МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Емоційна оцінка тональності тексту»

Завідувач випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Кузіков Б.О.**

Студент гр. ІН–61 **Супрун О.П.**

Суми 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІКТ

Затверджую_____

Зав. кафедрою Довбиш А.С.

“_____” _____2020 р.

ЗАВДАННЯ

до випускної роботи

Студента четвертого курсу, групи ІН-61 спеціальності “Комп’ютерні науки” денної форми навчання Супруна Олександра Павловича.

Тема: “**Емоційна оцінка тональності тексту**”

Затверджена наказом по СумДУ

№_____ від _____2020 р.

Зміст пояснювальної записки:1)огляд існуючих рішень; 2)постановка завдання й формування завдань дослідження; 3)вивчення принципів роботи нейронних мереж; 4) програмна реалізація та її опис; 5)висновки.

Дата видачі завдання “_____”_____2020 р.

Керівник випускної роботи_____ Кузіков Б.О.

Завдання прийняв до виконання _____ Супрун О.П.

РЕФЕРАТ

Записка : 37 с., 24 рис., 4 табл., 2 додатки, 4 розділи, 11 джерел.

Об'єкт дослідження: методи оцінювання емоційної тональності тексту.

Мета: розробка інформаційної системи для оцінювання емоційної тональності коротких текстів, а також реалізація та демонстрації роботи.

У роботі проведено аналіз існуючих методів та рішеннь для аналізу емоційної тональності.

Результати:

- створено та навчено модель рекурентної нейронної мережі для аналізу та оцінки емоційної тональності тексту для коротких текстів з емоційним забарвленням;
- використано модель Word2Vec для обробки та аналізу текстів нейронною мережею на основі векторних представлень слів.
- використано сучасну модель рекурентної нейронної мережі з модифікацією для корегування обчислень зважаючи на попередні та наступні данні.

Нейронну мережу отриману в результаті даної роботиможна використовувати для цоінки емоційної тональності коротких текстів. Задля збільшення точності обчислювань рекомендується додатково натренувати модель нейронної мережі на типових для середовища використання даних.

ОБРОБКА ПРИРОДНОЇ МОВИ, АНАЛІЗ ЕМОЦІЙНОЇ ТОНАЛЬНОСТІ ТЕКСТУ, ВЕКТОРНІ ПРЕДСТАВЛЕННЯ СЛІВ, ДВОНАПРАВЛЕННІ РЕКУРЕНТНІ НЕЙРОННІ МЕРЕЖІ, BIDERECTIAN LSTM.

Зміст

Вступ

Класифікація емоційного забарвлення повідомлень дуже важлива для аналізу відношення суспільства до продуктів та послуг компаній з метою визначення їх переваг та недоліків, аналізу реакцій відносно подій у світі, новини та телепередач задля збільшення рейтингу перегляду каналу тощо. Така класифікація забезпечує більш тісну взаємодію з користувачами та допомагає приймати рішення щодо подальшого розвитку у напрямах, які користуються більшим попитом.

Текст не має жодних власних емоцій, проте він може викликати емоції у особи, яка його читає, а також може відобразити або виразити емоційний стан людини, яка його написала [9]. Класичні методи оцінки тональності тексту працюють на певному наборі заданих правил, за якими визначається належність тексту до певного класу емоцій. Використання наборів правил для оцінювання потребує детального аналізу великої кількості даних для додавання нових або корегування старих правил , відповідно до змін у суспільстві.

Альтернативний метод для визначення емоцій у тексті є використання технік та методів машинного навчання . У машинному навчанні проблема аналізу тональності тексту вирішується не за допомогою певного набору запрограмованих правил, а завдяки створенню певної моделі, яка може аналізувати та оцінювати дані для класифікування емоції [6]. Частиною машинного навчання є глибоке навчання, яке, в свою чергу, є частиною штучного інтелекту [7]. Використовуючи глибоке навчання для вивчення даних, нейронні мережі будуть корисними інструментом, який може вирішити поставлене завдання [6].

Використання нейронних мереж для аналізу даних підвищує ефективність та швидкість обробки даних і не потребує трудомістких дій для корректної роботи.

Таким чином, метою дослідження є розробка інформаційної системи оцінювання тональності тексту, опис принципів її роботи та практична реалізація.

У процесі виконання поставленої задачі були виконані наступні дії:

- аналіз існуючих методів, підходів та рішень у галузі обробки природної мові та класифікації тексту за емоційним забарвленням;
- розглянуто варіанти підвищення ефективності існуючих рішень щодо аналізу тональності текстів;
- розроблено програмний продукт для розв’язання поставленої задачі;
- проведено аналіз варіантів покращень роботи системи;

Предметом дослідження є використання рекурентних нейронних мереж для оцінки тональності тексту на основі векторних представлень слів.

1. Інформаційний огляд

1.1. Огляд відомих рішень

Існуючі рішення у сфері оцінки тональності є аналітичні платформи Textrics, Brand24, Tweet Sentiment Visualization та SocialMention. Усі програми здатні до семантичного аналізу тексту .

1.1.1. Brand24

Програмний продукт з веб інтерфейсом для збору та оцінювання інформації щодо згадувань заданого ключового слова у соціальних мережах, засобах масової інформації, повідомленнях, а також класифікації позитивних та негативних згадувань у режимі реального часу. Унікальність інструменту Brand24 від інших полягає у тому, що він дозволяє не лише оцінити тональність тексту, але й надає статистичні дані щодо відношення негативних та позитивних оцінок відносно досліджуваного об’єкту. (Рисунок 1.1).

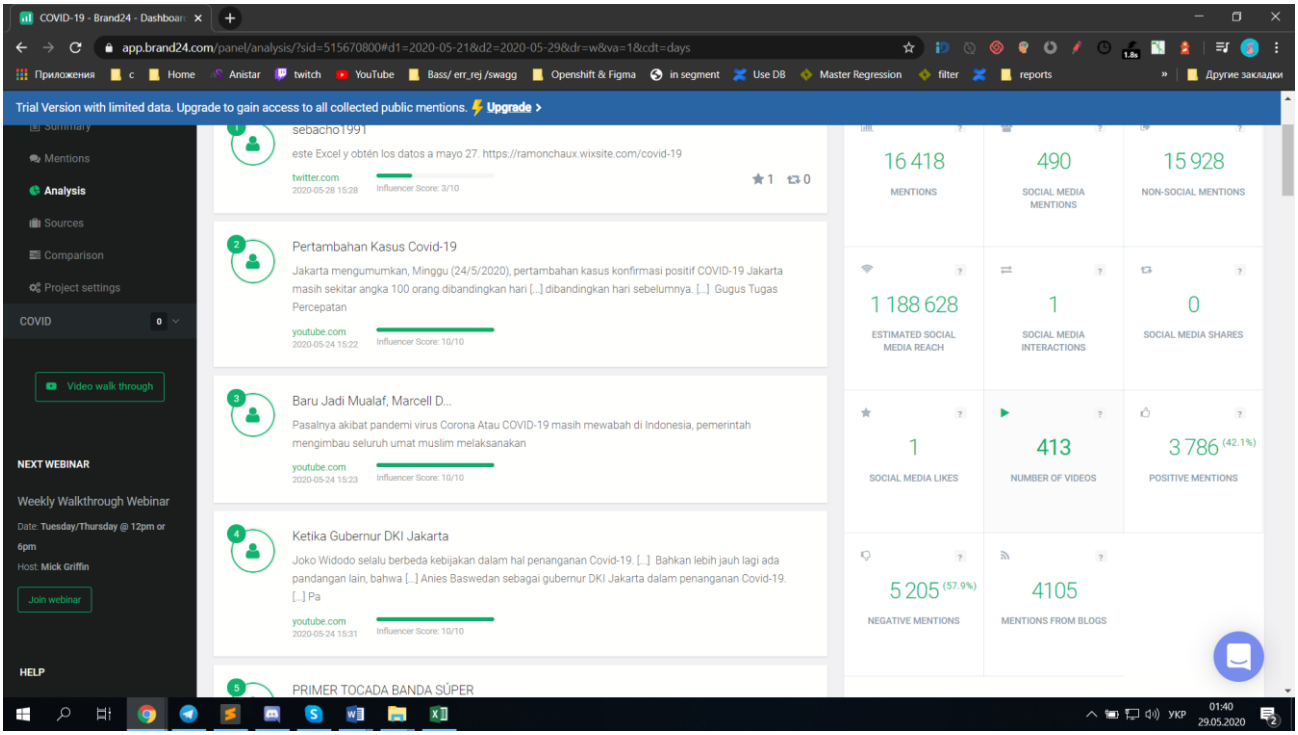


Рисунок 1.1 Приклад роботи сервісу Brand24

1.1.2. Tweet Sentiment Visualization

Онлайн сервіс ,який спеціалізується на оцінці та візуалізації емоцій для коротких, неповних фрагментів тексту. Принцип класифікації емоції визначається як для візуалізації даних про згадування ключових слів у повідомленнях з певним емоційним забарвленням, кількість повідомлень за певний проміжок часу ґрунтуючись на даних зібраних даних у соціальній мережі Twitter (Рисунок 1.3).

