УДК 681.3.093:044.3
DOI https://doi.org/10.32838/2663-5941/2019.3-1/38

Яременко В.С.
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Тарасенко М.В.
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

ПОРІВНЯЛЬНИЙ АНАЛІЗ ПРОГРАМНИХ БІБЛІОТЕК ДЛЯ КЛАСИФІКАЦІЇ ТЕКСТОВИХ ДАНИХ ІЗ ВИКОРИСТАННЯМ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

У цій роботі розглянуті бібліотеки для вирішення задач машинного навчання. Виконанно порівняльний аналіз даних бібліотек. Для порівняння було обрано задачу класифікації текстових даних. Навчання моделі відбувалось методами навчання з вчителем. Для навчання були використані штучні нейронні мережі з нейронами, що мають довгу короткочасну пам'ять. Оцінка бібліотек відбувається за точністю класифікації, швидкістю навчання моделі в однакових середовищах та наявністю засобів, що полегшують побудову та апробацію цієї моделі.

Ключові слова: *штучні нейронні мережі, глибоке навчання, інтелектуальний аналіз текстових даних, класифікація текстів, навчання з вчителем.*

Постановка проблеми. Задача класифікації текстових даних є однією з багатьох актуальних задач науки про машинне навчання. За останні роки було розроблено велику кількість методів для вирішення цієї задачі. При цьому залишається відкритим питання, які саме підходи мають перевагу і за яких обставин, а також – які програмні реалізації (бібліотеки) для машинного навчання є кращими для вирішення таких задач.

Аналіз останніх досліджень і публікацій. У праці "Sentiment Analysis using LSTM" [1], пропонується скористатись моделлю, що має архітектуру рекурентної нейронної мережі, побудованої на блоках довгої короткочасної пам'яті. При цьому автор пропонує як вхідні дані використовувати векторні представлення слів.

Іншим підходом, який розглядається в статті Д. Браунлі [2], є використання згорткових нейронних мереж і обчислення «мішка слів». Він має недоліки, адже не здатен врахувати порядок слів.

Одним з інструментів для вирішення задачі класифікації текстів є векторне представлення слів (word embedding). На практиці часто вживаними є такі моделі векторного представлення слів, як word2vec і GloVe. Обидві моделі вивчають представлення слів у вигляді векторів, базуючись на тому, як часто і в яких комбінаціях слова з'являються разом у великих текстових корпусах. Моделі відрізняються тим, що word2vec є прогнозуючою моделлю, тоді як GloVe − це частотна модель.

У прогнозуючих моделях вектори слів обробляються таким чином, щоб зменшити похибку між цільовими та контестними словами. У word2vec це реалізовано як нейронна мережа прямого поширення.

Прогнозуючі моделі економно використовують пам'ять, але необхідність зменшення похибки для кожної одиниці даних робить паралелізацією досить складною, що робить обробку великих текстових корпусів довгою [3].

Частотні моделі працюють із матрицями одночасної появи слів. Над матрицями здійснюється операція зменшення вимірності, залишаються лише найголовніші ознаки, шум відсікається. Після цієї операції кожен рядок отриманої матриці позначає вектор для кожного слова.

Частотні моделі вимагають великих обчислювальних потужностей та велику кількість пам'яті, але при цьому такі моделі легко розпаралелюються, що в теорії дає змогу проводити навчання на текстах розмірами в сотні гігабайтів, що має збільшити точність моделі [3].

Моделлю навчання з учителем, що добре підходить для класифікації текстових даних, є рекурентна нейронна мережа, побудована на нейронах із довгою короткочасною пам'яттю. Рекурентні нейронні мережі дають змогу обробляти дані, в яких є важливим порядок одиниць даних, подібно до того, як є важливим порядок слів у людській мові [4].

Довга короткочасна пам'ять − це архітектура РНН, ключова перевага якої для вирішення задачі семантичного аналізу речень − здатність реагувати на ключові для визначення класу слова незалежно від того, як далеко ці слова знаходяться один від одного [5; 6].

Блок ДКЧП складається з вхідного, вихідного та забувального вентилів, що контролюються відповідними векторами і визначають внутрішній стан блоку. На рисунку 1 проілюстровано будову вентилю.

