



BERT



Agenda

60
min

■ BERT

→ Pré treino e Fine Tuning

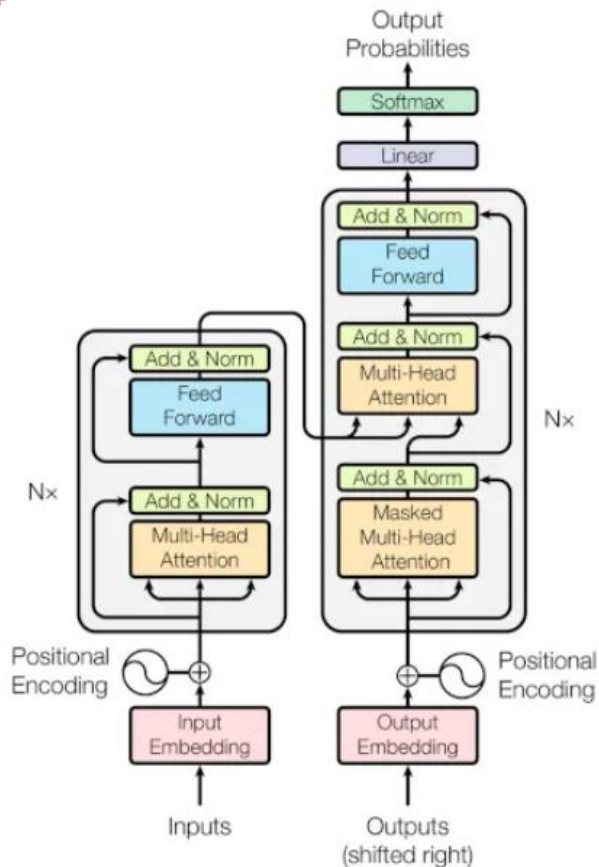
→ Tokenizers & Embeddings

→ Rorschach test com BERT Viz

→ Análise Outputs (Notebook)

BERT

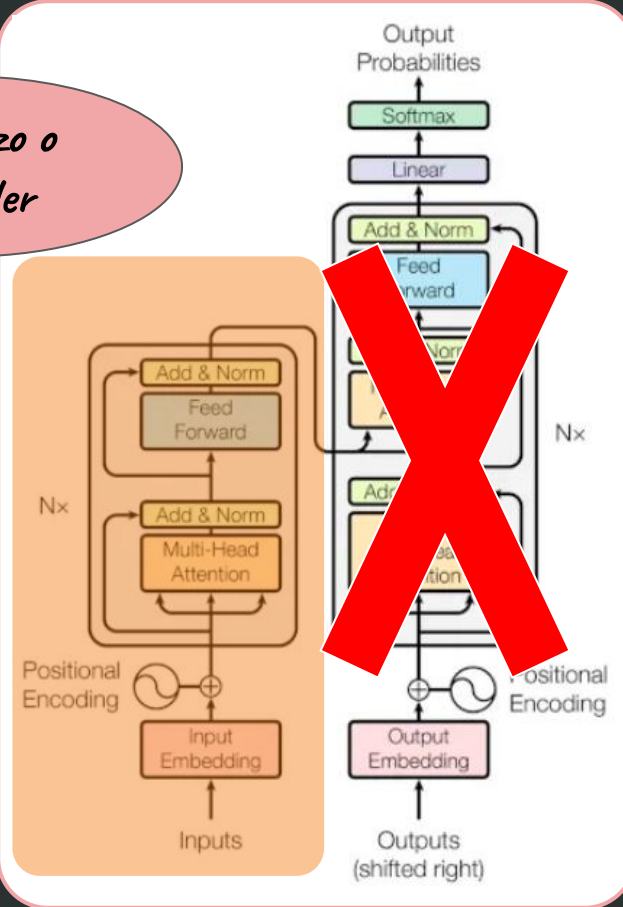
<https://arxiv.org/abs/1706.03762>



BERT



*Só utilizo o
Encoder*



BERT - GLUE Tasks

■ **CoLA** Se a sentença está escrita gramaticalmente correta

■ **MRPC** Se a sentença é paráfrase

■ **QQP** Se 2 questões são similares

■ **QNLI** Se uma sentença "B" contém uma resposta para a sentença "A"

■ **SST-2** Análise de reviews de filmes

■ **STS-B** Similaridade entre 2 sentenças

■ **MLNI** Se a sentença "B" é uma contradição de uma sentença "A"

■ **RTE** Se a sentença "B" está vinculada com uma sentença "A"

Se uma sentença "B" substitui
■ **WNLI** corretamente o pronome de uma sentença "A"

BERT - GLUE Score


















System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

BERT - GLUE Tasks

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

Benchmark ~ 90 avg.
([DeBERTA](#), Microsoft)

SuperGLUE
leaderboard
1/Dez/21

Rank	Name	Model	URL	Score
1	ERNIE Team - Baidu	 ERNIE		91.1
2	AliceMind & DRL	 Struct <u>BERT</u> + CLEVER		91.0
3	liangzhu ge	 <u>DEBERTa</u> + CLEVER		90.9
4	DeBERTa Team - Microsoft	 <u>DeBERTa</u> / TuringNLRv4		90.8
5	HFL iFLYTEK	 Mac <u>ALBERT</u> + DKM		90.7
	6	PING-AN Omni-Sinitic	 <u>ALBERT</u> + DAAF + NAS	90.6
7	T5 Team - Google	 T5		90.3
8	Microsoft D365 AI & MSR AI & GATECH	MT-DNN-SMART		89.9
	9	Huawei Noah's Ark Lab	 NEZHA-Large	89.8
	10	Zihang Dai	 Funnel-Transformer (Ensemble B10-10-10H1024)	89.7

Text to model

sentença: Quero um cartão adicional



Text to model



sentença: Quero um cartão adicional

• 1o passo: **Tokenizar** := [Quero][um][cartão][adicional]

Text to model



sentença: Quero um cartão adicional

• 1o passo: Tokenizar := [Quero][um][cartão][adicional]

• 2o passo:

one hot 😊

Quero [1,0,0,0,...,0]

um [0,1,0,0,...,0]

cartão [0,0,0,1,...,0]

adicional [0,0,1,0,...,0]



Text to model

sentença: Quero um cartão adicional

▪ 1o passo: **Tokenizar** := [Quero][um][cartão][adicional]

▪ 2o passo:

one hot 😞

Quero [1,0,0,0,...,0]

um [0,1,0,0,...,0]

cartão [0,0,0,1,...,0]

adicional [0,0,1,0,...,0]

▪ 2o passo:

embedding 😊

Quero [2.21,-3.32,...,0.89]

um [-1.27,2.80,...,4.05]

cartão [0.37,-1.98,...,3.09]

adicional [0.77,-0.88,...,2.16]

BERT - Tokenize



Formas de tokenização

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!

por
espaços: [Chewie,] [we're] [home!]

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!



por
espaços:

[Chewie , we ' re home !]

SUBÓTIMO

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!

por
espaços: [Chewie] [,] [we] ['] [re] [home] [!]

SUBÓTIMO

por
pontuação: [Chewie] [,] [we] ['] [re] [home] [!]

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!

por
espaços: [Chewie] [,] [we] ['] [re] [home] [!]

SUBÓTIMO

por
pontuação: [Chewie] [,] [we] ['] [re] [home] [!]

we're != we are ?

