

# Processamento de Linguagem Natural

## Transformers na prática



Orientação: Prof. Paulo Henrique Maia

Aluno: Anderson Martins Gomes

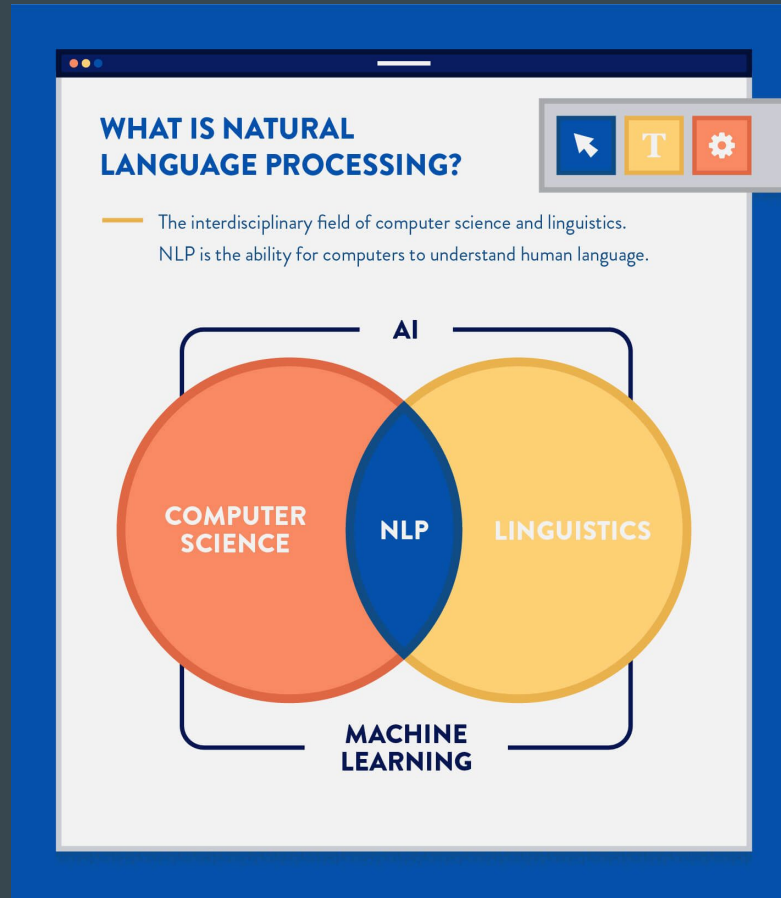
Agosto/22

1. Tarde 1 (Fundamentando)
  - a. **Conceitos iniciais**
    - i. NLP
    - ii. Tarefas comuns
    - iii. Caderno 1 (*Análise de Sentimentos, NER e Fill-Mask*)
  - b. **Transformers.Pipeline()**
    - i. Fases da Pipeline
    - ii. Transfer Learning
    - iii. Caderno 2 (*Behind the Pipeline*)
  - c. **Transformers**
    - i. História
    - ii. Arquitetura
    - iii. Caderno 3 (*Zero-Shooting Classification, Question-Answering e Translation*)
2. Tarde 2 (Aprofundando)
  - a. **Hugging Face**
    - i. Portal (hub)
    - ii. Licença de Uso
    - iii. Caderno 4 (*Text Generation, Summarization e Sentence Similarity*)
  - b. **Limitações e Customizações**
    - i. Bias Limitations
    - ii. Caderno 5 (Bias)
    - iii. Fine Tuning
  - c. **Caderno 6**
    - i. Automatic speech recognition (ASR)
    - ii. Image classification
    - iii. Object detection
3. Extra: coletânea de referências

# Natural Language Processing?



# Natural Language Processing!



# NLP Tasks?



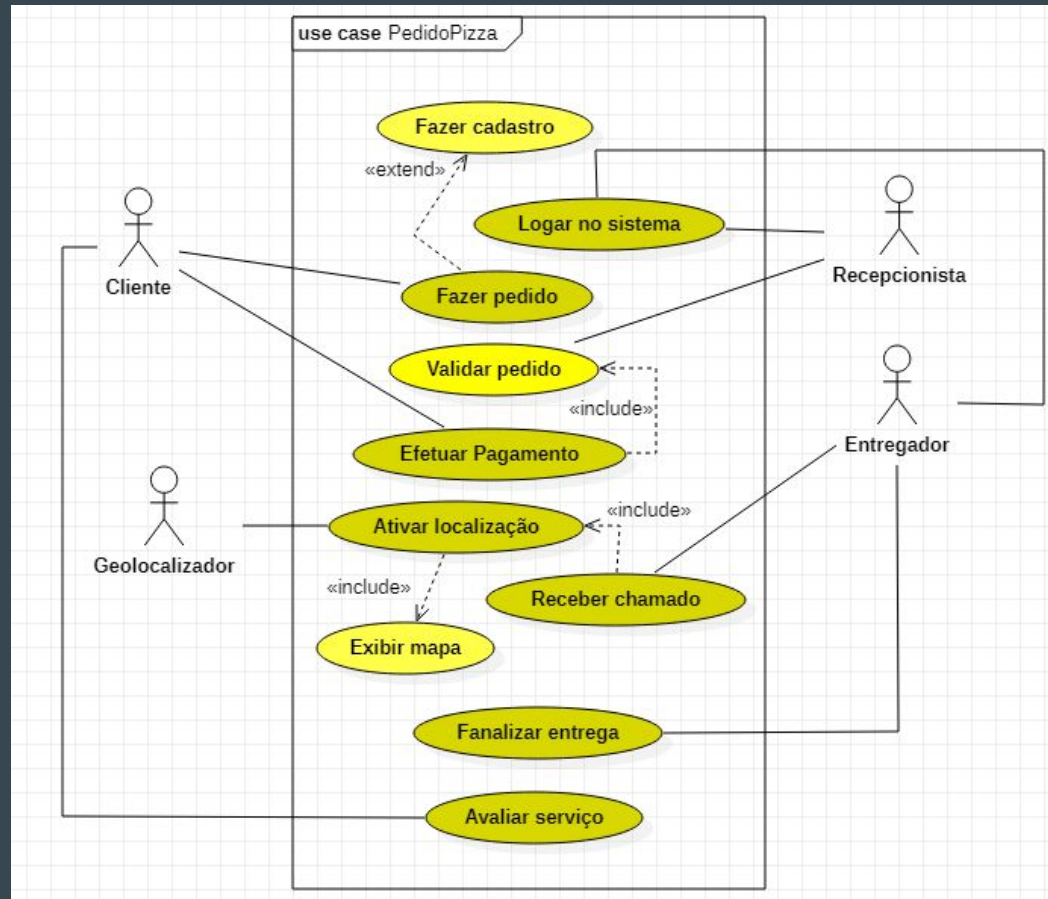
# NLP Tasks!



Image by NLPlanet



# Sistema de NLP x Sistema com NLP



# Sistema de NLP x Sistema com NLP

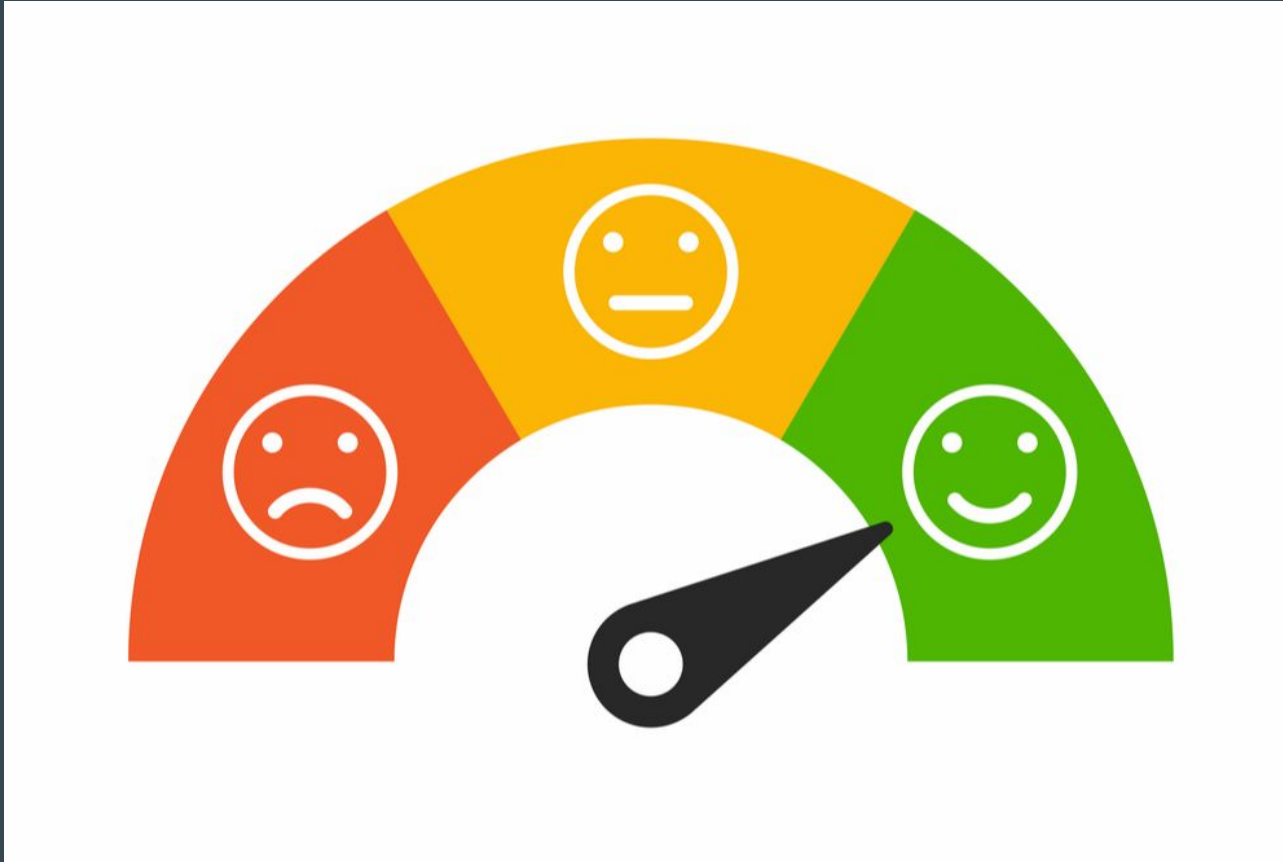




**Nossas três primeiras peças de lego**



# NLP Tasks: Sentiment Analysis





# NLP Tasks: Named Entity Recognition

ORGANISATION

LOCATION

DATE

PERSON

WEAPON

The **ISIS** ORG has claimed responsibility for a suicide bomb blast in the **Tunisian** LOC capital **earlier this week** DATE, the **militant group** ORG 's **Amaq news agency** ORG said on **Thursday** DATE. A **militant** PER wearing an **explosives belt** WEAPON blew himself up in **Tunis** LOC



