Применение матричных фильтров и теории кос для процедурной генерации архитектур нейронных сетей*

О. А. Лукьянова†, О. Ю. Никитин, А. С. Кунин

Вычислительный центр ДВО РАН, Хабаровск, Россия
†Контактный e-mail: ollukyan@gmail.com

Представлены результаты исследований, связанных с автоматическим формированием архитектур нейронных сетей, состоящих из набора модулей. Реализован алгоритмический подход, основанный на формировании матриц активных модулей. Предложены способы процедурной генерации архитектур нейронных сетей для решения задач классификации. Дается описание процесса автоматического формирования архитектуры PathNet, реализованной на основе новых подходов, а также рассматриваются примеры генерации трех новых архитектур глубоких нейронных сетей (3DNN, GraphNet и BraidNet). Архитектура BraidNet включает в себя построение графа связей сети на основе теории кос. Исследование на примере задачи классификации изображений MNIST показало применимость всех четырех предложенных нейронных сетей к распознаванию образов.

Ключевые слова: нейронная сеть, архитектура нейронных сетей, процедурная генерация, низкоразмерная топология, теория кос, передача информации, глубокое обучение.

Библиографическая ссылка: Лукьянова О.А., Никитин О.Ю., Кунин А.С. Применение матричных фильтров и теории кос для процедурной генерации архитектур нейронных сетей // Вычислительные технологии. 2019. Т. 24, № 6. С. 69–78. DOI: 10.25743/ICT.2019.24.6.009.

Введение

В настоящее время задание структуры нейронных сетей производится эмпирически, в процессе создания интеллектуальной системы анализа данных. В исследовательских работах имеется ряд подходов, предполагающих в той или иной степени автоматизацию процесса создания структуры нейронных сетей с помощью определенных алгоритмов. Так, могут автоматически выбираться модули, включенные в сеть, или количество слоев сети [1–4]. Все способы алгоритмического задания структур нейронных сетей можно обобщить как процедурную генерацию нейронных сетей. В вычислительной технике процедурная генерация — это метод создания данных алгоритмически, а не вручную [5]. Данный метод можно применять для формирования архитектур глубоких нейронных сетей, так как их оптимизация для конкретной решаемой задачи часто требует больших усилий и множества экспериментов.

*Title translation and abstract in English can be found on page 78.

Цель исследования — разработка способов процедурной генерации архитектур нейронных сетей, пригодных для задания различных видов нейронных сетей. Ниже рассматривается автоматическое формирование четырех архитектур нейронных сетей разной степени сложности, состоящих из модулей. Для этого применяется алгоритмический подход, разработанный авторами и основанный на формировании матриц активных модулей.

1. Матричные фильтры и модульные нейронные сети

Рассмотрим способы процедурной генерации архитектур нейронных сетей с помощью матричных фильтров. Фильтрами называются алгоритмы, выделяющие (или удаляющие) из исходного объекта некоторую часть с заданными свойствами. Матричными фильтрами, в свою очередь, являются фильтры, использующие определенную матрицу. Для автоматического создания архитектур нейронных сетей применяются бинарные матричные фильтры. Такие фильтры задают активные модули сети, тем самым меняя пути передачи информации между слоями. Они являются бинарными матрицами, в которых единица означает активацию модуля с соответствующим номером в определенном слое, а нуль — исключение модуля из сети (рис. 1).

Полученную бинарную матрицу назовем фильтром активных модулей F_a. Итоговая архитектура активных модулей нейронной сети представлена матрицей активных модулей M_{act}. Она получается с помощью произведения Адамара бинарного фильтра F_a на матрицу возможных активных модулей M_{full} нейронной сети:

$$M_{act} = M_{full} \circ F_a. \tag{1}$$

Активация и обратное распространение ошибки происходят только по активным модулям. Матричные фильтры использовались в данной статье для процедурной генерации как известной архитектуры нейронной сети PathNet (рис. 2, *б*), так и предложенных авторами. Ниже представлены три новых архитектуры нейронных сетей, различающиеся по сложности. Эти сети позволяют показать применимость метода матричных бинарных фильтров для построения как простых сетей, так и более сложных. Схематичные изображения архитектур исследуемых нейронных сетей представлены на рис. 2.

Архитектура, в которой структура модулей не меняется динамически и не имеет пересечений между потоками данных, приведена на рис. 2, *а*. Такая сеть предполагает глубокую нейронную сеть из нескольких параллельных потоков, в данном случае — трех, и названа 3DNN.

