Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мережевих систем факультету інформатики

Сучасні підходи використання нейронних мереж в аналізі текстів

Курсова робота

за спеціальністю 121 «Інженерія програмного забезпечення»

Керівник курсової роботи

Глибовець А. М.

(*прізвище та ініціали*)

_____ (*підпис*)

«____» _____ 2025 р.

Виконала студентка БП ІПЗ-3 Реуцька А. О.

(*прізвище та ініціали*)

«____» _____ 2025 р.

Зміст

Анотація

У роботі досліджено основні архітектури нейронних мереж, які застосовуються в задачах обробки природної мови (NLP). Метою роботи було проаналізувати переваги, недоліки та особливості таких архітектур, як RNN, LSTM, CNN, трансформери та гібридні моделі. Для досягнення мети було вивчено принципи роботи кожної архітектури, її здатність обробляти контекст, а також відповідні приклади застосування. Результатом дослідження стала реалізація моделей на основі кожної архітектури для виконання задачі аналізу емоційності слів та тексту, а також було проведено детальний аналіз результатів, який дозволяє краще зрозуміти, яку архітектуру доцільно застосовувати для конкретних NLP-завдань. Робота також акцентує увагу на сучасних тенденціях у розвитку нейронних мереж і ролі гібридних підходів для підвищення якості обробки текстової інформації.

Вступ

В теперішній час, поки обсяги текстової інформації стрімко зростають, з’являється й потреба в автоматизованих інструментах, що допомагають проводити обробки та аналізи. Всі інформаційні джерела, такі як соціальні мережі, новинні портали, електронні книги, документи – вони генерують великі обсяги текстів, що містять нові та важливі знання, тенденції, або ідеї, що можуть повпливати на прийняття критичних рішень. Відповідно ручне оброблення таких даних є неефективним способом, тож наразі спостерігається розвиток комп’ютерних систем, здатних до самостійного аналізу текстів.

Обробка природної мови (англ. Natural Language Processing, NLP) є однією з фундаментальних галузей в області штучного інтелекту, що розробляє алгоритми та моделі для «розуміння» тексту та мови людини комп’ютерами. Раніше для цього використовувалися методи з чітко заданими правилами та статистичними підходами, але подальший розвиток глибокого навчання, особливо нейронних мереж, посприяв великим змінам в даній сфері.

Важливість цієї теми полягає в широкому практичному застосуванні методів аналізу тексту: в бізнес-аналітиці, медіа, освіті, праві, інформаційній безпеці та електронній комерції. Організації та компанії по всьому світу все частіше впроваджують інструменти аналізу тексту на основі нейронних мереж, щоб покращити якість обслуговування клієнтів, автоматизувати рутинні завдання та отримати нові знання з поточних даних. Тим часом розробники відповідно продовжують удосконалювати архітектури моделей, інтегрувати їх та оптимізувати для різних мов і застосувань.

Метою даної курсової роботи є дослідження сучасних архітектур нейронних мереж, розглянути як вони застосовуються в аналізах текстів,

порівняти їхні можливості, переваги та недоліки, а також продемонструвати приклади їх використання на практиці. Для досягнення цієї мети було розглянуто як традиційні, так і ті, що набирають зараз популярність, зокрема: рекурентні нейронні мережі (RNN) та їх модифікації (LSTM, GRU), які здатні обробляти послідовності слів у реченні, зберігаючи контекст; згорткові нейронні мережі (CNN), що успішно застосовуються для виявлення локальних шаблонів у тексті; архітектура трансформерів, яка на сьогодні є провідною у більшості NLP-завдань завдяки паралельній обробці; гібридні моделі, що поєднують переваги декількох підходів для досягнення ще кращих результатів.

Основи NLP та машинного навчання:

У першому розділі розглянуто базові поняття Natural Language Processing й машинного навчання, та описано, як еволюціонували методи обробки природної мови. Пояснено сучасні підходи, що поєднують лінгвістику, штучний інтелект та новітні архітектури для підвищення точності обробки мовлення, зокрема в умовах малоресурсних мов, таких як українська.

Аналіз текстів:

Другий розділ розглядає основні етапи та методи аналізу текстів у рамках NLP: від попередньої обробки до представлення тексту у вигляді векторів та вирішення прикладних задач, таких як класифікація, аналіз емоцій, розпізнавання сутностей та тематичне моделювання.

Основні архітектури нейронних мереж:

У третьому розділі представлено ключові архітектури нейронних мереж, що використовуються в NLP. Розглянуто переваги та обмеження RNN, удосконалення у вигляді LSTM та GRU, а також застосування CNN для виявлення локальних ознак у тексті. Окрему увагу приділено

трансформерам як найсучаснішому підходу до моделювання контексту, а також гібридним архітектурам, які поєднують сильні сторони різних моделей для покращення результатів у складних завданнях обробки мови.

Розробка моделей на основі різних архітектур для бінарного сентимент-аналізу українських слів:

У четвертому розділі описується практичне застосування цих моделей на реальній задачі – емоційний аналіз тексту. Наведено повний список інструментів, що використовувалися під час дослідження, описано процес роботи, здійснено порівняльний аналіз результатів та зроблено висновки щодо їх ефективності та обчислювальної складності.

1. Основи NLP та машинного навчання

Natural Language Processing (NLP) є ключовою галуззю штучного інтелекту, яка займається дослідженням та розробкою алгоритмів, що дозволяють комп'ютерам працювати з природною мовою, тобто мовою, якою спілкуються люди. Метою NLP є створення систем, здатних автоматично аналізувати, розуміти, інтерпретувати та навіть генерувати текст чи усне мовлення на основі тих самих принципів, які використовує людина під час мовленнєвої діяльності. [1] Застосування технологій NLP є широким та доволі популярним, як-от пошукові системи, машинний переклад, віртуальниі асистенти, а також аналіз емоційності тексту та перевірка правопису.

NLP являє собою багаторівневу систему, що включає фонетичний, морфологічний, синтаксичний, семантичний, прагматичний і дискурсивний рівні обробки. [2] Це означає, що аналіз тексту не обмежується лише словами чи реченнями, оскільки система повинна враховувати контекст, цілі комунікації та міжтекстові зв'язки. Як приклад можна навести звичайний діалог з користувачем, де важливо не лише розпізнати запит, а й побудувати логічну відповідь, яка буде релевантною та опиратися на попередню історію діалогу.

