Распознавание пола и возраста по видеоизображению лица на основе сверточных нейронных сетей

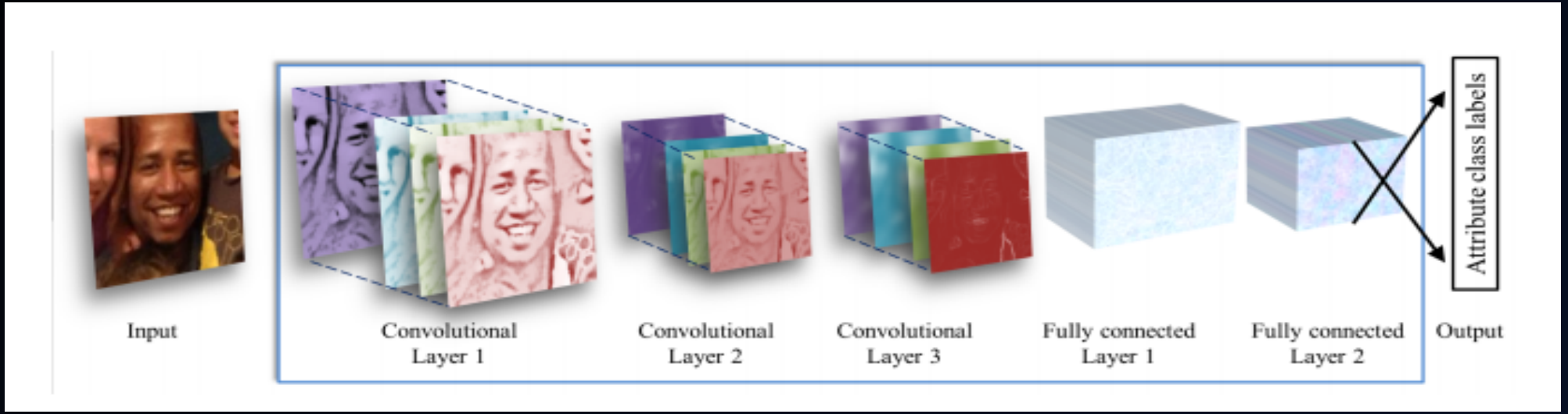
РАБОТУ ВЫПОЛНИЛА:
ХАРЧЕВНИКОВА АНГЕЛИНА
СТУДЕНТ ГРУППЫ 14БИ-1

- В данной работе задача распознавания пола и возраста на видеоизображении рассматривается как задача выбора наиболее надежного решения из нескольких на основе синтеза коллективов решающих правил (КРП, комитетов классификаторов)

Обзор Литературы

- **Levi, G.** Age and gender classification using convolutional neural networks / Levi G., Hassner T.

Строение модели



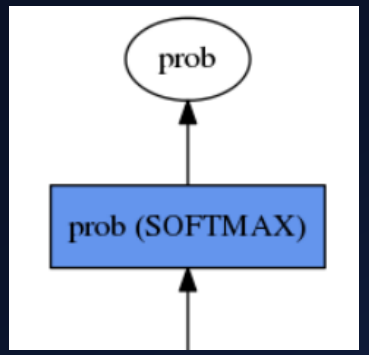
Входной объект

Вероятности

Выход нейросетевой модели обычно получается в слое Softmax, который выдает вероятности принадлежности объекта к каждому классу

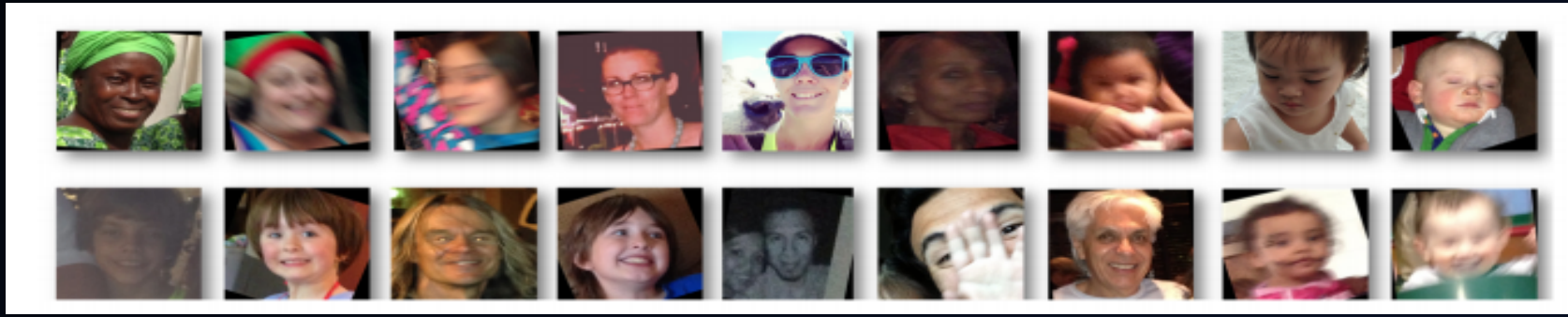
$$P(X(t)|l) = \text{softmax } z_l(t) = \frac{\exp(z_l(t))}{\sum_{j=1}^{L} \exp(z_j(t))}, l = 1,2,\ldots,L,$$

