Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

РОЗРОБКА ТА ДОСЛІДЖЕННЯ МЕТОДУ РОЗПІЗНАВАННЯ ОБЛИЧ ДЛЯ СИСТЕМ БЕЗПЕКИ НА БАЗІ РЕЗУЛЬТАТІВ ПРОВЕДЕНОГО АНАЛІЗУ СУЧАСНИХ МЕТОДІВ ДЕТЕКТУВАННЯ ТА РОЗПІЗНАВАННЯ ОБ'ЄКТІВ

Виконав:
студент 2 курсу, групи ІНФм-20-1
Ковтуненко А.Р.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Яковлева О.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Ковтуненко Андрію Романовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження методу розпізнавання облич для систем безпеки на базі результатів проведеного аналізу сучасних методів детектування та розпізнавання облич
затверджена наказом по університету від «_22_» жовтня 2021 року № 1574Ст.
2. Термін подання студентом роботи до екзаменаційної комісії _22 _листопада____ 2021 р.
3. Вихідні дані до роботи Методи детектування облич на зображенні, MTCNN, FaceBoxes, DSFD, RetinaFace, CenterFace, SCRFD, методи класифікації, SVM, FAISS, датасети для порівняння якості моделей, згорткові нейронні мережі, методи нормалізації облич.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд сучасних методів детектування.
2. Огляд сучасних методів класифікації.
3. Методи MTCNN, FaceBoxes, DSFD, RetinaFace, CenterFace, SCRFD.
4. Аналіз якості детектування обличчя при зміні кута нахилу.
5. Аналіз якості детектування при зміні розміру обличчя.
6. Аналіз швидкодії методів детектування.
7. Аналіз та вибір методів нормалізації облич.
8. Аналіз методів класифікації.
9. Створення pipeline для системи безпеки.
10. Реалізація застосунку системи безпеки на основі аналізу методів детектування, нормалізації та розпізнавання.
11. Аналіз якості застосунку.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність задачі, методи детектування, методи класифікації, результати порівняння моделей детектування, нормалізація обличчя, приклади роботи системи безпеки.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант дотримання діючих стандартів та норм	Доцент Бєлова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.10.2021	
2	Аналіз завдання, підбір літератури	22.10.21-25.10.21	
3	Аналіз літератури з досліджуваної проблеми	25.10.21-28.10.21	
4	Аналіз технічних засобів	28.10.21-02.11.21	
5	Розробка методу	02.11.21-04.11.21	
6	Програмна реалізація	04.11.21-08.11.21	
7	Оформлення пояснювальної записки	09.11.21-17.11.21	
8	Перевірка на плагіат	18.11.2021	
9	Рецензування	19.11.2021	
10	Підготовка презентації та доповіді	20.11.2021	
11	Занесення роботи в електронний архів	21.11.2021	
12	Попередній захист кваліфікаційної роботи	01.12.2021	

Дата видачі завдання __22__ __жовтня_____ 2021 р.

Студент _____
(підпис)

Керівник роботи _____ __доц. Яковлева О.В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 84 с., 1 табл., 77 рис., 2 дод., 71 джерело.

ДЕТЕКТУВАННЯ ОБЛИЧ, РОЗПІЗНАВАННЯ ОБЛИЧ, НОРМАЛІЗАЦІЯ ОБЛИЧ, EMBEDDINGS, LANDMARKS, SVM, FAISS, WIDER FACE, ДАТАСЕТ, ПОРІВНЯННЯ ШВИДКОДІЇ, CNN, MTCNN, FACEBOXES, DSFD, RETINAFACE, CENTERFACE, SCRFD, ARCFACE.

Робота присвячена розробці та дослідженню методу розпізнавання облич для систем безпеки на базі результатів проведеного аналізу сучасних методів детектування та розпізнавання об’єктів.

Метою дослідження існуючих методів детектування та класифікації було обрання таких, які якнайкраще задовольняють вимогам систем безпеки для середніх підприємств, та створення із обраних методів pipeline для вирішення задачі розпізнавання облич у реальному часі. Для порівняльного аналізу було обрано методи детектування: MTCNN, FaceBoxes, DSFD, RetinaFace, CenterFace, SCRFD та методи класифікації SVM та FAISS. Для проведення досліджень були створені власні датасети та використані існуючі.

За результатами досліджень було розроблено програмних застосунок розпізнавання облич для системи безпеки підприємства. Було використано мови: C++, JavaScript; бібліотеки: OpenCV, React.

FACE DETECTION, FACE RECOGNITION, FACE NORMALIZATION, EMBEDDINGS, LANDMARKS, SVM, FAISS, WIDER FACE, DATASET, SPEED COMPARISON, CNN, MTCNN, FACEBOXES, DSFD, RETINAFACE, CENTERFACE, SCRFD, ARCFACE.

The work is devoted to development and research of the face recognition method for the security system based on the results of the analysis of up-to-date methods of object detection and recognition.

The aim of the study of present detection and classification methods was to select those that meet the requirements of security systems for medium-sized enterprises at best, and to create the pipeline for solving the problem of face recognition in real time on the basis of searched methods. For comparative analysis, MTCNN, FaceBoxes, DSFD, RetinaFace, CenterFace, SCRFD were chosen as detection methods and SVM, FAISS as classification methods. For the research, the own datasets were created and the present ones were used.

According to the results of the research, the software application of face recognition for the enterprise security system was developed. The following languages were used: C++, JavaScript; libraries: OpenCV, React.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CNN – Convolutional Neural Network

FPN – Feature Pyramid Network

RPN – Regional Proposal Network

NMS – Non-maximum Suppression

LBP – Local Binary Patterns

AP – Average Precision

mAP – Mean Average Precision

PR – Precision Recall

MTCNN – Multi-task Cascaded Convolutional Networks

DSFD – Dual Shot Face Detector

SCRFD – Sample and Computation Redistribution for Efficient Face Detection

DBSCAN – Density-Based Spatial Clustering of Applications with Noise

ARI – Adjusted Rand Index

AMI – Adjusted Mutual Information

ВСТУП

Досягнення у сфері комп’ютерного зору не стоять на місці, так само, як і прогрес у цілому. До існуючих проблем додаються нові, які, у свою чергу, потребують також нових і ефективних рішень. Вони створюються для певної галузі та проблеми, наприклад, глибоке навчання прийшло на зміну класичним методам комп’ютерного зору [1, 2], бо була потреба у збільшенні якості задачі класифікації, на той час, класичні методи вирішували її не достатньо якісно. Так само можна сказати і про задачу ідентифікації, а саме про детектування та розпізнавання облич. Проблема захисту майна, об’єкту або іншого ресурсу не нова і досі є актуальною. Для її вирішення застосовують різні комплекси мір, засобів та систем безпеки. Розвитку останніх як раз сприяв розвиток комп’ютерного зору, бо набагато зручніше встановити відео камери, та у автоматичному режимі аналізувати що відбувається. Наприклад, схожий сценарій реалізований у аеропортах, для підтримки безпеки. Він аналізує особи людей і шукає їх по базам розшуку, аналізує емоції, стан людини і багато інших характеристик.

Робота присвячена створенню системи безпеки підприємства спираючись на успіхи у задачах детектування та розпізнавання. Детектування обличчя є першим і головним кроком для аналізу та розпізнавання облич, від якого залежать подальші кроки та результати усього процесу. Вже створено безліч методів і ще буде створено, бо як було сказано раніше, задачі та умови змінюються. Для одного середовища та потреб необхідний один метод, для інших – другий. Тому необхідно проаналізувати існуючі методи, щоб зробити висновок, який із безлічі краще підходить для поставлених завдань і умов, бо завжди є взаємовиключні аспекти: якість і швидкість. Так само і для задачі розпізнавання.

Для створення системи безпеки були протестовані сучасні методи детектування та розпізнавання на відкритих та власних датасетах та були обрані кращі методи для необхідного середовища та умов.

1 СУЧАСНИЙ СТАН ПИТАННЯ РОЗПІЗНАВАННЯ ОБЛИЧ

1.1 Успіхи у вирішенні задач аналізу облич та огляд методів

Задачі детектування та розпізнавання облич є головними при створенні систем безпеки, а саме детектування є першим етапом, від якого залежать наступні результати. Саме детектування вже вважається достатньо вирішеною задачею, проте, все ще потребує дослідження для адаптації його до специфічних умов, пришвидшення та інше. Перші методи детектування облич були засновані на класичних підходах, при яких з зображення або його частини вилучалися ознаки, які задавала людина: наявність очей, носу, роту, їх взаємне розташування, пошук за шаблоном. Згодом були методи, які вже вилучали структурні ознаки з зображень, які у наступному кроці передавалися класифікатору: каскад Хаара [3], який аналізує зображення набором примітивів у сковзному вікні, аналіз гістограми направлених градієнтів (HOG) [4]. Такі методи значно покращили детектування у той час і все ще є актуальними на сьогодні, але їх точність вже не може конкурувати з більш новими підходами [5], особливо при змінах умов детектування, часткового перекритих обличчях. Стрімкий ріст якості детектування припав на появу глибоких архітектур нейронних мереж, перевага полягає у тому, що навчена для багатокласової класифікації мережа може сама виявити необхідні ознаки, які характеризують обличчя та знайти його. Перші такі методи використовували ідеї каскадного підходу, де зображення сканувало одразу декілька мереж і відібрані регіони далі сканувалися наступним набором мереж. Такий підхід ще покращив якість детектування, проте швидкість детектування залежала від кількості облич на зображенні [6]. Наступним етапом розвитку стали підходи, які використовували ідеї R-CNN [7] та Faster-RCNN [8] методів, бо на той час вони були лідерами у вирішенні задачі детектування об’єктів. У них пошуком кандидатів займається окремий модуль мережі (RPN – Region Proposal Network), який

одночасно відбирає частини зображення, які потім будуть передані класифікатору для остаточного висновку – це лице, чи ні. Перевагою було те, що починаючи з Faster-RCNN, навчання проводилося для усієї мережі, а не окремо для кожного її модулю, але головним недоліком такого підходу була швидкість, тому з'явилися методи, які проводили детектування за один прохід (One Stage) та доповнили їх методи на основі пірамід ознак (Feature Pyramid Network) [9], що дало змогу об'єднати семантично слабкі ознаки з семантично сильними. У свою чергу це дозволило покращити точність детектування облич малого розміру, з перепадами яскравості або частково перекритих іншими об'єктами. Ще одна група методів – це методи, які засновані на теплових картах, які видає згорткова мережа. Тобто метод не вертає точні координати знаходження обличчя, а видає попіксельну ймовірність належності пікселю до класу обличчя. На цьому еволюція методів для детектування не завершена і все ще ведеться робота для досягнення кращих результатів у цій задачі. На рисунку 1.1 представлені приклади методів та роки їх створення [10].

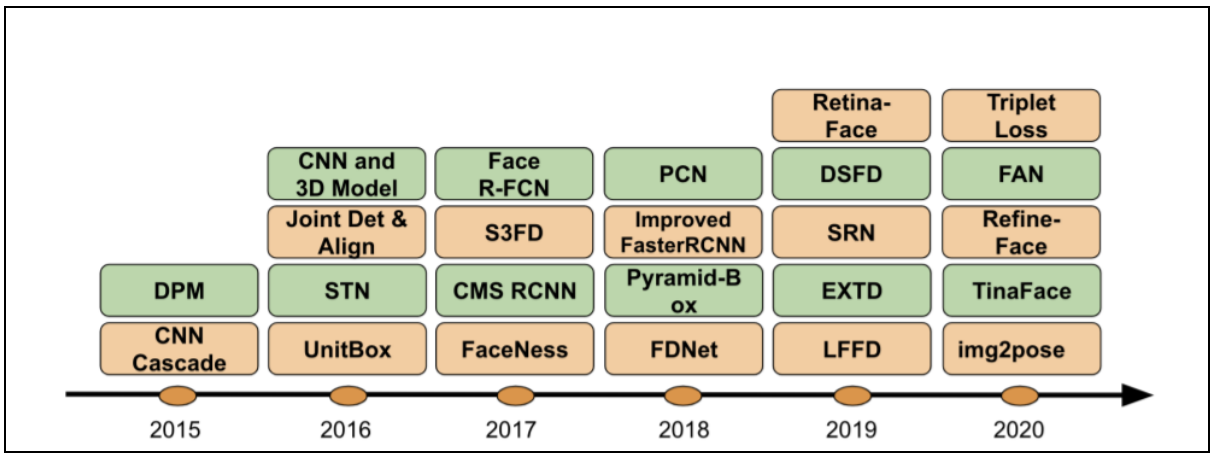


Рисунок 1.1 – Розвиток детекторів на основі глибоких нейронних мереж [10]

Задача розпізнавання, у свою чергу, також пройшла шлях розвитку від простого зіставлення кожного пікселю з еталонним зображеням та підрахування відмінностей до теперешніх методів на основі глибокого навчання. Перші підходи переводили обличчя у інший, більш менший

математичний вимір, на основі припущень про статистичний розподіл, для подальшої класифікації. Далі з'явилися методи на основі локальних характеристик: Габор [11], LBP [12], які вже більш точно могли описати обличчя. Еволюція цих підходів показана на рисунку 1.2.

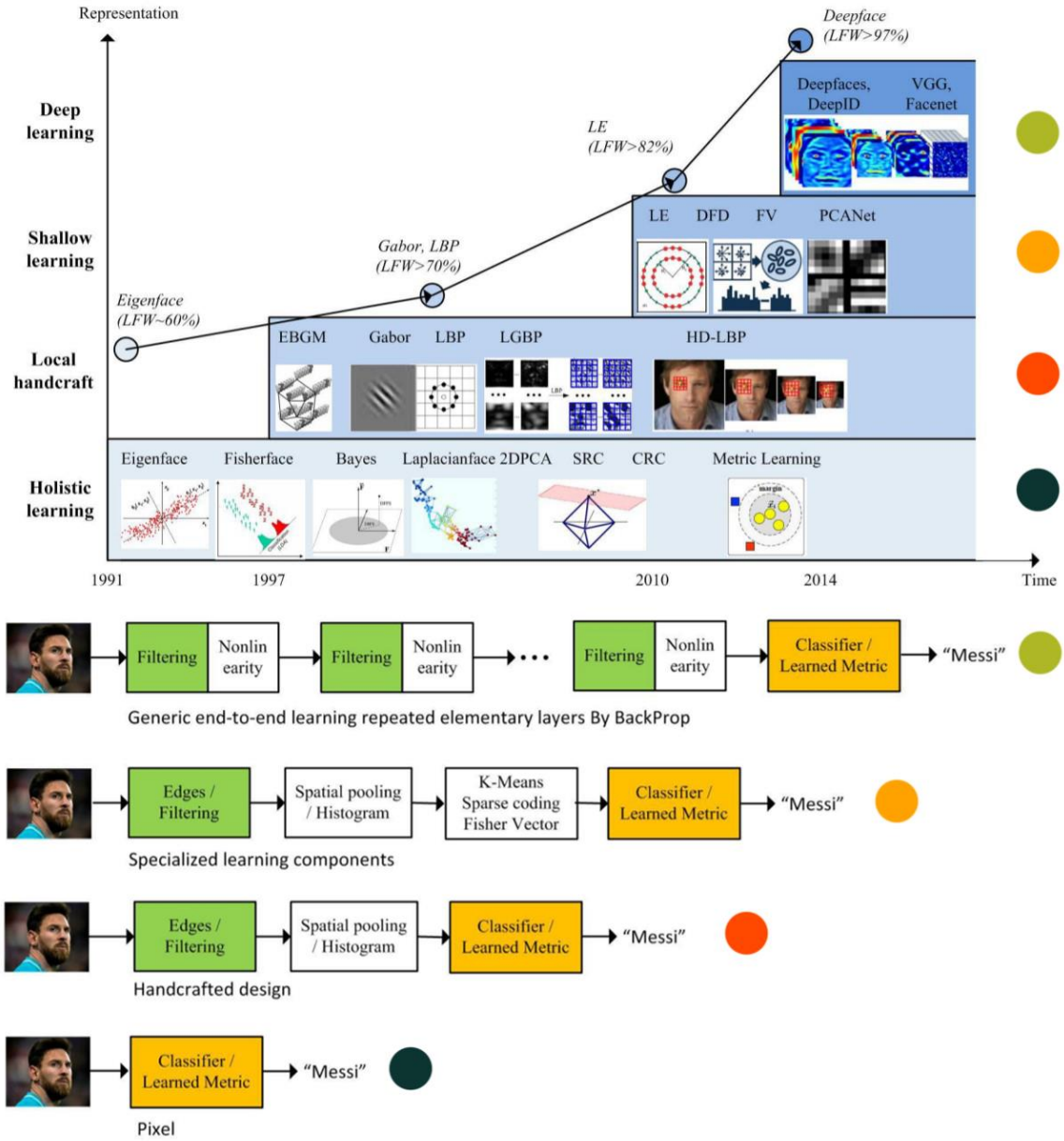


Рисунок 1.2 – Еволюція підходів до розпізнавання обличчя [13]

Зараз дослідження і прогрес у цій області дозволили створити системи, які працюють на телефонах, планшетах, дверних замках, і дозволяють проводити верифікацію користувача за секунди. Наприклад, система FaceID

дозволяє проводити верифікацію для підтвердження дій, банк «ПриватБанк» запустив систему FacePay24 для оплати покупок за обличчям без використання фізичної банківської карти, системи безпеки аеропортів для перевірки людей за базами даних розшуку та інші.

1.2 Зміст задачі розпізнавання облич та етапи її вирішення

Під задачею розпізнавання мається на увазі віднесення обличчя до заздалегідь сформованої бази осіб. Після того, як обличчя було знайдене, його віддають на розпізнавання. Задачу розпізнавання облич зазвичай розбивають на такі кроки: підготовка обличчя (пошук обличчя на зображенні; пошук ключових точок обличчя (face landmarks) та вирівнювання зображення за знайденими ключовими точками; перевірка обличчя на справжність); вилучення ознак, які описують обличчя (embeddings vector); зіставлення облич за вектором ознак (класифікація) (рис. 1.3) [13, 14].

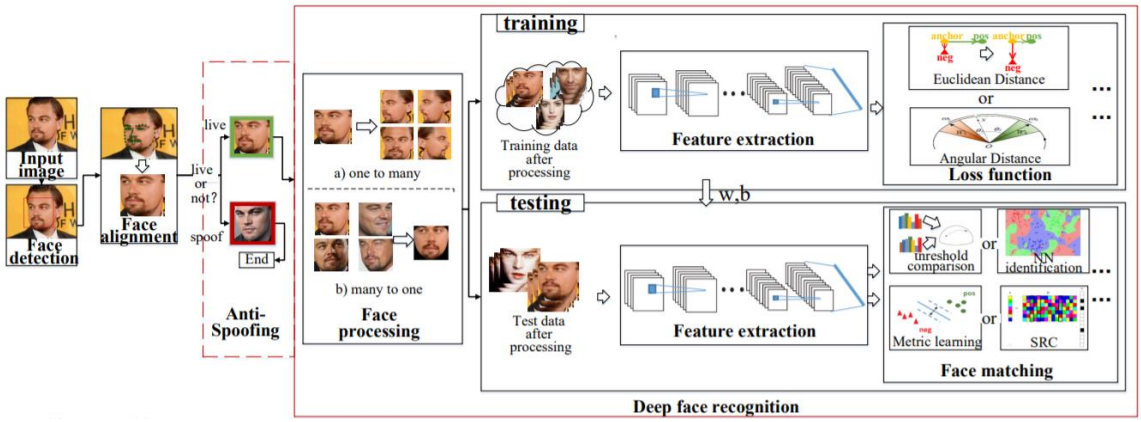


Рисунок 1.3 – Етапи розпізнавання обличя [13]

Для вилучення вектору ознак, який описує обличчя, за допомогою глибоких неронних мереж необхідно зробити наступне: обрати або створити архітектуру нейронної мережі для класифікації та провести її навчання на

великому обсязі даних. При навчанні на етапі, який відповідає за преобробку облич, існує два підходи для вирівнювання: один до багатьох (аугментація) – на цьому етапі на основі одного обличчя синтезуються певна кількість нових зображень, що дозволяє моделі бути більш стійкою до змін положення обличчя; багато до одного – різні положення: повороти, нахили обличчя – зводять завжди до еталонного (вирівнюють). Після завершення навчання у моделі «відрізаються» останні шари, які відповідають за класифікаційо і залишається лише блок вилучення ознак. При прямому проходженні даних по такій мережі, результат виходу з неї і є вектором, який описує обличчя. Для розпізнавання на етапі навчання також важливу роль також відіграє функція помилки, бо за її результатом саме і відбувається перерозподіл ваг у нейронній мережі, навіт зміна функції помилки давала приріст якості роботи методу [15]. На рисунку 1.4 продемонстрований розвиток функцій помилок [13].

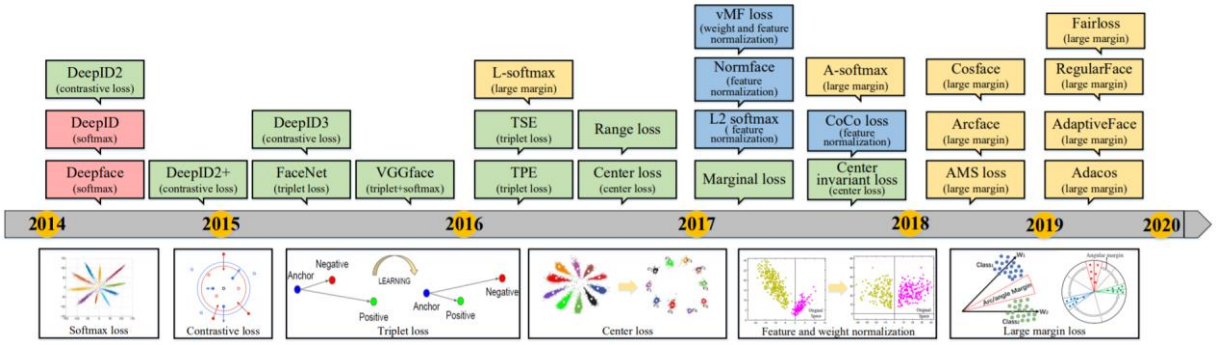


Рисунок 1.4 – Етапи розвитку функцій помилок [13]

Після того, як був отриманий вектор, що описую знайдене обличя, за ним, для розпізнавання, можна провести його зіставлення з відомою базою інших векторів, провести кластеризацію і виявити схожих та окремо порівняти ступінь схожості з кожним. Окремо для систем безпеки проблемою є те, що людину потрібно розпізнавати за малим набором фотографій, які система повинна вивчити. Цей підхід називається One Shot Learning [16, 17], до якого належать сіамські нейронні мережі [18], які

порівнюють два обличчя і видають міру їх схожості, але при такому порівнянні необхідно все одно спочатку знайти з чим порівнювати, бо база може бути великою.

1.3 Етап детектування облич

1.3.1 Оцінка якості детектування

Для оцінки якості детектування необхідно ввести поняття IoU (Intersection Over Union) (1.1) та наступні величини:

– TP – вірне детектування, $IoU \geq threshold$;

– FP – помилкове детектування, $IoU < threshold$;

– FN – не знайдений об’єкт.

$$IoU = \frac{area(detection \bigcap groun\ truth)}{area(detection \bigcup ground\ truth)}, \tag{1.1}$$

де detection – результат детектування;

ground truth – істинний результат детектування.

Найбільш популярні метрики будуються на підставі precision (1.2) та recall (1.3)

$$precision = \frac{TP}{TP + FP}, \tag{1.2}$$

$$recall = \frac{TP}{TP + FN}. \tag{1.3}$$

Першою метрикою є precision recall (PR) curve, для якої усі детектування впорядковуються за впевненістю, яку вертає метод, і

послідовно для кожного, з додаванням попередніх TP, FP, FN, рахуються значення precision та recall, значення яких для кожного детектування відображаються на графіку PR. Порівняння методів за цією метрикою іде за допомогою накладання графіків і кращий той метод, у якого графік вище.

Накладання не завжди зручне, тому наступна метрика average precision (AP) рахується як площа під графіком PR curve і вже більш наглядно демонструє точність детектування. Для підрахування площі під графіком використовують методи інтерполяції за 11 точками (1.4) та інтерполяцію за всіма точками (1.5)

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{\text{int}}(r), \quad p_{\text{int}} = \max_{\tilde{r}: \tilde{r} \geq r} p(\tilde{r}), \tag{1.4}$$

де $p(\tilde{r})$ – значення precision при recall $\tilde{r}$,

$$AP = \sum_{n=0} (r_{n+1} - r_n) p_{\text{int}}(r_{n+1}), \quad p_{\text{int}}(r_{n+1}) = \max_{\tilde{r}: \tilde{r} \geq r_{n+1}} p(\tilde{r}). \tag{1.5}$$

Ще одними із метрик є F_1-міра, яка пов’язую precision та recall (1.6), та mean average precision (mAP), яка використовується для оцінки багатокласової детекції і рахується як середнє значення AP для усіх класів, або середнє значення для усіх значень AP при різних значення IoU threshold [19]

$$F_1 = 2 * \frac{precision * recall}{precision + recall}. \tag{1.6}$$

1.3.2 Датасети для навчання моделей та оцінки якості детектування

Заміри за попередніми метриками робляться на окремих даних, які метод не бачив на етапі навчання, зазвичай це окремі датасети. Популярними є наступні:

– WIDER Face різних ступенів складності [20];

– FDDB [21];

– Annotated Faces in the Wild [22].

WIDER Face став найбільш популярним, бо має три підмножини зображень: Easy, Medium, Hard – які розподілені за складністю детектування облич за ступенями зміни масштабу обличчя, освітленості, пози, перекриття обличчя іншими предметами, емоцій. На рисунку 1.5 та 1.6 показані приклади зображень з датасетів.

Рисунок 1.5 – Приклад датасету WIDER Face [20]

Рисунок 1.6 – Приклад зображеннь з датасету FDDB [21]

На рисунку 1.7 представлений графік [23], на якому відображені методи зі значенням метрики AP по роках. Результати AP були отримані на датасеті WIDER Face Hard.

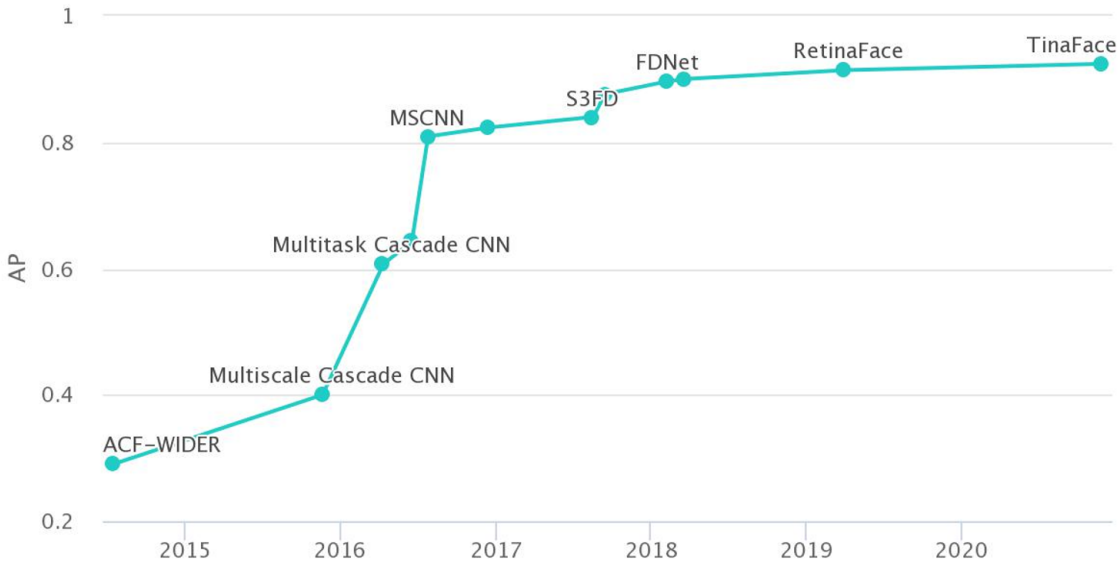


Рисунок 1.7 – Графік розвитку методів за якістю
на датасеті WIDER FACE Hard [23]

1.4 Етап розпізнавання

1.4.1 Оцінка якості розпізнавання

Оцінка якості розпізнавання, як і детектування, також спирається на значення precision та recall, але додається ще одна величина до матриці помилок TN – вірне відкидання класу, бо задачу розпізнавання можна віднести до задачі класифікації. Наприклад можна взяти наступні метрики: accuracy (1.7), precision, recall, F_1-міру, AP, mAP, receiver operating characteristic (ROC Curve) крива, площа під ROC (AUC ROC)

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}. \quad (1.7)$$

ROC Curve – це крива, яка відображає значення TPR (Recall) на осі ординат, та значення FPR (1.8) на осі абсцис

$$FPR = \frac{FP}{FP + TN}. \tag{1.8}$$

Ідея її полягає у тому, що результат, який вертає метод, інтерпретується за допомогою порогового значення. Змінюючи значення порогу мі будемо отримувати різні значення TP, FP, TN, FN, які і впливають на значення графіку. Кращим вважається той метод, у якого графік знаходиться вище. AUC ROC – це значення площі під графіком ROC. Чим це значення більше, тим краще працює метод.

