УДК 512.6, 517.9, 519.6

МОДЕЛИ КВАНТОВОГО ГЛУБОКОГО МАШИННОГО ОБУЧЕНИЯ: ПРОГРАММНО-АЛГОРИТМИЧЕСКАЯ ПЛАТФОРМА КВАНТОВОГО «СИЛЬНОГО» ВЫЧИСЛИТЕЛЬНОГО ИИ

Боровинский Виталий Викторович[1], Капков Роман Юрьевич[2], Решетников Андрей Геннадьевич[3], Тятюшкина Ольга Юрьевна[4], Ульянов Сергей Викторович[5], Решетников Геннадий Павлович[6]

[1]Предприниматель, инженер, MBA;
Московский государственный технический университет им. Н.Э. Баумана;
Россия, 105005, г. Москва, ул. 2-я Бауманская, 5, с. 1;
e-mail: vitali.borovinsky@yandex.ru.

[2]Предприниматель, преподаватель-исследователь;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: info@kapkov.pro.

[3]Кандидат технических наук, доцент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
Старший научный сотрудник;
Объединенный институт ядерных исследований;
Россия, 141980, Московская обл., г. Дубна, ул. Жолио-Кюри, 6;
e-mail: agreshetnikov@gmail.com.

[4]Кандидат технических наук, доцент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: tyatyushkina@mail.ru.

[5]Доктор физико-математических наук, профессор;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
Главный научный сотрудник;
Объединенный институт ядерных исследований;
Россия, 141980, Московская обл., г. Дубна, ул. Жолио-Кюри, 6;
e-mail: ulyanovsv46_46@mail.ru.

[6]Кандидат физико-математических наук, доцент
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
Старший научный сотрудник;
Объединенный институт ядерных исследований;
Россия, 141980, Московская обл., г. Дубна, ул. Жолио-Кюри, 6;
e-mail: genresh@mail.ru.

Рассмотрена цель машинного обучения состоит в том, чтобы обучить компьютер извлекать определенные свойства из заданного набора данных без явного кодирования или набора правил, а затем использовать полученные результаты для изучения этих свойств в новых данных в целях прогнозирования, классификации или построения модели исследуемого объекта. Обсуждаются наиболее популярные модели машинного обучения такие как контролируемое обучение (supervised learning) или обучение с учителем, при котором машина предварительно обучается с использованием некоторых помеченных данных. Другие формы обучения, такие как неконтролируемое и с подкреплением (unsupervised and reinforced), также широко изучались и

применялись в различных областях. Тремя наиболее широко используемыми алгоритмами контролируемого машинного обучения, относящимися к квантовым вычислениям, являются (а) нейронная сеть (NN – neural networks) для синтеза квантовой логики, физического отображения и декодирования квантовых ошибок, протокол QKD (quantum key distribution), квантовый ускоритель ML, квантовые нейронные сети (QNN – quantum neural networks); (б) Обучение с подкреплением (RL) для декодирования квантовых ошибок и (в) Метод опорных векторов (или SVM – support vector machine) для квантового машинного обучения. В исследовании также обсуждаются различные модели обучения ML, включая метод случайного поиска для квантовой коммуникации. Работа рассчитана на повышение квалификации ИТ – специалистов, применяющих методы «сильного» ИИ.

Ключевые слова: ускоритель для квантового машинного глубокого обучения, квантовая нейронная сеть, вариационные квантовые схемы, квантовое машинное обучение, алгоритм Гровера.

Для цитирования:
Модели квантового глубокого машинного обучения: программно-алгоритмическая платформа квантового «сильного» вычислительного ИИ / В. В. Боровинский, Ю. Р. Капков, А.Г. Решетников [и.др.] // Системный анализ в науке и образовании: сетевое научное издание. 2025. № 1. С. 23-64. EDN: JJXGPN. URL: https://sanse.ru/index.php/sanse/article/view/649.

MODELS OF QUANTUM DEEP LEARNING: SW ALGORITHMIC TOOLKIT OF QUANTUM “STRONG” COMPUTATIONAL AI ДЛЯ ИТ - НАЧИНАЮЩИХ

Borovinsky Vitalii V.[1], Kapkov Roman Yu.[2], Reshetnikov Andrey G.[3], Tyatyushkina Olga Yu.[4], Ulyanov Sergey V.[5], Reshetnikov Gennady P.[6]

[1]Businessman, engineer, MBA;
Bauman Moscow State Technical University;
5, bld. 1, 2nd Bauman str., Moscow, 105005, Russia;
e-mail: vitali.borovinsky@yandex.ru.

[2]Businessman, teacher-researcher;
Dubna State University,
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: info@kapkov.pro.

[3]PhD in Engineering sciences, associate professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
Senior Researcher;
Joint Institute for Nuclear Research;
6 Joliot-Curie Str., Dubna, Moscow region, 141980, Russia;
e-mail: agreshetnikov@gmail.com.

[4]PhD in Engineering sciences, associate professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: tyatyushkina@mail.ru.

[5]Grand PhD in Physical and Mathematical Sciences, professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
Chief Researcher;
Joint Institute for Nuclear Research;
6 Joliot-Curie Str., Dubna, Moscow region, 141980, Russia;
e-mail: ulyanovsv46_46@mail.ru.

[5]PhD in Physical and Mathematical Sciences, associate professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
Senior researcher;

Joint Institute for Nuclear Research;
6 Joliot-Curie Str., Dubna, Moscow region, 141980, Russia;
e-mail: genresh@mail.ru.

The goal of machine learning (ML) is to train a computer to extract certain properties from a given data set without explicit coding or a set of rules, and then applying the results to learn these properties from new data for the purpose of prediction, classification, or developing a model of the object under study. The most popular models of machine learning are discussed, such as supervised learning, or learning with a teacher, in which the machine is pre-trained using some labeled data. Other forms of learning, such as unsupervised and reinforced, have also been widely studied and applied in various fields. Three of the most widely used supervised machine learning algorithms related to quantum computing are (a) neural networks (NN) for quantum logic synthesis, physical mapping and quantum error decoding, quantum key distribution (QKD) protocol, quantum ML accelerator, quantum neural networks (QNN); (b) reinforcement learning (RL) for quantum error decoding and (c) support vector machine (SVM) for quantum machine learning. The study also discusses various ML learning models including random search method for quantum communication. The work is intended to improve the skills of IT specialists applying "strong" AI methods.

Keywords: accelerator for quantum deep learning, quantum neural network, variational quantum circuits, quantum machine learning, Grover's algorithm.

For citation:
Borovinsky Vitalii V., et al. Models of quantum deep learning: SW algorithmic toolkit of quantum "strong" computational AI. *System analysis in science and education*, 2025;(1):23-64 (in Russ). EDN: JJXGPN. Available from: https://sanse.ru/index.php/sanse/article/view/649.

Введение

Классические модели машинного обучения (*ML – machine learning*) как составляющая ИИ [1] применялись в различных областях квантовых вычислений [2-12], таких как квантовая коррекция ошибок, квантовая коммуникация, квантовая криптография, сопоставление квантовых алгоритмов с классическими алгоритмами в машинном обучении и т.д. С другой стороны, были разработаны квантовые аналоги существующих алгоритмов *ML*, такие как квантовая нейронная сеть (*QNN*), квантовый метод опорных векторов (или квантовая вспомогательная векторная машина – *QSVM – quantum support machine*), которые, как ожидается, будут работать лучше, чем классические компьютеры, из-за экспоненциального увеличения пространства поиска. Более того, были разработаны квантово-классические гибридные алгоритмы [13 - 21], которые в некотором смысле имитируют принцип работы алгоритмов *ML*. Определялся глобальный оптимум для функции затрат, но для этого использовали квантовые компьютеры и обсуждался вопрос соотношения инженерии машинного обучения с квантовыми основами (математическое и концептуальное понимание квантовой теории в построении квантово-подобных, квантово-инспирированных, гибридных квантово-классических и т.п. моделей).

Цель машинного обучения состоит в том, чтобы обучить компьютер извлекать определенные свойства из заданного набора данных без явного кодирования или набора правил, а затем использовать полученные результаты для изучения этих свойств в новых данных в целях прогнозирования или классификации. Например, ожидается, что компьютер, который ранее видел большое количество изображений опухолей и был обучен распознавать злокачественные опухоли, сможет правильно идентифицировать невидимую фотографию опухоли. Этот тип алгоритма *ML* называется контролируемым обучением (*supervised learning*) или обучение с учителем, при котором машина предварительно обучается с использованием некоторых помеченных данных. Другие формы обучения, такие как неконтролируемое и с подкреплением (*unsupervised and reinforced*), также широко изучались и применялись в различных областях.

Тремя наиболее широко используемыми алгоритмами контролируемого машинного обучения, относящимися к квантовым вычислениям, являются (*а*) Нейронная сеть (*NN – neural networks*) для синтеза квантовой логики, физического отображения и декодирования квантовых ошибок, протокол *QKD* (*quantum key distribution*), квантовый ускоритель *ML*, квантовые нейронные сети (*QNN - quantum neural networks*); (б) Обучение с подкреплением (*RL*) для декодирования квантовых ошибок и (в)

Метод опорных векторов (или *SVM – support vector machine*) для квантового машинного обучения. В исследовании также обсуждаются различные модели обучения *ML* [22-27], включая метод случайного поиска для квантовой коммуникации [11,12].

Примечание. Напомним, что однослойная нейронная сеть, построенная на методе обратного распространения ошибки аппроксимации и прямого действия исследуемой функции (сеть *FFNN – feed - forward neural network*) состоит из входного слоя нейронов и выходного слоя нейронов. Задачей нейронной сети является аппроксимация функции с заданной точностью и минимальной потерей информации, содержащейся в исходной функции. В структуре многослойной сети с прямой связью первый уровень является входным уровнем, который принимает входной сигнал, а последний уровень является выходным уровнем. Между этими двумя слоями могут быть несколько скрытых слоев. Сигнал от входного слоя проходит через эти скрытые слои к выходному. Связь между парой узлов (нейронами) в двух соседних слоях имеет соответствующий вес, который указывает на силу связи между ними. Входные данные для определенного слоя умножаются на вес, чтобы получить внутреннее значение, которое изменяется на пороговое значение перед передачей в функцию активации для получения выходных данных этого слоя. Эти выходные данные передаются следующему слою в качестве входных данных. Конечный уровень предоставляет результаты работы сети. На каждой итерации веса и пороговые значения обновляются для получения более точного значения.

Функция потерь должна непосредственно содержать меру расстояния. Однако ее вычисление требует больших вычислительных затрат даже при использовании квантового компьютера. Поэтому используют неявную функцию потерь, полученную из показателя точности, который выражается как

$$l_{\text{fid}}\left(\left(x_i, y_i\right),\left(x_j, y_j\right)\right)=\left[\left|\left\langle x_i \middle| x_j \right\rangle\right|^2-\frac{1}{2}\left(1+y_i y_j\right)\right]^2$$

Эта потеря точности может быть эффективно вычислена с помощью теста *SWAP* или прямого измерения перекрытия состояний (см. рис. 1).

NQE (*Neural Quantum Embedding*) - эффективный метод, который использует возможности классических нейронных сетей для обучения оптимальному квантовому встраиванию для данной задачи. *NQE* может повысить разделимость квантовых данных, превосходящую возможности квантовых каналов, тем самым расширяя фундаментальные пределы квантового обучения под наблюдением. Подход позволяет избежать критических проблем, присущих существующим методам, таких как увеличение числа вентилей и глубины квантовой схемы, а также риск возникновения «бесплодных плато». Численное моделирование и эксперименты с устройствами *IBM* подтверждают эффективность *NQE* в повышении производительности *QML* по нескольким ключевым показателям машинного обучения. Эти улучшения распространяются на точность обучения, способность к обобщению, обучаемость и устойчивость к шуму, превосходя возможности существующих методов квантового встраивания.

Экспериментальные результаты демонстрируют эффективность *NQE* в совершенствовании алгоритмов *QML* [28]. Эксперимент состоит из трех основных этапов: применение *NQE*, обучение *QCNN* (*quantum convolution neural networks*) с использованием моделей *NQE* и без них и оценка точности классификации для обученной квантовой сверточной нейронной сети *QCNN* с использованием моделей *NQE* и без них. Унитарное преобразование, которое отображает x_i в квантовое пространство признаков, определяется выходными данными классической нейронной сети, обозначаемыми как $g\left(x_i, w\right)$, где w представляет обучаемые параметры. Результирующее квантовое состояние имеет вид: $\left|x_i\left(w\right)\right\rangle=V\left(g\left(x_i, w\right)\right)\left|0\right\rangle^{\otimes n}$. Цель обучения - создать функции отображения, которые могут разделить два класса данных на два ортогональных подпространства. Эффективный расчет точности между двумя квантовыми состояниями, полученными с помощью отображения характеристик объектов, выполняется с помощью квантового компьютера.

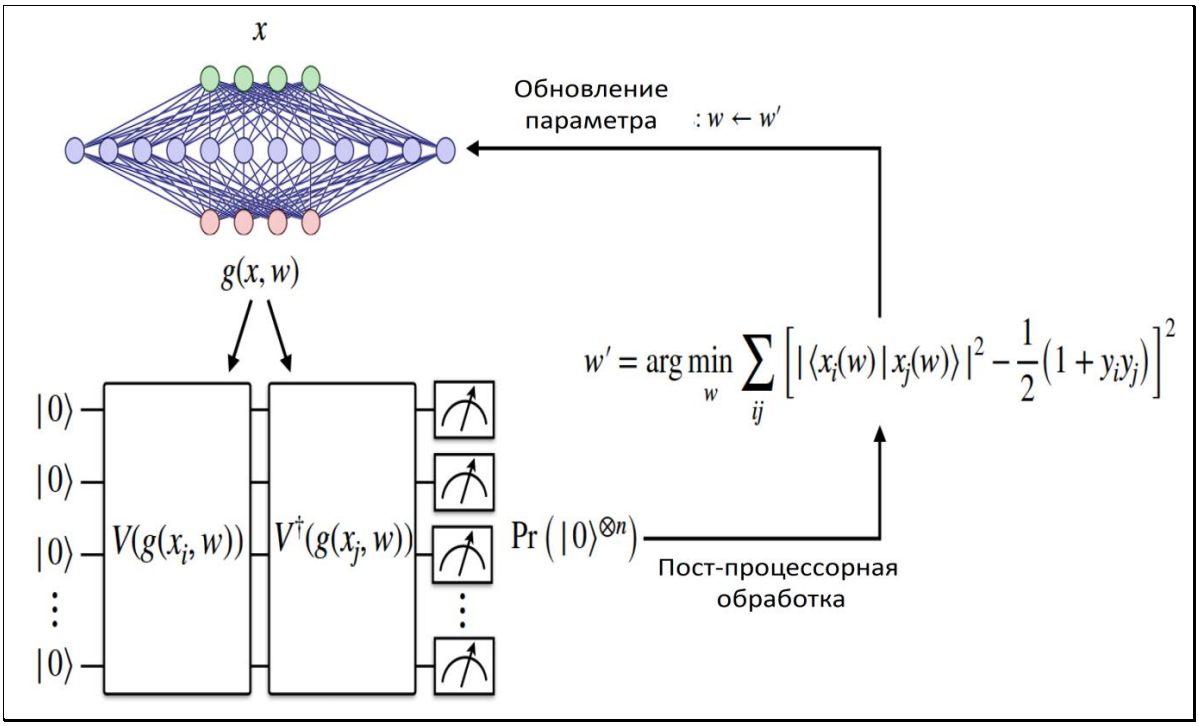
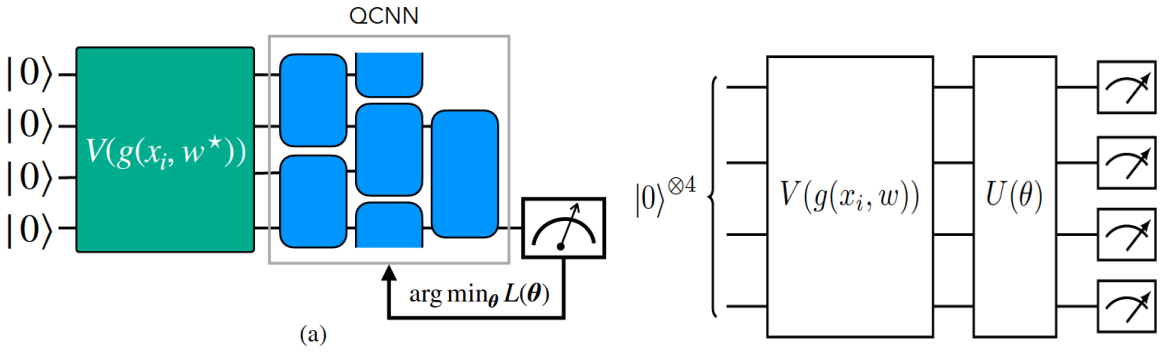


Рис. 1. Схема обучения NQE (Neural Quantum Embedding)

Схематическое изображение квантовой схемы, использованной в экспериментах, приведена на рис. 2.



(а) *(б)*

Рис. 2. (а) Схематическое изображение квантовой схемы, использованной в экспериментах. Зеленая секция указывает на нейронное квантовое встраивание (NQE), которое преобразует классические данные x_i в квантовое состояние $|x_i\rangle$. Синяя часть представляет архитектуру квантовой сверточной нейронной сети (QCNN); (б) Квантовая схема, используемая для оценки локальной эффективной размерности. Отображение характеристик объекта, обозначенное как $V\left(g\left(x_i, w\right)\right)$, воздействует на начальное состояние $|0\rangle^{\otimes 4}$ для кодирования входного вектора x_i. Затем для изменения состояния применяется параметризованное унитарное преобразование $U\left(\theta\right)$, при этом параметры выбираются таким образом, чтобы минимизировать конкретную функцию потерь.

На рис. 3 приведен пример гибридной квантово-классической модели *ML*.

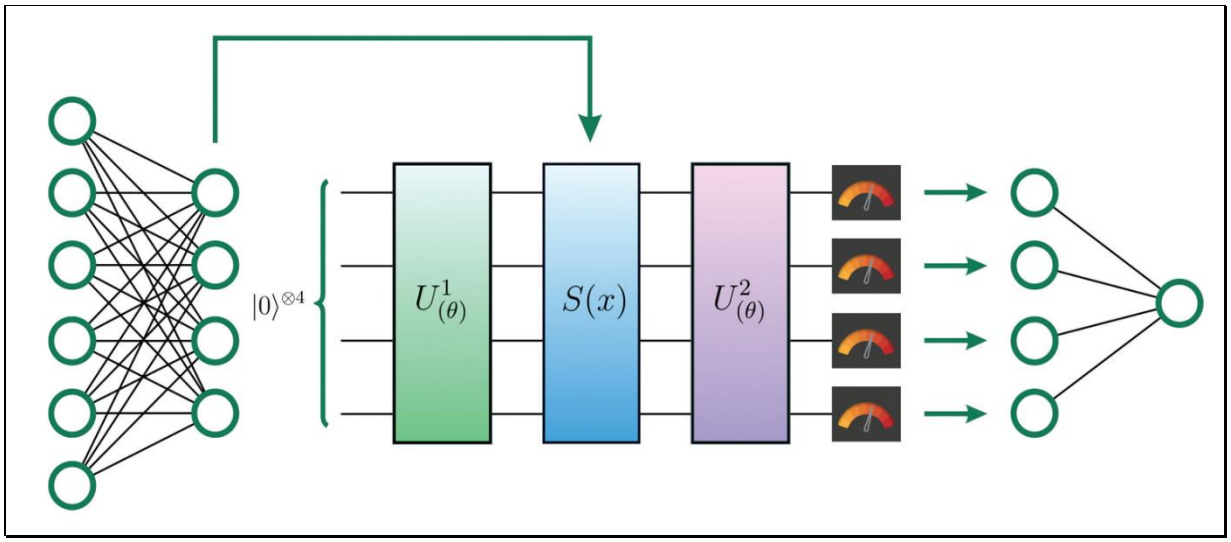


Рис. 3. Пример гибридной квантово-классической модели ML

В этом случае входные данные передаются в полностью подключенный классический многослойный персептрон, а его выходные данные подаются на встраивание квантовой схемы. В зависимости от настройки выполняются некоторые измерения этой квантовой схемы, которые затем передаются на другой полностью подключенный уровень, выходные данные которого можно сравнить с меткой.

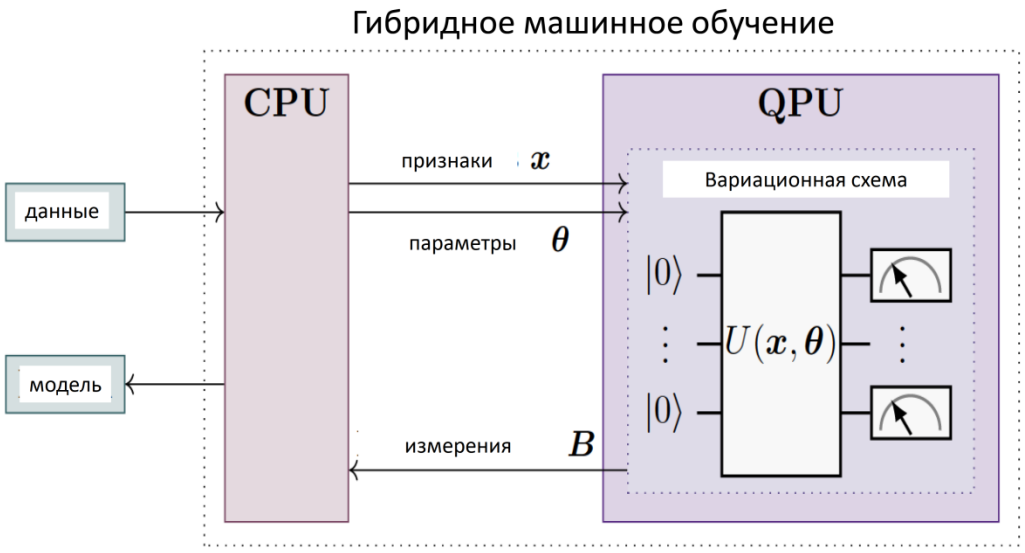
Междисциплинарная область *QML* находится на стыке *ML* и квантовых вычислений. Она охватывает несколько направлений исследований, в которых рассматриваются данные и алгоритмы, которые могут быть классическими, квантовыми или их гибридной комбинацией. Сосредоточимся на квантово-усовершенствованном *ML*, в котором рассматривается использование квантовых алгоритмов для улучшения задач *ML* для классических данных. Обычно это достигается с помощью гибридных алгоритмов, которые включают классическую и квантовую части. Такого рода алгоритмы уже могут быть запущены на современных устройствах *NISQ*.

