|  |  |
| --- | --- |
|  |  |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное автономное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика, искусственный интеллект и системы управления

КАФЕДРА Системы обработки информации и управления

**ОТЧЕТ**

***ПО ДОМАШНЕМУ ЗАДАНИЮ***

***НА ТЕМУ:***

**Обучение нейронной сети на собранном датасете и создание Web-приложения**

Студент \_\_ИУ5-22М\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_П.А.Бибиков \_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_А.И.Канев\_\_**

(Подпись, дата) (И.О.Фамилия)

2025 г.

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 1](#_Toc169725683)

[ВВЕДЕНИЕ 3](#_Toc169725684)

[Задание 3](#_Toc169725685)

[Часть 1. Сбор датасета 4](#_Toc169725686)

[Часть 2. Обучение нейронной сети 6](#_Toc169725687)

[Часть 3. Web-приложение 13](#_Toc169725688)

[ЗАКЛЮЧЕНИЕ 15](#_Toc169725689)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 16](#_Toc169725690)

ВВЕДЕНИЕ

Распознавание видов техники — типичная задача компьютерного зрения, в которой небольшие объёмы данных и высокое внутриклассовое разнообразие изображений создают серьёзные трудности для обучения «с нуля». В данной работе ставилась цель построить компактную, но достаточно точную модель для классификации трёх кузовов машин — седанов, пикапов и кабриолетов — по изображениям и развернуть её в форме web-приложения.

Чтобы компенсировать ограниченный объём исходных данных (300 снимков, по 100 на класс), была применена обширная аугментация, регуляризация весов и перенос обучения. В качестве базовой использовалась лёгкая свёрточная архитектура MobileNet V2 × 0.5, предварительно обученная на CIFAR-100; это позволило начать обучение с уже сформированными низкоуровневыми признаками и существенно ускорило сходимость. Для доказательства полезности transfer-learning также были поставлены контрольные эксперименты:

* полносвязная сеть «с чистого листа»;
* MobileNet с заморозкой части слоёв;
* полное дообучение MobileNet с разными наборами аугментаций и гиперпараметров.

# **Задание**

Необходимо создать и разметить собственный набор данных, состоящий из изображений. Набор содержит не менее 3 классов и не менее 100 экземпляров каждый. Изображения можно скачать из интернета или объединить несколько существующих датасетов. Создать web-приложение для классификации изображений полученного набора данных. Использовать аугментацию данных, регуляризацию, перенос обучения.

# **Часть 1. Сбор датасета**

По заданию необходимо обучить нейронную сеть, способную распознавать различные виды кузовов машин: седанов, пикапов и кабриолетов.

Для решения этой задачи необходимо найти в интернете изображения машин для формирования обучающей и тестовой выборки.

Изображения были найдены и проверены вручную.

В результате получилось:

1. Седаны – 100 изображения
2. Пикапы – 105 изображений
3. Кабриолеты – 108 изображений

Примеры изображений указаны на рисунках ниже:



Рис. 1 - Пример седана.



Рис. 2 - Пример пикапа.



Рис. 3 - Пример кабриолета.

# **Часть 2. Обучение нейронной сети**

Так как датасет включает в себя всего 300 изображений, будем использовать аугментацию (из ЛР 3) в 100% случаев.

Также ввиду малого количества обучающих данных будем использовать регуляризацию в виде штрафа за сложность модели. Сглаживание меток не показало своей эффективности.

В качестве оптимизатора был выбран AdamW, как показавший наивысшую эффективность в предыдущих ЛР.

Ради эксперимента обучим полносвязную нейросеть с самой успешной архитектурой из ЛР 1 (один скрытый слой на 100 нейронов).