Более сложна нейронная сеть, основанная на графе активных модулей (рис. 2, *в*). В ней активные модули могут произвольно выбираться из более широкого набора. Граф сети задается матрицей активных модулей и может меняться динамически в процессе решения задачи. Такую сеть можно назвать GraphNet.

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Рис. 1. Примеры бинарных матричных фильтров

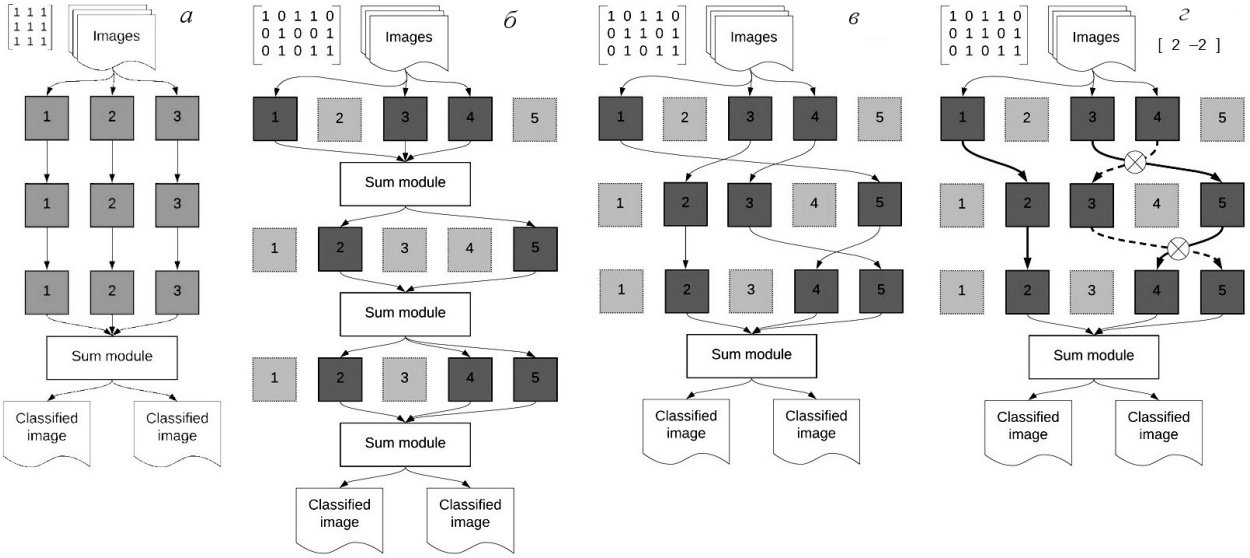

Рис. 2. Схематичное изображение архитектур нейронных сетей и матричных фильтров для их построения: (*а*) — 3DNN, (*б*) — PathNet, (*в*) — GraphNet, (*г*) — BraidNet

Третьей и самой сложной из предложенных архитектур является такая, где к архитектуре GraphNet добавлены дополнительные правила и ограничения динамики, задаваемые в соответствии с математической теорией кос (рис. 2, *г*). Это позволяет отслеживать динамику работы сети на фоне изменения изотопов косы (изотопами называются такие косы, которые могут быть непрерывно, без разрывов и склеек, продеформированы в трехмерном пространстве одна в другую [6]), а также очень кратко записывать структуру нейронной сети, избежав полного описания графа всех соединений. Такая сеть названа BraidNet.

2. Процедурная генерация нейронных сетей

Одной из простейших архитектур сетей с точки зрения процедурной генерации является глубокая нейронная сеть (DNN). На рис. 2, *а* изображена архитектура сети 3DNN — нейронной сети, состоящей из трех глубоких нейронных сетей, работающих параллельно и объединенных только на последнем суммирующем модуле (Sum module), где обобщается работа сети и рассчитывается ошибка для последующей коррекции весов.

Изображения подаются на первый слой в каждую из тех нейронных сетей и передаются от слоя к слою, не смешиваясь между тремя сетями. В сети 3DNN все модули в каждом слое активны и не отключаются. В этой и других сетях (рис. 2) все модули в слоях сетей являются полносвязными. Алгоритм формирования матричного фильтра для 3DNN представлен ниже (алгоритм 1).

Если набор потенциально активных модулей больше, чем итоговый набор активных модулей, то можно менять структуру связей модулей, в том числе динамически. Далее приведены примеры применения такого подхода для формирования различных архитектур сетей. Первой рассмотрим известную архитектуру PathNet, предложенную в [1]. Главной целью данного типа нейронных сетей является обеспечение так называемого трансферного обучения, т. е. многократного обучения одной и той же сети разным признакам.

