УДК 004.93'1

АВТОНОМНАЯ СИСТЕМА РАСПОЗНАВАНИЯ ЖЕСТОВ

Ерамкович В.А., студент

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Вашкевич М.И. – д-р. техн. наук, профессор

Аннотация. В работе представлено проектирование автономной системы распознавания жестов на базе несложного устройства. При проектировании использовались готовые решение, а также методы доработки их. Использовался набор данных FreiHAND, детектор на базе MediaPipe, архитектура сверточной нейросети MobileNetV2 и метод опорных векторов. В работе показаны изображения данных, формулы подготовки значений для обучения, объяснения особенностей семейства MobileNe, таблица с параметрами MobileNetV2 и информация об построении метода опорных векторов.

Ключевые слова. Сверточные нейронные сети, машинное обучение, компьютерное зрение.

Введение. В современном мире технологии «Умного дома» и Интернета вещей (IoT) становятся неотъемлемой частью повседневной жизни. Основным вектором развития интерфейсов взаимодействия человека с вычислительными системами является переход от контактных методов (клавиатура, мышь, сенсорные экраны) к бесконтактным, естественным интерфейсам. Управление с помощью жестов является одним из наиболее интуитивных способов взаимодействия, так как оно повторяет естественную моторику человека.

Однако существующие решения часто требуют дорогостоящего оборудования (сенсоры, перчатки с датчиками) или обладают высокой вычислительной сложностью. Разработка эффективного программного модуля для распознавания жестов, который не требует большого количества ресурсов является актуальной задачей в наше время. Поэтому целью работы является разработка автономной системы распознавания жестов на базе несложного устройства.

Данные для обучения. Для проектирования автономной системы распознавания жестов на базе несложного устройства использовался набор FreiHAND [1]. Данный набор данных состоит из двух выборок, тренировочной (130240 изображений) и валидационной (3960 изображений). Каждое изображение имеет размер 224x224.

Выборка для тренировки моделей представлена из изображений на зеленом фоне, количество таких изображений 32560 и данное количество является основной выборкой, остальные изображения получились путем аугментации основной. На рисунке 1 представлены изображения рук на зеленом фоне.

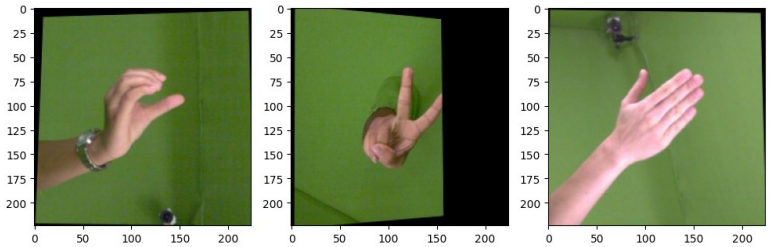


Рисунок 1 – Изображения рук на зеленом фоне и их черно-белое представление

Также в данном наборе есть данные для построения ключевых точек и скелета руки, что очень сильно помогает в реализации системы (рисунок 2)

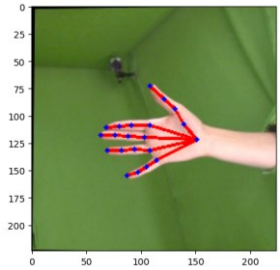


Рисунок 2 – Изображения рук на зеленом фоне и их черно-белое представление

Такие данные представлены в формате $[x, y, z]$, где x — значение по горизонтали, y — по вертикали и z отображает глубину (расстояние от камеры до руки), все эти изображения нормированы и принимают значения от 0 до 1 в метрах. Помимо этого, присутствуют данные, которые относятся к характеристикам камер, на которых происходили съемки всех изображений. Благодаря этому можно

сделать предобработку, чтобы привести значения к универсальному формату. Лучший формат — это привести все значения $[x, y, z]$, в $[u, v, z_{rel}]$ по формулам:

$$u = f_x \frac{x}{z} + c_x, \quad (1)$$

$$v = f_y \frac{y}{z} + c_y, \quad (2)$$

$$z_{rel}^i = z_{abs}^i - z_{abs}^0, \quad (3)$$

где f_x, f_y — фокусное расстояние объектива (в пикселях), c_x, c_y — координаты оптического центра (обычно центр картинки), z_{abs}^i — расстояние от камеры до сустава i, z_{abs}^0 — расстояние от камеры до запястья. Деление на z создает эффект перспективы: чем дальше объект (больше z), тем ближе его точки сходятся к центру и тем меньше он на экране.