1.4.2 Датасети для навчання моделей та оцінки якості розпізнавання

Для навчання необхідно зібрати, або взяти доступні набори даних, які будуть вже розбиті на класи, за якими буде проводитись навчання або тестування. Найбільш поширеними є наступні датасети:

- VGGFace2 [24];
- CelebA [25];
- IJB-B [26]
- LFW [27].

На рисунку 1.8 показані приклади зображень з датасету CelebA.

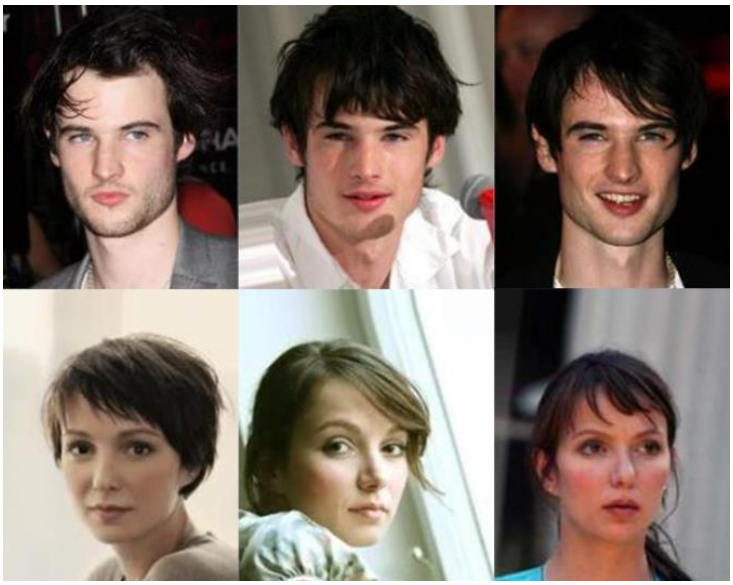


Рисунок 1.8 – Приклад зображень з датасету CelebA

1.5 Формування вимог до системи безпеки підприємства

Для системи безпеки головними э два аспекти: швидкодія та точність. У свою чергу точність необхідно розглядати як для усієї системи так і окремо для кожної її частини. Система безпеки повинна вміти детектувати людей у полі зору при різних умовах швидко, записувати ці події у журнал і паралельно розпізнавати. Людина не повинна робити окремих дії для її розпізнавання, усі дії повинні відбуватися у фоновому процесі. Система повинна бути легко конфігурованою та працювати з багатьма камерами одночасно.

Для системи також необхідно:

– обробляти один кадр за не більше ніж 100 ms;

– бути нечуттєвою до зміни положення камер, знаходженню людини, зміни освітленості;

– детектувати обличчя не менші ніж 100 px;

– сповіщати про зниження якості розпізнавання;

– якість системи повинна бути не менше 98% за значенням AUC ROC.

1.6 Постановка задачі дослідження

Метою даної роботи є розробка та дослідження методу розпізнавання облич для систем безпеки на базі результатів проведеного аналізу сучасних методів детектування та розпізнавання об’єктів.

Об’єктом дослідження є проблема розпізнавання облич у реальному часі.

Для досягнення мети необхідно вирішити такі задачі:

– проаналізувати сучасний стан розвитку питання детектування облич;

– проаналізувати існуючі підходи для детектування;

– створити датасет для аналізу якості існуючих підходів детектування облич при зміні умов положення та масштабу;

– оцінити якість існуючих методів детектування;

– оцінити швидкість існуючих методів детектування;

– обрати метод, який найкраще підходить до поставленого завдання;

– вивчити стан та розвиток методів нормалізації облич;

– проаналізувати сучасний стан розвитку задачі розпізнавання та методи її вирішення;

– провести аналіз методів класифікації облич;

– зібрати дані для створення датасету для оцінки якості розпізнавання;

– створити pipeline усієї системи;

– спроєктувати та створити програмну реалізацію;

– обрати метод для покращення роботи системи при зменшенні кількості вірних розпізнавань під час її функціонування.

2 РОЗРОБКА МЕТОДУ РОЗПІЗНАВАННЯ ОБЛИЧ ДЛЯ СИСТЕМ БЕЗПЕКИ НА ОСНОВІ АНАЛІЗУ МЕТОДІВ ДЕТЕКТУВАННЯ ТА КЛАСИФІКАЦІЇ

2.1 Аналіз методів детектування

2.1.1 Математичні моделі методів детектування

2.1.1.1 MTCNN

Метод MTCNN (Multi-task Cascaded Convolutional Networks) [28] виконує відразу декілька задач: пошук обличчя, знаходження ключових точок та вирівнювання. Запропонований він був у 2016 році. На вхід подається зображення, для якого будується піраміда зображень різних масштабів (рис. 2.1), і яка є входом наступної триступеневої каскадної мережі. Після обробки зображення мережею MTCNN на виході отримуємо позиції облич і їх ключові точки.

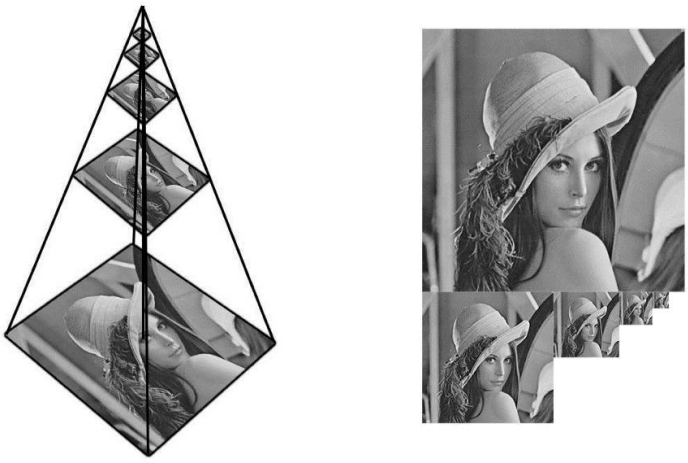


Рисунок 2.1 – Побудова піраміди масштабів вхідного зображення [29]

Пропозиційний модуль P-Net. P-Net модуль – це повнозгорткова нейрона мережа (FCN), яка використовується для прогнозування регіонів, де знаходяться обличчя. Для цього використовується техніка bounding box regression. Структура мережі представлена на рисунку 2.2.

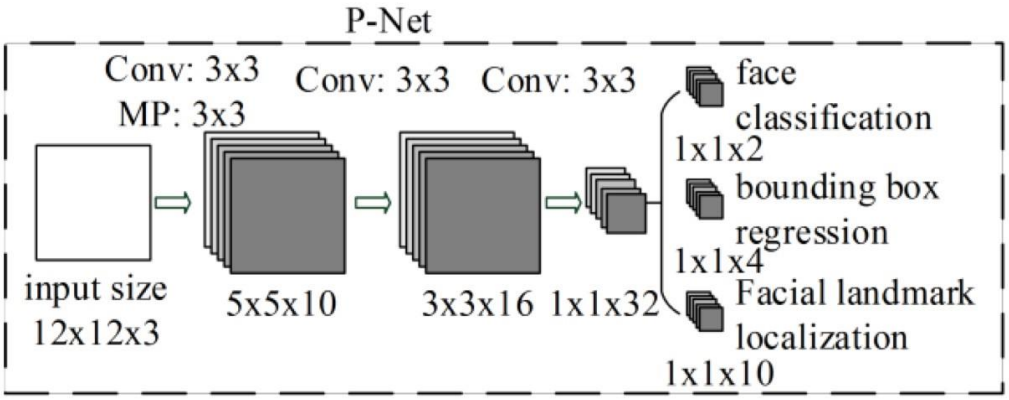


Рисунок 2.2 – Структура пропозиційного модулю мережі MTCNN [28]

Уточнюючий модуль R-Net. Усі кандидати із попереднього модулю надходять в уточнюючу CNN мережу. R-Net (рис. 2.3) додатково зменшує кількість кандидатів, виконує метод калібрування за допомогою регресії граничних областей і використовує метод не максимального придушення (NMS) для об’єднання кандидатів, які перекривають один одного.

На виході мережа видає результат, чи є вхід обличчям, позицію обмежувальної рамки з чотирьох точок і вектор ключових точок обличчя з десяти елементів.

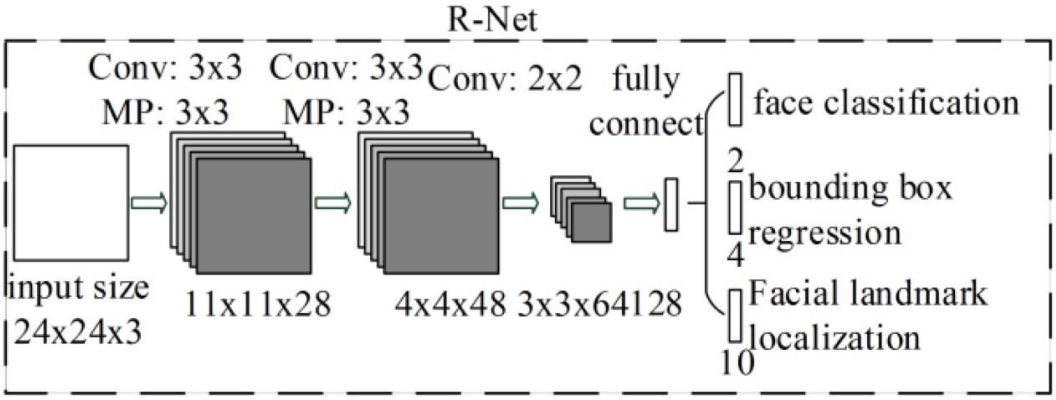


Рисунок 2.3 – Структура уточнюючого модулю мережі MTCNN [28]

Вихідний модуль O-Net. O-Net (рис. 2.4) модуль на виході видає положення лиця і відносно нього положення ключових точок – очі, ніс, ріт.

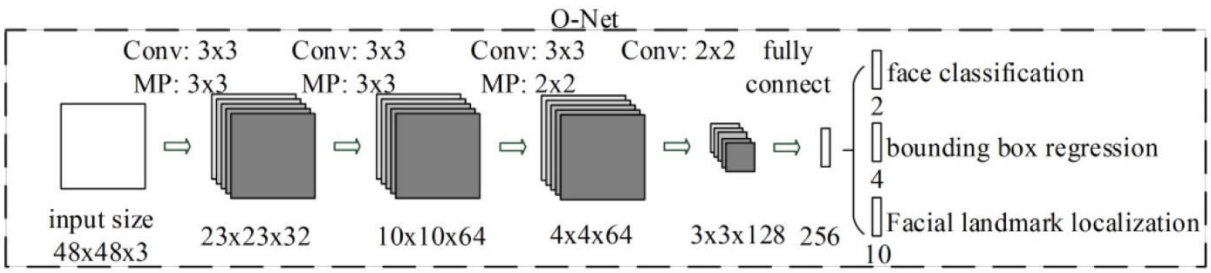


Рисунок 2.4 – Структура вихідного модулю мережі MTCNN [28]

Повний алгоритм роботи нейронної мережі MTCNN проілюстрований на рисунку 2.5.

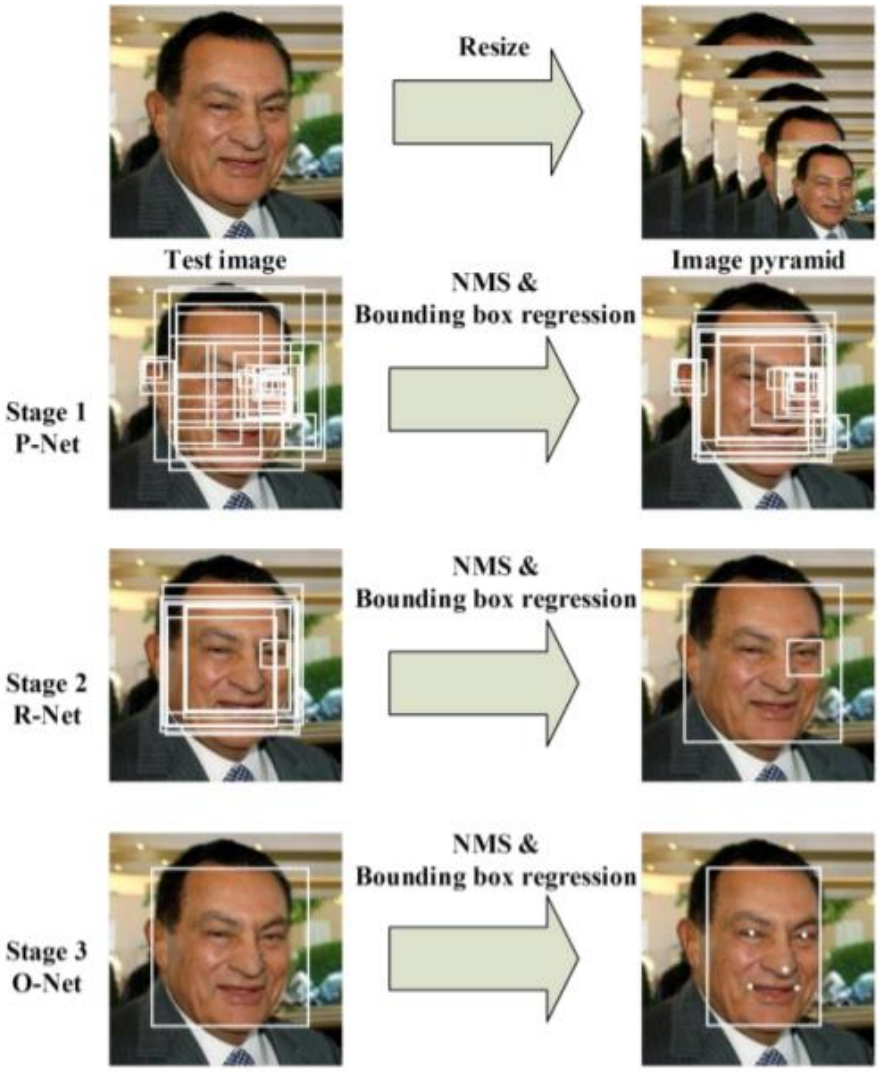


Рисунок 2.5 – Алгоритм роботи MTCNN мережі [28]

2.1.1.2 FaceBoxes

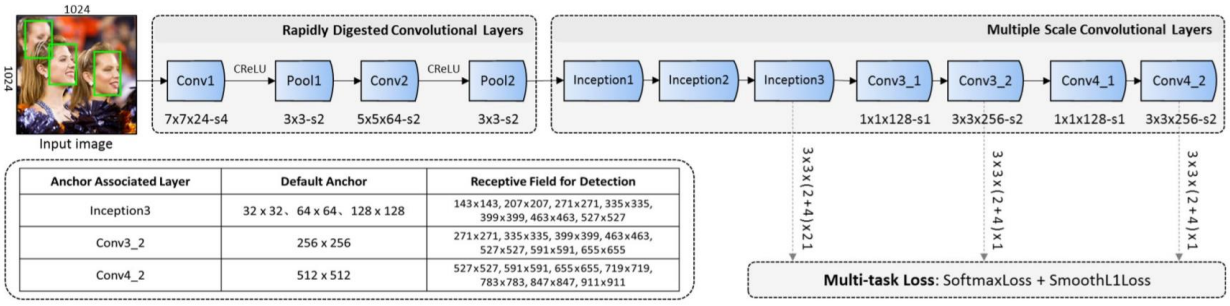
При розробці методу FaceBoxes [30] у 2017 році головною ціллю було зробити детектор облич у реальному часі, але не втратити у якості. Для цього були внесені наступні зміни:

– були розроблені згорткові шари Rapidly Digested Convolutional Layers (RDCL) для досягнення детектування у реальному часі;

– розроблені згорткові шари Multiple Scale Convolutional Layers (MSCL) для знаходження облич різних масштабів;

– розроблена нова методика генерування якорів розташування облич.

На рисунку 2.6 представлена архітектура мережі FaceBoxes.



Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

Рисунок 2.6 – Архітектура мережі FaceBoxes [30]

Rapidly Digested Convolutional Layers. Більшість методів для детектування облич використовує звичайні згортки, що зумовлює їх повільність, особливо на CPU. Для вирішення цієї проблеми у методі FaceBoxes було запропоновано створити шар, який буде швидко зменшувати розмірність вхідних даних. Для цього було зроблено наступне:

– використано великий крок зсуву для згорток та пулінгу у блоках шару RDCL. Як показано на рисунку 2.7, розмір кроку Conv1, Pool1, Conv2 і Pool2 становлять 4, 2, 2 і 2, відповідно;

– підібрані розміри ядер, бо вони повинні на перших декількох рівнях в мережі бути невеликими, для її прискорення, і в той же час повинні бути

достатніми для запобігання втрати інформації, через просторове зменшення інформації;

– зменшена кількість вихідних даних з шару. Для цього була використана функція активації C.ReLu. Це зумовлено тим, що фільтри на перших шарах зазвичай утворюють пари протилежних примітивів. З цього спостереження C.ReLU може замістити віхідні дані, об’єднавши заперечені виходи перед застосуванням ReLu (рис. 2.8).

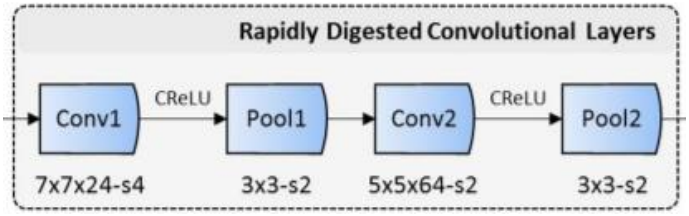


Рисунок 2.7 – Шар RDCL [30]

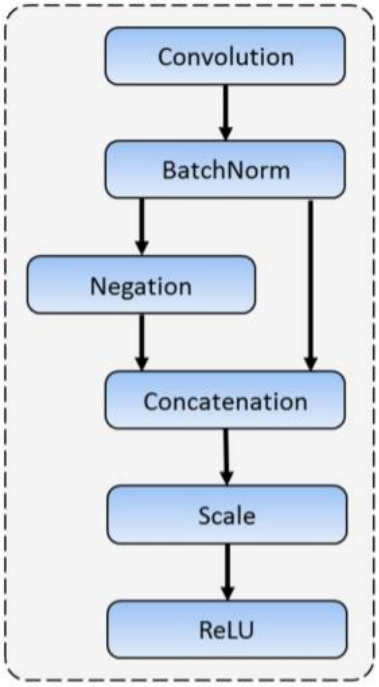


Рисунок 2.8 – Функція C.ReLu [30]

Multiple Scale Convolutional Layers. Метод заснований на RPM (Region Proposal Network), який повинен пропонувати варіанти лише одного класу – обличчя. Однак, як окремий детектор осіб, RPN не в змозі отримати

конкурентоспроможні результати. Це зумовлено двома аспектами. По-перше, якоря в RPN пов’язані тільки з останнім згортковим шаром, чиї вилучені ознаки та розмір занадто слабкі для обробки осіб різних розмірів. По-друге, шар, пов’язаний з якорем, відповідає за виявлення осіб у відповідному діапазоні масштабів, але він має тільки одне рецептивне поле, яке не може виявляти різні масштаби. Для цього у шарі MSCL було запропоноване наступне:

– шари зменшуються у розмірах прогресивно і формують багатомаштабні карти ознак;

– модуль Inception складається з декількох гілок згорток (рис. 2.9). Це дозволяє знаходити обличчя різних розмірів.

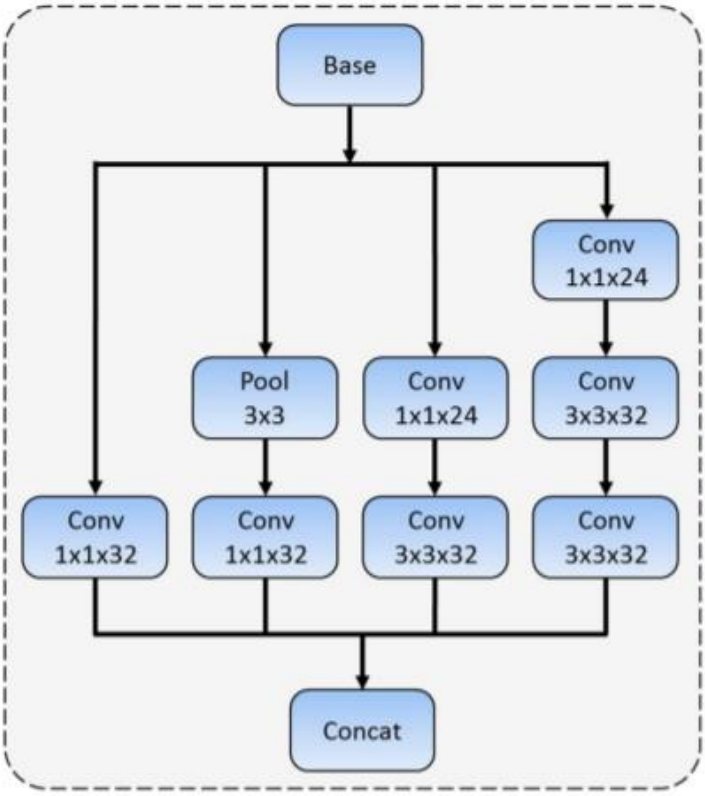


Рисунок 2.9 – Модуль Inception [30]

Anchor densification strategy. Для якорів встановлюється співвідношення сторін 1:1, тому що обличчя має приблизно квадратну форму. Масштаб якорю для шарів Inception3 становить 32, 64 і 128 пікселів,

шару Conv3_2, Conv4_2 – 256 і 512 пікселів, відповідно. Інтервал чергування якоря на зображенні дорівнює розміру кроку відповідного шару, пов’язаного з якорем. Якщо порахувати щільність якорів, то отримаємо значення на шарах 1, 2, 4, 4 – що відображає дисбаланс у щільності, тому, щоб це виправити для типу якоря *n*, необхідно рівномірно розташувати n^2 якорів навколо одного рецептурного поля замість того, щоб мати тільки один у центрі цього рецептивного поля для прогнозування (рис. 2.10).

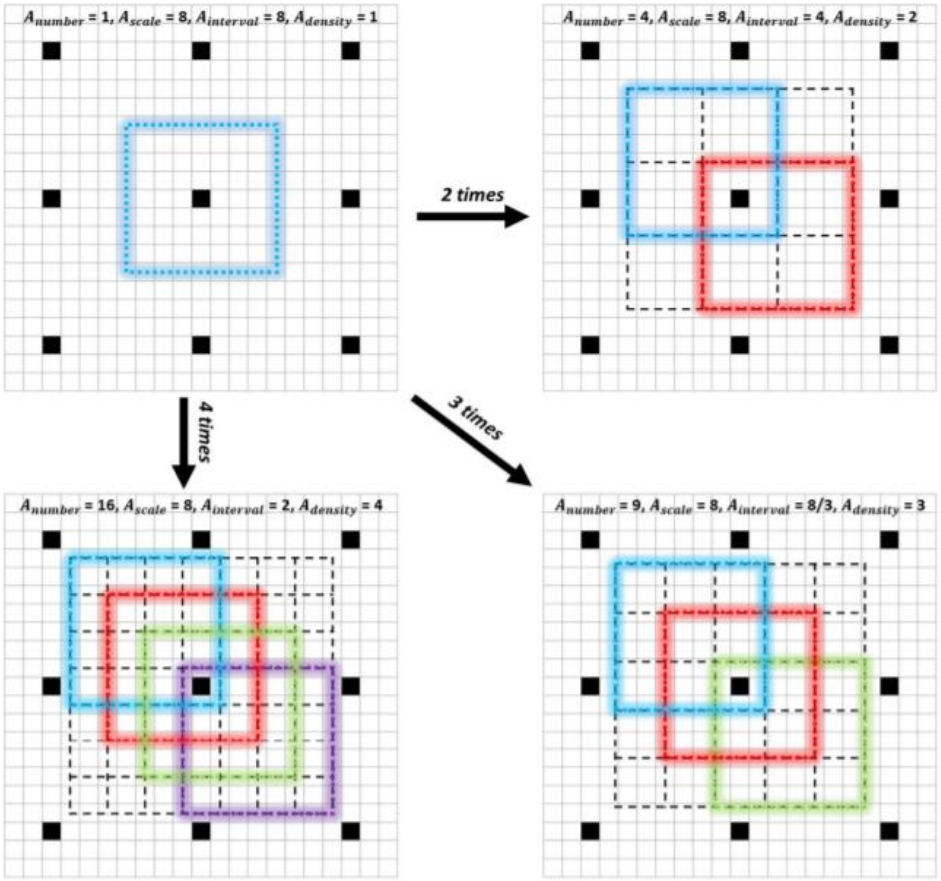


Рисунок 2.10 – Розташування якорів для знаходження облич у методів FaceBoxes [30]

2.1.1.3 Dual Shot Face Detector (DSFD)

Детектор DSFD [31, 32] в більшій мірі заснована на архітектурі детектора SSD [33]. Він був вперше запропонований у 2018 році. Ця

архітектура відрізняється від мереж з RPN тим, що в ній немає окремого кроку пропозиції регіонів. Координати і знаходження обличчя передбачаються безпосередньо з карти ознак, що швидше ніж окремий крок. Архітектура представлена на рисунку 2.11. В ній такий самий модуль вилучення перших характерник ознак (backbone) VGG [34]/ResNet [35], як і у SSD. Ключова відмінність полягає е тому, що шість карт ознак різної глибини перетворюються в шість «поліпшених» карт за допомогою модуля Feature Enhance Module (FEM).

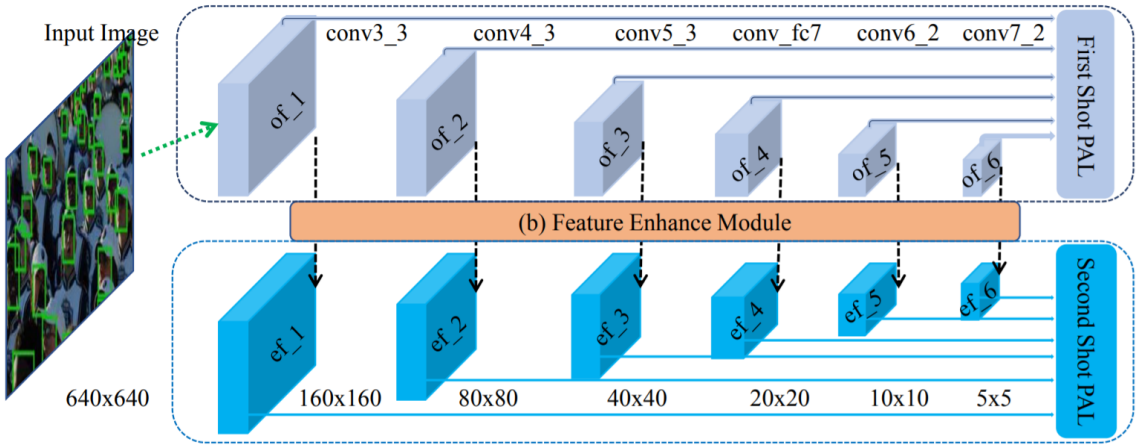


Рисунок 2.11 – Структура мережі DSFD [31]

Класифікатори та регресорів навчаються і застосовуються як до вихідних, так і до розширених карт ознак. First Shot PAL і Second Shot PAL – дві функції втрат для навчання.

Детальний опис FEM модулю представлений на рисунку 2.12. У ньому поелементно множаться карти ознак поточного рівня і наступного рівня глибини. Отримана карта розбивається на три частини. Кожна з них проходить серію від однієї до трьох розширених (dilated) [36] (рис. 2.13) згорток із внутрішньою відстанню між пікселами (rate) 3, після чого виконується об’єднання в повну карту ознак того ж розміру, що і вхідна поточна.