BERT - Tokenize



Formas de tokenização

sentença: Chewie, we're home!

por
espaços: [Chewie, we're home!]

SUBÓTIMO

por
pontuação: [Chewie, we're home!]

SUBÓTIMO

we're != we are

BERT - Tokenize



Vocab. Reduzido
+
Boas
Representações

▣ Subword tokenizers

- ▣ Byte-Pair Encoding BPE
- ▣ Unigram Subword
- ▣ Wordpiece
- ▣ Sentence piece

BERT - Tokenize



Vocab. Reduzido
+
Boas
Representações

Subword tokenizers

Byte-Pair Encoding BPE

Unigram Subword

Wordpiece

Sentence piece

`("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)` pré-tokenização

• BPE :=

BERT - Tokenize



Vocab. Reduzido
+
Boas
Representações

Subword tokenizers

- Byte-Pair Encoding BPE
- Unigram Subword
- Wordpiece
- Sentence piece

```
("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)
```

```
["a", "b", "i", "m", "t", "u", "ú"]
```

pré-tokenização
vocab. base

• BPE :=

BERT - Tokenize



Vocab. Reduzido
+
Boas
Representações

Subword tokenizers

- Byte-Pair Encoding BPE
- Unigram Subword
- Wordpiece
- Sentence piece

```
("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)
```

```
["a", "b", "i", "m", "t", "u", "ú"]
```

```
("m" "a" "u", 10), ("b" "a" "ú", 5), ("i" "t" "a" "ú", 7), ("u" "a" "u", 4)
```

pré-tokenização
vocab. base

• BPE :=



BERT - Tokenize

Vocab. Reduzido
+
Boas
Representações

■ Subword tokenizers

- Byte-Pair Encoding BPE
- Unigram Subword
- Wordpiece
- Sentence piece

```
("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)
```

```
["a", "b", "i", "m", "t", "u", "ú"]
```

```
("m" "a" "u", 10), ("b" "a" "ú", 5), ("i" "t" "a" "ú", 7), ("u" "a" "u", 4)
```

```
("m" "au", 10), ("b" "aú", 5), ("i" "t" "aú", 7), ("u" "au", 4)
```

pré-tokenização
vocab. base

■ BPE :=



BERT - Tokenize

Vocab. Reduzido
+
Boas
Representações

Subword tokenizers

Byte-Pair Encoding BPE

Unigram Subword

Wordpiece

Sentence piece

```
("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)
```

```
["a", "b", "i", "m", "t", "u", "ú"]
```

```
("m" "a" "u", 10), ("b" "a" "ú", 5), ("i" "t" "a" "ú", 7), ("u" "a" "u", 4)
```

```
("m" "au", 10), ("b" "aú", 5), ("i" "t" "aú", 7), ("u" "au", 4)
```

```
["a", "b", "i", "m", "t", "u", "ú", "au", "aú"]
```

pré-tokenização

vocab. base

novo vocab. base

BPE :=



BERT - Tokenize

Vocab. Reduzido
+
Boas
Representações

- Subword tokenizers**
- Byte-Pair Encoding BPE
 - Unigram Subword
 - Wordpiece
 - Sentence piece

BPE :=

trino

("mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)

["a", "b", "i", "m", "t", "u", "ú"]

("m" "a" "u", 10), ("b" "a" "ú", 5), ("i" "t" "a" "ú", 7), ("u" "a" "u", 4)

("m" "au", 10), ("b" "aú", 5), ("i" "t" "aú", 7), ("u" "au", 4)

["a", "b", "i", "m", "t", "u", "ú", "au", "aú"]

⋮

⋮

pré-tokenização

vocab. base

novo vocab. base

vocab. base final



BERT - Tokenize

Vocab. Reduzido
+
Boas
Representações

- Subword tokenizers**
- Byte-Pair Encoding BPE
 - Unigram Subword
 - Wordpiece
 - Sentence piece

BPE :=

GPT-2
Roberta
XLNet
~50K tokens

training

["mau", 10), ("baú", 5), ("itaú", 7), ("uau", 10)

["a", "b", "i", "m", "t", "u", "ú"]

["m" "a" "u", 10), ("b" "a" "ú", 5), ("i" "t" "a" "ú", 7), ("u" "a" "u", 4)

["m" "au", 10), ("b" "aú", 5), ("i" "t" "aú", 7), ("u" "au", 4)

["a", "b", "i", "m", "t", "u", "ú", "au", "aú"]

...

pré-tokenização vocab. base

novo vocab. base

vocab. base final

prever "auê" → ["au", "<unk>"]

mBERT, BERTimbau e BERTaú - Tokenizers



■ sentença: Never tell me the odds.

■ mBERT

■ BERTimbau

■ BERTaú

mBERT, BERTimbau e BERTaú - Tokenizers



■ sentença: Never tell me the odds.

■ mBERT

`['never', 'tell', 'me', 'the', 'odds', '.']`

■ BERTimbau

■ BERTaú

mBERT, BERTimbau e BERTaú - Tokenizers



■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

mBERT, BERTimbau e BERTaú - Tokenizers



■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

['neve', '##r', 'tel', '##l', 'me', 'the', 'od', '##ds', '.']

mBERT, BERTimbau e BERTaú - Tokenizers



■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

['neve', '##r', 'tel', '##l', 'me', 'the', 'od', '##ds', '.']

■ sentença: Quero um cartão de crédito.

■ mBERT

■ BERTimbau

■ BERTaú



mBERT, BERTimbau e BERTaú - Tokenizers

■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

['neve', '##r', 'tel', '##l', 'me', 'the', 'od', '##ds', '.']

■ sentença: Quero um cartão de crédito.

■ mBERT

['quer', '##o', 'um', 'carta', '##o', 'de', 'credito']

■ BERTimbau

■ BERTaú



mBERT, BERTimbau e BERTaú - Tokenizers

■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

['neve', '##r', 'tel', '##l', 'me', 'the', 'od', '##ds', '.']

■ sentença: Quero um cartão de crédito.

■ mBERT

['quer', '##o', 'um', 'carta', '##o', 'de', 'credito']

■ BERTimbau

['Quer', '##o', 'um', 'cartão', 'de', 'crédito']

■ BERTaú



mBERT, BERTimbau e BERTaú - Tokenizers

■ sentença: Never tell me the odds.

■ mBERT

['never', 'tell', 'me', 'the', 'odds', '.']

■ BERTimbau

['Ne', '##ver', 'tel', '##l', 'me', 'the', 'o', '##dd', '##s', '.']

■ BERTaú

['neve', '##r', 'tel', '##l', 'me', 'the', 'od', '##ds', '.']

■ sentença: Quero um cartão de crédito.

■ mBERT

['quer', '##o', 'um', 'carta', '##o', 'de', 'credito']

■ BERTimbau

['Quer', '##o', 'um', 'cartão', 'de', 'crédito']

■ BERTaú

['quero', 'um', 'cartao', 'de', 'credito']

BERTaú - Tokenize



Tokens especiais

■ BERTaú - special tokens

0: [PAD]

1: [UNK]

2: [CLS]

3: [SEP]

4: [MASK]

é relacionado com o `max_len` ex. para `max_len=10`

"Uma sentença": [2, 335, 12286, 155, 951, 3, 0, 0, 0, 0]

"E outra": [2, 37, 929, 3, 0, 0, 0, 0, 0, 0]

BERTaú - Tokenize



Tokens especiais

• **BERTaú** - special tokens

0: [PAD]

1: [UNK]

2: [CLS]

3: [SEP]

4: [MASK]



quando não existe a word/subword no vocab. base.
O BERTaú possui [UNK] para qualquer número
(dados anonimizados).

