МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

П.П. Маслянко

Системи Data Science

Практикум з системної інженерії систем Data Science

Інженерія системи Data Science визначення емоційного забарвлення публікацій в соціальних мережах

Спеціальність 113 Прикладна математика

Спеціалізація - Наука про дані та математичне моделювання

Ухвалено кафедрою прикладної математики
(протокол № _13_ від _16.06.22_)
Погоджено Методичною комісією факультету прикладної математики[1]
(протокол № _9_ від _24.06.22_)

Київ – 2022

[1]Методичною радою університету– для загальноуніверситетських дисциплін.

АНОТАЦІЯ

Метою проекту є розробка математичного і програмного забезпечення системи Data Science визначення емоційного забарвлення публікацій в соціальній мережі Твіттер, що характеризують війну в Україні.

Проведено аналіз існуючих теоретичних інструментів та засобів аналізу тональності тексту. Розглянуто такі методи векторного представлення слів: Bag-of-Words, TF-IDF, Word2Vecта GloVe. Також розглянуто такі методи класифікації текстів: Наївний Баєсівський класифікатор, метод опорних векторів, нейронна мережа LSTM. Проведено порівняльний аналіз характеристик методів векторизації та класифікації тексту та обрано модель GloVe для векторного представлення тексту та нейронну мережу LSTM для побудови моделі аналізу тональності твітів.

На основі обраних методів теорії систем, системної інженерії, теоретичних інструментів аналізу тональності тексту було розроблено модель системи Data Science для визначення емоційного забарвлення публікацій в соціальній мережі Твіттер, що характеризують війну в Україні та математичне і програмне забезпечення.

Проведено верифікацію і валідацію системи.

Пошук існуючих рішень, експериментальні розрахунки та презентація проекту виконані за участі Вікторії Хоменко.

Ключові слова: Системна інженерія систем Data Science, аналіз настроїв, машинне навчання, Twitter, війна в Україні.

Keywords: System engineering of Data Science systems, sentiment analysis, machine learning, Twitter, war in Ukraine

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОСИЛАНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

BOW – Bag of Words

CBOW – Continuous Bag of Words

SVM – Support Vector Machine

RNN – Recurrent Neural Network

LSTM – Long Short Term Memory

NLP – Natural Language Processing

UML – Unified Modeling Language

TF-IDF – частота слова – обернена частота документа (англ. Term Frequency – Inverse Document Frequency)

МН – Машинне Навчання

НЛП – Нейрон Лінгвістичне Програмування

Ключові слова: Системна інженерія систем Data Science, сентимент-аналіз, машинне навчання, Twitter, війна в Україні

АКТУАЛЬНІСТЬ ВИРІШЕННЯ ЗАВДАНЬ ПРОЕКТУ

Емоційний стан суспільства в умовах війни в Україні характеризують численні публікації у засобах масової інформації. Одним із джерел визначення емоційного забарвлення публікацій, що характеризують війну в Україні є соціальні мережі. Однією із них є Твіттер що є однією з найпопулярніших платформ соціальних мереж. Сотні мільйонів активних користувачів у щоденних твітах та їх метаданих продукують величезну кількість інформації. Завдяки аналізу настроїв у Твіттер компанії можуть дізнатися, як-от думки клієнтів про їхні бренди та продукти, щоб приймати кращі бізнес-рішення. Крім того, аналіз настроїв у Твіттер є популярним способом вивчення поглядів громадськості на політичні кампанії чи інші актуальні теми.

Російсько-українська війна на сьогоднішній день є одним із широко обговорюваних питань у Твіттер. У цій роботі ми з цих твітів, використовуючи методи сентимент аналізу, зможемо отримати уявлення людей про триваючу війну.

Розробка системи Data Science визначення емоційного забарвлення публікацій, що характеризують війну в Україні, є надзвичайно важливою проблемою, що вимагає нагального вирішення.

Основна ідея методу системної інженерії систем Data Science полягає у застосуванні методології системної інженерії та бізнес-профіля Еріксона–Пенкера для формалізації упорядкованого способу розроблення системи Data Science.

Імплементація системи Data Science визначення емоційного забарвлення публікацій, що характеризують війну в Україні дає можливість встановити рівень напруженості у суспільстві, емоційний стан різних верств громадян та їх відношення до наслідків війни.

1. ПОСТАНОВКА ЗАДАЧІ

Об’єктом дослідження є найбільш відомі методи обробки неструктурованих текстів, методи векторизації та класифікації тексту, існуючі системи та програмні засоби аналізу тональності тексту.

Предметом дослідження є математичне та програмне забезпечення системи для визначення емоційного забарвлення публікацій що характеризують війну в Україні на основі методів машинного навчання.

Для досягнення мети роботи потрібно вирішити такі завдання:

1) Провести огляд існуючих теоретичних інструментів та існуючих засобів аналізу тональності тексту;
2) Обґрунтування та вибір методів векторизації тексту;
3) Обґрунтування та вибір методів сентимент аналізу тексту;
4) Розробка системи визначення емоційного забарвлення публікацій, що характеризують війну в Україні в соціальній мережі Твіттер;
5) Верифікація та валідація системи.

Метою роботи є системна інженерія математичного і програмного забезпечення системи Data Science для визначення емоційного забарвлення публікацій в соціальній мережі Твіттер що характеризують війну в Україні.

Кінцевим результатом роботи є система Data Science, математичне та програмне забезпечення системи Data Science, що визначає емоційне забарвлення публікацій що характеризують війну в Україні.

2. ОГЛЯД ТА АНАЛІЗ ІСНУЮЧИХ ТЕОРЕТИЧНИХ ІНСТРУМЕНТІВ І ЗАСОБІВ АНАЛІЗУ ТОНАЛЬНОСТІ ТЕКСТУ

2.1 Підходи до сентимент аналізу тексту

Існує три популярних підходи, які широко використовуються для сентимент аналізу: на основі лексикону, на основі машинного навчання та гібридні. Усі ці підходи передбачають різні обчислювальні кроки. Підхід, заснований на лексиконі, визначає полярність або настрої, щоб класифікувати твіти за трьома різними категоріями: негативні, нейтральні та позитивні. З іншого боку, підхід, заснований на МН, також популярний серед дослідників, особливо коли доступна велика кількість позначених зразків даних.

2.1.1 Підхід на основі лексики

Одним із підходів або прийомів семантичного аналізу є підхід на основі лексики . Ця техніка обчислює орієнтацію настроїв. Техніки на основі лексики використовують прикметники та прислівники для виявлення семантичної спрямованості тексту. Для розрахунку будь-якої орієнтації тексту комбінації прикметників і прислівників виділяються зі значенням орієнтації на сентимент. Потім їх можна перетворити в одну оцінку для цілого значення.

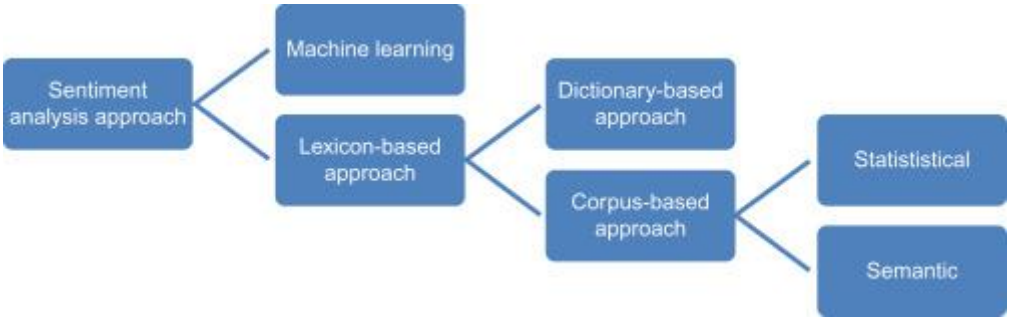


Рисунок 2.1 [1]

Різновиди підходу на основі лексики:

1. Підхід на основі словника.

У цьому підході словник створюється, спочатку взявши кілька слів. Тоді онлайн-словник, можна використовувати для розширення цього словника шляхом включення синонімів та антонімів цих слів. Словник розширюється до тих пір, поки до цього словника не можна буде додати нові слова. Словник можна уточнити шляхом ручної перевірки.

2. Корпусний підхід

Статистичний підхід: слова, які демонструють непостійну поведінку в позитивній поведінці, вважаються позитивними. Якщо вони показують негативповторення в негативному тексті вони мають негативну полярність. Якщо частота однакова як у позитивному, так і в негативному тексті, то слово має нейтральну полярність.

Семантичний підхід: цей підхід присвоює значення настрою окремим словам і словам, які семантично ближчі до цих слів; це можна зробити, знайшовши синоніми та антоніми щодо цього слова.

Найбільшо проблемою методів заснованих на лексиконі є важкість процесу складання словника. Він потребує великих затрат часу та зазвичай корисний лише у дуже вузькій предметній області.

2.1.2 Підхід на основі машинного навчання.

Інший спосіб, який можна використати для сентимент аналізу - це машинне навчання, яке в свою чергу поділяється на методи навчання з учителем, методи навчання без вчителя та напівконтрольоване навчання.

Методи навчання з учителем потребують наявність розмічених навчальних даних, які використовуються для навчального процесу. Задача сентимент аналізу тексту відноситься до задач класифікації, коли класифікатору подається на вхід текст,представлений у вигляді вектору, і який повертає відповідний клас.У процесі

навчання модель вчиться присвоювати вхідним даним відповідну мітку на основі зразків з навчального набору даних. Підходи з використанням методів машинного навчання з учителем дають доволі високу точність класифікації.

2.2 Попередня обробка тексту.

У будь-якій задачі машинного навчання або завдання аналізу даних першим і головним кроком є очищення та обробка даних. Дані із соціальних мереж в основному публікуються у формі вільного тексту та мають свої особливості, можуть містити різні спеціальні символи, смайли, емодзі і т.д.

Загалом використовують наступні кроки пов’язані з попередньою обробкою текстових даних, як-от Twitter [2]:

a) Видалення шуму. Цей крок видаляє всі шуми, такі як хештег, зображення профілю, ретвіти, емодзі, URL-адреси, імена користувачів, цифри, розділові знаки, пробіли, теги, знак @ тощо.
b) Зміна у нижньому регістрі: твіти потім змінюються на малі літери.
c) Токенізація: Токенізація — це процес НЛП, за допомогою якого широкі текстові дані поділяються на менші частини, які називаються токенами. Він поділяє речення на групу слів. Вона може бути двох типів: лексема слів і лексема речення. У цій роботі ми використали токенізацію слів. Приклад можна побачити на рисунку (2.2).

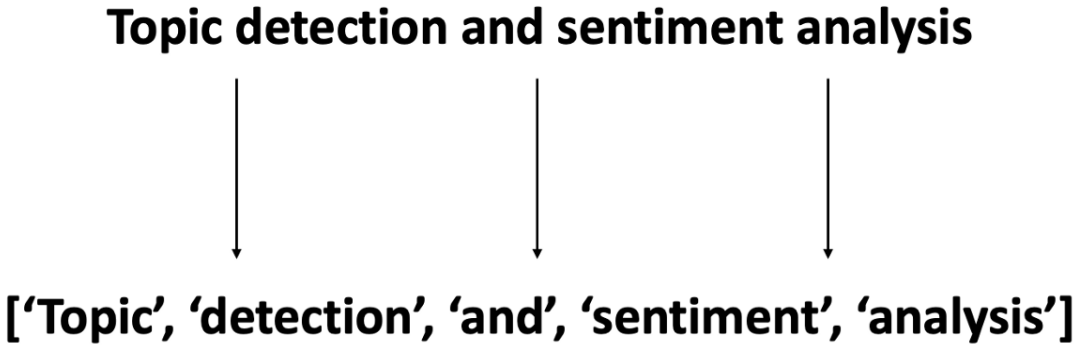


Рисунок 2.2 [3]

d) Видалення стоп-слів: усі артиклі, прийменники та сполучники, такі як am, are, an, the, is тощо, є прикладом стоп-слів і є часто вживаними словами в НЛП. Вони не несуть багато дискримінаційного вмісту. Також не відіграє важливої ролі в процесі класифікації настроїв і є марною інформацією, що міститься в будь-якому тексті, як-от твіт, тому його потрібно видалити.

е) Стеммінг та лематизація: це парування (стеммінг), яке об'єднує споріднені слова з асоційованим коренем. Наприклад, текстовий документ, що містить однину, множину, різні часи й інші варіації, як-от reads, reading, що належить до однієї основи, буде перетворено у відповідне слово read. Твіт може містити форму однини або множини одного слова, різні часи тощо. Лемматизація – це тип парування, який використовує конкретну частину мови для визначення кореневої форми слова. Правила нормалізації залежать від частини мови, а отже, вони дуже специфічні для мови. Лемматизацію вважають відмінною від стеммінгу, оскільки вона виходить за рамки простих правил видалення та використовує морфологічні корені слів.

2.3 Методи векторизації тексту.

Алгоритми машинного навчання виконуються з використанням таких математичних речей, як статистика, алгебра, обчислення тощо. Вони очікують, що дані будуть числовими, наприклад, двовимірним масивом із рядками як екземплярами та стовпцями як ознаками. Проблема з природною мовою полягає в тому, що дані мають форму необробленого тексту, тому текст потрібно перетворити у вектор. Процес перетворення тексту у вектор зазвичай називають векторизацією тексту. Це фундаментальний процес обробки природної мови, оскільки жоден з алгоритмів машинного навчання не розуміє текст, навіть комп'ютери [4].

2.3.1 Bag-of-Words

Модель «мішок слів» (Bag-of-Words) є одним із найпростіших методів які використовуються для вилучення ознак із текстових даних. Це простий у застосуванні та легкий для інтерпретації підхід. Незважаючи на простоту, вона часто дає конкурентоспроможні результати в порівнянні зі складними підходами вилучення ознак. Він підраховує частоту кожного унікального терміна в корпусі і створює числовий вектор ознак, тим самим зменшуючи складність навчання моделі. BoW можна уявити у вигляді матриці, в якій кожному рядку відповідає окремий докумен, а кожному стовпцю — унікальне слово. На перетині рядка і стовпця знаходиться кількість входжень слова у даний документ [5].

Наведемо приклад на трьох відгуках на фільми [6]:

- Відгук 1: This movie is very scary and long
- Відгук 2: This movie is not scary and is slow
- Відгук 3: This movie is spooky and good

Спочатку ми створимо словниковий запас з усіх унікальних слів у трьох вищенаведених оглядах. Словниковий запас складається з 11 слів: ‘This’, ‘movie’, ‘is’, ‘very’, ‘scary’, ‘and’, ‘long’, ‘not’, ‘slow’, ‘spooky’, ‘good’. Тепер ми можемо взяти кожне з цих слів і позначити їх зустріч у трьох оглядах фільмів вище 1 і 0. Це дасть нам 3 вектори для 3 оглядів:

	1 This	2 movie	3 is	4 very	5 scary	6 and	7 long	8 not	9 slow	10 spooky	11 good	Length of the review(in words)
Review 1	1	1	1	1	1	1	1	0	0	0	0	7
Review 2	1	1	2	0	0	1	1	0	1	0	0	8
Review 3	1	1	1	0	0	0	1	0	0	1	1	6

Рисунок 2.3 [6]

Вектор відгуку 1: [1 1 1 1 1 1 1 0 0 0 0]

Вектор відгуку 2: [1 1 2 0 0 1 1 0 1 0 0]

Вектор відгуку 3: [1 1 1 0 0 0 1 0 0 1 1]

І це основна ідея моделі Bag of Words (BoW).

