NATIONAL RESEARCH UNIVERSITY

ЭФФЕКТИВНЫЕ АЛГОРИТМЫ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ

Савченко А.В.
Докт. техн. наук, проф. каф. ИСиТ,
в.н.с. международной лаборатории алгоритмов и технологий анализа сетевых структур
НИУ ВШЭ-НН
Email: avsavchenko@hse.ru
URL: http://www.hse.ru/staff/avsavchenko

15.06.2021

План

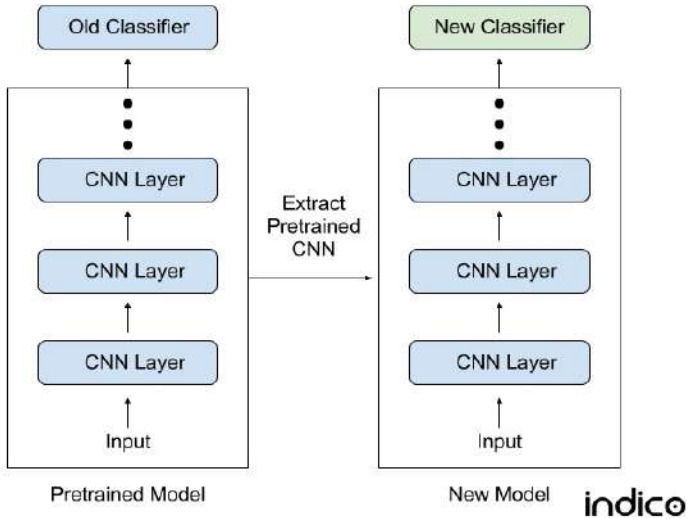
1. Постановка задачи
2. Выбор архитектуры нейронной сети
3. Повышение скорости прямого прохода (inference) в нейронной сети
4. Повышение вычислительной эффективности классификации
5. Пример прикладной задачи

Постановка задачи

Требуется отнести входное изображение X к одному из C классов. Задано обучающее множество из N эталонных изображений $\{X_n\}$, $n \in \{1,\dots N\}$ с известной меткой класса $c_n \in \{1,\dots C\}$

Проблема малых выборок: число изображений каждого класса недостаточно для обучения сложного классификатора: N/C<const

1. Дообучение (Fine-tuning) сверточной нейронной сети (CNN)



2. Классифицировать извлеченный с помощью CNN вектор признаков (*embeddings*) $\mathbf{x}=[x_1,\dots, x_D]$.

- Низкая вычислительная эффективность прямого прохода по глубокой сверточной нейронной сети

- Высокая вычислительная сложность классификации. Для полного перебора (instance-based learning) сложность: O(*RD*).

- Многокритериальная оптимизация (минимизируем вероятность ошибочной классификации и время принятия решений:

$$\bar{\alpha} \to \min \qquad \bar{t} \le t_0$$

Особенности мобильных устройств

–**Мало памяти!?**
- 64-256 Гб на топовых устройствах

–**Низкая вычислительная эффективность!?**

Но MobileNet inference time:
- 50-60 мс на Samsung S9+
- 20 мс на MacBook 2016

–**Энергопотребление!**

–**Другая платформа (Android, iOS). Где Python?**

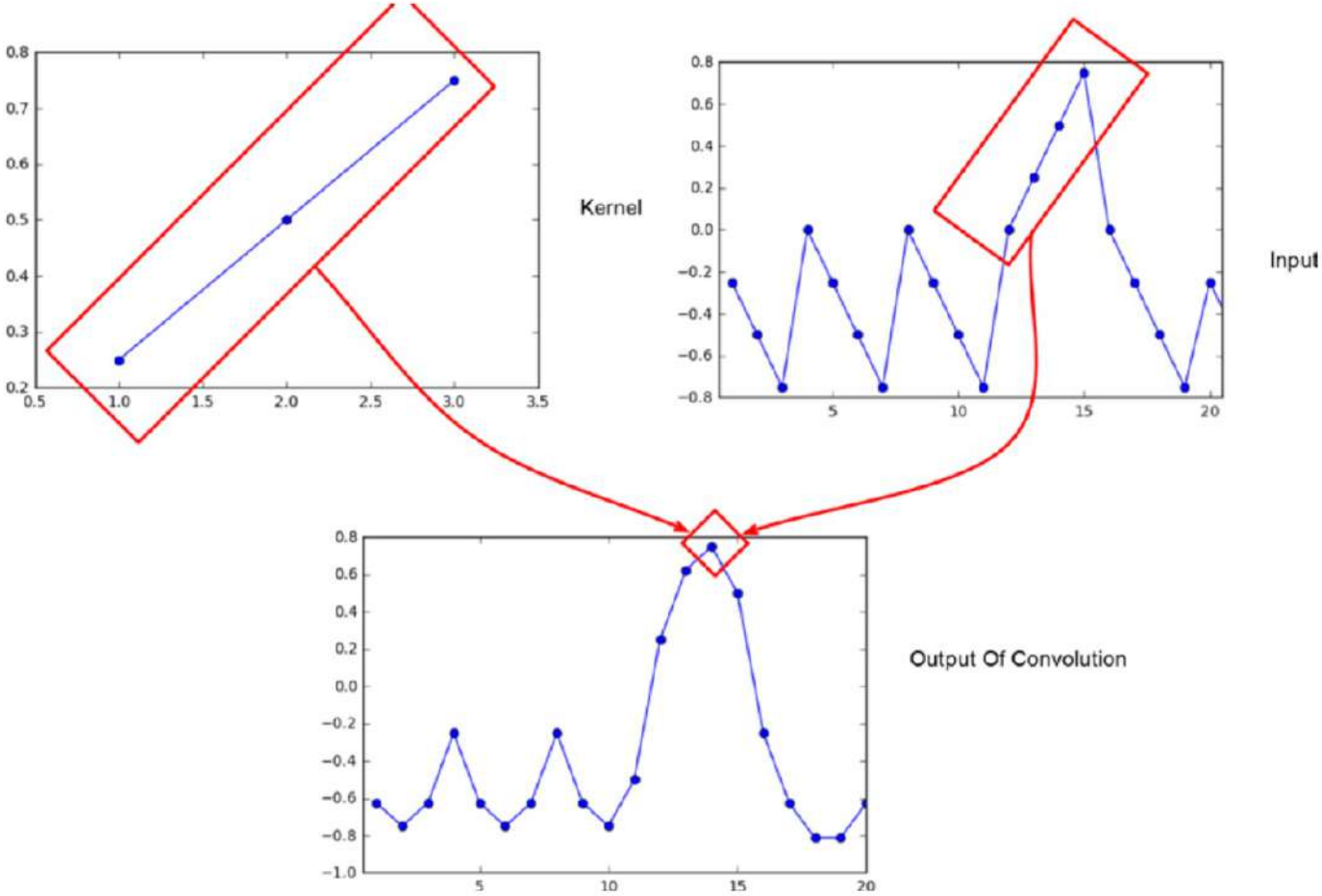
Выбор архитектуры нейронной сети

Свертка (convolution)

$$s(t) = \sum_a I(t-a) \cdot K(a)$$

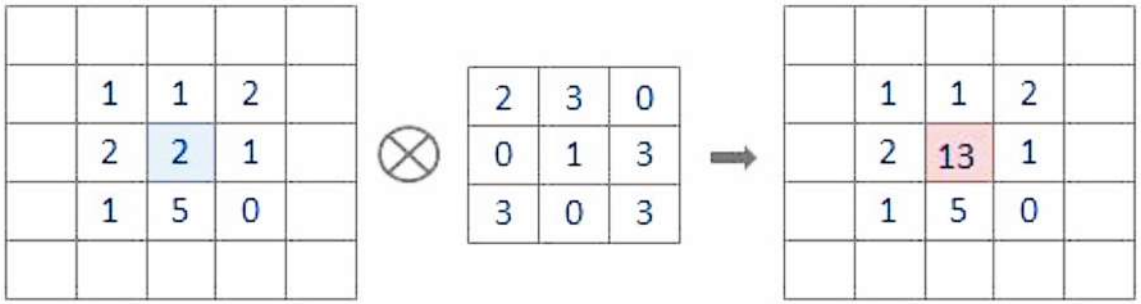
Кросс-корреляция

$$s(t) = \sum_a I(t+a) \cdot K(a)$$

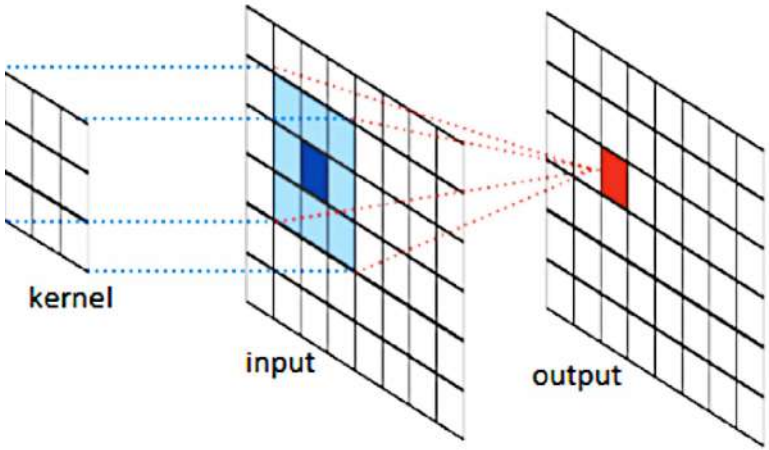


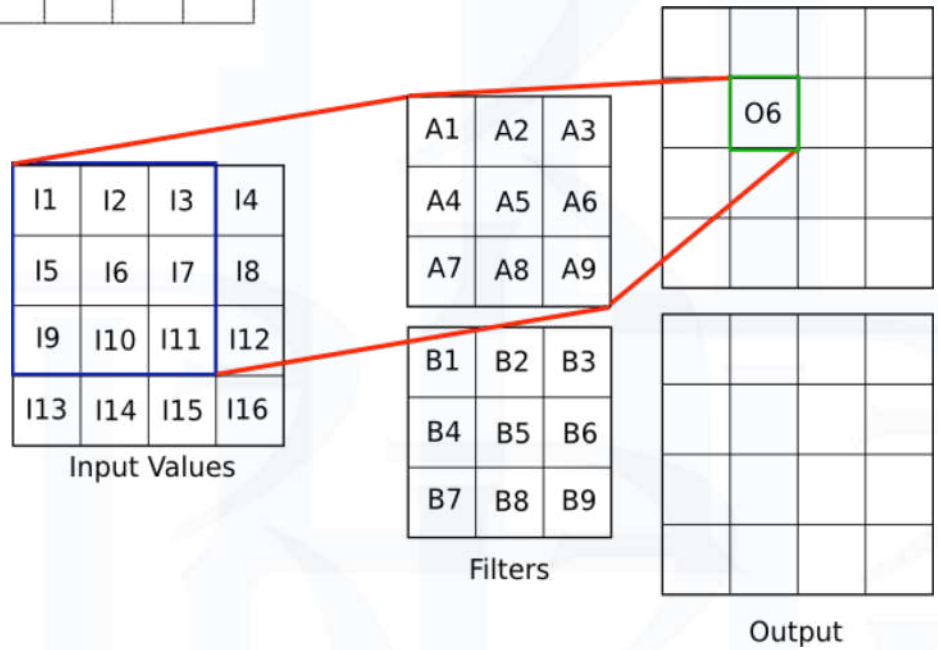
Ketkar “Deep Learning with Python”

Линейная фильтрация (2). Двумерный сигнал (изображение)