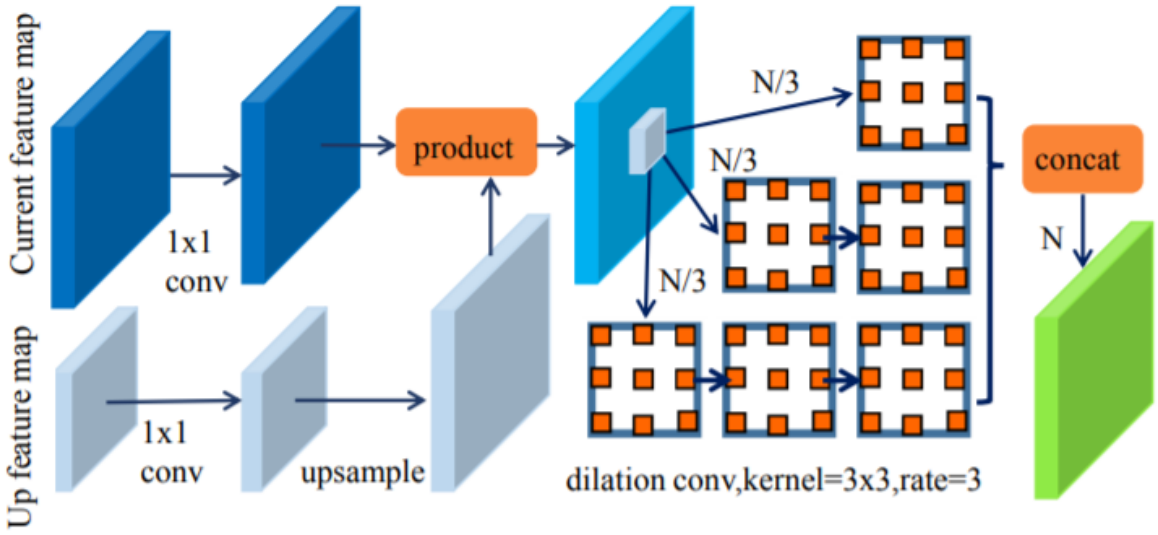


Рисунок 2.12 – Структура Feature Enhance Module [31]

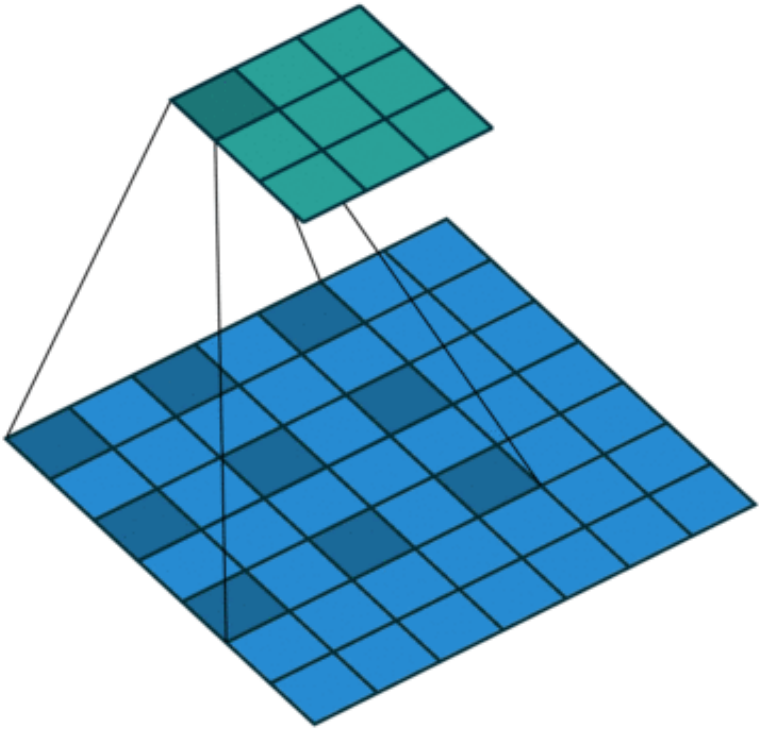


Рисунок 2.13 – Ілюстрація розширеної згортки [37]

Для навчання мережі використовували функцію втрат, яка є зваженою сумою двох функцій втрат набору якорів (2.1)

$$L_{PAL} = L_{FSL}(sa) + L_{SSL}(a), \tag{2.1}$$

де *a* – набір якорів для підрахування first shot loss;

sa – менший набір якорів для підрахування second shot loss.

2.1.1.4 RetinaFace

Детектор RetinaFace [38] – це state of the art детектор 2019 року. Він виконує декілька задач одночасно:

– детектування обличчя;

– вирівнювання обличчя;

– 3D реконструкцію обличчя.

Для вилучення ознак з зображення на першому кроці використовується архітектура ResNet разом з Feature Pyramid Network (FPN) [39] (рис. 2.14), які потім передаються у контекстний модуль. Для посилення можливостей контекстного моделювання в цьому модулі використовується деформаційну згортку (DCN) [40] над картами ознак, крім звичайної згортки 3×3.

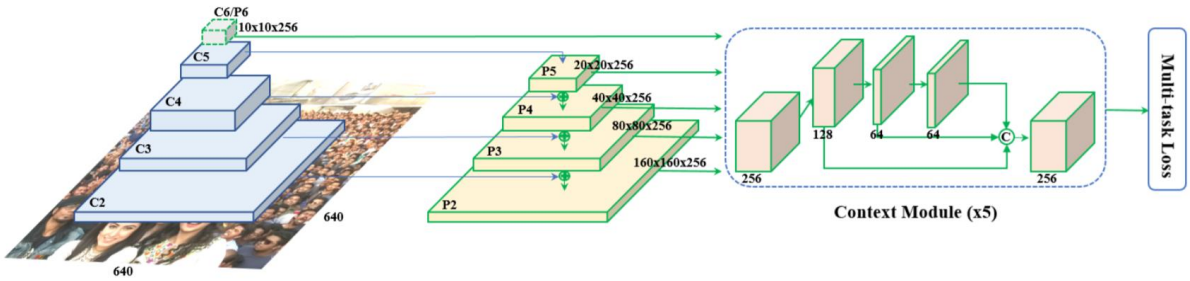


Рисунок 2.14 – Архітектура RetinaFace [38]

Звичайні глибокі згорткові мережі при прямому проході створюють карти ознак з різним ступенем просторового розділу, який зменшується, і через це деякі семантичні ознаки втрачаються, але знаходяться також і нові, що робить знаходження об’єктів різного масштабу, насамперед невеликих, складним. FPN передбачає об’єднання даних знизу-догори, зверху-донизу, та бічне. Метод видає пропорційні за розміром карти ознак на декількох рівнях,

де більш загальні семантичні ознаки з вищих шарів об’єднуються з попередніми (рис. 2.15).

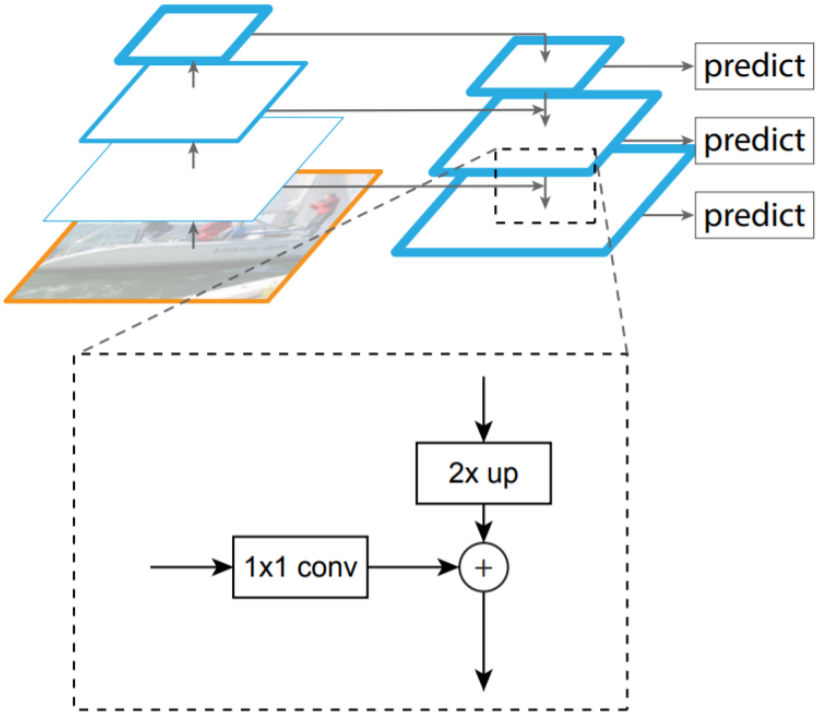


Рисунок 2.15 – Структура мережі FPN [39]

Для навчання мережі RetinaFace була застосована multi-task loss (2.2) рисунок 2.16

$$L = L_{cls}\left(p_i, p_i^*\right) + \lambda_1 p_i^* L_{box}\left(t_i, t_i^*\right) + \lambda_2 p_i^* L_{pts}\left(l_i, l_i^*\right) + \lambda_3 p_i^* L_{pixel}, \qquad (2.2)$$

де p_i – передбачена ймовірність якоря i обличчя;

p_i^* – дорівнює 1 для вірного якорю, інакше дорівнює 0;

L_{cls} – похибка softmax бінарної класифікації обличчя;

$t_i = \{t_x, t_y, t_w, t_h\}_i$ – координати передбаченої рамки обличчя;

$t_i^* = \{t_x^*, t_y^*, t_w^*, t_h^*\}_i$ – координати вірної рамки обличчя для вірного якорю;

$L_{box}(t_i, t_i^*) = R(t_i - t_i^*)$ – robust loss function [41];

$L_{pts}\left(l_i, l_i^*\right)$ – функція помилки ключових точок обличчя як і L_{box};

$l_i = \left\{l_{x_1}, l_{y_1}, ..., l_{x_5}, l_{y_5}\right\}_i$ – координати передбачених ключових точок обличчя;

$l_i^* = \left\{l_{x_1}^*, l_{y_1}^*, ..., l_{x_5}^*, l_{y_5}^*\right\}_i$ – координати вірних ключових точок обличчя;

$\lambda_1 - \lambda_3$ – параметри 0,25; 0,1; 0,01.

$$L_{pixel} = \frac{1}{W * H}\sum_{i}^{W}\sum_{j}^{H}\left\| R\left(D_{Pst}, P_{cam}, P_{ill}\right)_{i,j} - I_{i,j}^* \right\|_1, \qquad (2.3)$$

де $R\left(D_{Pst}, P_{cam}, P_{ill}\right)$ – 2D обличчя;

$I_{i,j}^*$ – якір обличчя шириною W і висотою H.

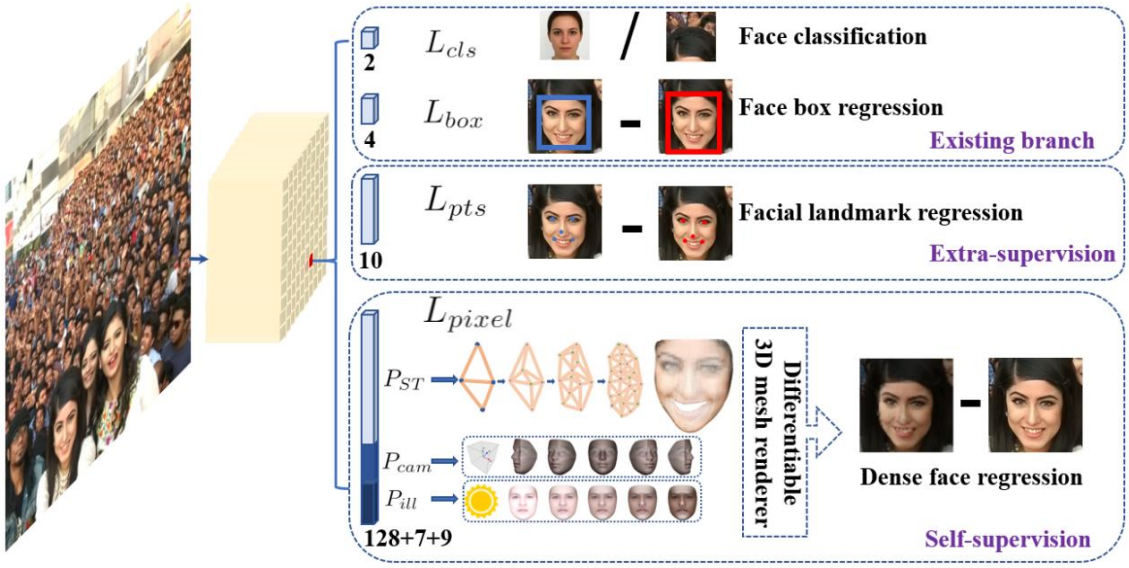
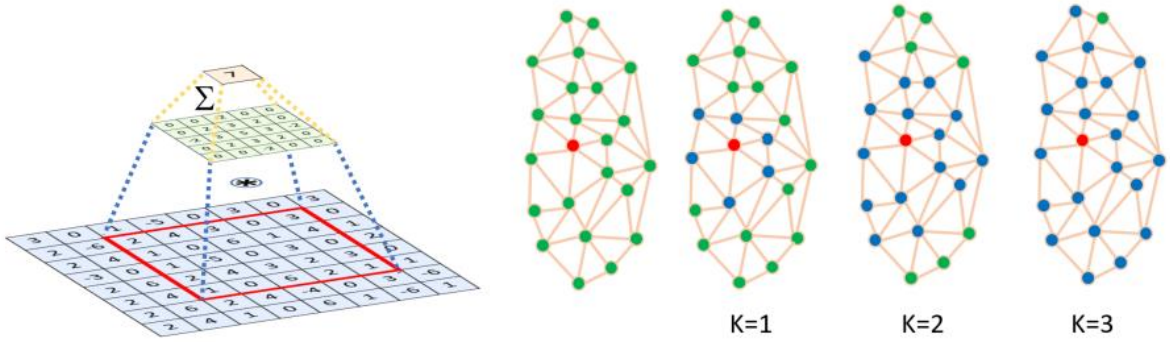


Рисунок 2.16 – Multi-task loss [38]

Для отримання 3D реконструкції обличчя був розроблений mesh decoder. Він заснований на методів графової згортки яка представлена на рисунку 2.17.



а) б)

Рисунок 2.17 – Ілюстрація згорток для RetinaFace mesh decoder [38]:
а) звичайна згортка; б) графова

2.1.1.5 CenterFace

Попередні методи, у яких використовувалися якорі для локалізації облич мали головний недолік, для покращення детектування потрібно було збільшувати щільність якорів, насамперед для невеликих за розміром. Наприклад, для попереднього детектору – RetinaFace, кількість якорів для зображення 640×640 пікселів становить близько 100 тисяч. Також якорі статистично обчислюються на основі певного набору даних, тому вони не завжди підходять для інших умов детектування. Детектор CenterFace був розроблений у 2019 році без застосування якірного підходу, його архітектура представлена на рисунку 2.18. Головною ідеєю цього методу є те, що обличчя описується центральною точкою рамки, що його описує. Центральна точка, розмір рамки та інші ключові точки підраховуються на основі теплових карт. Шукані точки відповідають пікам на теплових картах з шарів, які відповідають за знаходження відповідних точок [42].

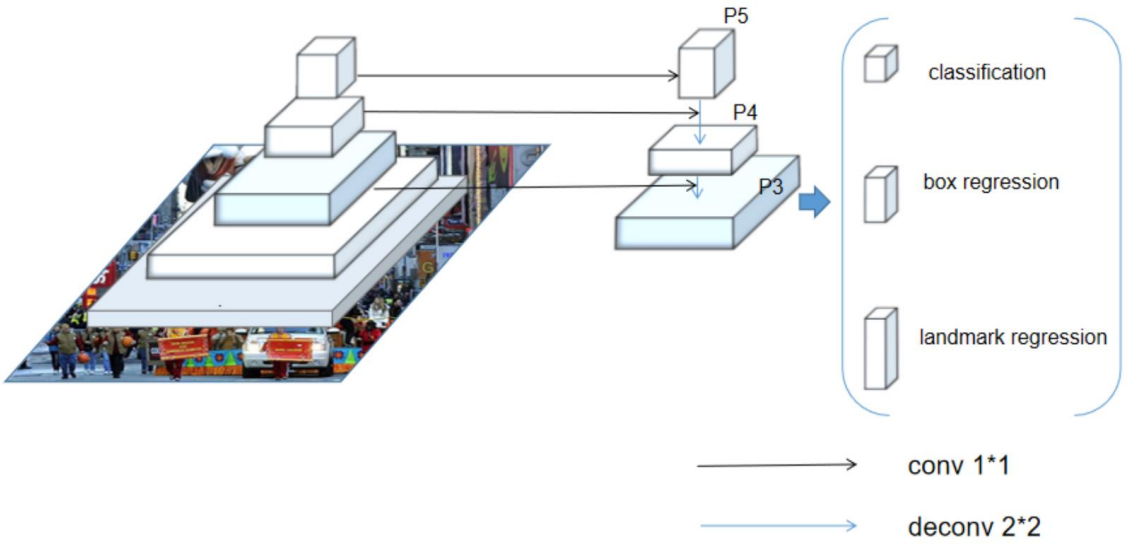


Рисунок 2.18 – Структура мережі CenterFace [42]

У архітектурі для вилучення перших ознак (backbone) була застосована мережа MobileNetV2 [43] з послідуючою передачею їх до FPN. Через те, що обличчя знаходиться за центральною точкою, лише один шар у FPN використовується для його виявлення. FPN складається з шарів, які зазначені як {*P*-*L*}, *L* = 3, 4, 5, де *L* означає рівень піраміди. Шар *Pl* має розмірність $\frac{1}{2^L}$ від вхідних даних. Усі рівні піраміди мають *C* = 24 канали.

Для навчання була застосована функція втрат на основі focall loss [44], (2.3). Еталонні дані у датасеті були закодовані за допомогою Гаусіану

$$L_c = \begin{cases} -(1-\hat{Y}_{xyc})^{\alpha} * \log(\hat{Y}_{xyc}), \; if \; Y_{xyc} = 1, \\ -(1-\hat{Y}_{xyc})^{\beta} * (\hat{Y}_{xyc})^{\alpha} * \log(1-\hat{Y}_{xyc}) \; otherwise, \end{cases} \tag{2.4}$$

де α – параметр з focal loss;

β – параметр з focal loss;

$\hat{Y}_{xyc}$ – значення з теплової карти.

Додатково детектор вертає п’ять ключових точок обличчя: очі, ніс та два кути рота.

2.1.1.6 SCRFD

Sample and Computation Redistribution for Efficient Face Detection [45] (SCRFD) метод 2021 року, який опирається на ідеї state of the art методу 2020 року TinaFace [46]. У TinaFace при роздільній здатності 640×480 пікселів були перевірені шари з якорями різного кроку за швидкодією, та було виявлено, що більше усього займають розрахунки якорів з кроком 4 (рис. 2.19).

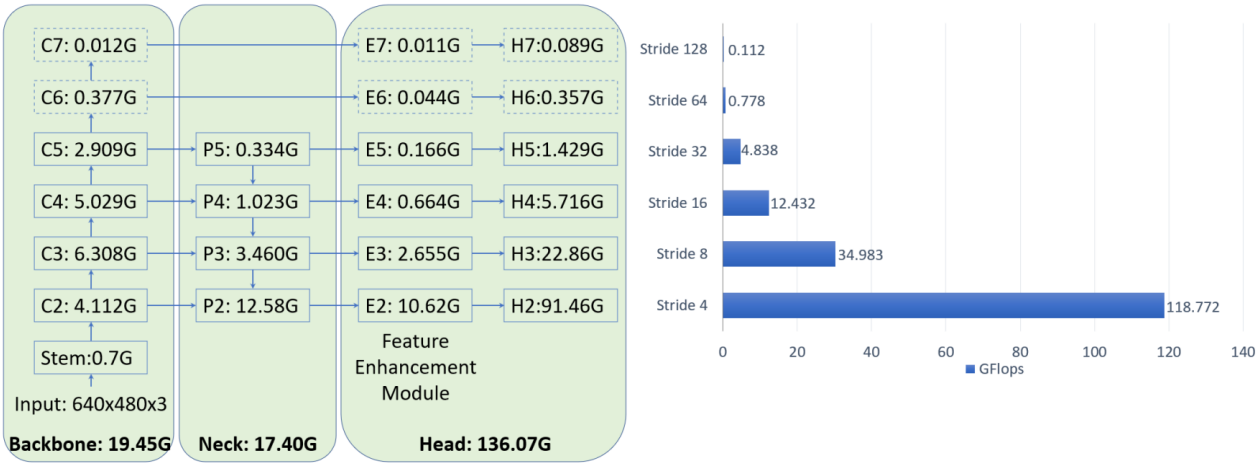


Рисунок 2.19 – Кількість операцій при підрахуванні якорів з різними кроками при розмірі входу 640×480 пікселів [45]

Додатково був побудований розподіл облич за розміром датасету WIDER Face [20] різного рівню складності (easy $\subset$ medium $\subset$ hard) (рис. 2.20). За цим розподілом був зроблений висновок, що при розмірі зображення 640×480 більшість облич для легкого детектування більше ніж 32 пікселі, середнього більше 16 пікселів і лише 13,36% менше 8 пікселів, тому було вирішено видалити якорі із кроком 4, що суттєво повинно покращити швидкодію мережі.

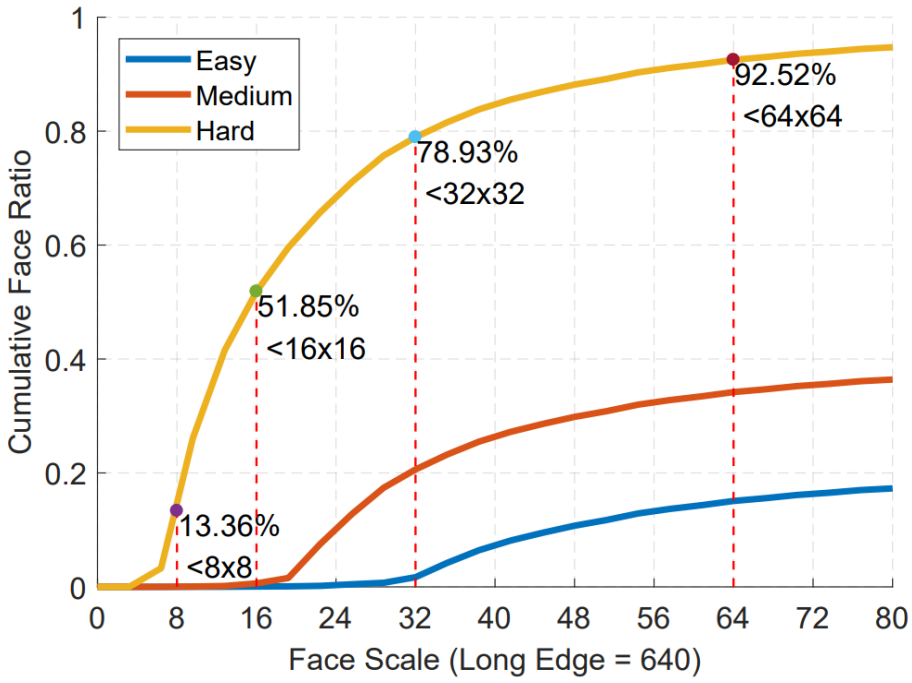


Рисунок 2.20 – Розподіл облич за розміром у датасеті WIDER FACE [45]

Усі попередні детектори були розроблені людиною на підставі якихось експертних знань, доводів або окремої оптимізації якоїсь із частин (backbone, neck, head) попередніх архітектур. У SCRFD структуру нейронної мережі шукали інші нейронні мережі окремо для кожної частини (backbone, neck, head) що є глобальною оптимізацією. У фіксованому просторі пошуку Neural Architecture Search (NAS) автоматично знайде кращу модель з цього простору. Для цього були використані: DetNas [47] для пошуку backbone структури детектору, CR-NAS [48] для оптимізаціх backbone, NAS-FPN [49] для пошуку структури FPN за допомогою reinforcement learning [50]. Вищеназвані методи оптимізують розподіл обчислень для backbone, neck, head компонентів, грунтуючись на статистиці, отриманої від групи випадково згенерованих моделей. Це скорочує простір пошуку і знаходить стабільний розподіл, що значно покращує продуктивність моделі. На рисунку А.1 продемонстровані зв’язки між шарами однієї із відібраних структур. Структури відрізняються точністю та швидкодією.

2.1.2 Порівняння розглянутих детекторів

2.1.2.1 Залежність якості детектування від кута нахилу обличчя

Для порівняння детекторів на якість детектування при різних кутах нахилу голови людини був створений власний датасет. У ньому було синтезоване обличчя людини та створено з нього 3D модель. Цю 3D модель у графічному редакторі Blender було відрендерено з кроком повороту один градус за двома осями окремо зліва-направо від -90 градусів до +90 градусів (вісь X) та кутом підйому голови знизу доверху від -90 до +90 градусів (вісь Y). Приклад зображення з датасету представлений на рисунках 2.21, 2.22, а результати представлені на рисунках 2.23 – 2.34.

