ЧИСЛЕННЫЕ МЕТОДЫ И АНАЛИЗ ДАННЫХ

Детектирование книг на книжных полках при помощи глубоких нейронных сетей

М.О. Калинина [1], П.Л. Николаев [1]
[1] Московский авиационный институт (национальный исследовательский университет), 121552, Россия, г. Москва, ул. Оршанская, д. 3

Аннотация

Глубокие нейронные сети в настоящее время получили широкое распространение в различных сферах деятельности человека, в том числе тех, где требуется работа с большим объемом данных, а также операции по получению и обработке информации из окружающего мира. В данной статье рассмотрено создание сверточной нейронной сети на основе архитектуры YOLO по детектированию книг в режиме реального времени. Описаны процесс создания собственного набора данных и обучение на нем глубокой нейронной сети. Приведена структура полученной нейронной сети, и рассмотрены наиболее часто используемые метрики для оценки качества ее работы. Также сделан краткий обзор существующих видов архитектур нейронных сетей. Выбранная в качестве основы для нейросети архитектура обладает рядом преимуществ, позволяющих ей в значительной мере конкурировать с другими моделями нейросетей и делающих ее наиболее подходящим вариантом для создания сети, нацеленной на детектирование объектов, так как при ее разработке были значительно снивелированы некоторые часто встречающиеся недостатки подобных сетей (проблемы с распознаванием схожих по оформлению, имеющих одинаковый цвет обложек или расположенных под наклоном книг). Результаты, полученные в ходе обучения глубокой нейронной сети, позволяют использовать ее в качестве основы для дальнейшей разработки приложения, целью которого будет являться детектирование книг по книжным корешкам.

Ключевые слова: распознавание изображений; детектирование объектов; компьютерное зрение; машинное обучение; искусственные нейронные сети; глубокое обучение; сверточные нейронные сети.

Цитирование: **Калинина, М.О.** Детектирование книг на книжных полках при помощи глубоких нейронных сетей / М.О.Калинина, П.Л. Николаев // Компьютерная оптика. – 2020. – Т. 44, № 6. – С. 968-977. – DOI: 10.18287/2412-6179-CO-731.

Citation: Kalinina MO, Nikolaev PL. Book spine recognition with the use of deep neural networks. Computer Optics 2020; 44(6): 968-977. DOI: 10.18287/2412-6179-CO-731.

Введение

В настоящее время нейронные сети активно используются при решении различных проблем, в том числе связанных с детектированием объектов и распознаванием текста. Решение этих проблем может в значительной степени поспособствовать эффективной работе в некоторых сферах, например, тех, для которых характерно большое количество манипуляций с крупными объемами книг. Поиск нужных книг, основанный на использовании нейронной сети, способной осуществлять локализацию книжных корешков и последующее распознавание текста на них, с большой долей вероятности был бы более эффективным и быстрым, чем если бы он выполнялся вручную.

Существует множество вариантов использования нейросетей, сочетающих в себе локализацию книжных корешков и распознавание текста на них, например:

– Инвентаризационно-поисковые системы в библиотеках, способные облегчить задачи поиска требуемых экземпляров и их инвентаризации.

– Смарт-очки, которые можно использовать в библиотеках, книжных магазинах и на складах для подсчета количества книг и быстрого нахождения нужных экземпляров.

– Мобильное приложение, с помощью которого при осмотре книг в книжных магазинах и библиотеках возможно быстро определить названия книг на книжных полках, а также сопутствующие ему сервисы по поиску таких же книг в других местах и по меньшей цене.

В рамках работы рассматривается разработка нейронной сети для детектирования книг по книжным корешкам. Данная задача относится к задачам компьютерного зрения. При решении задач по распознаванию изображений наилучшие результаты достигаются с применением сверточных нейронных сетей.

Разрабатываемая нейронная сеть будет являться частью программы по определению и распознаванию названий книг на книжных полках. Детектирование книг предполагается выполнять в режиме реального

времени. При наведении камеры на объекты приложение должно сразу же детектировать книги по их корешкам и в случае необходимости предоставлять интересующую пользователя информацию о них. В связи с этим необходимо построить и обучить нейронную сеть, способную распознавать достаточное количество кадров в секунду и при этом давать достаточно точные результаты.

Нейросеть должна анализировать входное изображение на предмет наличия на нем книг и обеспечивать пользователя информацией о детектируемых объектах. Иными словами, на выходе на поданном изображении должно быть отмечено местоположение книг с помощью ограничивающего прямоугольника («рамки» вокруг корешка обнаруженной книги), а также производиться считывание текста, расположенного на книжном корешке, который затем может подаваться в поисковую систему или заноситься в базу данных.

В данной статье рассматривается первый этап разработки вышеописанной программы, представляющий собой разработку глубокой нейронной сети для детектирования книг по книжным корешкам.

В рамках данной работы основными задачами являлись:

– сбор данных (изображений книг, у которых явно заметны торцевые части переплета (корешки)), предназначенных для обучения глубокой нейронной сети;

– дальнейшая подготовка собранных данных к процессу разметки;

– разметка данных (выявление необходимых объектов и определение координат «рамок») и их редактирование с целью последующего использования в процессе обучения нейросети;

– создание модели нейронной сети для детектирования книг и ее дальнейшее обучение.

