Оглавление

[Оглавление 1](#_Toc421305867)

[Введение 2](#_Toc421305868)

[Глава 1. Существующие форматы хранения нейронных сетей 3](#_Toc421305869)

[1.1. Форматы для теоритического использования 3](#_Toc421305870)

[1.1.1. PMML 3](#_Toc421305871)

[1.1.2. PFA 3](#_Toc421305872)

[1.2. Форматы для практического использования 3](#_Toc421305873)

[1.2.1. ONNX 3](#_Toc421305871)

[1.2.2. TorchScript 3](#_Toc421305871)

[1.2.3. NAS-lib 3](#_Toc421305871)

[1.3. Требования к новому формату 3](#_Toc421305873)

[Выводы по главе 1 3](#_Toc421305874)

[Глава 2. Теоретическое исследование 4](#_Toc421305875)

[Выводы по главе 2 4](#_Toc421305876)

[Глава 3. библиотека FAMLINN 5](#_Toc421305877)

[3.1. Реализация чего-то, предложенного в главе 2 5](#_Toc421305878)

[3.2. Сравнение с аналогами 5](#_Toc421305879)

[3.3. Описание внедрения 5](#_Toc421305880)

[Выводы по главе 3 5](#_Toc421305881)

[Заключение 6](#_Toc421305882)

[Библиографический список 7](#_Toc421305883)

[Приложения 8](#_Toc421305884)

[Приложение 1. Пример 8](#_Toc421305885)

[Приложение 2. Еще пример 8](#_Toc421305886)

Введение

Машинное обучение – активно развивающееся направление науки и технологий. Одно из направлений, получивших широкий спектр применимости и высокие показатели эффективности – нейронные сети. На сегодняшний день нейронные сети используются во всех без исключения высокотехнологичных производствах, крупных коммерческих проектах и научных исследованиях.

Нейронная сеть – это модель вычислений, прототипом которой стала модель человеческого мозга. Последовательность нейронов(аксонов), соединённая ребрами(синопсисами), позволяет человеку учиться, творить и мыслить. Первые попытки формализовать человеческий мозг математической моделью были предприняты У. Маккалоком и У. Питтсом в 1943 году. В дальнейшем модели были существенно усовершенствованы, и человечество, получив вычислительные возможности, смогли реализовать данные модели и получать от них практические результаты.

С развитием вычислительных возможностей наука получила возможность не просто реализовывать конкретную нейронную сеть, но и различными способами улучшать её параметры и архитектуру. Данная область машинного обучения является сравнительно новой и называется автоматическое машинное обучение. Однако существующие форматы хранения нейронных сетей были приспособлены для фиксированной архитектуры, и не были приспособлены для её изменения в исследовательских или оптимизационных целях.

Цель данной дипломной работы – разработать формат хранения нейронных сетей для использования в задачах автоматического машинного обучения. Данный формат получил название FAMLINN – Formatted Auto Machine Learning Iterating Neural Network

Задачи:

- реализовать библиотеку FAMLINN на языке Python

- создать алгоритм получения сети из фреймворка PyTorch

- реализовать сохранение сети для хранения и передачи по сети

- иметь возможность менять архитектуру сети

В первой главе будут описаны существующие форматы хранения нейронных сетей. Во второй главе описываются идея и технические решения библиотеки FAMLINN. И в третьей главе будут приведены результаты сравнения с существующими форматами, а также бенчмарки эффективности

# Существующие форматы хранения нейронных сетей

В данной главе будут рассмотрены способы представления нейронных сетей. Они будут разделены согласно их применимости для решения практических и теоретических задач

## Форматы для теоретического использования

Как будет показано ниже, данные форматы разрабатываются исключительно для исследовательских целей, и их изучение в рамках приведенной работы требуется для общего понимания контекста и способов взглянуть на доменную область под новым ракурсом.

### PMML

Формат хранения моделей вычислений, базирующийся на XML структуре. Используется для описания моделей нейронных сетей и формата данных для них. Содержит:

1. Заголовок – общая информация о модели и её применимости
2. Словарь данных – описывает типы всех переменные и константы в модели
3. Преобразование данных – описывает набор действий для подготовки входных данных
4. Модель – описание архитектуры, алгоритма и параметров
5. Схема данных – список всех значений переменных и констант
6. Результат – описание процедуры интерпретации выхода модели и значений целевой функции

PMML является форматом результатов исследований, однако не является исполняемым, что не дает возможности применить его на практике

### PFA

Формат хранения моделей вычислений, использующийся для задач машинного обучения и обработки данных. Представляет из себя JSON файл с описанием:

1. Входных данных
2. Выходных данных
3. Списка действий в формате псевдокода
   1. Объявление переменных и присваивания
   2. Циклы, операторы ветвления
   3. Создание и вызов функций

Данный формат хорошо подходит для понятного изложения алгоритмов работы с данными, но не применим на практике, так как не является исполняемой программой, а лишь её описанием

## Форматы для практического использования

### ONNX

Открытый стандарт хранения нейронных сетей, поддерживаемая в том числе такими компаниями как Facebook и Microsoft, созданный для обмена обученными моделями. Так как модели могут быть обучены с использованием разных технологий, то формат имеет внутри себя стандартизированное множество вычисляемых функций, которые используются в графе вычислений, который хранится в явном виде.

