СИСТЕМНИЙ АНАЛІЗ

SYSTEM ANALYSIS

УДК 004.6, 004.8, 004.942, 004.021: 519.6
DOI https://doi.org/10.32782/tnv-tech.2024.5.13

ПРОЄКТУВАННЯ ТА НАВЧАННЯ МОДЕЛІ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ПЛАНУВАННЯ Й ОЦІНКИ РИЗИКІВ ПРОЄКТІВ

Михайлов Н. О. *– аспірант*
Київського національного університету імені Тараса Шевченка
ORCID ID: 0009-0002-9374-3403

У сучасному світі, де управління проєктами стає дедалі складнішим і багатогранніш им, традиційні підходи проектного менеджменту втрачають свою актуальність. Ефективне управління ресурсами, прогнозування строків виконання завдань і оцінка потенційних ризиків є запорукою успіху будь-якого проєкту. У цих умовах технології штучного інтелекту, набувають усе більшої популярності, завдяки їхній здатності аналізувати великі обсяги даних, ідентифікувати приховані закономірності та робити точні прогнози.

Ця стаття присвячена розробці та навчанню моделі штучного інтелекту, яка здатна автоматизувати процес планування проєктів і оцінки ризиків. У межах роботи детально розглянуто весь цикл створення такої моделі: від налаштування середовища до побудови нейронної мережі та її навчання. Особливу увагу було приділено підготовці даних для моделі, вибору оптимальної архітектури нейронної мережі та експериментальній оцінці її ефективності на реальних даних.

Модель, описана в статті, орієнтована на вирішення двох основних задач. Перша – це прогнозування строків виконання завдань, що дозволяє оптимізувати розподіл ресурсів та уникати затримок. Друга задача – це автоматизована оцінка ризиків, яка допомагає заздалегідь ідентифікувати потенційні проблеми та запропонувати заходи для їх мінімізації. Завдяки використанню алгоритмів глибокого навчання модель може обробляти дані з багатьох джерел, таких як системи управління проєктами, і адаптуватися до нових умов. Ключовими аспектами створення моделі є її гнучкість і адаптивність. Навіть за умов обмеженого доступу до даних, модель демонструє здатність до навчання і прогнозування. Водночас, із розширенням обсягів даних її продуктивність і точність лише покращуються. Модель здатна автоматично оновлювати прогнози та коригувати планування у реальному часі, що значно підвищує ефективність управління великими та складними проєктами.

Ключові слова: *планування проєктів, оцінка ризиків, штучний інтелект, машинне навчання, нейронні мережі, прогнозування тривалості, оптимізація ресурсів, аналіз даних, алгоритми прогнозування, автоматизація управління проєктами, управління ризиками, інтелектуальні системи, адаптивне планування, навчання моделей, рекурентні нейронні мережі, нормалізація даних, підготовка даних для моделі.*

Mykhailov N. O. Designing and training an artificial intelligence model for project planning and risk assessment

In the modern world, where project management is becoming increasingly complex and multifaceted, traditional approaches to project management are losing their relevance. Effective

resource management, accurate task duration forecasting, and risk assessment are crucial for the success of any project. In this context, artificial intelligence technologies are gaining increasing popularity due to their ability to analyze large volumes of data, identify hidden patterns, and make accurate predictions.

This article focuses on the development and training of an artificial intelligence model capable of automating project planning and risk assessment processes. The work examines the entire cycle of creating such a model: from setting up the environment to building and training a neural network. Special attention is paid to data preparation for the model, selecting the optimal neural network architecture, and experimentally evaluating its effectiveness on real-world data. The model described in the article addresses two main tasks. The first is predicting task completion durations, which allows for resource optimization and the avoidance of delays. The second task is automated risk assessment, which helps to identify potential issues in advance and propose measures to mitigate them. By utilizing deep learning algorithms, the model can process data from multiple sources, such as project management systems, and adapt to new conditions. Key aspects of the model's design include its flexibility and adaptability. Even with limited access to data, the model demonstrates the ability to learn and make accurate predictions. Moreover, as the volume of data increases, its performance and accuracy improve further. The model can automatically update predictions and adjust planning in real-time, significantly enhancing the management of large and complex projects.

