УДК 004.9

ТЕХНОЛОГІЯ ІДЕНТИФІКАЦІЇ РЕРАЙТУ В ТЕКСТОВОМУ КОНТЕНТІ НА ОСНОВІ МЕТОДІВ МАШИННОГО НАВЧАННЯ

Холодна Н. М. – студент кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна.

Висоцька В. А. – канд. техн. наук, доцент, доцент кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна.

АНОТАЦІЯ

Актуальність. Перефразований текстовий контенту або рерайт є однією із складних проблем виявлення академічного плагіату. Більшість систем ідентифікації плагіату призначені для виявлення спільних слів, послідовності лінгвістичних одиниць та незначних змін, але не здатні виявити суттєві семантичні та структурні зміни. Тому більшість випадків плагіату із застосуванням перефразування залишаються непоміченими.

Мета – розроблення технології виявлення перефразувань у тексті на основі моделі класифікації та методів машинного навчання через використання сіамської нейронної мережі на основі рекурентних та типу Transformer – RoBERTa для аналізу рівня подібності речень текстового контенту.

Метод. Для даного дослідження у якості ознак обрані такі метрики семантичної подібності або показники: коефіцієнт Жаккара для спільних N-грам, косинусна відстань між векторними поданнями речень, Word Mover's Distance, відстані за словниками WordNet, передбачення двох ML-моделей: сіамської нейронної мережі на основі рекурентних та типу Transformer – RoBERTa.

Результати. Розроблено інтелектуальну систему виявлення перефразувань у тексті на основі моделі класифікації та методів машинного навчання. Розроблена система використовує принцип стекінгу моделей і інжиніринг ознак (feature engineering). Додаткові ознаки вказують на семантичну приналежність речень або нормовану кількість спільних N-грам. Додатково налаштована (fine-tuned) нейронної мережі RoBERTa (із додатковими повнозв'язними шарами) має меншу чутливість до пар речень, що не є перефразуваннями один одного. Така специфічність моделі може сприяти неправильному звинуваченню у плагіаті або некоректному об'єднанню згенерованого користувачами контенту. Додаткові ознаки збільшують як загальну точність класифікації, так і чутливість моделі до пар тих речень, що не є перефразуваннями один одного.

Висновки. Створена модель показує відмінні результати класифікації на тестових даних PAWS: зважена влучність (precision) – 93%, зважена повнота (recall) – 92%, F-міра (F1-score)–92%, точність (accuracy) – 92%. Результати дослідження показали, що NN типу Transformer можуть бути успішно застосовані для виявлення перефразувань у парі текстів із досить високою точністю без потреби додаткового генерування ознак.

КЛЮЧОВІ СЛОВА: опрацювання природної мови, NLP, ідентифікація рерайту, виявлення перефразувань у тексті, машинне навчання з вчителем, глибинне навчання, класифікація тексту, аналіз тексту, векторне вкладення слів, WordNet, семантична подібність.

АБРЕВІАТУРА

БД – база даних;
ІС – інтелектуальна система;
ІТ – інформаційна технологія;
ПЗ – програмне забезпечення;
ML – machine learning;
NLP – natural language processing;
NN – neural network;
ТРР – text pre-processing;
QA – question answering.

НОМЕНКЛАТУРА

S – система ідентифікації рерайту;
I – множина вхідних даних;
O – множина вихідних даних;
R – основні правила опрацювання потоку вхідних даних в ІС ідентифікації рерайту;
U – параметри опрацювання вхідних даних;
N – нейронна мережа;
α – оператор скачування вхідних даних;
β – оператор опрацювання вхідних даних;
γ – оператор пошуку рівня подібності речень;
μ – оператор попереднього опрацювання тексту;
χ – NLP-оператор;
ω – оператор машинного навчання ІС на достовірних текстових даних;
λ – оператор визначення перефразувань текстів;
i_1 – множина даних ідентифікації;
i_2 – сховище даних тексту/посилань на джерела;
i_3 – множина аналогічних робіт автора/користувача;
i_4 – конкретний запит/текст автора/користувача;
o_1 – запити з ІС до конкретних джерел тексту;
o_2 – колекція джерел, звідки запозичений текст;
o_3 – множина ідентифікованих перефразувань;
r_1 – правила алгоритму взаємодії;
r_2 – NLP-правила;
r_3 – правила алгоритму нейронної мережі;
r_4 – правила алгоритму ідентифікації рерайту;
u_1 – множина рівнів доступу;
u_2 – множина вимог доступу;
u_3 – множина NLP-вимог;
u_4 – множина метрик машинного навчання;
u_5 – множина вимог ідентифікації рерайту.

ВСТУП

Процес перефразування (рерайту) полягає у переписуванні тексту для зміни слів та послідовності

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

зі збереженням початкового сенсу. Ідентифікація рерайту відіграє важливу роль у різних NLP-задачах, включаючи виявлення плагіату, визначення авторства, використання у QA-системах, узагальнення тексту, машинний переклад, аналіз тексту в цілому тощо. Більш загальна задача вимірювання семантичної подібності текстів є важливим в NLP-галузі. Перефразований плагіат є однією із складних проблем, з якими стикаються системи виявлення плагіату. Більшість подібних систем ідентифікації плагіату призначені для виявлення спільних слів і незначних змін, але не здатні виявити серйозні семантичні та структурні зміни. Тому багато випадків рерайту залишаються непоміченими.

Генерація парафраз означає перетворення речення природної мови на нове речення, яке має те саме семантичне значення, але іншу синтаксичну або лексичну форму. Детекцію та генерування перефразувань застосовують у таких напрямках:

– виявлення порушення авторського права – перевірка на плагіат, визначення авторів тексту;

– поєднання дублікатів контенту, згенерованого користувачами, на інформаційних ресурсах;

– поєднання дублікатів записів однієї теми або питання, що дозволяє отримати більшу повноту (recall) для релевантного контенту при пошуку;

– машинний переклад (спрощення речень);

– QA – отримання додаткової інформації шляхом генерування варіантів запиту для отримання відповідей з БД та перефразування відповідей;

– text summarization (підвид перефразування);

– генерація природньої мови (рерайт речень);

– зміна стилю письма.

Виявлення парафраз тісно пов'язане із NLP-задачею оцінки семантичної подібності тексту. У той час як ML-модель повертає ймовірність перефразування або результат бінарної класифікації пар речень, ML-модель для оцінки семантичної подібності повертає ступінь подібності за певною метрикою, напр. оцінку від 1 до 5 (завдання SentEval). При перевірці на перефразування або визначення семантичної подібності до основними проблемами є:

– відсутність спільних слів, наприклад: Is there a Quora user who have seen an UFO? Have you seen an alien?;

– усі слова спільні, наприклад: How did Portural's team performed in match against Germany? How did Germany's team performed in match against Portugal?;

– ручна розмітка пар документів – результат анотації є суб'єктивним в залежності від ситуації;

– орфографічні і синтаксичні помилки, використання сленгу – неправильно написані слова ідентифікуються системою як нові або абсолютно відмінні від правильного значення;

– омоніми – слова мають однакове написання, однак семантичне значення є залежним від контексту.

Вищезазначені проблеми перешкоджають стрімкому розвитку детекції перефразувань як однієї з задач з NLP-галузі. До основних методів виявлення перефразувань належать:

– модель векторного простору (vector space model), основою якої є векторизація або вкладення тексту та розрахунок відстані/подібності між двома текстами (коефіцієнт Жаккара, Евклідова відстань, косинусна відстань, відстань Word Mover's тощо);

– штучні нейронні моделі глибинного навчання (згорткові, рекурентні, сіамські, encoder-decoder, трансформери з алгоритмом уваги тощо);

– розраховані відстані та результати NN-класифікації як окремих ознак навчання фінального класифікатора.

Одним із перших підходів до вимірювання семантичної подібності між текстовим контентом є модель векторного простору (VSM) для задачі області пошуку інформації [1]. Метою VSM є подання кожної сутності колекції (літер у словах, слів у реченнях, речень у контенті, контенту у корпусі) як точки в n-вимірному просторі, тобто як вектор у VSM [2]. Чим ближче розташовані точки в цьому просторі, тим більше вони є семантично подібними, і навпаки. Для заданого набору з k текстів $D=\{D_1, D_2,\ldots, D_k\}$ текст D_i подають у вигляді вектора $D_i=(w_{i1}, w_{i2}, \ldots, w_{in})$. У класичному VSM на основі слів кожен вимір відповідає одному терміну/слову з набору текстів. Вагу визначають на основі різних схем зважування; Bag-of-Words та TF-IDF зазвичай використовують в VSM на основі слів. Подібність між двома текстами D_i і D_j обчислюють за коефіцієнтом Жаккара, Евклідовою відстанню, косинусною відстанню тощо. Основними недоліками цієї моделі є висока розмірність, розрідженість і проблеми зі словником. Тому існують різні модифікації та узагальнення VSM.

Як і попередні методи, у глибинному навчанні документи або тексти подають у вигляді векторів за допомогою методу Doc2Vec. Окрім того, слова також подані як вектори на основі методу Word2Vec [3]. Існують варіанти навчання векторного подання слів на основі методу матричної декомпозиції, наприклад, LSA. Інший алгоритм використовує методи на основі контексту, наприклад, skip-grams, Continious Bag of Words. Ці вектори порівнюють за допомогою косинусної відстані або іншої міри подібності.

Поява моделі Word2Vec спонукала дослідників створити інші векторні моделі, такі як Doc2Vec, FastText, GloVe, USE та ELMO. Усі ці моделі є моделями *2Vec, оскільки вони перетворюють текст (у вигляді слів, фраз, речень, розділів і цілих документів) у векторну форму, створюючи n-вимірні векторні простори. NN-навчання з текстами з великих немаркованих корпусів призводить до створення VSM з використанням довільних параметрів, найважливішими з яких є: розмірність векторного простору, мінімальна частота слів, швидкість навчання та розмір вікна/контексту спостереження кожного слова. Word2Vec складається з двох підмоделей: CBOW (безперервний мішок слів), що прогнозує пропущене слово, якщо надаємо моделі

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

контекст пропущеного слова, тоді як skip-gram прогнозує контекст даного слова.

Метою дослідження є розроблення ІТ для детекції перефразувань, яка дозволить спростити і водночас покращити перевірку текстів на плагіат або об'єднання даних на інформаційних ресурсах за однаковими темами або запитаннями. Проектована система має виявляти дублікати за перефразування за допомогою пошуку аналогічних текстів. Для досягнення мети були поставлені такі завдання:

– розробити та описати функціональні вимоги проектованої системи згідно з методологією Rational Unified Process та Unified Model Language;

– проаналізувати state-of-the-art методи, що використовуються для детекції перефразувань;

– розробити та описати NN з різною архітектурою (згорткові, рекурентні, сіамські NN тощо);

– обрати найбільш оптимальну модель у контексті ТРР, векторного вкладення або векторизації, вибору та генерування ознак, ML-алгоритму та відповідних параметрів;

– реалізація відповідної ІС ідентифікації рерайту та відповідної апробації отриманих результатів.

1 ПОСТАНОВКА ПРОБЛЕМИ

Систему ідентифікації рерайту S подано кортежем:

$$S = < I, O, R, U, N, \alpha, \beta, \gamma >,$$

де $I = \{i_1, i_2, i_3, i_4\}$, $O = \{o_1, o_2, o_3\}$, $R = \{r_1, r_2, r_3, r_4\}$, $U = \{u_1, u_2, u_3, u_4, u_5\}$.

Основними процесами ІС ідентифікації рерайту є «Попереднє опрацювання тексту», «NLP», «Машине навчання» та «Визначення перефразувань».

Процес попереднього опрацювання вхідного тексту ІС ідентифікації рерайту опишемо суперпозицією:

$$C_{AU} = \mu \circ \beta \circ \alpha,$$
$$C_{AU} = \mu(\beta(\alpha(i_1, i_2, i_4), r_1, u_1), u_2).$$

NLP-процес ІС NLP опишемо суперпозицією: $C_{CU} = \chi \circ \beta \circ \alpha$, тобто

$$C_{CU} = \chi(\beta(\alpha(C_{AU}, i_2, i_3, i_4), r_1, u_3), r_2).$$

Процес машинного навчання на достовірних даних ІС ідентифікації рерайту опишемо суперпозицією:

$$C_{UL} = \omega \circ \gamma \circ \beta \circ \alpha,$$
$$C_{UL} = \omega(\gamma(\beta(\alpha(C_{CU}, i_2), i_3), u_4), r_3).$$

Процес визначення перефразувань ІС ідентифікації рерайту опишемо суперпозицією:

$$C_{US} = \lambda \circ \gamma \circ \beta \circ \alpha,$$
$$C_{US} = \lambda(\gamma(\beta(\alpha(C_{US}, i_2), i_4), u_5), r_4).$$

Задача перефразування можна розділити на дві підзадачі як виявлення і генерування перефразувань.

У задачі виявлення парафразувань результатом є ймовірність від 0 до 1, де значення, близьке до 1, означає пару речень як перефразування один одного, 0 – різні за семантичним навантаженням значення.

Використання глибоких NN для NLP значно зросло за останні роки. Для індентифікації перефразувань у дослідженнях використовують сіамські, згорткові, рекурентні, складні архітектури моделей на основі поєднання згорткових та рекурентних NN, трансформерів тощо. Точність моделі залежить від таких факторів, як кількість класів, на які розподіляються дані, вибору методів традиційного або глибинного навчання та їх комбінації з методами векторизації (вкладення) слів і ТРР. При побудові ІС визначення перефразування у тексті для нового набору даних необхідно оцінити не лише різні методи і архітектури ML-моделей, а й ТРР-алгоритми і подання тексту у числовому форматі та можливі комбінації їх поєднання для отримання найкращої можливої якості (точності) функціонування типової ІС ідентифікації рерайту.

Об'єктом дослідження є процеси детекції рерайту у тексті із застосуванням оптимального конвеєру (pipeline), що включає ТРР, векторизацію або вкладення слів, вибір та генерування ознак, бінарну класифікацію за допомогою певного ML-алгоритму і подальше опрацювання отриманих результатів для виявлення перефразувань на інформаційних джерелах. Наукова новизна полягає у застосуванні NN з новою архітектурою, огляд state-of-the-art методів та порівняння різних етапів конвеєру (pipeline) дасть змогу визначити таку їх комбінацію, яка дозволить отримати якісну модель визначення перефразувань в тексті для обраних контрольних наборів даних. Проектована ІС дозволить покращити процес визначення рерайту у тексті. Розроблювана ІС може використовуватися модераторами інформаційних ресурсів та соціальних мереж для оцінки підтримки якості публікацій та об'єднання даних за темами або запитаннями. Окрім того, детекція перефразувань може бути модулем до існуючих систем перевірки текстів на плагіат.

2 АНАЛІЗ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

Більшість ІС перевірки на плагіат порівнюють частини речення або спільні слова, однак детекція перефразувань досі залишається актуальним завданням у галузі опрацювання природньої мови і не є реалізованою у більшості онлайн-платформ і додатків. Окрім того, наявні системи перевірки на плагіат не можуть із достатньою точністю виявити перефразування та вказати оригінальне джерело [4].

