МЕТОДИКА ПЕРЕНОСА ВЕСОВ НЕЙРОННОЙ СЕТИ ИЗ ПРОГРАММНОЙ В АППАРАТНУЮ РЕАЛИЗАЦИЮ

БересневА. П.,Зоев И. В.
Научный руководитель: МальчуковА.Н.
Национальный исследовательский Томский политехнический университет
apb3@tpu.ru

Введение

В настоящее время довольно бурно развиваются исследования в сфере глубинного обучения нейронных сетей (НС), в частности сверточных нейронных сетей (СНС). В данной статье рассматривается СНС, распознающая объекты на изображениях. Существующие программные реализации имеют свой предел скорости работы, который упирается в количество необходимых вычислений [1]. Чтобы оптимизировать скорость работы СНС возможно создание аппаратной реализации на программируемых логических интегральных схемах (ПЛИС) [2]. Реализовать СНС на ПЛИС возможно, однако реализация обучения такой сети довольно сложна, а полученная конфигурация системы будет занимать большое количество логических ячеек. Проще использовать готовую программную реализацию СНС, обучить её, затем извлечь веса из модели и перенести на аппаратную реализацию. В итоге, программно обученная СНС будет работать на аппаратной реализации.

Выбор программной платформы для программной реализации

Существует достаточное количество фреймворков для создания программной СНС. Необходимо выбрать такой, в котором было бы удобно описать модель СНС, обучить её, а затем извлечь веса.

Согласно сравнительной характеристике из статей [3-4] фреймворки Caffe, H20, MXNet Deeplearning4j позволяют описать модель нейронной сети на высоком уровне абстракции. Другие, например, Tensorfolw, CNTK, Theano требуют изучения их программного интерфейса и более детального описания самой модели, что позволяет оптимизировать вычисления на программном уровне.

Для наших целей была выбрана программная платформа Caffe, так как она не только позволяет описать СНС на высоком уровне абстракции, но и предоставляет возможность для предварительной обработки входных данных, что немаловажно. А также позволяет сохранять веса.

Описание модели

Чтобы создать модель, используя Caffe, необходимо описать архитектуру нейронной сети в «prototxt» файлах. В этом файле описывается структура сети, представленная в виде слоев (layers) и их параметров, использующихся в модели и другие значения [5]. Для СНС характерно использование сверточных слоев, слоев пулинга, слои полносвязной сети нейронов, также слои с функциями активации и описание выходов нейронной сети.

Формально каждый слой имеет имя (name), тип (type), параметры top и bottom, в которых задаются имена верхнего и нижнего слоев соответственно. Также возможно указание дополнительных параметров, таких как размер ядра (kernel_size) для сверточных слоев или количество выходов (num_output) для полносвязных слоев.

Пример описания сверточного слоя представлен ниже:

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  convolution_param {
    num_output: 6
    kernel_size: 7
    stride: 1}
}
```

Также необходимо задать входные данные, которые для Caffe могут быть представлены в виде изображений (тип ImageData), HDF5 входного файла, или LevelDB файла.

Для того чтобы обучить созданную модель нейронной сети создается еще один «prototxt» файл, в котором задается «prototxt» файл описанием модели, количество итераций обучения, интервалы для тестирования нейронной сети, различные параметры, влияющие на скорость обучения и другие.

После обучения созданной нейронной сети на обучающей выборке Caffe создает «caffemodel» файл, в котором сохраняет настроенные веса. Этот файл и необходим для переноса весов с программной реализации в аппаратную.

Требуемое представление формата весов

В аппаратной реализации для более быстрой работы используются другое представление чисел. А именно, 14 разрядный вариант числа с плавающей точкой половинной точности. Структура такого формата представлена на рис.1.

Данный формат отличается от 16-ти разрядного числа тем, что два младших бита мантиссы числа отсекаются.

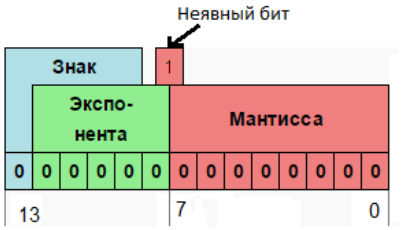

Рис.1. 14 разрядный формат числа с плавающей точкой половинной точности

Необходимо выполнить еще одно преобразование, так как аппаратная реализация разрабатывается в среде Modelsim фирмы Altera. Веса в аппаратной реализации хранятся в блоках памяти, которые инициализируются с использованием специальных файлов Hexadecimal File (*.mem). Числа в таких файлах хранятся в шестнадцатеричном формате. Пример внутреннего представления данного файла представлен ниже.

@0 26cc 0738 0957 0908 0851 2a5a 2a69 096f
@8 09b0 0766 0a9c 0a6e 29b5 2aa6 2835 088d
@10 07e4 0b1a 0b2d 095c 2a74 2986 09e4 278d
@18 0a84 0b81 0a68 28d8 29ec 0671 2a2b 274e
…

Методика переноса весов нейронной сети

До переноса весов, нужно привести к единому формату представления полученных весов и используемых в аппаратной реализации. В Caffe используется 32 битное представление чисел. Однако, нам необходимо 14 битное представление. Для этого необходимо перевести из 32-х разрядного представления в 16 разрядное, затем 16 разрядное перевести в 14 разрядное представление.

Сам процесс переноса весов можно описать в три этапа:

1. Получение HDF5 файла с весами из внутреннего представления фреймворка Caffe из файла caffemodel;
2. Приведение чисел из HDF5 [6] файла к требуемому формату;
3. Запись преобразованных чисел в требуемый формат .mem файла.

Для первого этапа используется скрипт из фрейморка Mocha [7], который компилируется вместе с Caffe.

На втором этапе из HDF5 файла веса необходимо преобразовать к виду, используемому в аппаратной реализации.

Второй и третий этап реализован в виде скрипта на языке Python. Данный скрипт берет на вход HDF5-файл с весами обученной нейронной сети, который был получен в результате первого этапа. Затем, происходит преобразование чисел, согласно описанию, приведенному выше.

Выходом являются файлы, в которых записываются числа в шестнадцатеричном формате, содержащие в себе обученные веса. Данные файлы являются совместимыми со средами разработки аппаратных реализаций, таких как Quartus и Modelsim.

Заключение

Данная работа является важным этапом создания аппаратных нейронных сетей, поскольку необученные нейронные сети не несут на себе никакой функциональной нагрузки.

Представленный метод не зависит от структуры СНС и может применяться для различных конфигураций модели нейронной сети.

Возможна небольшая модификация данной методики с целью загрузки разного рода данных в аппаратную реализацию. Например, для тестирования может появиться необходимость загрузить тестовые изображения, которые должны быть представлены в таком же формате данных.

В качестве дальнейших планов работы, процесс переноса весов может быть упрошен для пользователя посредством еще большей автоматизации через создание более сложных скриптов.

Список использованных источников

1. K. He and J. Sun. Convolutional neural networks at constrained time cost. arXiv:1412.1710, 2014.

2. Зоев И.В. Разработка вычислителя для плавающей точки для нейронных сетей / науч. рук. А.Н. Мальчуков // Информационные технологии в науке, управлении, социальной сфере и медицине: сборник научных трудов III Международной научной конференции, Томск 23-26 мая 2016 г. — Томск: Изд-во ТПУ, 2016. – Ч. 1. – С. 162-164.

3. Fox J., Zou Y., Qiu J. Software Frameworks for Deep Learning at Scale. Internal Indiana University Technical Report July 29 2016.

4. S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, “Comparative study of deep learning software frameworks,” arXiv preprint arXiv:1511.06435, 2015.

5. Layers//Caffe [Электронныйресурс] / Deep Learning framework by the BVLC. – URL: http://caffe.berkeleyvision.org/tutorial/layers.html (датаобращения 20.10.2016).

6. HDF5 Software Documentation. [Электронныйресурс] / The HDF5 Group – URL: https://support.hdfgroup.org/HDF5/ (датаобращения 20.10.2016).

7. Importing trained model form Caffe. [Электронный ресурс] / Mocha Documentation//Tools – URL: http://mochajl.readthedocs.io/en/latest/user-