$$l^*(t) = \underset{l=1,2,\ldots,L}{\text{argmax}} P(X(t)|l).$$



Результаты тестирования моделей

- Тестирование проводилось с помощью dataset Audience collection (Audience benchmark)
- Коллекция содержит 26000 нефильтрованных изображений, полученных с камер смартфонов



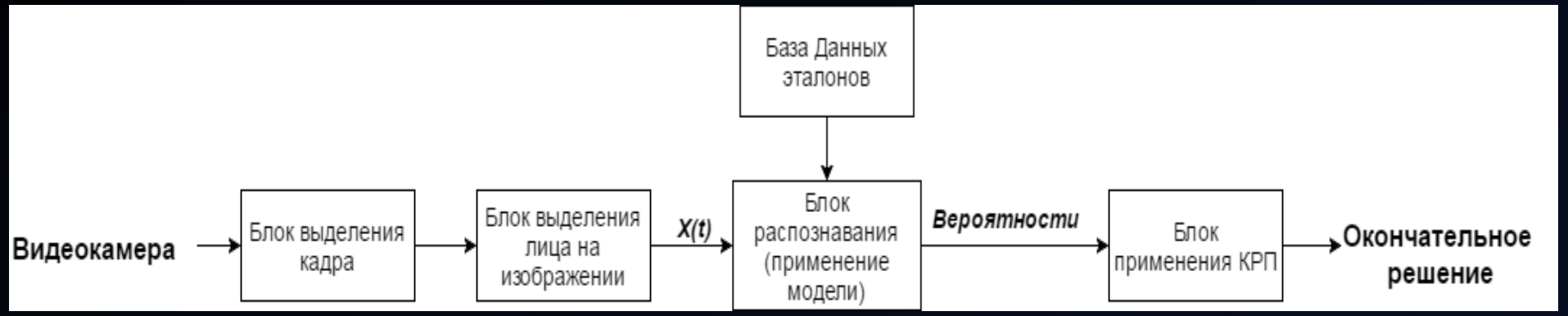
	0-2	4-6	8-13	15-20	25-32	38-43	48-53	60-	Total
Male	745	928	934	734	2308	1294	392	442	8192
Female	682	1234	1360	919	2589	1056	433	427	9411
Both	1427	2162	2294	1653	4897	2350	825	869	19487

Table 1. **The AdienceFaces benchmark.** Breakdown of the AdienceFaces benchmark into the different Age and Gender classes.

- Точность: распознавание пола – 87%

  распознавание возраста – 85%

Предлагаемый алгоритм



- Задача распознавания видеоизображения состоит в том, чтобы отнести вновь поступающую (на вход) последовательность кадров $\{X(t)\},\ X(t) = \|x_{uv}(t)\|,\ t = \overline{1,T}$ с изображением одного объекта к одному из *L* классов
- Распознавание пола: L = 2
- Распознавание возраста: L = 8

Применение КРП

- Простое голосование

$$l^* = \arg\max_{l=\overline{1,L}} \sum_{t=1}^{T} \delta(l^*(t) - l).$$

- Метод усреднения вероятностей

1. $$l^* = \operatorname{argmax}_{l=\overline{1,L}} \frac{1}{T} \sum_{t=1}^{T} P\left(X(t)\middle|l\right)$$

2. $$l^* = \operatorname{argmax}_{l=\overline{1,L}} \prod_{t=1}^{T} P\left(X(t)\middle|l\right) = \operatorname{argmax}_{l=\overline{1,L}} \sum_{t=1}^{T} \log P\left(X(t)\middle|l\right)$$