Tom has fully \_\_\_\_ illness.

Best Guess : recovered from his



# Caderno 1/6

# Questionário de Revisão 1



<https://take.quiz-maker.com/QWMB65K3E>

**Transformer.Pipeline()**



# Transformers.Pipeline()



## Transformers



English | 简体中文 | 繁體中文 | 한국어

State-of-the-art Machine Learning for JAX, PyTorch  
and TensorFlow

**“Its[Transformers Library] goal is to provide a single API through which any Transformer model can be loaded, trained, and saved.”**

(Hugging Face’s documentation)





# Transformers.Pipeline()

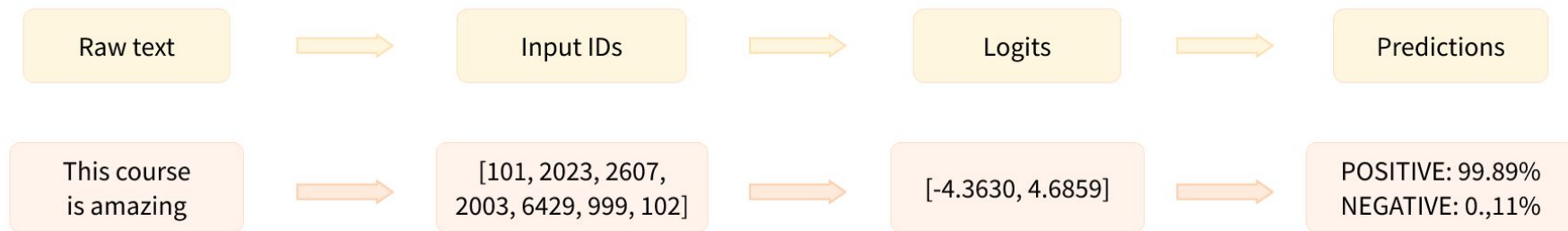
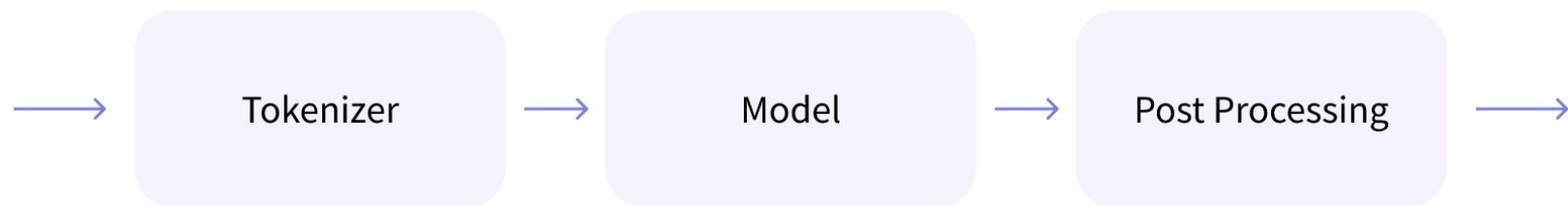


**“The pipeline function returns an end-to-end object that performs an NLP task on one or several texts.”**

(Hugging Face’s documentation)



# Transformers.Pipeline()



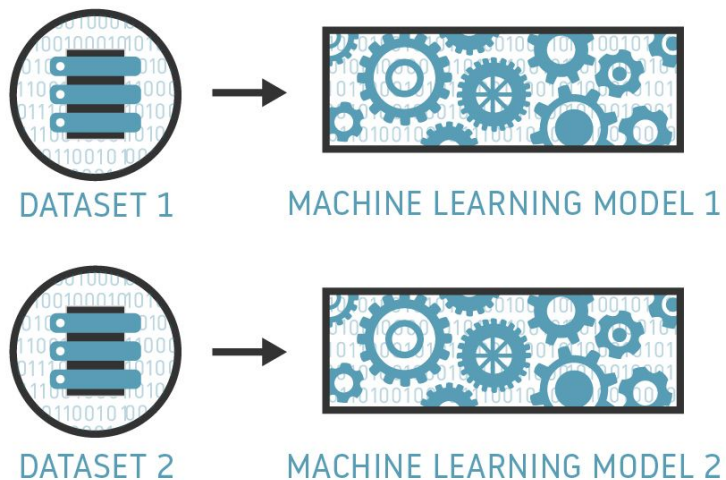
“Some of the currently available pipelines are: feature-extraction (get the vector representation of a text), fill-mask, ner (named entity recognition), question-answering, sentiment-analysis, summarization, text-generation, translation, zero-shot-classification etc”

(Hugging Face’s documentation)

