УДК 004.032.26
DOI: 10.25559/SITITO.019.202302.324-332

Оригинальная статья

Исследование влияния batch size на качество обучения нейронных сетей

А. А Лисов[*1], А. Г. Возмилов[1], В. Г. Урманов[2], С. А. Панишев[1]
[1] ФГАОУ ВО «Южно-Уральский государственный университет (национальный исследовательский университет)», г. Челябинск, Российская Федерация
Адрес: 454080, Российская Федерация, г. Челябинск, пр. Ленина, д. 76
[2] ФГБОУ ВО «Башкирский государственный аграрный университет», г. Уфа, Российская Федерация
Адрес: 450001, Российская Федерация, г. Уфа, ул. 50-летия Октября, д. 34
[*] lisov.andrey2013@yandex.ru

Аннотация
Нейронные сети обучаются с использованием градиентного спуска — метода оптимизации, при котором оценка ошибки, используемая для обновления весов модели нейросети, рассчитывается на основе подмножества обучающего набора данных. Количество примеров из набора обучающих данных, используемых для оценки градиента ошибки, называется размером пакета (batch size), он является важным гиперпараметром, влияющим на динамику алгоритма обучения. В статье приведен анализ влияния размера пакета обучения для нейросетей разных типов — нейросетей глубокого обучения, сверточных, рекуррентных и больших языковых моделей на точность прогнозирования. Однако, как выяснилось в процессе исследования, неоднократное упоминание в источниках того, что размер batch size влияет на скорость обучения, на практике данное высказывание не было подтверждено экспериментальными значениями. С этой целью был проведен эксперимент проверки влияния размера пакета обучающей выборки не только на точность распознавания (accuracy) и величину потерь (loss — разница между полученным значением предсказания и реальным), но и на затраченное время на процесс обучения. Результаты исследования размера пакета выявили, что он оказывает решающее влияние на точность распознавания изображений сверточных нейронных сетей, рекуррентных, нейросетей глубокого обучения и больших языковых моделей. Чем больше значение параметра, тем выше точность прогнозирования. С другой стороны, большое значение размера пакета приводит к увеличению требований к вычислительным ресурсам.

Ключевые слова: batch size, нейронные сети, точность, время обучения, градиентный спуск

Конфликт интересов: авторы заявляют об отсутствии конфликта интересов.

Для цитирования: Исследование влияния batch size на качество обучения нейронных сетей / А. А. Лисов [и др.] // Современные информационные технологии и ИТ-образование. 2023. Т. 19, № 2. С. 324-332. doi: https://doi.org/10.25559/SITITO.019.202302.324-332

Original article

The Impact of Batch Size on the Quality of Training of Neural Networks

A. A. Lisov[a*], A. G. Vozmilov[a], V. G. Urmanov[b], S. A. Panishev[a]
[a] South Ural State University (National Research University), Chelyabinsk, Russian Federation
Address: 76 Lenin prospekt, 454080 Chelyabinsk, Russian Federation
[b] Bashkir State Agrarian University, Ufa, Russian Federation
Address: 34 50th anniversary of October St., 450001 Ufa, Russian Federation
[*] lisov.andrey2013@yandex.ru

Abstract
Neural networks are trained using gradient descent, an optimization technique in which the error estimate used to update the weights of the neural network model is calculated based on a subset of the training dataset. The number of examples from the training dataset used to estimate the error gradient is called the batch size, and is an important hyperparameter that affects the dynamics of the learning algorithm. The article analyzes the impact of the batch size for neural networks of various types - deep learning neural networks, convolutional, recurrent and large language models on the accuracy of forecasting. However, as it turned out during the study, the repeated mention in the sources that the size of the batch size affects the speed of learning, in practice, this statement was not confirmed by experimental values. For this purpose, an experiment was conducted to check the impact of the size of the training sample packet not only on recognition accuracy and the amount of losses (the difference between the obtained prediction value and the real one), but also on the time spent on the learning process. The results of the study of the batch size revealed that it has a decisive influence on the accuracy of image recognition of convolutional neural networks, recurrent neural networks, deep learning neural networks and large language models. The larger the parameter value, the higher the prediction accuracy. On the other hand, a large value of the packet size leads to an increase in the requirements for computing resources.

Keywords: batch size, neural networks, accuracy, training time, gradient descent

Conflict of interests: The authors declare no conflict of interests.

For citation: Lisov A.A., Vozmilov A.G., Urmanov V.G., Panishev S.A. The Impact of Batch Size on the Quality of Training of Neural Networks. *Modern Information Technologies and IT-Education.* 2023;19(2):324-332. doi: https://doi.org/10.25559/SITITO.019.202302.324-332

Введение

В последние годы появляется все больше задач на основе больших данных, решением которым являются нейронные сети глубокого обучения; это, например, классификация ImageNet, обработка естественного языка и распознавание лиц. Наборы данных часто включают десятки миллионов изображений, текстов, голосов или другой информации, соответственно, нейронные сети для работы с ними также содержат несколько миллионов параметров. Для таких сложных задач уделяется много внимания в вопросе о том, как повысить эффективность обучения.