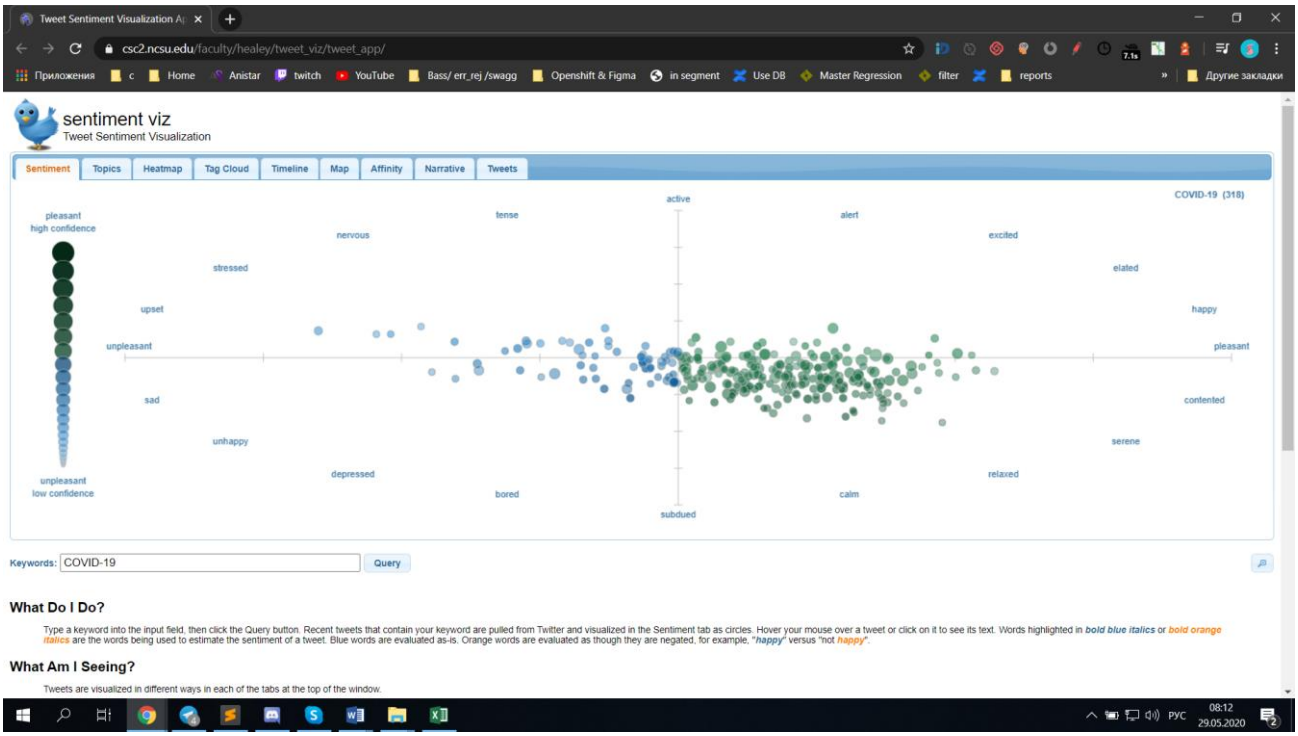


Рисунок 1.2 Приклад роботи сервісу Tweet Sentiment Visualization

1.1.3. Textrics

Програмний продукт з веб інтерфейсом, особливістю якого є визначенні сутностей, відношень та семантики у тексті й його класифікація на позитивний, негативний та нейтральний з відображенням вірогідності належності тексту до певної емоції. Інструмент працює на власній базі даних, яка періодично оновлюється. Textrics здатний аналізувати текст незалежно від джерела з якого він був отриманий та відображеної у ньому теми (Рисунок 1.3).

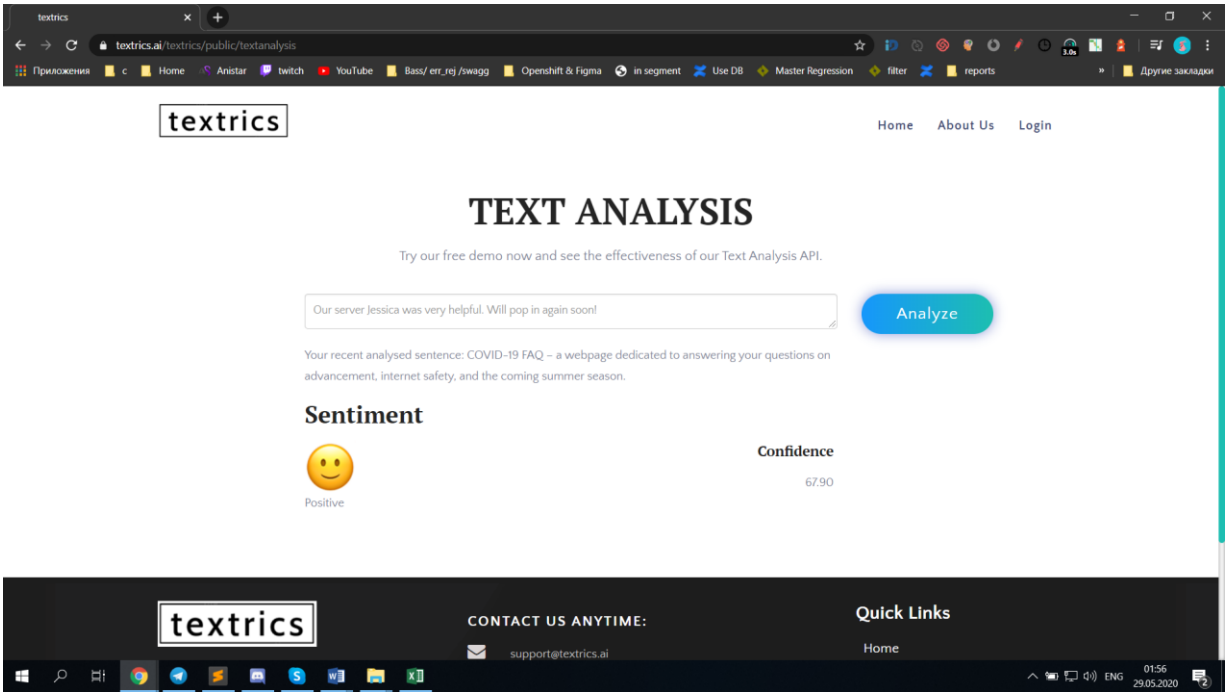


Рисунок 1.3 Приклад роботи сервісу Textrics

1.1.4. Social Mention

Social Mention – це платформа пошуку та аналізу інформації у всесвітній мережі, яка спеціалізується на відстежуванні відгуків користувачів щодо певного предмету, бренду, новини тощо. Система дозволяє дуже легко відстежувати та оцінювати відгуки користувачів у режимі реального часу. Social Mention проводить моніторинг великої кількості соціальних мереж та інтернет-ресурсів, в тому числі Twitter, Facebook, YouTube тощо. За ключовим словом (новиною, назвою продукту, назвою бренду тощо) можливо отримати інформацію щодо кількості позитивних, негативних та нейтральних згадувань їх пропорційне відношення один до одного, а також кількість унікальних публікацій, їх пересилань та статистичні дані щодо частоти пошуку за згадуваним ключовим словом (Рисунок 1.4).