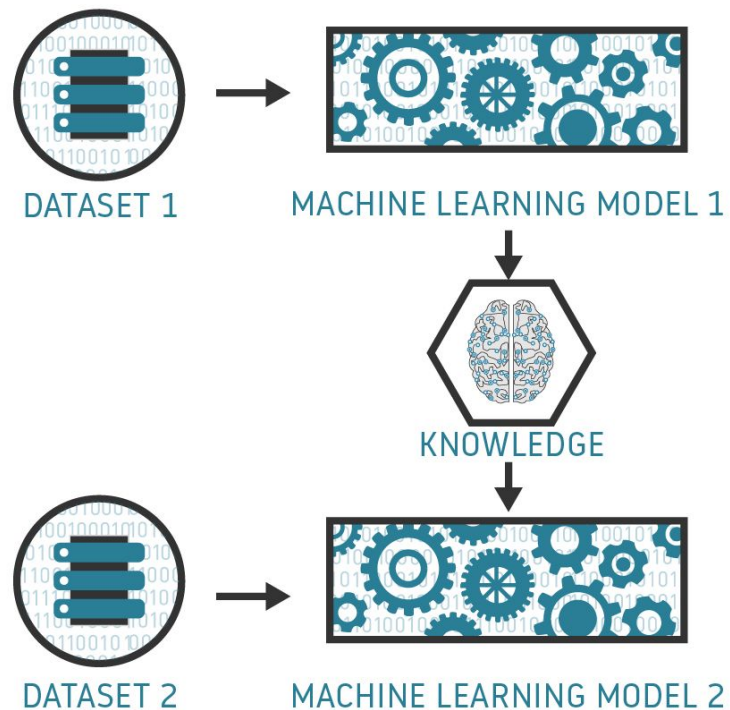
# Transfer Learning

# Transfer Learning

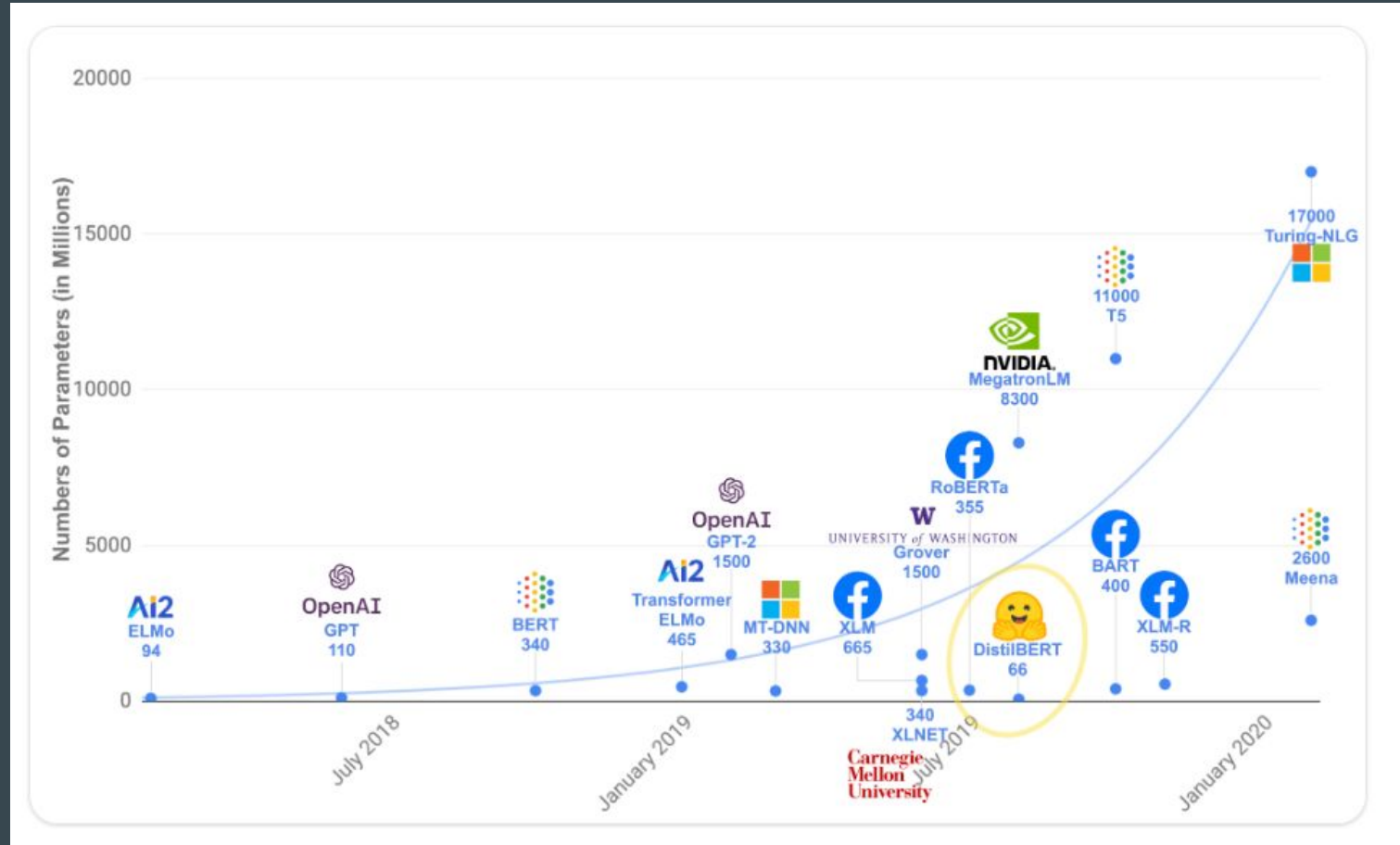
## TRADITIONAL MACHINE LEARNING



## TRANSFER LEARNING

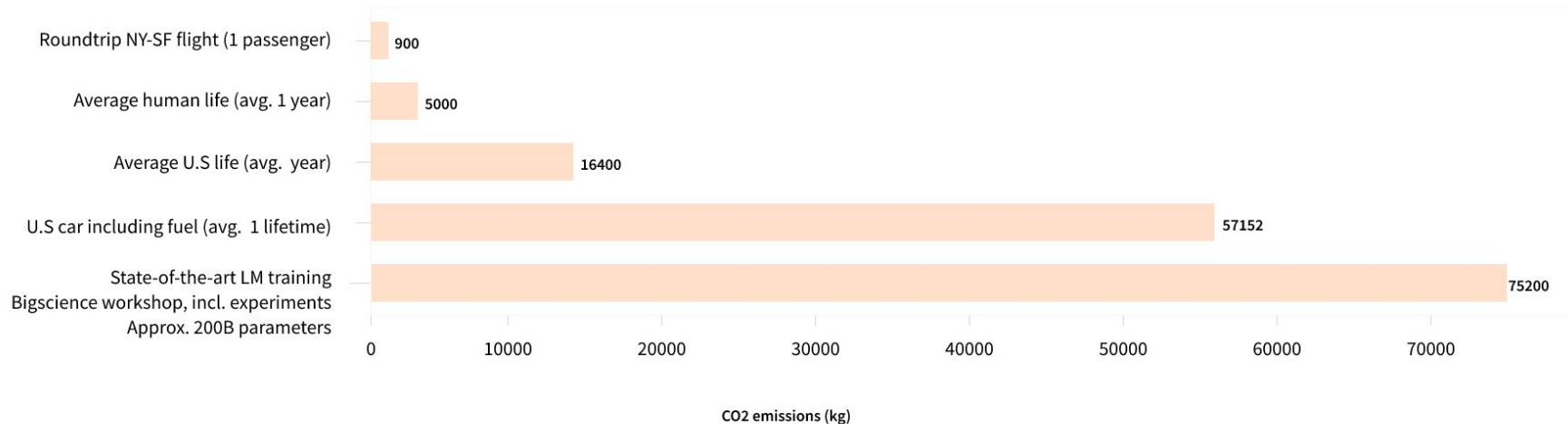


# Transfer Learning



# Transfer Learning

CO2 emissions for a variety of human activities



# Caderno 2/6

# Questionário de Revisão 2



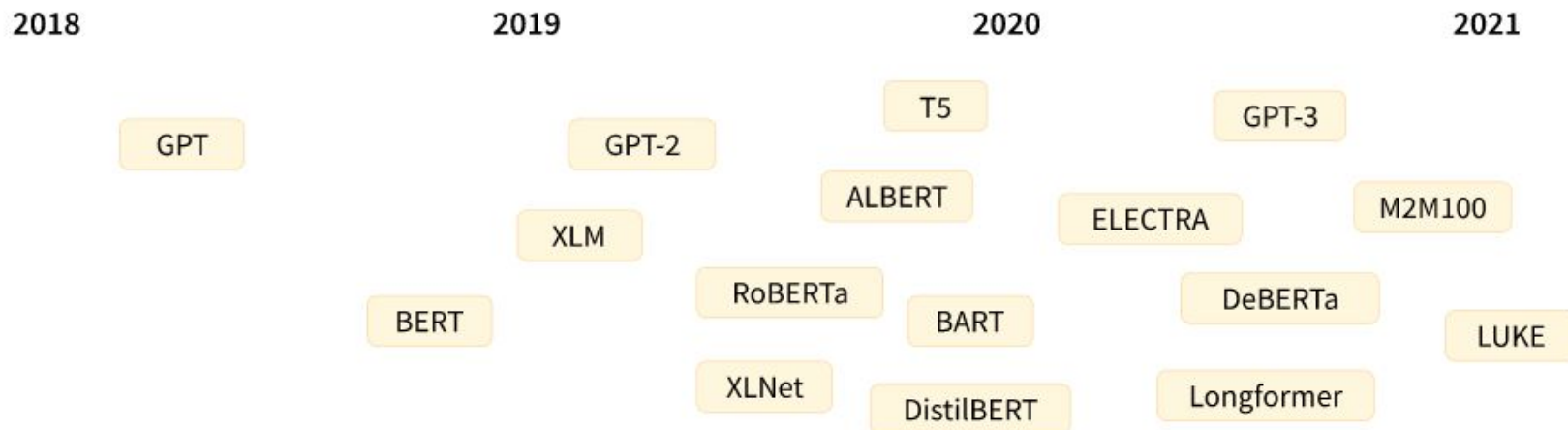
<https://take.quiz-maker.com/Q8GWCZRL1>



**Um pouco de História...**

## A bit of Transformer history

Here are some reference points in the (short) history of Transformer models:





Cornell University

the Sin

arXiv > cs > arXiv:1706.03762

Search...

Help | Advanced

Computer Science > Computation and Language

[Submitted on 12 Jun 2017 (v1), last revised 6 Dec 2017 (this version, v5)]

## Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Comments: 15 pages, 5 figures

Subjects: **Computation and Language (cs.CL)**; Machine Learning (cs.LG)

Cite as: arXiv:1706.03762 [cs.CL]

(or arXiv:1706.03762v5 [cs.CL] for this version)

<https://doi.org/10.48550/arXiv.1706.03762>

---

## Improving Language Understanding by Generative Pre-Training

---

Alec Radford  
OpenAI  
alec@openai.com

Karthik Narasimhan  
OpenAI  
karthikn@openai.com

Tim Salimans  
OpenAI  
tim@openai.com

Ilya Sutskever  
OpenAI  
ilyasu@openai.com

### Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding. Our general task-agnostic model outperforms discriminatively trained models that use architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test), 5.7% on question answering (RACE), and 1.5% on textual entailment (MultiNLI).



arXiv > cs > arXiv:1810.04805

Search...

Help | Advance

Computer Science > Computation and Language

[Submitted on 11 Oct 2018 (v1), last revised 24 May 2019 (this version, v2)]

## BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova

We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

Subjects: **Computation and Language (cs.CL)**

Cite as: [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) [cs.CL]

(or [arXiv:1810.04805v2](https://arxiv.org/abs/1810.04805v2) [cs.CL] for this version)

<https://doi.org/10.48550/arXiv.1810.04805>





Cornell University

the Si

arXiv > cs > arXiv:1910.10683

Search...

Help | Advance

Computer Science > Machine Learning

[Submitted on 23 Oct 2019 (v1), last revised 28 Jul 2020 (this version, v3)]

## Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu

Transfer learning, where a model is first pre-trained on a data-rich task before being fine-tuned on a downstream task, has emerged as a powerful technique in natural language processing (NLP). The effectiveness of transfer learning has given rise to a diversity of approaches, methodology, and practice. In this paper, we explore the landscape of transfer learning techniques for NLP by introducing a unified framework that converts all text-based language problems into a text-to-text format. Our systematic study compares pre-training objectives, architectures, unlabeled data sets, transfer approaches, and other factors on dozens of language understanding tasks. By combining the insights from our exploration with scale and our new "Colossal Clean Crawled Corpus", we achieve state-of-the-art results on many benchmarks covering summarization, question answering, text classification, and more. To facilitate future work on transfer learning for NLP, we release our data set, pre-trained models, and code.

