ВЕСТНИК ЮГОРСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
2021 г. Выпуск 4 (63). С. 47–54

DOI: 10.17816/byusu20210447-54
УДК 004.032.26

Е. А. Годовников, О. А. Петухова, Т. В. Пронькина, Р. Т. Усманов, А. В. Шицелов

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ОПРЕДЕЛЕНИЯ ТРАНСПОРТНОГО ПОТОКА НА ВИДЕОЗАПИСИ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННОЙ СЕТИ

В данной статье рассматривается проблема определения транспортных потоков на перекрестке путем идентифицирования автомобилей посредством применения нейронных сетей. В работе описаны модель нейронной сети, используемой для обнаружения транспортных средств на изображении, алгоритм идентифицирования автомобилей на перекрестке, а также алгоритм получения траектории движения автомобиля.

Ключевые слова: Нейронная сеть, интеллектуальная система, перекресток, Nvidia SSD

E. A. Godovnikov, O. A. Petuhova, T. V. Pronkina,R. T. Usmanov, A. V. Shitzelov

DEVELOPMENT OF AN AUTOMATED SYSTEM FOR DETERMINING TRANSPORT FLOW ON VIDEO USING A NEURAL NETWORK

This article discusses the problem of determining traffic flows at an intersection by identifying vehicles using neural networks. A model of a neural network used to detect vehicles in the image is described. The paper describes an algorithm for identifying cars at an intersection. The article also describes an algorithm for obtaining the trajectory of the vehicle.

Keywords: Neural network, intelligent system, crossroads, Nvidia SSD

Современный уровень развития методов искусственного интеллекта позволяет разрабатывать методы и приложения, которые обеспечивают эффективное управление транспортными потоками. Проект «Умный город» является примером использования нейронных сетей, с помощью которых решается современная проблема пробок. Пробки возникают по нескольким причинам, а именно: час-пик, ремонтные работы, аварии, светофоры и многое другое [1].

Сейчас данный класс проблем принято решать путем построения математической модели транспортной сети. При этом сама модель может быть выполнена в классическом стиле или может быть реализована с использованием современного подхода, который заключается в применении математических моделей, основанных на теории машинного обучения и нейронных сетей.

К сожалению, какой бы подход к решению данной проблемы не был выбран, остается одна очень важная проблема – получение первоначальных статистических данных о транспортном потоке. Эти данные очень важны как для самой модели, особенно если модель транспортного потока будет основана на теории машинного обучения, так и для этапа апробации результатов работы модели, а также для последующего корректирования параметров ее работы.

Для получения статистических данных можно выделить 2 подхода:

- Сбор данных с помощью человеческих ресурсов;
- Автоматизированный сбор данных.

Сбор данных с использованием человеческих ресурсов основан на том, что для подсчета количества транспортных средств, проезжающих через ключевой транспортный узел, используется человек. У данного подхода имеются как хорошие, так и плохие стороны.

Хорошие стороны:

- Легко организовать;

– Требует малых начальных вложений;
– Нет необходимости использовать сложное техническое оборудование.

Плохие стороны:

– Масштабируем только для некоторого предела;
– Сложно, либо невозможно повторно собрать статистические данные;
– Человек подвержен усталости;
– Нестабильное качество работы;
– При больших объёмах входной информации требует серьёзных затрат.

Автоматический сбор данных основан на обработке видеопотока на ЭВМ с автоматическим обнаружением и подсчетом на нем транспортных средств.

Хорошие стороны данного подхода:

– Масштабируем для очень большого числа транспортных узлов;
– Стабильная работа (качество) сбора данных.

Плохие стороны:

– Требует больших затрат на первоначальном этапе.

На данный момент из-за развития области машинного обучения и нейронных сетей, а также появления общедоступных предобученных моделей для распознавания образов, первоначальные расходы на системы сбора статистических данных о движении транспортных средств снизились, а также повысилась их точность.

