УДК 519.688 Вестник СПбГУ. Сер. 10. 2016. Вып. 4

И. С. Дрокин

ОБ ОДНОМ АЛГОРИТМЕ ПОСЛЕДОВАТЕЛЬНОЙ ИНИЦИАЛИЗАЦИИ ВЕСОВ ГЛУБОКИХ НЕЙРОННЫХ СЕТЕЙ И ОБУЧЕНИИ АНСАМБЛЯ НЕЙРОННЫХ СЕТЕЙ

Санкт-Петербургский государственный университет, Российская Федерация,
199034, Санкт-Петербург, Университетская наб., 7–9

Использование механизма предобучения многослойных персептронов позволило значительно улучшить качество и скорость обучения глубоких сетей. В статье предлагается еще один способ начальной инициализации весов с применением принципов обучения с учителем, подхода self-taught learning и transfer learning, были проведены тесты, показавшие работоспособность подхода, а также указаны дальнейшие шаги и направления для его развития. Получены и описаны итеративный алгоритм инициализации весов, основанный на уточнении весов скрытых слоев нейронной сети через решение исходной задачи классификации или регрессии, следующий из него способ построения ансамбля нейронных сетей. Проведены тесты, показывающие эффективность подхода. Библиогр. 14 назв. Ил. 5. Табл. 2.

Ключевые слова: глубокое обучение, инициализация весов нейронных сетей, ансамбли нейронных сетей.

I. S. Drokin

ABOUT AN ALGORITHM FOR CONSISTENT WEIGHTS INITIALIZATION OF DEEP NEURAL NETWORKS AND NEURAL NETWORKS ENSEMBLE LEARNING

St. Petersburg State University, 7–9, Universitetskaya nab.,
St. Petersburg, 199034, Russian Federation

Using of the pretraining of multilayer perceptrons mechanism has greatly improved the quality and speed of training deep networks. In this paper we propose another way of the weights initialization using the principles of supervised learning, self-taught learning approach and transfer learning, tests showing performance approach have been carried out and further steps and directions for the development of the method presented have been suggested. In this paper we propose an iterative algorithm of weights initialization based on the rectification of the hidden layers of weights of the neural network through the resolution of the original problem of classification or regression, as well as the method for constructing a neural network ensemble that naturally results from the proposed learning algorithm, tests showing performance approach have been carried out. Refs 14. Figs 5. Tables 2.

Keywords: deep learning, neural networks weights initialization, ensemble of neural networks.

Использование механизма предобучения многослойных персептронов позволило значительно улучшить качество и скорость обучения глубоких сетей. В данной статье предлагается еще один способ начальной инициализации весов с использованием принципов обучения с учителем и подхода self-taught learning [1]. В настоящее время широко применяются методы инициализации весов многослойного персептрона на базе ограниченных машин Больцмана [2], авто-энкодеров (stacked auto-encoders, SAE) [3–5], метода главных компонент [6]. Они построены на методах, которые могут использовать неразмеченные данные, что во многих задачах позволяет существенно увеличить объем выборки, доступный для исследований и тестов, и тем самым повы-

Дрокин Иван Сергеевич — аспирант; ivan.s.drokin@bk.ru

Drokin Ivan Sergeevich — postgraduent student; ivan.s.drokin@bk.ru

 DOI: 10.21638/11701/spbu10.2016.406

сить качество обучения. Разработаны итеративный алгоритм инициализации весов, основанный на уточнении весов скрытых слоев нейронной сети через решение исходной задачи классификации или регрессии, а также естественно следующий из предложенного алгоритма обучения способ построения ансамбля нейронных сетей.

Опишем алгоритм инициализации весов нейронной сети:

Algorithm 1: Предобучение сети с использованием SAE

for $j = 1 \to k$ **do**

1. Инициализация сети $\tilde{N}^j$ с одним скрытым слоем размера $u(L_j)$, входным слоем размера $u(L_{j-1})$ и выходным слоем размера $u(L_{j-1})$. Пусть $\tilde{W}$ — веса скрытого слоя.
2. Обучение сети $\tilde{N}^j$ на множестве примеров $\{x_{i,j-1}, x_{i,j-1}\},\ i = 1, \ldots, K$.
3. $\hat{W}_j = \tilde{W}$.

end

Result: $\{\hat{W}_j\}_{j=1}^k$.