Недоліки використання моделі Bag-of-Words (BoW) [5]:

У наведеному вище прикладі ми можемо мати вектори довжиною 11. Однак ми починаємо стикатися з проблемами, коли зустрічаємо нові речення:

1. Якщо нові речення містять нові слова, то розмір нашого словникового запасу збільшиться, а отже, збільшиться і довжина векторів.
2. Крім того, вектори також будуть містити багато нулів, що призведе до розрідженої матриці (чого б ми хотіли уникнути)
3. Ми не зберігаємо жодної інформації ні про граматику речень, ні про порядок слів у тексті.

2.3.2 TF-IDF

Термін частота інверсної частоти документа — це текстовий векторизатор, який перетворює текст у придатний для використання вектор. Він надає зважені функції в порівнянні з простими функціями підрахунку від BoW. Він обчислює вагу кожного терміна в корпусі, і ці зважені характеристики є більш ефективними при навчанні моделей. TF-IDF включає TF і IDF, де перший підраховує появу кожного унікального терміну, а другий призначає ваги кожному терміну на основі його появи в різних документах. TF можна обчислити за допомогою формули (1) [7].

$$TF = t_D/D_t \qquad (2.1)$$

де t_D - кількість разів, коли термін t з'являється в документі D, а D_t - кількість термінів в документі D.

IDF можна обчислити як (2):

$$IDF = log\left(\frac{N}{N_{D,t}}\right), \quad (2.2)$$

де N – кількість документів, а $N_{D,t}$ – кількість документів, що містять термін t. Отже, TF-IDF можна обчислити як

$$TF - IDF = TF * IDF \ , \quad (2.3)$$

Обмеження:

- Він корисний лише як функція лексичного рівня.
- Синонімами нехтують.
- Це не фіксує семантику.
- Найвища оцінка TF-IDF може не мати сенсу з темою документа, оскільки IDF надає високу вагу, якщо DF терміна низький.
- Він нехтує послідовністю термінів.

2.3.3 Word2Vec

Word2Vec — це підхід до вилучення ознак, розроблений Google, відрізняється від BoW і TF-IDF, оскільки вони використовують повторення терміну в корпусі, тоді як Word2Vec використовує подібність між словами та дозволяє представляти слово у вигляді вектора (часто називають вбудовуванням). Він використовує косинусну подібність, щоб знайти подібність між словами. Представлення слів за допомогою Word2Vec можна здійснити двома основними методами: Skip-gramта Continious Bag of Words (CBOW) [8].

- Метод безперервного пакета слів (CBOW) – цей метод допомагає закінчити часткове неповне речення, передбачаючи слова, які можна вмістити в середину речення на основі контексту слів. Контекст передбачення залежить

від кількох слів перед і після передбаченого слова. Ці методи називають методами мішка слів, оскільки порядок слів у контексті не важливий.

- Метод Skip-Gram – Цей метод використовується для передбачення слів контексту або оточуючих слів, заданих поточним словом у тому самому реченні. Модель Skip-gram приймає кожне слово великого корпусу як вхідні дані, а прихований або вбудовуваний шар, використовуючи вагові коефіцієнти вбудовування, прогнозує контекстні слова.

  Нижче представлена базова архітектура моделей CBOW і Skip-Gram.

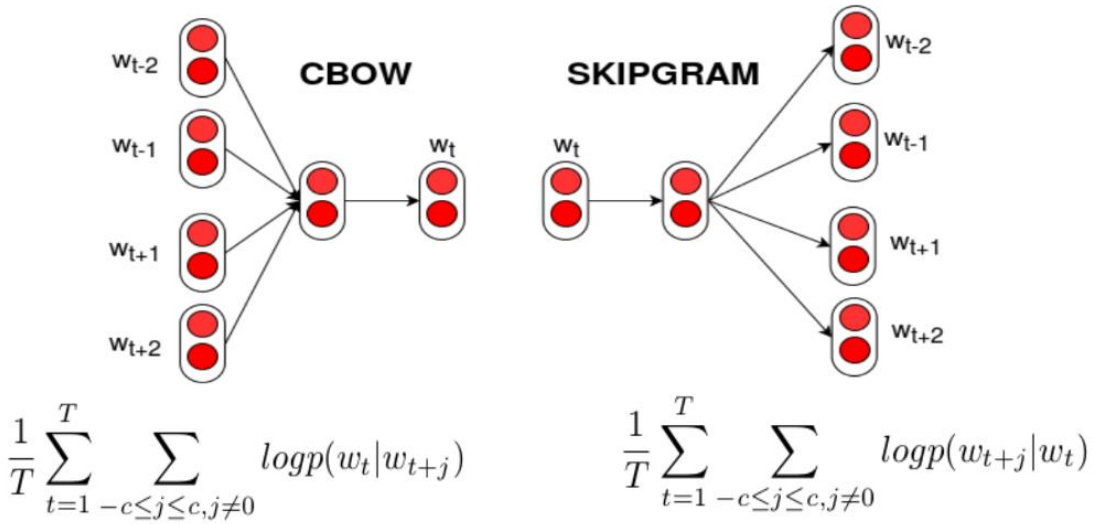


Рисунок 2.4 [9]

Можливо, найбільшою проблемою word2vec є нездатність обробляти невідомі слова або поза словникові слова. Якщо модель раніше не зустрічала слова, вона не матиме уявлення, як його інтерпретувати або як побудувати для нього вектор. Тоді ви змушені використовувати випадковий вектор, який далекий від ідеалу. Також Word2Vec вивчає вбудовування, пов’язуючи цільові слова з їх контекстом. Однак він ігнорує, чи з’являються деякі контекстні слова частіше, ніж інші. Для Word2Vec часте спільна поява слів створює більше навчальних прикладів **,** але не несе додаткової інформації [10]**.**

2.3.4 GloVe

Слово GloVe є поєднанням двох слів – Global та Vectors. Поглиблено, GloVe— це модель, яка використовується для представлення поширених слів. Ця модель представляє слова у вигляді векторів за допомогою неконтрольованого алгоритму навчання. Цей алгоритм навчання без нагляду відображає слова в простір, де семантична подібність між словами спостерігається за відстанню між словами. Навчання виконується з використанням базової моделі, заснованої на підрахунку, на агрегованій глобальній матриці спільної появи від слова до слова в текстовому корпусі і результат навчання зазвичай представляє підпростір слів, які нас цікавлять. Він розроблений як проект із відкритим кодом у Стенфорді та був запущений у 2014 році.

Модель GloVe поєднує у собі властивості двох моделей, а саме, методів глобального розкладу матриць, та локального контекстного вікна. Вона починає працювати зі створення великої матриці, яка складається з інформації про спільні появи слів корпусу. Ідея цієї матриці полягає в тому, щоб вивести зв’язок між словами зі статистики. Матриця спільної появи повідомляє нам інформацію про входження слів у різні пари. Подивимося на зображення рисунок 2.5 [8] .

Тут ми маємо матрицю, яка утворена шляхом розгляду унікальних слів із двох речень, наведених зліва від зображення, і на ньому ми бачимо, що слово «the» нараховано 3 рази за допомогою «cat»,«fast» і«hat», тому значення в матриці рівне 3 кожного разу.

Тепер за допомогою матриці ми можемо обчислити відношення ймовірностей між будь-якими двома парами слів. Для прикладів скажемо ймовірності (cat/fast) = 1 і (cat/the)=0,5.

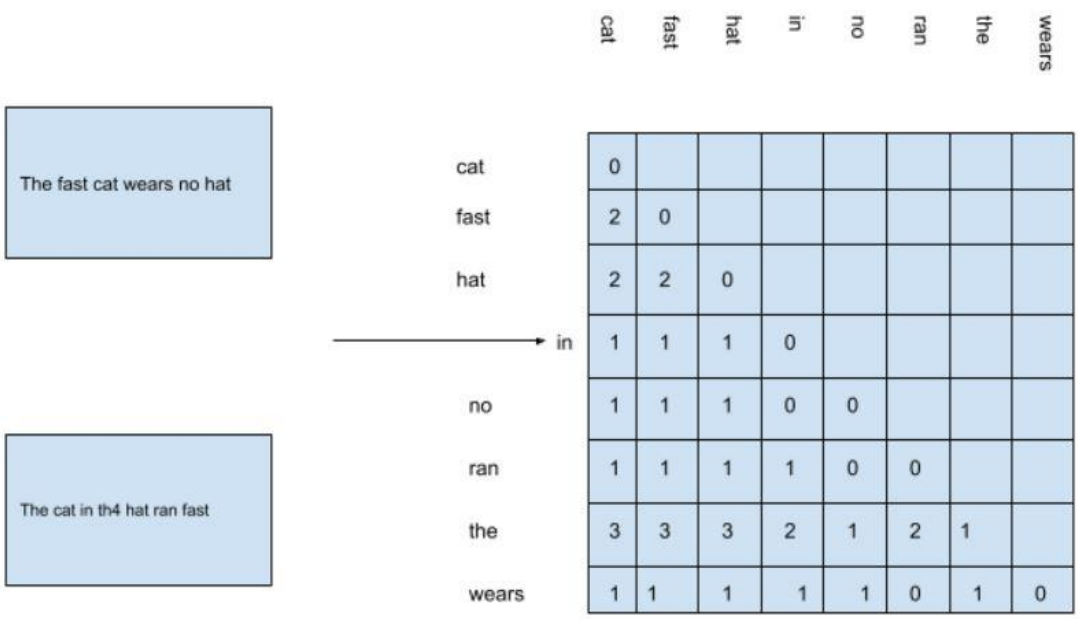


Рисунок 2.5 [8]

Відношення цих ймовірностей буде 2 і за цим співвідношенням ми можемо зробити висновок, що «fast» є більш релевантним, ніж «the». На основі ймовірностей метод GloVe можна представити математично як

$$F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}, \qquad (2.4)$$

де P_{ik} – ймовірність що слово ksустрілося в контексті слова i;

P_{jk}- ймовірність того, що слово k зустрілося в контексті слова j.

На відміну від Word2Vec, GloVe підкреслює, що частота спільних зустрічей є життєво важливою інформацією , і її не слід «витрачати» як додаткові навчальні приклади. Натомість GloVe створює вбудовування слів таким чином, що комбінація векторів слів безпосередньо пов’язана з ймовірністю спільної появи цих слів у корпусі.

2.4 Методи МН для аналізу тональності тексту.

Розглянемо більш детально методи МН, які використовуються для сентимент аналізу тексту, а саме: Наївний Байєсівський класифікатор, метод опорних векторів (SVM) та рекурентні нейронні мережі.

2.4.1 Наївний Байєсівський класифікатор

Наївний Байєс є найбільш часто використовуваним класифікатором в обробці природної мови (NLP). Головною перевагою цього класифікатора є його простота, а також передбачення правильного класу для нового екземпляр. Наївний класифікатори Байєса припускає, що кожна ознака або атрибут екземпляра розглядається незалежно від усіх інших ознак у даному класі. І в результаті він помножить усі члени вектора ознак у даному класі, щоб обчислити байєсівську ймовірність. Формула наведена нижче [11]:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \qquad (2.5)$$

де: P(A|B) - ймовірність гіпотези A за умови події B;

P(B|A) - екземпляр відбувся в конкретному класі для кожного значення A;

P(A) - попередня ймовірність класу;

P(B) — гранична ймовірність

Коли теорема Байєса використовується для класифікації текстових документів, клас c конкретного документа d визначається як:

$$C_{map} = \text{argmax}\,(P(c|d)\,) = \text{argmax}\,(P(c)/P(t_k|d)), \qquad (2.6)$$

де: t_k – терміни документа;

c – набір класів використовуваних для класифікації;

P(c | d) – умовна ймовірність класу с даного документа;

P(c) – пріоритетна ймовірність класу с;

$P(t_k|d)$ – умовна ймовірність токену t_k даного класу.

Тобто, щоб відшукати, до якого класу належить новий документ, необхідно оцінити добуток вірогідності кожного терміна документа, що задається певним класом та ймовірності конкретного класу. Підсумувавши попереднє для всіх класів множини С, оберемо той, який містить найбільшу ймовірність.

Для того щоб запобігти переливу плаваючої точки при обчисленні замість максимізації добутку ймовірностей, максимізуємо суму їх логарифмів:

$$C_{map} = \text{argmax} (\log P(c) + \sum_{1 \le k \le nd} \log P(t_k|c)). \qquad (2.7)$$

Це означає, що будемо обирати клас із найвищим рівнем протоколу, а не з найбільшою ймовірністю.

Основною проблемою цього методу є те, що не обов'язково певна ознака (слово) зустрінеться у певному класі. У такому випадку умовна ймовірність дорівнює 0. Для вирішення цієї проблеми використовується згладжування Лапласа, шляхом додавання 1 до кожного рахунку:

$$P(t|c) = (T_{ct} + 1)/ \sum_{ctV}(T_{ct'} + 1) = (T_{ct} + 1)/ \sum_{ctV}(T_{ct'}) + B') \qquad (2.8)$$

де: B′ дорівнює кількості термінів які знаходяться у словнику V.

2.4.2 Метод опорних векторів (SVM)

Машина опорних векторів — це контрольований алгоритм навчання, здебільшого використовується для задач класифікації. У SVM точки даних зображуються в n-вимірному просторі, де n — кількість ознак. Потім класифікація виконується шляхом вибору відповідної гіперплощини, яка розрізняє

два класи. Більш детально це можна побачити на наступних рисунках (2.6) – (2.7) [12].

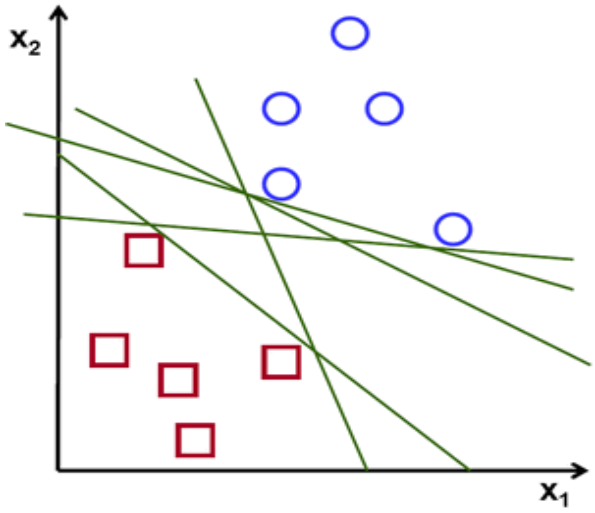


Рисунок 2.6 - Найбільш ймовірні лінії розділу для досліджуваних даних [12]

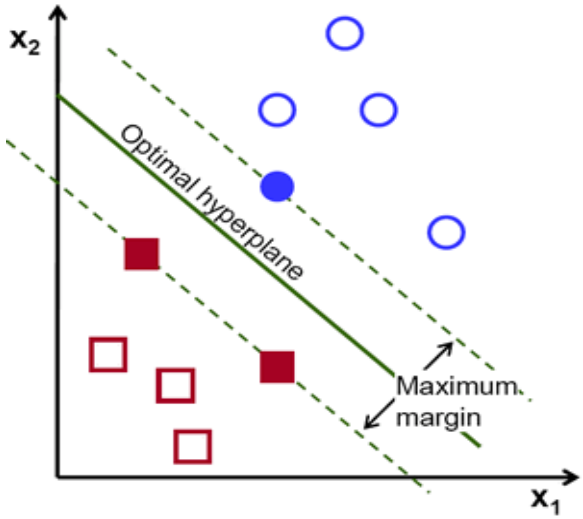


Рисунок 2.7 – SVM алгоритм [12]

SVM знаходить гіперплощину, яка правильно розділяє навчальні дані, максимізуючи відстань між двома класами. Алгоритм використовує функції ядра для відокремлення лінійних нероздільних наборів даних шляхом відображення вихідних векторів даних у простір вищого розміру, який ще називають простором ознак.