Поэтому для решения поставленной задачи о сборе данных для математической модели предлагается использовать автоматическую систему с использованием нейронных сетей.

В основе автоматические системы подсчета траффика лежит нейронная сеть, которая представляет из себя математическую модель, построенную с применением теории из области искусственного интеллекта и «обученную» специальным образом.

Под термином «обучение» подразумевается процесс численной минимизации ошибки, возникающей между предсказанными значениями, моделью и реальными данными. Данными процесс представляет собой численную оптимизацию внутренней структуры модели, направленную на уменьшение ошибки между предсказаниями и реальными данными.

Реальные данные (или же данные для обучения) представляют собой видеозапись, полученную с видеокамер, где все объекты были заранее размечены. Также такой вид данных называется «датасетом».

Одним из самых больших датасетов данных для обучения является ImageNet. ImageNet представляет собой набор данных, представленных изображениями с высоким разрешением, которые люди помечали вручную. Насчитывается более 20 тысяч видов категорий, на которые разбиты данные изображения, а самих изображений – миллионы. Свое начало ImageNet берет в 2010 году. В рамках конкурса «Pascal» ежегодно люди размечают изображения, которые в последствии будут использованы для составления датасетов. Благодаря данным датасетам появилась возможность проверки и сравнения производительности нейронных сетей для классификации объектов [2].

Так как в данной работе необходима работа с видеопотоком (представляющим собой последовательные кадры), а именно отслеживание автомобилей на перекрестке, необходима модель, которая будет находить автомобили на изображении.

В качестве основы для разработки была выбрана модель под названием Nvidia SSD (наименование модели SSD – Single Shot MultiBox Detector) [3]. Ее архитектура указана на рисунке 1.

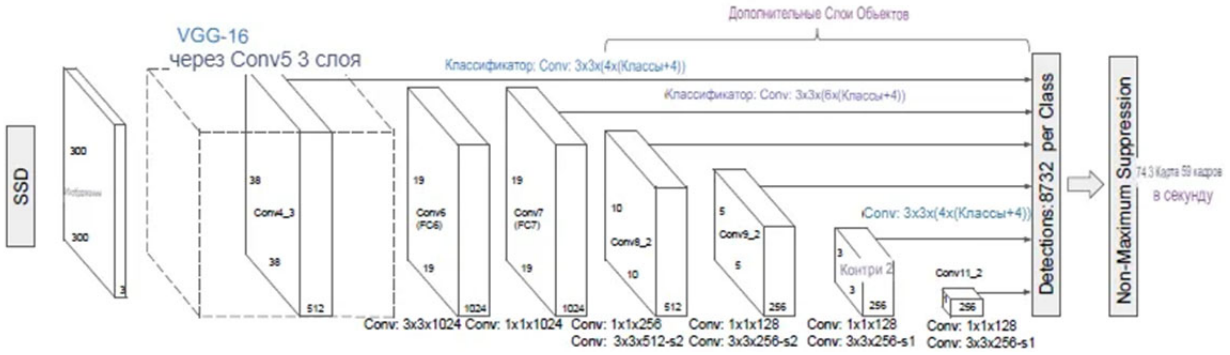

Рисунок 1. Архитектура модели Nvidia SSD

Основными достоинствами для ее выбора послужили высокая скорость работы и хороший уровень точности по сравнению с другими моделями.

Принцип работы данной модели проходил в два этапа. На первом этапе модель сама разделяла видеопоток на пакеты (отдельные изображения в определенном количестве), на втором этапе модель обрабатывала полученные пакеты и создавала новое видео, в котором были обведены рамками автомобили (см. Рис. 2).

Рисунок 2. Кадр после обработки

Идентифицирование автомобилей на перекрестке

Для реализации данной интеллектуальной информационной системы, исходя из поставленных задач, необходимо разработать алгоритм идентифицирования автомобилей на перекрестке.

