МОВС ВШЭ

Дипфейк-детектор: Первая Половина Пути

Годовой проект магистратуры на тему

“Определение дипфейков по фотографии”.

Выполнил:

Тимур Ермешев

Куратор:

Вячеслав Пирогов

Москва 2024

Введение

Всем нам знакомо увлечение подрисовкой усов на картинках в детстве, будь то учебник или афиша. С развитием компьютеров мы переключились на фотошоп, внося изменения в изображения. Даже вырезание и приклеивание чьего-то лица на другое изображение казалось безобидной шуткой. Но с появлением глубоких нейронных сетей создание фейков стало более изощренным.

Люди научились подделывать звук, видео и картинки на уровне, неотличимом для невооруженного глаза. Технологии генерации контента достигли высокого уровня, позволяя легко создавать изображения, неотличимые от реальных фотографий. Синтез голоса актеров стал настолько точным, что озвучивание персонажей фильмов стало делом прошлого.

Однако, с ростом возможностей по созданию реалистичных фейков, появляется потенциальная угроза. Дипфейки, искусственные подделки, проникают в сферу создания фальшивых видео с участием известных личностей. Актеры, политики, предприниматели - все подвержены этому новому виду манипуляций.

Сегодня мы сфокусируем внимание на инновационном решении - детекторе дипфейков. Давайте вместе исследуем, как этот инструмент обеспечивает безопасность и достоверность в мире, где граница между реальностью и виртуальным становится все более размытой.

Что такое Дипфейк

Технология Deepfake представляет собой уникальный процесс синтеза изображений и создания звуковых дорожек с установленными параметрами, осуществляемый при помощи нейронных сетей. Они обучаются на обширных наборах данных, включающих сотни или тысячи примеров лиц и соответствующих голосов.

Source Target Result

Своё название технология получила в 2017 году от пользователя Reddit, который создал несколько видео с использованием лиц знаменитостей. Подписавшись под никнеймом "Deepfake", этот пользователь стал вдохновением для присвоения термина этим инновационным технологиям.

Глубокое обучение нейронных сетей позволяет создавать впечатляющие и реалистичные подделки, перенося лица и голоса на новые контексты. Такие технологии, несмотря на свою потенциальную опасность, продолжают эволюционировать, вызывая важные обсуждения в области кибербезопасности и этических вопросов.

План работ

1. Первый этап.
- Создать репозиторий.
- Подобрать готовые датасеты для работы и/или подобрать методы для генерации датасета.
- Произвести анализ фотографий из датасета.
- Подготовить скрипт по оценке фотографий на чтение, произвести очистку при необходимости.

2. Второй этап (ML)
- Подготовить функцию для перевода картинки в вектор.
- Произвести выбор модели для бинарной классификации картинок (дипфейк или нет).
- Произвести обучение для выбранной модели/моделей.

3. Третий этап (DL)
- Выбор архитектуры предобученной нейросети.
- Создать функцию для загрузки изображений.
- Создание функции для обучения, валидации и предсказания модели.
- Обучение и тестирование модели

4. Четвертый этап (DL)
- Другие методы определения дипфейков.
- Трансформеры

5. Пятый этап
- Создание веб-сервиса/ТГ бот, в который пользователь загружает фотографию и получает в ответ решение о том, является ли картинка дипфейком или нет.

Датасет

Для данной работы использовались 2 датасета с видео файлами.

- FaceForensics++
  - 3068 manipulated videos.
  - 363 original source actor videos.
- Celeb-DF
  - 590 original videos collected from YouTube
  - 5639 corresponding DeepFake videos

Для извлечения картинок из видео и разделения датасетов написан скрипт (frames_from_video.py), который:

- Делит видео на на train, val и test в соотношении (60:20:20)
- Извлекает 4 картинки из каждого видео случайным образом.
- Создает уникальное имя картинке модулем uuid, добавляя к нему имя датасета, лейб, порядковый номер картинки.
- Раскладывает картинки по директориям, с соответствующими названиями.
- Записывает новые имена файлов, лейблы, название фазы обучения в dataset_names.csv

Для подсчета Mean, Variance, Standard deviation написана отдельная функция (get_variance) в ноутбуке EDA.ipynb.