Так как все изображения 224x224, то дополнительно проводится нормализация полученных u, v для отображения на изображении такого разрешения. Данная нормализация проводится по формулам:

$$u = \frac{u' - x_b}{S}, \quad (4)$$

$$v = \frac{v' - y_b}{S}, \quad (5)$$

где u', v' — полученные значения u, v по формулам 4.1 и 4.2, значения x_b, y_b — левый верхний угол рамки, найденной детектором (находит область руки), S — размер стороны квадрата детектора. Полученные значения находятся в диапазоне от 0 до 1.

Чтобы перенести такие значения на изображения нужно их привести к нужному разрешению по формулам:

$$x = uS + u_{min}, \quad (6)$$

$$y = vS + v_{min}, \quad (7)$$

где u_{min}, v_{min} — координаты левого верхнего угла этой рамки на исходном кадре.

Стоит отметить, что тренировочная выборка делилась для обучения и для тестирования, чтобы можно было проверять правильно ли, формируются ключевые точки и давать направление для улучшения.

В валидационной выборке расположены только изображения на разном фоне (рисунке 3). Также есть данные о характеристиках камер для приведения в нужный формат. Данные о ключевых точках для построения скелетов отсутствуют.

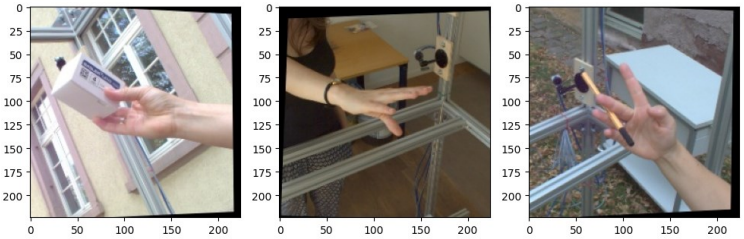


Рисунок 3 – Тестовая выборка изображений

Для обучения классификатора были отобраны 7 групп жестов. В таблице 1 представлена разметка классов и жестов для обучения. классификатора

Таблица 1 – Разметка классов и жестов

Номер класса жеста	Название жеста
0	Жест «Плохой» (большой + указательный палец)
1	Раскрытая ладонь
2	Жест «V» (указательный + средний палец)
3	Жест «Три» (большой + указательный + средний)
4	Большой палец вверх
5	Кулак
6	Указательный палец

Из всей тренировочной выборки были выбраны соответствующие жесты. На рисунке 4 показана диаграмма распределения по выбранным жестам, а также сколько пошло изображений на обучение и тестирование. Исходя из диаграммы мы видим, что присутствует неравномерное распределение по

классам. Из этого делаем вывод, что при обучении классификатора будет использоваться не метрика правильности, а F1-мера (macro). Используется такая метрика, так как она берет во внимание распределение по классам, а так как в полученном наборе есть неравномерное распределение, модель классификатора не будет отдавать предпочтение только тем классам, у которых больше всего значений, т.е. не будет переобучена.