Отже, детекція перефразувань є актуальною проблемою і наразі у більшості академічних робіт дослідники використовують методи глибинного навчання і архітектури мовленнєвих моделей, що мають мільйони параметрів. Однак історія вирішення цього завдання бере початок із більш простих методів розрахунку семантичної відмінності як відстані між двома термінами у базі даних WordNet [5]. Це БД семантичних зв'язків між словами (синсетами).

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Зв'язки містять синоніми, гіпоніми (підтип), мероніми (частина) тощо. Ієрархічна структура БД дає змогу розрахувати семантичну спорідненість окремих слів за різними формулами. В [6] запропонувано підхід, заснований на значеннях семантичної спорідненості слів із словника WordNet. У основі методу – ідея порівняння пари слів відповідно до частини мови.

Дослідники використали шість різних відстаней семантичної подібності основі WordNet [7–12]. Максимальну подібність шукають лише всередині класів слів з однаковою частиною мови. Для впровадження двонапрямленності використовують середнє арифметичне двох значень запропонованої функції, значення якої залежить від значення максимальної подібності пари слів та їх специфічності. Аналогічно можна поєднати усі шість відстаней, усереднивши показники фінальної оцінки семантичної подібності. Семантична подібність слів згідно з [7] залежить від довжини найкоротшого шляху між синсетами як вузлами графа. За [8] подібність слів визначається як перетин множин слів визначень, що відповідають заданим термінам. Функція відстані в [9] залежить від глибини двох концептів у таксономії та глибини найближчого спільного предка, за [10] – від ймовірності зустріти спільного предка у великому корпусі, за [11–12] – від ймовірності зустріти як спільного предка, так і два синсети, що порівнюються.

Автори в [13] запропонували новий алгоритм вимірювання семантичної спорідненості коротких речень із використанням методів, заснованих на корпусі (pointwise mutual information та latent semantic analysis) або на вищезазначених відстанях згідно даних з WordNet. Формула семантичної спорідненості речень поєднує метрики подібності пари слів та їх специфічності. Дана формула є потенційно гарним індикатором подібності двох введених текстів, оскільки отримані значення точності, влучності, повноти і F-міри є кращими у порівнянні із базовим підходом вимірювання косинусної відстані між двома вхідними реченнями, переведеними до векторного формату на основі TF-IDF.

У роботі [14] запропонований метод визначення семантичної спорідненості, який генерує семантичний профіль для слів на основі основних концептуальних ознак, зібраних з енциклопедичних знань. Основна ідея моделі – значення слова визначають поняттями, які знаходяться в прямому контексті. Даний метод складається з двох основних кроків. Спочатку на основі Wikipedia створюють анотований корпус основних концептуальних слів з відповідними посиланнями. Далі корпус використовують для вимірювання семантичної подібності слів та текстів.

В [15] адаптували формулу косинусної відстані між векторами для розрахунку семантичної подібності текстів. Запропонований алгоритм використовує інформацію про подібність слів, отриману із словників WordNet. Для розрахунку подібності між парами речень використовується косинусна відстань між векторами, що подають речення, та матриці семантичної подібності, що містять інформацію про подібність пар слів, отриману з шести семантичних відстаней WordNet, заснованих на ієрархії. Кожне речення подано у вигляді двійкового вектора (з елементами, рівними 1, якщо слово присутнє у реченні, і 0 в іншому випадку).

У [16] проведено порівняльне дослідження між векторними вкладеннями слів, отриманими за допомогою NN, і традиційними векторними поданнями слів, заснованими на підрахунку спільних появ. Автори виконали два завдання невеликого масштабу (розбір значень слів і подібність речень), і два великомасштабних (виявлення парафразувань і тегування актів діалогу). Для задачі розпізнавання перефразувань дослідники використали косинусну відстань між векторами. Використання векторних вкладень слів, отриманих із використанням NN, значно покращує якість моделі для виявлення перефразувань та тегування актів діалогу, однак не дає значного приросту точності у розв'язанні завдань невеликого масштабу.

В [17] автори запропонували метод вимірювання семантичної подібності текстів з використанням вимірювання семантичної подібності слів на основі корпусу (pointwise mutual information) та нормалізованої і модифікованої версії алгоритму відповідності рядків найдовшої спільної послідовності (LCS). Дослідження зосереджено на вимірюванні подібності між двома реченнями або між двома короткими абзацами. Метод оцінений на основі даних Microsoft Paraphrase Corpus (MSRP).

В [18] удосконалили процес перевірки тексту на плагіат за допомогою використання різних NLP-алгоритмів: кількість однакових 3-грам, Language Model Probability, Longest Common Subsequence, Dependency Relations Matching. Отримані ознаки використані для навчання наївного Байєсівського класифікатора за двома категоріями в залежності від наявності плагіату та за чотирма категоріями в залежності від рівня та типу перефразування. Результати експериментів є задовільними для класифікації документів у корпусі на дві категорії (з рерайтом та унікальні): 37 із 38 документів без плагіату правильно класифіковані, і лише декілька документів із плагіатом класифіковані як без плагіату (5 з 57). Розмежування між трьома різними рівнями плагіату виявилося набагато складнішим завданням. Незадовільна точність моделі класифікації на типи плагіату залежить як від складності завдання, так і від можливої помилки анотаторів.

В [19–20] розробили дві системи для визначення семантичної подібності пар коротких речень. Усі міри подібності слів засновані на WordNet. Щоб обчислити оцінку подібності для пари слів, автори взяли максимальний бал подібності для всіх можливих пар понять – синсетів WordNet та використали бібліотеку NLTK для обчислення мір подібності Leacock and Chodorow і Lin. Для подібності слів на основі корпусу

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

автори використали дистрибутивну лексичну семантичну модель. Вони використали латентний семантичний аналіз (LSA) на великому корпусі для оцінки розподілів. Система навчена прогнозувати оцінки подібності речень, використовуючи регресійну модель опорного вектора з вимірюванням кількох характеристик ступеня накладання спільних слів, подібності синтаксису і семантичної близькості слів.

У [21] подано метод для ідентифікації наявності плагіату із застосуванням перефразувань. Цей метод використовує класичний ML-алгоритм – логістичну регресію. В даній системі ознаками виступають певні особливості лексики, синтаксису, семантики та структури, які отримуються з «підозрілого» документу та оригіналу. До ознак особливостей лексики належать: Dice Coefficient (кількість спільних символів), Jaro Distance, Jaccard Coefficient (відношення спільних термінів до загальної кількості термінів), Levenshtein Distance, адаптована Manhattan Distance, Ngram Distance – аналог Manhattan Distance для N-грамів, Soundex Distance. До синтаксичних ознак належать POS N-gram Distance та Noun Ratio, семантичних – Semantic Similarity Distance, структурних – Stopword N-gram Distance, Word Pair Order, String Length Ratio. Результати експерименту на корпусі даних PAN@CLEF2013 показують, що використання різних типів ознак дає змогу отримати більш точні результати.

В [22] запропонували нову архітектуру глибокого навчання Bi-CNN-MI для виявлення/ідентифікації рерайту шляхом порівняння речень на кількох рівнях деталізації (уніграми, *N*-грами та речення) на основі згорткової NN та моделюючи особливості взаємодії на кожному рівні. Отримані характеристики є вхідними ознаками для бінарного класифікатора на основі логістичної регресії.

У [23] подана система для вирішення задачі детекції перефразувань шляхом виявлення відмінностей між реченнями та оцінки того, наскільки речення є різними. Такий метод також дозволяє виявити такі перефразування, що містять незначну кількість додаткової інформації. У якості алгоритму класифікації використано метод опорних векторів.

В [24] порівняли класичні ML-алгоритми (метод опорних векторів, наївний Байєсів класифікатор, maximum entropy classifier). Ознаками є дані про лексичну і семантичну спорідненість речень. Група ознак «word overlap» містить відношення кількості однакових N-грамів до загальної кількісті слів у реченнях. До лексичної групи ознак також належать skip-grams та longest common subsequence. До семантичних ознак належить noun/verb semantic similarity measure – кількість спільних іменників або дієслів, proper name – кількість спільних власних назв. Семантична ознака cardinal number дає змогу фіксувати числа з порівнянням «більше ніж» та «менше ніж», а також числа, записані словами. За результатами дослідження, найкращі показники досягнуті за допомогою методу опорних векторів.

У [25] показали, що метрики якості машинного перекладу (BLEU, NIST, WER, PER) можуть бути застосовані як ознаки у системі визначення і детекції перефразувань для навчання класифікатора і отримання якісних результатів. Окрім того, дослідники розробили класифікатор, заснований на Position independent word error rate (PER) та інформації про розподіл частин мови у реченнях.

В [26] повторно дослідили гіпотезу про те, що метрики, розроблені для автоматизованої оцінки якості машинного перекладу, можуть бути використані як ознаки класифікатора для детекції перефразувань. Результати показали, що мета-класифікатор, ознаками якого є лише метрики якості машинного перекладу, значно перевершує показники точності, отримані у попередніх дослідженнях. Усього використано три алгоритми класичного машинного навчання для класифікації: логістична регресія, SMO імплементацію методу опорних векторів і варіацію алгоритму найближчих сусідів. До метрик якості машинного перекладу належать: BLEU, NIST, TER, TERp, METEOR, SEPIA. Окреме використання метрики TERp забезпечує непогані результати і перевершує багато інших методів класифікації на основі численних складних ознак.

В [27] розробили систему для детекції перефразувань у коротких текстах на основі методів глибинного навчання. Архітектура класифікатора заснована на шарах згортки та рекурентних (long-short term memory) NN, що опрацьовують отриманий результат. Результат опрацювання рекурентними NN є семантичним поданням речення, тому ознаками фінального класифікатора є різниця між двома векторами виходу з рекурентних NN. Згорткова NN перетворює парну матрицю подібності усіх слів на вектор семантичної подібності речень, що використовується як ознаки для класифікатора. Додатковими ознаками є косинусна відстань між векторами двох речень, середнє значення відстані між іменниками, дієсловами, прикметниками, обчисленої на основі WordNet, відношення кількості спільних уні-, 2-, 3-грам до загальної кількості N-грамів тощо.

У роботі [28] поданий підхід до ідентифікації перефразувань на основі рекурсивних автокодерів, що досліджують вектори ознак для фраз за допомогою синтаксичних дерев. Ці ознаки використані для вимірювання подібності слів і фраз між двома реченнями. Оскільки речення мають довільну довжину, результуюча матриця подібності має змінний розмір. Додано новий рівень динамічного пулінгу, який обчислює подання фіксованого розміру з матриць змінного розміру. Дане подання використовувалося як вхідні дані для класифікатора.

У дослідженні [29] на противагу створенню та відбору великої кількості ознак використано рекурентну NN із шарами довгої-короткочасної пам'яті. NN приймає на вхід речення змінної довжини і навчається завданню регресії – передбаченню ступеня семантичної спорідненості пари речень. NN

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

має сіамську архітектуру і навчена з використанням функції втрат Mean Squared Error. Для подання слів у вигляді векторів використані попередньо навчені векторні вкладення word2vec, отримані на великих нерозмічених корпусах даних.

Автори в [30] поєднали сіамську рекурентну NN із двонапрямленими рекурентними мережами із шарами довгої-короткочасної пам'яті на рівні символів. Така модель є нечутливою до помилок орфографії, заміни синонімів і зайвих слів.

В [31] протестували декілька різних типів сіамських NN для задачі детекції перефразувань: LSTM, Bi-directional LSTM, GRU, Bi-directional GRU, LSTM + Attention, GRU + Attention, GRU + Capsule + Flatten. Найкращі результати отримані для NN з клітинами типу gated recurrent unit.

В [32] використали сіамську NN із довгою-короткочасною пам'яттю для класифікації пари текстів арабською мовою, відповідно до того, чи є вони перефразуванням один одного. Для векторного подання слів дослідники застосували метод векторного вкладення сімейства word2vec, що має назву Glove. Для розрахунку семантичної подібності текстів використана косинусна відстань між двома векторними поданнями речень.

В [33] використали сіамську NN для детекції парафраз у текстах мови те́лугу. Більшість даних були зібрані вручну з різних газет. Окрім того, дослідники порівняли три методи векторного вкладення (Word2Vec, Glove, Fasttext) та їх комбінацію. Найкраща точність на тестовому наборі даних досягнута для комбінації цих трьох методів.

У [34] поданий підхід глибокого навчання з підкріпленням до генерації парафразів. Описана нова структура розв'язку даної задачі, яка складається з генератора та оцінювача. Генератор, побудований як ML-модель на основі рекурентних NN архітектури sequence-to-sequence, перефразовує вхідне речення. Оцінювач класифікує, чи є два речення перефразуваннями одне одного. Генератор спочатку тренується за допомогою глибокого навчання, а потім додатково налаштовується за допомогою навчання з підкріпленням, під час якого оцінювач дає винагороду. Для навчання оцінювача дослідники пропонують два методи, засновані на навчанні із вчителем та навчанні з оберненим підкріпленням відповідно, залежно від типу навчальних даних.

В [35] розроблено систему SimAll, що поєднує методи, засновані на порівнянні стрічок, корпусі та знаннях, і підтримує англійську та арабську мови. Усього система підтримує 61 алгоритм оцінки семантичної подібності речень, результати оцінки кожного алгоритму після нормування належать інтервалу [0, 1]. Користувач має обрати метод агрегування результатів: середнє, сума або максимальне значення. До метрик, заснованих на знаннях, належить 6 відстаней семантичної схожості: Path (path), Leacock & Chodorow (lch), Wu & Palmer, Resnik (res), Lin (lin), Jiang & Conrath (jcn). Метрики доступні лише для англійської мови.

В [36] запропоновано два варіанти деревоподібної моделі LSTM для опрацювання дерев залежностей або синтаксичних дерев. Створені моделі протестовані на задачі визначення ступеня семантичної подібності двох речень.

У [37] розроблено системи, що поєднує згорткові та рекурентні NN для вимірювання семантичної подібності речень. Дослідники використали мережу згортки для врахування локального контексту слів і LSTM для врахування глобального контексту речень. Така поєднання мереж допомагає зберегти відповідну інформацію про речення та покращує обчислення ступеня подібності речень.

В [38] презентували великомасштабний корпус паралельних даних, утворений за допомогою моделей глибинного навчання BERT, RoBERTa, Longformer архітектури Transformer. Набір даних включає параграфи з наукових праць у arXiv, тези, а також статті Вікіпедії та їх перефразовані аналоги (всього 1,5 млн абзаців). Архітектура глибинного навчання Transformer дає змогу досить точно класифікувати оригінальний та перефразований тексти із використанням статичних векторних вкладень (fastText) [39]. Модель RoBERTa досягла найкращих результатів у задачі виявлення перефразувань.

В [40] запропонували і застосували новий підхід до створення паралельний корпусів перефразованих текстів за допомогою генеративних NN архітектури Transformer. Відповідно до результатів досліджень, хоча згенеровані NN набори даних дають змогу покращити точність систем виявлення перефразувань, менший, але створений людиною набір даних ще більше покращує точність класифікації.

В [41] застосували додаткове налаштування (fine-tuning) моделі глибинного навчання BERT архітектури Transformer. Модель BERT має 110 мільйонів параметрів у базовій версії, 340 мільйонів – у «великій» [42]. Попереднє налаштування BERT складається з двох завдань. Першим завданням є моделювання мови, за якого модель має передбачити випадково обраний і замаскований токен. Іншим завданням є передбачення наступного речення, де модель має визначити, чи є два речення послідовними. Додаткове налаштування моделі для основного завдання відбувається за допомогою відповідного набору даних.