Опорні вектори – це вектори даних на краях гіперплощини і впливають на положення та орієнтацію гіперплощини.

У SVM ми беремо вихід лінійної функції, і якщо цей вихід більше 1, ми ідентифікуємо це з одним класом, і якщо вихід дорівнює -1, ми ідентифікуємо це з іншим класом. Тобто, ми отримуємо діапазон значень підкріплення ([-1,1]), який діє як запас.

В алгоритмі SVM ми прагнемо максимізувати межу між точками даних і гіперплощиною. Функція втрат, яка допомагає максимізувати маржу, - це втрата шарнірів.

$$\mathbf{c}(x, y, f(x)) = (1 - y * f(x))_+ \qquad (2.9)$$

Вартість дорівнює 0, якщо прогнозована і фактична вартість мають однаковий знак. Якщо їх немає, ми розраховуємо величину втрат. Ми також додаємо параметр регуляризації функцію вартості. Метою параметра регуляризації є збалансувати максимізацію маржі та втрати. Після додавання параметра регуляризації функції вартості виглядають, як показано нижче.

$$min_w \lambda \|w\|^2 + \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle)_+ \qquad (2.10)$$

Тепер, коли у нас є функція втрат, ми беремо часткові похідні відносно ваг, щоб знайти градієнти. Використовуючи градієнти, ми можемо оновити наші ваги. Якщо немає помилкової класифікації, тобто наша модель правильно прогнозує клас нашої точки даних, нам залишається лише оновити градієнт із параметра регуляризації.

$$w = w - \alpha \cdot (2\lambda w) \qquad (2.11)$$

Коли відбувається неправильна класифікація, тобто наша модель робить помилку в передбаченні класу нашої точки даних, ми включаємо втрату разом з параметром регуляризації для виконання оновлення градієнта.

$$w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w) \quad (2.12)$$

2.4.3 Рекурентні нейронні мережі

Рекурентні нейронні мережі (Recurrent neural networks, RNN) – вид рекурсивних нейронних мереж, які «пам'ятають» свій попередній стан та роблять обчислення з урахуванням цього стану.

В розгорнутому стані структура RNN виглядає наступним чином [13]:

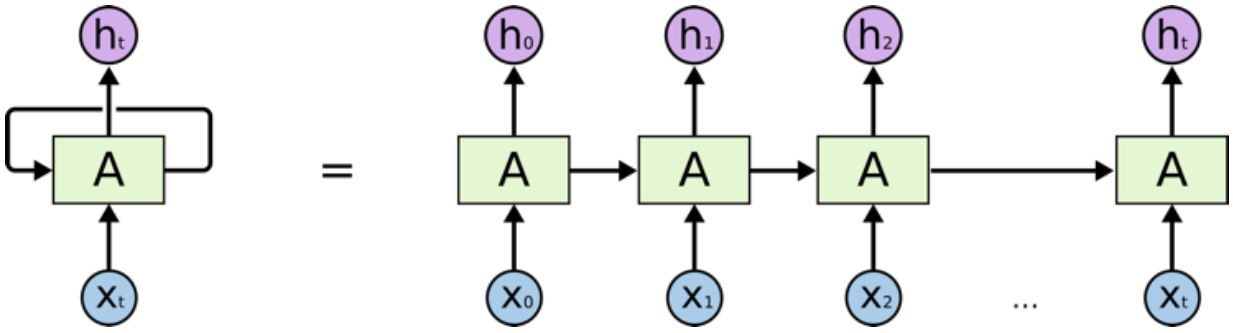


Рисунок 2.8 [13]

Типовий RNN зображений на рисунках (2.9) – (2.10), де X(t) є вхідним параметром, h(t) є вихідним, а A є нейтронною мережею, яка отримує інформацію з попереднього кроку в циклі. Вихід одного блоку переходить до наступного, і інформація передається.

У деяких випадках, щоб виконати поточне завдання, нам потрібно лише переглянути останню інформацію. Розглянемо як приклад мовну модель, яка на основі попередніх слів намагається передбачити слово яке буде наступним. Намагатимемося передбачити останнє слово у реченні «хмари на *небі*», тут не потрібен додатковий контекст, бо цілком очевидно, що наступне слово це небо. Тому у випадках, коли

розрив між відповідною інформацією та розташуванням невеликий, RNN можуть вивчитися використовувати минулу інформацію. Але існують випадки, які потребують більше контексту. Спробуймо передбачити останнє слово в реченні «Я виріс у Франції… я вільно розмовляю *французькою* ». Останні данні дають відомості про те, що наступне слово, ймовірно, є назвою мови, проте для того щоб звузити мову, нам потрібен контекст Франції з дальнього краю. Цілком вірогідно, що розрив між відповідною інформацією та місцем, яке її потребує, стане дуже великим.

На жаль, оскільки цей розрив зростає, RNN не можуть навчитися зв’язувати інформацію. Під час навчання RNN, оскільки інформація знову і знову обертається, що призводить до дуже великих оновлень ваг моделі нейронної мережі. Це пов’язано з накопиченням градієнтів помилок під час оновлення і, отже, призводить до нестабільної мережі. У крайньому випадку значення ваг можуть стати настільки великими, що переповнюються і призводять до значень NaN. Вибух відбувається через експоненціальне зростання шляхом багаторазового множення градієнтів через шари мережі, значення яких більше 1, або затухання відбувається, якщо значення менші за 1.

2.4.4 LSTM

LSTM (довготривала короткочасна пам’ять) є популярним рішенням проблеми затухаючого градієнта в RNN. LSTM є розширенням для RNN, оскільки воно дозволяє RNN запам'ятовувати введені дані протягом тривалого періоду часу .

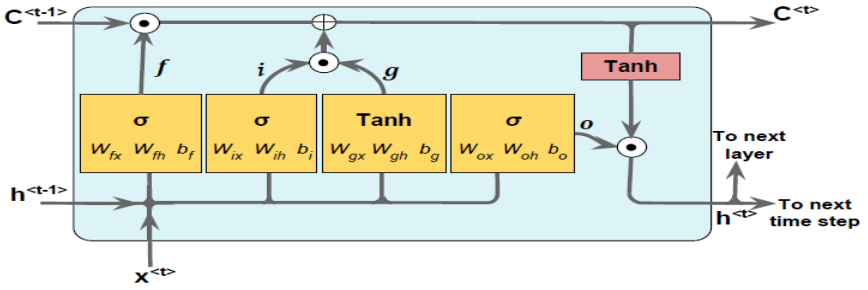


Рисунок 2.9 [14]

Комірка LSTM приймає прихований стан попередньої точки часу h<t−1> і стан комірки попередньої точки часу c<t−1> і виводить стан комірки c<t> і прихований стан h<t>. LSTM може мати довгострокову та короткочасну пам’ять, використовуючи 3 вентилі (або ворота): вхідний, забуття та вихідний. Вхідний вентиль визначає пропускати вхід чи ні; ворота забуття видаляють інформацію, яка не є важливою; вихідний вентиль визначає, чи вплине інформація на вихід у поточному стані чи ні. Більше того, оскільки всі ворота активуються за допомогою сигмоїдної функції, яка коливається від 0 до 1. Проблема затухаючого градієнта вирішується за допомогою LSTM. Тому LSTM зазвичай має відносно високу точність і короткий час навчання порівняно з традиційними RNN.

2.5 Порівняння описаних методів класифікації тексту

Таблиця 1.1 – порівняльний аналіз методів класифікації тексту

	Наївний Баєсів класифікатор	Метод опорних векторів	Нейронна мережа LSTM
Швидкість навчання	Швидко	Швидко	Ресурсномісткий і вимагає часу
Інтерпретація результатів	Результати цілком зрозумілі	Результати цілком зрозумілі	Важче інтерпретувати результати
Переваги	• Швидкість обчислення • Можливість фіксувати контекст	• Добре працює коли є чіткий поділ між класами • Ефективний для пам’яті	• Має потенціал отримання найбільш точних результатів • Може вивчати важливі ознаки • Ефективна у запам’ятовуванні важливої інформації • Фіксує контекст
Недоліки	• Покладається лише на повний і репрезентативний набір даних • Сильне припущення про незалежність у моделі • Проблема «нульової частини»	• Погано працює на великих наборах даних • Не дуже добре працює, коли набір даних має більше шуму	• Найдорожчі навчання та експлуатація

Насправді, ефективність класифікації настроїв за допомогою різних комбінацій алгоритмів і методів вилучення ознак (векторизації) може давати досить різноманітні результати. Виходячи з огляду теоретичних матеріалів та різних досліджень продуктивність алгоритму нейронної мережі є надзвичайною, у порівнянні з іншими методами машинного навчання. Загалом, методи SVM та НБК мають майже однакову ефективність. Крім того, на відміну від НБК та SVM, на результати моделювання, отримані нейронною мережею також не впливають суттєво ні збалансовані, ні незбалансовані умови складання даних. Результат завжди високий. Саме тому у подальшій роботі ми будемо використовувати нейронну мережу LSTM.

2.6 Висновки до розділу

У розділі булу розглянуто основні підходи до сентимент аналізу тексту.

1. Розглянули такі методи векторизації тексту: Bag-of-Words, TF-IDF, Word2Vec та GloVe. Bag of Words просто створює набір векторів, що містить кількість зустрічей слів у документі (рецензії), тоді як модель TF-IDF містить інформацію як про важливіші слова, так і про менш важливі. Вектори Bag of Words легко інтерпретувати. Однак TF-IDF зазвичай краще працює в моделях машинного навчання. Хоча як Bag-of-Words, так і TF-IDF були популярні у своїх власних відносинах, все ще залишалася порожнеча в розумінні контексту слів. Щоб виявити схожість між словами «spooky» та «scary», потрібно набагато більше інформації про документи. Ось тут на допомогу приходять такі методи вбудовування слова, як Word2Vecта GloVe.Word2Vec приймає тексти як навчальні дані для нейронної мережі. Отримане вбудовування визначає, чи з’являються слова в подібних контекстах. GloVe зосереджується на спільних зустрічах слів у всьому корпусі. Його вбудовування пов’язані з ймовірністю того, що два слова з’являться разом.

2. Також було розглянуто такі методи машинного навчання для для класифікації тексту: наївний Баєсів класифікатор, метод опорних векторів та

нейронні мережі LSTM. На основі порівняння характеристик цих методів було визначено, що найкращим вибором для побудови системи буде нейронна мережа типу LSTM.

3. МАТЕМАТИЧНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ВИЗНАЧЕННЯ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ПУБЛІКАЦІЙ ЩО ХАРАКТЕРИЗУЮТЬ ВІЙНУ В УКРАЇНІ В СОЦІАЛЬНІЙ МЕРЕЖІ ТВІТТЕР

3.1 Модель системи. Діаграма компонентів.

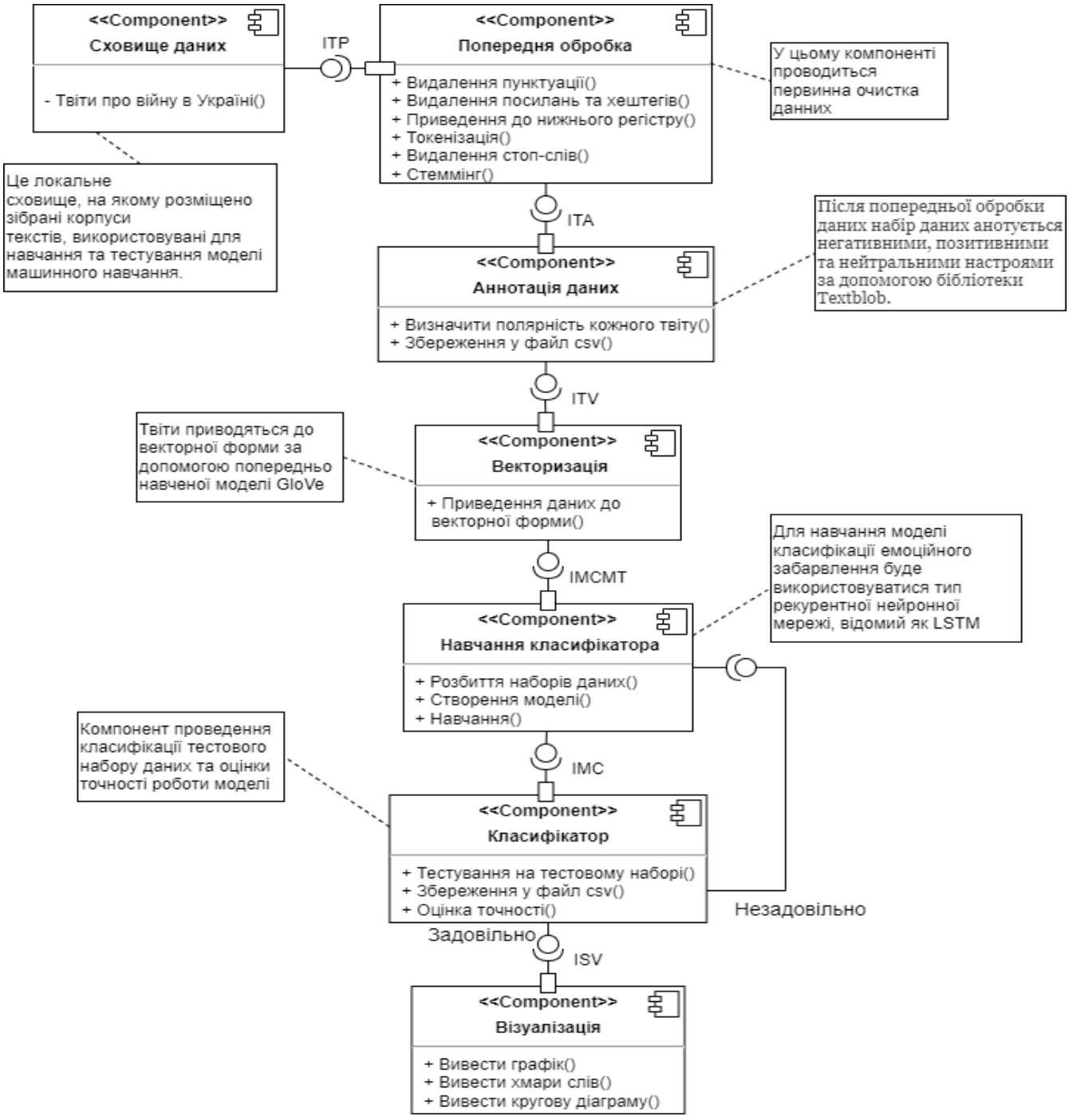


Рисунок 3.1 Модель системи. Діаграма компонентів у нотації UML.

3.1.1 Опис функціональності і компонентів системи

Компонент «Сховище даних» – це сховище, на якому розміщені зібрані корпуси текстів, використовувані для навчання та тестування класифікатора.