Comments: Final version as published in JMLR

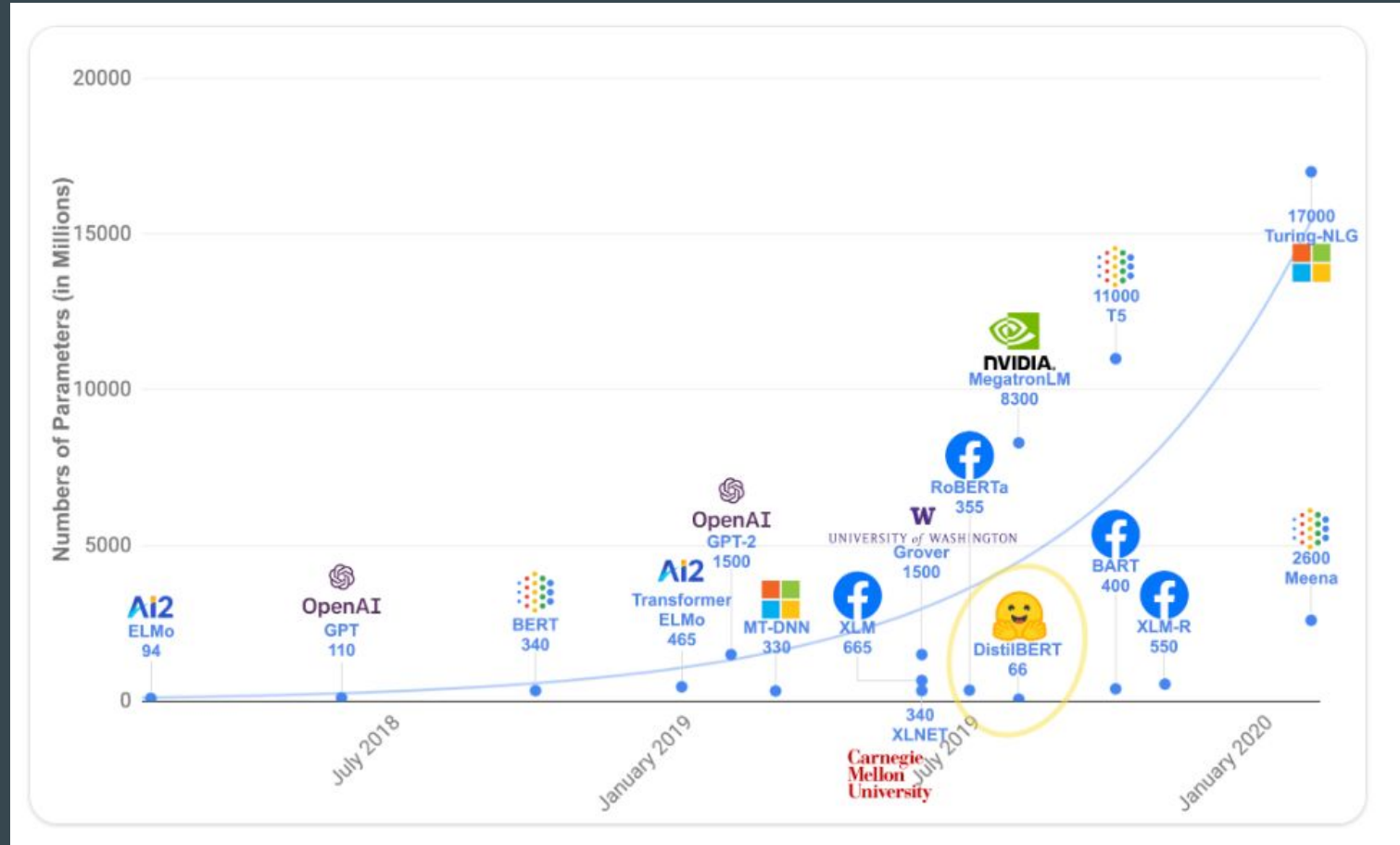
Subjects: **Machine Learning (cs.LG)**; Computation and Language (cs.CL); Machine Learning (stat.ML)

Cite as: [arXiv:1910.10683](https://arxiv.org/abs/1910.10683) [cs.LG]

(or [arXiv:1910.10683v3](https://arxiv.org/abs/1910.10683v3) [cs.LG] for this version)

<https://doi.org/10.48550/arXiv.1910.10683> 

# Transformers: a história

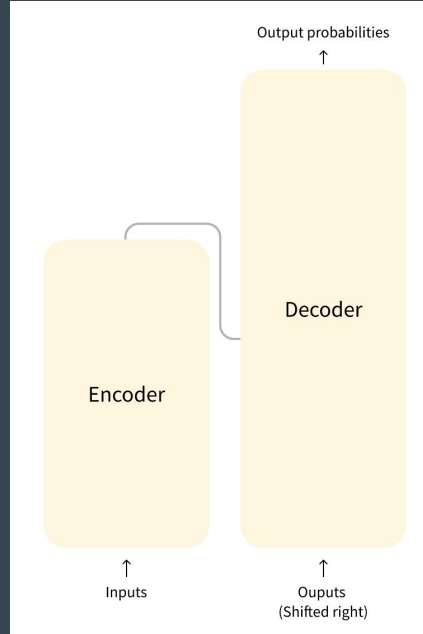


# A arquitetura Transformers



# A Arquitetura Transformers

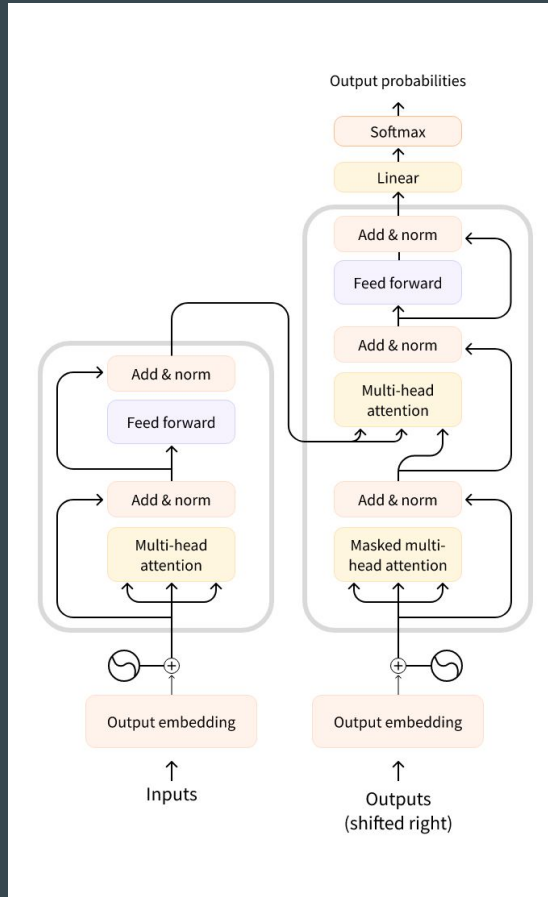
**Encoder-Only  
Models**



**Decoder-Only  
Models**

**Encoder-Decoder  
Models**

# A Arquitetura Transformers



# A Arquitetura Transformers

Model	Examples	Tasks
Encoder	ALBERT, BERT, DistilBERT, ELECTRA, RoBERTa	Sentence classification, named entity recognition, extractive question answering
Decoder	CTRL, GPT, GPT-2, Transformer XL	Text generation
Encoder-decoder	BART, T5, Marian, mBART	Summarization, translation, generative question answering

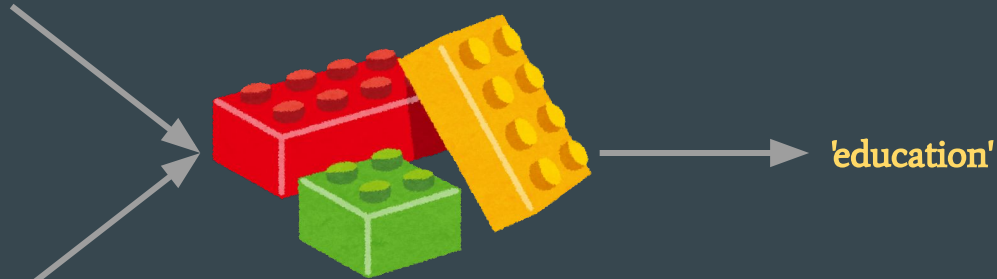
**Mais peças de lego**



# NLP Tasks: Zero-Shooting Classification

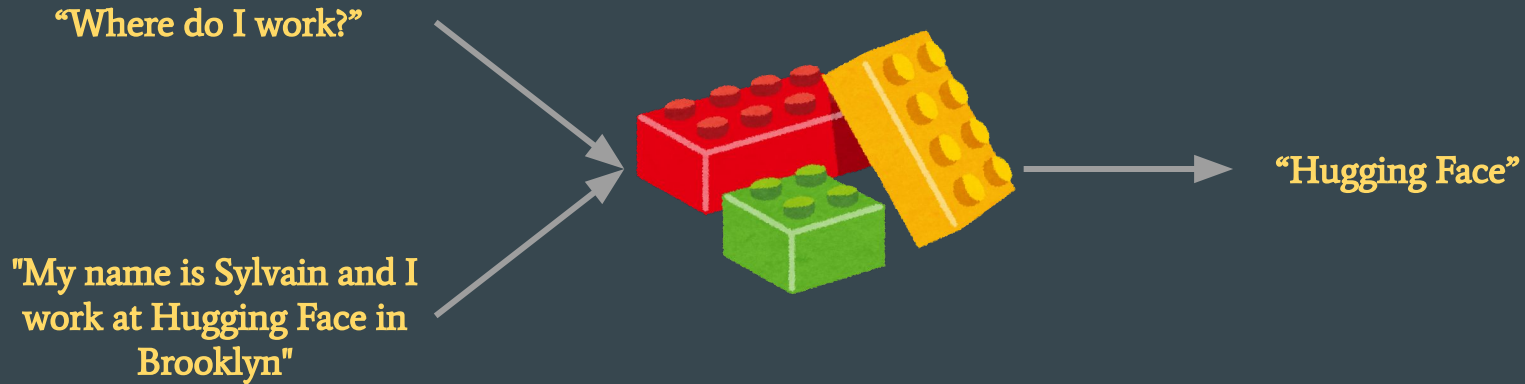
“This is a course about the  
Transformers library”

['education', 'business',  
'politics']





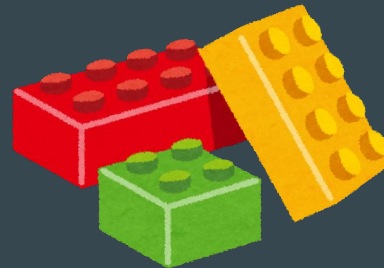
# NLP Tasks: Question-Answering





# NLP Tasks: Translation

“Where do I work?”