Алгоритм 1. Формирование матричного фильтра для 3DNN

Input: количество слоев сети = L; количество модулей сети на каждом слое = M; M = 3; модуль m ∈ M
Output: матрица L × 3, каждый элемент которой равен единице
Procedure *3DNN-mask(L, M)*
  foreach *L* **do**
    foreach *M* **do**
      m = 1;
    end
  end
end

На рис. 2, *б* представлено схематичное изображение архитектуры сети PathNet. Темно-серыми квадратами обозначены активные модули, светло-серыми — выключенные. В архитектуре PathNet алгоритм повторно использует ограниченные наборы активных модулей нейронной сети для разных задач. Для каждой задачи оптимальные архитектуры активных модулей подбираются алгоритмически. Так как в ходе обучения PathNet его модули активируются и отключаются, возникает необходимость в матричных фильтрах, с помощью которых задается, какие модули в определенный момент времени будут включены или выключены (алгоритм 2).

Полученный подход к генерации архитектуры PathNet может быть доработан для создания нейронных сетей с другими архитектурами. Архитектуру сети можно задавать с помощью графа пересечений между модулями. Отказавшись от суммирующих модулей, с помощью матричных фильтров можно задавать схему активных модулей на каждом слое сети. При этом связи между модулями разных слоев могут выбираться

Алгоритм 2. Формирование матричного фильтра для инициализации популяции PathNet

Input: количество слоев сети = L; количество модулей сети на каждом слое = M; количество активных модулей на каждом слое = N; N ∈ M; модуль m ∈ M
Output: матрица L × M, со случайными элементами, равными единице
Procedure *PathNet-mask(L, M, N)*
  foreach *L* **do**
    n = случайное значение из {1,...,N};
    foreach *M* **do**
      m = 0;
      while *(сумма всех m) < n* **do**
        случайно выбранный m = 1;
      end
    end
  end
end

Алгоритм 3. Формирование матричного фильтра для GraphNet

Input: количество слоев сети = L; количество модулей сети на каждом слое = M; количество активных модулей на каждом слое = N; N ∈ M; модуль m ∈ M
Output: матрица L × M, со случайными элементами, равными единице
Procedure *GraphNet-mask(L, M, N)*
  foreach *L* **do**
    foreach *M* **do**
      m = 0;
      while *(сумма всех m)* < *N* **do**
        случайно выбранный m = 1;
      end
    end
  end
end

случайно, формируя граф (рис. 2, *в*). Последний суммирующий модуль используется для обобщения работы сети. Такой вид архитектуры можно назвать GraphNet, так как архитектура сети задается графом (см. алгоритм 3)

3. BraidNet: сеть на основе кос

На основе изучения процессов изменения информации и применения теории кос для записи топологии сети, которые подробнее описаны в работах [7–9], можно предложить метод процедурной генерации архитектуры глубокой нейронной сети BraidNet (рис. 2, *г*).

Данный вид архитектуры схож с архитектурой GraphNet, рассмотренной выше, однако связи между модулями от слоя к слою задаются не случайно, а с помощью правил пересечения между нитями в теории кос [7]. Для автоматического задания структуры такой сети предложены два матричных фильтра. Первый аналогичен матричному фильтру для сети GraphNet, а второй — матричный вариант записи “слова” косы (комбинации символов, которые задают порядок пересечений в косе) (алгоритм 4).

Для описания кос можно закодировать их пересечения с использованием слов кос. Это может быть набор положительных и отрицательных целых чисел, например: [1, 2, −1]. Если нить n проходит под нитью $n+1$, такое пересечение будет обозначаться отрицательным числом и наоборот.

Применение теории кос позволяет сравнивать информационные характеристики различных слоев сети попарно, находя между ними показатели взаимной энтропии, что может быть полезным для отслеживания динамики обучения сети и обеспечения эволюции ее структуры в процессе тренировки. Кроме того, запись в форме косы позволяет значительно сократить длину описания нейронной сети, что может быть применено при записи структуры нейронной сети в форме генетического кода для ее последующей генетической оптимизации.

