NLP & Text Mining tools

Dr. Lorenzo Vaiani
Dipartimento di Automatica e Informatica
Politecnico di Torino

NLP or Text mining?

- Text Mining deals with the raw text, NLP deals with the underlying/latent meaning.
- Frequency counts of words, length of the sentence, presence/absence of certain words etc. is text mining.
- Text keywords, text category(topic), entity finding, etc. is NLP.

Natural Language Processing, today

- Researchers use Text Mining techniques to pre-preprocess data
- Natural Language Processing is used when the semantic meaning of text is involved in the task.
- Text Mining use as source data text documents, natural language processing deals with any form of human communication.

Research/Industrial Tools

- The de-facto standard programming language is python.

NLTK

spaCy

GENSIM
topic modelling for humans

Stanza

Transformers

Research/Industrial Tools

NLTK

- NLTK: Natural Language ToolKit is a complete package including low-level functions to manipulate text
  - Sentence/Word Tokenization
  - Multi-Lingual Lemmatization
  - Part of Speech tagging
  - Dependency tree parsing
  - ...

```
import nltk

print(nltk.word_tokenize("The students attended the lesson"))
>>> ["The", "students", "attended", "the", "lesson", "."]
```


Research/Industrial Tools

- GenerateSimilar: is the reference tool for semantic modelling of text
  - Word2Vec and FastText model management (training, inference, ...)
  - Latent semantic analysis (LSA, LSI, SVD)
  - Non-negative matrix factorization (NMF)
  - Latent Dirichlet allocation (LDA)
  - ...

```
model = Word2Vec(long_training_text, size=100, window=5)
model.save("word2vec.model")
print(model.most_similar(positive="keyboard"))

>>> ["mouse", "screen", "printer"]
```


Research/Industrial Tools

spaCy

- SpaCy: is among the most complete python packages and integrates both NLP and text mining techniques.
  - Support 15 languages (limited support up to 52 languages)
  - Already contains pretrained Word Vectors
  - Possible integration with deep learning methods
  - Easy integration of in the general NLP pipeline.
  - ...

```
doc = nlp("These are apples. These are oranges.")
for sent in doc.sents:
    print(sent)
>>> These are apples.
>>> These are oranges.
```


Research/Industrial Tools

Stanza

- **Stanza**: it is a collection of accurate and efficient tools for the linguistic analysis of many human languages.
  - It is developed and mantained by Stanford NLP Group.
  - it serves as a connector for the CoreNLP tool
  - It can be used for a variety of tasks in multiple languages (up to 66).
  - The most complete for NLP practitioner

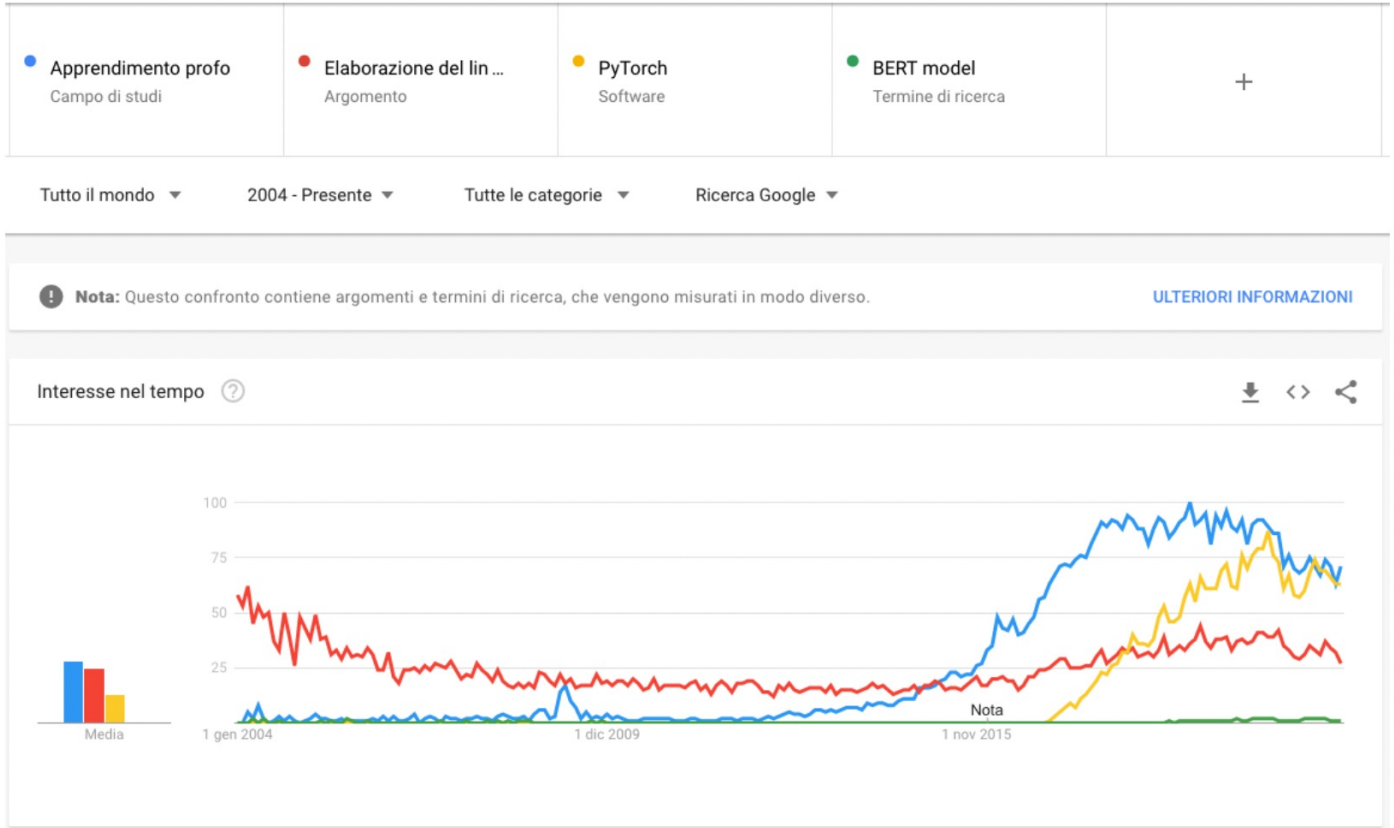
```
import stanza
lang_code = "en"
stanza.download(lang_code)

nlp = stanza.Pipeline(lang='en', processors='tokenize,ner')

text = "This is a very short text. However, it contains multiple sentences and some references to
Barack Obama or Cristiano Ronaldo"
doc = nlp(text)
print ("Named entities found:")
print(*[f'entity: {ent.text}\ttype: {ent.type}' for ent in doc.ents], sep='\n')

>>> Named entities found:
>>> entity: Barack Obama    type: PERSON
>>> entity: Cristiano Ronaldo   type: PERSON
```