BERTaú - Tokenize



Tokens especiais

• **BERTaú** - special tokens

0: [PAD]

1: [UNK]

2: [CLS]

3: [SEP]

4: [MASK]

[CLS] Uma sentença. [SEP] E outra! [SEP]

[2, 335, 12286, 155, 951, 3, 37, 929, 3, 0]

BERTaú - Tokenize



Tokens especiais

• **BERTaú** - special tokens

0: [PAD]

1: [UNK]

2: [CLS]

3: [SEP]

4: [MASK]



usado durante o pré-treino e downstream tasks de LM

BERTaú e BERT - Tokenizers



Tokens especiais

• **BERTaú** - special tokens

0: [PAD]

1: [UNK]

2: [CLS]

3: [SEP]

4: [MASK]

• **mBERT** - special tokens

0: [PAD]

100: [UNK]

101: [CLS]

102: [SEP]

103: [MASK]



Uma sentença. E outra!



```
['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
```

```
['[PAD]']  
max_length=12
```

↑ tokens

Uma sentença. E outra!



↑ input_ids

```
['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
```

```
 '[PAD]']  
max_length=12
```

↑ tokens

Uma sentença. E outra!



```
[ 2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0 ]
```

↑ input_ids

```
['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
```

```
 '[PAD]']
```

max_length=12

↑ tokens

Uma sentença. E outra!



↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



[0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



```
[ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0 ]
```

↑ attention_mask

```
[ 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0 ]
```

↑ token_type_ids

```
[ 2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0 ]
```

↑ input_ids

```
['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
```

```
 '[PAD]']
```

↑ tokens

max_length=12

Uma sentença. E outra!



↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']

↑ tokens

max_length=12

Uma sentença. E outra!



https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0

]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']

↑ tokens

max_length=12

Uma sentença. E outra!



↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']

↑ tokens

max_length=12

Uma sentença. E outra!



↑ transform_Layer_12
:
:
↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



↑ transform_Layer_12
:
:
↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]
↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]
↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]
↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]
↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
[PAD]']

max_length=12

↑ tokens

Uma sentença. E outra!



↑ transform_Layer_12
:
:
↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]
↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]
↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]
↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]
↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,
[PAD]']

max_length=12

↑ tokens

Uma sentença. E outra!



↑ transform_Layer_12
:
:
↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

['[PAD]']
max_length=12

↑ tokens

Uma sentença. E outra!



↑ transform_Layer_12
:
:
↑ transform_Layer_1

[0	,	1	,	2	,	3	,	4	,	5	,	6	,	7	,	8	,	9	,	10	,	11]		
	↑																									
[1	,	1	,	1	,	1	,	1	,	1	,	1	,	1	,	1	,	1	,	1	,	1	,	0]
	↑																									
[0	,	0	,	0	,	0	,	0	,	0	,	0	,	0	,	1	,	1	,	1	,	1	,	0]
	↑																									
[2	,	335	,	12286	,	155	,	951	,	18	,	3	,	37	,	929	,	5	,	3	,	0]		
	↑																									
['[CLS]'	,	'uma'	,	'sente'	,	'##n'	,	'##ca'	,	'.'	,	'[SEP]'	,	'e'	,	'outra'	,	'!'	,	'[SEP]'	,	,			

['[PAD]']

max_length=12

↑ tokens

Uma sentença. E outra!



Prediction

'[CLS]'

↑ transform_Layer_12

⋮

↑ transform_Layer_1

[0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11]

↑ positional_encoding

[1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0]

↑ attention_mask

[0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 , 1 , 0]

↑ token_type_ids

[2 , 335 , 12286 , 155 , 951 , 18 , 3 , 37 , 929 , 5 , 3 , 0]

↑ input_ids

['[CLS]', 'uma', 'sente', '##n', '##ca', '.', '[SEP]', 'e', 'outra', '!', '[SEP]', ,

'[PAD]']

↑ tokens

max_length=12

Uma sentença. E outra!

BERT - transfer learning

Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$



BERT - transfer learning



Pre training + Fine Tuning = Transfer Learning

\$\$\$\$

\$

2 tasks durante o treinamento

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

BERT - NSP



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

NSP: O BERT é alimentado com pares de sentenças. Metade das vezes a 2a sentença segue imediatamente a primeira e a outra metade não.

BERT - NSP



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

NSP: O BERT é alimentado com pares de sentenças. Metade das vezes a 2a sentença segue imediatamente a primeira e a outra metade não.

"Help me, Obi-Wan Kenobi. You're my only hope."

BERT - NSP



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

NSP: O BERT é alimentado com pares de sentenças. Metade das vezes a 2a sentença segue imediatamente a primeira e a outra metade não.

"Help me, Obi-Wan Kenobi. You're my only hope."

É NS?

BERT - MLM



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

MLM: Por que precisa de [MASK] 🙄?

BERT - MLM

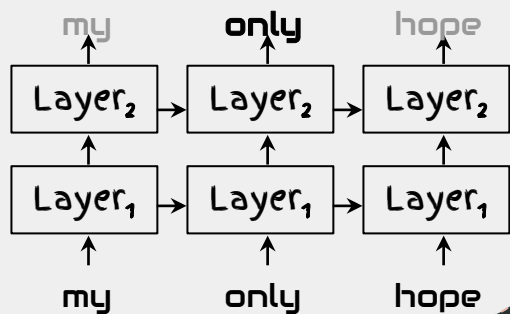


Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

MLM: Por que precisa de [MASK] ☹️?

Contexto Unidirecional (LSTM)



BERT - MLM



Pre training + Fine Tuning = Transfer Learning

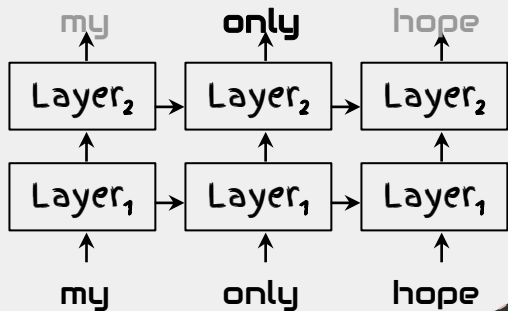
\$\$\$\$

\$

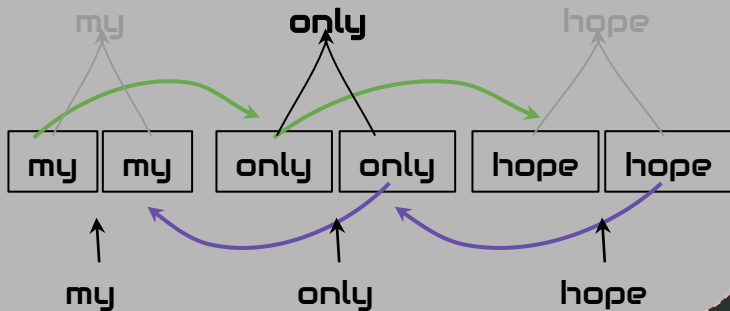
MLM: Por que precisa de [MASK] 😊?