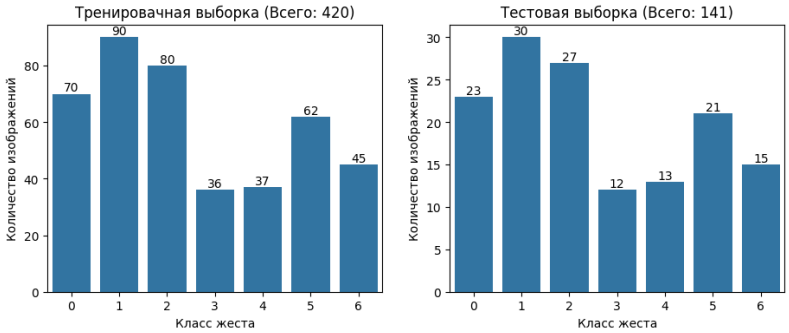


Рисунок 4 – Диаграммы распределения классов

Модели построения ключевых точек и классификатор. Детектирование области с рукой (ладонью) реализовывала уже готовая модель от MediaPipe, а именно MediaPipe Hands Detector [2]. Данная модель обучена определять область изображения с рукой и фиксировать ее. Зафиксированная область имеет разрешение 224x224.

Для формирования ключевых точек использовалась модель из семейства MobileNetV2 [3]. Есть два основополагающих блока в MobileNetV2 (MN блоки), где операции выполняются с одним шагом или с двумя. Блок с одним шагом представлен так:

1. Расширение исходного пространства сверткой с размером ядра 1x1 и активационной функцией ReLU6 (рисунок 6) [4]. Свертка выполняется для каждой карты признаков отдельно;
2. Глубинная свертка. Данная свертка делается ядром размером 3x3 и активационной функцией ReLU6, так же сворачивается каждая отдельная карта, а не все вместе;
3. Объединение карт признаков после глубинной свертки. Данный процессов происходит благодаря свертке ядром 1x1 с функцией ReLU6. При сверкте взаимодействуют все карты признаков вместе;
4. Выход (блок Add). На выходе складываются результаты, полученные с последнего блока (сворачивание пространства) и входные данные. Важное условие для отработки блока это соответствие размера карт и их количетсво. Формула для обработки блока выглядит так:

$$y = F(x) + x, \tag{8}$$

где $F(x)$ – результат прохождения данных через сверточные слои внутри блока, x — входные данные.

Блок с шагом два устроен точно также, но глубинная свертка выполняется с шагом два и ответствует блок Add.

На рисунке 5 представлены два MN блока.

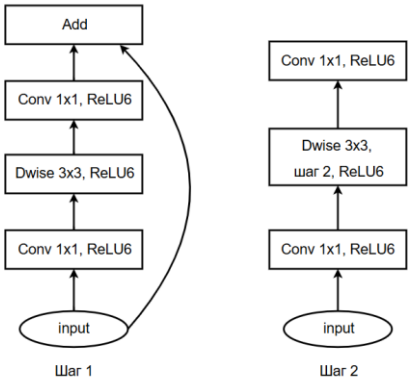
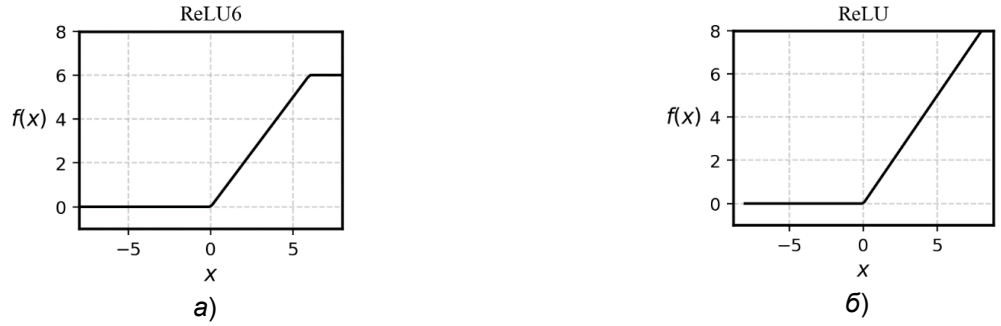


Рисунок 5 – MN блоки

Для лучшего обучения моделей используется механизм связи (Skip Connection). Этот механизм позволяет обходить сложные операции свертки при обратном распространении ошибки и обучать намного эффективнее.

Также есть слой обработки и остаточный блок., которые предназначены для обработки входного изображения и уточнения признаков. Структура такого блока такая же, как и у MN блока с шагом один. Механизма связи в. слое обработки нет, а в остаточный блок есть, что ускоряет обучение.