В [43] презентували метрику відстані між документами Word Mover's Distance на основі векторних вкладень. Word Mover's Distance вимірює відмінність між як мінімальну відстань, яку «мають пройти» векторні вкладення слів першого речення до векторних вкладень слів другого речення.

3 МАТЕРІАЛИ ТА МЕТОДИ

ІС виявлення перефразувань призначена для модераторів інформаційних ресурсів та соціальних мереж для оцінки та підтримки якості публікацій і

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

об’єднання даних за темами або запитаннями. Окрім того, алгоритм детекції перефразувань може бути доданий до існуючих систем перевірки текстів на плагіат. Наведемо приклад опису вимог у контексті ІС перевірки унікальності академічних робіт.

Функціонування ІС забезпечується науковим керівником, автором тексту, особою, що відповідальна за перевірку тексту та програмним забезпеченням. Науковий керівник (або викладач) здійснює контроль на всіх етапах написання автором своєї роботи і проводить ТРР до автоматизованої перевірки на оригінальність. Автор активно взаємодіє з керівником і реагує на його критику, виправляє роботу відповідно до його зауважень і потім передає свою роботу на перевірку. Відповідальна особа завантажує текст у програму і отримує результат, повідомляє автора та наукового керівника, формує звіт. Зацікавлені особи прецеденту та їх вимоги:

– ПЗ для перевірки текстів має відповідати визначеним системним вимогам, повинне забезпечити максимальну точність при перевірці результатів та формуванні звітів; має автентифікувати користувача, перевірити роботу і повернути результат, зберегти результат у обліковому записі користувача;

– автор хоче отримати підтвердження оригінальності роботи для її публікації або захисту; повинен написати текст, оформити роботу згідно вимог і надати її для подальшого опрацювання;

– відповідальна за перевірку особа повинна отримати результат автоматизованої перевірки, впевнитись в тому, що програма правильно визначила скопійовані фрагменти та першоджерела перефразованих речень.

Користувач ІС: відповідальна за перевірку особа, що буде використовувати ІС, для перевірки текстів на унікальність. Передумови прецеденту:

– наукова робота має бути належним чином оформлена і подана в одному з форматів, які підтримує система;

– комп’ютер, за допомогою якого здійснюватиметься перевірка, підключений до Інтернету та має встановлене необхідне ПЗ;

– відповідальна за перевірку особа має бути успішно авторизована.

Основний успішний сценарій:

– відповідальна за перевірку особа завантажує текст у програму і отримує результат;

– результат експертизи підлягає додатковому аналізу щодо кількості виявлених співпадінь та адекватності посилань на першоджерела.

Альтернативні потоки:

– помилка у роботі програми:

1. Відповідальний за перевірку звертається до служби технічної підтримки (розробника) і повідомляє про помилку у програмі.

2. Розробник усуває помилку і надає нову версію або надає інформацію про способи її самостійного усунення (точка повернення).

Пост-умови:

– дані про результат перевірки занесені до БД.

Спеціальні вимоги: ІС має точно встановлювати відсоток унікальності тексту, підсвічувати частини тексту відповідно до їх типів (скопійований текст, оригінальний текст, перефразований текст, цитата або неправильне вказування першоджерела) та надати список сайтів (робіт) з відсотком збігу, повноцінно підтримувати українську та англійську мови, легко інсталюватися, бути загальнодоступною, підтримувати різні формати (.txt, .doc, .docx тощо).

На діаграмі використання (рис. 1) основними акторами є користувачі та додатки, що по-різному реалізують функції системи. На цій діаграмі не зображені деталізовані варіанти використання, як-от реєстрація користувача, завантаження файлу або взаємодія адміністратора із системою, що у свою чергу містить налаштування системи, навчання NN, обрання методів виявлення перефразувань (рис. 2–3).

Індивідуальний користувач платформи може використовувати систему як додаток для перевірки на плагіат та унікальність вмісту. Така система може використовуватись для перевірки академічної чесності студентів та наукових співробітників. Окрім того, система перевірки на плагіат може використовуватись для встановлення автора тексту. Завдяки додатковій перевірці на наявність перефразувань така система матиме більшу повноту (recall). У такому випадку збільшиться частка правильно вказаних джерел від усіх істинних.

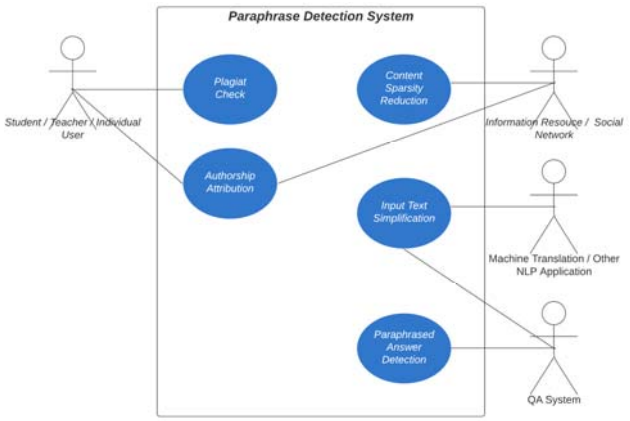


Рисунок 1 – Діаграма use case ІС виявлення рерайту

Інформаційний ресурс (у тому числі і соціальні мережі) можуть використовувати систему для зменшення розрідженості контенту шляхом об’єднання публікацій або запитань, що відносяться до однієї спільної теми. Також ці ресурси можуть використовувати систему виявлення перефразувань для встановлення автора тексту. Як додатки машинного перекладу, так і QA-системи можуть застосовувати функціонал запропонованої ІС для спрощення вхідного повідомлення. Окрім того, QA-системи також можуть застосовувати детекцію перефразувань для більш розширеного пошуку відповідей на запитання.

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Для побудови діаграми класів визначено сім основних класів: Individual User (користувач системи), Document (документ, унікальність якого має бути перевірена), Paraphrase Detection Software (додаток для перевірки на плагіат), User Database (БД користувачів), Documents Database (БД документів), Data Processing Server (сервер опрацювання даних), ML Model (ML-модель). Діаграма класів на рис. 5 позначає застосування системи лише індивідуальним користувачем для перевірки унікальності власних текстів. На діаграмі класів застосування системи виявлення перефразувань, що використовується сторонніми додатками, мають бути визначені додаткові методи обміну повідомленнями із головною програмою за допомогою запитів. Користувач може завантажити та змінити документ, здійснити його перевірку та отримати результати.

Клас додатку перевірки на плагіат позначає програму, що реалізовує користувацький інтерфейс та базові методи перевірки на плагіат шляхом порівняння стрічок. Така програма може бути встановлена локально на комп'ютері або реалізована як онлайн-платформа. Додаток відправляє запити до обчислювального серверу та бази даних користувачів.

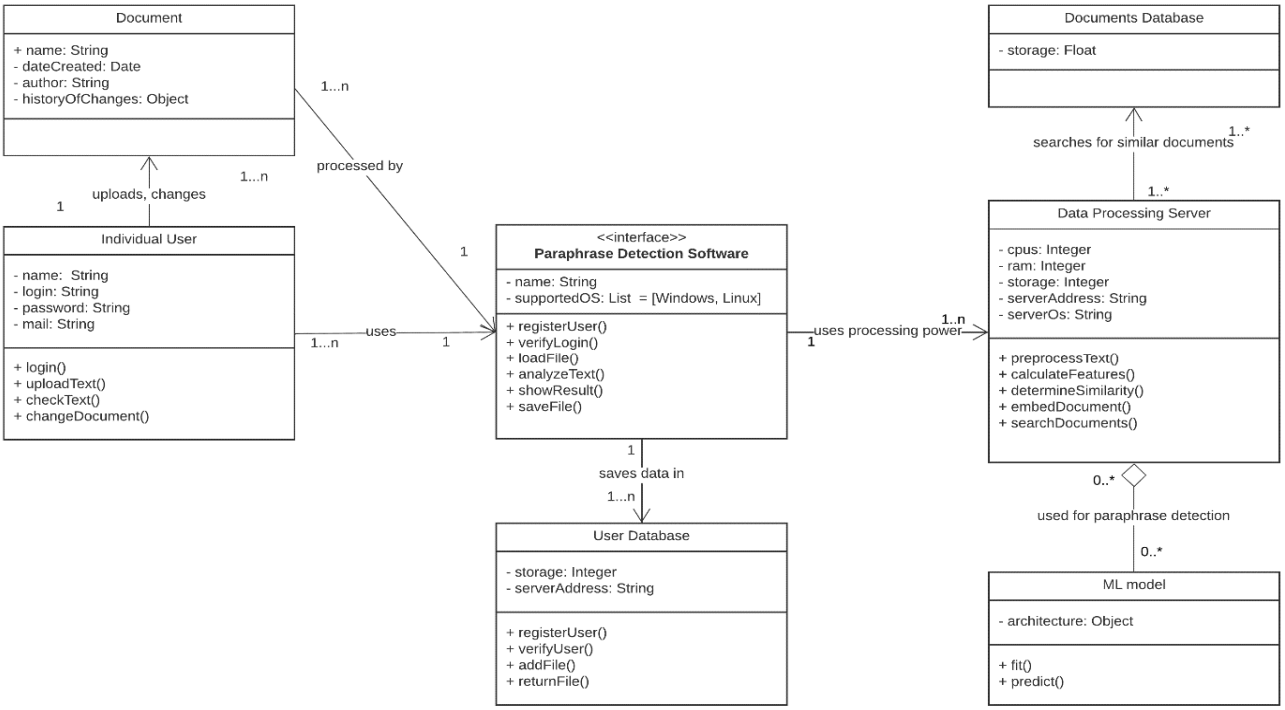


Рисунок 2 – Діаграма класів ІС виявлення перефразувань у тексті

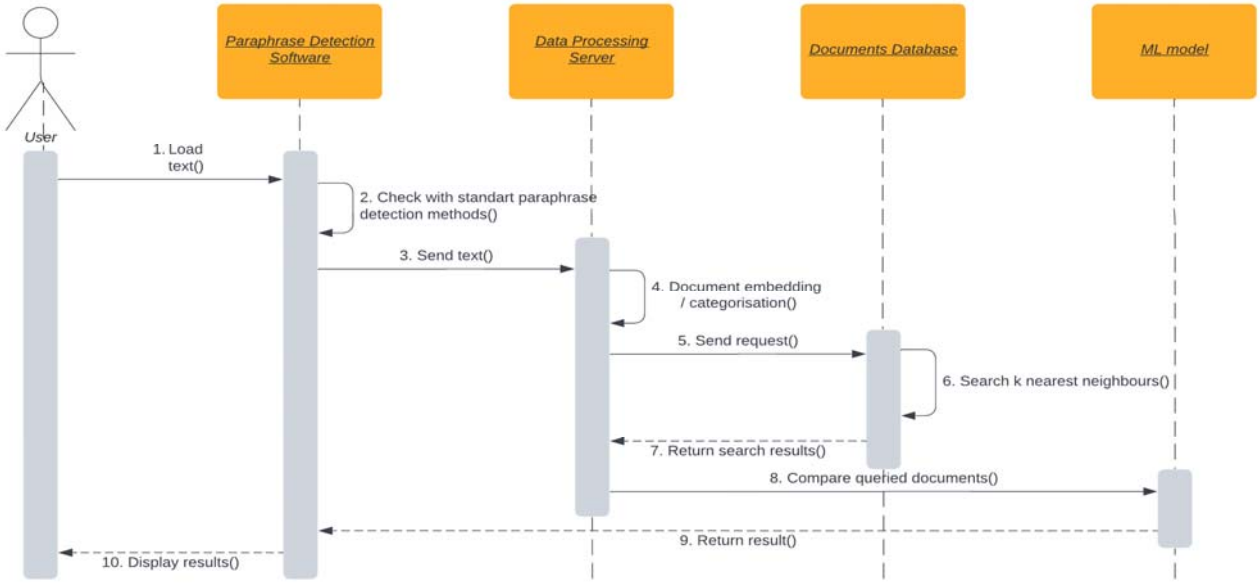


Рисунок 3 – Діаграма послідовності системи виявлення перефразувань

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Сервер опрацювання даних містить методи для порівняння документів за допомогою ML-алгоритмів та застосовує векторне вкладення або рубрикацію документів для пошуку найближчих сусідів, оскільки попарне порівняння тексту із усіма документами, наявними в БД, є надто ресурсозатратним процесом. Сервер опрацювання даних також має розраховувати додаткові ознаки, що будуть використовуватись як вхідні дані класифікатора. Такими додатковими ознаками можуть бути спільна кількість іменованих сутностей або частин мови (рис. 3). Розрахунок таких ознак вимагатиме додаткової реалізації спеціальних парсерів. Діаграма послідовності показує часові аспекти взаємодії компонентів системи. Оминаючи налаштування системи, реєстрацію і вхід користувача, після завантаження файлу наступним кроком є перевірка текстів на унікальність за допомогою базових алгоритмів, що засновані на порівнянні стрічок та їх фрагментів. Для більш детальної перевірки на наявність перефразувань документ спочатку має бути рубрикований або переведений у векторне подання. Такий підхід застосовується з метою збереження обчислювальних ресурсів та зменшення часу опрацювання запиту.

До важливих внутрішніх налаштувань системи належить кількість документів, які будуть порівняні із користувацьким. Завелика кількість подібних документів сповільнить перевірку, замала – зменшить її точність і повноту. Результати визначення ступеня семантичної подібності або бінарної класифікації повертаються і мають бути візуалізовані разом з результатами попередньої простої перевірки.

Важливою частиною розробки системи автоматичного виявлення перефразувань є створення та навчання ML-моделі для задачі класифікації текстів. Для отримання максимальної якості перевірки необхідно дослідити різні методи розрахунку відстаней, ML-алгоритми, TPP-методи.

Діаграма діяльності розподілена на дві частини (рис. 4–5) і позначає усі можливі розгалуження під час TPP та побудови системи відповідно. Отже, перед розробниками системи постає система розгалужень з усіх можливих методів та їх комбінацій. У випадку бінарної класифікації для початку потрібно впевнитись, що обсяг корпусу є достатньо великим і збалансованим, оскільки чисельна перевага записів певного класу може призвести до упередженого рішення штучної NN.

Якщо даних не є достатньо, застосовують методи генерації штучних даних або зміни наявних записів. Після цього потрібно видалити "шум", що може бути зайвими символами, що не містять смислове навантаження. Короткі тексти, особливо публікації у соціальних мережах із великою ймовірністю будуть містити emoji. Emoji можуть вказувати на тональне або емоційне забарвлення тексту, тому розробники системи можуть замінити їх відповідними словами або видалити. Наступним кроком є розбиття речень на токени. Стоп-слова не містять семантичного забарвлення, тому їх видалення є опціональним. До наступних кроків TPP належать лематизація (початкова форма слова), стемінг (основа слова), виправлення орфографічних помилок та видалення цифр. Як і у випадку із видаленням стоп-слів, застосування вищезгаданих кроків не є обов'язковим, однак може по-різному впливати на точність класифікації. Після TPP набір даних розділяється на навчальну та тренувальну вибірки.

