НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ноябрь–декабрь 2022 Том 22 № 6 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
November–December 2022 Vol. 22 No 6 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

doi: 10.17586/2226-1494-2022-22-6-1178-1186
УДК 004.8+ 65.011.56

Мультиагентная адаптивная маршрутизация агентами-клонами на основе многоголового внутреннего внимания с использованием обучения с подкреплением

Тимофей Александрович Грибанов[1], Андрей Александрович Фильченков[2]✉, Артур Александрович Азаров[3], Анатолий Абрамович Шалыто[4]

[1,2,3,4] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[3] Северо-Западный институт управления – филиал РАНХиГС, Санкт-Петербург, 199178, Российская Федерация

[1] t.hrybanau@gmail.com, https://orcid.org/0000-0002-1151-3405
[2] afilchenkov@itmo.ru✉, https://orcid.org/0000-0002-1133-8432
[3] artur-azarov@yandex.ru, https://orcid.org/0000-0003-3240-597X
[4] shalyto@mail.ifmo.ru, https://orcid.org/0000-0002-2723-2077

Аннотация

Предмет исследования. Регулярным условием, характерным для пакетной маршрутизации, а также задач транспортировки грузов и управления потоками, является изменчивость графа, на котором осуществляется маршрутизация. Это условие учитывают алгоритмы адаптивной маршрутизации, использующие обучение с подкреплением. Однако при значительных изменениях графа существующим алгоритмам маршрутизации требуется полное переобучение. **Метод.** Предложен новый метод, основанный на мультиагентном моделировании с агентами-клонами, с использованием новой архитектуры нейронной сети с многоголовым внутренним вниманием, которая предобучена в рамках парадигмы обучения с нескольких взглядов. Агент в такой парадигме использует вершину как вход, а его клоны помещены в вершины графа и осуществляют выбор соседа, которому следует передать объект. **Основные результаты.** Выполнен сравнительный анализ с существующим алгоритмом мультиагентной маршрутизации *DQN-LE-routing* по следующим этапам: предобучение и симуляция. Для каждого этапа рассмотрены запуски с помощью изменения топологии в процессе тестирования или симуляции. Эксперименты показали, что предложенный метод повышения адаптивности обеспечивает глобальную адаптивность, увеличивая время доставки при глобальных изменениях не более чем на 14,5 % от оптимального. **Практическая значимость.** Предложенный метод может быть использован для решения задач маршрутизации со сложными функциями оценки пути и динамически меняющимися топологиями графов, например, в транспортной логистике и для управления конвейерными лентами на производстве.

Ключевые слова

маршрутизация, мультиагентное обучение, обучение с подкреплением, адаптивная маршрутизация

Благодарности

Исследование выполнено за счет гранта Российского научного фонда (проект № 20-19-00700).

Ссылка для цитирования: Грибанов Т.А., Фильченков А.А., Азаров А.А., Шалыто А.А. Мультиагентная адаптивная маршрутизация агентами-клонами на основе многоголового внутреннего внимания с использованием обучения с подкреплением // Научно-технический вестник информационных технологий, механики и оптики. 2022. Т. 22, № 6. С. 1178–1186. doi: 10.17586/2226-1494-2022-22-6-1178-1186

Multi-agent adaptive routing by multi-head-attention-based twin agents using reinforcement learning

Timofey A. Gribanov[1], Andrey A. Filchenkov[2]✉, Artur A. Azarov[3], Anatoly A. Shalyto[4]

[1,2,3,4] ITMO University, Saint Petersburg, 197101, Russian Federation

[3] North-West Institute of Management – branch of the Russian Presidential Academy of National Economy and Public Administration, Saint Petersburg, 199178, Russian Federation

[1] t.hrybanau@gmail.com, https://orcid.org/0000-0002-1151-3405

[2] afilchenkov@itmo.ru✉, https://orcid.org/0000-0002-1133-8432

[3] artur-azarov@yandex.ru, https://orcid.org/0000-0003-3240-597X

[4] shalyto@mail.ifmo.ru, https://orcid.org/0000-0002-2723-2077

Abstract

A regular condition, typical for packet routing, for the problem of cargo transportation, and for the problem of flow control, is the variability of the graph. Reinforcement learning based adaptive routing algorithms are designed to solve the routing problem with this condition. However, with significant changes in the graph, the existing routing algorithms require complete retraining. To handle this challenge, we propose a novel method based on multi-agent modeling with twin-agents for which new neural network architecture with multi-headed internal attention is proposed, pre-trained within the framework of the multi-view learning paradigm. An agent in such a paradigm uses a vertex as an input, twins of the main agent are placed at the vertices of the graph and select a neighbor to which the object should be transferred. We carried out a comparative analysis with the existing *DQN-LE-routing* multi-agent routing algorithm on two stages: pre-training and simulation. In both cases, launches were considered by changing the topology during testing or simulation. Experiments have shown that the proposed adaptability enhancement method provides global adaptability by increasing delivery time only by 14.5 % after global changes occur. The proposed method can be used to solve routing problems with complex path evaluation functions and dynamically changing graph topologies, for example, in transport logistics and for managing conveyor belts in production.

Keywords

routing, multi-agent learning, reinforcement learning, adaptive routing

Acknowledgements

The study was supported by the grant from the Russian Science Foundation (project no. 20-19-00700).

For citation: Gribanov T.A., Filchenkov A.A., Azarov A.A., Shalyto A.A. Multi-agent adaptive routing by multi-head-attention-based twin agents using reinforcement learning. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2022, vol. 22, no. 6, pp. 1178–1186 (in Russian). doi: 10.17586/2226-1494-2022-22-6-1178-1186

Введение

Задача маршрутизации состоит в поиске оптимального пути между двумя вершинами в графе [1]. Для решения данной классической оптимизационной постановки применяются алгоритмы дискретной математики, которые можно разделить на два типа: дистанционно-векторные алгоритмы (*distance-vector*) [2] и алгоритмы состояния канала связи (*link-state*) [3], использующие алгоритмы кратчайших путей.

Более сложные постановки учитывают специфику предметных областей или конкретных условий, в которых необходимо осуществлять маршрутизацию. Регулярным условием, характерным для пакетной маршрутизации, а также задач транспортировки грузов и управления потоками, является изменчивость графа, на котором осуществлена маршрутизация. Адаптивная маршрутизация [4, 5] — направление исследований, в котором предполагается находить решения для задачи маршрутизации, устойчивые к такого рода изменениям. Алгоритмы адаптивной маршрутизации основаны на машинном обучении и преимущественно используют мультиагентный подход на основе обучения с подкреплением [6, 7]. Отметим, что появляются научные работы, основанные на других принципах машинного обучения, например — на использовании трансформеров [8].