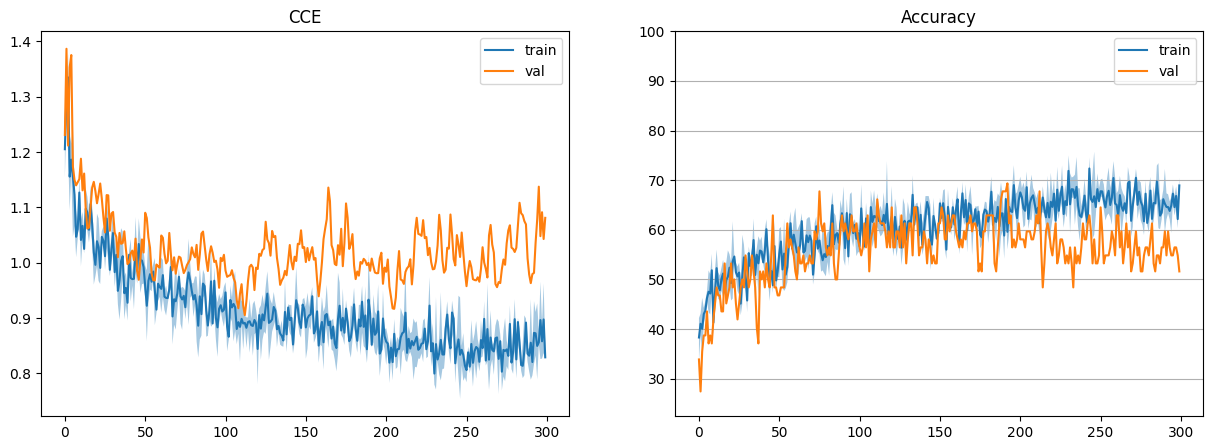


Рис. 4 - графики функции ошибки и точности для полносвязной сети.

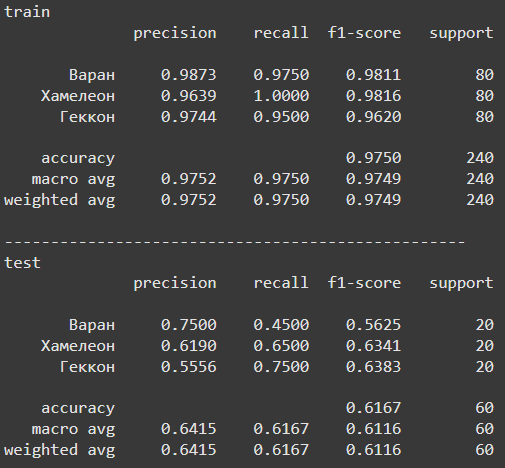


Рис. 5 – метрики полносвязной нейросети.

Высокая точность на обучении (≈ 97 %) при умеренной на тесте (≈ 62 %) указывает на сильное переобучение. Полносвязная архитектура не способна эффективно извлекать пространственные признаки изображений.

Далее дообучим mobilenet с заморозкой двух последних слоёв.

