

ИСПОЛЬЗОВАНИЕ НЕЙРОННОЙ СЕТИ ДЛЯ РАБОТЫ С ВИДЕОФАЙЛАМИ

Иванов Вадим Олегович

Санкт-Петербургский политехнический
университет Петра Великого
vadim1999ivanov@gmail.com

Амосов Владимир Владимирович

К.т.н., доцент, Санкт-Петербургский
политехнический университет Петра Великого
amosov_vv@spbstu.ru

Коликова Татьяна Всеволодовна

Старший преподаватель, Санкт-Петербургский
политехнический университет Петра Великого
kolikova_tv@spbstu.ru

USING A NEURAL NETWORK TO WORK WITH VIDEO FILES

**V. Ivanov
V. Amosov
T. Kolikova**

Summary. The stages of developing a program for searching for a car in a video sequence and for tracking an object in the frame are described. The search for the car and the tracking of the object in the frame was carried out using a neural network. This article describes the methods of working with video files. The types and constructions of a neural network are considered.

Keywords: neural network, method, video file, video sequence, frame, Keras, Connect, Mobile Net, OpenCV, data set.

Аннотация. Описываются этапы разработки программы для поиска автомобиля в видеоряде и для отслеживания объекта в кадре. Поиск автомобиля и отслеживание объекта в кадре осуществлялся с помощью нейронной сети. В данной статье описываются методы работы с видеофайлами. Рассматриваются виды и построения нейронной сети.

Ключевые слова: нейронная сеть, метод, видеофайл, видеоряд, кадр, Keras, ConvNet, MobileNet, OpenCV, дата-сет.

Введение

В работе описан процесс создания нейронной сети и рассмотрены аналоги похожих сетей. Также рассмотрен алгоритм работы нейронной сети с видеофайлами, её структура и какие средства для этой работы понадобились. Будут представлены различные способы обучения нейронной сети и составления дата-сета для обучения.

1. Постановка общей проблемы

В современном мире каждая дорога оснащена видеокамерами, но зачастую поиск автомобилей происходит без их участия. Для решения данной проблемы разрабатываются множество средств поиска автомоби-

лей. Одним из таких средств является нейронная сеть. Предлагаемая в данной работе нейронная сеть решает эту проблему, обеспечивая поиск объекта, улучшение качества изображения объекта из видеофайла и возможность отслеживания объекта.

2. Рассмотрение известных методов решения

В одном из методов решения по результатам исследований был предложен подход, использующий в качестве классификатора рекуррентную нейронную сеть (Recurrent neural network; RNN) с долгой краткосрочной памятью (Long short-term memory; LSTM), одним из преимуществ которой является способность обработки длинных последовательностей информации. [1]

В другом методе его авторы представили систематический обзор литературы (SLR) по обработке видео с использованием DL. Они исследовали приложения, функциональные возможности, методы, наборы данных, проблемы и вызовы, и сформулировали соответствующие исследовательские вопросы (RQS). Они классифицировали технику глубокого обучения для обработки видео, основанную на CNN, DNN и RNN. [2]

Ещё в одном методе рассмотрены основные способы, на которых строятся системы распознавания данных. В качестве основного метода был выбран нейросетевой подход. Разработанный аппаратно-программный комплекс распознавания изображений и образов для управления контрольно-пропускным пунктом позволил достичь точности распознавания до 80%. Результаты были получены на тестовых образцах фотоизображений. [3]

3. Постановка задач

Целью, данной работы является разработка нейронной сети, позволяющей улучшить качество видеофайла, нахождение и слежением за объектом в кадре.

Требуется построить такую нейронную сеть, которая сможет различать между собой большое количество объектов в видеоряде. Необходимо оптимизировать данную нейронную сеть для того, чтобы она эффективно и быстро могла обучаться.

Требуется, чтобы нейронная сеть работала в двух режимах (обучения, распознавания). В режиме обучения она использует ConvNet для распознавания объекта в тестовом режиме и улучшения качества видеоряда из базы данных, а также сохраняет результаты обучения.