Изменения в топологии можно условно разделить на локальные и глобальные. Локальные — удаление одной вершины или одного ребра, глобальные — существенное изменение всей топологии графа, например, его увеличение или уменьшение в два раза. Так как текущие алгоритмы маршрутизации с использованием обучения с подкреплением в своей основе содержат мультиагентное обучение [9, 10], они очень хорошо адаптируются к локальным изменениям, которые происходят в контексте одной вершины. Однако в случае глобальных изменений этим алгоритмам требуется полное переобучение, так как они адаптированы еще к старому графу.

В настоящей работе предложен алгоритм, решающий проблему глобальной адаптации с использованием обучения с подкреплением.

Алгоритмы маршрутизации

Маршрутизация и адаптивная маршрутизация. Пусть дан граф $G = \langle V, E \rangle$, $E = \{(u,v,param) | u,v \in V\}$, где *param* — свойства ребра, такие как пропускная способность или задержка. Также могут быть известны дополнительные параметры в зависимости от конкретной области применения, в том числе и к вершинам. Задана стоимость доставки объекта между каждой парой вершин $cost(u,v,cont)$, где $u,v \in V$ и *cont* — обстоятельства передачи (например, время).

Большинство современных алгоритмов адаптивной маршрутизации направлены на работу со временем или стоимостью доставки, которые задаются переберно, и

время (стоимость) маршрута — как сумма значений функций для каждого ребра, принадлежащего маршруту. Такие функции называются декомпозируемыми. В более сложном случае оценка решения может производиться недекомпозируемыми функциями.

Изменения в топологии выполним путем удаления или восстановления ребер. Строгого и формального разделения на локальные и глобальные изменения провести не представляется возможным, однако, назовем изменение локальным, если оно включает удаление или восстановление одного ребра на значительном промежутке времени, чтобы алгоритм успел провести адаптацию и показать релевантные результаты. Глобальные изменения — удаление, восстановление, добавление вершин и соответствующих им ребер.

Для оценки глобальных изменений введем понятие δ-окрестности графа. δ-окрестность исходного графа $G = \langle V, E \rangle$ — любой граф, полученный из исходного с помощью удаления или добавления δ вершин из или в граф. Таким образом, новый граф $G' = \langle V', E' \rangle$ будет удовлетворять следующему неравенству: $|V| - \delta \le |V'| \le |V| + \delta$.

Алгоритмы адаптивной маршрутизации недекомпозируемых функций. Для решения проблемы изменений в топологии большое распространение получили алгоритмы, использующие обучение с подкреплением. Родоначальником таких алгоритмов стал *Q-routing* [10], применяющий мультиагентный подход для обучения. В данном алгоритме каждой вершине соответствует отдельный независимый агент, который отвечает только за маршрутизацию через одну свою вершину. Сами агенты используют алгоритм обучения с подкреплением *Q-learning* [11], который после каждого получения награды обновляет функцию полезности Q состояний, на основе которой впоследствии будет выбрана стратегия принятия решения. В *Q-routing* каждому агенту соответствует таблица $Q(n,d)$ с оценками минимальной стоимости пути от текущей вершины до вершины d, если следующей вершиной на пути будет вершина n, которая обновляется за счет коммуникации между агентами после пересылок. Такая модель доказала свою эффективность при локальных изменениях и показала высокий уровень адаптации по сравнению с дистанционно-векторными алгоритмами и алгоритмами состояния канала связи. Однако алгоритм *Q-routing* предполагает, что оптимизуемая функция декомпозируема.

Более современные решения для аппроксимации Q-функции используют глубокие сети. Первой была предложена *Deep Q-Network* (*DQN*) [12]. На основе этого решения разработан алгоритм маршрутизации *DQN-routing* [13], объединяющий алгоритм *Q-routing* и идеи *DQN*, который привнес следующие изменения.

— Вместо таблицы Q использована нейронная сеть с прямым распространением, а обновления происходят в ее весах при помощи градиентного спуска.
— Оценка минимальной стоимости пути от текущей вершины до конечной заменена на математическое ожидание оценок при возможных переходах.
— Для принятия решения о следующей вершине в пути вместо простого нахождения минимума среди возможных переходов использована функция *Softmax* [14], которая применена к полученным значениям.
— Использован этап предобучения с помощью обучения с учителем. Данные для предобучения сгенерированы случайным образом на имеющемся графе без изменений, а в качестве правильного ответа взята длина кратчайшего пути (алгоритм Дейкстры).

Архитектура нейросети — сеть прямого распространения с двумя слоями по 64 нейрона. В качестве функции активации выбран гиперболический тангенс. На вход нейронной сети подаются вершины в *one-hot* кодировании: текущая вершина, конечная вершина объекта и соседи текущей вершины. Также в качестве дополнительного параметра на вход подается матрица смежности графа.

Алгоритм *DQN-routing* показал приемлемые результаты для оптимизации сложных функций, оценивающих качество доставки, однако его недостатком является размер нейронной сети. Этот размер зависит от размера самого графа, так как на его выходе сформирован вектор предсказаний для всех вершин.

Следующим логичным усовершенствованием алгоритма *DQN-routing* стал *DQN-LE-routing* [15]. Все основные компоненты алгоритма *DQN-routing* оставлены, изменения носят точечный характер и служат для избавления от его недостатков: вместо *one-hot* кодирования подающихся на вход вершин использованы их графовые векторные представления на основе алгоритма *Laplacian Eigenmaps* (*LE*) [16]; на входе графа подаются не все соседи сразу, а только один, и аналогично на выходе получается предсказание только для одного конкретного соседа; функцией активации вместо гиперболического тангенса стала *ReLu*: $\max(x, 0)$.

Описание разработанного метода

Основные положения нового метода. Предлагаемый метод основан на трех идеях.