Структура квантового встраивания включает в себя множество схем-аналогов (рис. 4), что приводит к неразрешимым скрытым представлениям для универсальных квантовых вычислений. На рис. 4 показан упрощенный вариант такого гибридного *ML*-конвейера [29].

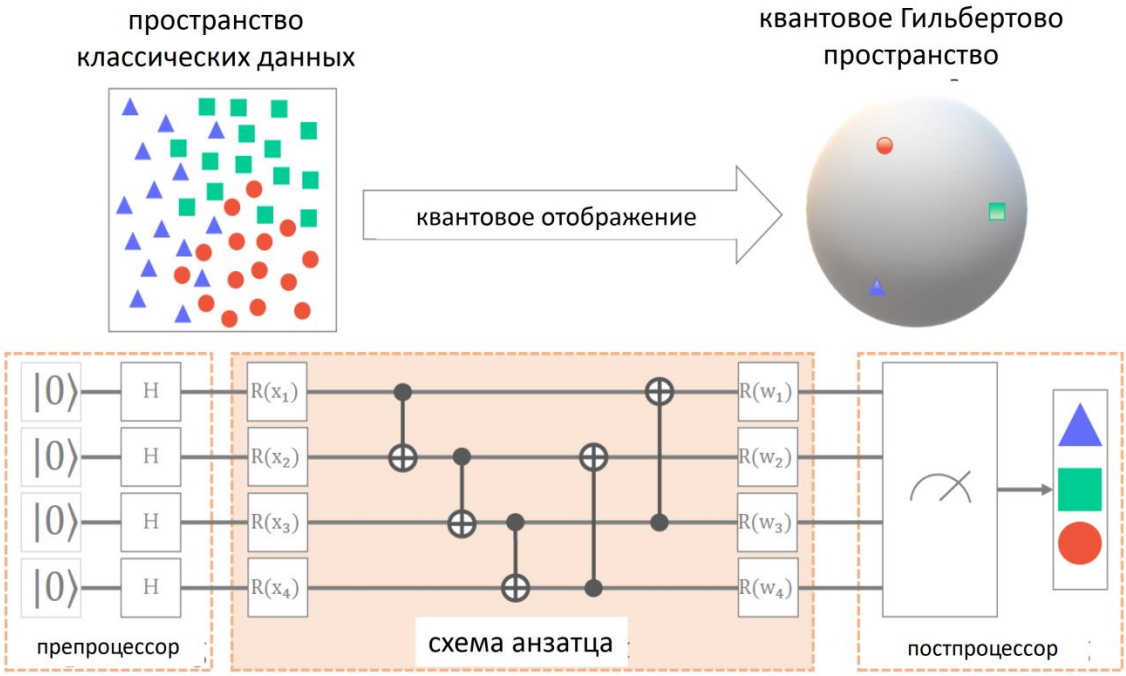
Подобно методам ядра, квантовое вложение преобразует наблюдения в классическом пространстве данных в квантовое гильбертово пространство квантовых состояний, которое может быть представлено внутренним произведением квантовых представлений. Представлена архитектура модели квантового машинного обучения, сформированной из выбранного набора квантовых схем. Схема - анзатц играет решающую роль в модели схемы, обеспечивая веса модели обучения w в соответствии с входными данными x.

Классические данные передаются на классический головной процессор (обычно состоящий из (или более) центральных процессоров (*CPU*), графических процессоров (*GPU*) и оперативной памяти (*RAM*)), который взаимодействует с *QPU*. На *QPU* выполняется квантовая схема. Эта схема управляется функциями и вариационными параметрами классического головного процессора. Признаки в этом контексте относятся к соответствующим образом предварительно обработанным классическим точкам данных, которые могут быть закодированы в схеме с помощью подходящей комбинации элементов таким образом, что результирующее квантовое состояние содержит соответствующую информацию, например, в виде амплитуд или углов.

Предварительно обработанные функции x и параметры классического процессора (представленные здесь центральным процессором) управляют вариационной схемой, которая выполняется на *QPU*. Результаты измерений B возвращаются на классический процессор и позволяют выполнить квантово-классическую оптимизацию параметров. В конечном итоге может быть реализована подходящая модель для предложенной задачи *ML*. В перспективе такой конвейер включает в себя несколько процессоров и *QPU*, а также графические процессоры.



(а)



(б)

Рис. 4. (а) Упрощенный эскиз гибридного конвейера ML в виде вариационного квантового алгоритма; (б) Квантовые вложения для контролируемого квантового машинного обучения

Пример. В машинном обучении значения Шепли используют теорию игр для определения точного вклада каждого игрока. Кроме того, метод Шепли объясняет прогнозы, сделанные с помощью нелинейных моделей. В *Python* функции значений Шепли используются для интерпретации моделей машинного обучения. Значения Шепли (*SV*), которые представляют собой хорошо известный не зависящий от модели инструмент в классическом *ML* для оценки важности каждой характеристики для прогнозирования модели. В частности, предлагается применять *SV* для квантовых вентилей, которые являются строительными блоками моделей *QML* на квантовых компьютерах на основе вентилей. Называется такой подход квантовым Значением Шепли (*QSV*).

На рис. 5 приведены два подхода [30] для *SVS* в *QML*.

В обоих случаях функция значения, определяется результатами измерений. Показана примерная вариационная схема с четырьмя кубитами, состоящая из блока кодирования данных для *k*-мерного

вектора признаков *x* и набора из пяти параметризованных элементов, как показано на рис.5, где каждый параметризованный элемент зависит от вектора параметров.

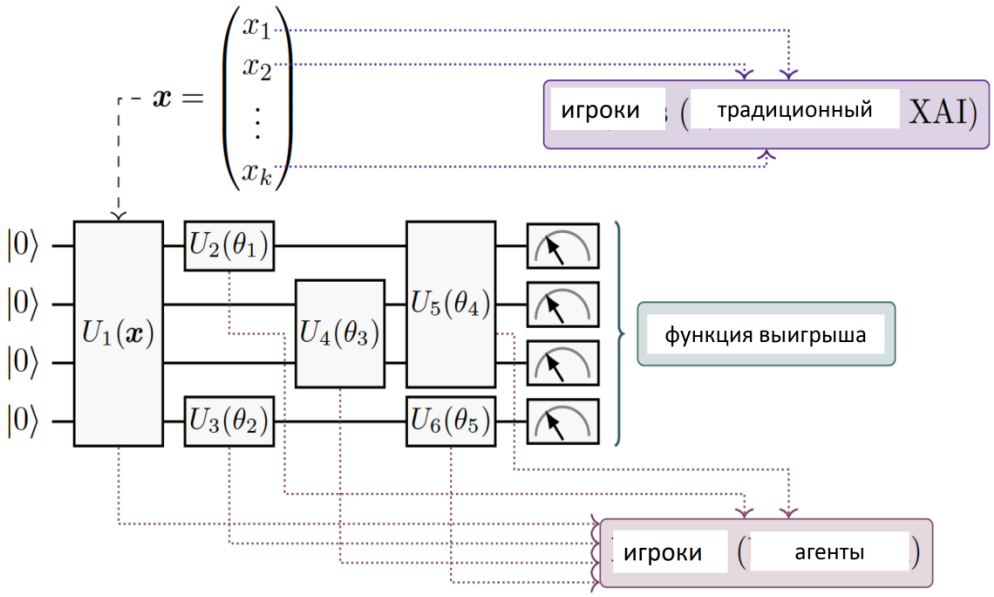


Рис. 5. Два возможных подхода к SV в QML, использующих вариационные квантовые схемы: (а) SV из классического ML, для которых объекты представляют игроков; (б) предложенные QSV, для которых квантовые элементы представляют игроков

Квантовые нейронные сети (*QNN*) - подкласс вариационных квантовых алгоритмов, состоящий из квантовых схем, которые содержат параметризованные логические операции. Информация сначала кодируется в квантовое состояние с помощью процедуры подготовки состояния или отображения свойств. Выбор отображения свойств обычно направлен на повышение производительности квантовой модели и, как правило, не оптимизируется и не обучается. Как только данные кодируются в квантовое состояние, применяется вариационная модель, содержащая параметризованные элементы управления, которая оптимизируется для конкретной задачи. Это происходит за счет минимизации функции потерь, когда выходные данные квантовой модели могут быть извлечены из классической функции пост-обработки, которая применяется к результату измерения. Применяемая модель изображена на рис. 6.

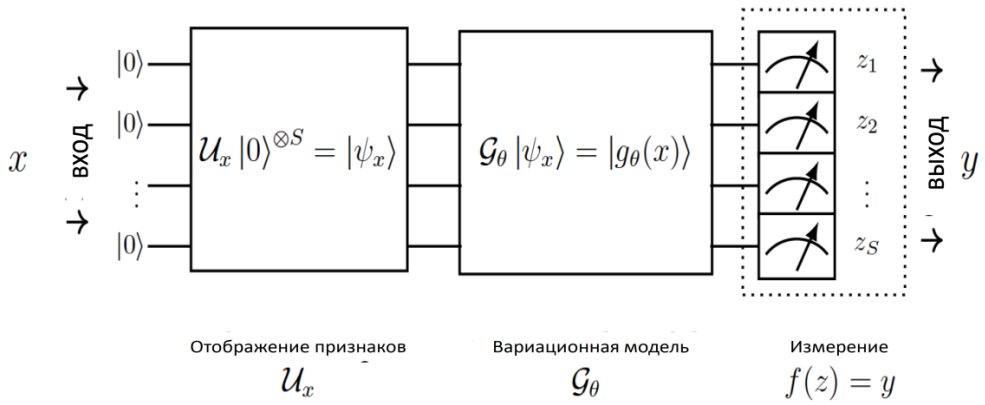


Рис .6. Модель квантовой нейронной сети [21]

Входные данные кодируются в гильбертово пространство *S*-кубита путем применения отображения свойств $|\psi_x\rangle := U_x|0\rangle^{\otimes S}$. Затем сформированное состояние преобразуется с помощью

вариационной формы $\left|g_\theta(x)\right\rangle := \mathrm{G}_\theta \left|\psi_x\right\rangle$, где параметры $\theta \in \Theta$ выбираются таким образом, чтобы минимизировать определенную функцию потерь. Наконец, выполняется измерение $z = (z_1, \ldots, z_S)$, результаты которого подвергаются последующей обработке для извлечения выходных данных модели $y := f(z)$.

Классические данные $x \in \square^{S_{\text{in}}}$ кодируются в гильбертово пространство *S*-кубита, используя отображение свойств U_x. Сначала к каждому кубиту применяются гейты Адамара. Затем нормированные значения отображенных данных кодируются с помощью R_Z-гейтов с углами поворота, равными значениям отображенных данных. Затем это сопровождается R_{ZZ} -гейтами, которые кодируют данные более высокого порядка, т.е. контролируемые значения поворота зависят от произведения значений признаков отображенных данных. Затем повторяются применения R_Z и R_{ZZ} -гейтов. Как только данные закодированы, модель оптимизирует вариационную схему G_θ, содержащую параметризованные логические R_Y – гейты с *CNOT* – слоями, формируя запутанные состояния между каждой парой кубитов, где $\theta \in \Theta$ обозначает обучаемые параметры.

На этапе пост-обработки измеряются все кубиты в базисе σ_z и классическим образом вычисляется четность выходных битовых строк. Для простоты рассмотрим бинарную классификацию, где вероятность наблюдения класса 0 соответствует вероятности наблюдения четной четности, и аналогично, для класса 1 с нечетной четностью. Выбор этой архитектуры модели обусловлен двумя причинами: отображение свойств служит полезной стратегией внедрения данных, которую, как считается, трудно смоделировать классическим способом по мере увеличения глубины и ширины массива, что значительно увеличивает мощность модели; и вариативная форма направлена на создание более выразительной схемы для квантовых алгоритмов.

1. Методы квантового глубокого обучения.

Рассмотрим некоторые распространенные методы квантового глубокого обучения [6-12].

А. *Метод опорных векторов*

Метод опорных векторов (*SVM - Support Vector Machine*) - алгоритм классификации с контролируемым машинным обучением, который создает разделяющую гиперплоскость для классификации. *SVM* обычно используется для бинарной классификации, и в этом разделе часто будем ссылаться на две бинарные метки в $\{0,1\}$. Цель применения *SVM*, состоит в том, чтобы максимально увеличить резерв. Сам резерв определяется как расстояние между разделяющей гиперплоскостью и обучающими выборками (как из 0, так и из 1 экземпляров), которые находятся ближе всего к этой гиперплоскости.

Ключевой идеей квантового *SVM* являются ядра, которые отличают ее от классических ядер *SVM* (таких как гауссовы или полиномиальные ядра). Функция квантового отображения признаков нелинейно преобразует классические точки данных, т. е. входные данные $\vec{x}$, в нелинейное квантовое состояние, т.е. реализует отображение:

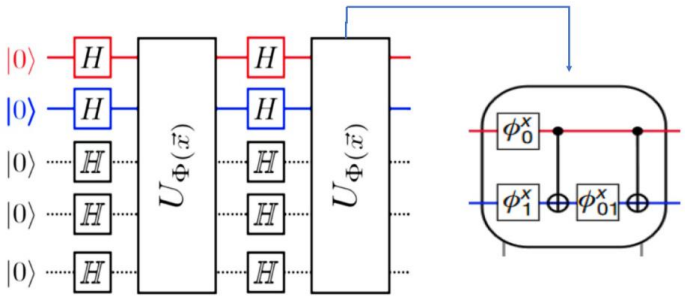
$$\phi : \vec{x} \mapsto \left|\phi(\vec{x})\right\rangle\left\langle\phi(\vec{x})\right|.$$

Далее создается отображение свойств объектов, математические свойства которых, с одной стороны, достаточно сложны, чтобы данные можно было легко разделить после их отображения, и в то же время достаточно просты, чтобы их можно было реализовать на реальном квантовом процессоре. Одно из возможных отображений свойств, которые используются в реализации, определяется с помощью унитарного:

$$U_{\Phi(x)} = \exp\left(i \sum_{S \subseteq [1,n]} \phi_S(\vec{x}) \prod_{i \in S} Z_i\right),$$

где $S \in \{0,1,\ldots,n-1,(0,1),(0,2),\ldots,(n-2,n-1)\}$, $\phi_i(\vec{x}) = x_i$, $\phi_{(i,j)}(\vec{x}) = (\pi - x_i) * (\pi - x_j)$.

Примечание. Отображение ϕ представляет собой отображение классического состояния в квантовое, тогда как каждое из отображений ϕ_i просто преобразует отдельные координаты; $\mathrm{U}_{\phi(\vec{x})}$ это функция отображения объектов, определенная на *n*-кубитах, размерность которых экспоненциально зависит от размера данных и номера кубита, генерируемая унитарным отображением: $\mathrm{U}_{\phi(\vec{x})} = U_{\phi(\vec{x})} H^{\otimes n} U_{\phi(\vec{x})} H^{\otimes n}$, которое преобразует классические входные данные $\vec{x}$ в квантовое состояние. Более конкретно, чтобы реализовать отображение ϕ, применяется отображение $\mathrm{U}_{\phi(\vec{x})}$ к фиксированному квантовому состоянию. Таким образом, точки данных встроены в единое унитарное отображение $\mathrm{U}_{\phi(\vec{x})}$, а не являются входным состоянием. Унитарную схему $\mathrm{U}_{\phi(\vec{x})}$ часто называют “отображением свойств объектов”. Унитарная схема $\mathrm{U}_{\phi(\vec{x})}$ строится путем повторения применения блока $U_{\phi(\vec{x})} H^{\otimes n}$, и этот блок повторяется d = 2 раза (d - это глубина блока $U_{\phi(\vec{x})} H^{\otimes n} U_{\phi(\vec{x})} H^{\otimes n}$ в схеме). Однако для создания более сложных и глубоких отображений признаков это количество может быть увеличено. Таким образом, параметры $\left(d, \phi(x)\right)$ можно настраивать для алгоритма классификации. Схема на рис. 7 для отображения объектов состоит из двух этапов, которые повторяются два раза (d = 2).



(а) *(б)*

Рис. 7. Схема отображения свойств

Сначала ко всем кубитам применяются гейты Адамара, затем диагональный элемент, который зависит от данных. Одна из идей, лежащих в основе этих отображений, заключается в необходимости кодировать данные таким образом, чтобы классические алгоритмы не могли имитировать их, что дает возможность получить преимущество перед классическими подходами. Отображение $U_{\phi(\vec{x})}$ для случая с двумя кубитами имеет простую форму (см. рис. 7(б)), где гейты - просто вращения, следующие этому правилу:

$$\phi_i(\vec{x}) = x_i, \quad \phi_{(i,j)}(\vec{x}) = (\pi - x_i) * (\pi - x_j).$$

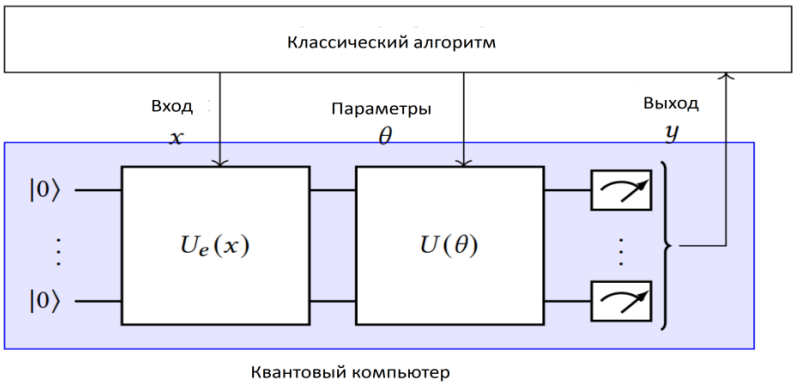
Наконец, классический ввод данных загружается путем применения унитарной функции к исходному квантовому состоянию $|0\rangle^{\otimes n}$: $\left|\phi(\vec{x})\right\rangle = U_{\phi(\vec{x})} |0\rangle^{\otimes n}$. В зависимости от графа связности квантового устройства квантовое отображение признаков имеет около $\frac{n(n-1)}{2}$ (n обозначает количество кубитов) дополнительных параметров, которые могут быть использованы для кодирования большего количества данных. После отображения объекта к его состоянию применяется квантовая схема малой глубины. Затем выходные данные обрабатываются с помощью нескольких уровней параметрических гейтов путем настройки параметров (весовых коэффициентов).

Б. *Вариационная схема для вариационных алгоритмов*

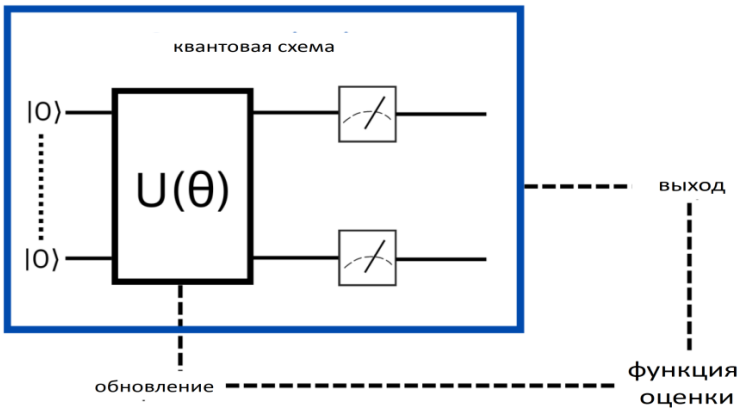
Общий квантовый классификатор будет состоять из двух взаимосвязанных схем. Первая - схема “отображения свойств объектов”, которую обсуждали выше. Вторая, в сочетании с измерением, реализует гиперплоскостную классификацию в пространствах свойств объектов. Параметрами этого

второго контура являются параметры обучения рассматриваемой модели. Этот второй контур является вариационным контуром, также называемым квантовым контуром малой глубины $W\left(\vec{\theta}\right)$. Вариационная схема - гибридная квантово-классическая схема, состоящая из параметризованных гейтов, зависящих от набора параметров $\vec{\theta}$, а также алгоритма обучения и целевой функции.

В рассматриваемом случае обучим схему правильно обозначать точки данных. Если использовать некоторые параметры схемы для ввода входных данных в схему и вычисления результатов (выходы), то можем видеть, что вариационная схема обладает такой же интуитивностью, как и другие модели с улучшенным обучением. Вариационные схемы используются для различных краткосрочных применений, таких как оптимизация и в контексте машинного обучения [31]. Вариационные квантовые схемы обладают универсальными свойствами и уже показали положительные результаты в реальных тестах, но их природа сильно отличается. В целом они основаны на подходе, показанном на рис. 8.



(а)



(б)

Рис. 8. (а) Гибридный алгоритм, использующий VQC; (б) Представление схемы оптимизации вариационной квантовой схемы

VQC всегда состоит из трех частей. Первая часть - уровень кодирования, который переводит классические данные в квантовое состояние, применяя унитарный оператор $U_e\left(x\right)$ в зависимости от данных. Вторая часть - вычислительный уровень, который преобразует квантовое состояние путем применения унитарного оператора $U\left(\theta\right)$ в зависимости от параметров θ. Заключительная часть - уровень измерений, который получает классические данные из квантового состояния. Классический алгоритм взаимодействует со схемой, передавая входные данные *x* в качестве параметров на уровень кодирования, передавая параметры θ на уровни расчета и получая выходные данные от уровня измерения (рис. 8 (а)).

Аналог представляет собой миниатюрную схему, состоящую из множества элементов, имеющих регулируемые характеристики, включая угол наклона элемента, который управляет поворотом. Затем измеряется результирующее квантовое состояние, которое должно дать правильные ответы на поставленную задачу (классификация, регрессия). Этот процесс повторяется до тех пор, пока схема не даст приемлемых результатов.

