Колледж Автономной некоммерческой образовательной организации высшего образования «Научно-технологический университет «Сириус»

Реферат по предмету “Введение в специальность” по теме:

“Библиотека TensorFlow в Python”

Работу выполнил:

Cтудент группы К0709-24/2:

Братик Э. Г.

Преподаватель:

Яковлева С.В.

Сириус 2024 – 2025 уч. г.

**Оглавление**

[Введение 3](#_gjdgxs)

[Основная часть 4](#_30j0zll)

[1 История создания TensorFlow 4](#_1fob9te)

[1.1 История создания 4](#_3znysh7)

[2.2 Типы данных и форматы тензоров в TensorFlow 7](#_1t3h5sf)

[2.3 Слои 9](#_4d34og8)

[2.4 TensorFlow 2.x 11](#_2s8eyo1)

[2.4.1 Eager Execution по умолчанию 12](#_17dp8vu)

[2.4.2 Keras как основной API 12](#_3rdcrjn)

[2.4.3 Поддержка TensorFlow Hub и TensorFlow Lite 13](#_26in1rg)

[3 Применение TensorFlow 13](#_lnxbz9)

[3.1 Области применения 13](#_35nkun2)

[4 Преимущества и ограничения 14](#_44sinio)

[Вывод 16](#_2jxsxqh)

[Список литературы 17](#_z337ya)

# 

# **Введение**

**Тема реферата:** библиотека TensorFlow в Python

В современном мире технологии глубокого и машинного обучения занимают центральное место в развитии информационных систем. Среди платформ, предназначенных для реализации этих технологий, широкой популярностью пользуется библиотека TensorFlow.

Была разработана компанией Google. TensorFlow позволяет создавать, обучать и внедрять нейронные сети для решения разнообразных задач: от анализа естественного языка до распознавания объектов.

Актуальность исследования обусловлена быстрым развитием искусственного интеллекта и широким применением данной библиотеки в различных проектах.

Цель проекта: анализ архитектуры, функциональных возможностей и практического применения TensorFlow, а также оценка его преимуществ и ограничений.

# 

# **Основная часть**

## **1 История создания TensorFlow**

## 

TensorFlow - одна из самых мощных и популярных библиотек для машинного обучения. Разработана в 2015 году. TensorFlow был представлен как преемник предыдущей системы распределённого обучения DistBelief. Библиотека быстро завоевала популярность благодаря своей гибкости, масштабируемости и возможности реализации сложных нейронных сетей на различных вычислительных платформах.

### **1.1 История создания**

TensorFlow разрабатывалась Google Brain - командой исследователей Google, занимающейся передовыми разработками в области искусственного интеллекта. В 2015 году компания Google открыла этот инструмент всему миру, выпустив TensorFlow под открытой лицензией Apache 2.0 после длительного периода внутреннего использования. Теперь любой желающий может применять TensorFlow в своих проектах, будь то создание сложных систем распознавания изображений или разработка чат-ботов с искусственным интеллектом.

Открытый доступ к TensorFlow привёл к формированию сообщества, которое активно участвует в совершенствовании библиотеки. Благодаря этому, TensorFlow постоянно развивается, повышается его эффективность и появляются новый возможности.

**2 Архитектура TensorFlow**

Архитектура TensorFlow состоит из нескольких ключевых компонентов, обеспечивающих удобство разработки и высокую производительность вычислений.

**2.1 Основные принципы и концепции**

В основе работы TensorFlow лежит концепция построения вычислительного графа, узлы которого представляют собой математические операции, а рёбра – данные (тензоры), передаваемые между узлами. Такая модель позволяет разбивать задачи на подзадачи, а также выполнять их на разных устройствах параллельно. Это существенно увеличивает скорость и производительность.

Граф направленный и ациклический. Это означает отсутствие циклов, а также то, что данные перемещаются только в одном направлении - от входов к выходам.

Основные элементы вычислительного графа:

* **Операции** - это вычисления, которые происходят в нейросети. Например, сложение, умножение, вычисление ошибок (разницы между реальным и предсказанным значением) и т.д.

В TensorFlow операции создают вычислительный граф, связывая тензоры и создавая нужные вычисления. Например, если мы создаем нейросеть для распознавания изображений, операция будет заключаться в том, чтобы передать изображение через слои нейросети, применяя различные операции с тензорами для вычислений и принятия решения.

* **Тензоры** (Tensors) - это многомерные массивы данных, представляющие входные и выходные значения для операций.

