Narzędzie do rozpoznawania języka oparte na bibliotece fastText

Marcin Rajs, Kacper Skurski, Michał Słota

Wstęp

Rozpoznawanie języka to proces, w którym komputer określa, w jakim języku napisano dany tekst. Rozpoznawanie języka jest ważne, ponieważ:

- umożliwia tworzenie aplikacji, które są bardziej elastyczne i uniwersalne - są bardziej przyjazne dla użytkowników z różnych części świata,

- ułatwia analizę danych tekstowych - możliwe jest automatyczne przetwarzanie i analizowanie dużych zbiorów danych tekstowych

- umożliwia tworzenie aplikacji do tłumaczenia maszynowego przydatnych w sytuacjach takich jak podróże zagraniczne, nauka języka obcego czy też komunikacja z osobami mówiącymi innymi językami.

Biblioteka fastText jest jednym z popularnych narzędzi do rozpoznawania języka. Została opracowana przez Facebooka tak by współpracowała z wieloma językami, szczególnie uwagę poświęcono klasyfikacji tekstu dzięki temu, umożliwia szybkie i skuteczne rozpoznawanie języka na podstawie dużych zbiorów danych tekstowych. Według Facebooka fastText jest minimalnie szybszy niż popularna biblioteka “word2vec” i jest w stanie wytrenować model na 1 miliardzie słów w mniej niż 10 minut na procesorze o standardowej wydajności.

Biblioteka ta jest szczególnie przydatna w przypadku języków, w których brakuje dużych zbiorów danych treningowych, ponieważ umożliwia tworzenie modeli językowych na podstawie niewielkich zbiorów danych, jednak wymaga wtedy więcej pracy by dobrać odpowiednio parametry aby uzyskać lepszą dokładność.

Celem zadania jest wytrenowanie sieci neuronowej i wykorzystanie jej do rozpoznawania języka, w którym zostały podane przykładowe zdania.

Rozwinięcie

Aby można było skorzystać z biblioteki fastText najpierw należało znaleźć i rozpakować zbiór danych treningowych. Skorzystano z danych dostępnych w dokumentacji do biblioteki fastText:

http://downloads.tatoeba.org/exports/sentences.tar.bz2

Następnie z rozpakowanego archiwum odczytano plik sentences.csv, a linie w pomieszanej kolejności zapisano do pliku all.txt:

```
awk -F"\t" '{print"__label__"$2" "$3}' < sentences.csv | shuf > all.txt
```

Na końcu podzielono plik all.txt na dwa pliki, które posłużą do nadzorowanego trenowania sieci neuronowej. Liczba wszystkich zdań w pliku *all.txt* to 10 875 396.

```
head -n 10000 all.txt > valid.txt
tail -n +10001 all.txt > train.txt
```

Przedstawione polecenia pochodzą z dokumentacji biblioteki fastText.

Wykonanie treningu

Badanie wykonano w wielu warunkach, zarówno na zbiorach testowych, jak i własnych zdaniach.

Przypadek 1:

```
model = fasttext.train_supervised(input="train.txt", dim=16)
model.save_model("train1.bin")
```

Przypadek 2:

```
model = fasttext.train_supervised(input="train.txt", dim=16, minn=2,
maxn=4)
model.save_model("train2.bin")
```

Przypadek 3:

```
model = fasttext.train_supervised(input="train.txt", dim=16, minn=2,
maxn=4, loss="hs")
model.save_model("train3.bin")
```

Następnie każdy z modeli można było odczytać z pliku:

```
model = fasttext.load_model("train1.bin")
```

Badanie wyników

W celu sprawdzenia jakości treningów, wczytano wszystkie zapisane modele i wykorzystano je do weryfikacji przewidywań na zbiorze kontrolnym. Odpowiada za to następujący kod:

```
length, precission, recall = model.test("valid.txt")
print("Dokładność: {}".format(precission))
```

Wyniki testów:

	Przypadek		
	1	2	3
Dokładność	95,74%	97,87%	97,92%
words/sec/thread	148 976	124 837	472 365

Można zauważyć, że przypadek 1. charakteryzuje się najmniejszą dokładnością. Natomiast różnice pomiędzy przypadkiem 2. a 3. widać tylko w szybkości treningu. Wykorzystanie funkcji *hierarchical softmax* wpływa na jego znaczące przyspieszenie.