В нашей реализации выход исходной модели был изменен. На финальном этапе карты признаков вытягиваются в вектор и проходят через полносвязные слои с функцией ReLU (рисунок 6) [5]. После этих слоев получается 63 значения, которые отвечают за наши координаты ключевых точек. Данное решение называется регрессионной головой. Полученная архитектура MobileNetV2 состоит из 2.57 миллионов параметров и представлена в таблице 2:



а) *б*)

Рисунок 6 – Функции активации: а) ReLU6; б) ReLU [4-5]

Таблица 2 – Структура нейросети MobileNetV2

№	Название / Тип блока	Входная размерность	Выходная размерность	Коэффициент расширения	Шаг	Повторы
1	Слой обработки	224×224×3	112×112×32	-	2	1
2	Блок MN	112×112×32	112×112×16	1	1	1
3	MN + Остаточный блок	112×112×16	56×56×24	6	2	2
4	MN + Остаточный блок	56×56×24	28×28×32	6	2	3
5	MN + Остаточный блок	28×28×32	14×14×64	6	2	4
6	MN + Остаточный блок	14×14×64	14×14×96	6	1	3
7	MN + Остаточный блок	14×14×96	7×7×160	6	2	3
8	Блок MN	7×7×160	7×7×320	6	1	1
9	Слой объединения (свертка 1x1)	7×7×320	7×7×1280	-	1	1
10	Векторизация	7×7×1280	1×1×1280	-	-	1
11	Регрессионная голова	1×1280	63	-	-	1

В процессе обучения использовались данные с тренировочной выборки. Метрика при обучении использовалась средняя квадратическая ошибка (MSE), в качестве оптимизатора использовалась Adam [5], изначальная скорость обучения составляла 1×10^{-3}, но в процессе дообучения была уменьшена и состовляла 1×10^{-6}, планировщик скорости использовался ReduceLROnPlateau, который изменяет скорость обучения если ошибка перестала улучшаться (падать) [6]. По результатам обучения ошибка MSE составила 0.000724 (RMSE = 0.027) или 6 пикселов, что является хорошим результатом.

Для реализации классификационной модели использовался один из методов машинного обучения, метод опорных векторов [8]. На вход классификатора идут значения ключевых точек, которые сформировала модель MobileNetV2. Но так как каждая точка имеет три координаты, то перед тем, как подавать значения идет предобработка, все значения вытягиваются в вектор с 63-мя значениями. После чего классификатор вычисляет значения решающей функции на основе ядра (линейное, полиномиальное и др.), а именно схожесть с теми значениями, на основе которых строился классификатор. Так как решается задачи многоклассовой классификации, то используется стратегия один против всех. Для 7 классов строится 21 классификаторов. После чего идет «голосование», где отбирается нужный класс.

Для поиска параметров данного метода использовался GridSearchCV [9]. Он ускоряет поиск параметров, подбирая параметры, которые указал для поиска пользователь.

Параметры для поиска:

1. Ядро: линейное, полиномиальное, сигмоидальное, RBF (радиальная базисная функция);

2. Коэффициент регуляризации — параметр, который контролирует компромисс между сложностью модели и ошибкой на обучающей выборке.

3. Коэффициент ядра — параметр, используемый для нелинейных ядер. Определяет «дальность» влияния каждого отдельного тренировочного примера на форму разделяющей поверхности. Может быть масштабируемым, автоматически подобрал моделью или фиксированным значением.

4. Степень полинома (только для полиномиального ядра);
5. Веса классов. Данный параметр помогает бороться с неравномерным распределением классов.

Лучшими параметрами для классификатора были следующие: коэффициент регуляризации — 50, используемое ядро — RBF, коэффициент ядра — масштабируемый. В процессе обучения метрика F1-мера (macro) равнялась 0.8038. Матрица спутывания для этой модели представлена на рисунке 7.

Рисунок 7 – Матрица спутывания

Стоит отметить, что классификатор выдает вероятности для классов и для того, чтобы не было ложных срабатываний был внесен порог, который равен 0.7.