Набір даних було отримано з сайту kaggle.com «Russia vs Ukraine Tweets Data set» [15]. Він містить твіти, пов'язані з війною в Україні починаючи з 1 січня 2022 по 6 березня 2022. Ця шкала включає період перед і після вторгнення. Всього присутні 1316605 рядків, і кожен рядок містить інформацію про один твіт. Інша інформація включає дату та час, мову, хеш теги, місце, кількість вподобань, кількість ретвітів, посилання тощо.Після процедури feature enginiring було видалено дублікати і неангломовні твіти і отримано 1204118. Далі розглядаються лише 2% набору даних. Вибірка формується таким чином, що кожен рядок має однакове зважування ймовірності, і можна було відібрати рядок лише один раз, і він не буде доступний для вибору знову. У вибірковому наборі даних загалом присутні 24084 твіти.

Компонент «Попередня обробка» призначений для видалення непотрібної, зайвої інформації з даних, які не сприяють прогнозуванню цільового класу. У ньому виконуються наступні кроки:

- Видалення посилань та хештегів;
- Видалення розділових знаків;
- Видалення нульових значень;
- Перетворення в нижній регістр;
- Токенізація.
- Стеммінг;
- Видалення стоп-слів.

Компонент «Анотація даних». Після попередньої обробки даних набір даних анотується негативними, позитивними та нейтральними настроями за допомогою бібліотеки Textblob. Textblob є одним з найбільш широко використовуваних методів оцінки настроїв, заснованих на лексиці. Він знаходить оцінку полярності заданого

тексту, яка коливається від -1 до 1. Ці настрої використовуються в цьому дослідженні як цільовий клас для навчання моделі класифікації. Після анотації даних набір даних розбивається на навчальні та тестові набори даних для навчання та тестування моделей відповідно. Співвідношення навчальних і тестових даних становить 80:20, де 80% даних використовується для навчання, а 20% – для тестування.

Розглянемо деякі основи TextBlob, він використовує NLTK (Natural Language ToolKit), а вхідні дані містять одне речення. Результатом TextBlob є не лише полярність, а і суб'єктивність, інтенсивності негативного, нейтрального, позитивного та compound. Оцінка compound знаходиться між (-1 до 1), де -1 визначає найбільш негативні слова, такі як «огидно», «жахливо», «жалюгідно», а 1 визначає найбільш позитивні слова, як-от «відмінно», «найкраще». Оцінка суб'єктивності знаходиться між (0 і 1), якщо речення має високу суб'єктивність, тобто близьке до 1. Проте у роботі ми її використовувати не будемо. Твіттер є платформою де люди виражають свої особисті думки, вважається, що всі твіти є суб'єктивними.

Компонент «Векторизація» перетворює кожне зі слів у вбудоване слово. Вбудовуванням слів називають векторні уявлення, які фіксують контекст базових слів по відношенню до інших слів у реченні. Ця трансформація призводить до того, що слова, що мають подібне значення, згруповані ближче один до одного в гіперплощині, а різні слова розташовані далі в гіперплощині. Ми будемо використовувати попередньо навчену модель вбудовування слів, відому як GloVe [16]. GloVe в основному є моделлю на основі підрахунку, де спочатку створюється велика матриця спільної появи, а потім виконуємо зменшення розмірності на ній. Після факторізації кожен вектор-рядок в отриманій матриці нижнього розміру використовується для представлення кожного слова у словнику. Цей процес досягається шляхом навчання логарифмічної білінійної регресії з метою мінімізації втрат при реконструкції. Основна ідея GloVe полягає в тому, щоб визначити відносне співвідношення між трьома словами за допомогою відношення

ймовірностей спільної появи. Наприклад, є три слова w_i ,w_j і w_k. Тепер припустимо, що w_k з'являється частіше в контексті w_i, ніж w_j. Таким чином, відношення спільної появи $\frac{P(w_k|w_i)}{P(w_k|w_i)}$ має бути великим. Якщо w_k з'являється частіше в контексті w_j, ніж w_i, коефіцієнт спільної появи $\frac{P(w_k|w_i)}{P(w_k|w_i)}$ має бути малим. Якщо w_k часто з'являється як в контексті w_j, так і в w_i, або рідко з'являється в обох контекстах, тоді коефіцієнт спільної появи має бути близьким до одиниці. Роблячи це, ми можемо відрізнити відповідні слова від невідповідних слів. Отримані уявлення слів демонструють цікаву лінійну властивість у векторному просторі, так що різниця між векторами може охоплювати значення, визначене паралельністю двох слів.

Так як процес навчання вбудовування слів вимагає великої кількості даних і є дуже ресурсномістким, було вирішено використовувати для нашої системи попередньо навчений GloVe, як ініціалізацію шару вбудовування слів у нашій моделі аналізу настроїв. Тут ми використовуємо ту саму модель, що й вище, але замість того, щоб обчислювати матрицю спільної появи на основі наших власних навчальних даних, ми збираємося використовувати попередньо навчену матрицю спільної появи, розраховану на основі всієї сторінки у Вікіпедії. Імовірно, це набагато більш повний і вичерпний словник векторів слів і може мати більш точні вектори, однак Вікіпедія не містить багато неформальної чи сленгової мови, і тому якщо слово не має представлення Glove, воно буде представлене з нульовою матрицею.

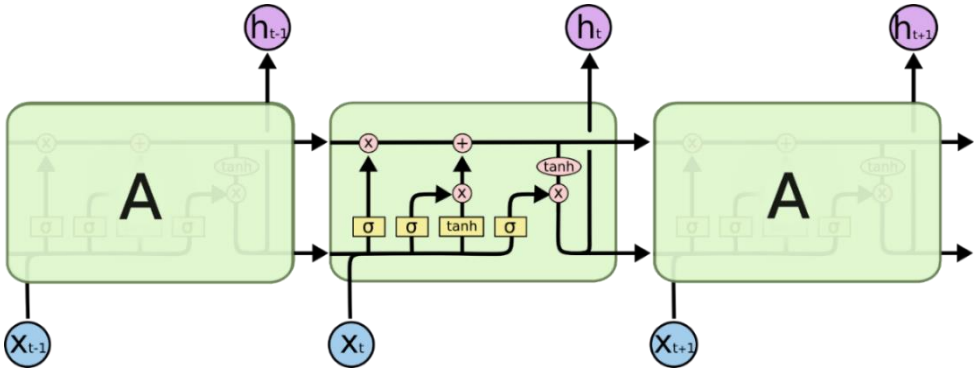


Рисунок 3.2 [13]

Компонент «Навчання класифікатора». Основним складовим цього компонента є власне модель класифікації. Для навчання ми будемо використовувати тип рекурентної нейронної мережі LSTM.

Робота LSTM полягає в тому, що використовує вхідні вентилі, які поєднують поточний вхід (xt) і вихід попереднього блоку LSTM, використовуючи функцію активації tanH, щоб об'єднати наші входи в один, а потім виконувати поелементне множення на, як правило, деякі функції активації сигмовидної форми.

Потім цей вхід проходить через адитивне перетворення, в результаті чого до нього додається нове значення St-1. Це внутрішній стан нашої комірки LSTM і перетворює наше попереднє вхідне значення. Як тільки це сталося, наш вхід проходить через ворота забуття. Ворота забуття — це, як правило, ще один рівень сигмоїдних функцій, через які протікає наш вхідний елемент, який вирішує, які значення скинути, помноживши різні частини нашої вхідної матриці на значення від 0 до 1. Нарешті, наше значення проходить через вихідні ворота, який є просто ще одним множенням на функцію активації, щоб визначити, яким має бути вихід нашої комірки LSTM.

При навчанні ставиться завдання мінімізувати цільову функцію помилки нейронної мережі, яка знаходиться методом найменших квадратів [17]:

$$E(w) = \frac{1}{2}\sum_{j=1}^{p}(y_i - d_i)^2, \qquad (3.1)$$

де p– число нейронів у вихідному шарі;

y_j – значення j-го виходу мережі;

d_j – цільове значення j-го виходу.

Навчання нейронної мережі виконується методом градієнтного спуску, тобто на кожній ітерації відбувається зміна ваги за формулою:

$$\Delta w_{ij} = -h\frac{\partial E}{\partial w_{ij}}, \qquad (3.2)$$

де h – параметр який визначає швидкість навчання.

Розпишемо більш детально складові мережі LSTM [13]:

Вентиль забуття(ворота забуття): рішення про те яку інформацію викинути зі стану клітиниприймається сигмовидним шаром:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{3.3}$$

Вхідний вентиль:вирішує яку нову інформацію ми збираємося зберігати в сані клітини. Спочатку сигмовидний шар, який називається «вхідним вентилем (воротами)», вирішує, які значення ми будемо оновлювати. Далі шар thanстворює вектор нових значень-кандидатів, $\tilde{C}$, що можна було б додати до стану.

$$i_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_i\right) \tag{3.4}$$

$$(c\tilde{}t)^{\sim} = \tanh(W_{\downarrow}c \cdot [[h_{\downarrow}(t-1), x_{\downarrow}t] + b_{\downarrow}c) \tag{3.5}$$

Комірка пам'яті: тепер треба оновити стан клітини C_{t-1}у новий стан комірки C_t. Ми множимо старий стан f_t, забувши речі, які ми вирішили забути раніше, потім додаємо$i_t * \tilde{C}$. Це нові значення-кандидати, масштабовані згідно до того, наскільки ми вирішили обновити значення кожного стану.

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \tag{3.6}$$

Вихідний вентиль: тут ми вирішуємо, що ми передаємо на вихід. Спочатку ми запускаємо сигмовидний шар, який приймає рішення, які частини стану клітини виводити. Потім передаємо стан клітини than (щоб отримати значення мід -1 та 1) і множимо його на вихід від сигмовидних воріт, що вивести лише ті частини, які ми вирішили.

$$o_{\downarrow}t = Sigmoid(W_{\downarrow}o \cdot [[h_{\downarrow}t1, x_{\downarrow}t] + b_{\downarrow}o) \tag{3.7}$$

$$h_t = o_t \cdot tanh(c_t) \quad (3.8)$$

Компонент “Класифікатор” призначений для визначення тональності твітів на тестовій вибірці і оцінки точності результатів. Основним складовим цього компоненту є вже навчена модель, якій ми передаємо тестовий набір даних для того щоб оцінити точність класифікації.

Існують різні метрики оцінки роботи моделей класифікацій. Базовою метрикою, яка використовується для оцінки моделі, часто є точність (accuracy), що описує кількість правильних передбачень у всіх прогнозах:

$$Accurasy = P/N, \quad (3.9)$$

де P – кількість документів за якими класифікатор прийняв вірне рішення;

N – розмір навчальної вибірки.

Однак даний показник сильно залежить від збалансованості даних. Так як наші дані є незбалансованими було вирішено використовувати показник F-міри, який є середнім гармонійним точності (precision) та повноти (recall) [18]. Precision використовується коли хибно-позитивний клас небажаний, recall, коли хибно-негативний. Розраховуємо за наступними формулами:

$$Precision = TP/(TP+FP), \quad (3.10)$$

$$Recall = TP/(TP+FN), \quad (3.11)$$

де TP – істино-позитивне рішення;

FP – хибно-позитивного рішення;

FN – хибно-негативне рішення.

F - міра–загальна міра точності моделі, вона об'єднує у собі інформацію стосовно точності та повноти обраного алгоритму. Розраховується за формулою:

$$F = (Precision * Recall) / (Precision + Recall) \quad (3.12)$$

Таким чином, F-міра, Recall та Precision є головними метриками при оцінці адекватності моделі.

Компонент “Візуалізація” відображає кругову діаграму для того щоб оцінити загальну думку, графік розподілу настроїв за датою, а також хмари слів з найбільш популярними словами для позитивних, негативних та нейтральних твітів. Хмари слів — одна з найкращих візуалізацій частоти слів у текстових документах. По суті, він створює зображення зі словами, які часто з’являються в текстовому документі, де найпоширеніші слова відображаються з великими розмірами шрифту, а менш часті слова з меншими.

3.1 Висновки до розділу

У цьому розділі було розроблено математичну модель системи визначення емоційного забарвлення публікацій що характеризують війну в Україні у соціальній мережі Твіттер. Було розроблено компонентну модель системи яка складається знаступних компонентів: «Сховище даних», «Попередня обробка», «Анотація даних», «Векторизація», «Навчання класифікатора», «Класифікатор», «Візуалізація». Було описано набір даних який ми використовуємо для навчання та тестування нашої моделі, влгоритм анотації даних за допомогою бібліотеки TextBlob, метод навчання нейронної мережі та метрики оцінки точності роботи нашої системи. Було визначено використовувати F-міра, Recall та Precision для оцінки точності системи.

4. ІМПЛЕМЕНТАЦІЯ СИСТЕМИ ВИЗНАЧЕННЯ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ПУБЛІКАЦІЙ ЩО ХАРАКТЕРИЗУЮТЬ ВІЙНУ В УКРАЇНІ В СОЦІАЛЬНІЙ МЕРЕЖІ ТВІТТЕР

4.1 Модель системи. Діаграма діяльності.

На основі компонентної моделі та математичних методів реалізації компонентів розроблено алгоритм роботи системи.

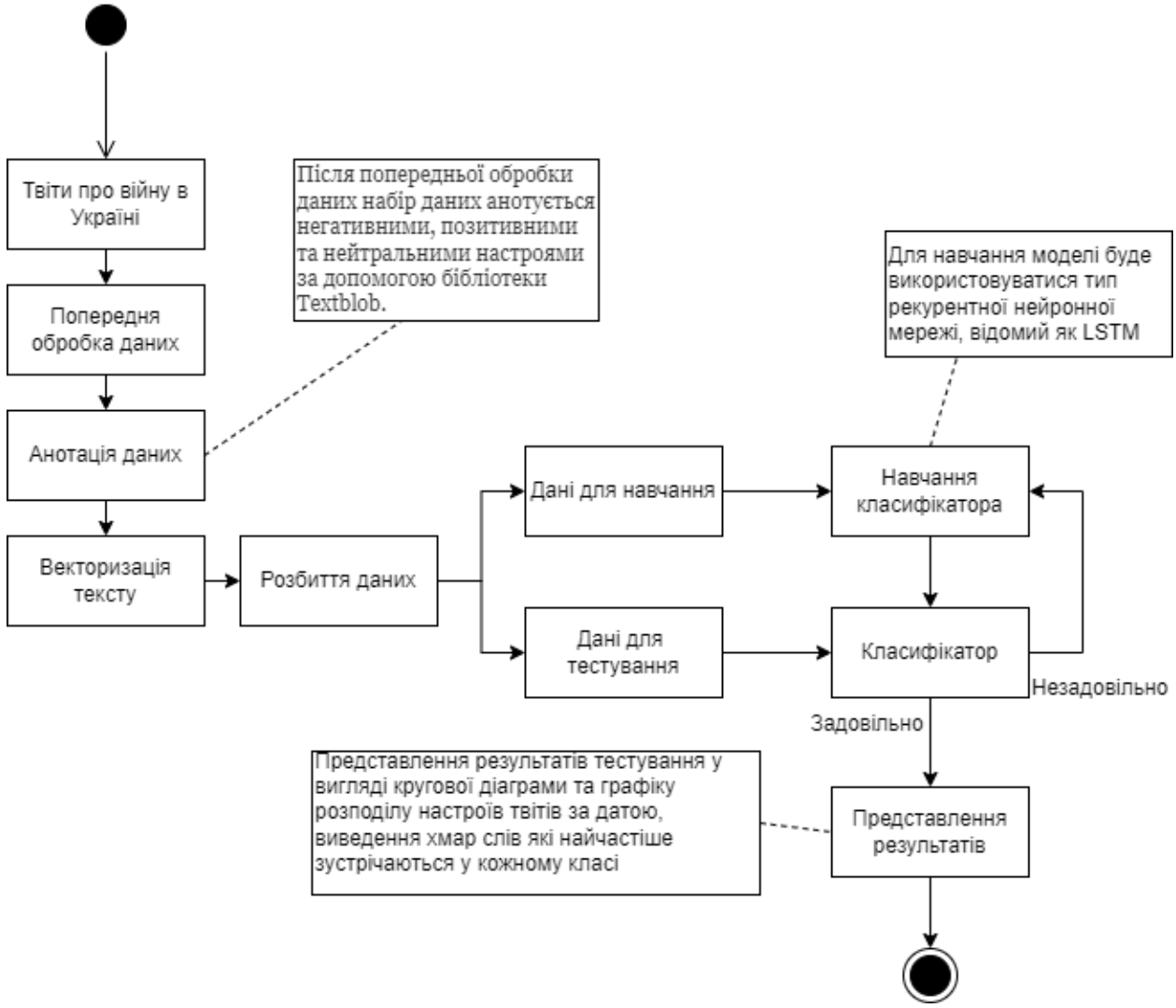


Рисунок 4.1 – Діаграма діяльності системи визначеня емоційного забарвлення твітів на тему війни в Україні

Для розробки системи аналізу емоційного забарвлення твітів що характеризують війну в Україні було використано рекурентні нейронні мережі, зокрема LSTM. Для навчання модель використовуватиме набір даних з соціальної мережі Твіттер який ми анотували використовуючи бібліотеку Textblob. Тестові дані містять твіти, які попередньо обробляються, а потім перетворюються у вектори. Для векторного представлення слівбуло обрано GloVe Embeddings. Ці вектори надаються як вхідні дані для класифікатора. Після того, як класифікація твітів виконана, вона зберігається у файлі csv. Ці дані можна візуалізувати за допомогою стовпчастих діаграм, кругових діаграм, таблиць, хмари слів. Системя реалізована з використанням мови програмування Python.