Рассмотрим множество пар $\{x_i, y_i\},\ i = 1,\ \ldots,\ K$, где $\{x_i \in R^n, y_i \in R^m\}$, x_i — входной пример, y_i — желаемый отклик. Необходимо построить функцию $f(x)$ такую, что $f(x_i) = y_i,\ i = 1,\ \ldots,\ K$. Для описания предлагаемого алгоритма возьмем многослойный персептрон N. Пусть $L = \{L_j\}_{j=1}^k = \{L_1 \ldots L_k\}$ есть множество его скрытых слоев, L_0 — входной слой, $\{x_{i,j}\}_{i=1}^K$ — выходы j-го слоя. Пусть $u(L_j)$ равно количеству нейронов в слое j, $W(L_j) = W_j$ — веса нейронов j-го скрытого слоя, $N(j)$ — j-й скрытый слой сети N. Алгоритм предобучения нейронной сети на основе авто-энкодеров кратко можно описать так, как показано в алгоритме 1.

Обучение ансамбля нейронных сетей. Использование алгоритма 2 позволяет применять множество $\{\hat{W}_j\}_{j=1}^k$ для достаточно хорошей инициализации весов скрытых слоев сети для обучения всей сети методом обратного распространения ошибки. Также этот подход учитывает входные данные, для которых неизвестен желаемый отклик. Предлагается дополнить этот подход следующим образом. Веса j-го скрытого слоя после предобучения авто-энкодером можно уточнить, обучая сеть с одним скрытым слоем на исходной задаче или близкой ей, на основе концепции transfer learning [1, 7, 9], т. е. применять полученные на втором пункте j-го шага алгоритма веса $\tilde{W}$ для инициализации сети с одним скрытым слоем, входным слоем размера $u(L_{j-1})$ и выходным размера m и обучить ее на множестве примеров $\{x_{i,j-1}, y_i\},\ i = 1, \ldots, K$. На j-й итерации алгоритма также можно уточнить веса скрытых слоев с первого по j-й через обучение сети с j-ми скрытыми слоями исходной или близкой исходной: определенные на предыдущих шагах веса $\{\hat{W}_t\}_{t=1}^j$ берутся для инициализации сети с j-ми скрытыми слоями, входным слоем размера n и выходным размера m, сеть обучается на множестве примеров $\{x_i, y_i\},\ i = 1, \ldots, K$.

Данную настройку параметров имеет смысл делать для всех скрытых слоев, кроме первого и последнего, так как для первого слоя эта настройка в точности повторяет описанную ранее процедуру, а для последнего является задачей обучения всей сети. Таким образом, к исходному алгоритму добавляются два дополнительных процесса обучения. Общий алгоритм можно описать так, как показано в алгоритме 2.

В результате предобучения сети (см. алгоритм 2) будут получены веса для инициализации сети N исходной конфигурации. В качестве небольшого бонуса при предобучении будут также получены веса для $k-1$ сети с меньшим числом слоев. Данные

Algorithm 2: Последовательное предобучение сети

for $j = 1 \to k$ **do**

1. Инициализация сети $\tilde{N}^j$ с одним скрытым слоем размера $u(L_j)$, входным слоем размера $u(L_{j-1})$ и выходным слоем размера $u(L_{j-1})$. Пусть $\tilde{W}$ — веса скрытого слоя.
2. Обучение сети $\tilde{N}^j$ на множестве примеров $\{x_{i,j-1}, x_{i,j-1}\},\ i = 1, \ldots, K$.
3. Инициализация сети $\grave{N}^j$ с одним скрытым слоем размера $u(L_j)$, входным слоем размера $u(L_{j-1})$ и выходным слоем размера m, веса скрытого слоя инициализируются $\tilde{W}$.
4. Обучение сети $\grave{N}^j$ на множестве примеров $\{x_{i,j-1}, y_i\},\ i = 1, \ldots, K$.
5. Инициализация сети $\hat{N}^j$ с j-ми скрытыми слоями размера $u(L_t), t = \overline{1,j}$, входным слоем размера n и выходным слоем размера m, веса скрытых слоев инициализируются $\{\hat{W}_t\}_{t=1}^{j-1}$, $\hat{W}_j = W(\grave{N}(1))$.
6. Обучение сети $\hat{N}^j$ на множестве примеров $\{x_i, y_i\},\ i = 1, \ldots, K$.
7. $\hat{W}_t = W(\hat{N}^j(t)), t = \overline{1,j}$.

end

Result: $\{\hat{W}_j\}_{j=1}^{k}$.