Research/Industrial Tools

- Transformers: it is the de-facto standard for deep learning methodologies applied to NLP.
  - Thousands of pretrained models to perform tasks on texts.
  - Provides APIs to download and use those pretrained models.
  - Supports the popular deep learning libraries — Jax, PyTorch and TensorFlow
  - Pipeline API = Preprocessing + Model training/inference

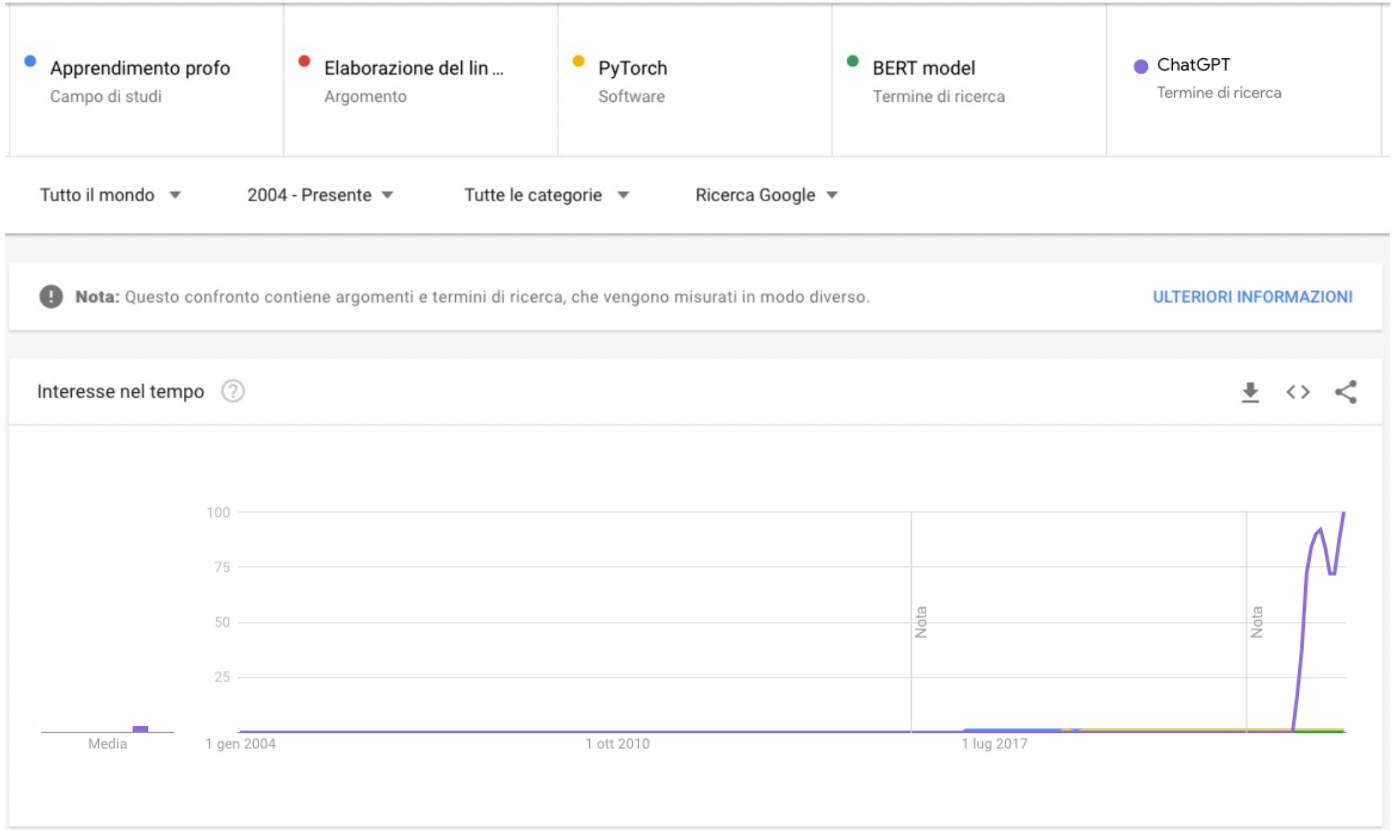
```
from transformers import pipeline

# Allocate a pipeline for sentiment-analysis
classifier = pipeline('sentiment-analysis')
classifier('We are very happy to introduce pipeline to the transformers repository.')

>>> [{'label': 'POSITIVE', 'score': 0.9996980428695679}]
```