Традиційно задачі NLP вирішувалися за допомогою експертних систем і ручного кодування правил. Наприклад, для розпізнавання частин мови застосовувалися граматичні правила, які вручну укладалися лінгвістами. Проте цей підхід мав серйозні обмеження, бо він не враховував контексту, був трудомістким у підтримці та погано масштабувався для великих обсягів текстів або нових мов. [3] Все змінилося з приходом машинного навчання, оскільки воно дозволяє моделям автоматично вивчати закономірності з великих корпусів тексту без явного програмування

правил. Одним із перших ефективних методів був Naive Bayes, цей метод наразі є одним з найпопулярніших алгоритмів машинного навчання через свою простоту, легку інтерпретацію та ефективність. [5] Згодом поширилися методи підтримуючих векторних машин (SVM), логістична регресія та дерева рішень. [4]

Можна вважати, що великий поштовх в розвитку NLP стався завдяки появі глибоких нейронних мереж. Першими ефективними моделями були рекурентні нейронні мережі, що дозволяли обробляти послідовності тексту, згодом їх вдосконалили за допомогою LSTM та GRU. Однак саме трансформери змінили парадигму обробки мови. Вони забезпечили ефективну обробку залежностей у тексті незалежно від позиції слів, завдяки механізму self-attention. [6] На основі трансформерів були створені моделі, які стали стандартом у багатьох NLP-задачах, серед них BERT [7], GPT, RoBERTa. Їхня ефективність пояснюється можливістю попереднього навчання на величезних корпусах і подальшого адаптивного донавчання на конкретних прикладних задачах, що значно зменшило потребу у великих розмітках для кожного нового проєкту.

У більш свіжих дослідженнях з NLP спостерігається поєднання традиційної лінгвістики, нейронних моделей та інших галузей штучного інтелекту. Наприклад, активно розвивається мультимодальне NLP, що уявляє собою поєднання тексту з візуальною чи аудіоінформацією. Крім того, важливою тенденцією стало використання knowledge graphs – це онтологічні баз знань, які допомагають моделі краще інтерпретувати фактичні зв’язки. Zero-shot та few-shot learning є ще одними напрямки, які стали привертати значну увагу. [6] Ці підходи дозволяють системам NLP виконувати нові завдання без попереднього донавчання або з мінімальною кількістю прикладів, що значно підвищує універсальність моделей і

розширює їхнє застосування, зокрема для малоресурсних мов, таких як українська.

Підсумовуючи цей розділ, сучасна обробка природної мови є складним, багаторівневим процесом, який активно інтегрується з машинним навчанням, знаннями про світ, логічним міркуванням та навіть зображеннями й звуком. Із суто формального аналізу тексту NLP перетворилася на складну когнітивну інженерію, здатну моделювати людиноподібне розуміння мови.

2. Аналіз текстів

2.1 Визначення та значення аналізу текстів

Аналіз текстів є одним з основних та поширених завдань рамках NLP, оскільки це дозволяє комп'ютерам видобувати сенс та ідеї з неструктурованої природної мови. Мова йде саме про процеси, що дозволяють системам автоматично інтерпретувати, класифікувати, організовувати, узагальнювати або навіть створювати текстові дані. Застосування аналізу текстів охоплює багато сфер, як-от автоматична фільтрація спаму, виявлення емоцій у повідомленнях користувачів, аналіз публічної думки в соціальних мережах, навіть звичайний автоматичний переклад. Наразі в світі, який вже давно став цифровим, і де обсяги текстової інформації зростають кожну хвилину, такі методи стають критично необхідними для перетворення необроблених текстів на структуровану й аналітично корисну інформацію.

2.2 Попередня обробка текстів

Перш ніж модель зможе ефективно працювати з текстовими даними, їх потрібно підготувати, тобто здійснити попередню обробку. Цей етап є фундаментальним і водночас часто недооціненим, хоча він безпосередньо впливає на результати моделювання. [8] Підготовка тексту передбачає кілька послідовних трансформацій. В першу чергу це очищення тексту, що включає в собі видалення зайвих символів, HTML-тегів, чисел, спеціальних знаків тощо. Часто це все призводиться до нижнього регістру. Наприклад, речення «Найкраща ЦІНА пРосто ЗАРАЗ!!!» перетворюється після очищення на «найкраща ціна просто зараз». Далі проводиться токенізація – це розбиття тексту на базові одиниці, зазвичай слова або речення. У класичних підходах для англійської мови токенізація реалізується за пробілами й розділовими знаками, однак у більш складних

мовах, зокрема в українській, вона вимагає урахування морфології й синтаксису.

Після токенізації часто виконується лематизація або стемінг. У випадку лематизації слова зводяться до своєї граматичної основи (леми), наприклад, «пішов», «йду», «підеш» зводяться до дієслова «йти». Це дозволяє агрегувати всі варіанти одного й того ж поняття. Стемінг, натомість, є грубішим методом, оскільки він просто обрізає слова до їх «стебел», що може бути менш точним, але швидшим. Інколи такий підхід може спотворювати зміст через своє агресивне скорочення. [8] Для української мови обидва підходи потребують складніших алгоритмів через багату морфологію.

Крім того, під час обробки видаляються так звані «стоп-слова». Це службові частини мови, які зустрічаються дуже часто, але зазвичай не несуть смислового навантаження в контексті машинного аналізу, такі як «і», «що», «але». Цей етап знижує розмірність векторного простору та зменшує вплив слів, що не мають вагомого значення для семантичного аналізу. Проте в задачах аналізу стилю, сарказму або граматичних залежностей стоп-слова можуть бути інформативними, тому їх видалення є опціональним та дуже залежить від контексту задачі. [8]

2.3 Представлення тексту у числовому вигляді

Після попередньої обробки виникає необхідність перетворити текст на числове подання — вектор, з яким можуть працювати алгоритми машинного навчання. Одним із найстаріших і водночас найпростіших підходів є модель «мішка слів» (Bag of Words, BoW), у якій текст представлений у вигляді набору слів без урахування порядку, а інформація подається у вигляді частот появи тих чи інших термінів у документі. Цей метод, хоча і доволі грубий, часто демонструє прийнятні результати на

простих задачах класифікації. Більш вдосконаленою версією BoW є TF-IDF (Term Frequency-Inverse Document Frequency), яка зважує частоти слів у конкретному документі, враховуючи, наскільки ці слова унікальні або поширені в усьому корпусі. Це дозволяє «підсилити» рідкісні, але потенційно значущі слова, і «послабити» загальновживані. [9]

У більш нових підходах набули популярності словникові вектори або embeddings, такі як Word2Vec [10], GloVe [11], fastText [12]. Ці моделі дозволяють представляти слова як щільні багатовимірні вектори, які відображають їх семантичну подібність. Наприклад, слова «король» і «королева» будуть розташовані близько в такому векторному просторі. Для багатьох мов, включно з українською, вже існують попередньо навчені embedding-моделі, як-от fastText або українські вектори від lang-uk. [13].

2.4 Основні задачі аналізу тексту

Після перетворення тексту в числову форму відкривається простір для вирішення численних задач. Однією з найпоширеніших є класифікація текстів. Це задача, в якій системі потрібно навчитися розподіляти текст по певних категоріях, наприклад, розпізнавати тематику статті чи новин (спорт, політика, технології) або виявляти мову тексту. Для таких задач добре працюють як класичні алгоритми на кшталт логістичної регресії чи SVM, так і сучасні нейронні моделі. [14]

Ще одним важливим напрямом є аналіз емоційності тексту (sentiment analysis). У цьому випадку система має виявити, чи є повідомлення позитивним, негативним або нейтральним. Подібні методи широко застосовуються в аналізі відгуків на товари, коментарів у соцмережах або реакцій на політичні події. В українському контексті ця задача

ускладнюється відсутністю великих валідаційних корпусів, проте існують зусилля з розробки відповідних датасетів.