При высоких показателях метрики модель может быть испытана на другом датасете. Например с ресурса Roboflow

Примеры картинок

Обработанные картинки

faceforensics_fake_041708b34a-3.jpg
resolution 1920x1080

celebdf_2_fake_a775eb622d-3.jpg
resolution 944x500

Не обработанные картинки

celebdf_2_real_8079bab1d1-3.jpg
resolution 782x440

faceforensics_real_15183f504f-0.jpg
resolution 1920x1080

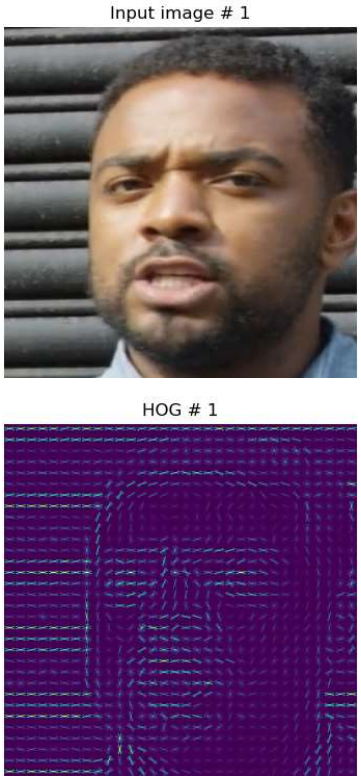
Подготовка фотографий

Класс ImageProcessing в файле ml_functions.py:

- считывает картинки из заданной директории;
- создает для них лейблы;
- переводит в черно белый вариант (при необходимости);
- вырезает лицо методом HAAR Cascade (при необходимости);
- изменяет размер на заданный;
- get_hog получает маску с направленными градиентами (Histogram of Oriented Gradients, HOG).
- на выходе выдает список картинок, лейблов, лиц, hog, links.

Функция convert_data

- Переводит каждую картинку списка в вектор и кодирует лейблы в соответствии с поданным словарем



ML модель

Для задачей ML были проведены тесты для различных размеров картинок, с вырезанным лицом, цветных и черно белых картинок.

Подбор варианта для обучения на базовых фото

- Вырезать лицо? – не вырезать;
- Оставлять цветным? - сделать серым;
- Какой выбрать размер? – 144x144 пикселей.

Выбор моделей:

- LogisticRegression
- LGBMClassifier
- RandomForestClassifier

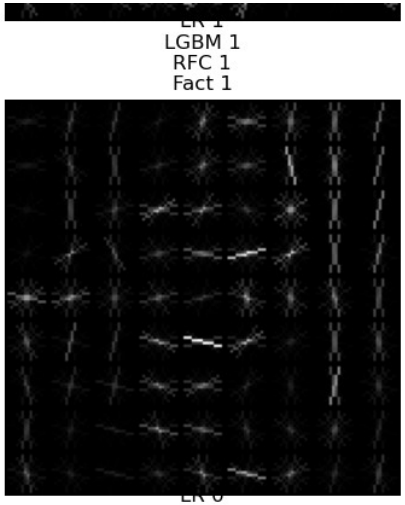
Среднее значение метрики для моделей LGBMClassifier и RandomForestClassifier для базовых фотографии и HOG маск на тестовых данных получилось 0.715

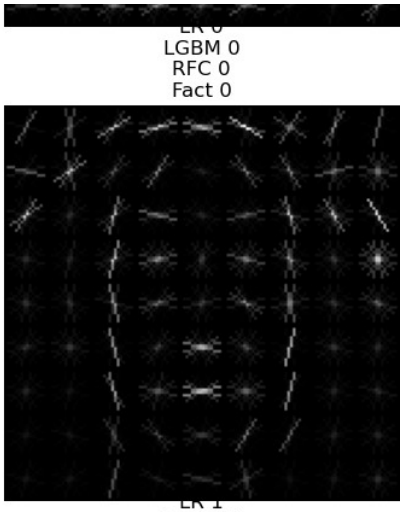
Base picture		face_crop	
		True	False
is gray	True	0.548	0.661
	False	0.59	0.641

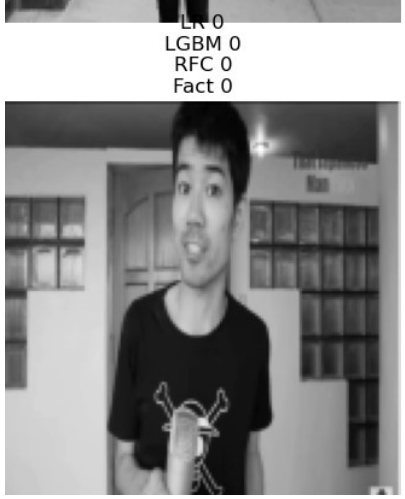
Size	Score base	Score HOG
256	0.617	0.58
224	0.633	0.614
192	0.614	0.58
176	0.639	0.596
160	0.655	0.564
144	0.668	0.62
128	0.607	0.62
112	0.643	0.606
96	0.617	0.592
80	0.653	0.603
64	0.647	0.589
32	0.625	0.601