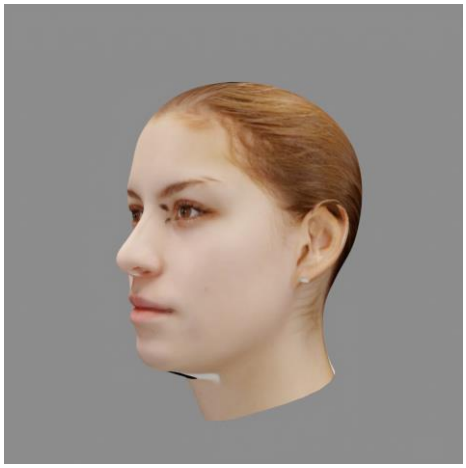


Рисунок 2.21 – Приклад зображення з поворотом по осі X на -50 градусів

Рисунок 2.22 – Приклад зображення з поворотом по осі Y на -20 градусів

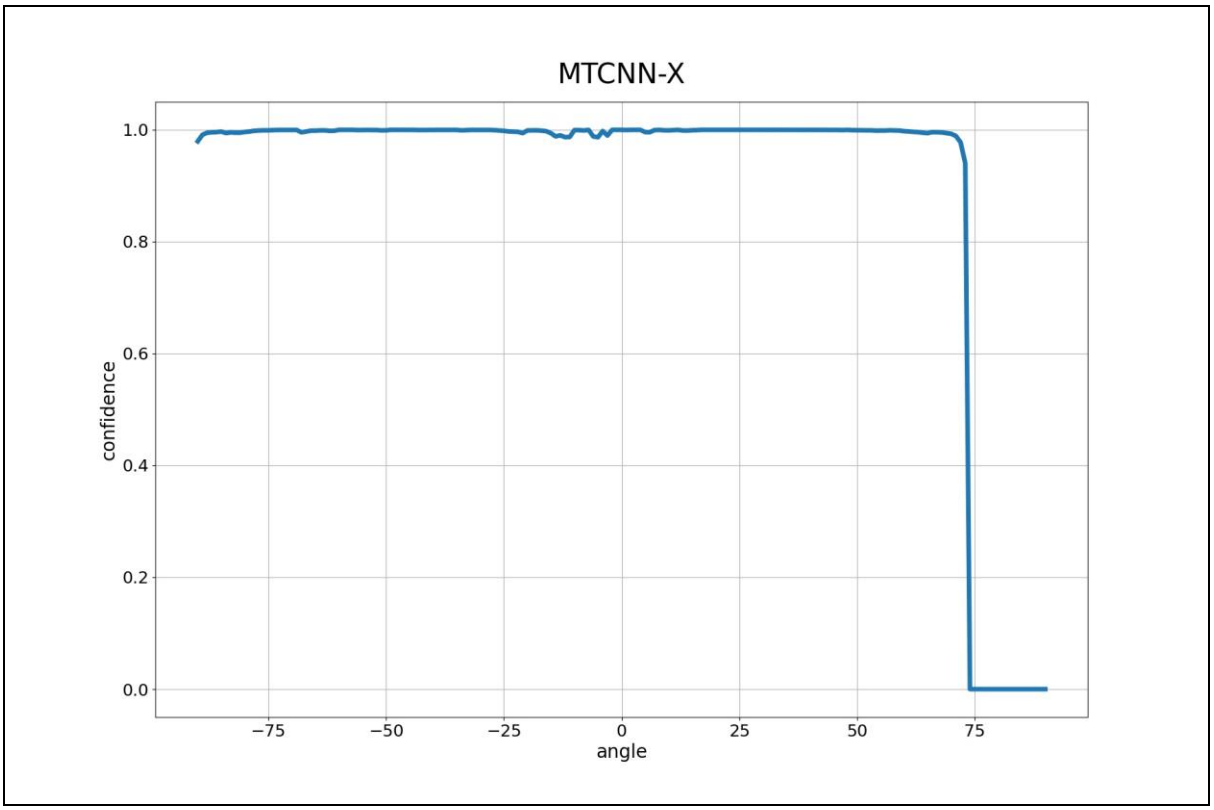


Рисунок 2.23 – MTCNN

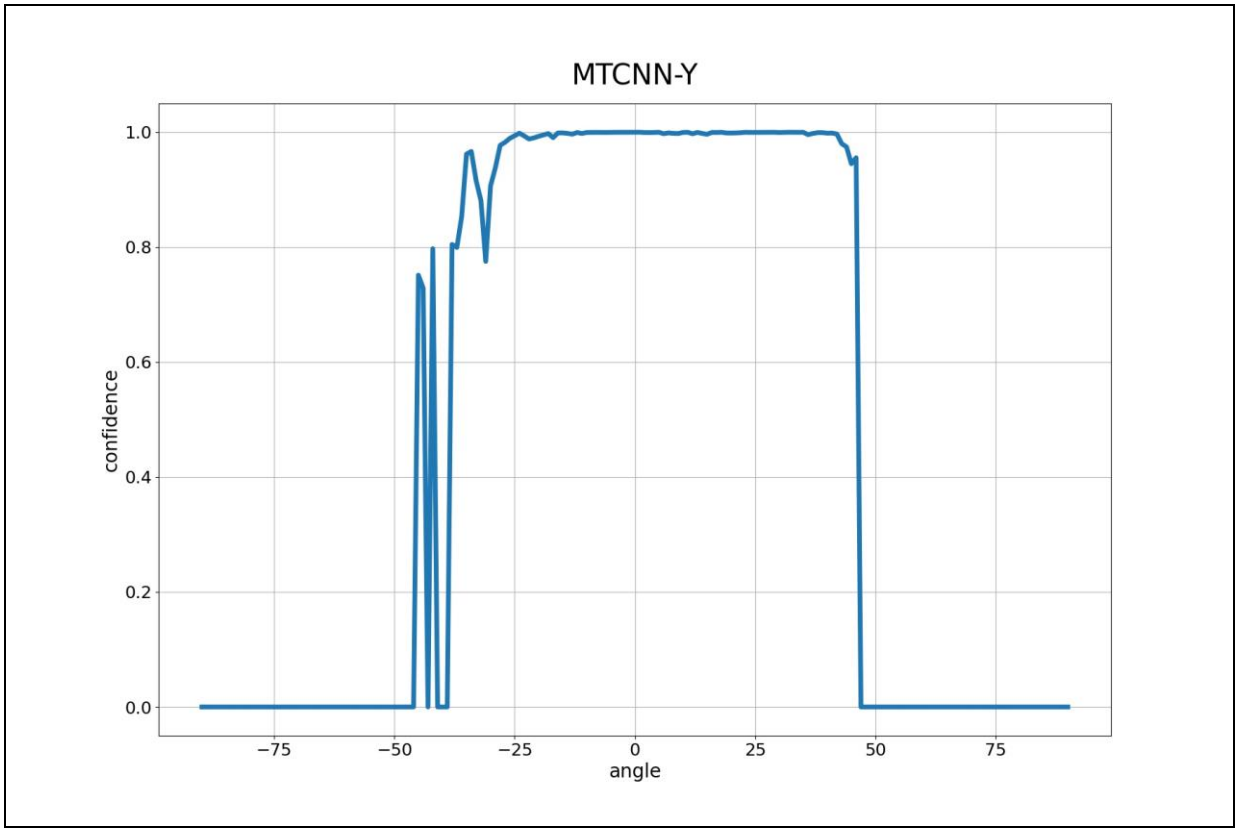


Рисунок 2.24 – MTCNN

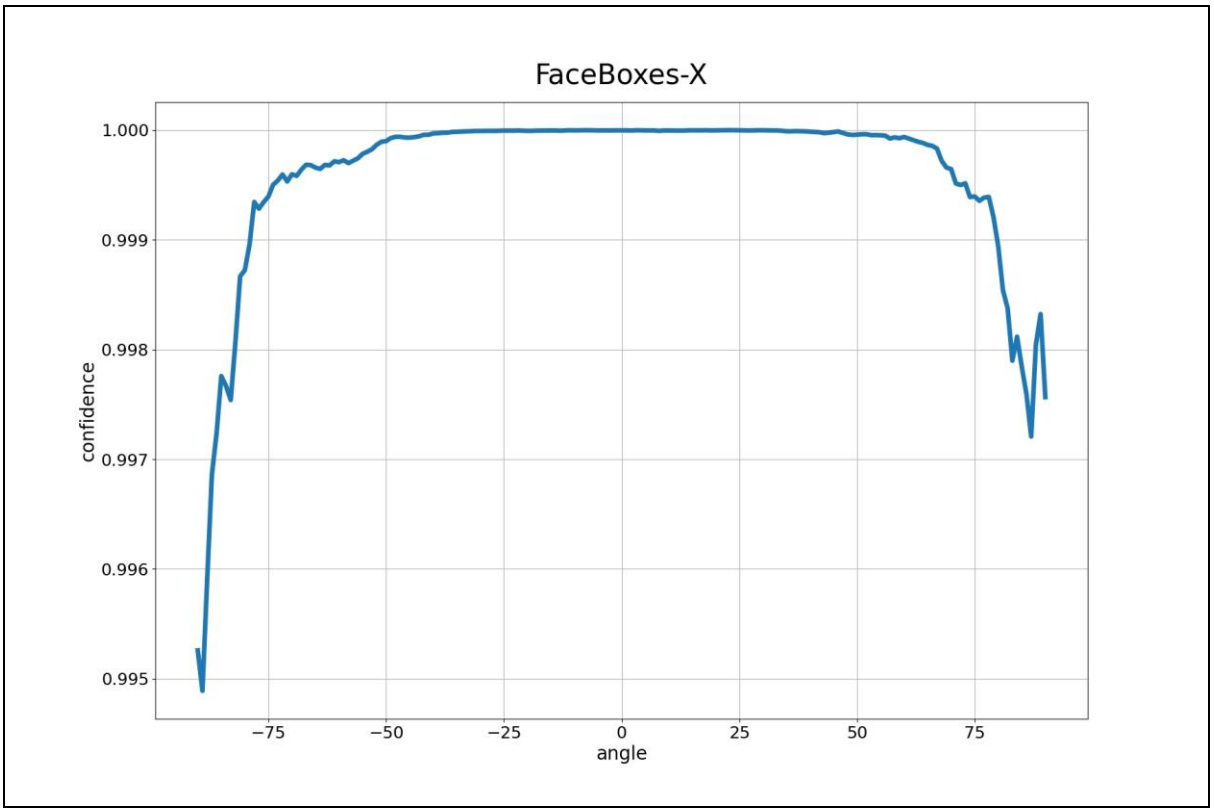


Рисунок 2.25 – FaceBoxes

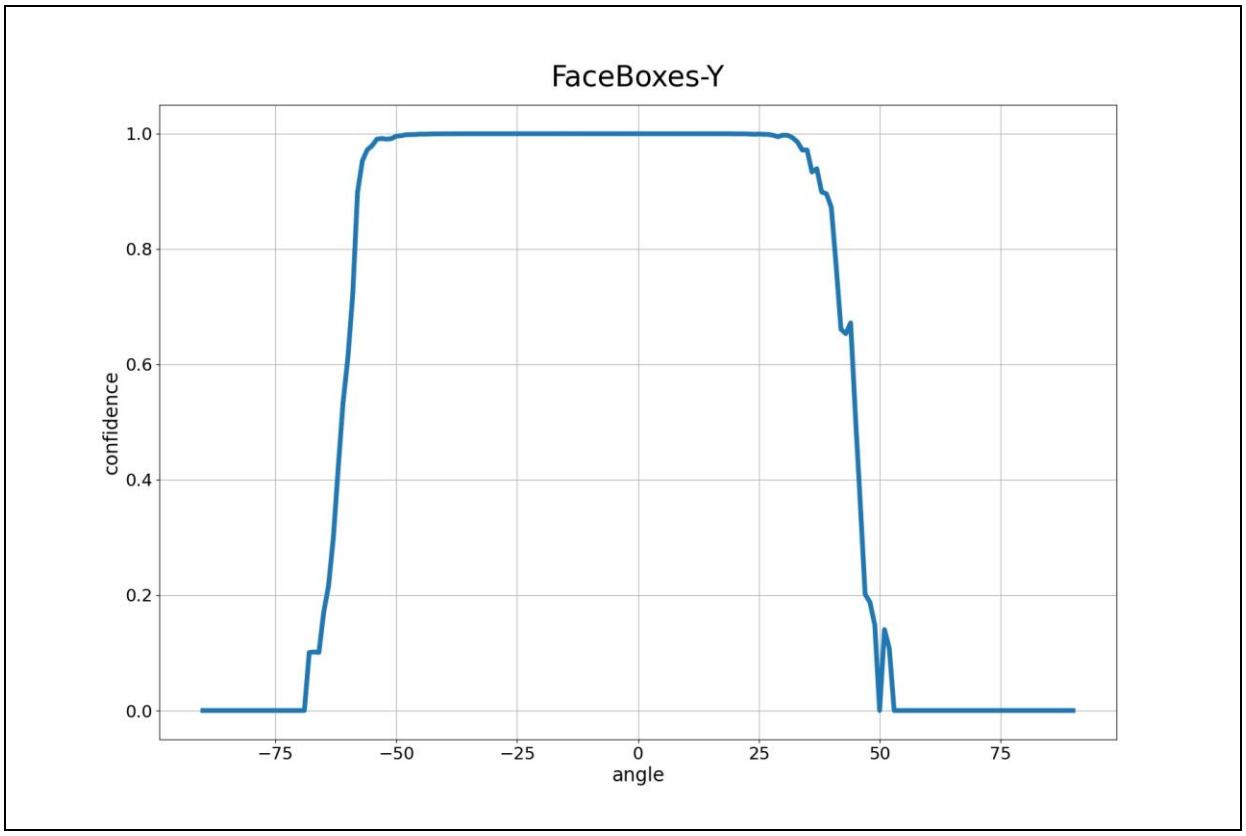


Рисунок 2.26 – FaceBoxes

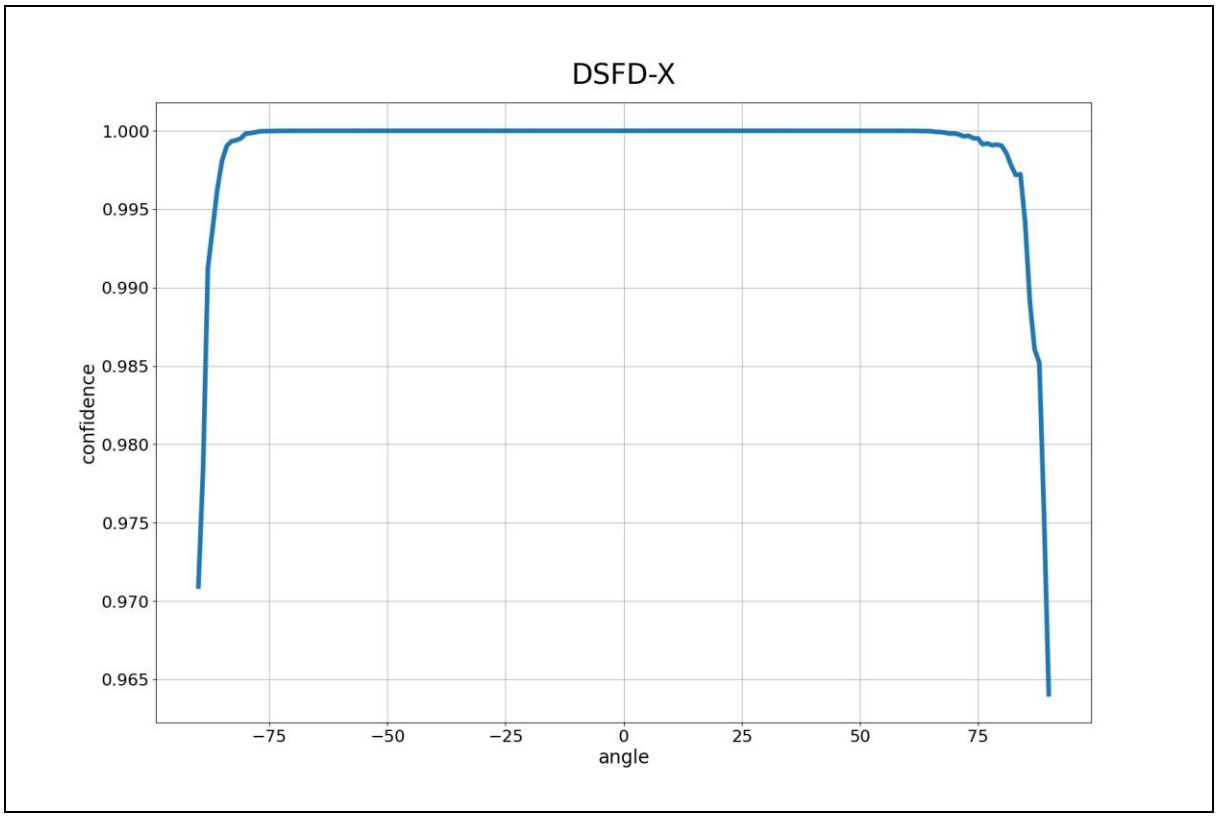


Рисунок 2.27 – DSFD

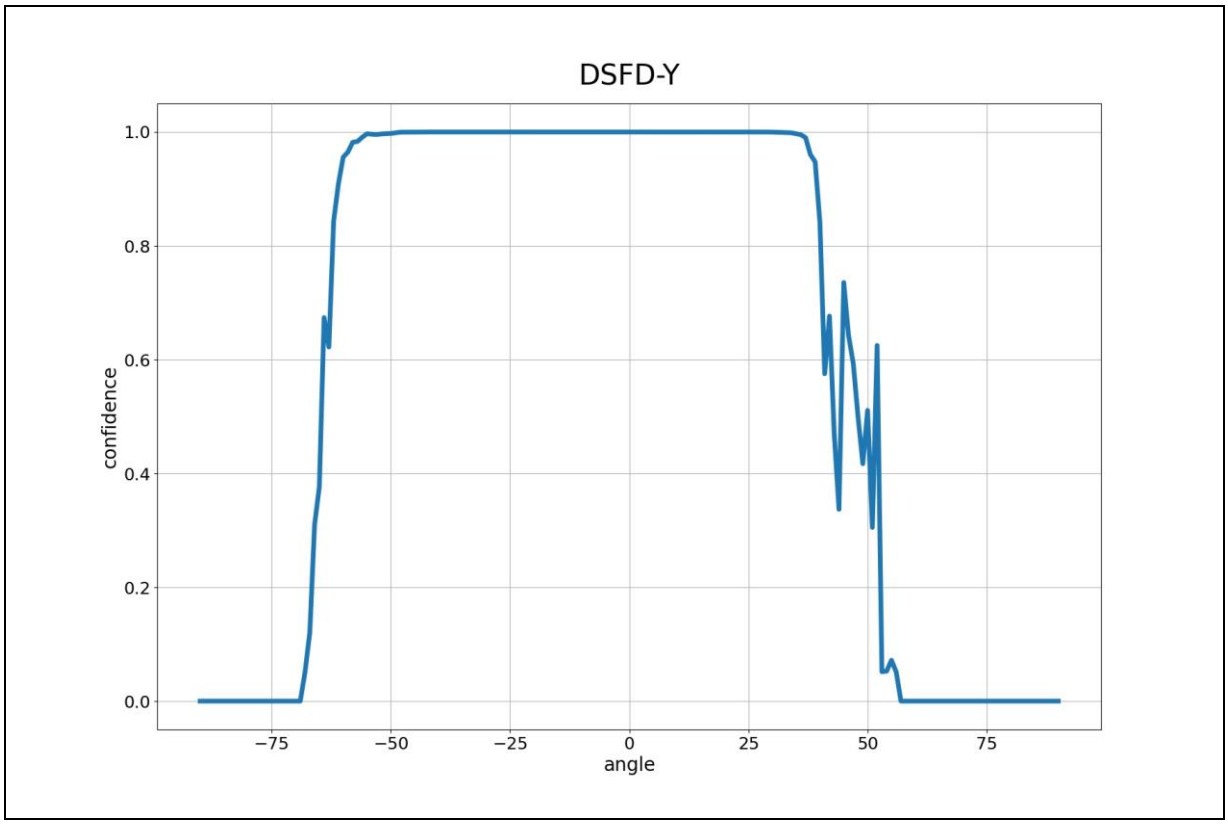


Рисунок 2.28 – DSFD

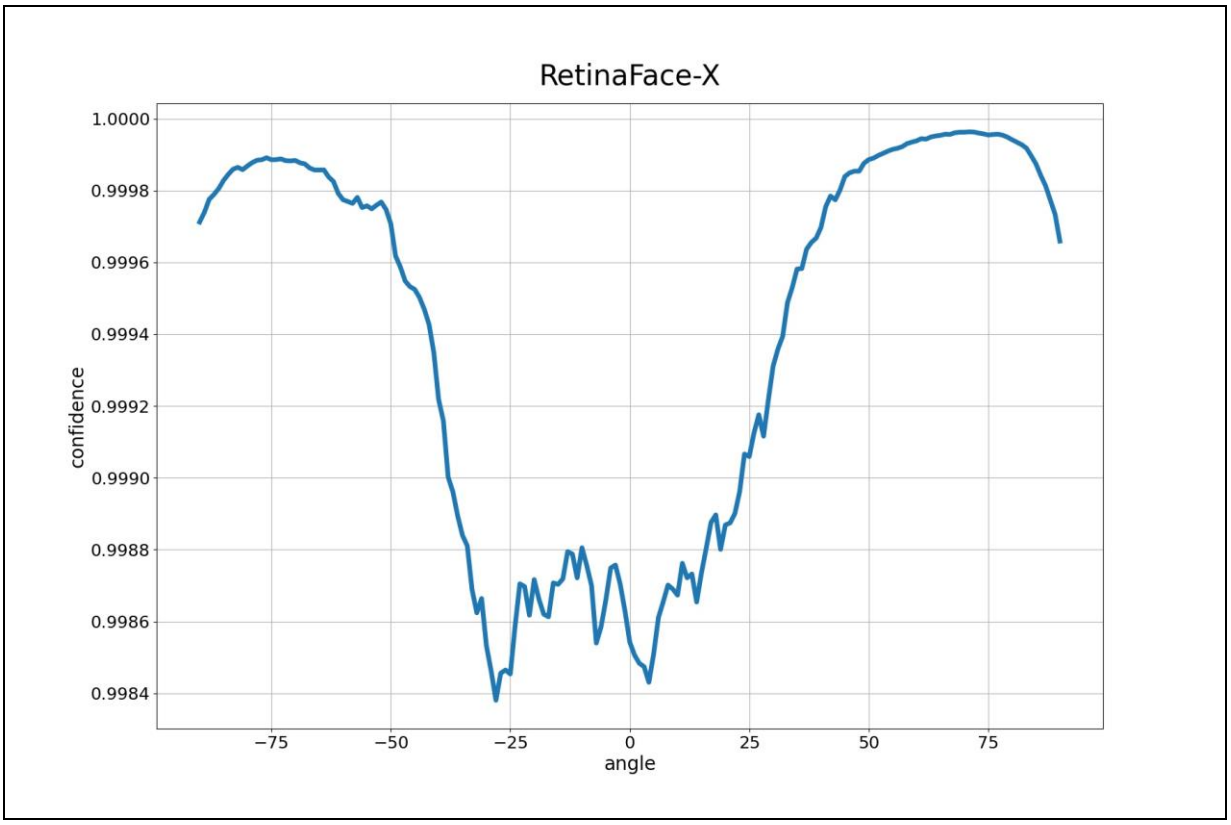


Рисунок 2.29 – RetinaFace

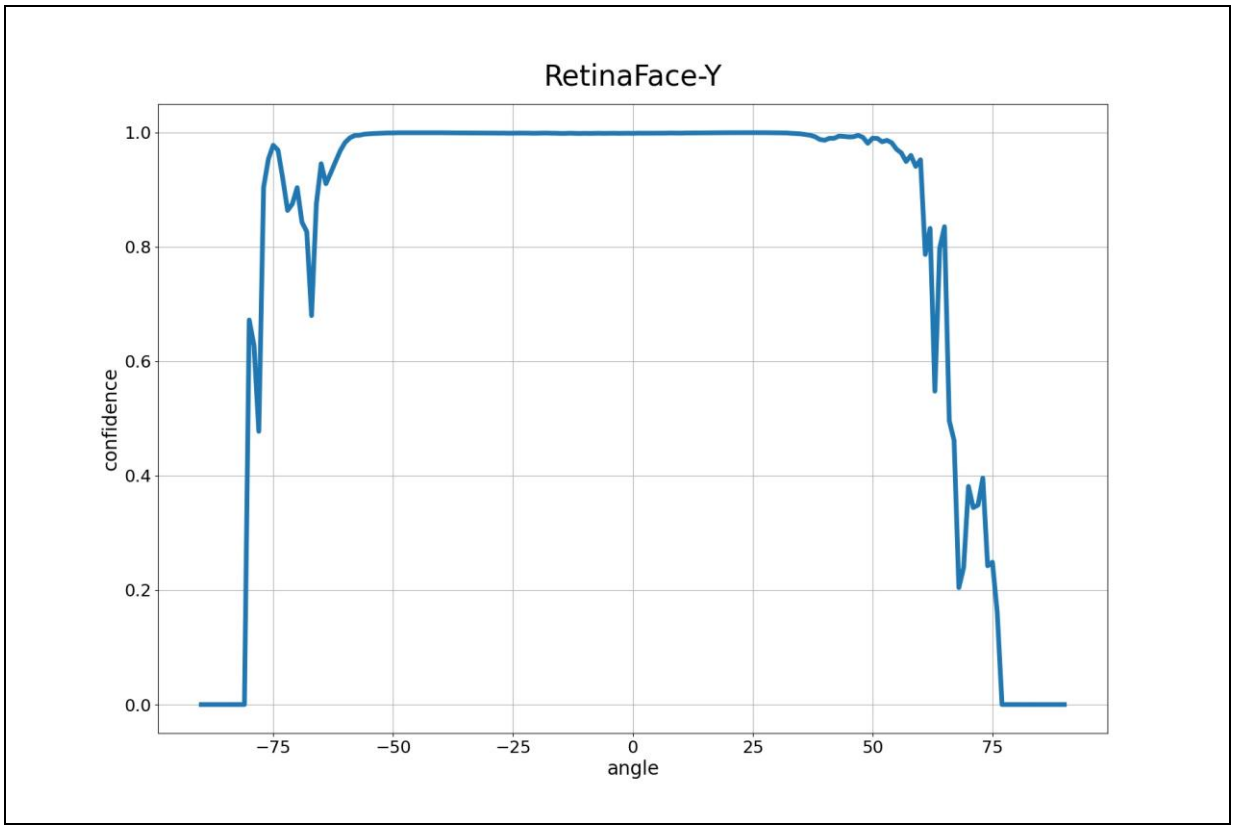


Рисунок 2.30 – RetinaFace

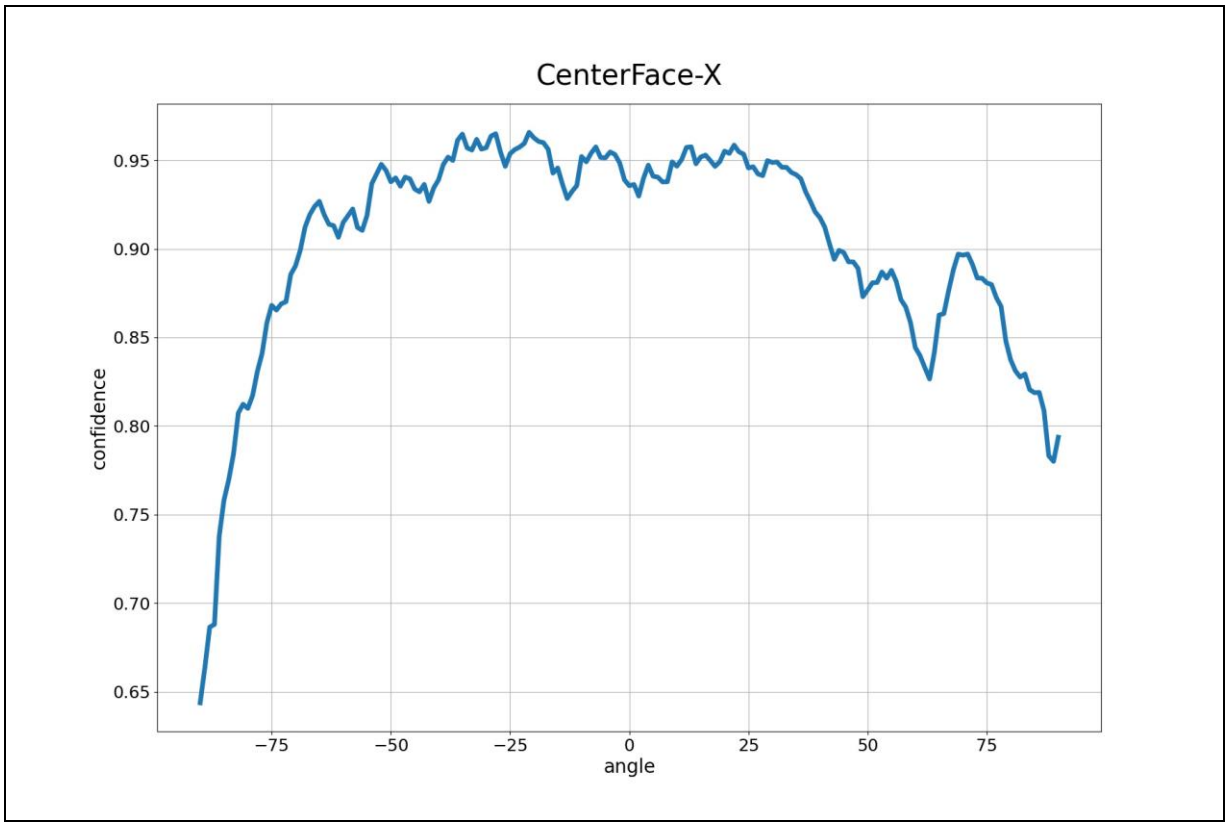


Рисунок 2.31 – CenterFace

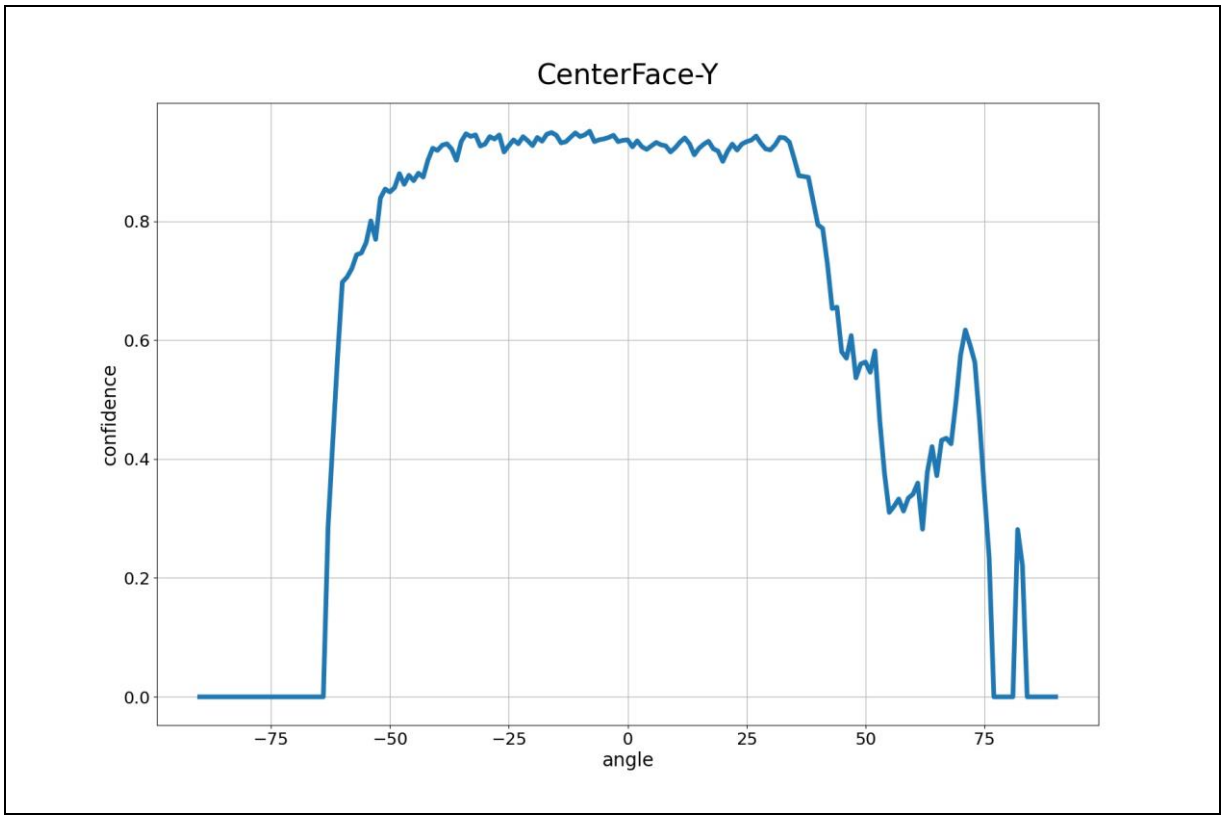


Рисунок 2.32 – CenterFace

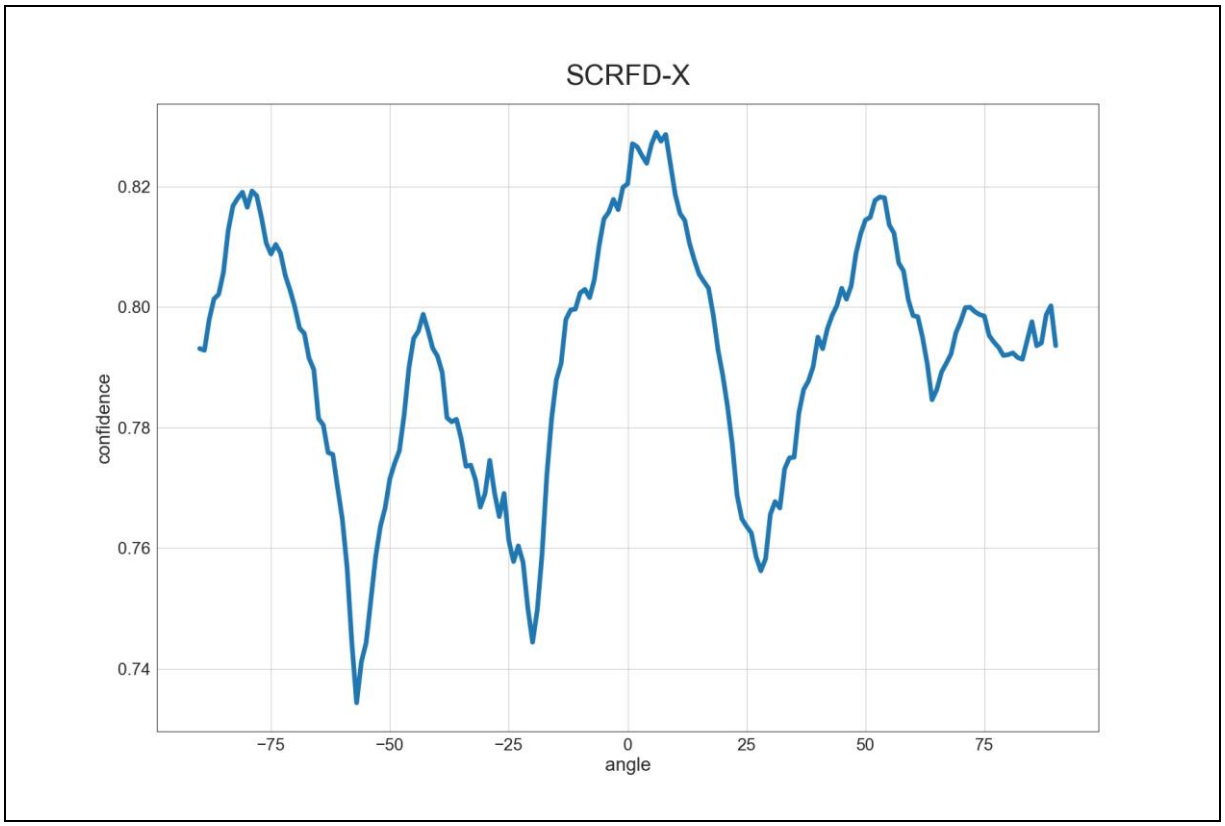


Рисунок 2.33 – SCRFD

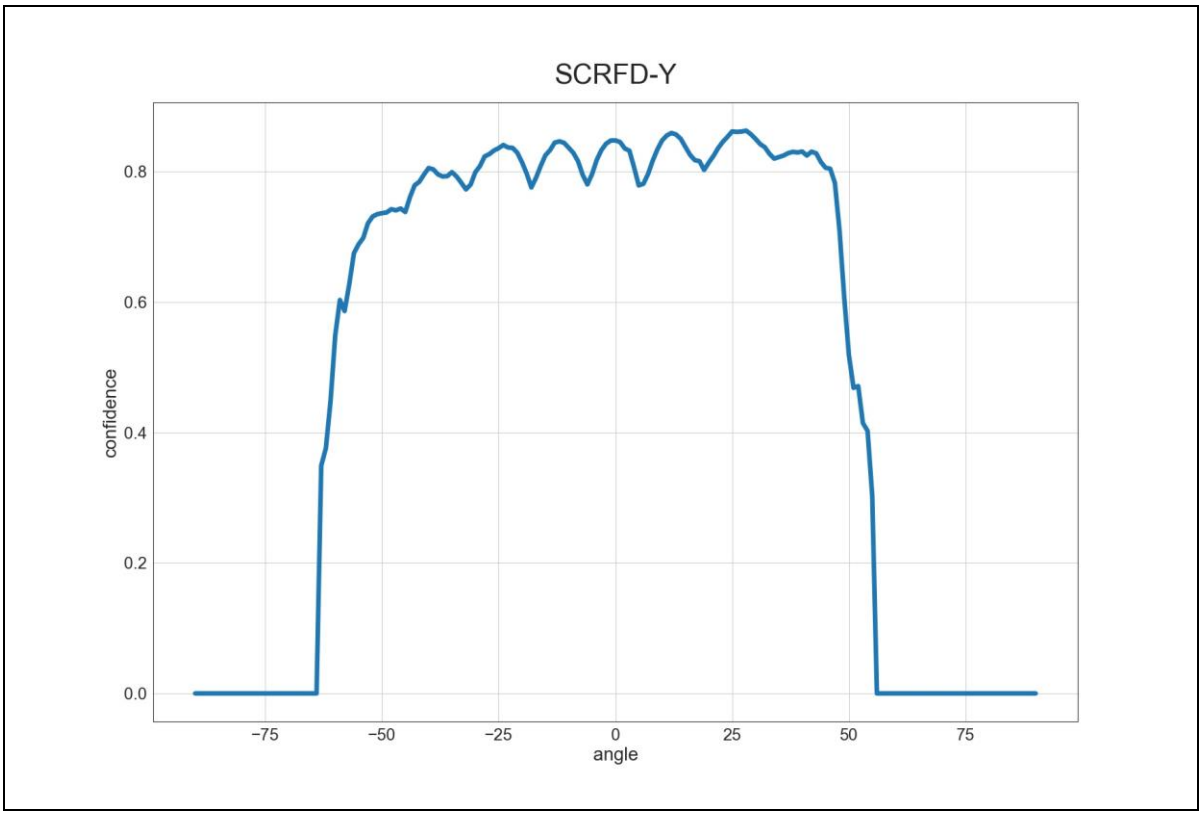


Рисунок 2.34 – SCRFD

2.1.2.2 Залежність якості детектування від розміру облич

Додатково було створено зображення із штучно згенерованими обличчями [51]. Обличчя були розподілені по зображення із кроком зміни розміру у 10 пікселів. Найменший розмір обличчя – 20 пікселів на 20 пікселів, найбільший – 310×310 пікселів. Для детекторів, які використовують NMS для об’єднання результатів, був виставлений поріг 0,3 для NMS. Зображення передавалося методам без додаткової його нормалізації чи змін у розмірі. Знайдені обличчя обводяться рамкою, колір якої залежить від вірогідності детектування, яку вернув метод та додатково були показані знайдені детектором ключові точки, якщо вони закладені у архітектуру. Результати тестів зображені на рисунках 2.35 – 2.40.

