ИНФОРМАТИКА И УПРАВЛЕНИЕ В ТЕХНИЧЕСКИХ И СОЦИАЛЬНЫХ СИСТЕМАХ

УДК 004.932 ***DOI: 10.46960/1816-210X_2022_4_7***

МЕТОД ПОВТОРНОЙ ИДЕНТИФИКАЦИИ ОБЪЕКТОВ В МНОГОКАМЕРНЫХ СИСТЕМАХ С НИЗКИМ ЭНЕРГОПОТРЕБЛЕНИЕМ С ИСПОЛЬЗОВАНИЕМ ИЕРАРХИЧЕСКИХ НЕЙРОННЫХ СЕТЕЙ

М.Б. Багиров
ORCID: **0000-0003-1656-0849** e-mail: **bagirov_mirabbas@mail.ru**
Нижегородский государственный технический университет им. Р.Е. Алексеева
Нижний Новгород, Россия

Представлена новая модель иерархических нейронных сетей для решения задачи повторной идентификации объекта (reID). Предложена иерархическая архитектура DNN, использующая метки атрибутов в обучающем наборе данных с целью повышения эффективности поиска объектов. Применяется библиотека C++ глубинного обучения tiny-DNN, разработанная для условий ограниченных ресурсов. На каждом узле иерархии tiny-DNN идентифицирует разные атрибуты изображения, на каждом конечном узле проводится реидентификация подмножества датасета с изображениями, состоящими только из изображений с атрибутами, идентифицированными по конкретному пути от корня к листу. Изображение запроса реидентифицируется сразу после обработки с помощью нескольких tiny-DNN. Проведен анализ эффективности предложенного решения. Использованы два набора данных изображений: VRAI для повторной идентификации транспортных средств и Market-1501 для повторной идентификации человека. Для оценки точности разработанного метода используется метрика ранжирования (Rank-1) и средней точности (mAP). Предложенный метод повторной идентификации объектов имеет наименьшее время запроса и энергопотребления при реализации на устройствах Raspberry Pi 3 и NVIDIA Jetson Nano. В результате сравнения разработанного метода с существующими вариантами анализа объектов на видеопотоке и повторной идентификации объектов показано, что, несмотря на потерю точности в среднем 4 %, разработанный подход обеспечивает значительную экономию ресурсов: в среднем требует на 74 % меньше памяти, на 72 % снижается объем производимых операций, на 67 % сокращено время задержки обработки запросов, что в целом приводит к снижению энергопотребления на 65%.

Ключевые слова: низкое энергопотребление, реидентификация, глубокие нейронные сети, иерархическая структура, компьютерное зрение.

ДЛЯ ЦИТИРОВАНИЯ: Багиров, М.Б. Метод повторной идентификации объектов в многокамерных системах с низким энергопотреблением с использованием иерархических нейронных сетей // Труды НГТУ им. Р.Е. Алексеева. 2022. № 4. С. 7-19. DOI: 10.46960/1816-210X_2022_4_7

A METHOD FOR LOW-POWER MULTI-CAMERA OBJECT RE-IDENTIFICATION USING HIERARCHICAL NEURAL NETWORKS

M.B. Bagirov
ORCID: **0000-0003-1656-0849** e-mail: **bagirov_mirabbas@mail.ru**
Nizhny Novgorod State Technical University n.a. R.E. Alekseev
Nizhny Novgorod, Russia

Abstract. The novel model of hierarchical neural networks is provided to address the problem of object re-identification (reID). The proposed is DNN hierarchical architecture that uses attribute labels in the training data set to increase the efficiency of object search. A tiny-DNN deep learning C++ library designed for resource constrained environment is applied. In each hierarchy node, a tiny-DNN identifies different image attributes. In each bottom hierarchy

node, a subset of image dataset containing only images with attributes identified by the specific path from the root to a leaf is re-identified. The query image is re-identified immediately following processing with several tiny-DNNs. The efficiency of the proposed solution is analyzed. Two image datasets are used: VRAI for re-identification of vehicles and Market-1501 for person re-identification. To assess the precision of the developed method, ranking accuracy (Rank-1) and mean Average Precision (mAP) metrics are used. When implemented on Raspberry Pi 3 and NVIDIA Jetson Nano devices, the proposed method for object re-identification provides the shortest query time and the lowest power consumption. In comparison to existing methods of detecting objects in a video stream and of object re-identification, the developed method, although demonstrating average loss in precision by 4 %, provides significant resource savings. It enables, in average, less memory usage by 74%, less operating intensity by 72 %, shorter timeout for processing of queries by 67 %, which altogether lead to lower power consumption by 65 %.

***Key words**:* low power consumption, re-identification, deep neural networks, hierarchical structure, computer vision.

FOR CITATION: M.B. Bagirov. A method for low-power multi-camera object re-identification using hierarchical neural networks. Transactions of NNSTU n.a. R.E. Alekseev. 2022. № 4. Pp. 7-19.
DOI: 10.46960/1816-210X_2022_4_7

Введение

Повторная идентификация объекта (реидентификация) является важной задачей компьютерного зрения. Методы повторной реидентификации могут использоваться для повышения общественной безопасности [1], управления большим количеством скопления людей, обнаружения требуемых событий при наблюдении с помощью нескольких камер [2]. Системы реидентификации могут быть развернуты на уже встроенных устройствах (дорожные камеры и беспилотные летательные аппараты) [3]. Вычислительные ресурсы для них ограничены, а энергоэффективность реализованных технологий имеет решающее значение. Исследования показывают, что существующие методы идентификации объектов не подходят для встраиваемых устройств, ориентированных на тяжелые глубокие нейронные сети, требующие больших вычислительных ресурсов и памяти [4].