Алгоритм 4. Формирование матричного фильтра для BraidNet

Input: количество слоев сети = L; количество модулей сети на каждом слое = M; количество активных модулей на каждом слое = N; N ∈ M; модуль m ∈ M

Output: матрица L × M, со случайными элементами, равными единице, и вектор-строка 1 × (L-1)

Procedure *BraidNet-mask(L, M, N)*

```
foreach L do
    foreach M do
        m = 0;
        while (сумма всех m) < N do
            случайно выбранный m = 1;
        end
    end
end
случайно инициализировать вектор-строку косы B, размерностью 1 × (L-1);
foreach b ∈ B do
    b = случайное значение из {-(N-1),...,(N-1)};
end
```

end

Таким образом, матричные фильтры могут быть применены для формирования структур различных нейронных сетей. Эффективность работы таких сетей в задачах классификации будет изучена ниже.

4. Оценка работы нейронных сетей

Для анализа работоспособности описанных выше архитектур, полученных с применением матричных фильтров, данные архитектуры были применены для распознавания и классификации изображений рукописных цифр из набора данных MNIST [10]. Оценивалась задача “один против одного” (one-vs-one). Размер пакета данных (batch size) составлял 16, одна эпоха обучения длилась 50 шагов. Нейронные сети реализованы на языке Python с применением фреймворка PyTorch и алгоритмов, изложенных выше. Обучение проводилось с критерием останова по достижении точности 98 %.

В процессе обучения для архитектур PathNet и GraphNet на каждой эпохе производилось сравнение точности работы архитектуры, сформированной по текущей матрице активных модулей, со случайно выбранной матрицей. Для следующего шага выбиралась та матрица активных модулей, которая формировала сеть с более высокой точностью работы.

Для архитектуры BraidNet исследовалась коса $[1, -1]$. Для нее на каждом шаге выбирались возможные изотопы, текущий изотоп сравнивался со случайно выбранным, и в следующем эпизоде использовался тот изотоп, который достигал наибольшей точности. В архитектуре 3DNN модули всегда представляли собой три параллельные глубокие нейронные сети, объединенные суммирующим слоем и функцией ошибки на выходе.

Результаты тренировки сетей, построенных с помощью четырех разных алгоритмов

Сеть	Эпохи (точность 98 %)	Эпохи (точность 97 %)	Точность после 1-й эпохи
3DNN	190	**4**	50 %
PathNet	194	24	**74 %**
GraphNet	**165**	6	54 %
BraidNet	375	9	**74 %**