Обзор существующих работ

Идея распознавания книг по их корешкам послужила темой для значительного количества исследований. Следует отметить, что многие из разработанных в результате них систем были направлены не только на простое сегментирование либо детектирование книг, но также на распознавание текста, помещенного на книжные корешки. Тем не менее, какого-либо универсального подхода к этой задаче выработано не было. Так, [1] предлагают двухмодульную систему, в основе которой лежит высокочастотная фильтрация, после прохождения которой на изображении выделяются участки с высокой частотой, т.е. с рядами книг на полках. Методика сегментации отдельных книг основана на алгоритме M-estimator SAmple and Consensus (MSAC), с чьей помощью производится вычисление доминантных исчезающих точек – Dominant Vanishing Points (DVP). Как показали результаты, система не всегда в состоянии правильно определить каждый ряд книг. В [2] предложена клиент-серверная система, распознающая текст на книжных корешках и использующая его в качестве ключевых слов для поиска соответствующей информации в базе данных. Текст детектируется с помощью метода максимально стабильных экстремальных областей – Maximally Stable Extremal Regions (MSER) и библиотеки Standard Widget Toolkit (SWT), далее он считывается с помощью оптического распознавания символов – Optical Character Recognition (OCR). Система достаточно сложна в реализации, к тому же она не дает удовлетворительных результатов при низком разрешении. В [3, 4] рассматривается создание клиент-серверной системы дополненной реальности на базе метода оценки параметров модели на основе случайных выборок – RANdom SAmple Consensus (RANSAC), функционирующей на смартфонах. Система работала без нажатия кнопок и выдавала пользователю сведения о цене издания, оценках читателей, а также предоставляла дополнительную функцию в виде просмотра обложки. Для распознавания текста применяли систему оптического распознавания Tesseract OCR, которая является свободным программным обеспечением и доступна для использования под многие языки программирования в качестве специальной библиотеки. Отличительной особенностью рассматриваемой системы является то, что ее функционирование не связано с глубокими нейронными сетями – в основе лежит сравнение изображений искомых книжных корешков с изображениями, содержащимися в базе данных. Иными словами, книга определяется по принципу похожести двух изображений, что может значительно затруднить распознавание книги, если ее изображение не занесено в базу данных. В [5] рассматривается система для инвентаризации книг в библиотеках (фотография книжной полки сравнивается с фотографиями полок, занесенных в базу данных, затем определенной книге присваивается статус опознанной, перемещенной либо потерянной) и поиска потерянных книг. Для упрощения поиска применяется квантизация обрабатываемых изображений. Но так как книжные корешки опознаются в основном по цвету, то присутствуют проблемы с книгами, имеющими одинаковые цвета или схожее оформление (например, периодические издания). [6] предлагают систему, состоящую из модуля для сегментирования книжных корешков (границы корешков выявляются с помощью алгоритма Line Segment Detector (LSD)) и модуля для распознавания текста на основе Tesseract OCR. Для системы характерно появление проблем, связанных с бинаризацией изображения – из-за теней, курсивного текста, дополнительных графических символов и т.д. Система, предложенная [7], также состоит из 2 модулей: первый отвечает за исключение цифрового шума и детектирует книжные полки с помощью оператора Кэнни, второй детектирует конкретные отдельные

книги, для чего используется канонический корреляционный анализ – Canonical Correlation Analysis (CCA). Несомненным плюсом системы является то, что она способна учитывать, что книги могут быть расположены под углом, а не только строго вертикально. Тем не менее, программа требует доработки в области определения книг и объектов, не относящихся к книгам. Система [8] несколько проще – она реализует лишь сегментацию книжных корешков без распознавания текста на основе фреймворка Active Contour Model (snakes). Для поиска полок с книгами на изображении используется RANSAC, для сегментирования корешков – алгоритм Maximum Weight Independent Set (MWIS). В общем система показала себя достаточно хорошо, однако, как и многие другие, не всегда справлялась с корешками, расположенными под углом. Кроме того, сегментирование могло затрудняться в тех случаях, когда корешок присутствовал в кадре не полностью либо был перекрыт другим объектом. Приложение для инвентаризации книг на полках, рассмотренное в [9], сегментирует книги и считывает текст с книжных корешков. Приложение состоит одновременно из 2 нейронных сетей – сверточной и реккурентной. Распознавание текста в данном случае основано на последовательной маркировке. Эта система также столкнулась с проблемой корректного распознавания текста в случае, когда заголовки были одинаковыми или похожими. Также затруднялось распознавание похожих по форме и оформлению книг. [10] предлагают систему для детектирования текста на книжных корешках с помощью разработанной ими сети Rotational Region CNN (R2CNN), основанной на Faster R-CNN [11] и обученной на датасете International Conference on Document Analysis and Recognition (ICDAR) 2015 [12]. Из-за недостаточного количества данных для обучения ограничивающие прямоугольники не являются особенно точными. Эту проблему решали не с помощью переобучения, а с помощью добавления дополнительных методов для повышения точности вычисления.

Таким образом, мы наблюдаем значительное количество разнообразных подходов к обозначенной задаче / проблеме, но универсальной системы, предназначенной для ее решения, получено не было. Многие из них достаточно сложны для непосредственной реализации, другие не учитывают необходимость считывания и распознавания текста на корешках книг. Очень многие системы оптимально могут работать только с горизонтально расположенными книгами, так как не учитывают возможность размещения книг под углом. Кроме того, во многих случаях распознавание объектов может значительно затягиваться по времени вследствие того, что детектирование производится по сделанному фото, а не в режиме видеопотока. Стоит также отметить, что лежащие в основе многих систем не нейросетевые алгоритмы значительно уступают по точности нейронным сетям. Нашей итоговой целью является создание системы детектирования книг по книжным корешкам и распознавания текста на них, максимально учитывающей недостатки рассмотренных приложений.

Сверточные сети

Изначально сверточные нейронные сети были предложены для решения задач по классификации изображений. Типовая архитектура подобной сверточной сети выглядит следующим образом: классифицируемое изображение проходит через чередующиеся слои свертки и пулинга, в результате чего формируются карты признаков, далее они поступают в блок классификации, состоящий из полносвязных слоев. В блоке классификации происходит определение класса, к которому принадлежит изображение. В дальнейшем было предложено использовать сверточные сети для решения задач по детектированию и сегментации объектов на изображении.

Существующие сети для детектирования объектов на изображении можно разделить на 2 типа согласно технике, которой они придерживаются в процессе анализирования входного изображения:

– one-shot detectors (YOLO [13 – 15], SSD [16], RetinaNet [17], DetectNet [18]);

– two-shot detectors (R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN [11, 19–21]).

Сетям, относящимся к типу one-shot detectors, который характеризуется значительной быстротой, но меньшей точностью полученных результатов, для определения объекта на изображении достаточно проанализировать это изображение всего лишь один раз. Именно поэтому эти сети отличаются достаточно высокой скоростью работы.

Общий принцип работы этих сетей заключается в следующем: изображение проходит через несколько слоев свертки, в результате чего происходит формирование слоя, содержащего $m \times n$ локаций (где m и n – количество локаций по горизонтали и вертикали соответственно). Для каждой из этих локаций определяется заданное число ограничивающих прямоугольников, отличающихся по размеру и расположению; для каждого ограничивающего прямоугольника, в свою очередь, вычисляется принадлежность объекта к классу [22–23].