сети можно применять для дальнейшего обучения и объединения построенных сетей в комитет. Приведенная схема предобучения позволяет также использовать на этапах 4 и 6 алгоритма не исходный набор данных для обучения, а близкий ему. Например, если исходная задача состоит в обучении на наборе данных CIFAR-100, то во время предобучения можно взять набор CIFAR-10 [10]. Очевидно, что данный алгоритм обобщается и для сверточных и рекуррентных нейронных сетей, для чего могут быть использованы рекуррентные [11] и сверточные [12] авто-энкодеры соответственно.

Согласно предложенному алгоритму, в п. 6 каждой итерации обучается нейронная сеть $\hat{N}^j$, решающая поставленную задачу. Таким образом, после окончания работы алгоритма имеются $k-1$ частично обученных моделей, решающих данную задачу, а также веса для инициализация всех скрытых слоев для обучения исходной нейронной сети. Согласно такому алгоритму, первая из $k-1$ частично обученных моделей сеть $\hat{N}^1$ содержит лишь один скрытый слой размера $u(L_1)$, вторая сеть $\hat{N}^2$ состоит из двух скрытых слоев размером $u(L_1)$ и $u(L_2)$ соответственно, сеть $\hat{N}^{k-1}$ — $k-1$ скрытых слоев размерами $u(L_1), u(L_2), \ldots, u(L_{k-1})$. Таким образом, нейронная сеть $\hat{N}^{k-1}$ аналогична по архитектуре исходной нейронной сети за вычетом последнего скрытого слоя. Опираясь на полученные модели, можно составить ансамбль нейронных сетей различной архитектуры с увеличивающейся сложностью от итерации к итерации.

В данной статье будет рассматриваться построение комитета нейронных сетей на основе простого усреднения выходов моделей. Поэтому целесообразно добавлять к полностью обученной исходной сети не все промежуточные $\hat{N}^j$, а только последние $p-1$ моделей. Состав ансамбля в таком случае следующий: $N, \hat{N}^{k-1}, \hat{N}^{k-1}, \ldots, \hat{N}^{k-p}$, всего p нейронных сетей различной архитектуры, каждая из которых получает начальные значения весов скрытых слоев в результате работы алгоритма 2.

Описание экспериментов. Для проведения эксперимента использовался набор данных CIFAR-10 [10]. Он представляет из себя набор из 50 000 обучающих примеров и 10 000 тестовых примеров цветных изображений размером 32 на 32 пикселя, разбитых на десять классов. Тесты проводились для двух классов нейронных сетей: многослойного перцептрона и сверточной нейронной сети.