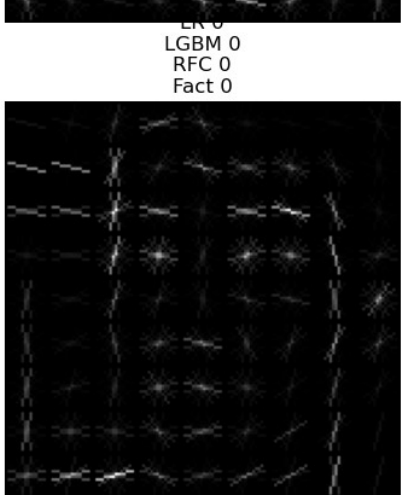
	Base	HOG
LogisticRegression	0.624	0.628
LGBMClassifier	0.654	0.69
RandomForestClassifier	0.655	0.645

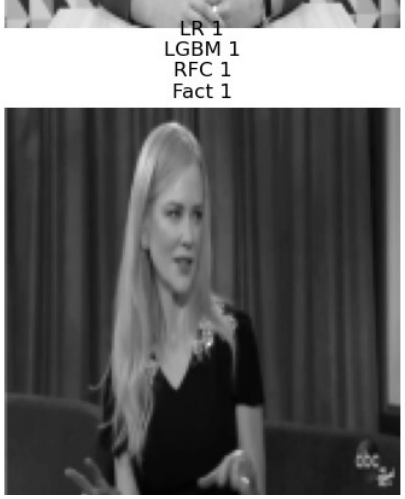
ML результаты

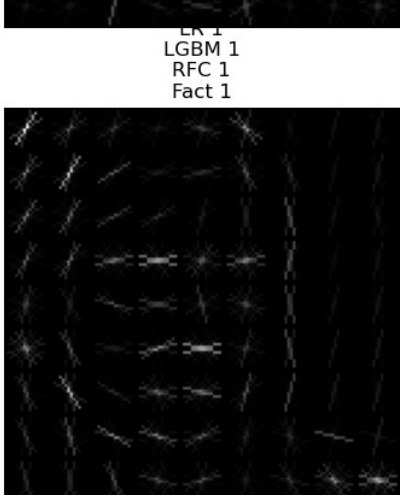












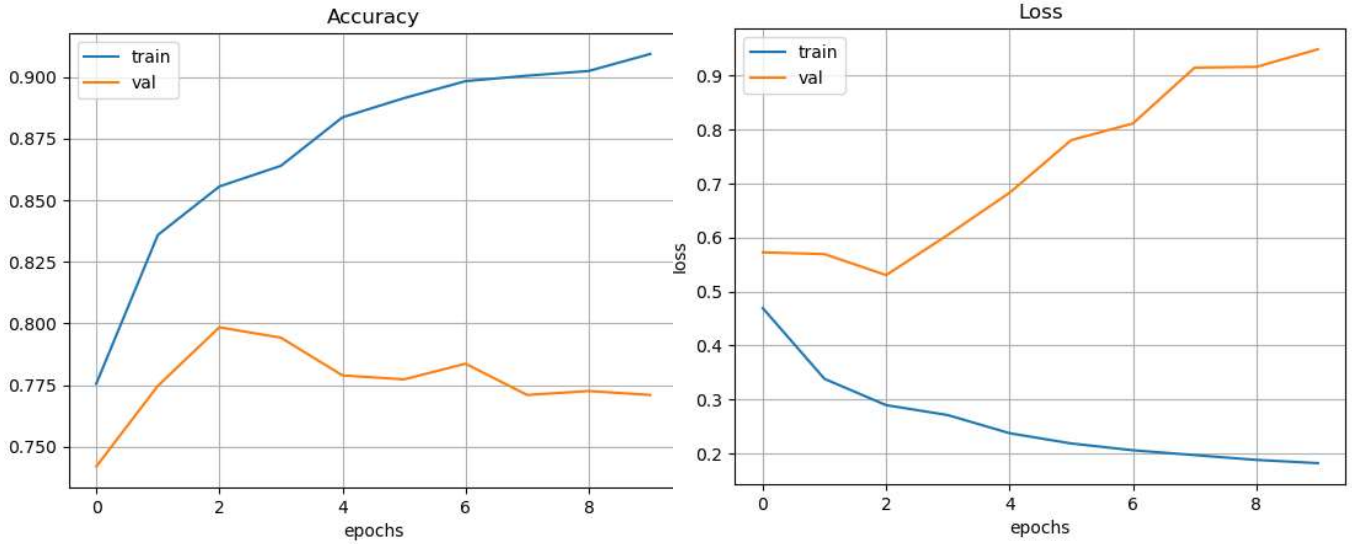
DL модель (предобученная)

На данном этапе выбрана предобученная модель EfficientNet_V2_M с весами IMAGENET библиотеки torchvision, которая имеет довольно высокую метрику классификации и не очень тяжелая.