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

Contexto Unidirecional (LSTM)



Contexto Bidirecional (Bi-LSTM)



BERT - MLM

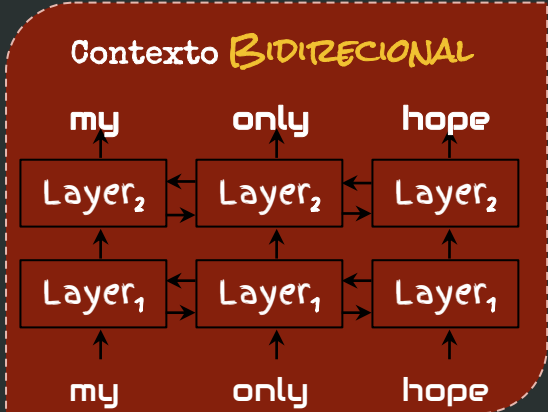
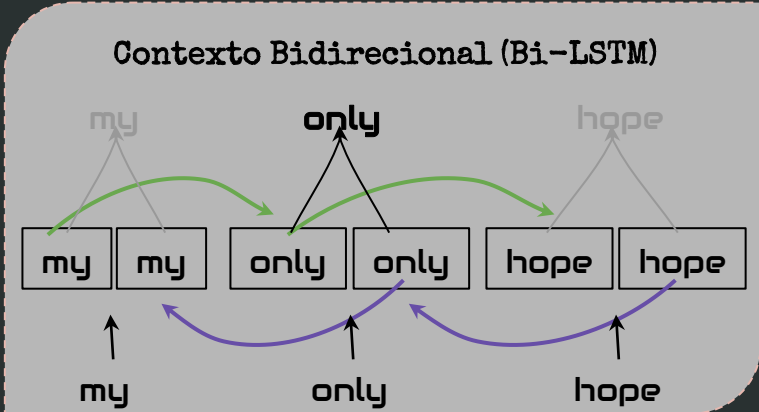
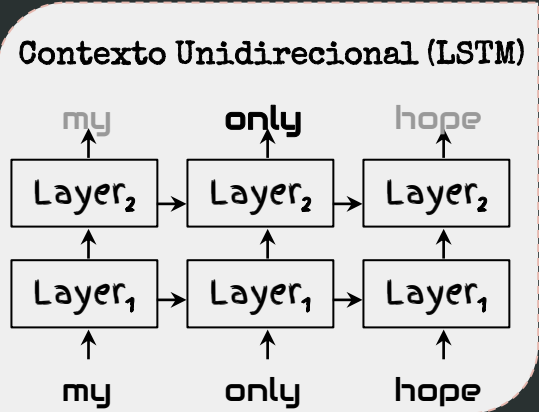


Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

- 1. Next Sentence Prediction (NSP)
- 2. Masked Language Model (MLM)

MLM: Por que precisa de [MASK] 🤖?

BERT
BIDIRECIONAL



BERT - MLM



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

MLM: Ao todo 15% do dataset é transformado com MLM que possui 3 formas:

80% de 15%: "[MASK] me, Obi-Wan Kenobi. You're my only hope."

BERT - MLM



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

MLM: Ao todo 15% do dataset é transformado com MLM que possui 3 formas:

80% de 15%: "[MASK] me, Obi-Wan Kenobi. You're my only

hope."

10% de 15%: "Find me, Obi-Wan Kenobi. You're my only hope."

BERT - MLM



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

1. Next Sentence Prediction (NSP)
2. Masked Language Model (MLM)

MLM: Ao todo 15% do dataset é transformado com MLM que possui 3 formas:

80% de 15%: "[MASK] me, Obi-Wan Kenobi. You're my only

hope."

10% de 15%: "Find me, Obi-Wan Kenobi. You're my only hope."

10% de 15%: "Help me, Obi-Wan Kenobi. You're my only hope."

BERT - Pre training



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

Uma instância de treino consiste de um pedaço de texto com MLM + NSP. As previsões de MLM e NSP são combinadas em uma única loss. De acordo com o artigo, apêndice A.2: *"The training loss is the sum of the mean masked LM likelihood and the mean next sentence prediction likelihood."*

BERT - Pre training



Pre training + Fine Tuning = Transfer Learning
\$\$\$\$ \$

Uma instância de treino consiste de um pedaço de texto com MLM + NSP. As previsões de MLM e NSP são combinadas em uma única loss. De acordo com o artigo, apêndice A.2: *"The training loss is the sum of the mean masked LM likelihood and the mean next sentence prediction likelihood."*

Exemplo de uma instância de treino

Help me, Obi-Wan Kenobi. You're my **unique** [MASK].

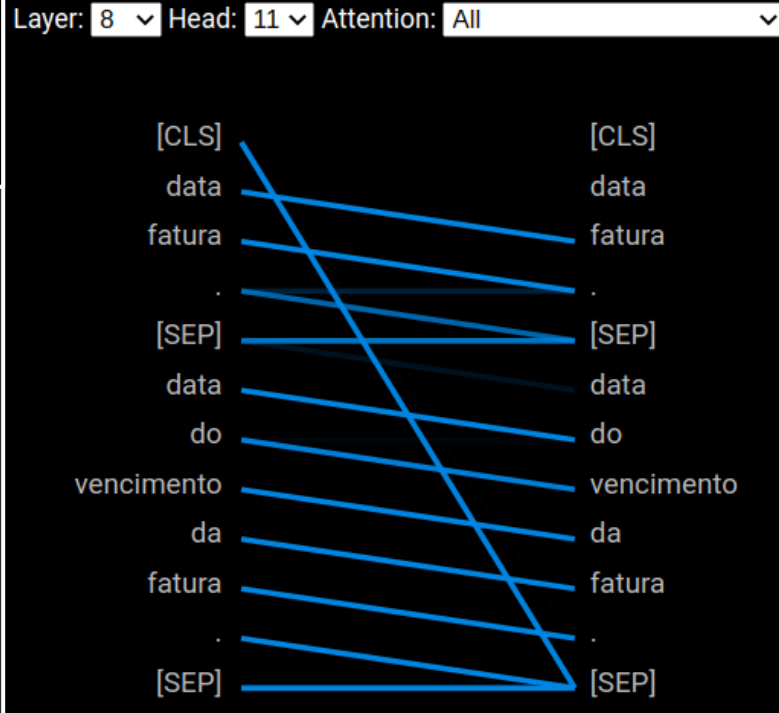
Label: IsNext



BERT_{pt} - Rorschach Test

- sentença 1: Data fatura.
- sentença 2: Data do vencimento da fatura.

