СОПОСТАВЛЕНИЕ ПРОГРАММНЫХ РЕАЛИЗАЦИЙ, АЛГОРИТМЫ И ИНСТРУМЕНТЫ НАСТРОЙКИ НЕЙРОННЫХ СЕТЕЙ

Таранчук В. Б.
Кафедра компьютерных технологий и систем, Белорусский государственный университет
Минск, Республика Беларусь
E-mail: taranchuk@bsu.by

Будут обсуждаться итоги комплексного сравнительного анализа современных подходов к построению, обучению и оптимизации нейронных сетей на примере задачи сглаживания ряда данных (представительный набор данных наблюдений при наличии шумов, помех, выбросов, потерь). Рассмотрены не только теоретические аспекты архитектуры и обучения моделей, но и практические вопросы автоматизации поиска оптимальных параметров с использованием двух различных технологических платформ – Wolfram Mathematica и Python (TensorFlow, Keras).

Введение

Искусственный интеллект становится неотъемлемой частью современного мира – от автоматизации производства и диагностики заболеваний до создания транспортных систем, анализа финансовых рынков. Одним из наиболее заметных прорывов становятся искусственные нейронные сети (ИНС), которые уже не воспринимаются, как абстрактная технология из научных публикаций, а активно используются в повседневной жизни.

Параллельно с ростом популярности нейронных сетей быстро эволюционируют программные инструменты для их разработки, модификации и тестирования. Специалистам доступны универсальные вычислительные системы компьютерной математики, например, система компьютерной алгебры Wolfram Mathematica (WM) [1], гибкие фреймворки, в частности, на языке Python ([2]): TensorFlow, Keras, PyTorch, а также специализированные библиотеки для автоматизации подбора гиперпараметров – Optuna, Ray Tune и другие. Выбор подходящей экосистемы становится самостоятельной задачей, от которой зависит успех внедрения интеллектуальных технологий в промышленность, медицину, транспорт, сферу цифровых сервисов.

I. Инструменты Mathematica для построения и настройки ИНС.

WM предоставляет множество инструментов для удобного построения и настройки нейронных сетей без необходимости вручную программировать каждую деталь. В состав Mathematica входит модуль Neural Networks, который позволяет разработчикам использовать простой и понятный синтаксис для быстрого создания многослойных нейросетевых моделей практически любой сложности.

Конструкция нейронной сети в Mathematica строится декларативно, путём последовательного описания слоёв. Важной особенностью WM Neural Networks является наличие широкого набора базовых и специализированных слоёв. Пользователь может создавать последовательные и разветвлённые композиции слоёв, комбинируя: LinearLayer – для выполнения линейных преобразований; PoolingLayer – для уменьшения размерности и повышения устойчивости к шуму; ElementwiseLayer и BatchNormalizationLayer – для реализации поэлементных и нормализующих операций; ConvolutionLayer – для выделения пространственных признаков в изображениях; ReLU, Tanh, Softmax и другие функции активации – для внедрения нелинейности и корректного формирования выходного сигнала.

Система WM поддерживает различные форматы входных и выходных данных. Благодаря встроенным кодировщикам и декодерам, сеть может работать не только с числовыми векторами, но и с изображениями, текстами, аудиосигналами и их комбинациями. Дополнительным преимуществом Neural Networks WM является наличие большого числа предобученных моделей и специализированного Wolfram Neural Net Repository, где представлены архитектуры для решения самых разных задач. Пользователь может импортировать готовую сеть, адаптировать её под собственные данные, визуализировать структуру и ход работы, а также анализировать метрики качества интерактивно.

После того как архитектура нейронной сети определена и подготовлены данные, следующим шагом становится непосредственно обучение модели. В WM для этого используется функция NetTrain – позволяет запустить процесс обучения, указав обучающий набор данных и нужные параметры оптимизации. В NetTrain пользователь задаёт количество эпох (число полных проходов по всем обучающим данным), размер мини-пакета данных (batch size), а также выбирает устройство, на котором будет выполняться обучение (например, центральный процессор или графическая видеокарта). Чтобы контролировать качество обучения и избежать проблемы переобучения, можно указать дополнительный набор данных (валидационный) для оценки про-

межуточных результатов. Замечательной особенностью Neural Networks WM является встроенная визуализация процесса обучения. В режиме реального времени WM показывает, как изменяется ошибка, как растёт точность модели по мере её обучения. Пользователь может остановить обучение, скорректировать архитектуру нейросети или изменить гиперпараметры.