В основе же сетей типа two-shot detectors, более медленных, но показывающих при этом более точные результаты, лежит несколько иной принцип работы. Решение поставленной задачи разбивается на два этапа – определение регионов и их дальнейшую обработку. Суть работы сводится к тому, что детектирование объектов происходит по регионам. На входном изображении выделяются т.н. регионы-кандидаты, на которых предположительно должны присутствовать необходимые объекты. В сети R-CNN данные регионы формировались с помощью выборочного поиска (Selective Search) и классификация

происходила с помощью метода опорных векторов – support vector machine (SVM) [11]. Позднее в сети Fast R-CNN были предложены некоторые модификации, позволившие ускорить сеть, в частности, на ее вход стали подавать сразу все изображения, а не поочередно регионы-кандидаты, которые затем уже накладывались на полученную карту признаков. В то же время в Fast R-CNN классификация все еще происходила аналогично классификации в R-CNN, что замедляло работу сети.

В сети Faster R-CNN регионы начали вычисляться уже по картам признаков. Вместо выборочного поиска, был предложен заменивший его алгоритм детектирования объектов – object detection algorithm. После того, как регионы-кандидаты прошли обработку с помощью специального пулингового слоя – Region of interest (RoI) pooling layer – происходит их дальнейшая классификация, выполняемая этим же слоем [11].

Сети семейства R-CNN, хотя и отличаются достаточно высоким уровнем точности, не позволяют распознавать значительное количество кадров в секунду, что требуется в нашей задаче, поэтому наилучшим решением будет использовать сети, относящиеся к типу one-shot detectors. Наиболее предпочтительным вариантом в качестве основы нашей сети для детектирования книг по корешкам на книжных полках из данных сетей является сеть с архитектурой YOLO.

Точность полученных результатов при использовании последней версии YOLO находится на одном уровне с точностью результатов в случае использования большинства других моделей, однако первая модель значительно быстрее (например, архитектура SSD медленнее YOLOv3 примерно в 3 раза). Сети же, основанные на архитектуре R-CNN и ее улучшенной версии Fast R-CNN, YOLO и вовсе оставляет далеко позади благодаря отказу от использования регионов – таким образом, пока на вход других сетей подается лишь часть изображения, для которого необходимо сделать предсказание, YOLO получает полное изображение, которое делит на ячейки, предсказывая для каждой из них вероятность присутствия в ней искомых объектов. Такое преимущество модели дает использование единой сети свертки – как следует из названия данной архитектуры, свертка для каждого входного изображения производится лишь единожды.

Таким образом, при внимательном рассмотрении YOLO в значительной мере способна составить конкуренцию другим типам архитектур сетей.

Сети на основе архитектуры YOLO

YOLO (You Only Look Once) представляет собой модель, нацеленную на обнаружение или детектирование предметов (object detection) в реальном времени. На данный момент последней версией этой модели является YOLOv3, созданная на базе предыдущей версии – YOLOv2.

В сетях модели YOLO входное изображение делится на ячейки посредством наложения на него сетки определенного размера. Для каждой формирующейся при этом ячейки выполняется предсказание массива чисел, в состав которого входят координаты ограничивающего прямоугольника (верхняя левая и правая нижняя точки) или координаты центра прямоугольника с его шириной и высотой, а также вероятность класса, к которому относится обнаруженный на изображении объект.

Как уже говорилось ранее, в сетях архитектуры YOLO все эти манипуляции выполняются при помощи единой сверточной сети [24]. После того, как сеть получила входное изображение и выполнила необходимые операции по разбивке его на ячейки, применяется алгоритм подавления немаксимумов, который способствует отсеиванию излишних предсказаний – non-max suppression [25]. Входными данными для алгоритма служит список, включающий значения предсказанных ограничивающих прямоугольников, вероятностей присутствия искомого объекта на изображении и порогов перекрытия, на выходе же формируется список из предсказаний, успешно прошедших своего рода фильтрацию. Ее алгоритм представляет собой следующий набор действий: в предварительно созданный пустой список заносится значение предсказания с максимальным значением вероятности присутствия объекта на изображении, вычисляется значение IOU для данного и всех остальных предсказаний. Предсказанные ограничивающие прямоугольники, для которых значения IOU превышают заданный порог перекрытия, удаляются. В общем смысле, весь процесс фильтрации сводится к использованию единственного порогового значения, которое напрямую влияет на эффективность выполняемых действий.

В последнем сверточном слое получаются координаты рамки (или координаты центра с шириной и высотой).

Согласно [13], сеть YOLO обладает рядом преимуществ по сравнению с другими сетями для детектирования:

– работает с очень высокой скоростью, позволяя распознавать до 35 кадров в секунду (согласно [24], при тестировании сети на наборе данных COCO скорость работы последней версии YOLO составила 35 fps);

– работает сразу с целым изображением, а не с его отдельными частями, и меньше ошибается при отделении искомых объектов от фона в отличие от сетей типа R-CNN;

– изучает обобщаемые представления объектов, что позволяет сети лучше работать на новых данных.

Особенность представления данных на выходе является непосредственным ключом к понимаю того, что представляет из себя сеть YOLO. Изображение, подающееся на вход нейросети, делится на $S \times S$ ячеек. Для каждого изображенного на рисунке объекта существует ячейка, ответственная за его предсказание, – это ячейка, в которую попадает центр объекта.

В ячейках осуществляется предсказание B ограничивающих прямоугольников и C вероятностных оценок классов. При предсказании ограничивающих прямоугольников рассматриваются 5 компонентов:

– x и y, представляющие собой координаты центра прямоугольника; при этом осуществляется нормализация координат, вследствие чего их значения оказываются в интервале [0,1];

– w и h, соответственно представляющие собой ширину и высоту ограничивающего прямоугольника; их значения нормализуются аналогично значениям x и y;

– степень уверенности, которая отображает вероятность присутствия либо отсутствия объекта определенного класса на изображении.

Количество значений, получаемых на выходе, таким образом, равно $S\times S\times B\times 5$.

Вероятностная оценка класса осуществляется для каждой ячейки лишь один раз, независимо от количества ограничивающих прямоугольников B, и вычисляется $S\times S\times C$ раз.

В результате на выходе нейросети мы получаем $S\times S\times(B\times 5+C)$-й тензор.

С целью повышения стабилизации и производительности работы нейронных сетей может применяться метод, в основе которого лежит предварительная обработка данных, которыми в дальнейшем будет оперировать сеть, до состояния, при котором набор значений, поданных на вход нейросети, характеризуется нулевым математическим распределением, а дисперсия равна 0. Данный метод известен как пакетная нормализация (batch-normalization) [26]; чаще всего он заключается в нормализации входных слоев при помощи масштабирования набора данных.

