*1 слайд. Титульник*

Добрый день, уважаемые члены комиссии. Вашему вниманию предлагается выпускная квалификационная работа на тему «Разработка программы имитационного моделирования моделей нейронных сетей».

*2 слайд. Актуальность темы*

За последние 5 лет в машинном обучении развивается новый интересный подход - in-memory computing. Применить такой подход стало возможно за счет разработки нового вида оперативной памяти ReRAM, то есть Resitive RAM - памяти, построенной на переменных резисторах. Для оперативной памяти преимущества такой системы заключаются в том, что по скорости она не уступает транзисторной ОЗУ, но при выключении питания вся информация сохраняется. Для машинного обучения применение ReRAM заключается в том, что самую частую операцию взвешивания входного вектора можно выполнять не в цифровом виде на процессоре, как это делается сейчас, а в аналоговом. Каждая ячейка ReRAM выступает в роли делителя напряжения, который изменяет входное напряжение на определенный коэффициент. Она может выполнять умножение весовых коэффициентов в аналоговом виде, а операция умножения является самой частой в машинном обучении. Проблема в таком подходе заключается в том, что аналоговый принцип обработки информации в ReRAM вносит погрешности из-за неизбежного влияния шумов и помех.

Переход на цель – актуальность задачи является оценка того можно ли на рерам аппаратно запустить готовые модели нейронных сетей

(будут ли на рерам готовые модели нейронных сетей работать с приемлемой точностью)

*3 слайд. Цель и задачи*

Актуальностью темы является оценка того, можно ли на рерам аппаратно запустить готовые модели нейронных сетей, поэтому можно выделить следующую цель этой работы: разработать программу, при помощи которой возможно определить, будут ли на ReRAM готовые модели нейронных сетей работать с приемлемой точностью.

Для выполнения данной цели на работу были поставлены задачи, представленные на слайде.

*4 слайд. Обзор программ модели моделирования ИНС*

Для разработки программы необходимо определиться с выбором средства. Есть много программ, которые могут моделировать искусственные нейронные сети (такие как Deductor или Orange), но они не подойдут для решения данной задачи, т.к. в них не реализована возможность низкоуровневого вмешательства в конфигурацию весов. Python и Matlab, которые являются популярными средствами для создания нейронных сетей, могут дать возможность реализовать это. Но т.к. Matlab распространяется на коммерческой основе, выбор пал на Python.И это оптимально, т.к. он содержит способы как создания нейронных сетей, так и распараллеливания задач.

*5 слайд. Общий принцип работы*

Для того, чтобы ускорить работу программы, можно разбить одну большую задачу на много маленьких подзадач, чтобы выполнить параллельные расчеты. При данном подходе скорость расчетов возрастет, но потребуется достаточно мощности и аппаратных характеристик системы, чтобы выполнить это.

*6 слайд. Проблема GIL и ее решение*

GIL – это своеобразная блокировка, позволяющая только одному потоку управлять интерпретатором Python. Это означает, что в любой момент времени будет выполняться только один конкретный поток. Работа GIL может казаться несущественной для разработчиков, создающих однопоточные программы. Но во многопоточных программах отсутствие GIL может негативно сказываться на производительности процессоро-зависимых программ. Поскольку GIL позволяет работать только одному потоку даже в многопоточном приложении, он заработал репутацию «печально известной» функции.

Поэтому необходимо отойти от концепции потоков и попытаться использовать мощность процессора на полную, задействовав все его физические и виртуальные ядра. При данном подходе исключено блокирование операций, но требуется большой объем оперативной памяти для того, чтобы хранить много копий функций, которые будут разбиваться на процессы, т.к. каждый процесс использует собственную локальную память.

*7 слайд. Проект программы*

На данном слайде представлена диаграмма последовательности разработанной программы. Как можно заметить, в программе используются 4 виджета, которые используются последовательно.