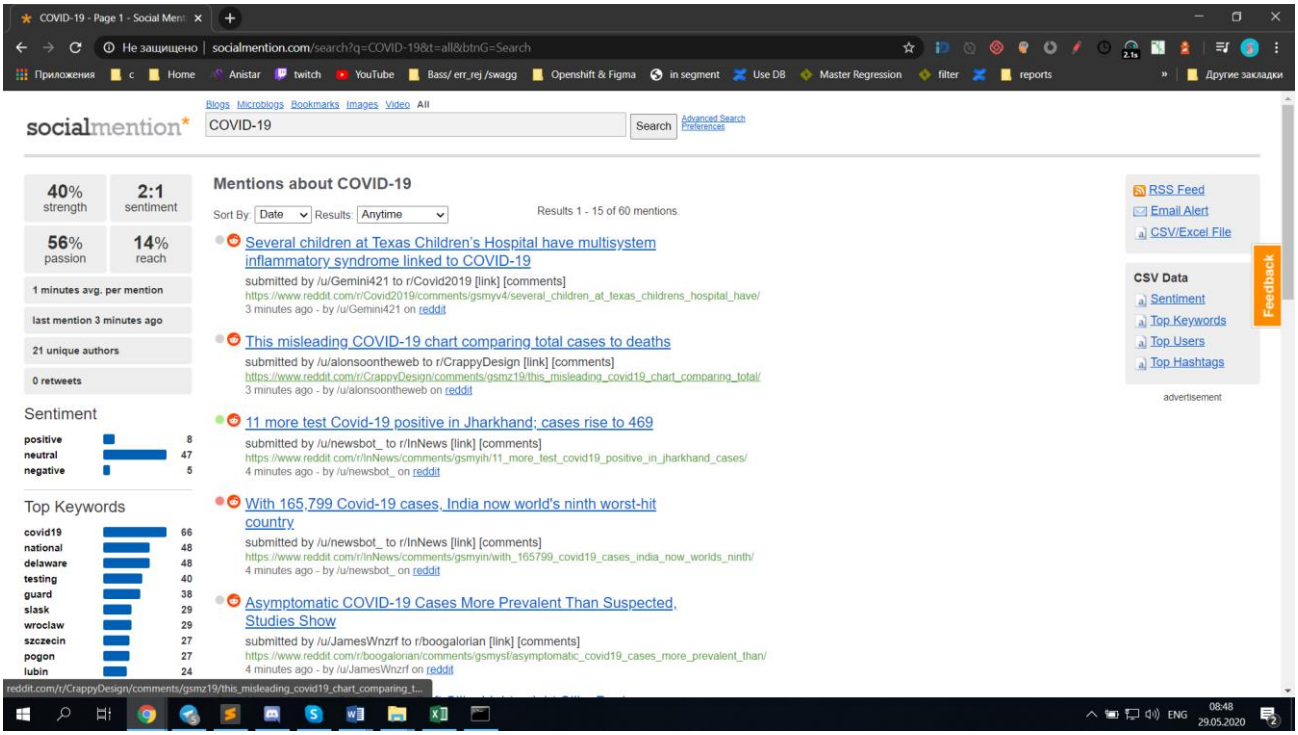


Рисунок 1.4 Приклад роботи сервісу Socialmentions

1.2.Опис відомих методів визначення тональності

1.2.1 Словниковий метод

Словниковий метод – це один із, якщо не найпростіший метод визначення тональності тексту. Він базується на словниках з чисельними значеннями так званих «ключових» слів, які несуть тональну характеристику й розподіляються на «негативні» та «позитивні» слова. Оцінка тональності вираховується як результат певної функції відштовхуючись від числової характеристики «негативних» та «позитивних» слів у реченні.

Переваги:

- дешевий та простий у реалізації;
- гарна оцінка відносно коротких простих речень;

Недоліки:

- не підходить для визначення тональності складних речень;
- не може визначити відносну характеристику у порівняннях, як результат оцінка може бути протилежна задуму;

- не вміє визначати омоніми (слова, які мають різні значення залежно від контексту);
- необхідність постійного збагачення словників для актуальної оцінки тональності;
- Велика вірогідність отримати протилежній результат при звичайних вхідних даних.

1.2.2. Статистичний метод

Статистичний метод – покращена реалізація словникового методу, яка працює за рахунок комбінації оцінки словникового методу та значень статистики використання певних слів та фраз із ними.

Переваги:

- більша точність;
- незначне зменшення впливу омонімів;

Недоліки:

- ті ж недоліки як і в словниковому методі;
- складніший у реалізації;
- потребує більше ресурсів.

1.2.3 Об’єктно-залежний метод

Об’єктно-залежний метод – це метод, який бере основу від своїх попередників, проте при оцінюванні тональності зважає на так званий «залежний» об’єкт, який отримує тональну оцінку, завдяки чому може будувати статистику та ієрархію кількох об’єктів, які згадуються у порівнянні.

Переваги:

- краща точність ніж у попередників;
- зменшення впливу омонімів за рахунок належності до об’єкту;
- можливість розрахунку відносної оцінки у порівнянні об’єктів;
- можливість побудови ієрархії порівняних об’єктів;

- може приймати та коректно обраховувати складні речення.

Недоліки:

- складний у реалізації;
- потребує велику кількість обчислювальних ресурсів;

1.3.Постановка задачі

Виявлення емоцій у текстових діалогах є складною проблемою за відсутності міміки людини та голосової модуляції зі зміною тембру голосу. Більше того, під час розмови та листування контекст постійного діалогу може повністю змінитися і, як результат, емоція теж буде змінена відповідно до змін у контексті діалогу. При порівнянні усього контексту діалогу з окремими його частинами виявлені емоції можуть кардинально відрізнятися одна від одної.

Розглянемо фразу: «Цей краєвид занадто прекрасний. Я зараз заплачу» Зауважимо, що вислів «Я зараз заплачу» , сприймається як вияв «**сумної**» емоції, проте зважаючи на контекст попереднього речення в цьому прикладі можна сказати, що людина відчуває крайнє збудження та щастя від споглядання прекрасного і плач показує степінь «**радості»** людини.

Звичайно, розгляд контексту для оцінки емоцій висловлювання тексту стає ще більш важливим для вищезгаданого сценарію.

Коректне інтерпретування емоцій є необхідним для покращення роботи цифрових помічників та розмовних агентів, через їх текстовий розмовний інтерфейс. Це завдання збільшило науковій інтерес до розпізнавання емоцій у тексті.

Метою моєї роботи є створення нейронної мережі, яка може визначати емоційне забарвлення текстів та класифікувати репліки за класом емоцій, які присутні в тексті.

2. Технології розв’язку поставленої задачі

2.2.Рекурентні нейронні мережі

Рекурентні нейронні мережі (англ. Recurrent neural network; RNN) – це клас нейронних мереж особливістю яких є тип з’єднання між вузлами, а саме утворення направленої послідовності елементів, представленої у вигляді орієнтованого графу. Відмінністю рекурентних нейронних мереж від попередніх поколінь нейронних мереж є можливість нейронів передавати не тільки результат оброблених дані керуючись бінарною логікою, але й у вигляді обробленої послідовності. Також подання кожного нейрону у вигляді невід’ємного набору цілих чисел сприяло більш детальному представленню його стану. Рекурентні нейронні мережі, завдяки можливості використовувати внутрішню пам’ять кожного нейрона для обробки довільних послідовностей вхідних даних широко використовуються у задачах обробки даних, у яких об’єкт поділений на кілька частин, наприклад:

- розпізнавання рукописного тексту ;
- розпізнавання мови;
- моделювання мови;
- переклад тексту.

Принцип архітектури рекурентної нейронної мережі фрагмент полягає в тому, що комірка A приймає вхідні данні x_t та повертає аене оброблене значення h_t. Наявність циклу дозволяє передавати інформацію від одного кроку до іншого (Рисунок 2.1).

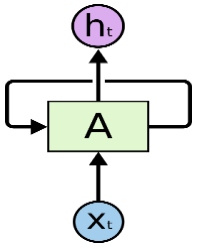


Рисунок 2.1 Зображення рекурентної нейронної мережі[3]

Для простішого розуміння можна уявити рекурентну нейронну мережу як декілька послідовних ідентичні копій з однаковим функціоналом (Рисунок 2.2).

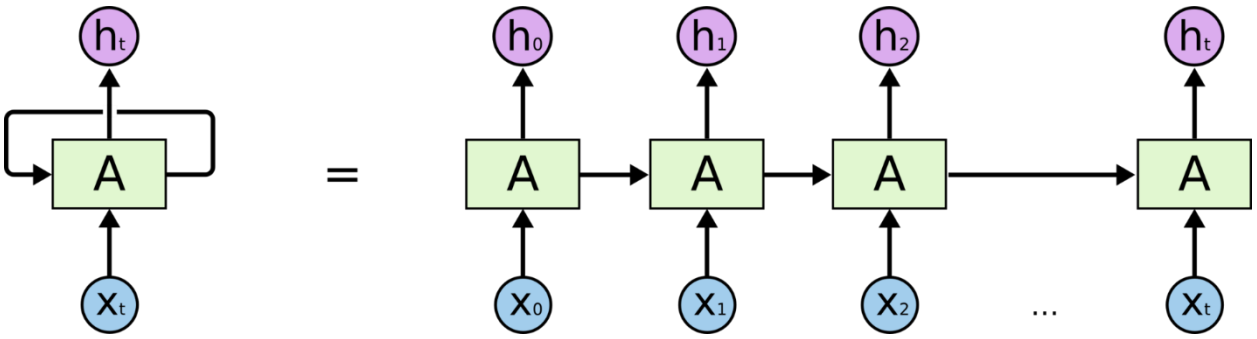


Рисунок 2.2 Розгорнуте зображення рекурентної нейронної мережі [3]

Перевагою RNN мереж є вміння використовувати інформацію, збережену у своїй внутрішній пам'яті з попередніх тестів та використовувати її для обробки нової, а отже така будова архітектури допомагає не лише в обробці майбутніх даних, але й покращує точність прогнозів майбутніх результатів. Наприклад при аналізуючи вислів «У понеділок йшов дощ» мі розуміємо, що мова йде про **понеділок** та **дощ.** У випадку, коли нові дані будуть потребувати інформацію, отриману з попередніх тестів і кількість проміжних тестів невелика, RNN можуть використати попередні дані для виконання поставленої задачі(Рисунок 2.3).