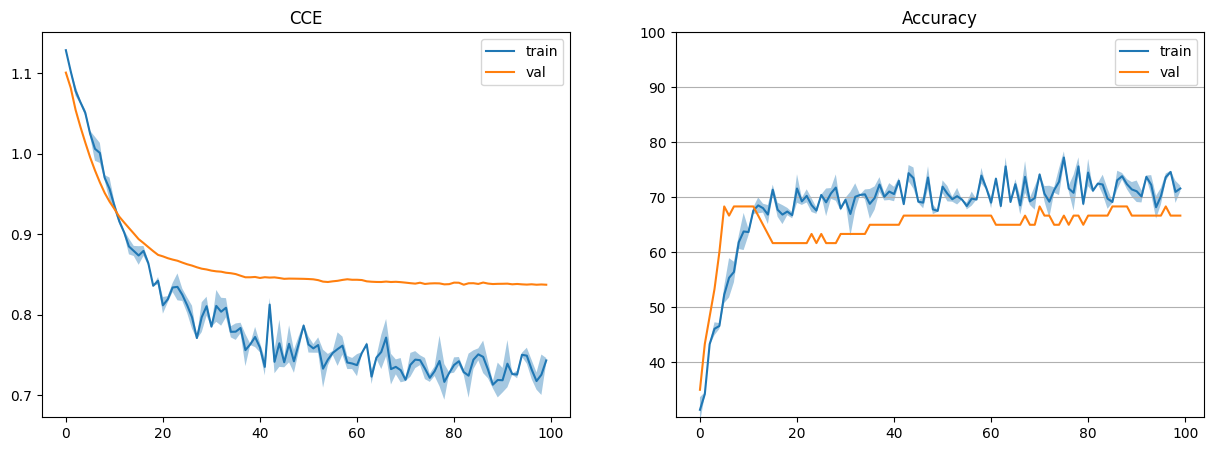


Рис. 6 - графики функции ошибки и точности mobilenet с заморозкой двух последних слоёв.

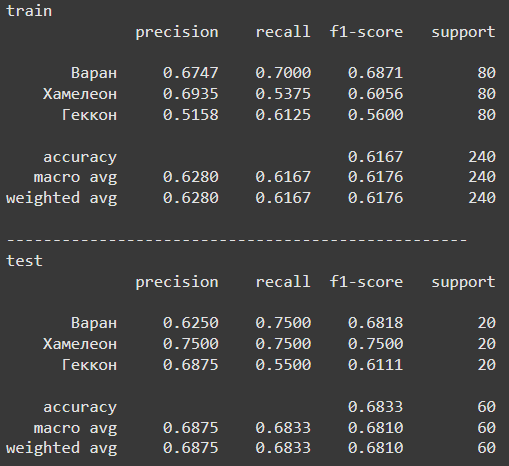


Рис. 7 – метрики mobilenet с заморозкой двух последних слоёв.

Низкая обучающая точность (≈ 62 %) и чуть более высокая тестовая (≈ 68 %) говорят об недообучении — замороженная часть не адаптируется под новый домен, а лишь последние слои не успевают выучить устойчивые признаки.

Полностью разморозим нейросеть:

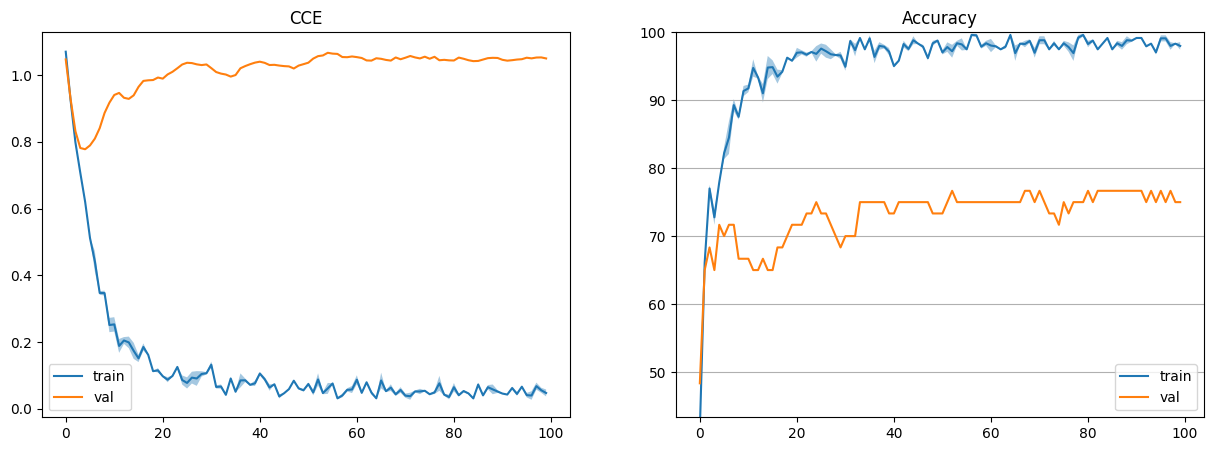


Рис. 8 - графики функции ошибки и точности размороженной mobilenet.