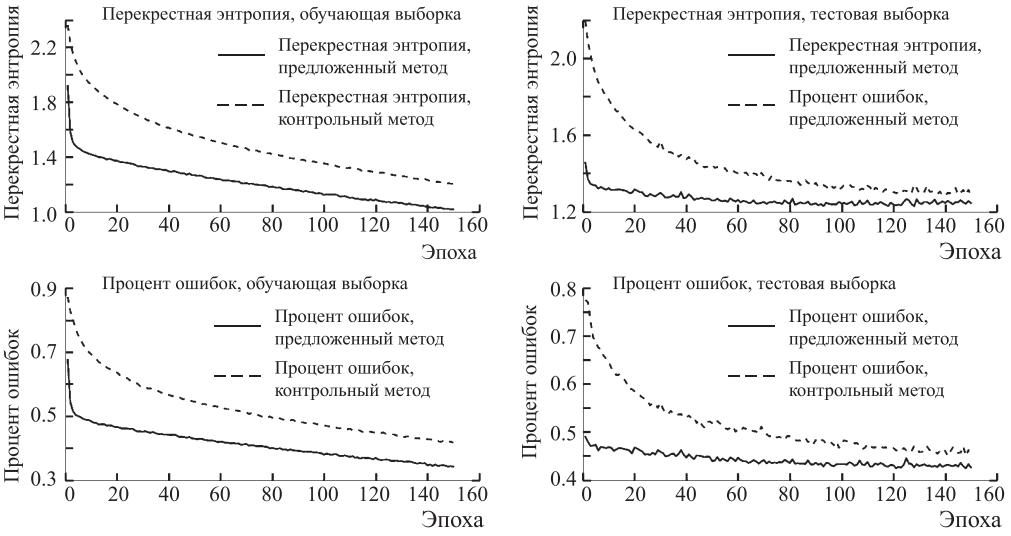

Рис. 1. Ошибки на тестовой и обучающей выборках для CIFAR-10 и многослойного персептрона

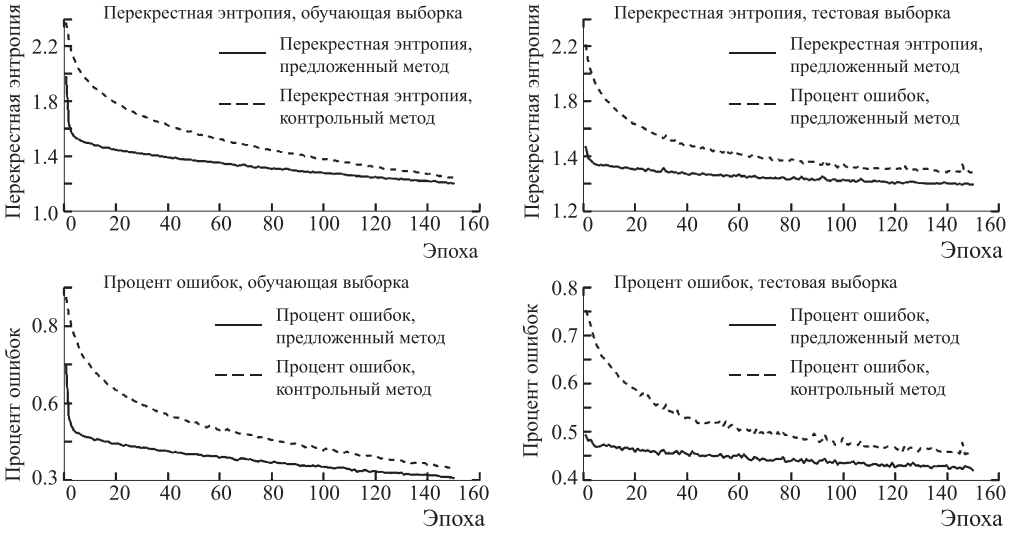

Рис. 2. Ошибки на тестовой и обучающей выборках для CIFAR-10 и многослойного персептрона с расширением датасета