В режиме распознавания, нейронная сеть, использует сохранённые результаты обучения для быстрого и точного определения вида объекта в улучшенном качестве в видеоряде.

4. Структура нейронной сети

1.1. Порядок составления дата-сета для обучения

Для этого используется набор данных DIV2K, в котором содержится несколько раг файлов, содержащих кадры в хорошем и плохом качестве.

1.2. Генератор данных

Если используется генератор изображений от Keras, то это ImageDataGenerator “поток” изобра-

жений из классифицированных каталогов. Он прост в использовании, и можно быстро увеличить объем данных (манипулирование изображениями для получения большего количества образцов в «ряду»). Такой генератор приемлем для распознавания “одиночных” изображений, но не для последовательности кадров, так как:

1. Генератор изображений выдает данные (N, W, H, C), где N-размер пакета, W и H-ширина и высота, а C-количество каналов (3 для RGB, 1 для изображений в оттенках серого) .

Но нам нужно отправить последовательность (несколько кадров). Необходимая форма (N, F, W, H, C) — где F-количество кадров для нашей последовательности. Например, если мы обучим последовательность из 5 изображений, которые являются RGB и имеют размер 112x112, форма должна быть (N, 5, 112, 112, 3).

Генератор данных изображений от Keras не может создать такой формат.

2. У каждого видео не одинаковое количество кадров. Если видео имеет длину 5 секунд и кодируется со скоростью 24 кадра в секунду, таким образом, для всего видео имеется $5 \times 24 = 120$ изображений. Но другие видео могут быть длиной 10 секунд или 2 секунды. [4] [5]

В следующем разделе будет описано как можно модернизировать данный генератор под нужды нашей нейронной сети.

1.3. Методы работы с набором данных

Для этого рассмотрим методы работы с данными в дата-сете. Первый метод состоит в том, чтобы получить 5 первых кадров из 0 секунды, затем из 1 секунды и так далее.

Это возможно только в том случае, если есть уверенность, что “действие”, которое нужно предсказать, происходит в этих 5 кадрах. Но это бывает редко. У каждого набора данных из набора кадров в видео, которое используется, есть немного (или много) времени в начале и в конце видео, где “не происходит никаких действий”. [6]

Второй метод состоит в выборе 5 распределенных кадров из всего видео.

Это более эффективно, потому что, если в первом или последнем кадре видео нет действия, то нужно будет избегать их. Другое преимущество заключается

в том, что теперь есть все движения в последовательности.

Единственное, что нужно сделать, это подготовить пользовательский генератор (который расширяется Sequence), который может:

- ◆ прочитать видео (с использованием OpenCV)
- ◆ выбрать N кадров
- ◆ возвратом пакетов с кадрами

Составленный генератор может извлечь некоторые кадры, изменить их размер, создать пакет. Он “подгоняет” модель под время обучения, также создает данные проверки, разделяя входной набор данных.

Генератор использует OpenCV для захвата видео. И поскольку нам не нужно тренироваться со всем набором кадров, он знает, как извлечь часть кадров из видео.

Составленный генератор видеок кадров может использовать ImageDataGenerator для получения набора кадров с расширенным набором данных из входных видео.

Наконец, генератор также может сохранять не дополненные исходные кадры в кэше и пытаться выбирать кадры, избегая первых, чтобы быть уверенным, что в кадрах “что-то происходит».

Что касается кэша кадров, то он заполняется при запуске первой фазы обучения. Затем, когда генератор обработает все видео, он будет выбирать кадры из памяти вместо повторного чтения видеофайла. [7] [8] [9]

1.4. Принцип работы генератора

Создадим набор данных из базы данных DIV2K, содержащий только 2 класса из предоставленных файлов rag.

Создаем каталог с именем “input” с 2 подкаталогами, названными именами классов. Теперь можно использовать генератор, входящий в комплект библиотеки keras.

- ◆ генератор может производить увеличение данных
- ◆ он создает проверочный набор данных — здесь 33% видео, отобранных из набора данных
- ◆ он может использовать “кэш кадров”, который должны «деактивировать», если у компьютера небольшой объем оперативной памяти.