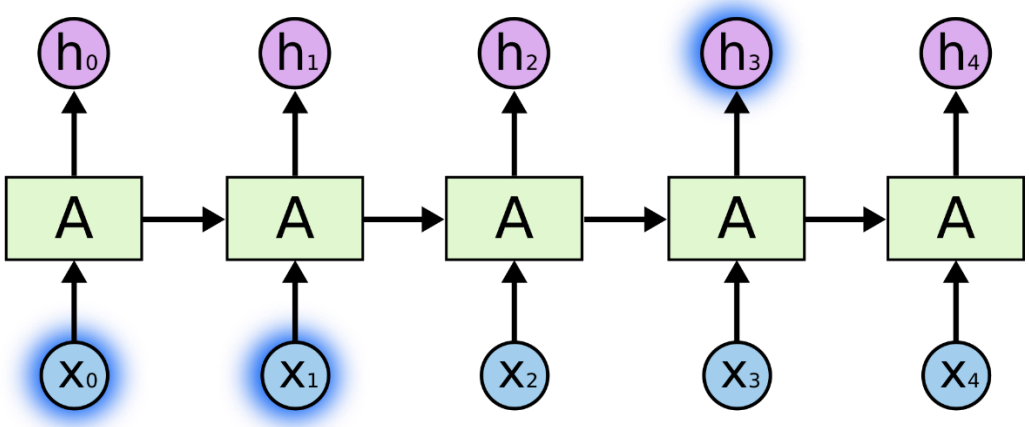


Рисунок 2.3 Приклад використання старих даних у новому аналізі при невеликій відстані між пов'язаними даними [3]

Проте нерідкі випадки, коли «відстань» між даними доволі велика і дані погано зв’язуються з пам’яттю. Наприклад у випадку аналізу розповіді про подорож до іншої країни сама назва країни згадується лише кілька разів на початку проаналізувати, що опис різних об’єктів, спогадів, несе значно більше інформації, тому інформація про назву країни може видалитися з пам’яті (Рисунок 2.4).

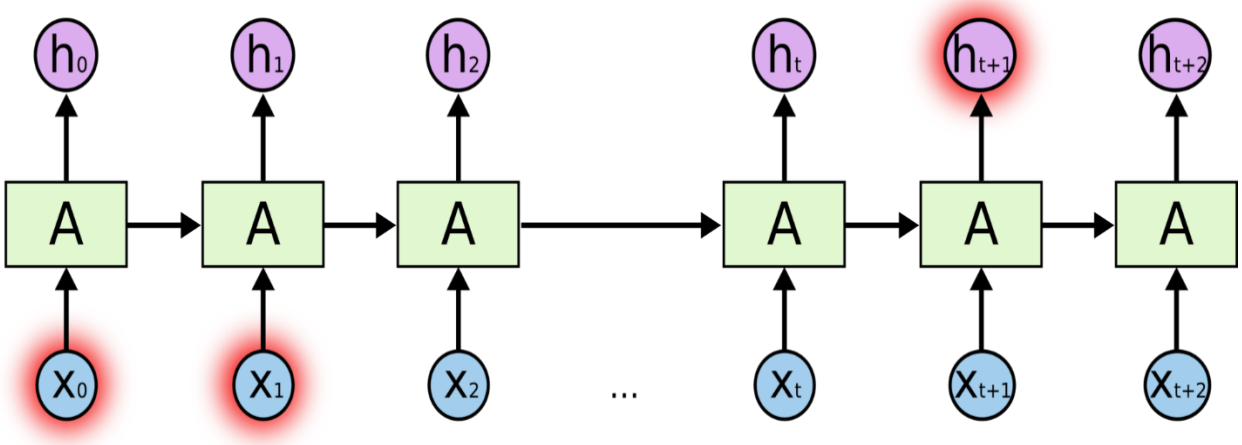


Рисунок 2.4 Велика відстань між пов’язаними даними [3]

2.3.Тривала короткострокова пам'ять LSTM

Тривала короткострокова пам'ять (Long short-term memory; LSTM) – це покращення архітектури RNN мережі, яка була спеціально розроблена з метою зменшення градієнтних втрат даних при великій відстані між даними за рахунок навчання довготривалій залежності.

Будь-яка рекурентна нейронна мережа має форму ланцюжка повторюваних модулів нейронної мережі. У звичайній RNN структура одного такого модуля дуже проста, наприклад, він може являти собою один шар з певної функцією активації, результати якої знаходяться в межах від наприклад: гіперболічний тангенс, SoftPlus, сигмоїд, Maxout тощо. (Рисунок 2.5)

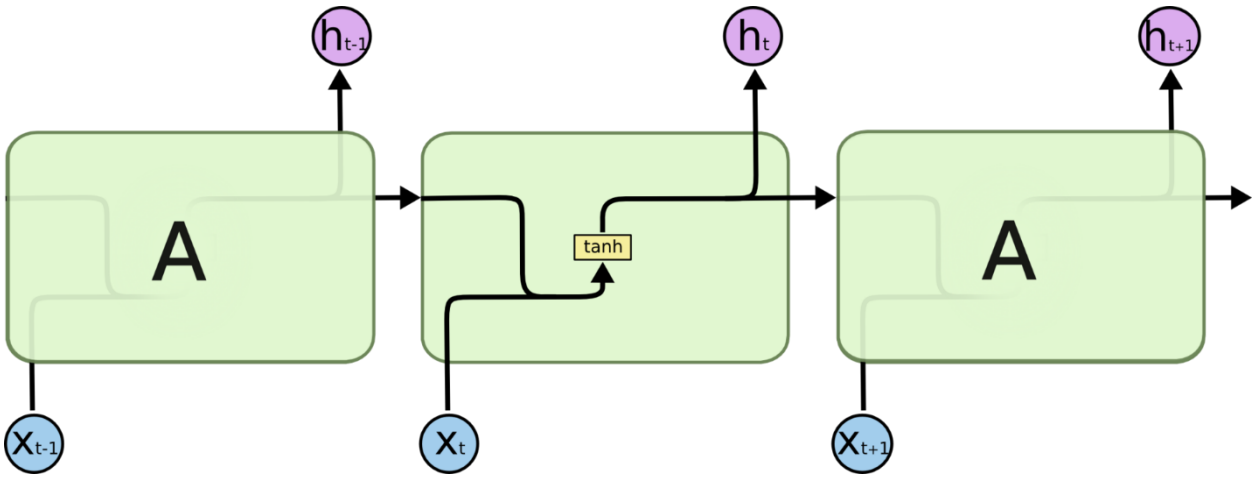


Рисунок -2.5 Повторюваний модуль в стандартній одношаровій RNN [3]

Оскільки LSTM – це покращена реалізація структури RNN, вона наслідує властивість повторення однакових блоків від свого попередника, проте структура цих блоків набагато складніша (Рисунок 2.6).

