Киселева А.С.
Козырев С.В.

Научно-исследовательская работа на тему

«Выбор процессорной архитектуры для обучения нейронных сетей»

1.Введение

Вычислительные системы прошли долгий путь с тех пор, как в середине XX века были разработаны первые электронные компьютеры. За последнее десятилетие технологический прогресс привел к значительному повышению вычислительной мощности, скорости и эффективности решения задач обработки данных. Двумя ключевыми компонентами современных вычислительных систем являются процессоры и видеокарты, которые за прошедшие годы претерпели впечатляющее развитие.

Стоит понимать, что архитектуры процессора и видеокарты значительно отличаются друг от друга, в связи с этим их сфера применения до определенного времени была весьма узконаправленной: процессоры (CPU) отвечали за выполнение последовательных операций и вычислений, в то время как видеокарты (GPU) отвечали за обработку 3D-графики. С течением времени архитектуры CPU совершенствовались, но достигли определенного потолка своих возможностей, в то время как архитектура видеокарт (GPU) стремительно наращивала свои обороты, и в последнее время стала применима для параллельной обработки больших объемов данных, и, как следствие, для задач, требующих огромной вычислительной мощности, таких как игры, рендеринг видео и машинное обучение.

Развитие подобных архитектур сыграло ключевую роль в эволюции вычислительных систем, предоставляя возможность производить все более сложные вычисления, но вместе с этим появилась потребность в выборе конкретных вычислительных мощностей, которые необходимо использовать для эффективного решения поставленных задач.

2.Развитие микропроцессорных систем

Микропроцессор — это небольшой компьютер на микросхеме, который может выполнять вычисления и другие операции. Микропроцессоры проектируются и разрабатываются специализированными группами инженеров и ученых, работающих в таких компаниях, как Intel, AMD, ARM и Qualcomm.

Первый микропроцессор Intel 4004 был выпущен в 1971 году. Он имел 4-битную архитектуру и был способен выполнять до 60 000 инструкций в секунду. С появлением микропроцессоров значительно уменьшились и габариты вычислительных систем, так как за определенный набор инструкций теперь отвечает компактный микрочип.

В течение следующих нескольких лет микропроцессоры продолжали совершенствоваться. Появились 8-разрядные Intel 8080 и 16-разрядные Intel 8086. Разрядность процессора определяла размер обработки данных за один такт. Такой параметр, как увеличение разрядности процессора, способствовал развитию персональных компьютеров, которые произвели революцию в способах работы, развлечений и общения людей.

Одним из наиболее значительных достижений в архитектуре микропроцессоров стало появление в 1980-х годах архитектуры CISC (Complex Instruction Set Computing). Процессоры CISC были разработаны для выполнения сложных операций, что сделало их более универсальными, чем предыдущие архитектуры. Однако, процессоры с подобной архитектурой отличались медленной скоростью обработки и своей дороговизной.

В ответ на это в 1980-х годах была представлена новая архитектура под названием RISC (Reduced Instruction Set Computing). Процессоры RISC были разработаны для выполнения более простых операций, что делало их быстрее и эффективнее, чем процессоры CISC. Они также были дешевле в производстве, что делало их более доступными для широкого спектра применений.

В 1990-х годах была представлена новая архитектура под названием MISC (Minimal Instruction Set Computing). Процессоры MISC были разработаны для очень быстрого выполнения небольшого количества операций, что

делало их идеальными для приложений, требующих быстрой обработки, но не требующих универсальности процессоров CISC или RISC.

Еще одним значительным достижением в области микропроцессоров стало появление в 1980-х годах архитектуры ARM (Advanced RISC Machine). Процессоры ARM были разработаны с низким энергопотреблением, что сделало их подходящими для использования в портативных устройствах, таких как смартфоны и планшеты. Сегодня процессоры ARM используются в подавляющем большинстве смартфонов и других портативных устройствах.

Помимо микропроцессоров, еще одним ключевым событием в области компьютерных технологий стало появление в 1990-х годах графических процессоров (GPU). Графические процессоры были разработаны для выполнения сложных параллельных вычислений, необходимых для рендеринга высококачественной графики в видеоиграх и других приложениях. Сегодня графические процессоры используются не только для игр, но и для научных симуляций, машинного обучения и других приложений, интенсивно использующих данные.