1. Алгоритм должен рассматривать текущую вершину как вход. Данный алгоритм, будучи обученным, позволяет добавлять новые вершины к топологии, не переобучаясь каждый раз на новой. В результате исследований возможных способов реализации данной идеи осуществлен переход на обучение модели на всем графе, а не на каждой вершине отдельно. Полученный алгоритм остается мультиагентным, однако все агенты являются клонами (*twin agents*) [17].
2. Вследствие отказа от мультиагентности, модель единого агента должна быть сложнее, чем модели агентов, использующихся в рассмотренных в разделе «Алгоритмы маршрутизации» алгоритмах, поскольку он должен быть способен оперировать информацией как о графе в целом, так и о ситуации в конкретных вершинах.
3. Обучение сложных моделей требует большей обучающей выборки, поэтому использовано множественное предобучение на графах разного размера в пределах некоторой δ-окрестности изначального графа. Данное условие дает возможность применения предобученной модели на любом случайном графе в этой δ-окрестности.

Ранее в предобучении для одного графа использовалось большое количество синтетических данных, на несколько порядков превышающее размеры самого графа. Это давало возможность обучить нейронную сеть максимально точно под конкретный граф. В новом подходе нет необходимости в точном обучении под конкретный граф, более того, оно может ухудшать результаты, так как каждый раз будет возникать проблема переобучения нейронной сети на каждом графе. Для успешной работы предобучение должно охватывать как можно больше разной информации от разных графов из δ-окрестности. По этой причине для каждого нового графа при множественном обучении генерируется небольшой набор синтетических данных, что позволяет увеличить число графов, используемых при обучении. Такая задача в машинном обучении имеет название обучение с нескольких взглядов (*few-shot learning*) [18] и заключается в том, что нейронная сеть обучается на данных, содержащих в себе ограниченное количество информации. Такой способ обучения позволяет модели быть более обобщенной, что в контексте задачи настоящей работы означает способность обобщаться по графам. Традиционные модели, обученные классическим способом, не способны различать классы, отсутствующие в обучающем наборе данных, в то время как метод обучения с нескольких взглядов позволяет нейронным сетям разделять два и более классов, которых нет в обучающих данных.

Мультиагентное управление с агентами-клонами. Для сохранения совместимости с моделью мультиагентного обучения использовано множество агентов. Все агенты, прикрепленные к вершинам, имеют второстепенный характер и привязаны к одному дополнительно введенному главному агенту, передающего логику обработки решений по маршрутизации.

Управляющая модель основана на отправлениях сообщений между агентами, которые определяют их логику поведения. Основные сообщения бывают двух видов: необходимость совершить какое-то действие (переслать объект для маршрутизации); получение новой информации (награды за совершенное раннее действие). Также имеются другие служебные сообщения, которые необходимы для полноценной работы симуляции.

В управляющей системе каждый второстепенный агент хранит в себе лишь информацию о том, к какой вершине он принадлежит, и ссылку на главного агента. При получении сообщения второстепенные агенты пересылают его главному агенту, добавляя к сообщению информацию о своей вершине.

Обработка сообщений главным агентом схожа с аналогичной обработкой любым из агентов рассмотренных ранее алгоритмов. Добавим функционал для поддержки сообщений от второстепенных агентов и чтения информации о вершине, которая теперь приходит вместе с сообщением, а не хранится в самом агенте. Изменения в схеме взаимодействия агентов при исходном мультиагентном управлении и управлении на основе агентов-клонов показаны на рис. 1.

Архитектура нейронной сети. Рассмотрим вход предлагаемой нейронной сети. При одном агенте информации только о текущей и конечной вершинах недостаточно, так как нейронная сеть теперь должна делать предсказания в контексте всего графа. Для решения данной задачи необходимо поочередно подавать на вход векторные представления всех соседей вершины, в которую пришел объект. Такой же подход применен и в алгоритме *DQN-LE-routing*. Ключевое отличие — добавление матрицы смежности к входным переменным, так как необходимо, чтобы нейронная сеть была способна подстраиваться и обучаться на графах разных размеров и топологий, а информации о соседних вершинах в таком случае недостаточно.

Отметим, что добавление матрицы смежности для использования на графах разных размерностей добав-

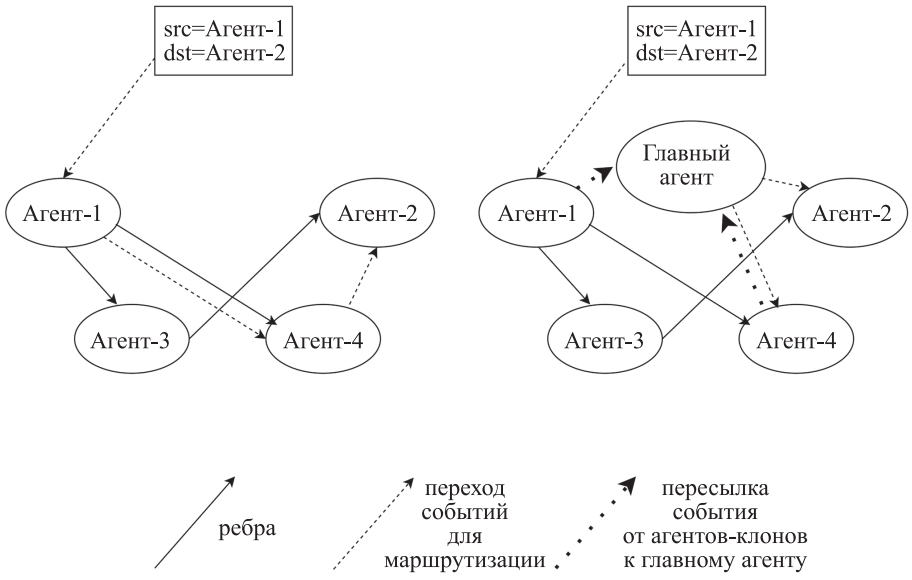

Рис. 1. Изменение схемы взаимодействия агентов в управляющей модели

scr — текущая вершина, dst — вершина назначения

Fig. 1. Changes in the agent interaction schemes in the control model, scr is the operating vertex, dst is the destination vertex

ляют новую проблему, суть которой состоит в том, что число значений на входе у нейронной сети не является константой. В основу решения проблемы входит понятие δ-окрестности графа. Значение δ задается и фиксируется в самом начале работы алгоритма, поэтому число вершин в наибольшем возможном графе в процессе всех изменений будет равно $n_{\max} = |V| + \delta$. Это ограничение позволяет передавать на вход нейронной сети матрицу смежности всегда определенного размера $n_{\max} \times n_{\max}$, которая будет заполняться нулевыми строками и столбцами в случае, когда текущий граф не максимального размера.

Для того чтобы увеличить сложность модели, использован механизм многоголового внутреннего внимания (*multi-head self-attention*) [19], способный к выявлению закономерности между разнесенными по времени подачи входами.

