Рецензия

на выпускную квалификационную работу студента СПбГУ
кафедры системного программирования, обучающегося в группе 16.Б11-мм,
Соколова Ярослава Сергеевича
по теме: “Система автодополнения фрагментами кода”

Целью данной выпускной квалификационной работы является создание плагина для PyCharm IDE автодополнения кода фрагментами из нескольких токенов вместо обычного автодополнения одним токеном. Эта цель является актуальной поскольку на данный момент для среды разработки PyCharm нет таких плагинов с открытым исходным кодом, которые бы позволяли улучшать качество их работы внутри компании.

Для достижения поставленной цели были сформулированы и выполнены следующие задачи: проведение обзора существующих систем автодополнения, выбор подхода к автодополнению фрагментами кода, реализация и оптимизация выбранной системы для использования в PyCharm IDE и проведение апробации плагина. По итогам всего проекта было реализовано клиент-серверное приложение, позволяющее из PyCharm IDE вызывать автодополнение фрагментами кода.

Структурно работа разбита на несколько частей: введение, постановка цели и задач, обзор существующих алгоритмов и выбор подхода, реализация алгоритма автодополнения и его оптимизация с последующей апробацией системы.

В обзоре рассмотрены существующие системы автодополнения кода одним токеном и рассмотрены их различия. Затем были приведены существующие аналоги, решающие задачу автодополнения фрагментами кода, таких как проприетарная система Deep TabNine с закрытым исходным кодом. На основе этого была спроектирована система автодополнения фрагментами кода, которая и была реализована в рамках данной работы.

В основной части работы был описан процесс разработки спроектированной системы, которая состоит из нескольких подсистем: токенизатор кода, авторегрессионая модель предсказания токенов, алгоритм выбора лучших кандидатов из сгенерированных цепочек токенов на основе лучевого поиска (beam search). Стоит упомянуть, что токенизатор кода был специально адаптирован для работы со специфической доменной областью исходного кода, что позволило добиться сразу двух целей – ускорение работы и улучшение качества предсказаний. Еще одно улучшение было сделано в алгоритме выбора следующих кандидатов в случае наиболее частого использовании системы автодополнения – вызов на последнем недописанном слове. Исходный код находится в открытом доступе.

В разделе апробации были измерены качество и скорость работы полученной системы, а также проведено сравнение с существующим автодополнения в PyCharm IDE и Deep TabNine. Разработанная в рамках данной работы существенно превосходит аналоги, однако уступает в скорости. Также было измерено влияние предложенных

модификаций для доменной области: они позволяют генерировать цепочки токенов в примерно в 1.5 раза быстрее и с заметным улучшением итогового качества.

К моментам, которые можно было бы улучшить в данной работе, можно отнести:

- Большое время работы полученной системы даже с учётом клиент-серверной архитектуры с мощной GPU (даже по сравнению с локально работающей на CPU моделью Deep TabNine).
- Использование контекста лишь с одной стороны, в то время когда код обычно модифицируется в положении, когда контекст есть с обоих сторон.
- Упоминание Transformer-XL в работе, но его отсутствие в части с апробацией системы.
- Слабое обоснование выбора автогрессионных моделей.

На основании вышеизложенного считаю, что данная выпускная квалификационная работа выполненна на высоком уровне, заслуживает оценки “отлично”, а ее автор – присвоения ему степени бакалавра.

Булычев Егор Геннадьевич
Machine Learning Engineer at “IntelliJ Labs” Co. Ltd
Дата: 08.06.2020