3. Проблемы развития микропроцессоров

По мере того, как процессоры становились меньше и имели более плотное расположение транзисторов, их стало труднее охлаждать, что могло привести к перегреву и снижению производительности. Кроме того, поскольку размер транзисторов уменьшился, количество энергии, необходимой для их работы, увеличилось, что приводило к накоплению тепла и ограничению производительности.

Существуют также фундаментальные физические ограничения размера транзисторов и скорости, с которой они могут переключаться. Эти ограничения наложены квантовой механикой и тем фактом, что на атомном уровне электроны ведут себя непредсказуемо и трудно поддаются контролю. Эти пре-

делы называются «концом закона Мура», который заключается в том, что количество транзисторов в микрочипе удваивается примерно каждые два года, но это не может продолжаться бесконечно.

В результате этих физических ограничений ученые и инженеры изучают новые способы дальнейшего повышения производительности, например, за счет использования новых материалов, новых архитектур и специализированных процессоров для конкретных задач. Одной из таких архитектур, как раз и стала архитектура графических процессоров, производительность которой, за последние годы показала колоссальный прирост. На рисунке 1 демонстрируется производительность GPU в сравнении с CPU. Исходя из этого можно сделать вывод о дальнейших перспективах развития данной архитектуры

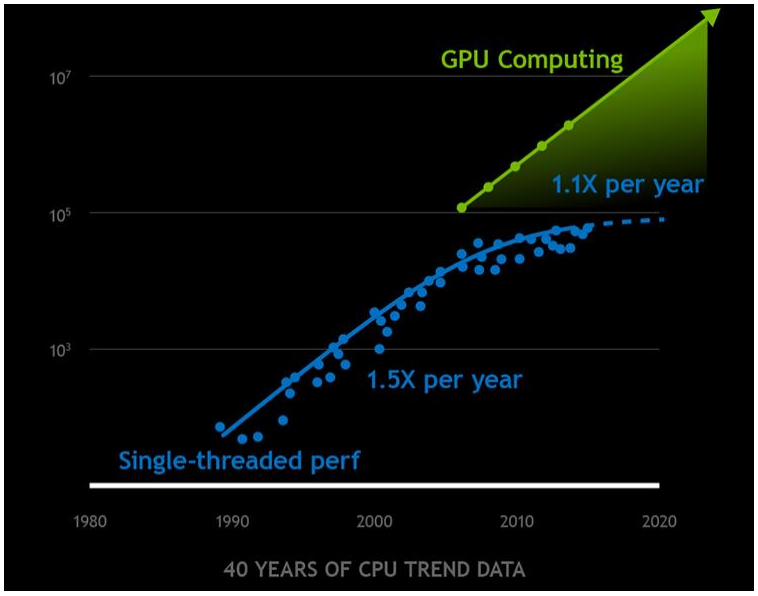

Рисунок 1 - сравнение производительностей CPU и GPU

Далее мы более детально изучим ранее упомянутые архитектуры, рассмотрим их плюсы и минусы, а также проведем эксперимент, в котором сравним эффективность использования графического и центрального процессоров при работе с нейронными сетями.

4.CPU

Центральный процессор — это микропроцессор, который выполняет инструкции, заданные программой на основе арифметических операций, операций управления, логики и ввода-вывода. Сегодня во многих компьютерах используется многоядерный процессор, который представляет собой один чип, содержащий два или более процессора (ядра). Процессоры с несколькими ядрами могут запускать несколько процессов параллельно, таким образом значительно ускоряя время выполнения.

4.1 Архитектура

Структура архитектуры ЦП состоит из нескольких компонентов, которые работают вместе для обработки данных и выполнения вычислений:

Блок управления (CU) отвечает за выборку и декодирование инструкций из памяти, а также за управление потоком данных между различными компонентами процессора. CU также управляет планированием и распределением ресурсов между различными процессами и потоками.