4.2 Попередня обробка навчальних даних

Попередня обробка даних для аналізу настроїв вимагає видалення всього, крім чистих текстових даних. Для цього ми видаляємо посилання, хештеги, символи, розділові знаки та перетворюємо текст у нижній регістр. Також було видалено всі твіти не англійською мовою та повторювані твіти (тобто ретвіти). На рисунку 4.2 можна побачити текст твітів до та після процедури очищення.

	content
189611	Biden's @PressSec Jen Psaki at briefing says p...
60692	If we wake up tomorrow morning and Putin has i...
55306	@MarkAClarkson LOL. The token response to Puti...
145267	@cliveatkinson @trussliz @DmytroKuleba Army of...
100650	Vlady Putin is a very stupid human. The coward...

	content
189611	biden s jen psaki at briefing says planning ...
60692	if we wake up tomorrow morning and putin has i...
55306	lol the token response to putin is comical ...
145267	army of the rf on its territory this is...
100650	vlady putin is a very stupid human the coward...

Рисунок 4.2

Далі набір даних анотується негативними, позитивними та нейтральними настроями за допомогою бібліотеки Textblob. На рисунку 4.3 зображено деякі твіти які було анотовано за допомогою цієї бібліотеки.

	content	sentiment	polarity	subjectivity	neg	neu	pos	compound
0	biden s jen psaki at briefing says planning for neptune strike exercise began in 2020 long before the current escalation of tensions from russia s aggressive posture towards ukraine she noted it s 1st time since cold war a us carrier group under nato command	negative	-0.216667	0.600000	0.208000	0.792000	0.000000	-0.827100
1	if we wake up tomorrow morning and putin has invaded ukraine and is on his way to kiyv i hope some ukrainian pilots don t bomb the border troops but fly right on to moskow to drop their bombs there it will have a more definitiv impact war is hell	negative	0.528571	0.645238	0.278000	0.690000	0.032000	-0.951300
2	lol the token response to putin is comical the germans will be begging russian gas before too long biden s sanctions and 800 troops is mere virtue signalling and in the words of paul keating is akin to throwing toothpicks at a growling bear	positive	0.150000	0.520000	0.000000	0.877000	0.123000	0.680800
3	army of the rf on its territory this is the right of any country kiev violates the minsk agreements the us and uk are pushing ukraine to war is this normal for you do you think killing people in donbass is normal	negative	0.195238	0.611905	0.224000	0.735000	0.041000	-0.888500
4	vlady putin is a very stupid human the coward wants no nato troops on his border but if he shit hole country invades ukraine he will get more troops on his border dumb ass	negative	-0.145833	0.500000	0.426000	0.574000	0.000000	-0.964000

Рисунок 4.3

Окрім полярності ми також маємо 4 ключі neg, neu, posta compound. Оцінка compoundрозраховується шляхом нормалізації інших 3 балів (neg, neu, pos) між -1 та 1 і визначає остаточний сентимент до якого належить твіт.

Після початкового процесу очищення чистий текст проходить через конвеєр NLP, що складається з токенізації, стеммінгу та видалення стоп-слів.При обробці вхідних даних використовуємо набір інструментів природної мови Python – «nltk». Python має вбудований метод рядка: string. split(), який розбиває будь-який заданий рядок на список на основі символу розділення. На рисунку 4.4 зображено функцію яку було написано для токенізації тексту.

```
#Appliyng tokenization
def tokenization(text):
    text = re.split('\W+', text)
    return text
```

Рисунок 4.4

Функції видалення стоп-слів та стеммінгу (приведення до спільного кореня) Зображено на рисунку 4.5. Більше функцій, наприклад видалення посилань та розділових знаків, можна подивитися у лістингу коду програми Додаток А.

```
#Removing stopwords
stopword = nltk.corpus.stopwords.words('english')
def remove_stopwords(text):
    text = [word for word in text if word not in stopword]
    return text

tw_list['nonstop'] = tw_list['tokenized'].apply(lambda x: remove_stopwords(x))
```

```
#Appliyng Stemmer
ps = nltk.PorterStemmer()

def stemming(text):
    text = [ps.stem(word) for word in text]
    return text

tw_list['stemmed'] = tw_list['nonstop'].apply(lambda x: stemming(x))
```

Рисунок 4.5

Перед поданням даних для навчання і тестування моделі їх необхідно представити увекторному вигляді. Ми використовуємо попередньо навчений GloVe як ініціалізацію шару вбудовування слів у нашій моделі аналізу настроїв. Ця модель перетворює слова у вектори.Для початку необхідно завантажити попередньо підготовлену модель GloVe [16]. Потім ми створюємо словник унікальних слів, зіставляючи кожен вектор GloVe з кожним словом. Наша мережа буде приймати дані, де кожне слово було трансформовано у вектор, який складається зі 150 значень.

4.3 Побудова та навчання нейронної мережі

Для навчання нейронної мережі, було використано бібліотеки numpy, tensorflow, keras. Для побудови моделі використовується клас Sequential, з бібліотеки keras. На рисунку 4.6 зображено код побудови нейронної мережі з відповідними шарами та параметрами.

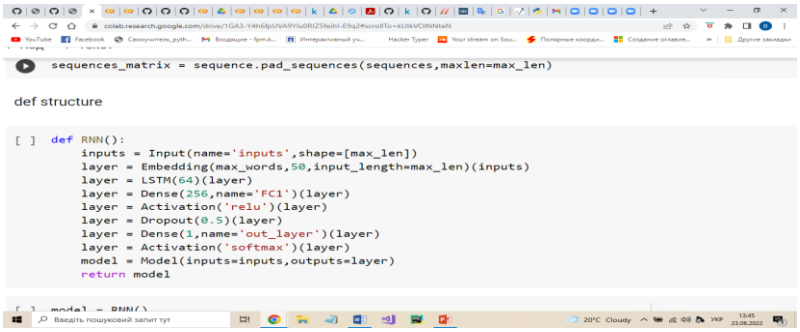


Рисунок 4.6

Запропонована архітектура нейронної мережі має наступний набір шарів:

- Шар Embedding – вхідний шар нейронної мережі, що використовується як векторний простір із попередньо навченим вбудовуванням GloVe. Генерує вектор вбудовування для кожної вхідної послідовності.
- Шар LSTM – рекурентний шар нейронної мережі. Цей рівень додає всі вентилі введення, виведення та забуття, необхідні для реалізації шару вузлів LSTM. Далі йде вихідний шар Dence з функцією активації ReLu.
- Шар Dropout – потрібний, щоб уникнути перенавчання нейронної мережі. Як параметр подається значення 0.5, що допомагає нашій моделі запобігти перенавчанню шляхом випадкового вимикання вузлів у мережі. (50% виходу шару LSTM обнуляється у випадкових місцях на кожній навчальній ітерації).
- Шар Dense – вихідний шар з функцією активації softmaxі нейронами що дорівнюють кількості унікальних категорій (у нашому випадку 3). Використовується для остаточної класифікації.

Оптимізатором для навчання є Адам. І так як це проблема багатокласової класифікації використовуваною функцією втрат є «категорична кроссентропія». Розмір партії обрали 64. Кількість епох під час навчання моделі було встановлено на 10.

Для верифікації моделі аналізу настроїв було застосовано 5-кратну перехресну валідацію. Під час виконання перехресної перевірки данні розбиваються на N частин, тренування проводиться на N-1 частинах, а остання використовується для

тестування моделі. Це робиться для того щоб перевірити працездатність алгоритма на даних які він раніше не бачив (незалежних даних).

Нарисунку 4.7 наведені метрики оцінювання (тобтоточність, повнота та F1) для нашоїмоделі класифікації. Найкращі значення розпізнавання на тестовому наборі даних були отримані для негативного сентименту і склали precision = 89%, recall = 89 %, F1 = 89%. Середня оцінка точності моделі становить 87%. Таким чином можна стверджувати, що розроблене математичне та програмне забезпечення системи визначення емоційного забарвлення твітів що характеризують війну в Україні може відображати результати з хорошою точністю.

	precision	recall	f1-score
0	0.89	0.89	0.89
1	0.87	0.88	0.88
2	0.86	0.85	0.85
accuracy			0.88
macro avg	0.87	0.87	0.87
weighted avg	0.88	0.88	0.88

Рисунок 4.7

4.4 Засоби візуалізації результатів

Для візуалізації результатів ми використовуємо кругову діаграму для того щоб швидко оцінити загальну думку людей. Також було використано хмару слів для віртуалізації даних для представлення текстових даних. Хмара слів — це техніка, яка представляє текст різного розміру залежно від частоти чи важливості. Ми завантажили та встановили три необхідні модулі: matplotlib, pandas та wordcloud. Також буде наведено загальний графік розподілу настроїв за датою та коментарі з приводу цих оцінок.

4.5 Верифікація і валідація системи

Стандарт ICO 9000:2000 визначає ці терміни таким чином:

«**Верифікація (**verification — перевірка,**)** — підтвердження на основінадання об’єктивних свідчень того, що *встановлені вимоги були виконані*».

«**Валідація** (validation — надання законної сили)— підтвердження наоснові надання об’єктивних свідчень того, що встановлені вимоги, *призначені для конкретного використання і застосування, виконані*».

Таким чином можна зробити **висновок:**

верифікація — проводиться методом порівняння значень конкретних характеристик продукції, товарів і послуг з заданими в документації значеннями цих характеристик для формування висновку про відповідність продукції, товарів і послуг заданим в документації вимогам;

валідація — проводиться при необхідності і виконується методом аналізу заданих умов застосування/експлуатації і оцінки відповідності значеньхарактеристик продукції, товарів і послуг цим умовам, з наступним висновкомпро можливість застосування продукції, товарів і послуг для встановлених умов застосування/експлуатації [19].

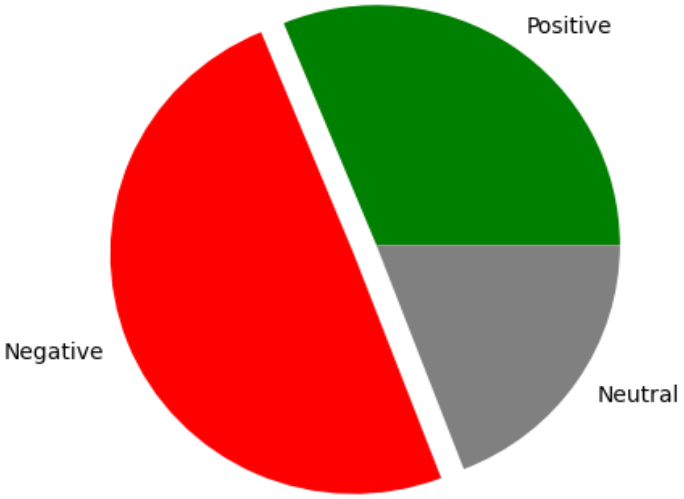


Рисунок 4.4 Кругова діаграма настроїв твітів

```
TotalPositiveTweets% : 31.22
TotalNegativeTweets% : 49.62
TotalNeutralTweets% : 19.16
```

З аналізу настроїв ми можемо чітко помітити, що більшість людей негативно ставляться до війни. Вони проти цього. Тепер подивимося на хмари слів які були отримані у результаті класифікації.

Рисунок 4.6 Слова які найчастіше зустрічаються у нейтральних твітах

Рисунок 4.7 Слова які найчастіше зустрічаються у негативних твітах

Деякі ключові слова у відповідних класах включають:

Позитивні: peace, support, independent

Нейтральні: переважно власні іменники

Негативні: crisis, war, invasion

Варто зазначити, що це доволі загальне представлення, оскільки як видно багато слів зустрічаються як у негативному так і позитивному і нейтральних класах. Для більш детального представлення пропонується у майбутньому використати такий підхід як тематичне моделювання, який допомагає виділити важливі теми з даних. Він допоможе визначити важливість теми за частотою її появи та її зв’язком з іншими темами і буде більш інформативним, оскільки ми зможемо побачити ставлення людей до конкретних тем.

Далі подивимося як змінювалися настрої у Твіттері за період з 1 січня по 6 березня 2022 року.