Нейронные сети обучаются с использованием алгоритма оптимизации стохастического градиентного спуска. Этот процесс включает в себя использование текущего состояния модели для создания прогноза, сравнение прогноза с ожидаемыми значениями и использование разницы в качестве оценки градиента ошибки. Затем этот градиент ошибки используется для обновления весов модели, и после процесс снова повторяется.

Batch size (размер пакета) — это гиперпараметр, определяющий количество выборок, которые необходимо обработать перед обновлением внутренних параметров модели. В конце анализа пакета прогнозы сравниваются с ожидаемыми выходными переменными и вычисляется ошибка. Ошибка используется для улучшения модели, этот алгоритм известен как алгоритм обратного распространения ошибки (backpropagation algorithm). Соответственно, от размера набора обучающих будет зависеть точность распознавания и время обучения модели нейронной сети.

Один из простых вариантов решения поставленной задачи — использовать большой размер пакета обучающей выборки. Однако в этом случае большой размер пакета ухудшит точность прогнозирования модели и уменьшит сходимость [1-5], в то же время увеличение размера пакета помогает найти более плоские минимумы [6, 7].

Когда используются все обучающие сэмплы (samples, образцы) для создания одной партии пакета, то в этом случае алгоритм обучения называется пакетным градиентным спуском (batch gradient descent). Когда пакет имеет размер одного образца, алгоритм обучения называется стохастическим градиентным спуском (stochastic gradient descent). Когда размер пакета больше одной выборки и меньше размера обучающего набора данных, то алгоритм обучения называется мини-пакетным градиентным спуском (mini-batch gradient descent). В случае мини-пакетного градиентного спуска популярные размеры пакетов включают в себя 16, 32, 64 и 128 образцов [8-11].

Материалы и методы

Как уже упоминалось ранее, нейронные сети обучаются с использованием алгоритма оптимизации стохастического градиентного спуска для вычисления оценки градиента ошибки. Градиент ошибки является статистической оценкой. Чем больше обучающих примеров используется в оценке, тем точнее будет эта оценка и тем больше вероятность того, что веса сети будут скорректированы таким образом, чтобы улучшить производительность модели. Более точная оценка градиента ошибки достигается за счет выполнения моделью большего количества прогнозов (увеличения batch size), прежде чем вычислять оценку и обновлять веса [11].Использование же меньшего количества примеров приводит к менее точной оценке градиента ошибки, которая сильно зависит от конкретных используемых обучающих примеров. Что приводит к зашумленной оценке, которая, в свою очередь, приводит к зашумленным обновлениям весов модели, однако при этом задействуется куда меньше вычислительных ресурсов. Стоит отметить, что иногда эти зашумленные обновления могут привести к более надежной модели.

Как уже упоминалось ранее, когда пакет имеет размер одного сэмпла, алгоритм обучения называется стохастическим градиентным спуском (stochastic gradient descent). Когда размер пакета больше одной выборки и меньше размера обучающего набора данных, то алгоритм обучения называется мини-пакетным градиентным спуском (mini-batch gradient descent). Когда используются все обучающие образцы для создания одной партии пакета, то в этом случае алгоритм обучения называется пакетным градиентным спуском (batch gradient descent).

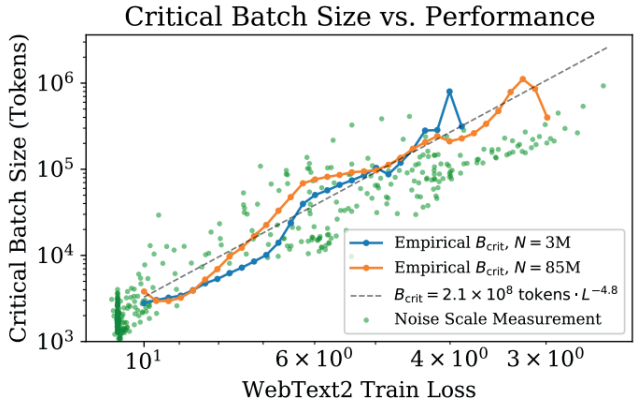
Размер пакета 32 означает, что 32 сэмпла из обучающего набора данных будут использоваться для оценки градиента ошибки до обновления весов модели. Одна эпоха обучения означает, что алгоритм обучения сделал один проход через набор обучающих данных, где примеры были разделены на случайно выбранные группы batch size.

На практике, учитывая очень большие наборы данных, которые часто используются для обучения нейронных сетей глубокого обучения, размер пакета редко устанавливается равным размеру обучающего набора данных. Меньшие размеры пакетов используются по двум основным причинам:

- меньшие размеры пакетов создают небольшой шум, предлагая эффект регуляризации и меньшую ошибку обобщения.
- пакеты меньшего размера упрощают размещение в памяти одного пакета обучающих данных и снижают затраты на вычислительные мощности и необходимую память.

Оптимальный размер пакета при обучении large language models (LLM)

Large language models (большие языковые модели) — это тип нейронных сетей глубокого обучения, которые могут анализировать и создавать текст. Их обучают с использованием больших объемов текстовых данных, что помогает им лучше справляться с такими задачами, как генерация текста. Языковые модели являются основой для многих приложений применения по обработке естественного языка, таких как преобразование речи в текст и анализ настроений. Примеры LLM — ChatGPT, LaMDA, PaLM и т.д.Оптимальный размер пакета [12] в этом случае примерно равен степени потерь, и его можно определить путем измерения шкалы градиентного шума, примерно равняется 1-2 миллионам токенов при сходимости для самых больших моделей. Согласно исследованию [12], критический размер партии Bcrit (рис. 1) подчиняется степенному закону потерь при увеличении производительности и не зависит напрямую от размера модели. В [12] обнаружили, что критический размер пакета примерно удваивается на каждые 13 % снижения потерь. Bcrit измеряется эмпирически по данным, но также грубо предсказывается по шкале градиентного шума.



