ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ № 3(17), 2013, с.??—??

УДК 004.415

С.А. Лобов, Н.А. Сергиевский, А.А. Харламов

Адаптация алгоритма сверточных нейронных сетей на ПЛИС

Ключевые слова и фразы: ПЛИС, нейронные сети

Введение

Алгоритмы сверточных нейросетей [1] являются перспективными с точки зрения качества распознавания изображений. Наиболее надежным способом измерения качества систем машинного зрения является тестирование на больших базах, таких как VOC Pascal [2,3] и Image-net [4]. Тестирование на этих базах позволяет оценить способность алгоритма решать задачи классификации и детектирования объектов в реальных условиях. Алгоритмы на основе сверточных сетей уверенно лидируют уже несколько лет на этих базах [5, 6] и в решении подобных задача [7]. Круг применения их расширяется с каждым годом [8]. Однако реализация сверточных нейронных сетей обладает существенными требованиями по быстродействию. Для «стационарных» решений наиболее распространенным способом обеспечения такой производительности является использование для вычисления универсальных графических процессоров [5]. Однако в составе встраиваемых и мобильных решений следует обратить внимание на возможность использования программируемых логических интегральных схем (ПЛИС) [7,9].

ПЛИС прочно занимает нишу компонентов для штучных и мелкосерийных устройств, для которых с одной стороны требуется сложная логическая структура, а с другой стороны экономически не целесообразно разрабатывать и заказывать специализированную микросхему. В настоящий момент известны реализации сверточных сетей: устройства Tegra 4 произведены в 2013-2014 по технологии с проектными нормами 28 нм, процессоры Xilinx Zynq-7000 также выпущены в 2013м году по технологии с такими же проектными

нормами. Таким образом, реализация сверточных сетей в виде микро интегральных устройства в настоящее время весьма актуальна.

1. Структура сверточной нейронной сети

Главной задачей исследования было промоделировать прямое распространение сверточной нейронной сети (СНС). В сверточной сети последовательно применяются операции свертки, подвыборки (max-pooling) и нелинейной функции активации (ReLU [5] или logsig). Рассмотрим нейронную сеть представленную на Рис. 1. На вход данной сети подается цветное изображение размером 224x224x3 и оно сворачивается с 96 фильтрами размером 11x11x3 и шагом 4, в результате получается набор карт признаков размером 55x55x96. Этот набор карт признаков проходит через нелинейную функцию ReLU, далее осуществляется свертка с 256 фильтрами 5x5x48 (нейронная сеть разбита на 2 части) и применяется операция подвыборки (max-pooling) и нелинейность ReLU. И так далее, последний слой содержит 4096 признаков.

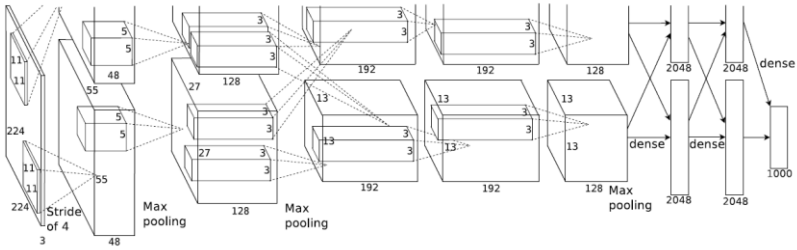

Рис. 1. Сверточная нейронная сеть для классификации изображений [5]

2. Реализация сверточной нейронной сети на программируемой логической интегральной схеме

Будем рассматривать входной поток данных в виде беззнаковых байт 0-255. ПЛИС содержит элементы ядра — 16-битные чис-

ла с фиксированной точкой и 12 битами дробной части. Это обусловлено следующими соображениями: реализация операций с плавающей точкой на ПЛИС требует существенно больших ресурсов, а в задаче сверточных сетей диапазон значений ядра является хорошо предсказуемым.

Одной из сложностей в реализации СНС на плис является недостаток памяти и умножителей на ПЛИС. То есть нельзя одновременно на одном кристалле осуществить свертку со всеми входными ядрами сети либо осуществить свертку всего кадра с входным окном.

Схема управления (см. Рис. 2) подключена с помощью шины AXI4-Lite. Загрузка данных из основной памяти и отправка результата осуществляется с помощью DMA через шину AXI4.

Реализуемые функции:

- Загрузка ядер
- Обработка
- Замена ядра
- Сброс.

Загрузка ядер представляет собою запись всех ядер (с использованием DMA) сразу в память ядер, размещенную на ПЛИС.

Рис. 2. Общая схема подключения модуля

Процесс обработки данных:

(1) Получение данных от источника
(2) Заполнение буфера FIFO для строк
(3) Вычисление свертки в каждой позиции
(4) Обновление и сохранение значения для операции подвыборки (maxpool)
(5) Сохранение итогового значения (карты признаков) в локальной памяти
(6) Сдвиг буфера строк

В качестве хранилища для строк изображения использовалось 11 (по размеру ядра) блоков памяти в режиме FIFO. Для хранения ядер используется еще один блок памяти (одно ядро занимает 244 байта). В качестве арифметического модуля используются блоки DSP48, они обеспечивают фиксированную задержку в 1 такт на операцию умножение и сложение (в режиме потоковой обработки) Рис. 3.