Выходные данные нейронной сети представляют собой изображение, на котором обозначены объекты. Объекты, которые обнаруживает нейронная сеть, делятся на классы. Для модели SSD300 v1.1. список классов достаточно обширен, но в данной системе необходимым классом обнаружения являются автомобили.

После обработки кадра видеопотока, нейронная сеть с некой вероятностью выдает координаты объекта класса, индекс класса и изображение. Координаты объектов являются последовательностью координат, а именно значение левого верхнего и правого нижнего углов прямоугольника. Значения координат точки определяются относительно высоты (длины) и ширины изображения. Индексом класса является число, соответствующее обозначению данного класса. На основе полученных данных при помощи простой функции происходит выделение области и классификация данного объекта.

Таким образом, нейронная сеть классифицирует объекты на последовательных кадрах исходного видеоряда.

Алгоритм идентифицирования автомобилей необходим для создания хранилища автомобилей, которое является таблицей с ключом и значением, где ключ – число, а значение – последовательные координаты левого верхнего и правого нижнего углов прямоугольника. Для понимания того, как работает алгоритм, необходимо рассмотреть два последовательных кадра обработанных нейронной сетью. За основу необходимо взять любой кадр выходных данных нейронной сети (см. Рис. 3).

Рисунок 3. Начальный кадр видео

Следующим шагом необходимо взять следующий кадр (см. Рис. 4).

Рисунок 4. Второй кадр видео

Дифференцирование – это наложение двух изображений друг на друга путем изменения прозрачности одного из изображений. После выполнения дифференцирования данных кадров получаем результат, представленный на рисунке 5. Данный подход используется для обнаружения движущихся объектов по маске [4]. Исходя из полученного результата можно сделать следующие выводы: очертания автомобилей или их рамка в соседних кадрах пересекаются, и расстояние между центрами рамок в соседних кадрах не выходит за пределы очертания автомобиля первого кадра из-за достаточного значения кадров в секунду.

Рисунок 5. Дифференцирование кадров видео

На основе полученных выводов был разработан алгоритм идентифицирования автомобилей, для которой достаточно выполнение двух условий.

Во-первых, очертания автомобиля, его контур в соседних кадрах должны пересекаться любым способом. При пересечении контура несколькими контурами, необходима вторая проверка.

Во-вторых, для избегания ситуаций ложного идентифицирования, необходимо помимо проверки пересечения контуров также проверять расстояние между центрами данных контуров. На рисунке 5 видно, что расстояние между центрами автомобиля в соседних кадрах минимальное. Когда возникают спорные ситуации с несколькими пересечениями, для корректировки данных выбирается то пересечение, которое имеет наименьшее расстояние между центрами.

Разработка алгоритма идентифицирования на основе выходных данных нейросети

Алгоритм идентифицирования автомобилей необходим для получения траектории движения транспортного средства на перекрестке. Выходными данными нейронной сети являются координаты очертаний автомобиля, которые для упрощения приведены к виду прямоугольника и имеют вид последовательных чисел значений левого верхнего и правого нижнего углов прямоугольника.

Так как рамки очертания автомобилей в двух последовательных кадрах пересекаются, необходимо использовать данный вывод в разработке алгоритма.

Первым условием, по которому возможно определить, происходит ли пересечение рамок, является наличие точки центра их второго кадра в рамке первого. В Python для решения данной задачи существует библиотека, проверяющая нахождение точки в полигоне. Полигоном в данном случае является рамка первого кадра, состоящая из двух точек, из которых возможно получить все четыре точки рамки. Далее из полученных точек необходимо создать полигон и проверить, находится ли точка из последующего кадра в данном полигоне (Листинг кода 1) [5].

Листинг кода 1: проверка на принадлежность точки полигону

```
poly = Polygon(coords)
p1 = Point(getCentroid(Ex, Ey, Gx, Gy))
if p1.within(poly):
      return True
```

Недостаток данного решения в том, что при большой скорости автомобиля рамки могут пересекаться, но центр может не попадать в область рамки из предыдущего кадра. Исходя из этого, необходима доработка алгоритма, с проверкой на пересечения рамок.