Завдання розпізнавання іменованих сутностей, або Named Entity Recognition полягає в тому, щоб ідентифікувати в тексті конкретні категорії об’єктів: імена людей, назви компаній, географічні локації, дати тощо. Це важливо для автоматичної побудови баз знань або інформаційного пошуку. Нейромережеві підходи, зокрема, моделі на основі Bi-LSTM та CRF, виявилися надзвичайно ефективними в цій галузі. [15]

Іншим корисним методом є тематичне моделювання (topic modeling), яке дозволяє автоматично виявляти основні теми в корпусі документів. Один з найвідоміших алгоритмів Latent Dirichlet Allocation (LDA) дозволяє без попереднього маркування виявити, про що «говорить» текст, тобто структурувати великі обсяги інформації за змістовими групами. [16] Це особливо актуально для аналізу новин, академічних статей або звернень громадян.

2.5 Особливості аналізу української мови

Аналіз текстів українською мовою в рамках NLP становить окремий науково-практичний інтерес через низку лінгвістичних та технічних особливостей. Хоча загальні підходи до обробки природної мови (як-от токенізація, лематизація, векторизація) залишаються подібними для всіх мов, для української вони вимагають певної адаптації через складну морфологію, багатий словотвір і граматичну гнучкість.

Однією з головних складностей є багатоформність слів, притаманна флективним мовам. Українська мова має складну систему відмінювання іменників, прикметників і дієслів. Це створює додаткове навантаження на

алгоритми лематизації та стемінгу: неправильне зведення слова до базової форми може істотно вплинути на якість подальшої обробки. Бібліотеки, такі як Stanza від Stanford NLP [17], вже підтримують українську мову на базовому рівні, надаючи модулі для токенізації, лематизації, POS-аналізу, однак якість часто поступається моделям, створеним для англійської.

Ще одним викликом є нестача великих анотованих корпусів текстів українською. Для навчання моделей глибокого навчання потрібні великі обсяги даних: це мільйони речень із розміченими частинами мови, іменованими сутностями, залежностями тощо. Для англійської мови існують такі стандарти, як Penn Treebank або OntoNotes, а для української — подібних масштабів набагато менше. Проте останніми роками з’являються ініціативи зі створення відкритих ресурсів: зокрема, Universal Dependencies надає корпуси з базовою синтаксичною розміткою, включаючи й українську мову. [18]

Водночас існують і позитивні зрушення. Проєкти на кшталт Lang-uk працюють над розвитком українських лінгвістичних ресурсів, зокрема лематизаторів, словників, корпусів. Lang-uk також розробляють моделі трансформерів, адаптовані під українську, наприклад як Ukrainian BERT – multi-cased base BERT модель, розроблена Сергієм Тютюнником. [13]

Підсумовуючи цей розділ, можна зазначити, що аналіз українськомовних текстів активно розвивається, хоча й потребує додаткових ресурсів та досліджень. З технічного боку він вимагає адаптації інструментів під морфологічні особливості мови, а також розширення корпусної бази для навчання і тестування моделей. З практичного боку українська мова поступово інтегрується в міжнародну екосистему NLP завдяки відкритим ініціативам, підтримці дослідників та появі спеціалізованих інструментів.

3. Основні архітектури нейронних мереж

3.1. Рекурентні нейронні мережі

Рекурентні нейронні мережі (RNN) стали однією з перших архітектур, які спробували враховувати природу послідовних даних. Їх ключовою особливістю є здатність зберігати інформацію про попередні елементи входу через внутрішні стани, які передаються від кроку до кроку. Цей механізм робить RNN цінними для завдань, у яких контекст і порядок слів мають значення: мовне моделювання, синтез тексту, розпізнавання мови, машинний переклад. Однак практичне застосування RNN виявило суттєві проблеми. Найбільш критичною є проблема затухання або вибуху градієнтів (vanishing/exploding gradients), яка унеможливлює ефективне навчання на довгих послідовностях. У таких випадках RNN забувають раніший контекст або, навпаки, накопичують нестійкі значення у градієнтах під час зворотного поширення помилки. Це призводило до втрати релевантної інформації при обробці текстів довжиною понад кілька речень. [19]

3.2. LSTM та GRU

Щоб вирішити проблеми стандартних RNN, були розроблені вдосконалені архітектури, такі як довготривала короткочасна пам'ять (LSTM) та ґейтові рекурентні одиниці (GRU). LSTM вводить механізми «вхідних», «забувальних» та «вихідних» ґейтів, які контролюють потік інформації через мережу. Це дозволяє ефективно зберігати та використовувати контекст на довгих відстанях у тексті. GRU є спрощеною версією LSTM з меншою кількістю ґейтів, що зменшує обчислювальні витрати при збереженні подібної ефективності. Обидві архітектури широко використовуються в NLP для завдань, що вимагають розуміння контексту, таких як аналіз тональності, розпізнавання іменованих сутностей та машинний переклад. [6]

3.3. Згорткові нейронні мережі

Зазвичай згорткові нейронні мережі (CNN) традиційно асоціюються з обробкою зображень, але вони також ефективно застосовуються до текстових даних. У контексті NLP CNN використовуються для виявлення локальних шаблонів у тексті, таких як фрази або словосполучення, незалежно від їхнього положення в реченні. CNN обробляють текст, застосовуючи ядра згортки до векторизованого представлення слів, що дозволяє виявляти ключові особливості для класифікації або інших завдань. Перевагами CNN є їхня здатність до паралельної обробки та ефективність у навчанні, що робить їх популярними для задач, таких як класифікація текстів та аналіз тональності.

3.4. Трансформери

Архітектура трансформерів, представлена у 2017 році, стала революційною в NLP. [20] Як зазначалося в розділі 1, основною інновацією трансформерів є механізм self-attention. Це являє собою концепцію, що дозволяє моделям оцінювати значення різних слів у послідовності в динаміці. Такий підхід усуває обмеження RNN, полегшуючи розпаралелювання та підвищуючи ефективність навчання. [6] Моделі, засновані на трансформерах, такі як BERT, GPT та їхні варіації, досягли видатних результатів у багатьох завданнях NLP, включаючи класифікацію текстів та машинний переклад.

3.5. Гібридні архітектури

Останні дослідження у сфері NLP демонструють ефективність комбінування кількох типів мереж для досягнення кращої продуктивності. Однією з популярних гібридних архітектур є BiLSTM-CNN — модель, яка об’єднує двосторонню рекурентну нейронну мережу з CNN. У такій архітектурі BiLSTM (Bidirectional LSTM) спочатку проходить текст у двох

напрямках — зліва направо і справа наліво — що дозволяє врахувати як попередній, так і майбутній контекст слова. Далі CNN застосовується до послідовностей станів BiLSTM, виділяючи найбільш інформативні локальні шаблони. Це дає змогу поєднувати глибоке контекстуальне розуміння з ефективним витягуванням ознак.