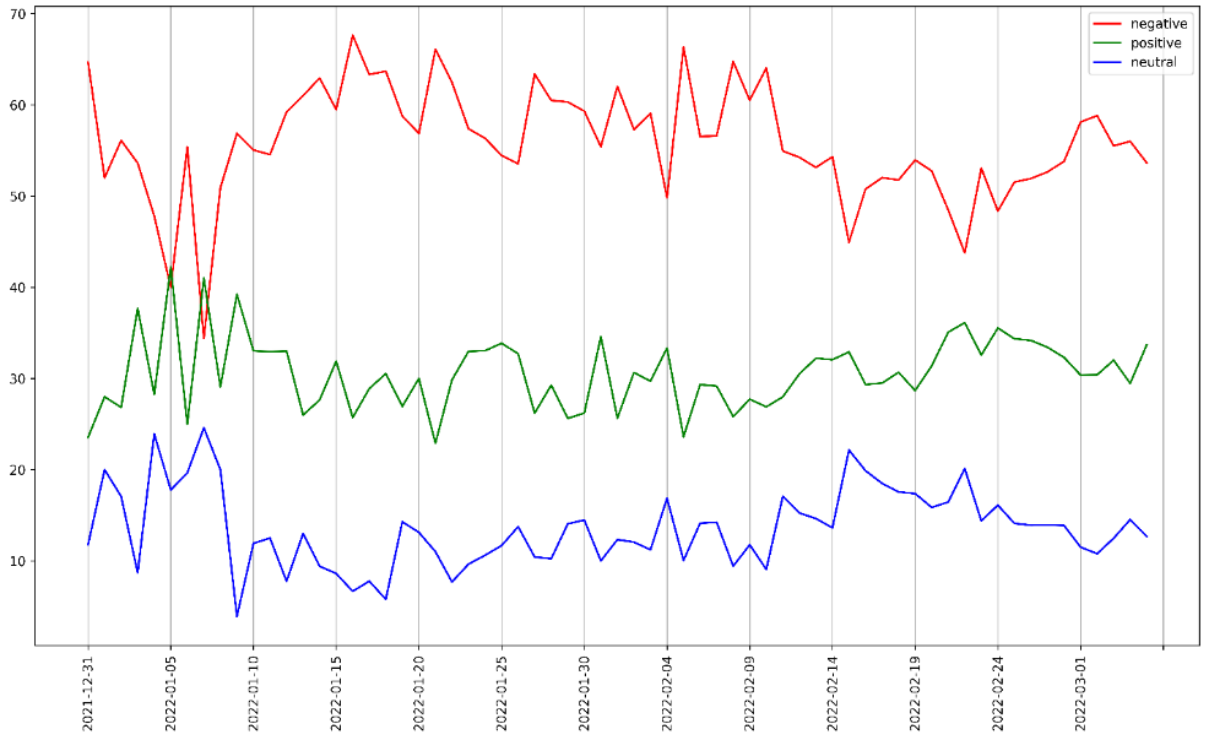


Рисунок 4.8 Графік розподілу настроїв твітів за датою

По цьому графіку ми можемо зіставляти основні події які відбувалися у цей період для того, щоб зрозуміти реакцію громадськості на них. На графіку з рисунка 4.8видно, як твіти з негативними настроями завжди мали більшу кількість, ніж позитивні або нейтральні твіти. 24 лютого 2022 року, коли Путін оголошує війну Україні, на цьому етапі збільшується кількість твітів із негативними настроями і позитивні настрої твіти зменшилися, отже, з цього подання можна спостерігати збільшення полярності. До цього, коли 17 січня з’являється новина про прибуття російських військ на кордони України, на цьому етапі також можна спостерігати сплеск негативних настроїв у твітах.

У таблиці 1.2 наведено приклад деяких основних подій в ці періоди.

Таблиця 1.2 – Деякі основні події в ці періоди

Дата	Новина
17 січня2022	Російські війська почали прибувати до союзника Росії Білорусі
24 січня 2022	НАТО перевело війська в режим очікування
10 лютого 2022	Росія та Білорусь почали 10-денні військові маневри
28 лютого 2022	Упродовж ночі навколо Маріуполя точилися бої

4.6 Висновок до розділу

У цьому розділі було розроблено архітектуру та програмне забезпечення системи визначення емоційного забарвлення публікацій що характеризують війну в Україні у соціальній мережі Твіттер. Описано алгоритми процеси попередньої обробки та анотації даних, векторне представлення, процес побудови та навчання нейронної мережі LSTM. Проведено валідацію і верифікацію системи і визначено точність системи на тестовій вибірці, яка склала 87%. Також було описано використані інструменти та бібліотеки для реалізації програмного забезпечення.

ВИСНОВКИ

1. Розглянуто основні підходи до сентимент аналізу тексту. Було досліджено та встановлено такі методи векторизації тексту: Bag-of-Words, TF-IDF, Word2Vec таGloVe. Після порівняння їх характеристик будо обрано модель GloVe для векторного представлення слів. Проведено аналіз методів машинного навчання для для класифікації тексту: наївний Баєсів класифікатор, метод опорних векторів та нейронні мережі LSTM. На основі порівняння характеристик цих методів було встановлено, що найкращим вибором для побудови системи буде нейронна мережа типу LSTM. Також було проведено огляд існуючих програмних продуктів та порівняно іх характеристики.

2. Було розроблено математичну модель системи визначення емоційного забарвлення публікацій що характеризують війну в Україні у соціальній мережі Твіттер. Розроблено компонентну модель системи яка складається з таких компонентів: «Сховище даних», «Попередня обробка», «Анотація даних», «Векторизація», «Навчання класифікатора», «Класифікатор», «Візуалізація». Було описано набір даних який ми використовуємо для навчання та тестування нашої моделі, алгоритм анотації даних за допомогою бібліотеки TextBlob, метод навчання нейронної мережі та метрики оцінки точності роботи нашої системи. Було визначено використовувати F-міра, Recall та Precision для оцінки точності системи.

3. Розроблено архітектуру та програмне забезпечення системи визначення емоційного забарвлення публікацій що характеризують війну в Україні у соціальній мережі Твіттер. Описано алгоритми процесів попередньої обробки та анотації даних, векторне представлення, процес побудови та навчання нейронної мережі LSTM. Проведено валідацію і верифікацію системи і визначено що найкращі значення розпізнавання на тестовому наборі даних були отримані для негативного сентименту і склали precision = 89%, recall = 89 %, F1 = 89%. Середня ж оцінка точності моделі становить 87%.

4. Дані соціальних мереж можна використовувати як інструмент для оцінки настроїв суспільства з приводу будь-яких тем. Перспективи подальших досліджень полягають у знаходженні розподілу настроїв людей з різних каїн, за допомогою даних про місцезнаходження, моделюванні тем до яких висловлються найбільш позитивно або найбільш негативно, розширенні метрик оцінки тональності публікацій у соціальних мережах до класифікацій емоцій (злість, сум, відраза, спокій, радість і т.д.), а також створенні теплової карти проросійських та проукраїнських настроїв.

ПЕРЕЛІК ПОСИЛАНЬ

1. Lexicon-basedapproach [Електронний ресурс] - Режим доступу: https://www.sciencedirect.com/topics/computer-science/lexicon-based-approach
2. H.S.Hota, Dinesh K. Sharma, Nilesh Verma.14 Lexicon-based sentiment analysis using Twitter data: a case of COVID-19 outbreak in India and abroad [Електронний ресурс] – Режим доступу: https://doi.org/10.1016/B978-0-12-824536-1.00015-0
3. OmarGawade [Електронний ресурс] – Режим доступу: https://miro.medium.com/max/1400/0*rVCzUQTsXNfJKZlV.png
4. Luthfi Ramadhan. TF-IDFSimplified. [Електронний ресурс] – Режим доступу:https://towardsdatascience.com/tf-idf-simplified-aba19d5f5530
5. E. Lee, F. Rustam, I. Ashraf, P. B. Washington, M. Narra and R. Shafique, "Inquest of Current Situation in Afghanistan Under Taliban Rule Using Sentiment Analysis and Volume Analysis," in IEEE Access, vol. 10, pp. 10333-10348, 2022, doi: 10.1109/ACCESS.2022.3144659
6. Purva Huilgol .Quick Introduction to Bag-of-Words (BoW) and TF-IDF for Creating Features from Text. [Електронний ресурс] – Режим доступу: https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/
7. Che-Wen Chen, Shih-Pang Tseng, Ta-Wen Kuan, Jhing-Fa Wang “Outpatient Text Classification Using Attention-Based Bidirectional LSTM for Robot-Assisted Servicing in Hospital” vol. 19, 2020, [Електронний ресурс] – Режим доступу: https://www.researchgate.net/publication/339343963_Outpatient_Text_Classification_Using_Attention-Based_Bidirectional_LSTM_for_Robot-Assisted_Servicing_in_Hospital
8. Yugesh Verma, “Word2Vec vs GloVe – A Comparative Guide to Word Embedding Techniques”, 2021, [Електронний ресурс] – Режим доступу:

https://analyticsindiamag.com/word2vec-vs-glove-a-comparative-guide-to-word-embedding-techniques/

9. machine learning interview.com [Електронний ресурс] – Режим доступу:https://machinelearninginterview.com/wp-content/uploads/2019/02/CBOW_eta_Skipgram.png
10. Timo Böhm, “The general ideas of word embeddings”, 2018, [Електронний ресурс] – Режим доступу: https://towardsdatascience.com/the-three-main-branches-of-word-embeddings-7b90fa36dfb9
11. Koushiki Dasgupta Chaudhuri “Building Naive Bayes Classifier from Scratch to Perform Sentiment Analysis”, 2022, [Електронний ресурс] – Режим доступу: https://www.analyticsvidhya.com/blog/2022/03/building-naive-bayes-classifier-from-scratch-to-perform-sentiment-analysis/
12. Rohith Gandhi, “Support Vector Machine - Introduction to Machine Learning Algorithms”, 2018, [Електронний ресурс] – Режим доступу: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
13. colah.github.io [Електронний ресурс]: Understanding LSTM networks. Режим доступу:http://colah.github.io/posts/2015-08-Understanding-LSTMs/
14. github.com [Електронний ресурс] – Режим доступу: https://github.com/rasbt/stat479-deep-learning-ss19/blob/master/L14_intro-rnn/L14_intro-rnn-part2_slides.pdf
15. kaggle.com [Електронний ресурс] – Режим доступу: https://www.kaggle.com/datasets/foklacu/ukraine-war-tweets-dataset-65-days
16. nlp.stanford.edu [Електронний ресурс]: GloVe: Global Vectors for Word Representation. Режим доступу: https://nlp.stanford.edu/projects/glove/
17. “LSTM Gradients”, 2020, [Електронний ресурс] – Режим доступу: https://towardsdatascience.com/lstm-gradients-b3996e6a0296

18. Teemu Kanstrén, “A look at precision, recall, and F1-score’, 2020, [Електронний ресурс] – Режим доступу: https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec

19. Системи Data Science. Методичні вказівки та рекомендації для виконання лабораторних робіт [Електронний ресурс] / КПІ ім. Ігоря Сікорського ; уклад. П. П. Маслянко. – Електронні текстові дані (1 файл: 1.37 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 38 с. – Назва з екрана. https://ela.kpi.ua/handle/123456789/49981

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
# Import Libraries
from textblob import TextBlob
import sys
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import os
import nltk
import re
import string
import seaborn as sns
from PIL import Image
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.stem import SnowballStemmer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from sklearn.feature_extraction.text import CountVectorizer
```

Data loading

```
# for mounting the google drive
from google.colab import drive
drive.mount('/content/drive')

# The dataset consists of tweets from total 8 hashtags and present in separated csv files. All those csv files are loaded.
tweets=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Russia_invade.csv")
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Russian_border_Ukrai
ne.csv"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Russian_troops.csv"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/StandWithUkraine.csv
"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Ukraine_border.csv"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Ukraine_nato.csv"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Ukraine_troops.csv"))
tweets=tweets.append(pd.read_csv("/content/drive/MyDrive/Colab Notebooks/sentiment analysis/Ukraine_war.csv"))

#checking the dataframe and total number of tweets in the dataset
tweets.shape

tweets.keys() #checking all the keys in dataset

tweets["date"] #checking the date format in dataset
```

Data preprocessing

```
tweets['date'] =tweets.date.str.slice(0, 10) #slicing the date , and removing the times
tweets.drop_duplicates(inplace = True) #remove duplicates
tweets.shape #after removing duplicates
```

```
print(tweets['date'].unique()) #checking all the unique dates in the dataset
len(tweets['date'].unique()) #checking how many days of data in present in the dataset
print(tweets["lang"].unique()) #checking how many unique language tweets are present in the dataset
len(tweets["lang"].unique()) #total number of language present
print(tweets.shape) #before removing the non-english tweets
tweets = tweets[tweets['lang'] == 'en'] #removing all the tweets expect the non-english tweets
print("After removing non-english Tweets")
print(tweets.shape) #only the number of english tweets
tweets=tweets.sample(frac =.02) #we are doing random sampling of the dataset , here we are taking 2% of the entire data
set
tweets.shape #checking the number of tweets present in the sample dataset
tweets[["content"]].head(5)

#Removing RT, Punctuation etc
remove_rt = lambda x: re.sub('RT @\w+: ',"",x)
rt = lambda x: re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)","",x)
tweets["content"] = tweets.content.map(remove_rt).map(rt)
tweets["content"] = tweets.content.str.lower()
tweets.head(10)
tweets[["content"]].head(5)

nltk.download('vader_lexicon')
#Calculating Negative, Positive, Neutral and Compound values

tweets[['polarity', 'subjectivity']] = tweets['content'].apply(lambda Text: pd.Series(TextBlob(Text).sentiment))
for index, row in tweets['content'].iteritems():
    score = SentimentIntensityAnalyzer().polarity_scores(row)
    neg = score['neg']
    neu = score['neu']
    pos = score['pos']
    comp = score['compound']
    if neg > pos:
        tweets.loc[index, 'sentiment'] = "negative"
    elif pos > neg:
        tweets.loc[index, 'sentiment'] = "positive"
    else:
        tweets.loc[index, 'sentiment'] = "neutral"
    tweets.loc[index, 'neg'] = neg
    tweets.loc[index, 'neu'] = neu
    tweets.loc[index, 'pos'] = pos
    tweets.loc[index, 'compound'] = comp

tweets.head(10)
tweets.to_csv("tweets_clean.csv") #saving the CSV file with cleaned tweets
#tweets=pd.read_csv("tweets_clean.csv")
tweets.keys()
def color_yellow(compound):
  background = 'yellow'
  return 'background: %s' % background
```

```
tweets[["content","sentiment","polarity", "subjectivity","neg","neu","pos", "compound"]].head(5).style.applymap(color_
yellow, subset=['compound'])
tw_list=pd.DataFrame(tweets['content'])

#Removing Punctuation
def remove_punct(text):
    text  = "".join([char for char in text if char not in string.punctuation])
    text = re.sub('[0-9]+', '', text)
    return text

tw_list['punct'] = tw_list['content'].apply(lambda x: remove_punct(x))

#Appliyng tokenization
def tokenization(text):
    text = re.split('\W+', text)
    return text

tw_list['tokenized'] = tw_list['punct'].apply(lambda x: tokenization(x.lower()))

nltk.download('stopwords')
#Removing stopwords
stopword = nltk.corpus.stopwords.words('english')
def remove_stopwords(text):
    text = [word for word in text if word not in stopword]
    return text

tw_list['nonstop'] = tw_list['tokenized'].apply(lambda x: remove_stopwords(x))
#Appliyng Stemmer
ps = nltk.PorterStemmer()

def stemming(text):
    text = [ps.stem(word) for word in text]
    return text

tw_list['stemmed'] = tw_list['nonstop'].apply(lambda x: stemming(x))
#Cleaning Text
def clean_text(text):
    text_lc = "".join([word.lower() for word in text if word not in string.punctuation]) # remove puntuation
    text_rc = re.sub('[0-9]+', '', text_lc)
    tokens = re.split('\W+', text_rc)    # tokenization
    text = [ps.stem(word) for word in tokens if word not in stopword]  # remove stopwords and stemming
    return text

#GloVe embeddings
# Download GloVe
!wget http://nlp.stanford.edu/data/glove.6B.zip
!unzip glove.6B.zip
# Select embeddings
GLOVE_EMB = 'glove.6B.300d.txt'
```

```
EMBEDDING_DIM = 300
# Map words to known embeddings by parsing the data of pre-trained embeddings
embeddings_index = {}

f = open(GLOVE_EMB)
for line in f:
  values = line.split()
  word = value = values[0]
  coefs = np.asarray(values[1:], dtype='float32')
  embeddings_index[word] = coefs
f.close()

print('Found %s word vectors.' %len(embeddings_index))

# Create embedding matrix and space which can be loaded into the "embedding layer" of our LSTM model for use in def
ining the context of each token
embedding_matrix = np.zeros((VOCAB_SIZE, EMBEDDING_DIM))
for word, i in word_index.items():
  embedding_vector = embeddings_index.get(word)
  if embedding_vector is not None:
    embedding_matrix[i] = embedding_vector
count_vect_df = pd.DataFrame(countVector.toarray(), columns=countVectorizer.get_feature_names())
count_vect_df.head()
```

LSTM