Key words: *project planning, risk assessment, artificial intelligence, machine learning, neural networks, task duration forecasting, resource optimization, data analysis, predictive algorithms, project management automation, risk management, intelligent systems, adaptive planning, model training, recurrent neural networks, data normalization, data preparation for modeling.*

Вступ. У сучасних умовах глобалізації та цифровізації, управління проєктами постає як складний і багатогранний процес, що потребує швидких, точних і адаптивних рішень. Традиційні методи планування та оцінки ризиків, такі як ручний аналіз даних та використання статичних моделей, часто не відповідають сучасним викликам, зокрема зростанню обсягів даних, швидкій зміні умов і необхідності враховувати численні взаємозалежні фактори. Успіх проєкту тепер значною мірою залежить від здатності ефективно управляти ресурсами, точно прогнозувати строки виконання завдань і вчасно ідентифікувати ризики, які можуть виникнути під час його реалізації.

У цьому контексті технології штучного інтелекту, зокрема нейронні мережі, стали потужним інструментом для автоматизації процесів управління проєктами. Вони здатні аналізувати великі обсяги даних, ідентифікувати складні закономірності, що не завжди є очевидними для людини, та надавати точні прогнози у реальному часі. Штучний інтелект не лише дозволяє зменшити витрати часу та ресурсів на планування, але й сприяє підвищенню точності прийняття рішень, що в кінцевому підсумку покращує результативність і стабільність виконання проєкту.

Налаштування середовища розробки. Налаштування середовища для розробки моделі штучного інтелекту є критично важливим етапом, який визначає ефективність і стабільність усіх подальших процесів. Воно має бути оптимізованим для роботи з великими обсягами даних і відповідати вимогам навчання нейромереж. Основною мовою програмування обрано Python завдяки її багатій екосистемі бібліотек, що забезпечують повний цикл розробки моделі. Для побудови нейронної мережі використовується TensorFlow, що дозволяє створювати складні архітектури та підтримувати апаратне прискорення за допомогою GPU. Високорівневий API Keras спрощує створення та налаштування архітектури, тоді як Pandas, NumPy і Scikit-learn використовуються для обробки даних, нормалізації та роботи з категорійними змінними [1, с. 129–133].

Апаратне забезпечення є одним із ключових факторів, що впливають на швидкість навчання моделі. Графічні процесори (GPU) дозволяють значно прискорити обчислення порівняно з центральними процесорами (CPU), тому їх використання

є обов'язковим для великих задач машинного навчання. У разі відсутності потужного локального обладнання застосовуються хмарні сервіси, такі як Google Colab, що надає безкоштовний доступ до GPU і TPU. Це особливо корисно для тестування та навчання моделей із помірним набором даних. Альтернативно використовуються комерційні хмарні сервіси, зокрема AWS та Azure, які пропонують ресурси для масштабованих обчислень. Перед початком роботи з GPU перевіряється коректність встановлення драйверів CUDA та cuDNN, необхідних для роботи з TensorFlow [2, с. 389–400].

Для організації ізольованого програмного середовища створюється віртуальне середовище Python. Воно забезпечує стабільність роботи та уникає конфліктів між залежностями різних проєктів. Пакети встановлюються за допомогою файлу requirements.txt, у якому перелічено всі необхідні бібліотеки. Наприклад, серед обов'язкових пакетів зазначаються TensorFlow, Pandas, NumPy, Scikit-learn і Matplotlib. Також можливе використання Anaconda, яка надає зручний інтерфейс для керування середовищами.

Для отримання реальних даних про завдання інтегрується система управління проєктами JIRA. За допомогою бібліотеки JIRA Python налагоджується автоматичний збір інформації про завдання, включаючи тривалість виконання, початкові оцінки, пріоритети та імена виконавців. Ці дані формують основу для навчання моделі, дозволяючи враховувати специфіку реальних проєктів. Автоматизація отримання даних із JIRA знижує витрати часу на ручний збір і забезпечує регулярне оновлення інформації [3, с. 295–300].

