

ВИКОРИСТАННЯ МОЖЛИВОСТЕЙ ОБРОБЛЕННЯ ПРИРОДНОЇ МОВИ У ПРИКЛАДНИХ ПРОГРАМАХ

Оброблення текстів, створених природною мовою, є на даний момент окремою галуззю комп'ютерних наук, що займається автоматичним аналізом та обробленням мовленнєвих даних, отриманих від людини. Дана галузь включає цілий ряд окремих задач різного рівня, ефективно розв'язання яких дозволяє підвищити ефективність взаємодії з користувачем, який фактично може надавати текст як вхідні дані, а може сприймати оброблений текст на виході. Фактично ці задачі можуть включати комплексні задачі витягання змісту, закладеного в тексті, виконання генерації нового тексту потрібної структури, виконання аналізу тональності, машинного перекладу, а також цілий ряд інших задач та більш простих завдань. При цьому слід розуміти, що від виконання навіть окремих простих завдань ефективним чином може залежати зручність роботи користувача з програмою, при тому, що основна функціональність програмного забезпечення може не бути пов'язаною безпосередньо з цими завданнями оброблення тексту.

До основних етапів оброблення текстів, створених природною мовою, належать токенизація, лексичний, синтаксичний, семантичний та дискурсивний аналіз.

Сучасний стан галузі комп'ютерних наук характеризується наявністю готових рішень для багатьох завдань. Це стосується і даної області, де потужним засобом є бібліотека NLTK [1] для мови програмування Python. Враховуючи доступність в ній широкого набору інструментів та орієнтованість Python на швидке створення готових рішень, можна констатувати, що додавання можливостей оброблення природної мови у прикладні програми може бути відносно простим і ефективним процесом. У такому випадку загалом витрачені ресурси можуть бути оцінені як достатньо ефективні. Безпосередньо пакет NLTK містить інструменти фактично для кожного етапу оброблення текстів, а також для супутніх завдань, як наприклад, класифікація текстів.

Реалізуючи дослідження в межах даної роботи, було виконано реалізацію двох програмних модулів, які можуть бути застосовані для оброблення текстів у складі більш широких програмних систем, які в свою чергу можуть стосуватися медичної діагностики, аналізу соціальних мереж, відеоігор тощо.

Перший модуль дозволив виконати оцінку коментарів, отриманих з відеохостингу youtube, на їх позитивність, негативність або нейтральність. Звичайно, що в конкретній ситуації коментарі можуть бути отримані з інтернет-магазину або багатьох інших сервісів. Розроблення даного модуля передбачало спочатку завантаження коментарів з відеохостингу, після цього формування вибірки. Для безпосередньо класифікації була застосована модель VADER (Valence Aware Dictionary and Sentiment Reasoner) [2], яка базується на виконанні лексичного аналізу, тобто вона порівнює слова та вирази зі списками слів з позитивним та негативним значенням, створеними на основі експертної оцінки, з використанням контекстної інформації для оцінки емоційного забарвлення тексту. Це дозволяє виявити слова, які живаються в негативному контексті, маючи позитивне значення, і віднести їх у підсумку до випадків сарказму.

Другий модуль дозволяє також виконувати класифікацію тексту, але застосовується вона для визначення жанру кінострічки на основі синопсису. Тобто фактично це також підзадача в загальній системі, яка відповідає за накопичення даних про кінострічки, що наприклад, демонструються на відповідному сервісі. Задача фактично є допоміжною і за великої кількості кінострічок та обмежених ресурсів для її реалізації потребує застосування простої, але практично придатної моделі. У результаті був застосований мультиноміальний наївний басів класифікатор. Кожен синопсис був отриманий з сервісу IMDb, токенізований та векторизований, а потім використовувався для розрахунку ймовірності того, що документ належить до кожного з класів, що відповідали доступним жанрам кінострічок.

Для реалізації модулів використовувалися такі бібліотеки Python як NLTK, Pandas [3] та Scikit-learn [4]. Обидва модулі демонструють застосування оброблення природної мови для аналізу та обробки текстових даних, що є підзадачами в межах відповідальності ширшої програмної системи і відповідно потребують простоти застосованих рішень і ефективності впровадження на практиці.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Natural Language Processing With Python's NLTK Package [Електронний ресурс]. – Режим доступу : <https://realpython.com/nltk-nlp-python/>.
2. Vader Lexicon [Електронний ресурс]. – Режим доступу : <https://www.kaggle.com/datasets/nltkdata/vader-lexicon>.
3. Pandas Guide [Електронний ресурс]. – Режим доступу : <https://pandasguide.readthedocs.io/en/latest/>.
4. API Reference – scikit-learn 0.21.3 documentation [Електронний ресурс]. – Режим доступу : <https://scikit-learn.org/0.21/modules/classes.html>.