Сама модель сохраняется на жесткий диск в Protobuf файлах. Модель содержит в себе граф, в каждой вершине которого хранятся вычислимая функция её обученные параметры. Стоит отметить, что вершины графа хранят в себе ссылку на следующую вершину, то есть ребра графа вычислений представлены списком смежности.

Данный формат позволяет эффективно вычислять значение целевой функции, что является его основной задачей, так как он разработан для коммерческого использования, и на него накладываются требования по простоте использования. В том числе есть расширения и дополнительные библиотеки, основанные на ONNX, позволяющие конструировать и соединять графы. Основная мотивация данных преобразований – это комбинация существующих моделей и дополнительная обработка данных перед и после подачи в модель.

### TorchScript

Формат хранения нейронных сетей, используемый в библиотеке *PyTorch.* Ключевой особенностью библиотеки является её динамичность, что создает дополнительные проблемы при получении архитектуры сети в явном виде.

Библиотека позволяет создавать сети следующим образом. Создается класс, в котором есть конструктор и функция вычисления. В конструкторе создаются модули нейронной сети. Модули бывают как примитивами – применением заранее известной функции к данным, так и другими, более сложными модулями. Когда определены составные части сети, в функции вычисления пишется код вычисления, согласно которому будут проведены преобразования над данными. Динамичность заключается в том, что способ исполнения становится известен только в момент вызова. Это логично ложится на концепцию интерпретируемости языка Python. Ключевой недостаток данного подхода применительно к NAS – информация о способе вычисления представлена в последовательности вызова функций. Кроме того, эта последовательность даже может зависеть от аргументов функции вычисления.

### NAS-lib

Формат хранения нейронных сетей, используемый в одноименно библиотеке для решения задач поиска архитектуры сети. Более правильно назвать это подходом к получению сети, поскольку как таковой формат отсутствует. В данной библиотеке используется алгоритм генерации PyTorch сети на основании шаблонного графа.

Для того, чтобы задать пространство поиска и оптимизации архитектур, в библиотеке создается граф, на котором запускают комбинаторные алгоритмы для выбора соответствующих вершинам функций. После задания этих функций производится генерация PyTorch сети. Как было описано выше, к модулю добавляются составные модули архитектуры – вершины шаблонного графа, а затем генерируется функция вычисления, согласно топологической сортировке графа. Известно, что на шаблонный граф, как и на нейронную сеть, накладываются требования на ацикличность, что и позволяет делать обход в глубину с топологической сортировкой.

## Требования к новому формату

Таким образом, появляются возможности совместить существующие наработки для выделения формата, пригодного конкретно к работе с самой архитектурой.

Прежде всего, для применения в задачах автоматического машинного обучения архитектура сети должна уметь изменяться не просто на уровне константных параметров функций, но и сами функции должны быть изменяемы. Более того, между парой слоёв нейронов требуется уметь вставлять слои, и должна быть возможность удалить слой.

Также формат должен позволять сохранять веса обученной модели для практического применения, и иметь возможность конвертации в библиотеки машинного обучения.

Выводы по главе 1

В данной главе были рассмотрены основные подходы к хранению нейронных сетей

# новый подход к хранению архитектуры

В этой главе будет предложен новый подход к хранению нейронных сетей, их архитектуры и параметров. Мы рассмотрим требуемый функционал формата, и алгоритмы работы с ним

## Явный граф вычислений

Пусть есть модель нейронной сети

Выводы по главе 2

1. Разделы выводов не нумеруются.

# Библиотека FAMLINN

В данной главе мы рассмотрим реализацию данного подхода в библиотеке *FAMLINN*. Данная библиотека реализована на *Python 3.9* и имеет возможность работы с библиотекой машинного обучения *PyTorch 1.11.0*

## Особенности реализации

Для автоматизации построения модели данных требуется разработать формальный подход, позволяющий построить модель данных по реализации алгоритма. Однако в общем случае это трудно решаемая задача.

Тут про приколы с тензорами и танцы с бубном то есть оннх

## Сравнение с аналогами

Тут про то что смогли сети бахнуть

Тут бенчмарки по времени\памяти

## Описание внедрения

Тут про то как внедрились

Выводы по главе 3

В данной главе были приведены детали реализации предложенного подхода. Также описаны полученные результаты, проведены сравнительные анализы вычислений с другими форматами по времени исполнения и объему занимаемой памяти.

Заключение

В данной работе был предложен и реализован подход хранения нейронных сетей с учетом возможности работы с архитектурой. Данный формат работы позволяет добиться сопоставимых по эффективности потребления ресурсов возможностей для хранения, при этом предоставляет расширенный функционал.

В новом формате сочетается простота хранения архитектуры для исследований, возможность работы с весами и параметрами, а также возможность вычисления целевой функции для прикладного использования. А также формат хорошо подходит для конвертации между различными способами представления нейронных сетей, так как имеет своей основой простую абстрактную модель.

Библиографический список

1. https://en.wikipedia.org/wiki/Portable\_Format\_for\_Analytics
2. https://en.wikipedia.org/wiki/Predictive\_Model\_Markup\_Language
3. https://en.wikipedia.org/wiki/Artificial\_neural\_network

Приложения

Приложение 1. Потребление ресурсов

Тут табличек

Приложение 2. Алгоритм конвертации из PyTorch

Вот так.