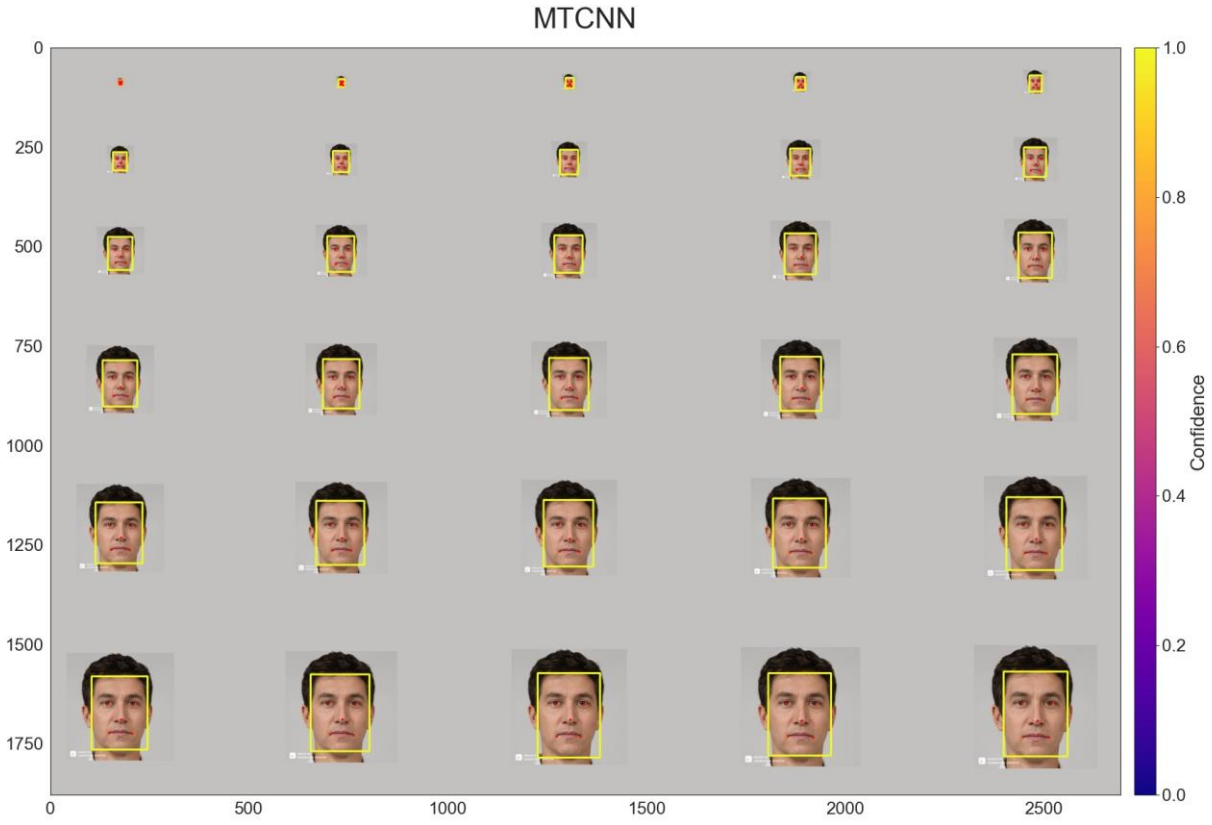


Рисунок 2.35 – MTCNN

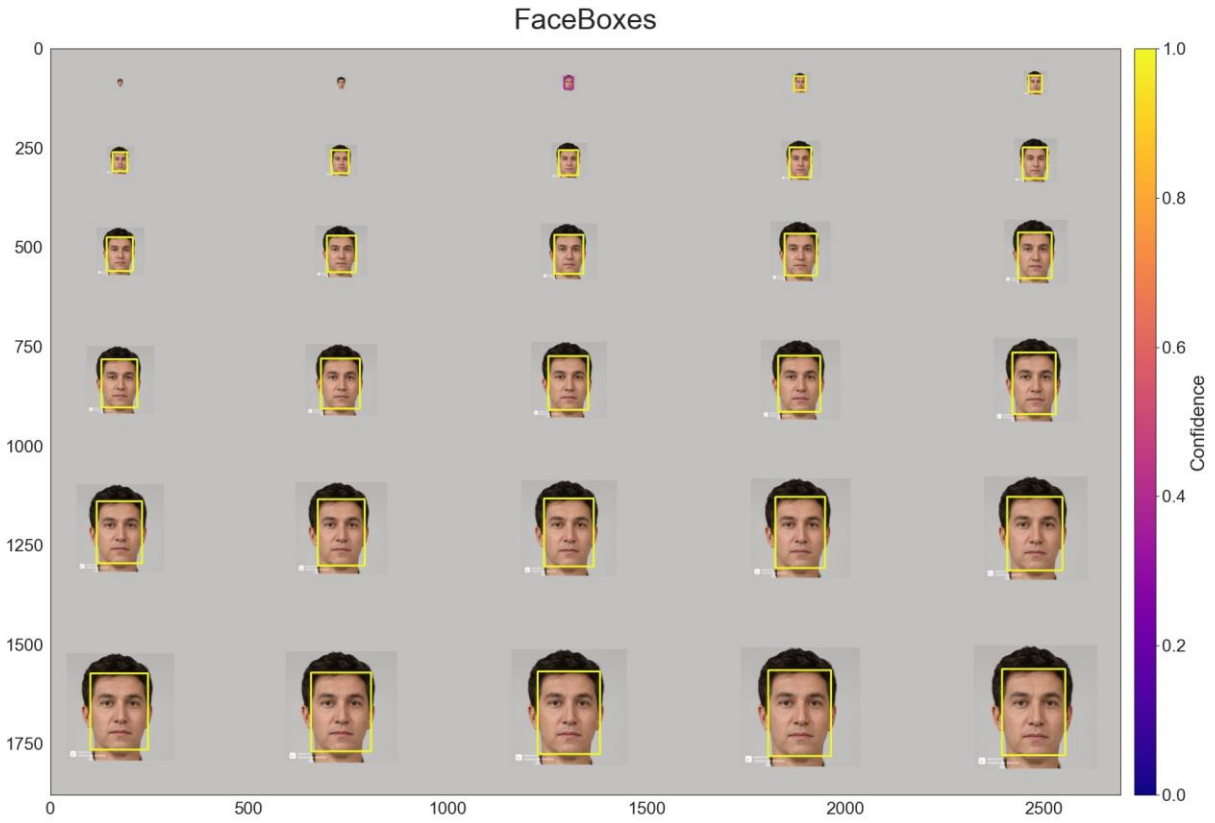


Рисунок 2.36 – FaceBoxes

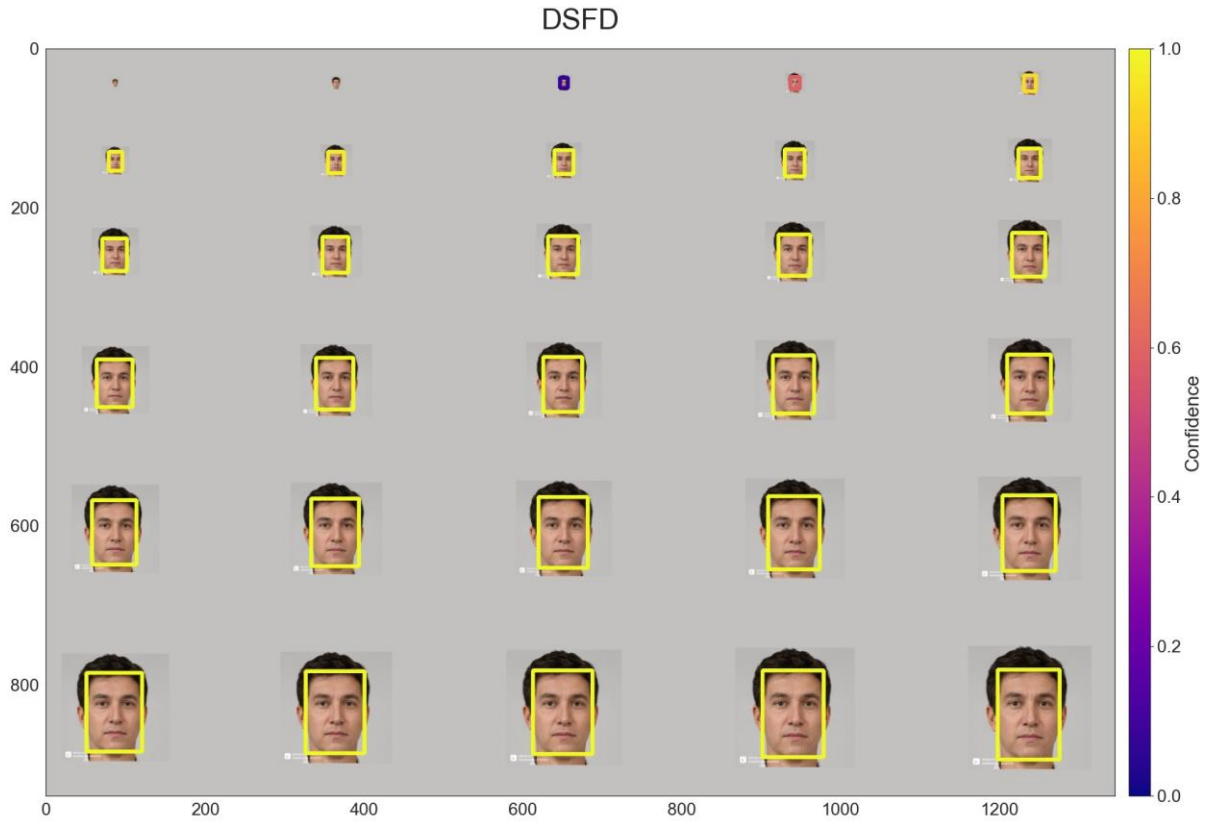


Рисунок 2.37 – DSFD

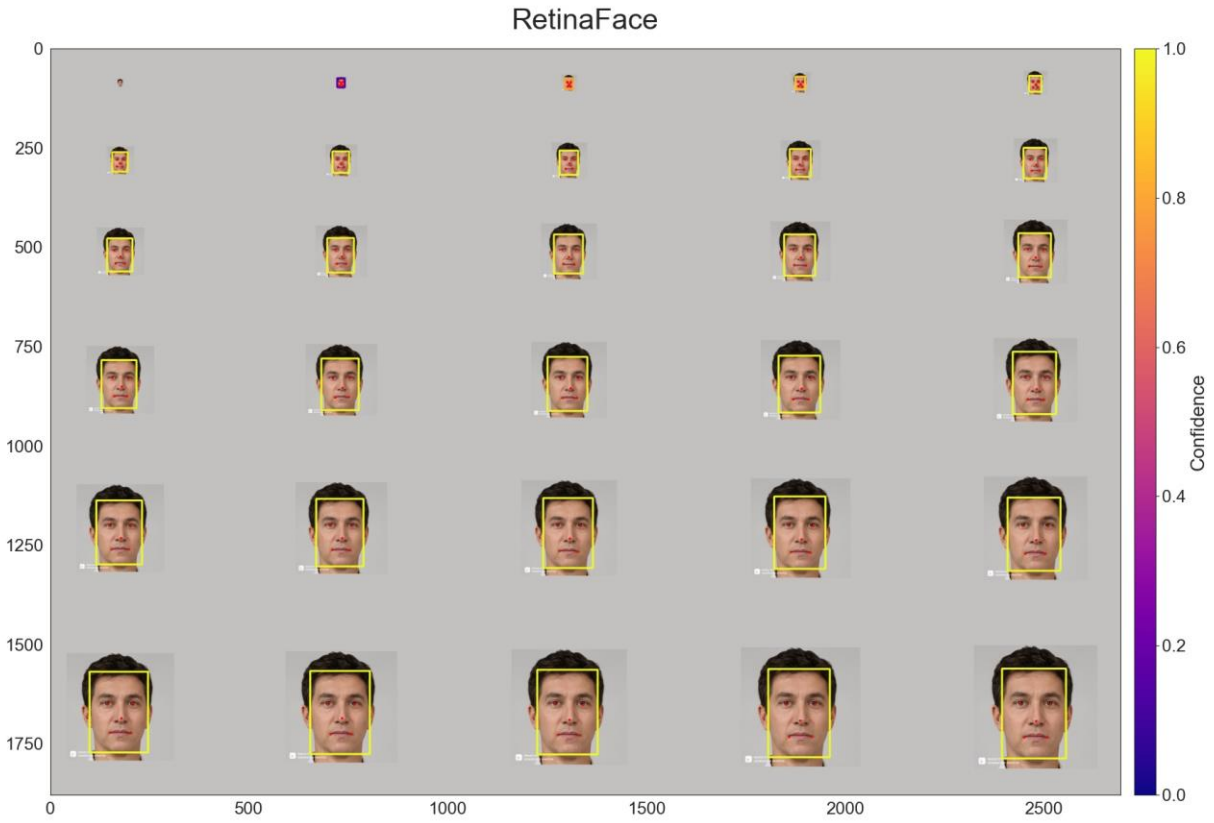


Рисунок 2.38 – RetinaFace

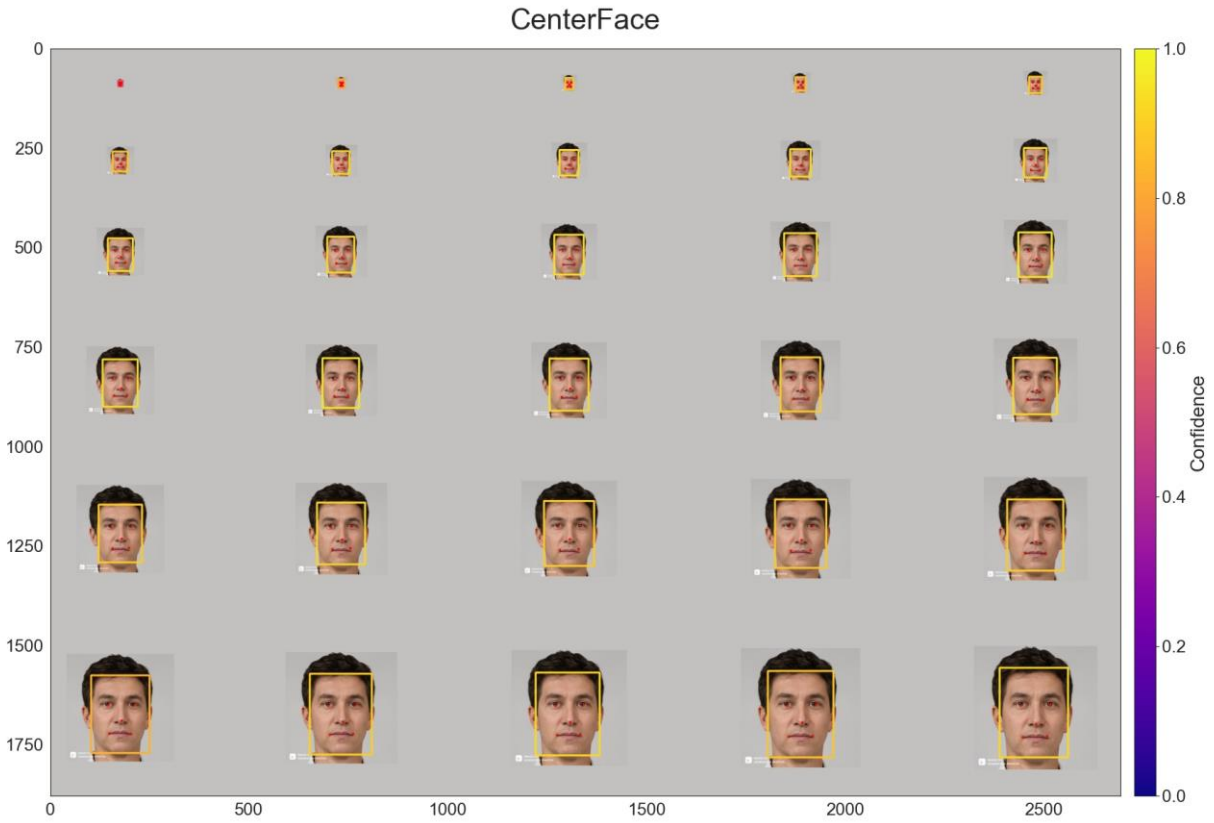


Рисунок 2.39 – CenterFace

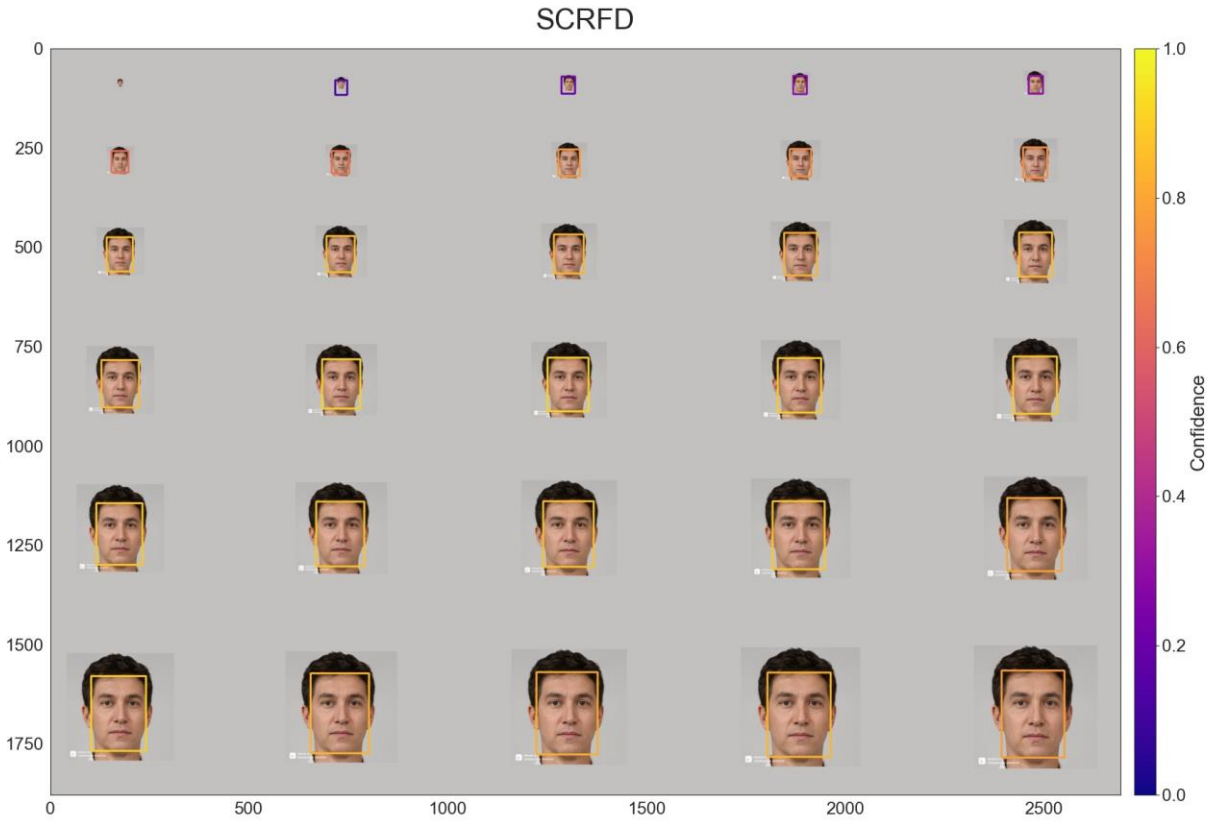


Рисунок 2.40 – SCRFD

2.1.2.3 Порівняння швидкодії

Порівняння методів за швидкодією було проведено на двох розмірах зображень: 640×480 (VGA) пікселів та 1280×720 (HD) пікселів. Це зумовлено тим, що ці розміри поширені для камер та, як було продемонстровано у попередніх тестах, детектування може відбуватися на обличчях навіть у 20 пікселів, тому необхідно дослідити швидкість мереж у різних умовах, бо можна зменшувати чи збільшувати роздільну здатність кадру для зміни швидкодії у ту чи іншу сторону. Якщо розмір кадру було змінено для пришвидшення детектування, то для етапу розпізнавання потрібно брати дані з оригінальних зображень та масштабувати знайдені рамки облич для них, бо для маленьких облич якість розпізнавання буде значно нижча. Результати замірів швидкості детектування облич показані на рисунках 2.41 – 2.52. Усі

заміри робилися з попереднім «прогрівом» мережі, пропуску першого детектування, на мобільній відеокарті NVIDIA GeForce 940MX.