Проще говоря, тензор - это данные, которые мы передаём в нейросеть. Это могут быть массивы, матрицы или более сложные структуры данных.

Например, если ты хочешь обучить нейросеть распознавать изображения, то каждое изображение будет представлено тензором (обычно это матрица чисел, где каждое число — это значение пикселя).

Тензор можно рассматривать как контейнер для данных, который хранит информацию в удобном для вычислений виде.

* **Переменные** - это данные, которые могут меняться в процессе обучения нейросети. Например, веса нейронов в нейросети.

Например, нейросеть пытается учиться и менять свои "настройки" (например, веса нейронов), чтобы делать более точные предсказания. Эти настройки и называются переменными.

* **Плейсхолдеры** - это "контейнеры", в которые мы можем помещать данные во время работы модели.

Например, при построении формулы, еще неизвестно, какие именно числа будут подставляться в ее места. Плейсхолдеры — это как пустые места в формуле, которые ты заполнишь позже конкретными данными.

При обучении модели, плейсхолдеры могут быть использованы для передачи данных обучения (например, изображения) и правильных ответов (меток).

* **Сессии** - это "интерпретаторы", которые выполняют вычисления на основе построенного вычислительного графа.

После того как мы создали граф с операциями, сессия выполняет этот граф и вычисляет результаты.

Можно представить, что мы написали инструкцию, а сессия - это тот, кто будет следовать этой инструкции, выполняя шаги один за другим.

* **Потоки данных** - это путь, по которому данные (тензоры) проходят через модель.

Когда мы запускаем сессию и передаём данные (например, изображения) в нейросеть, данные движутся через слои модели, и каждое вычисление выполняется с помощью операций. Это и есть поток данных.

В процессе обучения поток данных проходит через все слои нейросети, и каждый слой выполняет свои операции (например, находит важные признаки в изображении).

Раньше работа с вычислительными графами в TensorFlow требовала запуска сессий для выполнения операций. Однако в TensorFlow 2.x эту необходимость устранили, что позволило сосредоточиться на логике модели и упростило взаимодействие с данными. Об этом подробнее в пункте 2.4.

### **2.2 Типы данных и форматы тензоров в TensorFlow**

TensorFlow поддерживает множество типов данных. Ключевые типы данных следующие:

* tf.float32 и tf.float64 - типы с плавающей точкой. Типы 32 и 64 обозначают биты, выделенные под каждое число. С типом данных с плавающей точкой можно работать с рациональными числами.
* tf.int8, tf.int16, tf.int32, tf.int64 - целочисленные типы данных на 8, 16, 32 и 64 бита.
* tf.string - тип данных, использующийся для представления строк или текстовых данных.
* tf.bool - тип данных, использующийся для представления булевых значений - True или False.
* tf.complex64 и tf.complex128 - комплексные числа. Состоят из мнимой и действительной частей. Используются в тех отраслях, в которых требуется совершать сложные вычисления, например, в обработке сигналов.
* tf.uint8, tf.uint16, tf.uint32, tf.uint64 - беззнаковые (положительные) целые числа на 8, 16, 32 и 64 бита соответственно.
* tf.float16 / tf.half - тип данных, хранящий числа с плавающей точкой с урезанной точностью, что может быть полезно для сохранения памяти и ускорения обучения моделей, но только при условии, что небольшая потеря точности допустима.
* tf.quint8 - беззнаковый 8-битный целочисленный тип данных, который часто используется в квантовании - процессе, позволяющем сократить объем используемых ресурсов без значительной потери в качестве модели.
* tf.qint8, tf.qint16, tf.qint32 - квантованные версии целых знаковых чисел.
* tf.bfloat16 - тип данных хранит числа с плавающей точкой, как и tf.float32, но с большим диапазоном, чем tf.float16, при этом используя такой же объем памяти.

Тензоры в TensorFlow могут иметь разные размерности (или ранги), которые определяют, сколько направлений (или осей) имеет тензор. Основные форматы:

* 0D (скаляр): Тензор без осей, например, число.
* 1D (вектор): Тензор с одной осью.
* 2D (матрица): Тензор с двумя осями.
* nD (тензор с n осями): Тензор с произвольным числом осей.

### **2.3 Слои**

Слои в TensorFlow — это строительные блоки для создания нейросетей. Каждый слой выполняет определенную задачу, такую как преобразование входных данных, применение функций активации, извлечение признаков и т.д.. В TensorFlow (и особенно в его высокоуровневом API Keras) слои позволяют легко строить и настраивать архитектуры нейросетей.