Перед основною роботою з моделлю проводиться тестування налаштованого середовища. Навчається проста модель для перевірки працездатності TensorFlow і роботи GPU. Окремо тестується стабільність з'єднання з JIRA API для гарантованого доступу до даних. Контроль версій здійснюється за допомогою Git, що дозволяє документувати зміни та забезпечує повторюваність процесу навчання. Таким чином, налаштоване середовище є основою для успішної реалізації моделі штучного інтелекту. Його правильна конфігурація мінімізує технічні ризики, забезпечує інтеграцію з реальними даними та створює зручну платформу для навчання й оптимізації моделі, яка буде використовуватися для планування проєктів і оцінки ризиків [4, с. 436–444].

Вибір набору даних. Для успішного створення моделі штучного інтелекту ключову роль відіграє якість і репрезентативність вибраного набору даних. Основним джерелом даних у цьому проєкті є система управління завданнями JIRA, яка надає детальну інформацію про виконання завдань у попередніх спринтах. Ці дані включають тривалість виконання завдань, їхні початкові оцінки, пріоритети, типи, а також інформацію про виконавців та залежності між завданнями. Такі параметри є дуже важливими для навчання моделі, оскільки вони впливають на прогнозування тривалості завдань і оцінку ризиків.

Отримання даних із JIRA виконується за допомогою бібліотеки JIRA Python, що дозволяє автоматизувати процес збору та отримувати необхідні метрики через API. Цей підхід забезпечує регулярне оновлення даних, що є необхідним для створення адаптивної моделі. Крім того, для тестування та перевірки моделі можуть використовуватися публічно доступні набори даних із платформ, таких як Kaggle або OpenML, які імітують реальні сценарії планування проєктів.

Для забезпечення якості даних проводиться їх попередня обробка. Видаляються пропущені значення, аномалії та некоректні записи. Дані нормалізуються для зведення всіх параметрів до одного масштабу, що є важливим для стабільного

навчання моделі. Категорійні дані, такі як імена виконавців або типи завдань, кодуються в числовий формат для подальшої обробки нейронною мережею [5, с. 88–92].

Ретельний вибір і підготовка даних гарантують, що модель працюватиме коректно навіть із відносно невеликими обсягами інформації. Інтеграція з реальними даними з JIRA дозволяє забезпечити точність прогнозів і зробити модель більш практичною для застосування в управлінні проєктами.

Підготовка даних для моделі. Ефективність роботи моделі штучного інтелекту залежить від правильності й ретельності підготовки даних. У цьому процесі ключовими етапами є очищення даних, нормалізація числових значень, перетворення категорійних даних та формування навчальних і тестових вибірок.

Першим етапом є очищення даних, яке включає виявлення та усунення пропущених значень, аномалій або некоректних записів. Наприклад, якщо дані про тривалість виконання завдань або початкову оцінку часу відсутні, такі записи видаляються або заповнюються середніми значеннями для відповідного типу завдань. Також видаляються аномалії, такі як нереалістично високі оцінки часу або дублікати завдань, що можуть спотворити результати навчання.

Для забезпечення ефективного навчання всі числові значення, такі як оцінки тривалості та фактичний час виконання завдань, нормалізуються. Це дозволяє звести всі значення до одного масштабу, що запобігає домінуванню параметрів із великими числовими значеннями. Нормалізація виконується за формулою:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}, \text{ де}$$

x – вихідне значення, x_{min} та x_{max} – мінімальне та максимальне значення у вибірці. Цей підхід дозволяє моделі ефективно працювати з різними типами даних і знижує ризик помилок під час навчання [6, с. 315–350].

Категорійні дані, такі як виконавці завдань, типи завдань ("Bug", "Feature", "Improvement") та їхні пріоритети, перетворюються в числовий формат. Для цього використовується метод кодування. Наприклад, Label Encoding присвоює кожній категорії унікальне числове значення, а One-Hot Encoding створює бінарні стовпці для кожної категорії, що відображають її наявність. Це дозволяє моделі добре працювати з текстовими параметрами, перетвореними на числові значення.

Наступним кроком є розподіл оброблених даних на навчальну, тестову та валідаційну вибірки. Це необхідно для перевірки продуктивності моделі на даних, яких вона раніше не бачила. Дані розподіляються у співвідношенні 70:20:10: 70% – для навчання, 20% – для тестування, 10% – для валідації. Перемішування даних перед розподілом гарантує, що модель не буде залежати від порядку записів у наборі.