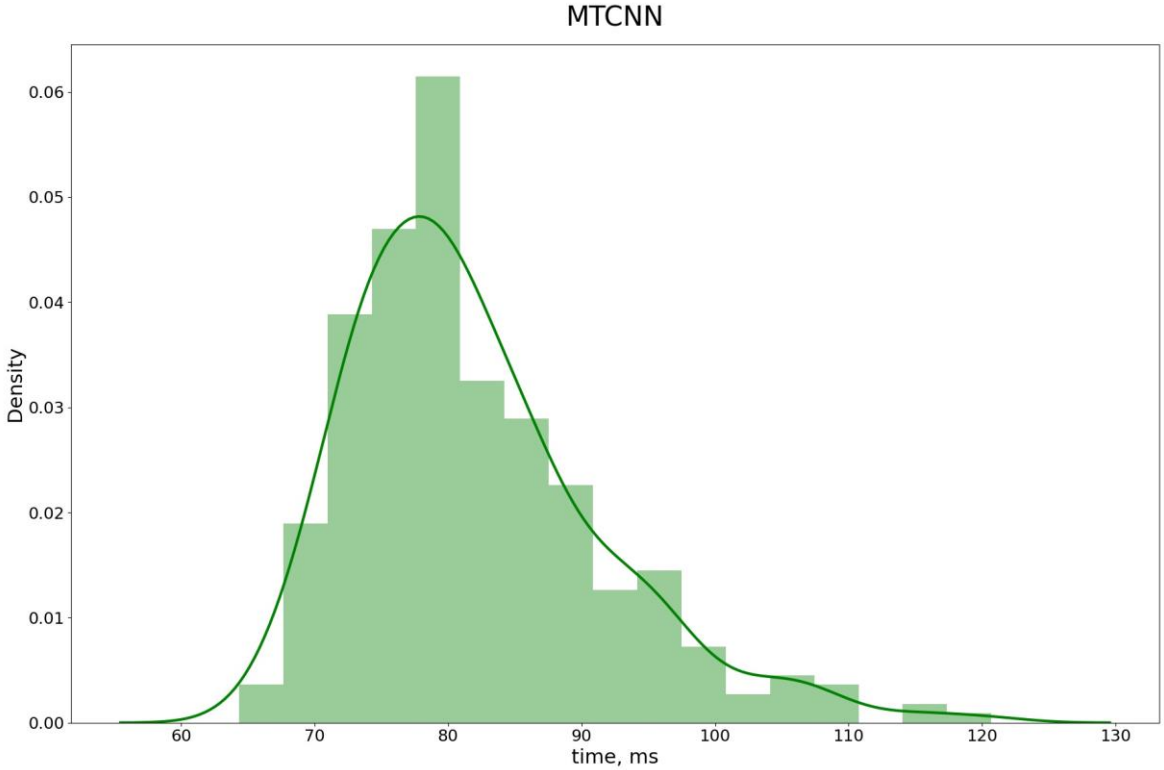


Рисунок 2.41 – MTCNN – 640×480 px

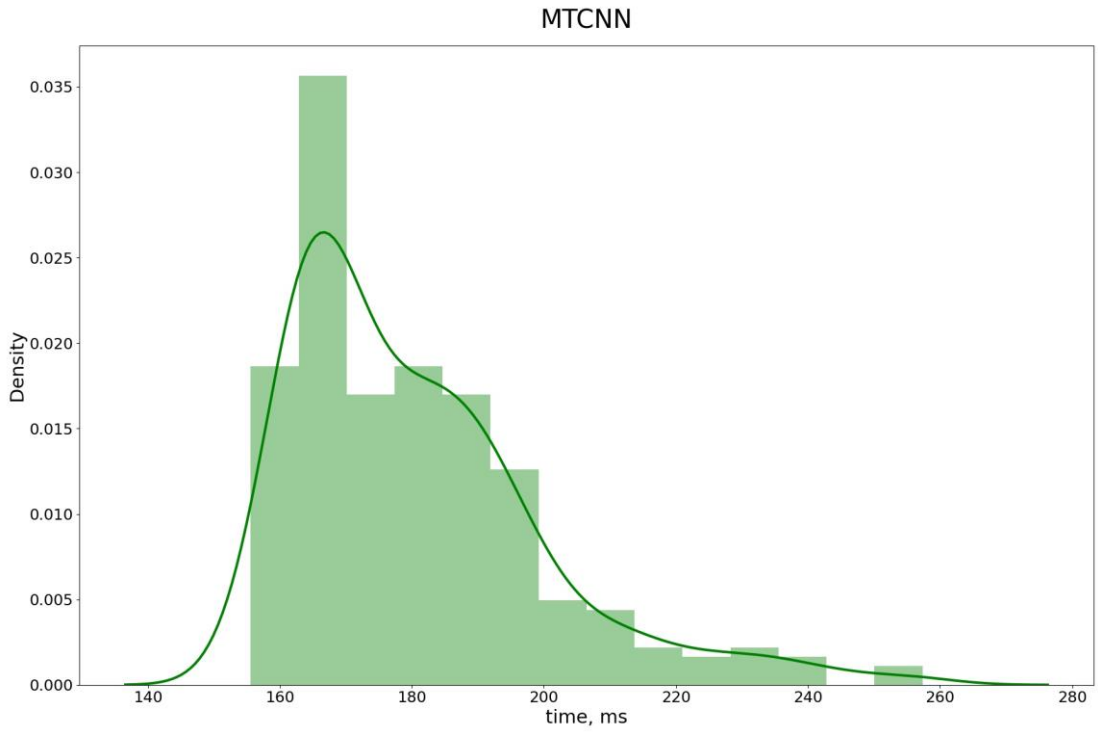


Рисунок 2.42 – MTCNN – 1280×720 px

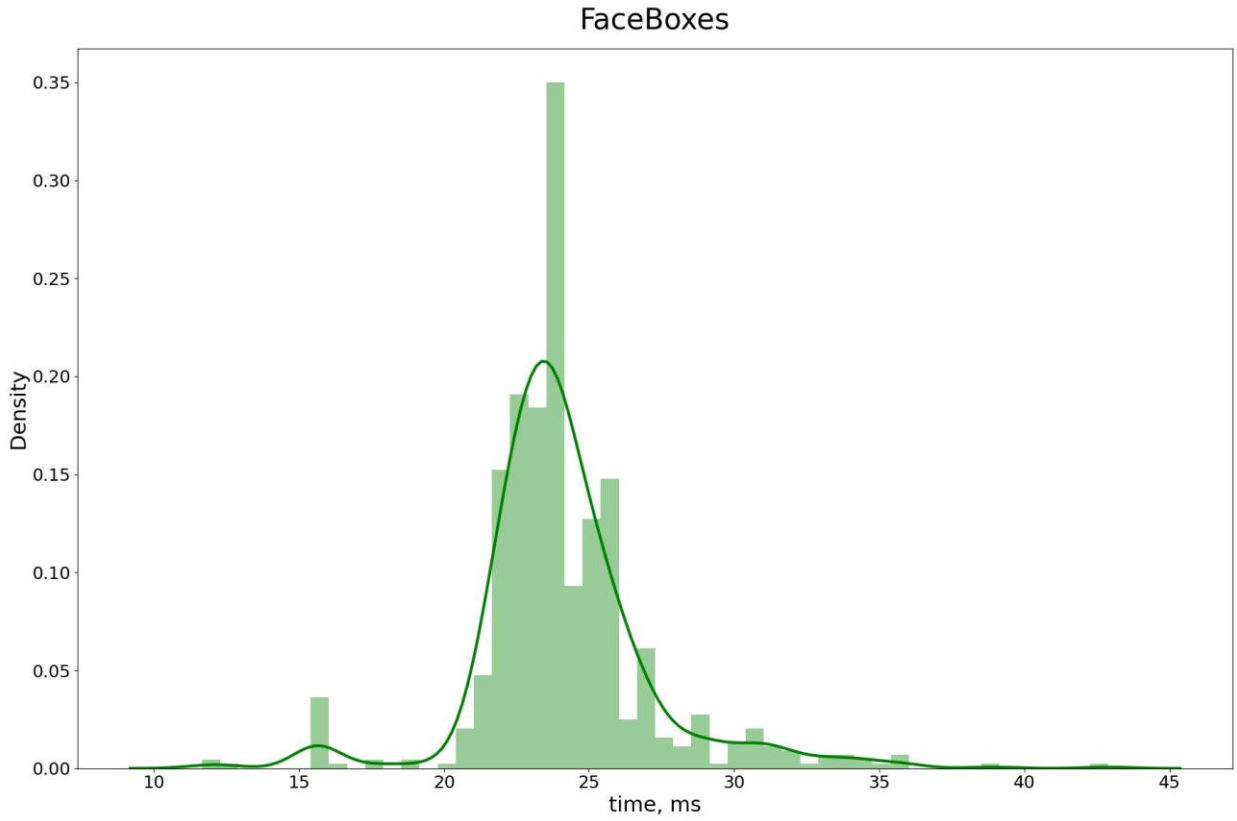


Рисунок 2.43 – FaceBoxes – 640×480 px

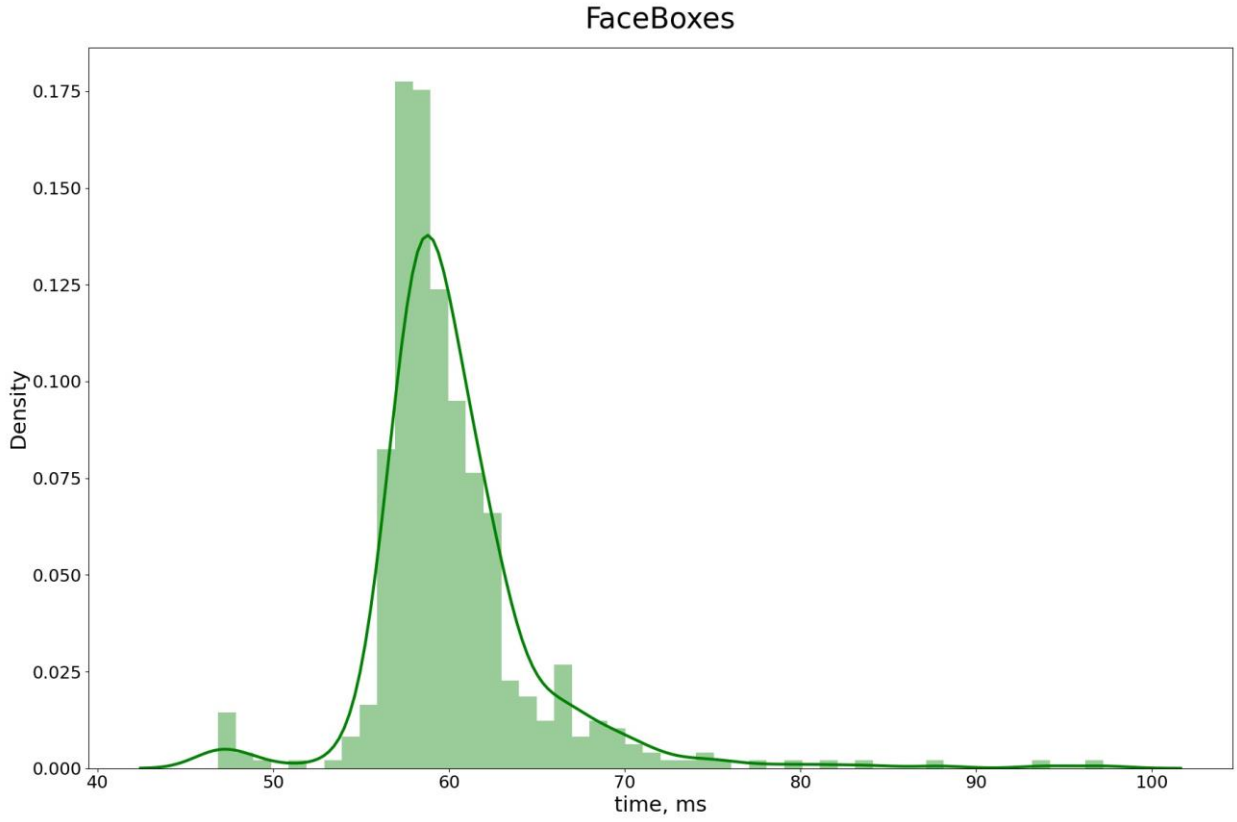


Рисунок 2.44 – FaceBoxes – 1280×720 px

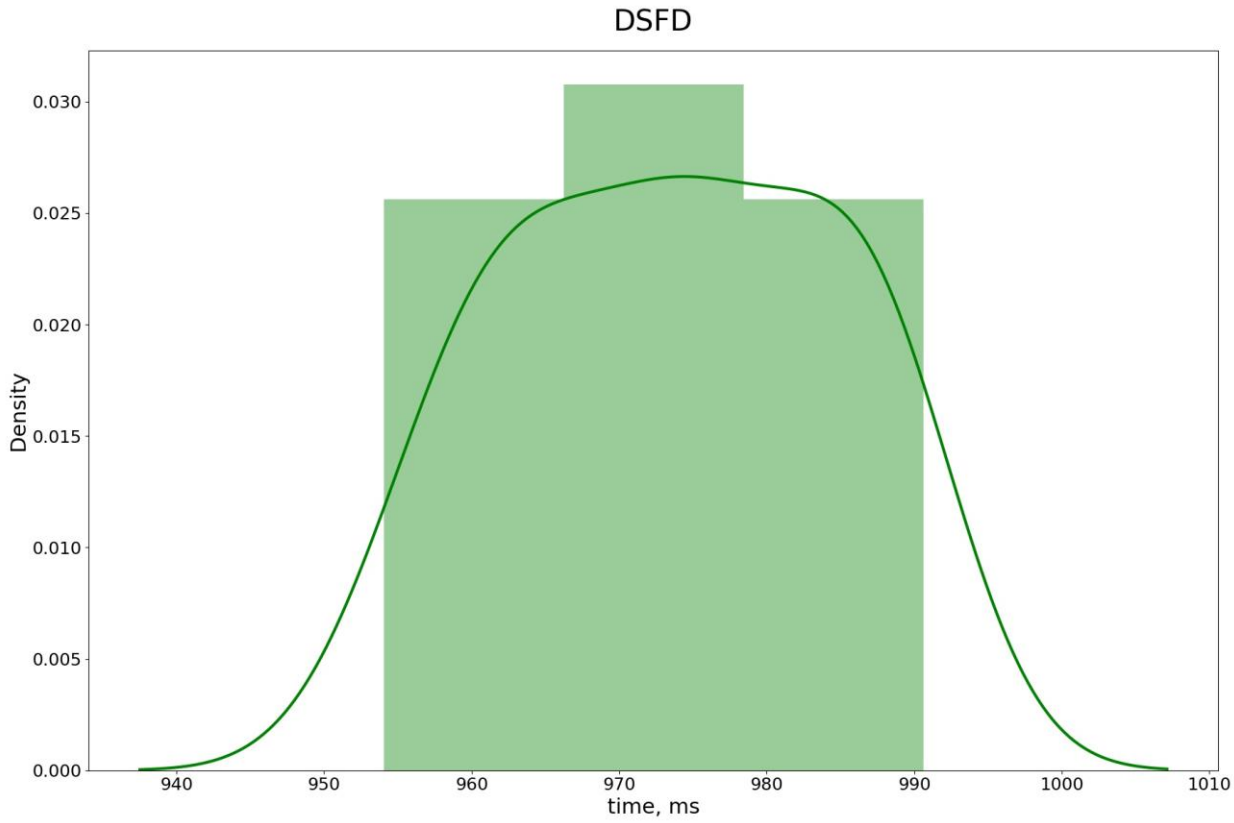


Рисунок 2.45 – DSFD – 640×480 px

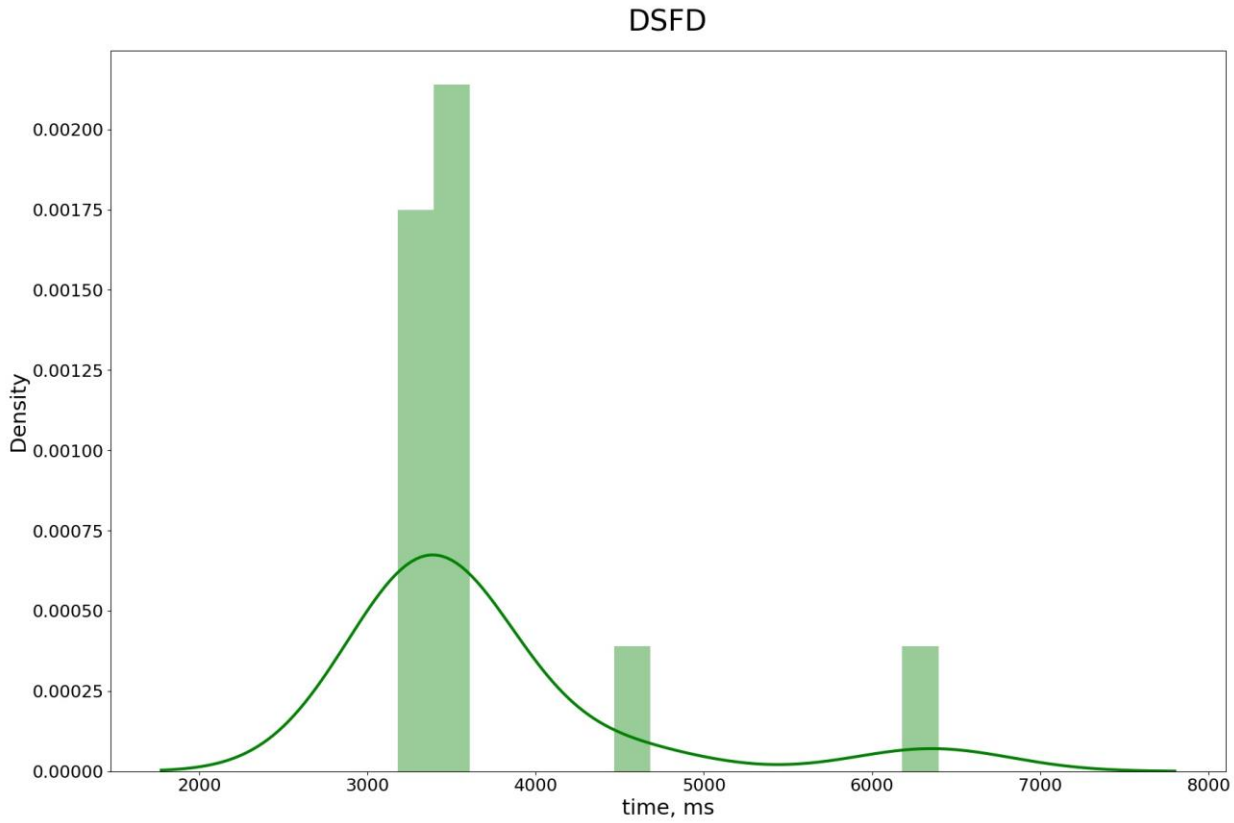


Рисунок 2.46 – DSFD – 1280×720 px

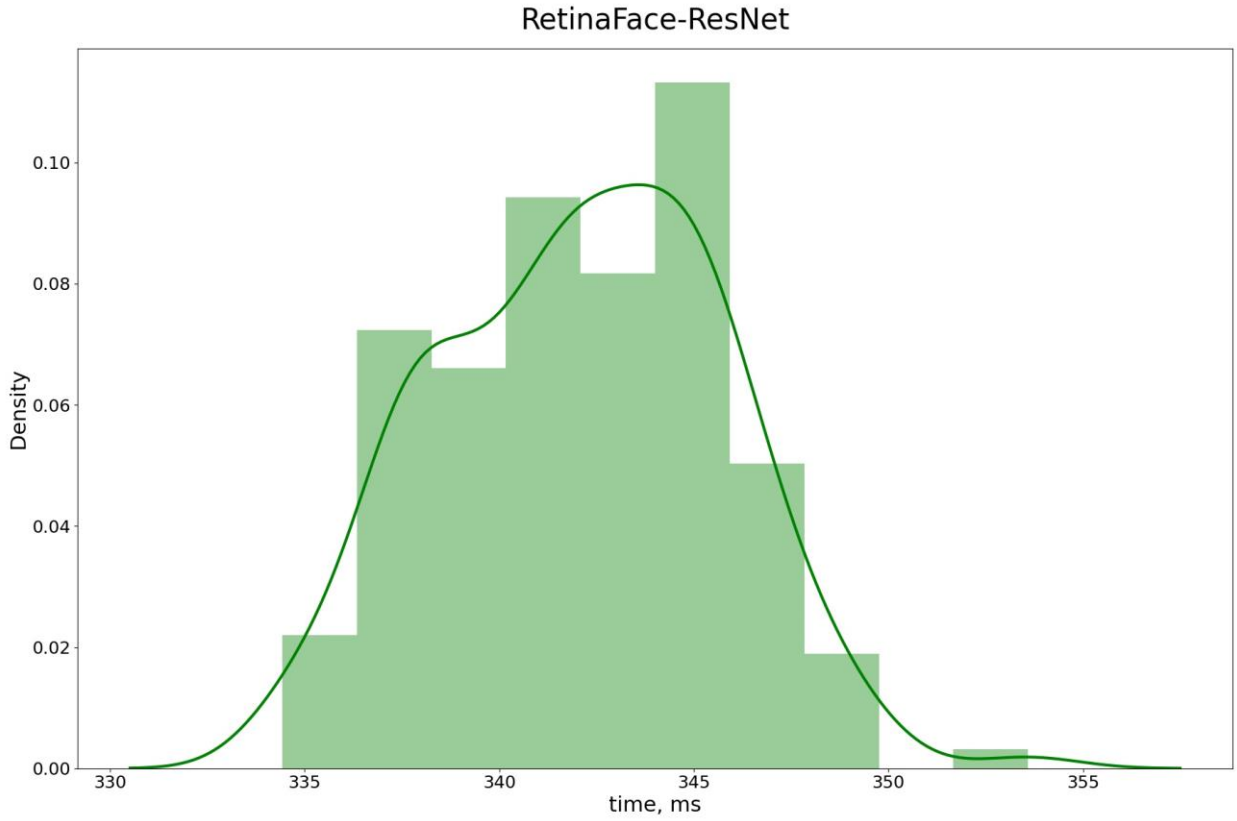


Рисунок 2.47 – RetinaFace – 640×480 px

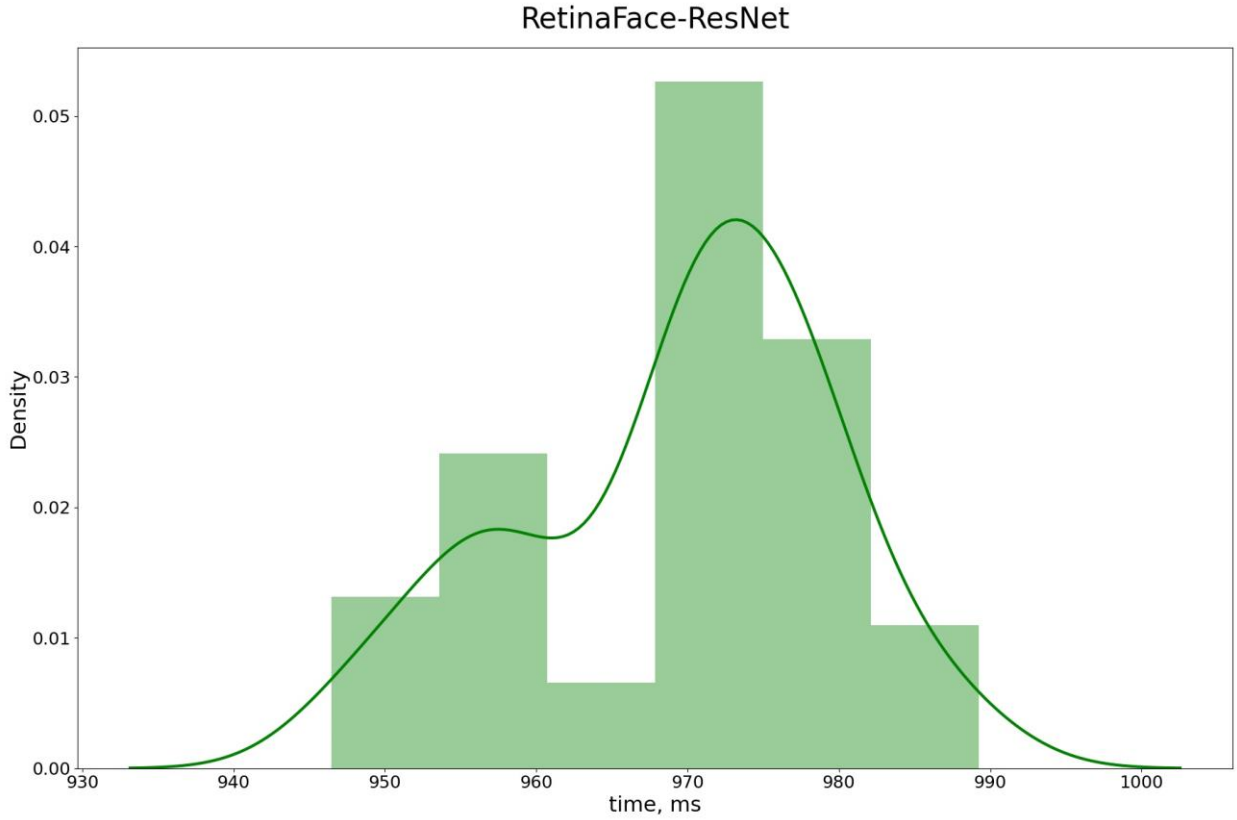


Рисунок 2.48 – RetinaFace – 1280×720 px

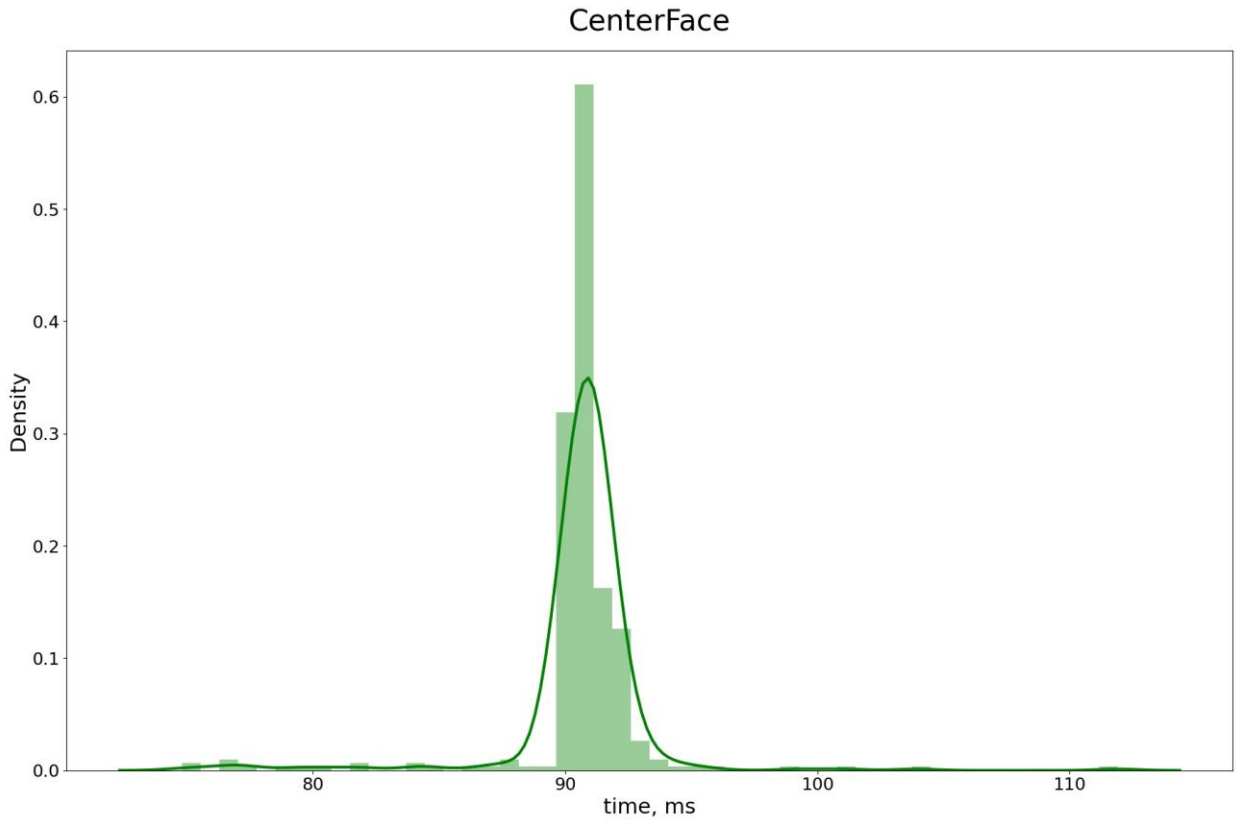


Рисунок 2.49 – CenterFace – 640×480 px

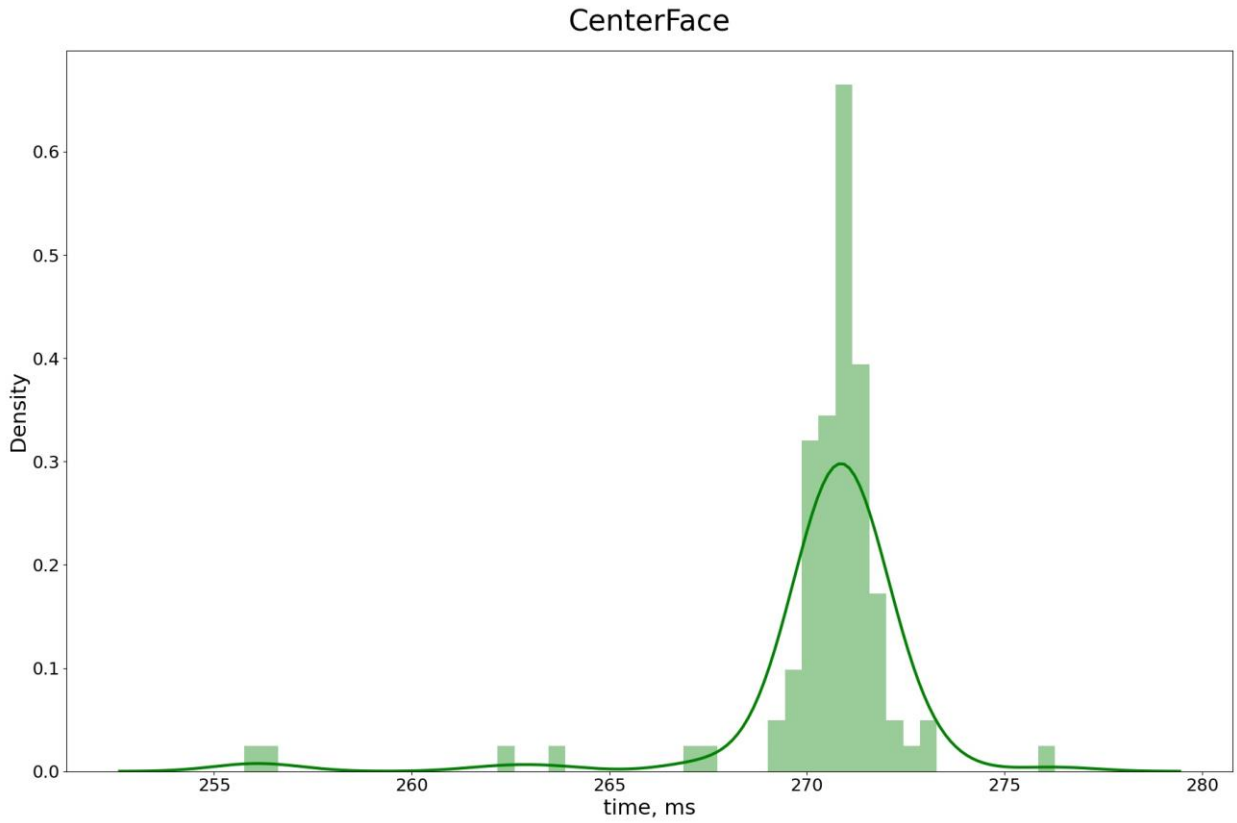


Рисунок 2.50 – CenterFace – 1280×720 px

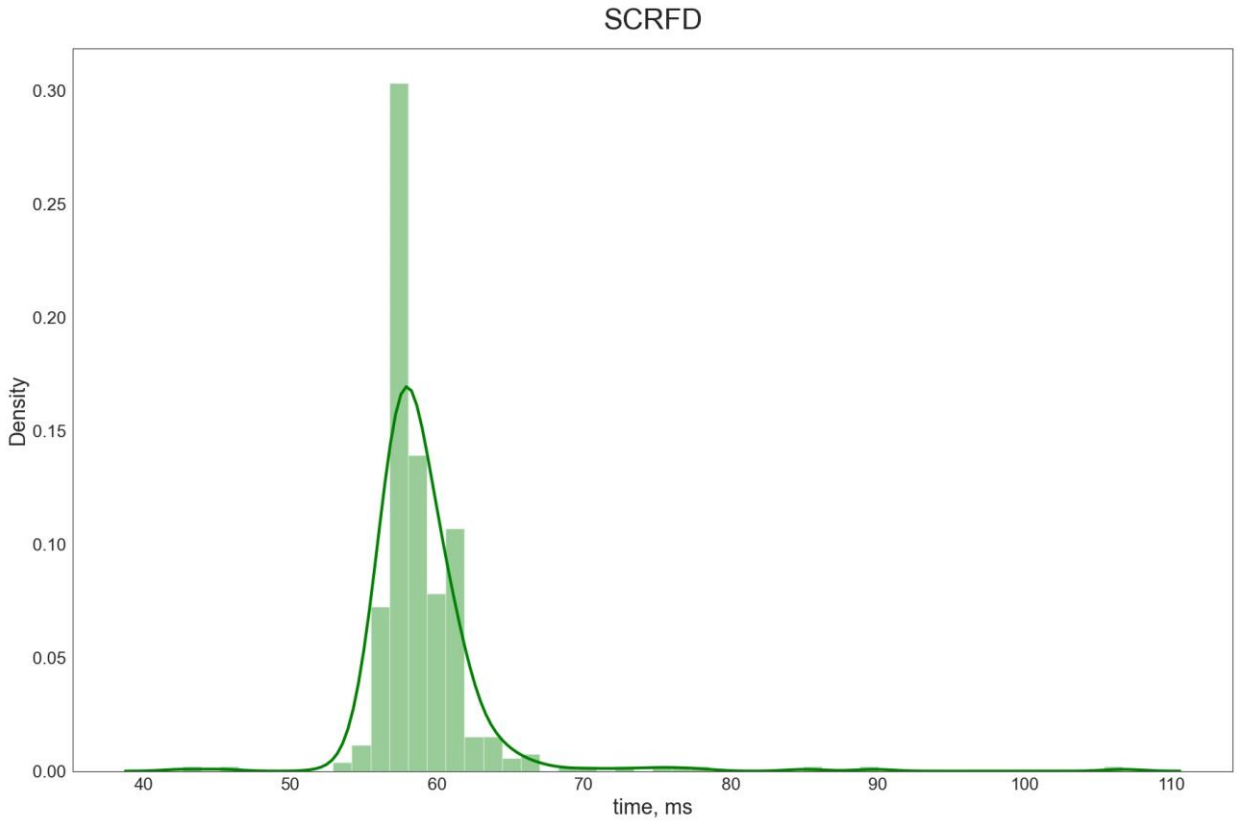


Рисунок 2.51 – SCRFD – 640×480 px

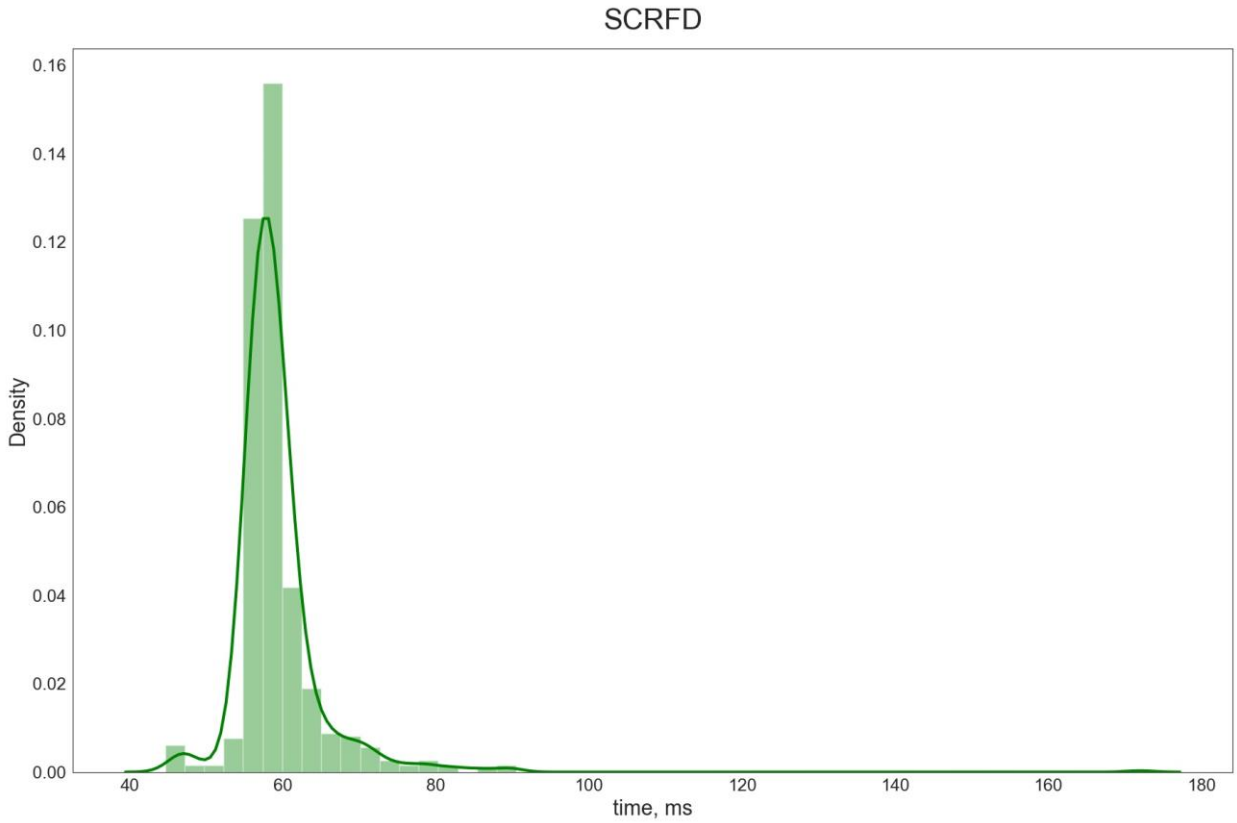


Рисунок 2.52 – SCRFD – 1280×720 px

2.1.3 Висновки щодо вибору методу детектування

Проаналізувавши результати тестів вони були зведені в таблицю 2.1.

Таблиця 2.1 – Результати досліджень

Метод	Діапазон кутів за віссю X, confidence ≥ 0,9	Діапазон кутів за віссю Y, confidence ≥ 0,9	Розмір першого обличчя (px), confidence ≥ 0,9	Середня швидкодія обробки одного кадру, ms		AP (1.5), WIDER Face Hard
				640×480	1280×720	
MTCNN	[-88;75]	[-30;46]	30×30	80	165	0,607
FaceBoxes	[-90;90]	[-60;35]	50×50	24	58	0,396
DSFD	[-90;90]	[-65;-40]	60×60	970	3200	0,900
RetinaFace	[-90;90]	[-65;60]	60×60	345	975	0,914
CenterFace	[-90;90]	[-43;35]	80×80	91	272	0,873
SCRFD0.5GF	∅	∅	–	54	55	0,694

Головним критерієм для створюваної системи є швидкодія, взявши за увагу ще те, що образу обробляється багато камер, тому обираємо поріг у 100ms на детектування, що дорівнює 10 FPS. За цим критерієм не проходять наступні детектори: RetinaFace, DSFD, CenterFace 1280×720, MTCNN 1280×720. FaceBoxes найшвидший, але варто врахувати те, що він не вертає ключові точки, за якими необхідно провести нормалізацію обличчя, тому необхідно після детектування додавати ще один метод для знаходження цих точок, що сповільнить весь процес. Далі аналізуємо кути детектування. Зважаючи на те, що після детектування йде розпізнавання, якість його стрімко падає, якщо значення кутів перевищую ±20 градусів, тому приймемо цей інтервал для оцінювання. З решти детекторів SCRFD не пройшов тест за порогом впевненості детектування, але, як видно з графіків, якщо поріг

зменшити до 0,7, цей детектор буде задовольняти умовам. Так само і із розміром облич, якщо понизити поріг до 70%, то усі детектори будуть задовольняти умовам. Останнім критерієм залишається AP, як бачимо, детектор FaceBoxes хоч і швидкий, гарно справляється з детектуванням під кутами і різними розмірами, проте на великому наборі даних його точність не здатна конкурувати з іншими детекторами, так само як і точність MTCNN. Також є дослідження [52], де MTCNN схильна до adversarial атаки [53]. Тому, проаналізувавши усі обрані детектори, робимо висновок, що для системи безпеки підприємства необхідно обрати детектор CenterFace 640×480 або SCRFD, але із зниженими порогами. Додатковою перевагою CenterFace є знаходження ключових точок для нормалізації обличчя для його розпізнавання, не має необхідності додавати окремий метод для їх знаходження, на відміну від SCRFD.

2.2 Математична модель нормалізації детектованного обличчя

Для нормалізації повороту обличчя навколо його центральної точки (носу) можна використати афінні перетворення, де кут повороту знаходиться за ключовими точками очей і буде дорівнювати α -90 (рис. 2.53). Між точками, які відповідають положенню очей, проводиться пряма, та перпендикуляр до її центру. Кут між перпендикуляром і прямою мінус 90 градусів і буде шуканим кутом, на протилежне значення якого необхідно провести афінні перетворення. Додатково необхідно порахувати коефіцієнт масштабування, щоб усі обличчя були однакового розміру. Це можна зробити, знаючи розмір поточної поточного обличчя, який вернув детектор, та необхідного.

Рисунок 2.53 – Обчислення кута повороту для нормалізації повороту обличчя

Ще одним методом нормалізації є нормалізація бокових поворотів за допомогою GAN мереж [54], які можуть генерувати нове зображення з існуючого. Цей підхід можна використати, подавши на вхід повернуте зображення та очікувати його перетвореним до фронтального положення. У роботі FreeStyleGAN [55] створили GAN мережу, яка видає зображення, в залежності від заданих параметрів розташування камери на сцені (рис. 2.54 та рис. 2.55).

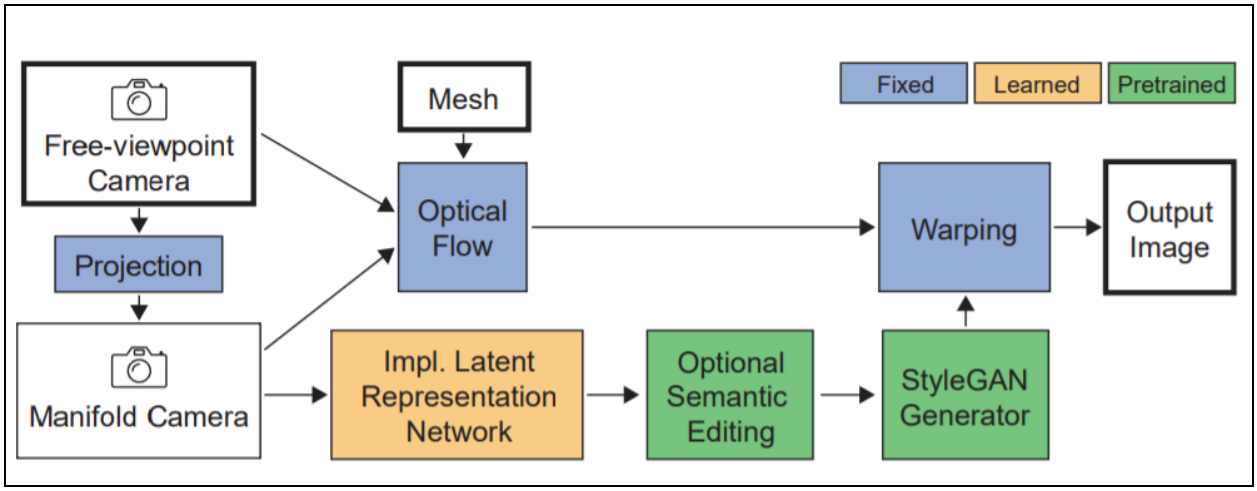


Рисунок 2.54 – Описання методу FreeStyleGAN [55]

Рисунок 2.55 – Приклад генерування одного облчиччя у різних положеннях [55]

Недолік цього підходу полягає у тому, що, по-перше, це потребує більше ресурсів, ніж перший метод, по-друге, створює нове зображення, ознаки якого можуть відрізнятися від оригінального. Враховуючи це, необхідно буде провести додаткове дослідження за якістю розпізнавання з цим підходом.

Для створення застосунку буде використаний перший метод нормалізації. Це зумовлено тим, що обраний детектор з пункту 2.1.3 вже знаходить ключові точки, за якими робиться нормалізація, та його значна перевага у швидкодії, у порівнянні з генеруванням нового обличчя.

2.3 Вилучення вектору характерних ознак

З нормалізованого обличчя необхідно за допомогою глибокої нейронної мережі вилучити вектор ознак, які будуть достовірно його

описувати для подальшого розпізнавання. Для цього була обрана модель ResNet [56], одна із її архітектур показана на рисунку 2.56. Вона була навчена за допомогою функцій втрат ArcFace [57] (рис. 2.57).

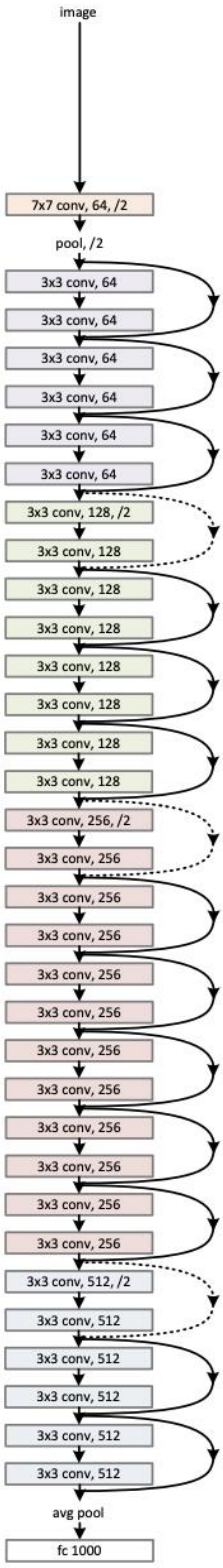


Рисунок 2.56 – Архітектура ResNet34 – residual [56]

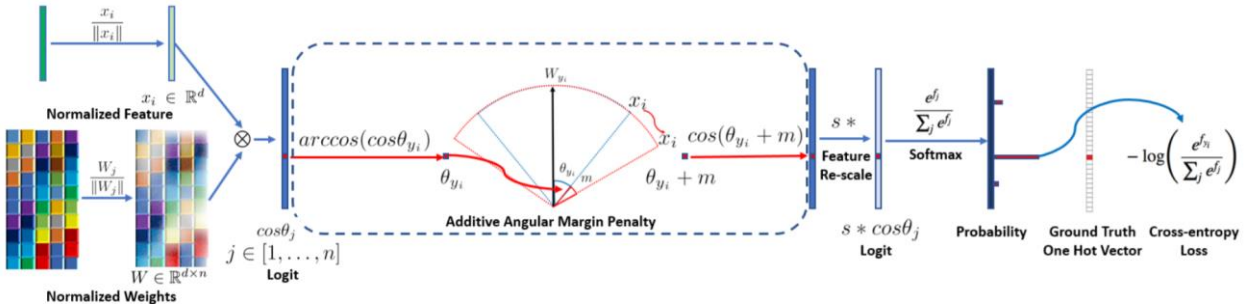


Рисунок 2.57 – Функція втрат ArcFace [57]

Перевагою ArcFace над іншими функціями втрат, наприклад, над популярною SoftMax, э здатність добре розділяти різні класи, але робити дуже схожими елементи всередині одного класу (рис. 2.58).

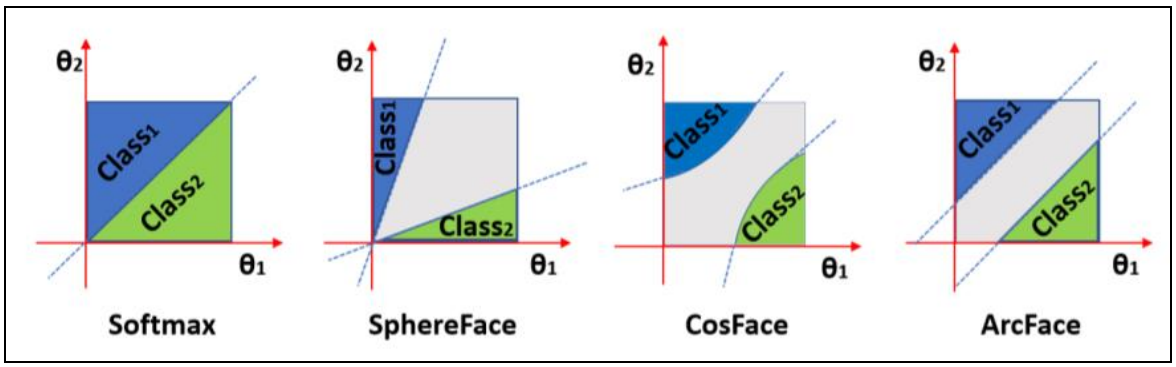


Рисунок 2.58 – Приклади візуалізації поділу класів різними функціями втрат [57]

2.4 Аналіз методів класифікації

2.4.1 Математичні моделі методів класифікації

2.4.1.1 SVM

Support Vector Machines (SVM) [58, 59] – метод навчання з учителем, який можна використовувати, також, і для задачі класифікації. Головною його ідеєю є знайти таку гіперплощину у N-мірному просторі, де N – розмірність даних, що класифікуються , яка чітко розділяє ці дані. Граничні елементи до гіперплощини називаються опорними векторами, w –

перпендикуляр до гіперплощини, що розділяє дані, $\frac{b}{\|w\|}$ – параметр, який дорівнює по модулю відстані від гіперплощини до початку координат (рис. 2.59).