Многоголовое внимание в предлагаемой архитектуре применяется дважды. Первый слой внимания распространяется на векторные представления вершин на входе для нахождения зависимостей в них. Для второго слоя механизма внимания к результату первого слоя добавляется вектор того же размера, отвечающий за матрицу смежности.

На рис. 2 представлена схема работы механизма внимания для преобразования векторного представления текущей вершины *scr*, используя информацию о векторных представлениях конечной и смежной вершин. В качестве *query*, *key* и *value* использовано одно и то же входное векторное представление соответствующей вершины.

Другим важным аспектом архитектуры нейронной сети являются слои нормализации. Нормализация предотвращает сильное изменение диапазона значений в слоях, что приводит к тому, что модель обучается быстрее и обладает лучшей способностью к обобщению [20].

Для использования второго слоя механизма внимания с помощью простой сети прямого распространения, состоящей из двух слоев, матрица смежности преобразовывается в вектор той же длины, что и векторное представление вершин. Затем этот вектор добавляется к измененным после первого слоя механизма внимания векторным представлениям вершин на входе, и полученные четыре вектора передаются на второй слой.

В связи с тем, что слой нейронной сети, представляющий сеть прямого распространения недостаточен для покрытия возросшего количества входных данных, он был расширен до четырех линейный слоев и одного *dropout*-слоя [21]. Также увеличено число нейронов в линейных слоях. Архитектура новой нейронной сети представлена на рис. 3.

Предобучения агента. Предобучение выполнено на синтетически сгенерированных данных под существующую нейронную сеть в конкретном графе. Процесс предобучения разбит на несколько этапов, в начале каждого из которых граф изменяется путем удаления или добавления δ вершин. Таким образом, за счет сэмплирования аппроксимируется предобучение на всех графах из δ-окрестности графа *G*. Суммарно итераций предобучения на δ-окрестностях производится достаточно много, но для каждого отдельного графа из этой окрестности их осуществляется достаточно мало. В результате нейронная сеть обучается на большом количестве разрозненных данных, что позволяет ей охватить и обучиться на всей δ-окрестности графа.

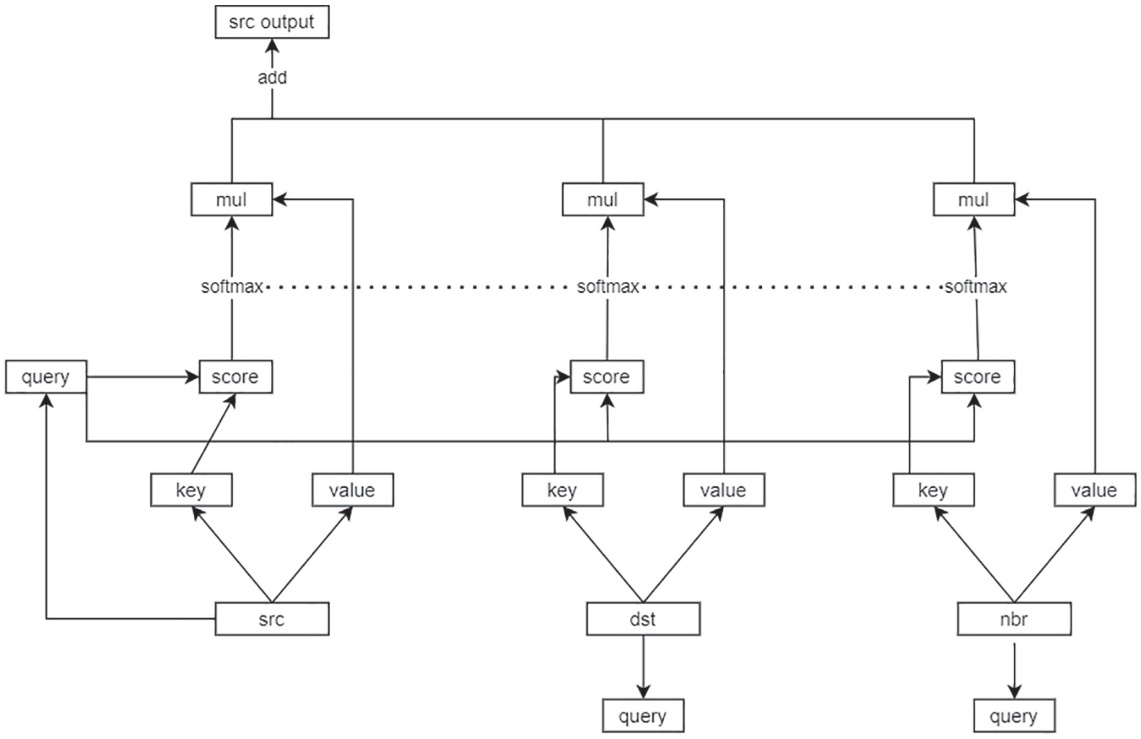

Рис. 2. Фрагмент архитектуры с внутренним вниманием для обработки входящей вершины scr

Fig. 2. Scheme of the self-attention mechanism for processing an input vertex scr

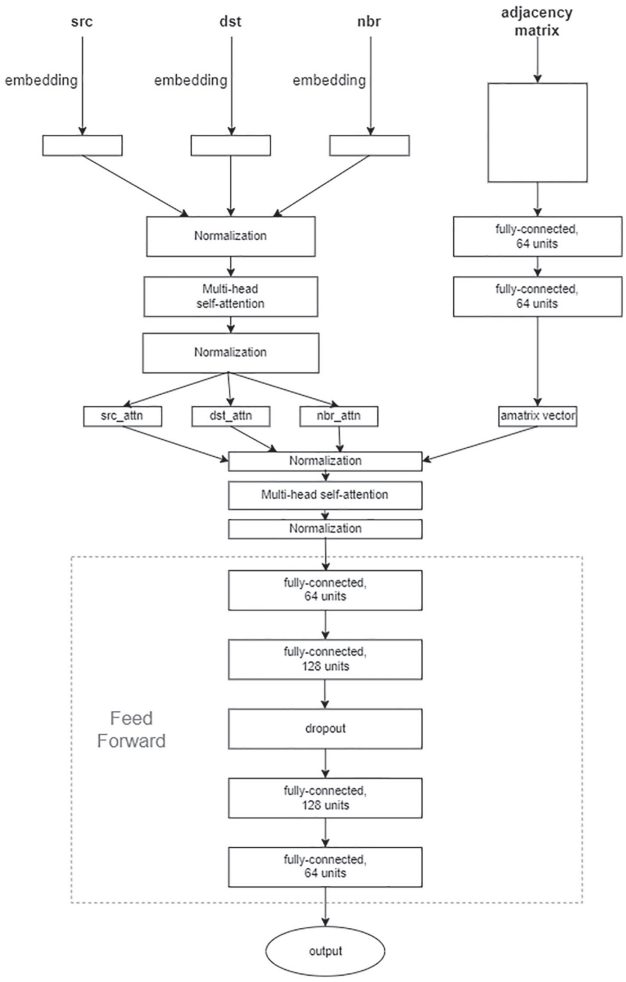

Рис. 3. Предлагаемая архитектура нейронной сети агента
Fig. 3. The proposed architecture of the neural network of the agent