При выполнении *VQC* необходимо оценить градиенты такой функции затрат относительно каждого параметра. В обычных нейронных сетях это обычно достигается путем обратного распространения между аналитическими процедурами. При использовании *VQC* процессы становятся чрезмерно сложными, и не можем достичь промежуточных квантовых состояний, если сначала не измерим их. В настоящее время усовершенствованный подход называется правилом сдвига параметров и требует применения схемы дважды для каждого параметра и измерения ее результата дважды. В отличие от этого, традиционное глубокое обучение требует всего лишь одного прямого и обратного прохождения по сети, чтобы собрать все тысячи градиентов. Правило сдвига параметров может быть распараллелено на нескольких симуляторах или квантовых устройствах. Однако это может оказаться непрактичным при большом количестве параметров.

2. Квантовый ускоритель для машинного глубокого обучения

За последние десятилетия был достигнут значительный прогресс в исследованиях ускорения нейронных сетей на классических процессорах, например *CPU, GPU, ASIC, FPGA*. Но с увеличением масштаба приложения возникает узкое место, связанное с объемом памяти, известное как "стена памяти". Именно здесь в качестве решения могут быть использованы усовершенствованные квантовые вычисления. Разработка машинного обучения с использованием классического аппаратного ускорителя может осуществляться в два этапа:

- Разработка аппаратного обеспечения с учетом особенностей нейронной сети: ускорители глубоких нейронных сетей - *DNN* (*deep neural networks*) на базе ПЛИС, оптимальный по стоимости дизайн на основе ПЛИС для сверточных нейронных сетей - *CNN* (*convolution neural networks*) с ограниченными временными рамками;
- Совместное проектирование нейронной сети и аппаратного ускорителя: интегрированная структура, выполняющая поиск адаптированных архитектур нейронных сетей, разработанных именно для безопасного вывода и способная напрямую изучать архитектуры для крупномасштабных целевых задач и целевой аппаратной платформы.

Для полного использования потенциала квантового компьютера необходимо выполнить совместное проектирование нейронной сети и квантовых схем. Система полного ускорения (см., рис. 9) состоит из трех блоков:

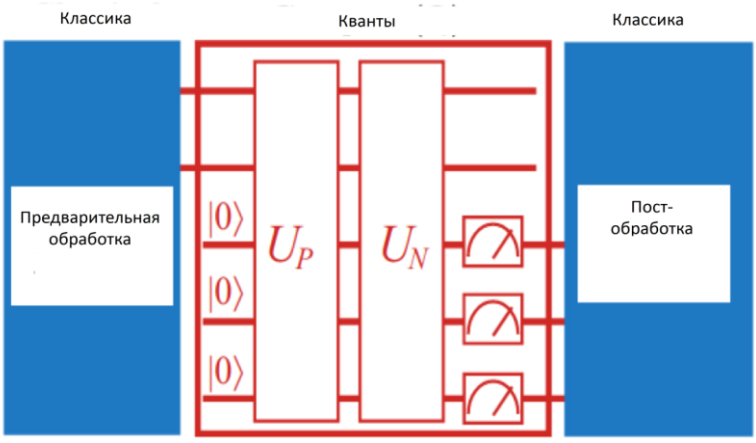


Рис. 9. Ускоритель квантовых вычислений

Предварительная и последующая обработка данных на классическом компьютере;

Ускоритель *NN* на квантовой схеме, включая подготовку квантового состояния (U_P);

Нейронные вычисления на основе *QC* (U_N).

Отметим, что название "вариационный" отражает вариационный подход к изменению параметров. Вариационные схемы имеют хорошее сходство с нейронными сетями, которые имеют веса, которые нам необходимо оптимизировать, и на заре квантовых вычислений их называли квантовыми нейронными сетями. Вариационный алгоритм состоит из двух частей: этапа обучения и этапа классификации. Для обучения дается набор помеченных точек для выполнения обучения (см. рис. 10).

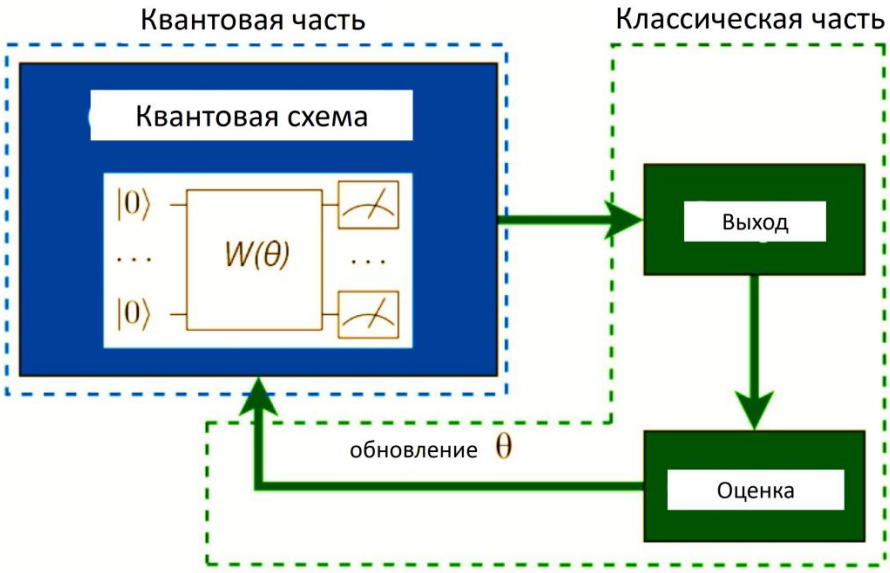


Рис. 10. Процесс создания вариационной схемы

Сначала мы обучаем классификатор, оптимизируя набор углов параметризации ($\vec{\theta}$).

Задача - найти оптимальную схему классификации $W\left(\vec{\theta}\right)$, которая разделяла бы наборы данных с разными метками. Для оптимизации нам необходимо определить функцию затрат, чтобы свести к минимуму вероятность присвоения решения неправильной метки. Ранее было объяснено, как работает квантовое отображение свойств признаков. Здесь рассмотрим схему отображения свойств признаков. Схема реализации показана на рис. 11.

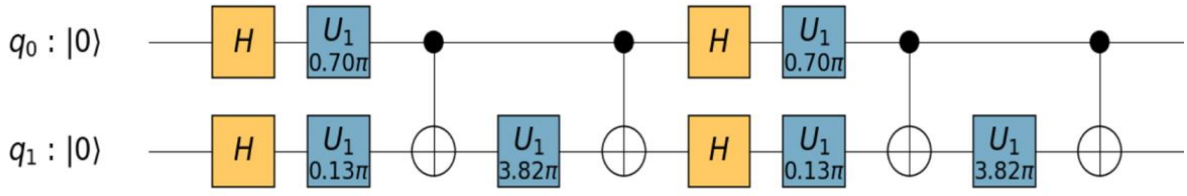


Рис. 11. Схема реализации отображения свойств признаков

Напомним, что в квантовом классификаторе значения входных векторов преобразуются в параметры в унитарном виде. В примере на Рис. 11 использованы точки данных $\begin{pmatrix} x_1 = 1.5 \\ x_2 = 0.3 \end{pmatrix}$, которые сгенерировали углы, как показано на рис. 11; H – гейт Адамара изначально применяется ко всем кубитам. Это позволяет подготовить равномерную суперпозицию всех битовых цепочек. За ним следует управляемый элемент управления U_1, который является управляемой версией однокубитного элемента управления U_1, и он полезен, поскольку позволяет применять квантовую

фазу и определять ее с помощью: $U_1(\phi)=\begin{pmatrix}1 & 0\\ 0 & e_{i\phi}\end{pmatrix}$. Кроме того, применяется многокубитный элемент управления, который не контролируется. Напомним, что элемент *controlled-NOT* переключает цель, когда кубит находится в состоянии $|1\rangle$. Элемент *Controlled-NOT* определяется как таковой:

$$C_X=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}.$$

Цель *ML*-модели состоит в том, чтобы изучить функцию *h*(*x*), которая аппроксимирует исходную *f*(*x*) с заданной точностью, при этом получая доступ к квантовому процессу E как можно реже (насколько это возможно) с небольшой средней ошибкой прогнозирования $|h(x)-f(x)|^2$, усредненной по некоторому заданному распределению входных данных $\mathrm{D}(x)$. Квантовая *ML*-модель обладает более широкими возможностями, чем классическая *ML*-модель.

На этапе прогнозирования классическая *ML*-модель ограничена обработкой классических данных, хотя и данных, полученных путем измерения квантовой системы на этапе обучения. В отличие от этого, квантовая *ML* может работать непосредственно в квантовой области. В некоторых задачах квантовая *ML*-модель обладает экспоненциальным преимуществом по сравнению с классической *ML*-моделью. Важный класс задач, связанных с квантовой механикой, заинтересован в прогнозировании функций вида $f(x)=Tr\left(O\mathrm{E}(|x\rangle\langle x|)\right)$, где *x* - классические входные данные, E произвольное (возможно, неизвестное) полностью положительное отображение с сохранением следа (*CPTP*), а *O* - известная наблюдаемая величина. Эта модель охватывает физический процесс, который использует классические входные данные и выдает действительное число в качестве выходных данных. Таким образом, эта общая настройка также включает в себя изучение классических функций, таких как функции, генерируемые нейронными сетями. Она также охватывает многие классические задачи *ML*, такие как классификация и регрессия.

Отображение E характеризует (потенциально очень сложную) квантовую эволюцию, происходящую в лаборатории. В зависимости от параметра *x* определяется квантовое состояние $\mathrm{E}(|x\rangle\langle x|)$. Наконец, в конце эксперимента экспериментатор измеряет определенное наблюдаемое значение *O*. Цель состоит в том, чтобы предсказать результаты измерений для новых физических экспериментов с другими значениями *x*, отличающимися от тех, которые были получены во время обучения.

Классические модели *ML* могут собирать классические данные измерений $\{(o_i,x_i)\}_{i=1}^{N_C}$, где o_i - результат, полученный при выполнении *POVM*-измерения состояния $\mathrm{E}(|x\rangle\langle x|)$. Через N_C обозначаем количество таких квантовых экспериментов, выполненных во время обучения в классической *ML*-среде. С другой стороны, мы рассматриваем квантовые *ML*-модели, в которых применение когерентно нескольких итераций *CPTP*-отображения E способно подготовить квантовое состояние, хранящегося в квантовой памяти, а предсказания производятся квантовым компьютером, имеющим доступ к этой квантовой памяти. Через N_Q обозначаем количество раз, которое использует E во время обучения в настройках процесса квантового *ML*.

Примечание. При сравнении не учитывается время работы классического или квантового компьютера, который генерирует прогнозы; интересует только то, сколько раз должен выполняться процесс E на этапе обучения в квантовых и классических условиях.

Пример. Рассмотрим задачу предсказания ожидаемых значений всех 4^n наблюдаемых величин Паули в неизвестном квантовом состоянии *n*-кубита ρ с небольшой ошибкой предсказания в худшем

случае. В этом случае используется функция $f(x) = \mathrm{Tr}\left(O\mathrm{E}_\rho\left(|x\rangle\langle x|\right)\right) = \mathrm{Tr}\left(P_x\rho\right)$, где $x \in \{I, X, Y, Z\}^n$, которая индексирует наблюдаемые значения по Паули и E_ρ подготавливает неизвестное состояние ρ, а затем P_x отображается в фиксированную наблюдаемую со значением O. Предсказание наблюдаемых величин Паули в среднем случае - гораздо более простая задача, поскольку большинство из 4^n ожидаемых значений экспоненциально малы по n.

Численный эксперимент на рис. 12 отображает результаты наиболее известных процедур применения *ML*.

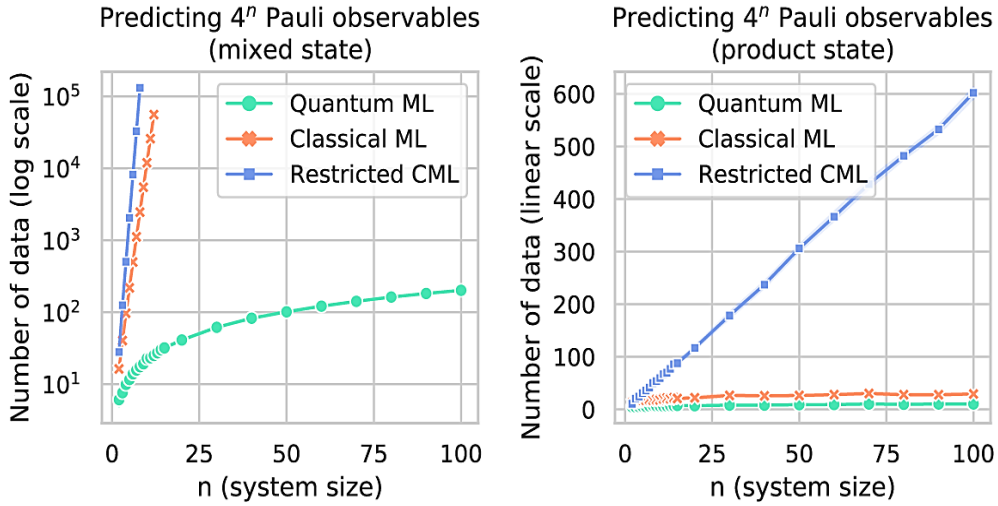


Рис. 12. Численные эксперименты – количество копий неизвестного состояния n-кубита, необходимое для прогнозирования ожидаемых значений всех 4^n наблюдаемых Паули с постоянной ошибкой прогнозирования в наихудшем случае

Смешанные состояния: квантовые состояния вида $(I + P) / 2^n$, где P - наблюдаемый n-кубит Паули. Состояния произведений: тензорные произведения состояний стабилизатора с одним кубитом.

Таким образом, существует экспоненциальное различие между количеством копий состояния ρ, необходимым для классического и квантового *ML* для предсказания ожидаемых значений, когда оно находится в классе смешанных состояний. Однако для класса состояний от произведения это различие гораздо менее выражено. Ограниченный классический *ML* может получить результаты $o_i \in \{\pm 1\}$ только с помощью $\mathrm{E}[o_i] = Tr\left(P_{x_i}\rho\right)$

Следовательно, каждая копия ρ предоставляет не более одного бита информации, и, следовательно, $\mathrm{O}(n)$ копий необходимы для прогнозирования ожидаемых значений всех 4^n, наблюдаемых по Паули. В отличие от этого, стандартный классический *ML* может выполнять произвольные измерения *POVM* для состояния ρ, поэтому каждая копия может предоставлять до n бит информации. Различие между классическим *ML* и квантовым *ML* является незначительным для состояний произведений.

Таким образом, точные результаты показывают, что классический *ML*, обученный с использованием данных квантовых измерений, может быть эффективным, укрепляя экспериментальные платформы, поддерживаемые классическим *ML*, и смогут плодотворно решать сложные квантовые задачи в физике, химии и материаловедении.

С другой стороны, строго установлен тот факт, что квантовый *ML* может иметь экспоненциальное преимущество перед классическим *ML* для определенных задач, где целью является достижение заданной ошибки прогнозирования в наихудшем случае. Важным направлением в будущем будет выявление дальнейших проблем в обучении, которые позволят получить

существенные квантовые преимущества, указывая на потенциальные практические применения квантовой технологии.

При разработке новых моделей *QNN* или тестировании производительности *QNN* на заданном наборе данных важным шагом является эффективное моделирование динамики обучения *QNN*. На данном этапе создан ряд платформ для квантового моделирования.

QNN - модели, которые используются далее, включают *QNN* на основе амплитудного кодирования и *QNN* на основе блочного кодирования (см. рис. 13).

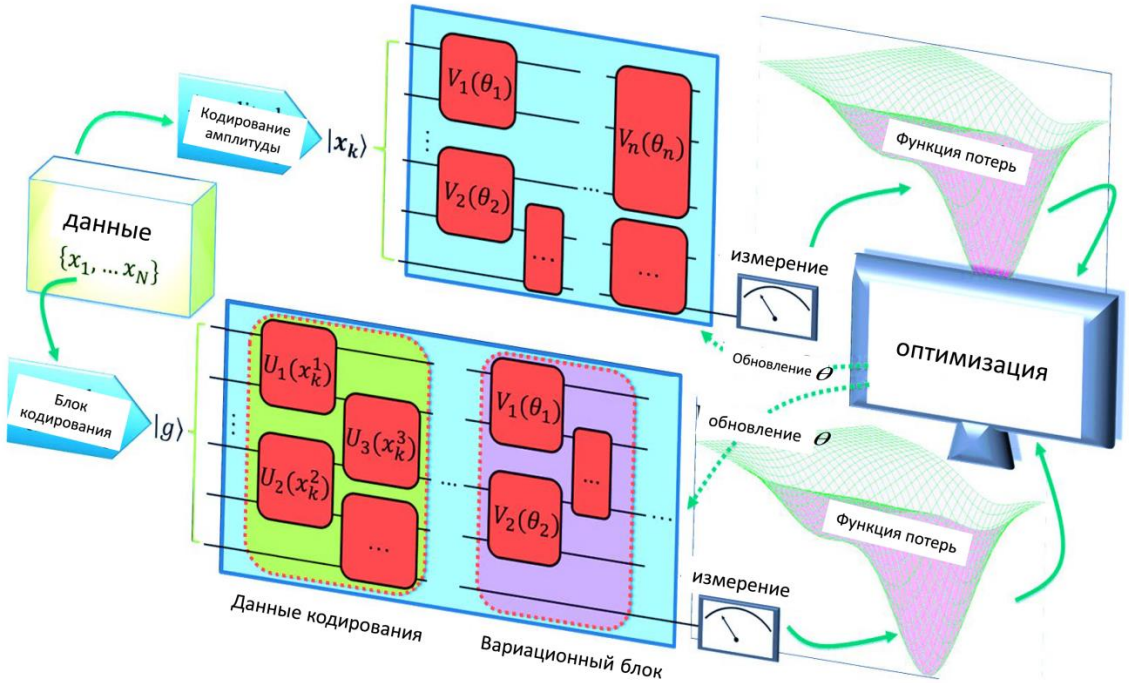


Рис. 13. Схематическая иллюстрация квантовых нейронных сетей (QNN) с двумя вариантами кодирования данных (двумя анзатцами): QNN на основе амплитудного кодирования и QNN на основе блочного кодирования [32]

Ожидаемые значения выходных состояний некоторых наблюдаемых величин часто используются в функции затрат для измерения расстояния между текущим и целевым прогнозами, в то время как классический оптимизатор используется для оптимизации параметров *QNN* для минимизации расстояния.

Более конкретно, *QNN*, основанные на амплитудном кодировании, обрабатывают данные, к которым есть прямой доступ. Например, если есть квантовая оперативная память для извлечения данных или данные поступают непосредственно из квантового процесса, можем предположить, что данные уже подготовлены в начале, и использовать вариационную схему для выполнения задачи обучения. В отличие от этого, *QNN*, основанные на блочном кодировании, обрабатывают данные, которые необходимо классически кодировать. Классическая стратегия кодирования может показаться неэффективной, но она более практична для экспериментальных демонстраций на устройствах *NISQ* по сравнению с *QNN*, основанными на амплитудном кодировании.

Благодаря успеху классического глубокого обучения, а также достижениям в области квантовых вычислений, квантовые нейронные сети (*QNN*), которые имеют сходство с классическими нейронными сетями и содержат вариационные параметры, привлекли большое внимание. Существует множество причин для разработки квантовой версии нейронных сетей.

Во-первых, квантовые компьютеры обладают потенциалом превосходить классические компьютеры по нескольким параметрам: некоторые алгоритмы, основанные на квантовом преобразовании Фурье, такие как алгоритм разложения на множители Шора, могут достигать экспоненциального ускорения по сравнению с наиболее известными классическими методами; Более того, доказано, что некоторые квантовые ресурсы, такие как квантовая нелокальность и

контекстуальность, обеспечивают безусловные квантовые преимущества в решение определенных вычислительных задач.

Эти захватывающие результаты стимулируют изучение потенциальных преимуществ моделей *QNN*, особенно в эпоху больших данных.

Во-вторых, когда пытаемся извлечь уроки из квантового набора данных, то есть набора данных, данные которого генерируются в результате квантового процесса, а не из классического набора данных, было бы более естественно использовать квантовую модель для решения задачи: извлечения достаточного количества информации из квантового состояния в классическое устройство это было бы очень сложно при масштабировании системы, в то время как модель *QNN*, которая может обрабатывать данные в экспоненциально большом гильбертовом пространстве, естественно, может обеспечить определенные преимущества. Этот момент был явно продемонстрирован, когда модель *QNN*, может распознавать квантовые состояния одномерных топологических фаз, защищенных симметрией, лучше, чем существующие подходы.

В-третьих, с точки зрения эффективной размерности, которая является свойством, связанным с эффективностью обобщения модели на новых данных, имеются предварительные данные, свидетельствующие о том, что квантовые нейронные сети могут быть способны достичь лучшей эффективной размерности и более быстрого обучения, чем сопоставимые сети прямого действия.

Пример: *Квантовая нейронная машина Борна* (*QNBM*). *QNBM* - квантовый аналог классической нейронной сети с прямой связью. Каждому нейрону в сети присваивается кубит, и он подключается к предыдущему слою нейронов с помощью подпрограммы квантовых нейронов, которая представляет собой схему повторения до достижения успеха (*RUS - Repeat-Until-Success*). Подпрограмма «*quantum neuron*» (квантовый нейрон) состоит из входного регистра $|x_{in}\rangle$, представляющего предыдущий уровень нейронов, выходного кубита, запущенного в состоянии $|\psi_{out}\rangle$, представляющем нейрон следующего уровня, а также вспомогательного кубита, первоначально находящегося в $|0_a\rangle$. Вспомогательный модуль используется для отображения функций активации от входного слоя нейронов $|x_{in}\rangle$ к каждому выходному нейрону $|\psi_{out}\rangle$ в следующем слое. Наглядное представление подпрограммы *RUS* для отдельного нейрона показано на рис. 14.