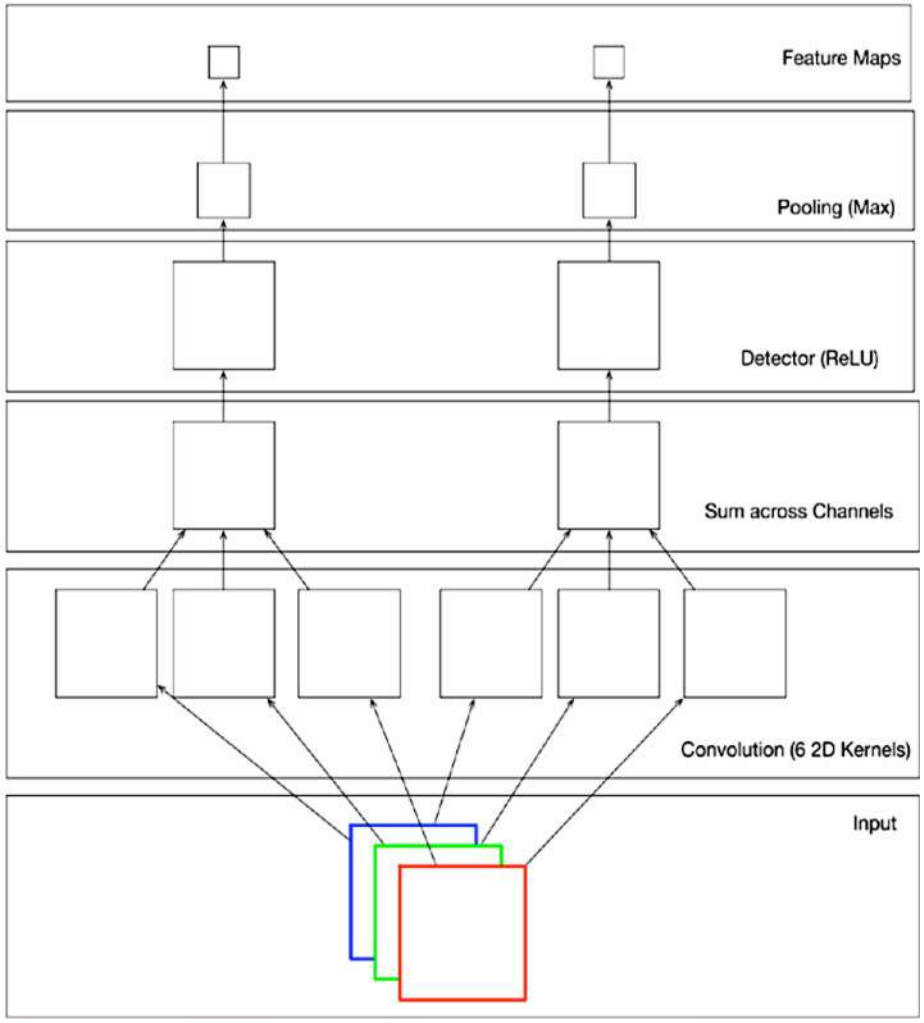
$$x_{ij} = \sum_{m=-M}^{M} \sum_{n=-N}^{N} p_{i-m,j-n} f_{m,n}$$



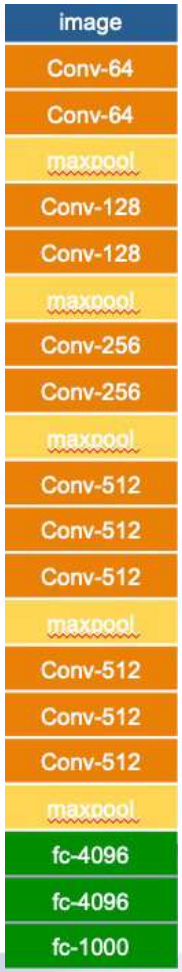


http://intellabs.github.io/RiverTrail/tutorial/

DLI-Teaching-Kit

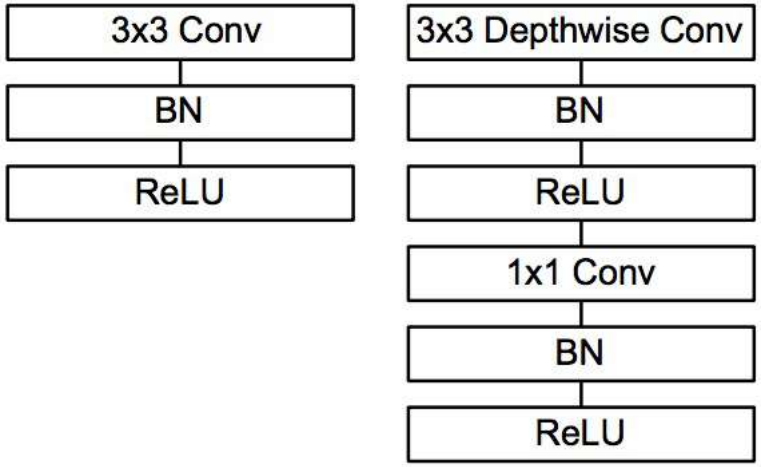


VGGNet (16-19 слоев, ~500 Мб)



InceptionResNet (550 layers)

Depthwise convolution

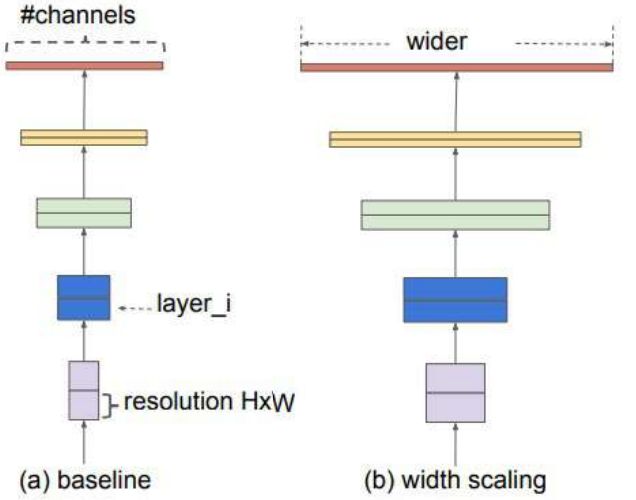


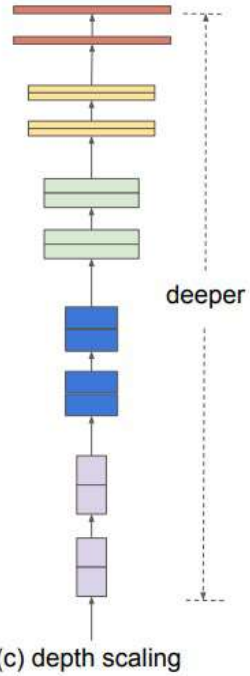
- **28 слоев**
- **ImageNet Top-5 Error rate: 12.81%**
- **4.2M весов**

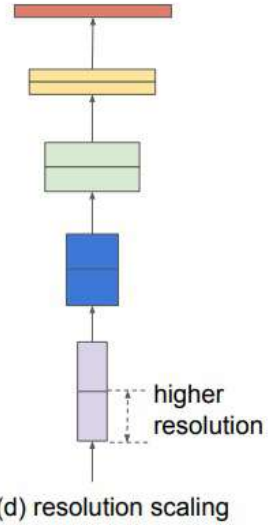
https://arxiv.org/abs/1704.04861

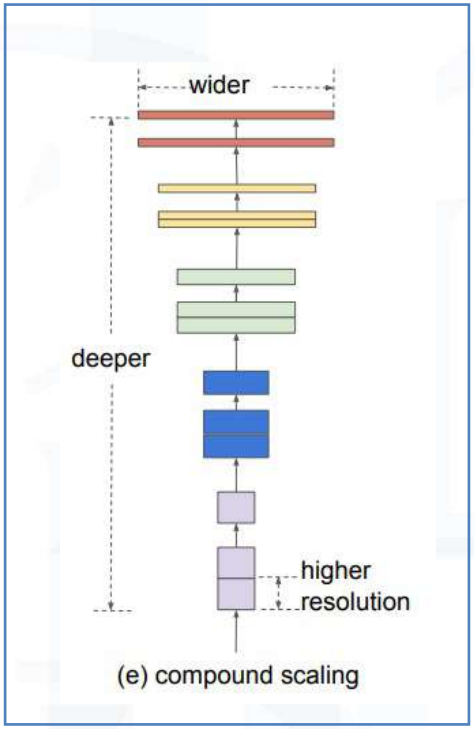
Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
5× Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

- Автоматический поиск архитектуры (MnasNet)
- Масштабирование (scaling) моделей





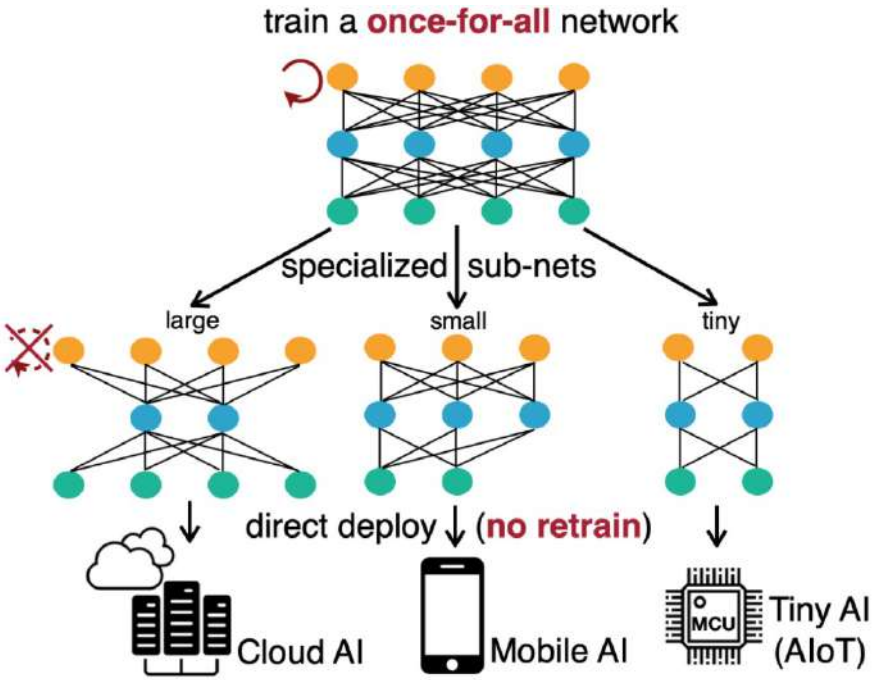




	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

https://arxiv.org/pdf/1905.11946.pdf

Генерация CNN для заданной архитектуры. SuperNet



Snapdragon 865

Model	Tflite CPU, ms	Tflite GPU, ms	Accuracy, %
enet-b0	28±1	14±1	76.1
Subnet-1	15±1	11±2	77.5
Subnet-2	9.4±0.6	7±2	76.5

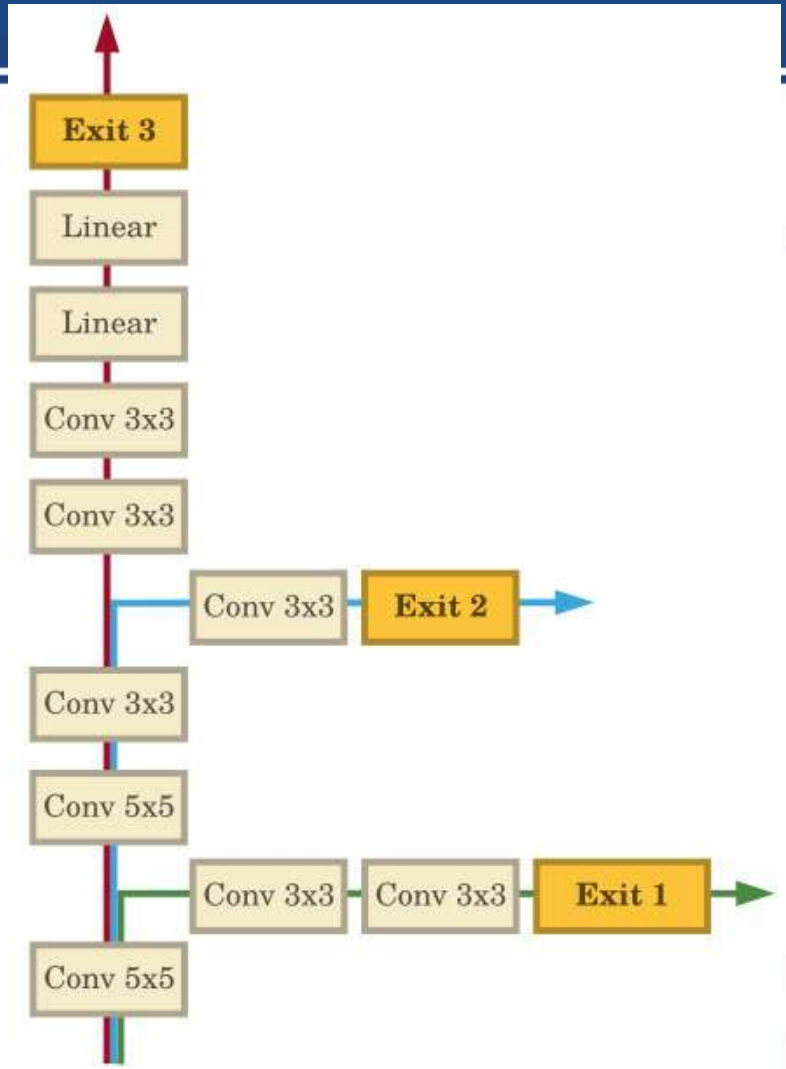
Snapdragon 765

Model	Tflite CPU, ms	Tflite GPU, ms	Accuracy, %
enet-b0	65±5	24±1	76.1
Subnet-1	38±2	18±1	77.9
Subnet-2	28±1	13±1	76.2

https://arxiv.org/pdf/1908.09791.pdf

Повышение скорости прямого прохода (inference) в нейронной сети

BranchyNet



https://arxiv.org/pdf/1709.01686.pdf

- Выбираются M > 1 промежуточных слоев (exit branches)
- К каждому выходу добавляются слои GAP (Global average pooling) и L2-norm.
- *X* описывается последовательностью векторов признаков $\mathbf{x}_1, ..., \mathbf{x}_M$.
- Последовательный анализ: *M* классификаторов предсказывают степени уверенности(confidence scores) $s_m^{(c)}(\mathbf{x}_m) \geq 0$

$$c_m^*(\mathbf{x}_m) = \underset{c \in \{1,...,C\}}{\operatorname{argmax}} s_m^{(c)}(\mathbf{x}_m). \quad (2)$$

$$\max_{c \in \{1,...,C\}} s_m^{(c)}(\mathbf{x}_m) > s_m. \quad (3)$$

Require: observed image
Ensure: class label
1: Initialize $\mathbf{z}_0$ by the RGB matrix of an input image X
2: **for** each intermediate layer $m \in \{1, ..., M\}$ **do**
3: Compute the output of the m-th layer $\mathbf{z}_m = f_{exit_m}(\mathbf{z}_{m-1}; \boldsymbol{\theta})$ (1)
4: Transform activations $\mathbf{z}_m$ into feature vector $\mathbf{x}_m$
5: Predict the confidence scores $\mathbf{s}_m(\mathbf{x}_m)$ using the m-th classifier
6: **if** $m = M$ OR condition (3) holds **then**
7: **return** class label $c_m^*(\mathbf{x}_m)$ (2)
8: **end if**
9: **end for**