Р и с. 1. Критический размер пакета для LLM

F i g. 1. Critical packet size for LLM

Источник: [12].

Source: [12].

Оптимальный размер пакета при обучении convolutional neural networks (CNN)

Convolutional neural networks (сверточная нейронная сеть) — тип нейронной сети глубокого обучения, которая может принимать на вход изображение и назначать важность различным аспектам/объектам на изображении и иметь возможность отличать одно от другого. Предварительная обработка, необходимая в CNN, намного ниже по сравнению с другими алгоритмами классификации. Они применяются в распознавании изображений и видео, рекомендательных системах [13] классификации изображений, сегментации изображений, анализе медицинских изображений, обработке естественного языка [14], интерфейсах мозг-компьютер [15] и финансовых временных рядах [16].

Процесс оптимизации функции F(w) называют обучением сети. Стохастический градиентный спуск (SGD) и его варианты часто используются для обучения сверточных сетей [17]. Эти методы минимизируют целевую функцию F, итеративно выполняя шаги в форме:

$$w_{t+1} = w_t - \eta \left(\frac{1}{|B|} \sum_{x \in B} \nabla f(x, w_t) \right)$$

где B – пакет (batch), отобранный из X;

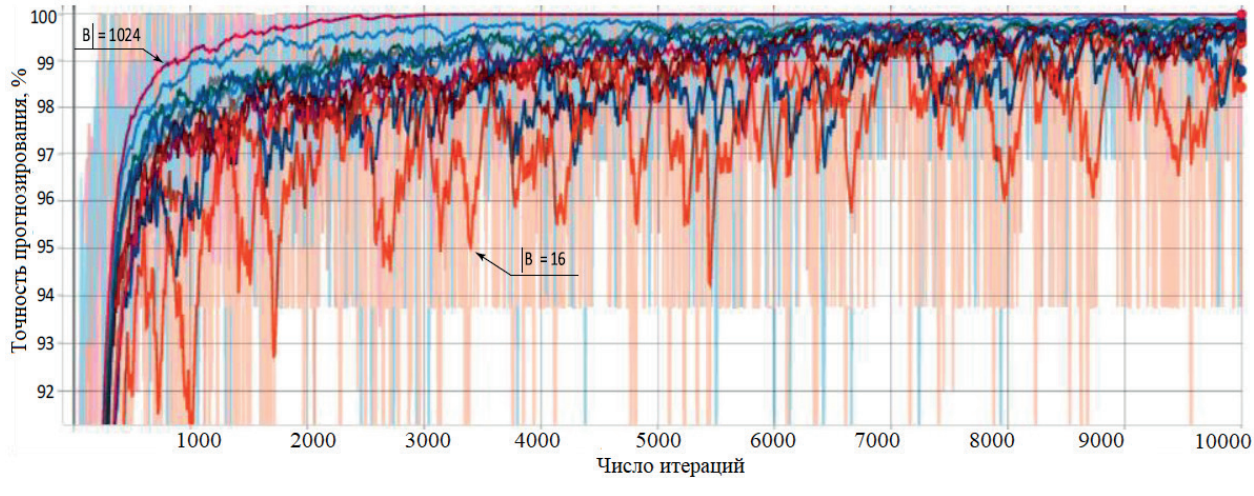
|B| — размер пакета;

η — скорость обучения;

t — индекс итерации.

Эти методы можно интерпретировать как градиентный спуск с использованием зашумленных градиентов, которые часто называют мини-пакетными градиентами спусками с заданным размером. SGD и его варианты используются в мелкосерийном режиме, где[8-10], [18].

Согластно исследованию [17], влияние размера пакета на точность распознавания сверточной нейронной сети на примере датасета MNIST имеет решающее значение (рис. 2) и состовляет 1024 семпла, что примерно равно 10 % об общего размера обучающей выборки.



Р и с. 2. Точность прогнозирования обученной CNN на наборе данных MNIST при различном размере пакета

F i g. 2. Prediction accuracy of a trained CNN on the MNIST dataset for different batch sizes

Источник: [17].

Source: [17].

Оптимальный размер пакета при обучении deep neural networks (DNN)

Deep neural networks (нейросеть глубокого обучения) — нейронная сеть содержащая не менее двух скрытых слоев. Обучение глубоких нейронных сетей с помощью стохастического градиентного спуска или его вариантов требует тщательного выбора как скорости обучения, так и размера пакета. В то время как меньшие размеры пакетов обычно сходятся за меньшее количество эпох обучения, большие размеры пакетов обеспечивают больший параллелизм и, следовательно, лучшую вычислительную эффективность.

Стохастический градиентный спуск (SGD) и его варианты

являются наиболее широко используемыми методами обучения DNN. Этот тип сетей имеет большой размер, и процесс обучения требует больших объемов данных. Поэтому при реализации обучающую выборку обычно делят серию пакетов некоторого фиксированного размера, меньшего, чем размер обучающей выборки, и большей, чем один [6].