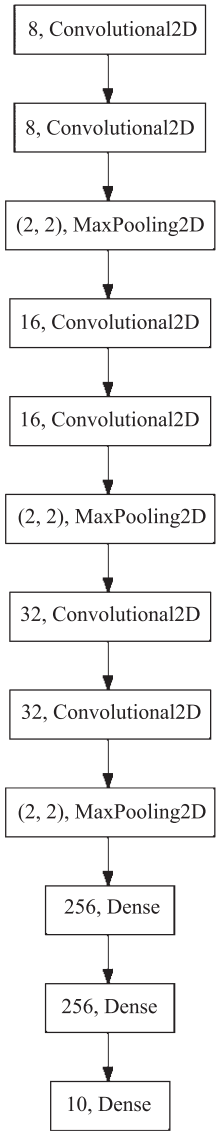

Рис. 3. Архитектура сверточной нейронной сети

Рассмотрим эксперименты, проведенные для многослойного перцептрона. Размер входного слоя сети равен 3072, размер выходного слоя равен 10 для набора данных CIFAR-10. В дальнейшем конфигурация многослойного перцептрона будет кратко записываться как $n - n_1 - n_2 - \cdots - n_k - m$, где n — размер входного слоя, $n_1, n_2, \ldots, n_k$ — количество нейронов в скрытых слоях, m — размер выходного слоя. Для всех скрытых слоев применялась функция активации $f(x) = \max(0, x)$ (ReLU), для выходного слоя — функция мягкого максимума (softmax). Функцией ошибки служила категориальная перекрестная энтропия (categorical crossentropy). Для каждого скрытого слоя использовалось исключение нейронов (dropout) с коэффициентом 0.5, размер одного пакета (batch) равен 16, в качестве оптимизатора применялся стохастический градиентный спуск с нестеровским моментом, параметр скорости обучения равен 0.0001, параметр момента — 0.9. Тесты проводились как с использованием техник увеличения объема выборки, так и без них. Увеличение выборки происходило перед началом каждой эпохи обучения, для этого изображение

1) поворачивалось на небольшой случайный угол;

2) зеркально отображалось относительно вертикальной оси;

3) сдвигалось на небольшую случайную величину по обеим осям.

В качестве наблюдаемых параметров выступали ошибки в процессе обучения на тестовой и обучающей выборках. Сравнивать предложенный алгоритм предлагается с классическим алгоритмом обучения нейронной сети с предобучением через последовательные авто-энкодеры. Все параметры обучения многослойных перцептронов и авто-энкодеров в обоих алгоритмах предобучения одинаковы. Для эксперимента на выборке CIFAR-10 будет рассматриваться конфигурация сети $3072 - 1536 - 1024 - 512 - 256 - 10$. Поведение ошибок исследовалось на первых 150 эпохах обучения. Результаты теста для сетей приведены на рис. 1 и 2.

Проведем аналогичный тест для сверточной нейронной сети. Архитектура сети представлена на рис. 3. В качестве функции активации использовалась функция $f(x) = \max(0, x)$, для каждого слоя агрегирования применялось исключение нейронов с коэффициентом 0.2, для полносвязных скрытых слоев — с коэффициентом 0.5, размер одного пакета равен 64, оптимизатором служил стохастический градиентный спуск с нестеровским моментом, параметр скорости обучения равен 0.0001, параметр момента — 0.9, предобучение выполнялось в течение 20 эпох, поведение ошибок изучалось на первых 350 эпохах обучения. Результаты теста приведены на рис. 4 и 5.