Усі підготовлені дані формуються у вектори, де кожен вектор представляє одне завдання. Наприклад: Task Vector=[Assignee, Priority, Type, Estimate, Actual Time], де кожен елемент є числовим значенням або закодованим параметром. Це спрощує введення даних у модель та дозволяє ефективно їх обробляти. Ретельна підготовка даних забезпечує високу якість і стабільність навчання моделі.

Побудова нейронної мережі для прогнозування тривалості та ризиків. Розробка нейронної мережі для прогнозування тривалості виконання завдань і оцінки ризиків базується на багатошаровій архітектурі, яка дозволяє ефективно працювати з різнорідними даними. Модель орієнтована на одночасне вирішення двох задач: регресії для оцінки тривалості та класифікації для визначення рівня ризиків. На першому етапі визначається структура вхідного шару. Дані про завдання

формуються у вигляді числових векторів, що включають такі параметри, як виконавець, пріоритет та тип завдання, початкову оцінка часу, а також інші характеристики, що можуть впливати на результати. Кожен елемент такого вектора відповідає певному параметру завдання, який попередньо оброблений та нормалізований.

Далі будуються приховані шари нейронної мережі. У даній архітектурі використовується два основні приховані шари. Перший шар, який містить 64 нейрони, відповідає за виявлення базових закономірностей у вхідних даних. Другий шар із 32 нейронами здійснює більш деталізований аналіз і дозволяє моделі адаптуватися до складних взаємозв'язків між параметрами. У прихованих шарах застосовується функція активації ReLU (Rectified Linear Unit), яка забезпечує ефективність обчислень і дозволяє уникнути проблеми "зникаючого градієнта" [7, с. 85–117].

Вихідна частина моделі складається з двох окремих шарів. Перший вихід прогнозує тривалість завдань, використовуючи лінійну функцію активації, яка забезпечує отримання безперервних числових значень. Другий вихід відповідає за оцінку ризиків. Для цього використовується сигмоїдна функція активації, яка перетворює вихідні дані у ймовірність ризику в діапазоні від 0 до 1.

Для забезпечення стабільного навчання моделі використовується оптимізатор Adam (Adaptive Moment Estimation). Цей алгоритм адаптивно регулює швидкість навчання та забезпечує швидку конвергенцію навіть у складних задачах із великою кількістю параметрів. Крім того, для кожного виходу моделі задаються окремі функції втрат: середньоквадратична похибка для регресійної задачі прогнозування тривалості та бінарна крос-ентропія для класифікаційної задачі оцінки ризиків.

Ключовою перевагою такої архітектури є її гнучкість і багатофункціональність. Модель здатна обробляти різні типи даних і одночасно вирішувати декілька задач. Завдяки адаптивній структурі, модель може бути легко модифікована для включення додаткових характеристик або змін у вимогах [8, 450–460].

Навчання моделі на історичних даних. Навчання нейронної мережі здійснюється на основі історичних даних, які містять ключову інформацію про виконання завдань у попередніх проєктах. Процес навчання починається з розділення даних на три частини: навчальний набір для налаштування ваг нейронної мережі, тестовий набір для перевірки її точності та валідаційний набір, що дозволяє контролювати узагальнювальну здатність моделі та уникати перенавчання. Такий підхід гарантує, що модель буде ефективною на нових даних.

Для оптимізації ваг застосовується алгоритм зворотного поширення помилок у поєднанні з оптимізатором Adam. Цей метод забезпечує швидке й стабільне налаштування моделі навіть у випадках, коли дані мають значні відмінності в масштабах. Оптимізація виконується через ітерації (епохи), протягом яких модель аналізує навчальні дані й коригує свої параметри для зменшення похибок.

Навчання моделі враховує дві задачі. Перша – це прогнозування тривалості завдань, що є регресійною задачею. Для неї використовується функція втрат середньоквадратичної похибки, яка дозволяє вимірювати відхилення між фактичними та прогнозованими значеннями. Друга задача – це оцінка ризиків, яка виконується як класифікація. Для цього застосовується функція втрат бінарної крос-ентропії, яка оптимізує модель для роботи з ймовірностями ризиків [9, с. 60–75].