Настройка и оптимизация геперпараметров в WM. Одним из важнейших этапов в обучении нейронных сетей является настройка гиперпараметров, от которых напрямую зависит точность модели, скорость сходимости и общая устойчивость к переобучению. Гиперпараметры, в отличие от параметров модели, не обучаются напрямую во время градиентного спуска, а определяются до начала процесса обучения. В WM реализация настройки гиперпараметров осуществляется с использованием встроенных возможностей функции NetTrain, а также при помощи процедурного программирования и визуализации. WM предоставляет пользователю прямой доступ к наиболее значимым гиперпараметрам в виде опций, передаваемых в функцию NetTrain. В частности, к числу часто настраиваемых параметров относятся: скорость обучения (LearningRate), тип оптимизатора (Method), размер минипакета (BatchSize), количество эпох (MaxTrainingRounds), а также состав и доля валидационного набора (ValidationSet). Mathematica предоставляет пользователю богатый инструментарий для самостоятельного построения процедурного поиска оптимальных значений. Для симуляции методов Grid Search и Random Search могут использоваться такие конструкции языка, как Table, RandomReal, Map, NestList и другие итеративные средства. Для оценок насколько успешно обучилась нейронная сеть, можно использовать предоставляемые специальные метрики качества. В WM для этих целей предусмотрена функция NetMeasurements, используя которую можно легко вычислить основные важные показатели, включая: точность, среднеквадратичная ошибка, кросс-энтропия.

Также Mathematica предоставляет полезную функцию NetInformation. С её помощью можно получить подробную информацию об активной модели: сколько в ней параметров, какой её размер, какую точность она показывает на обучающих и валидационных данных. Конечно же, в контексте простых моделей возможно использование стандартных WM функций оптимизации (FindMinimum, NMinimize, MachineLearningModelParameterOptimization).

Важно и следует напомнить, что пользователи WM могут комбинировать с внешними средами (в частности, Python) через интерфейс ExternalEvaluate, WolframScript или REST API, можно подключать сторонние оптимизаторы и расширять функциональность системы.

II. Python – о средствах настройки и оценки ИНС

Язык программирования Python занимает ведущие позиции благодаря своей универсальности, лаконичному синтаксису и огромному количеству специализированных библиотек. Среди наиболее популярных и востребованных библиотек для работы с нейронными сетями на языке Python выделяют TensorFlow, Keras, PyTorch и Theano. В данной работе внимание уделено библиотекам TensorFlow и Keras, так как их инструменты позволяют быстро создавать нейронные сети разного уровня сложности, оперативно проводить эксперименты и прототипирование, эффективно применять готовые решения в задачах обработки и анализа больших объёмов данных.

На примерах рассматриваемой в работе задачи для повышения эффективности и качества моделей в TensorFlow реализованы инструменты автоматического поиска оптимальных архитектур и гиперпараметров, в том числе реализована интеграция с модулями Keras Tuner и Optuna. Отмечено, как пользователь может экспериментировать с различными конфигурациями сети, автоматически подбирать размер слоёв, функции активации, оптимизаторы и другие параметры без необходимости вручную запускать десятки экспериментов. Примером поясняется выбор гиперпараметров с использованием Optuna, которая основана на методах байесовской оптимизации и реализует улучшенный алгоритм поиска – Tree-structured Parzen Estimator (TPE), позволяющий эффективно исследовать пространство гиперпараметров.

Заключение

Сопоставление результатов решения задачи устранения шумов [3], вычислительных экспериментов проведенных в двух рассмотренных средах показали, что всегда точность решения конкретной задачи зависит не только от выбранной платформы, но и от характеристик искажений (шума) исходных данных, а также от метода оптимизации гиперпараметров. Вместе с тем, при правильной настройке обе исследуемые платформы (Wolfram Mathematica и Python с библиотеками TensorFlow, Keras и Optuna) способны стабильно показывать хорошие результаты, хотя, не полностью совпадающие.

III. Список литературы

1. Neural Networks in the Wolfram Language [Electronic resource] / Wolfram Documentation. – Mode of access: `https://reference.wolfram.com/language/tutorial/NeuralNetworksOverview.html/`. – Date of access: 29.09.2025.
2. Шолле, Ф. Глубокое обучение на Python. 2-е издание / Франсуа Шолле – СПб. : Питер, 2023. – 1041 с.
3. Таранчук, В. Б. Средства и примеры интеллектуальной обработки данных для геологических моделей / В. Б. Таранчук // Проблемы физики, математики и техники. – 2019. – № 3 (40). – С. 117–122.