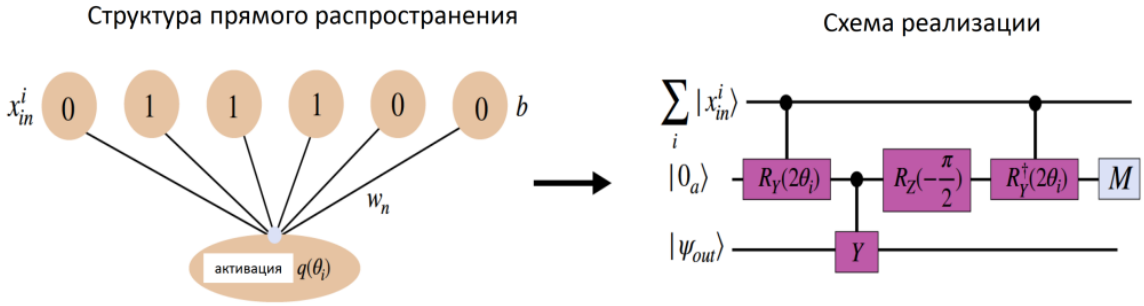


Рис. 14. Наглядная демонстрация отображения информации на один выходной нейрон в QNBM [33]

Слева показана структура обратной связи для активации нейрона, которая напоминает классическую сеть, содержащую обучаемые веса w_n и смещения b для отдельных битовых цепочек. Функция активации q вносит нелинейность в выходной нейрон на следующем уровне. Справа показана реализация квантовой схемы, которая создает это нелинейное отображение в виде квантовой схемы *RUS*. Схема передает информацию из суперпозиции битовых цепочек на входном уровне $\sum_i |x_{in}\rangle$ и выполняет нелинейную активацию на одном выходном нейроне $|\psi_{out}\rangle$. *QNBM* - просто многослойная сеть, состоящая из этих отдельных активаций квантовых нейронов. Основное различие между классической структурой (слева) и моделью квантовой схемы (справа) заключается в том, что квантовая сеть допускает суперпозицию входных данных на начальном слое.

Схема *RUS* выполняет нелинейную функцию активации для каждого выходного нейрона после суммирования весовых коэффициентов и смещений от нейронов предыдущего слоя. Конечное

состояние каждого выходного нейрона при успешной активации может быть описано следующим образом: $\sum_i F_i \left|x_{in}\right\rangle \otimes \left|0_a\right\rangle \otimes R_Y q\left(2\theta_i\right)\left|\psi_{out}\right\rangle$, где F_i - амплитудная деформация входного состояния во время отображения *RUS* и θ_i сумма весовых коэффициентов и смещений для каждой входной последовательности битов. Из-за вращения R_Y общая функция, выполняемая на выходном узле, равна $\sin^2\left(q\left(2\theta_i\right)\right)$. Ключевым отличием *QNBM* от классических нейронных сетей является ее способность выполнять функцию активации на основе суперпозиции дискретных битовых цепочек $\sum_i \left|x_i\right\rangle$.

Основным строительным блоком *QNBM* является квантовая нейронная схема (показана на рис. 15), которая ранее была представлена в качестве потенциального строительного блока для *QNN*.

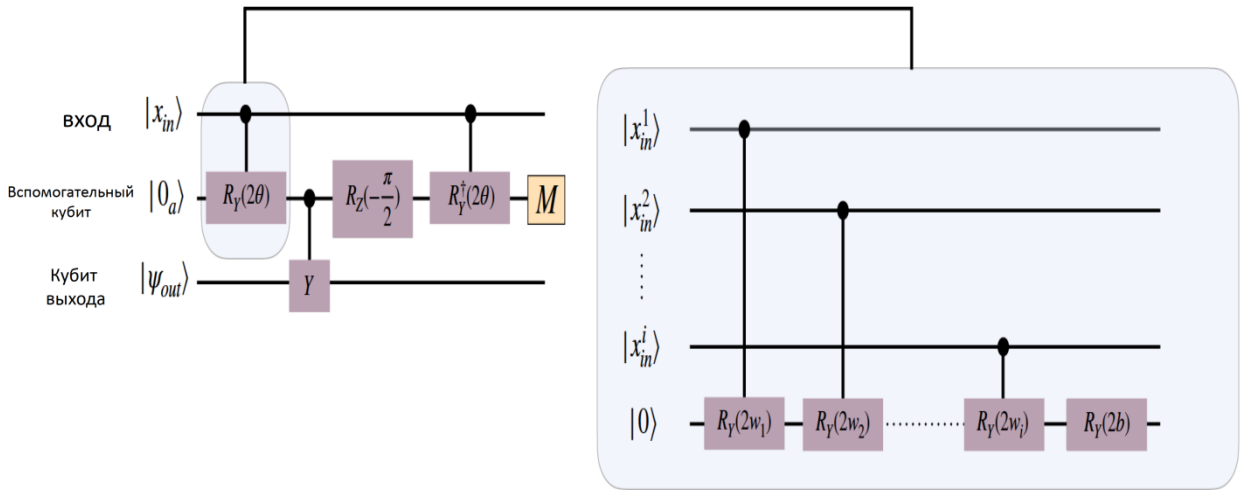


Рис. 15. Пример схемы квантового нейрона

Слева: Квантовая нейронная схема, соединяющая нейроны (кубиты) предыдущего слоя с одним нейроном следующего слоя. Если результат измерения вспомогательного элемента равен 0, схема $R_Y\left(2q\left(\theta\right)\right)$ успешно применена к выходному кубиту, а если результат измерения равен 1, необходимо применить $R_Y\left(-\pi/2\right)$ к выходному кубиту, элемент *NOT*, для вспомогательного элемента и снова применить схему. Следовательно, схема квантового нейрона - схема повторения до достижения успеха. Справа: реализация $R_Y\left(2\theta\right)$, где θ - функция весов, смещений и входных значений нейрона.

Каждому нейрону в сети присвоен кубит, поэтому схема *QN* имеет входной регистр $\left|x_{in}\right\rangle$, представляющий предыдущий слой, выходной кубит, инициализированный в состоянии $\left|\psi_{out}\right\rangle$, представляющем нейрон следующего слоя, а также вспомогательный кубит, первоначально включенный в $\left|0_a\right\rangle$. Предположим, что $\left|x_{in}\right\rangle$ изначально это одна строка битов, а не суперпозиция строк битов. Затем, непосредственно перед измерением (см. рис. 15 слева), объединенное квантовое состояние имеет вид:

$$\left|\psi\right\rangle = \left|x_{in}\right\rangle \otimes \left(\sqrt{p\left(\theta\right)}\left|0_a\right\rangle \otimes R_Y\left(2q\left(\theta\right)\right)\left|\psi_{out}\right\rangle + \sqrt{1-p\left(\theta\right)}\left|1_a\right\rangle \otimes R_Y\left(\pi/2\right)\left|\psi_{out}\right\rangle\right),$$

где θ - взвешенная сумма активаций нейронов на предыдущем слое: $\theta = \sum_{i=1}^{n} w_i x_i + b$, $w_i \in \left(-1;1\right)$ - веса, и $b \in (-1;1)$ - смещение. Аргумент поворота по оси y равен не 2θ, а $2q(\theta)$: это функция активации $q\left(\theta\right) = \arctan\left(\tan^2\left(\theta\right)\right)$. Функция активации имеет сигмовидную форму и определяется

последовательностью применяемых квантовых гейтов. Пока для упрощения используем только эту функцию активации.

В любом случае, просто измерив $|0_a\rangle$ с вероятностью $p(\theta) > \frac{1}{2}$ наличия вспомогательного кубита, получим правильную оценку состояния для следующего слоя:

$$|\psi\rangle = |x_{in}\rangle \otimes |0_a\rangle \otimes R_Y\left(2q(\theta)\right)|\psi_{out}\rangle .$$

Если вероятность $1 - p(\theta) \geq \frac{1}{2}$, то окажемся в состоянии: $|\psi\rangle = |x_{in}\rangle \otimes |1_a\rangle \otimes R_Y(\pi/2)|\psi_{out}\rangle$, которое может быть возвращено в исходное состояние с помощью элемента *NOT* на вспомогательном устройстве и $R_Y(-\pi/2)$ применено к выходному кубиту. Затем процесс будет повторяться до тех пор, пока вспомогательное измерение не даст результат $|0_a\rangle$, таким образом, схема квантового нейрона относится к классу схем повторения до успешного завершения (*RUS - Repeat Until Success*).

Важно отметить, что если мы позволим входному регистру находиться в общем состоянии - в виде суперпозиции битовых строк $\sum_i |x^i\rangle$, то схема успешно применит соответствующие преобразования к каждой из битовых строк в суперпозиции, что приведет к конечному состоянию: $\sum_i F_i |x_{in}^i\rangle \otimes |0_a\rangle \otimes R_Y q(2\theta^i)|\psi_{out}\rangle$. Здесь F_i относится к амплитудной деформации входного состояния во время отображения *RUS*. Хотя эти базовые блоки *QN* ранее использовались для проектирования маломасштабных квантовых сетей для задач обучения с контролем, здесь можем модифицировать схему *QN* для генеративного моделирования. В частности, добавляем параметризованные нелинейные преобразования в структуру «*Born Machine*».

3. От квантового нейрона (QN) к квантовой нейронной машине Больцмана (QNBM)

До сих пор только показывали, как один нейрон в слое может быть связан со всеми нейронами в предыдущем слое. Кроме того, подключение слоя k к слою $k-1$ в сети прямой связи является тривиальным расширением - можно повторить схему *QN* для каждого нейрона в слое k (как показано на рис. 16).

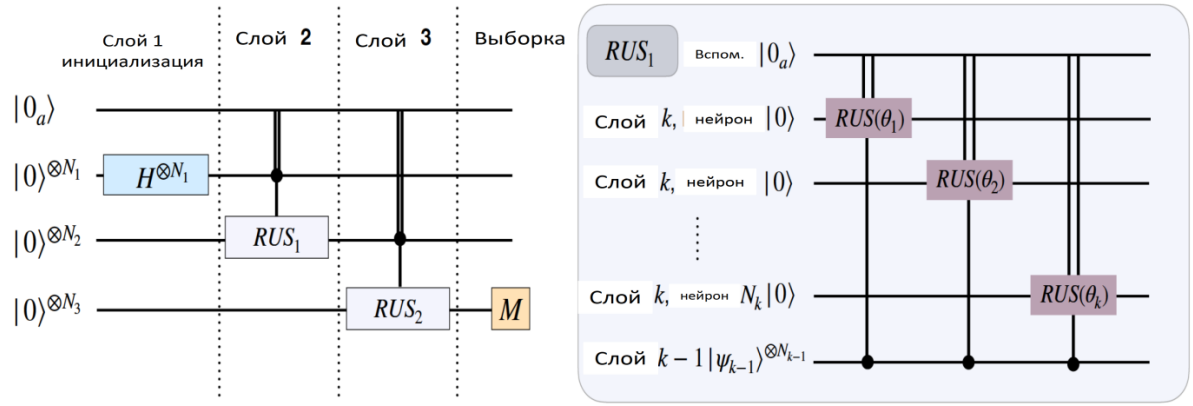


Рис. 16. Компонент квантовой схемы квантовой нейронной машины Борна слева: схема QNBM с 3 - мя слоями, где RUS_1 и RUS_2 представляют собой соединения между слоями

Входные нейроны инициализируются в однородной суперпозиции, а распределение вероятностей на выходе генерируется путем измерения выходного слоя. Справа: соединение двух слоев *QNBM*. Это равносильно запуску квантовых нейронных цепей *RUS* для каждого нейрона в слое k [34].

Ограничиваем эту работу значением $k = 3$, где количество нейронов во входном, скрытом и выходном слоях обозначается как (N_{in}, N_{hid}, N_{out}) соответственно. Также кубит «*ancilla*» $|0_a\rangle$ может быть повторно использован для каждой схемы *RUS*, поскольку после каждой схемы *ancilla* всегда включается и не коррелирует (отсоединяется) от всех нейронов. Таким образом, последующие измерения *ancilla* не влияют на уже подключенные нейроны. Аналогично, подключение всей сети сводится к повторению описанных выше действий для каждой пары соседних уровней.

Примечание. Предлагаемая здесь квантовая нейронная сеть не зависит от приложений и может, например, использоваться в классификации путем добавления встраивания данных и расчета затрат. Чтобы использовать ее в качестве генерирующей модели, вносим два изменения. Во-первых, квантовое состояние входного уровня инициализируем первый уровень в виде однородной суперпозиции битовых цепочек с помощью элементов Адамара, поскольку он способен вводить некоторую внутреннюю структуру в схему в качестве аналога классического предварительного распределения. Во-вторых, измеряем только кубиты, представляющие выходной слой, по аналогии с классическими нейронными сетями. Результирующее распределение вероятностей затем является результатом *QNBM* и оптимизируется точно так же, как и в *QCBM*, при этом веса и смещения выступают в качестве свободных параметров.

В квантовых вычислениях одной из наиболее сложных задач является моделирование гамильтониана. Моделирование гамильтониана - задача, для решения которой требуются алгоритмы, эффективно реализующие эволюцию квантового состояния. Ричард Фейнман предложил задачу моделирования гамильтониана в 1982 году, где он рассматривал квантовый компьютер в качестве возможного решения, поскольку моделирование общих гамильтонианов, по-видимому, растет экспоненциально относительно размера системы. В настоящее время эта проблема отражена практически во всех исследуемых системах, где хотим проанализировать физический дизайн и динамику в рамках вычислительной модели.

Задача гамильтонова моделирования определяется уравнением Шредингера, где оно дает эрмитову матрицу $H(2^n \times 2^n)$, которая воздействует на n кубит за время t, и максимальную ошибку моделирования ε, цель которой − найти алгоритм, который аппроксимирует оператор U таким образом, что $\|U - (e^{-iHt})\| \leq \varepsilon$, где e^{-iHt} - идеальная эволюция. Квантовое моделирование динамического поведения системы обычно выполняется за полиномиальное время *P, BQP* и т.д. Но его гамильтонова матрица растет экспоненциально $2^n \times 2^n$, что соответствует n кубитам исследуемых систем.

Таким образом, разрабатываются методы поиска наилучшего решения, допускающего ошибку. Существует две стратегии: «Юлий Цезарь» - («разделяй и властвуй» – «*divide and conquer*»); и алгоритм квантового блуждания. Вспомогательный элемент - локальный гамильтониан для некоторых конкретных гамильтонианов. *K*-локальный гамильтониан - эрмитова матрица, действующая на n кубитов, которая может быть представлена как сумма m членов гамильтониана, действующих не более чем на каждый из кубитов: $H = \sum_i H_i$. Это также приводит к d-разреженности. Говорят, что гамильтониан является d-разреженным (на фиксированной основе), если в любой строке или столбце содержится не более d ненулевых записей. Начиная с алгоритма "Разделяй и властвуй", первый шаг разбивает гамильтониан на сумму небольших и простых гамильтонианов, а второй шаг направлен на рекомбинацию сумм небольших и простых гамильтонианов.

Для этого есть три метода:

Первый метод − один из подходов, который представляет собой $e^{-i(A+B)t}$, где A и B - малые и простые гамильтонианы в приближении $\left(e^{-iAt/r}e^{-iBt/r}\right)^r$ для большого действительного значения r;

Второй метод сочетает в себе линейную комбинацию унитарных преобразований (*LCU - Linear Combination of Unitaries*) и усиление амплитуды без запоминания (*OAA - Oblivious Amplitude Amplification*).

Новейшей технологией является квантовая обработка сигналов (*QSP - Quantum Signal Processing*).

Квантовое машинное обучение (*QML*) исследует взаимодействие и использует преимущества идей и технологий квантовых вычислений и машинного обучения. Таким образом, *QML* - гибридная система, включающая как классическую, так и квантовую обработку, в которой квантовым устройствам предоставляются сложные с точки зрения вычислений подпрограммы. *QML* пытается использовать преимущества классического машинного обучения, которое работает лучше всего, и его стоимость, например, вычисление расстояния (внутреннее произведение), передавая его на квантовый компьютер, который может изначально вычислить его в гильбертовом векторном пространстве. В эпоху больших объемов классических данных и небольшого количества кубитов наиболее распространенным применением является разработка алгоритмов машинного обучения для классического анализа данных, выполняемых на квантовом компьютере, т.е. машинного обучения с квантовым расширением.

Обычные контролируемые процессы обучения в *QML* можно определить следующим образом:

• Отображение квантовых характеристик: это подготовка данных. В литературе этот этап называется подготовкой состояния;

• Квантовая модель: это создание модели. В литературе это называется унитарной эволюцией;

• Классическое вычисление ошибки: это этап вычисления ошибки, на котором модель наилучшим образом аппроксимирует входной набор; в машинном обучении этот этап известен как прогнозирование;

• Наблюдаемые: обычно этот этап включается в вычисление ошибки. Наблюдаемое проявляется в виде линейных операторов в гильбертовом пространстве, представляющем пространство состояний квантовых состояний. Собственные значения наблюдаемой - действительные числа, соответствующие возможным значениям; динамическая переменная, определяемая наблюдаемой, может быть измерена. В качестве наблюдаемой используем операторы Паули. Тем не менее, можем использовать некоторую линейную комбинацию этих операторов или некоторую аппроксимацию от них для выполнения конкретных измерений.

В типовом представлении квантовой модели (см. рис. 17) отображение характеристик объектов обычно рассматривается как фиксированный единственный блок в начале квантовой схемы и без повторений.

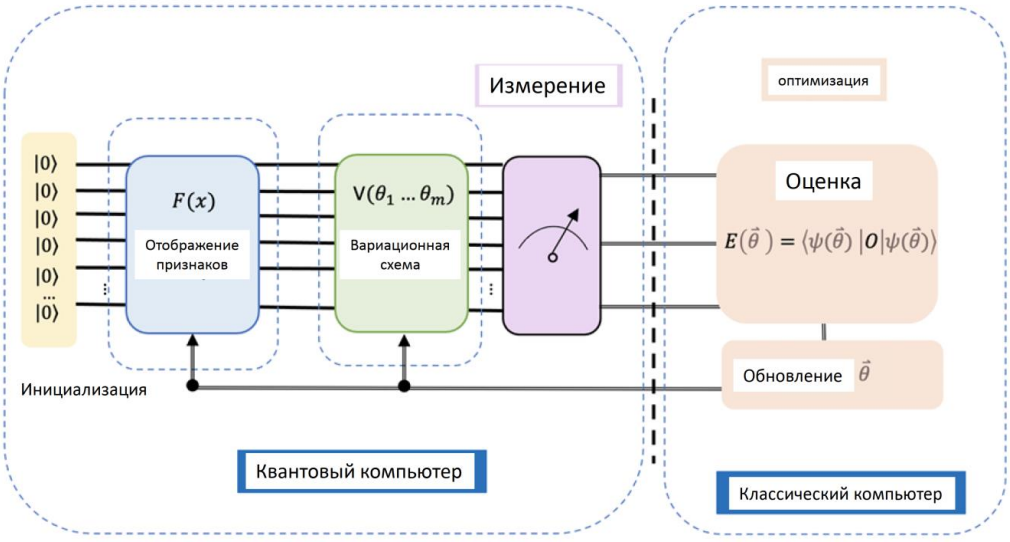


Рис. 17. Стандартная QML - модель только с одним встраиваемым блоком и анзатцем в виде параметризованной квантовой схемы с m параметрами

У этой модели есть ограничение: в худшем случае, когда входные данные не очень хорошо закодированы, для многих параметров, которые добавляем в параметрическую функцию/схему, не можем найти наилучшую модель, обобщающую входные данные.

Получаем классические данные, а квантовый компьютер обрабатывает их и возвращает результат после измерения. Существуют другие (три дополнительных) различных подхода и

комбинации для обработки классических или квантовых данных с помощью гибридных вычислений. В методах *QML* для обработки классических данных выполняются некоторые фундаментальные операции, называемые встраиванием. Процесс встраивания описывается как классическое кодирование входных данных в квантовые состояния. Называем отображение характеристик объектов контейнером, выполняющим указанную операцию отображения (встраивания). Кодирование встраивания может быть амплитудным, фазовым, базовым или гамильтоновым, что является ключевым процессом работы *QML* в вариационных квантовых схемах для глубокого обучения с подкреплением.

4. Квантовое машинное обучение (QML) с использованием вариационных квантовых схем (VQC)

Исследования в области *QML* существует множество подходов. Ожидаемое преимущество *QML* в значительной степени зависит от доступа к гильбертову пространству высокой размерности, предоставляемого квантовыми системами. Здесь кратко обсудим подходы квантовых алгоритмов обучения с подкреплением (*QRL*), основанных на вариационных квантовых схемах (*VQC*). *QML* часто имеет дело с математическими ожиданиями квантовых измерений. Математическое ожидание наблюдаемой величины *O* относительно квантового состояния $|\psi_i\rangle$ обозначается как $\langle O\rangle_\psi := \langle\psi|O|\psi\rangle$. В то время как *VQC* определяют новый класс *ML*-моделей, можно привести приблизительную аналогию с *NN*, где соотношение входных и выходных данных зависит от набора весовых коэффициентов.

Соответствующий гейт совершает поворот вокруг определенной оси на некоторый угол θ_0. Множество гейтов вращением образуют квантовую схему, где θ суммирует все свободные параметры. Изменение этих значений дает возможность определить эволюцию квантовой системы. Пусть U_θ обозначает соответствующий унитарный элемент. Схематический пример *VQC* показан на рис. 18.

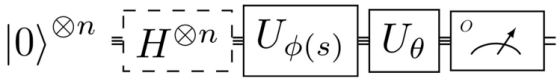


Рис. 18. Вариационная квантовая схема, состоящая из отображения характеристик объектов, вариационного слоя и измерения

Большинство задач *RL* используют концепцию состояний, на основе которых должно быть принято обоснованное решение. Эта информация о состоянии кодируется в квантовой системе с помощью соответствующего отображения характеристик. Как правило, входные данные *s* предварительно обрабатываются с помощью некоторой функции отображения Ф. Результаты Ф(*s*) могут быть аккуратно интегрированы в квантовую схему с помощью единого унитарного преобразования $U_{\Phi(s)}$. Для повышения выразительности *VQC* можно использовать более сложные процедуры кодирования данных, такие как повторная загрузка данных или инкрементная загрузка данных. В конечном счете, некоторые наблюдаемые величины должны быть измерены. Обычно используется вычислительная база с $O = Z^{\otimes n}$.