Оптимальный размер пакета при обучении recurrent neural networks (RNN)

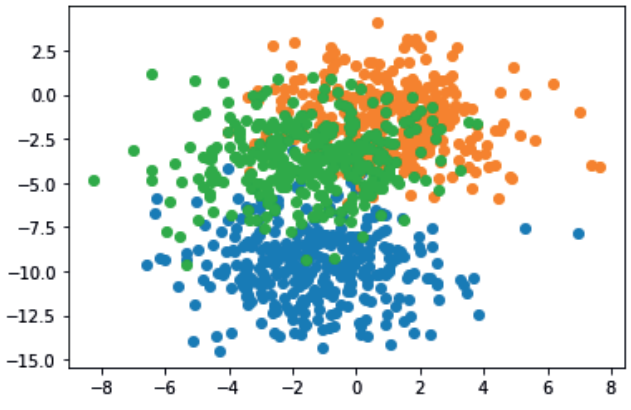
Recurrent neural networks (рекуррентные нейронные сети) — это тип искусственной нейронной сети, которая использует последовательные данные или данные временных рядов. Эти алгоритмы глубокого обучения обычно используются для порядковых или временных задач, таких как языковой перевод, обработка естественного языка, распознавание речи и субтитры к изображениям. Рекуррентные нейронные сети отличаются своей «памятью», поскольку они берут информацию из предыдущих входов, чтобы влиять на текущий ввод и вывод. В то время как традиционные глубокие нейронные сети предполагают, что входы и выходы независимы друг от друга, выходные данные рекуррентных нейронных сетей зависят от предшествующих элементов в последовательности.

Поскольку данный тип нейронных сетей отличается от нейросетей глубокого обучения наличием специальной «памяти», то вариант с пакетным градиентным спуском в этом случае потребует еще более значительных вычислительных ресурсов. Поэтому на практике также применяют метод с мини-пакетным градиентным спуском. С точки зрения производительности и соотношения размера пакета к обучающей выборке было проведено исследование [19].В этой статье была представлена эффективная реализация вычисления потерь и градиента RNN-T путем формулирования прямой и обратной рекурсии в матричной форме. Авторами были проведены эксперименты с различной длиной последовательностей и размерами пакетов на разных аппаратных архитектурах (CPU, GPU и TPU). Результаты тестов показывают, что при увеличении размера пакета с 1 до 32 можно повысить пропускную способность в 22,9 раза на GPU и в 3,2 раза на TPU. Кроме того, при размере пакета 32 они сумели достичь примерно в два раза большей пропускной способности при работе на одном ядре TPU по сравнению с графическим процессором Tesla P100.

Проверка полученных данных из приведенных выше исследований

Код для проведения эксперимента влияния размера пакета на точность, потери и время обучения был разработан на платформе Google.Collab на языке программирования Python, основным фреймворком задачи для машинного обучения являлся TensorFlow, а также специализированная библиотека Keras для удобной работы с фреймворком. Разработанный код программы можно найти на GitHub одного из авторов[1].Для всех методов была определена одна и та же архитектура модели нейронной сети, код которой приведен ниже.

```
# Определение модели
model = Sequential()
model.add(Dense(50, input_dim=2, activation='relu', kernel_
initializer='he_uniform'))
 model.add(Dense(3, activation='softmax'))
```



Р и с. 3. Датасет для исследования

F i g 3. Research dataset

Источник: здесь и далее в статье все рисунки и таблицы составлены авторами.

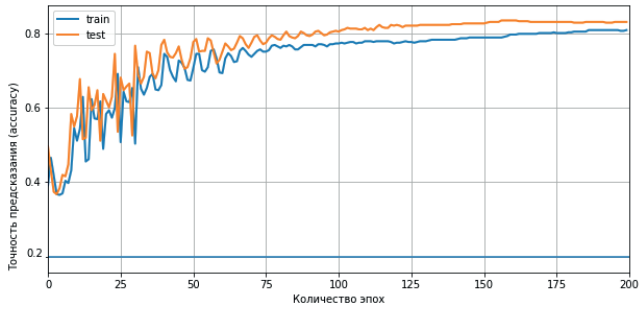
Source: Hereinafter in the article, all tables and figures are compiled by the authors.

В качестве датасета (рис. 3) для исследования при помощи библиотеки sklearn был сгенерирован двумерный набор данных выборок с тремя «каплями» в качестве задачи прогнозирования нескольких классов. Каждое наблюдение имеет два входа и 0, 1 или 2 значения класса. Функция make_blobs() из данной библиотеки позволяет сгенерировать этот набор для создания точек с гауссовым распределением.

