УДК 004
[1, 2] **Т.І. Легіневич**
Аспірант
[2] **А.М. Глибовець**
Доктор технічних наук, доцент
[1] *Інститут програмних систем НАН України, Київ*
[2] *Національний університет "Києво-Могилянська Академія", Київ*

АРХІТЕКТУРА СИСТЕМИ МОДЕЛЕЙ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗНАХОДЖЕННЯ ПОДІБНОСТІ ОБ'ЄКТІВ В ГЕТЕРОГЕННОМУ СЕРЕДОВИЩІ

Вступ. Глибока нейронна мережа – це найбільш обговорювана тема серед дослідників у галузі штучного інтелекту. Вона дала надзвичайно перспективні результати у вирішенні різноманітних задач, таких як: розуміння природної мови, мовного перекладу, розпізнання обличчя та інші[1]. Ось чому існує потреба у впровадженні глибокої нейронної мережі в інформаційні системи для вирішення задач. Існує декілька проблем, що потрібно вирішити під час процедури впровадження моделей. Наприклад, затрати ресурсів на обчислення при процесі навчання нейронних мереж, безперервне оновлення моделей та їх використання у динамічних системах.

Архітектура. Розглянемо архітектуру безперервного конвеєра для тренувань та розміщення моделей на серверах, як єдину інтегровану систему. Вона складається з трьох частин: управління даними, навчання та розміщення моделей.

Модуль управління даними відповідає за попередню обробку та зберігання даних. Загально відомо, що глибинні моделі нейронних мереж вимагають великої кількості даних для процесу навчання. Система управління базами даних загального призначення задовольняє усім вимогам поточного модуля. Ми обрали базу даних PostgreSQL [2] з відкритим вихідним кодом для системи. Кожний новий об'єкт з модуля розміщення зберігається в таблиці вихідних даних, а відповідна процедура додає його у чергу попередньої обробки. Декілька процесів витягують нові об'єкти з черги незалежно один від одного і виконують операції попередньої обробки, такі як очищення і приведення до нового представлення об'єкту в таблиці бази даних.

Навчальна черга створюється за запитом шляхом випадкового вибору обробленого об'єкта з відповідної таблиці бази даних. Навчальний модуль вимагає найбільше обчислювальних ресурсів та є фінансово затратним, тому зазвичай він працює на графічному процесорі. Існує кілька платформ з відкритим вихідним кодом, які підтримують графічні процесори і дозволять тренувати глибинні нейронні мережі, такі як: Theano [3] та Tensorflow [4].

Вирішальна частина полягає у необхідності управляти висхідною кількістю моделей та динамічно їх розміщувати на сервері для обробки запитів. З цією метою було вирішено, що кожна збережена модель містить 4 елементи:

ваги, структура, параметри моделі і статистика. Такий простий підхід дозволяє нам використовувати різні платформи для навчання та обслуговування моделей, все що потрібно зробити, це реалізація сценарію для завантаження збереженої моделі в певному форматі. Для обробки клієнтських запитів було впроваджено модельний диспетчер, який несе відповідальність за обслуговування моделей. Кількість моделей, що обслуговуються обмежується лише ресурсами сервера. Це дозволяє нам одночасно використовувати кілька моделей і переадресовувати запити на конкретну версію моделі або іншу, не змінюючи жодного API. Таким чином, система могла б одночасно використовувати декілька моделей, серед яких диспетчер, який матиме можливість вибирати модель, що повинна бути використана. Важливо розуміти які саме моделі мають найбільшу точність і використовують менше ресурсів. Інша важлива особливість інформаційної панелі керування – це можливість запускати навчання для нових нейронних мереж. У системі знаходження подібності між об'єктами модуль розміщення був реалізований на базі Flask платформи [5] для обробки клієнтських запитів. Також існує платформа під назвою TensorFlow Serving [6], яка має аналогічну функціональність для обслуговування моделей, але без інформаційної панелі керування.

Висновки. Запропонована архітектура дозволяє постійно тренувати моделі та порівнювати їх між собою. Важливіше, що модель може бути протестована на чинній системі для обраних клієнтів (бета-тестерів) і легко виключатись або змінюватись без впливу на інших користувачів або модифікації системи. Різниця буде лише в параметрі запиту, де слід вказати версію або ім'я бажаної нейронної мережі. Модуль розміщення може використовувати звичайний сервер без графічного процесора. Якщо система не навантажена це зменшує фінансові витрати на обчислювальні ресурси, особливо якщо ваша система розміщується в хмарі.

Список використаних джерел

1. LeCun Y. Deep learning / Y. LeCun, Y. Bengio, G. Hinton. // Nature. – 2015. – №512. – С. 436–444.

2. PostgreSQL 9.6.0, PostgreSQL Global Development Group. [Електронний ресурс]. – n.d. – Режим доступу до ресурсу: https://www.postgresql.org/docs/.

3. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions / Theano Development Team. – 2016. – arXiv e-prints abs/1605.02688. – URL: http://arxiv.org/abs/1605.02688

4. Abadi, M., Barham, P., Chen, J., Chen,Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al. Tensorflow: A system for large-scale machine learning,/ Abadi, M., Agarwal, A. et al.// Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). – Savannah, Georgia, USA, 2016.

5. Flask is a microframework for Python based on Werkzeug and Jinja2. [Електронний ресурс]. – n.d. – Режим доступу до ресурсу: http://flask.pocoo.org/.

6. TensorFlow Serving is an open-source software library for serving machine learning models. . [Електронний ресурс]. – n.d. – Режим доступу до ресурсу: https://tensorflow.github.io/serving/.