УДК 004.8

РАЗРАБОТКА И ИССЛЕДОВАНИЕ НЕЙРОСЕТЕВОГО СЖАТИЯ ДАННЫХ С ВИЗУАЛИЗАЦИЕЙ НА WEB-САЙТЕ

Балахнин Е. Е., студент гр. ПИб-192, III курс
Научный руководитель: Тайлакова А.А., старший преподаватель
Кузбасский государственный технический университет
имени Т.Ф. Горбачева
г. Кемерово

Существует такой класс задач, решаемых нейронными сетями как сжатие данных. Алгоритм, описанный в статье, не претендует на использование в реальных боевых условиях по причине существования более эффективных алгоритмов. Сразу оговорюсь, что речь пойдет только о сжатии без потерь.

Существует несколько основных популярных архитектур нейронных сетей, решающих задачу сжатия данных, но будет рассмотрен только один из них, а именно метод «бутылочного горлышка». Назван из-за формы нейронной сети, которая представляет из себя постепенно уменьшающиеся нейронные слои, до пика после которого они начинают увеличиваться. Главная задача такой сети, добиться тех же данных на выходе, что и на входе. Сжатие происходит без потерь, но в частных случаях целенаправленно допускаются потери.

Рассмотрим работу этой сети на практике, проанализируем результат и сделаем выводы. Для удобства, данные и обученные модели были интегрированы и визуализированы на сайте, в свою очередь сделанном с помощью библиотеки streamlit.

Для начала рассмотрим метод бутылочного горлышка. Для анализа работы сети поставим нашей нейронной сети невыполнимую задачу, сжать несжимаемые данные (рис. 1).

Полученные исходные данные представляют собой всевозможные комбинации 3-х байт. Перед сетью стоит следующая задача: сжать и разжать все цепочки данных без потерь. Наша модель состоит из 3-х слоев по 3, 2 и 3 нейрона соответственно. Получается модель с формой, имеющей название «Песочные часы» или «Бутылочное горлышко».

Моя модель, на 3-2-3 нейрона соответственно, выдает нам следующие данные (рис. 2):

Для визуализации данных в наш привычный вид, прогоним их через пороговую функцию с значением 0,5. То есть, все значения, которые меньше порогового, мы приравняем к нулю, все те, что больше или равны мы приравняем к единице.

Значение пороговой функции можно менять в процессе получения данных, от чего получается другой результат, имеющий тот же самый вердикт (рис. 3).

	0	1	2
0	0	0	0
1	1	1	1
2	0	1	0
3	1	0	1
4	1	1	0
5	0	0	1
6	0	1	1
7	1	0	0

Рисунок 1 – Исходные данные

	0	1	2
0	0.21701788902282715	0.2676132917404175	-0.18139472603797913
1	0.7829821705818176	0.7323865294456482	1.1813945770263672
2	-0.16602924466133118	0.7952606678009033	0.13877800107002258
3	1.1660292148590088	0.20473939180374146	0.8612220883369446
4	0.5233420729637146	0.41221511363983154	0.3984164595603943
5	0.4766578674316406	0.5877848267555237	0.6015836000442505
6	0.0936107188463211	1.1154322624206543	0.9217564463615417
7	0.9063893556594849	-0.11543220281600952	0.07824373245239258

Рисунок 2 – Выходные данные

Порог изменения значения для неправильной сети (переменная change_variable)

```
0.5
```

```
for i in range(len(res)):
    for j in range(3):
        if res[i][j] >= var1:
            res[i][j] = 1
        else:
            res[i][j] = 0")
```

Рисунок 3 – Пороговая функция и её значение

Ниже приведено сравнение исходных данных с данными, полученными от нейронной сети и преобразованные пороговой функцией с целевым значением равным 0,5 (рис. 4).

Правильный исход

	0	1	2
0	0	0	0
1	1	1	1
2	0	1	0
3	1	0	1
4	1	1	0
5	0	0	1
6	0	1	1
7	1	0	0

Полученный исход

	0	1	2
0	0	0	0
1	1	1	1
2	0	1	0
3	1	0	1
4	1	0	0
5	0	1	1
6	0	1	1
7	1	0	0

Рисунок 4 – Сравнение исходных данных с полученными

Правильный исход

	0	1	2
0	0	0	0
1	1	1	1
2	0	1	0
3	1	0	1
4	1	1	0
5	0	0	1
6	0	1	1
7	1	0	0

Полученный исход

	0	1	2
0	0	0	0
1	1	1	1
2	0	1	0
3	1	0	1
4	1	1	0
5	0	0	1
6	0	1	1
7	1	0	0

Рисунок 5 – Результат работы сети без сжатия

Хорошо видно, что некоторые строки идентичны, а некоторые, наоборот, отличаются на одну или несколько ячеек, что подтверждает изначальное предположение о несжимаемости этих данных. При изменение порогового значения функции на другое, некоторые данные восстанавливают свое исходное положение, но в то же время другие это положение теряют.

Архивация получилась с потерями. Такой результат не удовлетворяет поставленным требованиям.

На самом деле, такой исход можно предсказать еще на этапе обучения нашего нейронного архиватора, так как результат ошибки в ходе нашего обучения не был близок к нулю, а равнялся 0,0833. Причем, при увеличении эпох и времени обучения, результат оставался прежним.

Если мы добавим всего один нейрон, к нынешнему архиватору, результат будет получен без потерь (рис. 5).

Как видно, результат полностью соответствует ожиданиям. Все значения верны, а ошибка обучения сети примерно равна 3,2783e-13, что очень близко к нулю, по сравнению с предыдущем результатом.

По-прежнему остаются поля для экспериментов, например, можно попробовать избавиться от закономерности во входных данных, увеличивая их размерность, а также поработать над темпами уменьшения размерности слоев нейронной сети по мере приближения к ее пику.

Список литературы:

1. Нейросетевое сжатие данных - [электронных ресурс] – Режим доступа: .https://habr.com/ru/post/126497/ свободный (дата обращения: 13.01.2022).