Алгоритм, реализуемый в существующих методах, показан на рис. 1. Big-DNN извлекает вектор объектов из исходного изображения, затем он сравнивается с векторами объектов каждого изображения из датасета (рис. 1, изображения а-г), с использованием метрики евклидового расстояния, изображения из датасета ранжируются в зависимости от их расстояния до объекта поиска. При этом выполняется множество избыточных операций, поскольку объект поиска сравнивается с каждым изображением из датасета. Но изображение запроса (белый автомобиль с люком на крыше) достаточно сравнить только с другими белыми автомобилями с люками на крыше, что соответствует изображениям из датасета б и г (рис. 1).

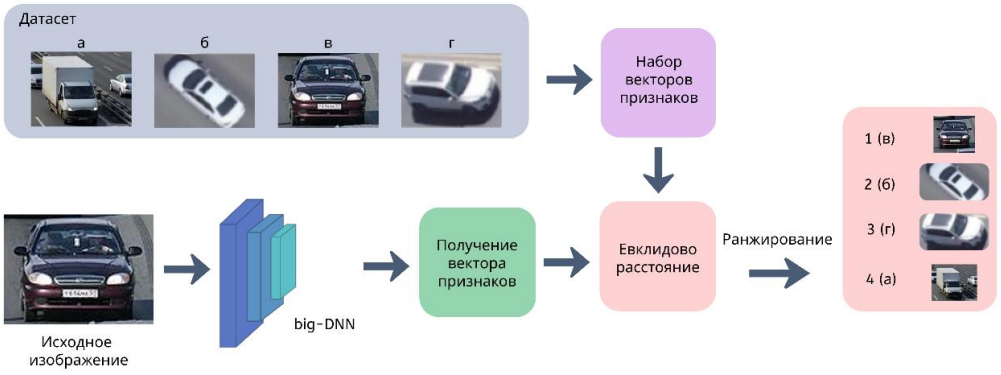

Рис. 1. Алгоритм сравнения набора характеристик исходного изображения с извлеченными векторами объектов для каждого изображения из датасета, требующий больших вычислительных затрат

Fig. 1. The algorithm for comparison of source image profile with extracted object vectors for every image from dataset requiring high computational efforts

В работе предлагается использовать иерархическую модель DNN для идентификации объектов с одинаковыми атрибутами (цвет, кузов, наличие люка на крыше и т.д.) для повышения точности и уменьшения количества операций за счет устранения избыточных. Иерархическая DNN-архитектура использует несколько DNN в форме иерархии для сокращения избыточных операций в компьютерном зрении. При этом возникают проблемы, связанные с их использованием для эффективного распознавания объектов. Для их решения в данной работе предлагается использовать иерархическую архитектуру DNN с модификацией метрик подобия и метода определения иерархической структуры.

В рассматриваемой архитектуре DNN каждый узел иерархии содержит tiny-DNN (модель нейронной сети, извлекающая вектор признаков из исходного изображения и передающая его последующим ветвям) (рис. 2). Исходное изображение обрабатывается первой tiny-DNN для получения вектора объекта, которая определяет, есть ли в крыше автомобиля люк. После того, как первой DNN изображение запроса классифицируется как транспортное средство с люком на крыше, датасет уменьшается до изображений б, в и г. Следующая DNN в иерархии продолжает обрабатывать вектор признаков и идентифицирует цвет транспортного средства. Эта классификация сводит множество изображений к двум (рис.1, б и г). Этот процесс продолжается до тех пор, пока не будет достигнут конечный узел. Вектор объектов из конечной DNN используется для выполнения сравнений с остальными изображениями из датасета для повторной идентификации объекта, которые содержат объекты с атрибутами, указанными в запросе. Поскольку каждая tiny-DNN специализируется на обработке только подмножества объектов (с определенными атрибутами), достигается высокая точность распознавания объектов.

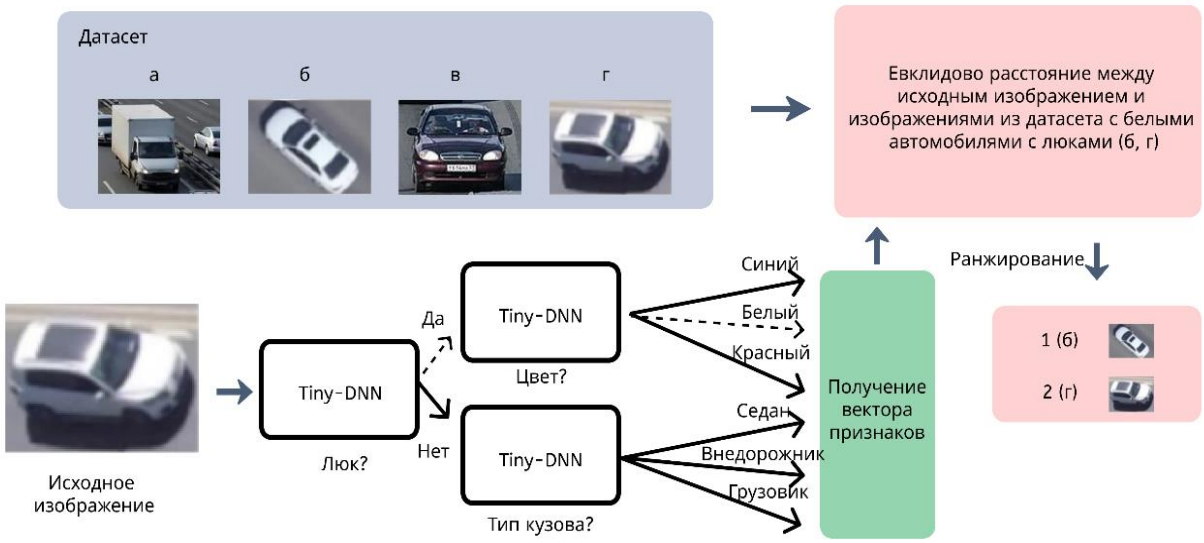

Рис. 2. Иллюстрация предлагаемого метода

Fig. 2. Illustration of the proposed method

В работе приводятся результаты экспериментов по реидентификации двух типов объектов: транспорта и людей, показывающие значительное снижение требований к потребляемым ресурсам: памяти – на 74-97 %, времени задержки запросов – на 67-89 %, количества операций – на 72-93 % и энергопотребления – на 65-88 %. Эксперименты проводились на двух встроенных устройствах: RaspberryPi 3 и NVIDIA JetsonNano (результаты измерений, на основе которых сделаны данные выводы, и их подробное описание приведены в разделе 4).

