|  |  |  |
| --- | --- | --- |
|  | **Софийски университет „Св. Кл. Охридски”**  Факултет по математика и информатика |  |

**Курсов Проект**

на тема: „Система за генериране на емотикони”

Студенти:

Божидар Павлов Димитров Ф.Н.: 7MI3400437

Маргарита Стефанова Моллова Ф.Н.: 0MI3400436

Курс: „ИИОЗ“, Учебна година: 2023/24

Преподавател: проф. Иван Койчев

=================================

Декларация за липса плагиатство:

* Плагиатство е да използваш, идеи, мнение или работа на друг, като претендираш, че са твои. Това е форма на преписване.
* Тази курсова работа е моя, като всички изречения, илюстрации и програми от други хора са изрично цитирани.
* Тази курсова работа или нейна версия не са представени в друг университет или друга учебна институция.
* Разбирам, че ако се установи плагиатство в работата ми ще получа оценка “Слаб”.

10.2.24 г. Подпис на студентите:

/Б. Димитров/

/М. Моллова/

**СЪДЪРЖАНИЕ**

[1 Увод 3](#_Toc158426851)

[2 Преглед на областa 3](#_Toc158426852)

[3 Проектиране 4](#_Toc158426853)

[3.1 Обща архитектура 4](#_Toc158426854)

[3.2 Модел на данните 4](#_Toc158426855)

[4 Реализация, тестване/експерименти 5](#_Toc158426856)

[4.1 Използвани технологии, платформи и библиотеки 5](#_Toc158426857)

[4.2 Реализация/Провеждане на експерименти 5](#_Toc158426858)

[5 Заключение 8](#_Toc158426859)

[6 Използвана литература 9](#_Toc158426860)

# Увод

В днешно време голяма част от комуникацията между хората се осъществява онлайн чрез социалните мрежи. Благодарение на тях са се изменили и видовете общуване – то вече не е само директно между група от хора, но може да бъде и индиректно – чрез публикации, чието съдържание не е насочено към конкретен човек или група от хора, а към всички, до които то би достигнало (в това число и напълно непознати). В много от случаите това се случва чрез текст, но по този начин за голяма част от хората може да бъде трудно да се предадат правилно емоциите, както и тяхната сила и експресивност. За да се улесни това, в текста се добавят емотикони – графични символи, които могат да представляват знаци за лице или символи за предмети и животни, които се тълкуват като израз на емоция или настроение. С развитието на технологиите се появяват много видове емотикони, които затрудняват избора на подходящ такъв за конкретния тект, защото смесват няколко емоции в себе си или често се използват в небуквален смисъл (например емотикона на клоун).

Целта на нашия проект е да улесни избора на емотикон към даден текст, предлагайки подходящ такъв за него. Целевата социална мрежа, която сме избрали е *X* (доскоро *Twitter*), тъй като тя е достъпна по цял свят, популярна е и в нея хората лесно могат да изразят своето мнение по дадена тема, както и да реагират на чуждото.

Задачата, която сме определили за курсовия проект, е по даден от потребител текст (бъдещ *tweet*) да бъде предложен подходящ емотикон, който обобщава цялостното настроение и емоцияга, която трябва да се предаде чрез текста.

# Преглед на областa

За решаване на проблема има няколко етапа [1], през които трбява да се премине:

* Намиране на подходящи набори от данни;
* Обработка на данните – разделяне на тренировъчни и тестови данни, трансформиране на отделните документи (*tweets*) в списъци от токени (*tonenization*) и последваща обработка на токените – редуциране до ниво лема (лематизация);
* Създаване на вектори, съответстващи на всеки документ (*tweet*) – реализирано чрез *Bag of Words (BoW)* и *Term Frequency-Inverse Document Frequency (TF-IDF)*;
* Обучаване на модел за класификация – реализирано чрез *Naïve Bayes, K-Nearest Neighbours (kNN), Support Vector Machines (SVM)* и *Random Forest*;
* Оценяване на получения модел чрез прилагането му върху тестовите данни.

Подобни подходи са избрани и в досега същестуващите решения на тази задача. Тъй като емотиконите са сравнително ново явление, няма много разработки на тази тема. През 2018 година такава задача е дадена на международния семинар по обработка на естествен език *SemEval*. Тогава са предложени редица решения [2], най-добрите от които използват *Bi-LSTM* с предварително тренирани *word2vec* вектори и *SVM* с *Bag of Words*. Тези и други различни подходи достигат до точност 40-47%. За обучение на моделите са използвани 500 000 данни, съдържащи 20 различни емотикона.