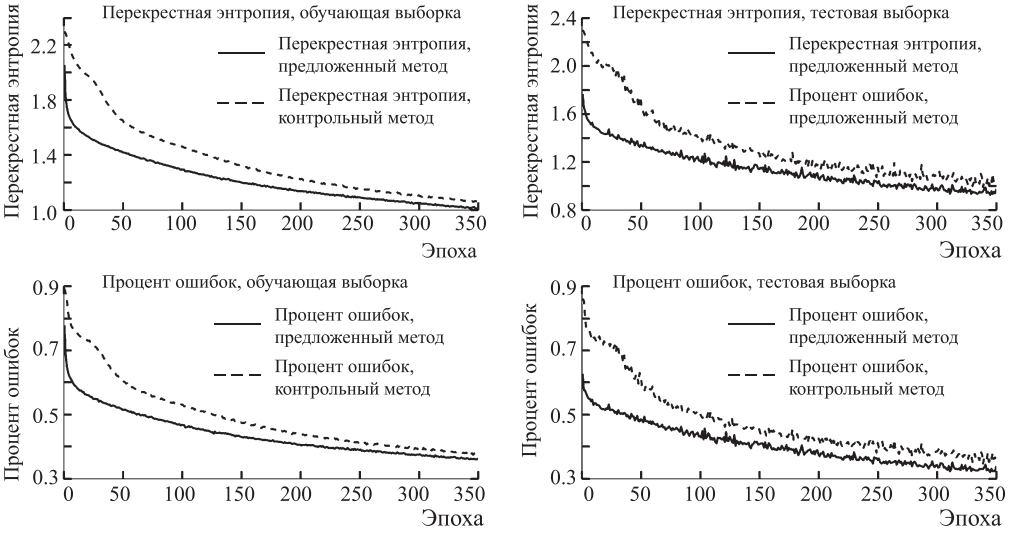

Рис. 4. Ошибки на тестовой и обучающей выборках для CIFAR-10 и сверточной сети

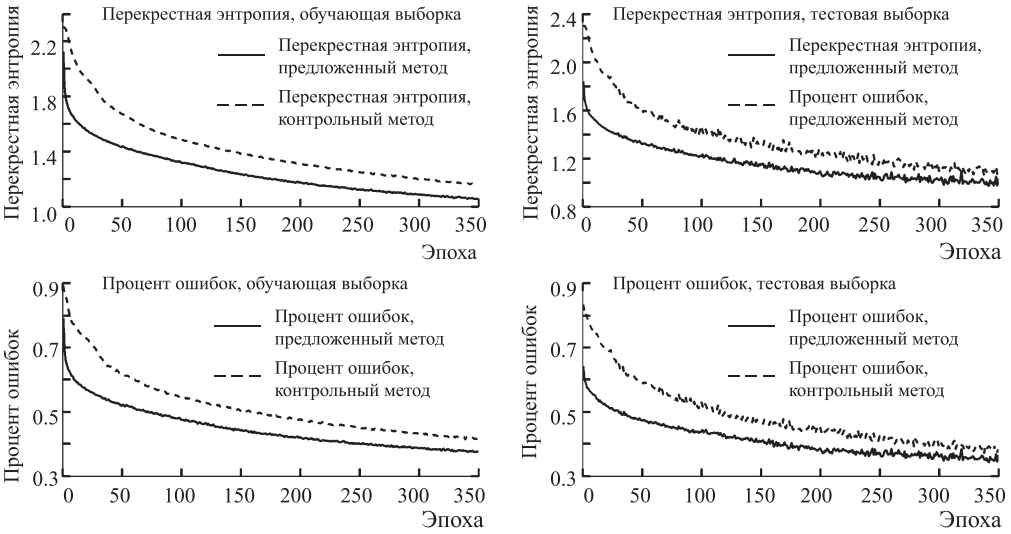

Рис. 5. Ошибки на тестовой и обучающей выборках для CIFAR-10 и сверточной сети с расширением датасета

Теперь перейдем к обсуждению результатов для ансамблей нейронных сетей, полученных в результате обучения. Для формирования ансамбля, помимо исходной сети, использовались сети, полученные в двух последних этапах работы алгоритма, т. е. $p = 3$. В табл. 1 и 2 введены следующие обозначения: BM — описание базовой модели, CA — в обучении использовался классический алгоритм, PA — предложенный алгоритм, EPA — ансамбль моделей, построенных в ходе обучения, MLP — многослойный перцептрон, CNN — сверточная сеть, lr — параметр скорости обучения, DA — в ходе эксперимента применялись техники расширения выборки.

Таблица 1. **Сравнительная таблица ошибок алгоритмов**

BM	CA, train	PA, train	EPA, train	CA, test	PA, test	EPA, test
MLP, $lr = 10^{-4}$	1.2022	1.0167	0.6489	1.2861	1.2435	1.1828
MLP, $lr = 10^{-4}$, DA	1.2394	1.1976	0.9056	1.2800	1.1909	1.1531
CNN, $lr = 10^{-4}$	1.0538	1.0099	0.8668	1.0325	0.9625	0.9241
CNN, $lr = 10^{-4}$, DA	1.1631	1.0535	0.9060	1.0931	1.0220	0.9443
CNN, $lr = 10^{-3}$, DA	0.7211	0.6587	0.5059	0.8237	0.6868	0.6670

Таблица 2. **Сравнительная таблица точности алгоритмов**

BM	CA, train	PA, train	EPA, train	CA, test	PA, test	EPA, test
MLP, $lr = 10^{-4}$	0.5832	0.6586	0.7949	0.5434	0.5741	0.5839
MLP, $lr = 10^{-4}$, DA	0.5698	0.5959	0.6893	0.5465	0.5804	0.5877
CNN, $lr = 10^{-4}$	0.6252	0.6391	0.6979	0.6391	0.6646	0.6786
CNN, $lr = 10^{-4}$, DA	0.5820	0.6262	0.6873	0.6178	0.6386	0.6694
CNN, $lr = 10^{-3}$, DA	0.7416	0.7669	0.8450	0.7203	0.7629	0.7762