2. Анализ работ в области применения иерархических глубоких нейронных сетей

Иерархические DNN используют несколько DNN, организованных в древовидную структуру [7], где входной сигнал зависит от пути, по которому он следует от корня к листу дерева (рис. 2). В последнее время эти методы используются для решения задачи классификации на каждом уровне дерева, чтобы уменьшить размерность задачи и повысить эффективность работы алгоритмов [8]. Иерархические архитектуры DNN строятся с использованием визуальных [9] или семантических сходств [10]. Однако эти методы неприменимы к реидентификации объектов. Иерархии, основанные на визуальном сходстве, необходимо переобучать каждый раз, когда встречается новый, ранее не распознанный объект. Между тем иерархии, основанные на семантическом сходстве, несут значительные потери в точности распознавания при их применении для решения задач компьютерного зрения. Для устранения этих недостатков предложено использовать комбинацию визуальных и семантических сходств, при этом показано, что такая комбинация хорошо подходит для эффективного поиска объектов.

При повторной идентификации объекта задача состоит в том, чтобы определить, существует ли объект в базе изображений объектов, которые появлялись ранее. Известны два популярных метода выполнения повторной идентификации объекта:

1) глобальные векторы: используются DNN для получения общего вектора объекта для каждого изображения [11];

2) локальные векторы признаков: объединяются несколько меньших векторов объектов (например, различные части кузова/транспортного средства), чтобы сформировать единый общий вектор объектов для каждого изображения.

В дополнение к векторам признаков некоторые методы используют вспомогательную информацию (например, атрибуты или синтетические изображения) для повышения точности повторной идентификации. Методы, использующие вспомогательные векторы объектов, часто требуют наборов данных, аннотированных метками атрибутов. Можно отметить, что квантование DNN уменьшает потребность в памяти, а сокращение DNN с помощью обрезки весов позволяет сократить количество операций. Применение такого подхода повышает эффективность существующих методов, но при этом, как правило, снижают точность распознавания объектов. В настоящее время не существует решений по использованию иерархических DNN для снижения энергопотребления при повторном распознавании объектов на встроенных устройствах.

Разработанный метод, применяющий глобальные функции и вспомогательную информацию для построения иерархии tiny-DNN для повторной идентификации объектов на встроенных устройствах с низким энергопотреблением, использует как семантические, так и визуальные сходства для построения иерархий для повторной идентификации объектов. Он обеспечивает высокую точность при низких требованиях к ресурсам за счет использования tiny-DNN, которые специализируются на обработке небольшого подмножества предложенных изображений из датасета. Проведенные эксперименты показывают, что разработанный метод эффективнее существующих методов с точки зрения требуемой памяти, количества операций и энергопотребления.

3. Описание модели нейронной сети для реидентификации объекта

Предлагаемый метод использует несколько tiny-DNN, которые извлекают векторы объектов из изображения-запроса и идентифицируют атрибуты, чтобы свести первоначальный датасет к новому меньшему подмножеству изображений. Чтобы построить иерархию сети, необходимо определить, какие атрибуты идентифицируются, и порядок, в котором они идентифицируются.

На рис. 3 показаны примеры изображений из набора данных Market-1501 с некоторыми их атрибутами. Идентификация всех атрибутов на изображениях не требуется.

Рис. 3. Примеры изображений в наборе данных Market-1501 с некоторыми их атрибутами

Fig. 3. Examples of images in the Market-1501 dataset with some of their attributes

Например, если большинство людей в базе данных, которые носят рюкзак, в то же время не носят сумку через плечо, идентификация того, носит ли человек сумку через плечо, скорее всего, будет излишней, если будет идентифицировано ношение рюкзака. Предложенный метод использует корреляции, чтобы определить, какие из атрибутов действительно следует идентифицировать. Под корреляцией понимается вероятность нахождения двух атрибутов на одном и том же объекте. Сильно коррелированные атрибуты не обязательно идентифицировать в одном и том же пути в иерархии, поскольку ожидаемое уменьшение мощности множества датасета будет невелико.

На рис. 4 а) и б) показаны векторы признаков, полученные с использованием предварительно обученной big-DNN для изображений в наборе данных Market-1501. Контуры добавляются к кластерам, чтобы подчеркнуть разную сложность идентификации атрибутов. Атрибуты с более четкими кластерами (например, пол) легче идентифицировать, чем другие (например, ношение сумки). В наборе данных Market-1501 мужчины и женщины визуально не похожи, но людей с сумками и без них трудно отличить. Следовательно, гендерная классификация должна выполняться ближе к корню иерархии. Это обусловлено тем, что tiny-DNN могут идентифицировать визуально непохожие атрибуты наиболее точно. Кроме того, сложные классификации объектов становятся ближе к верхним уровням иерархии, поскольку классификации выполняются уже на подмножествах изображений их датасета. Рис. 4 иллюстрирует сложность идентификации наличия сумок, когда датасет: б) – содержит все изображения, в) – только изображения женщин и г) – только изображения женщин в платьях. Сложность распознавания наличия сумок снижается, когда задача решается на меньшем количестве изображений в датасете. Чтобы гарантировать, что tiny-DNN могут точно выполнять повторную идентификацию объекта, необходимо определить порядок, в котором идентифицируются атрибуты.