На рис. 5 зображені усі основні методи та їх поєднання для бінарної класифікації пари речень в залежності від того, чи є вони перефразуванням один одного. Розробники системи можуть застосувати одну метрику для фінального рішення або агрегувати декілька значень за допомогою ML-методів, наприклад, логістичної регресії або випадкового лісу.

Для спрощення вигляду діаграми, на ній частково відсутні сторожові умови переходу між діяльностями. «Нотатками» позначені додаткові можливі розгалуження вибору методу розрахунку відстані за WordNet, методу агрегування відстаней, векторного вкладення слів, класифікації. На діаграмі розгdfgdgортання системи автоматичного виявлення перефразувань (рис. 6) зображені процесори Сервер опрацювання даних і дві бази даних. Користувач має доступ до функціоналу системи за допомогою графічного інтерфейсу завантаженої програми або онлайн-платформи. У БД зберігається інформація про користувачів та документи. Сервер опрацювання даних містить модулі ML-моделі, базової перевірки на плагіат і векторного вкладення або рубрикації документу. Альтернативним варіантом схеми розгортання є використання власних обчислювальних ресурсів персонального комп'ютера для аналізу даних, однак, з врахуванням того, що для отримання достовірного результату необхідно застосувати кілька етапів перевірки, особливо з використанням ML-методів і порівняння документів, що зберігаються у базі даних, така архітектура не є доцільною.

Для ML найпопулярнішими мовами є Python, R, Java, C++. Перевагою Python з-поміж інших мов саме для створення системи розпізнавання перефразувань є його підтримка великої кількості бібліотек:

– для роботи з ML-методами: Scikit-Learn;

– для створення штучних NN, глибинного навчання: TensorFlow, Keras, PyTorch;

– для опрацювання природньої мови: NLTK, spaCy, WordNet;

– для роботи із масивами, матрицями: NumPy;

– роботи з таблицями: pandas;

– візуалізації даних (в тому числі, інтерактивної): Matplotlib, seaborn, Plotly.

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

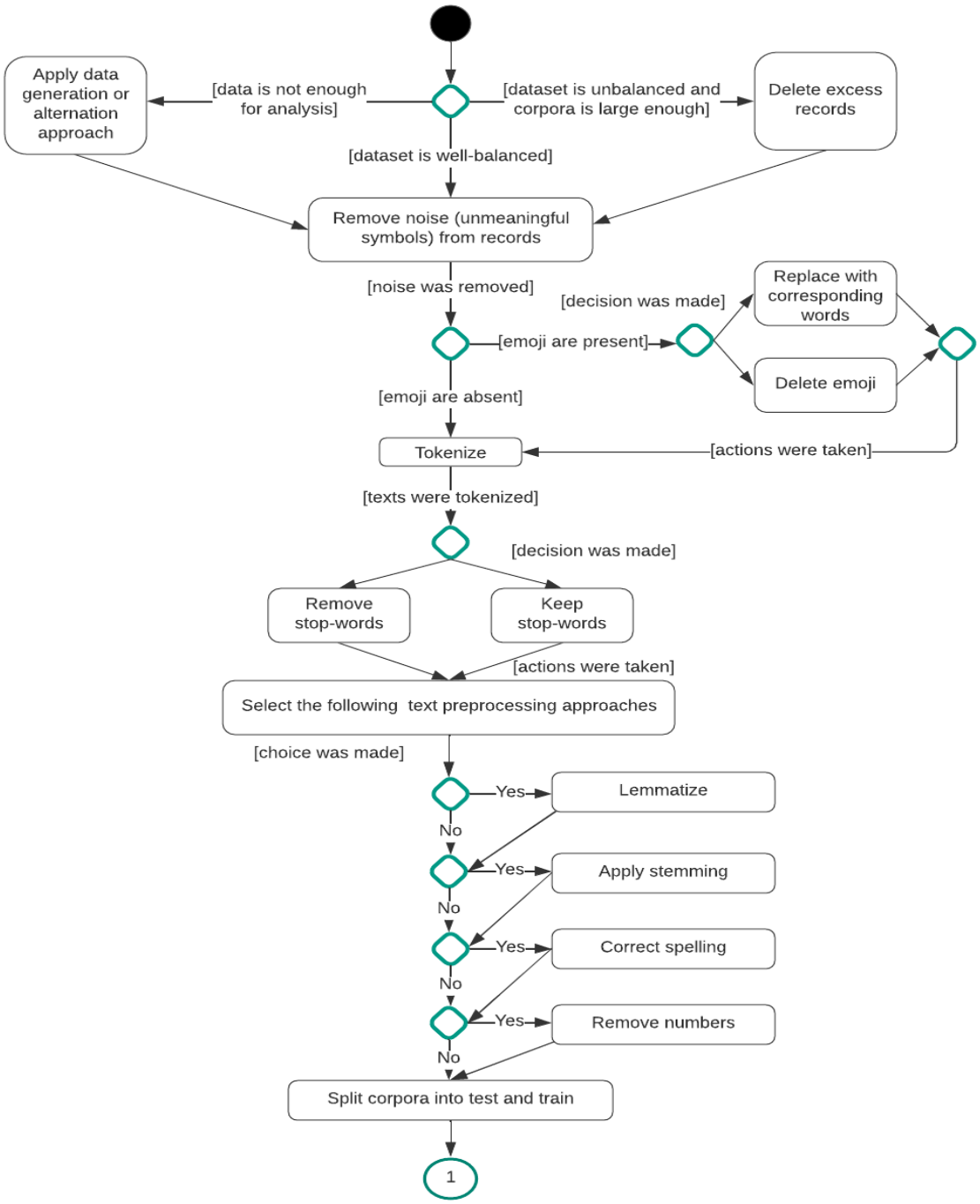


Рисунок 4 – Перша частина діаграми активності створення системи виявлення перефразувань

Бібліотека Scikit-Learn підтримує ТРР, зменшення розмірності даних, вибір ML-моделей для регресії, класифікації або кластерного аналізу. Однак Scikit-Learn не має комплексної підтримки для створення моделей глибинного навчання. Для створення штучних NN обрана бібліотека Keras, що слугує високорівневим API для TensorFlow 2. Keras дозволяє будувати послідовні моделі у вигляді графу, вершинами якого є шари (layers) певного типу із заданою кількістю вузлів. Також, за допомогою Keras можна поєднувати результати роботи кількох окремих частин NN для їх подальшого опрацювання, така структура не є лінійною. TensorFlow дає можливість імпортувати навчену ML-модель для її подальшого використання у інших програмах. TensorFlow також підтримує виконання низькорівневих операцій із тензорами за допомогою центральних процесорів, графічних процесорів та тензорних блоків опрацювання. Бібліотека NLTK у цьому дослідженні застосовується для ТРР: токенізації, видалення стоп-слів, стемінгу, лематизації. Також за допомогою функцій з цієї бібліотеки можна виявляти найпопулярніші N-грами і частини мови окремих токенів, розпізнавати іменовані сутності тощо. До додаткових бібліотек, що спрощують роботу із природньою мовою, належать Regex та emoji – для

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

використання регулярних виразів і заміни емоджі словами відповідно. Для роботи над задачами з дослідження даних і застосування ML зручним інструментом є Jupyter Notebook, який дозволяє запускати написаний код невеликими фрагментами – комірками. Одним із онлайн-сервісів, що надає змогу використовувати Jupyter Notebook без локального встановлення, є Google Colab. Цей сервіс дає можливість використовувати графічні процесори GPU і TPU, що значно пришвидшують навчання NN.

Для навчання і тестування ML-моделі, а також конструювання ознак обраний набір даних Paraphrase Adversaries from Word Scrambling [44], частина PAWS-Wiki Labeled (Final). PAWS-Wiki містить 65 401 пару речень, 44,2% з яких є перефразуваннями один одного. Оскільки у різних дослідженнях для визначення семантичної подібності та виявлення перефразувань використовуються відмінні одна від одної методології (від розрахунку кількості спільних N-грам до застосування глибинного ML-методів) [45], для отримання максимально великої точності класифікації необхідно порівняти та (або) поєднати різні алгоритми виміру семантичної подібності речень. Для даного дослідження у якості ознак обрані такі метрики семантичної подібності або показники: коефіцієнт Жаккара для спільних N-грам, косинусна

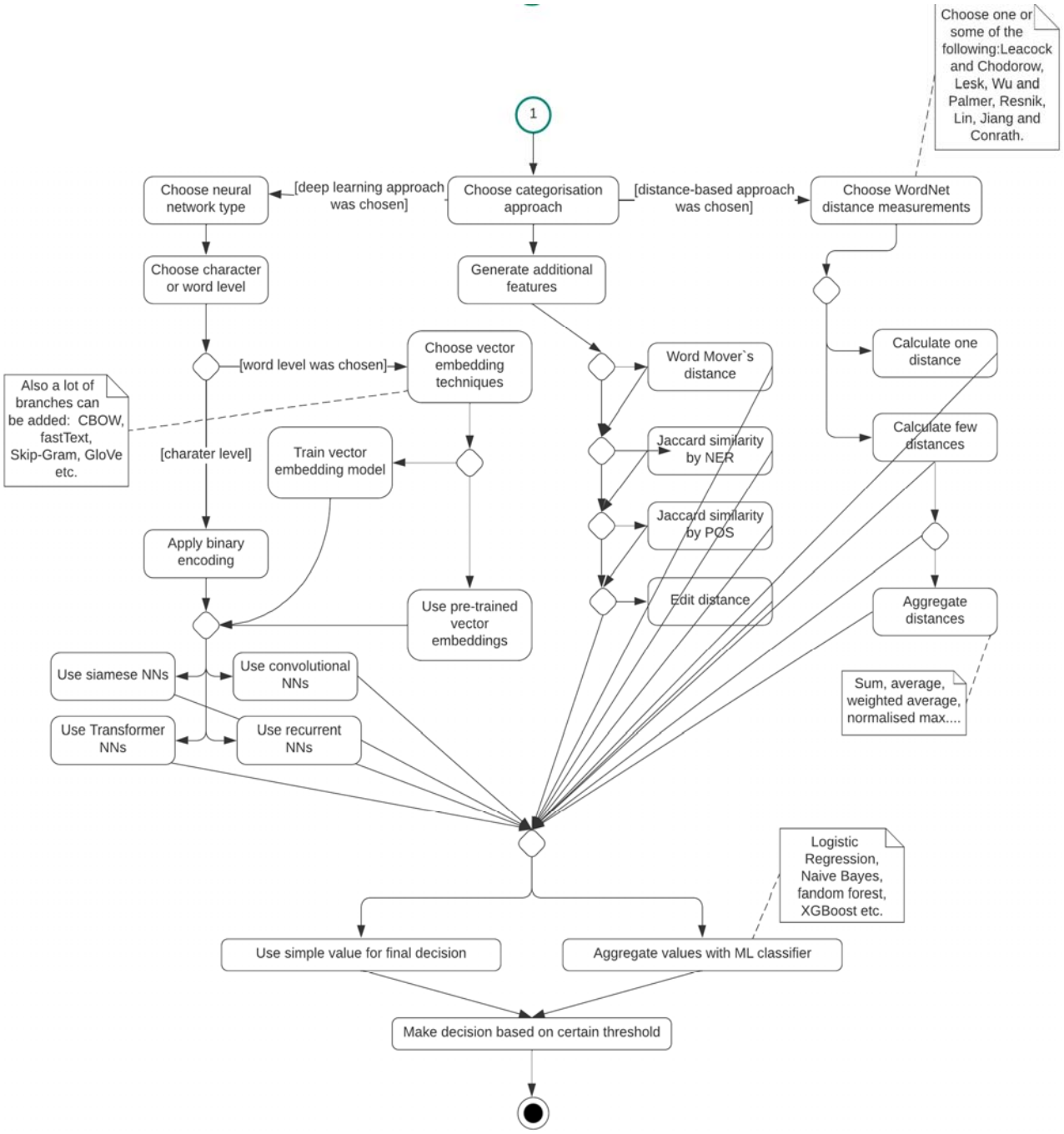


Рисунок 5 – Друга частина діаграми активності створення системи виявлення перефразувань

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

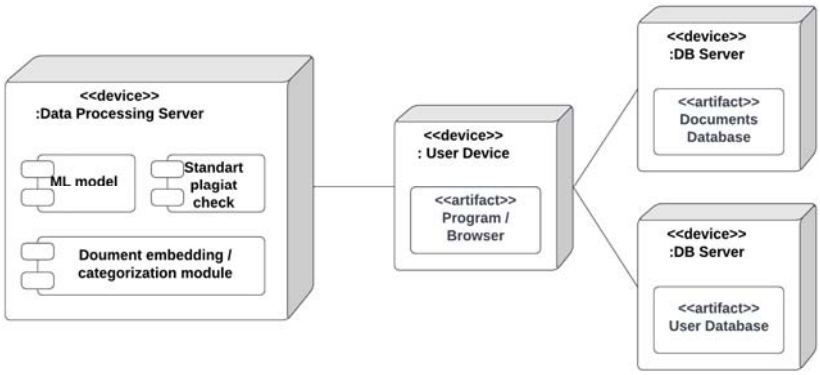


Рисунок 6 – Діаграма розгортання автоматичного виявлення перефразувань

відстань між векторними поданнями речень, Word Mover's Distance [43], відстані за словниками WordNet [1–2], передбачення двох ML-моделей: сіамської NN на основі рекурентних та типу Transformer – RoBERTa [46].

N-грам – послідовність з *n* слів. У контексті виявлення перефразувань у тексті кількість спільних N-грамів, нормована на загальну кількість *N*-грам у обох реченнях, допомагає виявити семантично подібні речення, що є близькими за семантичним навантаженням, однак не є перефразуваннями, оскільки друге речення отримане шляхом перестановки слів у першому реченні, і, відповідно, має зовсім відмінне значення. Коефіцієнт Жаккара для двох множин розраховуємо наступним чином:

$$J(A,B) = |A \cap B| / |A \cup B|.$$

Для розрахунку *N*-грам у кожному реченні речення спочатку приводяться до нижнього регістру, видаляються усі розділові і додаткові знаки. Усього розраховані коефіцієнти Жаккара для 2-, 3-, 4-грам.

Є велика різноманітність методів отримання векторних вкладень слів GloVe, Word2Vec: CBOW / Skip-Gram, fastText. У досліджені використані векторні вкладення моделі глибинного навчання BERT [42] для NLP архітектури Transformer. Базова модель BERT має 110 мільйонів параметрів, що налаштовуються, розширена версія – 345.

Особливістю архітектури Transformer є наявність механізму уваги, завдяки чому дані опрацьовують одночасно (на противагу рекурентним NN, де дані сприймаються послідовно). Окрім того, особливістю BERT є попереднє навчання NN для вирішення двох завдань: передбачення певного слова у реченні і визначення того, чи є друге речення логічним продовженням першого. У [42] ця ML-модель попередньо навчена на нерозмічених даних з BooksCorpus (800 мільйонів слів) та English Wikipedia (2 500 мільйонів слів). Попереднє навчання та механізм уваги дають змогу отримати контекстні векторні вкладення слів.