Next Word Attention

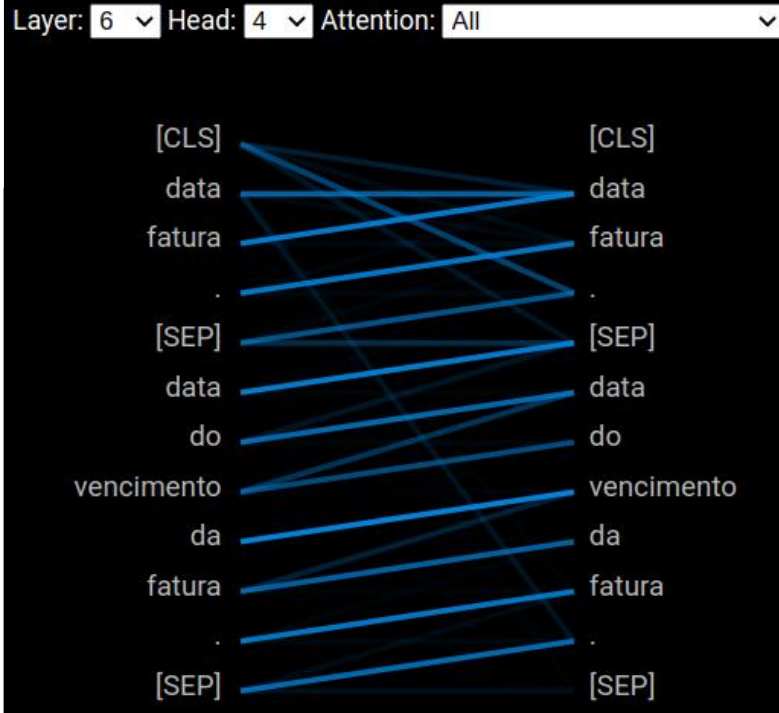




BERT - Rorschach Test

- sentença 1: Data fatura.
- sentença 2: Data do vencimento da fatura.

Previous Word Attention

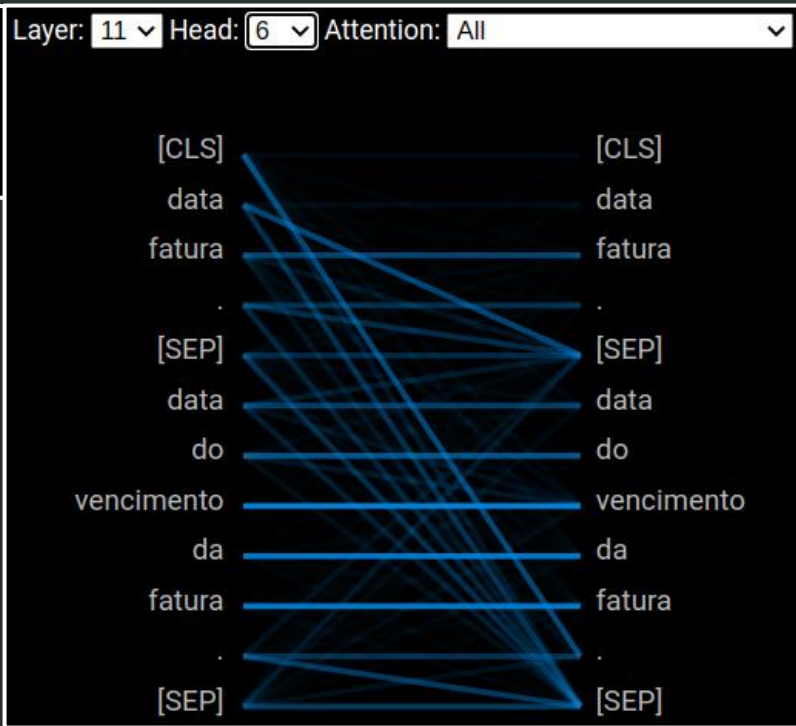




BERTaú - Rorschach Test

- sentença 1: Data fatura.
- sentença 2: Data do vencimento da fatura.

Identical Words Attention

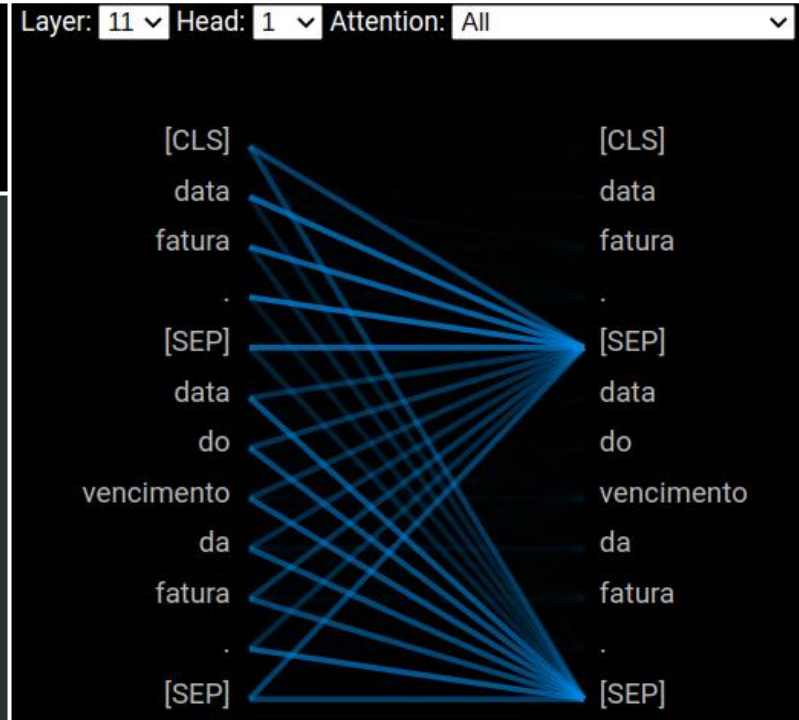




BERT_{pt} - Rorschach Test

- sentença 1: Data fatura.
- sentença 2: Data do vencimento da fatura.

