

Projekt nr 10

“Narzędzie do rozpoznawania języka oparte na bibliotece fastText”

autorzy:
Wojciech Bafia
Damian Kluczyński
Jakub Cudak

Politechnika Krakowska im. Tadeusza Kościuszki

1. Abstrakt:

W ramach projektu przygotowaliśmy program w języku Python, który z pomocą zaproponowanej biblioteki FastText [1] rozpoznaje język podanego tekstu. Stworzyliśmy również własny model do rozpoznawania języka oraz przygotowaliśmy porównanie kilku modeli na podstawie ich celności przy pracy na danych testowych.

2. Wstęp:

2.1 Cel:

Stworzenie rozwiązania do rozpoznawania języka podanego fragmentu tekstu przy wykorzystaniu biblioteki FastText [1]

2.2 Zakres:

Zakresem pracy jest stworzenie programu do rozpoznawania tekstu przy pomocy biblioteki FastText. My jednak postanowiliśmy także wytrenować własny model do rozpoznawania języka a także porównać jego działanie z modelami dostarczonymi przez twórców biblioteki FastText

2.3 Metodyka:

Wytrenowanie modelu do rozpoznawania języka na podstawie pliku zawierającego fragmenty tekstów w różnych językach wraz z przypisanymi do nich językami. A także wykorzystanie gotowego modelu [2] oraz przetestowanie funkcji rozpoznawania języka z

jego pomocą. A następnie porównanie dokładności dostępnych modeli.

3. Część teoretyczna:

Biblioteka FastText wykorzystuje koncept Bag-of-Words, dzięki któremu do konkretnych języków przypisany jest zbiór charakterystycznych słów, na podstawie których można rozpoznać język tekstu, a także tzw. Word Embedding - czyli przyporządkowanie słowu wektoru liczb, dzięki czemu możemy rozpoznawać nie tylko dokładnie takie same, ale także podobne do siebie słowa.

4. Część praktyczna:

Do zainstalowania biblioteki FastText na laptopie z systemem Linux należy użyć komendy:

```
sudo pip install fasttext
```

Twórcy tej biblioteki na swojej oficjalnej stronie udostępnili dwa modele do rozpoznawania języka [2]

Program:

```
fasttext_project.py

import fasttext as ft
import sys

def train_model():
    model2 = ft.train_supervised(input="datasets/dataset.txt")
    model2.save_model("our_model.bin")

#train_model()

def declare_model(model_name):
    try:
        return ft.load_model(model_name)
    except FileNotFoundError:
        print("Model not found")
        return None
```

```

def detect_language(text, model):
    if model is None:
        return "model is None"

    try:
        prediction = model.predict(text)
        return prediction[0][0]
    except ValueError:
        return "ValueError"

def compare_models():
    for filename in ["dataset_test.txt",
                    "english_test_samples_en_label.txt",
                    "german_test_samples_de_label.txt",
                    "hungarian_test_samples_hu_label.txt",
                    "italian_test_samples_it_label.txt",
                    "polish_test_samples_pl_label.txt"]:
        print('file: ' + filename)
        for model_name in ['lid.176.bin', 'lid.176.ftz', 'our_model.bin']:
            with (open("datasets/" + filename, "r") as file):
                model = declare_model(model_name)
                x = 0.0
                n = 0
                for line in file:
                    words = line.split()
                    text = words[1:]
                    if words[0] == detect_language(text, model)[0]:
                        x += 1
                    n += 1
                print(model_name + ':\t' + str(x / n) )

def main():
    original_stdout = sys.stdout
    original_stderr = sys.stderr
    sys.stdout = None
    sys.stderr = None

    model = declare_model('lid.176.bin')

    sys.stderr = original_stderr
    sys.stdout = original_stdout

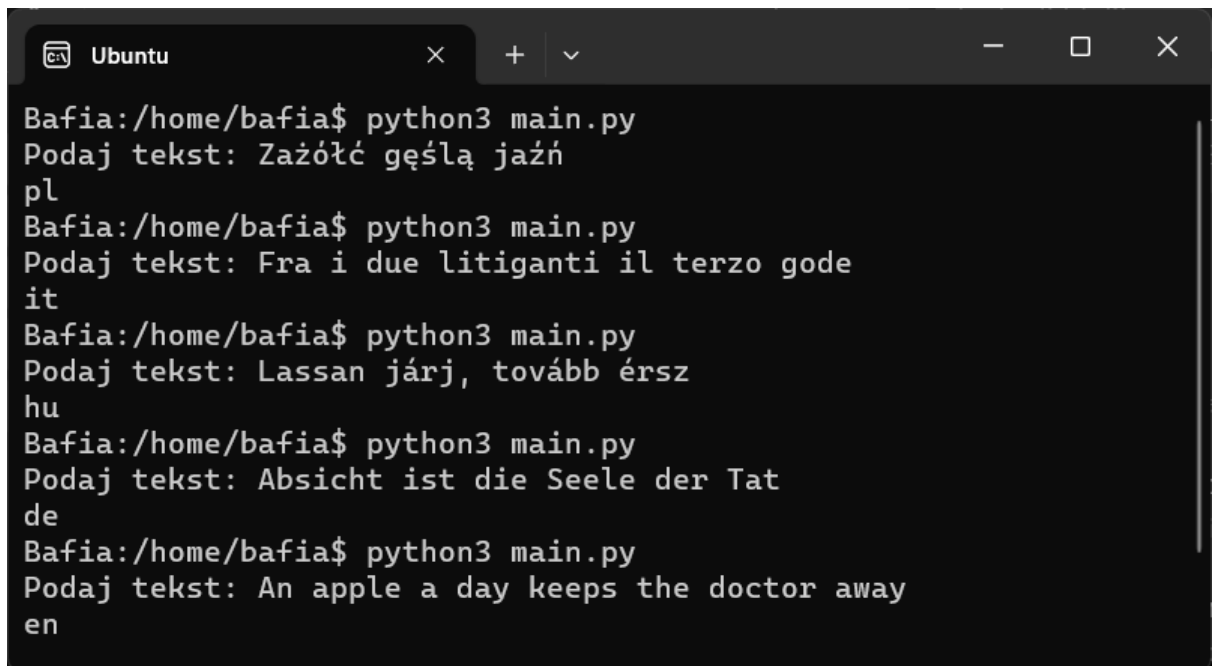
    text = input("Podaj tekst: ")
    print(detect_language(text, model)[-2:])

main()
#compare_models()