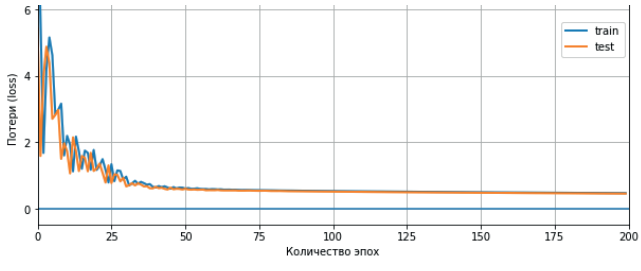
Результаты

Как уже упоминалось ранее, на основе сгенерированного датасета для 2D-классификации был проведен эксперимент влияния размера пакета на одну и ту же модель нейронной сети с целью проверки точности и скорости работы. На рис. 4 показан график процесса обучения тестовой модели методом пакетного градиентного спуска (BGD), на рис. 5 приведен график полученных в процессе обучения потерь. Время обучения в данном случае оказалось наименьшим (10,6 секунд) в сравнении со всеми остальными итерациями (см. таблицу 1), и поэтому это значение было принято за референсные 100 %. Стоит отметить, что точность при этом способе обучения оказалось одной из наилучших, причина этого объяснялась ранее — высокая устойчивость к шуму из-за наличия большого пакета, позволяющего усреднить уровень ошибки или вовсе свести к минимуму. Однако все это справедливо при условии использования довольно простой модели и небольшого датасета (в нашем случае размерностью в 500 семплов), в реальных условиях с наборами данных в несколько тысяч единиц информации этот метод не представляется оптимальным из-за очень высоких требований к вычислительным ресурсам и видеопамяти.

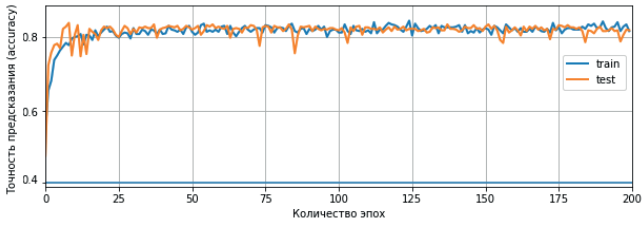
[1] Lisov A. Batch size study [Электронный ресурс] // GitHub, 2023. URL: https://github.com/AnLiMan/Batch_size_study (дата обращения: 26.02.2023)



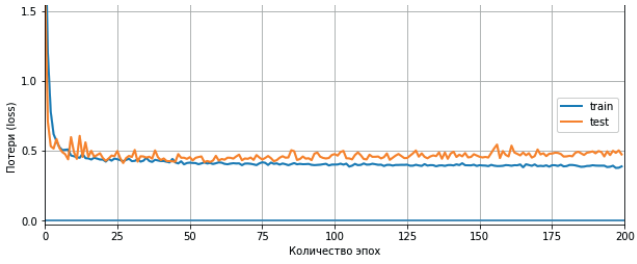
Р и с. 4. График процесса обучения BGD
F i g. 4. Graph of the BGD learning process



Р и с. 5. График потерь BGD
F i g. 5. Graph of BGD losses



Р и с. 6. График процесса обучения SGD
F i g. 6. Graph of the SGD learning process



Р и с. 7. График потерь SGD
F i g. 7. Graph of SGD losses

Далее был проведен опыт с методом стохастического градиентного спуска для обучения нейронной сети. Результаты опыта показали, что данный метод оказался самым медленным (см. таблицу 1) из представленных: 302,92 секунды занял весь процесс обучения. Рис. 6 наглядно показывает, что процесс обучения носит сильно зашумленный характер по сравнению с тем же BGD, это все объясняется малым размером пакета. Из-за небольшого batch size требуется намного больше итераций в одной эпохе обучения (500), при этом, в совокупности с заданным количеством эпох (200), в итоге получаем высокие временные затраты на процесс обучения. График потерь (рис. 7) также носит довольно сильно зашумленный характер по сравнению с BGD. Конечная точность обученной сети также оказалась ниже, чем при обучении методом пакетного градиентного спуска: 0,824 против 0,834. Поэтому, согласно исследованиям, приведенным выше, и полученному результату, можно прийти к выводу о нецелесообразности в большинстве случаев применения данного метода обучения для искусственных нейронных сетей.

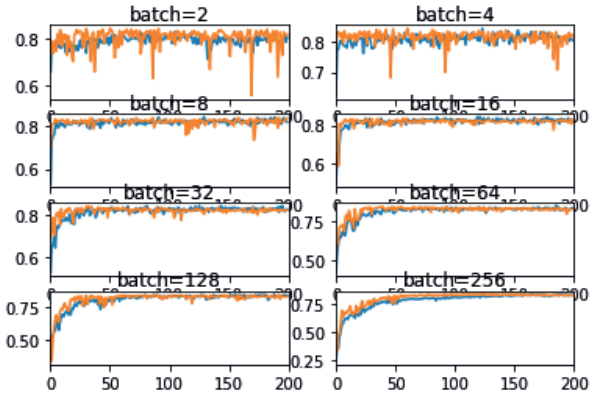
Для проверки метода мини-пакетного градиентного спуска была проведена серия проверок точности прогнозирования, величины потерь и времени, затраченного на обучения при размере пакета: 2, 4, 8, 16, 32, 64, 128 и 256 сэмпла. Результат работы приведен на рис. 8 и в таблице 1.