Deep learning (r)evolution



GPT-3

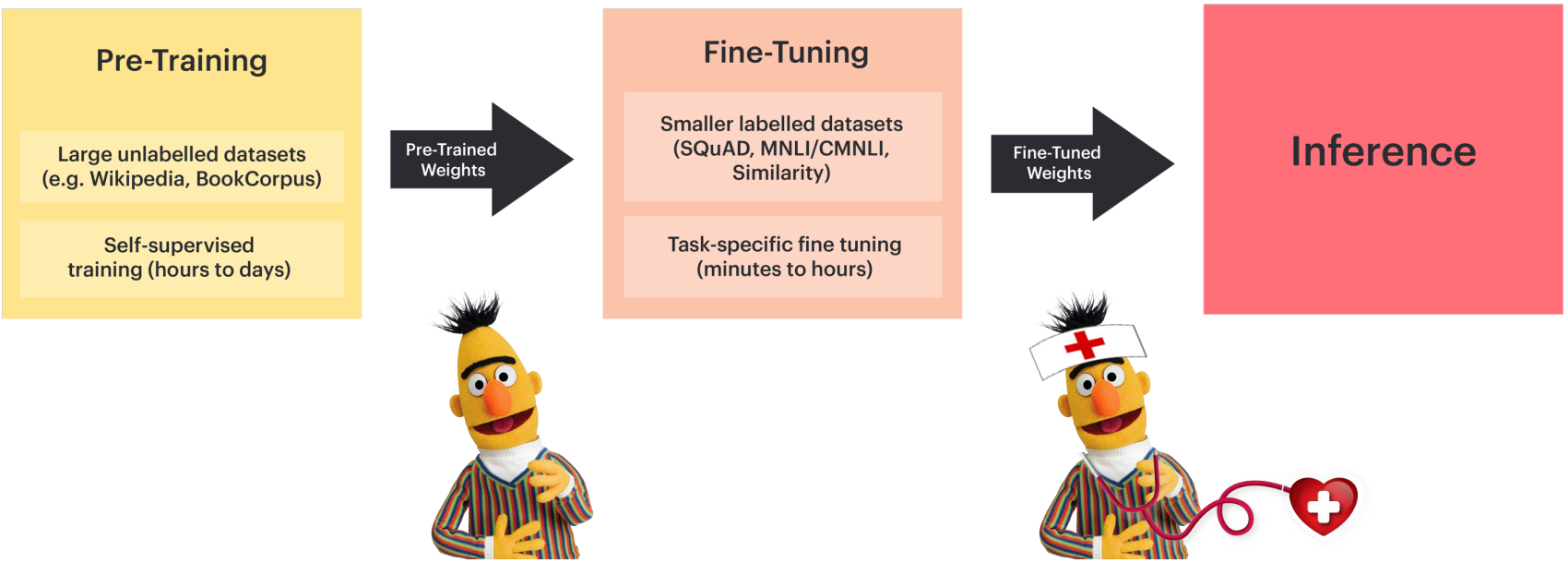
Deep learning (r)evolution



GPT-3

Deep learning (r)evolution

- Deep learning libraries define specific NN architectures.
- All modern NLP models are supported by each of those.
- Most architectures leverage *pretraining* + *fine-tuning* paradigm.