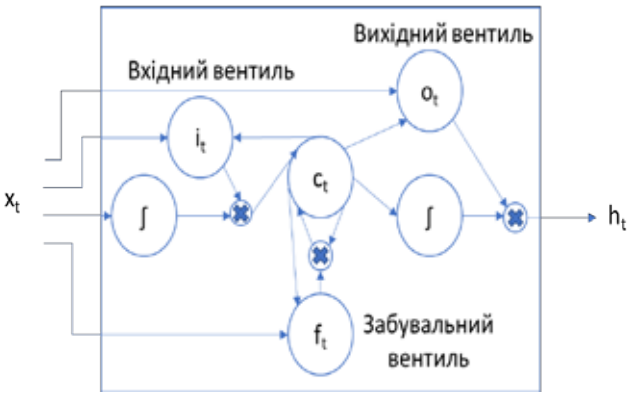

Рис. 1. Схема вузла ДКЧП

Вузол ДКЧП визначається такими формулами [7]:

$$f_t=\sigma_g\,(W_f x_t+U_f h_{t-1}+b_f),$$
$$i_t=\sigma_g\,(W_i x_t+U_i h_{t-1}+b_i),$$
$$o_t=\sigma_g\,(W_o x_t+U_o h_{t-1}+b_o),$$
$$c_t=f_t\circ c_{t-1}+i_t\circ\sigma_h\,(W_c x_t+Uc\, h_{t-1}+b_c),$$
$$h_t=\sigma_h\circ\sigma_h\,(c_t),$$

де f_t − вектор вентиля забуття, i_t − вектор вентиля входу, o_t − вектор вентиля виходу, c_t − вектор стану блоку, h_t − вектор виходу, x_t − вектор входу, σ_g − сигмоїдна функція, σ_c − гіперболічний тангенс, σ_h − функція, що повертає аргумент.

Постановка завдання. Основною метою є порівняння наявних програмних бібліотек для інтелектуальної обробки текстових даних, а саме – для вирішення задачі класифікації текстів методом рекурентних нейронних мереж. Для порівняння обрані TensorFlow, PyTorch і Keras. У цій роботі оцінка бібліотек відбувається за точністю класифікації, швидкістю навчання моделі в однакових середовищах та наявністю засобів, що полегшують побудову та апробацію цієї моделі.

Виклад основного матеріалу дослідження. Як набір даних використовується набір IMDB, що містить 50 000 рецензій на фільми і мітки класів «позитивний» та «негативний». Задача полягає в тому, щоб засобами трьох різних бібліотек розробити модель, що буде здійснювати семантичний аналіз висловлення і здатна визначати, чи є рецензія позитивною чи негативною. В наборі даних рецензії поділені на навчальні та тестові, в кожній категорії по 25 000 рецензій. У рамках цієї статті навчання моделей відбувається виключно на навчальних даних. Точність рахується як відсоток тестових рецензій, мітку класу яких модель змогла вказати вірно.

Конфігурація комп'ютера, на якому були проведені дослідження:

- CPU: AMD Ryzen 3 2300U
- GPU: Radeo Vega Mobile Gfx
- RAM: 8 GB DDR4 2666 MHz
- ROM: SK Hynix – 256GB M.2 SSD HFS256GD9TNG-62A0A

Схема первинної обробки. Окрім розробки моделі, необхідно провести первинну обробку вхідних даних, а саме почистити одиниці даних від шумів (теги HTML, розділові знаки, лишні пробіли, всі літери приводяться в нижній регістр), здійснити токенізацію слів, привести всі рецензії до однієї довжини. При цьому первинна обробка здійснюється, перш за все, засобами бібліотек, або самописними алгоритмами, якщо необхідних засобів немає.

Очистка даних від шумів здійснюється за допомогою методу рядків replace, що дає змогу замінювати необхідні підрядки на строки нульової довжини, методу lower, що перетворює всі символи на символи нижнього регістру, а також за допомогою засобів бібліотеки re, що дає змогу працювати з регулярними виразами.