```
X = total.clean_text
Y = total.TextBlob_Analysis
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)
max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(X_train)
sequences = tok.texts_to_sequences(X_train) #text to sequences!
sequences_matrix = sequence.pad_sequences(sequences,maxlen=max_len)
def RNN():
    inputs = Input(name='inputs',shape=[max_len])
    layer = Embedding(VOCAB_SIZE, EMBEDDING_DIM, weights=[embedding_matrix], input_length=MAX_SEQU
ENCE_LENGTH, trainable=False)(inputs)
    layer = LSTM(64)(layer)
    layer = Dense(256,name='FC1')(layer)
    layer = Activation('relu')(layer)
    layer = Dropout(0.5)(layer)
    layer = Dense(1,name='out_layer')(layer)
    layer = Activation('softmax')(layer)
    model = Model(inputs=inputs,outputs=layer)
    return model
```

```
model = RNN()
model.summary()
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
model.fit(X_train_final,y_train, epochs=10,validation_data=(X_test_final,y_test))
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = sequence.pad_sequences(test_sequences,maxlen=max_len)
ccr = model.evaluate(test_sequences_matrix,Y_test)
# Evaluate
model.evaluate(X_test_final,y_test)
# Predict test
lstm_pred = np.round(model.predict(X_test_final))
# Precision , Recall , F1-score
cr = classification_report(y_test,lstm_pred)
print(cr);
```

Visualisation

```
#pie chart
total_pos=len(tweets.loc[tweets['sentiment'] == "positive"])
total_neg=len(tweets.loc[tweets['sentiment'] == "negative"])
total_neu=len(tweets.loc[tweets['sentiment'] == "neutral"])
total_tweets=len(tweets)
print("Total Positive Tweets % : {:.2f}".format( (total_pos/total_tweets)*100))
print("Total Negative Tweets % : {:.2f}".format((total_neg/total_tweets)*100))
print("Total Neutral Tweets % : {:.2f}".format((total_neu/total_tweets)*100))

mylabels = ["Positive", "Negative", "Neutral"]
mycolors = ["Green", "Red", "Grey"]

plt.figure(figsize=(8, 5),
        dpi = 100) # Push new figure on stack
myexplode = [0, 0.1, 0]
plt.pie([total_pos,total_neg,total_neu],colors=mycolors,labels=mylabels,explode = myexplode)
plt.show()

#graph
pos_list=[]
neg_list=[]
neu_list=[]
for i in tweets["date"].unique():
  temp=tweets[tweets["date"]==i]
  positive_temp=temp[temp["sentiment"]=="positive"]
  negative_temp=temp[temp["sentiment"]=="negative"]
  neutral_temp=temp[temp["sentiment"]=="neutral"]
  pos_list.append(((positive_temp.shape[0]/temp.shape[0])*100,i))
  neg_list.append(((negative_temp.shape[0]/temp.shape[0])*100,i))
  neu_list.append(((neutral_temp.shape[0]/temp.shape[0])*100,i))

neu_list=sorted(neu_list, key = lambda x: x[1])
pos_list=sorted(pos_list, key = lambda x: x[1])
```

```
neg_list=sorted(neg_list, key = lambda x: x[1])

x_cord_neg=[]
y_cord_neg=[]

x_cord_pos=[]
y_cord_pos=[]

x_cord_neu=[]
y_cord_neu=[]

for i in neg_list:
  x_cord_neg.append(i[0])
  y_cord_neg.append(i[1])

for i in pos_list:
  x_cord_pos.append(i[0])
  y_cord_pos.append(i[1])

for i in neu_list:
  x_cord_neu.append(i[0])
  y_cord_neu.append(i[1])

plt.figure(figsize=(10, 5))
plt.plot(y_cord_neg,x_cord_neg,label="negative",color="red")
plt.plot(y_cord_pos,x_cord_pos,label="positive",color="green")
plt.plot(y_cord_neu,x_cord_neu,label="neutral",color="gray")
plt.xticks(np.arange(0, len(tweets["date"].unique()) + 1, 5))
plt.xticks(rotation=90)
plt.grid(axis = 'x')
plt.legend()