Существует несколько способов проверки пересечения прямоугольников, в том числе проверка принадлежности точки вершины одного из прямоугольников в области другого. При любом пересечении прямоугольников один из углов находится непосредственно в другом прямоугольнике, за исключением касательного характера, когда одна из граней одного прямоугольника находится на одной из граней другого прямоугольника. В данном случае из-за оптимальной частоты кадров касательное пересечение граней будет говорить о том, что это не один и тот же автомобиль.

Для проверки нахождения угла одного прямоугольника на площади другого прямоугольника необходима выполнение двух условий. Координаты данного угла должны находиться в диапазоне координат левого верхнего и правого нижнего углов черного прямоугольника (см. Рис. 6) [6].

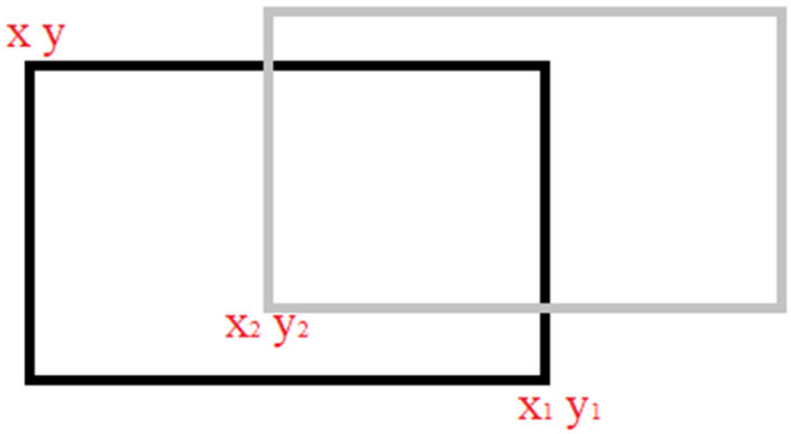

Рисунок 6. Пример пересечения прямоугольников

Как видно из примера (см. Рис. 3.2), необходимо выполнение двух неравенств, где:
x – координата ширины левого верхнего угла черного прямоугольника,
y – координата высоты левого верхнего угла черного прямоугольника,
x_1 – координата ширины правого нижнего угла черного прямоугольника,
y_1 – координата высоты правого нижнего угла черного прямоугольника,
x_2 – координата ширины левого нижнего угла серого прямоугольника,
y_2 – координата высоты левого нижнего угла серого прямоугольника,

$$x \leq x_2 \leq x_1, \quad (1)$$

$$y \leq y_2 \leq y_1. \quad (2)$$

Данные неравенства должны выполняться для одного из углов второго прямоугольника, в ином случае необходимо провести процедуру проверки принадлежности углов черного треугольника к серому [7].

При невыполнении данных условий следует вывод об отсутствии пересечения, следовательно, это два разных автомобиля.

Вследствие ракурса, возникают ситуации, когда происходит наложение нескольких рамок (см. Рис. 7).

Рисунок 7. Пример наложения рамок разных автомобилей

Для решения данной проблемы необходимо дополнения алгоритма идентифицирования сравнением минимального расстояния между рамками, с которыми было найдено пересечение. Так как центры имеют координаты на плоскости, необходимо воспользоваться формулой для нахождения расстояния между точками на плоскости где:

d – расстояние между координатами,

x_1 – координата ширины центра первой рамки,

y_1 – координата высоты центра первой рамки,

x_2 – координата ширины центра второй рамки,

y_2 – координата высоты центра второй рамки,

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad (3)$$

Далее необходимо сравнить полученные расстояния, – расстояние, которое окажется минимальным и будет означать, что рамка, которой принадлежит данная точка центра, относится к данному автомобилю.