Типы слоев:

1. Входной слой

Входной слой - это первый слой нейросети, через который проходят все входные данные (например, изображения, текст или другие признаки). Обычно этот слой не выполняет никаких вычислений, а просто задает форму данных, которые будут поступать в сеть.

1. Полносвязный слой

Полносвязный слой - это один из самых распространённых типов слоев в нейросетях. Каждый нейрон в таком слое связан с каждым нейроном предыдущего слоя.

В каждом нейроне происходит вычисление взвешенной суммы входных данных и применение функции активации.

Этот слой обычно используется в многослойных перцептронах и нейросетях для классификации.

1. Слой свёртки

Слой свёртки используется в свёрточных нейросетях, которые обычно применяются для обработки изображений, видео и других двумерных данных.

В свёрточных слоях фильтры проходят по входным данным и выполняют операцию свёртки. Это позволяет модели извлекать важные признаки, такие как края, текстуры и формы на изображении.

В каждом фильтре вычисляется взвешенная сумма части входных данных.

1. Слой подвыборки (слой пулинга)

Слой подвыборки или пулинга используется для уменьшения размерности данных и извлечения наиболее важных признаков.

Он обычно используется после сверточных слоев, чтобы уменьшить размер карты признаков и сделать модель менее подверженной переобучению.

Существует два типа пулинга:

* Максимальный пулинг: выбирает максимальное значение из области.
* Средний пулинг: вычисляет среднее значение из области.

1. Рекуррентные слои

Рекуррентные нейросети используют специальные слои для работы с последовательными данными, такими как текст или временные ряды. Рекуррентные слои "помнят" информацию о предыдущих элементах последовательности, что делает их полезными для задач, связанных с анализом последовательностей.

1. Слой нормализации

Batch Normalization (нормализация по батчу) - это техника, которая помогает нейронной сети обучаться быстро и стабильно. Благодаря ей можно избежать проблем с переобучением и нестабильными результатами.

1. Слой Dropout

Dropout - это метод регуляризации, который случайным образом отключает (зануляет) определенные нейроны в процессе обучения, чтобы предотвратить переобучение.

1. Слой активации

Слой активации - это слой, который применяет математическую функцию к выходу предыдущего слоя. Функция активации определяет, какие нейроны будут активны, а какие - нет.

1. Выходной слой

Выходной слой - это последний слой модели, который отвечает за преобразование результатов в формат, подходящий для задачи. Например, в задачах классификации это могут быть вероятности классов, а в регрессии - одно или несколько числовых значений.

### **2.4 TensorFlow 2.x**

TensorFlow 2.x — это обновлённая версия библиотеки TensorFlow, которая значительно улучшила удобство работы с ней, сделав её более доступной для разработчиков и исследователей. В версии 2.x были сделаны важные изменения и улучшения, которые упростили использование библиотеки. Ниже представлены основные особенности и изменения TensorFlow 2.x.

#### 

#### **2.4.1 *Eager Execution по умолчанию***

В TensorFlow 1.x для выполнения операций нужно было сначала построить вычислительный граф, а затем выполнять его в сессии. Это требовало довольно сложной работы с графами, плейсхолдерами и сессиями.

В TensorFlow 2.x все изменения и операции выполняются сразу, как только они вызываются. Это называется Eager Execution (немедленное выполнение). С помощью Eager Execution TensorFlow выполняет операции сразу, что упрощает код и делает его более читаемым. Это значит, что вместо статического вычислительного графа в TensorFlow 1.x появляется динамический.

Следовательно, если в TensorFlow 1.x нужно было использовать сессии и плейсхолдеры для работы с графом, чтобы запустить вычисления, то в TensorFlow 2.x этот процесс упростился. Сессии больше не нужны, и операции выполняются благодаря Eager Execution.

#### 

#### **2.4.2 Keras *как основной API***

До версии 2.x для создания нейросетей в TensorFlow использовались различные подходы, включая низкоуровневые API. В TensorFlow 2.x Keras стал основным высокоуровневым API для создания и обучения моделей. Keras позволяет значительно упростить работу с нейросетями, предоставляя удобные способы для построения и обучения моделей.

Keras предоставляет две основные возможности:

* Sequential API: Удобен для создания моделей с последовательными слоями.
* Functional API: Более гибкий подход для создания сложных моделей, таких как многоканальные нейросети или модели с несколькими входами/выходами.