Генерация синтетических данных для предобучения сопровождается дополнительным параметром, который означает, на сколько и в какую сторону нужно изменить граф. Удаление вершин происходит случайным образом и сопровождается проверкой связности полученного графа. При добавлении вершин требуется и добавление новых случайных ребер, число которых определяется также случайным образом в пределах некоторого диапазона, который задается в зависимости от размеров изначального графа. В качестве данных используются следующие значения: случайная начальная и конечная вершины; одна из вершин-соседей; наименьшая стоимость пути от стартовой вершины до конечной, проходящего через соответствующую смежную вершину; матрица смежности графа для возможности подсчета векторных представлений вершин во время предобучения.

Матрица смежности в генераторе синтетических данных передается заведомо большого размера для покрытия всей δ-окрестности. При этом во время предобучения для подсчета векторных представлений вершин матрица трансформируется в своей оригинальный вид без лишних строк и столбцов. Это необходимо для корректного подсчета векторных представлений вершин алгоритмом *LE* [16].

Описание работы агента. За предобучением следует фаза работы агента в среде. Агент использует ту же нейронную сеть, которая прошла стадию предобучения, и поэтому создает значительный прирост в показателях метрик. Так как обучению с подкреплением не приходится собирать информацию с нуля, это может привести к тому, что нейросеть «не сойдется» к оптимальному решению даже в простейших случаях [13].

В связи с измененной архитектурой нейронной сети, алгоритм обучения с подкреплением, отвечающий за интерпретируемый результат маршрутизации — определение следующей вершины на пути, состоит из следующих шагов.

Шаг 1. Нейронная сеть возвращает ожидаемую стоимость пути до конечной вершины при переходе в конкретного соседа.

Шаг 2. Запустив шаг 1 для каждого соседа, формируется список стоимостей возможных путей. Выбрав минимальный путь из полученного списка, можно было бы перейти в соответствующего соседа, но это — детерминистическая стратегия, и она не подходит для используемого подхода.

Шаг 3. К полученному списку применим стандартную функцию *Softmax* [14] для получения распределения вероятностных переходов, формируя тем самым стохастическую стратегию агента.

Шаг 4. На основе полученного распределения случайным образом выберем соседа, к которому будет совершен переход.

Экспериментальное исследование

Проведено экспериментальное исследование, которое состоит из двух частей. В первой части исследован этап предобучения, на котором проверено, что разработанный алгоритм действительно улучшает качество работы при увеличении числа подаваемых ему на вход графов. Во второй части проведено сравнение с алгоритмом *DQN-LE-routing* [15].

Для запуска процесса предобучения один из главных параметров — значение δ, в окрестности которого изменяются графы и происходит обучение. В качестве исходного графа для экспериментов выберем граф с 12 вершинами и 17 ребрами.

Для сравнения используем симуляционную систему, на которой проведено исследование алгоритма *DQN-LE-routing*. В системе смоделируем конвейерную систему, и оптимизируемую функцию, которая является суммой времени доставки и затраченной на передачу электроэнергии, зависящей от маршрутизации всех сумок.

Результаты предобучения разработанного метода. В предобучении используем метод нескольких взглядов. В результате сгенерируем большое число графов разных размеров из δ-окрестности исходного графа путем удаления случайных вершин с соответствующими ребрами или добавления новых случайно сгенерированных вершин. Отметим, что на вход нейронной сети подавался ограниченный набор синтетических данных для каждого сгенерированного графа.

Значение δ увеличим итеративно от запуска к запуску, таким образом проверяя модель на все бóльших

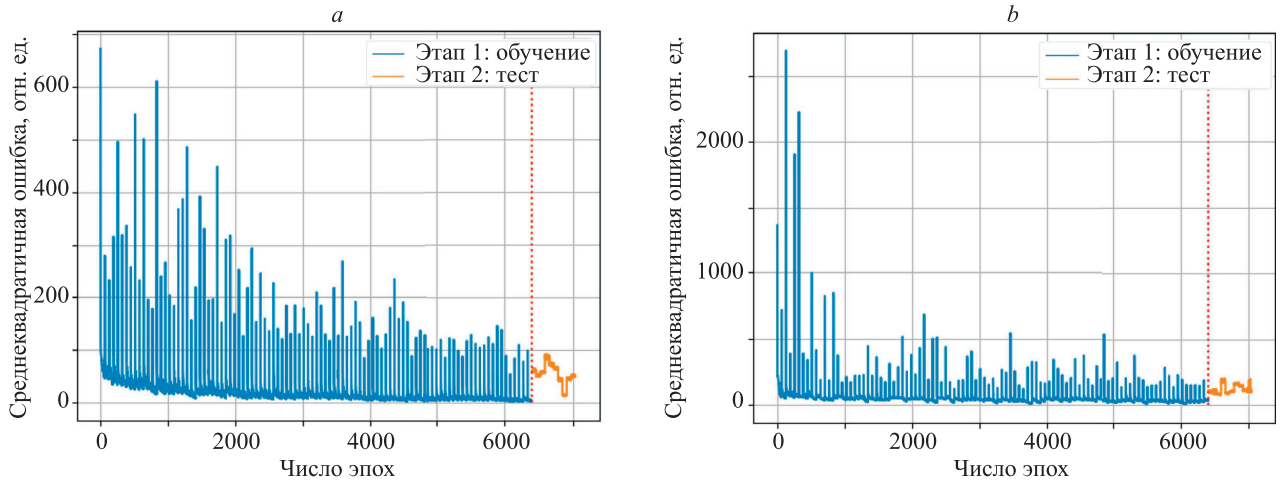

Рис. 4. Результаты предобучения со значениями δ = 1 (*a*) и δ = 5 (*b*)

Fig. 4. Results of pre-training with δ = 1 (*a*) и δ = 5 (*b*)

и бóльших изменениях. В качестве функции потерь выбрана среднеквадратичная ошибка. На рис. 4 показаны этапы результатов предобучения для стартового и последнего исследованных значений δ. Этап 1 — предобучение с градиентным спуском, которое длится до момента пересечения с красной пунктирной линией. Этап 2 — проверка нейронной сети.