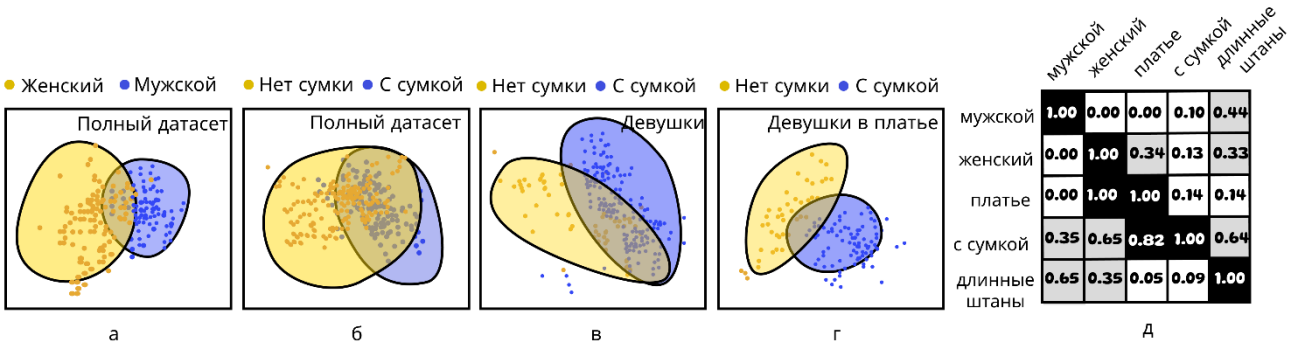

Рис. 4. Векторы признаков, полученные с использованием big-DNN для изображений в наборе данных Market-1501:

а), б) векторы признаков (после анализа основных компонентов) изображений из всего обучающего датасета Market-1501 распределены по а) полу, б) ношению сумки; в), г) векторы признаков подмножеств датасета Market-1501 распределены на: в) только женщины, г) только женщины в платьях; д) – выдержка из корреляционной матрицы для Market-1501, показывающая вероятность того, что у человека есть атрибут j, если у него есть атрибут i

Fig. 4. Feature vectors extracted using big-DNN for Market-1501 dataset images:

a), b) feature vectors (after principal component analysis) of images from the entire Market-1501 training dataset are categorized by a) gender, b) bag carrying; c), d) feature vectors of Market-1501 dataset subsets are categorized into: c) only women, d) only women wearing a dress; e) an extract from a cross-correlation matrix for Market-1501 showing than the person is likely to have attribute «j» if they have attribute «i».

Определим количественные оценки сложности идентификации и корреляции атрибутов. Чтобы определить, какой атрибут идентифицирован в корне, а также порядок последующих классификаторов атрибутов в иерархии, необходимо количественно оценить сложность идентификации. Более простую идентификацию атрибутов будем выполнять ближе к корню иерархии. Такой порядок целесообразен, поскольку иерархические DNN распространяют векторы выходных объектов от родительского элемента к дочернему. Каждая ветвь иерархической DNN представляет более глубокую DNN (имеющую большее количество уровней).

Чтобы измерить сложность классификации атрибутов, будем использовать линейную процедуру оценки *estimation procedure*. Линейный классификатор обучается на векторах признаков предварительно обученной DNN, при этом ошибка валидации используется для измерения сложности классификации. Атрибуты с большими ошибками валидации труднее классифицировать. Сортировка атрибутов по ошибке валидации позволяет создавать ранжированный список атрибутов на основе их сложности. Для набора данных Market-1501 пол является атрибутом с наивысшим рангом с линейной ошибкой классификации, равной 0,08, а наличие сумки – атрибут низшего ранга с линейной ошибкой классификации, равной 0,23. Атрибут с наивысшим рангом определяется в корне дерева. Последующие идентификации атрибутов для каждой ветви дерева определяются с использованием комбинации сложности идентификации и корреляций атрибутов.

Таким образом, первая классификация атрибутов (т.е. выбора атрибута с наивысшим рангом) иерархии определяется после количественной оценки сложности идентификации атрибутов. Затем для каждой ветви дерева необходимо определить, какие атрибуты классифицировать. Для этого выполняется рекурсивное получение корреляций между различными атрибутами. Сильно положительно и сильно отрицательно коррелированные атрибуты не идентифицируются в одной и той же ветви. Корреляционная матрица C(i, j), фрагмент которой приведен на рис. 4 д), показывает корреляцию P(j | i) между парами атрибутов – вероятность того, что у человека есть атрибут j, если у того же человека также есть атрибут i (1):

$$C(i, j) = P(i) = \frac{P(i, j)}{P(i)}, \quad (1)$$

Каждая запись матрицы C(i, j) получается путем деления количества изображений обучающего набора данных, содержащих атрибуты i и j, на количество изображений обучающего набора данных, содержащих атрибут i. Большое значение элемента матрицы C(i,j) указывает на высокую положительную корреляцию между атрибутами i и j (например, C(платье, женщина) = 1,00). Малое значение C(i, j) указывает на то, что i и j имеют высокую отрицательную корреляцию (например, C(платье, мужчина) = 0,00). Корреляционная матрица вычисляется в каждом узле дерева для каждого атрибута k относительно набора атрибутов от i до j, которые уже были идентифицированы по пути от корня к узлу: C(i...j, k) = P(k| i...j).