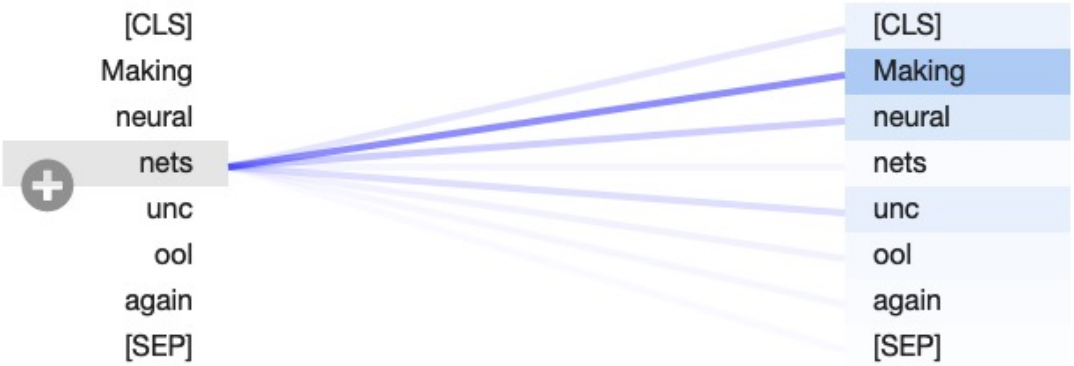
This is how all started

https://arxiv.org/abs/1706.03762

Attention Is All You Need

The Transformer is a model architecture which relies entirely on an attention mechanism to draw global dependencies between input and output. It allows for significantly more parallelization.

Multi-head attention, allows the model to jointly attend to information from different representation subspaces at different positions.



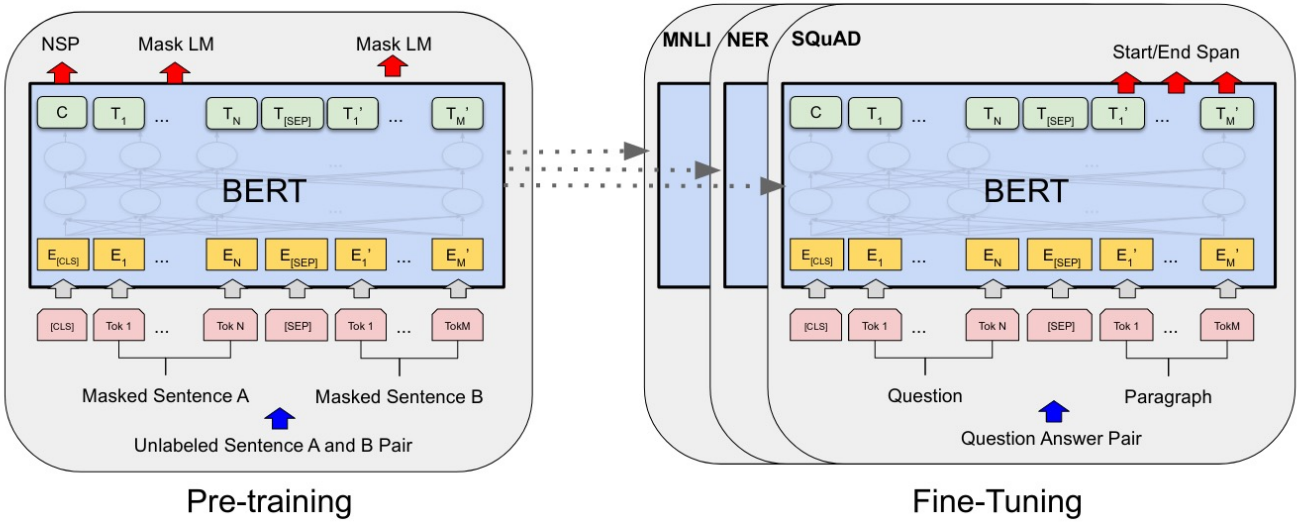
BERT

https://arxiv.org/abs/1810.04805

BERT makes use of Transformer (see prev. slide). It stacks multiple transformer encoders on top of each other.

It led to a real paradigm shift and internal revolution in the NLP community. It is the first real model to widely adopt the PT+FT paradigm (given the amount of data needed to train such model).

However, the proposed architecture is **discriminative** by nature.



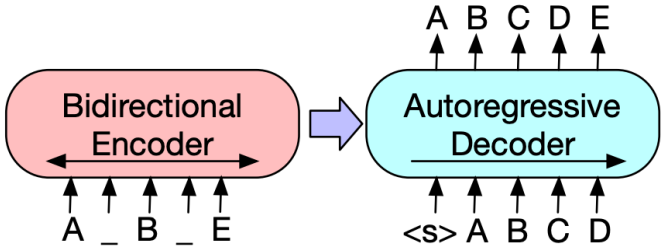
BART

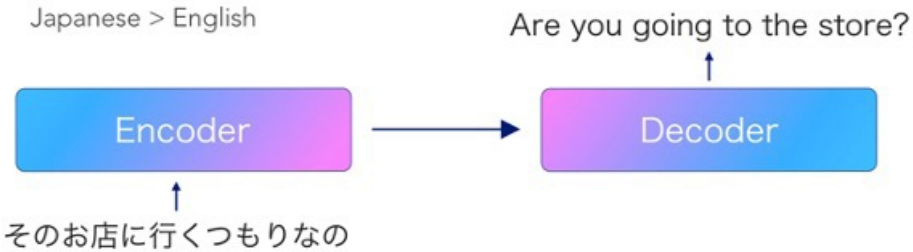
BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