В результате Train и valid генераторы могут генерировать последовательности кадров. Здесь используют 5 кадров с размером пакета до 8. Это означает, что

генератор создает 8 последовательностей из 5 кадров, выбранных из видео. [10]

1.5. Процесс обучения нейронной сети

Есть несколько способов построения модели. Используем известную модель, которую внедряем в распределенный по времени слой, или можем создать собственную. [11]

1.5.1. Пользовательская сеть ConvNet

Каждое входное изображение последовательности должно передаваться в сверточную сеть. Цель состоит в том, чтобы обучить эту модель для каждого кадра, а затем решить, какой класс выводить.

Концепция:

- ◆ Создается сеть ссылок и распределяется время для обнаружения “функций”
- ◆ Распределенный по времени вывод вводится в GRU или LSTM для обработки “временных рядов”.
- ◆ Затем для принятия решения применяется плотная сеть, чтобы “классифицировать” кадры с хорошим качеством и плохим для обучения на них.

Нужно создать сеть ConvNet. Создадим функцию, которая возвращает модель. Здесь используется простая модель. Было сделано несколько сверток с пакетной нормализацией.

Этой простой модели достаточно для обнаружения функций. Последний слой создает одно измерение для вывода, также можно использовать Flatten для преобразования формата изображений из 2d-массива (28 на 28 пикселей) в 1d-массив из $28 * 28 = 784$ пикселей, но GlobalMaxPool2D (операция глобального максимального объединения пространственных данных) уменьшит количество выходов (получая только максимальные значения из последней свертки). [12]

Теперь пришло время построить распределенную по времени модель с помощью LSTM (сети долгой краткосрочной памятью), это очень эффективно, но было применено GRU, так как он концептуально такой же, но использует меньше памяти.

Модель действий строит ConvNet сначала только с (112, 112, 3), shape [1:] удаляет индекс измерения “5” — затем он вводит его в TimeDistributed слой. Этому слою нужна полная форма (5, 112, 112, 3) в качестве входных данных. Это означает, что на входе “модель” требуется 5 изображений 112x112 с 3 каналами (RGB). Так как набор данных не очень велик, будет использо-

