**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Московский авиационный институт  
(национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ  
на тему:**

ОПТИМИЗАЦИЯ (РЕДУКЦИЯ) ПОЛНОСВЯЗНЫХ НЕЙРОННЫХ СЕТЕЙ

01.04.02 «Прикладная математика и информатика»

|  |
| --- |
| **Выполнил:**  магистрант  Рожлейс Иварс Андрисович |
|  |
| **Научный руководитель:**  ст. преп.  Аносова Наталья Павловна |
|  |
| **Рецензент:**  *звание, должность?*  *ФИО?* |

Москва, 2020

**Оглавление**

[Список условных обозначений 3](#_Toc38920726)

[Введение 4](#_Toc38920727)

[Часть 1. Долгая краткосрочная память. 8](#_Toc38920728)

[Часть 2. Сингулярное разложение. 13](#_Toc38920729)

[Часть 3. Обучение. 14](#_Toc38920730)

[Часть 4. Оптимизация. 16](#_Toc38920731)

[Список литературы 17](#_Toc38920732)

[Приложение 19](#_Toc38920733)

# Список условных обозначений

GRU – управляемый рекуррентный блок (англ. Gated Recurrent Unit, GRU)

LSTM – долгая краткосрочная память (англ. Long short-term memory, LSTM)

OBD – оптимальное повреждение мозга (англ. Optimal Brain Damage, OBS)

OBS – оптимальное прореживание мозга (англ. Optimal Brain Surgery, OBS)

ЯП – язык программирования

W – матрица весовых коэффициентов слоя

X – матрица входных значений

Y – матрица выходных значений

– состояние памяти ячейки

– скрытое состояние ячейки

# Введение

Процесс оптимизации заключается в том, чтобы улучшить какое-нибудь свойство. Приведи примеры параметров нейронной сети... Как правило в многопараметрических системах повышение качества одного параметра сопровождается понижением качества другого параметра. Иными словами, оптимизация без потерь как правило очень редко достижима, поэтому часто приходится иметь дело с компромиссным решением – улучшая одно свойство неизбежно ухудшается другое.

Идея редукции или, иными словами, упрощения нейронной сети заключается в том, чтобы избавиться от нейронов, которые вносят минимальное воздействие на итоговый результат работы сети, и лишних связей между ними. Некоторые нейроны могут оказаться абсолютно бесполезными и не влиять на итоговый результат работы сети, некоторые могут оказывать лишь минимальное воздействие. [1] Поэтому в соответствие с некоторым компромиссным соглашением между степенью упрощения и правдивостью результата работы сети можно отсечь те коэффициенты, которые вносят минимальный вклад в итоговый результат работы сети.

Процесс редукции нейронный сети можно описать как процесс оптимизации сети по критерию избыточности. Помимо процесса упрощения, который можно сказать работает по типу сверху-вниз, можно также отметить противоположный вариант оптимизации – наращивание сети, который работает по типу снизу-вверх. Нейроны и даже целые нейронные слои могут добавляться в сеть по мере требования. [2] В данной работе будет рассмотрен только процесс редукции нейронных сетей.

Нейроны в сети как правило группируются во взаимодействующие между собой нейронные слои, образуя таким образом различные топологии нейронных сетей. Возможно, стоит добавить различные типы топологий или привести ссылку. [3] В качестве математической модели взаимодействия между нейронами применяется матричная алгебра. Каждый нейрон имеет степень влияния или весовой коэффициент в пределах своего слоя. Весовые коэффициенты всех нейронов в пределах одного слоя образуют матрицу весовых коэффициентов этого слоя. Таким образом отсечение нейронов происходит посредством манипуляций с матрицей весовых коэффициентов слоя.

Наивным алгоритмом редукции нейронной сети может считать алгоритм, при котором малые коэффициенты принимаются равными нулю тем самым исключая нейроны с малыми весами из обработки. С учётом возможной оптимизации алгоритмов работы с разреженными матрицами, такой метод вполне может повысить быстродействие сети. [4] Тем не менее как показывает практика кажущийся в первую очередь верным, такой способ может быть редко применён, так как малые весовые коэффициенты не всегда оказывают малое влияние на итоговый результат нейронной сети. [5]

Более верным методом можно считать метод оптимального прореживания нейронных сетей OBD (англ. Optimal Brain Damage) предложенный Яном ЛеКуном (Yann LeCun) в 1990 году. Данный метод использует лучший критерий чем абсолютные значения весовых коэффициентов, а именно учитывает чувствительность нейронов к вариациям весовых коэффициентов других нейронов. [6] Этот алгоритм говорит, что безопасно можно принять равными нулю весовые коэффициенты только тех нейронов чья чувствительность к вариациям оказывается минимальной. Дальнейшее развитие этого метода получило название OBS (Optimal Brain Surgeon) и было предложено Бабаком Хассиби (Babak Hassibi) и Дэвидом Шторком (David G. Stork). В каком году? Этот алгоритм подстраивает значение зафиксированного параметра во время варьирования другого. [7]