Використовуючи попередньо навчену модель, матриця векторного подання для кожного речення має таку розмірність: torch.Size ([1, 128, 768]), де 1 – кількість речень у batch, 128 – максимальна довжина речення, 768 – розмірність векторного вкладення. Для векторного подання речень дані усереднені для кожного слова у реченні, в результаті отримуємо вектор вкладення довжиною 768 значень.

Формула розрахунку косинусної відстані:

$$C = (\sum A_i B_i)/((\sum A_i^2)^{1/2} (\sum B_i^2)^{1/2}).$$

Дана ознака потенційно виявляє семантично подібні речення, однак речення з однаковими словами, що не є перефразуваннями, матимуть невелику косинусну відстань.

Word Mover's Distance застосовує векторні вкладення для розрахунку семантичної відстані між реченнями. WMD-відстань вимірює різницю між двома текстовими документами як мінімальну відстань, яку векторні вкладення слів одного документа повинні «подолати», щоб досягти точок векторного вкладення слів іншого документа. Аналогічно до попередньої ознаки, така метрика відстані дозволить виявити семантично подібні пари слів, однак не допоможе виявити приклади неперефразованих речень із переставленими словами.

Для виміру семантичної подібності речень використовують дві метрики семантичної відстані синсетів за словниками WordNet: Leacock and Chodorow [1], Wu and Palmer [2]. Семантична подібність слів за Leacock and Chodorow [7] визначається за допомого наступної формули:

$$\text{sim}_{l\_ch} = -\log (\text{length}/(2*D)).$$

де length – довжина найкоротшого шляху між двома концептами (кількість вузлів), *D* – максимальна глибина відповідної таксономії.

Функція семантичної відстані Wu and Palmer [9] залежить від глибини двох концептів $d(\text{concept}_i)$ у таксономії та глибини їх найближчого спільного предка $d(LCS)$:

$$\text{sim}_{wu\_p} = 2*d(LCS)/(d(\text{concept}_1)+ d(\text{concept}_2)).$$

Оскільки за WordNet відстань розраховується для кожної пари синсетів окремо, для подання відстані між двома реченнями використаний підхід з [3]. Для впровадження двонапрямленності використовують середнє арифметичне двох значень певної відстані, значення якої залежить від максимальної подібності

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

пари слів. Однак, у даному випадку не враховується специфічність слів (відсутнє множення на значення TF-IDF). Таким чином, для розрахунку відстаней між парою речень потрібно порівняти кожне слово першого із кожним словом другого.

Сіамські NN використовують особливі функції втрат, оскільки даний тип NN вивчає такі внутрішні векторні подання даних, що однакові записи матимуть малу косинусну або Евклідову відстань. У даному дослідженні у якості функції втрат contrastive loss розрахована для Евклідової відстані:

$$L = 0{,}5\,(1-Y)\,E^2 + 0{,}5\,Y\,\{\max(0, m-E)\}^2,$$

де E – значення Евклідової відстані між передбаченими векторними поданнями записів, Y – істинне значення, m – поріг приналежності записів до одного класу: передбачена відстань між векторними поданнями різних класів має бути не $< m$, $m = 2$.

4 ЕКСПЕРИМЕНТИ

Оскільки навчальна вибірка набору даних PAWS-Wiki містить 49 401 запис і розрахунок кожної ознаки вимагає значних обчислювальних ресурсів, то для кожної ознаки спочатку створений репозиторій із відповідним скриптом опрацювання та результатами розрахунків. Проект має таку структуру (рис. 7).

Для кожної ознаки є репозиторій distances, що містить відповідні результати опрацювання. Кожний такий репозиторій містить по три файли – результати опрацювання тестової, навчальної, валідаційної вибірок відповідно. Для навчання NN (сіамської з рекурентними зв'язками та RoBERTa) використані обчислювальні ресурси Google Colab, тому передбачення сіамської NN завантажені локально, а у випадку RoBERTa завантажені лише збережені ваги у форматі .ckpt, опрацювання і класифікація відбувалась локально. Файл merge.py створений для об'єднання усіх ознак (рис. 8), в результаті утворюється тип даних бібліотеки Pandas DataFrame. Ці ж дані зберігаються у форматі .csv у репозиторії data/features для подальшого опрацювання.

З рис. 9–13 видно, що передбачення NN RoBERTa майже повністю співпадають з істинними мітками, що відповідають парі слів. Оскільки інші показники не є цілочисельними і позначають за своїм принципом зовсім різні ознаки, для їх об'єднання був обраний метод саме логістичної регресії. Її навчання прописане у файлі train_model.py, а сама модель (як і інші класичні ML-методи, що порівнювались) імпортована з бібліотеки sklearn. Файл model_all_features.pkl містить ваги логістичної регресії, модель якої була навчена на всіх згенерованих ознаках. Подальше опрацювання текстів за допомогою натренованих ML-моделей вимагає об'єднання розрахунку усіх ознак, що виконано у файлу main.py. Із відповідних файлів імпортовані відповідні функції і попередньо навчені моделі.

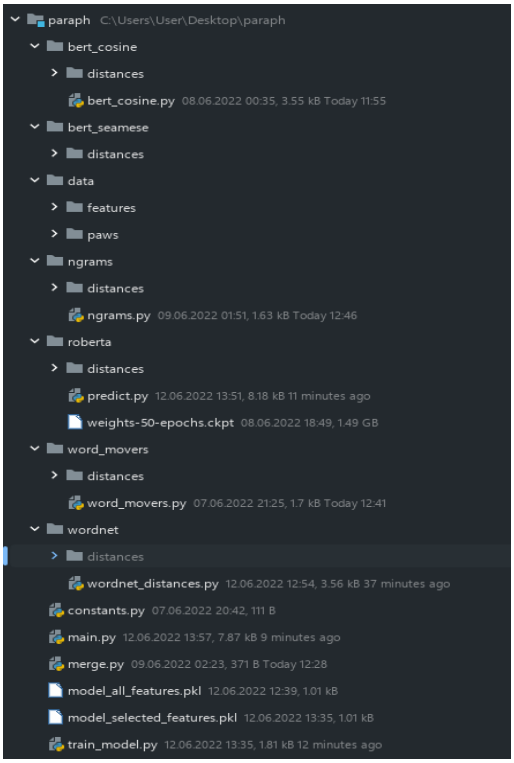


Рисунок 7 – Структура репозиторію проекту

```
import glob
import pandas as pd

for PART in ['dev', 'test', 'train']:

    files = glob.glob(f'C:/Users/User/Desktop/paraph/**/distances/{PART}_*.csv',
                      recursive=True)

    df = pd.concat(map(pd.read_csv, files), axis=1)
    df.drop(columns='id', inplace=True)

    df.to_csv(f'data/features/{PART}.csv', index=False)
    print('ready')
```

Рисунок 8 – Об'єднання файлів з ознаками

	id	sentence1	sentence2	label
1	1	This was a series of nested angular standards…	This was a series of nested polar scales , so…	0
2	2	His father emigrated to Missouri in 1868 but …	His father emigrated to America in 1868 , but…	0
3	3	In January 2011 , the Deputy Secretary Genera…	In January 2011 , FIBA Asia deputy secretary …	1
4	4	Steiner argued that , in the right circumstan…	Steiner held that the spiritual world can be …	0
5	5	Luciano Williames Dias (born July 25 , 1970 …	Luciano Williames Dias (born 25 July 1970) …	0
6	6	During her sophomore , junior and senior summ…	During her second , junior and senior summers…	1
7	7	The smallest number that can be represented i…	The smallest number that can be represented a…	0
8	8	His father emigrated to Missouri in 1868 , bu…	His father emigrated to Missouri in 1868 but …	1
9	9	The Villa Pesquera facilities are owned by th…	The facilities of Villa Pesquera are operated…	0
10	10	It is situated south of Köroğlu Mountains and…	It is situated south of Köroğlu - mountains a…	1

Рисунок 9 – Приклад вхідних даних

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

bert_cosine_distance	prediction_seamese_bert	3_grams_jaccard	2_grams_jaccard	4_grams_jaccard
0.99233836	0.2205543518066406	0.4117647058823529	0.5625	0.3142857142857143
0.98792297	0.4695497751235962	0.5517241379310345	0.8076923076923077	0.4827586206896552
0.9849439	0.6481984257698059	0.1724137931034483	0.44	0.0666666666666666
0.9759501	0.6250963807106018	0.2162162162162162	0.3823529411764705	0.1025641025641025
0.9834332	0.9546312689781188	0.375	0.5	0.25
0.9871587	0.7835149765014648	0.5769230769230769	0.72	0.5
0.98474044	0.856931209564209	0.2758620689655172	0.3928571428571428	0.2068965517241379
0.9874414	0.1621454060077667	0.8	0.88	0.72
0.9863758	1.0148042440414429	0.3478260869565217	0.4761904761904761	0.16
0.98016256	0.9825689196586608	0.4	0.5333333333333333	0.3571428571428571

Рисунок 10 – Відповідні вхідним даним ознаки

s_jaccard	predictions_raw	predictions	shortest_path_distance	wup_similarity	wm_distance
57142857143	0.0003076210268773	0	0.8808753618444751	0.9334199974323896	0.1638981500140085
86206896552	0.0001363550400128	0	0.9666656444455284	0.9749989694455368	0.1046792674400867
66666666666	0.9953380823135376	1	0.7899293829589504	0.8570820710088407	0.1082155853879519
41025641025	0.0146335149183869	0	0.8833327625003693	0.9286204319799678	0.258732279284138
0.25	0.004495037253946	0	0.9999987500015626	0.9999987500015626	0.0
0.5	0.9972121119499208	1	0.8849394652984164	0.9198043154376784	0.1023514166322542
65517241379	0.0016741530271247	0	0.9220770838260276	0.9545445867776484	0.1253511271784937
0.72	0.9963889122009276	1	0.9666656444455284	0.9749989694455368	0.1046792674400867
0.16	0.0004754920082632	0	0.999998819445843	0.999998819445843	0.0
28571428571	0.996845006942749	1	0.999998000004	0.999998000004	0.0

Рисунок 11 – Відповідні вхідним даним ознаки

```
from roberta.predict import Pairs_Dataset, config, Classifier_Model, predict
from transformers import AutoTokenizer
model_name = config['model_name']

tokenizer = AutoTokenizer.from_pretrained(model_name)

dataset_test = Pairs_Dataset(path, tokenizer, 'Quality', '#1 String', '#2 String')

BATCH_SIZE = config['batch_size']
test_loader = DataLoader(dataset_test, BATCH_SIZE, shuffle=False)

model_roberta = Classifier_Model.load_from_checkpoint('roberta/weights-50-epochs.ckpt', config=config)
model_roberta.to(device)

preditions_train_raw, predictions_train_int = predict(test_loader, model_roberta)
```

Рисунок 12 – Приклад класифікації за допомогою ML-моделі RoBERTa і імпортованих з інших файлів функцій

```
cos = get_cosine(df, 'sentence1', 'sentence2')
gr_2 = n_gr_sim(df, 2, 'sentence1', 'sentence2')
gr_3 = n_gr_sim(df, 3, 'sentence1', 'sentence2')
gr_4 = n_gr_sim(df, 4, 'sentence1', 'sentence2')
wn = wordnet_distance(df, 'sentence1', 'sentence2')
wm = word_movers(df, 'sentence1', 'sentence2')

X_test = pd.DataFrame({
    'bert_cosine_distance': cos,
    '2_grams_jaccard': gr_2,
    '3_grams_jaccard': gr_3,
    '4_grams_jaccard': gr_4,
    'predictions_raw': preditions_train_raw,
    'predictions': predictions_train_int,
    'shortest_path_distance': wn,
    'wm_distance': wm
})
```

Рисунок 13 – Створення таблиці з ознаками

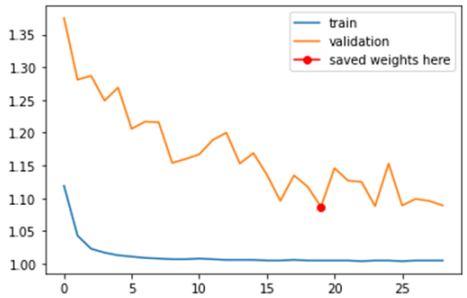


Рисунок 14 – Навчання сіамської NN

NN містить два шари двонапрямленої довгої короткочасної пам'яті (LSTM), два прихованих шари повнозв'язних нейронів (512 та 128 нейронів), вихідний шар, що містить 16 нейронів. Векторні вкладення слів були отримані шляхом прямого поширення у попередньо натренованій NN BERT. NN навчалась протягом 30 епох на навчальній вибірці із валідацією після кожної епохи (рис. 14).

Значна різниця між функціями втрат на навчальній та валідаційній вибірці свідчить про недостатню комплексність моделі (можливо, про недостатню кількість прихованих шарів або нейронів). Окрім того, точність класифікації може залежати від обраних гіперпараметрів, налаштування яких вимагає додаткового навчання і тестування моделі. Для подальшої класифікації ваги NN збережені за найменшого значення функції втрат (19-та епоха).

Попередньо навчена NN RoBERTa є покращеним аналогом BERT: основна її відмінність полягає у підборі гіперпараметрів під час попереднього навчання, збільшенні об'єму навчального корпусу

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

(16GB речень з Books Corpus та English Wikipedia, CommonCrawl News dataset (63 мільйони статей, 76 GB), Web text corpus (38 GB), Stories from Common Crawl (31 GB)), застосуванні динамічного маскування токенів для задачі передбачення слова у реченні: слово, яке необхідно передбачити у певному реченні, змінюється з кожною епохою. RoBERTa має 124 мільйони налаштовуваних параметрів (рис. 15). Оскільки результатом прямого поширення є матриця векторного вкладення речень розмірністю [batch size, max sentence length, embedding dimension], для подальшої класифікації додано прихований та повнозв'язний вихідний шар з 256 та 1 нейронами відповідно. Для знаходження векторного подання речень отримано середнє значення для кожної координати векторів вкладень. Результуюча матриця ваг між отриманими векторними поданнями слів та прихованим шаром із 256 нейронами матиме розмірність 768*256, між прихованим та вихідним шаром – 256*1. Активаційною функцією останнього шару є сигмоїда, функцією втрат – бінарна крос-ентропія.

	Name	Type	Params
0	pretrained_model	RobertaModel	124 M
1	dropout_layer	Dropout	0
2	hidden_layer	Linear	196 K
3	output_layer	Linear	257
4	loss_function	BCEWithLogitsLoss	0

Рисунок 15 – Загальна кількість параметрів NN

Попередньо навчена NN додатково налаштована (fine-tuned) для задачі виявлення перефразувань у тексті за допомогою навчальної та валідаційної вибірок. Усього NN додатково навчалась протягом 50 епох, ваги збережені за умови найменшого значення функції втрат на валідаційній вибірці (рис. 16).

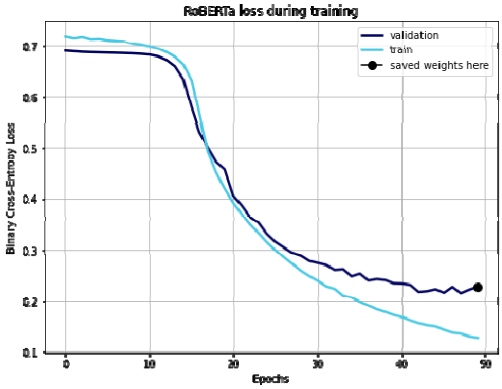


Рисунок 16 – Функція втрат під час навчання NN

Для фінального об'єднання ознак обрані як ймовірності приналежності двох записів до певного класу, так і самі передбачення. Поріг прийняття рішення = 0,5, тобто $\hat{y} = 1$ якщо p(y') > 0,5. Фінальна таблиця містить 10 ознак із вищезазначеними метриками для кожної пари речень, наявних у обраному наборі даних. Ці показники розраховані для навчальної, тестової і валідаційної вибірки.