“Onde eu trabalho?”

# Caderno 3/6



# Questionário de Revisão 3



<https://take.quiz-maker.com/QQZVMF2S6>

# Hugging Face: o portal

**Hugging Face** Search models, datasets, users...

Models Datasets Spaces Docs Solutions Pricing

**Tasks**

- Image Classification Translation
- Image Segmentation Fill-Mask
- Automatic Speech Recognition Token Classification
- Sentence Similarity Audio Classification
- Question Answering Summarization
- Zero-Shot Classification + 18 Tasks

**Libraries**

- PyTorch TensorFlow JAX + 28

**Datasets**

- wikipedia common\_voice squad glue
- bookcorpus emotion conll2003 xtreme
- + 1312

**Languages**

- English Spanish French German
- Chinese Arabic Russian Japanese + 184

**Licenses**

- apache-2.0 mit afl-3.0 + 45


**Other**


- AutoTrain Compatible Eval Results
- Carbon Emissions

**Models** 65,251 Filter by name Sort: Most Downloads

- microsoft/deberta-base**  
Updated Jan 13 • ↓ 28.6M • ♥ 23
- bert-base-uncased**  
Fill-Mask • Updated Jun 6 • ↓ 27M • ♥ 219
- Jean-Baptiste/camembert-ner**  
Token Classification • Updated Apr 3 • ↓ 18.5M • ♥ 13
- gpt2**  
Text Generation • Updated May 19, 2021 • ↓ 11.4M • ♥ 186
- distilbert-base-uncased**  
Fill-Mask • Updated May 31 • ↓ 10.5M • ♥ 73
- bert-base-cased**  
Fill-Mask • Updated Sep 6, 2021 • ↓ 10.4M • ♥ 35
- SpanBERT/spanbert-large-cased**  
Updated May 19, 2021 • ↓ 8.37M • ♥ 5
- roberta-base**  
Fill-Mask • Updated Jul 6, 2021 • ↓ 7.01M • ♥ 46
- xlm-roberta-base**  
Fill-Mask • Updated Jun 6 • ↓ 6.63M • ♥ 47
- distilroberta-base**  
Fill-Mask • Updated 25 days ago • ↓ 5.25M • ♥ 24
- distilbert-base-uncased-finetuned-sst-2-english**  
Text Classification • Updated about 4 hours ago • ↓ 5.18M • ♥ 82
- distilgpt2**  
Text Generation • Updated 25 days ago • ↓ 4.29M • ♥ 78
- tals/albert-xlarge-vitaminc-mnli**  
Text Classification • Updated 12 days ago • ↓ 3.95M
- deepset/roberta-base-squad2**  
Question Answering • Updated 1 day ago • ↓ 3.7M • ♥ 100
- deepset/xlm-roberta-base-squad2**  
Question Answering • Updated 1 day ago • ↓ 3.86M • ♥ 11
- bert-base-chinese**  
Fill-Mask • Updated 25 days ago • ↓ 3.5M • ♥ 111
- albert-base-v2**  
Fill-Mask • Updated Aug 30, 2021 • ↓ 3.14M • ♥ 16
- sentence-transformers/all-MiniLM-L6-v2**  
Sentence Similarity • Updated Jul 11 • ↓ 2.8M • ♥ 72
- hfl/chinese-macbert-base**  
Fill-Mask • Updated May 19, 2021 • ↓ 2.62M • ♥ 51
- bert-large-uncased**  
Fill-Mask • Updated May 18, 2021 • ↓ 2.3M • ♥ 9

# Hugging Face: o tipo licença

 main ▾ transformers / LICENSE Go to file ...






huggingface/transformers is licensed under the **Apache License 2.0**





A permissive license whose main conditions require preservation of copyright and license notices. Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

This is not legal advice. [Learn more about repository licenses.](#)

Permissions	Limitations	Conditions
✓ Commercial use	✗ Trademark use	① License and copyright notice
✓ Modification	✗ Liability	① State changes
✓ Distribution	✗ Warranty	
✓ Patent use		
✓ Private use		

 **sgugger** Copyright (#8970) ... ✓ Latest commit 00aa9db on Dec 7, 2020  History

 3 contributors

203 lines (170 sloc) | 11.2 KB Raw Blame    

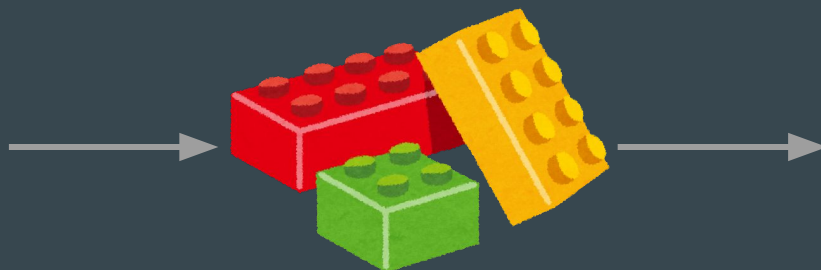
```
1 Copyright 2018- The Hugging Face team. All rights reserved.
2
3           Apache License
4           Version 2.0, January 2004
5           http://www.apache.org/licenses/
6
7 TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
8
```

**Mais peças de lego**



# NLP Tasks: Text Generation

“Hello, I’m a language  
model, ...”



“Hello, I’m a language  
model, and I’m trying to be as  
expressive as possible.”



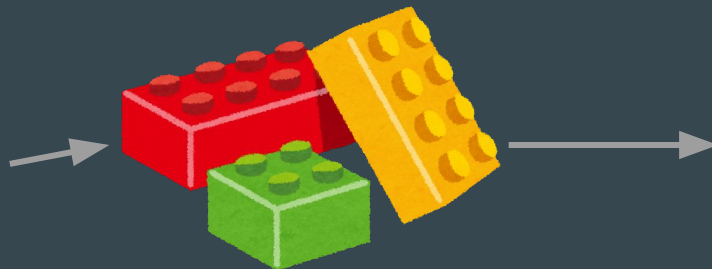
# NLP Tasks: Summarization

“New York (CNN)When Liana Barrientos was 23 years old, she got married in Westchester County, New York.

A year later, she got married again in Westchester County, but to a different man and without divorcing her first husband. Only 18 days after that marriage, she got hitched yet again. Then, Barrientos declared "I do" five more times, sometimes only within two weeks of each other.

In 2010, she married once more, this time in the Bronx. In an application for a marriage license, she stated it was her "first and only" marriage.

Barrientos, now 39, is facing two criminal counts of "offering a false instrument for filing in the first degree," referring to her false statements on the 2010 marriage license application, according to court documents <...>”



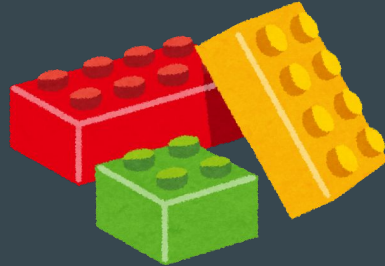
“Liana Barrientos, 39, is charged with two counts of "offering a false instrument for filing in the first degree" In total, she has been married 10 times, with nine of her marriages occurring between 1999 and 2002. She is believed to still be married to four men.”



# NLP Tasks: Sentence Similarity

“This is a red cat  
with a hat.”

“Have you seen  
my red cat?”