В целом, выходные данные *VQC*-модели можно описать следующим образом:

$$\langle O\rangle_{s,\theta} = \langle 0|\left(U_\theta U_{\Phi(s)}\right)^\dagger O U_\theta U_{\Phi(s)}|0\rangle := \langle 0|U_{s,\theta}^\dagger O U_{s,\theta}|0\rangle .$$

Для большинства задач это значение обрабатывается с помощью некоторой функции *f*. Сохраняя максимально общий вид, можно определить функцию потерь L для $f\left(\langle O\rangle_{s,\theta}\right)$ (на основе конкретной задачи). Обновление параметров может быть выполнено, например, с использованием методов, основанных на градиенте: $\theta \leftarrow \theta + \alpha \cdot \nabla_\theta \mathrm{L} f\left(\langle O\rangle_{s,\theta}\right)$. Требуемый градиент может быть получен с помощью правила сдвига параметров.

Общим является использование *VQC* в качестве аппроксиматора параметризованных функций. Типичный гибридный конвейер представлен на рис. 19.

Эта идея была впервые предложена для аппроксимации *Q*-функции и распространена на стратегию аппроксимации. *QPU* используется для аппроксимации соответствующей функции, в то время как предварительная и последующая обработка и оптимизация выполняются на классическом оборудовании. Взаимодействие с окружающей средой зависит от конкретного случая проблемы (например, классическая или квантовая среда).

Алгоритм следует понимать как гибридный, поскольку большая часть работы, особенно оптимизация, выполняется на классическом оборудовании. Агент наблюдает за текущим состоянием среды s_t, и применяет некоторую предварительную обработку ϕ. Результат кодируется с использованием отображения характеристик объектов $U_{\phi(s)}$. С текущими вариационными параметрами θ_t подготавливается квантовое состояние и измеряется наблюдаемая O_a (потенциально зависящий от действия). Ожидаемое значение $\langle O_a \rangle_{s,\theta}$ может быть подвергнуто последующей обработке для представления, например, функции значения состояния-действия $Q_\theta(s,a)$ или стратегии $\pi_\theta(a|s)$. В зависимости от экземпляра агент использует эту функцию для выборки действия и выполняет его в среде.

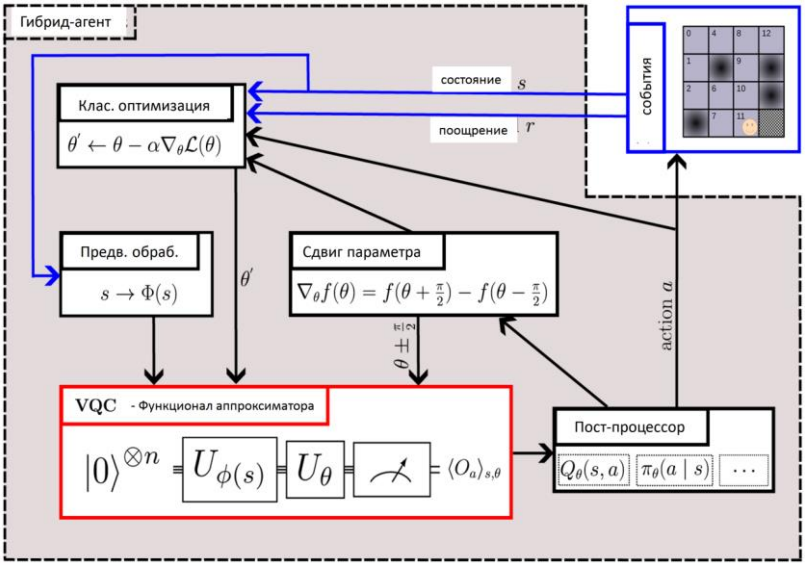


Рис. 19. Гибридный квантово-классический агент в типичном конвейере RL на основе VQC [35]

Вознаграждение r_t (и, возможно, также последовательное состояние s_{t+1}) отслеживается классическим оптимизатором. Чтобы включить обновление параметров на основе градиента, дополнительный гибридный модуль использует правило сдвига параметров для вычисления градиентов выходных данных *VQC* с учетом вариационных параметров θ_t. Классический оптимизатор определяет новый набор параметров θ_{t+1} и создает экземпляр *VQC* с этими обновленными параметрами. Эта общая итеративная процедура взаимодействия с окружающей средой, аппроксимации функций и обновления параметров повторяется в течение нескольких эпизодов точно так же, как, например, для *DRL*. Состояние *RL* интерпретируется как битовая строка, которая может быть закодирована с использованием идентичности $R_z(\pi)R_x(\pi)|0\rangle = |1\rangle$. Структура запутанности соединяет ближайших соседей с воротами *CZ*. Вариационные параметры учитываются при вращении отдельных кубитов вокруг осей *x*, *y* и *z*. Значение состояния-действия расшифровывается путем измерения наблюдаемых величин Паули-*Z* на нескольких кубитах, что соответствует количеству действий в окружающей среде. Полная схема *VQC* представлен на рис. 20.

Приближаясь к эпохе зашумленных квантовых вычислений промежуточного масштаба (*NISQ*), интересно исследовать, существует ли какой-либо квантовый алгоритм, который может не только решать фундаментальные проблемы обучения с обещанными квантовыми преимуществами, но и

может быть эффективно реализован на квантовых устройствах. Для достижения этой цели одним из наиболее вероятных решений является квантовая нейронная сеть (*QNN*), которая также называется вариационными квантовыми алгоритмами [36,37]. Конкретно, *QNN* состоит из вариационной квантовой схемы для подготовки квантовых состояний и классического контроллера для выполнения задач оптимизации [38].

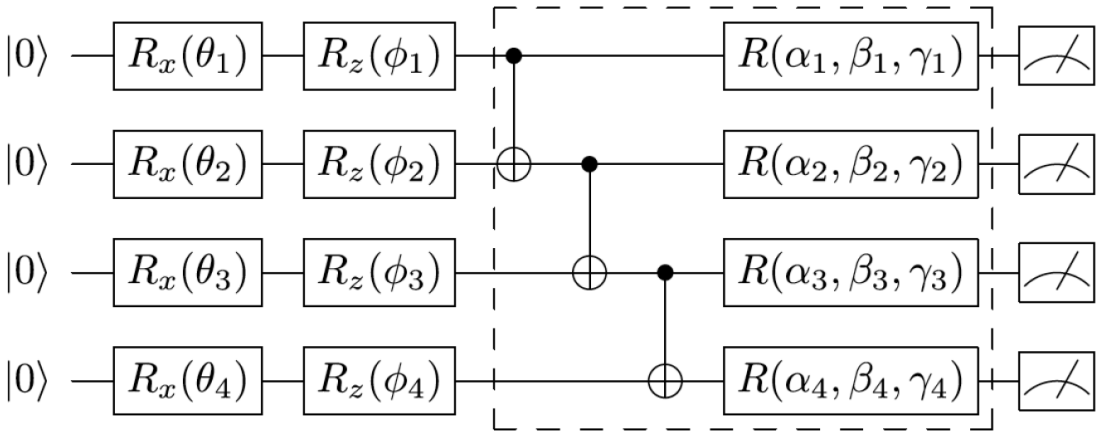


Рис. 20. Структура VQC

Элементы R_x и R_x используются для кодирования состояний. Несколько параметризованных уровней (пунктирная рамка) повторяются для формирования аппроксиматора *Q*-функции. Значения функции декодируются с использованием 1-кубитных наблюдаемых Паули-*Z*.

Частичным доказательством, подтверждающим это утверждение, является теоретический результат о том, что распределение вероятностей, генерируемое вариационной квантовой схемой, используемой в *QNN*, не может быть эффективно смоделировано классическими компьютерами. Учитывая высокую выразительность квантовых схем и схожую философию работы *QNN* и классической глубокой нейронной сети (*DNN*), вполне естественно использовать возможность реализации *QNN* на квантовых компьютерах в ближайшем будущем для выполнения определенных задач машинного обучения с более высокой производительностью по сравнению с классическими алгоритмами обучения.

5. Квантовое обучения квантовых нейронных сетей на основе поискового алгоритма Гровера

Конкретным примером достижения этой цели является квантовая нейронная сеть (*QNN*), которая была разработана для выполнения различных задач контролируемого обучения, таких как классификация и регрессия. Однако есть две основные проблемы, которые остаются неясными, когда *QNN* используется для выполнения задач классификации.

Во-первых, квантовый классификатор, который может эффективно сбалансировать вычислительные затраты, такие как количество измерений и эффективность обучения, остается неизученным. *Во-вторых*, неясно, могут ли квантовые классификаторы применяться для решения определенных задач, которые превосходят их классические аналоги. Здесь рассмотрим схему квантового обучения на основе поискового алгоритма Гровера (*GBLS - Grover-search based learning scheme*) для решения вышеуказанных проблем. Центральной концепцией *GBLS* является переформулировка задач классификации в задачу квантового поиска.

Хотя преимущество квантового алгоритма поиска по методу Гровера очевидно, способ преобразования задачи классификации в задачу поиска не является очевидным. Напомним, что в алгоритме поиска Гровера идентифицируется целевой элемент i^* в базе данных размера *K* путем

итеративного применения предопределенного оракула $U_f = I - 2\left|i^*\right\rangle\left\langle i^*\right|$ и оператора диффузии $U_{init} = 2\left|\varphi\right\rangle\left\langle\varphi\right| - I$ и $\left|\varphi\right\rangle = \frac{1}{\sqrt{K}}\sum_i \left|i\right\rangle$ к входному состоянию.

GBLS, как показано на рис. 21 а, использует заданную вариационную квантовую схему U_{L_1} и множество управляемых кубитов, расположенных вдоль оси *Z* (*MCZ*), для замены оракула U_f.

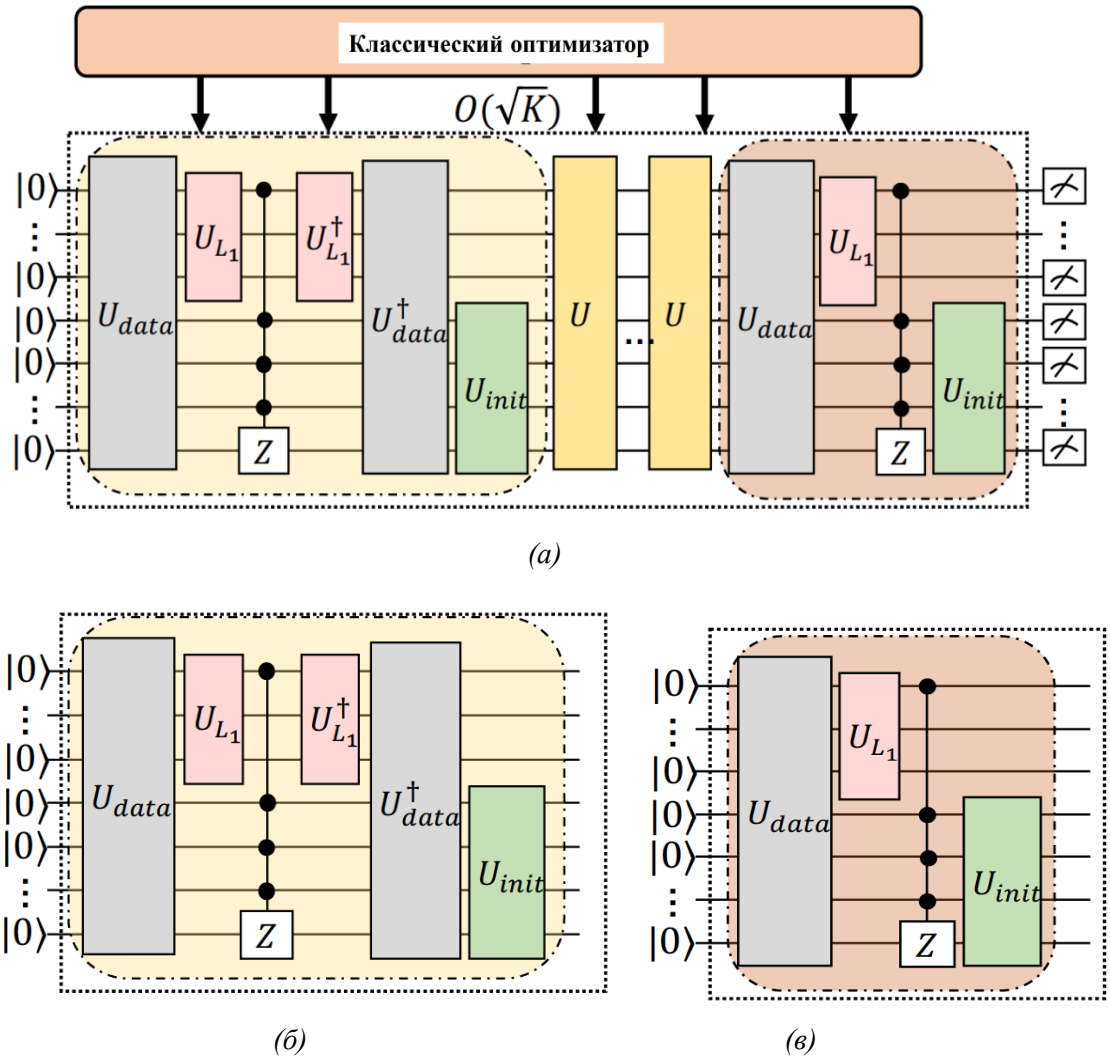


Рис. 21. Парадигма GBLS (а); Схемотехническая реализация оракула U (б); Схемотехническая реализация оракула UE (в)

В частности, вариационная квантовая схема условно переключает флаговый кубит (т.е. черную точку позади U_{L_1}, выделенную розовой областью) в зависимости от данных обучения. Обозначим все квантовые операции, относящиеся к одному циклу, как *U*, т.е.,

$$U := U_{init} \circ U_{data}^{\dagger}\left(U_{L_1} \otimes I\right)^{\dagger} \circ MCZ \circ \left(U_{L_1} \otimes I\right) \circ U_{data}.$$

Оператор *U*, определенный в уравнении состоит из унитарных операторов (т.е. U_{data}, U_{L_1}, *MCZ* и U_{init}), выделенных затененной желтой областью (рис. 21 (б)). В последнем цикле используется единая операция U_E, выделенная коричневой областью. Кубиты, взаимодействующие с U_{L_1} (или U_{init}), формируют регистр функций (или данных).

После переобозначения, определяем $U_{init} = I_F \otimes \left(2|\varphi\rangle\langle\varphi| - I_F\right)$ с помощью $|\varphi\rangle = \frac{1}{\sqrt{K}}\sum_i |i\rangle$. *GBLS* повторно применяет U к исходному состоянию $|0\rangle$, за исключением последнего цикла, в котором примененные унитарные операции заменяются на

$$U_E := U_{init} \circ MCZ \circ \left(U_{L_1} \otimes I\right) \circ U_{data},$$

как показано коричневой тенью на рис. 21 (в).

Следуя традиционному методу поиска по методу Гровера, *GBLS* запрашивает U и U_E в общей сложности $O\left(\sqrt{K}\right)$ раз, прежде чем выполнять квантовые измерения. На этом завершается квантовая часть *GBLS*.

Вариационные квантовые схемы и метод оптимизации рассмотрим на примере вариационных квантовые схем $U_{L_1}(\theta)$, используемые в *GBLS*. Метод оптимизации, т.е. правило сдвига параметров, которое используется для обучения U_{L_1} рассмотрим ниже.

Вариационные квантовые схемы, как отмечалось ранее, также называются параметризованными квантовыми схемами и состоят из обучаемых одиночных кубитных элементов и двух кубитных элементов (например, *CNOT* или *CZ*). В качестве схемы для устройств *NISQ* вариационные квантовые схемы были широко исследованы для решения задач генерации и распознавания образов с помощью вариационных гибридных квантово-классических алгоритмов. Одна из типичных вариационных квантовых схем называется многослойными параметризованными квантовыми схемами (*MPQC*), где расположение квантовых элементов в каждом слое идентично. Обозначим операцию, формируемую l-м слоем, как $U\left(\theta^l\right)$. Сгенерированное квантовое состояние из *MPQC* в виде

$$|\psi\rangle = \prod_{l=1}^{L} U\left(\theta^l\right)|0\rangle^{\otimes N_F},$$

где L - общее количество слоев. *GBLS* использует *MPQC* для построения U_{L_1}, т.е.,

$$U_{L_1}(\theta) = \prod_{l=1}^{L} U\left(\theta^l\right).$$

Схема расположения l-го слоя $U\left(\theta^l\right)$ показана на рис. 22.

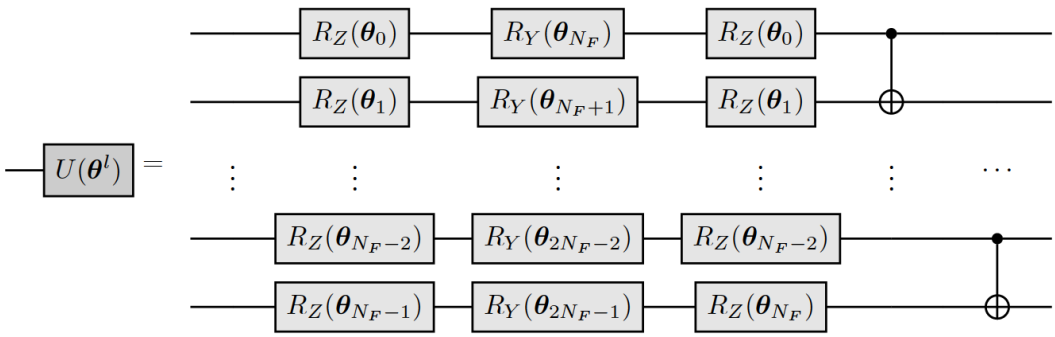


Рис. 22. Реализация l-го слоя $U\left(\theta^l\right)$

Предположим, что l-й слой $U\left(\theta^l\right)$ взаимодействует с N_F кубитами. Сначала к каждому кубиту применяются три обучаемых параметризованных элемента, R_Z, R_Y и R_Z, за которыми следуют N_F – 1 *CNOT*-гейт.

Правило обновления на *k*-й итерации выполняется следующим образом

$$\theta^{(k+1)} = \theta^{(k)} - \eta \frac{\partial \mathrm{L}\left(\theta^{(k)}, \mathrm{D}_k\right)}{\partial \theta},$$

где η - скорость обучения, а D_k - *k*-й пример обучения.

Расширим явную форму функции потерь $\mathrm{L}\left(\theta^{(k)}, \mathrm{D}_k\right)$ Основываясь на механизме алгоритма поиска по Гроверу, функция потерь в задаче классификации для *GBLS* может быть записана в следующем виде: $\min_{\theta} \mathrm{L}(\theta) := \mathrm{sign}\left(\frac{1}{2} - y_k\right) \mathrm{Tr}\left(\Pi \rho(\theta)\right)$, $\Pi = \left(|1\rangle\langle 1|\right) \otimes I \otimes \left(\left|i^*\right\rangle\left\langle i^*\right|\right)$ относится к оператору измерения, $\rho(\theta) = U_E U(\theta)^{O\left(\sqrt{K}\right)} |0\rangle\langle 0| \left(U_E U(\theta)^{O\left(\sqrt{K}\right)}\right)^{\dagger}$ – сгенерированное квантовое состояние, а $U(\theta)$ определена ранее (для ясности используем явную форму $U(\theta)$ вместо *U*).

Интуитивно понятно, что минимизированное значение $\mathrm{L}(\theta)$ соответствует тому факту, что в задаче классификации при $y_k = 1$ ($y_k = 0$) вероятность успешного выполнения выборки i^*, а также получения первого функционального кубита, равного "1" ("0"), максимизируется (минимизируется). Для оптимизации *GBLS* использует метод, основанный на градиенте, т.е. правило сдвига параметров. Градиенты $\mathrm{L}\left(\theta^{(k)}, \mathrm{D}_k\right)$ можно переписать в виде

$$\frac{\partial \mathrm{L}\left(\theta^{(k)}, \mathrm{D}_k\right)}{\partial \theta} = \mathrm{sign}\left(\frac{1}{2} - y_k\right) \frac{\partial \mathrm{Tr}\left(\Pi \rho\left(\theta^{(k)}\right)\right)}{\partial \theta},$$

где y_k относится к метке последней записи в D_k, *sign*( ) - обозначение знака функции, Π - оператор измерения и $\rho\left(\theta^{(k)}\right) = U_E U\left(\theta^{(k)}\right)^{O\left(\sqrt{K}\right)} |0\rangle\langle 0| \left(U_E U\left(\theta^{(k)}\right)^{O\left(\sqrt{K}\right)}\right)^{\dagger}$. В результате, *GBLS* использует правило сдвига параметров для достижения градиента $\frac{\partial \mathrm{Tr}\left(\Pi \rho\left(\theta^{(k)}\right)\right)}{\partial \theta}$.

Тогда правило обновления *GBLS* на *t*-й итерации для *j*-й записи таково

$$\theta_j^{(k+1)} = \theta_j^{(k)} - \eta \frac{\partial \mathrm{L}\left(\theta_j^{(k)}, \mathrm{D}_k\right)}{\partial \theta} = \theta_j^{(k)} - \eta \frac{\mathrm{Tr}\left(\Pi \rho\left(\theta_+^{(k)}\right)\right) - \mathrm{Tr}\left(\Pi \rho\left(\theta_-^{(k)}\right)\right)}{2} \mathrm{sign}\left(\frac{1}{2} - y_k\right).$$

Учитывая обучающий пример $x_i = \left(\omega_1^{(i)}, \omega_2^{(i)}\right) \in \square^2$ для всех $i \in [N-1]$, функция встраивания $g\left(\omega_1^{(i)}, \omega_2^{(i)}\right) : \square^2 \to \square^4$, которая используется для кодирования x_i в квантовые состояния, формулируется как $g\left(\omega_1^{(i)}, \omega_2^{(i)}\right) = \left(R_Y\left(g\left(\omega_1^{(i)}, \omega_2^{(i)}\right)\right) \otimes R_Y\left(\phi\left(\omega_1^{(i)}, \omega_2^{(i)}\right)\right)\right) |0\rangle^{\otimes 2}$, где $\phi\left(\omega_1^{(i)}, \omega_2^{(i)}\right)$ - указанная функция отображения.