Токенізація здійснюється за допомогою заміни кожного слова в повідомленні на список цілих чисел, де кожне число − порядковий номер слова в файлі з натренованою моделлю векторного перетворення слів (GloVe чи Word2Vec).

Приведення всіх повідомлень до однієї довжини є важливою, хоч і, на перший погляд, простою задачею, адже від довжини повідомлення залежить розмір вхідного шару нейронної мережі. Проблема полягає в тому, що, з одного боку, необхідно зменшити кількість вхідних нейронів, адже збільшення цього числа значно ускладнює розрахунки вагів при навчанні моделі. З іншого боку, якщо відкидувати частину слів із довгих повідомлень, може зникнути і важлива інформація, яка, можливо, і визначала суть повідомлення. Щоб досягти компромісу, розглянемо розподіл довжин повідомлень.

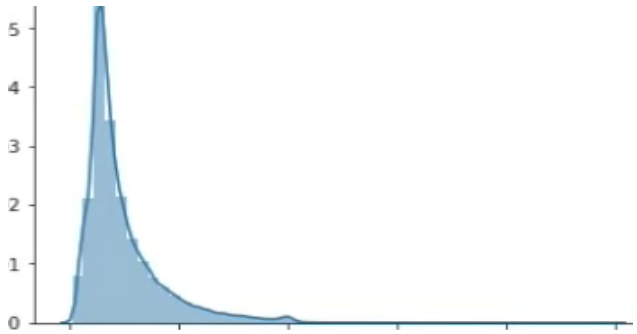

Рис. 2. Розподіл довжин повідомлень

Графік розподілення довжин повідомлень показує, що переважна кількість повідомлень є коротшою за 500 символів. Тому оберемо це число як оптимальне значення довжини одного повідомлення. Після фільтрації даних і приведення одиниць даних до однієї довжини можна за допомогою токенів замінити слова на вектори GloVe. Тепер дані готові до використання в навчанні моделі.

Слід зауважити, що бібліотека Keras має вбудовані засоби такої первісної обробки, для двох інших бібліотек необхідно розробляти необхідні програми власноруч.

Як модель машинного навчання було обрано рекурентну нейронну мережу з нейронами з довгою короткочасною пам'яттю. Кількість вхідних нейронів дорівнює розміру вхідних даних − 500, кількість вихідних дорівнює кількості класів − 2. Слід зауважити, що всі навчання моделей виконувались виключно на CPU.

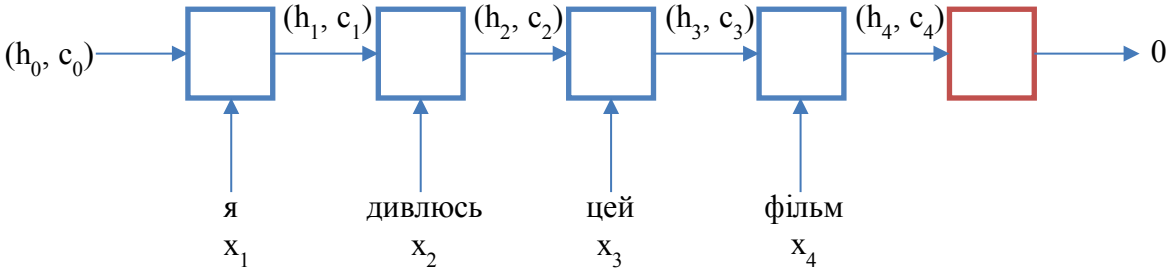

Рис. 3. Схема рекурентної нейронної мережі

TensorFlow. TensorFlow – це безплатна програмна бібліотека символьного програмування з відкритим сирцевим кодом для роботи з нейронними мережами і потоками даних. Моделі в цій бібліотеці представлені у вигляді графів потоків даних. Граф містить набір вузлів, що звуться операціями. Операції − це абстракції, що відображають певні обчислення, які можуть бути як простими, як-от додавання і віднімання, так і складними, як функції багатьох змінних. По цих графах потоків переходять від операції до операції тензори − багатовимірні матриці. Тому бібліотека і називається tensorflow −- потік тензорів. На вхід моделі завжди подаються тензори, повертає модель також тензори.