Т а б л и ц а 1. **Результаты исследования**
T a b l e 1. **Study Results**

Метод	Потери (val_loss)	Точность (val_accuracy)	Время обучения, сек	%
BGD	0,4465	0,834	10,6	100
SGD	0,4574	0,824	302,92	2857,7
MBGD (2)	0,6080	0,816	202,32	1908,7
MBGD (4)	0,5002	0,818	86,61	817,1
MBGD (8)	0,471	0,836	82,5	778,3
MBGD (16)	0,4366	0,826	40,4	381,1
MBGD (32)	0,4325	0,826	23,57	222,6
MBGD (64)	0,4243	0,824	17,69	166,9
MBGD (128)	0,3995	0,824	20,85	196,7
MBGD (256)	0,4145	0,828	20,85	196,7

Графики на рис. 8 показывают, что малый размер обучающей выборки (2, 4, 8) все так же имеет достаточно зашумленный процесс обучения. Стоит отметить нелинейность уменьшения времени, затраченного времени, а также тот факт, что размеры пакета в 16, 32 и в особенности 64 сэмпла показывают очень скорость обучения. Это объясняется тем, что в большинстве проектов, в том числе и в [8-10], наиболее часто используется именно эти размерности, поэтому код TensorFlow и иных библиотек для целей машинного обучения был оптимизирован именно под эти значения. Как и ожидалось от размерности 128

и 256, графики процесса обучения носят слабо зашумленный характер и при этом имеют малое время, затраченное на процесс обучения нейронной сети.



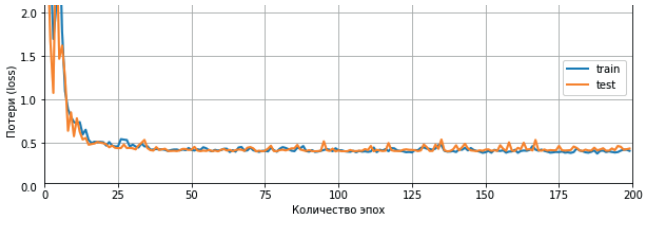
Р и с. 8. График процесса обучения MBGD с различными вариациями размера пакета

F i g. 8. Graph of the MBGD training process for different batch sizes

Ниже на рис. 9, 10 приведены графики точности прогнозирования и величины потерь для мини-пакетного градиентного спуска при размерности пакета, равной 64 образцам, как наиболее удачного для данного датасета размерностью 500 сэмплов.

Р и с. 9. График процесса обучения MBGD (batch size = 64)

F i g. 9. Graph of the MBGD learning process (batch size = 64)



Р и с. 10. График потерь MBGD (batch size = 64)

F i g. 10. Graph of MBGD losses (batch size = 64)

Обсуждение

Результаты опыта показывают, что размер пакета действительно имеет одно из решающих значений при обучении искусственных нейронных сетей методом оптимизации градиентного спуска, при применении алгоритма обратного распространения ошибки. Как и ожидалось, наиболее эффективным методом обучения, с точки зрения точности и затраченного времени, оказался пакетный градиентный спуск, однако этот способ целесообразен только при обучении небольших нейронных сетей, на небольшом наборе данных. Самым оптимальным для большинства прикладных задач и больших моделей оказался мини-пакетный градиентный спуск с наиболее популярным набором 16, 32, 64 и 128 сэмплов [20-22]. Эти значения обычно кратны степеням 2, что объясняется эффективной работой оптимизированных библиотек матричных операций [23]. Хотя в некоторых работах, таких как [24-26], предлагается устанавливать размер пакета, кратный 10, и получать высокие значения точности распознавания на разных наборах данных. Наименее оптимальным методом оказался метод стохастического градиентного спуска по причине количества затраченного времени на обучение.

Результаты существующих исследований также показали, что для сверточных нейронных сетей, нейросетей глубокого обучения, рекуррентных и для больших языковых моделей наиболее оптимальным методом обучения является mini-batch gradient descent.

Выводы

Результаты исследования размера пакета выявили, что он оказывает решающее влияние на точность распознавания изображений сверточных нейронных сетей, рекуррентных, нейросетей глубокого обучения и больших языковых моделей. Чем больше значение параметра, тем выше точность прогнозирования. С другой стороны, большое значение размера пакета приводит к увеличению требований к вычислительным ресурсам.

Результаты исследования показали, что для обучения нейронных сетей стоит выбирать размер пакета, равный 16, 32, 64 и 128 как наиболее оптимальный вариант с точки зрения точности и затраченного времени, поскольку именно под эти размерности и оптимизирован код TensorFlow и многих других библиотек для машинного обучения. Критического влияния на точность прогнозирования конкретной размерности пакета при этом не оказывается, таблица 1 и результаты исследования [25] это наглядно показывают.

References

[1] LeCun Y.A., Bottou L., Orr G.B., Müller KR. Efficient BackProp. In: Montavon G., Orr G.B., Müller KR. (eds.) Neural Networks: Tricks of the Trade. *Lecture Notes in Computer Science*. Vol. 7700. Berlin, Heidelberg: Springer; 2012. p. 9-48. https://doi.org/10.1007/978-3-642-35289-8_3

[2] Diamos G., Sengupta S., Catanzaro B., Chrzanowski M., Coates, A., Elsen E., Engel J., Hannun A., Satheesh S. Persistent RNNs: Stashing recurrent weights on-chip. In: Balcan M.F., Weinberger K.Q. (eds.) Proceedings of The 33rd International Conference on Machine Learning. New York, New York, USA: PMLR; 2016. Vol. 48. p. 2024-2033. Available at: https://proceedings.mlr.press/v48/diamos16.html (accessed 26.02.2023).