В качестве функции активации часто используется Leaky ReLU [27] (также называемая «ReLU с «утечкой»), которая представляет собой модифицированную версию обычной функции ReLU. В результате множественных экспериментов было установлено, что данная модификация позволяет успешно применять функцию активации в тех случаях, когда обычные ReLU рискуют выйти из строя, например, при слишком высокой скорости обучения нейронной сети.

Создание сети для детектирования книг

Наша сеть, основанная на архитектуре YOLOv3, обучалась с нуля, без использования готовых весов. Обучение нейросети проходило на созданном нами наборе данных, основой которого являются 500 изображений, каждому из которых соответствует один из файлов разметки в формате XML, включающий в себя информацию о присутствующих на изображении объектах искомого класса: метки класса, координат ограничивающего прямоугольника для каждого объекта, а также названия файла, высоты и ширины изображения, на котором производится детектирование. Координатами ограничивающего прямоугольника считаются координаты x и y верхнего левого и нижнего правого углов прямоугольника, стороны которого расположены параллельно соответствующим осям координат и в который можно вписать искомый объект.

Ниже можно увидеть пример файла разметки.

```
<?xml version="1.0"?>
<annotation verified="no">
  <folder>Book</folder>
  <filename>Bookshelf_0022</filename>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>600</width>
    <height>600</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>Book</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>17</xmin>
      <ymin>103</ymin>
      <xmax>493</xmax>
      <ymax>193</ymax>
    </bndbox>
  </object>
  <object>
    <name>Book</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>8</xmin>
      <ymin>189</ymin>
      <xmax>530</xmax>
      <ymax>302</ymax>
    </bndbox>
  </object>
</annotation>
```

Разметка изображений производилась в форме, аналогичной форме разметки в наборе данных PascalVOC [28]. Данный формат разметки очень популярен из-за удобства представления данных и их последующего использования и широко применяется при решении широкого спектра задач – классификации, сегментации и детектирования объектов. При разметке для каждого изображения создается XML-файл, в который вносятся данные об объектах, при-

сутствующих на изображении; в нашем случае (детектирование объектов) это координаты ограничивающего прямоугольника.

Для расширения изначального объема данных, содержащихся в созданном нами наборе данных, применялась техника аугментации, при которой при опоре на оригинальный датасет искусственно генерируются новые его элементы. Это необходимо, когда используемый набор данных недостаточно велик или же он достаточно однороден.

Аугментация производилось у случайным образом отобранных изображений из набора данных и включала в себя следующие действия:

– масштабирование (изменение изначального размера изображений производилось при помощи деления или умножения на специальный коэффициент со значением 1,5 – т.е. в результате происходило уменьшение или увеличение размера картинки в 1,5 раза);

– сдвиг (производился относительно осей x и y; те части изображения, которые выходили за пределы своей изначальной конфигурации, обрезаются);

– поворот и отражение (для выбранного изображения производился либо поворот против часовой стрелки на 90º (для увеличения количества изображений с горизонтально лежащими книгами), либо отражение относительно оси y; таким образом, к изображению применялась только одна из перечисленных функций);

– добавление цифрового шума (производилось с целью ухудшения качества изначального изображения для имитации работы сети в реалистичных условиях);

– искажения цветовой гаммы (производились с помощью изменения тонов (сдвигом цветовой гаммы на 180 тонов) и насыщенности цвета в цветовой модели HSV).

Для отображения разницы между полученным ответом и тем, который мы стремились получить (т.е. ответом с максимальной точностью), в нейронной сети применяется функция ошибки. В случае с сетями YOLO наиболее часто используется следующая функция ошибки [1]:

$$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}\left[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2\right]+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}\left[(\sqrt{w}_i-\sqrt{\hat{w}_i})^2+(\sqrt{h}_i-\sqrt{\hat{h}_i})^2\right]+$$
$$+\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}(C_i-\hat{C}_i)^2+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{noobj}(C_i-\hat{C}_i)^2+\sum_{i=0}^{S^2}1_{ij}^{obj}\sum_{c\in classes}(p_i(c)-\hat{p}_i(c))^2, \quad (1)$$

где x, y – предсказанные координаты, $\hat{x}$, $\hat{y}$ – реальные координаты из набора данных; w, h – ширина и высота ограничивающего прямоугольника; C – вероятность присутствия объекта искомого класса на изображении; $\hat{C}$ – IoU (пересечение множества предсказанных значений со множеством значений контрольных данных).

IoU (Intersection over union), или мера Жаккара, – это метрика, используемая для определения процента или доли перекрытия предсказанной области местонахождения объекта области, где он реально располагается [29–30]. IoU численно равна площади пересечения 2 областей (A и B), деленной на общую площадь обеих рассматриваемых областей:

$$J(A,B)=|A\cap B|/|A\cup B|. \quad (2)$$

Для оценки качества работы нейросетей применяются различные метрики. Рассмотрим некоторые из них.

Average precision (*AP*) является одной из наиболее популярных базовых метрик качества, используемых в различных задачах, например, там, где необходимо оценить качество ранжирования, классификации или детектирования объектов. *AP* для серии запросов можно определить как усредненное значение средних оценок точности для каждого запроса [31]. В нашем случае метрика определяет долю правильно детектированных объектов. Математически *AP* представляет собой расчет значений $p(r)$ – функции recall – в интервале значений $r=(0;1)$:

$$AP=\int_0^1 p(r)dr. \quad (3)$$

Среди других популярных метрик – precision (точность) и recall / sensitivity (полнота / чувствительность) [32]. Precision определяет соотношение количества объектов, принадлежащих к заданному классу, и общего количества предсказанных объектов (false positive rate), иными словами, – процент корректных предсказаний. Recall определяет соотношение количества объектов, принадлежащих к заданному классу, и общего количества объектов данного класса в наборе данных (false negative rate), т.е. определяет, насколько хорошо происходят предсказания.

$$\text{Precision}=TP/(TP+FP), \quad (4)$$

$$\text{Recall}=TP/(TP+FN), \quad (5)$$

где *TP* – true positive (объект рассматриваемого класса детектирован верно);

TN – true negative (объект верно не детектирован как объект рассматриваемого класса);

FP – false positive (объект ошибочно отнесен к рассматриваемому классу);

FN – false negative (объект ошибочно не детектирован как объект рассматриваемого класса).

