|  |  |
| --- | --- |
| **Изображение выглядит как эмблема, герб, символ, нашивка  Контент, сгенерированный ИИ, может содержать ошибки.** | **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 и системы управления

КАФЕДРА Системы обработки информации и управления

**Лабораторная работа №3**

**По курсу**

«**Разработка нейронных сетей**»  
**По теме «Аугментация данных»**

Выполнил:  
Студент группы ИУ5-22МБибиков П.А.  
21.04.2025

Проверил:

**Канев А.И.**

2025 г.

# Задание

По заданию выбрать свои классы и обучить сверточную нейронную сеть c использованием техник дропаут и аугментации данных. Сравнить три варианта дропаута и три варианта аугментации данных.

Проанализировать результаты обучения в Tensorboard

Классы из CIFAR100 по варианту 7:

1 класс = номер группы + 15 = 22 + 15 = **37 (дома, house)**

2 класс = номер варианта + 56 = 2 + 56 = **58 (пикап, pickup\_truck)**

3 класс = номер варианта + 21 = 2 + 21 = **23 (облака, cloud)**

# Часть 1. Обучение нейронной сети.

Cifar100\_CNN(

(seq): Sequential(

(0): Normalize()

(1): Conv2d(3, 64, kernel\_size=(3, 3), stride=(4, 4))

(2): ReLU()

(3): Dropout2d(p=0.2, inplace=False)

(4): Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(5): ReLU()

(6): AvgPool2d(kernel\_size=4, stride=4, padding=0)

(7): Dropout2d(p=0.3, inplace=False)

(8): Flatten(start\_dim=1, end\_dim=-1)

(9): Linear(in\_features=512, out\_features=3, bias=True)

)

)

Начальная конфигурация нейронной сети указана на рис.1

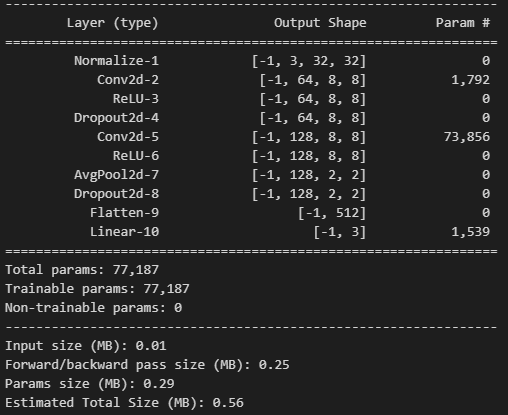


Рис. 1 – базовая конфигурация НС.

Проводим для неё обучение, обучаем 500 эпох, батч 128, метод – обучение с импульсом. Tensorboard Smoothing на графиках – 0.89.