Процесс свертки осуществляется непосредственно с помощью циклического списка очередей FIFO 1 — FIFO n из блоков памяти и набора регистров. Таким образом, значение интенсивности каждого следующего пиксела изображения последовательно сдвигается через регистры последней строки, а затем записывается в очередь FIFO n одновременно с перемножением на каждом такте на коэффициент ядра. Это эквивалентно горизонтальному движению ядра свертки. Затем, по достижении конца строки, адреса счетчиков FIFO меняются, что эквивалентно сдвигу ядра свертки вниз на один пиксел.

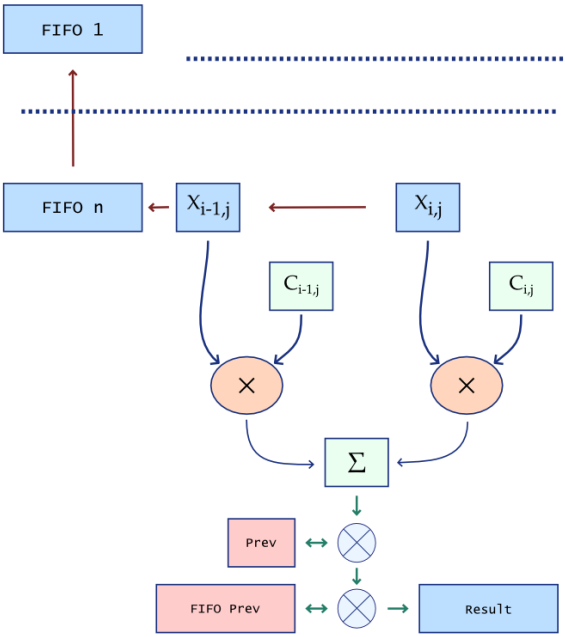

Рис. 3. Процесс обработки сигнала

Процесс свертки осуществляется непосредственно с помощью циклического списка очередей FIFO 1 — FIFO n из блоков памяти и набора регистров. Таким образом, значение интенсивности каждого следующего пиксела изображения последовательно сдвигается через регистры последней строки, а затем записывается в очередь FIFO n одновременно с перемножением на каждом такте на коэффициент ядра. Это эквивалентно горизонтальному движению ядра свертки. Затем, по достижении конца строки, адреса счетчиков FIFO меняются, что эквивалентно сдвигу ядра свертки вниз на один пиксел.

Для операции подвыборки результат свертки в нечетных столбцах нечетной строки записывается прямо в ячейку Prev предыдущего значения. В четных столбцах нечетной строки результат свертки сравнивается с ячейкой Prev и max записывается в блок памяти FIFO Prev. В следующей (четной) строке – последова-

тельность действий обратная, для нечетного столбца значение из памяти FIFO Prev читается и сравнивается со значением свертки в данной точке, затем записывается в Prev. Для четного столбца четной строки после сравнения с предыдущим значением максимальное значение (таким образом, являющееся подвыборкой по четырем ячейкам) записывается в память результата.

3. Технические подпрбности

Процессор 7Z020 серии Zynq-7000 является системой на кристалле – в одном корпусе содержится двухъядерный производительный процессор архитектуры ARM и ПЛИС. В состав ПЛИС входит 85 K программируемых ячеек, 140 модулей блочной памяти по 36 K каждый, и 220 модулей DSP.

В качестве тестовых стендов используется:

- Zynq-7000 SoC Video and Imaging Kit компании Xilinx
- Плата MicroZed с таким же процессором 7Z020 ядра.

Первое представляет собою готовый набор, для тестирования возможностей по обработки видео. Второе — миниатюрная плата 100x78 мм, допускающая мобильное использование.

Выводы

Алгоритмы на основе сверточных нейронных сетей получают все большую популярность в практических задачах. Эти алгоритмы могут сыграть большую роль во встраиваемых системах видеоаналитики, поэтому адаптация СНС на ПЛИС является перспективным направлением. В работе показано как можно реализовать прямое распространение сигнала первого слоя СНС на ПЛИС. При этом выработана эффективная (в плане использования ресурсов) стратегия, рассчитанная на специфику сверток СНС.

Литература

[1] LeCun Y. et al. Gradient-based learning applied to document recognition //Proceedings of the IEEE. – 1998. – Т. 86. – №. 11. – С. 2278-2324.

[2] Everingham M. et al. The pascal visual object classes (voc) challenge //International journal of computer vision. – 2010. – Т. 88. – №. 2. – С. 303-338.

[3] http://pascallin.ecs.soton.ac.uk/challenges/VOC/

[4] http://www.image-net.org/

[5] Krizhevsky A., Sutskever I., Hinton G. E. Imagenet classification with deep convolutional neural networks //Advances in neural information processing systems. – 2012. – С. 1097-1105.

[6] Girshick R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation //arXiv preprint arXiv:1311.2524. – 2013.

[7] Gokhale V. et al. A 240 G-ops/s Mobile Coprocessor for Deep Neural Networks //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. – 2014. – С. 682-687.

[8] Schmidhuber J. Deep Learning in Neural Networks: An Overview //arXiv preprint arXiv:1404.7828. – 2014.

[9] Pham P. H. et al. NeuFlow: dataflow vision processing system-on-a-chip //Circuits and Systems (MWSCAS), 2012 IEEE 55th International Midwest Symposium on. – IEEE, 2012. – С. 1044-1047