Результаты и выводы. Как следует из рис. 1–5, применение предложенного алгоритма позволяет уменьшить ошибку на тестовой выборке, что хорошо видно из сравнения процессов обучения для рассматриваемых алгоритмов инициализации весов. Представленный метод является эффективным вариантом инициализации весов как многослойного персептрона, так и сверточной сети. В ходе алгоритма также могут быть определены начальные веса для сетей меньшего размера, и, завершив их обучение, можно получить ансамбль нейронных сетей. Из приведенных табл. 1, 2 видно, что ансамбль достаточно эффективен для решения поставленных задач в сравнении с одной сетью. Таким образом, обучение комитета может рассматриваться как дополнительный вариант регуляризации. В качестве дальнейших направлений для исследований можно предложить, что

1) на этапах 3–6 следует добавлять дополнительные скрытые слои;

2) использовать этапы 3–6 только для первых нескольких слоев;

3) объединить в общий классификатор сети, полученные на этапах 4 и 6 на основе градиентного бустинга или аналогичного алгоритма;

4) провести оптимизацию параметров обучения сетей для дополнительной оценки качества обучения.

В основе предложения использовать этапы 3–6 только для первых нескольких слоев лежит наблюдение, называемое затуханием градиента [13, 14]. Согласно ему, первые слои многослойного персептрона обучаются хуже, и применение этапов 3–6 только на первых нескольких слоях позволит, с одной стороны, получить достаточно качественно приближение весов, с другой — сэкономить вычислительные ресурсы. Рассмотренный подход может быть также использован с ограниченными машинами Больцмана для предобучения многослойных персептронов и со сверточными автоэнкодерами и сверточными нейронными сетями.

Заключение. Был разработан алгоритм иерархического предобучения многослойного перцептрона на основе подходов self-taught learning и transfer learning, проведены тесты, показавшие его работоспособность, а также указаны дальнейшие шаги и направления для развития представленного метода. Был предложен алгоритм построения ансамбля нейронных сетей на базе рассмотренного алгоритма инициализации весов и проведены тесты, показавщие его эффективность.

Литература

1. *Raina R., Battle A., Lee H., Packer B., Ng A. Y.* Selftaught learning: Transfer learning from unlabeled data // Proc. of the 24th Intern. conference on machine learning. 2007. June 20–24. P. 759–766.

2. *Hinton G. E., Salakhutdinov R. R.* Reducing the dimensionality of data with Neural Networks // Science. 2006. 28 July. Vol. 313, N 5786. P. 504–507.

3. *Vincent P., Larochelle H., Lajoie I., Bengio Y., Manzagol P.-A.* Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion // The Journal of Machine Learning Research archive. 2010. Vol. 11. P. 3371–3408.

4. *Masci J., Meier U., Ciresan D., Schmidhuber J.* Stacked convolutional auto-encoders for hierarchical feature extraction // 21st Intern. conference on Artificial Neural Networks. Espoo, Finland, 2011. June 14–17. Pt I. P. 52–59.

5. *Gehring J., Miao Y., Metze F., Waibel A.* Extracting deep bottleneck features using stacked auto-encoders // Acoustics, speech and signal processing (ICASSP). IEEE Intern. conference. 2013. P. 3377–3381.

6. *Baldi P., Hornik K.* Neural Networks and principal component analysis: learning from examples without local minima // Neural Networks. 1989. Vol. 2. P. 53–58.

7. *Caruana R.* Multitask learning // Machine learning. 1997. Vol. 28. P. 41–75.

8. *UFDL.* URL: http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial (дата обращения: 19.04.2016).

9. *Ciresan D. C, Meier U., Schmidhuber J.* Transfer learning for Latin and Chinese characters with deep Neural Networks // The 2012 Intern. joint conference on Neural Networks (IJCNN). 2012. P. 1–6.

10. *CIFAR.* URL: http://www.cs.toronto.edu/~kriz/cifar.html (дата обращения: 19.04.2016).