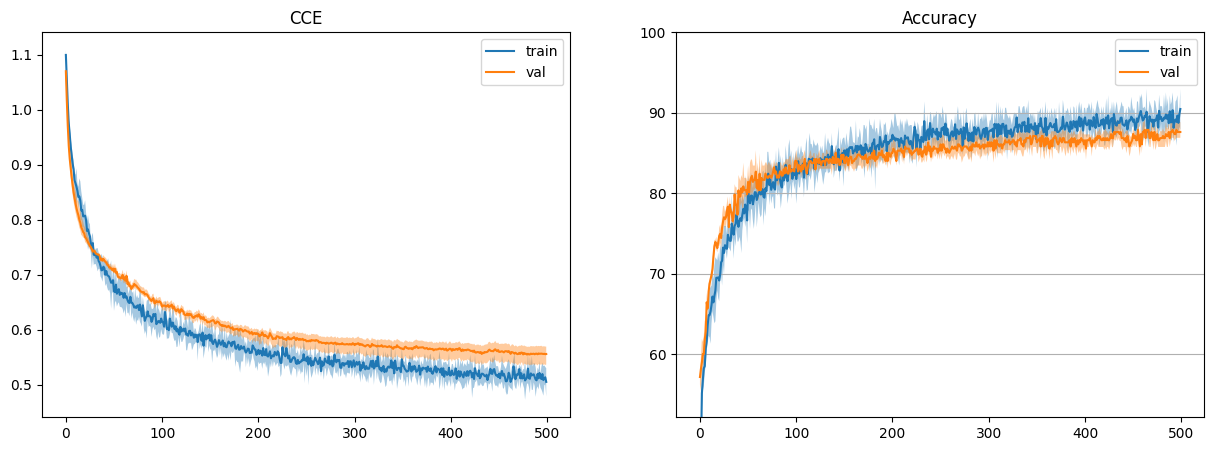


Рис. 2 – графики функции ошибки и точности для начальной конфигурации.

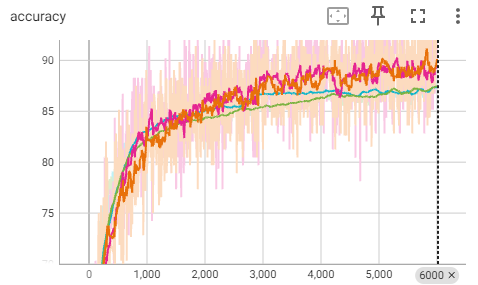


Рис. 3 – графики функции точности для начальной конфигурации.

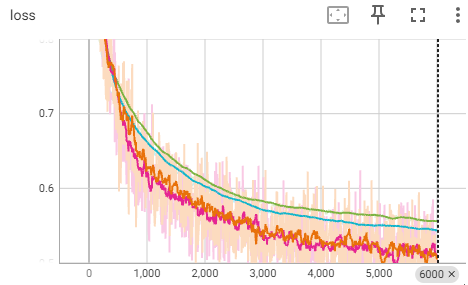


Рис. 4 – графики функции ошибки для начальной конфигурации.

Результаты обучения схожи с ЛР2 – начальная точность на тестовой выборке около 89 %. Однако, по графикам видно – модель куда позже начинает замедлять обучение. За 6000 итераций максимум получен на 4503, с точностью в 88.4%.

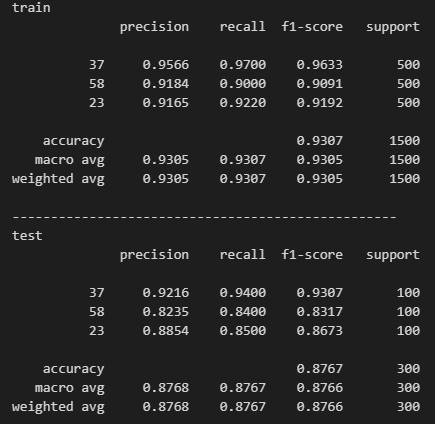


Рис. 5 базовые метрики модели на тестовой выборке.

# Часть 2. Обучение НС с конфигурацией из ЛР2.

Возьмём конфигурацию из ЛР2 и применим на ней техники регуляризации и аугментации. Ниже на рисунке конфигурация из ЛР2.

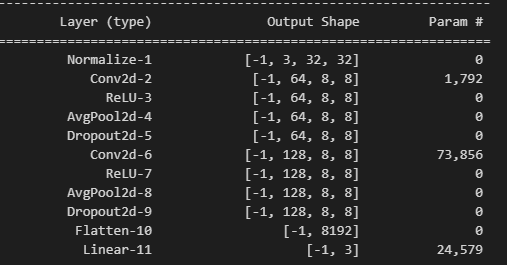


Рисунок 6 - Конфигурация нейронной сети из ЛР2.

В данной лабораторной работе к модели добавляется аугментация. Посмотрим на результат модели.

