Сеть готова, что дальше?

Лекция

Материал лекции размещен -
https://github.com/d-yacenko/NN_to-_production_impl.git

Цель

Создание алгоритма использующего NN. Цель и задачи.

- прикладная цель — решение конкретной технической проблемы с характеристиками решения не хуже существующих аналогов. В этой ситуации на выходе должно быть прикладной продукт — готовое к использованию в целевом юзкейсе ПО или программно-аппаратный комплекс
- демонстрация — например:
  - концепт демонстрирующий потенциально возможное продуктовое решение (стартап),
  - концепт демонстрирующий скилы создателя — портфолио, учебные цели - индивидуальны проект
  - научная работа, лишь демонстрация идей или возможностей, концепт не обязателен. Реализация предмета исследования — научная новизна/учебный продукт. В качестве продукта научная статья или учебные материалы.

Достижение

Вспомним, что работа с НС это learning/inference, соответственно и работа с сетью разделяется на две части. Во всех случаях кроме последнего (рассмотрим чуть позже) требуется реализация двух разных решений — (1) разработка алгоритма NN, решающего поставленную задачу со всеми сопутствующими условиями см. первую лекцию , (2) разработка прикладного приложения, соответствующего требованиям к ПО.

(1) Первая стадия состоит из:

- проектирования компонентов нейронной сети: Архитектура НН, Функция потерь, Метод оптимизации, Метрики
- реализации сети и ее обучении: выбор библиотек, создание пайплайна, выполнение процесса обучения на пайплайне, анализ полученных результатов, корректировка сети/рассмотрение новых гипотез
- фиксации результатов в виде документации, в виде опубликованного проекта, экспорт сети для дальнейшего использования

Примечание 1: первая стадия при реализации научной работы должна иметь признак научной новизны, теоретической значимости исследования, и должна удовлетворять критериям научности.

Примечание 2: первая стадия при реализации образовательной работы (методичка, пособие, учебный курс, демонстрационные проекты) в основном служит для демонстраций алгоритмов НС.

(2) Вторая стадия состоит из:

- анализа постановки прикладной задачи и выбор технологической платформы для реализации
- проектирование и разработка ПО использующего сериализованную НС для инференса по фактическим данным.

- имплементация прикладной системы как ПО и/или библиотек/фреймворков и/или сервисов и/или программно аппаратных комплексов

Примечание 1: Вторая стадия для случая научной работы — публикация статьи в научном журнале, участие в конференции.
Примечание 2: Вторая стадия для случая образовательной работы — публикация и внедрение образовательного материала.

Пример

Пример реализации цели «Индивидуальный проект IT Академия Samsung трек ИИ определение пожара на фотографии»
Поскольку эта цель подпадает под (портфолио, учебные цели) задачи:

1. получить датасет, спроектировать, реализовать, обучить НС, анализ и отчет,
2. спроектировать, реализовать, опубликовать концепт.

Реализуем первую стадию.

Проектирование.

Датасет.
Получаем датасет со следующими параметрами:

- две части train(80)/val(20)
- два класса фотографий fire(91)/nofire(138)
- состав — фото из разных источников 3872x2592 — 300x214

Вывод:

- датасет слишком маленький для полноценного обучения,
- датасет перекошенный,
- требуется предобработка фотографий — размер, обогащение.

Сеть.
Компоненты:

- Т.к. датасет маленький, используем transfer-learning - берем предобученную сеть Resnet18 (обучен на ImagNet ~15M) без fine-tunig
- К выходу сверточных слоев ResNet (признаковое описание ImageNet) подключаем полносвязный слой 512x2
- Некоторый трюк - CrossEntropyLoss
- SGD
- Метрики — Loss и Accuracy

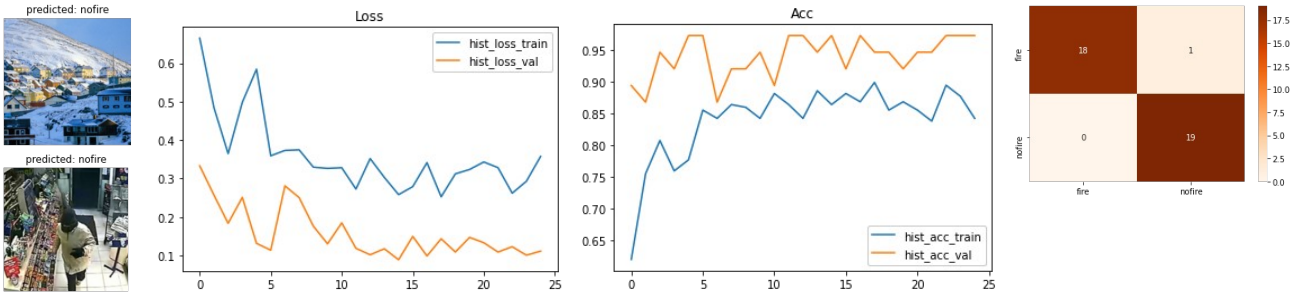
Реализация.