Реализация

- ПК Intel Core i5-3210M CPU, 64-разрядная операционная система Windows 7
- Opencv
- Opencv DNN module
- Gender_net caffemodel
- Age_net caffemodel

Детектирование лица

CourseWork_opencvBuild | (Global Scope) | detect(cv::Mat frame)

```
183     /** @function detectAndDisplay */
184     cv::Mat detect(cv::Mat frame)
185     {
186         std::vector<cv::Rect> faces;
187         cv::Mat frame_gray;
188         cv::Mat face;
189         cvtColor(frame, frame_gray, cv::COLOR_BGR2GRAY);
190         equalizeHist(frame_gray, frame_gray);
191         //-- Detect faces
192         face_cascade.detectMultiScale(frame_gray, faces, 1.1, 2, 0 | cv::CASCADE_SCALE_IMAGE, cv::Size(30, 30));
193         if (faces.size() == 0)
194         {
195             cout << "no face!" << endl;
196         }
197         else
198         {
199             cv::Rect face_i = faces[0];
200             face = frame(face_i);
201             rectangle(frame, face_i, CV_RGB(0, 255, 0), 1);
202             // Create the text we will annotate the box with:
203             string box_text_gender = "Gender= " + prediction_gender + "Age= " + prediction_age;
204             int pos_x = std::max(face_i.tl().x - 10, 0);
205             int pos_y = std::max(face_i.tl().y - 10, 0);
206             // And now put it into the image:
207             putText(frame, box_text_gender, cv::Point(pos_x, pos_y), cv::FONT_HERSHEY_PLAIN, 1.0, CV_RGB(0, 255, 0), 2.0);
208
209         }
210
211         display(frame);
212         return face;
213     }
```

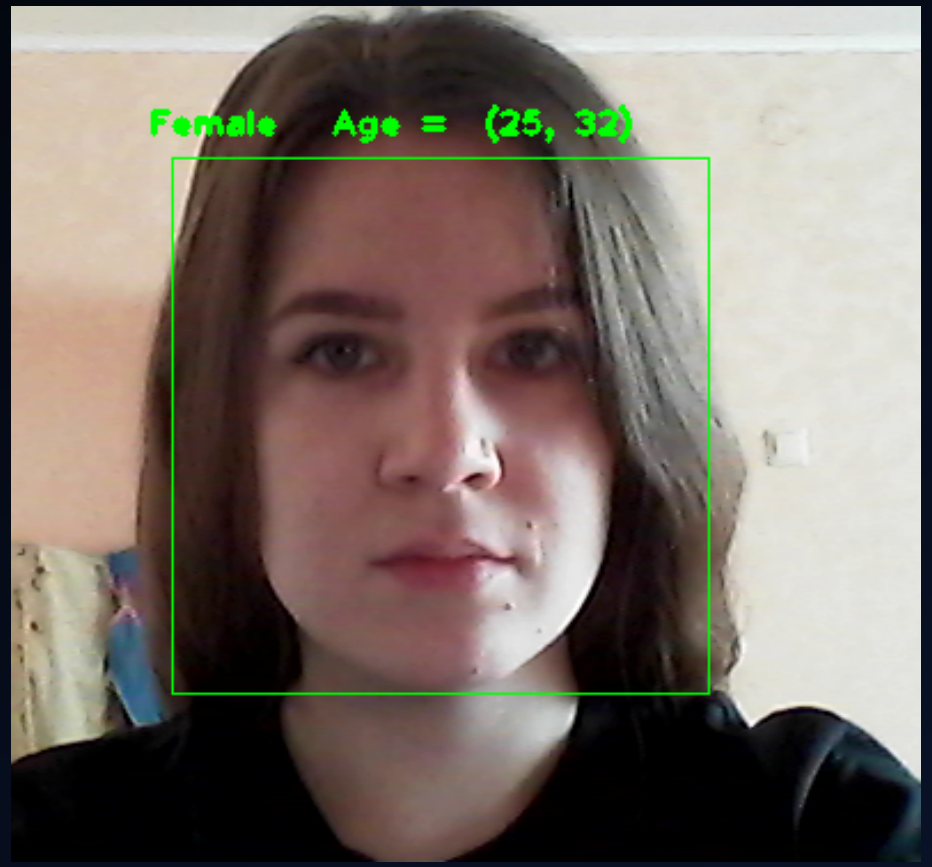
Распознавание пола и возраста

```
while (capture.read(frame))
{
    if (frame.empty())
    {
        printf(" --(!) No captured frame -- Break!");
        break;
    }
    //-- 3. Apply the classifier to the frame
    cv::Mat face = detect(frame);

    if (face.size() != cv::Size(0,0))
    {
        std::vector<float> probabs_gender(2);
        extractProbabs(face, &(probabs_gender[0]), net_gender);
        probabsAggregated_gender.push_back(probabs_gender);
        //makePrediction(gender_list, probabs_gender, "gender");

        std::vector<float> probabs_age(8);
        extractProbabs(face, &(probabs_age[0]), net_age, 8);
        probabsAggregated_age.push_back(probabs_age);
        //makePrediction(age_list, probabs_age, "age");

        countFaces++;
        if (countFaces == 10)
        {
            //prediction_gender = makePredictionSimpleVote(probabsAggregated_gender, gender_list);
            //prediction_age = makePredictionSimpleVote(probabsAggregated_age, age_list, 8);
            prediction_gender = makePredictionAverage(probabsAggregated_gender, gender_list);
            prediction_age = makePredictionAverage(probabsAggregated_age, age_list, 8);

            cout <<"gender: " << prediction_gender << endl;
            cout <<"age: " << prediction_age << endl;
            countFaces = 0;
```