# Проектиране

## Обща архитектура

Системата за генериране на емотикони се състои от няколко основни модула:

* Модул за обработка на данните – включва зареждане на данните; разделяне на тренировъчни и тестови; сформиране на списъци от токени; лематизацията на получените токени;
* Модул на класификаторите – включва обучаване на избран модел с избран метод за векторизация върху подадени тренировъчни данни; намиране точността на обучения модел чрез прилагане върху тестовите данни и сравнение на получените резултати с очакваните; методи за използване на модела чрез подаване на текст, за който той връща емотикон.
* Модул на интерфейс с команден ред (*command line interface - CLI*) – включва разработен *CLI* за потребителско взаимодействие с предварително обучени модели, т.е. потребителят може да избере кой метод за векторизиция и кой предварително обучен модел, който го използва, да се приложи за предлагането на подходящ емотикон при подаден от потребителя текст.

## Модел на данните

Избраните данни могат да се намерят свободно в онлайн платформата за наука от данни *Kaggle* [3]. Те се състоят от 20 файла, като всеки файл съответства на емотикон, който задължително се съжържа във всеки един от документите него (но може да има и други емотикони).

За обучаване и тестване на моделите сме използвали данни с избрани съответно 5 или 10 емотикони от наличните. Тъй като всеки *tweet* може да съдържа няколко емотикона, които могат да бъдат различни, сме използвали два подхода за предварителна филтрация на данните:

* Избрани са само тези tweets, които съдържат единствено емотикона, който съответства на файла, в който се намират, но се премахват всички негови повторения;
* Всички *tweets* от съответния файл се включват, като за всеки един от тях се оставя единствено емотикона, съответстващ на файла, в който се намират.

И при двата подхода 5 000 от данните са отделени за тестови. Всички данни се преобразуват в списъци от токени, на които се прилага лематизация, преди да бъдат подадени за обучаване/трениране на модел.

# Реализация, тестване/експерименти

## Използвани технологии, платформи и библиотеки

Избраният програмен език за реализация на проекта е *Python*. Той е подходящ за целта, защото е лесен за използване и има богата екосистема от библиотеки, свързани с обработката на данни, обработката на текст и машинно самообучение. За различните части от разработката на проекта сме избрали различни библиотеки:

* *gensim [4]* за превръщане на данните в списък от токени и създаването на речник от тях;
* *nltk [5]* за премахване на стоп-думите и лематизация;
* *sklearn [6]* за различните методи за създаване на вектори от документите, различните модели, използвани за обучаване и оценяването на модела с тестовите данни.

## Реализация/Провеждане на експерименти

Реалицията на проекта е разделена на описаните в точка 3.1 модули, връзките, между които могат да се видят на приложената Фигура 1. Потребителите могат да си взаимодействат както с интерфейса с командния ред за използване на някой вече съществуващ модел, така и с модула на класификаторите, за да изберат да обучат и запазят имплементиран модел със собствени данни. Всички данни минават през процес на обработка, след който данните могат да се запишат за по-бърз и лесен достъп до тях в бъдещето.

A yellow rectangular boxes with blue text

Description automatically generated

Фигура 1 – Диаграма на модулите

Реализираните типове векторизация на данните са:

* *Bag of Words* [1], [7]
* *TF-IDF* [1], [8]

Реализираните видове класификатори за обучаване на модел са:

* *Naïve Bayes* [1], [6]
* *kNN* [1], [6]
* *Suport Vector Machine (SVM)* [1], [6]
* *Random Forest* [6], [9]

Всеки от класификаторите е обучаван заедно с всеки от типовете векторизация.

Тестването на разбработената система се осъществява чрез обучаване на различни модели и намиране на точността им, когато са пуснати с тестовите данни, т.е. моделът предлага за всеки *tweet* подходящ емотикон, той се сравнява с истинския емотикон, който е бил използван, и се намира колко пъти двата емотикона съвпадат.

Използвани са модели, обучени както с използването на *Bag of Words* като метод за векторизация на данните, така и с *TF-IDF*. Моделите са обучавани и тествани както с данни, съдържащи общо 5 различни емотикона в тях, така и с такива с 10. Тестовите данни винаги са 5 000 на брой. Получените резултати могат да бъдат видени в приложените Фигура 2 и Фигура 3.

