# lab6

November 26, 2022

[ ]:
```
import pandas as pd
import os
import re
```

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[ ]:
```
!python -V
```

```
Python 3.7.15
```

[ ]:
```
!pip install transformers
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Collecting transformers
  Downloading transformers-4.24.0-py3-none-any.whl (5.5 MB)
     |                  | 5.5 MB 5.2 MB/s
Requirement already satisfied: tqdm>=4.27 in
/usr/local/lib/python3.7/dist-packages (from transformers) (4.64.1)
Collecting tokenizers!=0.11.3,<0.14,>=0.11.1
  Downloading
tokenizers-0.13.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (7.6
MB)
     |                  | 7.6 MB 24.0 MB/s
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.7/dist-packages (from transformers) (2022.6.2)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-
packages (from transformers) (1.21.6)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-
packages (from transformers) (3.8.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.7/dist-
packages (from transformers) (6.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-
packages (from transformers) (21.3)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-
```

```
packages (from transformers) (2.23.0)
Collecting huggingface-hub<1.0,>=0.10.0
  Downloading huggingface_hub-0.11.0-py3-none-any.whl (182 kB)
     |                  | 182 kB 44.7 MB/s
Requirement already satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from transformers) (4.13.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.7/dist-packages (from huggingface-
hub<1.0,>=0.10.0->transformers) (4.1.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from packaging>=20.0->transformers)
(3.0.9)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-
packages (from importlib-metadata->transformers) (3.10.0)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests->transformers) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (2022.9.24)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests->transformers) (1.24.3)
Installing collected packages: tokenizers, huggingface-hub, transformers
Successfully installed huggingface-hub-0.11.0 tokenizers-0.13.2
transformers-4.24.0
```

2. Download three Polish models from the Huggingface repository.

Zadanie zostało wykonane przy pomocy biblioteki transformers. Wybrane modele to: 1. bert-base-multilingual-cased, 2. xlm-roberta-base 3. clips/mfaq

Dużo modeli oznaczonych jako te działające z jezykiem polskim słabo sobie z nim radzi jak np clips/mfaq. Modele zostały wczytane przez pipeline

[ ]:
```
from transformers import pipeline
```

[ ]:
```
unmasker_bert = pipeline('fill-mask', model='bert-base-multilingual-cased')
```

```
Downloading:   0%|          | 0.00/625 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/714M [00:00<?, ?B/s]

Some weights of the model checkpoint at bert-base-multilingual-cased were not
used when initializing BertForMaskedLM: ['cls.seq_relationship.weight',
'cls.seq_relationship.bias']
- This IS expected if you are initializing BertForMaskedLM from the checkpoint
of a model trained on another task or with another architecture (e.g.
initializing a BertForSequenceClassification model from a BertForPreTraining
model).
- This IS NOT expected if you are initializing BertForMaskedLM from the
```

```
checkpoint of a model that you expect to be exactly identical (initializing a
BertForSequenceClassification model from a BertForSequenceClassification model).

Downloading:   0%|          | 0.00/29.0 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/996k [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/1.96M [00:00<?, ?B/s]
```

[ ]:
```
unmasker_roberta = pipeline('fill-mask', model='xlm-roberta-base')
```

```
Downloading:   0%|          | 0.00/615 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/1.12G [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/5.07M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/9.10M [00:00<?, ?B/s]
```

[ ]:
```
unmasker_clips = pipeline('fill-mask', model='clips/mfaq')
```

```
Downloading:   0%|          | 0.00/778 [00:00<?, ?B/s]

/usr/local/lib/python3.7/dist-packages/transformers/configuration_utils.py:370:
UserWarning: Passing `gradient_checkpointing` to a config initialization is
deprecated and will be removed in v5 Transformers. Using
`model.gradient_checkpointing_enable()` instead, or if you are using the
`Trainer` API, pass `gradient_checkpointing=True` in your `TrainingArguments`.
  "Passing `gradient_checkpointing` to a config initialization is deprecated and
will be removed in v5 "

Downloading:   0%|          | 0.00/1.11G [00:00<?, ?B/s]

Some weights of XLMRobertaForMaskedLM were not initialized from the model
checkpoint at clips/mfaq and are newly initialized:
['lm_head.layer_norm.weight', 'lm_head.dense.weight', 'lm_head.dense.bias',
'lm_head.layer_norm.bias', 'lm_head.bias']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.

Downloading:   0%|          | 0.00/464 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/5.07M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/9.08M [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/48.0 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/294 [00:00<?, ?B/s]
```

3. Devise a method to test if the langage model understands Polish cases. E.g. testing for nominal case could be expressed as “Warszawa to największe [MASK]”, and the masked word should be in nominative case. Create sentences for each case.

test_cases to funkcja ktora bierze najbardziej prawdopodobna odpowiedz z modelu nastepnie printuje dane slowo wraz z przypadkiem ktorym powinno ono byc.