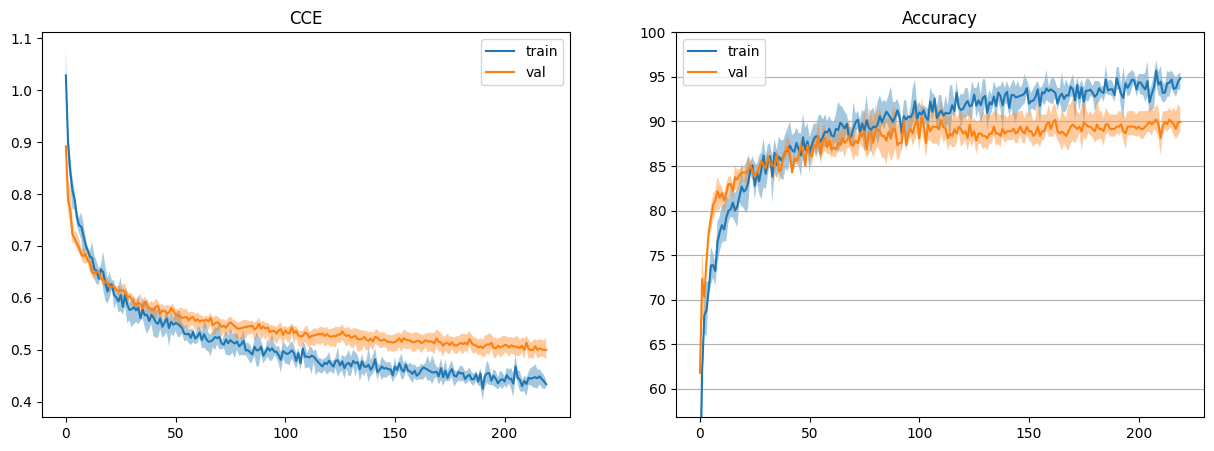


Рисунок 7 - График потерь для конфигурации ЛР2.

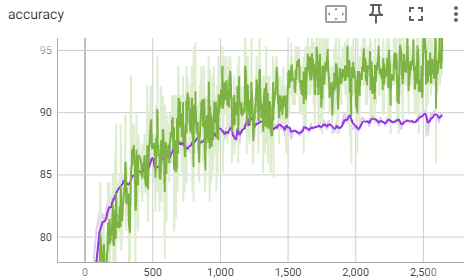


Рисунок 8 - График точности для конфигурации ЛР2

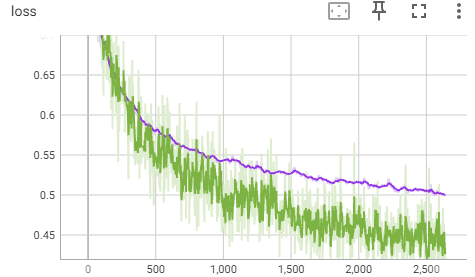


Рисунок 9 - График потерь для конфигурации ЛР2.

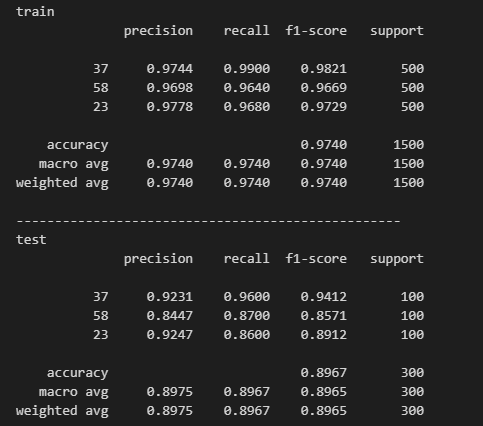


Рисунок 10 - Показатели модели ЛР2.

# Часть 3. Использование dropout.

Проведем анализ dropout, применив его к нашей ЛР. Сначала поставим к обоим слоям коэффициент равный 0.1, во втором – 0.2

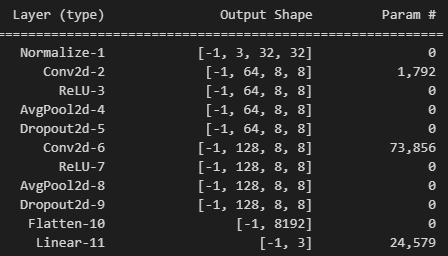


Рисунок 11 - Конфигурация нейронной сети из ЛР2 с dropout = 0.1 и 0.2.