Ещё одним методом оптимизации является метод упрощения сети с использованием штрафной функции. Идея метода заключается в том, чтобы организовать этап обучения сети таким образом, чтобы спровоцировать автоматическую минимизацию весовых коэффициентов маловажных нейронов тем самым по итогу обучения можно отбросить те нейроны чьё влияние на сеть минимальное. Иными словами, получаем автоматическое контрастирование сети ещё на этапе обучения. [8]

Выше представленные методы имеют ряд ограничений. Так, например, наивный метод является неточным и в ряде случаев приводит к ухудшению результатов работы нейронной сети. Далее опиши почему OBD и OBS плохо.

В последнее время часто применяется метод сингулярного разложения (англ. SVD Singular Value Decomposition) матриц весовых коэффициентов влияние которого и будет рассмотрено в данной работе.

Задача упрощения нейронных сетей имеет несколько важных следствий.

1. В силу того, что упрощённая сеть имеет меньше нейронов и меньше весовых коэффициентов, то, как следствие нужно производить меньше вычислений чтобы получить результат. Таким образом повышается быстродействие сети.
2. «Слабые» (по тому или иному критерию) нейроны оказывают меньший эффект воздействия на итоговый результат работы сети, однако вносят определённый уровень шума. Процесс контрастирования сети позволяет в определённой степени снизить шумовое воздействие «слабых» нейронов на результат работы сети. Таким образом результат освобождается от шумового воздействия.
3. Процесс упрощения сети подразумевает собой избавление от некоторых нейронов без существенной потери качества сети, что аналогично процессу сжатия информации с потерей данных. Итоговый размер сети становится меньше. В определённых ситуациях уменьшение объёма данных необходимых для хранения матриц весовых коэффициентов может оказаться полезным если, к примеру сеть реализуется на машине с небольшим объёмом памяти.
4. Прореживание нейронной сети также повышает её классифицирующую способность. [?]

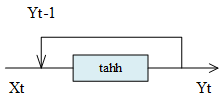
Почему это нужно?

Перед упрощением сети возможно провести также контрастирование её коэффициентов так чтобы в результате обучения сети сильные нейроны становились сильнее, а слабые слабее чтобы тем самым облегчить процесс выкидывания отдельных нейронов из сети и повысить эффективность упрощения.

# Часть 1. Долгая краткосрочная память.

Долгая краткосрочная память (англ. Long short-term memory, LSTM) это архитектурная разновидность рекуррентных нейронных сетей, которая была предложена Зеппом Хохрайтером и Юргеном Шмидхубером в 1997 году. (1) LSTM это архитектурное решение проектирования рекуррентных сетей, в состав которых входят несколько полносвязнных слоёв. Управляемый рекуррентный блок (англ. Gated Recurrent Network, GRU) ещё одна схожая архитектура рекуррентных сетей, предложенная Кёнхёном Тё в 2014 году. (2) По своим свойствам сеть GRU схожа с LSTM, однако имеет ряд допущений что в целом случаях позволяет ей быть более производительной взамен точности. Почему LSTM и GRU называют ячейками.

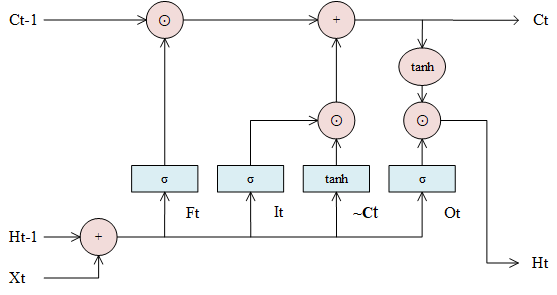
В основе работы рекуррентных нейронных сетей лежит принцип обратной связи, когда результат работы сети зависит от предыдущего результата. Ниже представлена схематическая диаграмма простой рекуррентной сети. Прямоугольником показан один слой с функцией гиперболического тангенса в качестве активационной функции. Стрелками показано направление потока данных.



Как видно из рисунка выходное значение Y на шаге t одновременно зависит как от входного значения X на этом же шаге t, так и от выходного значения Y посчитанного на предыдущем шаге t - 1. Обратная связь добавляет рекуррентным сетям временную зависимость и как следствие, если свёрточные сети хорошо обрабатывают информацию, распределённую в пространственной области, то рекуррентные сети в свою очередь хорошо справляются с информацией, распределённой во временной области.