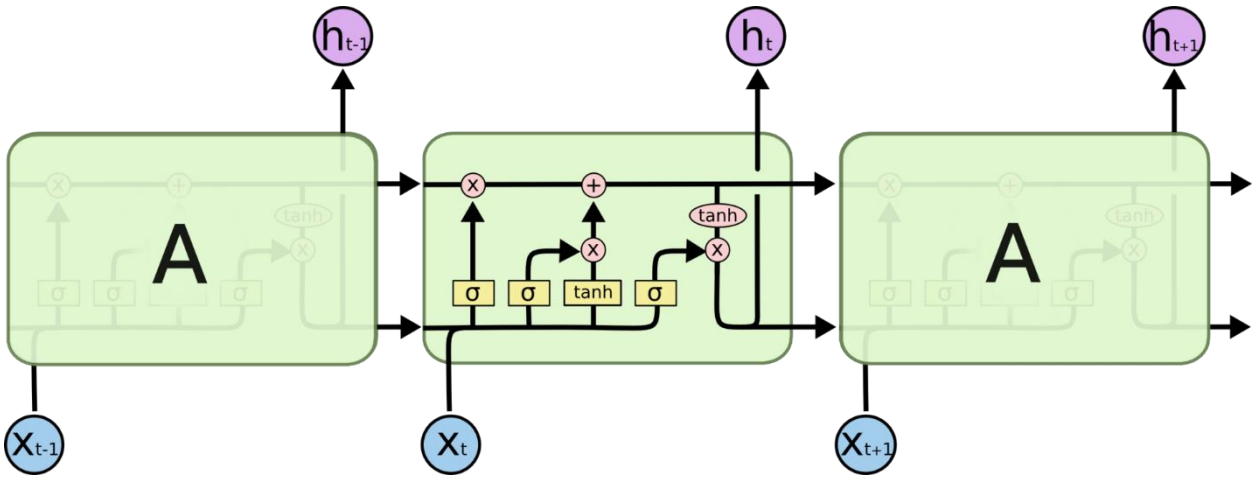


Рисунок 2.6 приклад будови повторювального блоку LSTM мережі [3]

Для LSTM мережі найважливішим моментом є стан комірки (cell state) – який зображується горизонтальною лінією у верхній частині схеми.(Рисунок 2.7)

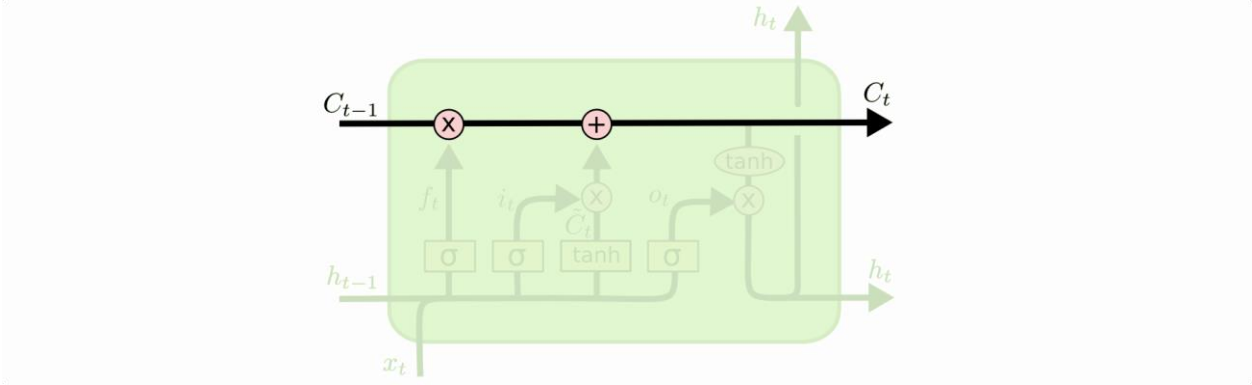


Рисунок 2.7 Стан комірки [3]

Він проходить через увесь ланцюжок нейронів, беручи участь в кількох лінійних перетвореннях протягом усього шляху. Інформація має можливість проходити крізь нейрон, не зазнаючи жодних змін. Проте, LSTM мережі можуть видаляти та заносити нову інформацію до стану комірки. Процеси видалення та збереження інформації регулюються спеціальними структурами, які називаються фільтрами(gates). Фільтри регулюють об’єм пропущеної інформації на основі проведених розрахунків всередині комірки. До складу фільтру входить один шар сигмоїдальної нейронної мережі та операція поточечного множення між отриманими результатами з сигмоїдального шару та поточного стану стану комірки (Рисунок 2.8.).

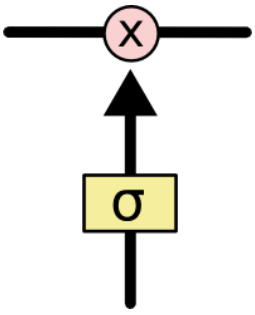
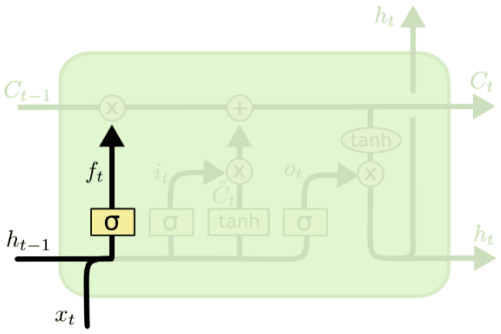


Рисунок 2.8 Використання фільтру [3]

На вихід з сигмоїдального шару подається набір чисел між нулем та одиницею, кожне з яких позначає необхідність передачі певної частки кожного блоку інформації слід далі по мережі. У випадку коли сигмоїдальний шар видає

нуль на вихід, то блок інформації не буде переданий до наступного нейрону (еквівалент операції видалення), передача одиниці означає передачу усього блоку інформації без змін. У LSTM наявні три фільтра, які контролюють стан комірки.

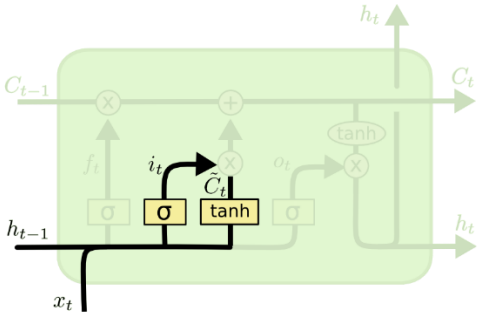
Значення комірки обчислюються поступово у кілька кроків. Перший крок – це аналіз та обчислення частини інформації з попередньої комірки, яку можна видалити. За це відповідає сигмоїдальний шар, який називається «шар фільтра забуття» (forget gate layer). Його задача – аналізувати отримані вхідні дані x_t, а також результат попереднього обчислення h_t та повернути число в діапазоні [0;1] для кожного числа із стану попередньої комірки C_{t-1}. (Рисунок 2.9)



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

Рисунок 2.9 Сигмоїдальний шар [3]

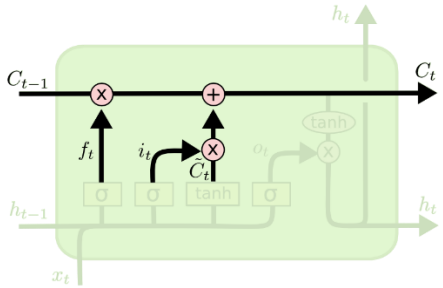
Наступний крок обчислення стану комірки – це виділення частини нової інформації, яка буде зберігатися в стані комірки. На цьому етапі послідовно виконуються дві дії. Спочатку сигмоїдальний шар вхідного фільтра (input gate layer) обчислює вхідні дані для виділення блоків інформації, які необхідно оновити. Після цього шар гіперболічного тангенсу (tanh) створює вектор нових значень, які будуть додані до стану комірки.(Рисунок 2.1)



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Рисунок 2.2 Другий крок обчислення стану [3]

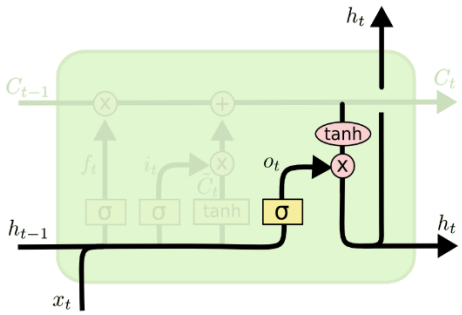
Наступна дія - заміна значення стану попередньої комірки на значення стану нової комірки. Які саме зміни нам потрібно виконати було вирішено на попередніх кроках, залишається тільки перерахувати дані . Перемноживши старе значення стану на значення f_t («шар фільтра забуття»). Потім додаємо добуток i_t * C_t. Нові значення, помножені на t – коефіцієнт, які показують на скільки потрібно оновити кожний блок інформації у стані комірки.(Рисунок 2.11)



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Рисунок 2.31 Зміна стану [3]

Останній етап це виділення частини інформації, яку потрібно передати. Вихідні дані будуть засновані на поточному стані комірки з застосуванням деяких фільтрів. Спочатку треба застосувати додатковий сигмоїдальний шар, який виділяє блоки інформації зі стану комірки буде для обчислення на наступному кроці. Потім значення стану осередку проходять через tanh-шар, щоб отримати на виході значення з діапазону від -1 до 1, після чого вони перемножуються з вихідними значеннями іншого сигмоїдального шару, що дозволяє виводити лише необхідну інформацію. (Рисунок 2.12)



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Рисунок 2.12 Вивід необхідної інформації[3]

2.4.Векторне подання слів

Векторне подання - загальна назва для різних підходів моделювання мови та навчання уявлень в обробці природної мови, спрямованих на зіставлення словами (іноді висловам) з певного словника векторів R^n відповідних значень n. Векторне представлення слів є невід'ємною частиною більшості підходів до створення NLP-систем із застосуванням глибокого навчання оскільки це один з найефективніших способів підготовки даних для подальшого використання в нейронних мережах. Серед існуючих моделей векторного представлення найчастіше використовують Word2Vec або GloVe.