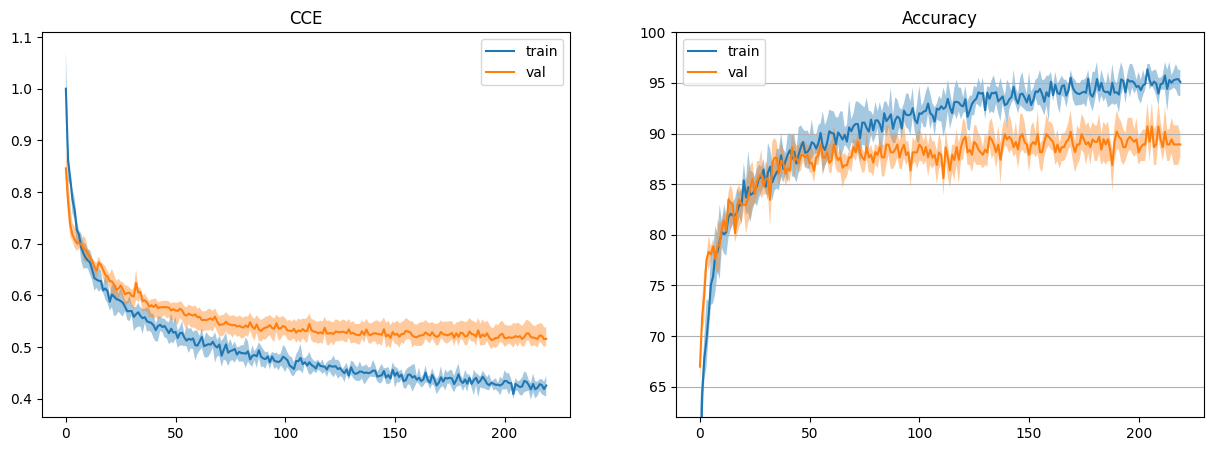


Рисунок 12 - График точности и потерь для конфигурации ЛР2 с dropout = 0.1.

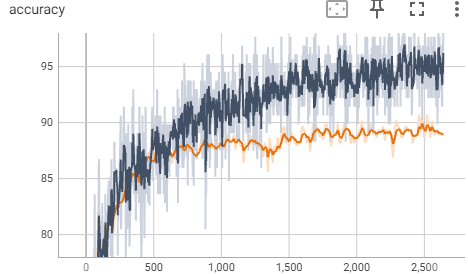


Рисунок 13 - График точности для конфигурации ЛР2 с dropout = 0.1.

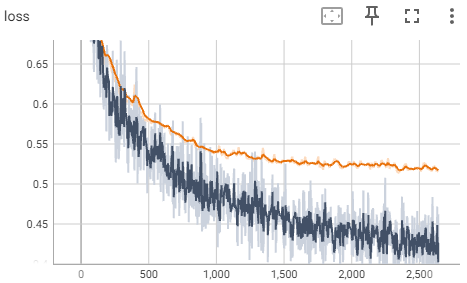


Рисунок 14 - График потерь для конфигурации ЛР2 с dropout = 0.1 и 0.2.

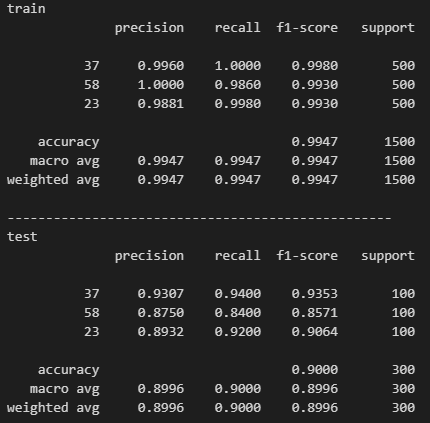


Рисунок 15 - Показатели модели ЛР2 с dropout= 0.1 и 0.2.

Поставим к обоим слоям коэффициент равный 0.2 и 0.3.

