Классификация объектов в помещениях сверточными сетями на встраиваемом вычислителе

И.С. Фомин[1], Г.К. Якименко[2]

[1]Центральный научно-исследовательский и опытно-конструкторский институт робототехники и технической кибернетики (ЦНИИ РТК), Санкт-Петербург; email: i.fomin@rtc.ru

[2]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург; email: yakimenko.gk@edu.spbstu.ru

Введение

Одной из наиболее распространенных задач в робототехнике является проблема распознавания объектов в области, в которой находится робот. Ее решение актуально как для управления роботами на улице, так и внутри помещения. Определение окружающих робота объектов может быть полезно для вычисления его местоположения, опознания препятствий, семантического анализа окружающей обстановки, вычисления объектов интереса в окружении для движения к ним и т.д. В качестве робототехнического устройства при решении задачи может выступать мобильный робот, разработанный для разведки местности, сервисный или транспортный робот, перемещающийся по помещениям в соответствии заданной оператором целью или робот для радиационной разведки, перемещающийся как по открытой местности, так и в помещениях. При условии использования классификации объектов, наблюдаемых камерой, возможно оценить, например, тип помещения, в котором сейчас находится робот. Это позволит, в частности, давать оператору информацию о местоположении робота в семантической форме, и использовать для навигации не указание координат, а описание маршрута в нечеткой форме путем указания типа помещений, которые должны быть преодолены для достижения цели маршрута. Безусловно, при использования такого типа глобальной навигации предполагается, что локальная навигация с объездом статических и динамических препятствий уже реализована.

Под роботом в данной работе предполагается небольшой наземный робот для исследования помещений, небольшого размера (около 45x33x14 см). Это налагает ограничения на устройства, которые могут быть использованы для запуска нейронных сетей. В качестве такого устройства может выступать встраиваемый вычислитель типа Intel NUC, различные нейронные вычислители от компании Intel или специально настроенные ПЛИС, но рассмотрение всех возможных вариантов выходит за рамки этой работы.

Рассмотрена классификация объектов в помещениях с использованием нескольких типов нейронных сетей. В качестве встраиваемого устройства задействованы различные версии Nvidia Jetson для сравнения качества классификации и производительности различных сетей. По этой причине в качестве инструмента для обучения и тестирования сетей используется фреймворк Nvidia-TLT. Он предоставляет возможность обучения широкого спектра архитектур нейронных сетей, как для классификации, так и для других задач. Кроме того, он разработан специально для обучения сетей для использования в будущем на Nvidia Jetson и предоставляет алгоритмы оптимизации сетей под конкретное устройство. Для обучения и тестирования использован гибридный набор данных, собранный из открытых наборов данных для классификации объектов в помещениях, изображений, свободно распространяемых в интернете и изображений, собранных непосредственно авторами. В работе будут сформулированы рекомендации по применению различных нейросетевых архитектур для того или иного устройства.

Во второй части работы будут рассмотрены использованные нейросетевые алгоритмы классификации объектов, а также использованные встраиваемые устройства; третья часть посвящена краткому описанию последовательности операций при обучении сетей с использованием Nvidia-TLT; четвертая часть посвящена описанию набора данных, полученных результатов и формированию рекомендаций по применению различных нейронных сетей на различных встраиваемых устройствах.

Обзор использованных сетей и устройств

На сегодняшний день использование нейронных сетей различных архитектур для классификации объектов на изображениях является основным способом в подавляющем большинстве случаев. Первые успешные опыты классификации были показаны в работах, посвященных сетям LeNet [1] и AlexNet [2], которые утвердили превосходство сверточных нейронных сетей над полносвязными нейронными сетями и классическими алгоритмами СТЗ в задаче классификации изображений. В дальнейшем взрывной рост числа работ на эту тематику породил группу архитектур, некоторые из которых успешно применяются и по сей день. Одной из таких архитектур стала VGG [3], названная по аббревиатуре группы исследователей Visual Geometry Group, в том числе она будет рассмотрена в дальнейшем в экспериментах по классификации. Два основных варианта, известных как VGG-16 (VGG-M в авторском названии) и VGG-19 (VGG-L) отличаются между собой количеством и размером слоев, в первой всего 16 слоев, а во второй, соответственно, 19 и увеличено количество карт признаков даже в тех слоях, которые присутствуют также и в VGG-16.