[3] Keskar N.S., Mudigere D., Nocedal J., Smelyanskiy M., Tang P.T.P. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In: 5th International Conference on Learning Representations (ICLR 2017). Toulon, France; 2017. p. 1-16. Available at: https://openreview.net/forum?id=H1oyRlYgg (accessed 26.02.2023).

[4] Goyal P., Dollar P., Girshick R., Noordhuis P., Wesolowski L., Kyrola A., Tulloch A., Jia Y., He K. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. arXiv:1706.02677. 2017. p. 1-12. https://doi.org/10.48550/arXiv.1706.02677

[5] Jastrzebski S., Kenton Z., Arpit D., Ballas N., Fischer, A., Bengio Y., Storkey A. Finding Flatter Minima with SGD. In: 6th International Conference on Learning Representations (ICLR 2018 Workshop Track). Vancouver Convention Center, Vancouver, BC, Canada; 2018. p. 1-4. Available at: https://openreview.net/forum?id=r1VF9dCUG (accessed 26.02.2023).

[6] Devarakonda A., Naumov M., Garland M. AdaBatch: Adaptive Batch Sizes for Training Deep Neural Networks. In: 6th International Conference on Learning Representations (ICLR 2018 Workshop Track). Vancouver Convention Center, Vancouver, BC, Canada; 2018. p. 1-4. Available at: https://openreview.net/forum?id=SkytjjU8G (accessed 26.02.2023).

[7] Smith S.L., Kindermans P., Ying C., Le Q.V. Don't Decay the Learning Rate, Increase the Batch Size. In: 6th International Conference on Learning Representations (ICLR 2018 Workshop Track). Vancouver Convention Center, Vancouver, BC, Canada; 2018. p. 1-11. Available at: https://openreview.net/forum?id=B1Yy1BxCZ (accessed 26.02.2023).

[8] Vozmilov A., Andreev L., Lisov A. Development of an Algorithm for the Program to Recognize Defects on the Surface of Hot-Rolled Metal. In: 2022 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM). Sochi, Russian Federation: IEEE Computer Society; 2022. p. 1004-1008. https://doi.org/10.1109/ICIEAM54945.2022.9787116

[9] Vozmilov A., Urmanov V., Lisov A. Using Computer Vision to Recognize Defects on the Surface of Hot-rolled Steel. In: 2022 International Ural Conference on Electrical Power Engineering (UralCon). Magnitogorsk, Russian Federation: IEEE Computer Society; 2022. p. 21-25. https://doi.org/10.1109/UralCon54942.2022.9906737

[10] Lisov A.A., Kulganatov A.Z., Panishev S.A. Using convolutional neural networks for acoustic-based emergency vehicle detection. *Modern Transportation Systems And Technologies*. 2023;9(1):95-107. (In Russ., abstract in Eng.) https://doi.org/10.17816/transsyst20239195-107

[11] Vozmilov A.G., Lisov A.A., Urmanov V.G., Sineva G.N. Determination of the type of potato leaves diseases with using machine learning. *Bulletin NGIEI.* 2023;3(142):7-16. (In Russ., abstract in Eng.) https://doi.org/10.24412/2227-9407-2023-3-7-16.

[12] Kaplan J., McCandlish S., Henighan T., Brown T.B., Chess B., Child R., Gray S., Radford A., Wu J., Amodei D. Scaling Laws for Neural Language Models. arXiv:2001.08361. 2020. https://doi.org/10.48550/arXiv.2001.08361

[13] Van den Oord A., Dieleman S., Schrauwen B. Deep content-based music recommendation. In: Burges C.J., Bottou L., Welling M., Ghahramani Z., Weinberger K.Q. (eds.) Advances in Neural Information Processing Systems. Curran Associates, Inc.; 2013. Available at: https://papers.nips.cc/paper_files/paper/2013/hash/b3ba8f1bee1238a2f37603d90b58898d-Abstract.html (accessed 26.02.2023).

[14] Collobert R., Weston J. A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning (ICML '08). New York, NY, USA: Association for Computing Machinery; 2008. p. 160-167. https://doi.org/10.1145/1390156.1390177

[15] Avilov O., Rimbert S., Popov A., Bougrain L. Deep Learning Techniques to Improve Intraoperative Awareness Detection from Electroencephalographic Signals. In: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC). Montreal, QC, Canada: IEEE Computer Society; 2020. p. 142-145. https://doi.org/10.1109/EMBC44109.2020.9176228

[16] Tsantekidis A., Passalis N., Tefas A., Kanniainen J., Gabbouj M., Iosifidis A. Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks. In: 2017 IEEE 19th Conference on Business Informatics (CBI). Thessaloniki, Greece: IEEE Computer Society; 2017. p. 7-12. https://doi.org/10.1109/CBI.2017.23

[17] Radiuk P.M. Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets. *Information Technology and Management Science*. 2017;20(1):20-24. https://doi.org/10.1515/itms-2017-0003

[18] Mishkin D., Sergievskiy N., Matas J. Systematic evaluation of convolution neural network advances on the Imagenet. *Computer vision and image understanding*. 2017;161:11-19. https://doi.org/10.1016/j.cviu.2017.05.007