Модель Word2Vec – це прогнозуюча модель, вона знаходить взаємозв'язок між словами згідно з припущенням, що в схожих контекстах зустрічаються семантично близькі слова. Word2Vec намагається прогнозувати цільове слово (архітектура CBOW) або контекст (архітектура Skip-Gram) базуючись на тексті й шукаючи слова, які найчастіше використовуються у подібних за структурою реченнях, тобто Word2Vec намагається мінімізувати функцію втрат корисної інформації, що добре доповнює особливості нейронної мережі LSTM

Модель GloVe - це модель векторного представлення, яка базується на підрахунку частоти використання слова у контексті на великому наборі даних та факторизації отриманої матриці векторних представлень у матрицю меншої розмірності. Процес зменшення розмірності відбувається шляхом мінімізації

втрат даних при зменшенні розмірності за рахунок пошуку представлень меншої розмірності, які можуть описати більшу частину дисперсії використання слова у матриці більшої розмірності.

Для поставленої задачі, а саме – оцінки емоційного забарвлення тексту, модель Word2Vec підходить краще, оскільки у векторному просторі цієї моделі існують співвідношення між цільовим словом та його контекстним словом, які покращують результати передбачення.

2.4 Принцип оцінення результату

Оцінювання результату проводиться на результатах передбачення нейронної мережі та значенні мікросервісної оцінки F1 для класифікації за шістьма класами емоцій:

- Щастя.
- Сум.
- Гнів.
- Страх
- Здивованість
- Огида

Для обчислення мікросервісної оцінки F1 обчислюється на основі двох інших метрик Precision та Recall.

Precision – це метрика, яка відображає відношення кількості правильно визначених екземплярів певного класу відносно кількості усіх екземплярів, які нейронна мережа визначила як екземпляр до цього класу. Використання метрики Precision вигідно у випадках коли ціна помилки передбачення належності об’єкту класу А до класу Б дуже велика. Значення Precision обчислюється за формулоюза формулою (2.4.1)

$$P\mu = \frac{\Sigma TPi}{\Sigma(TPi + FPi)} \forall i\epsilon \left\{ \begin{matrix} anger, disgust, fear, \\ happiness, \\ sadness, surprise \end{matrix} \right\} \quad (2.4.1)$$

Recall - це метрика, яка відображає відношення кількості правильно визначених екземплярів певного класу відносно усіх екземплярів цього класу. Використання метрики Recall вигідно у випадках коли ціна помилки передбачення належності , що об'єкт класу A не належить до класу A дуже велика. Значення Recall обчислюється за формулою (2.4.2)

$$R\mu = \frac{\Sigma T\ Pi}{\Sigma(TPi + FNi)} \forall i\epsilon \left\{ \begin{matrix} anger, disgust, fear, \\ happiness, \\ sadness, surprise \end{matrix} \right\} \quad (2.4.2)$$

У формулах 2.4.1 та 2.4.2 TP_i – це кількість зразків класу «i», які правильно прогнозуються нашою нейронною мережею, FNi - це кількість зразків класу «i», які помилково прогнозуються нашою нейронною мережою до іншого класу «j», FPi – кількість зразків інших класів емоцій, які помилково прогнозуються нашою нейронною до класу «i».

F1μ - це метрика , яка обчислюється як середнє гармонійне значень метрик Precision та Recall за формулою (2.4.3). Для задачі оцінки тональності тексту найкраще підходить саме метрика F1, оскільки вона передбачає нерівномірний розподіл даних до різних класів.

$$F1\mu = 2 \cdot \frac{P\mu \cdot R\mu}{P\mu + R\mu} \quad (2.4.3)$$

2.5 Обробка вхідних даних

Перед аналізом тексту нейронною мережею, його необхідно попередньо обробити видаливши неважливі елементи та привести до необхідного (числового) вигляду. Для попередньої оброки використано інструменты

быблыотеки ekphrasis з спеціальними налаштуваннями для розуміння спец. символів, хештегів, смайлів. У результаті отримано текст з мінімальною кількістю зайвої інформації, виправленою орфографією, нормалізованими словами та виразами, після цього буде проведено виправленого тексту та визначено, які дані не несуть в собі важливої інформації, та повинні буті відкинуті, а також дані, які потребують нормалізації та позначення спеціальними тегами. На етапі попередньої обробки та нормалізації тексту виконуються наступні операції:

- Замінюються електронні адреси URL, поштові адреси, дата та час, спеціальні імена, абревіатури, відсотки, спец символи(@,#,^ тощо), позначення валют та числа будуть замінені спеціальними тегами;
- Повторення, цензурність та навмисне подовження термінів відмічаються спеціальними мітками;
- Навмисно подовжені слова будуть скориговані до нормальних (словникових) значень.

Таблиця 2.1 Приклад попередньої обробки тексту

Вхідний(оригінальний) текст	Текст після обробки інтсрументами бібліотеки ekphrasis
Check this out. It's kinda realy COOOOOLLLLL https://www.youtube.com/watch?v=dmH95ft	check this out. it ' s kinda realy <allcaps> cool <elongated> </allcaps> <url>
The rent for last moth has raised. Now I have 40$ less in ma pocket :'-(:'-(:'-(	the rent for last moth has raised . now i have <money> less in ma pocket <sad> <sad> <sad>
It’s rainig ALL DAY. I’m so booooooorrrreeed	it ' raining <allcaps> all day</allcaps>. i it ' m so boored <elongated>

3 Програмна реалізація

3.1 Підготовка даних та створення нейронної мережі

Для початку обробки тексту необхідно виконати налаштування функції-обробника тексту з зазначенням відповідних параметрів для розпізнавання та коректної обробки спеціальних конструкцій таких як посилання, смайли, хештеги, згадування користувачів тощо (Додаток А.1).

Після зчитування та попередньої обробки тексту виконується перетворення слів з символьної форми у числову форму. Для цього завантажуємо матрицю векторних представлень для 658129 слів з векторами довжиною у 300 цілочисельних значень для кожного слова. Перетворюємо слова у числову форму відповідно до порядкового номеру слова у матриці векторних представлень та виконуємо процес перемішування тексту для підвищення якості навчання нейронної мережі за рахунок уникнення довготривалих повторень одного класу емоцій (Додаток А.2).

Останній етап обробки вхідних даних для нейронної мережі - це формування масивів однакової довжини для обробленого тексту та представлення міток емоцій у вигляді одномірного масиву з 0 та 1 довжини n, де n – це кількість класів емоцій, а 1 представляє клас емоції у тексті (Додаток А.3).

Створюємо модель нейронної мережі для аналізу тексту з передачею матриці векторних представлень для використання значень з масиву векторного представлення для кожного слова, довжини вхідного тексту та кількості нейронів у шарі LSTM та прихованому шарі (Додаток А.4). Інформація про побудовану моделі нейронної мережі (Рисунок Б.1)

Для оцінки якості роботи нейронної мережі за значенням метрик F1, Precision та Recall створено словник (Додаток А.5), який буде прораховувати

значення кожної метрики за відповідними їм формулами (див. розділ 2.4) під час роботи нейронної мережі.

3.2 Результати роботи програми та тестування

У процесі навчання нейронної мережі на 84943 повідомлень та перевірці на 21236 повідомлень, найкращий отриманий результат для значення F1 складає 84.08% (Рисунок 3.1), що є гарним результатом відповідно до значення нижньої границі значення F1 у 77% [2].

f1 0.8408268733850129

	precision	recall	f1-score	support
neutral	0.89	0.94	0.91	6321
anger	0.32	0.20	0.25	118
disgust	0.17	0.23	0.20	47
fear	0.67	0.12	0.20	17
happiness	0.63	0.47	0.54	1019
sadness	0.42	0.25	0.31	102
surprise	0.54	0.47	0.51	116
accuracy			0.84	7740
macro avg	0.52	0.38	0.42	7740
weighted avg	0.83	0.84	0.83	7740

Рисунок 3.1 Статистика метрик на перевірочному наборі даних

Створена нейрона мережа передбачена для оцінки тональності висловлювань довжиною у 30 – 35 слів. У якості вхідного тексту можна використати текст довільної довжини, проте під час перетворення тексту у числове представлення масив чисел буде зазнавати змін в залежності від довжини самого тексту. Це зумовлено виконанням вимоги нейронної мережі відносно вхідних даних.

У випадку, коли текст коротший ніж 35 слів - масив числового представлення тексту доповнюється 0 для досягнення необхідної довжини. Як результат нейронна мережа аналізуючи велику кількість 0 класифікує текст як нейтральний оскільки 0 представляє відсутність будь-якої емоції.

У випадку коли текст довший ніж 35 слів – перші n зайвих слів відкидаються. Відкидання слів з початку тексту зумовлено особливістю нейронної мережі аналізувати останні отримані дані.