https://arxiv.org/abs/1910.13461

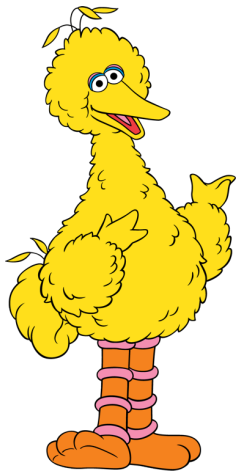
It is a **generative-model**. It also uses Transformer as backbone architecture, however, it is composed of:

- **Encoder**: it maps a sequence (of words) into a fixed vector representation). It's basically similar to BERT.
- **Decoder**: it takes as input a vector representation and produces a sequence (of words).





BigBird



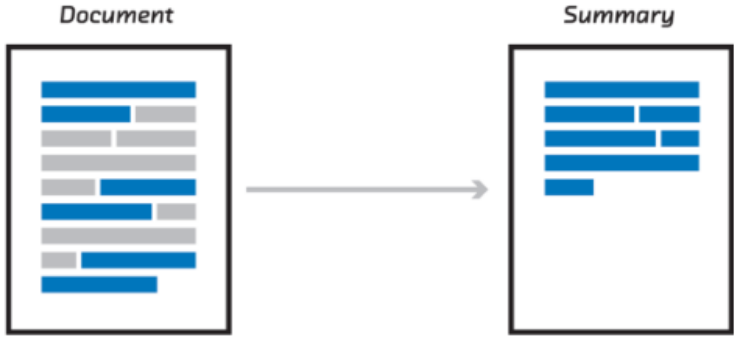
https://arxiv.org/abs/2007.14062

Big Bird: Transformers for Longer Sequences

One of the core limitation of transformers is the quadratic dependency (mainly in terms of memory) on the sequence length due to their full attention mechanism.

BigBird proposes a **sparse attention** mechanism that reduces this quadratic dependency to linear.

As a consequence of the capability to handle longer context, BigBird drastically improves performance on various NLP tasks such as question answering and summarization.