Практичестки бе установено, че за трениране на някои модели или за тренирането им върху голям набор от данни, е необходима по-добра техника от наличната при провеждането на експериментите. Това е причината резултати от тях да липсват.

Моделите, които използват *kNN* за класификатор са тествани с различен параметър за брой съседи (от 1 до 5), като в таблицата с резултати са записани само най-добрите постижения и съответно с колко на брой съседи са достигнати.

Наблюдава се, че при използваните тренировъчни и тестови данни моделите се справят значително по-добре, когато използват само 5 емотикона, а не 10. Това е така, защото данните са много разнообразни, не са свързани с някава обща тема и е трудно да се намерят общи връзки между различните *tweets,* за които е било използван един и същ емотикон. Забелязва се също, че разликата в резулататите при различна филтрацията на данните не е особено голяма, отново поради голямото различие в контекста на отделните *tweets*.

A screenshot of a computer

Description automatically generated

Фигура 2 – Резултати от тестването на различните модели, обучени върху данни с 5 различни емотикони

A screenshot of a social media account

Description automatically generated

Фигура 3 - Резултати от тестването на различните модели, обучени върху данни с 10 различни емотикони

Можем също да изведем като извод, че *Naïve Bayes* се представя значително по-добре от *kNN*. Това може да се дължи на редица фактори, първият от които е, че той третира думите в подадените му текстове като независими една от друга. В случая с конкретните данни думите в повечето случаи наистина са независими една от друга, което прави използването на този класификатор подходящо. Други причини, които биха могли да водят до по-добро представяне на *Naïve Bayes* сравнено с *kNN*, са свързани с недостатъците на алгоритъма, който *kNN* използва – при малък брой данни има голяма възможност да се получи *overfitting* или *underfitting*, освен това този подход е силно чувствителен към „шумни характеристики“ (такива, които не носят полезна информация и могат да бъдат объркващи за модела).

# Заключение

Получените резултати са добри, но показват, че за получаване на по-добра точност са необходими повече данни, които да обхващат различни теми и настроения. Създадените класификатори се справят поне частично с голямото разнообразие на данните, но по-сложни и задълбочени модели най-вероятно биха се представили по-добре.

За бъдещо развитие могат да се имплементират решения, използващи рекурентни невронни мрежи и да се сравнят получените резултати. Също така могат да се обучат вече създадените модели върху по-голям и пълен набор от данни и да се анализират разликите при тестване с тестовите данни.

# Използвана литература

# Койчев, Иван. *Извличане на информация* *към курса “Извличане на информация” воден в СУ „Св. Климент Охридски“*

# Barbieri, Francesco, et al. “SemEval 2018 Task 2: Multilingual Emoji Prediction”, *SemEval*, 2018, pp. 24-30, [www.aclanthology.org/S18-1003.pdf](http://www.aclanthology.org/S18-1003.pdf)

# “Tweets With Emoji”, *Kaggle*, [www.kaggle.com/datasets/ericwang1011/tweets-with-emoji](http://www.kaggle.com/datasets/ericwang1011/tweets-with-emoji)

# “Gensim – Topic modeling for humans”, *Radim Rehurek*, [radimrehurek.com/gensim/](https://radimrehurek.com/gensim/)

# “Natural Language Toolkit”, *NLTK*, [www.nltk.org/](http://www.nltk.org/)

# “Machine Learning in Python – Supervised Learning”, *scikit-learn*, [www.scikit-learn.org/stable/supervised\_learning.html#supervised-learning](http://www.scikit-learn.org/stable/supervised_learning.html#supervised-learning)

# “Introduction to Natural Language Processing – Apply a Simple Bag-of-Words Approach”, *OpenClassrooms*, [www.openclassrooms.com/en/courses/6532301-introduction-to-natural-language-processing/8081284-apply-a-simple-bag-of-words-approach](http://www.openclassrooms.com/en/courses/6532301-introduction-to-natural-language-processing/8081284-apply-a-simple-bag-of-words-approach)

# “Multi-Class Text Classification with Scikit-Learn using TF-IDF model”, *Medium*, [www.medium.com/@rohit\_batra/multi-class-text-classification-with-scikit-learn-using-tf-idf-model-161d395ce374](http://www.medium.com/@rohit_batra/multi-class-text-classification-with-scikit-learn-using-tf-idf-model-161d395ce374)

# “Understanding Random Forest Algorithms With Examples”, *Analytics Vidhya*, 2024, [www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/](http://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/)