[ ]:
```
def test_cases(unmasker):
  words = []
  words.append(unmasker(f"{unmasker.tokenizer.mask_token} Jagielloński to␣
 ↪największa uczelnia w Krakowie.")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Nie lubię␣
 ↪{unmasker.tokenizer.mask_token} Jagielońskiego na którym studiuję␣
 ↪")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Teraz przyglądam␣
 ↪się {unmasker.tokenizer.mask_token} Jagiellońskiemu na którym␣
 ↪studiuję")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Bardzo lubię␣
 ↪{unmasker.tokenizer.mask_token} Jagielloński na którym␣
 ↪studiuję")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Często rozmawiam z␣
 ↪{unmasker.tokenizer.mask_token} Jagiellońskim na którym studiuję␣
 ↪")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Często rozmawiam o␣
 ↪{unmasker.tokenizer.mask_token} Jagiellońskim na którym studiuję␣
 ↪")[0]['token_str'])
  words.append(unmasker(f"Jestem studentem na uniwersytecie. Hej! {unmasker.
 ↪tokenizer.mask_token} Jagielloński.")[0]['token_str'])

  words = [word.lower() for word in words]

  correct = ['uniwersytet', 'uniwersytetu', 'uniwersytetowi', 'uniwersytet',␣
 ↪'uniwersytetem', 'uniwersytecie', 'uniwersytecie']
  przypadki = ['Mianownik',␣
 ↪'Dopełniacz','Celownik','Biernik','Narzędnik','Miejscownik','Wołacz']
  count = 0
  for i in range(len(words)):
    # if correct[i] == words[i]:
    print(f'{przypadki[i]} - {words[i]}')
  #   else:
  #     print(f'{przypadki[i]} - {words[i]} --- Poprawna forma to {correct[i]}')
  #     count+=1
  # print(f'Poprawne formy: {7-count}')
```

[ ]:
```
test_cases(unmasker_bert)
```

```
Mianownik - uniwersytet
Dopełniacz - uniwersytetu
Celownik - uniwersytetu
Biernik - uniwersytet
Narzędnik - uniwersytet
Miejscownik - uniwersytecie
Wołacz - uniwersytet
```

[ ]:
```
test_cases(unmasker_roberta)
```

```
Mianownik - campus
Dopełniacz - uniwersytetu
Celownik - uniwersytetu
Biernik - kampus
Narzędnik - uniwersytetu
Miejscownik - uniwersytetu
Wołacz - jestem
```

[ ]:
```
test_cases(unmasker_clips)
```

```
Mianownik - watson
Dopełniacz - fqinj
Celownik - watson
Biernik - watson
Narzędnik - watson
Miejscownik - watson
Wołacz - watson
```

4. Devise a method to test long-range relationships such as gender. E.e. you can use two verbs where withe masculine and feminine gender, where one of the verbs is masked. Both verbs should have the same gender, assuming the subject is the same. Define at least 3 such sentences.

test_sex to funkcja ktora bierze najbardziej prawdopodobna odpowiedz z modelu i sprawdza płeć. w 1 zdaniu powinna być żeńska w 2 męska i w 3 inna.

[ ]:
```
def test_sex(unmasker):
  words = []
  words.append(unmasker(f"Dziewczynka poszła do sklepu i {unmasker.tokenizer.
 ↪mask_token} jajka")[0]['token_str'])
  words.append(unmasker(f"chłopiec grał na komputerze przez 2 godziny i␣
 ↪{unmasker.tokenizer.mask_token} wynik 500")[0]['token_str'])
  words.append(unmasker(f"Kacząтko było w jeziorze samo bo jako jedyne␣
 ↪{unmasker.tokenizer.mask_token} pływać")[0]['token_str'])

  words = [word.lower() for word in words]

  correct = ['uniwersytet', 'uniwersytetu', 'uniwersytetowi', 'uniwersytet',␣
 ↪'uniwersytetem', 'uniwersytecie', 'uniwersytecie']
  przypadki = ['F', 'M', 'N']
  count = 0
  for i in range(len(words)):
    print(f'{przypadki[i]} - {words[i]}')
```

[ ]:
```
test_sex(unmasker_bert)
```

```
F - miała
```

```
M - uzyskał
N - było
```

[ ]:
```
test_sex(unmasker_roberta)
```

```
F - kupi
M - miał
N - mógł
```

[ ]:
```
test_sex(unmasker_clips)
```

```
F - watson
M - qishloq
N - watson
```

5. Check if the model captures real-world knolwedge. For instance a sentence "[MASK] wrze w temperaturze 100 stopni, a zamarza w temperaturze 0 stopni Celsjusza." checks if the model "knows" the description of water. Define at least 3 such sentences.