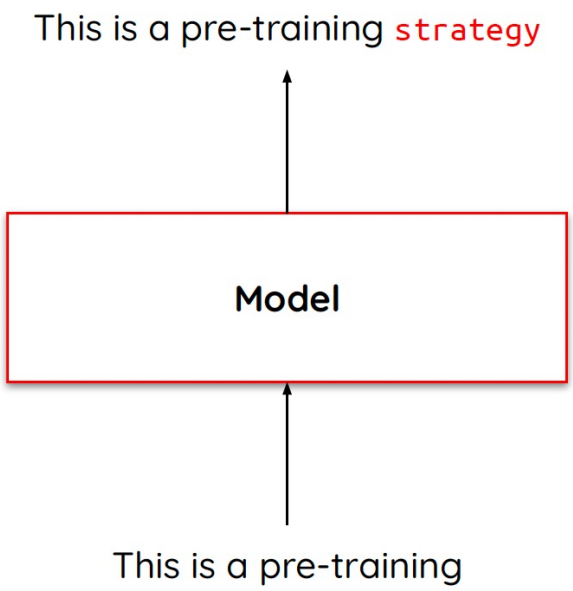


GPT-family

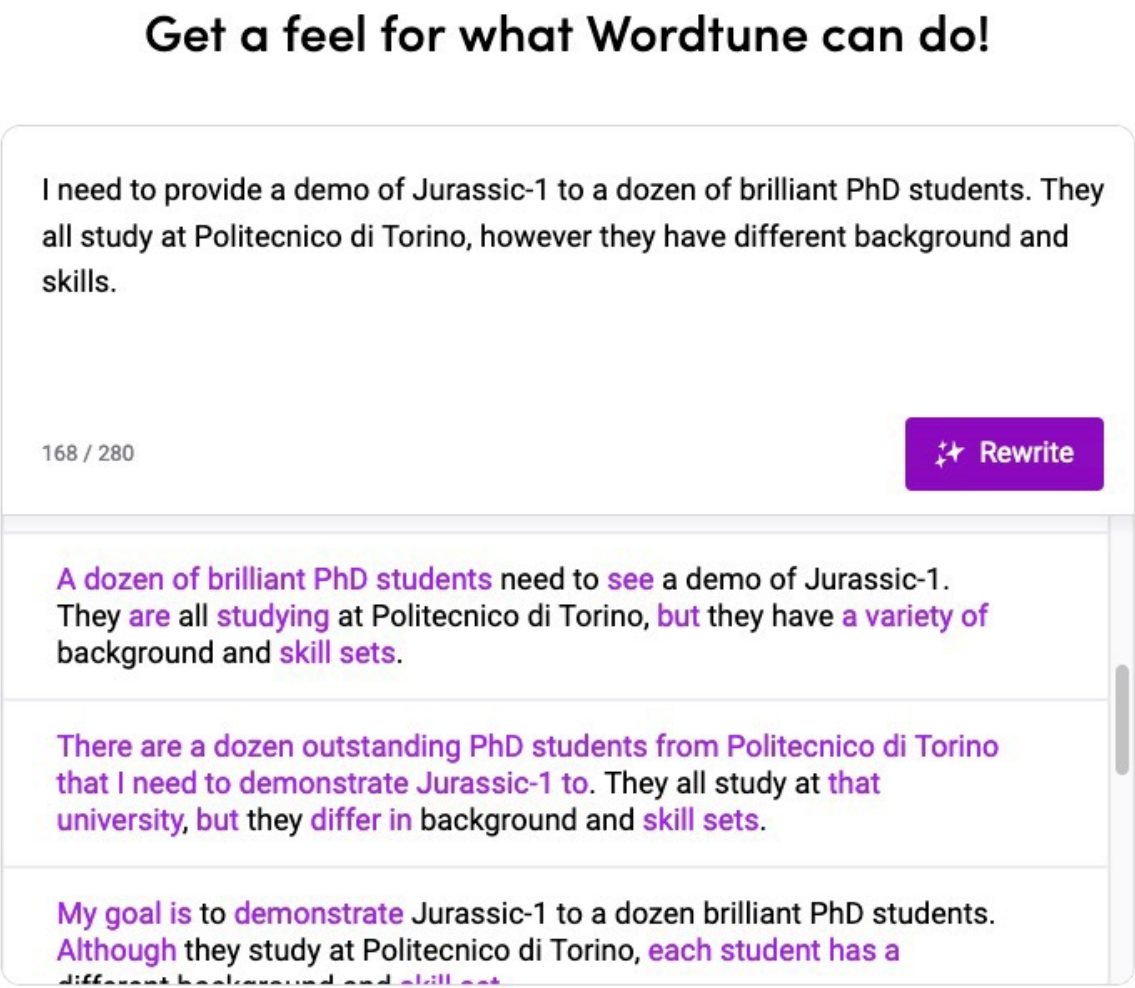
GPT: Generative Pre-trained Transformers
Generative models trained with next-word prediction task.

Family:

- GPT-2
- GPT-3
- ChatGPT
- GPT-3.5
- GPT-4



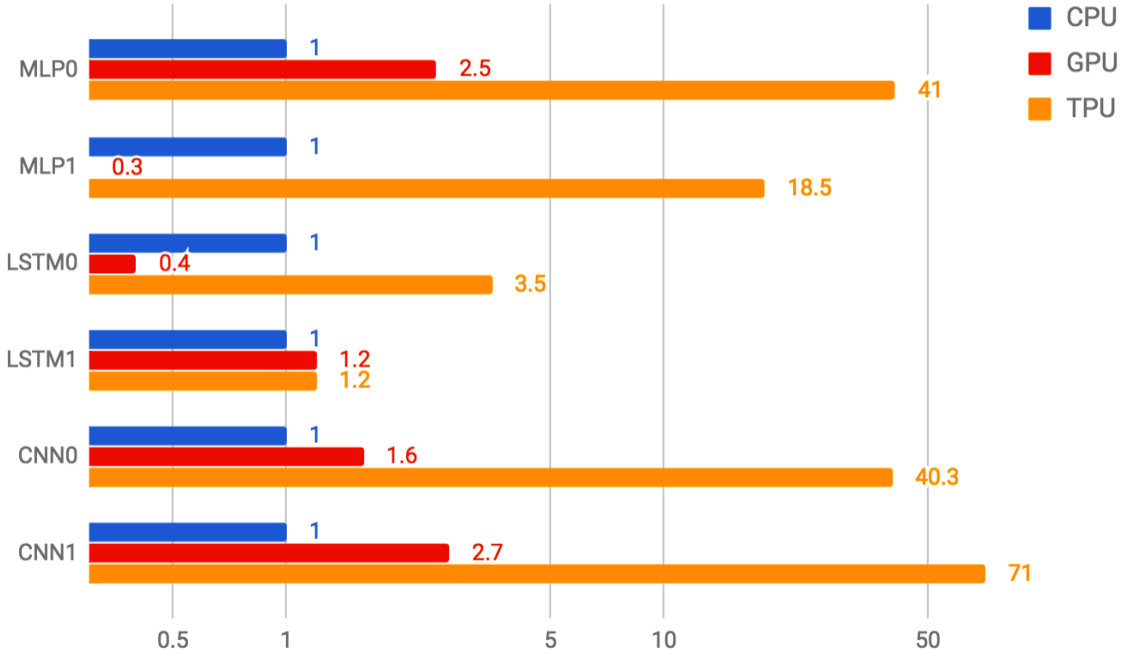
Jurassic-1 (GPT-3 competitor)



https://www.wordtune.com/

What do you need to go deep?