Взаимодействие системы. Автономная системы распознавания жестов реализована на базе Raspberry PI. К этому одноплатному компьютеру подключается камера, для отслеживания рук. Для отображения результатов работы системы подключается дисплей или удаленное устройство, например персональный компьютер.

Работа системы представлена следующим образом:
1. Получение изображения руки с камеры;
2. Определение области на изображении с рукой с помощью детектора;
3. Формирование координат ключевых точек через сеть MobileNetV2, отрисовка скелета кисти;
4. Определение класса жеста и отрисовка;
5. Вывод полученного результата.

На рисунке 8 показана схема работы системы.

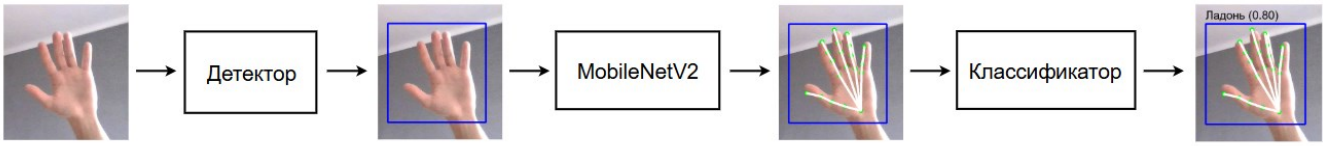


Рисунок 8 – Схема работы системы

Вывод. В результате работы получилось реализовать автономную систему распознавания жестов на базе детектора от Mediapipe, модели MobileNetV2, которая была переделана под задачу распознавания ключевых точек (ошибка построения точек состовляет 6 пикселов), и классификатора на основе метода опорных векторов (точность работы около 80%).

Список использованных источников:

1. Zimmermann C. et al. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images //Proceedings of the IEEE/CVF international conference on computer vision. – 2019. – С. 813-822.

2. Zhang F. et al. Mediapipe hands: On-device real-time hand tracking //arXiv preprint arXiv:2006.10214. – 2020.

3. Sandler M. et al. Mobilenetv2: Inverted residuals and linear bottlenecks //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2018. – С. 4510-4520.

4. ReLU6 — PyTorch 2.11 documentation [Электронный ресурс] // PyTorch. – Режим доступа: https://docs.pytorch.org/docs/stable/index.html. – Дата доступа: 21.03.2026.

5. ReLU — PyTorch 2.11 documentation [Электронный ресурс] // PyTorch. – Режим доступа: https://docs.pytorch.org/docs/stable/index.html. – Дата доступа: 21.03.2026.

6. Kingma D. P., Ba J. Adam: A method for stochastic optimization //arXiv preprint arXiv:1412.6980. – 2014.

7. Al-Kababji A., Bensaali F., Dakua S. P. Scheduling techniques for liver segmentation: Reducelronplateau vs onecyclelr //International conference on intelligent systems and pattern recognition. – Cham : Springer International Publishing, 2022. – С. 204-212.

8. Schölkopf B. et al. Support vector method for novelty detection //Advances in neural information processing systems. – 1999. – Т. 12.

9. GridSearchCV — scikit-learn 1.8.0 documentation [Электронный ресурс] // scikit-learn. – Режим доступа: https://scikit-learn.org/stable/api/index.html. – Дата доступа: 26.03.2021.

.

UDC 004.93'1

AN AUTONOMOUS GESTURE RECOGNITION SYSTEM

Ermakovich V.A student

Belarusian State University of Informatics and Radioelectronics
Minsk, Republic of Belarus

Vashkevich M.I. – Doctor of Science

Annotation. The paper presents the design of an autonomous gesture recognition system based on a simple device. During the design, ready-made solutions were used, as well as methods for refining them. The FreiHAND dataset, the MediaPipe-based detector, the MobileNetV2 convolutional neural network architecture, and the support vector method were used. The paper shows data images, formulas for preparing values for training, explanations of the features of the MobileNe family, a table with MobileNetV2 parameters, and information about the construction of the support vector method.

Keywords. Convolutional neural networks, machine learning, computer vision.