Арифметико-логическое устройство (ALU) выполняет арифметические и логические операции, такие как сложение, вычитание, умножение и деление. Оно также может выполнять логические операции, такие как И, ИЛИ и НЕ.

Регистры – это небольшие, быстродействующие ячейки памяти в процессоре, которые используются для хранения временных данных во время вычислений.

Кэш – это небольшая высокоскоростная память, в которой хранятся часто используемые данные. Он используется для уменьшения задержки при обращении к памяти и повышения производительности процессора.

Блок управления памятью (MMU) управляет иерархией памяти и обеспечивает эффективное хранение программ и данных и доступ к ним. Он переводит виртуальные адреса в физические, управляет защитой памяти и контролирует доступ к памяти.

Интерфейс соединяет ЦП с материнской платой компьютера и обеспечивает необходимую скорость передачи данных для высокоскоростной связи между ЦП и другими компонентами.

Вот основные характеристики процессора:

- Несколько ядер, интегрированых в один чип;
- Способность взаимодействовать с другими компонентами и регулировать распределение ресурсов;
- Использование небольших объемов встроенной памяти;
- Измерение скорости в герцах (частоту внутренних тактовых импульсов процессора в циклах в секунду);
- Измерение эффективности в IPC (количество операций за цикл).

5.GPU

Графический процессор (GPU) - это альтернативная архитектура процессора, основной задачей которого является отображение и обработка графики на электронном устройстве. Отчасти графический процессор похож на обычный, но внутреннее устройство архитектуры в нем кардинально отличается.

Графические процессоры в основном бывают двух типов:

- Интегрированные графические процессоры, встроенные в материнскую плату компьютера;
- Дискретные графические процессоры - это отдельный компонент, имеющий собственную плату, чип и память. Они больше и более энергозависимы, чем интегрированные графические процессоры.

GeForce 256 от NVIDIA был первым широко используемым GPU, специально разработанным для графики реального времени, приложений, требующих большого количества арифметических операций и высокой пропускной способности памяти.

5.1 Архитектура

Структура архитектуры GPU состоит из нескольких компонентов, которые работают вместе для обработки данных и выполнения вычислений:

Потоковые мультипроцессоры (SM) - это фундаментальный строительный блок архитектуры GPU. Каждый SM содержит набор вычислительных блоков, называемых ядрами CUDA, которые могут выполнять тысячи вычислений одновременно. SM также имеет собственный кэш инструкций, общую память и регистры.

Ядра CUDA - это вычислительные блоки внутри SM. Эти ядра предназначены для параллельного выполнения множества вычислений, что делает их идеальными для задач глубокого обучения, требующих большого количества матричных операций и сверток.

Иерархия памяти состоит из нескольких уровней кэшей, общей памяти и глобальной памяти. Кэши используются для хранения часто используемых данных, а общая память используется для обмена данными между потоками в рамках одного блока. Глобальная память используется для хранения данных, доступных всем потокам в GPU.

Блок управления отвечает за управление потоком данных и операций между различными компонентами архитектуры GPU.

Интерфейс соединяет GPU с материнской платой компьютера и обеспечивает необходимую скорость передачи данных для высокоскоростной связи между CPU и GPU.

Вот основные характеристики графического процессора:

- Большое количество арифметических логических блоков (ALU) способных обрабатывать огромное количество данных в несколько потоков. Кроме того, графические процессоры включают сотни ядер, которые могут управлять несколькими потоками обработки одновременно.
- Подключение через порты для подключения дискретного GPU к монитору можно использовать несколько портов.