Na początku sprawdzane jest przykładowe zdanie :

[MASK] wrze w temperaturze 100 stopni, a zamarza w temperaturze 0 stopni Celsjusza." Wybrane modele radzą sobie z nim słabo. Później sze przykłady działają trochę lepiej .

[ ]:
```
def test_ex(unmasker):
  return unmasker(f"{unmasker.tokenizer.mask_token} wrze w temperaturze 100␣
  ↪stopni, a zamarza w temperaturze 0 stopni Celsjusza.")
```

[ ]:
```
test_ex(unmasker_bert)
```

[ ]:
```
[{'score': 0.04173848032951355,
  'token': 21260,
  'token_str': 'Jego',
  'sequence': 'Jego wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.033460114151239395,
  'token': 14074,
  'token_str': 'Za',
  'sequence': 'Za wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.0281110480427742,
  'token': 11255,
  'token_str': 'Po',
  'sequence': 'Po wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.024846933782100677,
  'token': 10685,
  'token_str': 'Na',
  'sequence': 'Na wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
```

```
stopni Celsjusza.'},
 {'score': 0.023285038769245148,
  'token': 160,
  'token_str': 'W',
  'sequence': 'W wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'}]
```

[ ]:
```
test_ex(unmasker_roberta)
```

[ ]:
```
[{'score': 0.11466404795646667,
  'token': 229067,
  'token_str': 'Najlepiej',
  'sequence': 'Najlepiej wrze w temperaturze 100 stopni, a zamarza w
temperaturze 0 stopni Celsjusza.'},
 {'score': 0.10112352669239044,
  'token': 20,
  'token_str': '-',
  'sequence': '- wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.040479838848114014,
  'token': 239944,
  'token_str': 'Najczęściej',
  'sequence': 'Najczęściej wrze w temperaturze 100 stopni, a zamarza w
temperaturze 0 stopni Celsjusza.'},
 {'score': 0.03635391965508461,
  'token': 46,
  'token_str': '-',
  'sequence': '- wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.033538997173309326,
  'token': 167479,
  'token_str': 'Następnie',
  'sequence': 'Następnie wrze w temperaturze 100 stopni, a zamarza w
temperaturze 0 stopni Celsjusza.'}]
```

[ ]:
```
test_ex(unmasker_clips)
```

[ ]:
```
[{'score': 0.11825703084468842,
  'token': 115660,
  'token_str': 'amator',
  'sequence': 'amator wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.06405617296695709,
  'token': 100264,
  'token_str': '     ',
  'sequence': '      wrze w temperaturze 100 stopni, a zamarza w
temperaturze 0 stopni Celsjusza.'},
```

```
 {'score': 0.05478503182530403,
  'token': 85339,
  'token_str': '  ',
  'sequence': '   wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.03371739760041237,
  'token': 243172,
  'token_str': '    ',
  'sequence': '    wrze w temperaturze 100 stopni, a zamarza w temperaturze 0
stopni Celsjusza.'},
 {'score': 0.03235792741179466,
  'token': 122024,
  'token_str': 'qishloq',
  'sequence': 'qishloq wrze w temperaturze 100 stopni, a zamarza w temperaturze
0 stopni Celsjusza.'}]
```

[59]:
```
def test_wise(unmasker):
  print(*unmasker(f"{unmasker.tokenizer.mask_token} to największy kraj na␣
 ↪świecie"),sep='\n')
  print("-"*10)
  print(*unmasker(f"{unmasker.tokenizer.mask_token} to napój sporządzany z␣
 ↪palonych, a następnie zmielonych lub poddanych instantyzacji ziaren kawowca,␣
 ↪zwykle podawany na gorąco"),sep='\n')
  print("-"*10)
  print(*unmasker(f"{unmasker.tokenizer.mask_token} to gwiazda centralna Układu␣
 ↪Słonecznego, wokół której krąży Ziemia, inne planety tego układu, planety␣
 ↪karłowate oraz małe ciała Układu Słonecznego"),sep='\n')
```