0.61

# Caderno 4/6



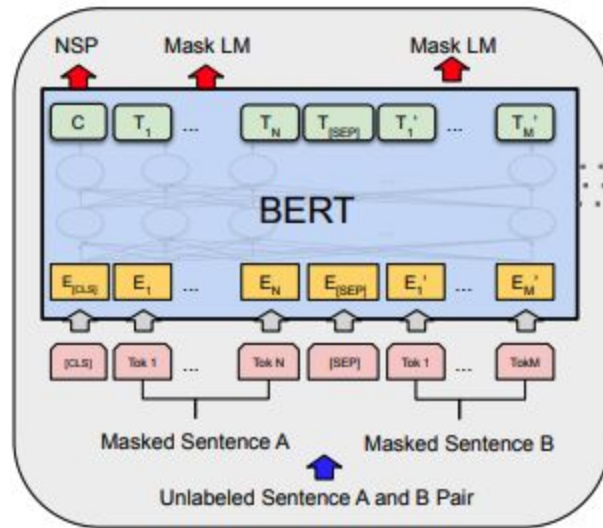
# Limitações e Customizações

```
from transformers import pipeline

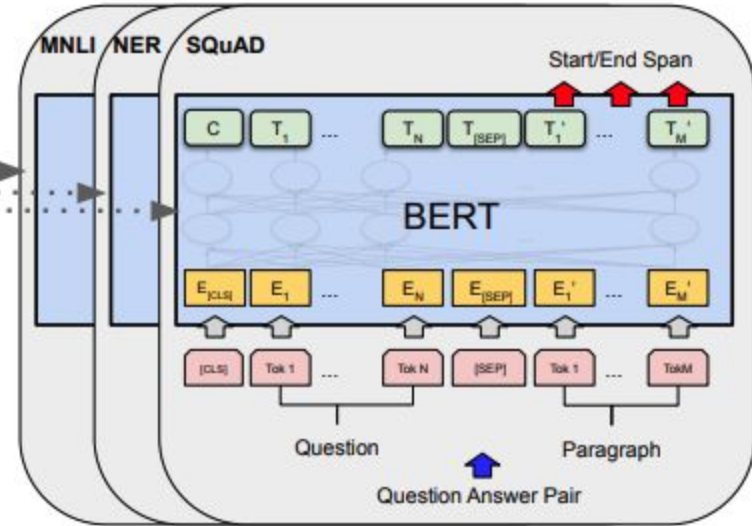
unmasker = pipeline("fill-mask", model="bert-base-uncased")
result = unmasker("This man works as a [MASK].")
print([r["token_str"] for r in result])

result = unmasker("This woman works as a [MASK].")
print([r["token_str"] for r in result])
```

# Fine Tuning

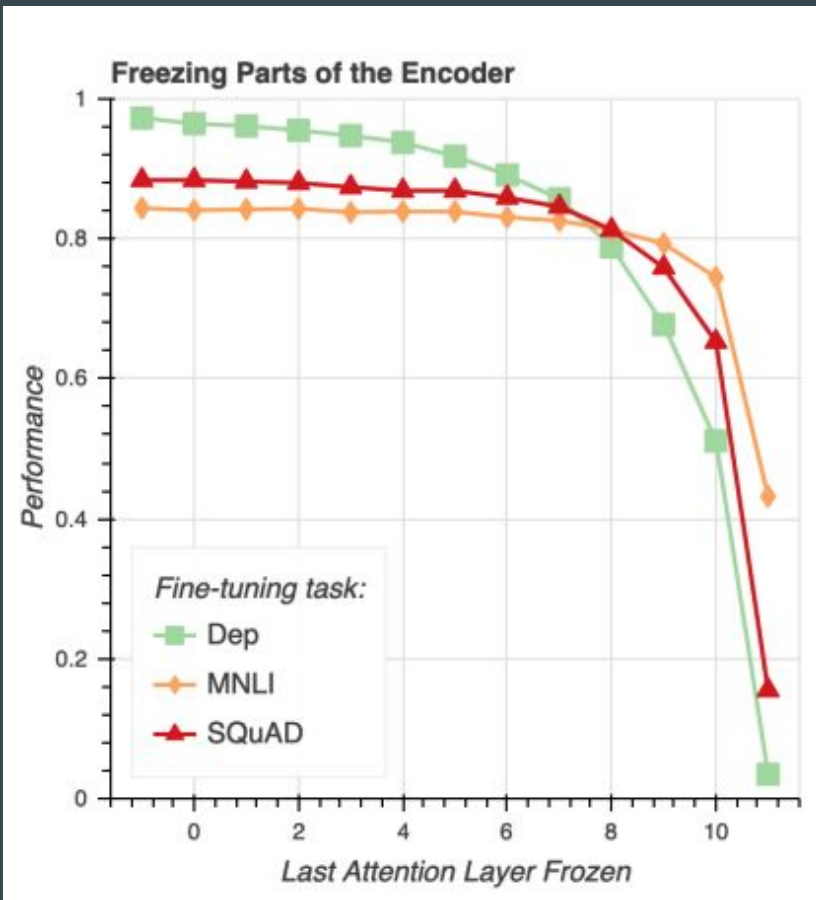


Pre-training

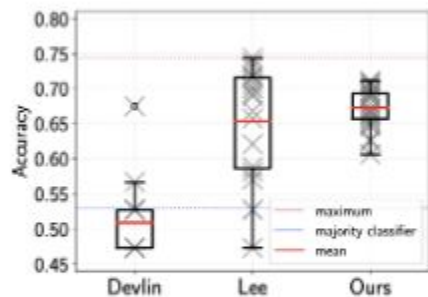


Fine-Tuning

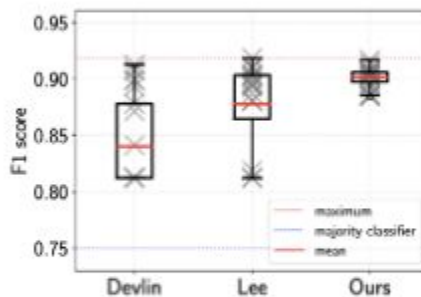
# Fine Tuning



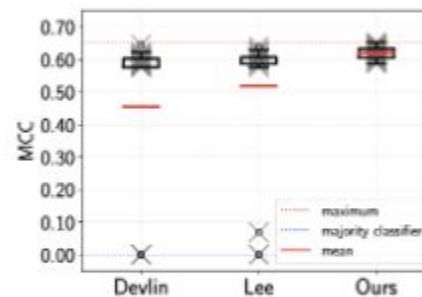
## Is fine-tuning stable?



(a) RTE



(b) MRPC



(c) CoLA

Fine-tuned model performance over 25 random seeds, following the fine-tuning recipe from Devlin et al (left-most points), and Mosbach et al (right-most points). Mosbach's recipe consistently yields more stable results.

Image credit: Mosbach et al 2021.

## Do we need to fine-tune at all?

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	<b>93.1</b>
Fine-tuning approach		
BERT <sub>LARGE</sub>	96.6	92.8
<b>BERT<sub>BASE</sub></b>	<b>96.4</b>	92.4
Feature-based approach (BERT <sub>BASE</sub> )		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
<b>Last Hidden</b>	<b>94.9</b>	-
Weighted Sum Last Four Hidden	95.9	-
<b>Concat Last Four Hidden</b>	<b>96.1</b>	-
Weighted Sum All 12 Layers	95.5	-

The feature-based approach performs close to the fine-tuned model on CoNLL-2003. Image credit: Devlin et al 2019.

## To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks

Matthew Peters<sup>1\*</sup>, Sebastian Ruder<sup>2,3†\*</sup>, and Noah A. Smith<sup>1,4</sup>

<sup>1</sup>Allen Institute for Artificial Intelligence, Seattle, USA

<sup>2</sup>Insight Research Centre, National University of Ireland, Galway, Ireland

<sup>3</sup>Aylien Ltd., Dublin, Ireland

<sup>4</sup>Paul G. Allen School of CSE, University of Washington, Seattle, USA

{matthewp, noah}@allenai.org, sebastian@ruder.io

### Abstract

While most previous work has focused on different pretraining objectives and architectures for transfer learning, we ask how to best adapt the pretrained model to a given target task. We focus on the two most common forms of adaptation, feature extraction (where the pretrained weights are frozen), and directly fine-tuning the pretrained model. Our empirical results across diverse NLP tasks with two state-of-the-art models show that the relative performance of fine-tuning vs. feature extraction depends on the similarity of the pretraining and target tasks. We explore possible explanations for this finding and provide a set of adaptation guidelines for the NLP practitioner.

Conditions			Guidelines
Pretrain	Adapt.	Task	
Any	❄️	Any	Add many task parameters
Any	🔥	Any	Add minimal task parameters ⚠️ Hyper-parameters
Any	Any	Seq. / clas.	❄️ and 🔥 have similar performance
ELMo	Any	Sent. pair	use ❄️
BERT	Any	Sent. pair	use 🔥

Table 1: This paper's guidelines for using feature extraction (❄️) and fine-tuning (🔥) with ELMo and BERT. Seq.: sequence labeling. Clas.: classification. Sent. pair: sentence pair tasks.

computationally cheaper as features only need to be computed once. On the other hand, 🔥 is convenient as it may allow us to adapt a general-purpose

[cs.CL] 11 Jun 2019



## Conclusion

Let's summarize the learnings with 3 bullets:

- **BERT's layers are hierarchical.** Early BERT layers learn more generic linguistic patterns, the equivalent of edges and corners in deep vision models, while the later BERT layers learn more task-specific patterns.
- **Fine-tuning is not always necessary.** Instead, the feature-based approach, where we simply extract pre-trained BERT embeddings as features, can be a viable, and cheap, alternative. However, it's important to not use just the final layer, but at least the last 4, or all of them.
- **Fine-tuning is brittle when following the recipe from Devlin et al.** This *fine-tuning instability* has been shown to go away when training for a larger number of epochs, and when using the original ADAM optimizer, instead of the modified version used in Devlin et al.



## NerPipeline

`class transformers.TokenClassificationPipeline`

[< source >](#)

```
( args_parser = <transformers.pipelines.token_classification.TokenClassificationArgumentHandler object at 0x7fce94c82040>, *args, **kwargs )
```

### Parameters

- **model** ([PreTrainedModel](#) or [TFPreTrainedModel](#)) — The model that will be used by the pipeline to make predictions. This needs to be a model inheriting from [PreTrainedModel](#) for PyTorch and [TFPreTrainedModel](#) for TensorFlow.
- **tokenizer** ([PreTrainedTokenizer](#)) — The tokenizer that will be used by the pipeline to encode data for the model. This object inherits from [PreTrainedTokenizer](#).
- **modelcard** (`str` or `ModelCard`, *optional*) — Model card attributed to the model for this pipeline.
- **framework** (`str`, *optional*) — The framework to use, either "pt" for PyTorch or "tf" for TensorFlow. The specified framework must be installed.