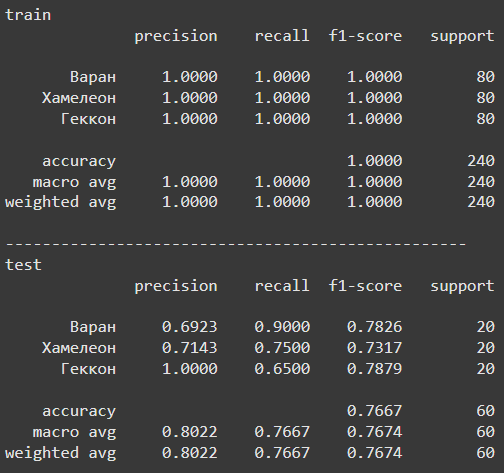


Рис. 9 – метрики размороженной mobilenet.

Дообучение всей сети резко повышает обучающую точность (≈ 99–100 %), тестовая растёт до ≈ 77 %. Переобучение вновь начинает проявляться, но результат существенно лучше, чем в п. 1–2.

К преобразованиям из 3 ЛР добавим несколько новых, итоговый список преобразований:  
 T.ColorJitter(brightness=0.1, contrast=0.1, saturation=0.2, hue=0.0),

T.RandomAffine(degrees=15, translate=(0.1, 0.1), scale=(0.8, 1.2), shear=5),

T.RandomHorizontalFlip(p=0.5),

T.RandomVerticalFlip(p=0.2),

T.RandomErasing(p=0.2, scale=(0.02, 0.02)),

T.RandomResizedCrop(size=(32, 32), scale=(0.8, 1.0)),

T.RandomGrayscale(p=0.1).

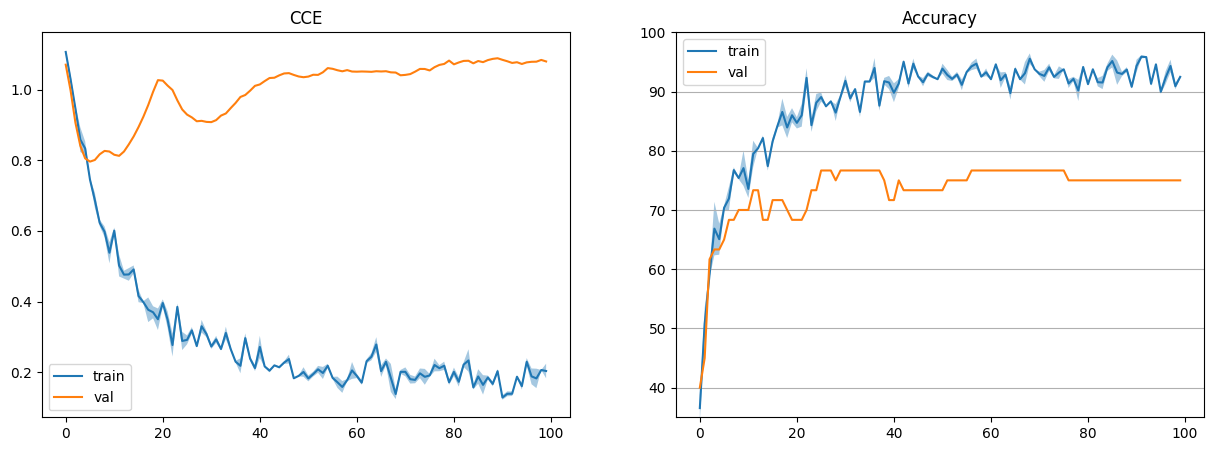


Рис. 10 - графики функции ошибки и точности mobilenet с дополнительной аугментацией.