[60]:
```
test_wise(unmasker_bert)
```

```
{'score': 0.12990902364253998, 'token': 10114, 'token_str': 'to', 'sequence':
'to to największy kraj na świecie'}
{'score': 0.06941397488117218, 'token': 10973, 'token_str': 'jest', 'sequence':
'jest to największy kraj na świecie'}
{'score': 0.032836467027664185, 'token': 118, 'token_str': '-', 'sequence': '-
to największy kraj na świecie'}
{'score': 0.031557779759168625, 'token': 19259, 'token_str': 'Jest', 'sequence':
'Jest to największy kraj na świecie'}
{'score': 0.015165765769779682, 'token': 11947, 'token_str': 'był', 'sequence':
'był to największy kraj na świecie'}
----------
{'score': 0.36692678928375244, 'token': 10973, 'token_str': 'jest', 'sequence':
'jest to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.048377253115177155, 'token': 19259, 'token_str': 'Jest', 'sequence':
'Jest to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
```

```
{'score': 0.04820689931511879, 'token': 117, 'token_str': ',', 'sequence': ', to
napój sporządzany z palonych, a następnie zmielonych lub poddanych instantyzacji
ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.042925771325826645, 'token': 118, 'token_str': '-', 'sequence': '-
to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.028311630710959435, 'token': 11769, 'token_str': 'ten', 'sequence':
'ten to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
----------
{'score': 0.48207923769950867, 'token': 19259, 'token_str': 'Jest', 'sequence':
'Jest to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.08427506685256958, 'token': 10973, 'token_str': 'jest', 'sequence':
'jest to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.055157314985990524, 'token': 118, 'token_str': '-', 'sequence': '-
to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne planety
tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.04379252344369888, 'token': 10114, 'token_str': 'to', 'sequence':
'to to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.03342774510383606, 'token': 33328, 'token_str': 'Alfa', 'sequence':
'Alfa to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
```

[61]:
```
test_wise(unmasker_roberta)
```

```
{'score': 0.08575551956892014, 'token': 54492, 'token_str': 'Polska',
'sequence': 'Polska to największy kraj na świecie'}
{'score': 0.08144792914390564, 'token': 49046, 'token_str': 'Izrael',
'sequence': 'Izrael to największy kraj na świecie'}
{'score': 0.06670422852039337, 'token': 5596, 'token_str': 'India', 'sequence':
'India to największy kraj na świecie'}
{'score': 0.047555841505527496, 'token': 72620, 'token_str': 'Nigeria',
'sequence': 'Nigeria to największy kraj na świecie'}
{'score': 0.04617858678102493, 'token': 8070, 'token_str': 'Tanzania',
'sequence': 'Tanzania to największy kraj na świecie'}
----------
{'score': 0.9105568528175354, 'token': 16819, 'token_str': 'Jest', 'sequence':
'Jest to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.02118770219385624, 'token': 834, 'token_str': 'jest', 'sequence':
'jest to napój sporządzany z palonych, a następnie zmielonych lub poddanych
instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.0036334272008389235, 'token': 105221, 'token_str': 'Kawa',
'sequence': 'Kawa to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
```

```
{'score': 0.0035889740101993084, 'token': 240751, 'token_str': 'Espresso',
'sequence': 'Espresso to napój sporządzany z palonych, a następnie zmielonych
lub poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.0031306094024330378, 'token': 130270, 'token_str': 'Beer',
'sequence': 'Beer to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
----------
{'score': 0.29079002141952515, 'token': 16819, 'token_str': 'Jest', 'sequence':
'Jest to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.21836650371551514, 'token': 118651, 'token_str': 'Saturn',
'sequence': 'Saturn to gwiazda centralna Układu Słonecznego, wokół której krąży
Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
{'score': 0.049439556896686554, 'token': 23342, 'token_str': 'Mars', 'sequence':
'Mars to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.04699844866991043, 'token': 193349, 'token_str': 'Orion',
'sequence': 'Orion to gwiazda centralna Układu Słonecznego, wokół której krąży
Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
{'score': 0.04277461767196655, 'token': 145255, 'token_str': 'Jupiter',
'sequence': 'Jupiter to gwiazda centralna Układu Słonecznego, wokół której krąży
Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
```