Важливою особливістю цієї бібліотеки є поняття компіляції графу і сесії обчислень, так бібліотека реалізована на базі архітектури «визначи і запусти», що означає, що спочатку компілюється власне потік даних, а потім в ньому запускаються обчислення. Таке рішення значно збільшує швидкість обчислень. Але таке рішення має численні недоліки, а саме неможливість використання засобів потоку (циклів, умовних операторів), і значно ускладнює відлагодження програми (неможливість користування функцією print, вбудованим відлагоджувачем і логуванням).

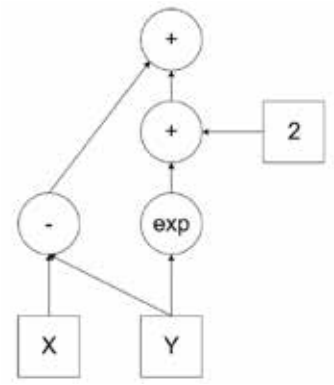

Рис. 4. Приклад елементарного графу потоку даних

Бібліотека TensorFlow є низькорівневою, жодних вбудованих інструментів для роботи з наборами даних у ній не передбачено, тому дані необхідно скачувати й обробляти вручну за допомогою бібліотеки requests та регулярних виразів.

Keras. Keras – це бібліотека для роботи з нейронними мережами з відкритим сирцевим кодом, написана мовою Python.

Ця бібліотека здатна працювати з бек-ендом у вигляді TensorFlow, Microsoft Cognitive Toolkit, Theano, чи PlaidML (ву дослідженні використовувався бек-енд TensorFlow). Ця бібліотека розроблена для того, щоб бути дружньою до користувача, модульною та легко розширюваною і дозволяти робити швидкі експерименти з нейронними мережами глибокого навчання. Keras представляє високорівневий, більш інтуїтивний набір абстракцій, який робить простим формування нейронних мереж незалежно від тилової бібліотеки наукових обчислень. Фактично, її можна цілком розглядати як синтаксичний цукор для низькорівневих бібліотек, який дає змогу замінити десятки рядків коду на TensorFlow кількома викликами класів, що дуже зручно, коли треба дуже швидко спробувати багато різних варіантів архітектури нейронних мереж.

Keras дає змогу використовувати різноманітні шари типових мереж (рекурентних, згорткових, рекурсивних, тощо) у вигляді класів, що імпортуються і додаються, як цеглини, в клас моделі, створюючи новий шар нейронної мережі. При цьому дуже легко можна задати розмір кожного шару, типи зв'язків, функцію активації.

Keras має вбудовану підтримку популярних наборів даних, у тому числі і набору IMDB, а також змогу провести токенізацію, векторне перетворення слів і приведення повідомлень до однієї довжини кількома командами. До його недоліків можна зарахувати те, що він не успадковує 100% функціоналу бібліотек, що стоять за ним. Мається на увазі, що іноді виникає необхідність вставити код, написаний на TensorFlow, всередину програми, написаної на Keras.

PyTorch. PyTorch – ще одна бібліотека для машинного навчання з відкритим сирцевим кодом. Її принципи засновані на бібліотеці Torch для мови програмування Lua. PyTorch є суттєво схожим на TensorFlow, але при цьому є дві ключові відмінності.

Перша відмінність – підтримка імперативного програмування, тобто відходження від обмежень символьного програмування. Це дає змогу робити програми більш гнучкими і зручними у розробці, адже можна використовувати всі можливості Python, такі як цикли, умовні оператори, засоби для відлагодження і контролю.

Друга ключова особливість частково випливає з першої і являє собою можливість використання динамічних графів потоків даних. Іншими словами, PyTorch використовує архітектуру «визначений запуском». Отже, граф починає формуватись у момент запуску. Це дає змогу зручно працювати з такими моделями, граф потоку даних яких може змінюватися. Наприклад, це можуть бути рекурентні нейронні мережі, рекурентний шар яких може збільшуватись чи зменшуватись, залежно від розмірності чинної одиниці даних.

Висновки. У результаті були отримані точності різних моделей, час їх навчання, а також факт наявності вбудованих засобів для роботи з наборами даних.