Значение корреляционной матрицы можно пояснить на примере. Следуя гендерной классификации, определение того, одет ли человек в платье, полезно только в том случае, если человек женского пола (C(женщина, платье) = 0,34). Идентификационный адрес не полезен для мужчин (C(мужчина, платье) = 0,00). Выполнение классификации одежды для мужчин не помогает уменьшить размер выборки из датасета, это избыточная операция. Идентификация атрибутов, которые имеют низкую корреляцию с ранее определенными атрибутами, уменьшает количество избыточных операций. Если имеется несколько неидентифицированных атрибутов с низкой корреляцией, атрибут с наивысшим рангом (самый простой для идентификации) выбирается в качестве следующей классификации атрибутов. Если нет неидентифицированных атрибутов с низкой корреляцией, классификация атрибутов больше не выполняется (конечный узел). Чтобы реализовать это, рассматриваем атрибуты с $C(i, j) \in [0{,}3, 0{,}7]$ как слабо коррелированные.

Предлагаемый метод повышает эффективность для общего случая, определяя только те атрибуты, которые слабо коррелируют с ранее идентифицированными атрибутами. Рассмотрим пример: в Market-1501 большинство мужчин не носят сумок (C (мужчина, сумка) = 0,10). Для повторной идентификации всех мужчин (с сумками или без них) определяются другие атрибуты (например, возраст), чтобы уменьшить размер выборки из датасета. Таким образом, для общего случая, т.е. мужчин без сумок, выполняется меньше избыточных операций. Как только структура иерархии определена, DNN строятся для каждого узла иерархии. При этом каждая DNN выполняет две задачи: извлекает векторы признаков и идентифицирует атрибуты. Каждый узел иерархии специализируется на обработке и повторной идентификации изображений из датасета. Архитектуры DNN необходимо выбирать таким образом, чтобы каждая DNN могла выполнять задачи точно и эффективно. Архитектуры big-DNN дают большую точность классификации, используя при этом больше ресурсов. Чтобы получить приемлемый компромисс между точностью и эффективностью модели, применим метод поиска архитектуры нейронной сети, где в качестве метрики используется изменение плотности точности для оценки выбора DNN (D_i+1) с i+1 слоями вместо DNN (D_i) с i слоями. Метрика изменения плотности точности определяется следующим образом:

$$\Delta D(D_i, D_{i+1}) = \frac{a_{i+1} - a_i}{m_{i+1} - m_i}, \quad (2)$$

где a_i и m_i – точность и требования к памяти для Di соответственно. Этот метод увеличивает размер DNN по одному слою за одну итерацию и вычисляет ΔD(Di, Di+1) до тех пор, пока не достигнет точки убывающей отдачи от точности ReID и точности классификации атрибутов с более крупными моделями. В работе в качестве a_i выбираем среднее значение средней точности ReID и точности классификации атрибутов.

Каждая модель HC tiny-DNN следует структуре плотного блока из DenseNet [12] из-за ее способности извлекать векторы информативных признаков. Корень иерархии использует ядро свертки 7×7×64 с шагом 2 и слой maxpool 2×2 для понижения дискретизации изображения. Каждый последующий слой является «плотным слоем», т.е. слоем, содержащим последовательность операций 1×1×128 и 3×3×32. Количество последующих плотных слоев опре-

деляется методом поиска нейронной архитектуры (NAS) [13]. Векторное усреднение используется для изменения размера тензоров перед извлечением вектора признаков 128×1 и выходных данных классификации. Карта активации перед слоем среднего пула используется в качестве входных данных для выбранной дочерней DNN. Каждая дочерняя DNN следует одной и той же структуре плотных слоев. В табл. 1 приведены примеры DNN, полученные данным методом.

Для обучения DNN иерархии используется метод обратного распространения ошибки. При этом будем использовать две функции потерь: тройную потерю с пакетным жестким майнингом для обучения DNN для извлечения вектора признаков таким образом, чтобы векторы признаков одного и того же объекта были похожи; кросс-энтропийную потерю – для выполнения классификации атрибутов.

Таблица 1.

DNN, построенные для набора данных Market-1501, обеспечивающие компромисс между точностью и эффективностью. Корневая DNN содержит 3 плотных слоя

Table 1.

DNNs built from Market-1501 dataset enabling precision and efficiency trade-off. The root DNN contains 3 dense layers

Root (пол)	Child 1 (длина штанов)	Child 2 (возраст)
conv7×7×64 maxpool2×2	maxpool2×2	maxpool2 ×2
conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32
conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32
conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32	conv1×1×128 conv3×3×32
avgpool	avgpool	conv1×1×128 conv3×3×32
Output2: 4×1 Output1: 128×1	Output2: 4×1 Output1: 128×1	avgpool
		Output2: 4×1 Output1: 128×1

Обе метрики оценки потерь используются отдельно, потому что они требуют разных конфигураций обучающей партии. Сначала DNN обучается с потерей триплетов для извлечения вектора признаков. Затем параметры извлечения вектора признаков замораживаются (не обновляются с помощью обратного распространения), а последующие слои классификации обучаются с помощью функции кросс-энтропийных потерь. DNN в иерархии обучаются в корневом порядке: сначала обучается корневой DNN, затем его дочерние элементы и так далее. Процесс реализация метода реидентификации объекта заключается в следующем. Изображения из датасета сначала обрабатываются обученной иерархической DNN. Атрибуты назначаются изображениям из датасета DNN, если атрибуты недоступны. Как видно на рис. 2, новое изображение запроса обрабатывается корневой DNN для извлечения вектора признаков и выполнения классификации атрибутов для выбора следующей DNN, затем вектор признаков обрабатывается выбранной DNN. Этот процесс продолжается до тех пор, пока не будет достигнут конец иерархии. В проведенных экспериментах в качестве метрики расстояния для сопоставления используется евклидово расстояние, при это измеряем расстояние только между изображением запроса и изображениями из датасета, которые содержат обнаруженные атрибуты, чтобы повторно идентифицировать объект. Часть полученной иерархии для набора данных Market-1501 изображена на рис. 5. Каждое изображение проходит один путь от корня дерева к листу. Разные пути определяют разные атрибуты.