5 РЕЗУЛЬТАТИ

Для зменшення навантаження на обчислювальні ресурси навчено іншу модель логістичної регресії без таких ознак: передбачення сіамської NN та відстані WordNet за Wu and Palmer. Точність класифікації на тестовій вибірці збільшилась з 92,4625% до 92,5%.

Для класифікації за допомогою NN RoBERTa дані подаються у наступному форматі: індекси слів, бінарна маска, що позначає частину корисного сигналу, істинна мітка, що відповідає парі записів як на рис. 17. Для отримання інших ознак дані приводяться до нижнього регістру, видаляються усі символи, окрім літер (рис. 18).

```
In [5]: dataset_test[0]
Out[5]: {'input_ids': tensor([    0,   713,    21,    10,   651,     9, 46902, 42970,  2820,  2156,
            98,    14, 19851,    11, 15001,   757,  5914,     8, 25361,   115,
            28,   626,  2024,    11, 13744, 34721,  5407,     7,     5,   364,
          3998, 10455,   636,   479,     2,     2,   713,    21,    10,   651,
             9, 46902, 13744, 21423,  2156,    98,    14, 19851,    11, 15001,
           757,  5914,     8, 25361,   115,    28,  3744,  2024,    11, 42970,
         34721,  5407,     7,     5,   364,  3998, 10455,   636,   479,     2,
             1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
             1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
             1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
             1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
             1,     1,     1,     1,     1,     1,     1,     1]),
 'attention_mask': tensor([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         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, 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]),
 'labels': tensor(0.)}
```

Рисунок 17 – Формат вхідних даних до нейронної мережі RoBERTa

bert_cosine_distance	2_grams_jaccard	3_grams_jaccard	4_grams_jaccard	predictions_raw	predictions	shortest_path_distance	wm_distance
0.992338	0.5625	0.411765	0.314286	0.000308	0	0	0.163898

Рисунок 18 – Отримані ознаки для тестової пари речень

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Для передбачення завантажуються ваги попередньо навченого класифікатора (логістичної регресії) як показано на рис. 19, рис. 20.

```
import pickle
loaded_model = pickle.load(open('model_selected_features.pkl', 'rb'))
predicted = loaded_model.predict(X_test)

print('Predicted label', predicted)
```

Рисунок 19 – Попередньо навчена модель

```
Predicted label [0]
```

Рисунок 20 – Результат передбачення

Результат класифікації, отриманий для наступного речення, подано на рис. 21. Логістична мережа правильно передбачила, що два речення не є перефразуваннями один одного, навіть попри те, що більшість слів у реченнях є спільними (рис. 22). Для наступного речення логістична регресія також правильно передбачила мітку, у цьому випадку пара речень є перефразуваннями один одного – подано на рис. 22.

```
['This was a series of nested angular standards , so that measurements in azimuth and elevation could be done directly in polar coordinates relative to the ecliptic .']
['This was a series of nested polar scales , so that measurements in azimuth and elevation could be performed directly in angular coordinates relative to the ecliptic .']
```

Рисунок 21 – Перша тестова пара речень

```
Predicted label 1
In [8]: print(df.loc[2, 'sentence1'])
   ...: print(df.loc[2, 'sentence2'])
   ...:
In January 2011 , the Deputy Secretary General of FIBA Asia , Hagop Khajirian , inspected the venue together with SBP - President Manuel V. Pangilinan .
In January 2011 , FIBA Asia deputy secretary general Hagop Khajirian along with SBP president Manuel V. Pangilinan inspected the venue .
```

Рисунок 22 – Друга тестова пара речень

Програма не потребує реалізації користувацького інтерфейсу, оскільки її використання заплановано лише у якості модулю системи виявлення плагіату або об'єднання згенерованого користувачами контенту.

6 ОБГОВОРЕННЯ

Для об'єднання ознак і фінальної класифікації обраний класичний ML-алгоритм – логістична регресія. Отримані результати такі результати:

– Точність на тестовому наборі даних – 92,462%, площа під ROC-кривою становить 97,05%, під кривою Precision-Recall – 94,96%.

– Точність на валідаційному наборі даних – 93,71% площа під ROC-кривою становить 97,66%, під кривою Precision-Recall – 96,12%.

Відповідно до показників влучності, повноти, $F_{0.5}$-міри (рис. 23) і матриці невідповідностей (рис. 24) результату класифікації тестового набору даних, логістична регресія помилково позначила майже вдвічі більше негативних записів як позитивні, ніж навпаки, позитивні – негативними. Даний результат може бути наслідком складності визначення не перефразованої пари речень, що були утворені в результаті заміни кількох слів місцями. Такі речення є дуже близькими семантично, однак мають абсолютно різне значення.

	precision	recall	f1-score	support
0	0.95	0.91	0.93	4464
1	0.89	0.94	0.92	3536
accuracy			0.92	8000
macro avg	0.92	0.93	0.92	8000
weighted avg	0.93	0.92	0.92	8000

Рисунок 23 – Повнота, влучність, F-міра для тестового набору даних

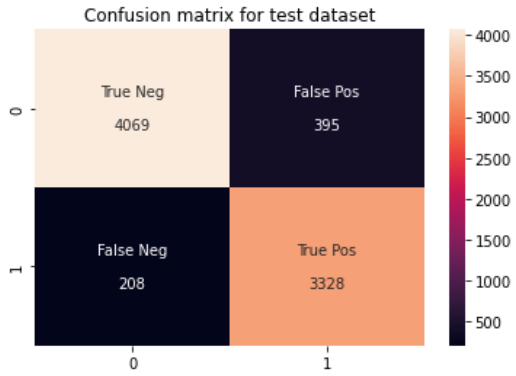


Рисунок 24 – Матриця невідповідностей для тестового набору даних

Відповідно до стовпчикової діаграми на рис. 25, найбільш важливими ознаками є передбачення ML-моделі RoBERTa, косинусна відстань між векторними поданнями рішень та найкоротший шлях між синсетами словника WordNet за Leacock and Chodorow. Значний коефіцієнт логістичної регресії (рис. 26), що відповідає передбаченням ML-моделі, свідчить про її потенційну можливість самостійно якісно класифікувати тексти без необхідності розрахунку додаткових ознак.

Без застосування методів глибинного навчання, використовуючи лише метод логістичної регресії для об'єднання результатів, отримуємо (рис. 27, рис. 28):

– Точність класифікації на тестовому наборі даних – 71,15%, площа під кривою Precision-Recall – 73,6%.

Найбільш важливою ознакою для класифікації є коефіцієнт Жаккарда для 3-грам (нормована кількість спільних 3-грам) та косинусна відстань між векторними поданнями речень.

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

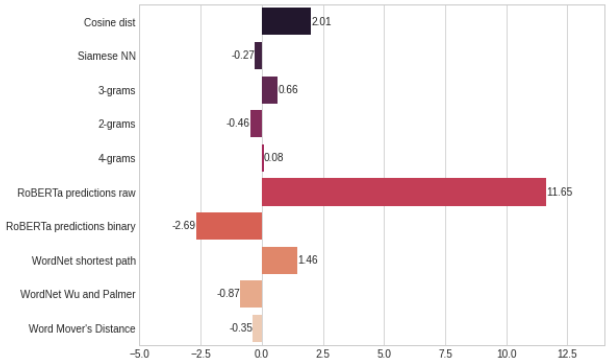


Рисунок 25 – Вагові коефіцієнти логістичної регресії, що відповідають певним ознакам семантичної подібності пари речень

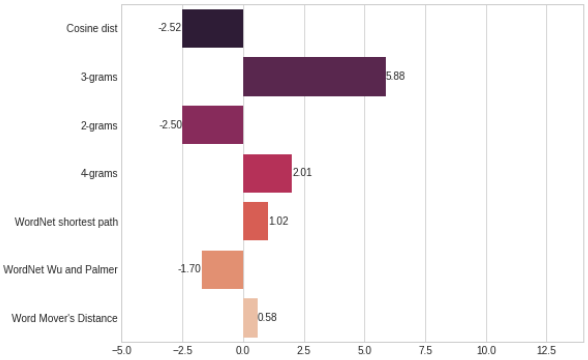


Рисунок 26 – Вагові коефіцієнти логістичної регресії, що відповідають певним ознакам семантичної подібності пари речень

На відміну від попереднього результату класифікації із використанням методів глибинного навчання, у цьому випадку вже більша частина позитивних випадків була класифікована як негативні, тобто модель логістичної регресії «має труднощі» із визначенням перефразувань.

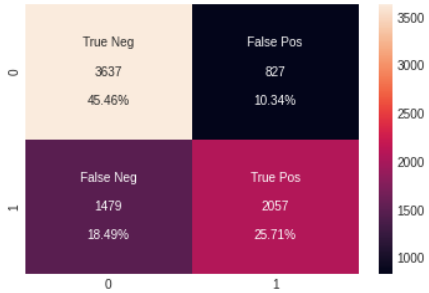


Рисунок 27 – Матриця невідповідностей для тестового набору даних

Використання лише однієї ознаки не є достатнім для якісної класифікації пари речень, оскільки передбачення такої моделі близькі за ймовірністю до «сліпого» вгадування: точність результатів класифікації тестового набору даних з використанням Word Mover's Distance становить 55,7%, косинусної відстані між векторними поданнями речень – 55,7%, передбачень сіамської NN – 58%, у той час як поєднання цих же метрик із коефіцієнтом Жаккарда для 3-грамів збільшує точність класифікації до 70%, а застосування коефіцієнту Жаккарда для 2-грам додає ще 1% точності.

Попри те, що дані ознаки можуть свідчити про семантичну подібність речень і можуть бути використані для виявлення пар подібних речень, вищезазначеної точності не є достатньо для повноцінного і якісного виявлення перефразувань.

Самостійне передбачення навченої ML-моделі RoBERTa є досить якісним і має високі показники якості: точність – 91,96%, площа під кривою Precision-Recall – 96,34%.

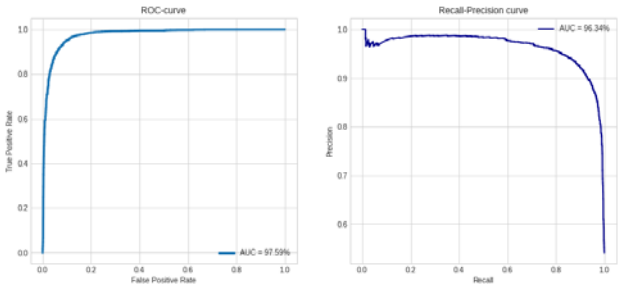
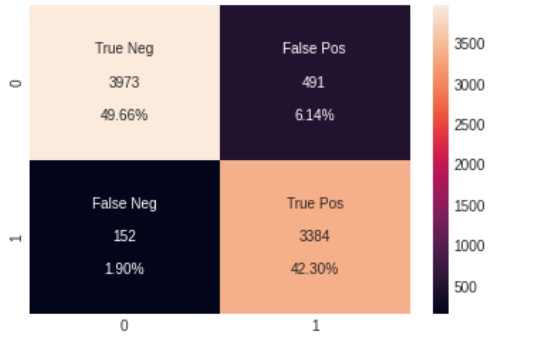


Рисунок 28 – Криві ROC та Precision-Recall для результатів класифікації моделі глибинного навчання RoBERTa

При цьому, RoBERTa значно більше (у 6 разів) класифікує негативні класи як позитивні (рис. 29), при цьому значно зменшуючи чутливість / повноту (recall) до пар речень, що не є перефразуваннями один одного.



	precision	recall	f1-score	support
0	0.96	0.89	0.93	4464
1	0.87	0.96	0.91	3536
accuracy			0.92	8000
macro avg	0.92	0.92	0.92	8000
weighted avg	0.92	0.92	0.92	8000

Рисунок 29 – Матриця невідповідностей, влучність, повнота, F-міра

При використанні такої моделі для виявлення плагіату більше робіт будуть неправильно вказані як першопочаткові джерела, при використанні для об'єднання згенерованого користувачами контенту (запитання, записи, теми на форумі) зростає ймовірність неправильно співвіднести певну кількість

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

записів. Окрім порівняння ML-методів також одночасно протестовано різну кількість ознак. Ознаки обрані в залежності від відповідної ваги моделі логістичної регресії. Найкраща точність була отримана для повного набору ознак та вищезазначеного алгоритму. Інші класичні ML-методи (протестовано наївний Байєсів класифікатор, метод опорних векторів, випадковий ліс, *k* найближчих сусідів багатошаровий перцептрон) не дають суттєвого приросту точності навчання. Щодо розробки проекту можна зробити такі висновки:

– для методу стекінгу моделей і ознак отримані високі показники точності, влучності, повноти, $F_{0.5}$-міри, площ під кривими ROC та Precision-Recall;

– «класичні» ознаки семантичної подібності, що використовуються у багатьох дослідженнях, справді здатні виявити семантично подібні речення, однак вони є «безсилими» у випадках, коли у реченні є переставлені слова (або є багато спільних слів) і, відповідно, друге речення не є перефразуванням першого і має зовсім інший зміст;

– основною передумовою такого результату є те, що сконструйовані ознаки не беруть до уваги порядок слів у реченні. Для боротьби з цим використовується коефіцієнт Жаккарда і об'єднання ознак за допомогою класичних ML-методів;

– попередньо навчені ML-моделі на основі архітектури Transformer показують відмінну точність класифікації. При цьому, навчена у процесі виконання роботи NN RoBERTa (із додатковими повнозв'язними шарами) має меншу чутливість до пар речень, що не є перефразуваннями один одного. Така специфічність моделі може сприяти неправильному звинуваченню у плагіаті або некоректному об'єднанню згенерованого користувачами контенту;

– перед розробниками ІС виявлення антиплагіату може постати питання вибору між точністю класифікації і збереженням обчислювальних ресурсів, оскільки розрахунок ознак також сприяє додатковому навантаженню на обчислювальний пристрій;

– NN архітектури Transformer не вимагають додаткової генерації ознак і здатні виявляти перефразування із досить високою точністю. Недоліком такого типу NN є значна кількість параметрів (великий час розрахунку результатів);

– Перевагою NN типу Transformer є попереднє навчання моделей для «розуміння мови» за допомогою завдань передбачення найбільш ймовірних слів у реченні та визначення того, чи є друге речення ідейним продовженням другого;

– для задачі виявлення перефразувань NN «дотренована» (fine-tuned) на наборі даних Paraphrase Adversaries from Word Scrambling. Перевагами обраного набору даних є 1) велика кількість навчальних записів – 49 тис. 2) збалансованість класів: 44.2% з усіх пар речень є перефразуваннями один одного. 3) частина прикладів була утворена шляхом заміни або перестановки слів, у такому випадку речення є близькими семантично, однак мають зовсім інші значення;

– NN типу Transformer застосовуються у багатьох NLP-задачах, їх також можна успішно використовувати для виявлення перефразувань у тексті з високою точністю. Єдиним недоліком такого типу мереж є значна кількість налаштовуваних параметрів – 110+ мільйонів, навчання такої NN і її застосування вимагають наявності значних обчислювальних ресурсів.