Таблиця 1

Порівняння бібліотек для машинного навчання

	Наявність засобів для роботи з наборами даних	**Час навчання моделі, с**	**Отримана точність, %**
TensorFlow	Ні	4719	79.2
Keras	Так	101	86.48
PyTorch	Так	418	86.02

Результати досліджень показують, що найбільшу точність показали моделі, реалізовані за допомогою Keras і PyTorch. При цьому модель на TensorFlow програє їм в точності і надзвичайно програє в часі навчання, що пов'язано з тим, що в усіх тестових запусках навчання, саме ця модель зіштовхувалась із проблемою перенавчання. Згідно з цим дослідженням, Keras показав найкращі результати, це може пояснюватись тим, що багато необхідних інструментів у цій бібліотеці створено спільнотою розробників, що гарантує високу якість реалізації моделей і меншу швидкість навчання.

У подальшому планується проводити дослідження з моделями, що містять різну кількість нейронів у рекурентному шарі, використати інші алгоритми для оптимізації моделі та векторного представлення слів. Також планується дослідити напрацювання за напрямом інтелектуального аналізу текстових даних великого об'єму.

Список літератури:

1. Samarth Agrawal. Sentiment Analysis using LSTM. Samarth Agrawal. 2019. URL: https://towardsdatascience.com/sentiment-analysis-using-lstm-step-by-step-50d074f09948. (Last accessed: 17.04.2019).

2. Jason Brownlee. Predict Sentiment From Movie Reviews Using Deep Learning. 2016. URL: https://machinelearningmastery.com/predict-sentiment-movie-reviews-using-deep-learning/. (Last accessed: 15.04.2019).

3. Christopher D. Manning. GloVe: Global Vectors for Word Representation. 2014. URL: https://www.aclweb.org/anthology/D14-1162. (Last accessed: 15.04.2019).

4. Grus J. Data Science from Scratch . Sebastopol, CA: O'Reilly Media, Inc., 2015. – 336 p.

5. Mandic D., Chambers C. Recurrent Neural Networks for Prediction: Learning Algorithms,Architectures and Stability. New York, NY, USA: John Wiley & Sons, Inc., 2001. 308 p.

6. Duyu T., Bing Q., Ting L. Document Modeling with Gated Recurrent Neural Networkfor Sentiment Classification. 2015. URL: https://www.aclweb.org/anthology/D15-1167. (Last accessed: 15.04.2019).

7. Sepp Hochreiter, Jürgen Schmidhuber. Long Short-term Memory. 1997. URL: https://www.researchgate.net/publication/13853244_Long_Short-term_Memory/download. (Last accessed: 15.04.2019)

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПРОГРАММНЫХ БИБЛИОТЕК ДЛЯ КЛАССИФИКАЦИИ ТЕКСТОВЫХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ

В работе рассмотрены библиотеки для решения задач машинного обучения. Выполнен сравнительный анализ данных библиотек. Для сравнения были выбраны задачу классификации текстовых данных. Обучение модели происходило методами обучения с учителем. Для обучения были использованы искусственные нейронные сети с нейронами, имеющими долгую кратковременную память. Оценка библиотек происходит по точности классификации, скорости обучения модели в одинаковых средах и наличию средств, облегчающих построение и апробацию данной модели.

Ключевые слова: *искусственные нейронные сети, глубокое обучение, интеллектуальный анализ текстовых данных, классификация текстов, обучение с учителем.*

COMPARATIVE ANALYSIS OF SOFTWARE LIBRARIES FOR THE CLASSIFICATION OF TEXT DATA USING ARTIFICIAL NEURAL NETWORKS

In this paper libraries for solving machine learning problems are being considered. Comparative analysis of library data is being performed. For comparison, the task of text data classification was selected. The model was taught using supervised learning methods. For training, artificial neural networks with neurons with long short-term memory were used. The evaluation of libraries is based on the accuracy of the classification, the speed of learning the model in the same environment and the availability of tools that facilitate the construction and testing of machine learning model.

Key words: *artificial neural networks, deep learning, texts mining, texts classification, supervised learning.*