Видно, что на графиках при этапе 1 кривые стабильно сходятся с присутствием пиков, которые соответствуют моментам изменения графа. Значения пиков убывают, вследствие чего можно сделать вывод, что нейронная сеть успешно обучается в текущей δ-окрестности, и при изменении топологии она не переобучается с нуля, о чем в том числе свидетельствует результаты этапа 2. Отметим, что сразу после достижения пика функция ошибки стремительно идет вниз, что означает успешное обучение нейронной сети и в рамках одного графа.

Так как цель предлагаемого алгоритма — выявить его способность адаптироваться к как можно большим изменениям, произведем запуск с увеличенным значением δ (рис. 4, *b*). Видно, что предложенный метод успешно справляется даже при изменениях топологии в два раза.

Сравнение с существующим алгоритмом. В качестве сравнения предобучения выбран алгоритм *DQN-LE-routing*, так как он показал лучшие результаты в адаптивной маршрутизации. Результаты сравнения при значении δ = 3 представлены на рис. 5. Видно, что предложенный в данной работе новый метод значительно превосходит показатели алгоритма *DQN-LE-routing*.

Для сравнения результатов симуляции был взят алгоритм *DQN-LE-routing* в единственном экземпляре. После предобучения алгоритм *DQN-LE-routing* и новый метод запущены в симуляционной модели, в которую была добавлена возможность полностью изменять топологию. Результаты запусков представлены на рис. 6.

Из рис. 6, *a* видно, что после изменения топологии, которое произошло в середине всей симуляции, алгоритм *DQN-LE-routing* более чем в пять раз ухудшил свое качество, после чего дообучился, однако все еще не достиг первоначальных показателей. При этом для разработанного метода изменение топологии никак не повлияло на результаты.

На рис. 6, *b* в процессе симуляции выполнено два изменения топологии. Заметно большое превосходство нового метода, над существующим *DQN-LE-routing*.

В рассмотренных экспериментах значение среднего времени доставки разработанного алгоритма при глобальных изменениях увеличивалось не более чем на 14,5 % в отличие от алгоритма *DQN-LE-routing*, у которого среднее время доставки увеличивалось в разы. На основе проведенных экспериментов можно заключить, что предложенный метод успешно сходится как на этапе предобучения, так и при последующей симуляции с использованием обучения с подкреплением.

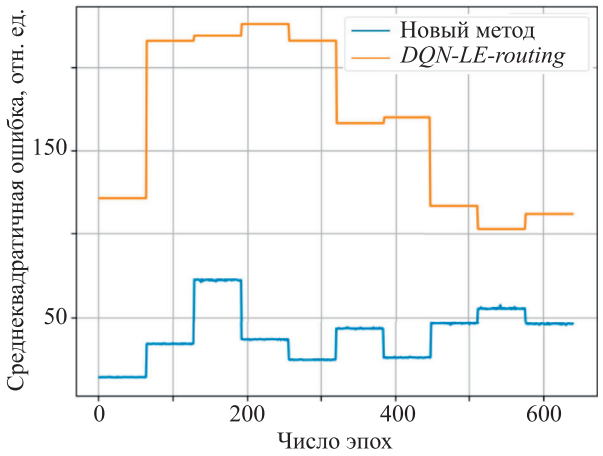

Рис. 5. Сравнение результатов предобучения нового метода и алгоритма *DQN-LE-routing* в нескольких экземплярах со значением δ = 3

Fig. 5. Comparison of the pre-training results of the new method and the *DQN-LE-routing* algorithm in several instances with the value δ = 3

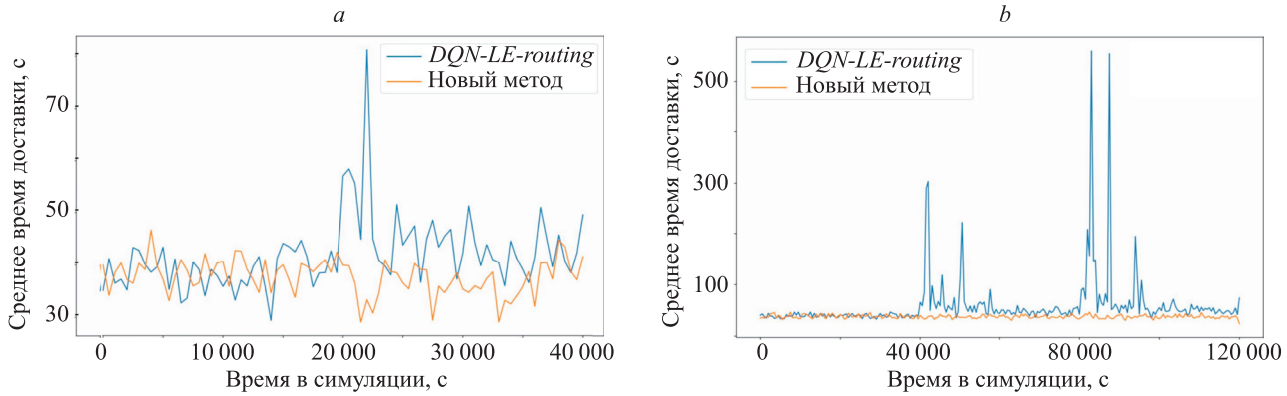

Рис. 6. Сравнение результатов работы нового метода и алгоритма *DQN-LE*-routing при однократном (*a*) и двукратном (*b*) изменении топологии

Fig. 6. Comparison of the results of the new method (New algo) and the *DQN-LE-routing* algorithm with a single (*a*) and two (*b*) topology change

Заключение

В работе предложен метод повышения адаптивности алгоритма мультиагентной маршрутизации, способный адаптироваться к любым изменениям графа, разработанный с учетом подходов, использующихся в существующих решениях в области обучения с подкреплением, и с применением нового подхода, как и в самой структуре и компонентах алгоритма, так и для обучения нейросети.