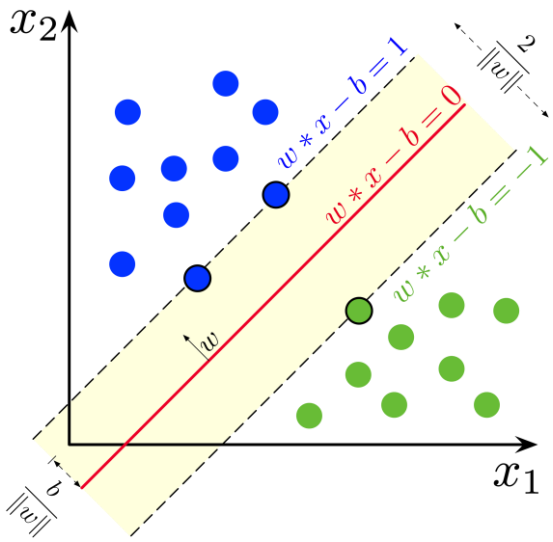


Рисунок 2.59 – Ілюстрація побудови гіперплощини у методі SVM [59]

Для розділення даних можна вибрати багато варіантів побудови гіперплощини, тому ставиться задача знайти таку площину, при якій margin, буде максимальний, тобто виконати задачу мінімізації $\|w\|$ для класів, що лінійно поділювані (2.5)

$$\begin{cases} \|w\|^2 \to \min, \\ c_i(w^* x_i - b) \geq 1, \quad 1 \leq i \leq n, \end{cases} \tag{2.5}$$

де c_i, x_i – компоненти елементу, що має вид $\{(x_1, c_1), ..., (x_n, c_n)\}$.

У свою чергу за допомогою теоремою Куна – Таккера та квадратичного програмування розв'язання має вигляд (2.6)

$$w = \sum_{i=1}^{n} \lambda_i c_i x_i, \quad b = w^* x_i - c_i, \quad \lambda_i > 0. \tag{2.6}$$

Для класів, які неможливо точно лінійно поділити, вводимо додаткові змінні $\xi_i \geq 0$, які характеризують величину помилки для $x_i, 1 \leq i \leq n$. Це назвали soft margin (2.7), або робимо kernel trick – перехід від скалярного добутку до довільних ядер (2.8) [60, 61]

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i, \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i = 1,...,n, \quad \xi_i \geq 0, \tag{2.7}$$

де C – параметр регуляризації.

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(X_i^T, X_j),$$
$$0 \leq \alpha_i \leq C, i = 1,...,n, \ \sum_{i=1}^{n} \alpha_i y_i = 0. \tag{2.8}$$

Це допомагає провести класифікацію таких даних, як на рисунку 2.60.

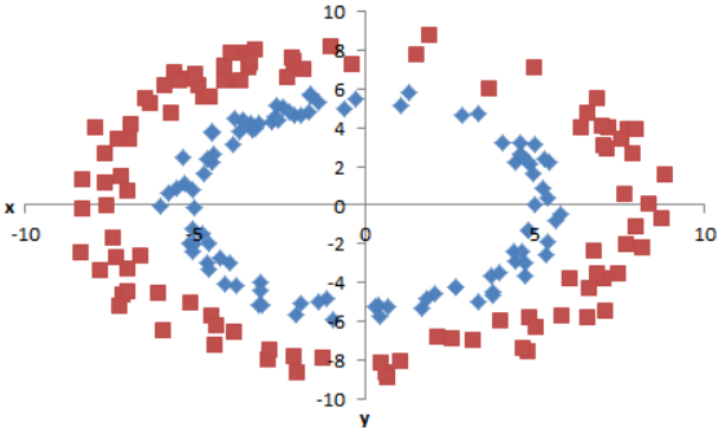


Рисунок 2.60 – Приклад даних, які неможливо лінійно розділити на класи [62]

До переваг методу SVM можна виділити наступні пункти:

– гарно підходить для класів, що лінійно поділяємі;
– потребує мало пам’яті;
– висока швидкодія при класифікації;
– підходить для елементів великої розмірності.

Проте має такі недоліки:

– не підходить для великого або не збалансованого об’єму даних [63];
– чутливий до шуму та гіперпараметрів;
– низька точність при обсязі даних менше, ніж розмірність елементів.

2.4.1.2 FAISS

FAISS [64] – бібліотека від Facebook, яка створювалася з головною метою перекласти пошук за подібністю, кластеризацію на GPU з розпаралелюванням, навіть якщо дані настільки великі, що не вміщаються у оперативну пам’ять під час обробки. Подібність рахується за L2 відстанню, косинусною, або за скалярним добутком. При обробці також можливо використовувати стиснення даних за допомогою квантування векторів, це вливає на точність, але дозволяє обробляти набагато більше даних. Таких результатів FAISS досягає, бо побудована навколо індексу – це KNN-граф, який будується за методом KNN (K-Nearest Neighbor) на етапі навчання, за яким швидко можна проводити пошук схожості вхідних даних та зберігаємих. Внутрішня структура індексу створена таким чином, щоб відповідати компромісам за: швидкістю пошуку, якістю пошуку, використанню пам’яті, часу навчання, кількості даних для навчання. На рисунку 2.61 показані результати швидкості пошуку KNN на датасеті SIFT1M [65].

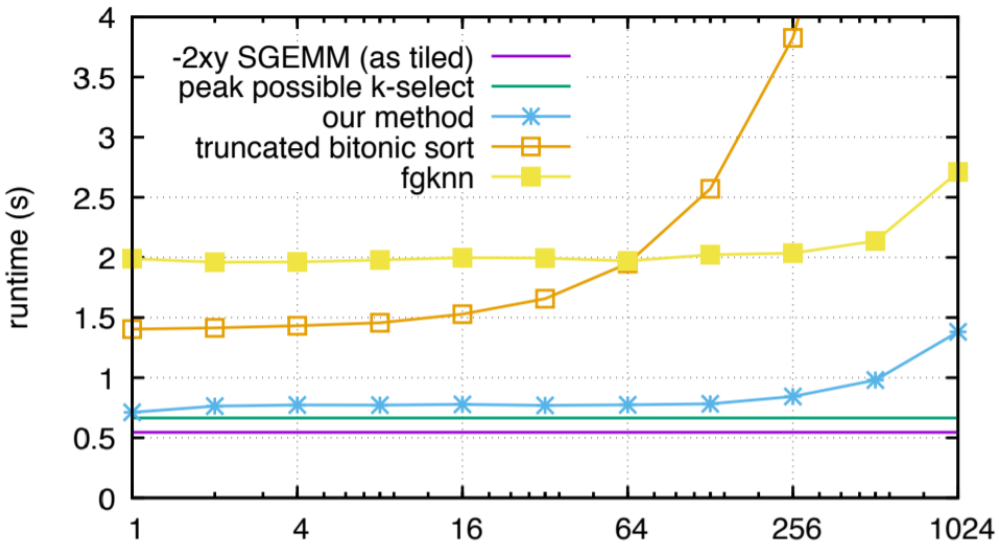


Рисунок 2.61 – Результати швидкодії FAISS на Titan X GPU [64]

2.4.2 Висновки щодо обрання методу класифікації для системи безпеки середнього підприємства

Проаналізувавши переваги та недоліки методів, були зроблені наступні висновки. Доцільно використовувати для невеликої кількості класів SVM, по-перше, через те, що він описується математично і витрачає менше ресурсів і пам’яті, на відміну від FAISS, бо усі вектори зберігаються у індексі, по-друге, метод SVM не використовує пошук класів за найменшою відстанню, яка є вхідним параметром для FAISS і суттєво змінює результати класифікації.

Для FAISS при малій кількості класів використовується повний перебір між даними, на яких був створений індекс, це ускладнює пошук, бо один вектор з шумом може призвести до помилки класифікації, бо функція відстані не володіє властивістю згладжувати викиди. Проте, FAISS, на відміну від SVM, може працювати з великими об’ємами даних і, доцільно, використовувати його, якщо кількість людей у системі безпеки підприємства буде більшою ніж 400 і буде продовжувати зростати. Для великої кількості класів є змога створити індекс, який використовує кластеризацію і буде

повернений той об’єкт, який знайшовся першим у потрібному блоці, проте не обов’язково це буде кращій результат. Така поведінка зумовлена компромісом між швидкодією та якістю класифікації.

2.5 Створення pipeline для розпізнавання облич на основі проведених досліджень для системи безпеки підприємства

Для функціонування системи безпеки підприємства необхідні наступні алгоритми: навчання, розпізнавання. На етапі навчання адміністратор системи створює профілі робітників, за якими буде проводитися розпізнавання. Адміністратор додає фотографії відповідної людини до кожного створеного профілю, на яких повинна бути лише одна людина, лице повинно займати 30% зображення та бути не меншим ніж 150×150 пікселів. Додатково фотографії можна додати з камер, коли за командою адміністратору людина стає перед камерою і виконує його команди. Після наповнення профілів користувачів адміністратор ініціює навчання SVM або FAISS.

Алгоритм розпізнавання складається з наступних кроків:

– детектування обличчя методом CenterFace;

– реєстрація факту детектування;

– перевірка впевненості детектування;

– перевірка, що лице справжнє;

– розрахунок кута обличчя;

– нормалізація обличчя (вирівнювання та вилучення необхідного регіону);

– розрахунок характерних ознак обличчя та переведення його у векторний опис за допомогою ResNet;

– класифікація за допомогою SVM.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ БЕЗПЕКИ ПІДПРИЄМСТВА

3.1 Вибір технологій

Для реалізації необхідно створити саму систему розпізнавання, систему для навчання, інтерфейс для взаємодії з користувачем. Для системи розпізнавання та навчання були обрані такі технології: мова програмування C++, бо нам дуже важлива швидкодія, бібліотека OpenCV [66], яка є відкритою і підтримує змогу працювати з моделями глибокого навчання, які були створені на за допомогою інших бібліотек. Для створення інтерфейсу користувача була обрана мова програмування JavaScript та бібліотека React [67]. Переваги її у тому, що вона швидка, легко розширюється та має велику за обсягом документацію та багато прикладів у вирішенні тієї чи іншою задачі.

3.2 Розбиття на мікросервиси

Розбиття системи на підсистеми дає змогу використовувати точково необхідні технології там, де вони необхідні, наприклад, основну та критичну, за часом виконання, логіку написати на C++ та взаємодіяти з базою на SQL специфікації ANSI, а іншу частину реалізувати, наприклад, за допомогою ORM (Object Relational Mapping). Це буде повільніше, але набагато простіше для розробника, та ще робить підтримку системи більш простішою.