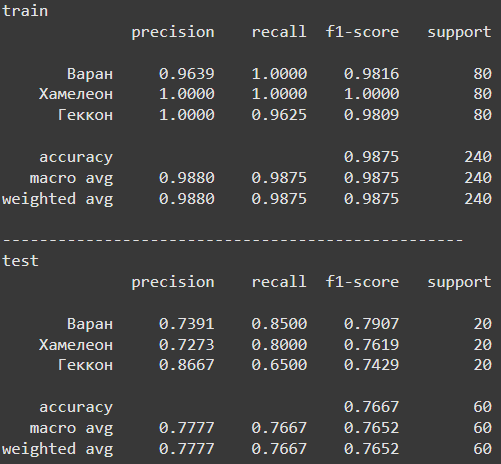


Рис. 11 – метрики mobilenet с дополнительной аугментацией.

Дополнительные преобразования повысили вариативность, но не компенсировали переобучение.

Уменьшим размер батча с 128 до 64 и изменим коэффициент планового уменьшения скорости обучения с 0.5 до 0.25:

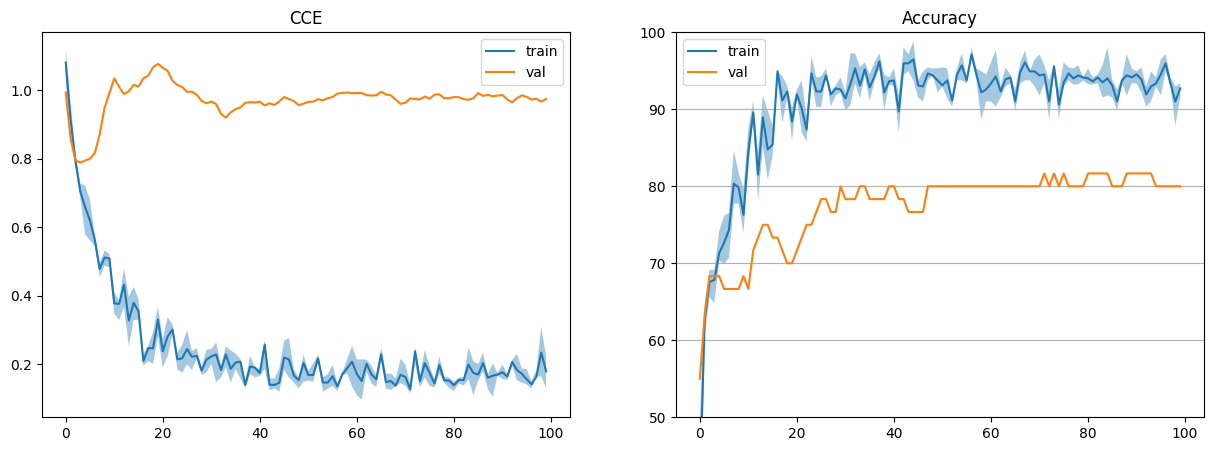


Рис. 12 - графики функции ошибки и точности настроенной mobilenet.

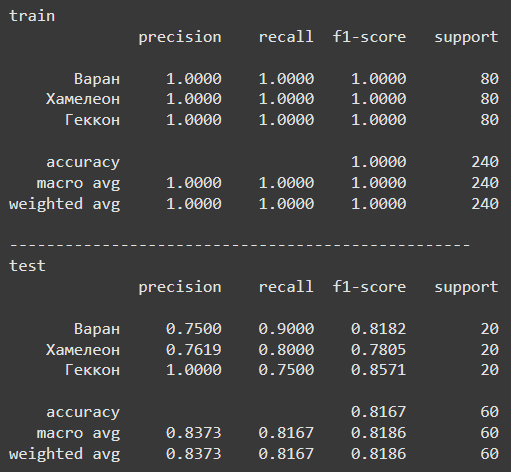


Рис. 13 – метрики настроенной mobilenet.

test-accuracy поднялась до ≈ 81–82 %, разница train\test-метрик сократилась. Меньшие батчи усилили стохастичность SGD-шага и стали «естественным» регуляризатором.