Например, смысл абзаца складывается из отдельных предложений, смысл каждого из которых складывается из отдельных слов. Или для того, чтобы продолжить последовательность нужно помнить значения её предыдущих членов. Здесь имеется временная зависимость. Так нельзя определить смысл предложения, не прочитав последовательно и не запомнив все слова как нельзя предсказать следующий член последовательности, не предсказав и не запомнив все предыдущие её члены. В обоих задачах появляется необходимость наличия контекста. Иными словами говоря, сеть приобретает состояние, которое зависит от результатов работы на предыдущих шагах и динамически изменяется от шага к шагу. Подобные состояния имеют LSTM и GRU архитектуры.

Ниже представлена схематическая диаграмма классической LSTM архитектуры. (3) Прямоугольники по-прежнему представляют отдельные слои. Окружности представляют поэлементные операции. Символ ⊙ обозначает произведение Адамара (покомпонентное произведение).



Как видно исходя из этой диаграммы LSTM сеть имеет четыре полносвязных слоя, три из которых имеют сигмоидальную активационную функцию и один слой имеет активационную функцию гиперболического тангенса. Также на диаграмме виды два потока состояний: память блока и скрытое состояние .

Слои LSTM сети по сути являются фильтрами, так как в зависимости от весовых коэффициентов производят выборочное преобразование элементов. Каждый фильтр имеет своё предназначение и на диаграмме они обозначены буквами F, I, C и O. Первый фильтр забывания F (англ. forget) определяет какую информацию нужно забыть или удалить из памяти блока Ct. Пара фильтров входных данных I (англ. input) и кандидатов C (англ. candidate) наоборот определяют какую новую информацию нужно добавить в состояние памяти Ct. Наконец фильтр O (англ. output) участвует при формировании выходного значения. (3)

Эти процессы можно описать математически. Для фильтров эти выражения принимают следующий вид:

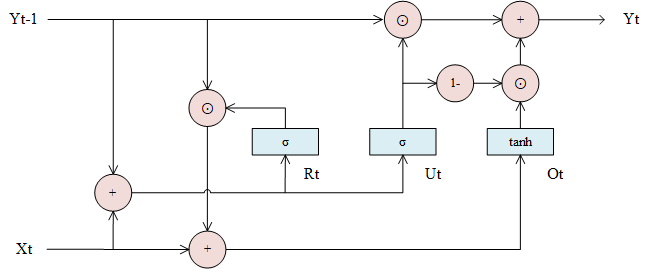
Для состояний и получаем выражения:

Можно рассмотреть предельный случай, когда и наблюдается режим устойчивого сохранения информации что можно интерпретировать как «не забывать ничего» () и «не добавлять новых данных» (). (3)

Так как область значений сигмоиды и гиперболического тангенса лежат в интервалах и соответственно, то на выходе скрытого состояния сети добавляется ещё одна матрица с весовыми коэффициентами, однако уже без активационной функции:

Таким образом обеспечивается масштабирование результата.

В данной работе будет использоваться архитектура LSTM тем не менее также полезно рассмотреть архитектуру GRU. Как уже было сказано выше архитектура GRU схожа с архитектурным решением LSTM сети. Ниже представлена диаграмма архитектуры GRU. Все условные обозначения сохранены из предыдущих диаграмм.



В отличие от сетей LSTM сети GRU имеют всего три фильтра, а также только одно скрытое состояние . Вследствие таких упрощений сети GRU работают быстрее чем LSTM, а также демонстрируют схожую эффективность, и тем не менее сети LSTM остаются сильнее. (4)

Можно добавить математическую модель GRU.

Напиши в чём преимущество использовать одну архитектуру вместо другой (LSTM vs GRU). Напиши, что в этой работе исследуется именно LSTM архитектура. Причины выбора именно LSTM можно не указывать.

В конце думаю нужно добавить, что умеет LSTM ячейка. Например, предсказывать временные ряди и т.д.

Приведи примеры погрешности при обучении на синусоиде.

# Часть 2. Сингулярное разложение.

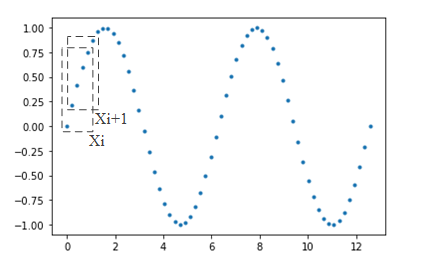
Сингулярное разложение (англ. Singular Value Decomposition, SVD) применяется…

Где ещё используется сингулярное разложение? В частности, используется при сжатии изображения.