После этапа движения по пути увеличения качества за счет количества слоев (глубины сети) исследователи достигли плато, и перешли к ново-

му подходу. Теперь появлялись дополнительные соединения, за счет которых сигнал мог обходить часть весов, и сети собирались из определенного вида блоков. Так появилась сначала GoogLeNet (или InceptionV1) [4], а когда было показано, что использование таких блоков позволяет увеличивать глубину сети и качество намного сильнее, чем раньше, в новых версиях InceptionV2 и InceptionV3 [5] эти идеи получили свое развитие.

Еще одна архитектура, которая шла тем же путем, но в своем направлении была сеть ResNet [6], которая также получила несколько версий, и в том числе была скрещена с архитектурой Inception. Несколько версий этой сети, используемые в данной работе, описаны в статье [7] и за подробным описанием отличий между сетями следует обратиться непосредственно к ней. Она также получила развитие в виде архитектуры ResNext, также после этого было создано еще несколько интересных архитектур, но так как они не представлены в составе Nvidia-TLT или не были исследованы, их рассмотрение выходит за рамки этого обзора.

Рассмотрим также встраиваемые устройства, производимые компанией Nvidia для решения различных задач машинного обучения, технического зрения и нейронных сетей. Jetson – это линейка встраиваемых модулей, работающих на базе GPU от Nvidia. Jetson является ведущей встраиваемой платформой для решения задач обработки изображений, видео и искусственного интеллекта. На данный момент линейка Jetson представлена 4-мя моделями устройств: TX1, TX2, Nano и Xavier. TX1 и TX2 – сравнительно дешевые и небольшие модули, появившиеся на рынке встраиваемых устройств уже несколько лет назад, но до сих пор они позволяют с приемлемой скоростью применять нейронные сети для классификации и даже некоторые детекторы. Jetson Nano – относительно недавно появившаяся модель, она позиционируется как устройство, предназначенное для обработки информации сразу, непосредственно рядом с ее источником после получения и оцифровки, единственный истинно рассчитанный на встраивание чип; несмотря на малый размер и небольшое энергопотребление подходит как для запуска алгоритмов обработки (предобработки) видео, так и для запуска несложных нейронных сетей, оптимизированных в TLT. Модель Jetson Xavier – наиболее мощный встраиваемый модуль, который имеет сравнительно скромные размеры и ограниченное энергопотребление; он подходит для запуска широкого спектра нейронных сетей, вплоть до тяжеловесных детекторов и сегментаторов, впрочем, ниже будет показана степень эффективности данного устройства при решении задачи классификации. Наконец, для проверки работоспособности обученного решения также проводится тестирование на том же устройстве, на котором производится обучение (видеокарта GTX 1660 Super)

Обучение сетей в Nvidia-TLT

Рассмотрим основной порядок обучения нейронных сетей в фреймворке Nvidia-TLT. Перед началом работы пользователь получает в личном кабинете токен для разрешения запуска расчетов, устанавливает и настраивает все необходимые библиотеки, docker-образ Nvidia-TLT с поддержкой видеокарты основной системы изнутри образа. Далее выбирается подходящий задаче скрипт для обучения (классификатор, детектор и пр.), архитектура нейронной сети и предобученные веса из списка доступных (программирование осуществляется в Jupyter Notebook, а выбор происходит в текстовом формате). После выбора предварительно обученной модели и подготовки файлов спецификации для настройки гиперпараметров обучения, наступает рабочий этап обучения нейронной сети. На первом шаге рабочего процесса предварительно обученная модель обучается с использованием новых данных, отобранных для решения какой-то конкретной задачи, согласно файлу спецификации.