[SEP]'s Attention

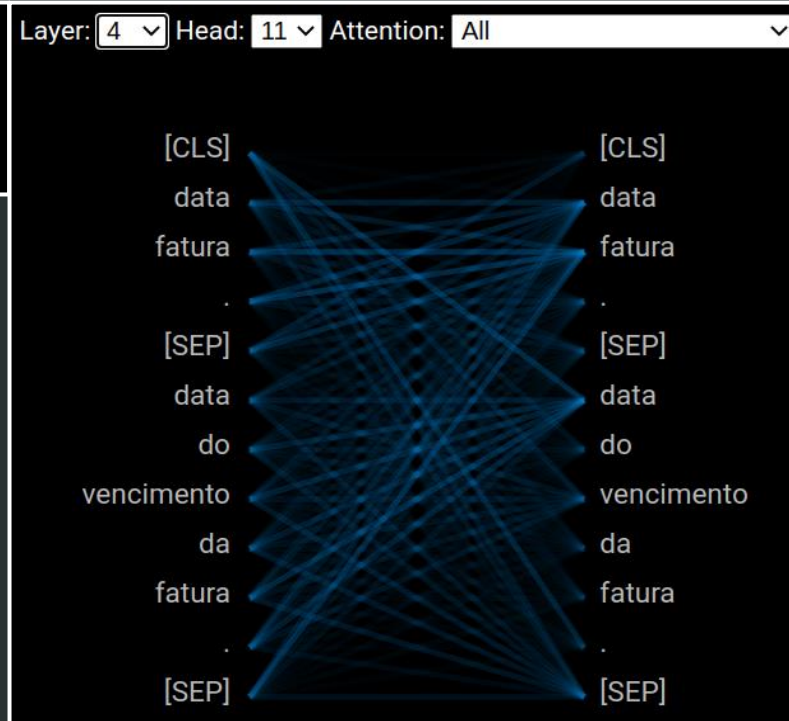


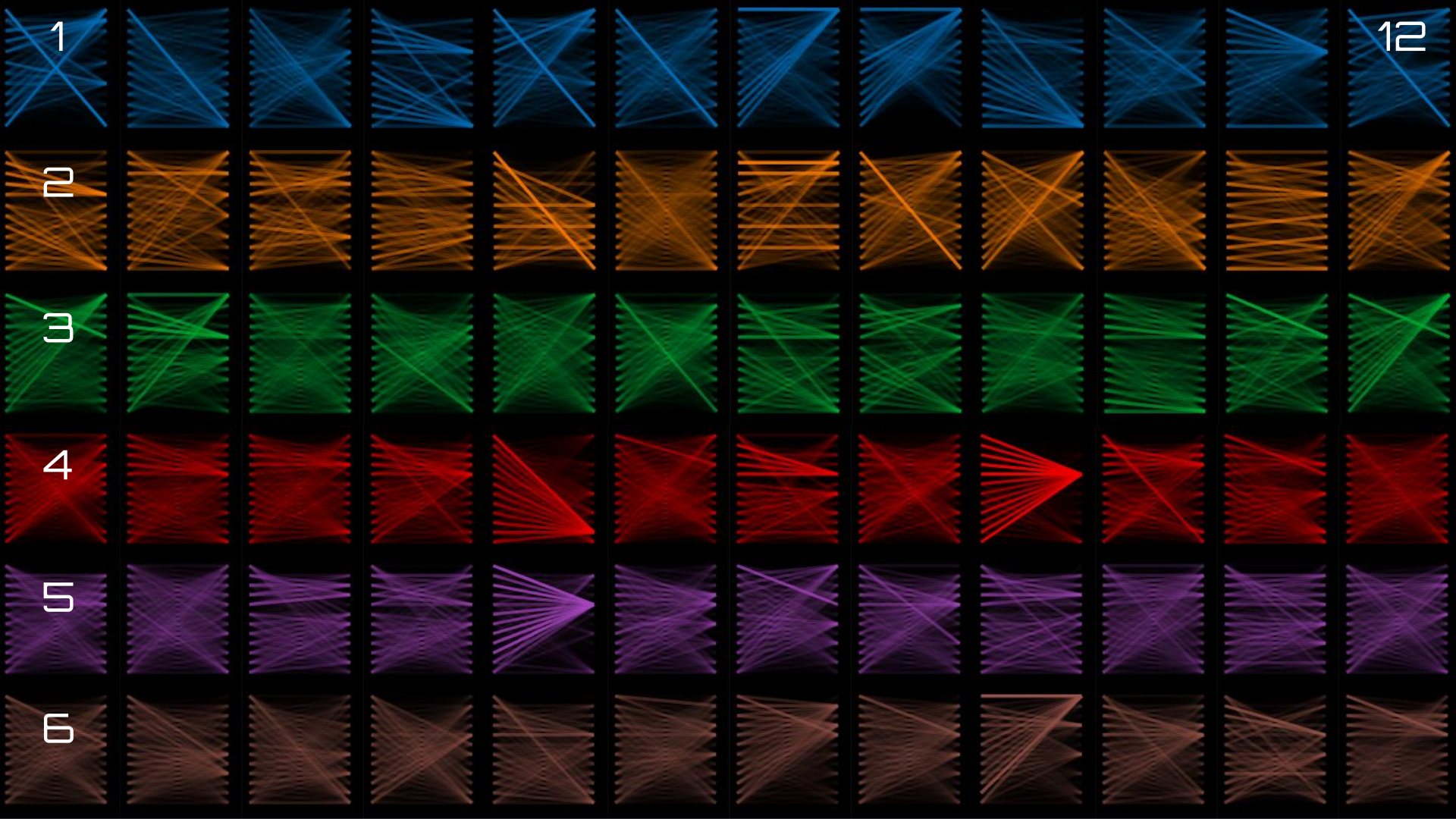


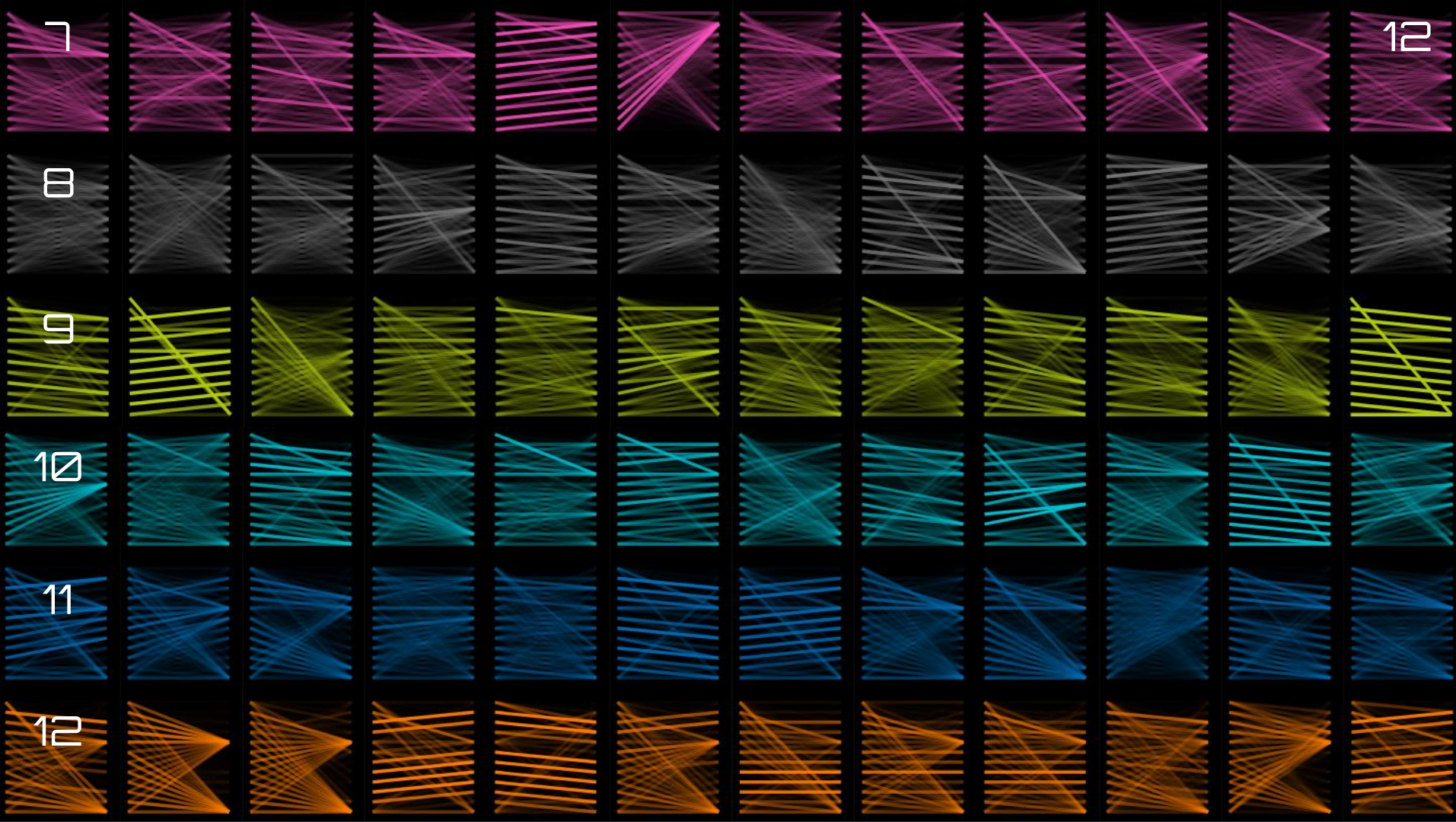
BERTaú - Rorschach Test

- sentença 1: Data fatura.
- sentença 2: Data do vencimento da fatura.

