Лабораторна робота №8

Навчання моделей spaCy

Мета роботи: Ознайомитись з додаванням власних прикладів до моделей spaCy та компонентом для класифікації текстів.

Короткі теоретичні відомості

Моделі spaCy дуже успішні для загальних цілей обробки природньої, таких як розуміння синтаксису речення, розбиття абзацу на речення та вилучення деяких сутностей. Однак іноді потрібно працювати над дуже специфічними доменами, які моделі spaCy не бачили під час навчання.

Якщо потрібне індивідуальне навчання моделі, зазвичай виконуються такі дії:

- Збір своїх даних.
- Анотування своїх даних.
- Вирішення чи оновити існуючу модель або навчити модель з нуля.

Першим кроком навчання моделі завжди є підготовка навчальних даних. Зазвичай збираються дані з джерела, наприклад, з логів клієнтів, а потім вони перетворюються у набір даних, наприклад, у форматі CSV або JSON. Навчальний код моделі spaCy працює з файлами JSON.

```
{           "sentence": «Я вилітаю з аеропорту Бориспіль. "
            "entities": {
                  "label": "LOC"
                  "value": “Бориспіль"          }
```

Не можна передати необроблений текст і анотації безпосередньо до spaCy. Натомість потрібно створити об’єкт Example для кожного навчального прикладу.

```
import spacy
from spacy.training import Example
nlp = spacy.load("en_core_web_md")
doc = nlp("I will visit you in London.")
annotations =  {"entities": [(20, 26, "GPE")]}
example_sent = Example.from_dict(doc, annotations)
```

Можна оновити вже існуючий компонент конвеєру spaCy. Наприклад, потрібно змінити розпізнавання сутностей.

Для цього:

1. Вимкнути всі інші компоненти статистичного конвеєра, включаючи тег частини мови і аналізатор залежностей.

2. Додати свої приклади домену до процедури навчання.

3. Оцінити нову модель розпізнавання сутностей.

Модель spaCy для розпізнавання сутностей — це модель нейронної мережі. Щоб навчити нейронну мережу, потрібно налаштувати деякі параметри, а також

надати навчальні приклади. Кожен прогноз нейронної мережі є сумою її вагових значень; отже, процедура навчання коригує ваги нейронної мережі за допомогою додаткових прикладів.

```
import random
import spacy
from spacy.training import Example
nlp = spacy.load("en_core_web_md")
trainset = [
        ("navigate home", {"entities": [(9,13, "GPE")]}),
        ("navigate to office", {"entities": [(12,18, "GPE")]}),
        ("navigate", {"entities": []}),
        ("navigate to Oxford Street", {"entities": [(12, 25, "GPE")]})
        ]
epochs = 20
other_pipes = [pipe for pipe in nlp.pipe_names if pipe != 'ner’]
with nlp.disable_pipes(*other_pipes):
  optimizer = nlp.create_optimizer()
  for i in range(epochs):
        random.shuffle(trainset)
  example = Example.from_dict(doc, annotation)
  nlp.update([example], sgd=optimizer)
  ner = nlp.get_pipe("ner")
  ner.to_disk("new_ner")
```

Також в spaCy можна використовувати моделі для класифікації. TextCategorizer є необов'язковим компонентом конвеєра, який можна навчати. Щоб його навчити, потрібно надати приклади та їхні позначки класів. Спочатку додаємо TextCategorizer до конвеєра обробки природньої мови, а потім виконуємо процедуру навчання. На відміну від моделей, які використовувались в scikit-learn, в spaCy застосовуються нейронні мережі.

```
from spacy.pipeline.textcat import
DEFAULT_MULTI_TEXTCAT_MODEL
config = {
  "threshold": 0.5,
    "model": DEFAULT_MULTI_TEXTCAT_MODEL
}
textcat = nlp.add_pipe("textcat_multilabel", config=config)
train_data = [ ("I loved this product, very easy to use.", {"cats": {"sentiment": 1}}), ("Truly horrible product. Very few amount of product for a high price. Don't recommend.", {"cats": {"sentiment": 0}})  ]
textcat.add_label("sentiment")
train_examples = [Example.from_dict(nlp.make_doc(text),label) for text,label in train_data]
textcat.initialize(lambda: train_examples, nlp=nlp)
```

```
epochs=20
with nlp.select_pipes(enable="textcat"):
  optimizer = nlp.resume_training()
  for i in range(epochs):
                    random.shuffle(train_data)
                    for text, label in train_data:
                     doc = nlp.make_doc(text)
        example = Example.from_dict(doc, label)
nlp.update([example], sgd=optimizer)
```

Завдання до лабораторної роботи

1. Створити кілька своїх прикладів у форматі json за тематикою варіанту (англійською або українською мовою) для розпізнавання нового типу сутностей (обрати самостійно). Створити програму, що додає ці приклади до існуючої моделі spaCy, навчає модель. Продемонструвати роботу.
2. Застосувати компонент TextCategorizer для визначення намірів. Дані для навчання за тематикою варіанту обрати самостійно або скористатись вказаним файлом (utterance містить висловлювання, intent - намір). Дані файли містять приклади діалогів користувачів з системою-помічником за певною тематикою, наприклад, замовлення квитків і т.д. Навчити компонент та продемонструвати роботу.

Варіант 1.
Тематика: музика. Файл music.json.
Варіант 2.
Тематика: події (наприклад, спортивні). Файл events.json.
Варіант 3.
Тематика: фільми. Файл movies.json.
Варіант 4.
Тематика: авіарейси. Файл flights.json.
Варіант 5.
Тематика: медіа. Файл media.json.
Варіант 6.
Тематика: таксі та оренда машин. Файл taxiandcars.json.
Варіант 7.
Тематика: автобусні рейси. Файл buses.json.
Варіант 8.
Тематика: послуги (наприклад, медичні). Файл services.json.
Варіант 9.
Тематика: оренда житла. Файл homes.json.
Варіант 10.
Тематика: календар подій. Файл calendar.json.

Варіант 11.
Тематика: банки. Файл banks.json.
Варіант 12.
Тематика: готелі. Файл hotels.json.
Варіант 13.
Тематика: події. Файл events.json.
Варіант 14.
Тематика: авіарейси. Файл flights.json.
Варіант 15.
Тематика: фільми. Файл movies.json.

Оформити звіт. Звіт повинен містити:

— титульний лист;
— код програми;
— результати виконання коду;

Продемонструвати роботу програми та відповісти на питання стосовно теоретичних відомостей та роботи програми.