11. *Rolfe J. T., LeCun Y.* Discriminative recurrent sparse auto-encoders // The Intern. conference on learning representations: Proceedings. 2013.

12. *Masci J., Meier U., Cires D. Bëan, Schmidhuber J.* Stacked convolutional auto-encoders for hierarchical feature extraction // Intern. conference artificial Neural Networks and machine learning. 2011. P. 52–59.

13. *Glorot X., Bengio Y.* Understanding the difficulty of training deep feedforward neural networks // Intern. conference on artificial intelligence and statistics. 2010. P. 249–256.

14. *Pascanu R., Mikolov T., Bengio Y.* Understanding the exploding gradient problem: Tech. Rep. Montreal: Universite de Montreal, 2012. 11 p.

Для цитирования: Дрокин И. С. Об одном алгоритме последовательной инициализации весов глубоких нейронных сетей и обучении ансамбля нейронных сетей // Вестник Санкт-Петербургского университета. Сер. 10. Прикладная математика. Информатика. Процессы управления. 2016. Вып. 4. С. 66–74. DOI: 10.21638/11701/spbu10.2016.406

References

1. Raina R., Battle A., Lee H., Packer B., Ng A. Y. Selftaught learning: Transfer learning from unlabeled data. *Proceedings of the 24th Intern. conference on machine learning*, 2007, June 20–24, pp. 759–766.

2. Hinton G. E., Salakhutdinov R. R. Reducing the dimensionality of data with Neural Networks. *Science*, 2006, 28 July, vol. 313, no. 5786, pp. 504–507.

3. Vincent P., Larochelle H., Lajoie I., Bengio Y., Manzagol P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research archive*, 2010, vol. 11, pp. 3371–3408.

4. Masci J., Meier U., Ciresan D., Schmidhuber J. Stacked convolutional auto-encoders for hierarchical feature extraction. *21st Intern. conference on Artificial Neural Networks*. Espoo, Finland, 2011, June 14–17, pt I, pp. 52–59.

5. Gehring J., Miao Y., Metze F., Waibel A. Extracting deep bottleneck features using stacked auto-encoders. *Acoustics, speech and signal processing* (*ICASSP*). *IEEE Intern. conference*, 2013, pp. 3377–3381.

6. Baldi P., Hornik K. Neural Networks and principal component analysis: learning from examples without local minima. *Neural Networks*, 1989, vol. 2, pp. 53–58.

7. Caruana R. Multitask learning. *Machine learning*, 1997, vol. 28, pp. 41–75.

8. *UFDL.* Available at http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial (accessed: 19.04.2016).

9. Ciresan D. C, Meier U., Schmidhuber J. Transfer learning for Latin and Chinese characters with deep Neural Networks. *The 2012 Intern. joint conference on Neural Networks* (*IJCNN*), 2012, pp. 1–6.

10. *CIFAR.* Available at http://www.cs.toronto.edu/~kriz/cifar.html (accessed: 19.04.2016).

11. Rolfe J. T., LeCun Y. Discriminative recurrent sparse auto-encoders. *The Intern. conference on learning representations*, 2013.

12. Masci J., Meier U., Cires D. Bëan, Schmidhuber J. Stacked convolutional auto-encoders for hierarchical feature extraction. *Intern. conference artificial Neural Networks and machine learning*, 2011, pp. 52–59.

13. Glorot X., Bengio Y. Understanding the difficulty of training deep feedforward neural networks. *Intern. conference on artificial intelligence and statistics*, 2010, pp. 249–256.

14. Pascanu R., Mikolov T., Bengio Y. *Understanding the exploding gradient problem.* Tech. Rep. Montreal, Universite de Montreal Publ., 2012, 11 p.

For citation: Drokin I. S. About an algorithm for consistent weights initialization of deep Neural Networks and Neural Networks ensemble learning. *Vestnik of Saint Petersburg University. Series 10. Applied mathematics. Computer science. Control processes*, 2016, issue 4, pp. 66–74. DOI: 10.21638/11701/spbu10.2016.406

Статья рекомендована к печати проф. Е. И. Веремеем.

Статья поступила в редакцию 19 мая 2016 г.

Статья принята к печати 29 сентября 2016 г.