- Как выбрать пороги?
- Кросс-валидация, зафиксировав вероятность ошибки второго рода (false acceptance rate, FAR) α_m

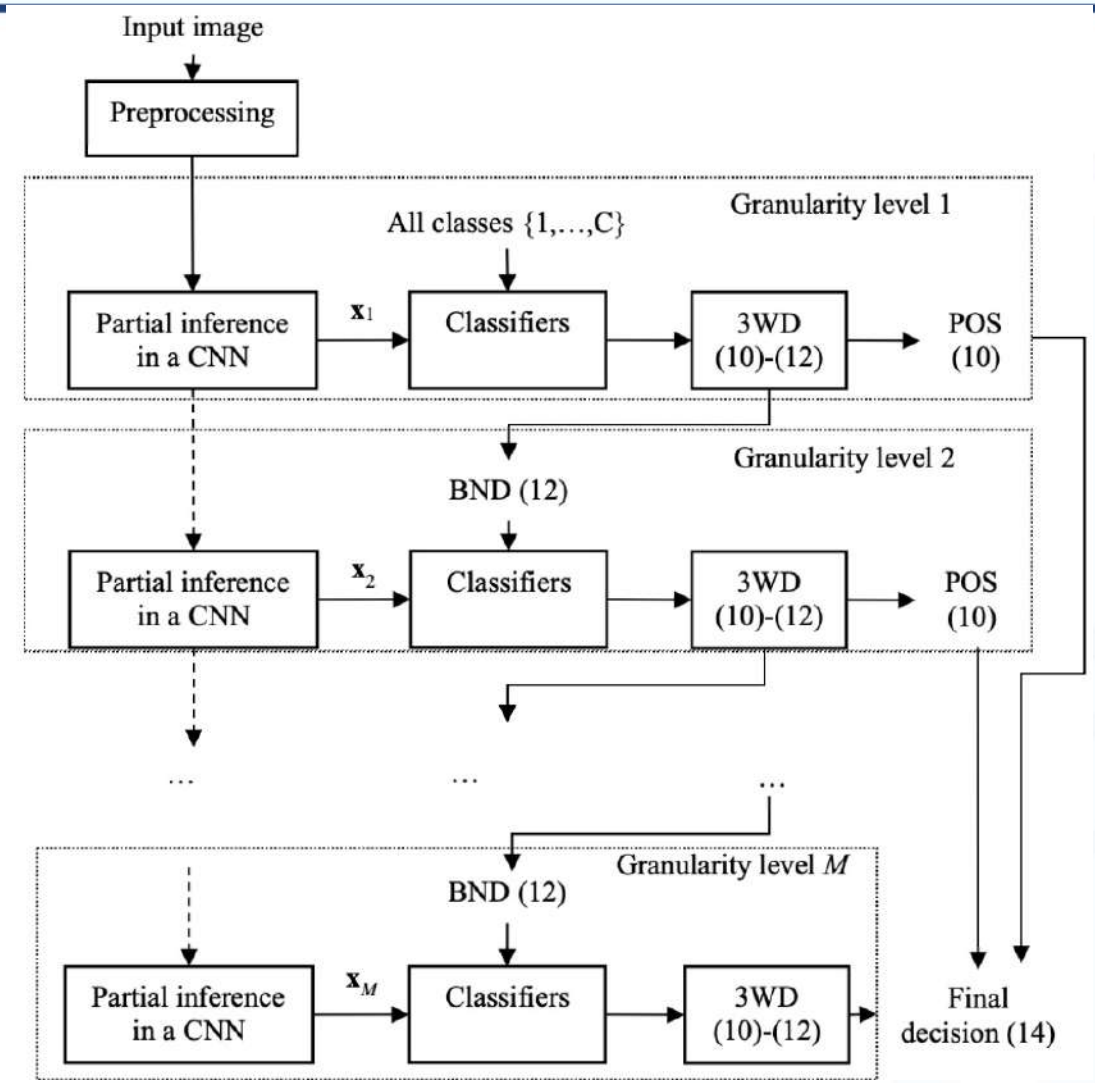
$$\sum_{n \notin \{n_1,\dots,n_K\}} H\left[\max_{c \neq c(n)} s_m^{(c)}(\mathbf{x}_{n;m}) - s_m\right] = \lfloor \alpha_m (N-K) \rfloor, \quad (4)$$

- Как выбрать FAR для *m*-го классификатора, если задан только уровень значимости для всей процедуры α?
- Проблема множественных сравнений, процедура Benjamini-Hochberg

$$\alpha_m = \alpha \cdot m / M$$

1: (Optional) Add fully-connected LR classifiers to all M branches and fine-tune the CNN on the given training set
2: **for** each intermediate layer $m \in \{1, ..., M\}$ **do**
3: **for** each training instance $n \in \{1, ..., N\}$ **do**
4: Feed the n-th image into a CNN and compute the outputs $\mathbf{x}_{n;m}$ at the m-th layer
5: **end for**
6: Split N instances in a stratified fashion using train/test split ratio δ to get indices $\{n_1, ..., n_K\}, K = \lceil N \cdot \delta \rceil$
7: Train the m-th classifier, e.g., the linear one-versus-all SVM in order to maximize the squared hinge loss, using feature vectors $\{(\mathbf{x}_{n_k;m}, c(n))\}, k \in \{1, ..., K\}$
8: Initialize a list of maximal intra-class scores $S = []$
9: **for** each validation instance $n \notin \{n_1, ..., n_K\}$ **do**
10: Append the maximal inter-class confidence score $\max_{c \neq c(n)} s_m^{(c)}(\mathbf{x}_{n;m})$ to the list S
11: **end for**
12: Assign the $\lfloor \alpha \cdot m(N-K)/M \rfloor$-th largest element from S to the threshold s_m (4) using the Benjamini-Hochberg correction
13: Retrain the m-th classifier using all feature vectors $\{(\mathbf{x}_{n;m}, c(n))\}, n \in \{1, ..., N\}$
14: **end for**
15: **return** M classifiers and their thresholds $\{s_m\}$

Предложенный подход на основе последовательного анализа



- Savchenko, IJCNN 2020
- Savchenko, Information Sciences 2021
- Савченко, Записки научных семинаров ПОМИ, 2021

Результаты экспериментов, EfficientNet-b7

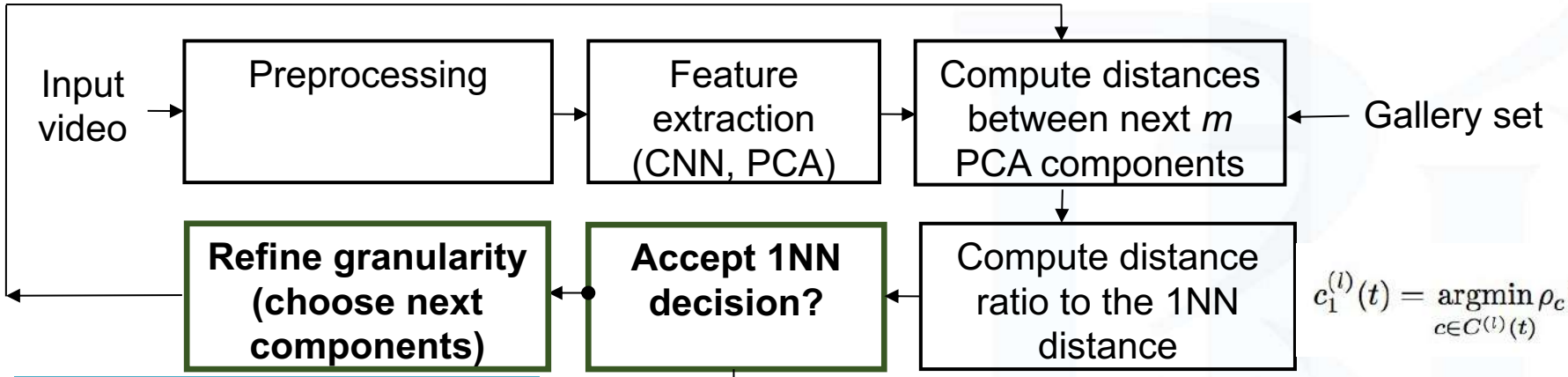
Classifier	Layers	Caltech-101		Caltech-256		Stanford Dogs	
		block6b_add+block6f_add+penultimate		block6f_add+block7b_add+penultimate		block7b_add+penultimate	
		accuracy, %	time, ms	accuracy, %	time, ms	accuracy, %	time, ms
RF	first	60.10	176.47	46.52	185.48	86.75	205.87
RF	last	91.40	215.18	82.90	215.41	92.80	215.06
Linear SVM	first	82.12	172.72	71.18	181.76	86.82	198.03
Linear SVM	last	95.24	211.63	92.16	211.66	93.41	207.40
Ours, fixed threshold	all	92.07	178.71	83.42	191.46	90.53	200.27
Ours, adaptive threshold	all	94.49	181.91	91.13	197.82	92.99	202.18

Пример для Caltech-101

Prediction at “block_6b_add” ($s_1 = 1.83$)	Confidence Class	1.14 scorpion	0.81 cannon	0.97 beaver	0.34 wrench	1.64 elephant
Prediction at “block_6f_add” ($s_2 = 1.36$)	Confidence Class	1.64 crab	1.66 anchor	2.03 dolphin	18.38 mayfly	1.52 kangaroo

Повышение вычислительной эффективности классификации

$$\rho_c(\tilde{\mathbf{x}}^{(l)}(t)) = \min_{r\in\{1,\dots,R\},c(r)=c} \rho(\tilde{\mathbf{x}}^{(l)}(t), \tilde{\mathbf{x}}_r^{(l)})$$



Strong theoretical foundations for the Jensen-Snannon and Kullback-Leibler divergences

- В разы быстрее полного перебора
- Значимо точнее, чем сравнение главных компонент

- [Savchenko, Information Systems, 2019]
- [Savchenko, Knowledge-Based Systems, 2016]

Вход Ближайшие эталонные изображения

(a) (b) (c) (d)