4. Розробка моделей на основі різних архітектур для бінарного емоційного аналізу українських слів та тексту

4.1 Огляд інструментів для реалізації

В цьому підрозділі розглянемо інструменти, які були використані під час виконання практичної частини роботи. Основна мова програмування, яка була задіяна – Python, оскільки це високорівнева мова з відкритим кодом, що має широку підтримку в науковому середовищі та спільноті штучного інтелекту. Python відзначається простим синтаксисом, гнучкістю та наявністю великої кількості бібліотек для машинного навчання, роботи з текстом та візуалізації даних. [21]

Для реалізації та виконання коду було використано інтерактивне середовище Google Colaboratory, яке є безкоштовною хмарною платформою від Google. [22] Воно дозволяє запускати та редагувати Jupyter Notebook без потреби встановлення локального середовища та є доволі популярним місцем роботи для тих, хто спеціалізується на машинному навчанні та Data Science. Colab підтримує апаратне прискорення за допомогою GPU або TPU, що істотно пришвидшує тренування моделей глибокого навчання. Ця платформа також забезпечує доступ до Google Drive, що спрощує збереження та завантаження наборів даних.

Уся модельна частина реалізована з використанням TensorFlow, що є популярним open-source фреймворком для побудови моделей машинного та глибокого навчання. [23] Було використано високорівневий API Keras, який є частиною TensorFlow та спрощує побудову нейронних мереж завдяки інтуїтивному синтаксису. Keras дозволяє легко реалізовувати складні моделі завдяки декларативному синтаксису.

Для роботи з передтренованою трансформерною моделлю було використано бібліотеку Transformers. [24] Вона забезпечує простий інтерфейс для тренування моделей, а також їх оцінки. Також бібліотека надала доступ до токенізатора, а також таких інструментів, як AutoTokenizer, AutoModelForSequenceClassification, Trainer, TrainingArguments та DataCollatorWithPadding. Ця бібліотека дозволяє швидко адаптувати потужні передтреновані моделі під свої завдання з мінімальними зусиллями. Із бібліотеки datasets від Hugging Face було використано клас Dataset, який дозволив швидко створити навчальний і тестовий набори з уже токенізованих текстів і міток. [25] Це спростило інтеграцію з Trainer, що автоматизував процес навчання, логування і збереження моделей.

Бібліотека pandas застосовувалась для зчитування CSV-файлів, фільтрації даних, перейменування колонок та базової обробки датафреймів. Для токенізації текстів і створення послідовностей слів застосовувалась функціональність із tensorflow.keras.preprocessing.text.Tokenizer, яка дозволяє перетворити сирий текст у числові представлення, необхідні для введення у нейронні мережі. Для машинного навчання та попередньої обробки міток також було використано scikit-learn, цю бібліотеку зазвичай використовують для класичних ML-завдань. Зокрема, було використано LabelEncoder для кодування текстових міток у числові, train_test_split для розбиття на навчальну і тестову вибірки, а також classification_report і confusion_matrix для оцінки якості класифікації.

Для візуалізації результатів тренування моделей (точність та функція втрат) використовувалась бібліотека matplotlib, яка дозволяє створювати графіки у пітонівському середовищі. Це дало змогу наочно оцінити процес навчання та переобучення моделей. А також застосовувалась бібліотека

seaborn для побудови теплових карток матриць плутанини, що надало візуальне уявлення про класифікаційні помилки моделей.

Бібліотека NumPy забезпечувала ефективну обробку масивів і векторів, що є основою для роботи з тензорами, прогнозами моделей та маніпуляціями з даними.

4.2 Процес виконання

Для початку порівняння ефективності п’ятьох різних архітектур нейронних мереж розроблено моделі для більш елементарної задачі на бінарну класифікацію емоційного забарвлення українських слів: позитивного або негативного. У ході цього експерименту досліджується якість класифікації кожної з моделей та їх здатність узагальнювати лінгвістичні патерни в україномовному корпусі.

Для навчання моделей було використано український датасет sentiment_ua.csv, який містить окремі слова з відповідною міткою емоційної полярності: -1 для негативних і 1 для позитивних. [26] Цей датасет є вдалим вибором, оскільки дозволяє моделі навчитися розрізняти тональні особливості коротких текстових одиниць, що є складнішою задачею, ніж класифікація повних речень, через відсутність контексту. Перед обробкою було проведено фільтрацію даних: очищено від пропусків та перетворено числові мітки на текстові (positive/negative). Усього в наборі даних залишилося 5747 зразків. Перед подачею на моделі дані було токенізовано за допомогою Tokenizer з параметром num_words=10000 для обмеження словника і включення спеціального токена <OOV> для невідомих слів. Кожне слово було перетворено в числову послідовність, а потім доповнено або обрізано до фіксованої довжини 10 токенів (maxlen=10), що дозволяє уніфікувати вхідний розмір для нейронної

мережі. Мітки (labels) були закодовані за допомогою LabelEncoder, після чого перетворені в числові значення (0 для negative, 1 для positive).

RNN-модель реалізована через стек із трьох основних шарів: Embedding для навчання векторних уявлень слів, SimpleRNN з 32 нейронами для збереження послідовної інформації та двох Dense-шарів: один проміжний та один вихідний з softmax-активацією. Модель навчалась 30 епох із розміром пакету 16, з використанням sparse_categorical_crossentropy як функції втрат. LSTM є вдосконаленням RNN, яке краще працює з довгими залежностями та розв’язує проблему згасаючого градієнта. Архітектура аналогічна до RNN, однак використовується LSTM-шар замість SimpleRNN. CNN, хоч і не має рекурентної природи, здатна ефективно витягати локальні патерни у тексті. У моделі використовується Conv1D з 128 фільтрами та фільтром розміру 3, після чого застосовується глобальне підсумовування (GlobalMaxPooling1D) і повнозв'язні шари.

Далі була реалізована гібридна модель, що поєднує можливості BiLSTM та CNN. У цій архітектурі після шару Embedding використовується Bidirectional LSTM, що дозволяє обробляти послідовність в обох напрямках та захоплювати як попередній, так і наступний контекст. Далі застосовується згортковий шар Conv1D, який дозволяє моделі фіксувати локальні шаблони у тексті після обробки довгих залежностей. Результат проходить через GlobalMaxPooling1D, а потім через два повнозв'язні Dense-шари. Така архітектура забезпечує гнучке поєднання глобального контексту й локальних ознак, що позитивно впливає на якість класифікації.