Обучение сети

Код нейронной сети был реализован на языке программирования Python 3.6. Для создания структуры сверточной сети, предназначенной для детектирования книг, использовались Python-библиотеки глубокого обучения Keras 2.2.4 и TensorFlow 1.12.0. Также дополнительно применялась библиотека OpenCV для считывания изображений и реализации функций аугментации. В процессе создания нейронной сети было опробовано несколько вариантов сетевой структуры, в том числе приближенной к tiny YOLO. Тем не менее, лучшие результаты были достигнуты в случае использования оригинальной структуры архитектуры YOLOv3. Схему структуры сети можно увидеть на рис. 1.

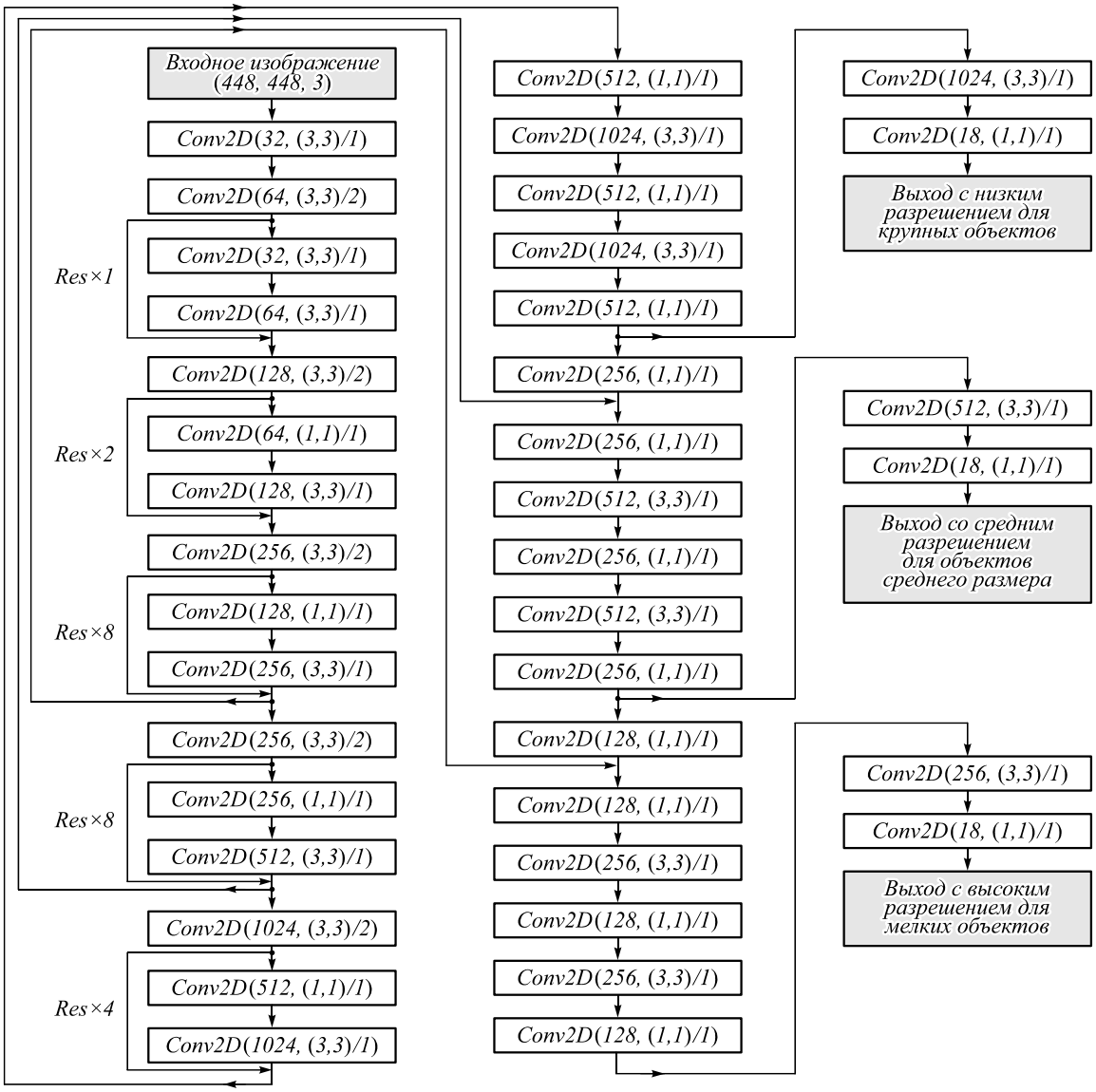

Рис. 1. Схема структуры нейронной сети архитектуры YOLOv3

Обучение сети производилось на GPU Nvidia GeForce GTX 1060 6GB при следующих параметрах:

– размер изображения на входе нейросети – 448×448×3;
– оптимизатор сети – Adam;
– коэффициент скорости обучения сети – 0,0001;
– размерность батча – 2 образца;
– функция ошибки – формула (1);
– метрика, используемая для оценки качества сети – Average Precision;
– число эпох обучения – 800.

Набор данных, на котором производилось обучение глубокой нейронной сети, состоял из 500 изображений с 5245 изображенными на них книжными корешками. Далее он делился на обучающий, валидационный и тестовый наборы в соотношении 6:1:3 соответственно. Наилучшие результаты были достигнуты на 710 эпохе. В табл. 1 показаны полученные в процессе обучения сети наилучшие значения AP и потерь для каждой из этих выборок.

Задача обучения нейронной сети сводится к минимизации функции потерь. График обучения с полу-

ченными значениями ошибки на каждой эпохе показан на рис. 2.