[19] Bagby T., Rao K., Sim K.C. Efficient Implementation of Recurrent Neural Network Transducer in Tensorflow. In: 2018 IEEE Spoken Language Technology Workshop (SLT). Athens, Greece: IEEE Computer Society; 2018. p. 506-512. https://doi.org/10.1109/SLT.2018.8639690

[20] He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE Computer Society; 2016. p. 770-778. https://doi.org/10.1109/CVPR.2016.90

[21] Krizhevsky A. One weird trick for parallelizing convolutional neural networks. arXiv:1404.5997v2. 2014. https://doi.org/10.48550/arXiv.1404.5997

[22] Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: 3rd International Conference on Learning Representations (ICLR 2015). arXiv:1409.1556. 2015. p. 1-15. https://doi.org/10.48550/arXiv.1409.1556

[23] Takác M., Bijral A., Richtárik P., Srebro N. Mini-Batch Primal and Dual Methods for SVMs. In: Dasgupta S., McAllester D. (eds.) Proceedings of the 30th International Conference on Machine Learning (PMLR). 2013;28(3):1022-1030. Available at: https://proceedings.mlr.press/v28/takac13.html (accessed 26.02.2023).

[24] Wilson D.R., Martinez T.R. The general inefficiency of batch training for gradient descent learning. *Neural networks*. 2003;16(10):1429-1451. https://doi.org/10.1016/S0893-6080(03)00138-2

[25] Li M., Zhang T., Chen Y., Smola A.J. Efficient mini-batch training for stochastic optimization. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '14). New York, NY, USA: Association for Computing Machinery; 2014. p. 661-670. https://doi.org/10.1145/2623330.2623612

[26] Lin Z., Courbariaux M., Memisevic R., Bengio Y. Neural Networks with Few Multiplications. In: Bengio Y., LeCun Y. (eds.) 4th International Conference on Learning Representations, ICLR 2016. San Juan, Puerto Rico, May 2-4, 2016. Conference Track Proceedings. 2016. https://doi.org/10.48550/arXiv.1510.03009

Поступила 26.02.2023; одобрена после рецензирования 29.04.2023; принята к публикации 20.05.2023.
Submitted 26.02.2023; approved after reviewing 29.04.2023; accepted for publication 20.05.2023.

Об авторах:

Лисов Андрей Анатольевич, аспирант кафедры электропривода, мехатроники и электромеханики, ФГАОУ ВО «Южно-Уральский государственный университет (национальный исследовательский университет)» (454080, Российская Федерация, г. Челябинск, пр. Ленина, д. 76), **ORCID: https://orcid.org/0000-0001-7282-8470**, lisov.andrey2013@yandex.ru

Возмилов Александр Григорьевич, профессор кафедры электропривода, мехатроники и электромеханики, ФГАОУ ВО «Южно-Уральский государственный университет (национальный исследовательский университет)» (454080, Российская Федерация, г. Челябинск, пр. Ленина, д. 76), доктор технических наук, **ORCID: https://orcid.org/0000-0001-7282-8470**, vozmiag44@rambler.ru

Урманов Виль Губаевич, доцент кафедры прикладной механики и компьютерного инжиниринга, ФГБОУ ВО «Башкирский государственный аграрный университет» (450001, Российская Федерация, г. Уфа, ул. 50-летия Октября, д. 34), кандидат технических наук, **ORCID: https://orcid.org/0009-0007-8328-0392**, uvg55@mail.ru

Панишев Сергей Алексеевич, аспирант кафедры электропривода, мехатроники и электромеханики, ФГАОУ ВО «Южно-Уральский государственный университет (национальный исследовательский университет)» (454080, Российская Федерация, г. Челябинск, пр. Ленина, д. 76), **ORCID: https://orcid.org/0000-0003-2753-2341**, panishef.serega@mail.ru

Все авторы прочитали и одобрили окончательный вариант рукописи.

About the authors:

Andrey A. Lisov, Postgraduate student of the Department of Electric Drive, Mechatronics and Electromechanics, South Ural State University (National Research University) (76 Lenin prospekt, 454080 Chelyabinsk, Russian Federation), **ORCID: https://orcid.org/0000-0001-7282-8470**, lisov.andrey2013@yandex.ru

Alexander G. Vozmilov, Professor of the Department of Electric Drive, Mechatronics and Electromechanics, South Ural State University (National Research University) (76 Lenin prospekt, 454080 Chelyabinsk, Russian Federation), Dr. Sci. (Eng.), **ORCID: https://orcid.org/0000-0002-1292-3975**, vozmiag44@rambler.ru

Vil G. Urmanov, Associate Professor of Applied Mechanics and Computer Engineering Department, Bashkir State Agrarian University (34 50th anniversary of October St., 450001 Ufa, Russian Federation), Cand. Sci. (Eng.), **ORCID: https://orcid.org/0009-0007-8328-0392**, uvg55@mail.ru

Sergei A. Panishev, Postgraduate student of the Department of Electric Drive, Mechatronics and Electromechanics, South Ural State University (National Research University) (76 Lenin prospekt, 454080 Chelyabinsk, Russian Federation), **ORCID: https://orcid.org/0000-0003-2753-2341**, panishef.serega@mail.ru

All authors have read and approved the final manuscript.