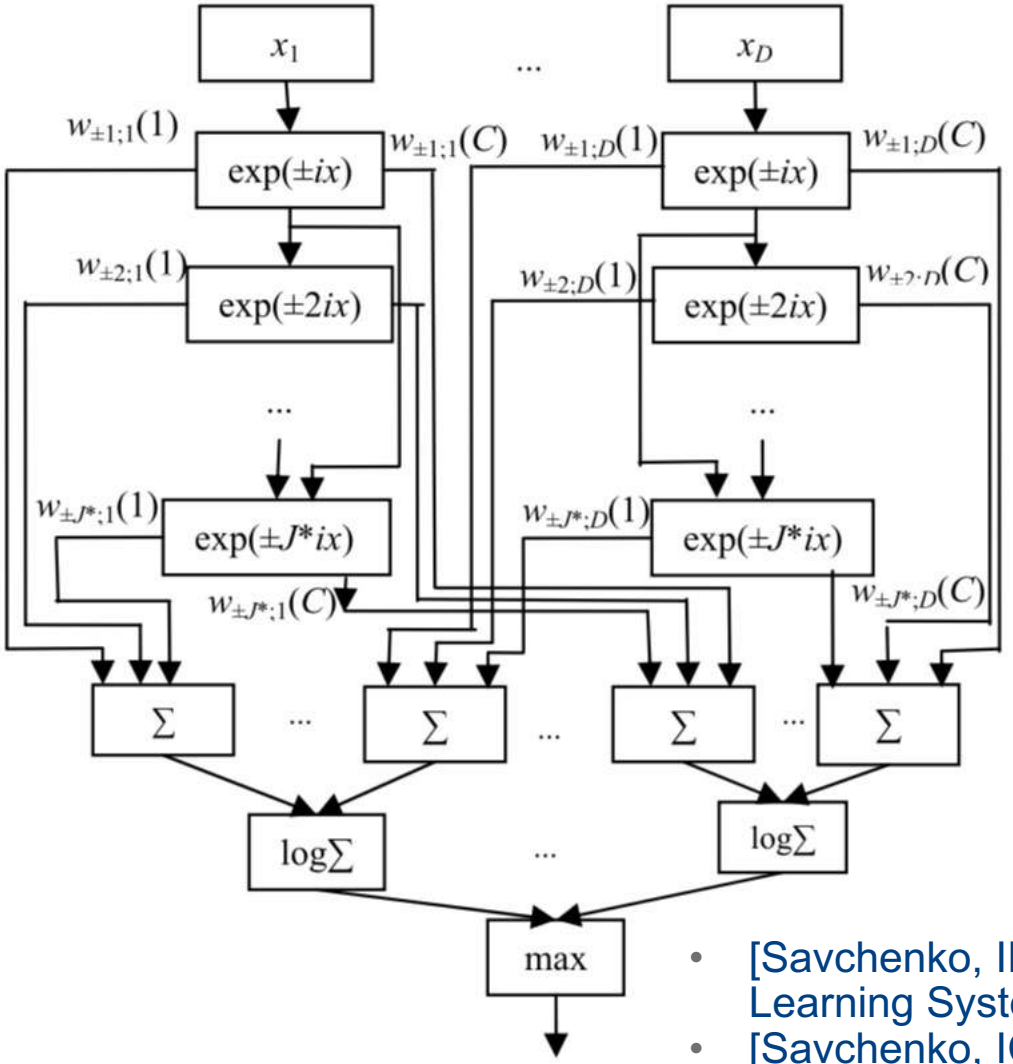
Результаты распознавания, порог 0.7

Subject	$l=1$		$l=2$		$l=3$	
	$\rho[r]$	$\rho_{\min}/\rho[r]$	$\rho[r]$	$\rho_{\min}/\rho[r]$	$\rho[r]$	$\rho_{\min}/\rho[r]$
Armstrong (d)	0.0086	0.87	0.0122	1	0.0129	1
Auriemma (b)	0.0074	1.00	0.0170	0.71	0.0195	0.66
McEwen (c)	0.0104	0.72	0.0188	0.65	-	-
Williams	0.0100	0.75	0.0300	0.41	-	-
Wirayuda	0.0103	0.73	0.0217	0.56	-	-
LeBron	0.0105	0.71	0.0200	0.61	-	-

Быстрее (не нужно сравнивать все признаки)

Высокая точность

Вероятностная нейронная сеть с тригонометрическими функциями активации



- Сходится к Байесовскому решению.
- Высокая скорость обучения: новый обучающий пример добавляется в реальном времени
- Классификация точнее и примерно в $(R/C)^{2/3}$–раз быстрее, чем instance-based learning (PNN, k-NN)

Проблемы

- Медленнее PNN при $R/C<5$
- Менее точна, чем SVM, RF,… при $R/C>25$

- [Savchenko, IEEE Transactions on Neural Networks and Learning Systems, 2020]
- [Savchenko, ICPR 2018]

Метод максимального правдоподобного перебора

$$r_{k+1} = \underset{\nu \in \{1,\dots,R\} - \{r_1,\dots,r_k\}}{argmax} \left(p_\nu \cdot \prod_{i=1}^{k} f\left(\rho(X, X_{r_i}) \middle| W_\nu \right) \right),$$

Дивергенция Кульбака-Лейблера асимптотически распределена как нецентральный хи-квадрат.

$$r_{k+1} = \underset{\mu \in \{1,\dots,R\} - \{r_1,\dots,r_k\}}{argmin} \left(\sum_{i=1}^{k} \varphi_\mu(r_i) - \ln p_\mu \right)$$

$$\varphi_\mu(r_i) \approx \frac{\left(\rho(X, X_{r_i}) - \rho_{\mu, r_i} \right)^2}{\rho_{\mu, r_i}}$$

- На порядок быстрее k-NN/PNN

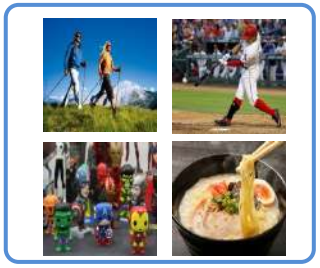
Проблемы

- Точность по сравнению с полным перебором снижается на 0.2-1%
- Дополнительные затраты памяти: не масштабируется на сотни тысяч классов
  - [Savchenko, Pattern Recognition, 2017]
  - [Savchenko, Optimization Letters, 2017]
  - [Savchenko, Pattern Recognition, 2012]

Пример прикладной задачи

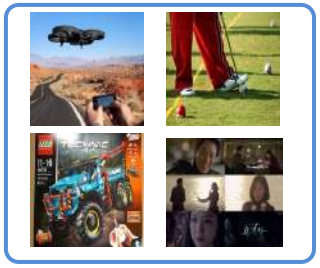
Анализ предпочтений по фото и видео на смартфоне

Генерация профиля пользователя

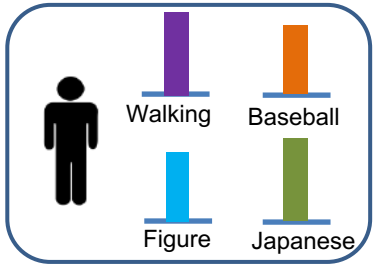


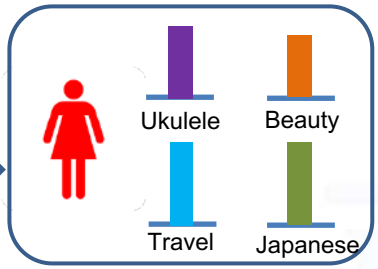
User 1

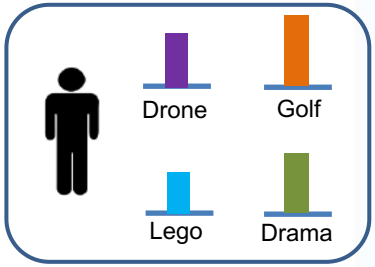
User 2



User 3







Рекомендации

Shop : Japanese (Sushi)

Product : Beauty (SKII)

Content : Drama (Hwayugi)

...

Funded by SAMSUNG

Профиль пользователя

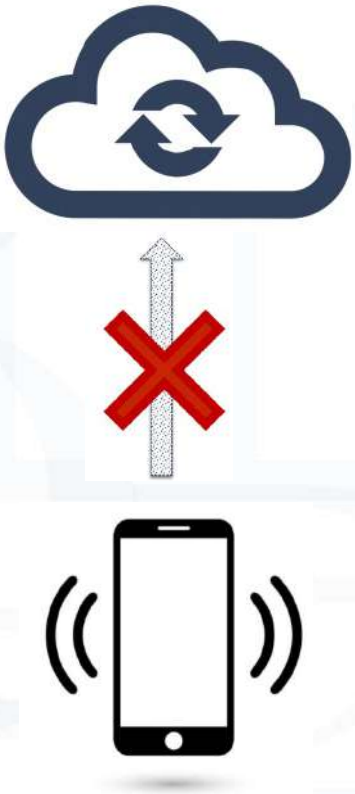
Hobbies	Food code	Pets
Restaurant	Junk food	Dog
Beach	Health - salad and etc.	Cat
Tracks	Sandwich	Fish
Bar	Meat	Horse

Sports	Household Income	Vacation
Fishing	Age	Ski
Golf	Gender	Museum
Diving	Car types	Tracks
Tennis	household	Monuments

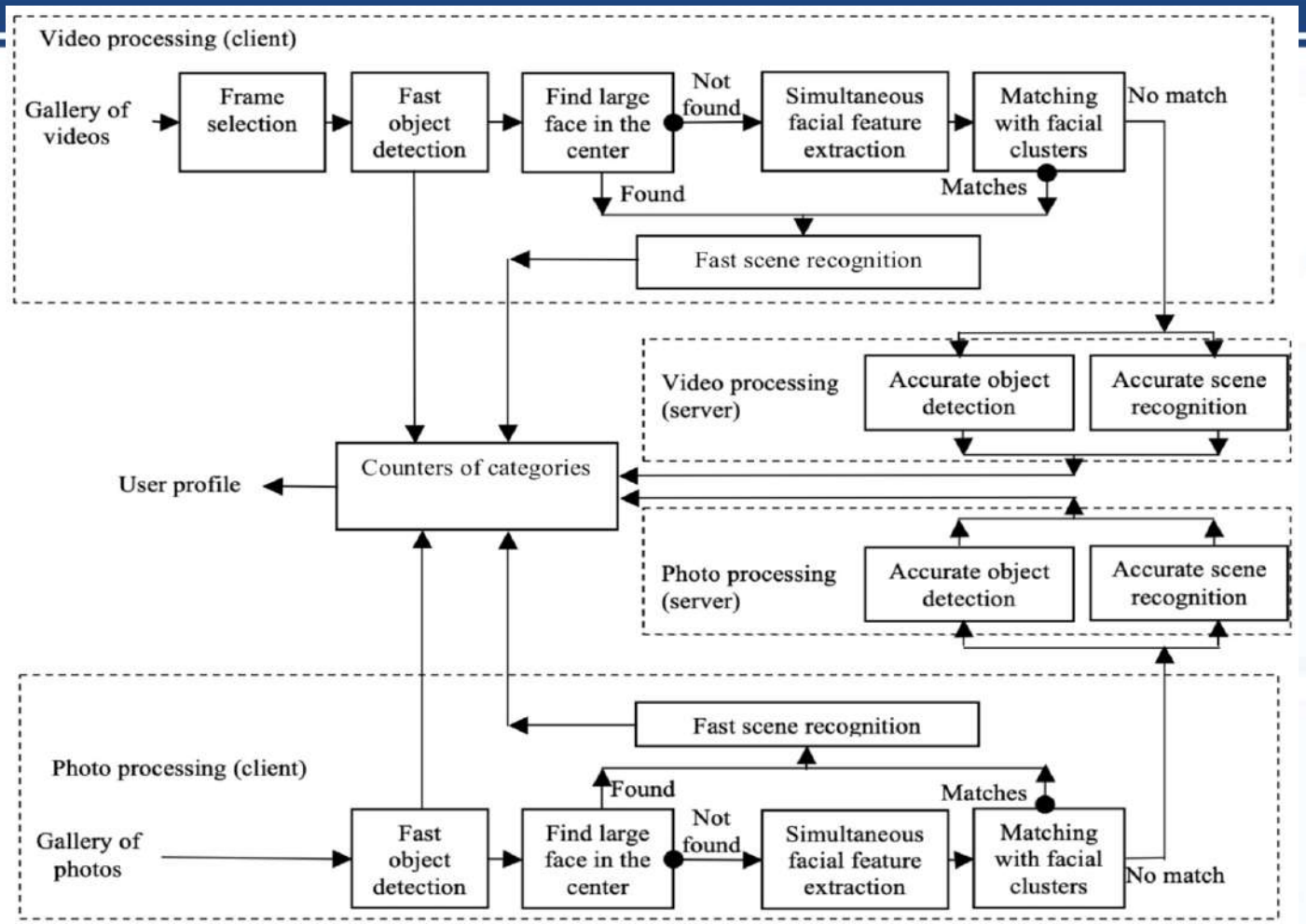
Фотографии как персональные данные

Public photos

Private photos



Предложенный подход



- Savchenko, IJCNN, 2021 (in print)
- https://arxiv.org/abs/1907.04519

Classifier training using both visual features and results of object detection

Scene recognition

- Sport
- Nature
- Athletic field

Embeddings

Object detection

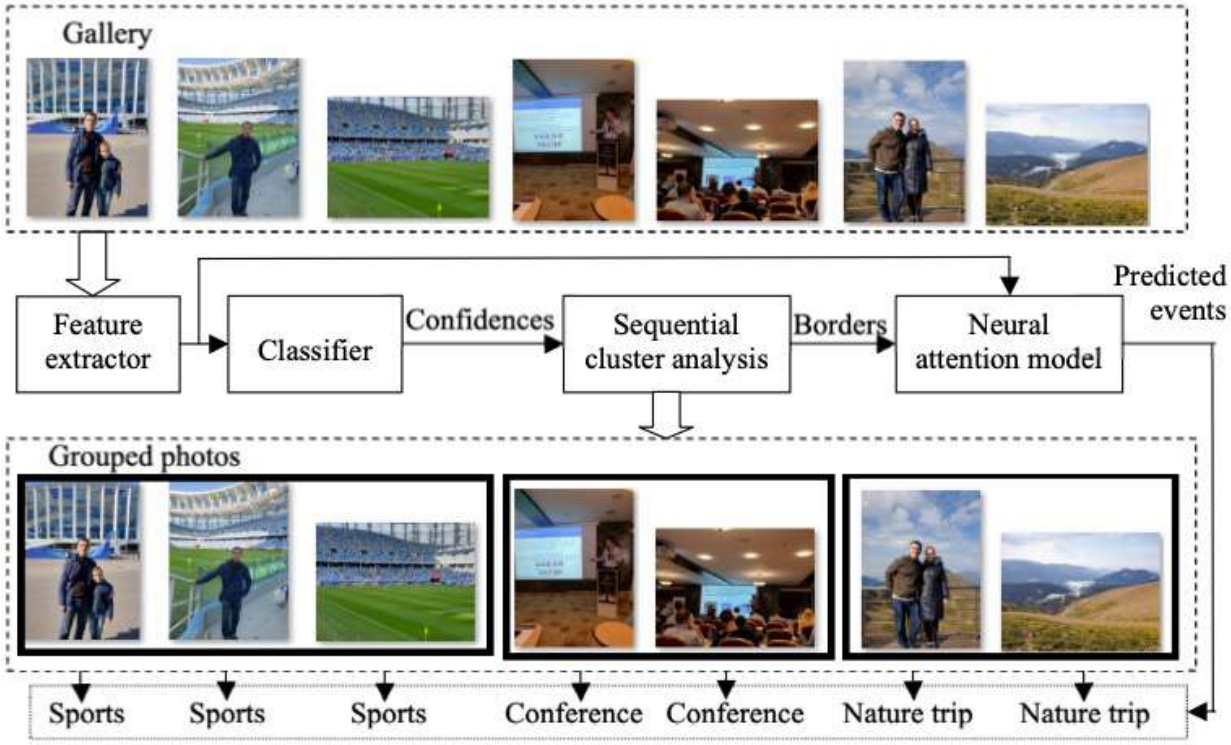
- Person
- Shoes
- Tree

Scores

Ensemble (blending)

