УДК

1 **Р.А.МИКОЛАЙЧУК**

Доктор технічних наук, доцент, доцент кафедри мережевих та інтернет технологій

1 **М.О.Агафонова**

Студентка 3 курсу кафедри мережевих та інтернет технологій факультету інформаційних технологій

1 **л.О.Лунін**

Студент 3 курсу кафедри мережевих та інтернет технологій факультету інформаційних технологій

1 *Київський національний університет імені Тараса Шевченка, м. Київ*

**Аналіз ефективності застосування нейронних мереж для прогнозування параметрів пандемії COVID-19 в Україні**

Пандемія коронавірусу, останнім часом стала проблемою, яка є головним епіцентром обговорення, ця проблема стала ребром у цілому світі. Коронавіруси – це тип вірусів, які можуть зустрічатися, як у людини, так і у представників тваринного світу. А головне питання зараз це все через те, що від поширення цієї хвороби та збільшення чи навпаки зменшення кількості хворих залежить економіка всіх держав світу, зокрема, і України. Пандемія також впливає і на суспільство. Для того, щоб мінімізувати негативний вплив пандемії і покращити методи з її усунення, необхідно здійснювати прогнози кількості хворих на COVID-19 та тих, що одужали чи померли. А це все для того, щоб можна було спрогнозувати якісь моменти та, можливо, застерегти людей від ще страшніших наслідків. Для цього застосовують різноманітні методи. Важливе місце серед них посідають штучні нейронні мережі (ШНМ). Перші спроби використання ШНМ для прогнозування кількості хворих на COVID-19 в Україні були здійснені центром даних з геоінформатики та сталого розвитку, проте використана нейромережа Back Propagation хоч і є найпоширенішою, але має низку недоліків. Найбільшим її недоліком є те, що в процесі навчання нейромережа повільно сходиться до мінімальної помилки, а це негативно впливає на час навчання. Таку ШНМ важко використовувати в режимі реального часу з наперед заданою високою точністю навчання.

Ми спробували зробити графік розвитку цієї хвороби на території України з самого початку - з березня 2020 року. Робили це за допомогою проекту TensorFlow. Він має багато унікальних деталей у своїй структурі, які нам допомогли зібрати дані та провести модель прогнозування з невеликими втратами. А саме такі деталі: основна бібліотека підходить для широкого сімейства технік машинного навчання, а не тільки для глибинного навчання; лінійна алгебра та інші нутрощі добре видно зовні; на додаток до основної функціональності машинного навчання, TensorFlow також включає власну систему логування, власний інтерактивний візуалізатор логів і навіть потужну архітектуру з доставки даних; модель виконання TensorFlow відрізняється від scikit-learn мови Python і від більшості інструментів в R.

TensorFlow є системою машинного навчання Google Brain другого покоління, випущеною як відкрите програмне забезпечення 9 листопада 2015 року. В той час як еталонна реалізація працює на одиничних пристроях, TensorFlow може працювати на декількох центральних та графічних процесорах (включно з додатковими розширеннями CUDA для обчислень загального призначення на графічних процесорах). TensorFlow доступна для 64-розрядних Linux, macOS, Windows, та для мобільних обчислювальних платформ, включно з Android та iOS. Обчислення TensorFlow виражаються як станові графи потоків даних. Назва TensorFlow походить від операцій, що такі нейронні мережі виконують над багатовимірними масивами даних. Ці багатовимірні масиви називають «тензорами».

Також, TensorFlow надає бібліотеку готових алгоритмів чисельних обчислень, реалізованих через графи потоків даних (data flow graphs). Вузли в таких графах реалізують математичні операції або точки входу/виводу, в той час як ребра графу представляють багатовимірні масиви даних (тензори), які перетікають між вузлами. Вузли можуть бути закріплені за обчислювальними пристроями і виконуватися асинхронно, паралельно обробляючи разом все підходящі до них тензори, що дозволяє організувати одночасну роботу вузлів в нейронної мережі за аналогією з одночасною активацією нейронів в мозку

**Тензорний процесор (ТП)**

У травні 2016 року Google анонсувала свій тензорний процесор, спеціалізовану мікросхему, побудовану спеціально для машинного навчання, й підігнану під TensorFlow. ТП розроблений для забезпечення високої продуктивності в арифметиці низької точності (наприклад, 8-бітній), і спрямованим радше на використання або виконання моделей, аніж на їхнє тренування. Google оголосила, що вони використовували ТП у своїх центрах обробки даних понад рік, і виявили, що вони забезпечують для машинного навчання на порядок краще оптимізовану продуктивність на ват.