Під час кожної ітерації модель перевіряється на валідаційному наборі для оцінки її здатності до узагальнення. Цей підхід дозволяє забезпечити стабільну продуктивність моделі та уникати перенавчання, що є критичним для успішного використання моделі в реальних проєктах [10, с. 12–25].

Висновки. Використання штучного інтелекту для планування та оцінки ризиків у проєктах відкриває значні можливості для підвищення ефективності управління

проєктами. Запропонована модель побудована на основі нейронної мережі, яка здатна працювати з великими обсягами даних, обробляти різноманітні характеристики завдань і забезпечувати точність прогнозів навіть у складних умовах. Основними перевагами розробленого підходу є його здатність одночасно вирішувати дві ключові задачі: прогнозування тривалості виконання завдань і оцінка потенційних ризиків.

У ході роботи було детально розглянуто всі етапи створення моделі. Налаштування середовища дозволило забезпечити стабільність і доступ до необхідних інструментів для моделювання. Етап вибору та підготовки даних охоплював роботу з історичними даними, нормалізацію вхідних характеристик і обробку категорійних змінних, що є важливим аспектом для забезпечення якості вхідної інформації. Побудова нейронної мережі враховувала потребу у двох виходах: регресійному для прогнозування тривалості та класифікаційному для оцінки ризиків. Навчання моделі було спроектовано з використанням алгоритму Adam, який забезпечу ефективну оптимізацію ваг і адаптацію до різних типів даних.

Подальший розвиток моделі може включати інтеграцію більш складних архітектур, таких як трансформери, розширення набору вхідних параметрів для покращення точності прогнозів і інтеграцію з популярними інструментами управління проєктами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. Kerzner, H. Project Management: A Systems Approach to Planning, Scheduling, and Controlling. John Wiley & Sons, 2017, 129-133.
2. Hastie, T., Tibshirani, R., & Friedman, J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd Edition, Springer, 2009, 389-400.
3. Alpaydin, E. Introduction to Machine Learning. MIT Press, 2014, 295-300.
4. LeCun, Y., Bengio, Y., & Hinton, G. Deep learning. Nature, 2015, 436-444.
5. Sommerville, I. *Software Engineering.* 10th Edition, Pearson, 2015, 88-92.
6. Goodfellow, I., Bengio, Y., & Courville, A. *Deep Learning.* MIT Press, 2016, 315-350.
7. Schmidhuber, J. *Deep Learning in Neural Networks: An Overview.* Neural Networks, 2015, 61, 85-117.
8. Russell, S., & Norvig, P. *Artificial Intelligence: A Modern Approach.* 4th Edition, Pearson, 2020, 450-460.
9. Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, 2006, 60-75.
10. Graves, A. *Supervised Sequence Labelling with Recurrent Neural Networks.* Springer, 2012, 12-25.

REFERENCES:

1. Kerzner, H. (2017) Project Management: A Systems Approach to Planning, Scheduling, and Controlling. John Wiley & Sons, 129-133.
2. Hastie, T., Tibshirani, R., & Friedman, J. (2009) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd Edition, Springer, 389-400.
3. Alpaydin, E. (2014) Introduction to Machine Learning. MIT Press, 295-300.
4. LeCun, Y., Bengio, Y., & Hinton, G. (2015) Deep learning. Nature, 436-444.
5. Sommerville, I. (2015) *Software Engineering.* 10th Edition, Pearson, 88-92.
6. Goodfellow, I., Bengio, Y., & Courville, A. (2016) *Deep Learning.* MIT Press, 315-350.
7. Schmidhuber, J. (2015) *Deep Learning in Neural Networks: An Overview.* Neural Networks, 61, 85-117.
8. Russell, S., & Norvig, P. (2020) *Artificial Intelligence: A Modern Approach.* 4th Edition, Pearson, 450-460.
9. Bishop, C. M. (2006) *Pattern Recognition and Machine Learning.* Springer, 60-75.
10. Graves, A. (2012) *Supervised Sequence Labelling with Recurrent Neural Networks.* Springer, 12-25.