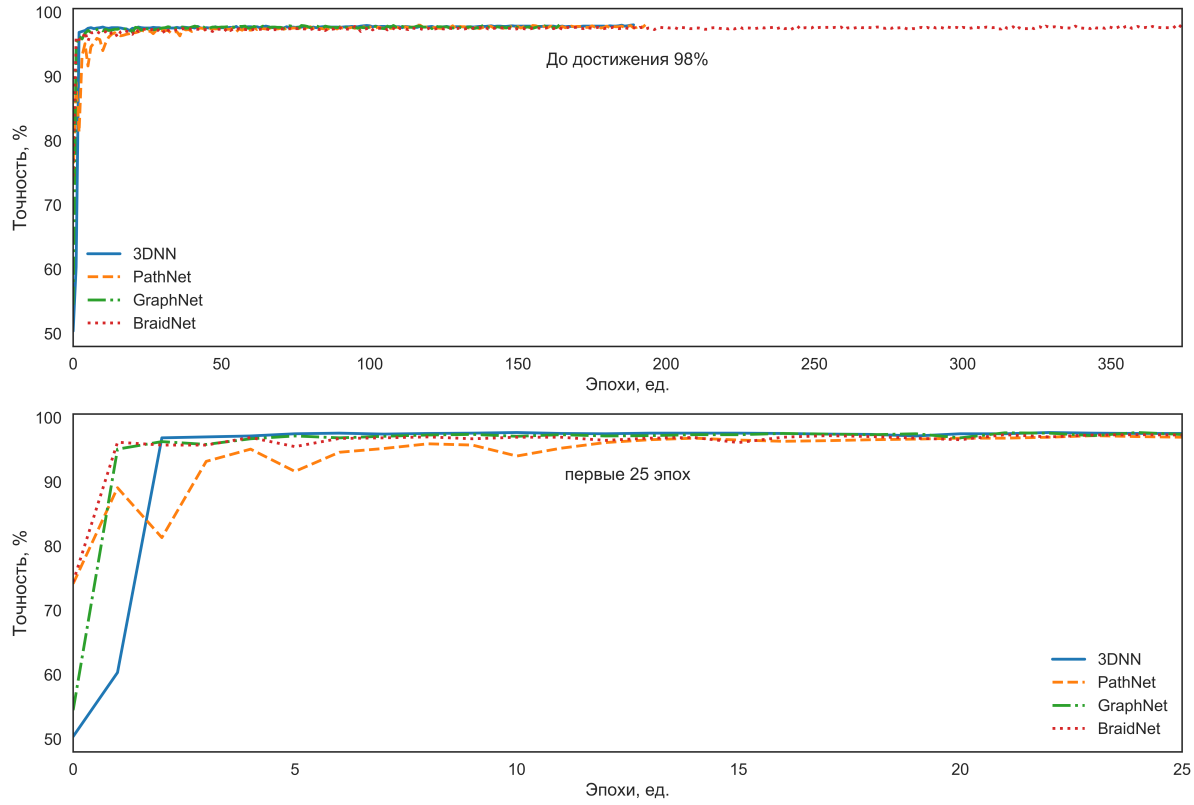

Рис. 3. Результаты тренировки сетей, построенных с помощью четырех разных алгоритмов

Процесс повышения точности в ходе обучения для четырех предложенных архитектур нейронных сетей отображен на рис. 3.

Исследование на примере задачи классификации показало, что все четыре архитектуры достигли требуемого результата. Тем не менее все представленные нейронные сети различаются по параметрам эффективности обучения, таким как скорость достижения требуемого результата и время достижения допустимой точности. Показатели обучения сетей представлены в таблице. Быстрее всех точности 98 % достигла сеть GraphNet (165 эпох). В свою очередь, сети PathNet и BraidNet уже после одной эпохи имели точность 74 %. Наиболее быстро повышала точность сеть 3DNN, которая уже к четвертому шагу достигла высокой точности — 97 %.

Все четыре сети имеют различные преимущества, которые могут быть исследованы на более крупных и многовариантных задачах, чем задача MNIST. Результаты работы архитектур, основанных на динамических матрицах (PathNet, GraphNet, BraidNet), говорят об их меньшей склонности к переобучению вследствие более поздней сходимости и наличию стохастической динамики в процессе обучения, видной на рис. 3.

Заключение

Показано, как процедурная генерация архитектур нейронных сетей позволяет избежать ручного задания структуры сети и формировать ее автоматически. Использование матричных фильтров упрощает процесс генерации архитектуры сети благодаря большому количеству возможных комбинаций модулей и связей между ними. На примере задачи классификации MNIST показано, как архитектуры, представленные в статье, решают реальные задачи распознавания образов.

Методы обучения с динамическим адаптивным изменением архитектуры сети позволяют быстрее достигать удовлетворительной точности, а также должны быть менее склонны к переобучению. Представленный в статье алгоритм BraidNet применим для попарного сравнения информационной динамики на каждом слое сети, а также для удобной краткой записи структуры нейронной сети в генетических алгоритмах. Такие особенности делают BraidNet перспективным алгоритмом для дальнейшего приложения и исследования в сложных задачах распознавания образов, в том числе с применением нейроэволюционных подходов.

Благодарности. Для выполнения расчетов использованы вычислительные ресурсы ЦКП “Центр данных ДВО РАН” [11]. Исследование выполнено при финансовой поддержке РФФИ (грант № 18-31-00188). Расчеты проводились с применением методов и технологии, разработанных при финансовой поддержке РФФИ в рамках проекта № 18-29-03196.

Авторы выражают благодарность А.А. Сорокину за ценные комментарии в процессе подготовки рукописи данной статьи.

Список литературы / References

[1] **Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A., Wierstra, D.** Pathnet: Evolution channels gradient descent in super neural networks. Available at: `https://arxiv.org/abs/1701.08734` (accessed 30.09.2019).

[2] **Larsson, G., Maire, M., Shakhnarovich, G.** FractalNet: Ultra-deep neural networks without residuals. Available at: `https://arxiv.org/abs/1605.07648` (accessed 30.09.2019).

[3] **Saxena, S., Verbeek, J.** Convolutional neural fabrics // Advances in Neural Inform. Proc. Systems (NIPS 2016), 2016. P. 4053–4061.

[4] **Zoph, B., Le, Q.V.** Neural architecture search with reinforcement learning. Available at: `https://arxiv.org/abs/1611.01578` (accessed 30.09.2019).

[5] **Togelius, J., Kastbjerg, E., Schedl, D., Yannakakis, G.N.** What is procedural content generation? Mario on the borderline // Proc. of the 2nd Intern. Work. on Procedural Content Generation in Games, New York, USA. NY: ACM, 2011. P. 3:1–3:6.