- Способность выполнять векторные вычисления и вычисления с плавающей точкой, то есть математические операции над приближенным представлением действительных чисел.
- Пригодность для параллельных вычислений

6.Сравнение CPU и GPU при обучении нейросетей

6.1 Экспериментальная платформа

Эксперименты проводились на компьютере под управлением Windows 10 Professional x64 Custom Version:

1.Процессор - Intel(R) Core(TM) i5-10400F CPU @ 2.90GHz 2.90 GHz -> TurboBoost Active 4.3GHz AllCore

2.Видеокарта - MSI NVIDIA GeForce RTX 3070 Gaming Z Trio GDDR6 8Gb

3.Материнская плата - ASRock B460M Pro4

4.Оперативная память - HyperX Fury GDDR4 Dual Channel 16gb

6.2 Экспериментальное обоснование выбора GPU

Основным преимуществом использования графического процессора для обучения нейронных сетей, как мы уже выяснили ранее, является его способность выполнять параллельную обработку данных. В отличие от центральных процессоров, которые обычно имеют несколько ядер, оптимизированных для последовательной обработки, графические процессоры имеют сотни или даже тысячи ядер меньшего размера, оптимизированных для параллельной обработки. Это позволяет выполнять множество вычислений одновременно, значительно сокращая время, необходимое для обучения нейронной сети.

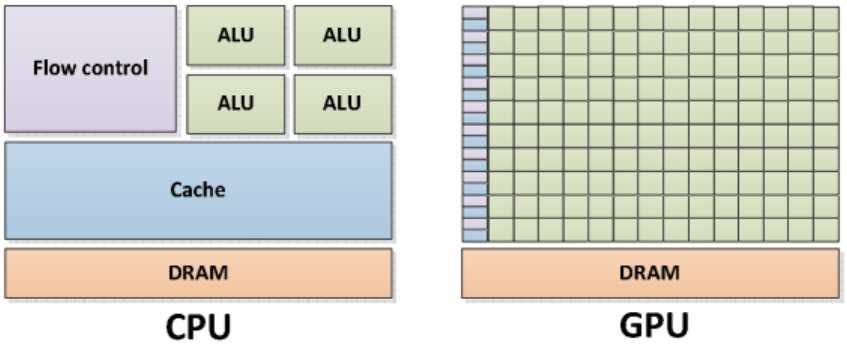

Рисунок 2 - Архитектура ЦП (слева) и архитектура ГП (справа)

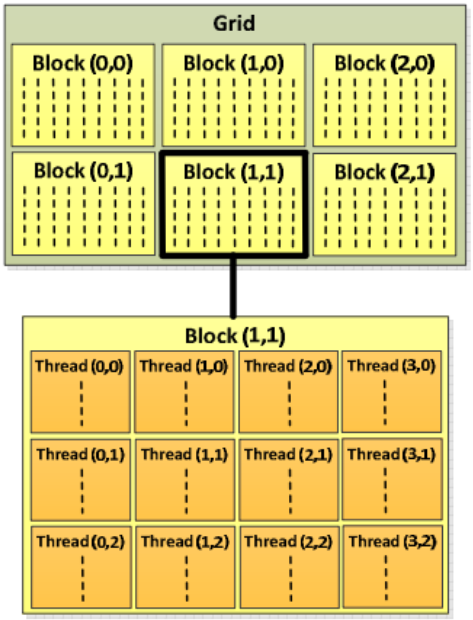

Рисунок 3 - Иллюстрация блоков и потоков на GPU

В дополнение к возможностям параллельной обработки графических процессоров они также имеют высокую пропускную способность памяти. Пропускная способность памяти относится к скорости, с которой данные могут передаваться между памятью графического процессора и его вычислительными ядрами. Это имеет решающее значение для обучения нейронных сетей, поскольку включает передачу больших объемов данных между слоями сети во время каждой итерации процесса обучения.

Среди нейросетевых моделей, которые демонстрируют наибольшую пользу от использования GPU: конволюционные (сверточные) нейронные сети (CNN), рекуррентные нейронные сети (RNN) и генеративные адверсарные

сети (GAN). Кроме того, многие популярные фреймворки глубокого обучения, такие как TensorFlow и PyTorch, имеют встроенную поддержку ускорения GPU. Это позволяет легко обучать нейронные сети на графических процессорах, не требуя значительных изменений в коде.

6.3 Выбор модели нейронной сети

Для проведения эксперимента мы будем использовать свёрточную нейронную сеть (CNN) для классификации изображений из набора данных MNIST. Основные оптимизации использования графического процессора включают использование слоев Conv2D и MaxPooling2D, которые можно эффективно распараллелить на графическом процессоре, а также использование метода подгонки для пакетного обучения модели.