ВИСНОВКИ

Результатом роботи є розроблена ML-модель для виявлення перефразувань шляхом бінарної класифікації пари текстів. Розроблене ПЗ використовує принцип стекінгу моделей і інжиніринг ознак (feature engineering). Додаткові ознаки вказують на семантичну приналежність речень або нормовану кількість спільних N-грам. Створена модель показує відмінні результати класифікації на тестових даних PAWS: зважена влучність (precision) – 93%, зважена повнота (recall) – 92%, F-міра (F1-score)–92%, точність (accuracy) – 92%. Результати дослідження показали, що NN типу Transformer можуть бути успішно застосовані для виявлення перефразувань у парі текстів із досить високою точністю без потреби додаткового генерування ознак.

Додатково налаштована (fine-tuned) NN RoBERTa (із додатковими повнозв'язними шарами) має меншу чутливість до пар речень, що не є перефразуваннями один одного. Така специфічність моделі може сприяти неправильному звинуваченню у плагіаті або некоректному об'єднанню згенерованого користувачами контенту. Додаткові ознаки збільшують як загальну точність класифікації, так і чутливість моделі до пар тих речень, що не є перефразуваннями один одного.

ПОДЯКИ

Роботу виконано в рамках держбюджетної теми «Методи та засоби функціонування систем підтримки прийняття рішень на основі онтологій» (ID:839 2017-05-15 09:20:01 (2459-315)). Дослідження провадилось в межах спільних наукових досліджень кафедри інформаційних систем та мереж НУ «Львівська політехніка» на тему «Дослідження, розроблення і впровадження інтелектуальних розподілених інформаційних технологій та систем на основі ресурсів баз даних, сховищ даних, просторів даних та знань з метою прискорення процесів формування сучасного інформаційного суспільства». Наукові дослідження провадилися також в рамках ініціативної тематики досліджень кафедри ІСМ НУ «Львівська політехніка» на тему «Розроблення інтелектуальних розподілених систем на основі онтологічного підходу з метою інтеграції інформаційних ресурсів».

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

ЛІТЕРАТУРА / ЛИТЕРАТУРА

1. Salton G. A vector space model for automatic indexing / G. Salton, A. Wong, C.-S. Yang // Communications of the ACM. – 1975. – Vol. 18(11). – P. 613–620. DOI: 10.1145/361219.361220
2. Turney P. D. From Frequency to Meaning: Vector Space Models of Semantics / P. D. Turney, P. Pantel // Journal of Artificial Intelligence Research. – 2010. – Vol. 37(1). – P. 141–188. DOI: 10.1613/jair.2934
3. Efficient Estimation of Word Representations in Vector Space / [T. Mikolov, K. Chen, G. s Corrado, J. Dean] // ArXiv. – 2013. DOI: 10.48550/arXiv.1301.3781
4. Do Online Plagiarism Checkers Identify Paraphrased Content?. – DotNek Software Development, 2021. – https://www.dotnek.com/Blog/Marketing/do-online-plagiarism-checkers-identify-paraph
5. Introduction to WordNet: An On-line Lexical Database/ [G. Miller, R. Beckwith, C. Fellbaum, D. Gross, K. Miller] // International Journal of Lexicography. – 1990. – Vol. 3(4). – P. 235–244. DOI: 10.1093/ijl/3.4.235
6. Corley C. Measuring the Semantic Similarity of Texts / C. Corley, R. Mihalcea // ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, Michigan, Jun. 2005 : proceedings. – P. 13–18.
7. Leacock C. Combining Local Context and WordNet Similarity for Word Sense Identification / C. Leacock, M. Chodorow // WordNet: An Electronic Lexical Database. – 1998. – Vol. 49(2). – P. 265–283. DOI: 10.7551/mitpress/7287.003.0018
8. Lesk M. E. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone / M. E. Lesk // SIGDOC '86: the 5th Annual International Conference on Systems documentation, Toronto, Ontario, Canada, June 1986 : proceedings. – P. 24–26. DOI: 10.1145/318723.318728
9. Wu Z. Verbs Semantics and Lexical Selection / Z. Wu, M. Palmer // ACL '94: the 32nd annual meeting on Association for Computational Linguistics, Las Cruces, New Mexico, June 27 – 30, 1994 : proceedings. – P. 133–138. DOI: 10.3115/981732.981751.
10. Resnik P. Using Information Content to Evaluate Semantic Similarity in a Taxonomy / P. Resnik // ArXiv. – 1995. DOI: 10.48550/arXiv.cmp-lg/9511007
11. Lin D. An Information-Theoretic Definition of Similarity / D. Lin // ICML, 1998. – https://www.cse.iitb.ac.in/~cs626-449/Papers/WordSimilarity/3.pdf
12. Jiang J. J. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy / J. J. Jiang, D. W. Conrath // 10th Research on Computational Linguistics International Conference, Taipei, Taiwan, Aug. 1997 : proceedings. – P. 19–33.
13. Mihalcea R. Corpus-based and Knowledge-based Measures of Text Semantic Similarity / R. Mihalcea, C. Corley, C. Strapparava // AAAI'06: Proceedings of the 21st national conference on Artificial intelligence, Boston, Massachusetts, July 2006 : proceedings. – Vol. 1. – P. 775–780.
14. Hassan S. Semantic Relatedness Using Salient Semantic Analysis / S. Hassan, R. Mihalcea // AAAI 2011: Twenty-Fifth AAAI Conference on Artificial Intelligence, San Francisco, California, USA, August 7–11, 2011 : proceedings. – https://web.eecs.umich.edu/~mihalcea/papers/hassan.aaai11.pdf
15. Fernando S. A Semantic Similarity Approach to Paraphrase Detection / S. Fernando, M. Stevenson // 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics, May 2008 : proceedings. – P. 45–52.
16. Milajevs D. Evaluating Neural Word Representations in Tensor-Based Compositional Settings / [D. Milajevs, D. Kartsaklis, M. Sadrzadeh, M. Purver] // ArXiv. – 2014. DOI: 10.48550/arXiv.1408.6179
17. Islam A. Semantic similarity of short texts / A. Islam, D. Inkpen // Current Issues in Linguistic Theory: Recent Advances in Natural Language Processing. – 2009. – Vol. 309. – P. 227–236. DOI: 10.1075/cilt.309.18isl
18. Chong M. Using Natural Language Processing for Automatic Detection of Plagiarism / M. Chong, L. Specia, R. Mitkov // IPC2010 : 4th International Plagiarism Conference, Newcastle-upon-Tyne, May 2010 : proceedings. – https://www.academia.edu/326444/Using_Natural_Language_Processing_for_Automatic_Detection_of_Plagiarism
19. TakeLab: Systems for Measuring Semantic Text Similarity / [F. Šarić, G. Glavaš, M. Karan, J. Šnajder, B. Dalbelo Bašić] // *SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), Montréal, Canada, May 2012 : proceedings. – P. 441–448.
20. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity / [E. Agirre, D. Cer, M. Diab, A. Gonzalez-Agirre] // *SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), Montréal, Canada, May 2012 : proceedings. – P. 385–393.
21. Detecting High Obfuscation Plagiarism: Exploring Multi-Features Fusion via Machine Learning / [L. Kong, Z. Lu, H. Qi, Z. Han] // International Journal of u- and e- Service, Science and Technology. – 2014. – Vol. 7. – P. 385–396. DOI: 10.14257/ijunnesst.2014.7.4.35
22. Yin W. Convolutional Neural Network for Paraphrase Identification / W. Yin, H. Schütz // North American Chapter of the Association for Computational Linguistics: Human Language Technologies Conferences, Denver, Colorado, May 2015 : proceedings. – P. 901–911. DOI: 10.3115/v1/N15-1091.
23. Qiu L. Paraphrase Recognition via Dissimilarity Significance Classification / L. Qiu, M.-Y. Kan, T.-S. Chua // Empirical Methods in Natural Language Processing Conference, Sydney, Australia, Jul. 2006 : proceedings. – P. 18–26.
24. Kozareva Z. Paraphrase Identification on the Basis of Supervised Machine Learning Techniques / Z. Kozareva, A. Montoyo // Advances in Natural Language Processing. FinTAL 2006. Lecture Notes in Computer Science. – 2006. – Vol. 4139. – P. 524–533. DOI: 10.1007/11816508_52
25. Finch A. Using machine translation evaluation techniques to determine sentence-level semantic equivalence / A. Finch, E. Sumita // IWP2005 : 3rd International Workshop on Paraphrasing, May 2005 : proceedings. – P. 17–24.
26. Madnani N. Re-examining Machine Translation Metrics for Paraphrase Identification / N. Madnani, J. Tetreault, M. Chodorow // North American Chapter of the Association for Computational Linguistics: Human Language

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Technologies Conference of the, Montréal, Canada, June 2012 : proceedings. – P. 182–190.
27. A Deep Network Model for Paraphrase Detection in Short Text Messages / B. Agarwal, H. Ramampiaro, H. Langseth, M. Ruocco // ArXiv. – 2017. DOI: 10.48550/arXiv.1712.02820
28. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection / R. Socher, E. H.-C. Huang, J. Pennington, A. Ng, C. D. Manning // NIPS'11: the 24th International Conference on Neural Information Processing Systems, Granada Spain, December 2011 : proceedings. – P. 801–809.
29. Thyagarajan A. Siamese Recurrent Architectures for Learning Sentence Similarity / A. Thyagarajan // The Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, February 12–17, 2016 : proceedings. – Vol. 30(1). – P. 2786–2792. DOI: 10.1609/aaai.v30i1.10350
30. Neculoiu P. Learning Text Similarity with Siamese Recurrent Networks / P. Neculoiu, M. Versteegh, M. Rotaru // Workshop on Representation Learning for NLP, Berlin, Germany, August 2016 : proceedings. – P. 148–157. DOI: 10.18653/v1/W16-1617
31. Ranasinghe T. Semantic Textual Similarity with Siamese Neural Networks / T. Ranasinghe, C. Orasan, R. Mitkov // RANLP 2019 : International Conference on Recent Advances in Natural Language Processing, Varna, Bulgaria, Sep. 2019 : proceedings. – P. 1004–1011. DOI: 10.26615/978-954-452-056-4_116
32. Mahmoud A. BLSTM-API: Bi-LSTM Recurrent Neural Network-Based Approach for Arabic Paraphrase Identification / A. Mahmoud, M. Zrigui // Arabian Journal for Science and Engineering. – 2021. – Vol. 46. – P. 4163–4174. DOI: 10.1007/s13369-020-05320-w
33. Reddy D. LSTM Based Paraphrase Identification Using Combined Word Embedding Features / D. Reddy, M. Kumar, S. Kp // Computing and Signal Processing. Advances in Intelligent Systems and Computing. – 2019. – Vol. 898. – P. 385–394. DOI: 10.1007/978-981-13-3393-4_40
34. Paraphrase Generation with Deep Reinforcement Learning / [Z. Li, X. Jiang, L. Shang, H. Li] // ArXiv. – 2017. DOI: 10.48550/arXiv.1711.00279
35. Gomaa W. SimAll: A flexible tool for text similarity / W. Gomaa, A. Fahmy // ESOLEC' 2017 : The Seventeenth Conference on Language Engineering, December 2017 : proceedings. https://www.academia.edu/35381793/SimAll_A_flexible_tool_for_text_similarity
36. Ahmed M. Improving Tree-LSTM with Tree Attention / M. Ahmed, M. R. Samee, R. E. Mercer // ArXiv. – 2019. DOI: 10.48550/arXiv.1901.00066
37. Pontes E. L. Predicting the Semantic Textual Similarity with Siamese CNN and LSTM / E. L. Pontes, S. Huet, A. C. Linhares, J.-M. Torres-Moreno // Actes de la Conférence TALN, Rennes, France, May 2018 : proceedings. – P. 311–320.
38. Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection / [J. P. Wahle, T. Ruas, N. Meuschke, B. Gipp] // ArXiv. – 2021. DOI: 10.48550/arXiv.2103.12450
39. Attention Is All You Need / [A. Vaswani, N. Shazeer, N. Parmar et al.] // ArXiv. – 2017. DOI: 10.48550/arXiv.1706.0376
40. Nighojkar A. Improving Paraphrase Detection with the Adversarial Paraphrasing Task / A. Nighojkar, J. Licato // ArXiv. – 2021. DOI: 10.48550/arXiv.2106.07691
41. Arase Y. Transfer fine-tuning of BERT with phrasal paraphrases / Y. Arase, J. Tsujii // ArXiv. – 2021. DOI: 10.48550/arXiv.1909.00931
42. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / [J. Devlin, M.-W. Chang, K. Lee, K. Toutanova] // ArXiv. – 2018. DOI: 10.48550/arXiv.1810.04805
43. From Word Embeddings to Document Distances / [M. J. Kusner, Y. Sun, N. I. Kolkin, K. Q. Weinberger] // JMLR: W&CP. – 2015. – Vol. 37. – P. 957–966.
44. Zhang Y. PAWS: Paraphrase Adversaries from Word Scrambling / Y. Zhang, J. Baldridge, L. He // ArXiv. – 2019. DOI: 10.48550/arXiv.1904.01130
45. Propaganda Detection in Text Data Based on NLP and Machine Learning / [V.-A. Oliinyk, V. Vysotska, Y. Burov, et al.] // Modern Machine Learning Technologies and Data Science (MoMLeT+DS 2020) : Workshop, Lviv-Shatsk, 2–3 June 2020 : CEUR workshop proceedings. – Aachen: CEUR-WS.org, 2020. – Vol. 2631. – P. 132–144.
46. RoBERTa: A Robustly Optimized BERT Pretraining Approach / [Y. Liu, M. Ott, N. Goyal et al.] // ArXiv. – 2019. DOI: 10.48550/arXiv.1907.11692

Стаття надійшла до редакції 23.06.2022.
Після доробки 28.09.2022.

УДК 004.9

ТЕХНОЛОГИЯ ИДЕНТИФИКАЦИИ РЕРАЙТА В ТЕКСТОВОМ КОНТЕНТЕ НА ОСНОВЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

Холодна Н. М. – студент кафедры «Информационные системы и сети», Национальный университет «Львовская политехника», Украина.

Высоцка В. А. – канд. техн. наук, доцент, доцент кафедры «Информационные системы и сети», Национальный университет «Львовская политехника», Украина.

АННОТАЦИЯ

Актуальность. Перефразированный текстовый контент или рерайт является одной из сложных проблем выявления академического плагиата. Большинство систем идентификации плагиата предназначены для выявления общих слов, последовательности лингвистических единиц и незначительных изменений, но не способны выявить существенные семантические и структурные изменения. Поэтому большинство случаев плагиата с применением перефразирования остаются незамеченными.

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

Целью исследования является разработка технологии обнаружения перефразировок в тексте на основе модели классификации и методов машинного обучения через использование сиамской нейронной сети на основе рекуррентных и типа Transformer – RoBERTa для анализа уровня подобия предложений текстового контента.