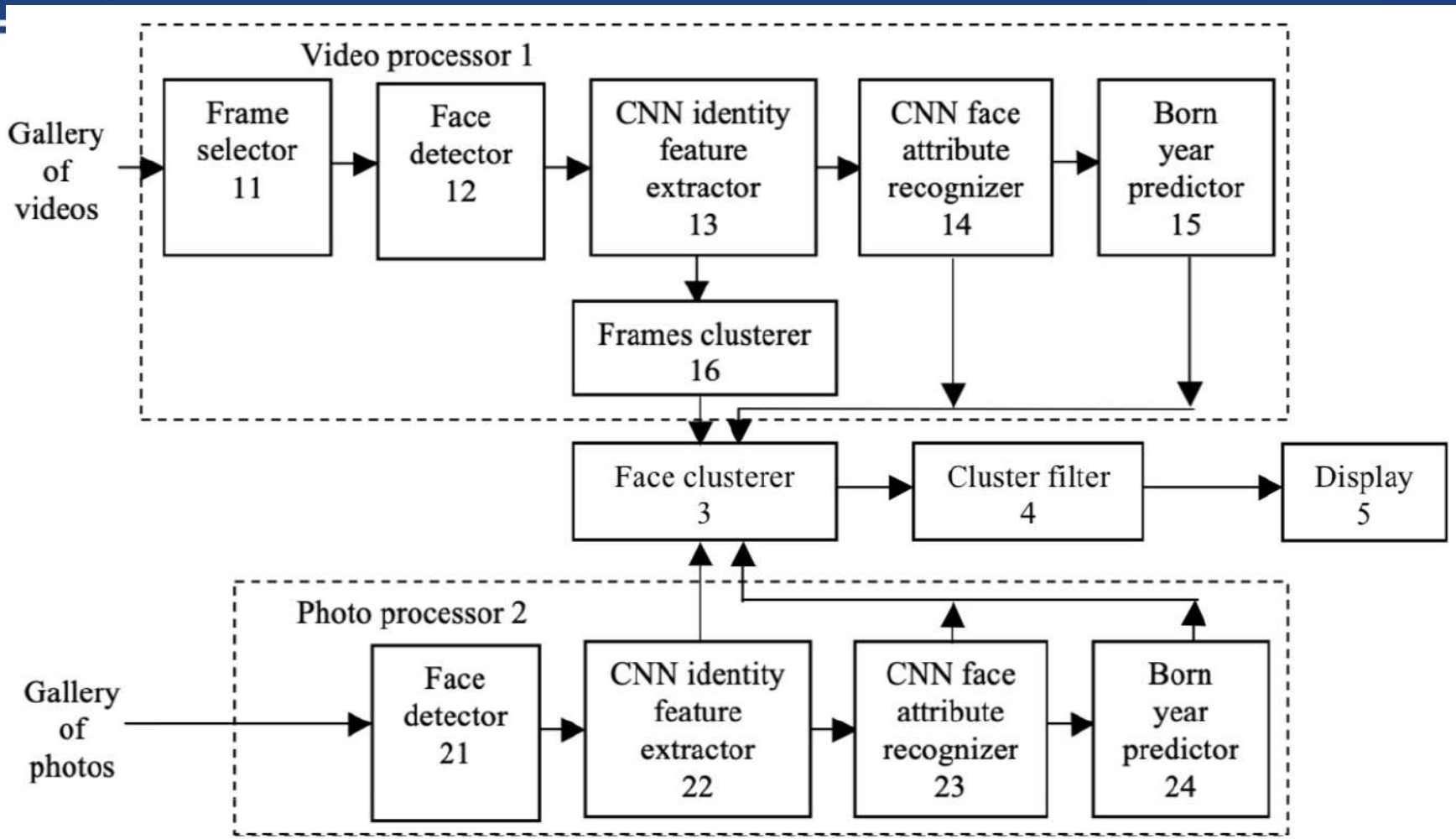
Initialize lists $T_f := [], T_o := []$
for each training image **do**
  Compute features $\mathbf{f}$ by feeding the image into a scene CNN
  Append $\mathbf{f}$ to the list T_f
  Compute maximal confidences of detected objects $\mathbf{o}$ by feeding the image into object detector
  Append $\mathbf{o}$ to the list T_o
end for
Train classifiers $\mathcal{C}_f, \mathcal{C}_o$ using T_f and T_o
for each validation image **do**
  Compute scene features $\mathbf{f}$
  Predict C-dimensional confidence scores $\mathbf{cs}_f$ using classifier $\mathcal{C}_f$
  Compute maximal confidences of detected objects $\mathbf{o}$
  Predict C-dimensional confidences $\mathbf{cs}_o$ using $\mathcal{C}_o$
end for
Assign $\alpha^* := 0$
for all possible weights $w_f \in [0, 1]$ **do**
  for each validation image **do**
    Compute confidences $[cs_1, ..., cs_C] := w_f \mathbf{cs}_f + (1 - w_f)\mathbf{cs}_o$
    Perform blending to predict the best class $c^* := \underset{c \in \{1,...,C\}}{\operatorname{argmax}}\ cs_c$
  end for
  Compute accuracy α using predictions for all validation images
  if $\alpha^* < \alpha$ **then**
    Assign $\alpha^* := \alpha, w_f^* := w_f$
  end if
end for
return classifiers $\mathcal{C}_f, \mathcal{C}_o$ and their weights $w_f^*, (1 - w_f^*)$

На практике: необходимо поставить в соответствие **каждой фотографии** X_t, $t \in \{1, \dots T\}$ из **галереи** один из C классов событий. Обучающее множество – аналогично задаче image-set recognition



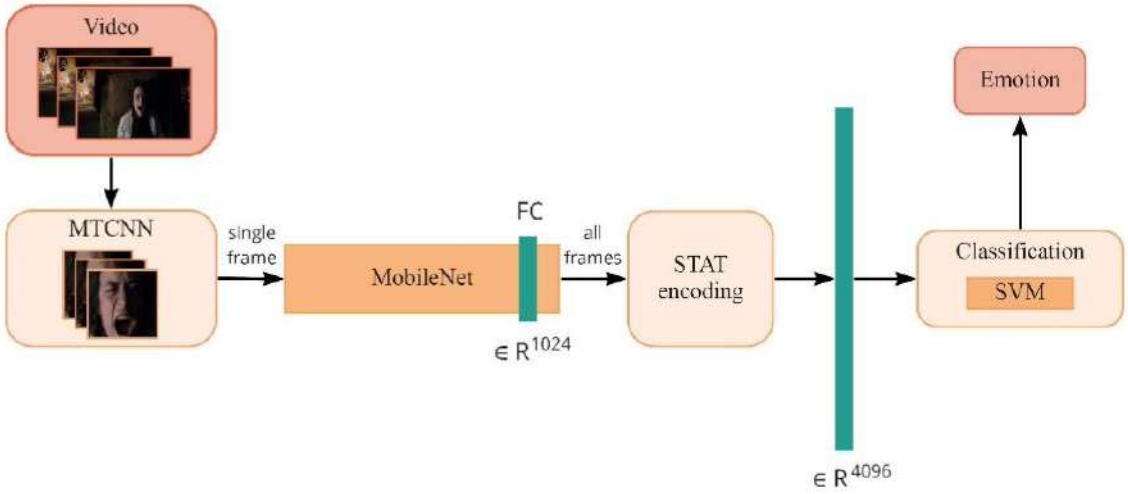
1. Группировка последовательных фото на основе расстояния между L_2-нормированными предсказаниями классификатора (**classifier's scores**)
2. Оценить порог, используя случайные перестановки альбомов из обучающего множества
3. **Механизм внимания**

Savchenko, IJCNN 2020



- Патент RU 2706960 / Савченко А.В., Патентообладатель: САМСУНГ (KR)
- Savchenko, PeerJ Computer Science, 2019

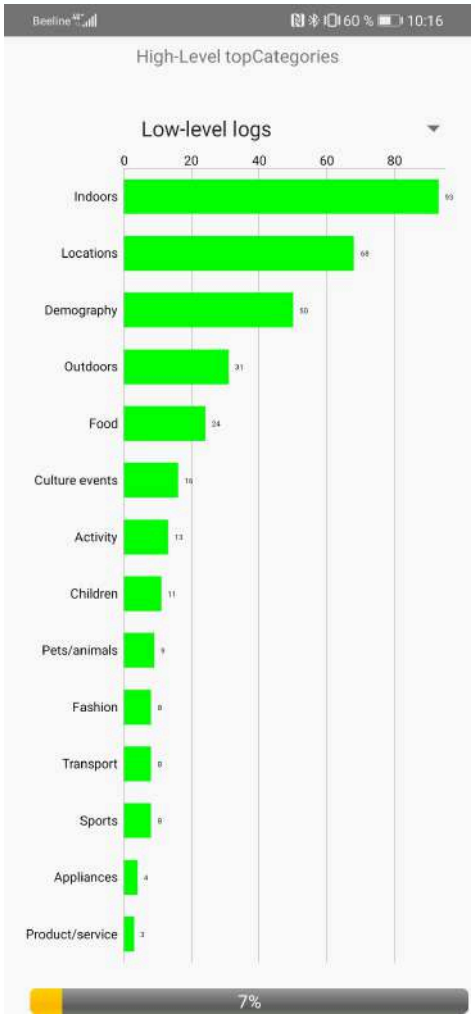
Распознавание эмоций на видео

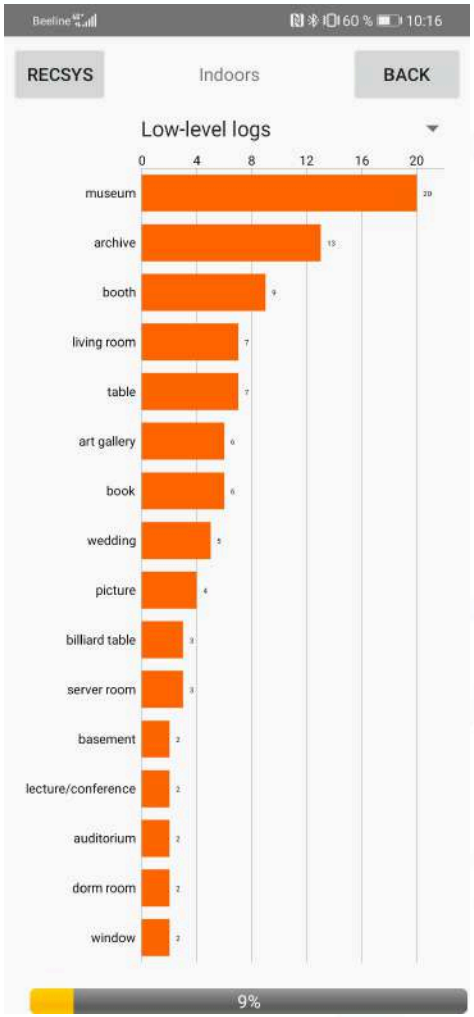


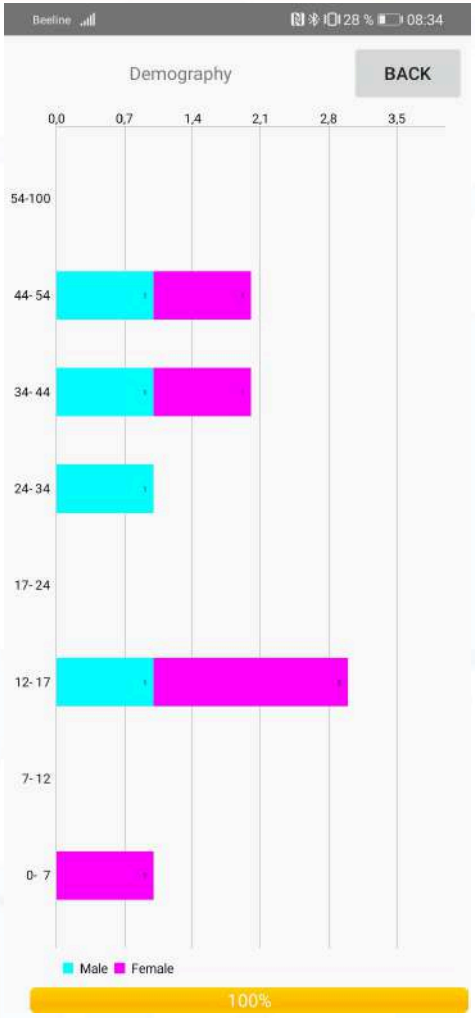
Rank	Model	Accuracy (8 emotion)	Accuracy ↑ (7 emotion)
1	**Emotion-GCN**		66.34
2	**Multi-task EfficientNet-B0**	61.32	65.74
3	**Distilled student**	61.60	65.4
4	**PAENet**	-	65.29
5	**DACL**	-	65.20
6	**ARM** (ResNet-18)	59.75	64.49
7	**CPG**	-	63.57
8	**CAKE**	-	61.7
9	**Facial Motion Prior Network**	-	61.52
10	**CNNs and BOVW + local SVM**	-	59.58