If no framework is specified, will default to the one currently installed. If no framework is specified and both frameworks are installed, will default to the framework of the model, or to PyTorch if no model is provided.

**Caderno 6/6**

# Referências

- Artigos Científicos “Seminais”:
  - Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017). <https://arxiv.org/abs/1706.03762>
  - Radford, Alec, et al. "Improving language understanding by generative pre-training." (2018). [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf)
  - Devlin, Jacob. 2018. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” Cornell University. <https://arxiv.org/abs/1810.04805>.
  - Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." J. Mach. Learn. Res. 21.140 (2020): 1-67. <https://arxiv.org/abs/1910.10683>
- Melhores livros:
  - <https://www.tableau.com/learn/articles/natural-language-processing-books>
- Outras fontes:
  - [https://www.youtube.com/playlist?list=PLBkQ2d3BHwmpAe95W\\_GWPSx8rP14OXwDx](https://www.youtube.com/playlist?list=PLBkQ2d3BHwmpAe95W_GWPSx8rP14OXwDx)
  - <https://www.deeplearning.ai/courses/>
  - <https://huggingface.co/course/chapter1/1>
  - <https://medium.com/nlplanet/awesome-nlp-18-high-quality-resources-for-studying-nlp-1b4f7fd87322>

# Referências Complementares

- Jurafsky, Dan, and James H. Martin. 2009. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. N.p.: Pearson Prentice Hall
- Mao, Fagui, et al. "Operational pattern based code generation for management information system: An industrial case study." 2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence,
- Falcon, William. 2019. "Lit BERT: NLP Transfer Learning In 3 Steps | by William Falcon." Towards Data Science. <https://towardsdatascience.com/lit-bert-nlp-transfer-learning-in-3-steps-272a866570db>.
- Linhares, Ticiana, and Márcia Lima. 2021. "PLN - Processamento de Linguagem Natural para Iniciantes – Insight Data Science Lab." Insight Lab. <https://insightlab.ufc.br/pln-processamento-de-linguagem-natural-para-iniciantes>.
- Mikolov, Tomas. 2013. "Efficient Estimation of Word Representations in Vector Space." Cornell University. <https://arxiv.org/abs/1301.3781>.
- Santos, Vinicius. 2021. "Term Frequency (TF) e Term Frequency - Inverse Document Frequency (TF-IDF)." Computer Science Master. <https://www.computersciencemaster.com.br/term-frequency-e-term-frequency-inverse-document-frequency/>.
- <https://www.deeplearningbook.com.br/o-que-e-bert-bidirectional-encoder-representations-from-transformers/>

# Slides Complementares

We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement). (Devlin 2018)

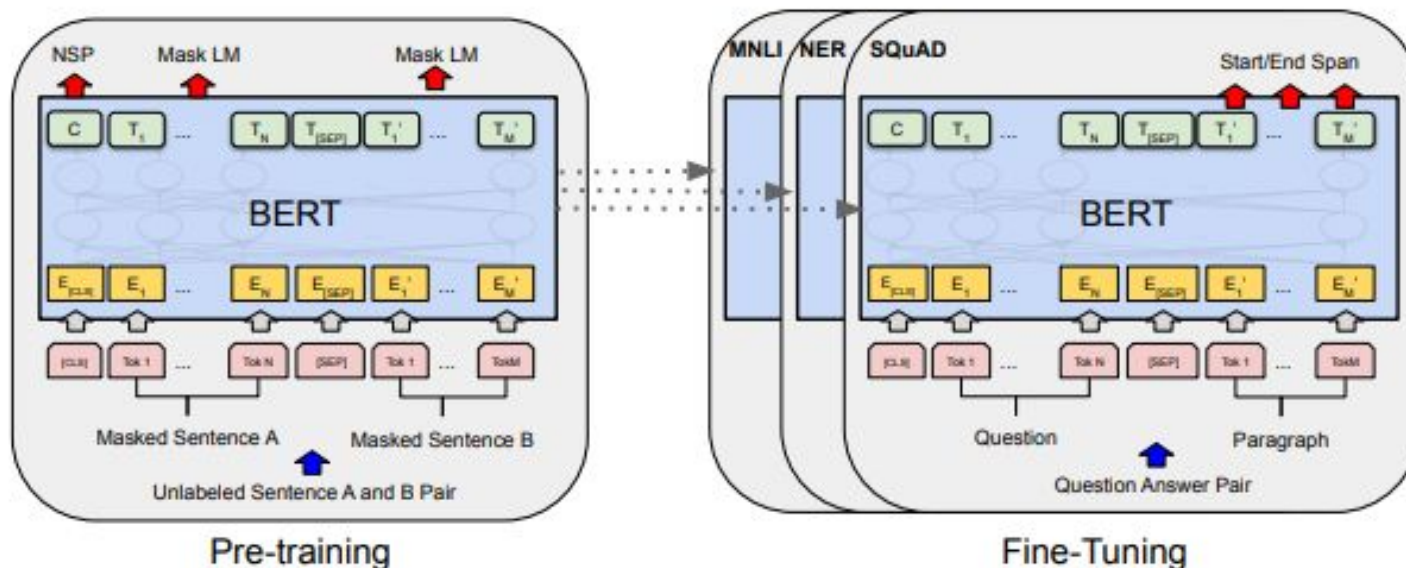
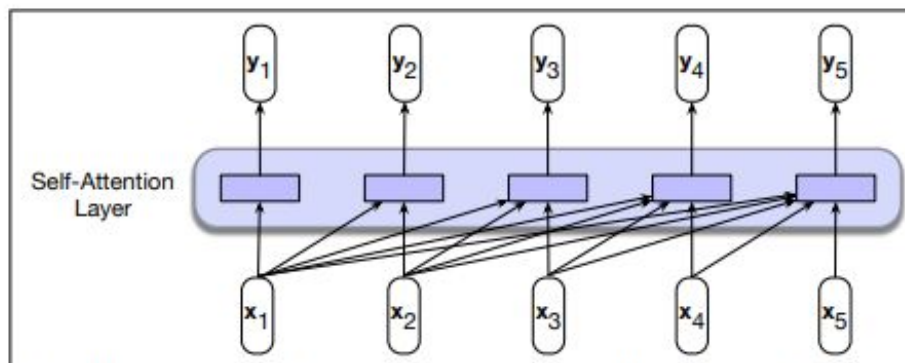


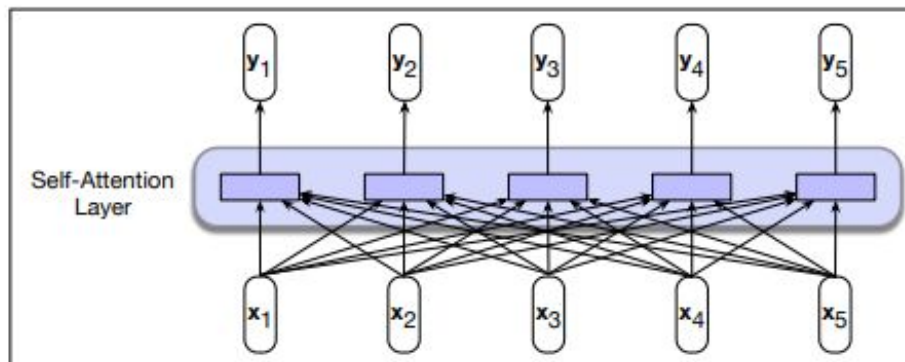
Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).



# Google BERT - Bidirectional Encoders

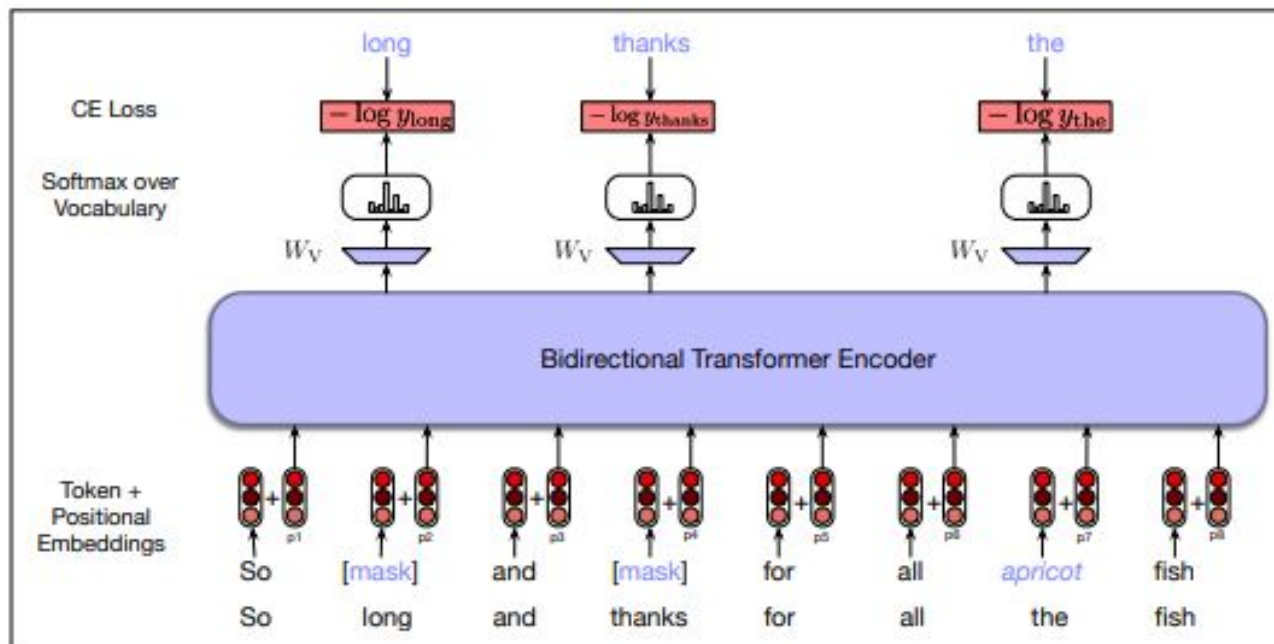


**Figure 11.1** A causal, backward looking, transformer model like Chapter 9. Each output is computed independently of the others using only information seen earlier in the context.