Во время обучения набор данных может быть искусственно расширен с целью повышения вариативности и получения более точной выходной рабочей модели нейронной сети. TLT содержит следующие инструменты расширения начального набора данных:

- масштабирование изображения,
- увеличение цветового пространства изображения,
- размытие изображения,
- поворот, сдвиг и отражение изображения,
- изменение оттенка, яркости и контраста изображения.

После первого обучения модели и проверки её работоспособности, наступает следующий шаг рабочего процесса – обрезка (pruning) модели. Его суть состоит в том, чтобы найти и удалить из модели узлы, не оказывающие существенного влияния на качество работы модели. Этот шаг позволяет значительно повысить производительность нейронной сети, уменьшая затраты на использование памяти вычислительного устройства и увеличивая пропускную способность нейронной сети. После этого в общем случае выполняется преобразование сети таким образом, чтобы при ее вычислении на конечном устройстве все операции выполнялись не в числах с плавающей точкой, а в пространстве чисел INT8. Это позволяет заметно увеличить частоту кадров на тех устройствах, для которых это возможно. Для тех устройств, для которых нет поддержки расчетов в INT8, данный этап опускается.

После экспорта модели на целевое устройство происходит ее сборка с помощью инструментов библиотеки TensorRT [8], а также дальнейшее развертывание и эксплуатация с помощью набора инструментов DeepStream.

Набор данных

За основу набора данных для классификации был взят набор MS COCO [9], из которого были

извлечены изображения тех объектов, которые имеют отношение к помещениям. Из списка полученных изображений были отобраны только те, которые действительно относятся к помещениям и не включают улицу по большей части. Размер некоторых изображений был частично сокращен, так, чтобы видна была только часть комнаты с ограниченным количеством объектов. Кроме того, набор данных был дополнен изображениями, собранными вручную самостоятельно и отсортированных соответственно выбранным классам. Общие характеристики собранного набора представлены в сводной таблице 1. Набор данных для классификации был размечен согласно формату «KITTI format dataset» [10].

Таблица 1

Параметры набора данных		
Класс	Обучающий набор	Тестовый набор
Человек	2860	818
Тв-монитор	783	115
Стул	402	224
Стол	376	108
Растение	368	106
Диван	354	102

Эксперименты

Как было сказано ранее, для обучения был использован пакет Nvidia-TLT. Во-первых, требовалось обучить сети на собранном наборе данных, для чего он был специальным образом подготовлен. Далее в таблице 2 будет приведена средняя нормированная матрица ошибок для нескольких из исследованных сетей.

Таблица 2

	Стул	Стол	Человек	Растение	Диван	Тв-монитор
Стул	0,76	0,00	0,12	0,00	0,08	0,04
Стол	0,52	0,08	0,32	0,08	0,00	0,00
Человек	0,03	0,02	0,92	0,01	0,01	0,01
Растение	0,05	0,03	0,10	0,82	0,00	0,00
Диван	0,20	0,00	0,00	0,04	0,76	0,00
Тв-монитор	0,03	0,00	0,03	0,06	0,00	0,88

Из приведенных данных можно сделать вывод о том, что изображения, которые использовались как в процессе обучения, так и в процессе тестирования имеют ряд особенностей, которые частично влияют на качественные показатели. Изображения в наборе MS COCO содержат зачастую больше одного объекта из классов, которым обучена нейронная сеть, поэтому, как показывает матрица ошибок, часто происходит путаница между классами. В таблице 3 показаны показатели качества работы сети при расчете качества по top-1 классу (т.е. учитывается только класс с наибольшей вероятностью). Там же показаны показатели качества для расчета по top-3, когда классификация считается верной, если хотя бы один из трех первых по вероятности классов совпадает с размеченным классом. Эти показатели существенно выше.

Таблица 3

	ResNet10	ResNet50	VGG19
Топ-1	0,93	0,92	0,92
Топ-3	0,95	0,97	0,95

В процессе влияния обрезки (pruning) избыточных связей выяснено, что некоторые сети существенно теряют в качестве в результате обрезки, некоторые качество сохраняют. В таблице 4 приведено сравнение влияния обрезки на различные архитектуры. Исходя из этих данных можно сделать вывод о том, что архитектуры ResNet18, VGG16 неразумно подвергать данной процедуре.