**В чому полягає проблема часових рядів?**

Проблеми часових рядів стосуються даних з плином часу. Наприклад, кількість співробітників компанії протягом 10 років, продажі комп’ютерів за останні 5 років, споживання електроенергії за останні 50 років.

Хронологія може бути короткою (секунди/хвилини) або довгою (роки/десятиліття). І проблеми, які ви могли б дослідити можна розділити на дві категорії: класифікація і передбачення

|  |  |  |
| --- | --- | --- |
| **Тип проблеми** | **Приклад** | **Результат** |
| Класифікація | Виявлення аномалій, ідентифікація часових рядів | Дискретний (мітка) |
| Передбачення | Передбачення цін на фондовому ринку, прогнозування майбутнього попиту на продукт, вимоги до запасів | Безперервний (число) |

В обох вищеописаних випадках часто використовується наглядовий підхід. Це означає, що у вас є деякі приклади даних і мітка, пов'язана з цими даними.

Наприклад, при прогнозуванні кількості випадків захворювання ваші дані можуть бути історичною кількістю за останній місяць, а мітка – кількістю захворювань сьогодні (мітка не може бути кількістю завтра, тому що це те, що ми хотіли б передбачити).

Отже, для передбачення кількості захворювань ми будемо використовувати Tensorflow, Numpy, MatPlotLib, Pandas бібліотеки.

**Актуальність** теми є надзвичайно високою, оскільки правильне прогнозування допоможе ефективно розподілити ресурси та мінімізувати ризики для здоров’я населення.

**Дані** були успішно експортовані з джерела сайту [www.worldometers.info](http://www.worldometers.info) та збережені в CSV-файл.

**Тренування** моделі відбувалось на основі наступного розподілу даних: 80% - тренувальні, 20% - тестові. Нами було використано 2 моделі: Naïve forecast та Recurrent neural networks (RNN/LSTM).

Оскільки перша з них не потребує тренування, то перейдемо до розгляду відразу другої моделі. В цьому ми використали команду .fit з настипними параметрами:

train\_windows – умовні Х для тренування,

train\_labels – умовні У для тренування,

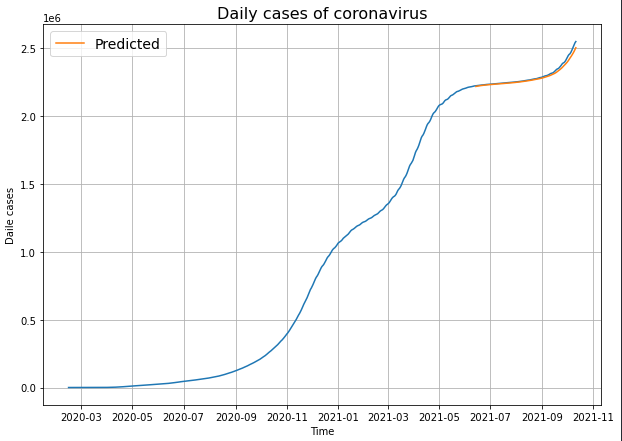
epochs=100 – кількість етапів тренування,

verbose=0 – логування відключене,

batch\_size=128 – кількість данних за етап,

validation\_data=(test\_windows, test\_labels) – вибір данних для перевірки тренування,

callbacks=[create\_model\_checkpoint(model\_name=model\_5.name)] – викликаємо функцію після тренування

В результаті ми можемо спостерігати що передбачення виявлень захворювань в день майже збігається з реальними показниками, а це може означати лише те, що модель успішно передбачила кількість випадків. 

Отже, ми змогли успішно застосувати TensorFlow для прогнозування даних.

**Список використаних джерел**

1. [Ukraine COVID](https://www.worldometers.info/coronavirus/country/ukraine/)
2. [10. Milestone Project 3: Time series forecasting in TensorFlow](https://github.com/mrdbourke/tensorflow-deep-learning/blob/main/10_time_series_forecasting_in_tensorflow.ipynb)
3. [Tensorflow](https://www.tensorflow.org/)