Приведенная выше формулировка подразумевает, что $g(x_i)$ может быть преобразована в последовательность квантовых операций, где ее реализация проиллюстрирована на верхней левой панели Рис. 23. Чтобы одновременно закодировать несколько обучающих примеров в квантовые состояния, необходимо реализовать $g(x_i)$ в качестве контролируемой версии, где реализация показана на верхней правой панели рис. 23.

Верхняя левая панель иллюстрирует схемную реализацию кодирования унитарных данных U_{data}, соответствующая отображение характеристик объектов $g(x_i)$. Нижняя панель демонстрирует

реализацию *GBLS* с учетом входных данных $\mathrm{D}_k = \left(x_i, x_j, x_k, x_l\right)$, где реализация квантовой операции «*controlled - g*(x_i)» показана на верхней правой панели.

Синтетический набор данных и производительность различных квантовых классификаторов в идеальных условиях представлены на рис. 24. На левой панели показан синтетический набор данных, используемый в численном моделировании. Условные обозначения "положительный" (или "отрицательный") означают, что метка данных равна 1 (или 0). Правая панель демонстрирует точность обучения и тестирования различных квантовых классификаторов.

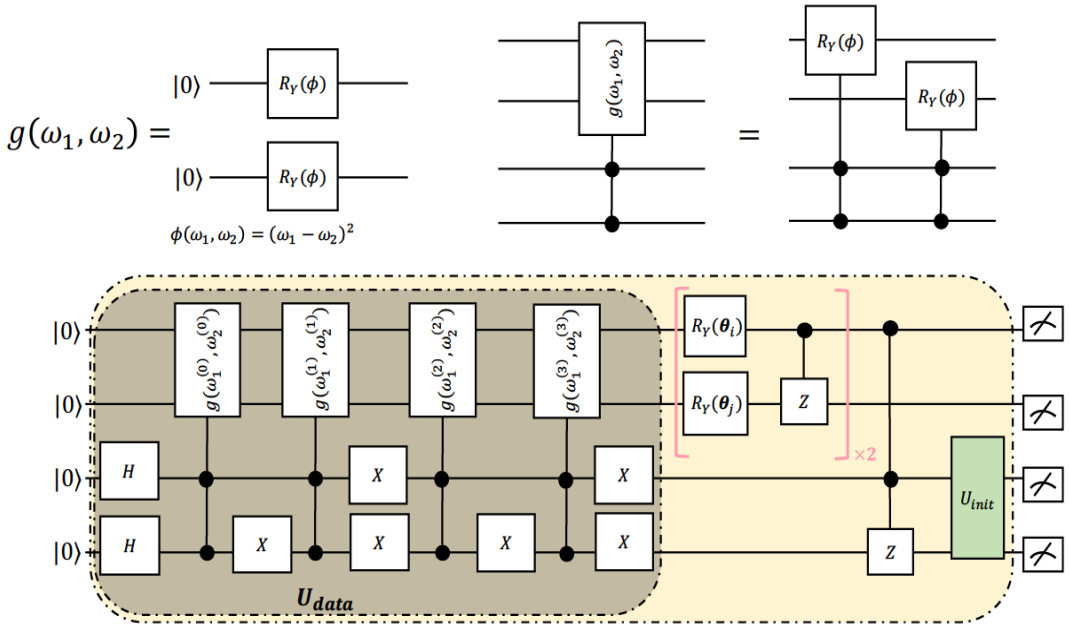


Рис. 23. Реализация GBLS, используемая при численном моделировании

Обозначения ‘*GBLS*’, "*BCE*", "*MSE*’ относятся к предлагаемому *GBLS*, классификатору квантового ядра с потерей двоичной перекрестной энтропии, и классификатору квантового ядра с потерей среднеквадратичной ошибки, соответственно. Вертикальные столбики отражают разницу в точности теста на каждой итерации.

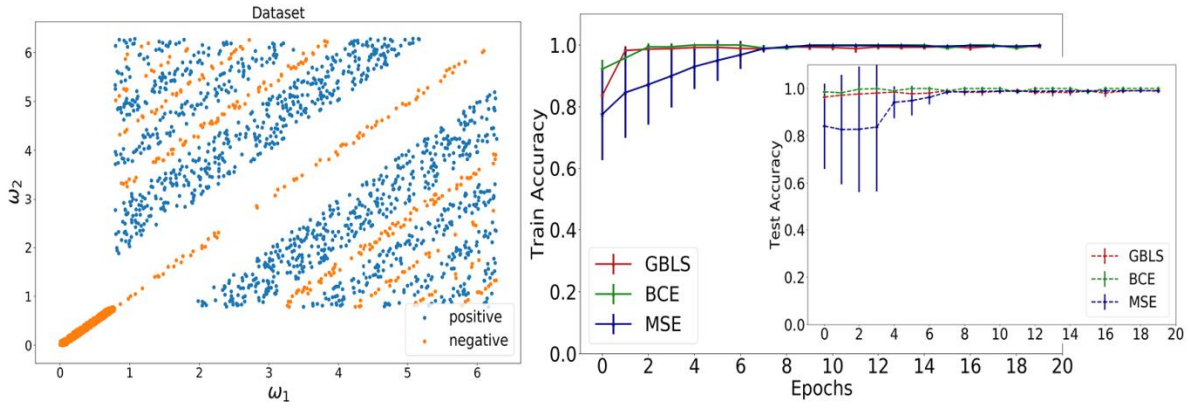


Рис. 24. Синтетический набор данных и производительность различных квантовых классификаторов в идеальных условиях

Выбор классификаторов квантового ядра в качестве эталонного основан на том факте, что этот метод обеспечивает современную производительность для классификации нелинейных данных.

Рассмотрим в качестве примера случай идеальной настройки. Оценим производительность различных квантовых классификаторов в идеальных условиях, когда квантовая система бесшумна, а

количество измерений бесконечно. Правая панель на рис. 25 иллюстрирует усредненную точность обучения и тестирования в зависимости от количества периодов.

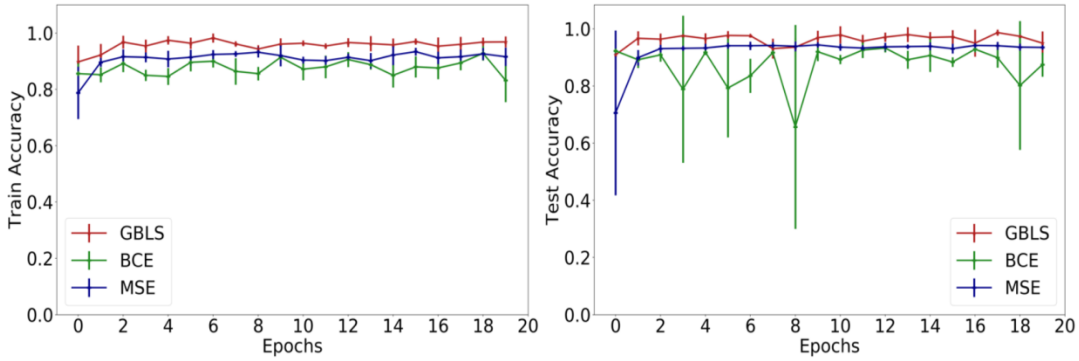


Рис. 25. Результаты моделирования различных квантовых классификаторов в режиме реалистичного шума

В частности, введенное предложение обеспечивает производительность, сравнимую с классификатором на квантовом ядре с потерей *BCE*, где точность обучения и тестирования достигает 99% в течение 2 эпох. Более того, эти два метода превосходят классификатор квантового ядра с потерей *MSE*, точность тестирования которого может достигать 95% только через 10 эпох. Разница в этих трех квантовых классификаторах после10 эпох становятся малой величиной, что означает, что все они сохраняют стабильную производительность при идеальных настройках. Между тем, шум измерения был применен ко всем квантовым классификаторам. Из-за относительно низкой производительности классификатора квантового ядра с потерей *MSE*, здесь сосредоточимся только на сравнении классификаторов *GBLS* и квантового ядра с потерей *BCE* и потерей *MSE*. Все настройки гипер-параметров идентичны тем, которые использовались в приведенных выше численных расчетах. Результаты моделирования представлены на рис. 25.

Обозначения "*GBLS*", "*BCE*" и "*MSE*" имеют то же значение, что и на рис. 24. Модель шума, которая извлекается из реального квантового оборудования, применяется к обучаемому унитарному $U_L(\theta)$ из этих трех классификаторов. В частности, три классификатора обеспечивают сопоставимую производительность. Такие результаты свидетельствуют об эффективности *GBLS*, поскольку требуемое количество измерений для *GBLS* сокращено в четыре раза по сравнению с двумя другими квантовыми классификаторами. Численные эксперименты показали, что *GBLS* может достичь производительности, сравнимой с другими передовыми квантовыми классификаторами, за счет использования меньшего количества измерений и может найти применение в квантовых устройствах.

Существует два основных направления применения *QNN*. Первый использует *QNN* для генерации квантовых состояний, которые минимизируют ожидаемое значение данного гамильтониана, например, в вариационных квантовых решателях собственных значений (*VQE*) для задач квантовой химии или в алгоритмах квантовой приближенной оптимизации (*QAOA*) для комбинаторных задач. задачи оптимизации. Второй путь использует *QNN* в качестве моделей машинного обучения, основанных на данных, для формирования дискриминативных и генеративных задачи, для решения которых *QNN* могли бы обладать большей выразительностью, чем их классические аналоги. Несмотря на то, что в исследования *QNN* вкладывается все больше усилий, есть свидетельства того, что их будет трудно обучать из-за плоских ландшафтов оптимизации, называемых бесплодными плато. Альтернативная схема обучения *QNN* с помощью максимально когерентного (т.е. квантового) протокола представлена на рис. 26.

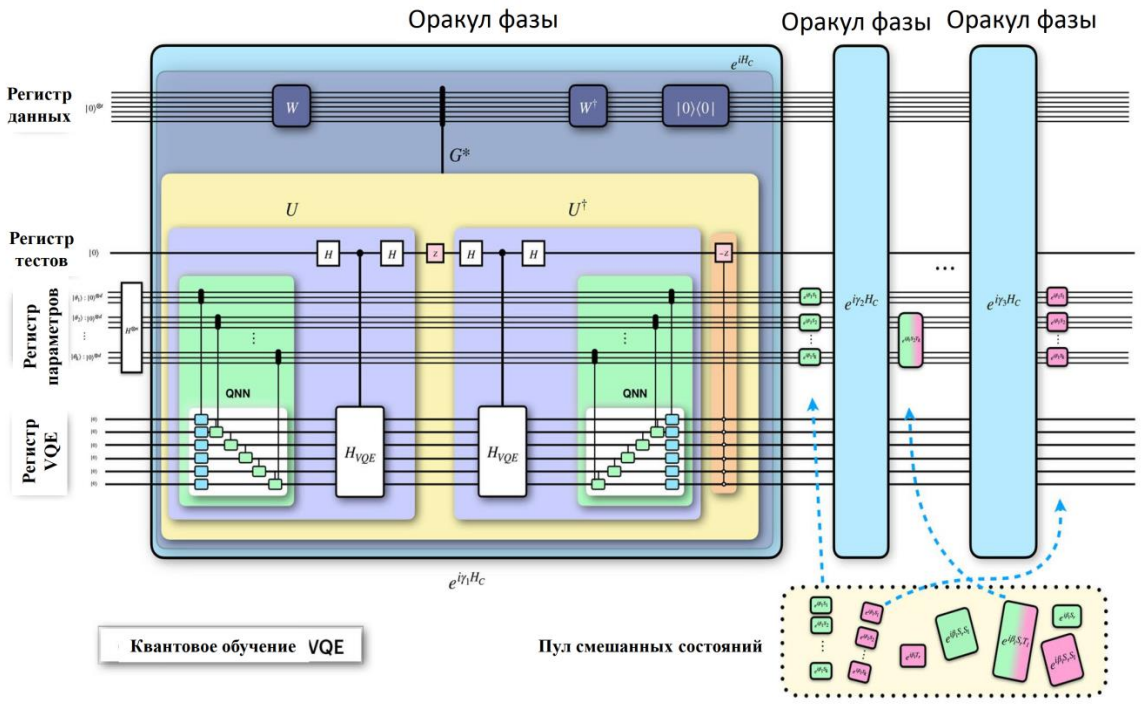


Рис. 26. Схема алгоритма квантового обучения для VQE

Здесь используется обучение *VQE* в качестве примера, чтобы представить принципиальную схему алгоритма квантового обучения для *QNN*.

Процесс квантового обучения может быть описан как эволюция состояния в совместном гильбертовом пространстве регистра параметров и регистра *QNN*. Их протокол квантового обучения состоит из двух чередующихся операций в режиме *QAOA* - первая операция воздействует как на регистр параметров, так и на регистр *QNN*, чтобы закодировать функцию стоимости *QNN* в относительную фазу состояния параметра. Вторая операция воздействует только на регистр параметров и представляет собой вариант оригинальных смесителей *QAOA*, адаптированный для случая, когда параметры в *QNN* являются непрерывными переменными. Эти две операции могут быть математически выражены как $e^{-i\gamma_i C(\theta)}$ и $e^{-i\beta_i H_M}$, где θ − параметры *QNN*, $C(\theta)$ - функция стоимости *QNN*, γ_i и β_i - настраиваемые гипер-параметры, H_M - гамильтониан смешанных состояний. Ожидается, что благодаря эвристической настройке гипер-параметров квантовое обучение приведет к достижению оптимальных параметров *QNN* после нескольких итераций чередующихся операций *QAOA*.

Данный процесс проиллюстрирован чередующиеся операции их квантового обучения на рис. 27. Протокол квантового обучения состоит из двух чередующихся операций в режиме оптимизации алгоритма *QAOA* — первая операция воздействует как на регистр параметров, так и на регистр *QNN*, чтобы закодировать функцию стоимости *QNN* в относительную фазу состояния параметра. Эта операция представлена синими блоками на рисунке. Вторая операция воздействует только на регистр параметров и представляет собой вариант оригинальных смесителей *QAOA*, адаптированный для случая, когда параметры в *QNN* являются непрерывными переменными.

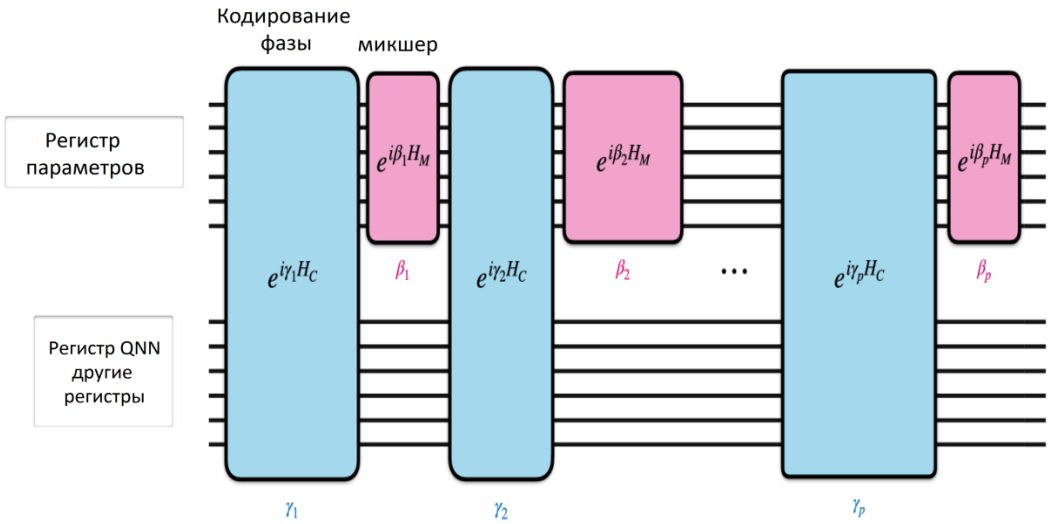


Рис. 27. Протокол обучения, аналогичный QAOA для QNN

Эта операция представлена розовыми блоками на рис. 27. Квантовое обучение может быть описано с помощью адаптивного алгоритма Гровера в качестве базовой линии перед внедрением системы квантового обучения с использованием *QAOA*.

Схема фреймворка для квантового обучения *QNN* представлена на рис. 28.

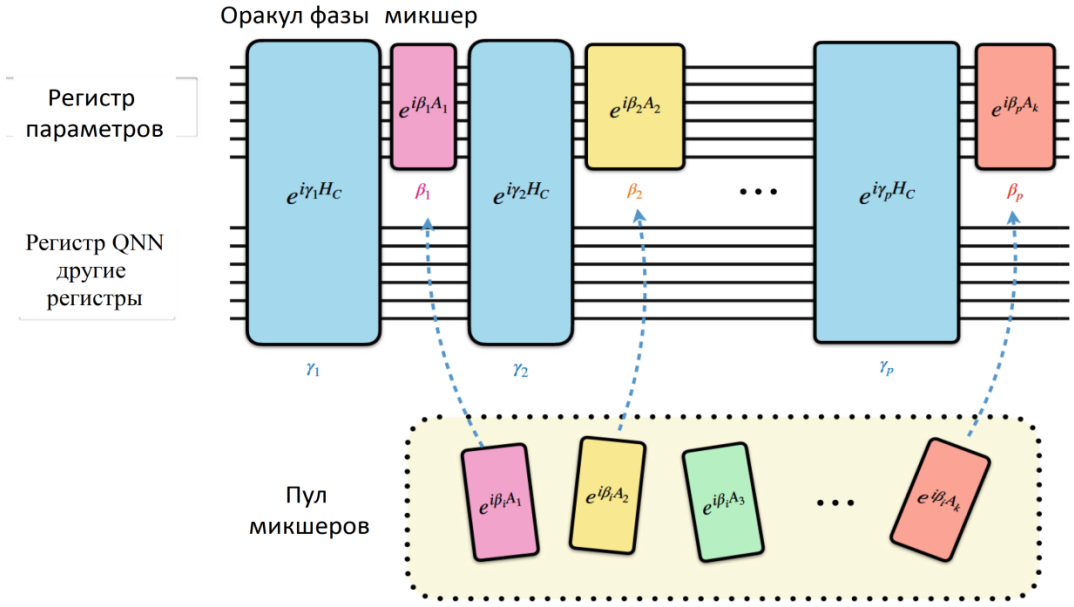


Рис. 28. Схема фреймворка для квантового обучения QNN

Примечание. Отметим некоторые фундаментальные различия адиабатического протокола и протокола *QAOA*. *QAOA* можно рассматривать как “упрощенную” версию адиабатической эволюции: гамильтонианы эволюции смешанных состояний являются начальным гамильтонианом в аналогичном адиабатическом алгоритме, а гамильтонианы затрат - конечным гамильтонианом. Однако *QAOA* с малой глубиной - на самом деле не оцифрованная версия адиабатической задачи, а скорее специальный отклик. *QAOA* способен детерминировано находить решение специально построенных задач оптимизации в случаях, когда квантовый отжиг не применим. В результате *QAOA* - алгоритм, основанный на учете помех, так что нецелевые состояния создают деструктивные помехи, в то время как целевые состояния создают конструктивные помехи.

На рис. 29 изображен этот процесс в виде интерференции *QAOA*. *QAOA* - алгоритм, основанный на учете помех, при котором нецелевые состояния создают деструктивные помехи, в то время как целевые состояния создают конструктивные помехи.

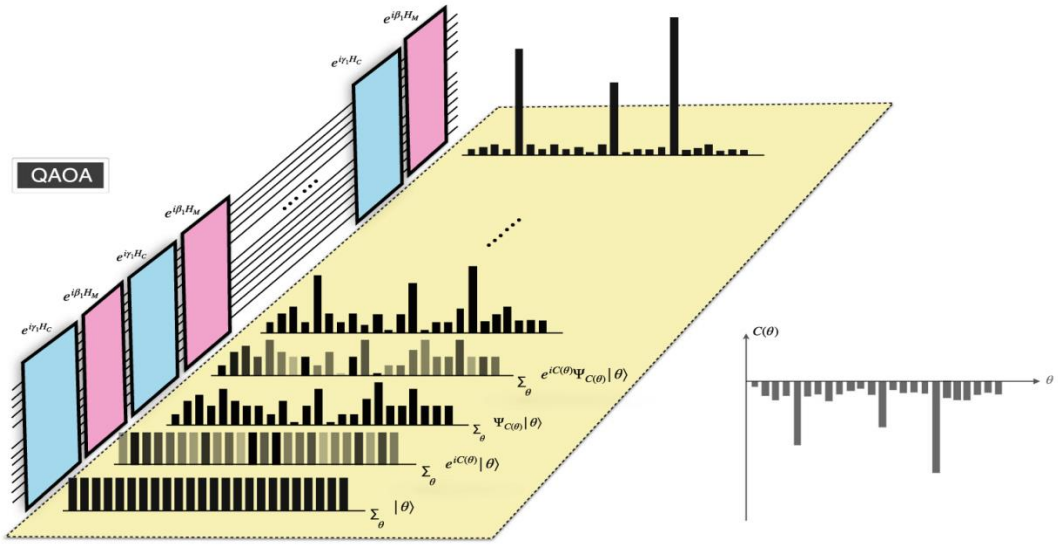


Рис. 29. Процесс интерференции QAOA

На рис. 29 иллюстрируется процесс взаимодействия, представляя эволюцию квантового состояния параметров (черные гистограммы на желтой плоскости) наряду с операциями *QAOA* (синие и розовые прямоугольники на линиях схемы, представляющие фазовое кодирование и смешанные состояния, соответственно). Начальное состояние $\sum_\theta |\theta\rangle$ (без учета коэффициента нормализации) является состоянием суперпозиции наложения всех возможных конфигураций параметров.