Табл. 1. Наилучшие значения AP и потерь для обучающей, валидационной и тестовой выборок

Выборка	AP	Потери (loss)
Обучающая	93,74 %	7,38
Валидационная	91,76 %	6,57
Тестовая	94,43 %	6,74

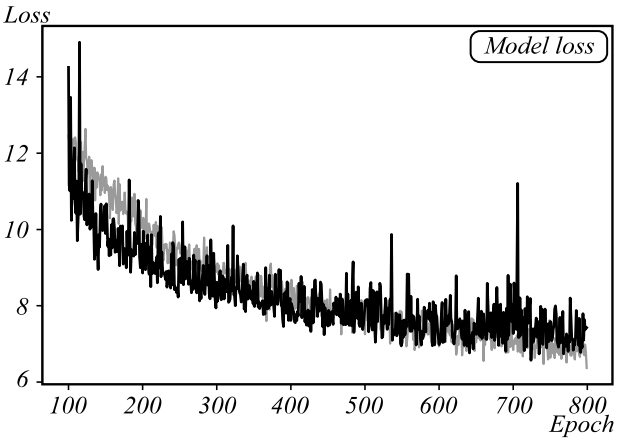

Рис. 2. График изменения значений ошибки на обучающей и валидационной выборках

Рис. 3*а* и 3*б* иллюстрируют работу нейросети после обучения. Над каждой полученной рамкой выведены значения вероятности, что найденный объект относится к заданному классу. Рис. 3*а* содержит изображения горизонтально и вертикально расположенных книг, рис. 3*б* – книг, схожих по оформлению, среди которых есть расположенные под углом. Мы видим, что нейронная сеть уверенно научилась распознавать и детектировать книги по книжным корешкам. Используемая в качестве базиса для создания приложения быстрая сеть YOLOv3 позволяет производить детектирование в режиме реального времени.

Заключение

В данной статье был рассмотрен первый этап создания приложения для детектирования книг по книжным корешкам. В качестве основы была использована нейронная сеть архитектуры YOLOv3, отличающейся значительной быстротой получения выходных данных и относительно небольшим значением ошибки, что делает ее пригодной для организации детектирования книг в режиме видеопотока. Результаты, полученные после обучения глубокой нейронной сети, показали, что сеть научилась в достаточной мере уверенно распознавать и детектировать объекты заданного класса, справляться с распознаванием корешков одинакового цвета, схожих по оформлению или расположенных под углом. В дальнейшем планируется продолжить работу по ее усовершенствованию и обучению распознавать текст на корешках книг.

Рис. 3. Пример, иллюстрирующий работу нейросети по детектированию

Литература

1. **Quoc, N.** A framework for recognition books on bookshelves / N. Quoc, W. Choi // Proceedings of the ICIC 2009: Emerging Intelligent Computing Technology and Applications. – 2009. – P. 386-395. – DOI: 10.1007/978-3-642-04070-2_44.
2. **Tsai, S.S.** Combining image and text features: A hybrid approach to mobile book spine recognition / S.S. Tsai, D. Chen, H. Chen, C. Hsu, K. Kim, J.P. Singh, B. Girod // Proceedings of the 2011 ACM international conference on Multimedia. – 2011. – P. 1029-1032. – DOI: 10.1145/2072298.2071930.
3. **Chen, D.** Low-cost asset tracking using location-aware camera phones / D. Chena, S. Tsaia, K. Kimb, C. Hsub, J.P. Singhb, B. Giroda // Proceedings of SPIE. – 2010. – Vol. 7798. – 77980R. – DOI: 10.1117/12.862426.
4. **Chen, D.** Mobile augmented reality for books on a shelf / D. Chen, S. Tsai, C. Hsu, J.P. Singh, B. Girod // Proceedings of the 2011 IEEE International Conference on Multimedia and Expo. – 2011. – P. 1-6. – DOI: 10.1109/ICME.2011.6012171.
5. **Lee, D.J.** Matching book-spine images for library shelf-reading process automation / D.J. Lee, Y. Chang, J.K. Archibald, C. Pitzak // Proceedings of the 2008 IEEE International Con-