- Deep learning architectures are computationally expensive. GPUs and purpose-specific processing unit (TPUs) speed up training and inference.



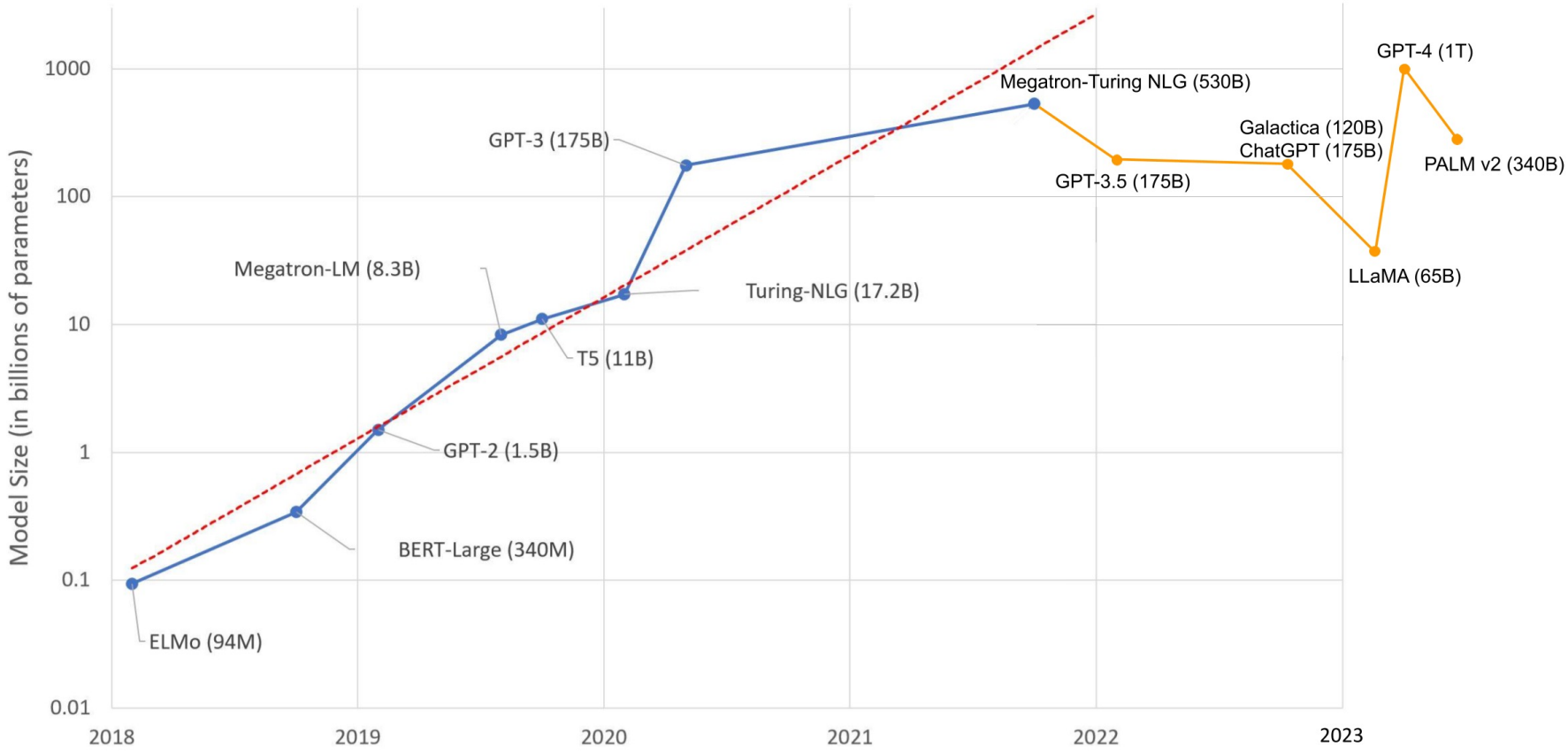
If you are interested:

Benchmarking TPU, GPU, and CPU Platforms for Deep Learning

https://arxiv.org/abs/1907.10701

What do you need to go deep?

- Deep learning architectures are computationally expensive.



What can you do in **your** research?

Different types of embedding

Embedding text to obtain its semantic representation can be essential for several end-tasks.

- Chars2vec: embedding sequences of characters, for spellcheckers.
- Word2Vec, GloVe, FastText: embedding words, used for a great variety of tasks.
- Deep neural models: they are not specifically designed for generating embeddings but to solve downstream tasks. The final-layer representation gives an accurate embedded representation of the source text.

Interesting projects

Sentence-BERT: https://www.sbert.net/

It is a Python framework for state-of-the-art sentence, text and image embeddings.

- It is really easy to setup and use
- It has really good performances (on medium-range GPU, encodes 900 sentences/s, on Intel i5 CPU encodes 5 sentences/s)
- It contains pretrained (aligned) models in multiple languages.
- ... etc.

Interesting projects

Transformer-interpret: https://github.com/cdpierse/transformers-interpret

Transformers Interpret is a model explainability tool designed to work exclusively with the 🤗 transformers package.