Наостанок було проведено експеримент з використанням трансформерної моделі xlm-roberta-base, що є багатомовною версією RoBERTa, попередньо натренованою на величезних корпусах текстів з

різних мов, включаючи українську. [27] Цей підхід є значно потужнішим, оскільки трансформери використовують вже зазначений тут механізм self-attention. А для підготовки даних використовувався токенізатор, сумісний з моделлю (AutoTokenizer), який перетворював слова у формат, зручний для вхідного шару трансформера. Навчання проводилося за допомогою бібліотеки Transformers від HuggingFace, використовуючи клас Trainer. Було визначено гіперпараметри: 3 епохи, розмір батчу 16, швидкість навчання 2e-5, та вагове згасання.

Моделі тестувались на контрольному наборі слів: ["любов", "війна", "успіх", "горе", "надія", "страх", "щастя", "депресія"]. Результати прогнозів свідчать про узгодженість у роботі всіх моделей, хоча деякі слова, як-от "успіх" або "страх", класифікувались по-різному в залежності від моделі, що свідчить про наявність нюансів у побудові векторних уявлень та логіки класифікації. Також моделі були протестовані на оцінку емоцій двох речень, окрім моделі на основі трансформеру, бо перевірялася на повноцінних абзацах, що дозволило підтвердити її здатність обробляти складні контекстуальні структури та зберігати стабільну якість класифікації при зміні масштабу вхідних даних.

```
full_texts = [
    "Сьогодні був дуже важкий день, все навалилось одночасно, і я почуваюсь спустошеною.",
    "Мене переповнює радість і вдячність за підтримку моїх близьких у важкий час.",
    "Я отримала підвищення на роботі і відчуваю себе на вершині світу.",
    "Ніч була жахливою, я не могла заснути від тривоги і постійних думок.",
    "День пройшов чудово: хороша погода, прогулянка з друзями, смачна кава та гарний настрій."
]
test_words_with_transformer(transformer_model, transformer_tokenizer, full_texts)
```

```
Сьогодні був дуже важкий день, все навалилось одночасно, і я почуваюсь спустошеною.: negative
Мене переповнює радість і вдячність за підтримку моїх близьких у важкий час.: positive
Я отримала підвищення на роботі і відчуваю себе на вершині світу.: positive
Ніч була жахливою, я не могла заснути від тривоги і постійних думок.: negative
День пройшов чудово: хороша погода, прогулянка з друзями, смачна кава та гарний настрій.: positive
```

Рис. 4.2.1 - Результати тестування моделі-трансформера на повноцінних реченнях

4.3 Порівняльний аналіз результатів

Починаючи з тренувальних метрик, можна відзначити, що класичні архітектури нейромереж, RNN, LSTM та CNN, продемонстрували доволі високі значення точності на тренувальних даних. Найвищий показник точності продемонструвала LSTM з 93.04%, слідом йде CNN з 91.97%, а RNN замкнула трійку з результатом 87.80%. Але при цьому спостерігається цікава динаміка, якщо звернути увагу на статистику за втратами. LSTM мала найнижчий відсоток втрат серед цих трьох моделей, лише 29.95%, що свідчить про стабільне навчання та ефективне засвоєння патернів у даних. CNN виявилася лише трохи менш ефективною, оскільки її тренувальний відсоток втрат склав 29.34%, але при цьому вона забезпечила вражаюче збалансовану продуктивність на обох етапах. RNN, навпаки, мала значно вищий відсоток 109.28%, що вказує на складнощі в процесі оптимізації та потенційні проблеми з узагальненням.

Під час навчання моделі, що поєднує CNN та BiLSTM, її тренувальна точність склала 91.12%, що виглядає конкурентно, але її відсоток втрат був так само доволі високим (104.61%), близьким до результату RNN. Це може свідчити про те, що хоч модель здатна до навчання, вона менш стабільно сходиться через ускладнену архітектуру. Що стосується трансформера, модель тренувалася лише три епохи, проте досягла середнього відсотку 51.09%. Тренувальна точність безпосередньо не виводилася, але навіть таке відношення втрат вказує на задовільне освоєння навчального корпусу. Однак, через суттєві відмінності у метриках для трансформера, пряме порівняння з іншими моделями тут є лише умовним.

Перейшовши до тестової продуктивності, бачимо, що LSTM показала найвищу точність — 92.61%, незначно випередивши гібридну модель (92.26%) та CNN (91.91%). Найнижчу точність серед класичних

моделей зафіксовано у RNN — 89.04%. Водночас, тестовий loss виявився найнижчим у LSTM (31.58%) і CNN (31.72%), що знову підкреслює стабільність цих моделей. Гібридна модель, попри високу точність, має досить високий тестовий loss — 95.48%, що вказує на ймовірну переобучуваність або складність адаптації до нових прикладів. RNN залишається найменш стабільною в цьому аспекті — її тестовий loss склав 90.30%. Трансформер, попри свою сучасну архітектуру, продемонстрував найнижчу загальну точність серед усіх — 84%, що свідчить про обмежену ефективність у контексті класифікації окремих слів, де контекстуальна інформація обмежена або повністю відсутня.

Більш компактний опис результатів тренувальних та тестових метрик можна переглянути за допомогою таблиці:

Модель	Train Accuracy	Train Loss	Test Accuracy	Test Loss
RNN	87.80%	109.28%	89.04%	90.30%
LSTM	93.04%	29.95%	92.61%	31.58%
CNN	91.97%	29.34%	91.91%	31.72%
CNN + BiLSTM	91.12%	104.61%	92.26%	95.48%
Transformer	—	51.09%	84.00%	—

Таблиця 4.3.1 - Результати загальних метрик моделей після їх навчання

Також після завершення навчання кожної моделі результати оцінювалися за допомогою розгорнутого класифікаційного звіту. Такий звіт для кожної моделі дає змогу точніше оцінити, як добре вони розпізнають як позитивні, так і негативні емоції. Перш ніж перейти до пояснення всіх звітів, розглянемо його основні компоненти на прикладі звіту для моделі RNN.

	precision	recall	f1-score	support
negative	0.97	0.84	0.90	3864
positive	0.75	0.95	0.84	1883
accuracy			0.88	5747
macro avg	0.86	0.90	0.87	5747
weighted avg	0.90	0.88	0.88	5747

Рис. 4.3.1 – Класифікаційний звіт для моделі на основі архітектури RNN

Цей звіт складається з кількох ключових компонентів:

- Precision (точність) – частка правильно передбачених позитивних випадків серед усіх передбачених як позитивні або негативні. Наприклад, для класу *positive* значення 0.75 означає, що з усіх прикладів, які модель класифікувала як позитивні, 75% дійсно були позитивними. Це важлива метрика у випадках, коли вартість помилково позитивного результату висока.
- Recall (повнота) є часткою правильно передбачених позитивних/негативних випадків серед усіх фактичних позитивних/негативних прикладі в у даних. Тобто recall 0.95 для класу *positive* означає, що з усіх справжніх позитивних прикладів модель правильно класифікувала 95%. Повнота є критично важливою в задачах, де пропуск позитивного прикладу має серйозні наслідки.
- F1-score – це гармонічне середнє між precision і recall. Ця метрика балансує обидва показники, і часто використовується як єдиний критерій оцінки, коли важлива як точність, так і повнота. Значення 0.90 для класу *negative* свідчить про високий рівень як точності, так і повноти при виявленні негативних прикладів.
- Support є кількістю прикладів кожного класу у тестовому наборі. Наприклад, клас *negative* має 3864 приклади, а клас *positive* — 1883. Ця інформація дозволяє оцінити збалансованість вибірки: якщо один

клас суттєво переважає, це може впливати на якість класифікації менш представленого класу.