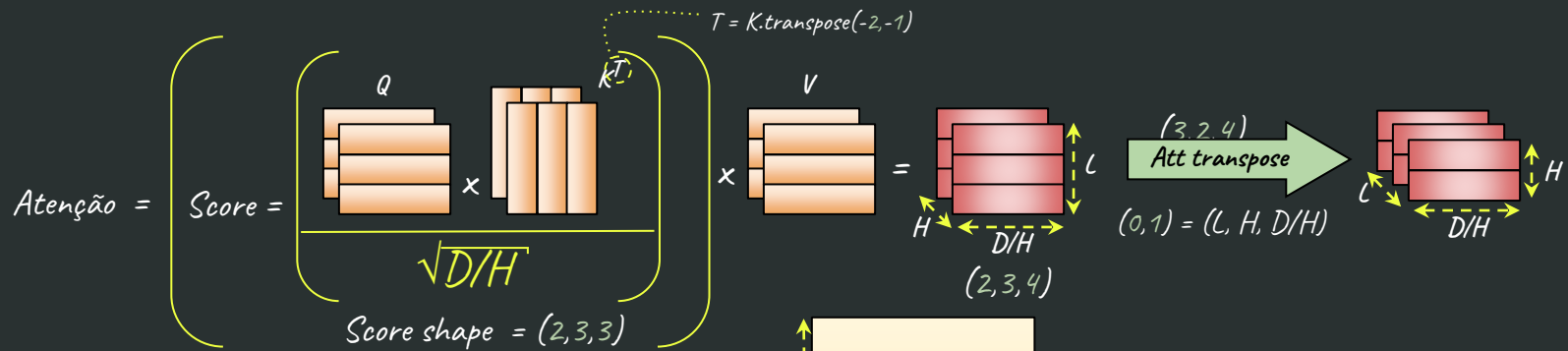
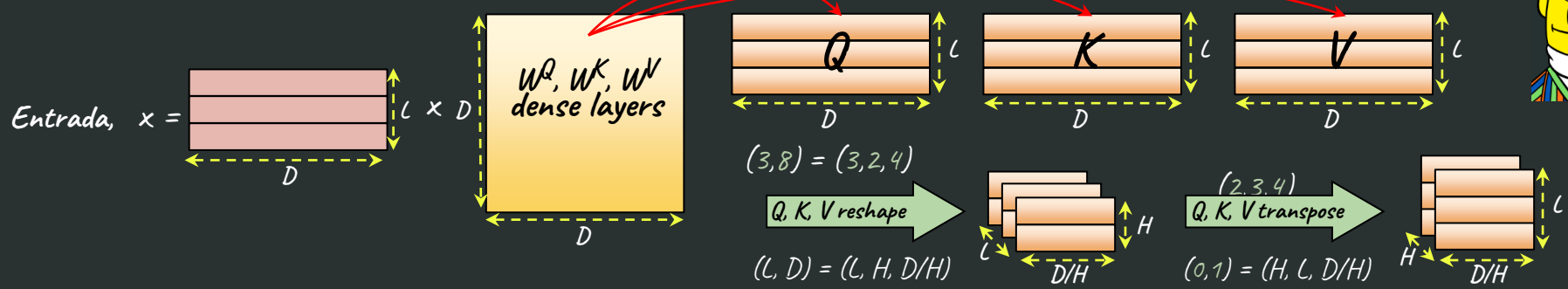
BoW
Attention



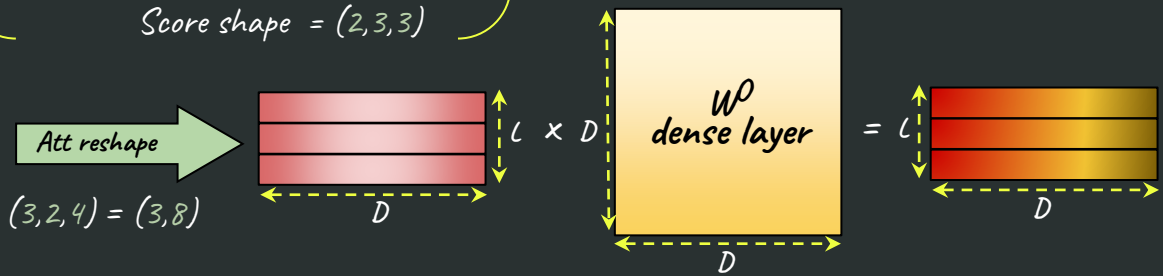




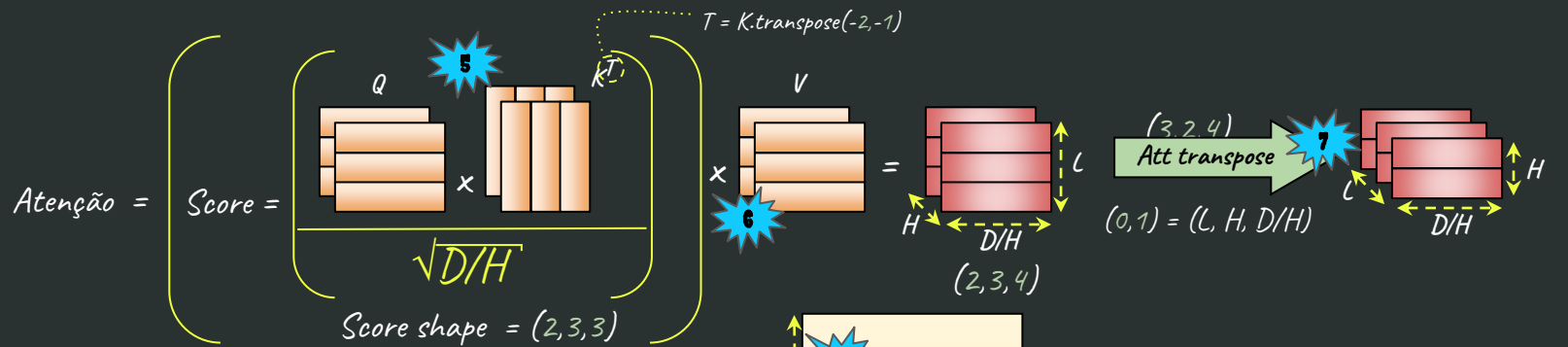
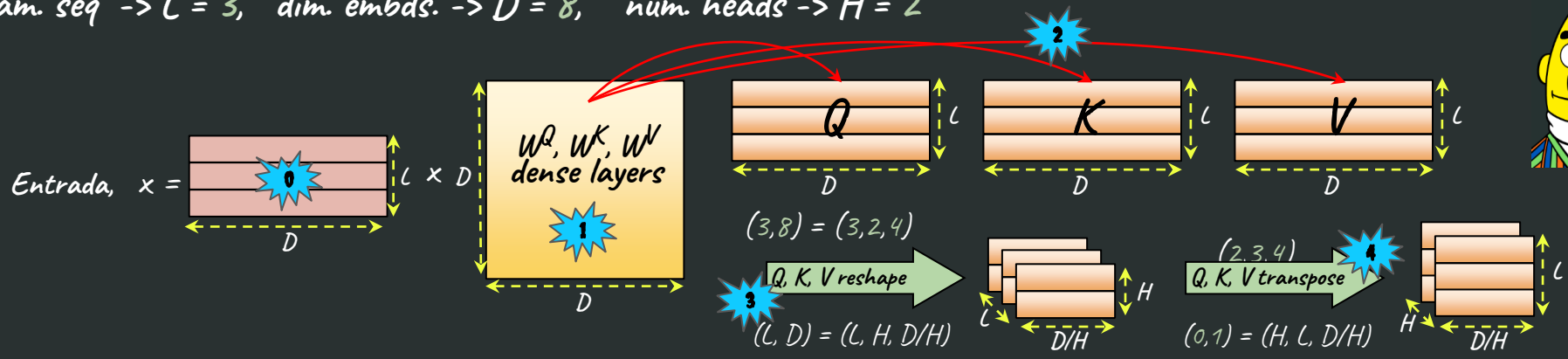
tam. seq $\rightarrow L = 3$, dim. embds. $\rightarrow D = 8$, num. heads $\rightarrow H = 2$