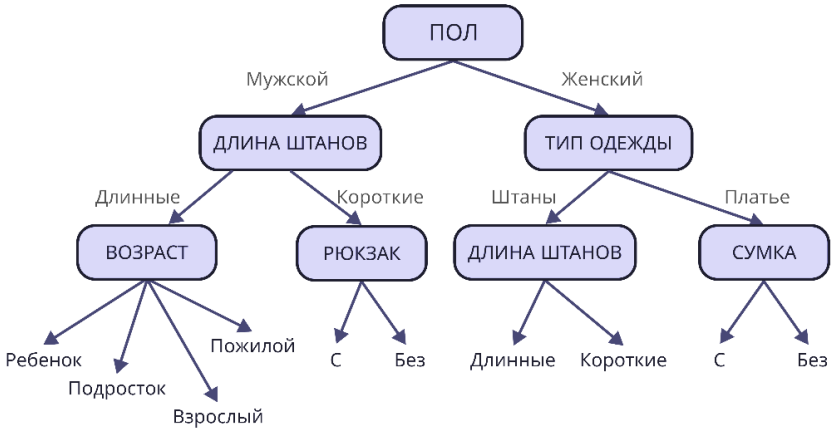

Рис. 5. Три уровня полученной иерархии для выполнения повторной идентификации человека из набора данных Market-1501

Fig. 5. Three levels of the resulted hierarchy to perform person re-identification in the Market-1501 dataset

Разные пути могут идентифицировать одни и те же атрибуты (например, «длина брюк»). Хотя соответствующие DNN концептуально перекрываются, они специализируются на обработке и идентификации атрибутов для различных подмножеств датасета и не могут использоваться взаимозаменяемо. Важно отметить, что для изображения запроса активируется только одна ветвь, поэтому избыточные вычисления не выполняются, даже если одни и те же атрибуты появляются в разных ветвях.

Отметим, что табл. 1 иллюстрирует архитектуру нейросети, а рис. 5 определяет метод реидентификации.

4. Результаты апробации разработанного метода реидентификации объекта

При проведении экспериментов использованы два набора данных изображений: VRAI для повторной идентификации транспортных средств и Market-1501 для повторной идентификации человека. VRAI содержит 66 113 изображений 6 302 различных транспортных средств. Market-1501 содержит 32 668 изображений, принадлежащих 1 501 разным пользователям. Этот набор данных также содержит 50 000 изображений для тестирования. Оба набора данных разделены на наборы для обучения и тестирования и аннотированы атрибутами.

Для оценки точности разработанного метода используем метрику ранжирования (Rank-1) и средней точности (mAP). Rank-1 – вероятность того, что правильное изображение появится как совпадение с наивысшим рангом, mAP измеряет среднюю производительность поиска при наличии нескольких совпадений. Оценки точности распознавания объектов определяются по данным тестирования. Требования к памяти и количество выполняемых операций (FLOP) для DNN содержатся в библиотеках torchsummary и thopPyTorch соответственно. Для разработанного метода определяются требования к памяти и количество операций (FLOP) в наихудшем случае: отношение суммы размеров моделей к FLOP DNN вдоль самого длинного пути от корня до листа. Измеритель мощности Yokogawa WT310E измеряет энергопотребление технологий на устройствах RaspberryPi 3 и NVIDIA JetsonNano. Структура иерархической DNN определяется с помощью протокола линейной оценки и корреляционной матрицы, затем создаются и обучаются DNN для каждого узла. Для обеспечения доступности обучающих данных длина ветви не увеличивается, если дочерняя DNN имеет менее 300 обучающих изображений. Самая глубокая ветвь иерархической DNN – четыре узла для набора данных VRAI и пять узлов для набора данных Market-1501. Во время обучения используем реализацию PyTorch по умолчанию с использованием триплетных потерь с пакетным жестким майнингом. Здесь пакеты формируются путем случайной выборки P объек-

тов, а затем случайной выборки K изображений каждого объекта, в результате чего получается пакет из P*K изображений. В экспериментах используем P = 8 и K = 4 (наибольший размер пакета, который помещается в доступную память GPU). Тренировка идет с тройной потерей до тех пор, пока потеря не достигнет постоянного значения. Скорость обучения начинается с 0,01 и уменьшается в 10 раз каждые 100 эпох. При обучении DNN с кросс-энтропийной потерей обучаем 100 эпох с размером партии, равной 32 и скоростью обучения, равной 0,001 с.

Для сравнения эффективности разработанного метода и существующих подходов к выполнению ReID для набора данных VRAI с целью повторной идентификации транспортных средств используются RAM-VGG (region-awaredeepmodel) [14], MultiTask [15] и DenseNet201 [16], для набора данных ReID с целью повторной идентификации человека сравниваем с пирамидальным ReID [17], DeepAnytimeRe-ID (DARE), Auto-ReID и Part-basedConvolutionalBaseline (PCB).

Таблица 2.

Сравнение размера модели, количества операций (FLOP), Rank-1 и mAP. Зеленый шрифт указывает наилучший результат

Table 2.