```

Przy wykorzystaniu gotowego modelu 'lid.176.bin' w funkcji main wywołujemy funkcję "detect_language", która zwraca język wpisanego przez użytkownika tekstu.

Wynik działania programu:



```
Ubuntu x + v
Bafia:/home/bafia$ python3 main.py
Podaj tekst: Zażółć gęślą jaźń
pl
Bafia:/home/bafia$ python3 main.py
Podaj tekst: Fra i due litiganti il terzo gode
it
Bafia:/home/bafia$ python3 main.py
Podaj tekst: Lassan járj, tovább érsz
hu
Bafia:/home/bafia$ python3 main.py
Podaj tekst: Absicht ist die Seele der Tat
de
Bafia:/home/bafia$ python3 main.py
Podaj tekst: An apple a day keeps the doctor away
en
```

Przygotowaliśmy także własny model na podstawie utworzonego przez nas pliku “dataset.txt” składającego się z ok. 20 tysięcy danych. Jego szkolenie znajduje się w funkcji “train_model”.



```
fasttext_project.py

def train_model():
    model2 = ft.train_supervised(input="datasets/dataset.txt")
    model2.save_model("our_model.bin")

#train_model()
```

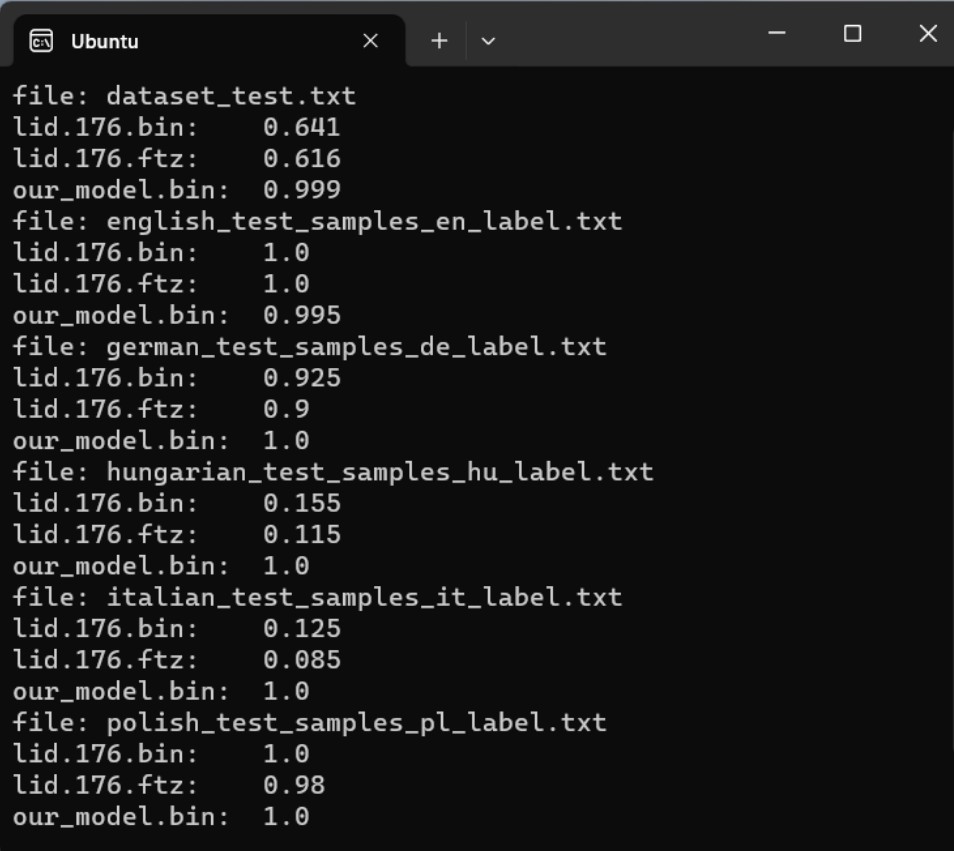
Funkcja “compare_models” na podstawie paru plików testowych weryfikuje i porównuje dokładność modeli przygotowanych przez twórców biblioteki FastText oraz nasz model:

```

def compare_models():
    for filename in ["dataset_test.txt",
                    "english_test_samples_en_label.txt",
                    "german_test_samples_de_label.txt",
                    "hungarian_test_samples_hu_label.txt",
                    "italian_test_samples_it_label.txt",
                    "polish_test_samples_pl_label.txt"]:
        print('file: ' + filename)
        for model_name in ['lid.176.bin', 'lid.176.ftz', 'our_model.bin']:
            with (open("datasets/" + filename, "r") as file):
                model = declare_model(model_name)