- Accuracy – загальна точність моделі, тобто частка всіх правильно передбачених прикладів серед загальної кількості прикладів. У випадку RNN точність становить 0.88, або 88%, що означає, що модель правильно класифікувала 88% всіх прикладів.
- Macro avg – це середнє значення метрик (precision, recall, f1-score) по всіх класах без урахування кількості прикладів у кожному класі. Це дозволяє оцінити, як модель працює для кожного класу незалежно, навіть якщо класи незбалансовані.
- Weighted avg – середньозважене значення метрик по класах, з урахуванням кількості прикладів у кожному класі. Цей показник дає уявлення про загальну продуктивність моделі, враховуючи дисбаланс у класах.

Таке розгорнуте представлення результатів дозволяє не лише бачити загальну точність моделі, але й оцінити її поведінку для кожного окремого класу, виявити потенційні упередження та краще порівнювати між собою різні архітектури моделей.

У випадку RNN найкращі метрики precision і recall — для негативного класу (0.97 і 0.84 відповідно), що говорить про її схильність до виявлення негативних емоцій. Проте у позитивному класі recall помітно вищий (0.95), ніж precision (0.75), що свідчить про переоцінку позитивних випадків, часто помилкову. LSTM, натомість, досягла відмінного балансу: recall у негативному класі — 0.99, а у позитивному — 0.81, при precision 0.97 та 0.91 відповідно. Такий розподіл демонструє сильну здатність до виявлення негативу, що особливо цінно у психологічному аналізі. CNN зберігає найкращий баланс: високі значення precision і recall у обох класах

(понад 0.90), що робить її найстабільнішою у задачі без упередження до одного з класів. Гібридна модель, на жаль, поступається в точності, хоча показує високу чутливість до обох класів, але точні метрики precision/recall по класах не виводилися. Transformer продемонстрував посередні результати: для негативного класу recall 0.91, precision – 0.85, а для позитивного recall – 0.69 та precision – 0.80. Це свідчить про нестабільну продуктивність при роботі з позитивними емоціями.

Ще варто зупинитися на тестуванні моделей на невеликому списку слів, які несуть очевидне емоційне забарвлення. Слова «любов», «війна», «успіх», «горе», «надія», «страх», «щастя» і «депресія» були класифіковані кожною з моделей. Тут модель на основі трансформеру показала себе добре: вона та RNN правильно класифікували слово «успіх» як позитивне, тоді як усі інші моделі визначили його як негативне. Водночас трансформер не зробив жодної помилки у класифікації інших емоційних слів, що свідчить про потенціал його використання у випадках, коли дані мають чітке емоційне забарвлення. Найбільше помилок зробила RNN, яка двічі помилилася, оскільки вона інтерпретувала «страх» та «депресію» як позитивні, демонструючи упередженість у бік позитивних класів. LSTM, CNN та гібридна моделі показали стабільні однакові результати результати, з лише однією помилкою кожна. LSTM чітко розпізнала всі негативні слова, що є важливою перевагою.

Повний вигляд результатів тестування на словах можна продивитися за таблицею, наведеною нижче:

Слово	RNN	LSTM	CNN	CNN + BiLSTM	Transformer
любов	Positive	Positive	Positive	Positive	Positive
війна	Negative	Negative	Negative	Negative	Negative
успіх	Positive	Negative	Negative	Negative	Positive
горе	Negative	Negative	Negative	Negative	Negative

надія	Positive	Positive	Positive	Positive	Positive
страх	Positive	Negative	Negative	Negative	Negative
щастя	Negative	Positive	Positive	Positive	Positive
депресія	Positive	Negative	Negative	Negative	Negative

Таблиця 4.3.2 - Результати тестування моделей на оцінку емоційності слів

Проводилося також додаткове тестування на здатність моделей робити оцінку емоцій цілих речень. Для тестування було взято два таких речення: «Сьогодні був важкий день, але вечір подарував трохи тепла і спокою. Я дуже вдячна за це» та «Сьогодні був для мене поганий день. Я проспала екзамен, запізнилася на автобус та забула свій ланч. В результаті мені довелося йти на перездачу, через це я була дуже засмучена». Результати виявилися неоднозначними. RNN і LSTM коректно класифікували обидва речення як позитивні, що частково відповідає реальній тональності першого речення, але є хибним для другого, в якому явно домінують негативні переживання. CNN, навпаки, неправильно оцінила перше речення як негативне, тоді як друге класифікувала як позитивне, що знову ж не відображає його справжнього емоційного тону. Гібридна модель виявилася єдиною, що коректно ідентифікувала негативну тональність другого речення, однак при цьому неправильно класифікувала перший текст як негативний. Ці результати свідчать про обмеження всіх моделей при роботі з багатозначним контекстом: CNN і гібридна модель можуть занадто фокусуватися на окремих словах, ігноруючи загальний зміст, тоді як RNN та LSTM, ймовірно, схильні до оптимістичної переоцінки через більш «м’яку» реакцію на негативні тригери.

В кінці навчання кожної моделі виводилися графіки, що показують відповідно результати навчання:

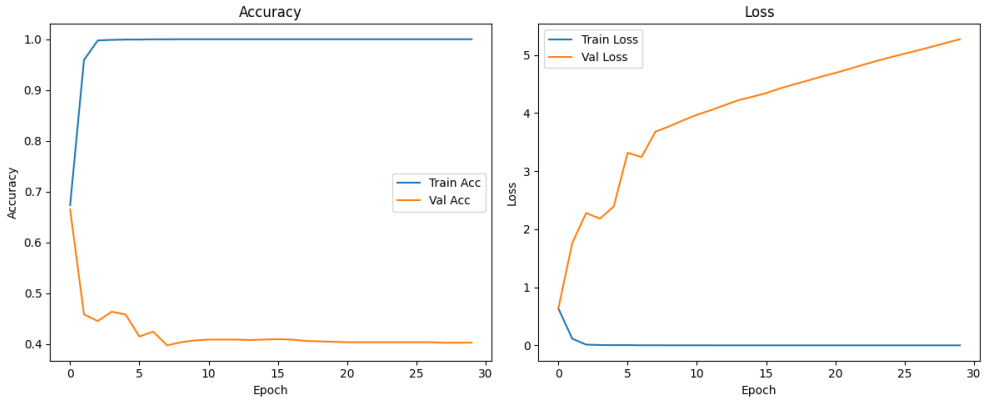

Рис. 4.3.2 – Графік точності та функції втрат для моделі на основі архітектури RNN