После первой операции кодирования фазы состояние становится $\sum_\theta e^{-iC(\theta)}|\theta\rangle$, для чего используется непрозрачность столбцов, чтобы указать значение фазы; значения амплитуд в состоянии остаются неизменными. После первого смешивания состояние становится $\sum_\theta \psi_{C(\theta)}|\theta\rangle$, в котором значения амплитуд в состоянии изменились.

Аналогичный процесс происходит со следующими операциями до тех пор, пока амплитуды оптимальных конфигураций параметров значительно не увеличатся (самая дальняя гистограмма). Серая гистограмма в правом углу - функция затрат, оптимизируемая *QAOA*.

Квантовая схема операций в исходном *QAOA* представлена на рис. 30.

Состояние инициализируется применением вентилей Адамара к каждому кубиту, представленных в виде $H^{\otimes n}$. Это приводит к состоянию равномерной суперпозиции всех возможных решений. *QAOA* состоит из чередующейся временной эволюции по двум гамильтонианам H_C и H_M для *p* раундов, где продолжительность в раунде *j* определяется параметрами γ_i и β_i соответственно. В исходном *QAOA* гамильтониан смешанных состояний H_M выбран равным $H_M = \sum_{j=1}^{n} X_j$.

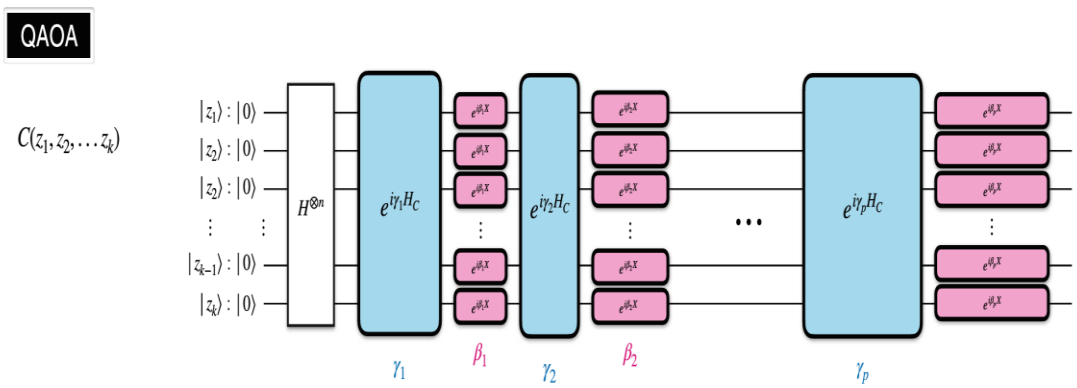


Рис. 30. Квантовая схема операций в QAOA

После всех *p* раундов состояние становится

$$|\beta,\gamma\rangle = e^{-i\beta_p H_M} e^{-i\gamma_p H_C} \ldots e^{-i\beta_2 H_M} e^{-i\gamma_2 H_C} e^{-i\beta_1 H_M} e^{-i\gamma_1 H_C} |s\rangle .$$

Алгоритм Гровера обычно используется в качестве метода поиска для нахождения набора желаемых решений из набора возможных решений. На практике применяют алгоритм Гровера для глобальной оптимизации, который называется адаптивный поиск Гровера - *Grover Adaptive Search* (*GAS*).

GAS был применен для обучения классических нейронных сетей и полиномиальной бинарной оптимизации. Отметим особенности *GAS*.

Рассмотрим функцию $f : X \to \square$, где для простоты представления предположим, что $X = \{0,1\}^n$. Нас интересует решение $\min_{x \in X} f(x)$. Основная идея *GAS* заключается в построении "адаптивного" оракула для заданного порогового значения *y* таким образом, чтобы он отмечал все состояния $x \in X$, удовлетворяющие $f(x) < y$, а именно оракул отмечает решение *x* тогда и только тогда, когда другая логическая функция g_y удовлетворяет $g_y(x) = 1$, где $g_y(x) = \begin{cases} 1 & \text{если } f(x) < y \\ 0 & \text{в другом случае} \end{cases}$. Затем оракул $\mathrm{O}_{\text{Гровер}}$ действует как $\mathrm{O}_{\text{Гровер}} |x\rangle = (-1)^{g_y(x)} |x\rangle$

Алгоритм Гровера выполняет поиск, чтобы найти решение $\tilde{x}$ со значением функции, лучшим, чем *y*. Затем устанавливает $y = f(\tilde{x})$ и повторяет до тех пор, пока не будут выполнены некоторые формальные критерии завершения - например, на основе количества итераций, времени или прогресса в *y*.

Обсудим использование адаптивного поиска *GAS* для выполнения глобальной оптимизации структуры *QNN*. Как представлено ранее, ядром адаптивного поиска *GAS* является адаптивный оракул, определенный как $\mathrm{O}_{\text{Гровер}}$. Рассмотрим кратко, как создать такой оракул для обучения *QNN*. Адаптивный $\mathrm{O}_{\text{Гровер}}$ оракул Гровера в контексте обучения *QNN* выступает в следующем виде: $\mathrm{O}_{\text{Гровер}} |\theta\rangle \otimes |0\rangle_{\text{QNN+ancillas}} = (-1)^{g(C(\theta)-C^*)} |\theta\rangle \otimes |0\rangle_{\text{QNN+ancillas}}$, в котором C^* - адаптивный порог для функции затрат, а функция *g* определяется как

$$g(x) = \begin{cases} 1 & x < 0 \\ 0 & \text{в других случаях} \end{cases} .$$

Когда $\mathrm{O}_{\text{Гровер}}$ воздействует на состояние суперпозиции параметров $\sum_\theta \omega_\theta |\theta\rangle$, имеем

$$\mathrm{O}_{\text{Гровер}} \sum_\theta \omega_\theta |\theta\rangle \otimes |0\rangle_{\text{QNN+ancillas}} = \sum_\theta (-1)^{g(C(\theta)-C^*)} \omega_\theta |\theta\rangle \otimes |0\rangle_{\text{QNN+ancillas}} .$$

QNN с оракулом $\mathrm{O}_{\text{Гровер}}$ может быть создана с помощью следующих шагов.

Амплитудное кодирование. Первым шагом является кодирование функции затрат *QNN* в амплитуду. В зависимости от формы функции затрат *QNN* амплитудное кодирование может быть достигнуто с помощью теста *swap* или теста Адамара. Рассмотрим амплитудное кодирование с помощью *swap* – теста. Для задачи обучения чистое состояние $|\psi\rangle = T|0\rangle$ (*T* – заданный унитарный оператор) функция стоимости представляет собой точность между сгенерированным состоянием из *QNN* и состоянием $|\psi\rangle = T|0\rangle$. В этом случае амплитудное кодирование может быть достигнуто с помощью теста, как показано в схеме на рис. 31.

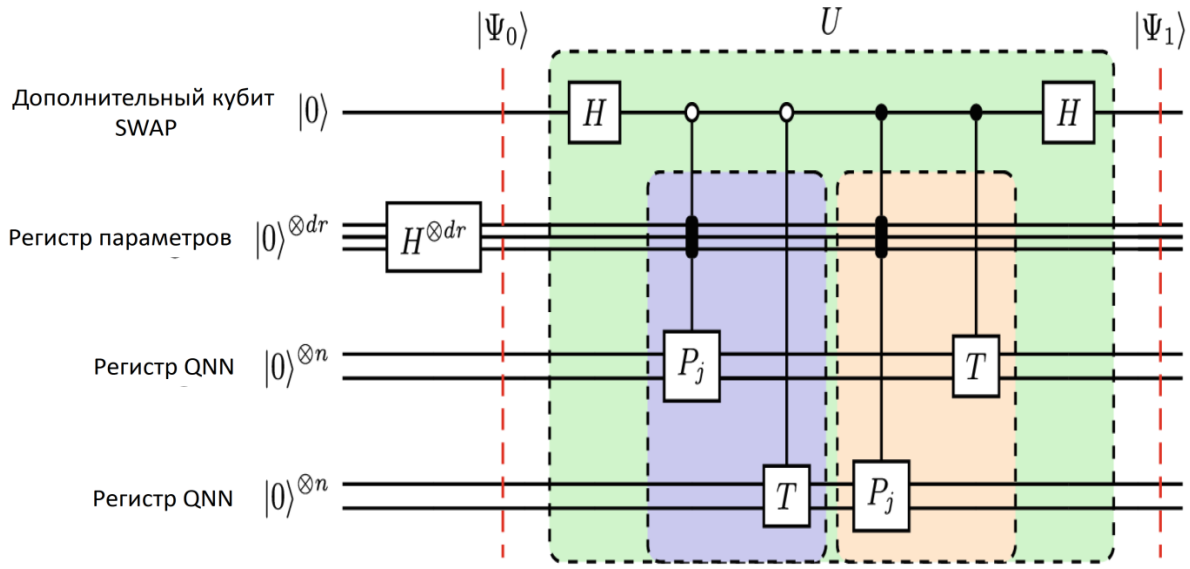


Рис. 31. Амплитудное кодирование с помощью swap – теста

Обозначим унитарный оператор для схемы *swap* – теста (выделенной зеленым прямоугольником) как *U*, а входное и выходное состояния *U* как $|\psi_0\rangle$ и $|\psi_1\rangle$ соответственно. Входные данные для *U*, $|\psi_0\rangle$, могут быть описаны в виде:

$$|\psi_0\rangle = |0\rangle \otimes \left(\sum_j |j\rangle\right) \otimes |0\rangle^n_{QNN1} |0\rangle^n_{QNN2}.$$

Тогда оператор *U* может быть записан явно в виде

$$U := [H \otimes I \otimes I \otimes I] \cdot \left[|0\rangle\langle 0| \otimes \left(\sum_j |j\rangle\langle j| \otimes P_j \otimes T \right) + |1\rangle\langle 1| \otimes \left(\sum_j |j\rangle\langle j| \otimes P_j \otimes T \right) \right] \cdot [H \otimes I \otimes I \otimes I]$$

Таким образом, *Pj* представляет *QNN* со специфической конфигурацией параметра, определяемой его управляющей битовой строкой *j*. Это представление применимо не только к одному гейту вращения, но и к случаю, когда имеется несколько параметризованных гейтов вращения. Можно доказать, что *U* можно переписать как $U = \sum_j |j\rangle\langle j| \otimes U_j$, где U_j - единичное значение индивидуального *swap* – теста для унитарного P_j и целевого унитарного *T*, определяемое как $U_j := [H \otimes I \otimes I] \cdot [|0\rangle\langle 0| \otimes P_j \otimes T + |1\rangle\langle 1| \otimes T \otimes P_j] \cdot [H \otimes I \otimes I]$. Результирующее состояние U_j, действующего на $|\psi_0\rangle$, равно $|\phi_j\rangle = U_j |0\rangle \otimes |0\rangle^n_{QNN1} |0\rangle^n_{QNN2}$

и имеет следующий вид: $|\phi_j\rangle = \sin\theta_j |u_j\rangle|0\rangle + \cos\theta_j |\upsilon_j\rangle|1\rangle$. Следовательно, конечное выходное состояние из *U*, $|\psi_1\rangle = U|\psi_0\rangle$, равно

$$|\psi_1\rangle = \sum_j |j\rangle \underbrace{\left(\sin\theta_j |u_j\rangle|0\rangle + \cos\theta_j |\upsilon_j\rangle|1\rangle\right)}_{|\phi_j\rangle} = \sum_j |j\rangle|\phi_j\rangle$$

.

Таким образом, функция стоимости (точность $|\langle p_j | t\rangle|^2$) для различных параметров была закодирована в амплитуды состояния $|\psi_1\rangle$.Поскольку анализ для случая теста Адамара очень похож на анализ для теста *swap*, опускаем здесь подробности.

2. *Оценка амплитуды*. Вторым шагом, следующим за амплитудным кодированием, является использование оценки амплитуды для извлечения и сохранения функции стоимости в дополнительном регистре, который называется “регистром амплитуды”. Далее подробно рассмотрим оценку амплитуды. После амплитудного кодирования с помощью теста *swap* вводим дополнительный регистр $|0\rangle^{t}_{\text{амплитуда}}$, и выходное состояние $|\psi_1\rangle$ (используя ту же запись) становится $|\psi_1\rangle = \sum_j |j\rangle|\phi_j\rangle|0\rangle^{t}_{\text{амплитуда}}$, где $|\phi_j\rangle$ может быть разложено в виде $|\phi_j\rangle = \frac{-i}{\sqrt{2}}\left(e^{i\theta_j}|\omega_+\rangle_j - e^{i(-\theta_j)}|\omega_-\rangle_j\right)$. Следовательно, имеем

$$|\psi_1\rangle = \sum_j \frac{-i}{\sqrt{2}}\left(e^{i\theta_j}|\omega_+\rangle_j - e^{i(-\theta_j)}|\omega_-\rangle_j\right)|0\rangle^{t}_{\text{амплитуда}}.$$

Общий Гровер - оператор G определяется как $G := UC_2U^{-1}C_1$, где C_1 - вентиль Z на вспомогательном кубите *swap*, а C_2 - унитарное «инверсное нулевое состояние». Можно показать, что оператор G может быть выражено как $G = \sum_j |j\rangle\langle j| \otimes G_j$, где G_j - индивидуальный Гровер - оператор, как определено в: $G_j = U_j C U_j^{\dagger}\left(Z \otimes I^{\otimes 2n}\right)$, где $Z = |0\rangle\langle 0| - |1\rangle\langle 1|$ - оператор Паули-*Z*; $C = I^{\otimes(2n+1)} - 2|0\rangle^{\otimes(2n+1)}\langle 0|^{\otimes(2n+1)}$.

Описанную процедуру и далее алгебраические процедуры вычисления $|\psi_2\rangle, |\psi_3\rangle, |\psi_4\rangle$ можно проиллюстрировать схемой [35-38], показанной на рис. 32.

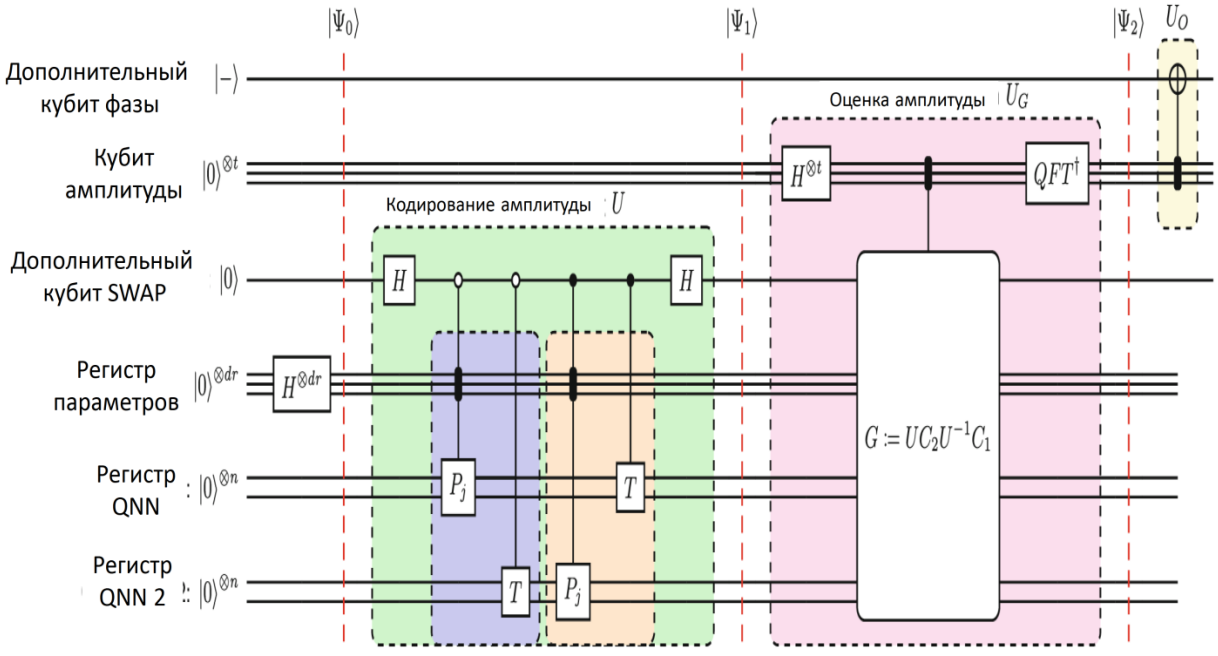


Рис. 32. Основные этапы создания оракула Гровера

Шаг 0: инициализируем систему, применяя гейты Адамара к регистру параметров, что приводит к состоянию

$$|\psi_0\rangle = |0\rangle_s \otimes \left(\sum_j |j\rangle\right) \otimes |0\rangle^{n}_{\text{QNN1}}|0\rangle^{n}_{\text{QNN2}}.$$

Шаг 1 (зеленый прямоугольник с пунктиром): амплитудное кодирование функции затрат, как показано на рис. 32; в результате чего получается состояние

$$|\psi_1\rangle = \sum_j |j\rangle\left(\sin\theta_j |u_j\rangle|0\rangle + \cos\theta_j |\upsilon_j\rangle|1\rangle\right),$$

в котором θ_i содержит функцию стоимости;

Шаг 2 (розовая рамка с пунктиром): оценка амплитуды для извлечения и сохранения функции стоимости в дополнительном регистре, который называется “регистром амплитуды”, в результате чего получаем состояние

$$|\psi_2\rangle = \sum_j \frac{-i}{2}\left(e^{i(\theta_j)}|j\rangle|\omega_+\rangle_j|2\theta_j\rangle - e^{i(-\theta_j)}|j\rangle|\omega_-\rangle_j|-2\theta_j\rangle\right);$$

Шаг 3 (желтая рамка с пунктиром): пороговый оракул кодирует функцию затрат в относительную фазу с помощью вспомогательного кубита фазы, что приводит к состоянию

$$|\psi_3\rangle = \sum_j \frac{-i}{2}(-1)^{g(\theta_j-\theta^*)}\left(e^{i(\theta_j)}|j\rangle|\omega_+\rangle_j|2\theta_j\rangle - e^{i(-\theta_j)}|j\rangle|\omega_-\rangle_j|-2\theta_j\rangle\right)$$

Опуская вычисление $|\psi_4\rangle$, выполняем вычисление *swap* теста, в результате получаем следующее состояние

$$|\psi_5\rangle = \sum_j (-1)^{g(\theta_j-\theta)}|j\rangle|0\rangle^n_{QNN1}|0\rangle^n_{QNN2}|0\rangle|0\rangle^t_{\text{амплитуда}}.$$

Как видно из данного уравнений, на вышеуказанных этапах был реализован оракул Гровера $O_{\text{Гровер}}$. После описанной выше процедуры к состоянию параметра была последовательно добавлена относительная фаза, которая зависит от функции стоимости *QNN* $\left|\langle p_j|t\rangle\right|^2$ и порогового значения. Важно отметить, что вычисление позволяет отделить регистр параметров от *QNN* и других регистров.

3. *Обучение QNN с помощью адаптивного QAOA*. Как показано на рис. 28, система квантового обучения *QNN* состоит из двух основных компонентов:

- *Фазовый оракул*. При этом функция стоимости *QNNs* последовательно кодируется в относительную фазу состояния суперпозиции в гильбертовом пространстве параметров;
- *Адаптивные микшеры*. Они используют скрытую структуру в задачах оптимизации *QNN* и, следовательно, могут обеспечить реализацию схем малой глубины.

Взаимодействия фазового оракула и адаптивных микшеров образуют процедуру *QAOA*, которая количественно определяет оптимальные параметры сети *QNN*.

Из-за характера алгоритма Гровера, основанного на глобальном поиске, квантовое обучение с использованием адаптивного поиска Гровера может обойти помехоустойчивое бесплодное плато; однако оно имеет определенные ограничения и недостатки, такие как: (1) не может справиться с вызванным шумом бесплодным плато; (2) требует экспоненциального числа вызовов к «контролируемому *QNN*» с чрезмерной длиной цепи и временем работы. Напротив, описанное квантовое обучение с совместным использованием адаптивного непрерывного *QAOA* может устранить ограничения, связанные с использованием адаптивного поиска Гровера.

Обсудим применение *QNN* в качестве квантового классификатора, который выполняет контролируемое обучение, что является стандартной задачей в машинном обучении.

Пример: *Обучение квантового классификатора.* Модель обучения, как и ранее, имеет следующий вид. Предположим, что X - набор входных данных, а Y - набор выходных данных. Задан набор данных $\mathrm{D} = \{(x_1, y_1),\ldots,(x_M, y_M)\}$ из пар так называемых обучающих входных данных $x_m \in \mathrm{X}$ и выходных данных цели обучения $y_m \in \mathrm{Y}$ для m = 1, ..., M, задача модели обучения состоит в том, чтобы предсказать выходные данные $y_m \in \mathrm{Y}$ для нового входного сигнала $x_m \in \mathrm{X}$. Для простоты в дальнейшем будем предполагать, что $\mathrm{X} = \square^N$ и $\mathrm{Y} = \{0,1\}$, что является задачей

бинарной классификации в *N*-мерном реальном входном пространстве. Таким образом, квантовый классификатор нацелен на изучение эффективной функции маркировки $\ell : \mathrm{X} \to \mathrm{Y}$.

Учитывая входные данные x_i и набор параметров θ, квантовый классификатор сначала вводит x_i в состояние квантовой системы с *n*-кубитами через схему подготовки состояния S_{x_i} таким образом, что $S_{x_i}|0\rangle = |\varphi(x_i)\rangle$, а затем использует обучаемую квантовую схему $U(\theta)$ (*QNN*) в качестве модели прогноза для принятия решений. Предсказанная метка класса $y^{(i)} = f(x_i, \theta)$ извлекается путем измерения указанного кубита в состоянии $U(\theta)|\varphi(x)\rangle$. Схема квантового классификатора представлена на рис. 33.

Таким образом, согласно рис. 33, и как предварительно отмечено выше, для точки обучающих данных (x_i, y_i) (слой (x_i, y_i), см., рис. 33) квантовый классификатор сначала вводит x_i в состояние квантовой системы с *n*-кубитами через схему внедрения данных S_{x_i} (фиолетовая рамка) таким образом, что $S_{x_i}|0\rangle = |\varphi(x_i)\rangle$ и впоследствии использует обучаемую квантовую схему $U(\theta)$ (*QNN*) в качестве модели логического вывода (здесь, для простоты, используем один и тот же символ θ для всех углов различных гейтов вращения $R(\theta)$ в слое *QNN*). Предсказанная метка класса $y^{(i)} = f(x_i, \theta)$ извлекается путем измерения указанного кубита в состоянии $U(\theta)|\varphi(x)\rangle$.