[6] **Murasugi, K., Kurpita, B.I.** Isotopy of braids // A Study of Braids. Book Series: Mathematics and Its Applications. 1999. Vol. 484. P. 96–112.

[7] **Lukyanova, O., Nikitin, O.** Modeling of extrasynaptic information transfer in neural networks using braid theory // Computer Sci. 2018. Vol. 145C. P. 306–311. DOI:10.1016/j.procs.2018.11.076.

[8] **Lukyanova, O., Nikitin, O.** Isotopic inheritance: A topological approach to genotype transfer. From Animals to Animats 15. SAB 2018 / P. Manoonpong, J. Larsen, X. Xiong, J. Hallam, J. Triesch (Eds) // Lecture Notes in Computer Science. 2018. Vol. 10994. P. 27–38.

[9] **Lukyanova, O., Nikitin, O.** Neuronal topology as set of braids: information processing, transformation and dynamics // Optical Memory and Neural Networks. 2017. Vol. 26, No. 3. P. 172–181.

[10] **LeCun, Y., Cortes, C., Burges, C.J.C.** The MNIST database of handwritten digits. Available at: `http://yann.lecun.com/exdb/mnist/` (accessed 30.09.2019).

[11] **Сорокин А.А., Макогонов С.В., Королев С.П.** Информационная инфраструктура для коллективной работы ученых Дальнего Востока России // Науч.-техн. информация. Сер. 1: Организация и методика информационной работы. 2017. № 12. С. 14–16.
Sorokin, A.A., Makogonov, S.V., Korolev, S.P. Information infrastructure for the collective work of scientists of the Russian Far East // Sci. and Technical Inform. Ser. 1: Organization and Methodology of Information Work. 2017. No. 12. P. 14–16. (In Russ.)

Поступила в редакцию 15 октября 2019 г.

Application of matrix filters and braid theory for the procedural generation of neural network architectures

Lukyanova, Olga A.*, Nikitin, Oleg Yu., Kunin, Alexey S.

Computing Center FEB RAS, Khabarovsk, 680000, Russia

*Corresponding author: Lukyanova, Olga A., e-mail: `ollukyan@gmail.com`

There are various approaches to the algorithmic specification of the network structure in the deep learning problems, which are successfully used in applications. These methods can be generalized by the concept of procedural generation of neural network architectures.

Methodology. In the work, we use binary matrix filters. The filters are obtained with the help of the Hadamard product. Such filters define active network modules, thereby changing the way information is transmitted between layers. To build various architectures, the theory of braids is used in the work. The article reproduces the well-known PathNet architecture. Examples of generating three new deep neural network architectures (3DNN, GraphNet, and BraidNet) are examined.

Findings. The paper shows how the procedural generation of neural network architectures allows avoiding manually setting the network structure and automatically forming it. The use of matrix filters simplifies the process of generating network architecture due to a large number of possible combinations of modules and connections between them. Using the MNIST classification problem as an example, it is shown how the architectures presented in the article solve real-world pattern recognition problems. The results of application of neural networks indicate their diminishing tendency to retraining due to the subsequent convergence and the presence of stochastic dynamics in the learning process.

Originality/value. Learning methods with dynamic adaptive changes in the network architecture allows achieving satisfactory accuracy faster and should also be less prone to retraining. The BraidNet algorithm presented in the article is applicable for

a convenient brief record of the structure of a neural network in genetic algorithms. Such features make BraidNet a promising algorithm for further application and research in complex problems of pattern recognition, including using neuroevolutionary approaches.

Keywords: neural networks, neural network architectures, procedural generation, low-dimensional topology, braid theory, information transfer, deep learning.

Cite: Lukyanova, O.A., Nikitin, O.Y., Kunin, A.S. Application of matrix filters and braid theory for the procedural generation of neural network architectures // Computational Technologies. 2019. Vol. 24, No. 6. P. 69–78. (In Russ.)
DOI: 10.25743/ICT.2019.24.6.008.

Acknowledgements. The computing resources of the Shared Facility Center "Data Center of FEB RAS" (Khabarovsk) were used to carry out calculations. The research was funded by RFBR project No. 18-31-00188. The calculations were carried out using methods and technologies which development was funded by RFBR according to the research project No. 18-29-03196.

The authors would like to express their gratitude to Dr. A.A. Sorokin for fruitful conversation and insightful comments during the manuscript preparation.

Received October 15, 2019