Розглянемо роботу нейронної мережі на прикладі двох текстів різної довжини на прикладі текстів зображених у (Таблиці 3.1)

Таблица 3.1 Тексти у символьному та числовому представленні

№	Текст	Текст у числовому представлені
1	The coronavirus COVID-19 pandemic is the defining global health crisis of our time.	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 11 97518 40461 23 11 17421 1648 692 3212 26 138]
2	The coronavirus COVID-19 pandemic is the defining global health crisis of our time and the greatest challenge we have faced since World War Two. Since its emergence in Asia late last year, the virus has spread to every continent except Antarctica.	[40461 23 11 17421 1648 692 3212 26 138 89 19 11 1564 1296 53 38 6818 425 178 1015 425 148 35056 22 3230 570 175 11 6163 126 2411 12 241 17869 1730]

Через невелику кількість слів у першому тексті масив числового представлення був доповнений 0 для досягнення необхідного розміру. Оскільки нейронна мережа розрахована на оцінку текстів у 30 – 40 слів, впевненість нейронної мережі щодо класифікації текстів меншої довжини знижується у результаті чого нейронна мережа відносить короткі висловлювання до нейтральних. Значення класифікації відносно кожного класу емоцій для тексту №1 наведені у Таблиці 3.2.

Таблиця 3.2 Класифікація тексту №1 нейронною мережею

neutral	anger	disgust	fear	happiness	sadness	surprise
0.913021	0.006313	0.00095	0.001618	0.075595	0.002057	0.000446

Рисунок 3.2 Діаграма класифікації тексту №1

Розглянемо роботу нейронної мережі на текстах з більшою кількістю слів, для цього візьмемо текст №2 з Таблиці 3.1 довжиною у 40 слів та оцінимо його нейронною мережею. Оскільки довжина тексту №2 близька до довжини

максимальної довжини очікуваного тексту, впевненість нейронної мережі щодо класів емоцій зростає за рахунок відсутності 0 . Значення відносно кожної емоції для тексту №2 наведені у Таблиці 3.3.

Таблиця 3.3 Класифікація тексту №2 нейронною мережею

neutral	anger	disgust	fear	happiness	sadness	surprise
0.000069	0.701039	0.026483	0.40867	0.144039	0.719647	0.000054

Рисунок 3. 3 Діаграма класифікації тексту №2

Для покращення роботи нейронної мережі можна виконати наступні дії:

- Збільшити кількість нейронів на у LSTM шарі та прихованому шарі нейронної мережі. Приклад отриманих результатів при меншій кількості нейронів зображено на рисунку Б.5.
- Корегування довжини вхідного тексту відносно середньої довжини усіх текстів у наборі тренувальному та тестовому наборі даних.
- Використати більшу кількість тренувальних даних приблизно однакової довжини.

Висновки

В даній роботі був проведений аналіз існуючих рішень та методів у одному з напрямів обробки природної мови – оцінки тональності тексту. На основі зібраної інформації було створено нейронну мережу типу LSTM для оцінки тональності тексту.

Програмна реалізація включає в себе створення та налаштування функцій для обробки тексту, завантаження та використання матриці векторних представлень слів, побудова моделі нейронної мережі та тренування з відстеженням точності роботи нейронної мережі на кожній епосі навчання. Історія тренування зображена на Рисунку Б.2.

Порівняно якість роботи нейронної мережі з різною кількістю нейронів на при оцінювання перевірочного набору даних (Рисунок Б.4, рисунок Б.5). Виконано навчання нейронної мережі з використанням іншої моделі векторних представлень слів. Результати роботи нейронної мережі з іншим векторним представленням зображені на Рисунку Б.6.

Реалізація нейронної мережі виконана на мові програмування Python 3.7.6. Для обробки даних та побудови нейронної мережі використані інструменти бібліотек keras, ekphrasis, tensorflow та sklearn.

Для навчання нейронної мережі було використано 84943 коротких текстів у якості тренувальних даних та додатково 21236 текстів для тестування.

Найкращий отриманий результат для моделі (Рисунок Б.1) складає 84%. Тестовий запуск та використання нейронної мережі описані в розділі 3.2. Варіанти покращення даної моделі надані в кінці розділу 3.2.

Список літератури

1. Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment.
2. Angelo Basile , Marc Franco-Salvador , Neha Pawar , Sanja Stajner , Mara Chinea Rios, Yassine Benajiba. 2019 SymantoResearch at SemEval-2019 Task 3: Combined Neural Models for Emotion Classification in Human-Chatbot Conversations
3. Christopher Olah Understanding LSTM Networks 2015 [Електронний ресурс] URL:https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (Дата звернення 15.04.2020).
4. J. Kaur and R. J. Saini, "Emotion Detection and Sentiment Analysis in Text Corpus: A Differential Study with Informal and Formal Writing Styles," International Journal of Computer Applications, розділ. 101, с. 1- 9, 09 2014
5. Jeffrey Pennington, Richard Socher, Christopher D. Manning. 2013.GloVe: Global Vectors for Word Representation
6. Mathieu Cliché. 2017. BB twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs
7. N.Buduma,Fundamentals of Deep Learning,Sebastopol:O‟Reilly Media,Inc, 2017.
8. P. Singhal and P. Bhattacharyya, "Sentiment Analysis and Deep Learning: A Survey," 2016.
9. Rosenblatt, Frank (1958), The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, c 386-408
10. Sayyed M Zahiri and Jinho D Choi. 2017. Emotion detection on tv show transcripts with sequencebased convolutional neural networks.
11. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean. 2017. Distributed Representations of Words and Phrases and their Compositionality

Додатки

Додаток А

Додаток А.1

Створення словника відповідності між символьними представленнями емоцій та їх значенням емоції у форматі ключ-значення.

```
emoji_symbolic_dictionary = {
    'angru': 'angry', "d-':':':-c': '<sad>',';-)': '<happy>',':-d': '<laugh>',':-(':
'<sad>',':-/': '<annoyed>','(^ ▪ ^)': '<happy>','=-d': '<happy>', ':'-)": '<happy>',
    '(○?○)': '<happy>',';-]': '<happy>',':\\/': '<sad>','d=<': '<annoyed>',
    '<annoyed>', ":'-(": '<sad>', ":-[": '<annoyed>','(^○^)': '<happy>',':-)': '<happy>',
    'xd': '<laugh>'}
```

Налаштування обробника тексту

```
text_processor = TextPreProcessor(
    normalize=['time','url','email','phone','user','date','percent','money','number'],
    annotate={"allcaps","repeated","hashtag", 'censored', "elongated", 'emphasis'},
    fix_html=True,      segmenter="twitter",unpack_hashtags=True,
unpack_contractions=True,
    corrector="twitter",
    spell_correct_elong=True,
    tokenizer=SocialTokenizer(lowercase=True).tokenize,
    dicts=[emoticons, emoji_symbolic_dictionary])

def text_tokenization(text):
    text = " ".join(text_processor.pre_process_doc(text))
    return text
```

Додаток А.2

Функція зчитування векторних представлень слів з файлу.

```
def getEmbeddings(file):
    embeddingsIndex = {}
    dimmension = 0
    with io.open(file, encoding="utf8", buffering = 2000000) as f:
        for line in f:
            values = line.split()
            word = values[0]
            embeddingVector = np.asarray(values[1:], dtype='float32')
            embeddingsIndex[word] = embeddingVector
            dimmension = len(embeddingVector)
    return embeddingsIndex, dimmension
```

Функція створення матриці векторних представлень.

```
def getEmbeddingMatrix(wordIndex, embeddings, dimmension):
    embeddingMatrix = np.zeros((len(wordIndex) + 1, dimmension))
    for word, i in wordIndex.items():
        embeddingMatrix[i] = embeddings.get(word)
    return embeddingMatrix
embeddings, dimmension = getEmbeddings('/content/drive/My Drive/Colab
Notebooks/glove.6B.300d.txt')
tokenizer = Tokenizer(filters='')
tokenizer.fit_on_texts([' '.join(list(embeddings.keys()))])
wordIndex = tokenizer.word_index
embeddings_matrix = getEmbeddingMatrix(wordIndex, embeddings,
dimmension)
X_train, X_val, y_train, y_val = train_test_split(texts_train, labels_train,
test_size=0.2, random_state=42)
```

Додаток А.3

Розбиття набору текстів та емоцій зі збереженням відповідності.

```
X_train, X_val, y_train, y_val = train_test_split(texts_train, labels_train,
test_size=0.2, random_state=42)
```

Приведення емоції до вигляду вектору

```
labels_categorical_train = to_categorical(np.asarray(y_train))
labels_categorical_val = to_categorical(np.asarray(y_val))
labels_categorical_dev = to_categorical(np.asarray(labels_dev))
```