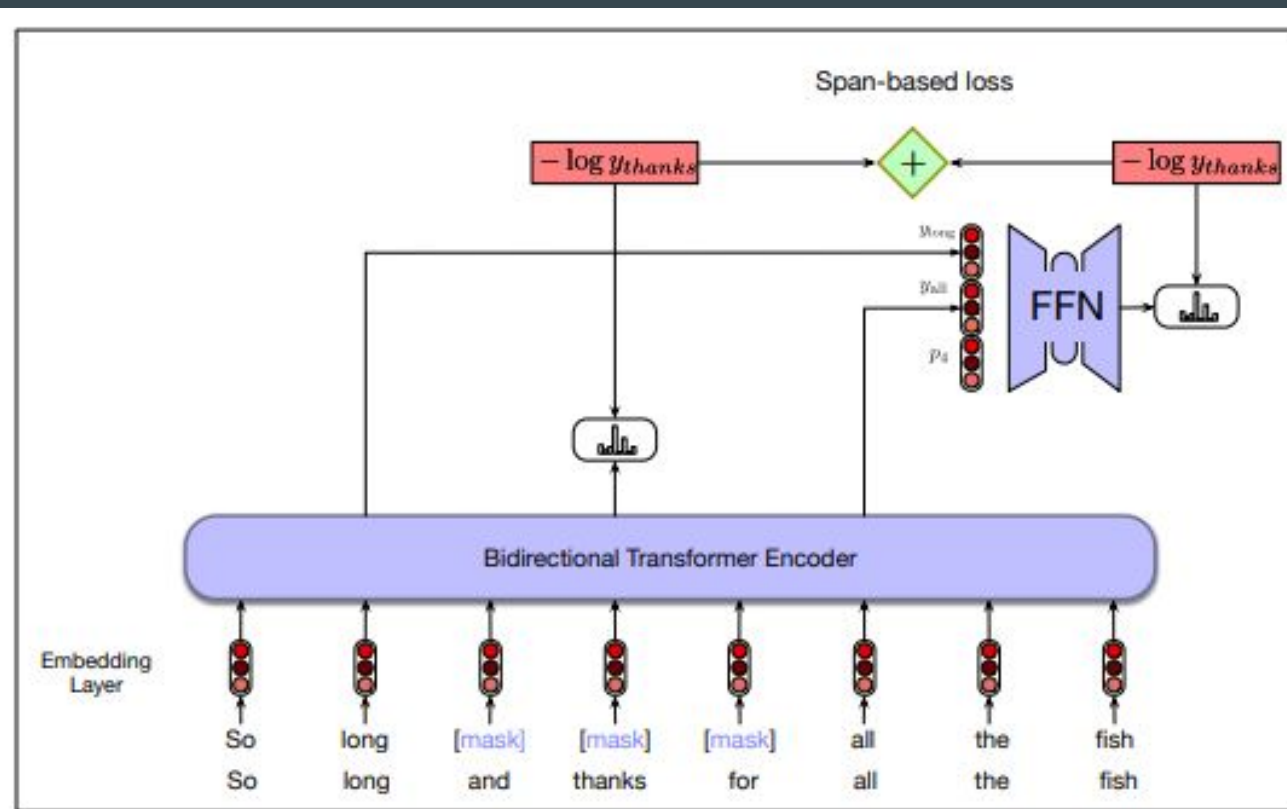
**Figure 11.2** Information flow in a bidirectional self-attention model. In processing each element of the sequence, the model attends to all inputs, both before and after the current one.

# Google BERT - Masked Language Model Training



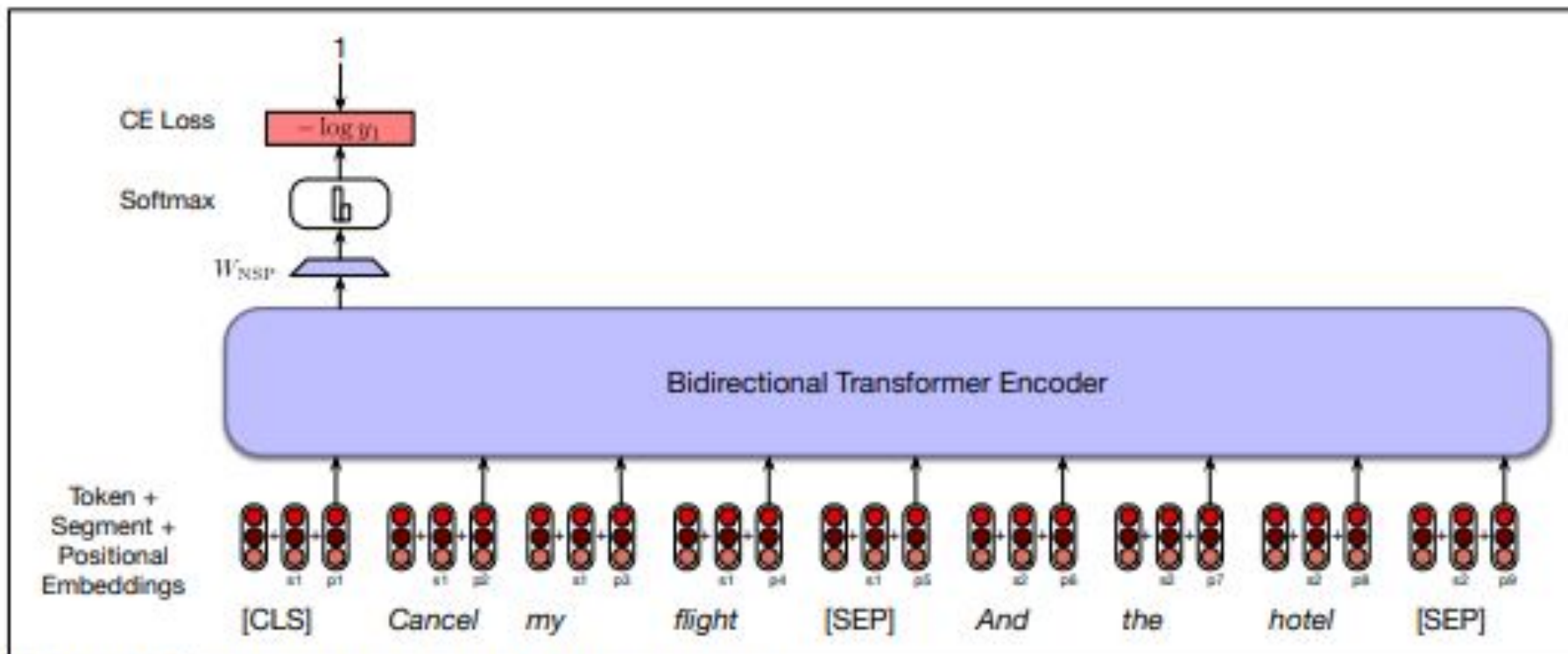
**Figure 11.5** Masked language model training. In this example, three of the input tokens are selected, two of which are masked and the third is replaced with an unrelated word. The probabilities assigned by the model to these three items are used as the training loss. (In this and subsequent figures we display the input as words rather than subword tokens; the reader should keep in mind that BERT and similar models actually use subword tokens instead.)

# Google BERT - Span-based Language Model Training



**Figure 11.6** Span-based language model training. In this example, a span of length 3 is selected for training and all of the words in the span are masked. The figure illustrates the loss computed for word *thanks*; the loss for the entire span is based on the loss for all three of the words in the span.

# Google BERT - Next Sentence Prediction



**Figure 11.7** An example of the NSP loss calculation.

# Google BERT - Pre-Training Procedure

In this work, we denote the number of layers (i.e., Transformer blocks) as  $L$ , the hidden size as  $H$ , and the number of self-attention heads as  $A$ .<sup>3</sup> We primarily report results on two model sizes: **BERT<sub>BASE</sub>** ( $L=12$ ,  $H=768$ ,  $A=12$ , Total Parameters=110M) and **BERT<sub>LARGE</sub>** ( $L=24$ ,  $H=1024$ ,  $A=16$ , Total Parameters=340M).

**Pre-training data** The pre-training procedure largely follows the existing literature on language model pre-training. For the pre-training corpus we use the BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia (2,500M words). For Wikipedia we extract only the text passages and ignore lists, tables, and headers. It is critical to use a document-level corpus rather than a shuffled sentence-level corpus such as the Billion Word Benchmark (Chelba et al., 2013) in order to extract long contiguous sequences.

Training of BERT<sub>BASE</sub> was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total).<sup>13</sup> Training of BERT<sub>LARGE</sub> was performed on 16 Cloud TPUs (64 TPU chips total). Each pre-training took 4 days to complete.