Записи о траектории движения транспортных средств хранятся в словаре, который имеет вид таблицы с ключом и значением. Ключом в данном словаре является счетчик, который при запуске программы имеет нулевое значение. Значением выступает двумерный массив, состоящий из массивов, в которых находится последовательность координат рамок автомобилей, а именно координаты левого верхнего и правого нижнего углов. Иными словами, словарь является таблицей состоящий из двух столбцов: ключ и значение.

Так как во время запуска словарь является пустым и алгоритм идентифицирования транспортных средств не может начать свою работу, необходимо предварительно заполнить его первоначальными данными. Во время того, как впервые появляются данные об автомобилях, происходит заполнение словаря путем присвоения ключа, с его последующей инкрементацией, и координат обнаруженных автомобилей (Листинг кода 2).

Листинг кода 2 – Первое заполнение словаря

```
for i in range(0, len(self.firstFrame)):
    self.dictCoordinate[self.countCars] = [self.firstFrame[i]
    self.countCars+=1
```

Заполнение словаря происходит в случае первого запуска системы или первого получения координат автомобилей, либо в случае полного очищения словаря (Листинг кода 3). Полное очищение словаря происходит во время отсутствия транспортных средств на перекрестке.

Листинг кода 3 – Проверка на наличие записей в словаре

```
if not len(self.dictCoordinate):
    self.firstFrame.append([x1, y1, x2, y2])
elif len(self.dictCoordinate):
    self.secondFrame.append([x1, y1, x2, y2])
```

Последующие заполнения словаря происходят путем прохождения полного алгоритма идентифицирования автомобилей. При появлении координат автомобиля, который только начал движение по перекрестку, совпадений с существующими записями не происходит и данный автомобиль создается по аналогии с первоначальным заполнением словаря (Листинг кода 4).

Листинг кода 4 – Заполнения словаря новыми автомобилями

```
if self.newBoxesfdict:
    for i in self.newBoxesfdict:
        self.dictCoordinate[self.countCars] = [i]
        self.countCars += 1
```

Очищение словаря происходит с помощью алгоритма получения статистических данных о движении транспортных средств по перекрестку.

Вывод

Как можно видеть, для реализации задачи по сбору данных с помощью автоматизированной системы был разработан и написан алгоритм, основанный на многих предыдущих и использующий сильные стороны других алгоритмов со сглаживанием или полным устранением плохих особенностей. Данная система была написана и протестирована на записях видео с перекрестков, при этом те же самые записи были даны людям для ручного подсчета трафика. Как следует из эксперимента, система показала сопоставимый с человеком результат.

Литература

1. Казарян, Д. Э. Нейросетевые подходы к управлению потоками транспорта Д. Э. Казарян, В. А. Михалев, Е. А. Софронова. – Текст : непосредственный // Вестник Российского университета дружбы народов. Серия: Инженерные исследования. – 2017. – Т. 18, № 1. – С. 97–106.

2. Хранилище классифицированных датасетов. – Текст : электронный // IM Genet. – URL: https://image-net.org/index.php (дата обращения: 15.01.2021).

3. Hui, J. SSD object detection: Single Shot MultiBox Detector for real-time processing // Jonathan Hui. – URL: https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06 (date of application: 10.10.2021).

4. Тукмачева, Ю. А. Обзор алгоритмов, используемых для определения интенсивности дорожного движения на перекрестке / Ю. А. Тукмачева. – Текст : непосредственный // Евразийское Научное Объединение. – 2021. – № 8-1 (78). – С. 61–64.

5. Документация Shapely // Shapely 1.8.0. – URL: https://pypi.org/project/Shapely/ (date of application: 30.01.2021).

6. Амелькин, В. В. Геометрия на плоскости. Теория, задачи, решения : в 2 частях / В. В. Амелькин В. Л. Рабцевич. – Мозырь : Белый Ветер, 2015. – Ч. 1. – 288 с. – Текст : непосредственный.

7. Документация Python math // Python https://docs.python.org/3/library/math.html (date of application: 30.01.2021).