- [Demochkina, Savchenko, ICPR Workshop, 2021]
- https://arxiv.org/abs/2103.17107

Пример мобильного приложения (1)

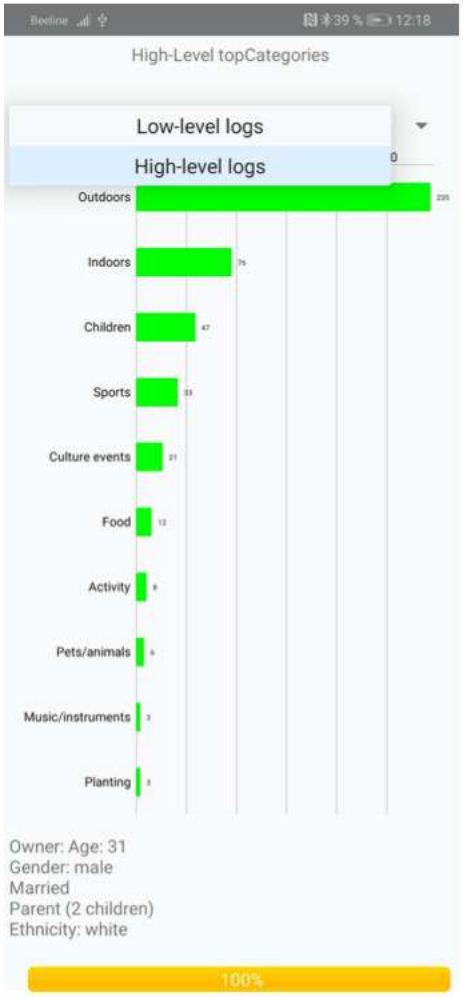


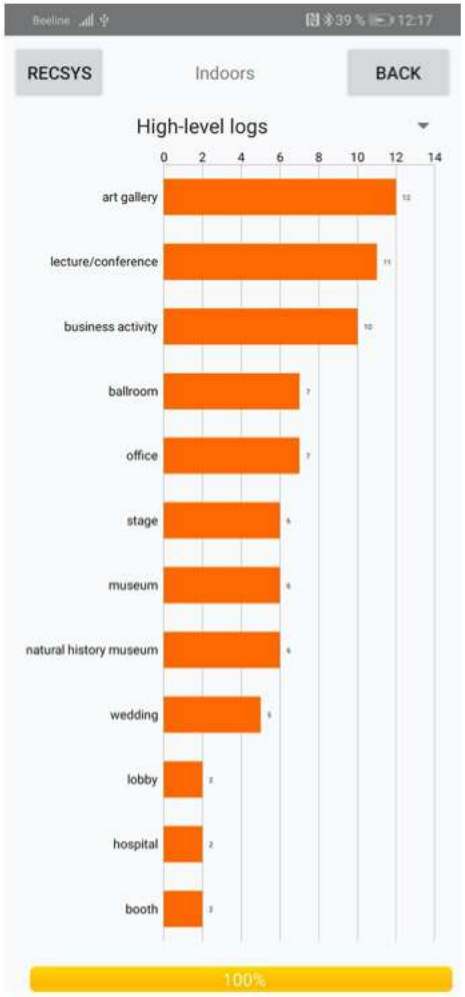




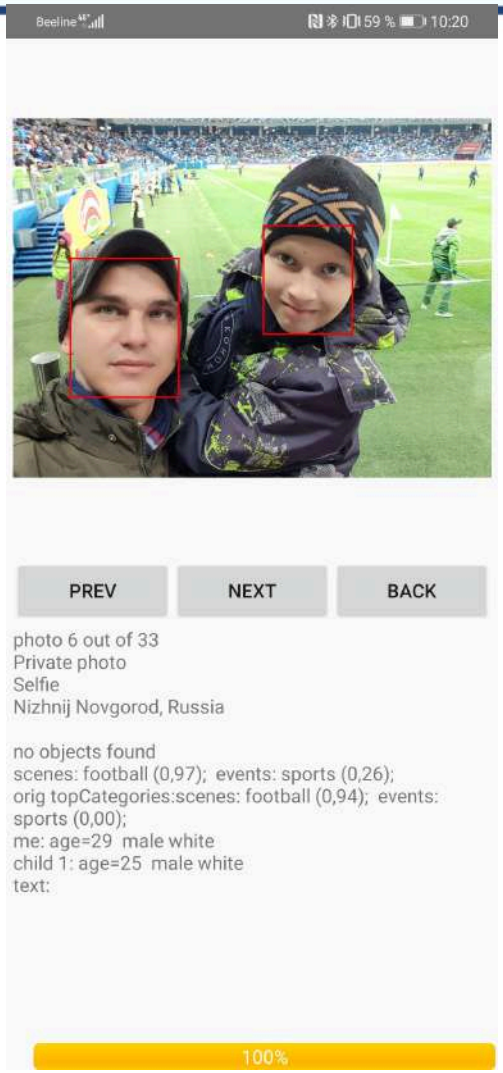
https://drive.google.com/drive/folders/1rQkJZifq_89pu0sT_UnYXziuxutTpNEN

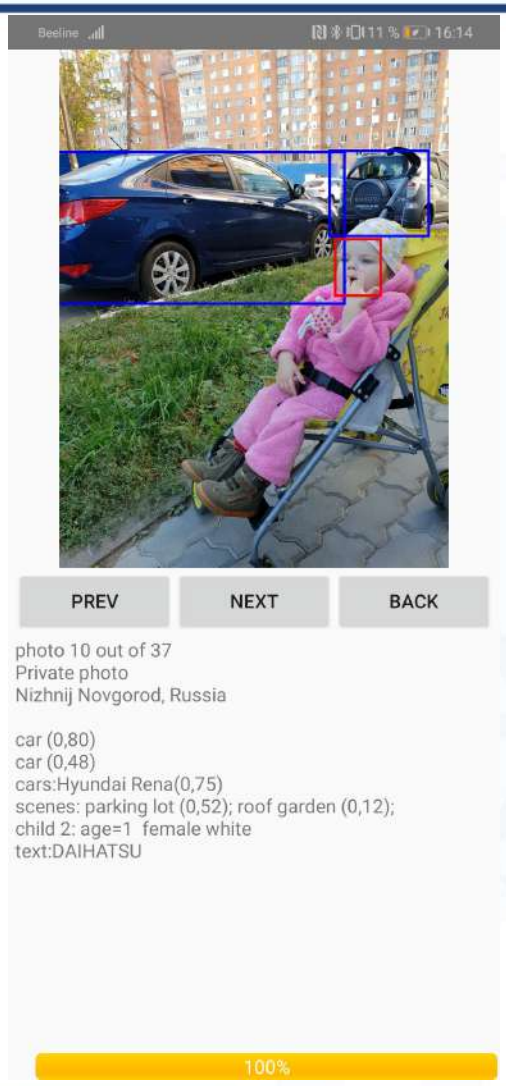
Пример мобильного приложения (2)

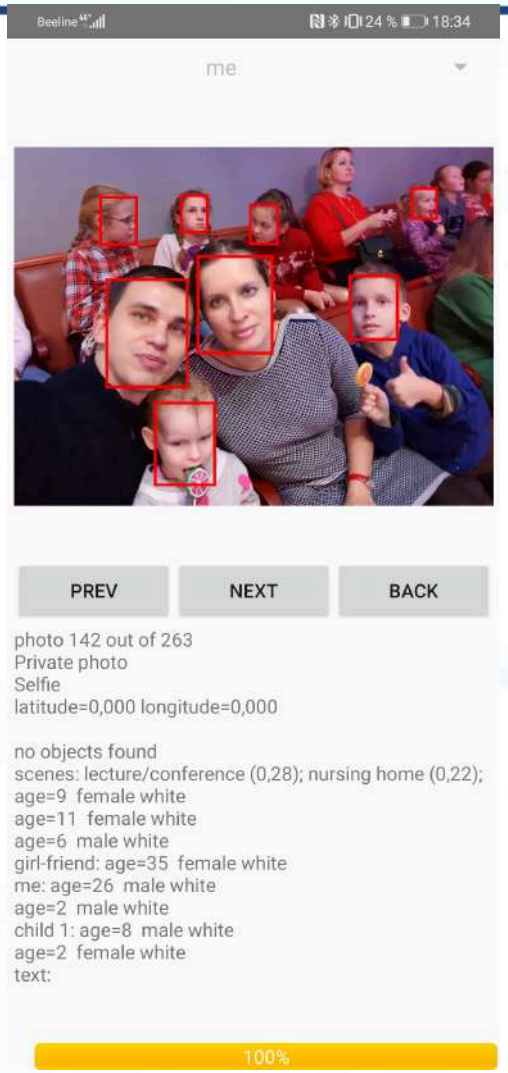




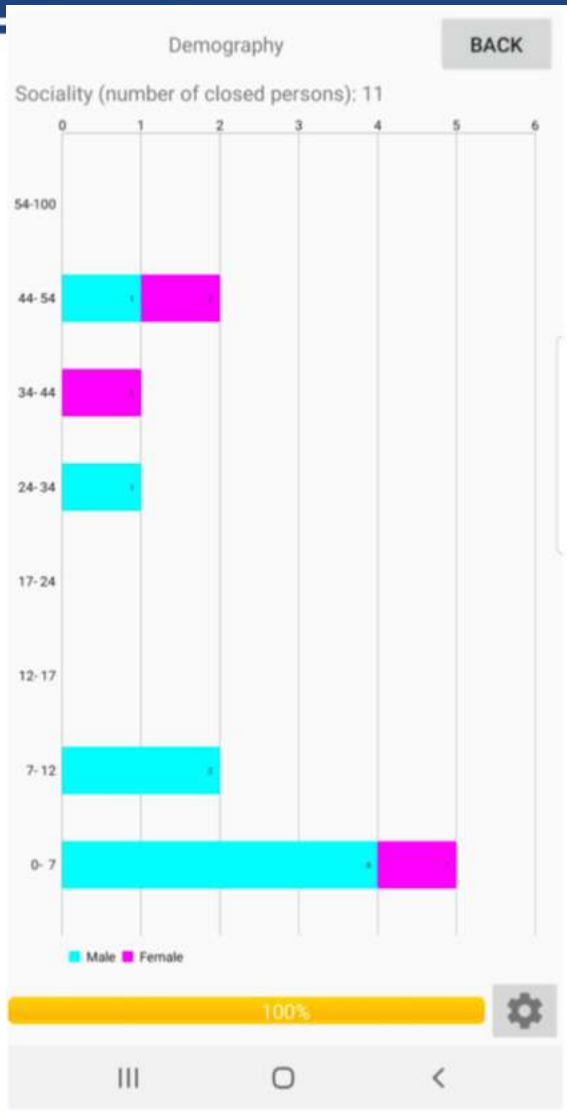
Пример мобильного приложения (3)