Получение вероятностей на основе модели

```
void init() {
    cv::dnn::initModule();
    net_gender = cv::dnn::readNetFromCaffe(PROTO_FILE_GENDER, PROTO_MODEL_GENDER);
    net_age = cv::dnn::readNetFromCaffe(PROTO_FILE_AGE, PROTO_MODEL_AGE);
}

void extractProbabs(cv::Mat& face_img, float* probabs, Net net_, int probabs_len = 2) {
    cv::Mat img_resized;
    cv::resize(face_img, img_resized, cv::Size(227, 227));
    cv::Mat img;
    img_resized.convertTo(img, CV_32FC1);
    cv::dnn::Blob inputBlob = cv::dnn::Blob::fromImages(img);   //Convert Mat to dnn::Blob batch of images
    net_.setBlob(".data", inputBlob);
    net_.forward();
    cv::dnn::Blob featureBlob = net_.getBlob(FEATURE_LAYER);
    cv::Mat featureMat = featureBlob.matRefConst().reshape(1, 1);
    int feat_len = std::min(featureMat.cols, probabs_len);
    for (int i = 0; i < feat_len; ++i) {
        probabs[i] = featureMat.at<float>(0, i);
    }
}
```

Результат эксперимента



Агрегация решений
Коллектив решающих правил

Простое голосование

```
string makePredictionSimpleVote(vector<vector<float>> probabsAggregated, string valueList[], int numberOfClasses = 2)
{
    vector<int> votesToClasses(numberOfClasses);
    for (int i = 0; i < probabsAggregated.size(); i++)
    {
        int pos = findMaxValuePosition(probabsAggregated[i]);
        votesToClasses[pos] ++;
    }
    return valueList[findMaxValuePosition(votesToClasses)];
}
```

Усреднение вероятностей

```
string makePredictionAverage(vector<vector<float>> probabsAggregated, string valueList[], int numberOfClasses = 2)
{
    vector<float> resultProb(numberOfClasses);
    for (int i = 0; i < numberOfClasses; i++)
    {
        float sum = 0;
        for (int j = 0; j < probabsAggregated.size(); j++)
        {
            sum += probabsAggregated[j][i];
        }
        sum = sum / probabsAggregated.size();
        resultProb[i] = sum;
    }
    return valueList[findMaxValuePosition(resultProb)];
}
```

Усреднение вероятностей

```
string makePredictionAverageGeom(vector<vector<float>> probabsAggregated, string valueList[], int numberOfClasses = 2)
{
    vector<float> resultProb(numberOfClasses);
    for (int i = 0; i < numberOfClasses; i++)
    {
        float sum = 0;
        for (int j = 0; j < probabsAggregated.size(); j++)
        {
            sum += log(probabsAggregated[j][i]);
        }

        resultProb[i] = sum;
    }
    return valueList[findMaxValuePosition(resultProb)];
}
```

План работы

- Проведение сравнительного анализа способов агрегации решений, полученных при распознавании каждого видеокадра
- Экспериментальное исследование их точности и быстродействия
- Выступление на конференции "Информационные системы и технологии 2017"
- Проведение хорошего эксперимента
- Применение более сложных способов агрегации решений

Спасибо за внимание!