ference on Automation Science and Engineering. – 2008. – P. 738-743. – DOI: 10.1109/COASE.2008.4626503.
6. **Nevetha, M.P.** Automatic book spine extraction and recognition for library inventory / M.P. Nevetha, A. Baskar // Management WCI '15: Proceedings of the Third International Symposium on Women in Computing and Informatics. – 2015. – P. 44-48. – DOI: 10.1145/2791405.2791506.
7. **Jubair, M.I.** A technique to detect books from library bookshelf image / M.I. Jubair, P. Banik // Proceedings of the 2013 IEEE 9th International Conference on Computational Cybernetics (ICCC). – 2013. – P. 359-363. – DOI: 10.1109/ICCCyb.2013.6617619.
8. **Talker, L.** Viewpoint-independent book spine segmentation / L. Talker, Y. Moses // Proceedings of the IEEE Winter Conference on Applications of Computer Vision. – 2014. – P. 453-460. – DOI: 10.1109/WACV.2014.6836066.
9. **Yang, X.** Smart library: Identifying books on library shelves using supervised deep learning for scene text reading / X. Yang, D. He, W. Huang, A. Ororbia, Z. Zhou, D. Kifer, C.L. Giles // Proceedings of the 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL). – 2017. – P. 1-4. – DOI: 10.1109/JCDL.2017.7991581.
10. **Anegawa, R.** Text detection on books using cnn trained with another domain data / R. Anegawa, M. Aritsugi // Proceedings of the 2019 IEEE International Conference on Dependable, Autonomic and Secure Computing 2019: – P. 170-176. – DOI: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00041.
11. **Gandhi, R.** R-CNN, Fast R-CNN, Faster R-CNN, YOLO – object detection algorithms [Electronical Resource] / R. Gandhi // – 2018. – URL: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e (request date 11.02.2020).
12. **Karatzas, D.** ICDAR 2015 competition on robust reading / D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V.R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida, E. Valveny // Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR). – 2015. – P. 1156-1160.
13. **Redmon, J.** You only look once: Unified, real-time object detection / J. Redmon, S. Divvala, R. Girshick, A. Farhadi // Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition. – 2016. – P. 779-788. – DOI: 10.1109/CVPR.2016.91.
14. **Redmon, J.** Yolo9000: Better, faster, stronger / J. Redmon, A. Farhady // Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. – 2017. – P. 6517-6525.
15. **Redmon, J.** YOLOv3: An incremental improvement [Electronical Resource] / J. Redmon, A. Farhady. – 2018. – URL: https://arxiv.org/pdf/1804.02767.pdf (request date 09.02.2020).
16. **Liu, W.** SSD: Single shot multibox detector / W. Liu, D. Anuelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. Berg. – In: Computer Vision – ECCV 2016 / ed. by B. Leibe, J. Matas, N. Sebe, M. Welling. – Cham: Springer, 2016. – DOI: 10.1007/978-3-319-46448-0_2.
17. **Lin, T.Y.** Focal loss for dense object detection [Electronical Resource] / T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár. – 2018. – URL: https://arxiv.org/pdf/1708.02002.pdf (request date 12.02.2020).
18. DetectNet: Deep Neural Network для Object Detection в DIGITS [Электронный ресурс]. – URL: https://habr.com/ru/post/310332/ (дата обращения 05.02.2020).
19. **Ren, S.** Faster R-CNN: Towards real-time object detection with region proposal networks [Electronical Resource] / S. Ren, K. He, R. Girshick, J. Sun // arXiv Preprint. – 2016. – URL: https://arxiv.org/pdf/1506.01497.pdf (request date 10.02.2020).
20. **He, K.** Mask R-CNN [Electronical Resource] / K. He, G. Gkioxari, P. Dollár, R. Girshick // arXiv Preprint. – 2018. – URL: https://arxiv.org/pdf/1703.06870.pdf (request date 09.02.2020).
21. Mask R-CNN: архитектура современной нейронной сети для сегментации объектов на изображениях [Электронный ресурс]. – 2018. – URL: https://habr.com/en/post/421299/ (дата обращения 11.02.2020).
22. **Liu, W.** SSD: Single shot multibox detector [Electronical Resource] / W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg. – ArXiv Preprint. – 2016. – URL: https://arxiv.org/pdf/1512.02325.pdf (request date 10.02.2020).
23. **Tsang, S.** Review: SSD – single shot detector (object detection) [Electronical Resource] / S. Tsang. – 2018. – URL: https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11 (request date 14.02.2020).
24. YOLO: Real-time object detection [Electronical Resource]. – URL: https://pjreddie.com/darknet/yolo/ (request date 09.02.2020).
25. **Sambasivarao, K.** Non-maximum suppression (NMS) [Electronical Resource] / K. Sambasivarao. – 2019. – URL: https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c (request date 12.02.2020).
26. **Bindal, A.** Normalization techniques in deep neural networks [Electronical Resource] // A. Bindal. – 2019. – URL: https://medium.com/techspace-usict/normalization-techniques-in-deep-neural-networks-9121bf100d8 (request date 09.02.2020).
27. **Sharma, H.** Activation functions: Sigmoid, ReLU, Leaky ReLU and Softmax basics for neural networks and deep learning [Electronical Resource] // H. Sharma. – 2019. – URL: https://medium.com/@himanshuxd/activation-functions-sigmoid-relu-leaky-relu-and-softmax-basics-for-neural-networks-and-deep-8d9c70eed91e (request date 08.02.2020).
28. The PASCAL Visual Object Classes homepage [Electronical Resource]. – URL: http://host.robots.ox.ac.uk/pascal/VOC/ (request date 11.02.2020).
29. **Arlen, T.C.** Understanding the mAP evaluation metric for object detection [Electronical Resource] / T.C. Arlen. – URL: https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3 (request date 14.02.2020).
30. **Saxen, S.** Precision vs Recall [Electronical Resource] / S. Saxen. – 2018. – URL: https://towardsdatascience.com/precision-vs-recall-386cf9f89488 (request date 08.02.2020).
31. **Sandeep, A.** Object detection – IOU – Intersection Over Union [Electronical Resource] / A. Sandeep. – 2019. – URL: https://medium.com/@nagsan16/object-detection-iou-intersection-over-union-73070cb11f6e (request date 09.02.2020).
32. **Bodla, N.** Improving object detection with one line of code [Electronical Resource] / N. Bodla, B. Singh, R. Chellappa, L.S. Davis. – ArXiv Preprint. – 2017. – URL: https://arxiv.org/pdf/1704.04503.pdf (request date 08.02.2020).

Сведения об авторах

Калинина Мария Олеговна, 1996 года рождения, магистр 2-го курса Московского авиационного института (национального исследовательского университета). Область научных интересов: программирование, машинное обучение, искусственный интеллект, глубокие нейронные сети, распознавание образов.
E-mail: *Ptaha-96@yandex.ru* .

Николаев Павел Леонидович, 1989 года рождения, старший преподаватель Московского авиационного института (национального исследовательского университета). Область научных интересов: программирование, машинное обучение, искусственный интеллект, глубокие нейронные сети, распознавание образов.
E-mail: *npavel89@gmail.com* .

ГРНТИ: 28.23.15; 28.23.37
Поступила в редакцию 4 апреля 2020 г. Окончательный вариант – 7 октября 2020 г.

Book spine recognition with the use of deep neural networks

M.O. Kalinina [1], P.L. Nikolaev [1]
[1]Moscow Aviation Institute (National Research University),
121552, Moscow, Russia, Orshanskaya 3

Abstract

Nowadays deep neural networks play a significant part in various fields of human activity. Especially they benefit spheres dealing with large amounts of data and lengthy operations on obtaining and processing information from the visual environment. This article deals with the development of a convolutional neural network based on the YOLO architecture, intended for real-time book recognition. The creation of an original data set and the training of the deep neural network are described. The structure of the neural network obtained is presented and the most frequently used metrics for estimating the quality of the network performance are considered. A brief review of the existing types of neural network architectures is also made. YOLO architecture possesses a number of advantages that allow it to successfully compete with other models and make it the most suitable variant for creating an object detection network since it enables some of the common disadvantages of such networks to be significantly mitigated (such as recognition of similarly looking, same-color book coves or slanted books). The results obtained in the course of training the deep neural network allow us to use it as a basis for the development of the software for book spine recognition.

Keywords: image recognition; object detection; computer vision; machine learning; artificial neural networks; deep learning; convolutional neural networks.

Citation: Kalinina MO, Nikolaev PL. Book spine recognition with the use of deep neural networks. Computer Optics 2020; 44(6): 968-977. DOI: 10.18287/2412-6179-CO-731.

References

[1] Quoc N, Choi W. A Framework for Recognition Books on Bookshelves. Proc ICIC 2009: Emerging Intelligent Computing Technology and Applications; 2009; 386-395. DOI: 10.1007/978-3-642-04070-2_44.

[2] Tsai SS, Chen D, Chen H, Hsu C, Kim K, Singh JP, Girod B. Combining image and text features: A hybrid approach to mobile book spine recognition. Proc 2011 ACM Int Conf on Multimedia 2011: 1029-1032. DOI: 10.1145/2072298.2071930.

[3] Chen D, Tsaia S, Kimb K, Hsub C, Singhb JP, Giroda B. Low-cost asset tracking using location-aware camera phones. Proc SPIE 2010; 7798: 77980R. DOI: 10.1117/12.862426.