Листинг 1 - код

```
import tensorflow as tf
import time

# Архитектура модели
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')
])

# Компиляция модели
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Загрузка датасета MNIST
mnist = tf.keras.datasets.mnist
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Нормализация данных
x_train, x_test = x_train / 255.0, x_test / 255.0
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
```

Далее нам нужно определить тип процессора, на котором будут происходить вычисления. За счет поддержки библиотекой TensorFlow работы с графическим процессором, нам достаточно дополнить код соответствующими функциями обучения с использованием CPU и GPU отдельно.

Листинг 2 - код

```
# Обучение модели на GPU
with tf.device('/GPU:0'):
    start_time_gpu = time.time()
    model.fit(x_train, y_train, epochs=10, batch_size=32, validation_data=(x_test, y_test))
    end_time_gpu = time.time()

# Обучение модели на CPU
with tf.device('/CPU:0'):
    start_time_cpu = time.time()
    model.fit(x_train, y_train, epochs=10, batch_size=32, validation_data=(x_test, y_test))
    end_time_cpu = time.time()

#Сравнение по затраченному времени на Обучение
print("Time taken to train the model on GPU: {:.2f} seconds".format(end_time_gpu - start_time_gpu))
print("Time taken to train the model on CPU: {:.2f} seconds".format(end_time_gpu - start_time_gpu))
```

6.4 Результат

При обучении модели с использованием GPU, мы можем заметить, что параметр шага обучения равен 2ms/step и суммарное время, затраченное на обучение, равняется 47.96 секундам. Процесс обучения представлен на рисунке 4.

```
Epoch 1/10
1875/1875 [==============================] - 7s 3ms/step - loss: 0.1809 - accuracy: 0.9463 - val_loss: 0.0672 - val_accuracy:
0.9789
Epoch 2/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0625 - accuracy: 0.9809 - val_loss: 0.0520 - val_accuracy:
0.9823
Epoch 3/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0425 - accuracy: 0.9872 - val_loss: 0.0505 - val_accuracy:
0.9834
Epoch 4/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0304 - accuracy: 0.9907 - val_loss: 0.0448 - val_accuracy:
0.9842
Epoch 5/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0215 - accuracy: 0.9930 - val_loss: 0.0471 - val_accuracy:
0.9852
Epoch 6/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0174 - accuracy: 0.9945 - val_loss: 0.0548 - val_accuracy:
0.9837
Epoch 7/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0133 - accuracy: 0.9956 - val_loss: 0.0538 - val_accuracy:
0.9847
Epoch 8/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0092 - accuracy: 0.9971 - val_loss: 0.0563 - val_accuracy:
0.9845
Epoch 9/10
1875/1875 [==============================] - 5s 2ms/step - loss: 0.0077 - accuracy: 0.9973 - val_loss: 0.0543 - val_accuracy:
0.9859
Epoch 10/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0066 - accuracy: 0.9979 - val_loss: 0.0670 - val_accuracy:
0.9830
Time taken to train the model on GPU: 47.96 seconds
```

Рисунок 4 - Процесс обучения на GPU

При обучении модели с использованием CPU, мы можем заметить, что параметр шага обучения равен 7ms/step и суммарное время, затраченное на обучение, равняется 136.17 секундам. Процесс обучения представлен на рисунке 5.

```
Epoch 1/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.1702 - accuracy: 0.9498 - val_loss: 0.0706 - val_accuracy:
0.9774
Epoch 2/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.0612 - accuracy: 0.9812 - val_loss: 0.0574 - val_accuracy:
0.9811
Epoch 3/10
1875/1875 [==============================] - 14s 7ms/step - loss: 0.0421 - accuracy: 0.9869 - val_loss: 0.0547 - val_accuracy:
0.9813
Epoch 4/10
1875/1875 [==============================] - 14s 7ms/step - loss: 0.0301 - accuracy: 0.9902 - val_loss: 0.0487 - val_accuracy:
0.9844
Epoch 5/10
1875/1875 [==============================] - 14s 7ms/step - loss: 0.0211 - accuracy: 0.9930 - val_loss: 0.0483 - val_accuracy:
0.9827
Epoch 6/10
1875/1875 [==============================] - 15s 8ms/step - loss: 0.0149 - accuracy: 0.9952 - val_loss: 0.0523 - val_accuracy:
0.9841
Epoch 7/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.0118 - accuracy: 0.9960 - val_loss: 0.0507 - val_accuracy:
0.9845
Epoch 8/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.0088 - accuracy: 0.9972 - val_loss: 0.0508 - val_accuracy:
0.9864
Epoch 9/10
1875/1875 [==============================] - 13s 7ms/step - loss: 0.0065 - accuracy: 0.9978 - val_loss: 0.0613 - val_accuracy:
0.9853
Epoch 10/10
1875/1875 [==============================] - 14s 7ms/step - loss: 0.0074 - accuracy: 0.9975 - val_loss: 0.0561 - val_accuracy:
0.9860
Time taken to train the model on CPU: 136.17 seconds
```