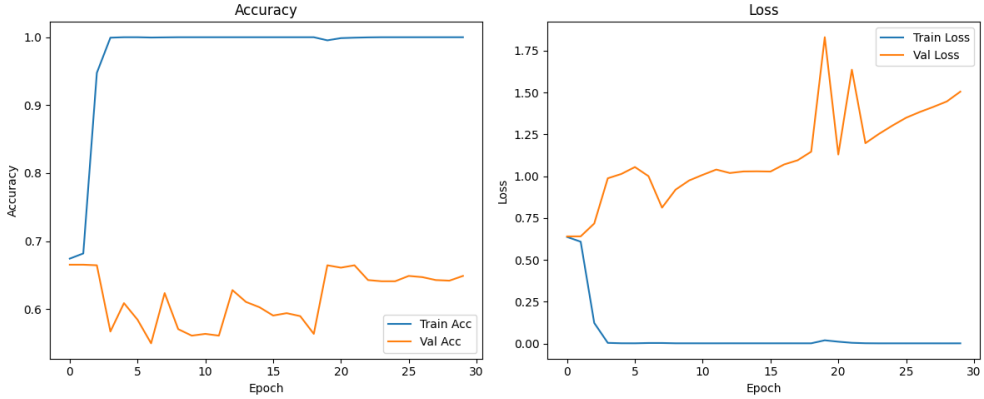

Рис. 4.3.3 – Графік точності та функції втрат для моделі на основі архітектури LSTM

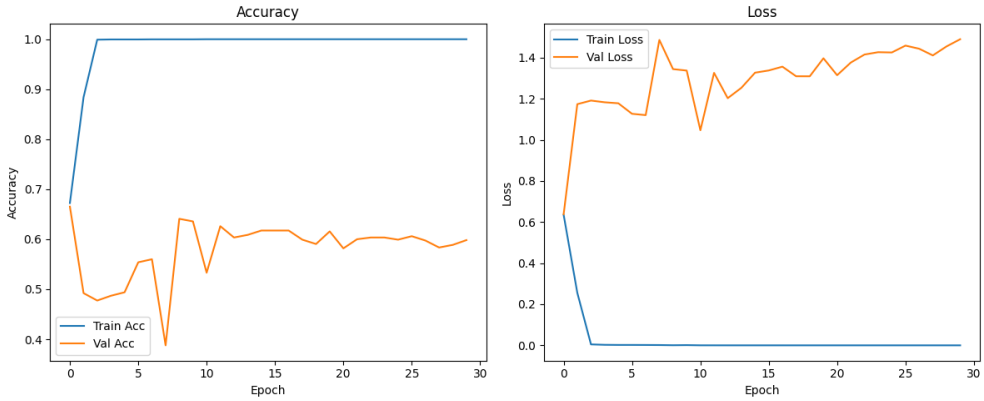

Рис. 4.3.4 – Графік точності та функції втрат для моделі на основі архітектури CNN

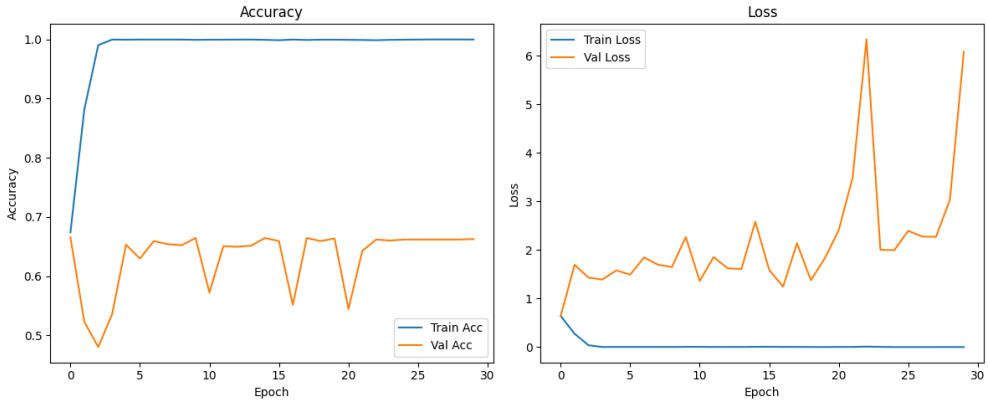

Рис. 4.3.4 – Графік точності та функції втрат для моделі на основі архітектури CNN + BiLSTM

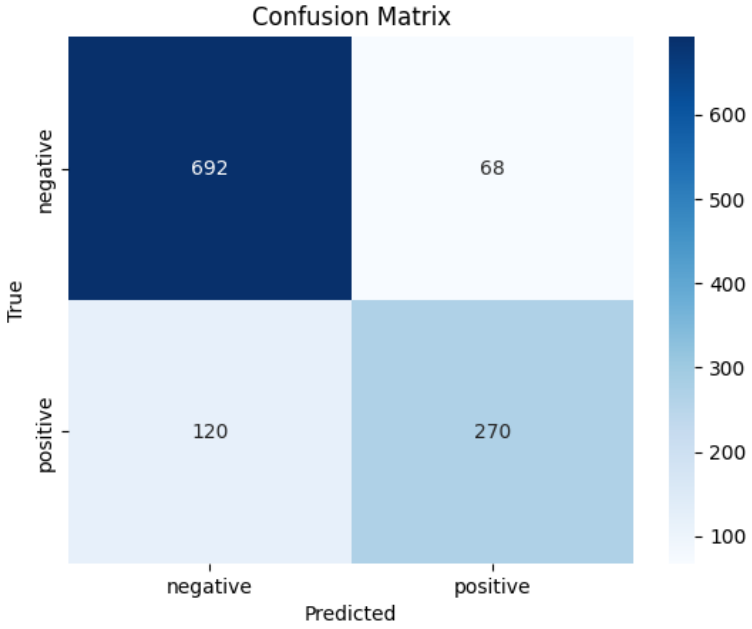

Рис. 4.3.4 – Матриця плутанини для моделі на основі архітектури Transformers

Останній графік був зроблений спеціально для моделі на основі Transformers для того, щоб показати, як часто модель правильно або неправильно класифікує зразки в кожну з категорій. Число 692 означає кількість слів, які були дійсно негативними й модель саме так їх передбачила, в той час як 68 – кількість негативних слів, які модель помилково передбачила як позитивні. Відповідно і з позитивними, 270 –

кількість слів, які модель правильно класифікувала, а 120 слів помилково передбачені як негативні.