*8 слайд. Алгоритмы имитации погрешностей весов*

Чтобы понять, как именно имитировать погрешность аналогового подхода, нужно понять, что …

Чтобы сымитировать погрешность, которая возникает на резисторах ReRAM, можно воспользоваться центральной предельной теоремой, суть которой заключается в том, что сумма достаточно большого количества слабо зависимых случайных величин, имеющих примерно одинаковые масштабы (ни одно из слагаемых не доминирует, не вносит в сумму определяющего вклада), имеет распределение, близкое к нормальному. Поэтому добавление алгоритма изменения весов, используя нормальное распределение, является основным способом имитации погрешности аналогового сигнала.

Метод равномерного распределения был добавлен для того, чтобы провести некоего рода стресс-тест модели, и посмотреть, как ее показатели точности изменятся при сильном изменении весов.

*9 слайд. Протокол взаимодействия с программой*

При установке модуля себе на ПК, пользователю-программисту необходимо написать дочерний класс от родительского класса модуля и описать 5 функций:

- загрузки модели в программу;

- загрузки тестовых данных;

- функции для получения весов и установки их в новую модель;

- функцию получения точности модели.

*10 слайд. Реализованная программа*

На этом слайде вашему вниманию представлены все 4 состояния программы.

Сначала программа проверяет описанные пользователем методы и выдает ошибки, если таковые имеются (например, модуль multiprocessing может принимать и разбивать файлы в двоичном виде, которые делает модуль Pickle, поэтому модель должна иметь возможность быть корректно обработана этим модулем).

Следующий виджет позволяет пользователю настроить конфигурацию теста, а именно: выбрать количество используемых ядер, алгоритм тестирования и список тестируемых параметров данного алгоритма. После правильной настройки конфигурации станет доступна кнопка «Начало тестирования».

Следующий виджет необходим для отображения процесса тестирования, он отображает также примерное время до конца тестирования.

В последнем окошке пользователь увидит график, полученный в результате тестирования, и также получит возможность сохранить результаты:

- в виде изображения;

- в виде файла .xlsx;

- в виде текстового файла.

*11 слайд. Benchmark (Компьютер)*

На данном слайде приведены результаты тестирования разработанной программы на компьютере.

Конфигурация компьютера следующая: объём оперативной памяти 16 Гб и частотой 3200 МГц и процессором AMD Ryzen 5 3600, имеющим тактовую частоту 3,6 ГГц и 6 физических (12 виртуальных) ядер (в дальнейшем Компьютер).

Тестирование производилось на двух разных моделях нейронный сетей. Первая содержит 800 нейронов на входном слое и 10 – на выходном (в сумме 810), а вторая нейронная сеть содержит 3000 нейронов (это та же самая модель, но с добавлением скрытых слоев).

На слайде вы видите уменьшение времени, затраченного на тестирование при увеличении используемых ядер процессора. Размер нейронной сети в данном случае увеличивал время работы программы в 2,5-3 раза.

*12 слайд. Benchmark (Ноутбук)*

На данном слайде приведены результаты тестирования разработанной программы на ноутбуке.

Конфигурация ноутбука следующая: объём оперативной памяти 8 Гб и частотой 2400 МГц и процессором Intel® Core(TM) i5-7200U, имеющим тактовую частоту 2,5 ГГц и 4 физических ядра (в дальнейшем Ноутбук).

Тестирование производилось на тех же моделях нейронных сетей, что и на компьютере.

Так как здесь количество ядер и частоты процессора и оперативной памяти ниже, чем на компьютере, процесс тестирования занимает больше времени, но, как и в случае с компьютером, увеличение количества ядер, используемых программой, дает прирост в скорости примерно в 2 раза.

*13 слайд. Заключение*

На данном слайде представлен список задач, которые были реализованы в рамках ВКР, а именно:

* реализована возможность тестировать любую модель нейронной сети;
* добавлены алгоритмы имитации аналоговых помех для тестирования работы нейронных сетей (алгоритм равномерного распределения и алгоритм нормального распределения);
* повышена производительность работы программы за счет использования метода распараллеливания процессов в целях сокращения времени тестирования;
* сформулированы выводы о работе реализованной программы тестирования.

*14 слайд. Спасибо за внимание.*

А этом у меня все, я готов ответить на ваши вопросы.