Рисунок 5 - Процесс обучения на CPU

Стоит отметить, что в момент обучения процент загруженности GPU и CPU, был разным. Даже с учетом того, что экспериментальный центральный процессор (CPU) находился в состоянии TurboBoost Active и его тактовая частота была постоянной и равнялась 4.3GHz, его загруженность при обучении не была пиковой и составляла от 55-60%, в то время процент загруженности видеокарты равнялся 80-85%. Для наглядной демонстрации была реализована таблица со средним процентом загруженности, она представлена на рисунке 6. Этот факт говорит нам о том, что при равных условиях CPU не может полностью раскрыть свой потенциал из-за своих архитектурных особенностей, в то время как GPU успешно распределил нагрузку и использовал максимум своей производительности.

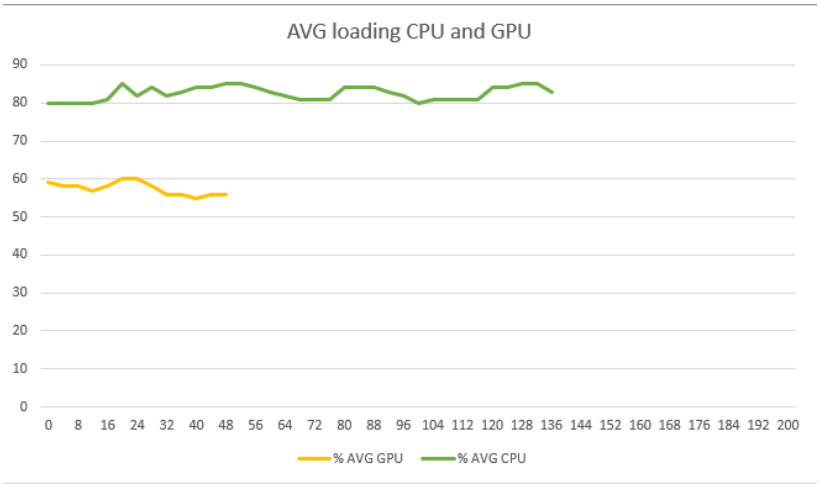

Рисунок 6 - Средний процент загруженности CPU и GPU

Вывод

Выполнив данное научное исследование, мы убедились в том, что архитектура графического процессора (GPU) является наиболее производительной и эффективной при работе с большими объемами данных в процессе обучения нейронных сетей. Данное преимущество обусловлено конструктивной особен-

ностью данной архитектуры и возможностью параллельной обработки больших объемов данных, в отличие от центральных процессоров, оптимизированных для последовательной обработки данных.

Список источников

1. Гудфеллоу Я., Бенджио И., Курвилль А. − Глубокое обучение, 2017 г.
2. Орельен Жерон − Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow, 2018 г.
3. Тарик Рашид – Создаем нейронную сеть, 2016 г.
4. Калачев А. – Многоядерные процессоры. Учебное пособие, 2014 г.
5. Таненбаум Эндрю, Остин Т. – Архитектура компьютера, 2019 г.
6. Таненбаум Эндрю, Бос Херберт – Современные операционные системы, 2019 г.