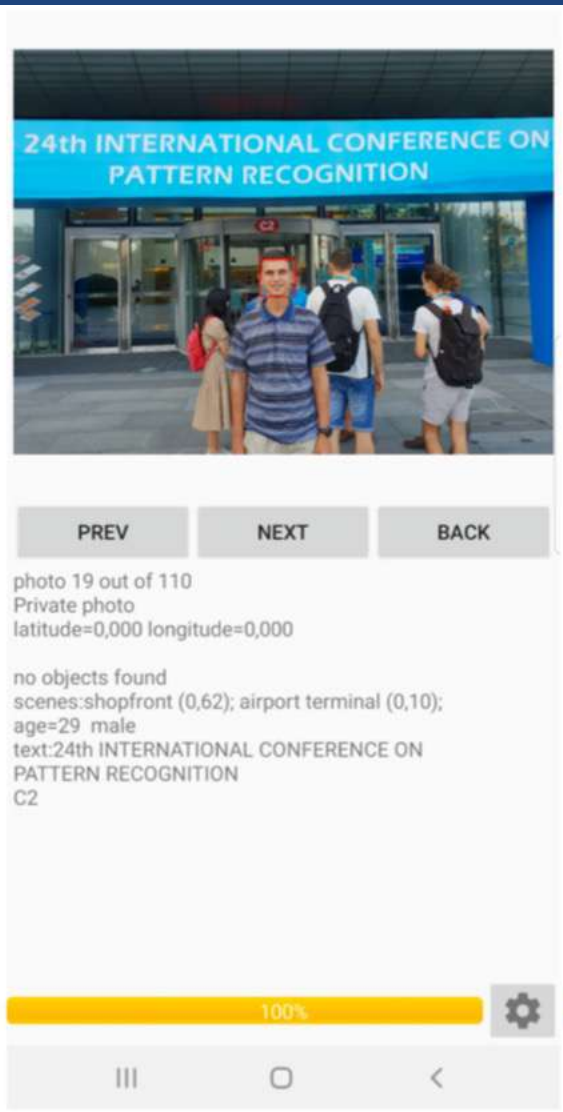


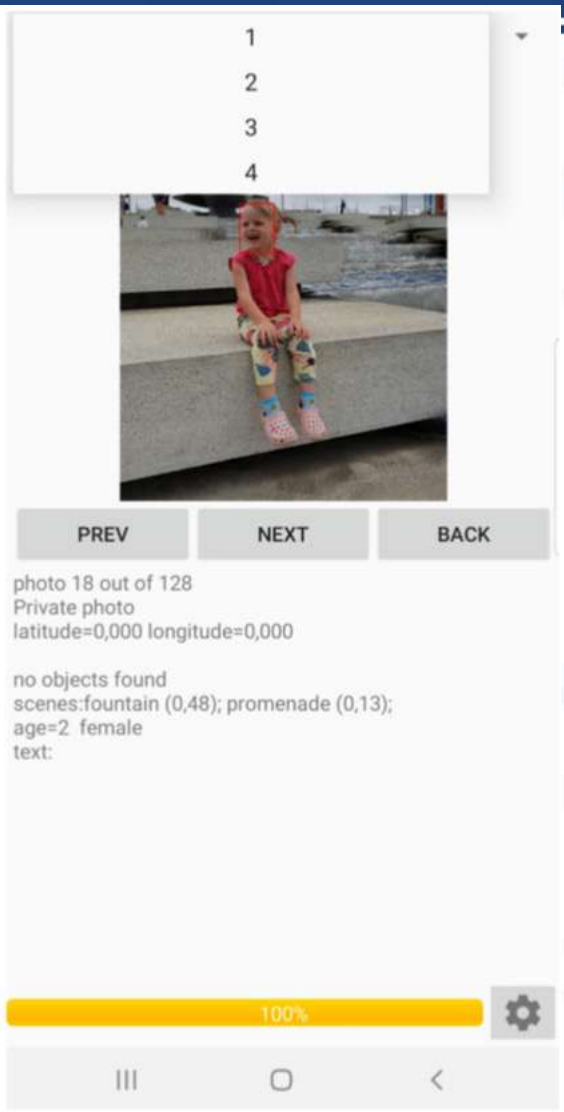




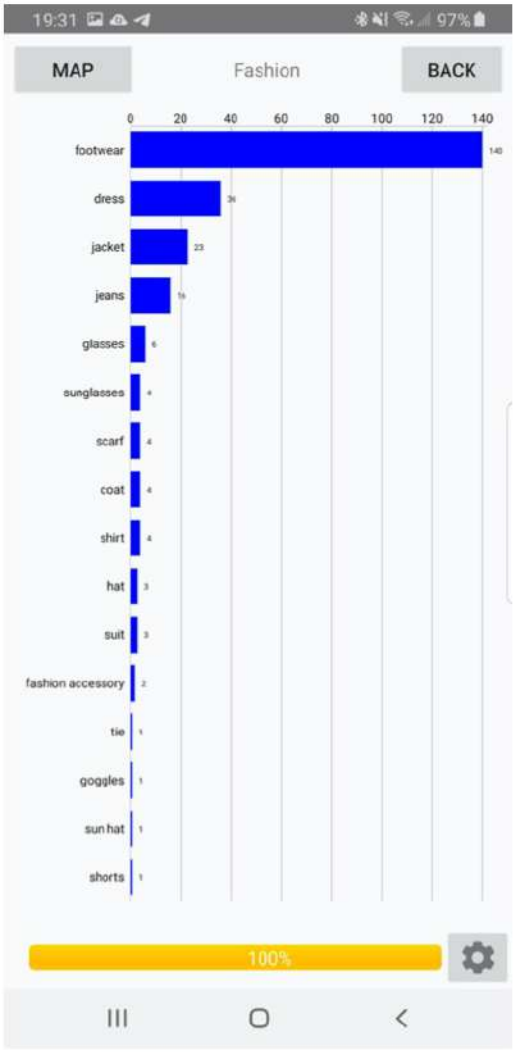
Анализ демографии

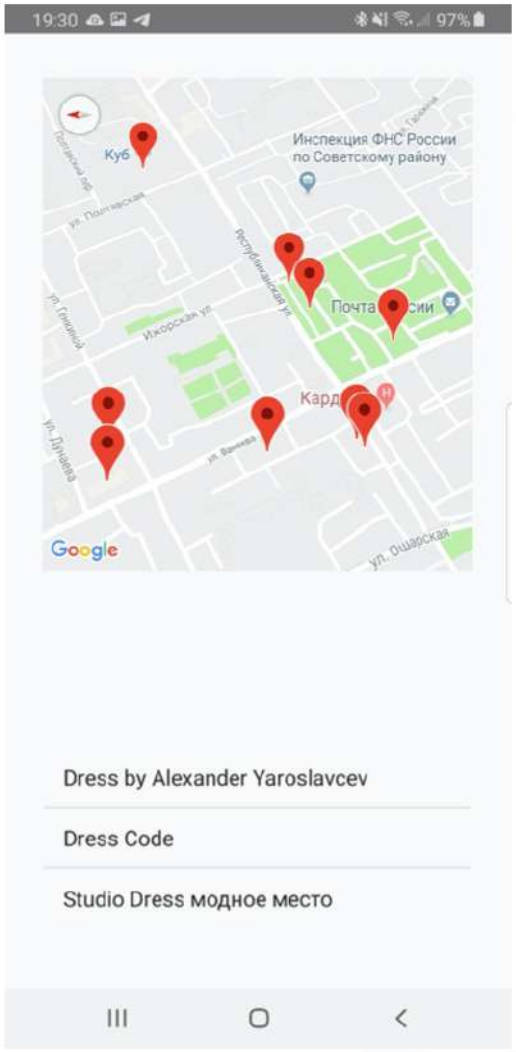


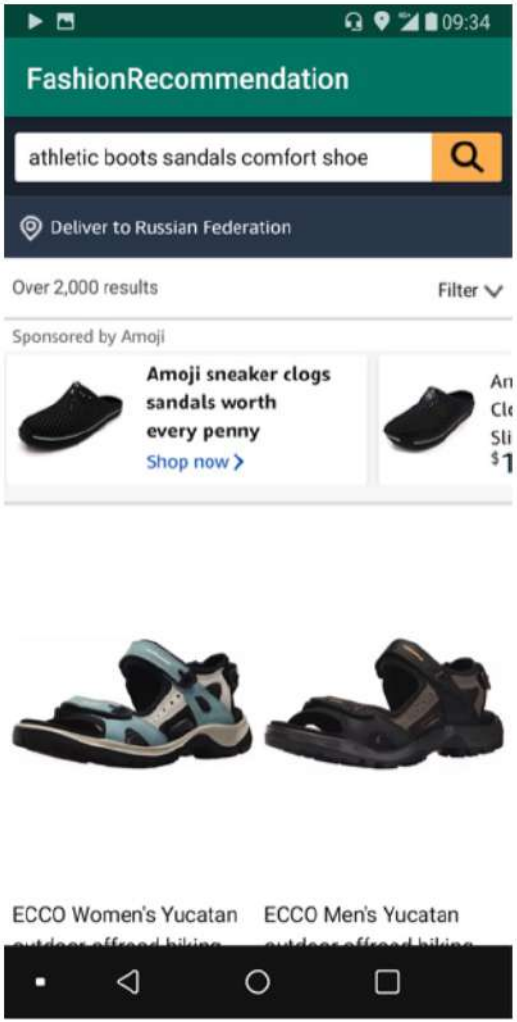




Рекомендации ближайших магазинов

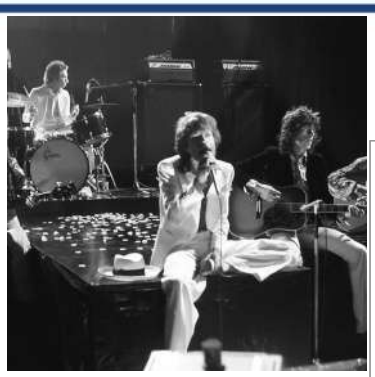




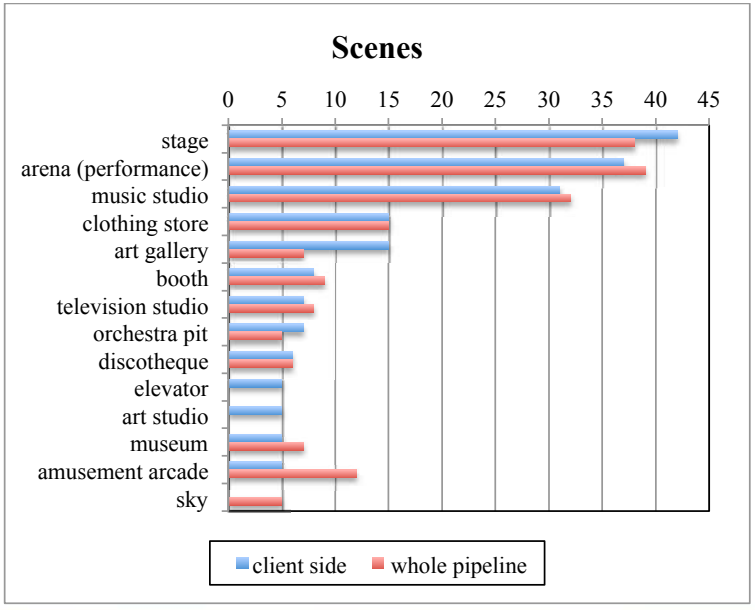


The Rolling Stones profile

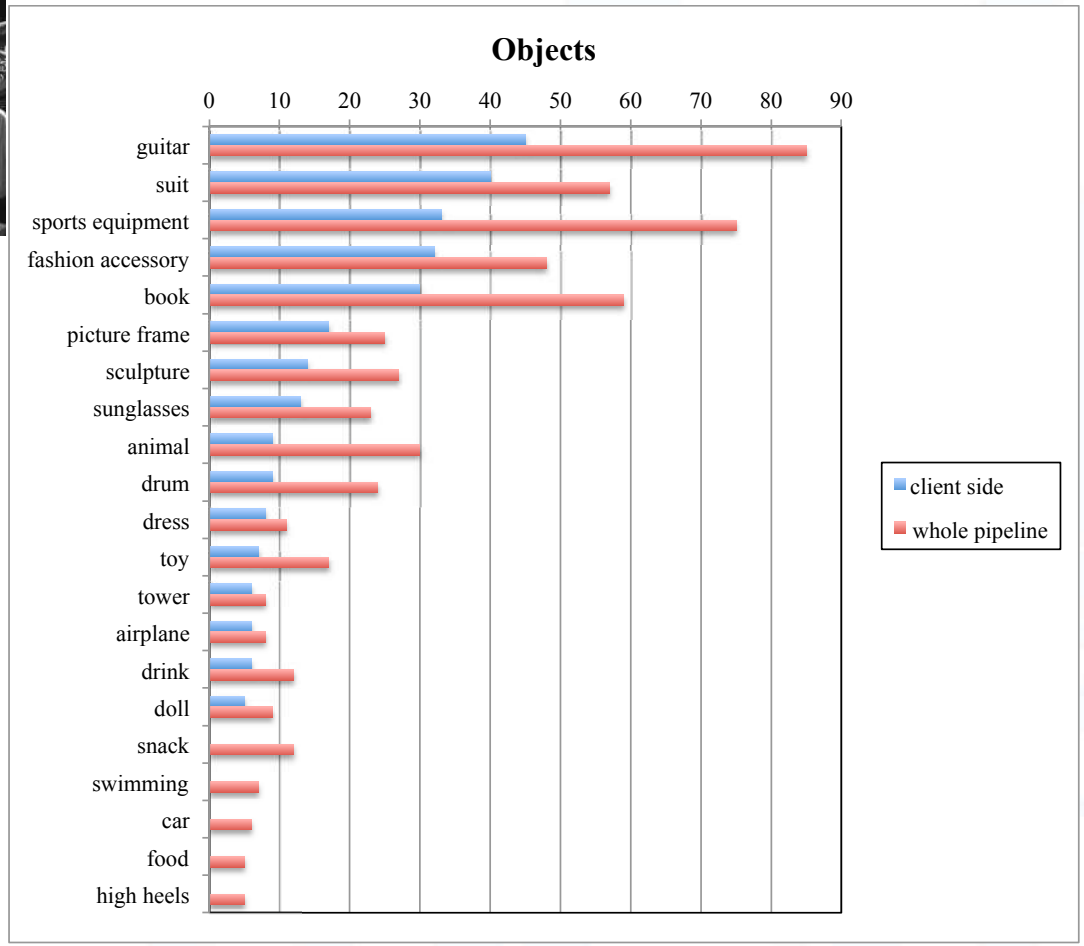
…



Scene recognition



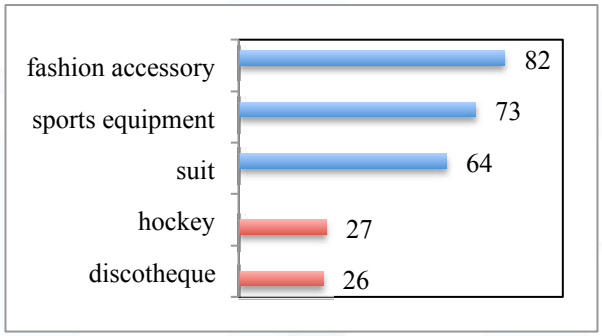
Object detection



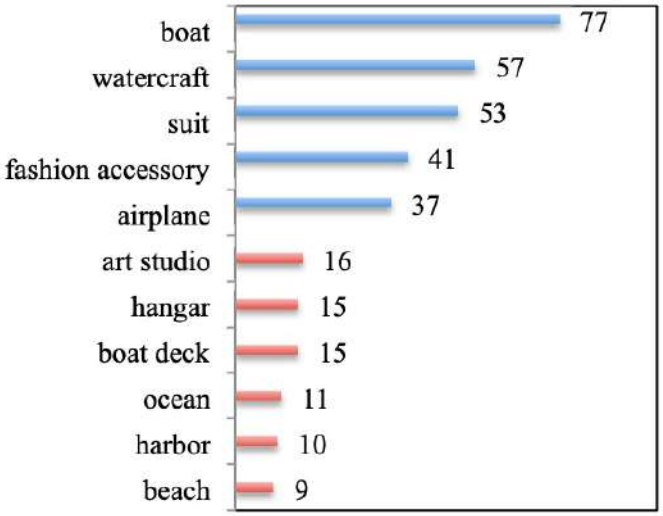
User's profiles from Instagram (1)

Alex Ovechkin

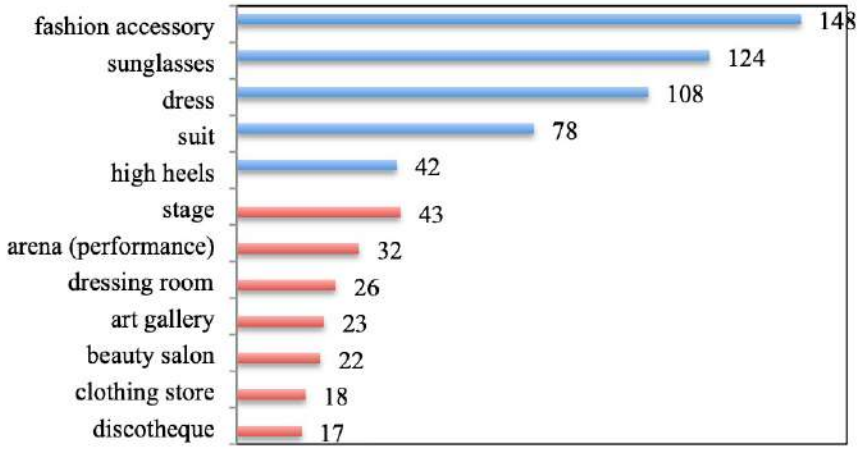
…



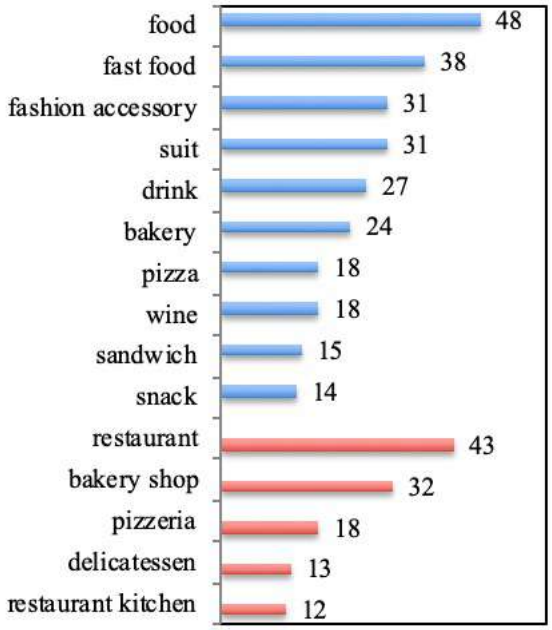
Fedor Konyukhov



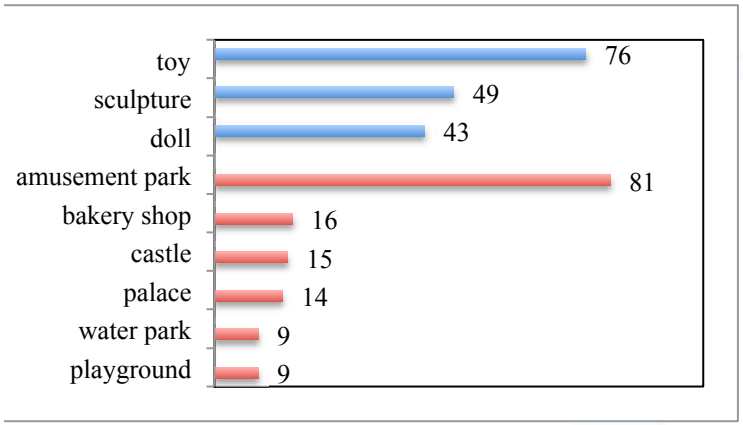
Beyonce



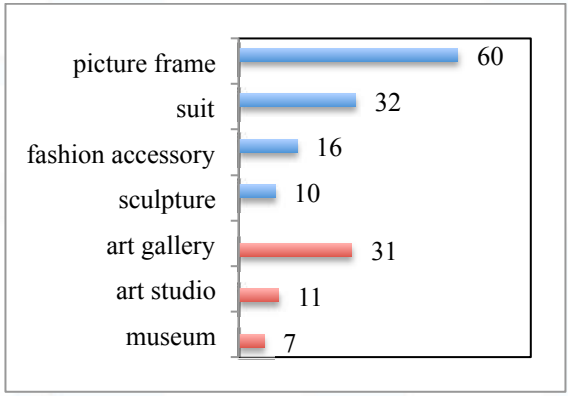
Gordon Ramsay



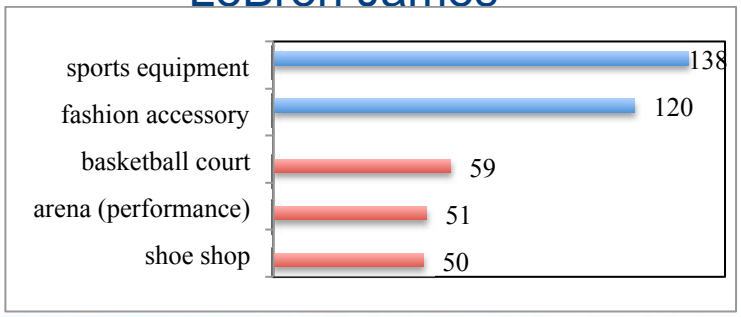
Disneyland



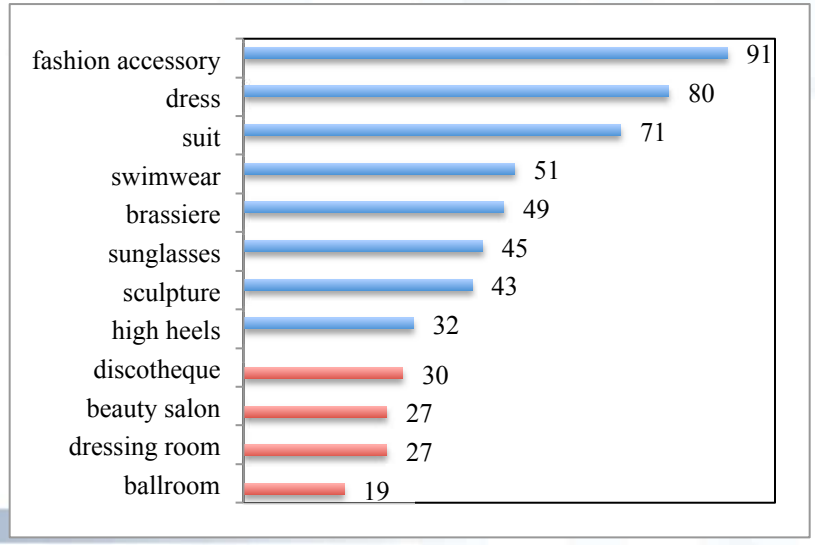
Kehinde Wiley



LeBron James



Kim Kardashian



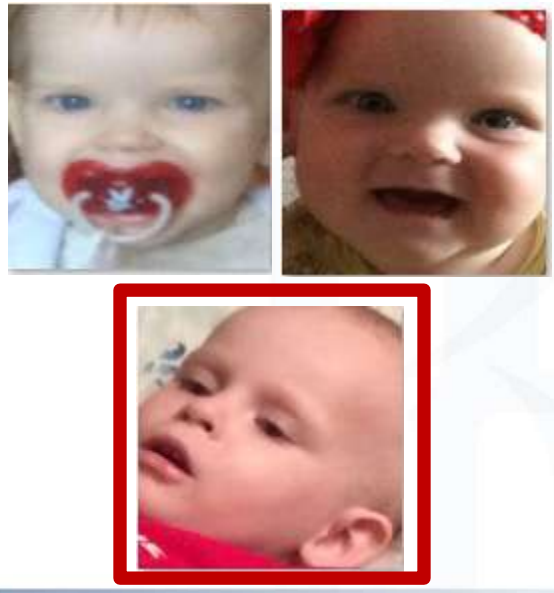
Результаты кластеризации лиц

male 1987

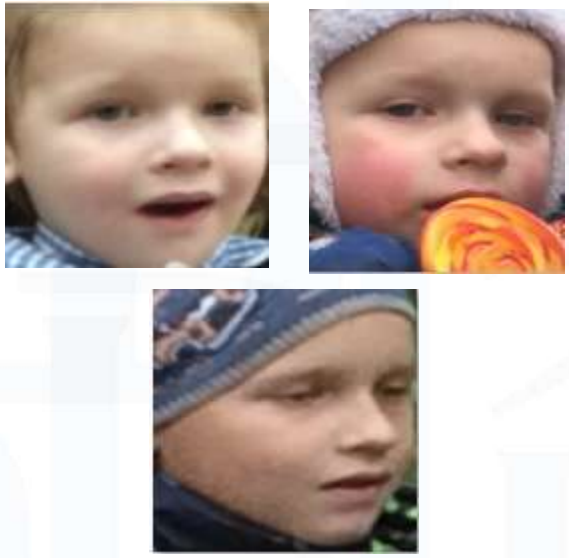
female 1982

female 2015

female 2016