[62]:
```
test_wise(unmasker_clips)
```

```
{'score': 0.1299787014722824, 'token': 122024, 'token_str': 'qishloq',
'sequence': 'qishloq to największy kraj na świecie'}
{'score': 0.07731964439153671, 'token': 115660, 'token_str': 'amator',
'sequence': 'amator to największy kraj na świecie'}
{'score': 0.07141003012657166, 'token': 189368, 'token_str': 'fqinj',
'sequence': 'fqinj to największy kraj na świecie'}
{'score': 0.05194125324487686, 'token': 139322, 'token_str': 'yaşıyor',
'sequence': 'yaşıyor to największy kraj na świecie'}
{'score': 0.0328538604080677, 'token': 235862, 'token_str': 'vizinhos',
'sequence': 'vizinhos to największy kraj na świecie'}
----------
{'score': 0.11393607407808304, 'token': 189368, 'token_str': 'fqinj',
'sequence': 'fqinj to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.0728994756937027, 'token': 241999, 'token_str': '    ',
'sequence': '    to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.0550517663359642, 'token': 235862, 'token_str': 'vizinhos',
'sequence': 'vizinhos to napój sporządzany z palonych, a następnie zmielonych
lub poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
```

{'score': 0.040677573531866074, 'token': 115660, 'token_str': 'amator',
'sequence': 'amator to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
{'score': 0.04013815522193909, 'token': 243172, 'token_str': '  ',
'sequence': '  to napój sporządzany z palonych, a następnie zmielonych lub
poddanych instantyzacji ziaren kawowca, zwykle podawany na gorąco'}
----------
{'score': 0.10600201785564423, 'token': 100264, 'token_str': '  ',
'sequence': '  to gwiazda centralna Układu Słonecznego, wokół której
krąży Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
{'score': 0.1001770943403244, 'token': 115660, 'token_str': 'amator',
'sequence': 'amator to gwiazda centralna Układu Słonecznego, wokół której krąży
Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
{'score': 0.05321836844086647, 'token': 65482, 'token_str': '  ', 'sequence':
'  to gwiazda centralna Układu Słonecznego, wokół której krąży Ziemia, inne
planety tego układu, planety karłowate oraz małe ciała Układu Słonecznego'}
{'score': 0.05090215057134628, 'token': 218756, 'token_str': 'szczególny',
'sequence': 'szczególny to gwiazda centralna Układu Słonecznego, wokół której
krąży Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}
{'score': 0.034241944551467896, 'token': 217728, 'token_str': 'bihotz',
'sequence': 'bihotz to gwiazda centralna Układu Słonecznego, wokół której krąży
Ziemia, inne planety tego układu, planety karłowate oraz małe ciała Układu
Słonecznego'}

Answer the following questions: 1. Which of the models produced the best results?

1. Was any of the models able to capture Polish grammar?
2. Was any of the models able to capture long-distant relationships between the words?
3. Was any of the models able to capture world knowledge?
4. What are the most striking errors made by the models?

1. Najlepszym modelem z tych 3 wydaje się być bert-base-multilingual-cased. Dobrze radzi sobie gdy jest wiecej informacji a [Mask] znajduje się pomiedzy nimi. Często jednak gdy jest mniej informacji a [Mask] jest na poczatku lub na koncu daje znaki typu -, :, ;, . badz powtarza wyraz. W takim wypadku dziala lepiej xlm-roberta-base ktory najlepiej wypadl przy sprawdzaniu real-world knolwedge. Clips/mfaq wypada najgorzej czesto nie rozpoznaje jezyka a jesli już to robi to proponowane tokeny nie maja wiekszego sensu

2. Tutaj widać przewagę bert-base-multilingual-cased w zadaniu 1 udało mu się zwrocić więcej poprawnych form niż pozostałym modelom. W 2 natomiast dobrze rozpoznał płeć. xlm-roberta-base w 1 rozpoznał o jedna forme mniej niż bert a w 2 rozpoznał płeć M i F. Najgorzej wypadł Clips/mfaq

3. Tak. bert-base-multilingual-cased oraz xlm-roberta-base moga wychwycić takie zależności(bert-base-multilingual-cased>xlm-roberta-base poprzez przykłady w 2)

4. Najlepiej działa xlm-roberta-base. Potrafi rozponzać niektóre definicje np “Kawa”

5. Czasami modele podają jako [Mask] znaki typu -, :, ;, . które nie pasują w danym miejscu - natomiast po drobnym formacie wyniki mogą być zupełnie inne. Często również powtarzają słowo przed lub po [Mask] co też nie ma raczej sensu. Ponadto część modeli które powinny działać z językiem polskim zupełnie sobie z nim nie radzą

[ ]: