МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій

«До захисту допущено»
В.о. завідувача кафедри

_____ Світлана ВАЩЕНКО

____ _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 «Комп'ютерні науки»,

освітньо-професійної програми «Інформаційні технології проектування»

на тему: «Прогнозування електрозабезпечення приватного будинку від відновлюваних джерел енергії за допомогою нейронних мереж»

Здобувача групи ІТ-01 Богачова Максима Володимировича
(шифр групи) (прізвище, ім'я, по батькові)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Максим БОГАЧОВ
(підпис) (Ім'я та ПРІЗВИЩЕ здобувача)

Керівник к.т.н. доц. Парфененко Юлія Вікторівна _____
(посада, науковий ступінь, вчене звання, Ім'я та ПРІЗВИЩЕ) (підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ

В. о. зав. кафедри ІТ

Світлана ВАЩЕНКО

«_____» _____ 2024 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Богачову Максиму Володимировичу

1 Тема роботи *Прогнозування електрозабезпечення приватного будинку від відновлюваних джерел енергії за допомогою нейронних мереж*

керівник роботи Парфененко Юлія Вікторівна, *к.т.н., доцент*,

затверджені наказом по університету від « *07* » травня 2024 р. №0482-VI

2 Строк подання студентом роботи « 26 » травня 2024 р.

3 Вхідні дані до роботи технічне завдання на розробку додатку з прогнозування електрозабезпечення приватного будинку від альтернативних джерел енергії

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, проєктування додатку, розробка додатку

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення планування робіт	20.03.2024	виконано
2	Оформлення технічного завдання	06.04.2024	виконано
3	Проведення аналізу предметної області	16.04.2024	виконано
4	Проведення проєктування додатку	25.04.2024	виконано
5	Розробка додатку	19.05.2024	виконано
6	Тестування додатку	23.05.2024	виконано
7	Оформлення пояснювальної записки	26.05.2024	виконано

Студент _____ Максим БОГАЧОВ
(підпис)

Керівник роботи _____ к.т.н., доц. Юлія ПАРФЕНЕНКО
(підпис)

ABSTRACT

Bachelor's qualification thesis topic «Prediction of electricity supply of a detached house from renewable energy sources using neural networks».

Bachelor thesis consists of introduc

The relevance of this work is in its potential to:

- Increase reliance on renewable energy: using accurate prediction of electricity supply from renewable sources, especially solar, for a detached house. So, this research can contribute to a more sustainable future with less use of fossil fuels.
- Improve homes' energy usage efficiency: predicting both energy consumption and energy generation could be beneficial in improving energy efficiency in private houses.
- Ease the energy grid management: the ability to predict energy grid behavior in such parameters as energy consumption and energy generation from renewable sources might be useful in facilitation of energy grid management, especially, empowering such processes as resource planning and payload balancing.
- Lower the costs of energy bills: the ability to predict possible energy usage could assist in more balanced load distribution over a certain period of time, especially in regions or countries where the electricity tariff is not fixed, but proportional (the more you use – the more you are charged proportionally).

Goal statement: Develop a desktop application that can accurately (with accuracy higher than 90%) predict the electricity supply (consumption and generation) of a detached house from renewable sources.

Results of this work were presented at the IMA-2024 conference. The application was developed that predicted energy consumption by detached houses in time periods of 24 hours. So, the user having 24 records of energy consumption for each hour of the

previous day, can predict the next 24 hours of energy consumption. Accuracy of such model was 93.6% using MAPE (mean absolute percentage error) to evaluate performance of model. However, this application had its disadvantages:

- Focus on hourly data: while using hourly data could increase the accuracy of short-term prediction, it comes at the cost of increased computational resources and diminishing returns for longer-term prediction. Also, recording hour-to-hour consumption data leads to the increasing complexity of data gathering, especially when it comes to long periods of time.
- Predicting only energy consumption: many modern households integrate renewable energy sources, mostly solar panels, into their power grid, which increases its complexity and this requires more efforts for planning and resource management. So, the prediction of energy generation was considered to be developed in further research.
- Relying only on historical data of consumption: during this research the model were developed relied only on historical data of consumption. While it could be beneficial in the matter of lower computational resources, the usage of historical weather data could increase the accuracy of the model, making the artificial neural network model more general and, therefore, efficient on different data sets.

In the first section the recent researches in this field are covered, especially focused on existing systems that serve the similar purpose as this work. Figured out, that crucial part of such application would be not only predicting the energy consumption, but also the energy generation by renewable energy sources. Thesis' goal, tasks and application's requirements were defined and the artificial intelligence model. Also, the architecture of the neural network for this work was defined through analysis of stated problems.

Second section dedicated to the functional modeling of desktop application. In this section contextual diagram, IDEF0 model and use-case diagram were created and described.

Third section dedicated to the application implementation and testing. Firstly, the Maven project was created and organized. Several dependencies was considered during this stage:

- Apache POI: library by Apache community for working with XLSX (Excel) files (reading, editing, writing), because XLSX format was chosen for data storage and organization.
- DL4j (DeepLearning4j): java framework for deep learning models' creation, training, testing and using.
- ND4j: library by which is backed DL4j. It serves as a tool for working with n-dimensional arrays, which is very helpful in the context of artificial intelligence.

Also, in this section several artificial neural network models were trained and evaluated to find the most efficient one. Models were separated into two groups: those that work with historical data of temperature, and those that work only with historical data of electricity supply. Also, the models were trained with different numbers of days that they're predicting. During training, the most efficient period to predict was 1 day and 14 days, which completely aligns with the task to introduce the medium-term prediction. While for energy consumption models temperature data slightly increased performance in short-term periods, this wasn't so helpful for energy generation (solar panels in this case) prediction, and the model without such data about temperature have slightly better results overall.

Result of this work is a developed desktop application that provides electricity supply prediction of a detached house from renewable energy sources over a medium-term time period. Its functionality simplifies resource planning and load balancing for households through understanding future values of energy consumption and generation.

The practical value is to create desktop application, that simplify resource planning for households. It increases household energy efficiency, especially when it comes to integration and use of renewable energy sources.

Keywords: artificial intelligence, neural networks, long short-term memory, deep learning, machine learning, electricity supply prediction, recurrent neural network, renewable energy sources, solar energy.

ЗМІСТ

ВСТУП

На сьогодні технології, що стрімко розвиваються, мають тенденцію до збільшення споживання електроенергії населенням та промисловістю. Це, у свою чергу, призводить до ускладнення енергетичних систем, що ускладнює планування ресурсів, особливо за наявності в енергосистемі відновлюваних джерел енергії.

Тож, актуальність даного дослідження полягає у прогнозуванні споживання та генерації електроенергії в обставинах невпинного росту складності електричних мереж, що спростить короткострокове та довготривале планування ресурсів і роботи домогосподарств, які використовують альтернативні джерела енергії для власних потреб або для отримання доходів з виробленої електроенергії.

Об’єкт – прогнозування енергозабезпечення від відновлюваних джерел енергії. Предмет –методи та програмні засоби прогнозування енергозабезпечення від відновлюваних джерел енергії.

Отже, метою даного дослідження є розробка програмного додатку для прогнозування електрозабезпечення приватного будинку від відновлюваних джерел енергії за допомогою нейронних мереж.

Для досягнення мети проєкту необхідно виконати наступні задачі:

- визначити актуальність роботи, дослідити предметну область та провести аналіз аналогів додатків;
- розробити нейронну мережу та провести її тестування;
- спроєктувати структуру додатку та розробити його;
- інтегрувати в додаток нейронну мережу;
- виконати тестування програмного додатку.

Практичне значення додатку буде полягати у спрощенні процесу планування використання електроенергії, зокрема і від альтернативних джерел, що в свою чергу призведе до оптимізації витрат домогосподарства на електроенергію та обслуговування енергоінфраструктури.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Штучний інтелект (ШІ) – це галузь комп'ютерних наук, яка вивчає створення програм та систем, що можуть виконувати завдання, які вимагають інтелектуальних здібностей людини. Системи ШІ можуть включати в себе машинне навчання, обробку природної мови, комп'ютерний зір, робототехніку та інші технології [1].

З розвитком технологій зростає попит на автоматизацію та оптимізацію процесів у різних галузях, включаючи виробництво, логістику, фінанси, охорону здоров'я та інші [2]. Багато бізнесів використовують ШІ для різних цілей. Така ситуація і у сфері енергетики, коли при збільшенні технологічності домоволодіння збільшується і комплексність електричної мережі, як локальної (на рівні будинку), так і рівнями вище (на рівні міст та регіонів) [3].

Алгоритми машинного навчання використовують для аналізу даних та прогнозування тенденцій, що допомагає в прийнятті ефективних стратегічних рішень. Він є корисним і в оптимізації рутинних бізнес-процесів, що призводить до збільшення ефективності та зниження витрат; його використовують у чат-ботах, персоналізованих системах рекомендацій та інших інструментів для поліпшення взаємодії з клієнтами [4-5]

Добре розроблена система ШІ володіє високою точністю та швидкістю вирішення задач. Успішні системи ШІ можуть адаптуватися до змінних умов та викликів. Якість даних є визначальною у навчанні систем ШІ, адже недостатня кількість даних або їх низька якість може призвести до поганих результатів та постійних помилок. Важливо зазначити, що добре зроблені системи ШІ можуть пояснити свої рішення та прогнози, що допомагає користувачам розуміти процеси та приймати обґрунтовані рішення [6].

Для роботи з хронологічно послідовними даними добре підходить архітектура нейромережі LSTM - Long Short-Term Memory. Через що її часто застосовують і зокрема для прогнозування споживання електроенергії [7-9].

Щоправда, LSTM нейромережа може бути недостатньо точною або недостатньо узагальненою. Для цього слід використовувати кілька методик:
- відсіювання - ігнорування частки даних на вхід до схованих шарів нейромережі задля того, щоб не копіювати паттерн даних, на яких вона тренується, а узагальнювати його;
- двосторонні LSTM мережі - для опрацьовування даних як у звичайному, так і у зворотному порядках, щоб мати можливість вловлювати більше контекстної інформації, що може збільшити точність;
- оптимізація гіперпараметрів - пошук збалансованої комбінації кількості шарів, їхнього розміру, епох, швидкості навчання, тощо [10].

Також слід зазначити, що LSTM нейромережі також мають ряд недоліків:
- обчислювальна складність - LSTM є складнішими в роботі ніж інші види нейромереж, що впливає на швидкодію при їх навчанні і роботі;
- велика кількість тренувальних даних - для точних результатів роботи нейромережа потребує великі обсяги тренувальних даних, інакше ефективність може бути незначною;
- недостатнє узагальнення - як і інші RNN нейромережі, LSTM вразлива до "вибухаючого" та "стираючого градієнту" [11], тож потребує оптимізації шляхом регуляризації та відсіювання [12-14].

Дослідження [15] показало, що LSTM моделі нейромереж ефективні у короткостроковому прогнозуванні, а саме інтервалами у 6 годин, 1 та 3 доби. У той час, як авторегресивні моделі не мали такого успіху. Найкращий результат був показаний при роботі з даними приватного будинку - середнє абсолютне відхилення у відсотках (MAPE) було рівне 0.1586% при прогнозуванні відрізками у 6 годин, 0.1631% при відрізках в 1 добу (24 години) та 0.1243% при відрізках в 3 доби.

Показово, що задля такої високої точності було використано дані споживання електроенергії за 6-10 сезонів. Для даної моделі було використано такі параметри - LSTM-шар - 100 нейронів, шар відсіювання та щільний шар (кожен нейрон даного шару підключений до кожного попереднього)[16] розміром в 1 нейрон. Таким чином, нейромережа складається з 52 501 параметрів, які слід натренувати. Як і було зазначено раніше, LSTM моделі дуже вибагливі до кількості даних для ефективності роботи, але також великі обсяги даних у моделях LSTM швидкість такого тренування зменшується, особливо зі збільшенням комплексності моделі (кількість параметрів). Тому, у цій роботі буде спроба реалізувати просту модель нейромережі LSTM - зменшивши кількість параметрів за рахунок меншої точності, щоб отримати швидкий алгоритм навчання та роботи нейромережі, щоб даний додаток був доступний до використання як на мобільних пристроях, так і на малопотужних персональних комп'ютерах.

Також, для створення додатку планується використати мову програмування Java з фреймворком DL4j для досягнення мультиплатформерності, оскільки даний фреймворк доступний для роботи з такими операційними системами - Windows, Linux, MacOS та Android.[17]

1.2 Аналіз програмних продуктів-аналогів

На жаль, точного аналогу по функціоналу не було знайдено. Однак деякі технологічні та енергетичні компанії, а також стартапи, використовують системи штучного інтелекту для аналізу даних з електроенергії. Ця галузь є високорозвиненою, однак майже не використовується малими бізнесами.

Для визначення вимог майбутнього програмного продукту було проведено дослідження існуючих сервісів які використовують системи ШІ для аналітики роботи електромереж, а саме «Schneider Electric», «Google DeepMind» та «Enel».

Schneider Electric надає рішення для управління електроенергією, включаючи системи штучного інтелекту для аналізу та оптимізації енергоспоживання в будівлях та промислових об'єктах. Їх підхід намагається звести до мінімуму ручне збирання даних, дозволяючи ШІ самому з'ясовувати як будівля з'єднана всередині. Проаналізувавши інформацію, ШІ провайдер пропонує список точок, на які можна впливати, кожна з яких має допустимий діапазон, в межах якого може діяти рішення, що приймає ШІ [18].

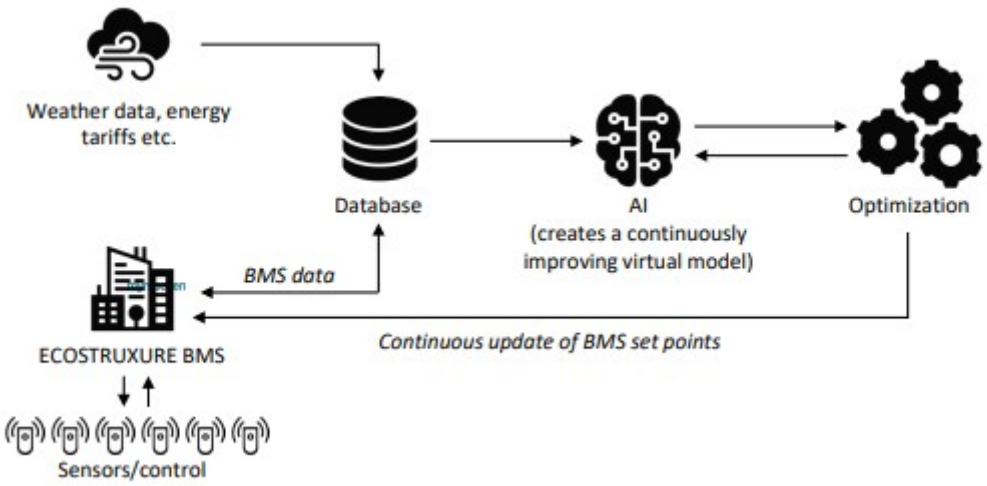


Рисунок 1.1 Ілюстрація потоків даних між рішенням ШІ та BMS.

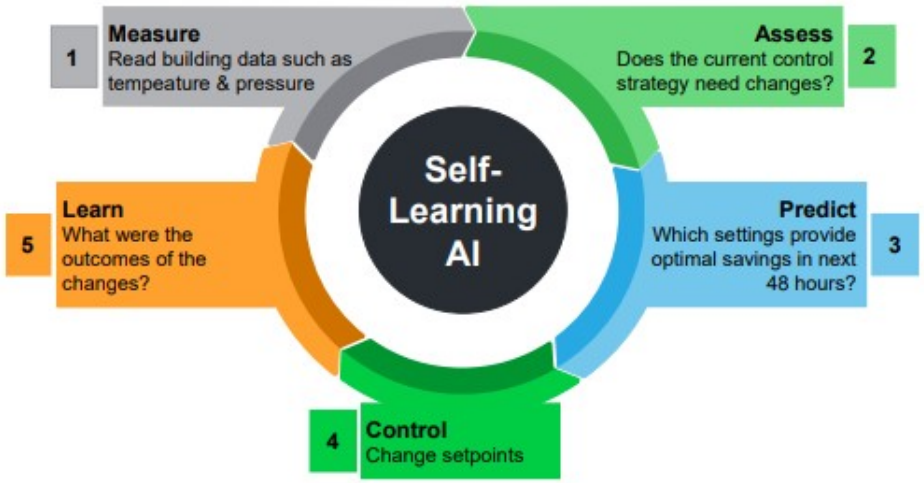
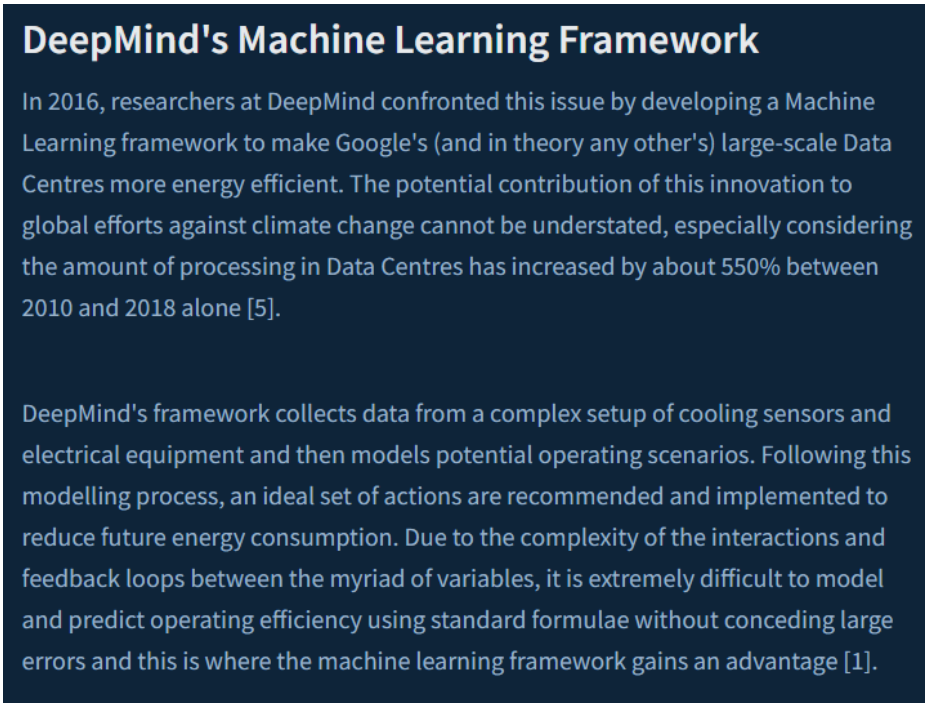


Рисунок 1.2 Як загальна модель ШІ навчається з часом

Google DeepMind використовує технології штучного інтелекту для оптимізації енергоспоживання в Центрах обробки даних Google. Їхні алгоритми допомагають знижувати витрати електроенергії та покращувати енергоефективність. [19]



DeepMind's Machine Learning Framework

In 2016, researchers at DeepMind confronted this issue by developing a Machine Learning framework to make Google's (and in theory any other's) large-scale Data Centres more energy efficient. The potential contribution of this innovation to global efforts against climate change cannot be understated, especially considering the amount of processing in Data Centres has increased by about 550% between 2010 and 2018 alone [5].

DeepMind's framework collects data from a complex setup of cooling sensors and electrical equipment and then models potential operating scenarios. Following this modelling process, an ideal set of actions are recommended and implemented to reduce future energy consumption. Due to the complexity of the interactions and feedback loops between the myriad of variables, it is extremely difficult to model and predict operating efficiency using standard formulae without conceding large errors and this is where the machine learning framework gains an advantage [1].

Рисунок 1.3 Дані про фреймворк навчання

Enel, одна з провідних енергетичних компаній, використовує технології штучного інтелекту для оптимізації виробництва електроенергії, прогнозування попиту та управління електромережами. [20]

Після аналізу продуктів конкурентів, було визначено їх переваги та недоліки. Результати аналізу представлені у таблиці 1.1.

Таблиця 1.1 — Порівняльна таблиця характеристик аналогів

Аналог	Google DeepMind	Enel	Schneider Electric
Прогнозування споживання електроенергії	+	+	+
Орієнтованість на приватні будинки	-	-	+
Прогнозування та оптимізація виробництва електроенергії	-	+	-
Використання погодних даних	-	-	+

Дані з таблиці 1.1 дають змогу під час роботи сконцетруватися на ключових моментах, які дали би переваги перед аналогами:

- прогнозування як споживання, так і генерації електроенергії;
- використання погодних даних для покращення ефективності роботи нейромережі;
- орієнтація на приватні будинки.

1.3 Мета та задачі дослідження

Метою даного дослідження є розробка додатку для прогнозування споживання електроенергії за допомогою ШІ для полегшення діяльності бізнесу у роботі з електроенергії шляхом надання точних прогнозів стосовно того, який обсяг електроенергії буде спожитий тим чи іншим закладом протягом наступної доби.

Основні вимоги до створюваного програмного продукту є наступними:

- здатність моделі ШІ узагальнювати дані таким чином, щоб не залежати від конкретних проміжків значень;
- забезпечити зручний інтерфейс взаємодії з користувачем;
- забезпечити точне прогнозування споживання та генерації електроенергії.

Для досягнення мети проєкту необхідно виконати наступні задачі:

- розробити механізм обробки даних;
- розробити нейромережу;
- протестувати нейромережу;
- спроєктувати додаток;
- розробити та протестувати додаток;
- створити графічний інтерфейс для взаємодії з користувачем.

1.4 Вибір та обґрунтування засобів реалізації

Вимоги до проєкту в цілому, структури додатку, видів забезпечення та функціонування системи описані у технічному завданні на розробку проєкту (додаток А).

Для реалізації даного додатку було обрано такі технології як мову програмування Java 11 та фреймворк DL4j для глибокого навчання ШІ.

Дану комбінацію було обрано виходячи з таких аспектів:

- портативність: Java може запускатися на будь-якому пристрої куди було портовано віртуальну машину Java (JVM), а отже на сьогоднішній момент це від різних linux-подібних систем (у т.ч. і Android) до Windows та MacOS. DL4j також підтримує роботу на Android, Linux, Windows, MacOS;
- широкий спектр інструментів та бібліотек: Java має багато як стандартних, так і від третіх сторін бібліотек, які суттєво спрощують роботу з різними аспектами розробки ПЗ;
- популярність: DL4j є найпопулярнішим та найбільш документованим фреймворком з глибокого навчання на Java.

1.5 Модель нейронної мережі

Для виконання даної роботи слід розробити архітектуру нейронної мережі, яка буде використана для прогнозування споживання та генерації електроенергії. У першу чергу, враховуючи те, що для прогнозування будуть використовуватись історичні дані за n днів, слід використати для досягнення цієї цілі архітектуру LSTM. Також, для того, щоб запобігти перенавчанню, слід використати dropout шар, а на виході буде використовуватись щільний шар. Для роботи з історичними даними, слід враховувати цикл в тиждень — 7 днів, адже в залежності від того, який саме це день тижня, буде різні показники по тому ж споживанню, отже як основу для визначення кількості нейронів на шар — слід використовувати числа кратні 7. Загалом, архітектуру збудованої мережі можна записати у вигляді таблиці 3.1 для моделі, що не використовує температурні дані і таблиці 3.2 для тієї, що використовує такі дані.

Таблиця 1.2 — Архітектура нейромережі без використання температурних даних

Тип шару	Кількість нейронів на вході	Кількість нейронів на виході
LSTM	1	7
Dropout	7	7
Dense	7	21
RNNOut	21	1

Таблиця 1.3 — Архітектура нейромережі з використанням температурних даних

Тип шару	Кількість нейронів на вході	Кількість нейронів на виході
LSTM	3	7
Dropout	7	7
Dense	7	21
RNNOut	21	1

2 ФУНКЦІОНАЛЬНЕ МОДЕЛЮВАННЯ ПРОЄКТУ

IDEF0 – це методологія та нотація для функціонального моделювання, що використовується для опису та формалізації бізнес-процесів. Контекстна діаграма зображена на рис. 2.1.

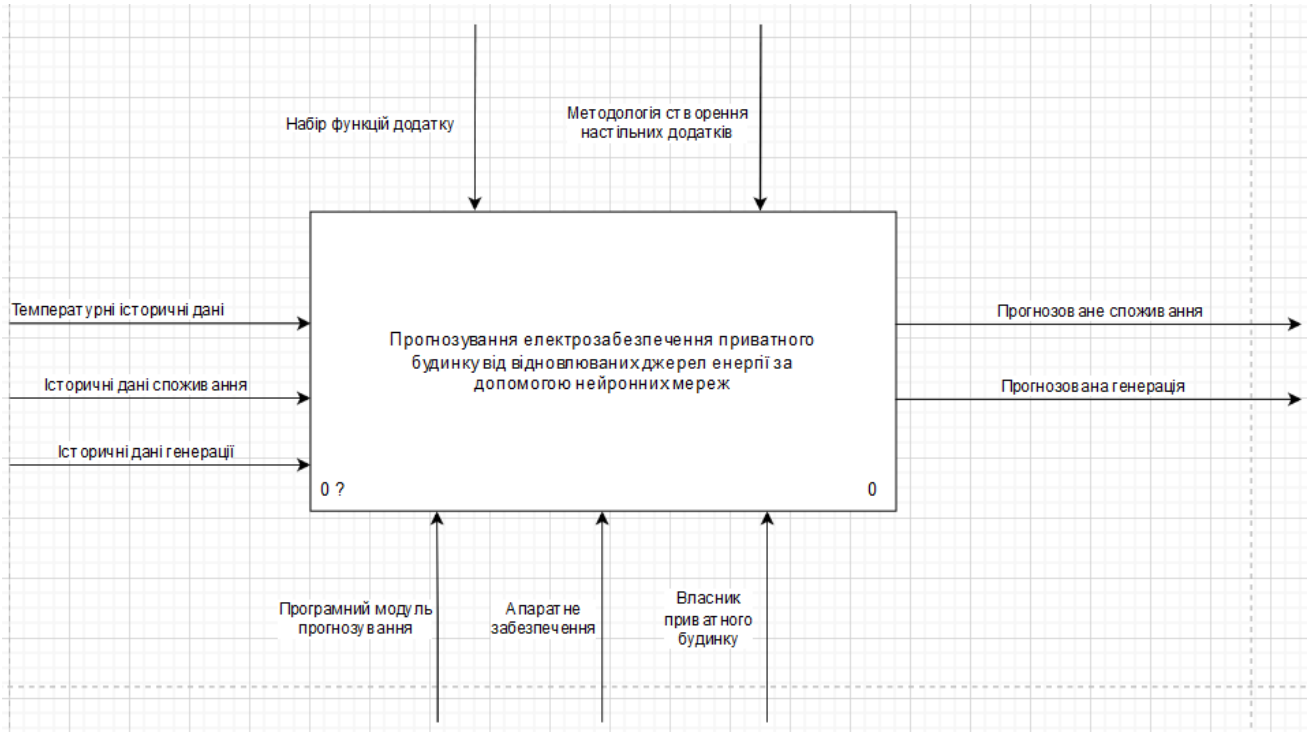


Рисунок 2.1 — Контекстна діаграма процесу прогнозування електрозабезпечення

На рисунку 2.2 наведена діаграма декомпозиції в нотації IDEF0, яка складається з етапів форматування і обробки даних, прогнозування споживання електроенергії приватним домогосподарством та прогнозування генерації електричної енергії для потреб приватного будинку.

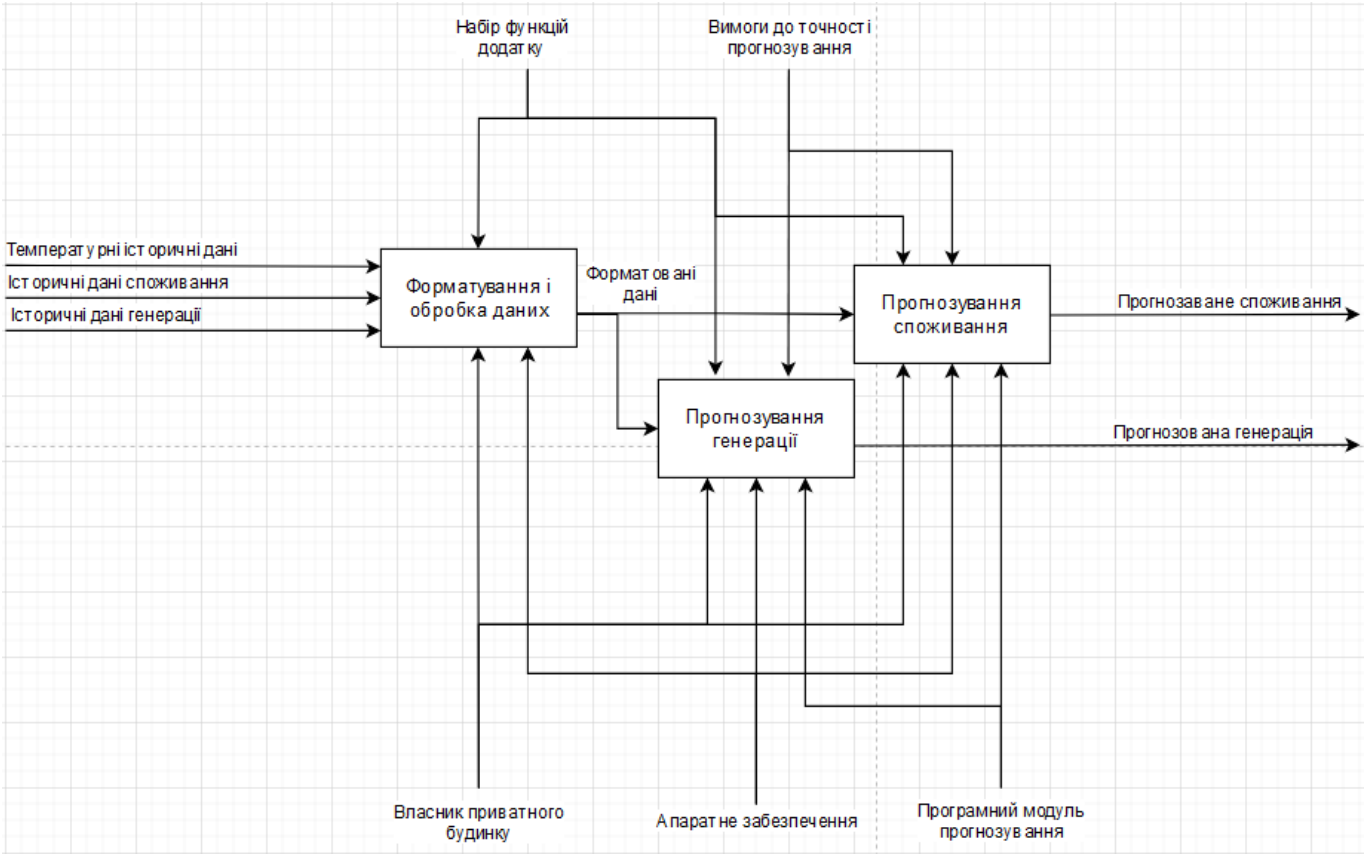


Рисунок 2.2 — Діаграма декомпозиції

Діаграма варіантів використання програмного додатку прогнозування електрозабезпечення наведена на рис.2.3.

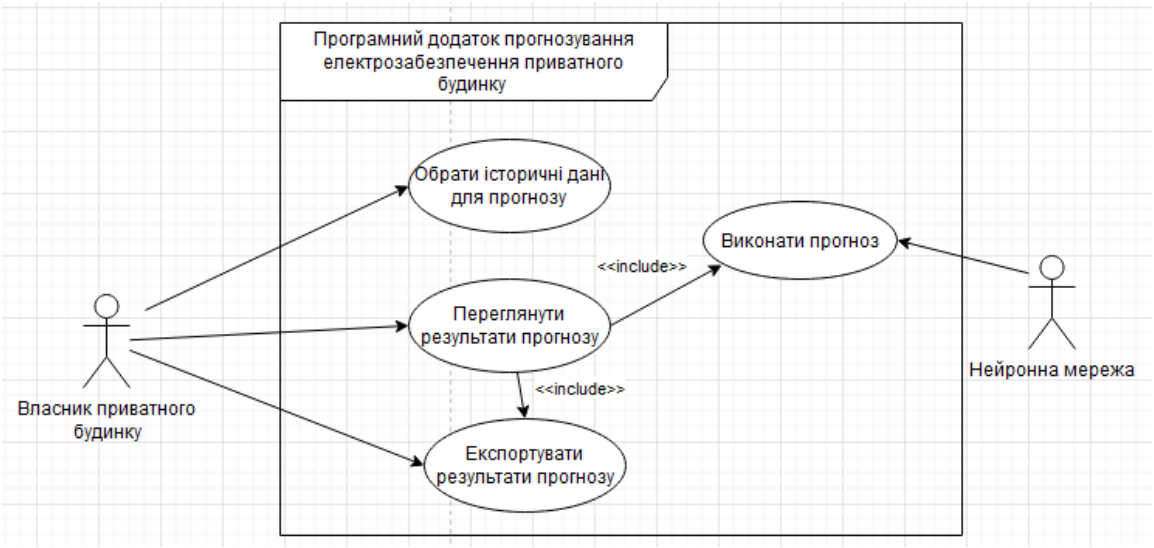


Рисунок 2.3 – Діаграма варіантів використання програмного додатку прогнозування

Далі наведений опис варіантів використання — таблиця 2.1.

Таблиця 2.1 — Опис варіантів використанням

Варіанти використання	Опис
Обрати історичні дані для прогнозу	Вибір файлу Excel, який містить потрібний обсяг даних для прогнозу
Переглянути результати прогнозу	Переглянути графік даних, що були спрогнозовані на основі попередніх історичних даних
Експортувати дані прогнозу	Вивантажити результати прогнозу у файл формату Excel
Виконати прогноз	Здійснити прогноз на наступний проміжок часу, використовуючи набір історичних даних за минулий проміжок часу

2 ПРАКТИЧНА РЕАЛІЗАЦІЯ ДОДАТКУ

3.1 Налаштування проєкту

Етап практичної реалізації починається з налаштування проєкту Java засобами інструменту для збірки проєктів Maven. Залежностями проєкту було визначено такі бібліотеки на фреймворки:

- Apache POI для роботи з XLSX файлами;
- DL4j для роботи з LSTM нейромережею та ND4j для роботи з багатовимірними масивами;
- Lombok для спрощення написання рутинного коду на Java.

Далі слід визначити структуру файлів проєкту. Для тренування нейромережі буде використовуватись клас Main, а для взаємодії з користувачем за допомогою інтерфейсу, щоб той міг скористатись натренованою моделлю для прогнозування електрозабезпечення приватного будинку — клас UI (рис. 3.1).

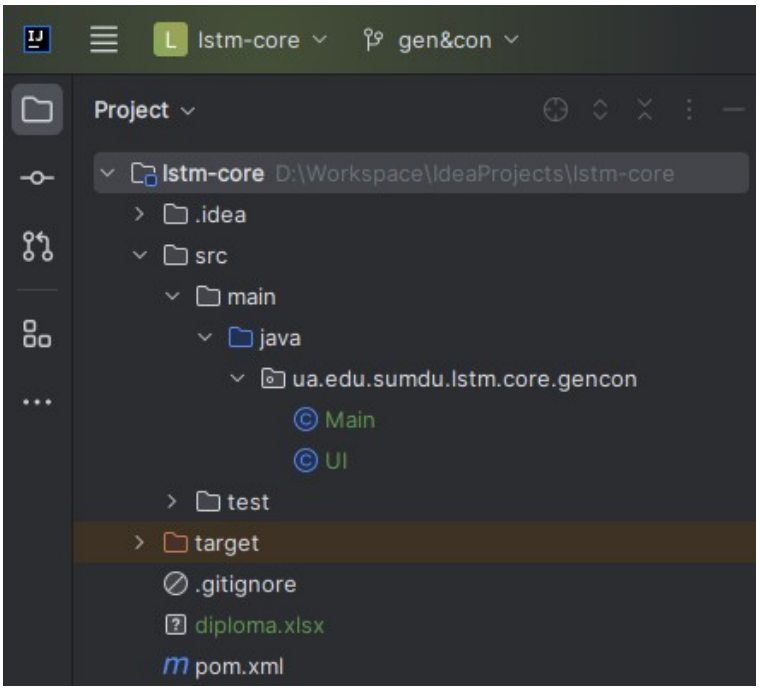


Рисунок 3.1 — Структура файлів проєкту

3.2 Тренування нейромережі

Для тренування нейронної мережі з прогнозування споживання та генерації було використано набір даних енергозабезпечення (споживання та генерація за допомогою сонячних панелей) приватного будинку за три з половиною роки з інформацією про температуру на початок (6:00) та кінець дня (18:00). Розподіл показників генерації впродовж даного періоду часу та показники температури можна побачити на рисунку 3.2. Розподіл значень споживання, у свою чергу, можна побачити на рисунку 3.3.

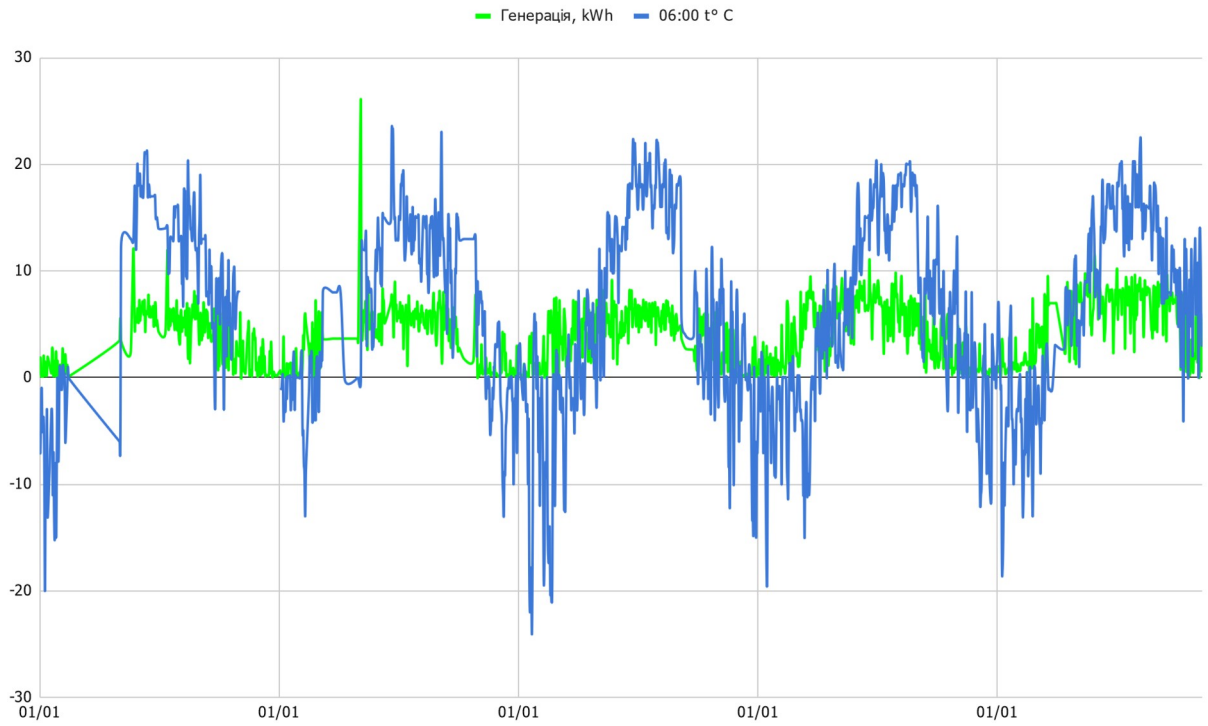


Рисунок 3.2 — Графік генерації електроенергії сонячними панелями

Рисунок 3.3 — Графік споживання електроенергії приватним будинком

Наступним кроком у розробці додатку є створення механізму обробки значень з набору даних. Зокрема розбиття масиву даних на два масиви, де перший буде використаний для тренування моделі нейромережі, а другий для тестування результатів тренування — наскільки добре нейромережа навчилась прогнозувати споживання та генерацію електроенергії. Для цього було використано половину вищезазначеного набору даних. Слід зазначити, що для маркування правильних даних, які мають бути спрогнозовані для n днів (n є {1, 2, 3, 5, 7, 14}) використовуються дані за n наступних днів.

Після формування масивів даних для тренування та тестування слід розробити саму нейромережу. Для цього буде використано фреймворк для глибокого навчання на Java – DL4j.

Після експериментів з різною кількістю днів, що прогнозуються, було отримано результати по точності кожного з варіантів. Похибки при прогнозуванні споживання електроенергії з використанням температурних даних можна побачити на таблиці 3.1, при генерації електроенергії з використанням температурних даних

— таблиця 3.2; без використання температурних даних: для споживання — таблиця 3.3, для генерації — таблиця 3.4.

Таблиця 3.1 — Похибки при прогнозуванні споживання з використанням температурних даних

Кількість днів	MSE	MAE	RMSE	MAPE
1	45.99	6.78	6.78	4.99%
2	811.91	21.22	28.49	20.29%
3	2412.76	44.32	49.12	43.59%
5	6242.34	78.26	79.01	75.21%
7	2593.57	43.57	50.93	40.43%
14	599.41	18.13	24.48	8.94%

Таблиця 3.2 — Похибки при прогнозуванні генерації з використанням температурних даних

Кількість днів	MSE	MAE	RMSE	MAPE
1	147231	383.71	383.71	7.00%
2	401357	610.61	633.53	13.20%
3	783581	710.62	885.20	15.85%
5	2017777	1163.86	1420.48	28.59%
7	2216379	1215.23	1488.75	27.62%
14	1208186	967.75	1099.18	16.13%

Таблиця 3.3 — Похибки при прогнозуванні споживання без використання температурних даних

Кількість днів	MSE	MAE	RMSE	MAPE
1	70.88	8.42	8.42	6.19%
2	497.14	19.55	22.30	18.33%
3	1915.77	34.60	43.77	34.78%
5	4110.77	62.87	64.12	60.85%
7	3266.37	48.98	57.15	45.36%
14	630.81	18.40	25.12	8.79%

Таблиця 3.4 — Похибки при прогнозуванні генрації без використання температурних даних

Кількість днів	MSE	MAE	RMSE	MAPE
1	3201	56.58	56.58	1.03%
2	1056244	900.53	1027.74	16.75%
3	802282	718.91	895.70	15.38%
5	1042304	778.94	1020.93	19.12%
7	1247517	908.02	1116.92	21.10%
14	1498755	1037.70	1224.24	16.85%

Як можна побачити з результатів, найточнішою моделлю є та, яка працює з періодом в 1 день, але враховуючи те, що прогнозування в 1 день спрощує лише короткострокове планування, але якщо ми говоримо про прогнозування в 14 днів, то модель менш точна, але за рахунок такого великого проміжку часу, навіть попри точність дані результати будуть корисними при коротко- та довгостроковому плануванні.

Якщо порівнювати між моделями, що використовують для прогнозування окрім історичних даних, температурні дані та тими, які використовують лише

історичні, то можна зробити такий висновок: для прогнозування споживання температурні дані покращують показники, а отже є сенс використовувати саме такі моделі; для прогнозування генерації ситуація ж зворотна — температурні дані зменшують ефективність нейромережі, можливо, не в останню чергу, через циклічність історичність даних залежну саме від сезону протягом року, а не конкретно температури.

3.3 Демонстрація роботи додатку

Інтерфейс меню складається з двох складових: головне меню та вікно відображення результатів прогнозування.

Головне меню містить три кнопки:

- вибір файлу історичних даних для прогнозування;
- вибір робочої теки для збереження даних прогнозування;
- перехід на вікно графіку прогнозованих даних.

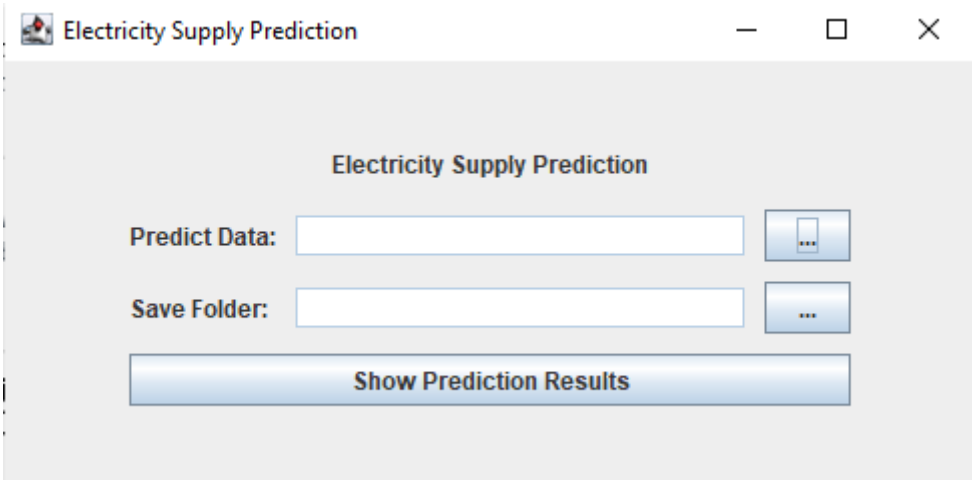


Рисунок 3.4 — Головне меню додатку

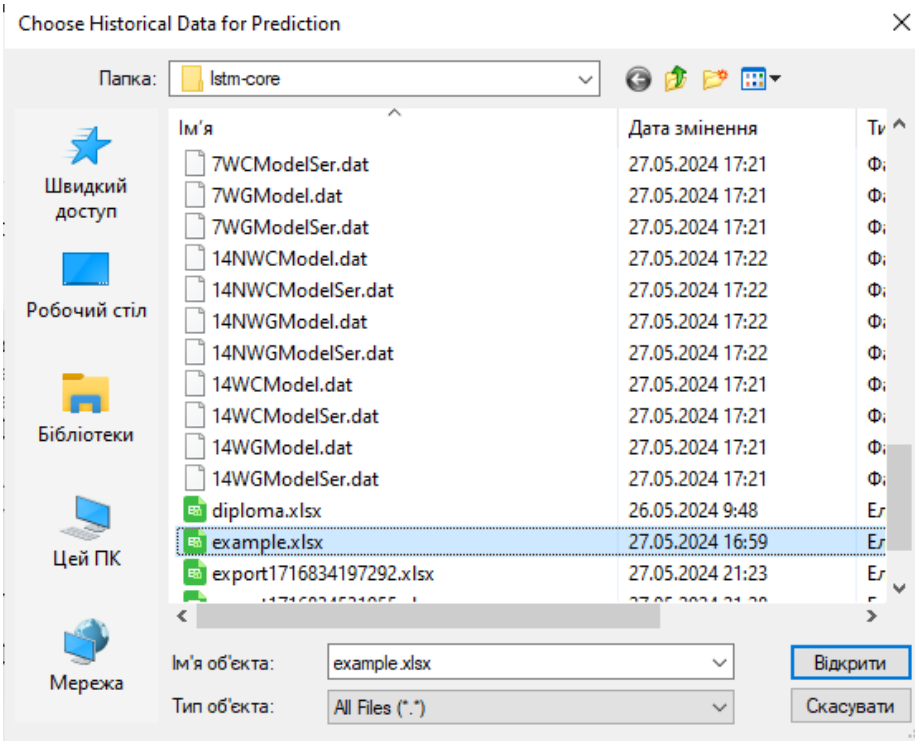


Рисунок 3.5 — Вибір історичних даних для прогнозування

example.xlsx — LibreOffice Calc

Файл Зміни Перегляд Вставка Формат Стилі Аркуш Дані

Arial 10 pt B I U A

F9 f_x Σ =

	A	B	C	D
1	Генерація, Wh	Споживання, Wh	06:00 t C	18:00 t C
2	330,00	168,00	-7,00	-5,00
3	1 890,00	56,00	-3,00	-3,00
4	220,00	38,00	-1,00	-2,00
5	270,00	76,00	-5,00	-2,00
6	200,00	80,00	-4,00	-3,00
7	240,00	21,00	-4,00	-5,00
8	2 070,00	246,00	-8,00	-5,00
9	170,00	82,00	-20,00	-7,00
10	1 520,00	51,00	-9,00	-9,00
11	310,00	46,00	-9,00	-9,00
12	190,00	0,00	-3,00	-3,00
13	360,00	85,00	-13,00	-3,00
14	1 950,00	125,00	-3,00	-9,00
15	440,00	86,00	-4,00	-1,00

Рисунок 3.6 — Приклад вигляду файлу, що приймається додатком

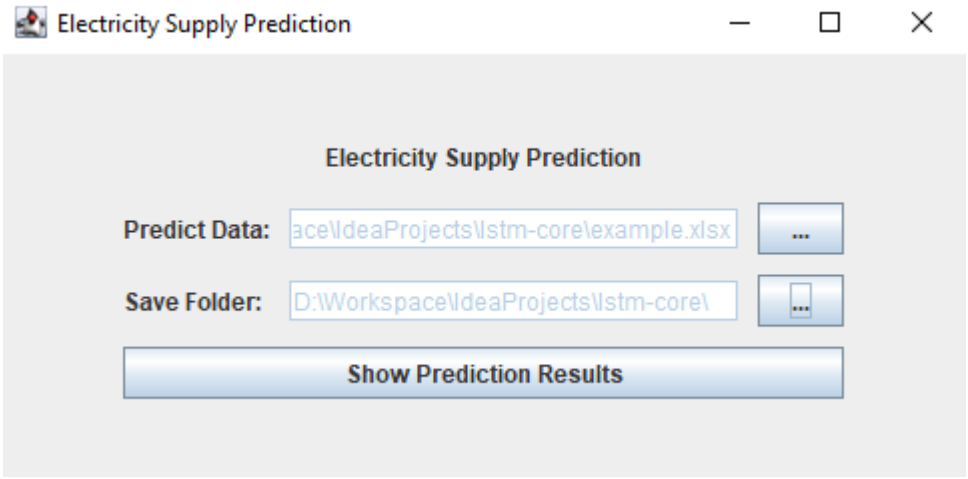


Рисунок 3.7 — Вигляд головного меню після того, як необхідні поля заповнені

Після заповнення полів можна перейти до перегляду графіку прогнозованих даних.

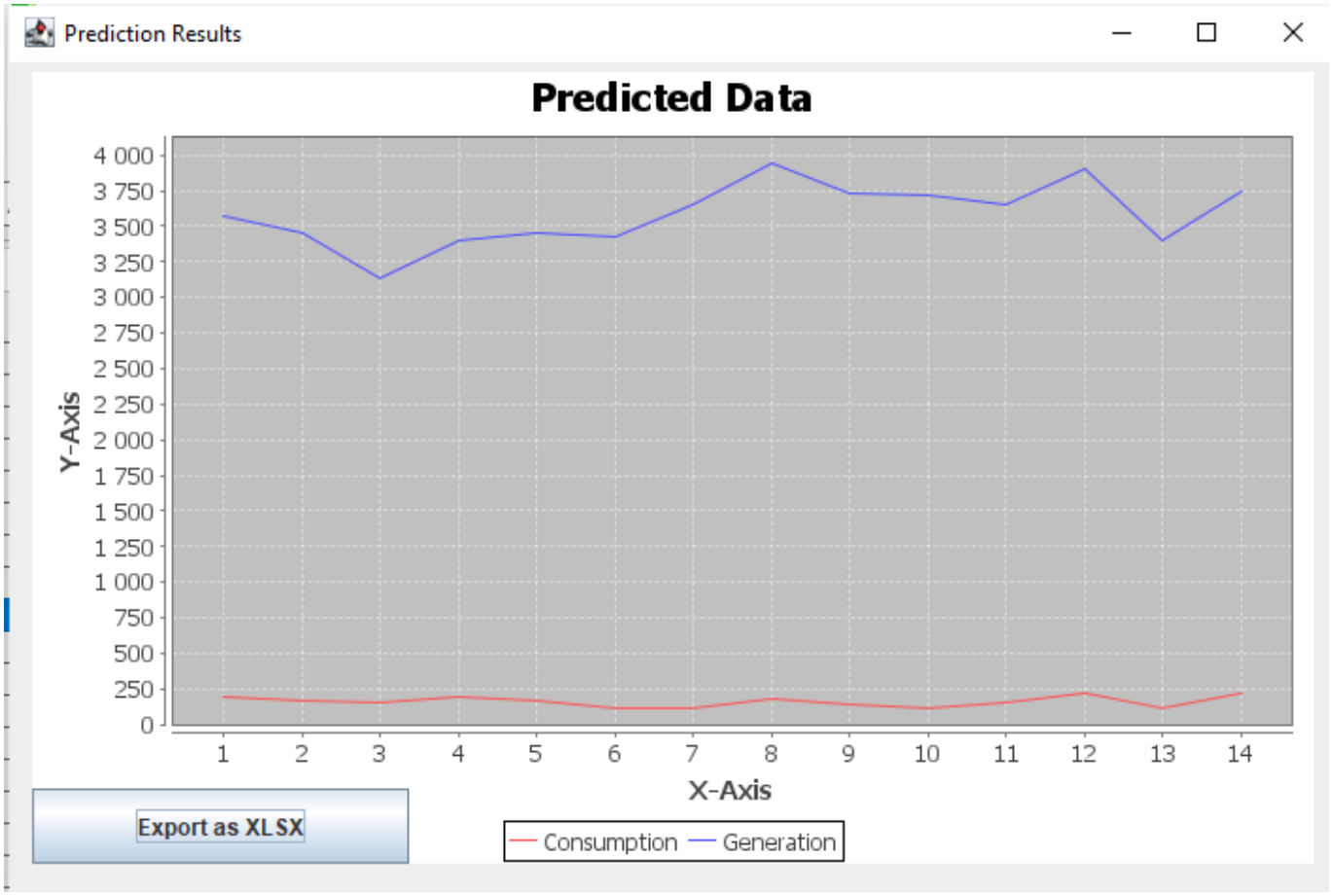


Рисунок 3.8 — Графік прогнозованих значень

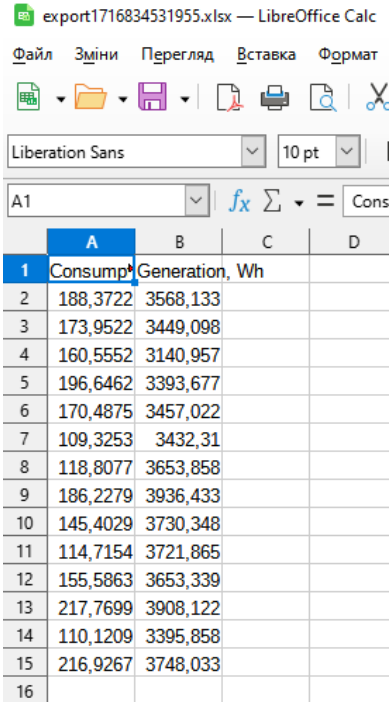


Рисунок 3.9 — Вигляд експортованого файлу Excel з результатами прогнозу

ВИСНОВКИ

У ході розробки додатку для прогнозування електрозабезпечення приватного будинку від альтернативних джерел енергії було проведено аналіз останніх досліджень у галузі та наявних технічних рішень на ринку програмного забезпечення, визначено їх переваги та недоліки, що допомогло у формуванні вимог до розроблюваного додатку. Серед іншого, було визначено, що важливо прогнозувати не лише споживання електроенергії, але й генерацію від альтернативних джерел, зокрема сонячних панелей. Також, що звернути увагу на прогнозування у середньо- та довгостроковій перспективах.

Для реалізації додатку було обрано певний перелік бібліотек і фреймворків, які дозволяють створювати, тренувати та використовувати моделі нейронних мереж, працювати з різними файлами і взаємодіяти з користувачем, використовуючи графічний інтерфейс.

Розроблений додаток дозволяє прогнозувати електропостачання приватного будинку від альтернативних джерел енергії, а саме споживання та генерацію електроенергії, за допомогою нейронних мереж. Він може бути використаний для прогнозування споживання електроенергії проміжком у 14 днів, із точністю до 91.06% для споживання та з точністю до 83.87% для прогнозування генерації електроенергії.

Даний додаток має практичне застосування у сфері планування ресурсів власниками та орендарями приватних будинків, які зокрема мають в електромережі сонячні панелі.

СПИСОК ДЖЕРЕЛ

1. What is Artificial Inteligence (AI)? | IBM. *IBM*. URL: https://www.ibm.com/topics/artificial-intelligence (Date of access: 07.04.2024)

2. Managing Power Grid Complexity. *Smart Energy International*. URL: https://www.smart-energy.com/industry-sectors/data-analytics/managing-power-grid-complexity-the-5-biggest-priorities-for-utilities-in-2024/ (Date of access: 07.04.2024)

3. Applications of Artificial Intelligence & Associated Technologies. Jatin Borana. *ETEBMS-2016*. URL: https://test.globalinfocloud.com/technodigisoftnew/wp-content/uploads/2019/07/Applications-of-Artificial-Intelligence-Associated-Technologies.pdf (Date of access: 07.04.2024)

4. Machine Learning: Algorithms, Real-World Applications and Research Directions. Iqbal H. Sarker. *SN Computer Science.* URL: https://link.springer.com/content/pdf/10.1007/s42979-021-00592-x.pdf (Date of access: 07.04.2024)

5. A Review of Machine Learning and Deep Learning Applications. Pramila P. Shinde, Seema Shah. *International Conference on Computing Communication Control and automation.* URL: https://doi.org/10.1109/ICCUBEA.2018.8697857 (Date of access: 07.04.2024)

6. Generalization in AI Systems | by Egor Dezhik. *Medium*. URL: https://towardsdatascience.com/generalization-in-ai-systems-79c5b6347f2c (Date of access: 07.04.2024)

7. A data-driven strategy using long short term memory models and reinforcement learning to predict building electricity consumption. Xinlei Zhou, Wenye Lin, Ritunesh Kumar, Ping Cui, Zhenjun Ma. *Applied Energy.* URL: https://doi.org/10.1016/j.apenergy.2021.118078 (Date of access: 07.04.2024)

8. Machine-Learning based methods in short-term load forecasting. Weilin Guo, Liang Che, Mohammad Shahidehpour, Xin Wan. *The Electricity Journal.* URL: https://doi.org/10.1016/j.tej.2020.106884 (Date of access: 07.04.2024)

9. A Review of Deep Learning Models for Time Series Prediction. Z. Han, J. Zhao, H. Leung, K. F. Ma and W. Wang. *IEEE Sensors Journal.* URL: https://ieeexplore.ieee.org/abstract/document/8742529 (Date of access: 07.04.2024)

10. How do you scale up your LSTM model to handle large or complex datasets? *LinkedIn*. URL: https://www.linkedin.com/advice/3/how-do-you-scale-up-your-lstm-model (Date of access: 07.04.2024)

11. The Exploding and Vanishing Gradients Problem in Time Series. *Medium*. URL: https://medium.com/metaor-artificial-intelligence/the-exploding-and-vanishing-gradients-problem-in-time-series-6b87d558d22 (Date of access: 07.04.2024)

12. Understanding LSTM: Architecture, Pros and Cons, and Implementation. *Medium*. URL: https://medium.com/@anishnama20/understanding-lstm-architecture-pros-and-cons-and-implementation-3e0cca194094 (Date of access: 07.04.2024)

13. LSTM — Implementation, Advantages and Diadvantages. *Medium*. URL: https://medium.com/@prudhviraju.srivatsavaya/lstm-implementation-advantages-and-diadvantages-914a96fa0acb (Date of access: 07.04.2024)

14. Deep Learning with Kernel Flow Regularization for Time Series Forecasting. Mahdy Shirdel, Reza Asadi, Duc Do, Micheal Hintlian. *ResearchGate*. URL: https://www.researchgate.net/publication/354865221_Deep_Learning_with_Kernel_Flow_Regularization_for_Time_Series_Forecasting (Date of access: 07.04.2024)

15. COMPARISON OF SHORT-TERM FORECASTING METHODS OF ELECTRICITY CONSUMPTION IN MICROGRIDS. Yu. V. Parfenenko, V. V. Shendryk, Ye. P. Kholiavka, P. M. Pavlenko. *Radio Electronics, Computer Science, Control.* URL: https://doi.org/10.15588/1607-3274-2023-1-2 (Date of access: 07.04.2024)

16. What is Dense Layer in Neural Network? *AIM*. URL: https://analyticsindiamag.com/topics/what-is-dense-layer-in-neural-network/ (Date of access: 07.04.2024)

17. Deeplearning4j Suite Overview. *DeepLearning4j*. URL: https://deeplearning4j.konduit.ai/ (Date of access: 07.04.2024)

18. About Us. *Schneider Electric Global*. URL: https://www.se.com/ww/en/about-us/ (Date of access: 07.04.2024)

19. About Google DeepMind. *Google DeepMind*. URL: https://deepmind.google/about/ (Date of access: 07.04.2024)

20. Enel Group: Energy, sustainability and innovation. *Enel Group*. URL: https://www.enel.com/ (Date of access: 07.04.2024)

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ
на розробку інформаційної системи
«Додаток для прогнозування електрозабезпечення приватного будинку від відновлюваних джерел енергії за допомогою нейронних мереж»

ПОГОДЖЕНО:

Доцент кафедри інформаційних технологій

_____ Парфененко Ю.В.

Студент групи ІТ-01

_____ Богачов М.В.

Суми 2024

1. Призначення й мета створення додатку

1.1 Призначення додатку

Додаток має надавати можливість клієнтам адаптувати інформаційну систему під свій приватний будинок, надавши масив даних споживання та генерації електроенергії за певний попередній проміжок часу, а також після цієї адаптації виконати прогнозування споживання та генерації електроенергії на певний проміжок часу в майбутньому, надавши масив даних споживання за попередній проміжок часу.

1.2 Мета створення додатку

Покращення планування клієнтами споживання та генерації електроенергії.

1.3 Цільова аудиторія

До цільової аудиторії додатку можна віднести власників та орендарів приватних будинків, які забезпечуються частково або повністю за допомогою відновлювальних джерел енергії.

2 Вимоги до додатку

2.1 Вимоги до додатку в цілому

2.1.1 Вимоги до структури й функціонування додатку

Додаток має бути десктопним та запускатися в операційній системі Windows. Для аналізу ефективності нейромережі слід використовувати такі метрики: MAPE (середнє абсолютне відхилення у відсотках), RMSE (кореневе середньоквадратичне відхилення) та MAE (середня абсолютна похибка). Основною метрикою для оцінювання має бути MAPE. Усі дані для роботи з ним мають зберігатися локально.

2.1.2 Вимоги до персоналу

Від персоналу не має вимагатися особливих технічних навичок для підтримки й експлуатації додатку, окрім загальних навичок роботи з персональним комп'ютером і пакетом Microsoft Office або LibreOffice для роботи з даними в табличному форматі.

2.1.3 Вимоги до збереження інформації

Уся інформація надана у додатку буде зберігатися на пристрої, на якому він встановлений.

2.1.4 Вимоги до розмежування доступу

Вимог для розмежування доступу немає, користувач має доступ до усіх функцій додатку.

2.2 Структура додатку

2.2.1 Загальна інформація про структуру додатку

Структура додатку є простою — одна форма, на якій розташовано всю потрібну інформацію та дії (кнопки).

2.2.2 Дизайн та структура додатку

Стиль додатку має бути простим та сучасним для спрощення взаємодії з користувачем.

Розташування елементів на головній меню додатку схематично показано на рисунку А.1.

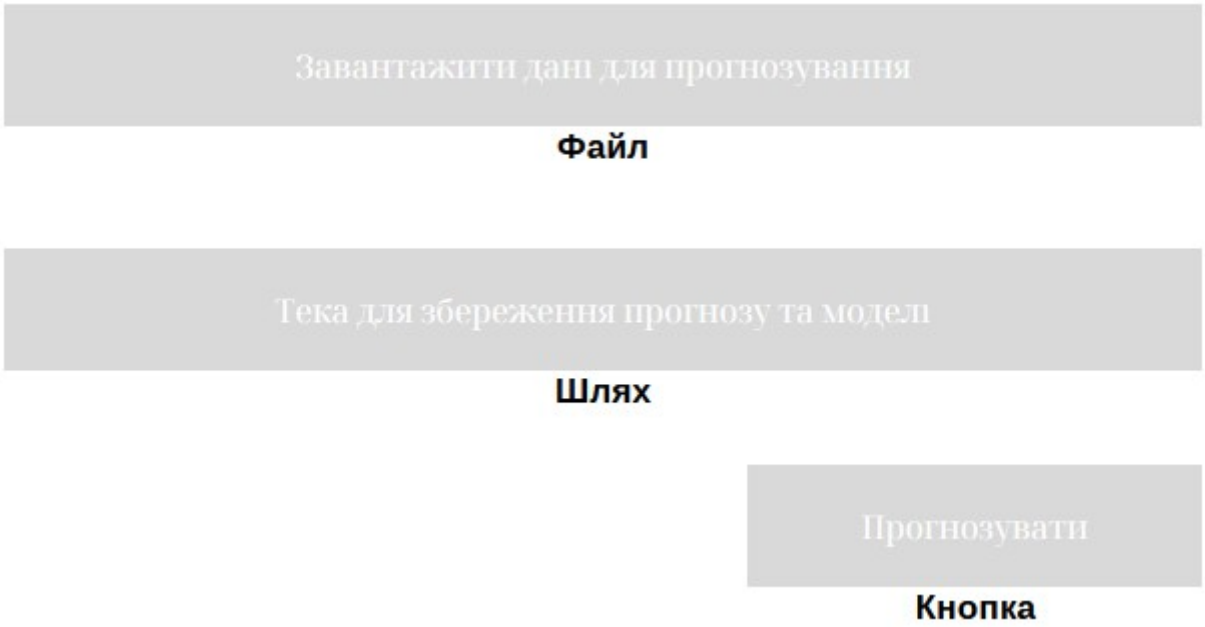


Рисунок А.1 – Схема графічного інтерфейсу додатку

2.3 Вимоги до функціонування системи

2.3.1 Потреби користувача

Потреби користувача, визначені на основі рішення замовника, представлені у таблиці А.1.

Таблиця А.1 – Потреби користувача

ID	Потреби користувача	Джерело
UN-01	Обрати робочу теку	Клієнт
UN-02	Обрати історичні дані для прогнозування	Клієнт
UN-03	Переглянути результати прогнозування	Клієнт
UN-04	Можливість зберегти результати прогнозування	Клієнт

2.3.2 Функціональні вимоги

На основі потреб користувача були визначені такі функціональні вимоги:

- вибір робочої теки;
- підказки при заповненні поля або виборі документу;
- неактивні поля для заповнення при незаповнених попередніх полях;
- проведення тренування нейромережі;
- проведення прогнозування.

2.3.3 Системні вимоги

Даний розділ визначає, розподіляє та вказує на системні вимоги, визначені розробником. Їх перелік наведений в таблиці А.2.

Таблиця А.2 – Системні вимоги

ID	**Системні вимоги**	**Пріоритет**	**Опис**
SR-01	Алгоритм тренування моделі нейронної мережі	M	Проводить тренування моделі нейромережі на наданих користувачем даних
SR-02	Алгоритм прогнозування нейронною мережею	M	Прогнозує обсяги споживання та генерації електроенергії
SR-03	Модуль збереження файлів програми	M	Використовується для збереження натренованої моделі та результатів прогнозування
SR-04	Модуль завантаження файлів	M	Завантажує потрібні файли для роботи програми зі вказаного користувачем шляху
SR-05	Графік по даних для тренування	C	Відображає графік значень

			датасету для тренування
SR-06	Графік по даних для прогнозування	C	Відображає графік значень даних на вхід до нейромережі
SR-07	Графік по прогнозу	C	Відображає графік результатів прогнозу

Умовні позначення в таблиці А.2:

Must have (M) – вимоги, які повинні бути реалізовані в системі;

Could have (C) – вимоги, які можуть бути реалізовані, але вони не є центральною ціллю проекту.

2.4 Вимоги до видів забезпечення

2.4.1 Вимоги до інформаційного забезпечення

Реалізація додатку відбувається з використанням:

- JDK 11;
- ND4j – для роботи з багатовимірними масивами;
- DL4j – для роботи з нейромережами: створення, тренування та використання моделей.
- Apache POI – для роботи з Excel файлами: створення, редагування та читання.

2.4.2 Вимоги до лінгвістичного забезпечення

Додаток має бути виконаний англійською мовою.

2.4.3 Вимоги до програмного забезпечення

Програмне забезпечення персонального комп’ютера, на якому встановлено даний додаток:

- Середовище виконання Java (JRE): версії 11 і вище.

3 Склад і зміст робіт зі створення додатку

Докладний опис етапів роботи зі створення додатку наведено в таблиці А.3.

Таблиця А.3 – Етапи створення додатку

№	Склад і зміст робіт	Строк розробки (у робочих днях)
1	Постановка цілей необхідних для досягнення певного результату	1 день
2	Складання технічного завдання	1 дні
3	Створення макету дизайну додатку	1 дні
4	Підготовка прототипу	2 дні
5	Створення модулю тестування і прогнозування	5 днів
6	Тестування та відлагодження ефективності створеного алгоритму	5 днів
7	Створення графічного інтерфейсу	1 день
8	Реалізація модулю збереження та завантаження файлів	2 день
9	Перевірка працездатності додатку	1 день
10	Завершення роботи	1 день
	Загальна тривалість робіт	20 днів

4 Вимоги до складу й змісту робіт із введення додатку в експлуатацію

Для того щоб, клієнти змогли скористатися додатком, слід створити інсталятор для зручного розповсюдження. Вимоги до ПЗ на ПК, на якому буде відбуватися робота з додатком мають відповідати вимогам ТЗ.

ДОДАТОК Б

Планування робіт

Попит у сфері аналітики електромереж на сьогоднішній день стрімко зростає, адже з кожним днем кількість та потужність пристроїв, які потребують електроенергії – збільшується, а отже і потреба у оптимізації навантаження локальних мереж. Тому створення додатку прогнозування споживання електроенергії дозволить спростити планування роботи домогосподарства за рахунок точного (більше 90%) прогнозування споживання електроенергії.

Деталізація мети проєкту методом SMART. Для успішності та конкурентоспроможності проєкту треба на концептуальному етапі правильно визначити його мету за допомогою SMART-методу. Вона має ширше формулювання. А саме: “Розробка додатку для прогнозування електрозабезпечення приватного будинку від відновлюваних джерел енергії за допомогою нейронних мереж на основі затвердженого технічного завдання для спрощення планування життєдіяльності приватних будинків до 01 червня 2024 року”. Результати деталізації мети методом SMART розміщені у таблиці Б.1.

Таблиця Б.1 – Деталізація мети методом SMART

Specific (конкретна)	Розробити додаток для прогнозування споживання електроенергії за допомогою ШІ із відповідним функціоналом, затвердженим у технічному завданні.
Measurable (вимірювана)	Створений додаток, який на вхід має приймати дані споживання електроенергії за певний проміжок часу і віддавати прогноз споживання на майбутній проміжок часу.
Achievable (досяжна, узгоджена)	Для виконання проєкту є затверджене технічне завдання, спеціалізовані знання й навички та доступ до використання відповідних технологій: Java, DL4J.
Relevant (реалістична)	Даний додаток суттєво спростить планування роботи домогосподарства за рахунок точного (більше 90%) прогнозування споживання електроенергії.
Time-framed (Обмежена в часі)	Термін досягнення мети проєкту визначено за домовленістю між замовником та виконавцем – кінець 4 курсу (01 червня 2024 року)

Планування змісту робіт. WBS (Work Breakdown Structure – ієрархічна структура робіт) – це графічний вигляд елементів проєкту, які згруповані ієрархією у єдине ціле з продуктом проєкту. Структура декомпозиції робіт орієнтована на досконале виконання робіт по частинам і сама є ключовою частиною проєкту, яка спрямована на організацію командної роботи. Елементами декомпозиції можуть бути продукти, дані та послуги. Більше того, WBS забезпечує необхідним каркасом для ретельної оцінки термінів та контролю та графіків роботи.

На найвищому (першому) рівні розміщений продукт проєкту. Основні дії та заходи, що забезпечують досягнення мети проєкту, зафіксовані на другому рівні декомпозиції. Декомпозиція робіт виконується до тих пір, поки вони не стануть елементарними (простими).

Елементарні роботи – це дії, які мають однозначний чіткий результат, на які призначена відповідальному одна конкретна особа, для якої можна обчислити витрати праці і тривалість виконання. На рисунку Б.1 представлено WBS з розробки додатку для прогнозування споживання генерації електроенергії за допомогою ШІ.

Планування структури виконавців. Наступним етапом після декомпозиції процесів є розробка організаційної структури виконавців або OBS, яка визначається як графічна структура відображення учасників або відповідальних осіб, які беруть участь у реалізації проєкту.

У ролі відповідальних осіб виступають співробітники, що відповідають за організацію і виконання елементарної роботи, що зазначена у WBS. Кожну елементарну роботи можна розглядати як окремий проєкт.

На рисунку Б.2 представлено організаційну структуру планування проєкту. Список виконавців, що функціонують в проєкті описано в таблиці Б.2.

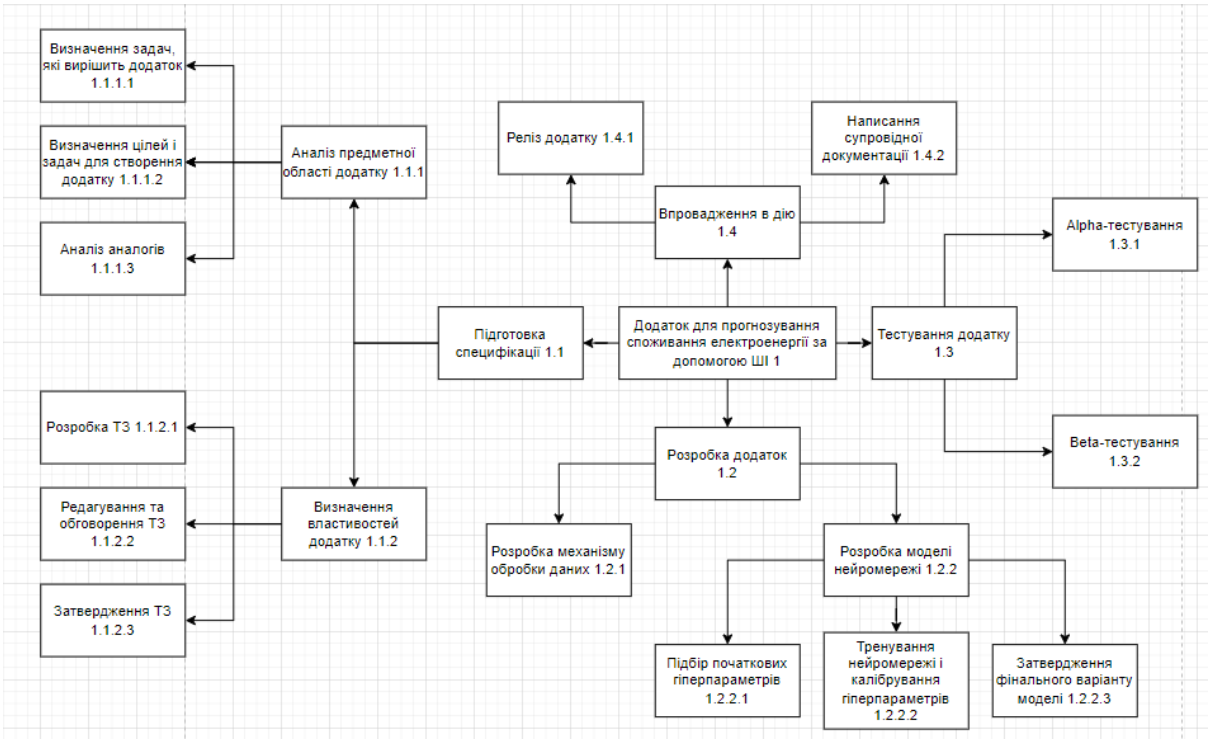


Рисунок Б.1 – WBS-структура робіт проєкту

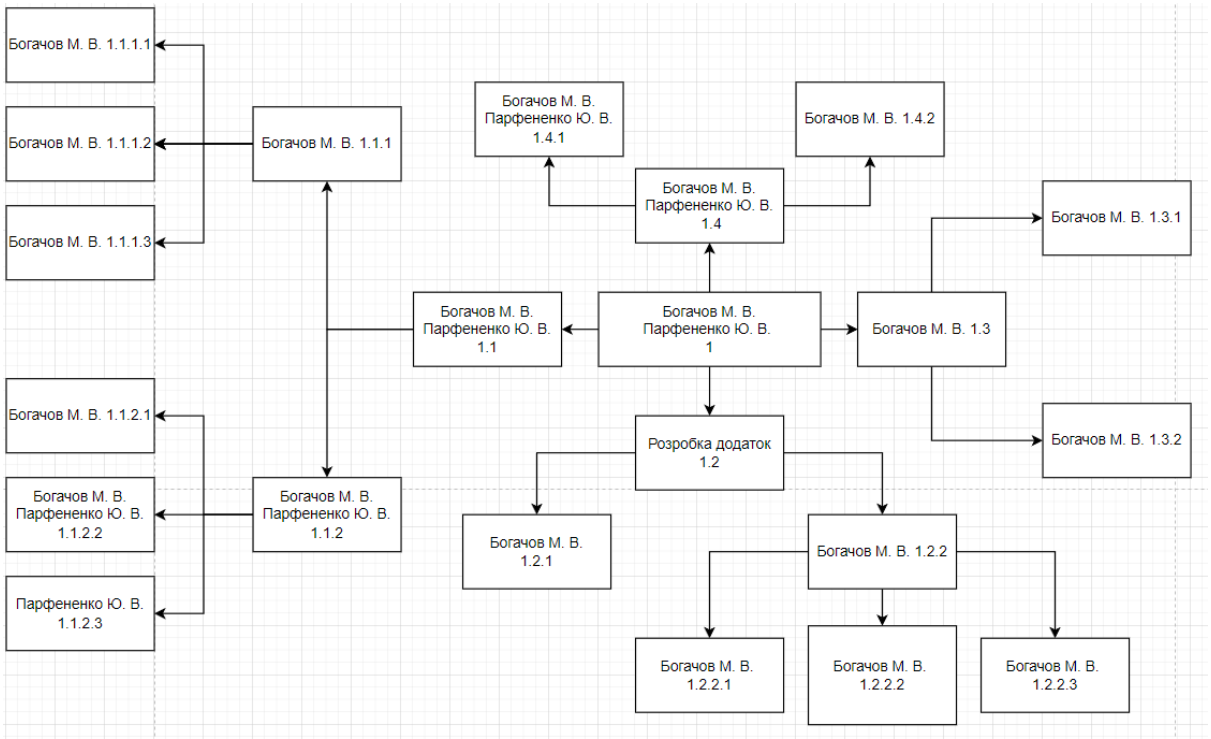


Рисунок Б.2 – OBS-структура робіт проєкту

Таблиця Б.3 – Виконавці проєкту

Роль	ПІБ	Проєктна роль
Розробник	Богачов М. В.	Виконує розробку та тренування моделі нейронної мережі.
Проєктувальник	Богачов М. В.	Виконує проєктування архітектури моделі нейромережі та додатку в цілому.
Тестувальник	Богачов М. В.	Відповідає за тестування функціоналу web-додатку
Менеджер проєкту	Богачов М. В.	Відповідає за виконання термінів та розподіл ресурсів. Виконує збір та аналіз даних.
Керівниця проєкту	Парфененко Ю. В.	Формує завдання на розробку проєкту.

Діаграма Ганта. Побудова календарного графіку (діаграми Ганта) є одним з важливих етапів планування проєкту, що виглядає як розклад виконання робіт з реальним розподілом дат. Завдяки йому можна отримати достовірне уявлення про тривалість процесів з обмеженнями у ресурсах, урахуванням вихідних днів та свят.

Календарний графік проєкту представлено на рисунках Б.3-Б.4.

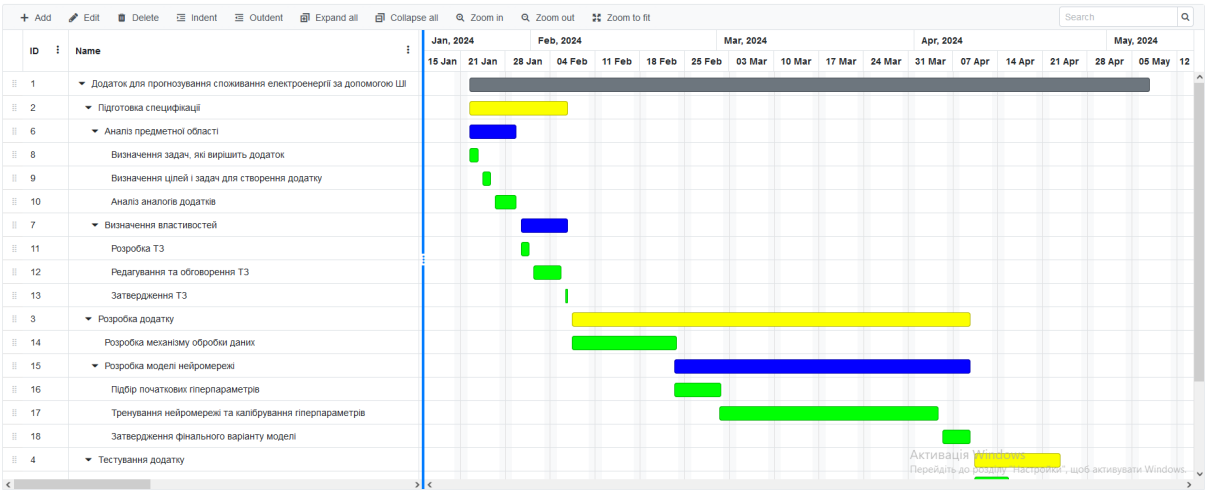


Рисунок Б.3 – Календарний графік проєкту

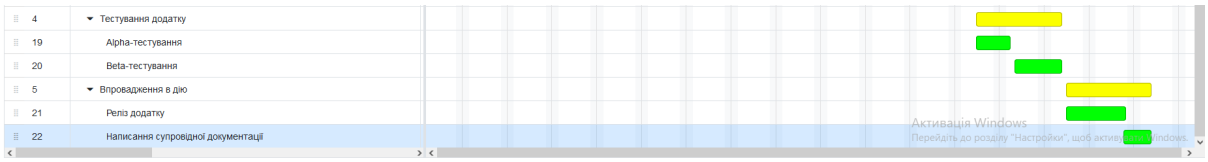


Рисунок Б.4 – Продовження календарного графіку проєкту

Управління ризиками проєкту. Під час виконання якісної оцінки ризиків треба визначити ризики, які мають бути усунені якнайшвидше. В залежності від ступеня важливості ризику – реагування буде відповідне. Наступним етапом є виконання кількісного оцінювання ризиків. Кількісне та якісне оцінювання можуть виконувати одночасно або окремо, що залежить від ступеня забезпечення проєкту. У таблиці Б.3 надано перелік ризиків даного проєкту. Результати оцінки ризиків надано у таблиці Б.4. Таблиця Б.5 представляє шкалу для класифікації ризиків за величиною впливу на проєкт та ймовірністю їх виникнення.

Таблиця Б.4 – Ризики проєкту

№ ризику	Назва (опис) ризику
1	Недостатня кількість тренувальних даних.
2	Недостатній рівень узагальнення натренованої моделі.
3	Потреба у великій обчислювальній потужності при тренуванні.

Таблиця Б.5 – Результати визначення ймовірності, впливу

№ ризику	Назва (опис) ризику	Ймовірність (0,1-0,9)	Вплив (0,05-0,8)	Ранг Й*В
1	Недостатня кількість тренувальних даних.	0,3	0,4	0,12
2	Недостатній рівень узагальнення натренованої моделі.	0,1	0,4	0,04
3	Потреба у великій обчислювальній потужності при тренуванні.	0,5	0,2	0,1
4	Хвороба менеджера проєкту.	0,1	0,4	0,04
5	Виведення ПК розробника з ладу.	0,1	0,8	0,08
6	Завеликий об’єм моделі та як наслідок –	0,3	0,4	0,12

	потреба в високих системних параметрах ПК.			
7	Витік даних для тренування та тестування нейромережі	0,1	0,8	0,08
8	Неправильне використання конфіденційних даних під час роботи додатку	0,1	0,8	0,08
9	Поява ефективнішого аналогу	0,1	0,8	0,08
10	Втрата актуальності відносно ринку ШІ	0,1	0,8	0,08

Таблиця Б.6 – Шкала оцінювання ризиків за ймовірністю виникнення та величиною впливу

Оцінка	Ймовірність виникнення	Вплив ризику	Тип ризику
1	Низька	Низький	Прийнятні
2	Середня	Середній	Виправдані
3	Висока	Високий	Недопустимі

Для того, щоб знизити негативний вплив ризиків на проєкт треба виконати планування реагування на них. До нього входить оцінка наслідків впливу на проєкт і розробка відповідних заходів. Аналіз виконується за показниками, які описані в таблиці Б.6. У результаті планування заходів реагування на ризики проєкту було отримано матрицю ймовірності виникнення та впливу ризиків (табл. Б.7). Зеленим кольором на матриці позначають прийнятні ризики, жовтим – виправдані, а червоним – недопустимі.

Таблиця Б.7 – Матриця відповідності ймовірності та впливу

Ймовірність виникнення ризику	Вплив ризику				
	0,05	0,1	0,2	0,4	0,8
0,9	0,045	0,09	0,18	0,36	0,72
0,7	0,035	0,07	0,14	0,28	0,56
0,5	0,025	0,05	0,10 R3	0,20	0,24
0,3	0,015	0,03	0,06	0,12 R1 R6	0,24
0,1	0,005	0,01	0,02	0,04 R2 R4	0,08 R5 R7 R8 R9 R10

Класифікація ризиків проєкту за рівнем, відповідно до отриманого значення індексу, представлена у таблиці Б.8. У таблиці Б.9 описано ризики та стратегії реагування на кожен із них.

Таблиця Б.8 – Шкала оцінювання ризику за рівнем

№	Назва	Межі	Ризики, які входять (номера)
1	Прийнятні	0,005<=R<=0,05	2, 4
2	Виправдані	0,05<R<=0,14	1, 3, 5, 6, 7, 8, 9, 10
3	Недопустимі	0,14<R<=0,72	-

Таблиця Б.9 – Ризики та стратегії реагування на них

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив	Ранг	Тип стратегії реагування	План А	План Б
RS_1	Новий	Недостатня кількість тренувальних даних.	Середня	Середній	0,12	Прийняття	Наближатись до якомога більшої ефективності за рахунок збільшення параметрів	Змінити архітектуру нейромережі, експерименти з використанням трансформерів
RS_2	Новий	Недостатній рівень узагальнення	Низька	Середній	0,04	Зменшення	Збільшити об’єм даних для тренувань	Розширити діапазони значень при пошуку тестових даних

		натренованої моделі.						
RS_3	Новий	Потреба у великій обчислювальній потужності при тренуванні.	Середня	Середній	0,10	Прийняття	Збільшення ефективності за рахунок найменшої кількості параметрів	Збільшення можливостей тренувальних потужностей за рахунок обчислень в хмарі
RS_4	Новий	Хвороба менеджера проєкту.	Низька	Середній	0,04	Прийняття	Зберігати баланс відпочинку та роботи протягом проєкту, зменшення темпу робіт	Запит до керівництва проєкту стосовно подовження строку проєкту
R	Н	Вив	Н	Сер	0	Змен	Регулярне	Використання

S_5	овий	едення ПК розробника з ладу.	изька	едній	,08	шення	обслуговування обладнання виконавців проєкту та використання віддалених репозиторіїв для роботи з файлами проєкту	ПК на базі університету для продовження проєкту
R S_6	Н овий	Заве ликий об'єм моделі та як наслідок – потреба в високих системних параметра	Се редня	Сер едній	0 ,12	Прий няття	Збільшення загальних системних вимог до ПК користувачів	

		х ПК.						
R S_7	Новий	Витік даних для тренування та тестування нейромережі	Низька	Середній	0,08	Адаптація	Проведення усіх обчислень у ізольованому середовищі, шифрування даних при роботі з ними, також для всіх виконавців – поліпшення безпеки облікових записів шляхом додавання двохфакторної аутентифікації, тощо.	
R S_8	Новий	Неправильне використання	Низька	Середній	0,08	Адаптація	Шифрування даних при роботі з ними	

		конфіденційних даних під час роботи додатку						
RS_9	Новий	Поява ефективнішого аналогу	Низька	Середній	0,08	Адаптація	Постійний аналіз наукових досягнень у сфері ШІ та випробовування нових методів і систем ШІ	Аналіз аналогу та покриття його сильних та слабких сторін за рахунок зміни або покращення існуючої моделі розроблюваного програмного продукту
RS_10	Новий	Втрата актуально	Низький	Середній	0,08	Прийняття	Постійний аналіз наукових досягнень у сфері ШІ	

		сті відносно ринку ШІ					та випробовування нових методів і систем ШІ	

ДОДАТОК В

Лістинг основних модулів

Лістинг класу Main.java

```
package ua.edu.sumdu.lstm.core.gencon;

import lombok.SneakyThrows;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.deeplearning4j.datasets.iterator.utilty.ListDataSetIterator;
import org.deeplearning4j.nn.api.OptimizationAlgorithm;
import org.deeplearning4j.nn.conf.BackpropType;
import org.deeplearning4j.nn.conf.NeuralNetConfiguration;
import org.deeplearning4j.nn.conf.layers.DenseLayer;
import org.deeplearning4j.nn.conf.layers.DropoutLayer;
import org.deeplearning4j.nn.conf.layers.LSTM;
import org.deeplearning4j.nn.conf.layers.RnnOutputLayer;
import org.deeplearning4j.nn.multilayer.MultiLayerNetwork;
import org.deeplearning4j.nn.weights.WeightInit;
import org.deeplearning4j.util.ModelSerializer;
import org.nd4j.evaluation.classification.ROC;
import org.nd4j.evaluation.regression.RegressionEvaluation;
import org.nd4j.linalg.activations.Activation;
import org.nd4j.linalg.api.ndarray.INDArray;
import org.nd4j.linalg.dataset.DataSet;
import org.nd4j.linalg.dataset.api.preprocessor.DataNormalization;
import org.nd4j.linalg.dataset.api.preprocessor.NormalizerMinMaxScaler;
import org.nd4j.linalg.dataset.api.preprocessor.NormalizerStandardize;
import org.nd4j.linalg.dataset.api.preprocessor.serializer.NormalizerSerializer;
import org.nd4j.linalg.dataset.api.preprocessor.serializer.StandardizeSerializerStrategy;
import org.nd4j.linalg.factory.Nd4j;
import org.nd4j.linalg.learning.config.Adam;
import org.nd4j.linalg.lossfunctions.LossFunctions;

import java.io.FileInputStream;
```

```
import java.io.IOException;
import java.time.LocalDate;
import java.util.Arrays;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;

public class Main {

    public static final int FIRST_YEAR = 2019;
//    public static final int lookback = 14;
    public static final double DATA_PERCENTAGE_TO_TRAIN = .5;
    public static final int ROWS_TO_SKIP = 3;
//    public static final boolean withWeather = false;

    public static void main(String[] args) {
        final boolean[] withWeather = {true, false};
        final int[] lookbacks = {1, 2, 3, 5, 7, 14};

        for (int i = 0; i < 2; i++) {

            for (int j = 0; j < 6; j++) {
                trainConsumptionModel(withWeather[i], lookbacks[j]);
                trainGenerationModel(withWeather[i], lookbacks[j]);
            }
        }
    }

    @SneakyThrows
    public static MultiLayerNetwork trainConsumptionModel(boolean withWeather, int lookback) {
        MultiLayerNetwork consumptionModel = buildModel(withWeather, lookback);

        List<InfoPiece> loadedData = new LinkedList<>();

        try (FileInputStream file = new FileInputStream("diploma.xlsx");
             Workbook workbook = new XSSFWorkbook(file)) {

            Iterator<Row> rowIterator;
            for (int year = FIRST_YEAR; year <= 2023; year++) {
                Sheet sheet = workbook.getSheet(String.valueOf(year));
```

```
        rowIterator = sheet.rowIterator();

        LocalDate previousDate = null;

        for (int i = 0; rowIterator.hasNext(); i++) {
            if (i < ROWS_TO_SKIP) {
                rowIterator.next();
                continue;
            }

            Row row = rowIterator.next();

            Cell dateCell = row.getCell(0);

            System.out.println(i);
            if (isValidForThisYear(year, dateCell)) {
                LocalDate date = dateCell.getLocalDateTimeCellValue().toLocalDate();

                if (previousDate != null && date.isEqual(previousDate)) {
                    continue;
                } else {
                    previousDate = date;
                }

                double generation = row.getCell(12).getNumericCellValue() * 1000;
                double consumption = row.getCell(16).getNumericCellValue();
                String temp6 = row.getCell(18).getStringCellValue();
                String temp18 = row.getCell(17).getStringCellValue();

                InfoPiece infoPiece = new InfoPiece(date, generation, consumption, temp6, temp18);

                loadedData.add(infoPiece);
            } else {
                break;
            }
        }
    }

    loadedData.forEach(System.out::println);
```

```
        final int dataToTrain = (int)(loadedData.size() * DATA_PERCENTAGE_TO_TRAIN) - (int)
(loadedData.size() * DATA_PERCENTAGE_TO_TRAIN) % lookback;
        final int dataBatchesToTrain = dataToTrain / lookback;

        double[] consumptionData = new double[dataToTrain];
        double[] temp6Data = new double[dataToTrain];
        double[] temp18Data = new double[dataToTrain];
        double[] generationData = new double[dataToTrain];

        Iterator<InfoPiece> infoPieceIterator = loadedData.listIterator();
        for (int i = 0; i < dataToTrain; i++) {
            InfoPiece currentInfoPiece = infoPieceIterator.next();

            consumptionData[i] = currentInfoPiece.consumption;
            temp6Data[i] = currentInfoPiece.temp6;
            temp18Data[i] = currentInfoPiece.temp18;
            generationData[i] = currentInfoPiece.generation;
        }

        List<DataSet> trainData = new LinkedList<>();
        List<DataSet> testData = new LinkedList<>();

        for (int i = 0; i < dataBatchesToTrain - 2; i++) {
            INDArray consumptionFeatures = Nd4j.create(Arrays.copyOfRange(consumptionData, i * lookback,
(i + 1) * lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp6Features = Nd4j.create(Arrays.copyOfRange(temp6Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp18Features = Nd4j.create(Arrays.copyOfRange(temp18Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray labels = Nd4j.create(Arrays.copyOfRange(consumptionData, (i + 1) * lookback, (i + 2) *
lookback))
                    .reshape(1, 1, lookback);

            INDArray features;
```

```
            if (withWeather) {
                features = Nd4j.hstack(consumptionFeatures, temp6Features, temp18Features)
                    .reshape(1, 3, lookback);
            } else {
                features = consumptionFeatures.reshape(1, 1, lookback);
            }

            trainData.add(new DataSet(features, labels));
        }
        for (int i = dataBatchesToTrain - 2; i < dataBatchesToTrain - 1; i++) {
            INDArray consumptionFeatures = Nd4j.create(Arrays.copyOfRange(consumptionData, i * lookback,
(i + 1) * lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp6Features = Nd4j.create(Arrays.copyOfRange(temp6Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp18Features = Nd4j.create(Arrays.copyOfRange(temp18Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray labels = Nd4j.create(Arrays.copyOfRange(consumptionData, (i + 1) * lookback, (i + 2) *
lookback))
                    .reshape(1, 1, lookback);

            INDArray features;

            if (withWeather) {
                features = Nd4j.hstack(consumptionFeatures, temp6Features, temp18Features)
                        .reshape(1, 3, lookback);
            } else {
                features = consumptionFeatures.reshape(1, 1, lookback);
            }

            testData.add(new DataSet(features, labels));
        }

        NormalizerSerializer serializer = new NormalizerSerializer();

        serializer.addStrategy(new StandardizeSerializerStrategy());

        DataNormalization normalizer = new NormalizerStandardize();
```

```
normalizer.fitLabel(true);

ListDataSetIterator<DataSet> trainIterator = new ListDataSetIterator<>(trainData, 28);
ListDataSetIterator<DataSet> testIterator = new ListDataSetIterator<>(testData, 1);

normalizer.fit(trainIterator);
trainIterator.setPreProcessor(normalizer);
testIterator.setPreProcessor(normalizer);

consumptionModel.init();

int epochs = 30;
for (int i = 0; i < epochs; i++) {
    System.out.println(Math.round(i * 1. / epochs * 100 * 100) / 100 + "%");

    consumptionModel.fit(trainIterator);
}


serializer.write(normalizer, lookback + (withWeather ? "W": "NW")  + "CModelSer.dat");
normalizer = serializer.restore(lookback + (withWeather ? "W": "NW")  + "CModelSer.dat");

System.out.println("100%");

RegressionEvaluation evaluation = new RegressionEvaluation(1);
var roc = new ROC(100);

double[] predictedArray = new double[1 * lookback];
double[] actualArray = new double[1 * lookback];

while (testIterator.hasNext()) {
    DataSet dataSet = testIterator.next();

    var predicted = consumptionModel.output(dataSet.getFeatures());
    var actual = dataSet.getLabels();

    normalizer.revertLabels(predicted);
    normalizer.revertLabels(actual);
```

```
            for (int i = 0; i < predicted.length(); i++) {
                predictedArray[i] = predicted.getDouble(i);
                actualArray[i] = actual.getDouble(i);
            }

            roc.evalTimeSeries(actual, predicted);
            System.out.println("AUC: " + roc.calculateAUC());
            evaluation.eval(actual, predicted);
            System.out.println(predicted);
            System.out.println(actual);
        }

        System.out.println("Regression evaluation metrics:");
        for (RegressionEvaluation.Metric metric : RegressionEvaluation.Metric.values()) {
            System.out.println(metric + ": " + evaluation.scoreForMetric(metric));
        }

        calcMAPE(actualArray, predictedArray);
//          testData.forEach(System.out::println);
        ModelSerializer.writeModel(consumptionModel, lookback + (withWeather ? "W": "NW")  +
"CModel.dat", true);
        return consumptionModel;
    } catch (IOException e) {
        return null;
    }
  }

  @SneakyThrows
  public static MultiLayerNetwork trainGenerationModel(boolean withWeather, int lookback) {
    MultiLayerNetwork generationModel = buildModel(withWeather, lookback);

    List<InfoPiece> loadedData = new LinkedList<>();

    try (FileInputStream file = new FileInputStream("diploma.xlsx");
         Workbook workbook = new XSSFWorkbook(file)) {

        Iterator<Row> rowIterator;
        for (int year = FIRST_YEAR; year <= 2023; year++) {
            Sheet sheet = workbook.getSheet(String.valueOf(year));
```

```
        rowIterator = sheet.rowIterator();

        LocalDate previousDate = null;

        for (int i = 0; rowIterator.hasNext(); i++) {
            if (i < ROWS_TO_SKIP) {
                rowIterator.next();
                continue;
            }

            Row row = rowIterator.next();

            Cell dateCell = row.getCell(0);

            System.out.println(i);
            if (isValidForThisYear(year, dateCell)) {
                LocalDate date = dateCell.getLocalDateTimeCellValue().toLocalDate();

                if (previousDate != null && date.isEqual(previousDate)) {
                    continue;
                } else {
                    previousDate = date;
                }

                double generation = row.getCell(12).getNumericCellValue() * 1000;
                double consumption = row.getCell(16).getNumericCellValue();
                String temp6 = row.getCell(18).getStringCellValue();
                String temp18 = row.getCell(17).getStringCellValue();

                InfoPiece infoPiece = new InfoPiece(date, generation, consumption, temp6, temp18);

                loadedData.add(infoPiece);
            } else {
                break;
            }
        }
    }

    loadedData.forEach(System.out::println);
```

```
        final int dataToTrain = (int)(loadedData.size() * DATA_PERCENTAGE_TO_TRAIN) - (int)
(loadedData.size() * DATA_PERCENTAGE_TO_TRAIN) % lookback;
        final int dataBatchesToTrain = dataToTrain / lookback;

        double[] consumptionData = new double[dataToTrain];
        double[] temp6Data = new double[dataToTrain];
        double[] temp18Data = new double[dataToTrain];
        double[] generationData = new double[dataToTrain];

        Iterator<InfoPiece> infoPieceIterator = loadedData.listIterator();
        for (int i = 0; i < dataToTrain; i++) {
            InfoPiece currentInfoPiece = infoPieceIterator.next();

            consumptionData[i] = currentInfoPiece.consumption;
            temp6Data[i] = currentInfoPiece.temp6;
            temp18Data[i] = currentInfoPiece.temp18;
            generationData[i] = currentInfoPiece.generation;
        }

        List<DataSet> trainData = new LinkedList<>();
        List<DataSet> testData = new LinkedList<>();

        for (int i = 0; i < dataBatchesToTrain - 2; i++) {
            INDArray generationFeatures = Nd4j.create(Arrays.copyOfRange(generationData, i * lookback, (i +
1) * lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp6Features = Nd4j.create(Arrays.copyOfRange(temp6Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray temp18Features = Nd4j.create(Arrays.copyOfRange(temp18Data, i * lookback, (i + 1) *
lookback));
//                    .reshape(1, 1, lookback);
            INDArray labels = Nd4j.create(Arrays.copyOfRange(generationData, (i + 1) * lookback, (i + 2) *
lookback))
                    .reshape(1, 1, lookback);

            INDArray features;

            if (withWeather) {
```

```
                features = Nd4j.hstack(generationFeatures, temp6Features, temp18Features)
                        .reshape(1, 3, lookback);
                } else {
                features = generationFeatures.reshape(1, 1, lookback);
                }

                trainData.add(new DataSet(features, labels));
            }
            for (int i = dataBatchesToTrain - 2; i < dataBatchesToTrain - 1; i++) {
                INDArray generationFeatures = Nd4j.create(Arrays.copyOfRange(generationData, i * lookback, (i +
1) * lookback));
//                      .reshape(1, 1, lookback);
                INDArray temp6Features = Nd4j.create(Arrays.copyOfRange(temp6Data, i * lookback, (i + 1) *
lookback));
//                      .reshape(1, 1, lookback);
                INDArray temp18Features = Nd4j.create(Arrays.copyOfRange(temp18Data, i * lookback, (i + 1) *
lookback));
//                      .reshape(1, 1, lookback);
                INDArray labels = Nd4j.create(Arrays.copyOfRange(generationData, (i + 1) * lookback, (i + 2) *
lookback))
                        .reshape(1, 1, lookback);

                INDArray features;

                if (withWeather) {
                features = Nd4j.hstack(generationFeatures, temp6Features, temp18Features)
                            .reshape(1, 3, lookback);
                } else {
                features = generationFeatures.reshape(1, 1, lookback);
                }

                testData.add(new DataSet(features, labels));
            }

            NormalizerSerializer serializer = new NormalizerSerializer();

            serializer.addStrategy(new StandardizeSerializerStrategy());

            DataNormalization normalizer = new NormalizerStandardize();
```

```
normalizer.fitLabel(true);

ListDataSetIterator<DataSet> trainIterator = new ListDataSetIterator<>(trainData, 28);
ListDataSetIterator<DataSet> testIterator = new ListDataSetIterator<>(testData, 1);

normalizer.fit(trainIterator);
trainIterator.setPreProcessor(normalizer);
testIterator.setPreProcessor(normalizer);

generationModel.init();

int epochs = 30;
for (int i = 0; i < epochs; i++) {
    System.out.println(Math.round(i * 1. / epochs * 100 * 100) / 100 + "%");

    generationModel.fit(trainIterator);
}

System.out.println("100%");

serializer.write(normalizer, lookback + (withWeather ? "W": "NW")  + "GModelSer.dat");
normalizer = serializer.restore(lookback + (withWeather ? "W": "NW")  + "GModelSer.dat");

RegressionEvaluation evaluation = new RegressionEvaluation(1);
var roc = new ROC(100);

double[] predictedArray = new double[1 * lookback];
double[] actualArray = new double[1 * lookback];

while (testIterator.hasNext()) {
    DataSet dataSet = testIterator.next();

    var predicted = generationModel.output(dataSet.getFeatures());
    var actual = dataSet.getLabels();

    normalizer.revertLabels(predicted);
    normalizer.revertLabels(actual);

    for (int i = 0; i < predicted.length(); i++) {
        predictedArray[i] = predicted.getDouble(i);
```

```
                actualArray[i] = actual.getDouble(i);
            }

            roc.evalTimeSeries(actual, predicted);
            System.out.println("AUC: " + roc.calculateAUC());
            evaluation.eval(actual, predicted);
            System.out.println(predicted);
            System.out.println(actual);
        }

        System.out.println("Regression evaluation metrics:");
        for (RegressionEvaluation.Metric metric : RegressionEvaluation.Metric.values()) {
            System.out.println(metric + ": " + evaluation.scoreForMetric(metric));
        }

        calcMAPE(actualArray, predictedArray);
//          testData.forEach(System.out::println);

        ModelSerializer.writeModel(generationModel, lookback + (withWeather ? "W": "NW")  +
"GModel.dat", true);
        return generationModel;
    } catch (IOException e) {
        return null;
    }
  }

  private static void calcMAPE(double[] actual, double[] predicted) {
    double sum = 0;

    for (int i = 0; i < actual.length; i++) {
        sum += Math.abs((actual[i] - predicted[i]) / actual[i]) * 100;
    }

    double error = sum / actual.length;

    double accuracy = 100 - error;

    System.out.println("MAPE: " + error + "%");
    System.out.println("Accuracy: " + accuracy + "%");
  }
```

```
    private static MultiLayerNetwork buildModel(boolean withWeather, int lookback) {
        return new MultiLayerNetwork(new NeuralNetConfiguration.Builder()
//                .seed(123)
                .optimizationAlgo(OptimizationAlgorithm.STOCHASTIC_GRADIENT_DESCENT)
                .updater(new Adam(0.001))
                .list()
                .layer(new LSTM.Builder()
                        .weightInit(WeightInit.XAVIER)
                        .activation(Activation.TANH)
                        .gateActivationFunction(Activation.SIGMOID)
                        .nIn(withWeather ? 3 : 1)
                        .nOut(7)
                        .build())
                .layer(new DropoutLayer(.85))
                .layer(new DenseLayer.Builder()
                        .weightInit(WeightInit.XAVIER)
                        .activation(Activation.TANH)
                        .nIn(21)
                        .nOut(21)
                        .build())
                .layer(new RnnOutputLayer.Builder(LossFunctions.LossFunction.MEAN_ABSOLUTE_ERROR)
                        .activation(Activation.IDENTITY)
                        .nIn(21)
                        .nOut(1)
                        .build())
                .backpropType(BackpropType.TruncatedBPTT)
                .tBPTTBackwardLength(7)
                .validateOutputLayerConfig(false)
                .build());
    }

    public static boolean isValidForThisYear(int year, Cell cell) {
        return cell != null && cell.getLocalDateTimeCellValue() != null &&
cell.getLocalDateTimeCellValue().getYear() == year;
    }

    public static class InfoPiece {
        public final LocalDate date;
        public final Double generation;
```

```
    public final Double consumption;
    public final Double temp6;
    public final Double temp18;

    public InfoPiece(LocalDate date, Double generation, Double consumption, String temp6, String temp18) {
        this.date = date;
        this.generation = generation;
        this.consumption = consumption;
        this.temp6 = Double.valueOf(replaceTempRelatedChars(temp6));
        this.temp18 = Double.valueOf(replaceTempRelatedChars(temp18));
    }

    private static String replaceTempRelatedChars(String rawTemp) {
        String result = rawTemp.replaceAll("C", "")
                .replaceAll("C", "")
                .replaceAll("\\.", "")
                .replaceAll(" ", "")
                .replaceAll(",", ".");

        return result;
    }

    @Override
    public String toString() {
        return "InfoPiece{" +
                "date=" + date +
                ", generation=" + generation +
                ", consumption=" + consumption +
                ", temp6=" + temp6 +
                ", temp18=" + temp18 +
                '}';
    }
  }
}
```

Лістинг класу UI.java

```
package ua.edu.sumdu.lstm.core.gencon;

import lombok.SneakyThrows;
```

```
import org.apache.commons.io.filefilter.DirectoryFileFilter;
import org.apache.commons.io.filefilter.RegexFileFilter;
import org.apache.commons.io.filefilter.SuffixFileFilter;
import org.apache.commons.io.filefilter.TrueFileFilter;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.deeplearning4j.nn.multilayer.MultiLayerNetwork;
import org.deeplearning4j.util.ModelSerializer;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.xy.DefaultXYDataset;
import org.jfree.data.xy.XYDataItem;
import org.jfree.data.xy.XYDataset;
import org.jfree.data.xy.XYSeries;
import org.nd4j.linalg.api.ndarray.INDArray;
import org.nd4j.linalg.dataset.api.preprocessor.DataNormalization;
import org.nd4j.linalg.dataset.api.preprocessor.serializer.NormalizerSerializer;
import org.nd4j.linalg.factory.Nd4j;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
import java.util.stream.IntStream;
```

```
import static ua.edu.sumdu.lstm.core.gencon.Main.isValidForThisYear;

public class UI extends JFrame implements ActionListener {

    private static final String HOME_DIR = System.getProperty("user.dir");

    private JLabel titleLabel, predictDataLabel, saveFolderLabel;
    private JTextField predictDataFileTextField;
    private static JTextField saveFolderPathTextField;
    private JButton predictButton;
    private JButton predictDataFileButton, saveFolderButton;
    private FileDialog predictFileDialog, saveFolderDialog;

    public UI() {
        super("Electricity Supply Prediction");

        // Create labels
        titleLabel = new JLabel("Electricity Supply Prediction", JLabel.CENTER);
        predictDataLabel = new JLabel("Predict Data:");
        saveFolderLabel = new JLabel("Save Folder:");

        // Create text fields
        predictDataFileTextField = new JTextField(20);
        predictDataFileTextField.setEnabled(false);
        saveFolderPathTextField = new JTextField(20);
        saveFolderPathTextField.setEnabled(false);

        // Create buttons
        predictButton = new JButton("Show Prediction Results");
        predictDataFileButton = new JButton("...");
        saveFolderButton = new JButton("...");

        // Create file dialogs
        predictFileDialog = new FileDialog(this, "Choose Historical Data for Prediction");
        predictFileDialog.setFilenameFilter(new SuffixFileFilter(".xlsx"));
        saveFolderDialog = new FileDialog(this, "Choose Save Folder");
        saveFolderDialog.setFilenameFilter(DirectoryFileFilter.DIRECTORY);

        // Set layout
```

```
setLayout(new GridBagLayout());
GridBagConstraints c = new GridBagConstraints();

// Title label
c.fill = GridBagConstraints.HORIZONTAL;
c.gridwidth = 3;
c.gridx = 0;
c.gridy = 0;
c.insets = new Insets(10, 10, 10, 10);
add(titleLabel, c);

// Predict data label
c.fill = GridBagConstraints.HORIZONTAL;
c.gridwidth = 1;
c.gridx = 0;
c.gridy = 2;
c.insets = new Insets(5, 5, 5, 5);
add(predictDataLabel, c);

// Predict data text field
c.fill = GridBagConstraints.HORIZONTAL;
c.gridwidth = 1;
c.gridx = 1;
c.gridy = 2;
c.insets = new Insets(5, 5, 5, 5);
add(predictDataFileTextField, c);

// Predict data file button
c.fill = GridBagConstraints.NONE;
c.gridwidth = 1;
c.gridx = 2;
c.gridy = 2;
c.insets = new Insets(5, 5, 5, 5);
add(predictDataFileButton, c);

// Save folder label
c.fill = GridBagConstraints.HORIZONTAL;
c.gridwidth = 1;
c.gridx = 0;
c.gridy = 3;
```

```
        c.insets = new Insets(5, 5, 5, 5);
        add(saveFolderLabel, c);

        // Save folder text field
        c.fill = GridBagConstraints.HORIZONTAL;
        c.gridwidth = 1;  // Spans two columns
        c.gridx = 1;
        c.gridy = 3;
        c.insets = new Insets(5, 5, 5, 5);
        add(saveFolderPathTextField, c);

        // Save folder button
        c.fill = GridBagConstraints.NONE;
        c.gridwidth = 1;
        c.gridx = 2;
        c.gridy = 3;
        c.insets = new Insets(5, 5, 5, 5);
        add(saveFolderButton, c);

        // Predict button
        c.fill = GridBagConstraints.HORIZONTAL;
        c.gridwidth = 4;
        c.gridx = 0;
        c.gridy = 4;
        c.insets = new Insets(5, 5, 5, 5);
        add(predictButton, c);

        // Add action listeners
        predictDataFileButton.addActionListener(this);
        predictButton.addActionListener(this);
        saveFolderButton.addActionListener(this);  // Add listener for save folder button

        // Set window properties
        setSize(500, 250);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
```

```
    if (e.getSource() == predictDataFileButton) {
        // Open predict data file dialog
        predictFileDialog.setVisible(true);
        String selectedFile = predictFileDialog.getDirectory() + predictFileDialog.getFile();
        if (selectedFile != null) {
            predictDataFileTextField.setText(selectedFile);

            System.out.println("File: " + predictFileDialog.getFile());
            System.out.println("Directory: " + predictFileDialog.getDirectory());
        }


    } else if (e.getSource() == predictButton) {
        if (!predictDataFileTextField.getText().isBlank()) {
            // Functionality to handle predict button click
            // You can add logic to access the predict data path from predictDataFileTextField
            // and perform prediction operations here
            System.out.println("Predict button clicked! Predict data path: " + predictDataFileTextField.getText());
            System.setProperty("user.dir", predictFileDialog.getDirectory());
            createPlotWindow(predictFileDialog.getFile());
        }
    } else if (e.getSource() == saveFolderButton) {
        // Open save folder dialog
        saveFolderDialog.setVisible(true);
        String selectedFolder = saveFolderDialog.getDirectory();  // Get directory path
        if (selectedFolder != null) {
            saveFolderPathTextField.setText(selectedFolder);
        }
    }
}

public static void main(String[] args) {
    new UI();
}

@SneakyThrows
private static void createPlotWindow(String dataLoadPath) {
    // Create a Swing frame for the pop-up window
    JFrame plotFrame = new JFrame("Prediction Results");
    plotFrame.setSize(720, 480);
```

```
plotFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE); // Close on X

// Create a panel to hold the plot (Swing)
JPanel plotPanel = new JPanel();
plotFrame.getContentPane().add(plotPanel);

List<Main.InfoPiece> loadedData = new LinkedList<>();

try (FileInputStream file = new FileInputStream(dataLoadPath);
     Workbook workbook = new XSSFWorkbook(file)) {

    Iterator<Row> rowIterator;
    Sheet sheet = workbook.getSheetAt(0);

    rowIterator = sheet.rowIterator();

    for (int i = 0; rowIterator.hasNext(); i++) {
        if (i < 1) {
            rowIterator.next();
            continue;
        }

        Row row = rowIterator.next();



        System.out.println(i);
        if (row.getCell(0) != null) {
            LocalDate date = null;

            double generation = row.getCell(0).getNumericCellValue();
            double consumption = row.getCell(1).getNumericCellValue();

            String temp6 = String.valueOf(row.getCell(2).getNumericCellValue());
            String temp18 = String.valueOf(row.getCell(3).getNumericCellValue());

            Main.InfoPiece infoPiece = new Main.InfoPiece(date, generation, consumption, temp6, temp18);

            loadedData.add(infoPiece);
        } else {
            break;
```

```
            }

        }

        Iterator<Main.InfoPiece> infoPieceIterator = loadedData.iterator();

        double[] consumptionData = new double[14];
        double[] generationData = new double[14];
        double[] temp6Data = new double[14];
        double[] temp18Data = new double[14];

        for (int i = 0; i < 14; i++) {
            Main.InfoPiece current = infoPieceIterator.next();

            consumptionData[i] = current.consumption;
            generationData[i] = current.generation;
            temp6Data[i] = current.temp6;
            temp18Data[i] = current.temp18;
        }

        System.setProperty("user.dir", HOME_DIR);
        MultiLayerNetwork consumptionModel = loadModel("14WCModel.dat");
        DataNormalization consumptionNormalization = loadNormalization("14WCModelSer.dat");

        INDArray toPredictC = Nd4j.hstack(Nd4j.create(consumptionData), Nd4j.create(temp6Data),
Nd4j.create(temp18Data)).reshape(1, 3, 14);

        consumptionNormalization.transform(toPredictC);

        INDArray predictedConsumption = consumptionModel.output(toPredictC);

        consumptionNormalization.revertLabels(predictedConsumption);

        MultiLayerNetwork generationModel = loadModel("14WGModel.dat");
        DataNormalization generationNormalization = loadNormalization("14WGModelSer.dat");

        INDArray toPredictG = Nd4j.hstack(Nd4j.create(generationData), Nd4j.create(temp6Data),
Nd4j.create(temp18Data)).reshape(1, 3, 14);

        generationNormalization.transform(toPredictG);
```

```
        INDArray predictedGeneration = generationModel.output(toPredictC);

        generationNormalization.revertLabels(predictedGeneration);

        // Generate some sample data (replace with your data source)
        double[] xDataC = IntStream.range(1, 15).asDoubleStream().toArray();
        double[] yDataC = predictedConsumption.reshape(14).toDoubleVector();
        double[] xDataG = IntStream.range(1, 15).asDoubleStream().toArray();
        double[] yDataG = predictedGeneration.reshape(14).toDoubleVector();

        // Create a basic line chart (using JFreeChart)
        LineChart plot = new LineChart(xDataC, yDataC, xDataG, yDataG);
        plotPanel.add(plot.getChartPanel());

        // Display the pop-up window
        plotFrame.setVisible(true);
    }
}

static class LineChart implements ActionListener {



    private final double[] xDataC;
    private final double[] yDataC;
    private final double[] xDataG;
    private final double[] yDataG;
    private final JFreeChart chart;
    private final JButton export;

    public LineChart(double[] xDataC, double[] yDataC, double[] xDataG, double[] yDataG) {

        this.xDataC = xDataC;
        this.yDataC = yDataC;
        this.xDataG = xDataG;
        this.yDataG = yDataG;

        // Create the dataset and add the data items
        DefaultXYDataset dataset = new DefaultXYDataset();
```

```
    dataset.addSeries("Consumption", new double[][]{xDataC, yDataC}); // Provide a series name (optional)
    dataset.addSeries("Generation", new double[][]{xDataG, yDataG}); // Provide a series name (optional)

    chart = ChartFactory.createXYLineChart(
            "Predicted Data", "X-Axis", "Y-Axis", dataset, PlotOrientation.VERTICAL,
            true, true, false);

    export = new JButton("Export as XLSX");

    export.setBounds(0, 380, 200, 40);

    export.addActionListener(this);
}

public JPanel getChartPanel() {
    ChartPanel chartPanel = new ChartPanel(chart);

    chartPanel.setLayout(null);
    chartPanel.add(export);

    export.setAlignmentX(0);

    return chartPanel;
}

@SneakyThrows
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == export) {
        String[] header = new String[] {"Consumption, Wh", "Generation, Wh"};
        double[][] data = new double[][] {yDataC, yDataG};

        XSSFWorkbook workbook = new XSSFWorkbook();
        XSSFSheet sheet = workbook.createSheet();
        XSSFRow headerRow = sheet.createRow(0);

        for (int i = 0; i < header.length; i++) {
            XSSFCell cell = headerRow.createCell(i);
            cell.setCellValue(header[i]);
```

```
            }

            for (int i = 1; i <= data[0].length; i++) {
                XSSFRow dataRow = sheet.createRow(i);
                for (int j = 0; j < data.length; j++) {
                    XSSFCell cell = dataRow.createCell(j);
                    cell.setCellValue(data[j][i - 1]);
                }
            }

            System.setProperty("user.dir", saveFolderPathTextField.getText());
            FileOutputStream os = new FileOutputStream("export" + Instant.now().toEpochMilli() + ".xlsx");
            workbook.write(os);
            os.close();
            System.out.println("Exported");
        }
    }
}
@SneakyThrows
static MultiLayerNetwork loadModel(String path) {
    MultiLayerNetwork model = ModelSerializer.restoreMultiLayerNetwork(path);

    return model;
}
@SneakyThrows
static DataNormalization loadNormalization(String path) {
    return NormalizerSerializer.getDefault().restore(path);
}
}
```

ДОДАТОК Г

Тези на наукову конференцію IMA-2024

Prediction of electricity supply of a residential house from renewable energy sources using neural networks

M. Bohachov, *student IT-01*; Yu. Parfenenko, *Associate Professor*

Sumy State University, Sumy, Ukraine

Nowadays, rapidly developing technologies tend to increase the consumption of electricity by the population and industry. However, this contributes to the complexity of energy systems, making the planning of resources sophisticated, especially having renewable energy sources in the power grid. So, the relevance of developing the neural network for predicting household energy consumption is in suggesting future power grid behavior and forecasting power consumption.

The purpose of the study is to facilitate resource management and electricity network optimization. House owners can obtain predictions about electricity consumption for the next 24 hours, providing information about power usage for the past 24 hours with an average error rate of 6.4%. Users should input a list of consumption records directly to the application or specify the file (XSLX or CSV) with such data. Also, there is variation between possible prediction periods, but the longer the predicted period, the higher the error rate amongst predictions and the length required of previous consumption data. After a series of experiments on configuration, we can assume that the 24-hour long period is the most efficient one.

The neural network application for predicting household power consumption is design as cross-platform because of the usage of Java Development Kit alongside such tools as Lombok, Apache Poi, DL4j, and ND4j. Due to its high efficiency in working with time-series sequences, the neural network type used in the application is LSTM, a subtype of Recurrent Neural Network. The cross-platform nature of JDK makes the developing application run on Windows, Linux, and Mac machines, but also Android-based systems. However, supporting this application to run on Android-based systems requires additional efforts on a separate user interface shell.

Therefore, the developed application is a valuable tool in resource planning for households and getting a better understanding of the operational dynamics of the power grid. It provides an efficient way to forecast the power usage of a household within a 24-hour long period with 93.6% accuracy.