ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ТА ПРАКТИЧНОСТІ ІНСТРУМЕНТІВ ДЛЯ КОДОГЕНЕРАЦІЇ

***Четвериков Г.Г.**, професор, кафедра ПІ, ХНУРЕ*
***Денисюк В.М.**, студент, кафедра ПІ, ХНУРЕ*

Дослідження має на меті розробку та оцінку ефективності методів кодогенерації застосунків на основі UML діаграм. Визначення методу кодогенерації та його ефективності є важливим оскільки дозволяє формалізувати процес вибору оптимального рішення для конкретних проектів розробки.

Основною проблемою генерації вихідного коду є недостатня стандартизація вхідних даних, різноманітність мов та їх особливостей. На основі специфікації обраної мови та UML діаграми на якій представлені взаємозв'язки, сутності або алгоритми – можна створити вихідний код.

Для генерації вихідного коду була проаналізовна архітектура Transformer. Ця модель глибокого навчання, яка була вперше представлена в дослідженні "Attention is All You Need" від Google Research. Transformer має перевагу у паралельній обробці даних, що прискорює навчання порівняно з послідовними моделями.

Основна ідея трансформерів полягає в використанні механізму уваги (attention mechanism) [1] для моделювання взаємодії між різними частинами послідовності. Математичний вигляд нейромережі наведено у формулі:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Q – це матриця запитів (query matrix). Кожен рядок цієї матриці відповідає одному запиту. K – це матриця ключів. Кожен стовпчик цієї матриці відповідає одному ключу. V – матриця значень (value matrix). Кожен стовпчик цієї матриці відповідає одному значенню. Формула рахує ваговану суму значень V з урахуванням відповідностей між запитами Q та ключами K. Це дозволяє моделі приділяти більше уваги важливим частинам вхідних даних. d_k – розмірність ключів та запитів. У формулі вона використовується для градації приведення QK^T, щоб уникнути градієнтів, які можуть виникати у великих розмірах. Функція softmax використовується для нормалізації ваг у відповідності до їхніх значень, щоб отримати ймовірностний розподіл. Це дозволяє їм ефективно вирішувати завдання, де важлива контекстуальна інформація, такі як машинний переклад, генерація тексту або аналіз послідовностей.

Використання архітектури Transformer є зручною для розробки ефективного інструменту кодогенерації. Її здатність глибоко розуміти структуру та семантику вхідних даних робить її ідеальною для генерації коду

на основі UML діаграм, де необхідно враховувати велику кількість зв'язків та залежностей між різними елементами.

Виведений підхід використання навченої моделі на прикладі генерації коду з UML діаграми класів може виглядати наступним чином.

1. Зчитування та аналіз UML діаграми – зчитайте XML або JSON файл з UML діаграмою класів та проведіть аналіз структури даних. Ідентифікуйте класи, методи, властивості та зв'язки між класами. Під час цього процесу потрібно притримуватися підходу Model Driven Engineering (MDE) [2].

2. Створення словника об'єктів – для кожного класу створіть об'єкт, який містить всі його атрибути та методи. Використовуйте словник Python або об'єкт класу, який відповідає цьому класу.

3. Визначення вхідних та вихідних даних – визначте, яку інформацію з цих об'єктів класів буде використовувати ваша модель як вхідні дані (наприклад, атрибути класу, його методи) і які дані вона має згенерувати як вихід (наприклад, код методів класу).

4. Перетворення даних у векторне представлення – перетворіть кожен клас, метод та атрибут у векторне представлення за допомогою Word Embeddings, One-Hot Encoding або іншої техніки. Наприклад, для кожного класу можна створити вектор, який містить інформацію про його назву, атрибути та методи.

5. Створення вихідних даних – створіть вихідні дані у форматі, який вказує, який код потрібно згенерувати для кожного класу, методу та атрибуту. Наприклад, для кожного методу можна створити рядок, який містить заголовок методу та його тіло.

6. На основі вихідного коду можна доповнити та оновити навчену модель використовуючи рішення навчання на основі повторного використання знань [3].

Список літератури

1. Wikipedia. (2020). Attention (machine learning). https://en.wikipedia.org/wiki/Attention_(machine_learning).
2. Bashir, S., & Galadanci, M.I.M. (б. д.). Automatic code generation from uml diagrams: the state-of-the-art. Science World Journal, 13(4).
3. Каратаєв, О., & Шубін, І. (2023). Проблеми повторного використання знань у процесі проєктування програмних систем. Innovative technologies and scientific solutions for industries, 2(24), 62. https://doi.org/10.30837/ITSSI.2023.24.0.