from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
text = open('tweets_clean.csv', mode='r', encoding='utf-8').read()
wc = WordCloud(background_color='white',
               stopwords=stopwords,
               height = 600,
               width = 400)
wc.generate(text)
wc.to_file('wc.png')
```

ДОДАТОК Б ІЛЮСТРАТИВНИЙ МАТЕРІАЛ

Рисунок Б.1 — Слайд 1

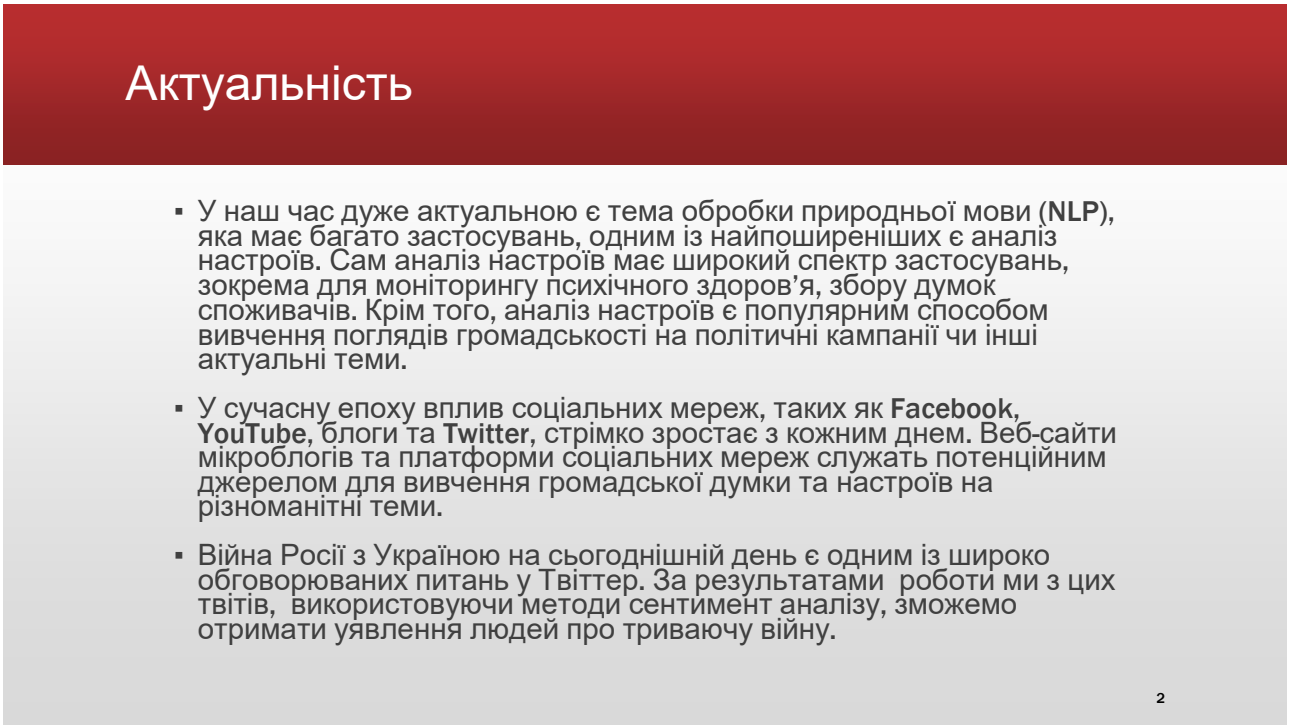


Рисунок Б.2 — Слайд 2

Мета роботи

- Метою роботи є розробка математичного і програмного забезпечення системи визначення емоційного забарвлення публікацій що характеризують війну в Україні в соціальній мережі Твіттер.
- Об’єктом дослідження є публікації в соціальній мережі Твіттер, методи обробки неструктурованих текстів, алгоритми та моделі векторного представлення слів, методи класифікації тексту.
- Предметом дослідження є математичне та програмне забезпечення системи для визначення емоційного забарвлення публікацій що характеризують війну в Україні на основі методів машинного навчання.

3

Рисунок Б.3 — Слайд 3

Поставлені завдання

Для досягнення мети роботи потрібно вирішити такі завдання:

- Провести огляд існуючих підходів до сентимент аналізу тексту, методів векторного представлення слів та методів класифікації тексту;
- Провести порівняльний аналіз методів векторного представлення слів та методів класифікації тексту;
- Розробити математичну модель системи та математичні методи реалізації її компонентів;
- Розробити програмне забезпечення системи;
- Провести верифікацію та валідацію системи на даних із соціальної мережі Твіттер.

4

Рисунок Б.4 — Слайд 4

Огляд моделей векторного представлення слів

Bag of Words

- Спочатку токенізувати кожне речення на слова та об'єднати всі слова, щоб створити вектори ознак.
- Кожен вектор ознак представляє кількість цього слова в конкретному реченні.
- напр. «The car is beautiful» — це речення створить 4 елементи зі стовпцями «The», «car», «is», «beauty», і кожна з них матиме значення 1 (кількість конкретного слова у реченні)

TF-IDF

- TF-IDF(t,d,D) = tf(t,d) *idf(t,D)
- $\mathrm{tf}(t, d) = \frac{f_d(t)}{\sum_{w \in d} f_w(t)}$

$$= \frac{frequency\ of\ the\ word\ in\ the\ document}{total\ number\ of\ words\ in\ the\ document}$$

- $\mathrm{idf}(t, D) = \ln(\frac{|D|}{|\{d \in D, t \in d\}|})$

$$= \ln(\frac{Total\ number\ of\ documents}{total\ number\ of\ documents\ containing\ the\ word\ t})$$

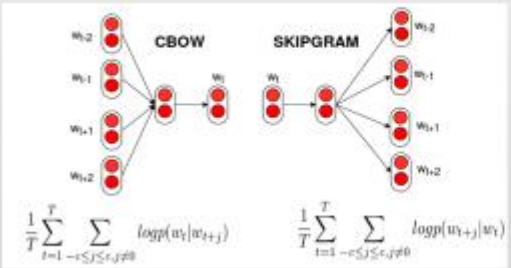
5

Рисунок Б.5 — Слайд 5

Огляд моделей векторного представлення слів

Word2Vec

- Представлення слів за допомогою Word2Vec можна здійснити двома основними методами: Skip-gram та Continious Bag of Words (CBOW)



GloVe - алгоритм навчання без нагляду для отримання векторних представлень слів.

- Створює матрицю, яка складається з інформації про спільне зустрічання слів

$$F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}$$

де P_{ik} – ймовірність що слово k зустрілося в контексті слова i;

P_{jk} - ймовірність що слово k зустрілося в контексті слова j.

6

Рисунок Б.6 — Слайд 6

Огляд методів класифікації тексту

- **Наївний баєсівський класифікатор**

Умовна імовірнісна модель: заданий екземпляр проблеми, що підлягає класифікації, представлений вектором x = {x1, x2, … , xn}, що представляє деякі n ознак (незалежних змінних), вона присвоює цьому екземпляру ймовірності

$$P(C_k|x_1, x_2, \dots, x_n) \qquad P(C_i, x) = \frac{P(x|C_i)P(C_i)}{P(x)}$$

- **Метод опорних векторів**

Базується на побудові гіперплощини, яка розділяє об'єкти вибірки найбільш оптимальним шляхом. Основна ідея в тому, що значення середньої похибки класифікатора оберненно пропорційне значенню відстанні між гіперплощиною та об'єктами розділенних класів.

- **Штучні нейронні мережі**

Рекурентні нейронні мережі, LSTM.

7

Рисунок Б.7 — Слайд 7

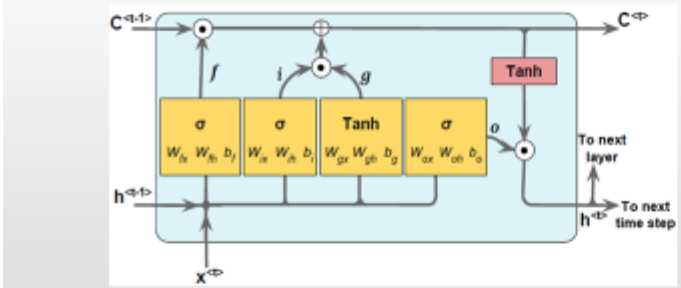
Порівняння методів класифікації тексту

	Наївний Баєсів класифікатор	Метод опорних векторів	Нейронна мережа LSTM
Швидкість навчання	Швидко	Швидко	Ресурсомісткий і вимагає часу
Інтерпретація результатів	Результати цілком зрозумілі	Результати цілком зрозумілі	Важче інтерпретувати результати
Переваги	• Швидкість обчислення • Можливість фіксувати контекст	• Добре працює коли є чіткий поділ між класами • Ефективний для пам'яті	• Має потенціал отримання найбільш точних результатів • Може вивчати важливі ознаки • Ефективна у запам'ятовуванні важливої інформації • Фіксує контекст
Недоліки	• Покладається лише на повний і репрезентативний набір даних • Сильне припущення про незалежність у моделі	• Погано працює на великих наборах даних • Не дуже добре працює, коли набір даних має більше шуму	• Найдорожчі навчання та експлуатація

8

Рисунок Б.8 — Слайд 8

LSTM (довготривала короткочасна пам'ять)



LSTM є розширенням для RNN, оскільки воно дозволяє RNN запам'ятовувати введені дані протягом тривалого періоду часу

- Вентиль забування

$$f_t = \sigma(W_f \cdot [h_{t1}, x_t] + b_f)$$

- Вентиль входу

$$i_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{c}_t = \tanh(W_c \cdot [[h_{t-1}, x_t] + b_c)$$

- Комірка пам'яті

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$$

- Вихідний вентиль

$$o_t = Sigmoid(W_o \cdot [[h_{t1}, x_t] + b_o)$$
$$h_t = o_t \cdot tahn(c_t)$$

9

Рисунок Б.9 — Слайд 9

Модель системи. Діаграма компонентів

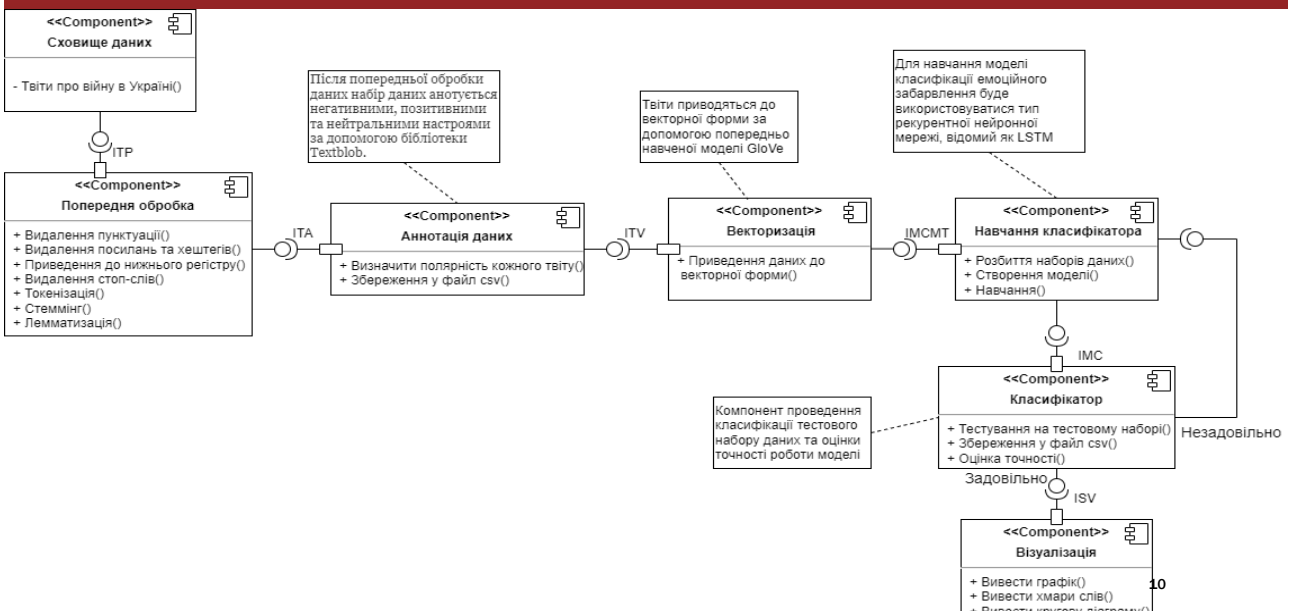


Рисунок Б.10 — Слайд 10

Модель системи. Діаграма діяльності

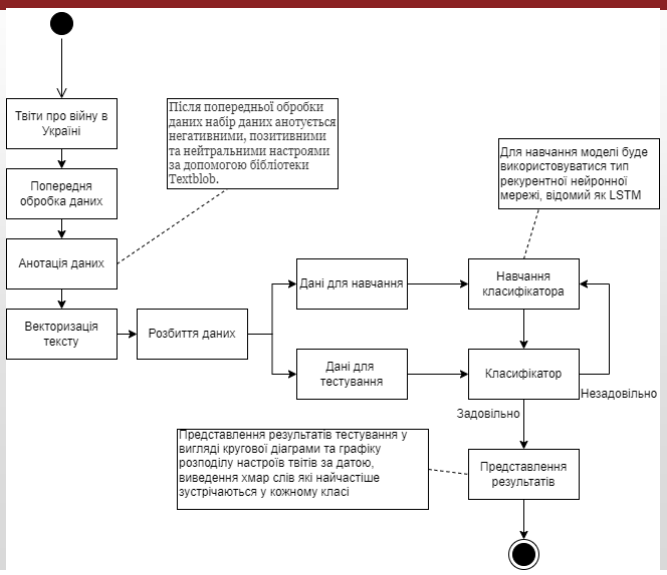


Рисунок Б.11 — Слайд 11

Опис даних і їх первинна обробка

1. Набір даних було отримано з сайту kaggle.com «Russia vs Ukraine Tweets Dataset». Він містить твіти зібрані за 65 днів з 1 січня 2022 року по 6 березня 2022 року. Ця шкала включає період перед і після вторгнення. Всього присутні 1316605 рядків, і кожен рядок містить інформацію про один твіт. Інша інформація включає дату та час, мову, хештеги, місце, кількість вподобань, кількість ретвітів, посилання тощо.
2. Після процедури feature enginiring отримуємо 1204118.
3. Далі розглядаються 2% набору даних. Вибірка формується таким чином, що кожен рядок має однакове зважування ймовірності, і можна було відібрати рядок лише один раз, і він не буде доступний для вибору знову. У вибірковому наборі даних загалом присутні 24084 твіти.
4. Потім розділові знаки, хештеги та посилання видаляються, а всі тексти приводяться у нижній регістр.
5. Далі дані анотуються за допомогою бібліотеки Textblob та подаються для навчання та тестування моделі аналізу настроїв. Співвідношення навчальних і тестових даних становить 80:20, де 80% даних використовується для навчання, а 20% – для тестування.

12

Рисунок Б.12 — Слайд 12

Навчання класифікатора

Для навчання класифікатора, було використано бібліотеки numpy, tensorflow, keras. Для побудови моделі використовується клас Sequential, з бібліотеки keras.

Запропонована архітектура нейронної мережі має наступний набір шарів:

- Шар Embedding - використовується як векторний простір із попередньо навченим вбудовуванням GloVe;
- Шар LSTM - додає всі вентилі введення, виведення та забуття, необхідні для реалізації шару вузлів LSTM;
- Шар Dropout – як параметр подається значення 0.5, що допомагає нашій моделі запобігти перенавчанню шляхом випадкового вимикання вузлів у мережі. (50% виходу шару LSTM обнуляється у випадкових місцях на кожній навчальній ітерації).
- Шар Dense – вихідний шар з функцією активації softmax і нейронами що дорівнюють кількості унікальних категорій (у нашому випадку 3). Використовується для остаточної класифікації.

13

Рисунок Б.13 — Слайд 13

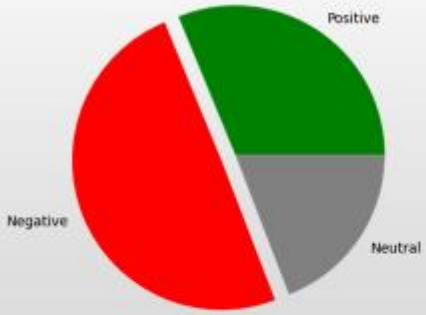
Демонстрація роботи програмного забезпечення

	content	sentiment	polarity	subjectivity	neg	neu	pos	compound
0	biden s jen psaki at briefing says planning for neptune strike exercise began in 2020 long before the current escalation of tensions from russia s aggressive posture towards ukraine she noted it s 1st time since cold war a us carrier group under nato command	negative	-0.216667	0.600000	0.208000	0.792000	0.000000	-0.827100
1	if we wake up tomorrow morning and putin has invaded ukraine and is on his way to kiyv i hope some ukrainian pilots don t bomb the border troops but fly right on to moskow to drop their bombs there it will have a more definitiv impact war is hell	negative	0.528571	0.645238	0.278000	0.690000	0.032000	-0.951300
2	lol the token response to putin is comical the germans will be begging russian gas before too long biden s sanctions and 800 troops is mere virtue signalling and in the words of paul keating is akin to throwing toothpicks at a growling bear	positive	0.150000	0.520000	0.000000	0.877000	0.123000	0.680800
3	army of the rf on its territory this is the right of any country kiev violates the minsk agreements the us and uk are pushing ukraine to war is this normal for you do you think killing people in donbass is normal	negative	0.195238	0.611905	0.224000	0.735000	0.041000	-0.888500
4	vlady putin is a very stupid human the coward wants no nato troops on his border but if he shit hole country invades ukraine he will get more troops on his border dumb ass	negative	-0.145833	0.500000	0.426000	0.574000	0.000000	-0.964000

14

Рисунок Б.14 — Слайд 14

Кругова діаграма настроїв твітів



- Total Positive Tweets % : 31.22
- Total Negative Tweets % : 49.62
- Total Neutral Tweets % : 19.16

На період з 1 січня по 6 березня 2022 року

15

Рисунок Б.15 — Слайд 15

Верифікація і валідація системи

Оцінювання якості класифікації емоцій вхідного повідомлення здійснювалася із використанням трьох метрик: точність, повнота і збалансована F1-міра.

- Precision $Precision = \frac{TP}{TP+FP}$
- Recall $Recall = \frac{TP}{TP+FN}$
- F-міра $F = 2\frac{Precision * Recall}{Precision + Recall}$

	precision	recall	f1-score
0	0.89	0.89	0.89
1	0.87	0.88	0.88
2	0.86	0.85	0.85
macro avg	0.87	0.87	0.87
weighted avg	0.88	0.88	0.88

16

Рисунок Б.16 — Слайд 16

Рисунок Б.17 — Слайд 17

Рисунок Б.18 — Слайд 18

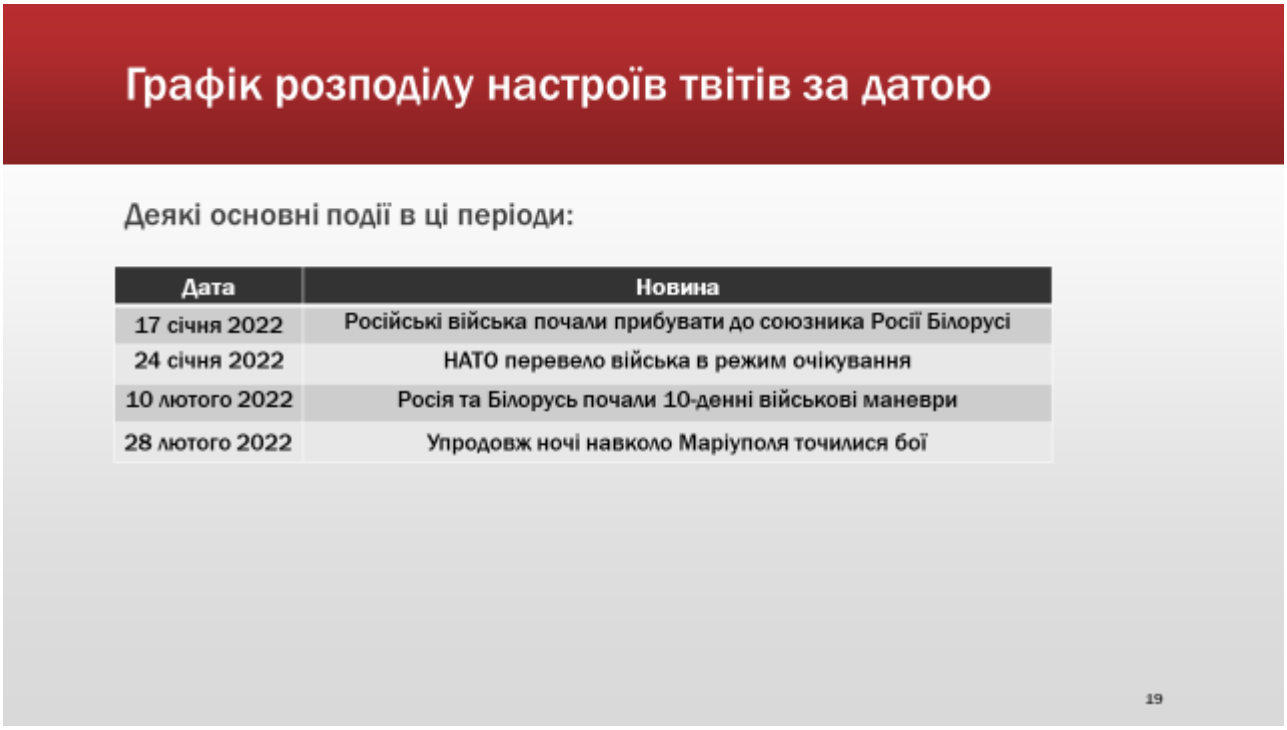


Рисунок Б.19 — Слайд 19

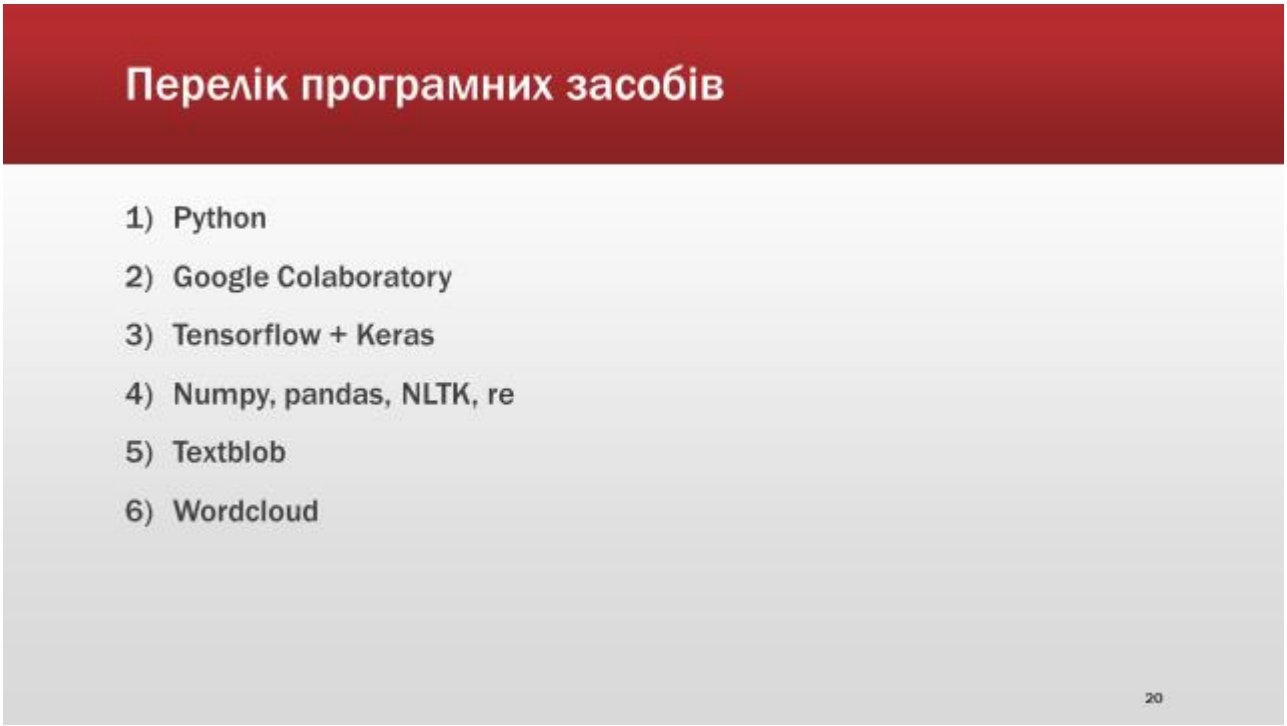


Рисунок Б.20 — Слайд 20

Висновки

1. Було розглянуто основні підходи до сентимент аналізу тексту. Було досліджено та встановлено такі методи векторизації тексту: **Bag-of-Words, TF-IDF, Word2Vec** та **GloVe**. Після порівняння їх характеристик будо обрано модель **GloVe** для векторного представлення слів. Проведено аналіз методів машинного навчання для для класифікації тексту: наївний Баєсів класифікатор, метод опорних векторів та нейронні мережі **LSTM**. На основі порівняння характеристик цих методів було встановлено, що найкращим вибором для побудови системи буде нейронна мережа типу **LSTM**.
2. Розроблено математичну модель системи аналізу настроїв у соціальній мережі Твіттер, яка складається з наступних компонентів: «Сховище даних», «Попередня обробка», «Анотація даних», «Векторизація», «Навчання класифікатора», «Класифікатор», «Візуалізація». Було описано набір даних який ми використовуємо для навчання та тестування нашої моделі, методи попередньої обробки, алгоритм анотації, метод навчання нейронної мережі та метрики оцінки точності роботи нашої системи: **F-міра, Recall** та **Precision**.
3. Розроблено програмне забезпечення системи. Проведено валідацію і верифікацію системи і визначено що найкращі значення клпсифікації на тестовому наборі даних були отримані для негативного сентименту і склали **precision = 89%, recall = 89 %, F1 = 89%**. Середня оцінка точності моделі становить **87%**.
4. Встановлено, що дані соціальних мереж можна використовувати як інструмент для оцінки настроїв суспільства з приводу будь-яких тем. Перспективи подальших досліджень полягають у знаходженні розподілу настроїв людей з різних країн, моделюванні тем до яких висловлються найбільш позитивно або найбільш негативно, розширенні метрик оцінки тональності публікацій у соціальних мережах до класифікацій емоцій (злість, сум, відраза, спокій, радість і т.д.).

21

Рисунок Б.21 — Слайд 21

Дякую за увагу!

22

Рисунок Б.22 — Слайд 22