The comparison between model size, number of operations (FLOP), Rank-1 and mAP. The green font indicates the best result

Используемый датасет	Методы	Размер модели	FLOPs	Качество ранжирования	mAP
VRAI	RAM-VGG	528	15,483 M	0,720	0,573
	MultiTask	103	3,882 M	0,685	0,693
	MultiTask + DP	351	11,172 M	0,803	0,786
	DenseNet201	77	1,082 M	0,671	0,700
	RandomTree	25	1,082 M	0,631	0,585
	Разработанный метод	14	1,082 M	0,781	0,737
Market 1501	Pyramidal	184	9,757 M	0,928	0,821
	DARE	89	2,891 M	0,868	0,693
	Auto-reID	55	2,050 M	0,938	0,834
	DG-Net	101	4,029 M	0,896	0,745
	ResNet50	103	3,882 M	0,872	0,685
	DenseNet201	77	4,000 M	0,860	0,699
	PCB	107	4,206 M	0,923	0,774
	RandomTree	27	1,736 M	0,788	0,535
	Разработанный метод	14	808 M	0,885	0,699

В табл. 2 приведены полученные результаты: размер модели, количество операций (FLOP), значение метрики качества ранжирования (Rank-1) и значение mAP для различных методов. Разработанный метод ReID (реидентификации объекта) требует наименьшего объема памяти. По сравнению с RAM-VGG, в наборе данных VRAI предложенный метод требует модель гораздо меньшего размера (на 97,3 % меньше (1−14/528 = 0,973)), снижается количество выполняемых операций (FLOP) на 80,7 % (1−808/4206 = 0,807) по сравнению с методом PCB на наборе данных Market-1501. ResNet50, DenseNet201 и DARE обеспечивают более низкую точность, чем предложенный метод ReID. Отметим, что среднюю точность распознавания (mAP) с помощью разработанного метода можно улучшить, используя алгоритм случайного выбора, оптимизацию ранжирования и повторное ранжирование изображений. Предполагаются дальнейшее изучение сферы оптимизации результатов.

Проведенные исследования с использованием дерева случайных решений показывают, что использование иерархических DNN может снизить потребление ресурсов при повторной идентификации объекта. Очевидно, что случайный выбор иерархии менее эффективен, чем подход с использованием интеллектуальной иерархии. Реализация RandomTree требует больше ресурсов в связи с большим (неоптимальным) размером DNN, используемых для выполнения сложных классификаций атрибутов близко к корню дерева. Выявлены избыточные атрибуты, и очевидно, что для уменьшения размера выборки датасета требуются более глубокие деревья. Средняя глубина дерева RandomTree составляет 6 для VRAI и 7 для Market-1501. Поскольку ошибка классификации увеличивается на каждом этапе иерархии, более высокие деревья обеспечивают более низкую точность повторной идентификации.

Таблица 3.

Сравнение времени запроса (сек/изображение) и энергопотребления (Дж/изображение) на двух встроенных устройствах: RaspberryPi 3 и NVIDIA JetsonNano. Зеленый шрифт указывает на лучший результат

Table 3.

The comparison between query time (sec/image) and power consumption (J/image) on two built-in devices: RaspberryPi 3 and NVIDIA JetsonNano. The green font indicates the best result

Используемый датасет	Методы	RaspberryPi 3		NVIDIA JetsonNano	
		Время запроса	Энерго-потребление	Время запроса	Энерго-потребление
VRAI	ResNet50	_	_	3,20	22,24
	DenseNet201	_	_	2,75	19,26
	RandomTree	4,85	21,99	0.78	5,68
	Разработанный метод	2,53	12,13	0,305	2,66
Market 1501	ResNet50	_	_	3,00	21,63
	DenseNet201	_	_	2,55	18,18
	DARE	11,41	55,83	1,09	7,92

В табл. 3 приведены значения времени запроса и энергопотребление методов. Результаты получены после усреднения значений обработки более 100 изображений, в которых используется самый длинный путь от корня к листу. Предложенный метод имеет наименьшее временя запроса и энергопотребления при реализации на обоих устройствах. Методы, вызывающие ошибки памяти во встроенных устройствах, обозначены знаком «-» или исключены из табл. 3. На рис. 6 показаны примеры реидентификации (ReID): два изображения запроса и пять изображений из датасета, которые возвращаются как совпадения. Разработанный метод дает хорошие результаты повторной реидентификации объектов (возвращает правильные совпадения с изображением запроса).

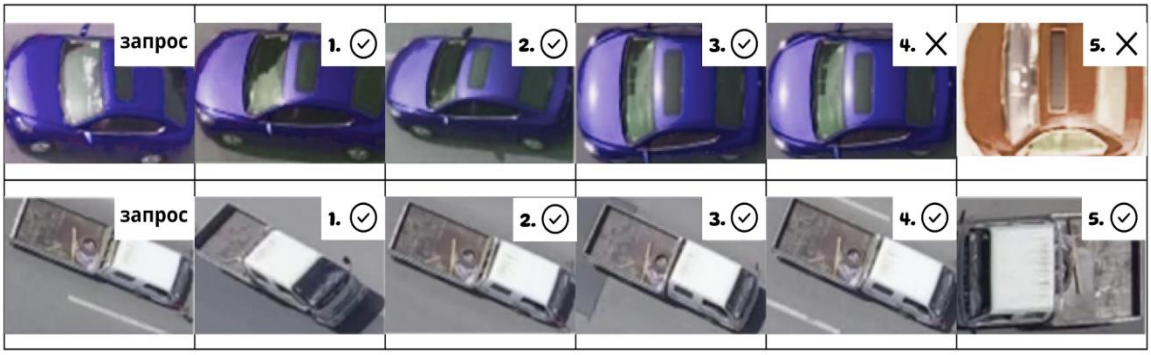

Рис. 6. Изображения, возвращенные как совпадения с использованием иерархического DNN для двух запросов. Правильные совпадения отмечены галочкой

Fig. 6. Images returned as matches using hierarchical DNN for two queries. Correct matches are ticked