[4] Chen D, Tsai S, Hsu C, Singh JP, Girod B. Mobile augmented reality for books on a shelf. Proc 2011 IEEE Int Conf on Multimedia and Expo 2011: 1-6. DOI: 10.1109/ICME.2011.6012171.

[5] Lee DJ, Chang Y, Archibald JK, Pitzak C. Matching book-spine images for library shelf-reading process automation. Proc 2008 IEEE Int Conf on Automation Science and Engineering 2008: 738-743. DOI: 10.1109/COASE.2008.4626503.

[6] Nevetha MP, Baskar A. Automatic book spine extraction and recognition for library inventory. Management WCI '15: Proc 3rd Int Symposium on Women in Computing and Informatics 2015: 44-48. DOI: 10.1145/2791405.2791506.

[7] Jubair MI, Banik P. A technique to detect books from library bookshelf image. Proc 2013 IEEE 9th Int Conf on Computational Cybernetics (ICCC) 2013: 359-363. DOI: 10.1109/ICCCyb.2013.6617619.

[8] Talker L, Moses Y. Viewpoint-independent book spine segmentation. Proc IEEE Winter Conf on Applications of Computer Vision; 2014: 453-460. DOI: 10.1109/WACV.2014.6836066.

[9] Yang X, He D, Huang W, Ororbia A, Zhou Z, Kifer D, Giles CL. Smart library: Identifying books on library shelves using supervised deep learning for scene text reading. Proc 2017 ACM/IEEE Joint Conf on Digital Libraries (JCDL); 2017: 1-4. DOI: 10.1109/JCDL.2017.7991581.

[10] Anegawa, R., Aritsugi, M. Text Detection on Books Using CNN Trained with Another Domain Data. Proc. 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing; 2019: 170-176. DOI: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00041.

[11] Gandhi R. R-CNN, Fast R-CNN, Faster R-CNN, YOLO – object detection algorithms. 2018. Source: ⟨https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e⟩.

[12] Karatzas D, Gomez-Bigorda L, Nicolaou A, Ghosh S, Bagdanov A, Iwamura M, Matas J, Neumann L, Chandrasekhar VR, Lu S, Shafait F, Uchida S, Valveny E. ICDAR 2015 competition on robust reading. Proc 2015 13th Int Conf on Document Analysis and Recognition (ICDAR) 2015: 1156-1160.

[13] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. Proc 2016 IEEE Conf Comp Vis Pattern Recogn 2016; 779-788. DOI: 10.1109/CVPR.2016.91.

[14] Redmon J, Farhady A. Yolo9000: Better, faster, stronger Proc 2017 IEEE Conf Comp Vis Pattern Recogn 2017: 6517-6525.

[15] Redmon J, Farhady A. YOLOv3: An incremental improvement. 2018. Source: ⟨https://arxiv.org/pdf/1804.02767.pdf⟩.

[16] Liu W, Anuelov D, Erhan D, Szegedy C, Reed S, Fu C, Berg A. SSD: Single shot multibox detector. In Book: Leibe B, Matas J, Sebe N, Welling M, eds. Computer Vision – ECCV 2016. Cham: Springer; 2016. DOI: 10.1007/978-3-319-46448-0_2.

[17] Lin TY, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. 2018. Source: ⟨https://arxiv.org/pdf/1708.02002.pdf⟩.
[18] Tao A, Barker J, Sarathy S. DetectNet: Deep Neural Network for Object Detection in DIGITS. Source: ⟨https://developer.nvidia.com/blog/detectnet-deep-neural-network-object-detection-digits/⟩.
[19] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. 2016. Source: ⟨https://arxiv.org/pdf/1506.01497.pdf⟩.
[20] He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. 2018. Source: ⟨https://arxiv.org/pdf/1703.06870.pdf⟩.
[21] Mask R-CNN: architecture of modern neuron network for object segmentation on image [In Russian]. 2018. Source: ⟨https://habr.com/en/post/421299/⟩.
[22] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC. SSD: Single shot multibox detector. 2016. Source: ⟨https://arxiv.org/pdf/1512.02325.pdf⟩.
[23] Tsang S. Review: SSD – single shot detector (object detection). 2018. Source: ⟨https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11⟩.
[24] YOLO: Real-time object detection. Source: ⟨https://pjreddie.com/darknet/yolo/⟩.
[25] Sambasivarao K. Non-maximum suppression (NMS). 2019. Source: ⟨https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c⟩.
[26] Bindal A. Normalization Techniques in Deep Neural Networks. 2019. Source: ⟨https://medium.com/techspace-usict/normalization-techniques-in-deep-neural-networks-9121bf100d8⟩.
[27] Sharma H. Activation functions: Sigmoid, ReLU, Leaky ReLU and Softmax basics for neural networks and deep learning. 2019. Source: ⟨https://medium.com/@himanshuxd/activation-functions-sigmoid-relu-leaky-relu-and-softmax-basics-for-neural-networks-and-deep-8d9c70eed91e⟩.
[28] The PASCAL Visual Object Classes homepage. Source: ⟨http://host.robots.ox.ac.uk/pascal/VOC/⟩.
[29] Arlen TC. Understanding the mAP evaluation metric for object detection. Source: ⟨https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3⟩.
[30] Saxen S. Precision vs Recall. 2018. Source: ⟨https://towardsdatascience.com/precision-vs-recall-386cf9f89488⟩.
[31] Sandeep A. Object detection – IOU – Intersection Over Union. 2019. Source: ⟨https://medium.com/@nagsan16/object-detection-iou-intersection-over-union-730`70cb11f6e ⟩.
[32] Bodla N, Singh B, Chellappa R, Davis LS. Improving object detection with one line of code. 2017. Source: ⟨https://arxiv.org/pdf/1704.04503.pdf ⟩.

Authors' information

Maria Olegovna Kalinina (b. 1996) Master's student, Moscow Aviation Institute (National Research University). Research interests are programming, machine learning, artificial intelligence, deep neural networks, pattern recognition. E-mail: *Ptaha-96@yandex.ru* .

Pavel Leonidovich Nikolaev, (b. 1989), Senior lecturer, Moscow Aviation Institute (National Research University). Research interests are programming, machine learning, artificial intelligence, deep neural networks, pattern recognition. E-mail: *npavel89@gmail.com* .

Received April 4, 2020. The final version – October 7, 2020.