Выполнено сравнение метода с алгоритмом *DQN-LE-routing*, являющимся последним и наилучшим алгоритмом маршрутизации с использованием обучения с подкреплением по части архитектуры нейронной сети. По результатам экспериментов установлено, что при глобальных изменениях структуры графа разработанный метод значительно превосходит указанный алгоритм как на этапе предобучения, так и во время работы.

Предложенный метод может быть использован для решения задач маршрутизации со сложными функциями оценки пути и динамически меняющимися топологиями графов, например, в транспортной логистике и для управления конвейерными лентами на производстве.

Литература

1. Toth P., Vigo D. An overview of vehicle routing problems // The Vehicle Routing Problem. SIAM, 2002. P. 1–26. https://doi.org/10.1137/1.9780898718515.ch1
2. Vutukury S., Garcia-Luna-Aceves J.J. MDVA: A distance-vector multipath routing protocol // Proc. 20th Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM). V. 1. P. 557–564. https://doi.org/10.1109/INFCOM.2001.916780
3. Clausen T., Jacquet P. Optimized link state routing protocol (OLSR). 2003. N RFC3626. https://doi.org/10.17487/RFC3626
4. Sweda T.M., Dolinskaya I.S., Klabjan D. Adaptive routing and recharging policies for electric vehicles // Transportation Science. 2017. V. 51. N 4. P. 1326–1348. https://doi.org/10.1287/trsc.2016.0724
5. Puthal M.K., Singh V., Gaur M.S., Laxmi V. C-Routing: An adaptive hierarchical NoC routing methodology // Proc. of the 2011 IEEE/IFIP 19th International Conference on VLSI and System-on-Chip. 2011. P. 392–397. https://doi.org/10.1109/VLSISoC.2011.6081616
6. Zeng S., Xu X., Chen Y. Multi-agent reinforcement learning for adaptive routing: A hybrid method using eligibility traces // Proc. of the 16th IEEE International Conference on Control & Automation (ICCA'20). 2020. P. 1332–1339. https://doi.org/10.1109/ICCA51439.2020.9264518
7. Ibrahim A.M., Yau K.L.A., Chong Y.W., Wu C. Applications of multi-agent deep reinforcement learning: models and algorithms // Applied Sciences. 2021. V. 11. N 22. P. 10870. https://doi.org/10.3390/app112210870
8. Bono G., Dibangoye J.S., Simonin O., Matignon L., Pereyron F. Solving multi-agent routing problems using deep attention mechanisms // IEEE Transactions on Intelligent Transportation Systems. 2021. V. 22. N 12. P. 7804–7813. https://doi.org/10.1109/TITS.2020.3009289
9. Kang Y., Wang X., Lan Z. Q-adaptive: A multi-agent reinforcement learning based routing on dragonfly network // Proc. of the 30th International Symposium on High-Performance Parallel and

References

1. Toth P., Vigo D. An overview of vehicle routing problems. *The Vehicle Routing Problem*. SIAM, 2002, pp. 1–26. https://doi.org/10.1137/1.9780898718515.ch1
2. Vutukury S., Garcia-Luna-Aceves J.J. MDVA: A distance-vector multipath routing protocol. *Proc. 20th Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM)*, vol. 1, pp. 557–564. https://doi.org/10.1109/INFCOM.2001.916780
3. Clausen T., Jacquet P. *Optimized link state routing protocol (OLSR)*, 2003, no. RFC3626. https://doi.org/10.17487/RFC3626
4. Sweda T.M., Dolinskaya I.S., Klabjan D. Adaptive routing and recharging policies for electric vehicles. *Transportation Science*, 2017, vol. 51, no. 4, pp. 1326–1348. https://doi.org/10.1287/trsc.2016.0724
5. Puthal M.K., Singh V., Gaur M.S., Laxmi V. C-Routing: An adaptive hierarchical NoC routing methodology. *Proc. of the 2011 IEEE/IFIP 19th International Conference on VLSI and System-on-Chip*, 2011, pp. 392–397. https://doi.org/10.1109/VLSISoC.2011.6081616
6. Zeng S., Xu X., Chen Y. Multi-agent reinforcement learning for adaptive routing: A hybrid method using eligibility traces. *Proc. of the 16th IEEE International Conference on Control & Automation (ICCA'20)*, 2020, pp. 1332–1339. https://doi.org/10.1109/ICCA51439.2020.9264518
7. Ibrahim A.M., Yau K.L.A., Chong Y.W., Wu C. Applications of multi-agent deep reinforcement learning: models and algorithms. *Applied Sciences*, 2021, vol. 11, no. 22, pp. 10870. https://doi.org/10.3390/app112210870
8. Bono G., Dibangoye J.S., Simonin O., Matignon L., Pereyron F. Solving multi-agent routing problems using deep attention mechanisms. *IEEE Transactions on Intelligent Transportation Systems*, 2021, vol. 22, no. 12, pp. 7804–7813. https://doi.org/10.1109/TITS.2020.3009289
9. Kang Y., Wang X., Lan Z. Q-adaptive: A multi-agent reinforcement learning based routing on dragonfly network. *Proc. of the 30th International Symposium on High-Performance Parallel and*