Przygotowano także własne zdania, które nie występowały w pobranym zbiorze i sprawdzono czy prawdopodobieństwo rozpoznania języka jest odpowiednio wysokie:

```
sentences = [
      "Would you like to talk to me in English?",
      "¿Querrías hablar conmigo en español?",
      "Czy możemy porozmawiać po polsku?"
]

language, probability = model.predict(sentences)
for index, sentence in enumerate(sentences):
      print("Język: {}, prawdopodobieństwo: {:.2%}".format(
language[index][0], probability[index][0] ))
```

Wszystkie języki rozpoznaje prawidłowo z bardzo wysokimi prawdopodobieństwami:

	Przypadek		
Zdanie	1	2	3
1	100.00%	100.00%	100,00%
2	100.00%	99,98%	100,00%
3	100.00%	99,98%	100,00%

Program jest także w stanie rozpoznawać zdania podawane bezpośrednio przez użytkownika:

```
sentence = input()
language, probability = model.predict(sentence)

print("Język: {}, prawdopodobieństwo: {:.2%}".format( language[0], probability[0] ))
```

```
Mein Beispielsatz lautet
Język: __label__deu, prawdopodobieństwo: 99.90%
```

Wykonano także testy na zdaniach w innych językach:

1. ਮੈਂ ਇੱਕ ਲੋਕਾਂ ਦਾ ਸਮੂਹ ਹਾਂ ਜੋ ਸਾਡੇ ਦੇ ਘਰ ਵਿੱਚ ਰਹਿੰਦੇ ਹਨ। (Punjabi)
2. أنا أعيش في مدينة كبيرة في قارة أفريقيا وأنا أشتغل كمهندس كهرباء. (Arabic)
3. אני גר בעיר גדולה באזור המזרח התיכון ואני עובד כמהנדס חשמל. (Hebrew)
4. मैं एक बड़े शहर में रहता हूँ और मैं एक विद्युत अभियंता हूँ। (Hindi)
5. 我住在一个大城市，并且我是一名电气工程师。 (Chinese)
6. ฉันอาศัยอยู่ในเมืองใหญ่และฉันเป็นวิศวกรไฟฟ้า。(Thai)
7. Μένω σε μια μεγάλη πόλη και είμαι ηλεκτρολόγος μηχανικός. (Greek)
8. მე მოვიდა დიდ ქალაქში და მე ვარ ელექტრომენტროლის მენეჯერი. (Georgian)
9. Mi loĝas en granda urbo kaj mi estas elektroteknikisto. (Esperanto)
10. Я живу (Russian)

```
for index, sentence in enumerate(sentences):
    print("Język: {}, prawdopodobieństwo: {:.2%}".format( language[index][0], probability[index][0] ))
```

```
Język: __label__pan, prawdopodobieństwo: 29.45%
Język: __label__ara, prawdopodobieństwo: 99.41%
Język: __label__heb, prawdopodobieństwo: 100.00%
Język: __label__hin, prawdopodobieństwo: 99.94%
Język: __label__cmn, prawdopodobieństwo: 99.35%
Język: __label__tha, prawdopodobieństwo: 98.32%
Język: __label__ell, prawdopodobieństwo: 99.83%
Język: __label__kat, prawdopodobieństwo: 99.80%
Język: __label__epo, prawdopodobieństwo: 100.00%
Język: __label__rus, prawdopodobieństwo: 98.69%
```

Podsumowanie

- Jak widać na powyższych przykładach model został wytrenowany wystarczająco dobrze, żeby rozpoznawać zdania.
- Również zdania w mniej popularnych językach udało mu się poprawnie rozpoznać.
- W przypadku pierwszego języka, pendżabskiego, prawdopodobieństwo było dość małe (~30%), ale według kodu ISO-639-2 język o skrócie “pan” to właśnie język pendżabski, zatem został rozpoznany poprawnie. Małe prawdopodobieństwo może wynikać z małej liczby dostępnych zdań w pliku, na którego podstawie trenowany był model.

- Biblioteka fastText jest bardzo przyjazna dla użytkownika, gdyż zawiera dużo przykładów oraz posiada dobrą dokumentację. Ponadto możliwe jest korzystanie z niej przy pomocy pythona co może ułatwić wielu osobom pracę z tą biblioteką.

Bibliografia

- https://fasttext.cc/blog/2017/10/02/blog-post.html
- https://fasttext.cc/docs/en/supervised-tutorial.html
- http://downloads.tatoeba.org/exports/sentences.tar.bz2
- https://research.facebook.com/blog/2016/08/fasttext/
- https://blog.insightdatascience.com/using-transfer-learning-for-nlp-with-small-data-71e10baf99a6
- https://github.com/facebookresearch/fastText/issues/5