#### **2.4.3 *Поддержка TensorFlow Hub и TensorFlow Lite***

* TensorFlow Hub: Это библиотека для переиспользования предварительно обученных моделей. С помощью TensorFlow Hub можно легко загружать и использовать модели, которые были обучены для разных задач, таких как классификация изображений, анализ текста и т. д.
* TensorFlow Lite: Это инструмент для оптимизации моделей TensorFlow и их внедрения на устройствах с ограниченными вычислительными ресурсами, таких как мобильные устройства и встраиваемые системы.

## **3 Применение TensorFlow**

### **3.1** **Области применения**

TensorFlow находит широкое применение в следующих направлениях:

* Обработка изображений и видео: Распознавание объектов, классификация изображений, анализ видеопотоков, генерация эскизов, улучшение качества графики.
* Обработка естественного языка: Автоматический перевод текстов, анализ тональности, генерация текста, синтез речи.
* Рекомендательные системы: Персонализация контента и товаров.
* Медицинская диагностика: Анализ медицинских изображений и прогнозирование заболеваний.

**3.2 Примеры реализации**

Примеры реализации:

* TensorFlow обеспечивает мобильное подтверждение покупки для своей программы лояльности в Coca-Cola.
* Spotify персонализирует рекомендации для пользователей с помощью TensorFlow.
* Twitter использовал TensorFlow для создания своей «ранжированной временной шкалы», сортирующей твиты.
* WPS Office: интеллектуальный офис на базе TensorFlow WPS Office реализует несколько бизнес сценариев, таких как распознавание изображений на устройстве и оптическое распознавание изображений на основе TensorFlow.

## **4** **Преимущества и ограничения**

Преимущества:

* Мощность и гибкость: TensorFlow позволяет строить как простые, так и очень сложные модели машинного обучения. Вы можете легко адаптировать его под разные задачи.
* Оптимизация для разных устройств: TensorFlow поддерживает обучение на GPU (графических процессорах) и TPU (тензорных процессорах), что позволяет ускорить вычисления и обработку больших данных.
* Поддержка распределенных вычислений: TensorFlow поддерживает обучение на нескольких машинах или устройствах, что полезно при работе с очень большими объемами данных.
* Широкое сообщество и поддержка: TensorFlow активно развивается и поддерживается огромным сообществом разработчиков по всему миру.

Ограничения:

* Сложность для новичков: Несмотря на улучшения в TensorFlow 2.x, для начинающих разработчиков освоение библиотеки может быть сложным, особенно если они не знакомы с математикой и концепциями машинного обучения.
* Ресурсоемкость: Обучение больших моделей требует значительных вычислительных ресурсов.
* Конкуренция: Хотя TensorFlow очень популярен, есть и другие библиотеки, такие как PyTorch, которые также активно развиваются и могут быть удобнее для некоторых задач.

# 

# **Вывод**

TensorFlow - это мощная и гибкая библиотека, которая делает доступным использование искусственного интеллекта для широкого круга разработчиков и исследователей. С ее помощью можно решать задачи машинного обучения и глубокого обучения, создавая нейросети для распознавания изображений, обработки текста, создания рекомендательных систем и многого другого.

Благодаря изменениям в TensorFlow 2.x, библиотека стала гораздо проще в использовании, и теперь даже начинающие разработчики могут эффективно работать с ней. Несмотря на определенные сложности, TensorFlow продолжает быть одним из самых популярных инструментов для создания моделей машинного обучения.

# 

# **Список литературы**

1. TensorFlow - Официальный сайт. Дата обращения: 02.03.2025. URL: https://www.tensorflow.org/
2. Библиотека глубокого обучения TensorFlow / Хабр. Дата обращения: 02.03.2025. URL: https://habr.com/ru/companies/ods/articles/324898/
3. TensorFlow: что это за библиотека, как ее установить и создать свою нейросеть / Skillbox Media. Дата обращения: 02.03.2025. URL: https://skillbox.ru/media/code/biblioteka-tensorflow-pishem-neyroset-i-izuchaem-printsipy-mashinnogo-obucheniya/
4. “Машинное обучение и TensorFlow”, Шакла Нишант. Издательство: ПИТЕР, 2019.
5. Подробный гайд по TensorFlow: немного теории, обзор и основные элементы - Code Lab. Дата обращения: 02.03.2025. URL: https://codelab.pro/podrobnyj-gajd-po-tensorflow-nemnogo-teorii-obzor-i-osnovnye-elementy/