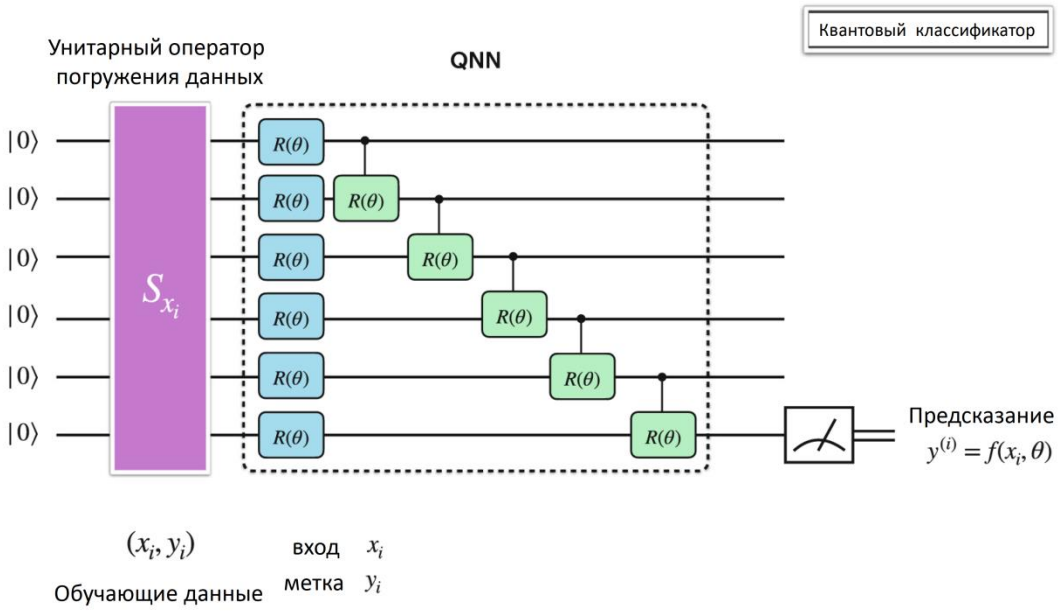


Рис. 33. Схема квантового классификатора

Примечание. Вариационный квантовый классификатор может работать как много классовый классификатор, здесь ограничиваемся случаем бинарной классификации, рассмотренным выше, и рассматриваем задачи с несколькими метками как набор подзадач бинарного распознавания.

Обозначим $p(\lambda)$ как вероятность того, что результат измерения на указанном кубите будет равен λ $(\lambda \in \{0,1\})$. Функция стоимости каждой точки обучающих данных $L_i(\theta)$, как функция y_i и $y^{(i)}$, и, следовательно, функция y_i, x_i, θ, которую обозначаем как $L(x_i, y_i, \theta)$, выбирается как вероятность получения результата измерения на проектируемый кубит, идентичный заданной метке, а именно $L_i(\theta) = L(x_i, y_i, \theta) := p(y_i)$. Тогда стоимость каждой выборки данных естественным образом закодирована в амплитуде выходного состояния *QNN*.

Здесь следует отметить, что чем больше вероятность, тем точнее прогноз, поэтому следует максимизировать затраты (чтобы соответствовать предыдущему описанию, используем “затраты” вместо обычно используемого “вероятности” для определения правильной метки для данных образец.)

С другой стороны, квантовое состояние системы (после подготовки состояния и вывода *QNN*) можно записать в виде

$$\left|\psi_i(\theta)\right\rangle = U(\theta)\left|\varphi(x_i)\right\rangle = \sqrt{p(0)}\left|0\right\rangle\left|u_\theta\right\rangle + \sqrt{1-p(0)}\left|1\right\rangle\left|\upsilon_\theta\right\rangle$$
$$= \begin{cases} \sqrt{p(y_i)}\left|0\right\rangle\left|u_\theta\right\rangle + \sqrt{1-p(y_i)}\left|1\right\rangle\left|\upsilon_\theta\right\rangle, \; y_i = 0 \\ \sqrt{p(y_i)}\left|0\right\rangle\left|u_\theta\right\rangle + \sqrt{1-p(y_i)}\left|1\right\rangle\left|\upsilon_\theta\right\rangle, \; y_i = 1 \end{cases},$$

в котором $\left|u_\theta\right\rangle$, $\left|\upsilon_\theta\right\rangle$ - некоторое нормализованное состояние, зависящее от θ.

Таким образом, стоимость каждой выборки данных $L(x_i, y_i, \theta)$ естественным образом кодируется в амплитуде выходного состояния $\left|\psi_i(\theta)\right\rangle$ *QNN*. Амплитудное кодирование функции затрат для квантового классификатора показано на рис. 34.

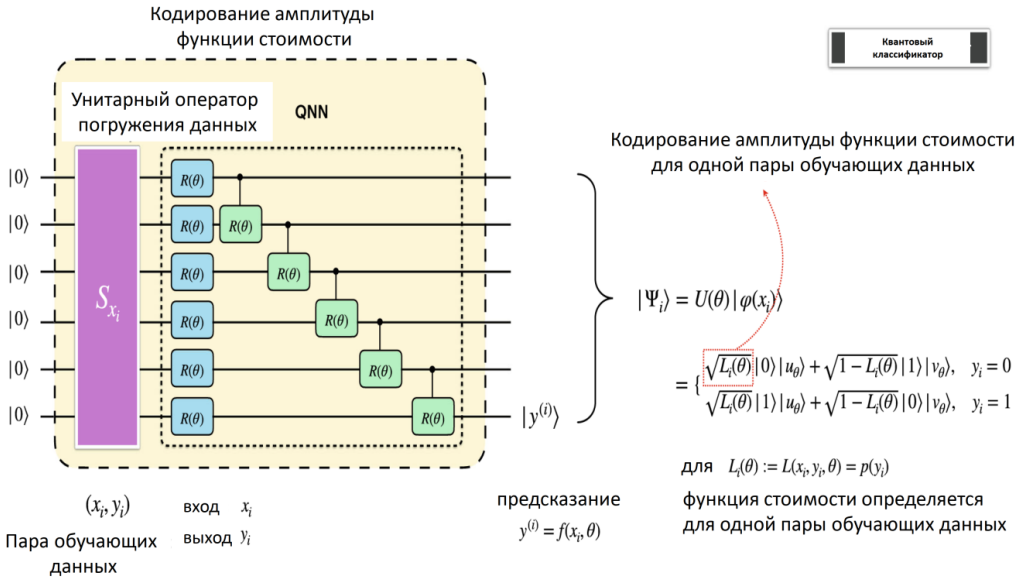


Рис. 34. Амплитудное кодирование функции стоимости для квантового классификатора

Построение “управляемой *QNN*” позволит параллельно кодировать амплитуду для всех конфигураций параметров.

Фазовое кодирование функции общей стоимости может быть реализовано путем накопления индивидуального фазового кодирования для каждой обучающей выборки; этот процесс можно проиллюстрировать на рис .35.

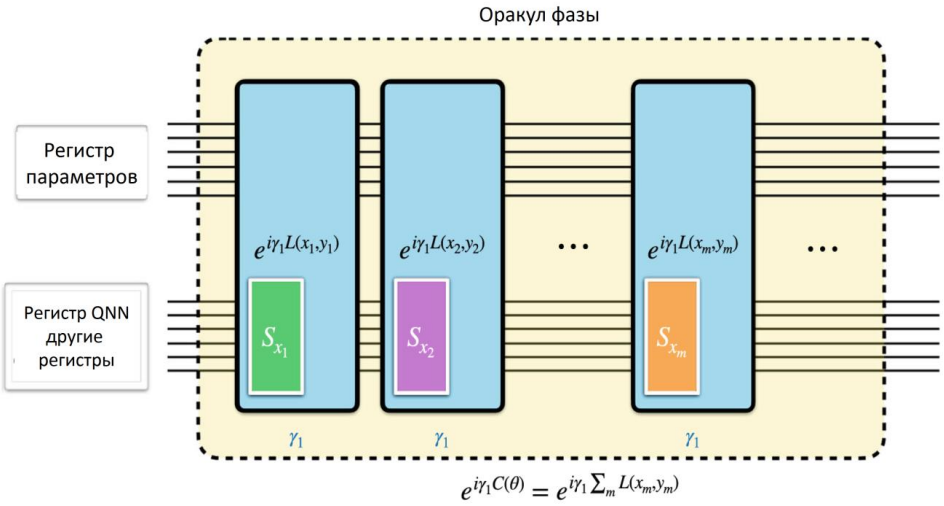


Рис. 35. Фазовое кодирование функции общей стоимости квантового классификатора

Функция общей стоимости всего обучающего набора может быть определена как $C(\theta) = \sum_i L(x_i, y_i, \theta)$. Следовательно, фазовое кодирование функции общих затрат (общий желтый прямоугольник) может быть реализовано путем накопления индивидуального фазового кодирования для каждой обучающей выборки (синие прямоугольники). Внутренние ячейки в синих полях представляют различные варианты встраивания данных для точек обучающих данных

На основе фазового кодирования функции общей стоимости, можем построить полный протокол квантового обучения, как показано на рис. 36.

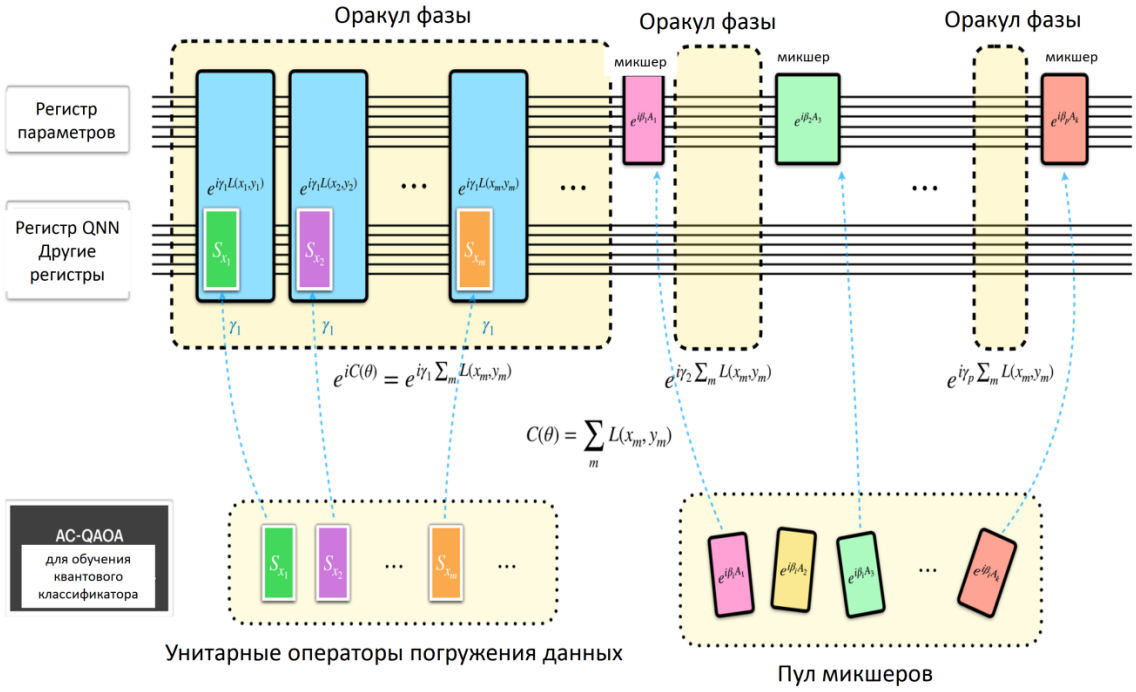


Рис. 36. Схема протокола квантового обучения для квантового классификатора

Полный протокол квантового обучения состоит из чередования фазового оракула, который обеспечивает согласованное фазовое кодирование функции затрат, и адаптивных микшеров, выбранных из пула микшеров. Фазовое кодирование функции общей стоимости для квантового классификатора подробно показано на рис. 35. Функция общей стоимости всего обучающего набора может быть определена как $C(\theta) = \sum_i L(x_i, y_i, \theta)$. Отсюда следует, что $e^{-i\gamma C(\theta)} = \prod_i e^{-i\gamma L(x_i, y_i, \theta)}$. Следовательно, фазовый оракул для функции общей стоимости (желтые прямоугольники в верхней

части этого рисунка) может быть реализован путем накопления индивидуального фазового кодирования для каждой обучающей выборки (синие прямоугольники). Цветные прямоугольники с белой каймой представляют различные варианты встраивания данных для точек обучающих данных. Цветные прямоугольники с черной каймой (за исключением синих для фазового кодирования) представляют различные выбранные микшеры.

После каждого измерения в регистре параметров получаем конкретную конфигурацию параметров *QNN*. Затем необходимо оценить функцию затрат для этой конкретной конфигурации параметров. Количество запусков *QNN* после каждой шкалы измерений равно $\mathrm{O}\left(1/\varepsilon\right)$, а общее количество *QNN* запусков всех измерений во время квантовой шкалы обучения равно $N_{\text{QNN}} \square \mathrm{O}\left(\frac{1}{\varepsilon}\right) N_{\text{измерений}}$. Для небольшого примера с 5-ю кубитами и 10 вентилями вращения в *QNN* для реализации протокола требуется примерно 60 кубитов. Таким образом, протокол реализуем на малых квантовых компьютерах.

Заключение

Рассмотрены цели и методы квантового машинного обучения как программно-алгоритмической платформы квантового «сильного» вычислительного ИИ. Приведено описание архитектуры и принципов работы трех наиболее применимых в квантовой инженерии методов квантового машинного обучения и области применения рассмотренных методов. Применение *QNN* рассмотрено в качестве квантового классификатора, который выполняет контролируемое обучение, что является стандартной задачей в машинном обучении.

Список источников

1. A high-bias, low-variance introduction to Machine Learning for physicists / P. Mehta [et al.] // Physics Reports. – 2019. – Vol. 810. – Pp. 1-124. – DOI: https://doi.org/10.1016/j.physrep.2019.03.001.
2. Machine learning and the physical sciences / G. Carleo [et al.] // Reviews of modern physics. – 2019. – Vol. 91. – No 4. – DOI: 10.1103/RevModPhys.91.045002.
3. AI for Next Generation Computing: Emerging Trends and Future Directions / S. S. Gill [et. al.] // arXive.org e-Print archive. – arXiv:2203.04159v1 [cs.DC] 5 Mar 2022.
4. Coles P. Thermodynamic AI and the fluctuation frontier // arXive.org e-Print archive. – arXiv:2302.06584v1 [cs.ET] 9 Feb 2023.
5. Intelligent Computing: The Latest Advances, Challenges and Future / Sh. Zhu [et al.] // arXive.org e-Print archive. – arXiv:2211.11281v1 [cs.AI] 21 Nov 2022.
6. Jha R. G. Notes on Quantum Computation and information // arXive.org e-Print archive. – arXiv:2301.09679v1 [quant-ph] 23 Jan 2023.
7. Dunjko V. Briegel H.J. Machine learning & artificial intelligence in the quantum domain: a review of recent progress // Reports on Progress in Physics. – 2018. – Vol. 81. – No 7. – Pp. 074001.– DOI: 10.1088/1361-6633/aab406.
8. Carrasquilla J. Machine learning for quantum matter // Advances in Physics X. – 2020. – Vol. 5. – No 1. – Pp. 1797528. – DOI: 10.1080/23746149.2020.1797528.
9. Wittek P. Quantum Machine Learning: What Quantum Computing Means to Data Mining. – Elsevier. – 2014.
10. Quantum machine learning for chemistry and physics / M. Sajjan [et al.] // Chemical Society Reviews. – 2022. – Vol. 51. – Pp. 6475. – DOI: 10.1039/d2cs00203e.
11. Haney B.S. Quantum machine learning: a patent review // JOURNAL OF LAW, TECHNOLOGY & THE INTERNET. – 2021. – Vol. 12 – No. 5.

12. Potential Applications of Quantum Computing at Los Alamos National Laboratory v0.1.0 / A. Bärtschi [et al.] // arXive.org e-Print archive. – arXiv: 2406.06625v1 [quant-ph] 7 Jun 2024.

13. Carlesso M. Lecture Notes on Quantum Algorithms in Open Quantum Systems // arXive.org e-Print archive. – arXiv:2406.11613v1 [quant-ph] 17 Jun 2024.

14. sQUlearn – A Python Library for Quantum Machine Learning / D. A. Kreplin [et al.] // arXive.org e-Print archive. – arXiv:2311.08990v1 [quant-ph] 15 Nov 2023.

15. Sahu H., Gupta H. P. Quantum Computing Toolkit From Nuts and Bolts to Sack of Tools // arXive.org e-Print archive. – arXiv:2302.08884v1 [quant-ph] 17 Feb 2023.

16. A Herculean task: Classical simulation of quantum computers / X. Xu [et al.] // arXive.org e-Print archive. – arXiv:2302.08880v1 [quant-ph] 17 Feb 2023.

17. Quantum Machine Learning for 6G Communication Networks: State-of-the-Art and Vision for the Future/ S. J. Nawaz [et al.] // IEEE Access. – 2019. – Vol.7. – Pp. 46317-46350. – DOI: 10.1109/ACCESS.2019.2909490.

18. Bötticher A., Seskir Z. C., Ruhland J. Introducing a Research Program for Quantum Humanities – Theoretical Implications // arXive.org e-Print archive. – arXiv:2212.12947 [physics.soc-ph] 25 Dec 2022.

19. Chen S.Y.-C., Yoo S. Federated Quantum Machine Learning // Entropy. – 2021. – Vol. 23. – No 4. – Pp. 460. – DOI: https://doi.org/10.3390/e23040460.

20. Quantum Federated Learning Experiments in the Cloud with Data Encoding / Sh. R. Pokhrel [et al.] // arXive.org e-Print archive. – arXiv:2405.00909v1 [cs.LG] 1 May 2024.

21. Quantum Distributed Deep Learning Architectures: Models, Discussions, and Applications / Y. Kwak [et al. // arXive.org e-Print archive. – arXiv:2202.11200v3 [quant-ph] 7 Apr 2022.

22. Abbas Φ. et al. The power of quantum neural networks // Nature computational science. – 2021. – Vol. 1. – Pp. 403–409. – DOI: https://doi.org/10.1038/s43588-021-00084-1.

23. Oh S., Choi J., Kim J. A Tutorial on Quantum Convolutional Neural Networks (QCNN) // arXive.org e-Print archive. – arXiv:2009.09423v1 [quant-ph] 20 Sep 2020.

24. A Review of Barren Plateaus in Variational Quantum Computing / M. Larocca [et al.] // arXive.org e-Print archive. – arXiv:2405.00781v1 [quant-ph] 1 May 2024.

25. TensorFlow Quantum: A Software Framework for Quantum Machine Learning / M. Broughton [et al.] // arXive.org e-Print archive. – arXiv:2003.02989v2 [quant-ph] 26 Aug 2021.

26. Quantum Federated Learning with Entanglement Controlled Circuits and Superposition Coding / W. J. Yun [et al.] // arXive.org e-Print archive. – arXiv:2212.01732v1 [quant-ph] 4 Dec 2022.

27. Tensor Circuit: a Quantum Software Framework for the NISQ Era / S.-X. Zhang [et al.] // arXive.org e-Print archive. – arXiv:2205.10091v2 [quant-ph] 27 Jan 2023.

28. Kulkarni V., Pawale S., Kharat A. A Classical-Quantum Convolutional Neural Network for Detecting Pneumonia from Chest Radiographs // arXive.org e-Print archive. – arXiv:2202.10452v1 [cs.CV] 19 Feb 2022.

29. Hur T., Araujo I. F., Park D. K. Neural Quantum Embedding: Pushing the Limits of Quantum Supervised Learning // arXive.org e-Print archive. – arXiv:2311.11412v1 [quant-ph] 19 Nov 2023.

30. Nguyen N., Chen K.-C. Quantum Embedding Search for Quantum Machine Learning // arXive.org e-Print archive. – arXiv:2105.11853v1 [quant-ph] 25 May 2021.

31. Explainable Quantum Machine Learning / R. Heese [et al.] // arXive.org e-Print archive. – arXiv:2301.09138v1 [quant-ph] 22 Jan 2023.

32. Quantum Machine Learning: Foundation, New Techniques, and Opportunities for Database Research / T. Winker [et al.] // SIGMOD-Companion '23, June 18–23, 2023, Seattle, WA, USA. – 2023. – Pp. 45-52. – DOI: https://doi.org/10.1145/3555041.3589404.

33. Li Weikang, Lu Zhide, Deng Dong-Ling. Quantum neural network classifiers: A tutorial // SciPost Physics Lecture Notes. – 2022. – Vol. 61. – DOI: https://doi.org/10.21468/SciPostPhysLectNotes.61.

34. Generative Modeling with Quantum Neurons / K. Gili, R. S. Kumar, M. Sveistrys, C. J. Ballance // arXive.org e-Print archive. – arXiv:2302.00788v1 [quant-ph] 1 Feb 2023.

35. Gili K., Sveistrys M., Balance C. Introducing Non-Linear Activations into Quantum Generative Models // arXive.org e-Print archive. – arXiv:2205.14506v4 [quant-ph] 8 Dec 2022.

36. A Survey on Quantum Reinforcement Learning / N. Meyer [et al.] // arXive.org e-Print archive. – arXiv:2211.03464v1 [quant-ph] 7 Nov 2022.

37. Variational quantum algorithms / M. Cerezo [et al.] // Nature reviews physics. – 2021. – Vol. 3. – Pp. 625-644. – DOI: https://doi.org/10.1038/s42254-021-00348-9.

38. Noisy intermediate-scale quantum algorithms / K. Bharti [et al.] // Reviews of Modern Physics. – 2022. – Vol. 94. – No 1. – Pp. 015004-1-015004-69. – DOI: https://doi.org/10.1103/RevModPhys.94.015004.