- пайплайна фактически нет, т.к. предполагаем что угадаем с первого раза.
- среда исполнения ноутбук Google Colab,
- фреймворк pytorch
- предобученная модель — torchvision.models.resnet18

Демонстрация

Результаты.

Training complete in 2m 0s
Best epoch:14 val Loss:0.088741 Acc: 0.973684
На таком маленьком датасете сеть обучилась очень быстро (обучали то два нейрона), точность 97% - выглядит неплохо, хотя на таком не репрезентативном датасете недостоверна и валидация. По хорошему нужно собрать отдельные данные для оценки достоверности результатов.



Забираем сериализованную модель — файл **Fires.pt**

Реализуем вторую стадию.

Анализ постановки прикладной задачи

- Если задача научная или учебная, на этой стадии оформляют весь собранный на предыдущих этапах материал в виде статьи/метод.материала с учетом требований. В качестве демонстрации можно рассмотреть любую статью в научном журнале [1]
- Если речь о концепте необходимо продумать форму удобной для представления, но с учетом минимальной функциональности и минимальной трудоемкости. Например чат-бот телеграмм. Демонстрация, Приложение 2.
- Если требуется разработка прикладного решения нужно рассмотреть требованию к решению и параметры результата, получившегося на первом этапе :
  - параметры инференса на полученной сети — время на один цикл, среда исполнения,
  - требования по скорости инференса,
  - требования по оперативной памяти,
  - по системе вычислений (TOPS/FLOPS),
  - по программной платформе,
  - архитектура целевой системы,
  - требования безопасности.

и выбор технологической платформы для реализации

- проектирование и разработка ПО использующего сериализованную НС для инференса по фактическим данным.

Литература

1. «TF-IDF vs Word Embeddings for Morbidity Identification in Clinical Notes: An Initial Study» Danilo Dess, Rim Helaoui, Vivek Kumar1, Diego Reforgiato Recupero and Daniele Riboni University of Cagliari, Cagliari, Italy{danilo dessi, vivek.kumar, diego.reforgiato, riboni}@unica.it Philips Research, Eindhoven, Netherlands rim.helaoui@philips.com
Нейронные сети и компьютерное зрение, Samsung Research Russia Open Education - 2019
Нейронные сети и обработка текста, Samsung Research Russia Open Education -2019
Bastiaan Sjardin, Luca Massaron, Alberto Boschetti - Large Scale Machine Learning with Python — 2016

Приложение 2.

Пример чат-бота.

app.py

```
# https://core.telegram.org/bots#
# https://pytorch.org/get-started/locally/
# pip3 install opencv-python
# pip3 install PyTelegramBotAPI==2.2.3
import telebot
import traceback
import torch
from torchvision import transforms
import config
from handler import *

bot = telebot.TeleBot(config.TOKEN)
classes=['fire','nofire']
model = torch.jit.load('fire_net.pt')
transform = transforms.Compose(
    [transforms.Resize(224),
     transforms.ToTensor(),
     transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])])

def get_photo(message):
    photo = message.photo[1].file_id
    file_info = bot.get_file(photo)
    file_content = bot.download_file(file_info.file_path)
    return file_content

@bot.message_handler(commands=['start'])
def start_message(message):
    bot.send_message(message.chat.id, 'Привет! Пришли фото сюда, а нейронная сеть определит наличие объекта.\nАвтор: @d-yacenko')

@bot.message_handler(content_types=['photo'])
def repeat_all_messages(message):
    try:
        file_content = get_photo(message)
        image = byte2image(file_content)
        image=transform(image)
        model.eval()
        image=torch.unsqueeze(image, 0)
        outputs = model(image)
        _, preds = torch.max(outputs, 1)
        bot.send_message(message.chat.id,text='Обнаружено: {}'.format(classes[int(preds)]))
    except Exception:
        traceback.print_exc()
        bot.send_message(message.chat.id, 'Упс, что-то пошло не так :( Обратитесь в службу поддержки!')

if __name__ == '__main__':
    import time
    while True:
        try:
            bot.polling(none_stop=True)
        except Exception as e:
            time.sleep(15)
            print('Restart!')
```

config.py

```
TOKEN='1645233456:klsdfmg kfjgn#$eer$#%^WE$RTzxcvbxcU'
```

handler.py

```
import cv2
from PIL import Image
import numpy as np

def byte2image(byte):
    arr = np.frombuffer(byte, dtype=np.uint8)
    image = cv2.imdecode(arr, cv2.IMREAD_COLOR)
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    im_pil = Image.fromarray(image)
    return im_pil
```