Підсумовуючи цей розділ, на основі глибокого аналізу можна зробити висновок, що модель побудована на основі архітектури CNN виявилася найоптимальнішою моделлю для завдання бінарної класифікації емоцій українських слів. Вона демонструє високу точність, мінімальні втрати, збалансованість у класифікації позитивних і негативних емоцій та хорошу узагальнювальну здатність. LSTM можна вважати другим за силою кандидатом, вона здатна точно виявляти негативні емоції. RNN, попри відносно хороші результати, поступається у стабільності й демонструє систематичне зміщення. Гібридна модель хоч й показує конкурентну точність, але її нестабільність та високі втрати свідчать про потребу в доопрацюванні. Модель на основі трансформеру показала ідеальні результати при практичному тестуванні, але через недостачу результатів метрик важко визначити в її збалансованості, тому в межах цієї задачі саме CNN залишається найбільш надійним і ефективним вибором, якщо цю модель відповідно доопрацювати.

Висновки

Метою роботи було проаналізувати переваги, недоліки та особливості таких архітектур, як RNN, LSTM, CNN, трансформери та гібридні моделі. Для досягнення мети було вивчено принципи роботи кожної архітектури, її здатність обробляти контекст, а також відповідні приклади застосування.

В ролі практичної частини курсової роботи виступило дослідження можливості автоматизованого визначення емоційної забарвленості українських слів за допомогою різних архітектур нейронних мереж. У ході дослідження було реалізовано декілька моделей машинного навчання: RNN, LSTM, CNN, гібридну модель CNN + BiLSTM, а також трансформерну модель xlm-roberta-base. Кожна з них була протестована на спеціально зібраному датасеті, що містив українські слова, розмічені як позитивні або негативні.

У результаті експериментів було визначено, що найкращі результати у задачі класифікації продемонструвала модель CNN, яка забезпечила високу точність, низькі втрати та збалансовану роботу з обома класами. LSTM також показала хороші результати, особливо у виявленні негативних емоцій, що робить її корисною в контекстах, де виявлення негативу є критично важливим. Гібридна модель, хоч і досягла високої точності, мала значні втрати, що свідчить про потенційну переобучуваність. RNN виявилась найменш ефективною серед усіх реалізованих моделей. Щодо моделі на основі трансформера, то вона продемонструвала відмінні показники під час практичних тестів, однак відсутність метрик ускладнює оцінку її збалансованості.

Список використаних джерел

[1] What is NLP (natural language processing)? [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/think/topics/natural-language-processing

[2] NLP: What is NLP? [Електронний ресурс] / A. Shakeri – ResearchGate, 2024. – Режим доступу до ресурсу: https://www.researchgate.net/publication/391012959_NLP_What_is_NLP

[3] History and Evolution of NLP [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/history-and-evolution-of-nlp/

[4] Natural Language Processing (NLP) 101: From Beginner to Expert [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/natural-language-processing-nlp-101-from-beginner-to-expert/

[5] Generalized Naive Bayes [Електронний ресурс] / A. Jha – ResearchGate, 2023. – Режим доступу до ресурсу: https://www.researchgate.net/publication/383494678_Generalized_Naive_Bayes

[6] Ok. E. Advanced AI and NLP Approaches [Електронний ресурс] / E. Ok, J. Olusegun, B. Barnty, O. Joseph. – 2025. – Режим доступу до ресурсу: https://www.researchgate.net/publication/389265319_Advanced_AI_and_NLP_Approaches

[7] Devlin, J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Електронний ресурс] – arXiv:1810.04805. – Режим доступу до ресурсу: https://arxiv.org/abs/1810.04805

[8] Patel A., Bhatt R., Patel V. Preprocessing Techniques for Text Mining - An Overview [Електронний ресурс]. – ResearchGate, 2020. – Режим доступу до ресурсу:

https://www.researchgate.net/publication/339529230_Preprocessing_Techniques_for_Text_Mining_-_An_Overview

[9] A Gentle Introduction to the Bag-of-Words Model. [Електронний ресурс] – Режим доступу до ресурсу: https://machinelearningmastery.com/gentle-introduction-bag-words-model/

[10] Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space [Електронний ресурс] / arXiv, 2013. – Режим доступу до ресурсу: https://arxiv.org/abs/1301.3781

[11] Pennington J., Socher R., Manning C. D. GloVe: Global Vectors for Word Representation. [Електронний ресурс] – Режим доступу до ресурсу: https://www.researchgate.net/publication/284576917_Glove_Global_Vectors_for_Word_Representation

[12] Bojanowski P., Grave E., Joulin A., Mikolov T. Enriching Word Vectors with Subword Information [Електронний ресурс] / arXiv, 2016. – Режим доступу до ресурсу: https://arxiv.org/abs/1607.04606

[13] lang-uk [Електронний ресурс] – Режим доступу до ресурсу: https://lang.org.ua/en/models/#word-vectors
https://lang.org.ua/uk/models/

[14] Jurafsky D., Martin J. H. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition [Електронний ресурс]. – 3rd ed. – 2023. – Режим доступу до ресурсу: https://web.stanford.edu/~jurafsky/slp3/

[15] Yadav V., Bethard S. A Survey on Recent Advances in Named Entity Recognition from Deep Learning Models // Proceedings of COLING 2018. [Електронний ресурс] – Режим доступу до ресурсу: https://aclanthology.org/C18-1182/

[16] Blei D. M., Ng A. Y., Jordan M. I. Latent Dirichlet Allocation // Journal of Machine Learning Research. [Електронний ресурс] – 2003. – Vol. 3. – Режим доступу до ресурсу: https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

[17] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. Stanza: A Python NLP Library for Many Human Languages. [Електронний ресурс] – Режим доступу до ресурсу: https://stanfordnlp.github.io/stanza/

[18] Universal Dependencies. Ukrainian UD Treebank. [Електронний ресурс] – Режим доступу до ресурсу: https://universaldependencies.org/treebanks/uk_iu/

[19] Otter D. W., Medina J. R., Kalita J. K. A survey of the usages of deep learning for natural language processing [Електронний ресурс] // arXiv. – 2018. – Режим доступу до ресурсу: https://arxiv.org/abs/1807.10854

[20] Vaswani, A. et al. Attention is All You Need. [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/abs/1706.03762

[21] Офіційний сайт Python. [Електронний ресурс] – Режим доступу до ресурсу: https://www.python.org/

[22] Google Colaboratory. [Електронний ресурс] – Режим доступу до ресурсу: https://colab.research.google.com/

[23] Tensorflow. [Електронний ресурс] – Режим доступу до ресурсу: https://www.tensorflow.org/

[24] Бібліотека Transformers. [Електронний ресурс] – Режим доступу до ресурсу: https://huggingface.co/docs/transformers/index

[25] Бібліотека Datasets. [Електронний ресурс] – Режим доступу до ресурсу: https://huggingface.co/docs/datasets/index

[26] Ukrainian-Sentiment-Analysis. The list of Ukrainian words for sentiment analysis and NLP. [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/skupriienko/Ukrainian-Sentiment-Analysis

[27] XLM-RoBERTa Base [Електронний ресурс] // Hugging Face. – Режим доступу до ресурсу: https://huggingface.co/FacebookAI/xlm-roberta-base