- Plug and play with SOTA deep learning models
- It supports visualizations in both notebooks and as savable html files.

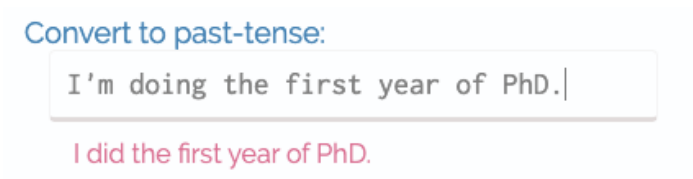
Legend: ■ Negative □ Neutral ■ Positive

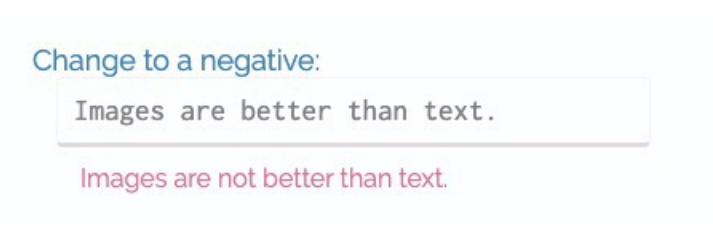
True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
POSITIVE	POSITIVE (1.00)	POSITIVE	2.02	[CLS] i love you , i like you [SEP]

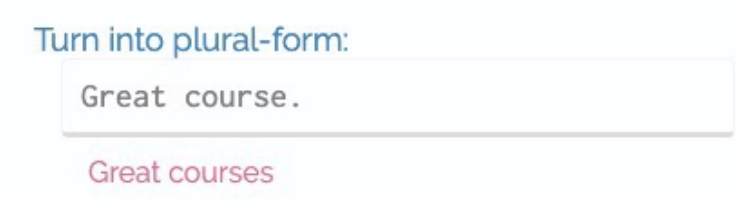
Interesting projects

Compromise http://compromise.cool/

A very light javascript-based framework to perform easy tasks directly on the browser.







Interesting projects

Python

- Sentence-BERT: https://www.sbert.net
- HF Transformers https://huggingface.co/transformers
- Allen NLP https://allennlp.org/
- Stanza https://stanfordnlp.github.io/stanza/

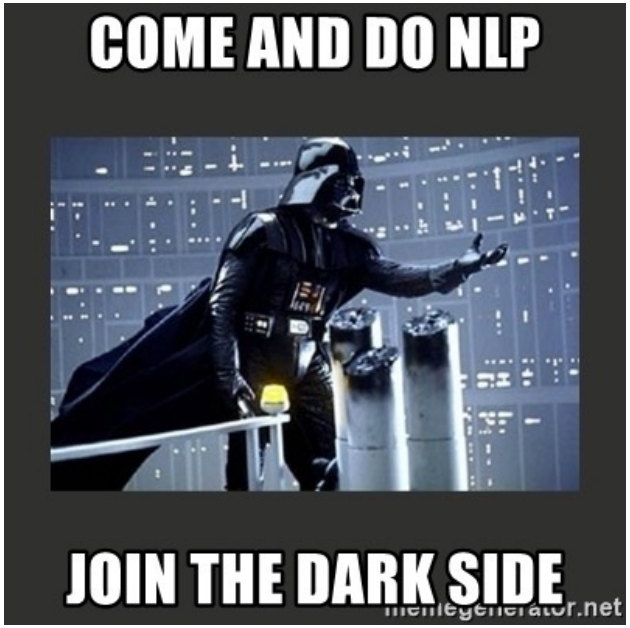
Java

- Stanford NLP https://nlp.stanford.edu/software/index.html
- NLP4J https://emorynlp.github.io/nlp4j/
- Apache Open NLP https://opennlp.apache.org/

Other Languages

- NLP.js https://github.com/axa-group/nlp.js (javascript)
- Compromise http://compromise.cool/ (javascript)
- wordVectors https://github.com/bmschmidt/wordVectors (R)

Intro to text mining and NLP



https://colab.research.google.com/drive/1ZzuVJGKl4eXjkqgv821rR09NWYJ1NGJQ?usp=sharing

Deep Dive into Hugging Face

https://colab.research.google.com/drive/1jjJ9037hI2CpE7SU-jSyieExsbJxcoFx?usp=sharing

Acknowlegdements and copyright license

- Copyright licence
  - Attribution + Noncommercial + NoDerivatives
- Acknowledgements
  - I would like to thank Dr. Moreno La Quatra, who created the first draft of the slides, and Prof. Luca Cagliero and Dr. Giuseppe Gallipoli, who collaborated to the revision of the teaching content.
- Affiliation
  - The author and his staff are currently members of the Database and Data Mining Group at Dipartimento di Automatica e Informatica (Politecnico di Torino) and of the SmartData interdepartmental centre
    - https://dbdmg.polito.it
    - https://smartdata.polito.it

Thank you!