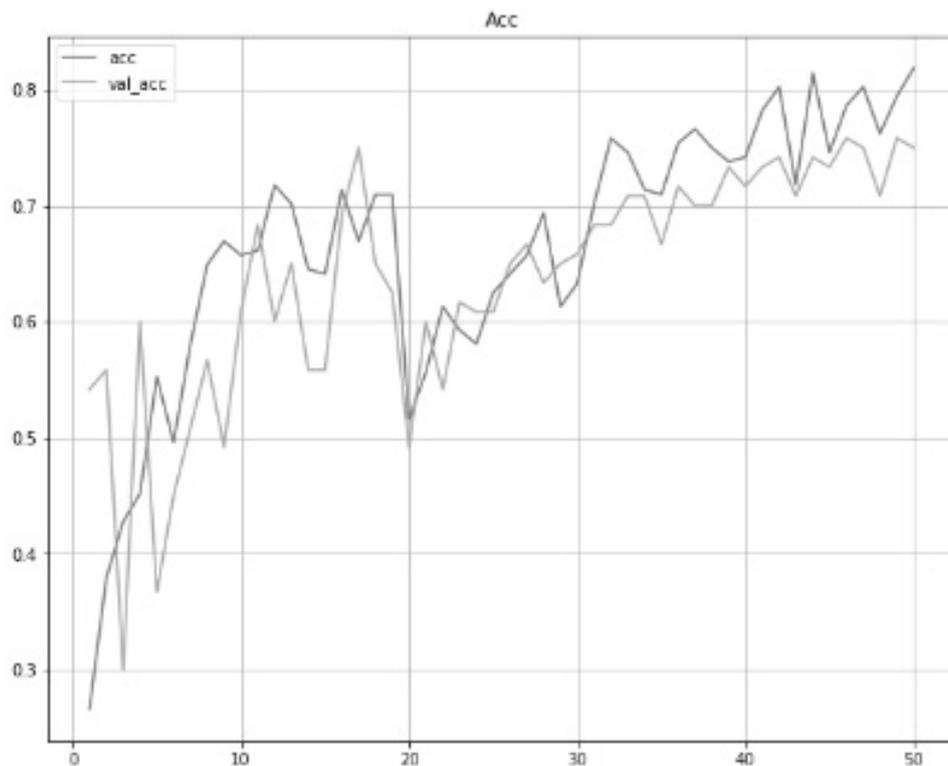


Рис. 1. График точности

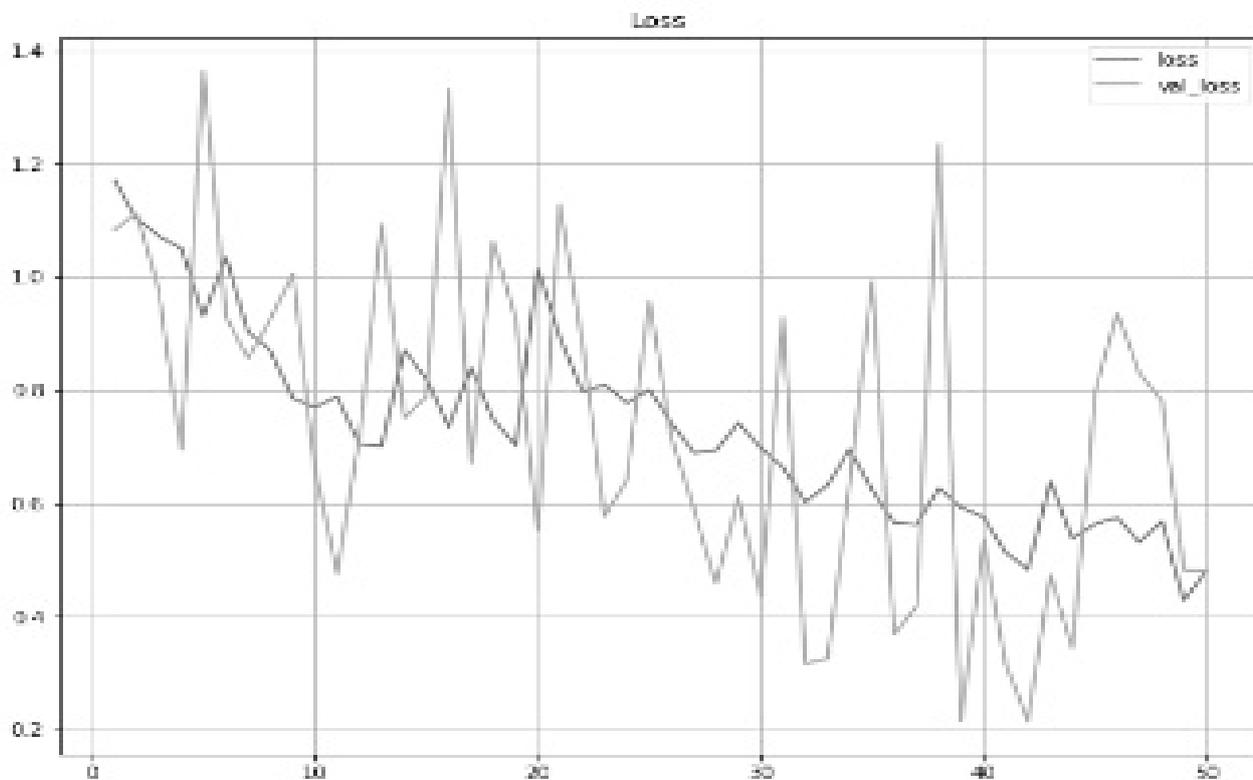


Рис. 2. График потерь

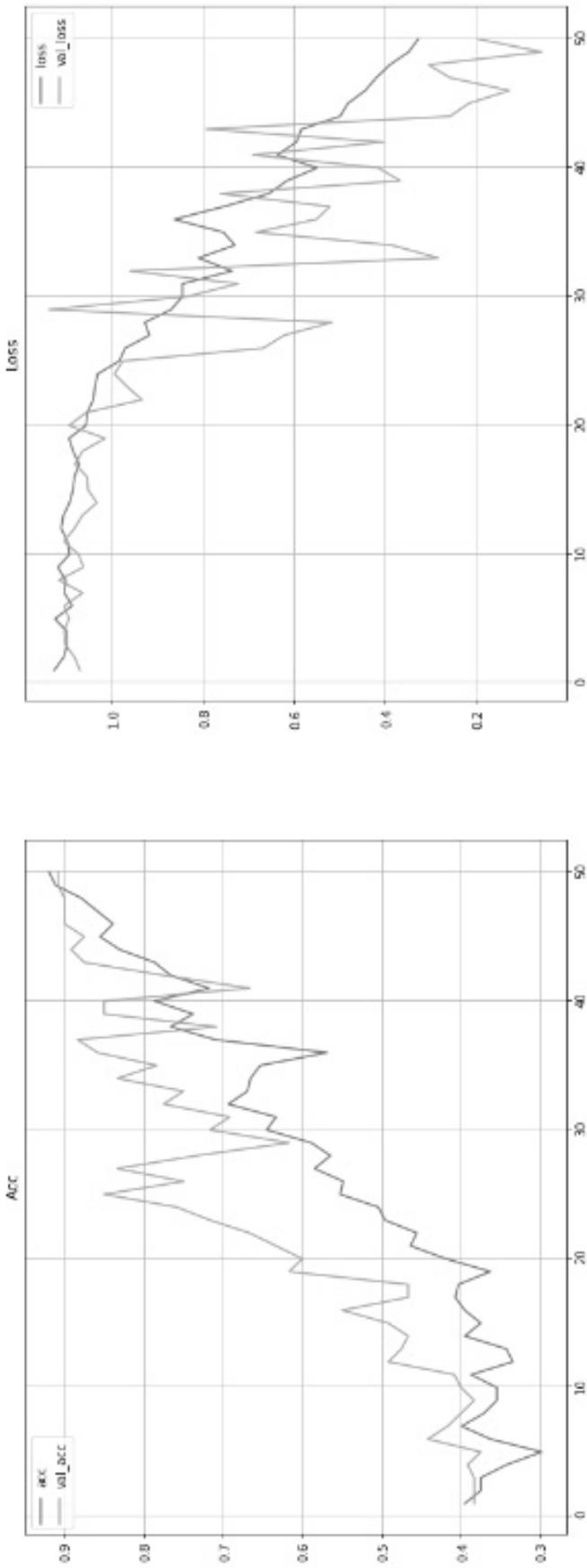


Рис. 3. График точности Рис. 4. График потерь

ваться Dropout, чтобы избежать переобучения. В дальнейшем устанавливаем настройки генератора. [13]

На этом этапе есть:

- ◆ Train и valib генераторы
- ◆ валидаторы предпочитают использовать класс последовательности, поэтому можно применить их с fit generator способ из Кераса
- ◆ model — время распределено

Теперь установим настройки генератора и выведем графики (рис 1. и рис 2.), чтобы проверить, как развивается модель.

На графике (Рис 1.) можно наблюдать то, что точность растёт, но не постоянно. Теперь посмотрим, что происходит с «потерями» (Рис 2.) .

Потеря обучения, кажется, в порядке, но можно наблюдать проблемы с потерей проверки. Возможные причины такого разброса являются:

- ◆ Недостаточно данных для проверки
- ◆ Недостаточно данных для обучения
- ◆ Модель, вероятно, недостаточно хорошо построена

1.5.2. Пользовательская сеть MobileNet

Используем MobileNet. Она быстра и хорошо обучается.

Нашей модели GRU (или LSTM) требуется одно измерение, но, если получим MobileNetV2 без “верхних слоев” (это выходные данные категории, они нам не нужны), будет слой “Свертки”. Нужно добавить один слой, чтобы форма одного измерения была совместима с GRU.

Кроме того, MobileNet изначально не предназначен для обнаружения “действий”, а только для распознавания изображений. Таким образом, если настроить все слои как “неподготовленные», не будет улучшений точности или “потерь”.