Таблица 1. Результаты.

|  |  |  |  |
| --- | --- | --- | --- |
| Конфигурация сети | Гиперпараметры | Точность | Пояснения |
| Полносвязная нейронная сеть | lr = 1e-3,  batch\_size = 128,  epochs = 300,  AdamW,  betas=(0.9, 0.999), eps=1e-8,  weight\_decay=1e-2 | test = 61.67%  train = 97.49% | Полносвязная нейронная сеть |
| MobileNet  keep\_last=2 | lr = 3e-4,  batch\_size = 128,  epochs = 60,  AdamW,  betas=(0.9, 0.999), eps=1e-8,  weight\_decay=1e-4 | test = 68.33%  train = 61.67% | mobilenet с заморозкой двух последних слоёв |
| MobileNet  keep\_last=all | lr = 3e-4,  batch\_size = 128,  epochs = 60,  AdamW,  betas=(0.9, 0.999), eps=1e-8,  weight\_decay=1e-4 | test = 76.67%  train = 100% | Размороженная mobilenet |
| MobileNet  keep\_last=all | lr = 3e-4,  batch\_size = 128,  epochs = 60,  AdamW,  betas=(0.9, 0.999), eps=1e-8,  weight\_decay=1e-4 | test = 76.67%  train = 98.75% | Увеличили аугментацию |
| MobileNet  keep\_last=all | lr = 3e-4,  batch\_size = 64,  epochs = 60,  AdamW,  betas=(0.9, 0.999), eps=1e-8,  weight\_decay=1e-4 | test = 81.67%  train = 100% | Уменьшили батч и коэффициент планового уменьшения скорости обучения |

# **Часть 3. Web-приложение**

Создаём веб-приложение с помощью фреймворка Django.

Изображение выглядит как текст, колесо, транспортное средство, Наземный транспорт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 14 - Вывод приложения при загрузке изображения с седаном.

Изображение выглядит как текст, транспортное средство, Наземный транспорт, колесо

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 15 - Вывод приложения при загрузке изображения с пикапом.

Изображение выглядит как текст, Наземный транспорт, транспортное средство, колесо

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 16 - Вывод приложения при загрузке изображения с кабриолетом.

ЗАКЛЮЧЕНИЕ

В ходе работы был построен полный цикл решения задачи визуальной классификации: от сбора и ручной разметки датасета до развёртывания готового приложения.

Transfer-learning с предобученной MobileNet V2 × 0.5 (CIFAR-100) оказался существенно эффективнее обучения «с нуля»: +20 п.п. к точности на тестовой выборке по сравнению с полносвязной базой.

Заморозка большинства слоёв привела к недообучению, а полное дообучение без дополнительных мер — к переобучению. Лучшие результаты достигнуты при полном дообучении, усиленной аугментации и тонкой настройке lr-scheduler.

Уменьшение батча неожиданно повысило accuracy с ≈ 77 % до ≈ 82 % и сократив разрыв между обучающей и валидационной метриками (вероятнее всего флуктуации градиента батча помогли оптимизатору выбраться из локального минимума).

Веб-приложение на Django демонстрирует практическую применимость модели: загрузив изображение, пользователь мгновенно получает вероятностный прогноз принадлежности к одному из трёх классов.

Итоговая точность модели ≈ 82 % подтверждает работоспособность подхода, учитывая малый объём исходных данных. Дальнейшее улучшение возможно за счёт увеличения датасета, тестирования других архитектур и тонкой оптимизации гиперпараметров.

​

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Методические указания к лабораторным работам. Источник: <https://github.com/iu5git/Deep-learning>
2. Материал видео-лекций по предмету “Разработка нейросетевых систем”. Источник: [Deep learning - YouTube](https://www.youtube.com/playlist?list=PLLELLTvDgUQ_d9eUj_3XVpAdGByuU37kT)