Також ще однією перевагою для підсистем є змога балансувати навантаження між ними на різних шарах моделі OSI. Якщо навантаження на якийсь компонент різко зростає, наприклад прийшов натовп людей і система розпізнавання не витримує, можна паралельно запустити ще один модуль розпізнавання та відправляти частину даних на нього. Таке балансування

можна робити, наприклад, через Kubernetes [68] – продукт, з відритим кодом, який розробляється компанією Google та робить зручним роботу з контейнерізованими застосунками. Контейнерізація – це механізм, за допомогою якого ядро операційної системи працює з декількома просторами користувача та вони ізольовані один від одного, тобто є можливість створити контейнер, який можна запустити і бути впевненим, що у нього є все необхідне для роботи і немає зайвого, що може спричинити збої у роботі. Наприклад, зараз популярним є Docker [69]. За його допомогою зручно створювати контейнери, які можна передавати іншим, запускати застосунки вже з налаштованими середовищами та зручно оновлювати їх версії, без остраху конфліктів залежностей.

Для створення системи були виділені наступні мікросервіси:

– система розпізнавання, яка детектує обличчя, записує дані у базу, вилучає ознаки обличчя, нормалізує його та класифікує;

– система навчання, яка навчає модель класифікації;

– front-end – інтерфейс користувача;

– back-end – прошарок між системою розпізнавання, системою навчання, базою та інтерфейсом користувача.

3.3 Створення датасету для аналізу якості роботи системи

Для аналізу якості роботи системи була створена функція збору даних для тестування за допомогою створеної системи. Вона дозволяє додавати у датасет дані з камер або окремо, які адміністратор вибрав і встановив належність їх до відповідного класу. Таку функцію винесено у інтерфейс користувача, що дає змогу легко та швидко додавати або видаляти дані та тестувати систему. Датасет містить більше 100 класів та близько 17000 фотографій. На кожний клас, у середньому, зібрано 150 прикладів зображень. Також система дозволяє вибірково виключати дані з тестування, що також

додає більшу зручність при тестування, коли, наприклад, необхідно підрахувати точність розпізнавання лише одного класу. Приклади даних представлені на рисунку 3.1. Через наявність у системі конфіденційних даних, на прикладах вони були приховані, а фотографії розмиті.

FACE	FULL NAME	DETECTOR	CREATE DATE	UPDATE DATE	ACTIVE	ANGLE
	User is hidden	CenterFace	09.07.2021 17:43:17	13.10.2021 12:24:31	✔	2.673172
	User is hidden	CenterFace	09.07.2021 17:43:17	12.07.2021 15:37:26	✔	0.17647934
	User is hidden	CenterFace	09.07.2021 17:43:17	12.07.2021 15:37:26	✔	1.5610237
	User is hidden	CenterFace	09.07.2021 17:43:17	12.07.2021 15:37:26	✔	4.0250244
	User is hidden	CenterFace	09.07.2021 17:43:17	12.07.2021 15:37:26	✔	6.677952
	User is hidden	CenterFace	09.07.2021 17:43:17	12.07.2021 15:37:26	✔	4.167568

Рисунок 3.1 – Приклад показу даних з датасету у UI

3.4 Ілюстрація роботи системи

На головній сторінці системи відображається активні камери, та відеотрансляція з них. Додатково є функція, якщо тип відеопотоку вказаний як recognizer, яка відображає на відеопотоці рамки облич та підписує їх після розпізнавання. З правої сторони у реальному часі оновлюється події, які були зафіксовані сьогодні системою з усіх камер (рис. 3.2). За бажанням останні події можна відсортувати за результатом – розпізнані або ні, перейти на список усіх подій.

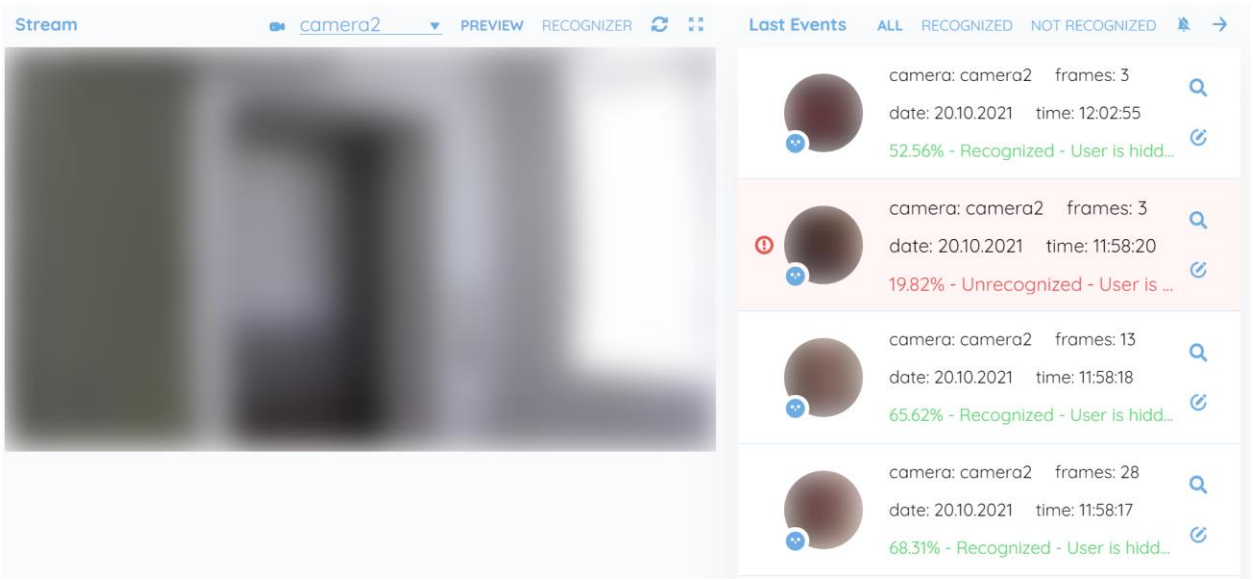


Рисунок 3.2 – Головна сторінка системи

Увесь список подій відображається на окремій сторінкі (рис. 3.3), а якщо необхідно більш детально подивитися подію, то для цього зроблений окремий розділ, де можна подивитися відео, яке було зафіксовано системою, та список людей, які були розпізнані (рис. 3.4).

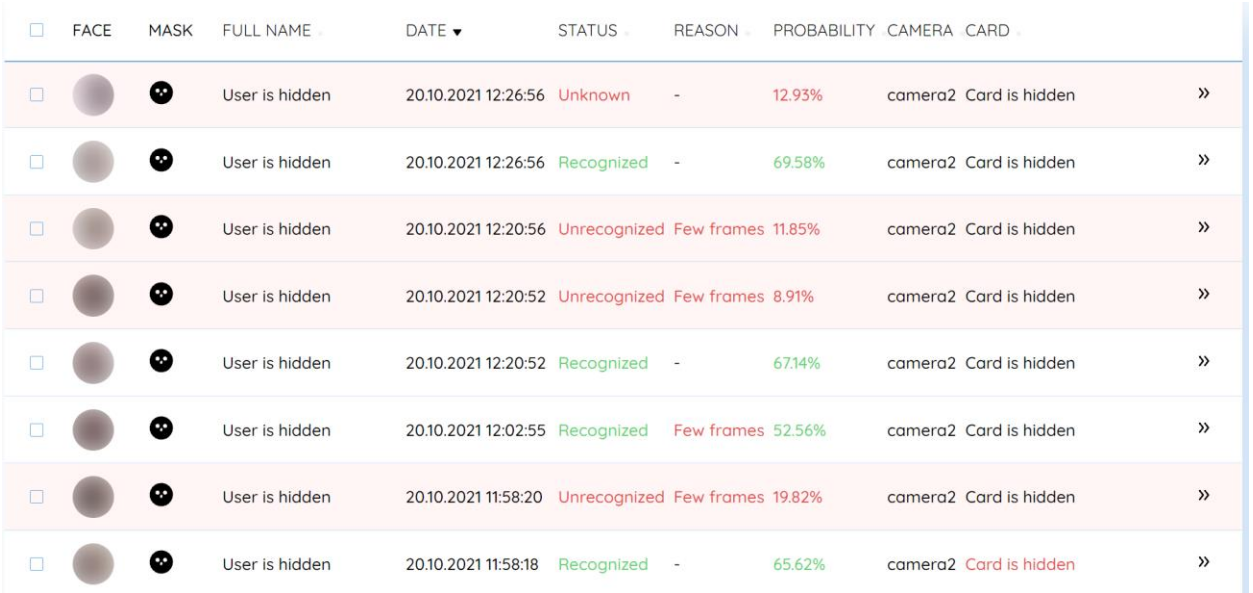


Рисунок 3.3 – Сторінка з усіма подіями у UI

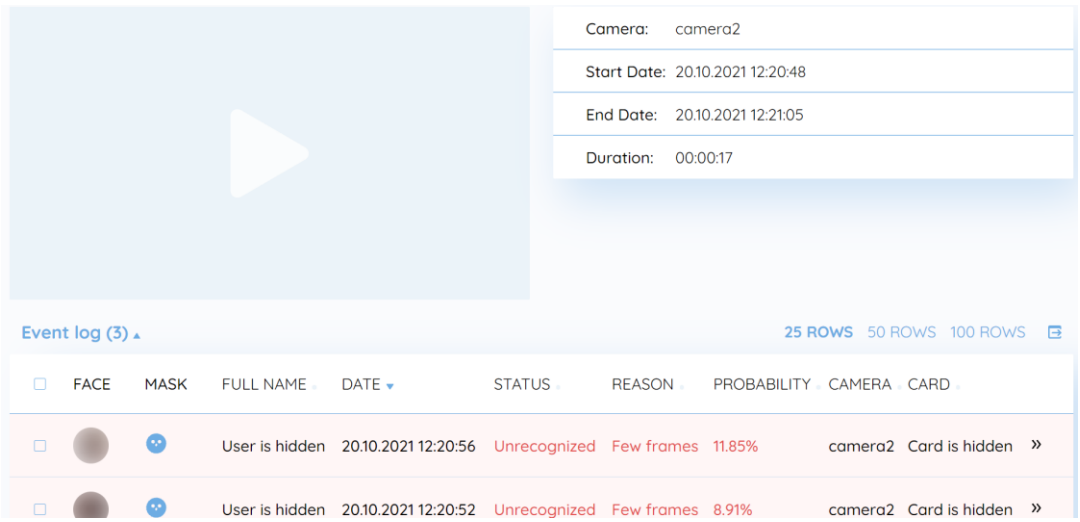


Рисунок 3.4 – Приклад сторінки з детальною інформацією зафіксованої системою події у UI

3.5 Аналіз якості розпізнавання розробленою системою

Для системи була передбачена та розроблена можливість тестування якості її роботи. Для цього на даних, які зібрані у пункті 3.3, запускався модуль класифікації який помічав результат помилкою, якщо профіль не співпав, та видав попередження, якщо ймовірність нижче заданого порігу. Результати записуються у базу та відображаються у інтерфейсі користувача, як показано на рисунку 3.5.

RUN ID	START DATE	END DATE	DURATION	ERRORS	WARNINGS	TOTAL	PASSED	RUN STATUS
3167	17.08.2021 13:04:12	17.08.2021 13:07:57	00:03:45	79	1127	17464	99.55%	Fail
2798	20.07.2021 11:05:18	20.07.2021 11:08:27	00:03:09	51	264	17464	99.71%	Fail
2764	13.07.2021 12:17:32	13.07.2021 12:20:43	00:03:10	63	833	17464	99.64%	Fail
2759	12.07.2021 17:19:09	12.07.2021 17:22:38	00:03:28	59	556	17464	99.66%	Fail
2758	12.07.2021 16:23:40	12.07.2021 16:27:07	00:03:26	59	2350	17464	99.66%	Fail
2755	12.07.2021 14:36:40	12.07.2021 14:40:37	00:03:56	61	2123	17464	99.65%	Fail
2745	09.07.2021 17:58:24	09.07.2021 18:01:55	00:03:31	61	2119	17464	99.65%	Fail

Рисунок 3.5 – Приклад результатів тестування системи у UI

Результати, які не пройшли класифікацію можна подивитися окремо, та побачити причину, чому вони були відкинуті (рис. 3.6). Для підрахування якості роботи усієї системи результати тестування були оброблені за допомогою AUC ROC для багатокласового випадку. Результат тестування становить 0,9999979211204167.

FACE	TEST CASE PROFILE	TEST CASE PROBABILITY	RECOGNIZED PROFILE	PROBABILITY	ERROR	ACTIVE
	User is hidden	55.00%	User is hidden	20.59%	probability	✓
	User is hidden	55.00%	User is hidden	20.07%	probability	✓
	User is hidden	55.00%	User is hidden	25.69%	probability	✓
	User is hidden	55.00%	User is hidden	34.38%	probability	✓
	User is hidden	55.00%	User is hidden	31.10%	probability	✓
	User is hidden	55.00%	User is hidden	26.07%	probability	✓
	User is hidden	55.00%	User is hidden	21.02%	probability	✓

Рисунок 3.6 – Приклад результатів, які не були класифіковані

3.6 Пропозиція щодо використання методу DBSCAN для покращення розпізнавання при зменшенні кількості вірних розпізнавань під час функціонування системи

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [70] є базовим алгоритмом для кластеризації на основі щільності. Він може виявити кластери різних форм і розмірів з великої кількості даних, що містять шум і викиди. Цей алгоритм не вимагає зазначення кількості кластерів вхідним параметром, як, наприклад, *k*-means, DBSCAN визначає кількість кластерів на основі даних і може виявити кластери довільної форми.

Основна ідея DBSCAN полягає в тому, що точка належить до кластеру, якщо вона близька до багатьох точок з цього кластеру.

Існує два ключові параметри DBSCAN:

– *eps* – відстань, що визначає сусідство. Дві точки вважаються сусідніми, якщо відстань між ними менше або дорівнює eps;

– *minPts* – мінімальна кількість точок даних для визначення кластера.

На основі цих двох параметрів точки класифікуються як основні точки, прикордонні точки або викиди. Основна точка – точка є основною, якщо в її околицях з радіусом eps є не менше minPts кількості точок, включаючи саму точку. Прикордонна точка – точка є прикордонною, якщо до неї можна дістатися з основної точки і в її околицях знаходиться менш minPts кількості точок. Викид – точка є викидом, якщо вона не є основною точкою і не досяжна ні з однією основною точки.

На основі обраного методу був створений окремий модулю, який використовується для кластеризації результатів класифікації, які не пройшли класифікацію або ймовірність класифікації менша 50%. Враховуючи, що мережа, яка вилучає embeddings, навчена за допомогою ArcFace, то embeddings всередині одного класу достатньо близькі, ніж для інших класів. Метод виконується один раз на день або після запуску адміністратором, коли система менше усього навантажена, та генерує підказки для адміністратора, щодо перенавчання існуючої моделі класифікації. Наприклад, якщо з’явилася нова людина, яка ще не додана у систему, то метод збирає усі події, де ця людина була помічена, та пропонує додати її до системи.

На датасеті CelebA, з якого були отримані embeddings для кожного зображення, метод видає наступні результати за метриками: однорідності, повноти, *V*-міри, ARI, silhouette coefficient, AMI. Результати показані на рисунку 3.7.

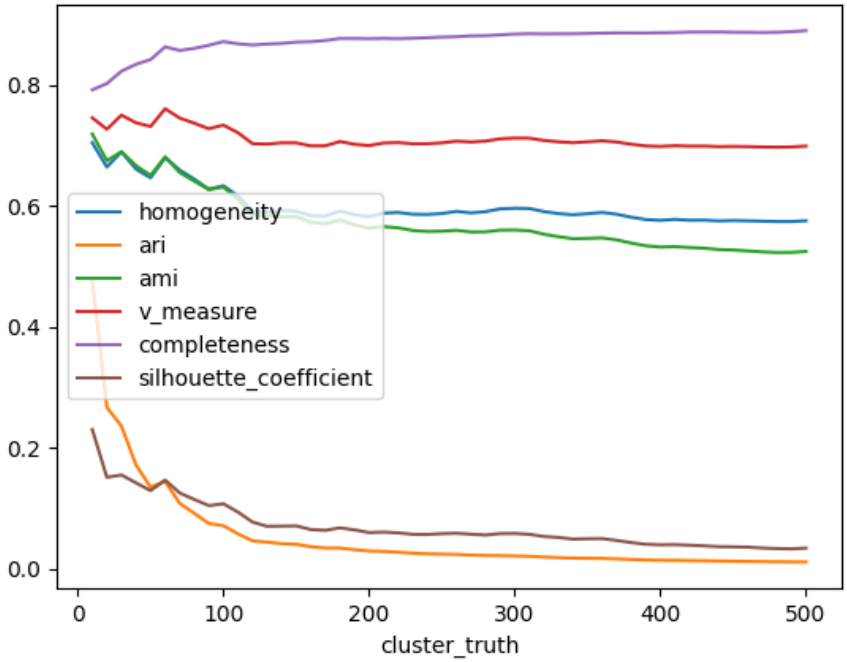


Рисунок 3.7 – Графік значень метрик методу DBSCAN на датасеті CelebA для 500 кластерів

На рисунку 3.8 показаний приклад, класу, який метод виділив з даних при тестуванні.

Рисунок 3.8 – Приклад виділеного класу методом DBSCAN

ВИСНОВКИ

У рамках кваліфікаційної роботи з розробки та дослідження методу розпізнавання облич для систем безпеки на базі результатів проведеного аналізу сучасних методів детектування та розпізнавання об’єктів було виконане наступне:

– проаналізовано сучасний стан розвитку питання детектування облич;

– проаналізовано існуючі підходи для детектування;

– створено датасет для аналізу якості існуючих підходів детектування облич при зміні умов положення та масштабу;

– оцінено якість існуючих методів детектування;

– оцінено швидкість існуючих методів детектування;

– обрано метод, який найкраще підходить до поставленого завдання;

– вивчено стан та розвиток методів нормалізації облич;

– проаналізовано сучасний стан розвитку задачі розпізнавання та методи її вирішення;

– проведено аналіз методів класифікації облич;

– зібрано дані для створення датасету для оцінки якості розпізнавання;

– створено pipeline усієї системи;

– спроєктовано та реалізоввано програмну реалізацію;

– обрано метод для покращення роботи системи при зменшенні кількості вірних розпізнавань під час її функціонування.

Для обрання кращого детектору за швидкістю роботи на зображеннях різного розміру, за впевненістю на малих та повернутих обличчях були протестовані методи детектування облич: MTCNN, FaceBoxes, DSFD, RetinaFace, CenterFace, SCRFD. Окремо для задачі класифікації були проаналізовані методи: SVM та FAISS. За результатами досліджень були

обрані методи та на їх основі була розроблена система безпеки підприємства, яка працює у режимі реального часу з багатьма камерами одночасно.

Дослідження показали, що одні детектори: FaceBoxes, SCRFD – мають кращу швидкодію ніж інші, проте, програють у якості детектування за значенням AP детекторам: DSFD, RetinaFace, CenterFace. Зважаючи на те, що DSFD та RetinaFace не можуть працювати з декількома камерами у 10 FPS, як ставилось у технічному завданні, бо їх час детектування більший ніж 100ms, був обраний детектор CenterFace 640×480 px. Додатковою його перевагою над FaceBoxes и SCRFD є те, що він вертає ключові точки обличчя, за якими проводиться нормалізація кута повороту.

Задача класифікації була вирішена за допомогою SVM та FAISS, які створені для різних цілей. SVM був обраний для систем з невеликою кількостю профілей для розпізнавання, а FAISS навпаки, великої кількості. Аналіз якості системи у цілому за допомогою ROC AUC дав результат 0,9999979211204167. ROC крива вже при нульовому порозі показує результат, який задовольняє вимогам до системи, що свідчить про те, що класи достатньо розрізняються, після вилучення характеристик обличчя мережею, яка навчена за допомогою функції ArcFace.

Для створення застосунку була використані мови програмування C++ та JavaScript разом з бібліотеками OpenCV та React. З їх допомогою були створені частини системи, які можуть працювати з багатьма камерами одночастно, витримувати великі навантаження та роблять зручним управління системою в цілому для адміністратора.

Результати досліджень були апробовані у вигляді 4 тез доповідей на 25-ому Міжнародному молодіжному форуму «Радіоелектроніка і молодь в XXI столітті» [5], XII-й Міжнародній науково-практичній конференції «Free and Open Source Software» [14], VII міжнародної науково-технічної конференції НТУ «ХПІ» «Інформатика, управління та штучний інтелект» [15] та на VIII міжнародної науково-технічної конференції НТУ «ХПІ» «Інформатика, управління та штучний інтелект» [71].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Yakovleva, O., & Nikolaieva, K. (2020). RESEARCH OF DESCRIPTOR BASED IMAGE NORMALIZATION AND COMPARATIVE ANALYSIS OF SURF, SIFT, BRISK, ORB, KAZE, AKAZE DESCRIPTORS. Сучасні інформаційні системи, 4(4), 89-101.

2. Kovtunenko, A., Yakovleva, O., Liubchenko, V., & Yanholenko, O. (2020). ДОСЛІДЖЕННЯ СУМІСНОГО ВИКОРИСТАННЯ МАТЕМАТИЧНОЇ МОРФОЛОГІЇ ТА ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ РОЗПІЗНАВАННЯ ЦІННИКІВ. Вісник Національного технічного університету «ХПІ». Серія: Системний аналіз, управління та інформаційні технології, (1 (3)), 24-31.

3. Viola, P., & Jones, M. (2001, December). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (Vol. 1, pp. I-I). Ieee.

4. Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (Vol. 1, pp. 886-893). Ieee.

5. Ковтуненко А.Р. (2021). Порівняння класичних методів детектуваня облич на зображенні та нейромережевих підходів за якістю та швидкодією. Матеріали XXV Міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті».

6. Li, H., Lin, Z., Shen, X., Brandt, J., & Hua, G. (2015). A convolutional neural network cascade for face detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5325-5334).

7. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

8. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, *28*, 91-99.

9. Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).

10. Minaee, S., Luo, P., Lin, Z., & Bowyer, K. (2021). Going Deeper Into Face Detection: A Survey. *arXiv preprint arXiv:2103.14983*.

11. Liu, C., & Wechsler, H. (2002). Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *IEEE Transactions on Image processing*, *11*(4), 467-476.

12. Ahonen, T., Hadid, A., & Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence*, *28*(12), 2037-2041.

13. Wang, M., & Deng, W. (2018). Deep face recognition: A survey. *arXiv preprint arXiv:1804.06655*.

14. Ковтуненко А.Р. (2020). Огляд бібліотек для вирішення задачі розпізнавання облич. Матеріали XII-ої Міжнародної науково-практичної конференції «Free and Open Source Software».

15. Ковтуненко А.Р. (2020). Дослідження підходу до розпізнання облич на основі глибокого навчання. "Інформатика, управління та штучний інтелект".

16. Vinyals, O., Blundell, C., Lillicrap, T., & Wierstra, D. (2016). Matching networks for one shot learning. *Advances in neural information processing systems*, *29*, 3630-3638.

17. Fei-Fei, L., Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, *28*(4), 594-611.

18. Koch, G., Zemel, R., & Salakhutdinov, R. (2015, July). Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop* (Vol. 2).

19. Padilla, R., Netto, S. L., & da Silva, E. A. (2020, July). A survey on performance metrics for object-detection algorithms. In *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)* (pp. 237-242). IEEE.

20. Yang, S., Luo, P., Loy, C. C., & Tang, X. (2016). Wider face: A face detection benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5525-5533).

21. Jain, V., & Learned-Miller, E. (2010). *Fddb: A benchmark for face detection in unconstrained settings* (Vol. 2, No. 4, p. 5). UMass Amherst technical report.

22. Zhu, X., & Ramanan, D. (2012, June). Face detection, pose estimation, and landmark localization in the wild. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 2879-2886). IEEE.

23. Papers with code - wider face (hard) benchmark (face detection). The latest in Machine Learning. URL: https://paperswithcode.com/sota/face-detection-on-wider-face-hard (дата звернення: 04.10.2021).

24. Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2018, May). Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)* (pp. 67-74). IEEE.

25. Liu, Z., Luo, P., Wang, X., & Tang, X. (2018). Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, *15*(2018), 11.

26. Whitelam, C., Taborsky, E., Blanton, A., Maze, B., Adams, J., Miller, T., ... & Grother, P. (2017). Iarpa janus benchmark-b face dataset. In *proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 90-98).

27. Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008, October). Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*.

28. Xiang, J., & Zhu, G. (2017, July). Joint face detection and facial expression recognition with MTCNN. In 2017 4th international conference on information science and control engineering (ICISCE) (pp. 424-427). IEEE.

29. Gradilla, R. (2020, July 27). Multi-task Cascaded Convolutional Networks (MTCNN) for Face Detection and Facial Landmark Alignment. Medium. URL: https://medium.com/@iselagradilla94/multi-task-cascaded-convolutional-networks-mtcnn-for-face-detection-and-facial-landmark-alignment-7c21e8007923 (дата звернення: 10.09.2021).

30. Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., & Li, S. Z. (2017, October). Faceboxes: A CPU real-time face detector with high accuracy. In 2017 IEEE International Joint Conference on Biometrics (IJCB) (pp. 1-9). IEEE.

31. Li, J., Wang, Y., Wang, C., Tai, Y., Qian, J., Yang, J., ... & Huang, F. (2019). DSFD: dual shot face detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5060-5069).

32. Ponchon, B. (2020, December 11). Face detectors: Understand DSFD and the state-of-the-art algorithms. Sicara. URL: https://www.sicara.ai/blog/2019-09-26-face-detectors-dsfd-state-of-the-art-algorithms (дата звернення: 12.09.2021).

33. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.

34. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

35. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

36. Yu, F., & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122.

37. Tsang, S.-H. (2019, March 20). Review: DilatedNet - DILATED CONVOLUTION (Semantic Segmentation). Medium. URL: https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5 (дата звернення: 12.09.2021).

38. Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., & Zafeiriou, S. (2019). Retinaface: Single-stage dense face localisation in the wild. *arXiv preprint arXiv:1905.00641*.

39. Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).

40. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 764-773).

41. Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).

42. Xu, Y., Yan, W., Yang, G., Luo, J., Li, T., & He, J. (2020). CenterFace: joint face detection and alignment using face as point. *Scientific Programming*, *2020*.

43. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).

44. Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988).

45. Guo, J., Deng, J., Lattas, A., & Zafeiriou, S. (2021). Sample and Computation Redistribution for Efficient Face Detection. *arXiv preprint arXiv:2105.04714*.

46. Zhu, Y., Cai, H., Zhang, S., Wang, C., & Xiong, Y. (2020). Tinaface: Strong but simple baseline for face detection. *arXiv preprint arXiv:2011.13183*.

47. Chen, Y., Yang, T., Zhang, X., Meng, G., Xiao, X., & Sun, J. (2019). Detnas: Backbone search for object detection. *Advances in Neural Information Processing Systems*, *32*, 6642-6652.

48. Liang, F., Lin, C., Guo, R., Sun, M., Wu, W., Yan, J., & Ouyang, W. (2019). Computation reallocation for object detection. *arXiv preprint arXiv:1912.11234*.

49. Ghiasi, G., Lin, T. Y., & Le, Q. V. (2019). Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7036-7045).

50. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

51. Unique, worry-free model photos. Generated Photos. URL: https://generated.photos/ (дата звернення: 26.09.2021).

52. Kaziakhmedov, E., Kireev, K., Melnikov, G., Pautov, M., & Petiushko, A. (2019, October). Real-world attack on MTCNN face detection system. In *2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)* (pp. 0422-0427). IEEE.

53. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

54. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, *27*.

55. Leimkühler, T., & Drettakis, G. (2021). FreeStyleGAN: Free-view Editable Portrait Rendering with the Camera Manifold. arXiv preprint arXiv:2109.09378.

56. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

57. Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4690-4699).

58. Gunn, S. R. (1998). Support vector machines for classification and regression. *ISIS technical report*, *14*(1), 5-16.

59. Wikimedia Foundation. (2021, September 30). Support-Vector Machine. Wikipedia. URL: https://en.wikipedia.org/wiki/Support-vector_machine (дата звернення: 05.10.2021).

60. Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144-152).

61. Hofmann, M. (2006). Support vector machines-kernels and the kernel trick. *Notes*, *26*(3), 1-16.

62. Yadav, A. (2018, October 22). Support Vector Machines(SVM). Medium. URL: https://towardsdatascience.com/support-vector-machines-svm-c9ef22815589 (дата звернення: 05.10.2021).

63. Batuwita, R., & Palade, V. (2013). Class imbalance learning methods for support vector machines. *Imbalanced learning: Foundations, algorithms, and applications*, 83-99.

64. Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

65. Jegou, H., Douze, M., & Schmid, C. (2010). Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, *33*(1), 117-128.

66. Home. OpenCV. (2021, June 16). URL: https://opencv.org/ (дата звернення: 07.10.2021).

67. React – a JavaScript library for building user interfaces. – A JavaScript library for building user interfaces. URL: https://reactjs.org/ (дата звернення: 07.10.2021).

68. Production-grade container orchestration. Kubernetes. URL: https://kubernetes.io/ (дата звернення: 07.10.2021).

69. Empowering app development for developers. Docker. URL: https://www.docker.com/ (дата звернення: 07.10.2021).

70. Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd* (Vol. 96, No. 34, pp. 226-231).

71. Ковтуненко А.Р. (2021). Створення системи безпеки на основі аналізу сучасних методів детектування облич та класифікації. "Інформатика, управління та штучний інтелект".