Метод. Для данного исследования в качестве признаков выбраны следующие метрики семантического подобия или показатели: коэффициент Жаккара для общих N-грамм, косинусное расстояние между векторными представлениями предложений, Word Mover's Distance, расстояния по словарям WordNet, предсказание двух ML-моделей: сиамской нейронной сети на основе рекуррентных и типа Transformer – RoBERTa.

Результаты. Разработана интеллектуальная система выявления перефразировок в тексте на основе модели классификации и методов машинного обучения. Разработанная система использует принцип стекинговых моделей и инжиниринг признаков (feature engineering). Дополнительные признаки указывают на семантическую принадлежность предложений или нормированное количество общих N-грамм. Дополнительно настроенная (fine-tuned) нейронная сеть RoBERTa (с дополнительными полносвязными слоями) имеет меньшую чувствительность к парам предложений, не являющимся перефразированием друг друга. Такая специфичность модели может способствовать неправильному обвинению в плагиате или некорректном объединении сгенерированного пользователями контента. Дополнительные признаки увеличивают как общую точность классификации, так и чувствительность модели к парам тех предложений, которые не являются перефразированием друг друга.

Выводы. Созданная модель показывает отличные результаты классификации на тестовых данных PAWS: взвешенная точность (precision) – 93%, взвешенная полнота (recall) – 92%, F-мера (F1-score) – 92%, точность (accuracy) – 92%. Результаты исследования показали, что NN типа Transformer могут быть успешно применены для обнаружения перефразирования в паре текстов с достаточно высокой точностью без необходимости дополнительного генерирования признаков.

КЛЮЧЕВЫЕ СЛОВА: обработка природного языка, NLP, идентификация рерайта, выявление перефразировок в тексте, машинное обучение с учителем, глубинное обучение, классификация текста, анализ текста, векторное вложение слов, WordNet, семантическое сходство.

UDC 004.9

REWRITING IDENTIFICATION TECHNOLOGY FOR TEXT CONTENT BASED ON MACHINE LEARNING METHODS

Kholodna N. – Student of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine.

Vysotska V. – PhD, Associate Professor of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine.

ABSTRACT

Context. Paraphrased textual content or rewriting is one of the difficult problems of detecting academic plagiarism. Most plagiarism detection systems are designed to detect common words, sequences of linguistic units, and minor changes, but are unable to detect significant semantic and structural changes. Therefore, most cases of plagiarism using paraphrasing remain unnoticed.

Objective of the study is to develop a technology for detecting paraphrasing in text based on a classification model and machine learning methods through the use of Siamese neural network based on recurrent and Transformer type – RoBERTa to analyze the level of similarity of sentences of text content.

Method. For this study, the following semantic similarity metrics or indicators were chosen as features: Jacquard coefficient for shared N-grams, cosine distance between vector representations of sentences, Word Mover's Distance, distances according to WordNet dictionaries, prediction of two ML models: Siamese neural network based on recurrent and Transformer type - RoBERTa.

Results. An intelligent system for detecting paraphrasing in text based on a classification model and machine learning methods has been developed. The developed system uses the principle of model stacking and feature engineering. Additional features indicate the semantic affiliation of the sentences or the normalized number of common N-grams. An additional fine-tuned RoBERTa neural network (with additional fully connected layers) is less sensitive to pairs of sentences that are not paraphrases of each other. This specificity of the model may contribute to incorrect accusations of plagiarism or incorrect association of user-generated content. Additional features increase both the overall classification accuracy and the model's sensitivity to pairs of sentences that are not paraphrases of each other.

Conclusions. The created model shows excellent classification results on PAWS test data: precision – 93%, recall – 92%, F1-score – 92%, accuracy – 92%. The results of the study showed that Transformer-type NNs can be successfully applied to detect paraphrasing in a pair of texts with fairly high accuracy without the need for additional feature generation.

KEYWORDS: natural language processing, NLP, rewrite identification, detection of paraphrasing in text, supervised machine learning, deep learning, text classification, text analysis, word embeddings, WordNet, semantic similarity.

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

REFERENCES

1. Salton G., Wong A., Yang C.-S. A vector space model for automatic indexing, *Communications of the ACM,* 1975, Vol. 18(11), pp. 613–620. DOI: 10.1145/361219.361220
2. Turney P. D., Pantel P. From Frequency to Meaning: Vector Space Models of Semantics, *Journal of Artificial Intelligence Research,* 2010, Vol. 37(1), pp. 141–188. DOI: 10.1613/jair.2934
3. Mikolov T., Chen K., Corrado G. s, Dean J. Efficient Estimation of Word Representations in Vector Space, *ArXiv*, 2013. DOI: 10.48550/arXiv.1301.3781
4. Do Online Plagiarism Checkers Identify Paraphrased Content? *DotNek Software Development,* 2021, https://www.dotnek.com/Blog/Marketing/do-online-plagiarism-checkers-identify-paraph
5. Miller G., Beckwith R., Fellbaum C., Gross D., Miller K. Introduction to WordNet: An On-line Lexical Database, *International Journal of Lexicography,* 1990, Vol. 3(4), pp. 235–244. DOI: 10.1093/ijl/3.4.235
6. Corley C., Mihalcea R. Measuring the Semantic Similarity of Texts, *ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, Michigan, Jun. 2005, proceedings,* pp. 13–18.
7. Leacock C., Chodorow M. Combining Local Context and WordNet Similarity for Word Sense Identification, *WordNet: An Electronic Lexical Database,* 1998, Vol. 49(2), pp. 265–283. DOI: 10.7551/mitpress/7287.003.0018
8. Lesk M. E. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone, *SIGDOC '86: the 5th Annual International Conference on Systems documentation*. Toronto, Ontario, Canada, June 1986, proceedings, pp. 24–26. DOI: 10.1145/318723.318728
9. Wu Z., Palmer M. Verbs Semantics and Lexical Selection, *ACL '94: the 32nd annual meeting on Association for Computational Linguistics.*Las Cruces, New Mexico, June 27–30, 1994, proceedings, pp. 133–138. DOI: 10.3115/981732.981751.
10. Resnik P. Using Information Content to Evaluate Semantic Similarity in a Taxonomy, *ArXiv*, 1995. DOI: 10.48550/arXiv.cmp-lg/9511007
11. Lin D. An Information-Theoretic Definition of Similarity, ICML, 1998, https://www.cse.iitb.ac.in/~cs626-449/Papers/WordSimilarity/3.pdf
12. Jiang J. J., Conrath D. W. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy, *10th Research on Computational Linguistics International Conference*. Taipei, Taiwan, Aug. 1997, proceedings, pp. 19–33.
13. Mihalcea R. Corley C., Strapparava C. Corpus-based and Knowledge-based Measures of Text Semantic Similarity, *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence, Boston, Massachusetts, July 2006 : proceedings*, Vol. 1, pp. 775–780.
14. Hassan S., Mihalcea R. Semantic Relatedness Using Salient Semantic Analysis, *AAAI 2011: Twenty-Fifth AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, August 7–11, 2011, proceedings, https://web.eecs.umich.edu/~mihalcea/papers/hassan.aaai11.pdf
15. Fernando S., Stevenson M. A Semantic Similarity Approach to Paraphrase Detection, *11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics, May 2008, proceedings*, pp. 45–52.
16. Milajevs D., Kartsaklis D., Sadrzadeh M., Purver M. Evaluating Neural Word Representations in Tensor-Based Compositional Settings, *ArXiv,* 2014. DOI: 10.48550/arXiv.1408.6179
17. Islam A., Inkpen D. Semantic similarity of short texts, *Current Issues in Linguistic Theory: Recent Advances in Natural Language Processing,* 2009, Vol. 309, pp. 227–236. DOI: 10.1075/cilt.309.18isl
18. Chong M., Specia L., Mitkov R. Using Natural Language Processing for Automatic Detection of Plagiarism, *IPC2010, 4th International Plagiarism Conference, Newcastle-upon-Tyne, May 2010, proceedings,* https://www.academia.edu/326444/Using_Natural_Language_Processing_for_Automatic_Detection_of_Plagiarism
19. Šarić F., Glavaš G., Karan M., Šnajder J., Dalbelo Bašić B. TakeLab: Systems for Measuring Semantic Text Similarity, **SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, Volume 1, Proceedings of the main conference and the shared task, and Volume 2, Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). Montréal, Canada, May 2012, proceedings, pp. 441–448.
20. Agirre E., Cer D., Diab M., Gonzalez-Agirre A. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity,**SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, Volume 1, Proceedings of the main conference and the shared task, and Volume 2, Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). Montréal, Canada, May 2012, proceedings, pp. 385–393.
21. Kong L., Lu Z., Qi H., Han Z. Detecting High Obfuscation Plagiarism: Exploring Multi-Features Fusion via Machine Learning, *International Journal of u- and e- Service, Science and Technology*, 2014, Vol. 7, pp. 385–396. DOI: 10.14257/ijunnesst.2014.7.4.35
22. Yin W., Schütz H. Convolutional Neural Network for Paraphrase Identification, *North American Chapter of the Association for Computational Linguistics: Human Language Technologies Conferences.* Denver, Colorado, May 2015, proceedings, pp. 901–911. DOI: 10.3115/v1/N15-1091.
23. Qiu L., Kan M.-Y., Chua T.-S. Paraphrase Recognition via Dissimilarity Significance Classification, *Empirical Methods in Natural Language Processing Conference.* Sydney, Australia, Jul. 2006, proceedings, pp. 18–26.
24. Kozareva Z., Montoyo A. Paraphrase Identification on the Basis of Supervised Machine Learning Techniques, *Advances in Natural Language Processing. FinTAL 2006. Lecture Notes in Computer Science*, 2006, Vol. 4139, pp. 524–533. DOI: 10.1007/11816508_52
25. Finch A., Sumita E. Using machine translation evaluation techniques to determine sentence-level semantic equivalence, *IWP2005, 3rd International Workshop on Paraphrasing, May 2005, proceedings*, pp. 17–24.
26. Madnani N., Tetreault J., Chodorow M. Re-examining Machine Translation Metrics for Paraphrase Identification, *North American Chapter of the Association for Computational Linguistics, Human Language Technologies Conference of the, Montréal*. Canada, June 2012, proceedings, pp. 182–190.
27. Agarwal B., Ramampiaro H., Langseth H., Ruocco M. A Deep Network Model for Paraphrase Detection in Short Text Messages, *ArXiv,* 2017. DOI: 10.48550/arXiv.1712.02820

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11

28. Socher R., Huang E. H.-C., Pennington J., Ng A., Manning C. D. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, *NIPS'11: the 24th International Conference on Neural Information Processing Systems*, Granada Spain, December 2011, proceedings, pp. 801–809.
29. Thyagarajan A. Siamese Recurrent Architectures for Learning Sentence Similarity, *The Thirtieth AAAI Conference on Artificial Intelligence,* Phoenix, Arizona, USA, February 12–17, 2016, proceedings, Vol. 30(1), pp. 2786–2792. DOI: 10.1609/aaai.v30i1.10350
30. Neculoiu P., Versteegh M., Rotaru M. Learning Text Similarity with Siamese Recurrent Networks, *Workshop on Representation Learning for NLP*. Berlin, Germany, August 2016, proceedings, pp. 148–157. DOI: 10.18653/v1/W16-1617
31. Ranasinghe T., Orasan C., Mitkov R. Semantic Textual Similarity with Siamese Neural Networks, RANLP 2019, *International Conference on Recent Advances in Natural Language Processing*. Varna, Bulgaria, Sep. 2019, proceedings, pp. 1004–1011. DOI: 10.26615/978-954-452-056-4_116
32. Mahmoud A., Zrigui M. BLSTM-API: Bi-LSTM Recurrent Neural Network-Based Approach for Arabic Paraphrase Identification, *Arabian Journal for Science and Engineering*, 2021, Vol. 46, pp. 4163–4174. DOI: 10.1007/s13369-020-05320-w
33. Reddy D., Kumar M., Kp S. LSTM Based Paraphrase Identification Using Combined Word Embedding Features, *Computing and Signal Processing. Advances in Intelligent Systems and Computing*, 2019, Vol. 898, pp. 385–394. DOI: 10.1007/978-981-13-3393-4_40
34. Li Z., Jiang X., Shang L., Li H. Paraphrase Generation with Deep Reinforcement Learning, *ArXiv,* 2017. DOI: 10.48550/arXiv.1711.00279
35. Gomaa W., Fahmy A. SimAll: A flexible tool for text similarity, *ESOLEC' 2017, The Seventeenth Conference on Language Engineering, December 2017*, proceedings. https://www.academia.edu/35381793/SimAll_A_flexible_tool_for_text_similarity
36. Ahmed M., Samee M. R., Mercer R. E. Improving Tree-LSTM with Tree Attention, *ArXiv,* 2019. DOI: 10.48550/arXiv.1901.00066
37. Pontes E. L., Huet S., Linhares A. C., Torres-Moreno J.-M. Predicting the Semantic Textual Similarity with Siamese CNN and LSTM, *Actes de la Conférence TALN*. Rennes, France, May 2018, proceedings, pp. 311–320.
38. Wahle J. P., Ruas T., Meuschke N., Gipp B. Are Neural Language Models Good Plagiarists? A Benchmark for Neural Paraphrase Detection, *ArXiv,* 2021. DOI: 10.48550/arXiv.2103.12450
39. Vaswani A., Shazeer N., Parmar N., J. Uszkoreit, Jones L., Gomez A. N., Kaiser Ł., Polosukhin I. Attention Is All You Need, *ArXiv*, 2017. DOI: 10.48550/arXiv.1706.0376
40. Nighojkar A., Licato J. Improving Paraphrase Detection with the Adversarial Paraphrasing Task, *ArXiv,* 2021. DOI: 10.48550/arXiv.2106.07691
41. Arase Y., Tsujii J. Transfer fine-tuning of BERT with phrasal paraphrases, *ArXiv,* 2021. DOI: 10.48550/arXiv.1909.00931
42. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *ArXiv,* 2018. DOI: 10.48550/arXiv.1810.04805
43. Kusner M. J., Sun Y., Kolkin N. I., Weinberger K. Q. From Word Embeddings to Document Distances, *JMLR: W&CP*, 2015, Vol. 37, pp. 957–966.
44. Zhang Y., Baldridge J., He L. PAWS: Paraphrase Adversaries from Word Scrambling, *ArXiv,* 2019. DOI: 10.48550/arXiv.1904.01130
45. Oliinyk V.-A., Vysotska V., Burov Y., Mykich K., Fernandes V. B. Propaganda Detection in Text Data Based on NLP and Machine Learning, *Modern Machine Learning Technologies and Data Science (MoMLeT+DS 2020), Workshop*, *Lviv-Shatsk, 2–3 June 2020, CEUR workshop proceedings.* Aachen, CEUR-WS.org, 2020, Vol. 2631, pp. 132–144.
46. Liu Y., Ott M., Goyal N., Du J., Joshi M., Chen D., Levy O., Lewis M., Zettlemoyer L., Stoyanov V. RoBERTa: A Robustly Optimized BERT Pretraining Approach, *ArXiv,* 2019. DOI: 10.48550/arXiv.1907.11692

© Холодна Н. М., Висоцька В. А., 2022
DOI 10.15588/1607-3274-2022-4-11