male 2009



Показатели точности реализованных в мобильном приложении методов

- **Детектирование объектов** (78 классов)

mAP: 66.3%, recall: 75.1%

- **Распознавание сцен** (328 categories)

Top-5 accuracy: 88.4%,

Top-1 accuracy: 58.6%

- **Распознавание лиц**:

- Кластеризация. B-cubed precision: 76.2-98.7%, B-cubed recall: 91.7-99.3%
- Распознавание пола:

Top-1 accuracy: 95-98%

- Предсказание возраста:

MAE (mean absolute error): 5,5-6 лет

- **Время работы на мобильном устройстве** (Samsung S9+):
- Распознавание изображений (лиц/сцен): 15-40 мс
- Детектирование объектов: 150-250 мс

Заключение

1. Простые основанные на MobileNet/EfficientNet модели могут достигать лучших (state-of-the-art) результатов в задачах распознавания сцен, событий, пола, возраста, расы, кластеризации лиц
2. Методы ускорения предобученных нейронных сетей, основанные на досрочном останове на ранних слоях, ограниченны в точности
3. Методы повышения эффективности классификации нейросетевых признаков не всегда существенно снижают общее время принятия решений

NATIONAL RESEARCH
UNIVERSITY

Спасибо!

Работа поддержана грантом РНФ 20-71-10010