Weight	Acc@1	Acc@5	Params	GFLOPS	Recipe
AlexNet_Weights.IMAGENET1K_V1	56.522	79.066	61.1M	0.71	link
ConvNeXt_Base_Weights.IMAGENET1K_V1	84.062	96.87	88.6M	15.36	link
EfficientNet_B7_Weights.IMAGENET1K_V1	84.122	96.908	66.3M	37.75	link
EfficientNet_V2_L_Weights.IMAGENET1K_V1	85.808	97.788	118.5M	56.08	link
EfficientNet_V2_M_Weights.IMAGENET1K_V1	85.112	97.156	54.1M	24.58	link
EfficientNet_V2_S_Weights.IMAGENET1K_V1	84.228	96.878	21.5M	8.37	link
GoogLeNet_Weights.IMAGENET1K_V1	69.778	89.53	6.6M	1.5	link
Inception_V3_Weights.IMAGENET1K_V1	77.294	93.45	27.2M	5.71	link

Написан class ModelTrainer в файле dl_functions.py, который производит обучение модели, валидацию, тестирование и предсказание.

Также написана функция transform_image для подготовки одного изображения для подачи на получение предсказаний.



Сервис

Для данного этапа реализован простой телеграмм бот, который на данный момент запускается локально с моего ноутбука.

При достижении высокой точности предсказаний я рассматриваю для себя 2 реализации:

- Web-страница: Этот вариант будет удобен для тех, у кого есть доступ в интернет. Вы видите возможность создания одной страницы с функционалом визуализации и вывода промежуточных результатов и изображений.

- Телеграмм бот: Этот вариант будет удобен для всех, у кого есть телефон с установленным приложением Telegram, независимо от модели телефона или планшета. Бот будет функционировать в режиме вопрос-ответ, принимая изображение и предоставляя вероятность того, что данное изображение является дипфейком.

Структура проекта

```
DeepFake_Detector
├── datasets
│   ├── frames
│   │   ├── test/
│   │   ├── train/
│   │   └── val/
│   ├── videos
│   │   ├── celebdf_2/
│   │   └── faceforensics/
├── models
│   ├── DL/
│   └── ML/
├── notebooks
│   ├── eda
│   │   └── EDA.ipynb
│   ├── train_models
│   │   ├── DL_model.ipynb
│   │   └── ML_model.ipynb
│   └── Get_pictures_from_video.ipynb
├── presentations/
├── src
│   ├── faceforensics_dataset_downloader
│   │   ├── faceforensics_download_v4.py
│   │   └── README.md
│   ├── dl_functions.py
│   └── ml_functions.py
├── tg_bot/
├── .env
├── .gitignore
├── dl_train.py
├── frames_from_video.py
├── ml_train.py
└── README.md
```

`datasets/`
- `frames/` - test, train and val frames with real and fake classes
- `videos/` - celebdf_2 and faceforensics++ video datasets

`models` - trained best ML and DL models

`notebooks/`
- `eda/EDA.ipynb` - notebook with exploratioin data analisys of the datasets.
- `train_models/` - notebooks with dataset preprocessing and training ML/DL models
- `Get_pictures_from_video.ipynb` - notebook for splitting datasets to train, test, val. Extracting frames from videos.

`presentations/` - checkpoint presentations

`src/`
- `faceforensics_dataset_downloader/` - faceforensics dataset downloader with readme file for running .py file.
- `dl_functions.py` - functions loading pictures, preprocessing and training DL models.
- `ml_functions.py` - functions loading pictures, preprocessing and training ML models.

`tg_bot/` - files, associated with tg bot

`.env` - environment constants

`dl_train.py` - download pictures, transform, train and save DL model

`frames_from_video.py` - splitting dataset, extracting frames from videos, save pictures

`ml_train.py` - download pictures, transform, train and save ML models

`README.md` - project description

Заключение

В заключение данной презентации хочу подчеркнуть важность и значимость проделанной работы в создании системы определения дипфейков. Мы успешно завершили первую половину комплексного подхода, начиная с формирования репозитория, подготовки датасетов и анализа фотографий. Пройдя через этапы подготовки данных и обучения моделей машинного обучения, мы достигли успешного завершения процесса обучения.

Обученные модели теперь способны выявлять дипфейки с определенной долей ошибки, что представляет собой важный шаг в борьбе с фальсификацией контента. Завершив этап работы, я начал внедрять систему в повседневную жизнь путем запуска Телеграмм бота, делая технологию доступной и удобной.

Я убежден, что данное исследование имеет не только академическое, но и практическое значение. В современном мире, насыщенном цифровым контентом, проблемы дезинформации и мошенничества требуют инновационных подходов, и моя работа в этом направлении может сделать важный вклад в обеспечение достоверности информации.

Спасибо

Спасибо за внимание и готов ответить на ваши вопросы.