Таблица 4

	До обрезки		После обрезки	
	ResNet18	VGG16	ResNet18	VGG16
Топ-3	0,92	0,89	0,94	0,63
Топ-5	1	0,97	0,97	0,93

Также, чтобы продемонстрировать, что удалось успешно обучить нейронную сеть для классификации в рамках нового фреймворка, и для проверки применимости в своих задачах по итогам экспериментов был собран собственный дополнительный тестовый датасет. Его отличительной особенностью является то, что каждый объект находится не в окружении других: одно изображение – один класс объекта. Всего по 100 изображений на каждый из классов, представленных в таблице 1. Результаты тестирования на этом наборе данных приведены в таблице 5. По этим данным видно, что вне условий сложных изображений качество классификатора позволяет использовать его в задачах классификации объектов в помещениях.

Таблица 5

	Стандартный набор данных		Собственный набор данных	
	ResNet10	ResNet18	ResNet10	ResNet18
Топ-3	0,91	0,91	0,93	0,92
Топ-5	0,99	0,99	0,99	1

Кроме того, одной из задач было исследование быстродействия работы различных сетей на разных устройствах линейки Jetson. Сводные результаты тестов быстродействия приведены в таблице 6.

Таблица 6

	Nano	TX2	Xavier
ResNet10	76	199	1548
ResNet18	48	122	755
ResNet50	-	35	402
VGG16	-	18	248
VGG19	-	7	176

По итогам исследования можно сделать вывод, что сети ResNet10 и ResNet18 отлично показывают себя на маломощных устройствах Jetson при достойном показателе качества, а архитектуры ResNet50, VGG16, VGG19 требуют существенную вычислительную мощность и требуют для своего запуска устройство типа Xavier, или даже персональный компьютер с графическим вычислителем.

Заключение

По итогам исследования использования платформы Nvidia-TLT для обучения и поставки нейронных сетей для классификации объектов были сделаны следующие выводы. Данный инструмент позволяет обучать классификаторы для использования на встраиваемых устройства. Качество обученного таким образом классификатора в общем случае не уступает другим способам. После обрезки избыточных связей и дообучения отдельные сети теряют в качестве. Быстродействие оптимизированных сетей превышает быстродействие сетей с той же архитектурой, обученных и запущенных на устройствах Jetson через стороннее ПО (OpenCV). Сформулированы рекомендации по использованию различных архитектур сетей в зависимости от модели устройства Jetson.

В дальнейшем планируется расширить номенклатуру исследуемых сетей на группу сетей для обнаружения объектов, а также проверить полученные результаты на нескольких разных датасетах, чтобы убедиться в независимости сделанных выводов от используемого набора данных.

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. *Krizhevsky A., Sutskever I., Hinton G. E.* Imagenet classification with deep convolutional neural networks //Advances in neural information processing systems. 2012.Т. 25. С. 1097-1105.
3. *Simonyan K., Zisserman A.* Very deep convolutional networks for large-scale image recognition //arXiv preprint arXiv:1409.1556. 2014.
4. *Szegedy C. et al*. Going deeper with convolutions //Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. С. 1-9.
5. *Szegedy C. et al.* Rethinking the inception architecture for computer vision //Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. С. 2818-2826.
6. *He K. et al.* Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. С. 770-778.
7. *Szegedy C. et al.* Inception-v4, inception-resnet and the impact of residual connections on learning //Proceedings of the AAAI Conference on Artificial Intelligence. 2017. Т. 31. №. 1.
8. *Settle S. O. et al.* Quantizing convolutional neural networks for low-power high-throughput inference engines //arXiv preprint arXiv:1805.07941. 2018.
9. *Lin T. Y. et al*. Microsoft coco: Common objects in context //European conference on computer vision. Springer, Cham, 2014. С. 740-755.
10. *Geiger A. et al*. Vision meets robotics: The kitti dataset //The International Journal of Robotics Research. 2013. Т. 32. №. 11. С. 1231-1237.