5. Заключение

Представлена новая иерархическая DNN для энергоэффективной повторной идентификации объектов (ReID) на встроенных устройствах. Используется несколько нейросетей tiny-DNN, организованных в виде иерархии, где каждая сеть обрабатывает входные данные и идентифицирует атрибут. Предложен подход, учитывающий как визуальное, так и семантическое сходства для построения иерархий для эффективного повторного идентификатора объекта. Это достигается путем количественной оценки сложности идентификации атрибутов и нахождения корреляции между атрибутами, чтобы определить, какие атрибуты идентифицируются и в каком порядке. Область поиска сужается каждый раз, когда идентифицируется атрибут. При этом иерархия tiny-DNN модернизируется для повторной идентификации только небольшого подмножества объектов для повышения точности решений при низких требованиях к ресурсам. Эксперименты подтверждают, что иерархическая модель DNN позволяет развертывать метод ReID объекта на двух встраиваемых устройствах начального уровня на примере RaspberryPi 3 и NVIDIA JetsonNanoза за счет системного подхода к построению иерархии и выбора размера DNN.

Библиографический список

1. **Кучеров**, **С.А.** О подходах к распознаванию и идентификации персоны по цифровым изображениям в задачах обеспечения общественной безопасности / С.А. Кучеров, А.Н. Самойлов, А.К. Маакот, М.А. Кучерова // Политематический сетевой электронный научный журнал Кубанского государственного аграрного университета. 2016. С. 630-639.
2. **Гордин**, **М.С.** Алгоритмы обнаружения тревожных событий для систем автоматизированного видеонаблюдения / М.С. Гордин, С.А. Иванов // Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2017. Т. 15. № 3. С. 21-30.
3. **Майоров**, **Н.Н.** Автоматизация процесса идентификации объектов при выполнении автономных полетных заданий беспилотной авиационной системой / Н.Н. Майоров, А.С. Костин // Известия Тульского государственного университета. Технические науки. 2022. № 2. С. 640-646.
4. **Захарова**, **М.В.** Исследование алгоритмов технического зрения для систем пространственного слежения в типовых режимах их функционирования. / М.В. Захарова, Г.Г. Шмигельский., В.В. Григорьев // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 3. С. 487-492.
5. **Маковецкая-Абрамова**, **О.В.** Алгоритм идентификации транспортных средств в различных дорожных условиях / О.В. Маковецкая-Абрамова, Г.А. Петров // Технико-технологические проблемы сервиса. 2012. № 3(21). С. 5-7.

6. **Барашко, Е.Н.** Современные решения идентификации человека. Распознавание лиц / Е.Н. Барашко, С.О. Мазуренко, А.А. Шадрин // The Scientific Heritage. 2019.№ 42-1(42). С. 40-42.
7. **Зарубин, Г.А.** Моделирование иерархической нейросетевой архитектуры для решения задач классификации данных // Интеллектуальный потенциал XXI века: ступени познания. 2011. С. 175-178.
8. **Энгель, Е.А.** Использование иерархических нейронных сетей для распознавания многоэлементных зрительных сцен / Е.А. Энгель, О.И. Завьялова // Сибирский аэрокосмический журнал. 2009. № 3(24). С. 39-43.
9. **Кочурко, В.А.** Обнаружение объектов системами компьютерного зрения: подход на основе визуальной салиентности. / В.А. Кочурко, К. Мадани, К. Сабуран, В.А. Головко. П.А. Кочурко. // Доклады Белорусского государственного университета информатики и радиоэлектроники. 2015. № 5(91). С. 47-53.
10. **Зотов, С.С.** Обнаружение объектов в реальном времени с помощью алгоритмов распознавания YOLO / С.С. Зотов, А.А. Яковлев, Д.А. Колчинцев // Синергия наук. 2018. № 26. С. 388-404.
11. **Немцев, Н.С.** Подход для повторной идентификации модели транспортного средства по его изображению // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 14. № 4. С. 722-729.
12. **Романов, А.С.** Методика идентификации автора текста на основе аппарата опорных векторов // Доклады Томского государственного университета систем управления и радиоэлектроники. 2009. С. 36-42.
13. DenseNet − плотная сверточная сеть (классификация изображений), 25 ноября 2018 г. [Электронный ресурс] // Режим доступа: URL: https://machinelearningmastery.ru/review-densenet-image-classification-b6631a8ef803/
14. Поиск нейронной архитектуры (NAS) − будущее глубокого обучения, 8 июня 2019 г. [Электронный ресурс] // Режим доступа: URL: https://machinelearningmastery.ru/neural-architecture-search-nas-the-future-of-deep-learning-c99356351136/
15. **Liu, X.** RAM: a region-aware deep model for vehicle re-identification / Liu X., Zhang S., Huang Q., Gao W. // In Proceedings of the 2018 IEEE International Conference on Multimedia and Expo (ICME). 2018. Pp. 1-6.
16. **Peng Wang**. Vehicle Re-Identification in Aerial Imagery: Dataset and Approach / Peng Wang, Bingliang Jiao, Lu Yang, Yifei Yang, Shizhou Zhang, Wei Wei, Yanning Zhang // In 2019 IEEE/CVF International Conference on Computer Vision (ICCV).
17. Densenet-201 model from Densely Connected Convolutional Networks [Электронный ресурс] // Режим доступа: https://pytorch.org/vision/main/models/generated/torchvision.models.densenet201.html
18. **Zheng F.** Pyramidal Person Re-IDentification via Multi-Loss Dynamic Training / Feng Zheng, Cheng Deng, Xing Sun, Xinyang Jiang, Xiaowei Guo, Zongqiao Yu, Feiyue Huang, Rongrong Ji // In 2019 Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

Дата поступления
в редакцию: 30.06.2022

Дата принятия
к публикации: 04.10.2022