Поскольку используется предварительно подготовленная модель, нужно использовать требуемую форму (224, 224, 3). [14] [15]

Был построен генератор для обучения нейронной сети и выведены графики модели (Рис 3. и Рис 4.) .

Здесь потеря валидации невелика, но достаточно хороша для экспериментов.

В итоге была выбрана пользовательская сеть MobileNet из-за быстрой и хорошей обучаемости по сравнению с ConvNet. Так же она показала лучшие результаты в потерях валидации и точности.

4. Заключение

В результате разработана и исследована искусственная нейронная сеть MobileNet, для улучшения качества видеофайлов и поиска объектов на ней.

При выполнении данной работы на основе разработанной нейронной сети была создана программа для улучшения качества определённого объекта, в нашем случае видеофайлов. В дальнейшем данная нейронная сеть должна будет научиться отслеживать объект в кадре. Это приложение было создано для того, чтобы люди могли найти свой автомобиль в городе и следить за ним.

ЛИТЕРАТУРА

1. А.Ю. Буйко, А.Н. Виноградов Выявление действий на видео с помощью рекуррентных нейронных сетей // программные системы: теория и приложения. 2017, с. 327–345.
2. Palash Feddewar | Dr. Bharti Deshmukh Convolution Neural Network (CNN) for Video Processing: A Survey. 2022, с. 147–152.
3. Akulov M.S. Gladkikh S.A. Lankina M. Yu. Baklanov A.N. PROCESSING PHOTOS AND VIDEOS BY USING NEURAL NETWORKS IN THE LABVIEW PROGRAM. // Modern high technologies. — 2019. — № 3 (part 1) — P. 12–17.
4. Васенков Д.В. Методы обучения искусственных нейронных сетей // Компьютерные инструменты в образовании. — СПб.: Изд-во ЦПО «Информатизация образования», 2007, № 1, С. 20–29.
5. Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, Li Fei-Fei. Large-scale Video Classification with Convolutional Neural Networks. 2014.
6. Барский, А.Б. Логические нейронные сети: Учебное пособие / А.Б. Барский. — М.: Бином, 2013. — 352 с.
7. Редько, В.Г. Эволюция, нейронные сети, интеллект: Модели и концепции эволюционной кибернетики / В.Г. Редько. — М.: Ленанд, 2015. — 224 с
8. Суровцев И.С., Ключкин В.И., Пивоварова Р.П. Нейронные сети. — Воронеж: ВГУ, 1994. — 224 с.
9. Уоссермен Ф. Нейрокомпьютерная техника: теория и практика. — М.: Мир, 1992.
10. Галушкин, А.И. Нейронные сети: основы теории. / А.И. Галушкин. — М.: РИС, 2014. — 496 с.
11. Интеллектуальные технологии на основе искусственных нейронных сетей / М.А. Басараб, Н.С. Коннова. — Москва: Издательство МГТУ им. Н.Э. Баумана, 2017. — 53 с.

12. Солдатова О.П. Нейроинформатика. Курс лекций. СГАУ, 2013. 130 с.
13. Игнатьев Н.А., Мадрахимов Ш.Ф. О некоторых способах повышения прозрачности нейронных сетей // Вычисл. технологии. 2003. Т. 8, № 6. С. 31–37.
14. Yann LeCun, J.S. Denker, S. Solla, R.E. Howard and L.D. Jackel: Optimal Brain Damage, in Touretzky, David (Eds), Advances in Neural Information Processing Systems 2 (NIPS*89), Morgan Kaufman, Denver, CO, 1990
15. Y. LeCun, L. Bottou, G. Orr and K. Muller: Efficient BackProp, in Orr, G. and Muller K. (Eds), Neural Networks: Tricks of the trade, Springer, 1998.

© Иванов Вадим Олегович (vladim1999ivanov@gmail.com),

Амосов Владимир Владимирович (amosov_vv@spbstu.ru), Коликова Татьяна Всеволодовна (kolikova_tv@spbstu.ru).

Журнал «Современная наука: актуальные проблемы теории и практики»



Санкт-Петербургский политехнический университет Петра Великого