                x = 0.0
                n = 0
                for line in file:
                    words = line.split()
                    text = words[1:]
                    if words[0] == detect_language(text, model)[0]:
                        x += 1
                    n += 1
                print(model_name + ':\t' + str(x / n) )

```

A oto wynik jej działania:



```

file: dataset_test.txt
lid.176.bin: 0.641
lid.176.ftz: 0.616
our_model.bin: 0.999
file: english_test_samples_en_label.txt
lid.176.bin: 1.0
lid.176.ftz: 1.0
our_model.bin: 0.995
file: german_test_samples_de_label.txt
lid.176.bin: 0.925
lid.176.ftz: 0.9
our_model.bin: 1.0
file: hungarian_test_samples_hu_label.txt
lid.176.bin: 0.155
lid.176.ftz: 0.115
our_model.bin: 1.0
file: italian_test_samples_it_label.txt
lid.176.bin: 0.125
lid.176.ftz: 0.085
our_model.bin: 1.0
file: polish_test_samples_pl_label.txt
lid.176.bin: 1.0
lid.176.ftz: 0.98
our_model.bin: 1.0

```

Jak widać powyżej nasz model ma większą dokładność przy rozpoznawaniu języka na dostarczonych danych testowych. Najprawdopodobniej jest to związane z tym, że modele twórców FastText [2] trenowane są na o wiele większej liczbie języków, przez co ich wykrywanie spośród 5-ciu języków jest znacznie gorsze

5. Podsumowanie:

Stworzyliśmy narzędzie do rozpoznawania języka tekstu przy pomocy modelu z biblioteki FastText, a także wytrenowaliśmy własny model działający na mniejszym zakresie języków, który dzięki temu jest dokładniejszy.

6. Bibliografia:

[1] Dokumentacja biblioteki FastText

<https://fasttext.cc/docs/en/support.html>

[2] Modele dostarczone przez twórców FastText:

[https://fasttext.cc/docs/en/language-identification.ht](https://fasttext.cc/docs/en/language-identification.html)

[ml](#)