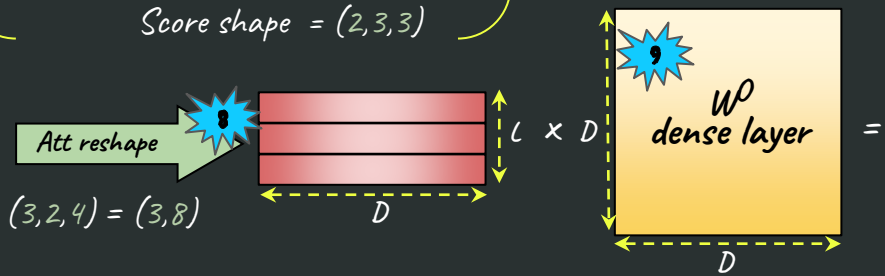
Hands On
Self Attention
Média
Ponderada



tam. seq $\rightarrow L = 3$, dim. embds. $\rightarrow D = 8$, num. heads $\rightarrow H = 2$



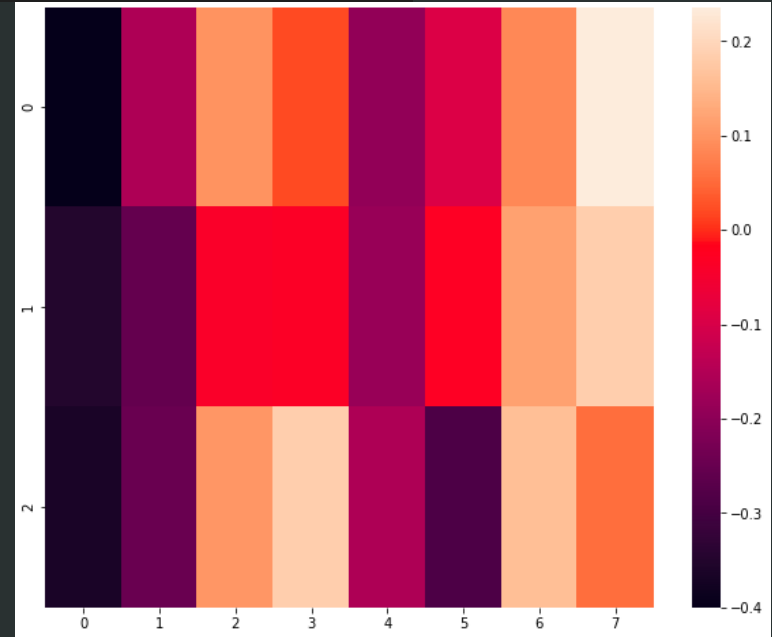
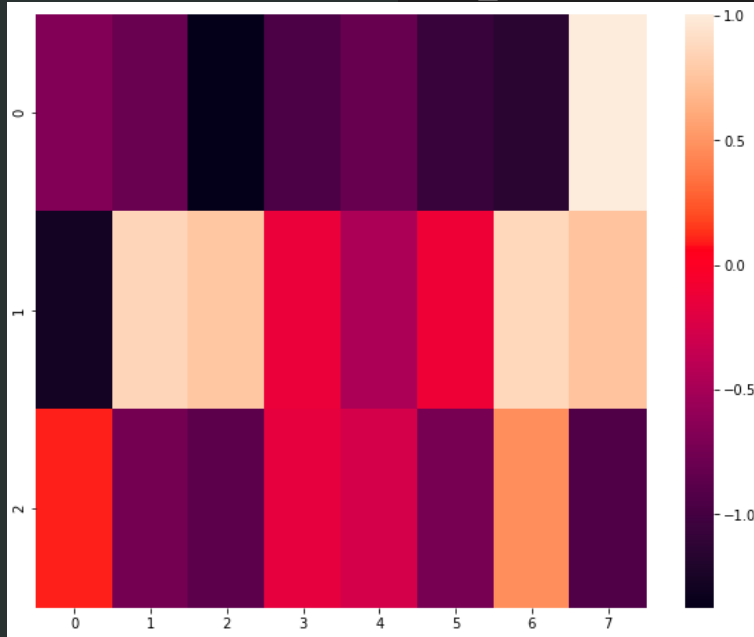
Hands On
Self Attention
Média
Ponderada



[Self Attention Step by Step.ipynb](#)
[Animação - SA.ipynb](#)



```
text_tokens = ['Quero', 'um', 'cartão']  
text_ids = [27, 11, 83] # tokens sintéticos
```



• Embeddings

• ATT

DeepMind Attention explained:

<https://www.youtube.com/watch?v=AliwuClvH6k&t=5559s>

BERTaú - Attention Viz in SA



Positive Sample

Legend: ■ Negative □ Neutral ■ Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
Positivo	Positivo (0.92)	Positivo	1.66	[CLS] atendimento muito bom , problema foi resolvido ! [SEP]

```
[(' [CLS]', 0.0), ('atendimento', 0.09), ('muito', 0.57), ('bom', 0.70), (',', 0.03), ('problema', -0.07), ('foi', 0.01), ('resolvido', -0.06), ('!', 0.38), (' [SEP]', 0.0)]
```

Negative Sample

Legend: ■ Negative □ Neutral ■ Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
Negativo	Negativo (0.03)	Neutro	-1.94	[CLS] quero cancelar, muito ruim , pessimo , nao resolveu meu problema . [SEP]

```
[(' [CLS]', 0.0), ('quero', 0.10), ('cancelar', 0.039), (',', 0.03), ('muito', -0.19), ('ruim', -0.18), (',', 0.001), ('pessimo', -0.83), (',', 0.01), ('nao', -0.19), ('resolveu', -0.29), ('meu', -0.02), ('problema', -0.10), ('.', -0.30), (' [SEP]', 0.0)]
```

Referências



1. Artigo BERT: [arxiv](#)
2. BERT Viz: [github-repo](#)
3. Transformers Interpret: [github-repo](#)
4. Artigo Attention is all you need: [arxiv](#)
5. Self Attention explained: [DeepMind-youtube channel](#)
6. Artigo (livro?) de IR com Transformers: [arxiv 155 pages](#)
7. UvA DLC (tutorial 6): [github page](#)