# Часть 3. Обучение.

В качестве языка программирования был выбран язык программирования Python с библиотекой TensorFlow в качестве библиотеки машинного обучения. В данной работе используется Python версии 3.7.6 и библиотека TensorFlow версии 1.15.2. В работе была использована собственная реализация LSTM сети (3) без использования заранее разработанных шаблонов, которые предоставляет, например библиотека Keras. В качестве проверки, модель была обучена на нескольких синтезированных последовательностях.

Обучение предсказания последовательностей сводится к тренировке сети таким образом чтобы для каждой входной последовательности, состоящей из дискрет сеть, выдавала предсказанную выходную последовательность также состоящую из дискрет. Количество дискрет из которых состоит одна последовательность далее будем называть размером этой последовательности. Ниже представлен график с выбранными точками для тренировки сети на первом шаге, когда :



На данном графике выбирается группа из дискрет. Эту группу определим как входное значение LSTM сети и назовём её . Смещённую ровно на одну дискрету группу будем считать выходным значением сети . Функцию потерь в свою очередь определим как , то есть минимизирующую разницу между входным и выходным значениями. При достижении необходимого значения функции потерь алгоритм обучения переходит к следующему набору дискрет на следующем шаге. Для простоты допустим, что все точки на графике являются тренировочными данными. Тогда алгоритм обучения продолжается до тех пор, пока не будет достигнут конец тренировочных данных.

Процесс обучения организован таким образом что имеется два цикла: внешний и внутренний. Внешний цикл задаёт количество эпох и задаёт количество проходов по всему тренировочному набору данных. Внутренний цикл задаёт количество повторений обучения одной последовательности из дискрет. Оба значения являются макропараметрами и могут быть использованы для тонкой настройки процесса обучения.

Добавь псевдоалгоритм показывающий как происходит обучение.

Покажи данные которые предсказывает твоя модель.

Расхождение в фазах между гребнями синусоид может быть связано с выбором недостаточно точного значения размера последовательности.

# Часть 4. Оптимизация.

А

# Список литературы

1. *Long short-term memory.* **S., Hochreiter и J., Schmidhuber.** б.м. : Neural Computaion, 1997 г.

2. **K., Cho, и др.** Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arxiv.* [В Интернете] 2014 г. https://arxiv.org/abs/1406.1078.

3. **A., Zhang, и др.** *Dive into Deep Learning.* [В Интернете] https://d2l.ai/.

4. **G., Weiss, Y., Goldberg и E., Yahav.** On the Practical Computational Power of Finite Precision RNNs for Language Recognition. *arxiv.* [В Интернете] https://arxiv.org/abs/1805.04908.

5. **Ф., Уоссермен.** *Нейрокомпьютерная техника: Теория и практика.* 1992.

6. **N., Shukla.** *Machine Learning with TensorFlow.* б.м. : Manning, 2017.

7. **T., Hope, Y., Resheff и I., Lieder.** *Learning TensorFlow.* Sebastopol : O’Reilly, 2017. 978-1-491-97851-1.

8. **С., Николенко, А., Кадурин и Е., Архангельская.** *Глубокое обучение.* Санкт-Петербург : Питер, 2018. 978-5-496-02536-2.

9. **Р., Тарик.** *Создаём нейронную сеть.* Санкт-Петербург : Диалектика, 2017. 978-5-9909445-7-2.

10. **С., Хайкин.** *Нейронные сети. Полный курс.* Москва : Вильямс, 2006. 5-8459-0890-6.

11. **C., Zhang, и др.** Understanding Deep Learning Requires Rethinking. *arxiv.* [В Интернете] 2016 г. https://arxiv.org/abs/1611.03530.

12. *Second order derivatives for network pruning: Optimal Brain Surgeon.* **B., Hassibi и D., Stork.** Stanford : б.н., 1992 г.

13. *Optimal Brain Surgeon: Extensions and performance comparisons.* **B., Hassibi, D., Stork и G., Wolff.** Stanford : б.н., 1993 г.

14. **Y., LeCun.** Optimal Brain Damage. *Yann LeCun's home page.* [В Интернете] 1990 г. http://yann.lecun.com/exdb/publis/pdf/lecun-90b.pdf.

15. **Дж., Райс.** *Матричные вычисления и математическое обеспечение.* Москва : Мир, 1984.

16. **Р., Хорн и Ч., Джонсон.** *Матричный анализ.* Москва : Мир, 1989. 5-03-001042-4.

# Приложение

А