Приведення тексту до розмірності у 35 слів (значення змінної MAX_SEQUENCE_LENGTH)

```
def get_sequances(texts, sequence_length):
    text = np.array(text)
message_first = pad_sequences(tokenizer.texts_to_sequences(texts.ravel()),
sequence_length)
message_first_message_train = get_sequances(X_train,
MAX_SEQUENCE_LENGTH)
message_first_message_val = get_sequances(X_val,
MAX_SEQUENCE_LENGTH)
message_first_message_dev = get_sequances(texts_dev,
MAX_SEQUENCE_LENGTH)
```

Додаток А.4

Функція створення нейронноъ мережі

```
def modelContruction(embeddings_matrix, sequence_length, lstm_dimmension,
hidden_layer_dimmension, num_classes,
            noise=0.1, dropout_lstm=0.2, dropout=0.2):
    turn1_input = Input(shape=(sequence_length,), dtype='int32')
    embedding_dimmension = embeddings_matrix.shape[1]
    embeddingLayer = Embedding(embeddings_matrix.shape[0],
embedding_dimmension,  weights=[embeddings_matrix],
input_length=sequence_length, trainable=False)
    turn1_branch = embeddingLayer(turn1_input)
    turn1_branch = GaussianNoise(noise, input_shape=(None, sequence_length,
embedding_dimmension))(turn1_branch)
    lstm1 = Bidirectional(LSTM(lstm_dimmension, dropout=dropout_lstm))
    turn1_branch = lstm1(turn1_branch)
    x = Dropout(dropout)(turn1_branch)
    x = Dense(hidden_layer_dimmension, activation='relu')(x)
    output = Dense(num_classes, activation='softmax')(x)
    model = Model(inputs=turn1_input, outputs=output)
    model.compile(loss='categorical_crossentropy', optimizer='rmsprop',
metrics=['acc'])
    return model

model = modelContruction(embeddings_matrix, MAX_SEQUENCE_LENGTH,
lstm_dimmension=64, hidden_layer_dimmension=64, num_classes=7)
model.summary()
```

Додаток А.5

```
metrics = {"f1_e": (lambda y_test, y_pred: f1_score(y_test, y_pred,
average='micro', labels=[emotion2label["neutral"], emotion2label["anger"],
emotion2label["disgust"], emotion2label["fear"], emotion2label["happiness"],
emotion2label["sadness"],emotion2label["surprise"]])),
   "precision_e": (lambda y_test, y_pred: precision_score(y_test, y_pred,
average='micro', labels=[emotion2label["neutral"], emotion2label["anger"],
emotion2label["disgust"],emotion2label["fear"],emotion2label["happiness"],
emotion2label["sadness"],emotion2label["surprise"]])),
   "recoll_e": (lambda y_test, y_pred: recall_score(y_test, y_pred, average='micro',
labels=[emotion2label["neutral"], emotion2label["anger"],
emotion2label["disgust"], emotion2label["fear"], emotion2label["happiness"],
emotion2label["sadness"],emotion2label["surprise"]])),}
_datasets = {}
_datasets["dev"] = [[message_first_message_dev],
                    np.array(labels_categorical_dev)]
_datasets["val"] = [[message_first_message_val],
                    np.array(labels_categorical_val)]

metrics_callback = MetricsCallback(datasets=_datasets, metrics=metrics)
checkpoint = ModelCheckpoint(filepath, monitor='val_acc', save_best_only=True,
save_weights_only=False,mode='auto', period=1)
tensorboardCallback = TensorBoard(log_dir='/content/drive/My Drive/Colab
Notebooks/Graph', histogram_freq=0, write_graph=True, write_images=True)
validation_data=(message_first_message_val,np.array(labels_categorical_val))
```

Додаток Б

У додатку наведені інформація щодо нейронної мережі, а також різниці використання різної кількості нейронів та інших векторних представлень .

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	(None, 35)	0
embedding_2 (Embedding)	(None, 35, 300)	197439000
gaussian_noise_2 (GaussianNo	(None, 35, 300)	0
bidirectional_2 (Bidirection	(None, 128)	186880
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 64)	8256
dense_4 (Dense)	(None, 7)	455

Total params: 197,634,591
Trainable params: 195,591
Non-trainable params: 197,439,000

Рисунок Б.1 Інформація про будову нейронної моделі

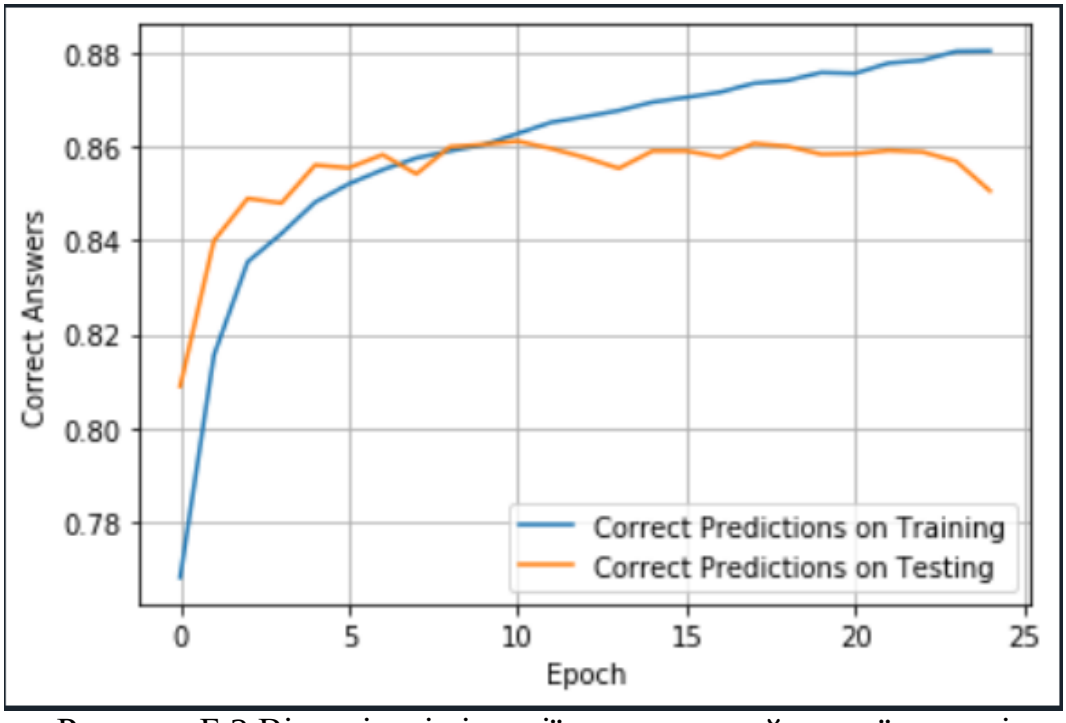


Рисунок Б.2 Візуалізація історії навчання нейронної мережі

f1 0.8408268733850129

	precision	recall	f1-score	support
neutral	0.89	0.94	0.91	6321
anger	0.32	0.20	0.25	118
disgust	0.17	0.23	0.20	47
fear	0.67	0.12	0.20	17
happiness	0.63	0.47	0.54	1019
sadness	0.42	0.25	0.31	102
surprise	0.54	0.47	0.51	116
accuracy			0.84	7740
macro avg	0.52	0.38	0.42	7740
weighted avg	0.83	0.84	0.83	7740

Рисунок Б.4 Результати метрик у фінальній моделі нейронної мережі (64 нейрони на прихованому шарі)

f1_e 0.8035974720466699

	precision	recall	f1-score	support
neutral	0.83	0.95	0.89	6321
anger	0.76	0.14	0.23	317
disgust	0.31	0.05	0.08	85
fear	0.57	0.02	0.05	161
happiness	0.64	0.47	0.55	1019
sadness	0.28	0.02	0.04	209
surprise	0.52	0.46	0.49	116
accuracy			0.80	8228
macro avg	0.56	0.30	0.33	8228
weighted avg	0.77	0.80	0.77	8228

Рисунок Б.5 Результати метрик у моделі нейронної мережі з меншою кількістю нейронів (30 нейронами на прихованому шарі)

f1_e 0.8391472868217055

	precision	recall	f1-score	support
neutral	0.89	0.93	0.91	6321
anger	0.36	0.17	0.23	118
disgust	0.30	0.15	0.20	47
fear	0.43	0.18	0.25	17
happiness	0.61	0.50	0.55	1019
sadness	0.43	0.25	0.32	102
surprise	0.38	0.63	0.47	116
accuracy			0.84	7740
macro avg	0.49	0.40	0.42	7740
weighted avg	0.83	0.84	0.83	7740

Рисунок Б.6 Результати метрик при використанні моделі GloVe