Distributed Computing. 2021. P. 189–200. https://doi.org/10.1145/3431379.3460650
10. Choi S., Yeung D.Y. Predictive Q-routing: A memory-based reinforcement learning approach to adaptive traffic control // Advances in Neural Information Processing Systems. 1995. V. 8. P. 945–951.
11. Watkins C.J., Dayan P. Q-learning // Machine Learning. 1992. V. 8. N 3. P. 279–292. https://doi.org/10.1023/A:1022676722315
12. Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M. Playing atari with deep reinforcement learning // arXiv. 2013. arXiv:1312.5602. https://doi.org/10.48550/arXiv.1312.5602
13. Mukhutdinov D., Filchenkov A., Shalyto A., Vyatkin V. Multi-agent deep learning for simultaneous optimization for time and energy in distributed routing system // Future Generation Computer Systems. 2019. V. 94. P. 587–600. https://doi.org/10.1016/j.future.2018.12.037
14. Gao B., Pavel L. On the properties of the softmax function with application in game theory and reinforcement learning // arXiv. 2017. arXiv:1704.00805. https://doi.org/10.48550/arXiv.1704.00805
15. Мухудинов Д. Децентрализованный алгоритм управления конвейерной системой с использованием методов мультиагентного обучения с подкреплением: магистерская диссертация. СПб.: Университет ИТМО, 2019. 92 с. [Электронный ресурс]. URL: http://is.ifmo.ru/diploma-theses/2019/2_5458464771026191430.pdf (дата обращения: 01.10.2022)
16. Belkin M., Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering // Advances in Neural Information Processing Systems. 2001. P. 585–591. https://doi.org/10.7551/mitpress/1120.003.0080
17. Benea M.T., Florea A.M., Seghrouchni A.E.F. CAmI: An agent oriented-language for the collective development of AmI environments // Proc. of the 20th International Conference on Control Systems and Computer Science (CSCS). 2015. P. 749–756. https://doi.org/10.1109/CSCS.2015.136
18. Wang Y., Yao Q., Kwok J.T., Ni L.M. Generalizing from a few examples: A survey on few-shot learning // ACM Computing Surveys. 2020. V. 53. N 3. P. 63. https://doi.org/10.1145/3386252
19. Liu J., Chen S., Wang B., Zhang J., Li N., Xu T. Attention as relation: learning supervised multi-head self-attention for relation extraction // Proc. of the 19th International Joint Conferences on Artificial Intelligence (IJCAI). 2020. P. 3787–3793. https://doi.org/10.24963/ijcai.2020/524
20. Sola J., Sevilla J. Importance of input data normalization for the application of neural networks to complex industrial problems // IEEE Transactions on Nuclear Science. 1997. V. 44. N 3. P. 1464–1468. https://doi.org/10.1109/23.589532
21. Baldi P., Sadowski P.J. Understanding dropout // Advances in Neural Information Processing Systems. 2013. V. 26. P. 26–35.

Distributed Computing, 2021, pp. 189–200. https://doi.org/10.1145/3431379.3460650
10. Choi S., Yeung D.Y. Predictive Q-routing: A memory-based reinforcement learning approach to adaptive traffic control. *Advances in Neural Information Processing Systems*, 1995, vol. 8, pp. 945–951.
11. Watkins C.J., Dayan P. Q-learning. Machine Learning, 1992, vol. 8, no. 3, pp. 279–292. https://doi.org/10.1023/A:1022676722315
12. Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M. Playing atari with deep reinforcement learning. *arXiv*, 2013, arXiv:1312.5602. https://doi.org/10.48550/arXiv.1312.5602
13. Mukhutdinov D., Filchenkov A., Shalyto A., Vyatkin V. Multi-agent deep learning for simultaneous optimization for time and energy in distributed routing system. *Future Generation Computer Systems*, 2019, vol. 94, pp. 587–600. https://doi.org/10.1016/j.future.2018.12.037
14. Gao B., Pavel L. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv*, 2017, arXiv:1704.00805. https://doi.org/10.48550/arXiv.1704.00805
15. Mukhudinov D. *Decentralized conveyor system control algorithm using multi-agent reinforcement learning methods*. MSc Dissertation. St. Petersburg, ITMO University, 2019, 92 p. Available at: http://is.ifmo.ru/diploma-theses/2019/2_5458464771026191430.pdf (accessed: 01.10.2022). (in Russian)
16. Belkin M., Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems*, 2001, pp. 585–591. https://doi.org/10.7551/mitpress/1120.003.0080
17. Benea M.T., Florea A.M., Seghrouchni A.E.F. CAmI: An agent oriented-language for the collective development of AmI environments. *Proc. of the 20th International Conference on Control Systems and Computer Science (CSCS)*, 2015, pp. 749–756. https://doi.org/10.1109/CSCS.2015.136
18. Wang Y., Yao Q., Kwok J.T., Ni L.M. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 2020, vol. 53, no. 3, pp. 63. https://doi.org/10.1145/3386252
19. Liu J., Chen S., Wang B., Zhang J., Li N., Xu T. Attention as relation: learning supervised multi-head self-attention for relation extraction. *Proc. of the 19th International Joint Conferences on Artificial Intelligence (IJCAI)*, 2020, pp. 3787–3793. https://doi.org/10.24963/ijcai.2020/524
20. Sola J., Sevilla J. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 1997, vol. 44, no. 3, pp. 1464–1468. https://doi.org/10.1109/23.589532
21. Baldi P., Sadowski P.J. Understanding dropout. *Advances in Neural Information Processing Systems*, 2013, vol. 26, pp. 26–35.

Авторы

Грибанов Тимофей Александрович — студент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, https://orcid.org/0000-0002-1151-3405, t.hrybanau@gmail.com

Фильченков Андрей Александрович — кандидат физико-математических наук, инженер, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, SC 55507568200, https://orcid.org/0000-0002-1133-8432, afilchenkov@itmo.ru

Азаров Артур Александрович — кандидат технических наук, научный сотрудник, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация; заместитель директора, Северо-Западный институт управления — филиал РАНХиГС, Санкт-Петербург, 199178, Российская Федерация, SC 56938354700, https://orcid.org/0000-0003-3240-597X, artur-azarov@yandex.ru

Шалыто Анатолий Абрамович — доктор технических наук, профессор, главный научный сотрудник, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, SC 56131789500, https://orcid.org/0000-0002-2723-2077, shalyto@mail.ifmo.ru

Authors

Timofey A. Gribanov — Student, ITMO University, Saint Petersburg, 197101, Russian Federation, https://orcid.org/0000-0002-1151-3405, t.hrybanau@gmail.com

Andrey A. Filchenkov — PhD (Physics & Mathematics), Engineer, ITMO University, Saint Petersburg, 197101, Russian Federation, SC 55507568200, https://orcid.org/0000-0002-1133-8432, afilchenkov@itmo.ru

Artur A. Azarov — PhD, Scientific Researcher, ITMO University, Saint Petersburg, 197101, Russian Federation; Deputy Director, North-West Institute of Management — branch of the Russian Presidential Academy of National Economy and Public Administration, Saint Petersburg, 199178, Russian Federation, SC 56938354700, https://orcid.org/0000-0003-3240-597X, artur-azarov@yandex.ru

Anatoly A. Shalyto — D. Sc., Professor, Chief Reseacher, ITMO University, Saint Petersburg, 197101, Russian Federation, SC 56131789500, https://orcid.org/0000-0002-2723-2077, shalyto@mail.ifmo.ru

Статья поступила в редакцию 28.10.2022
Одобрена после рецензирования 03.11.2022
Принята к печати 29.11.2022

Received 28.10.2022
Approved after reviewing 03.11.2022
Accepted 29.11.2022