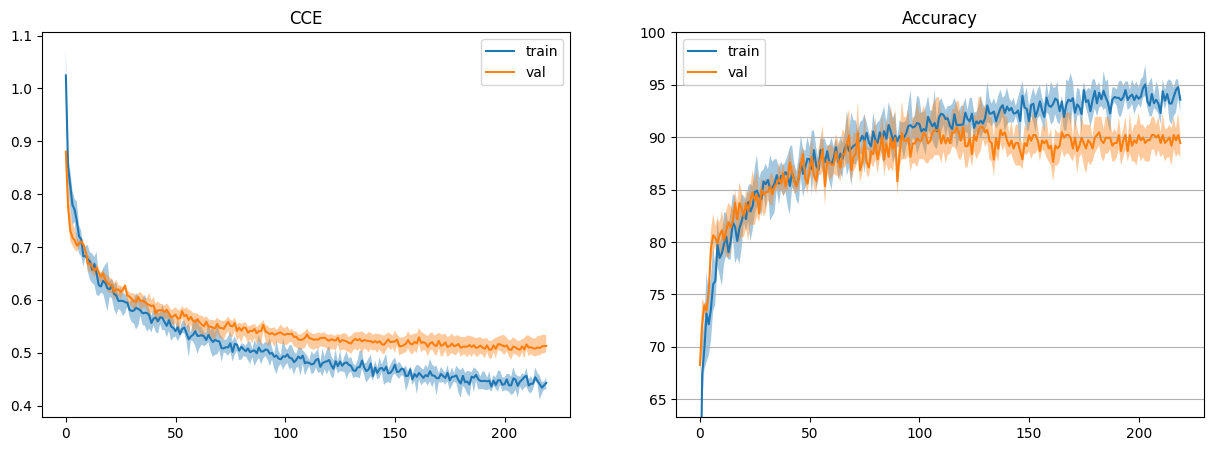


Рисунок 16 - График точности и потерь для конфигурации ЛР2 с dropout = 0.2 и 0.3.

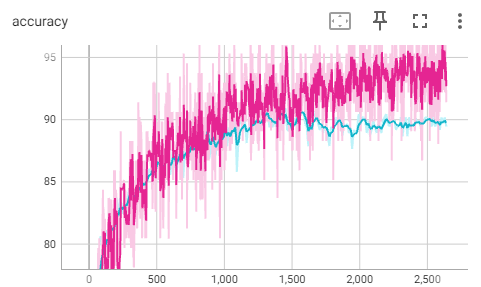


Рисунок 17 - График точности для конфигурации ЛР2 с dropout = 0.2 и 0.3.

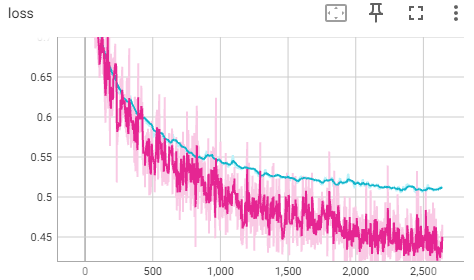


Рисунок 18 - График потерь для конфигурации ЛР2 с dropout = 0.2 и 0.3.

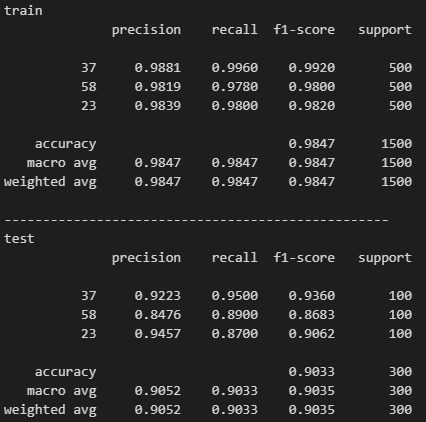


Рисунок 19 - Показатели модели ЛР2 с dropout= 0.2 и 0.3.

Поставим к обоим слоям коэффициент равный 0.3 и 0.4.

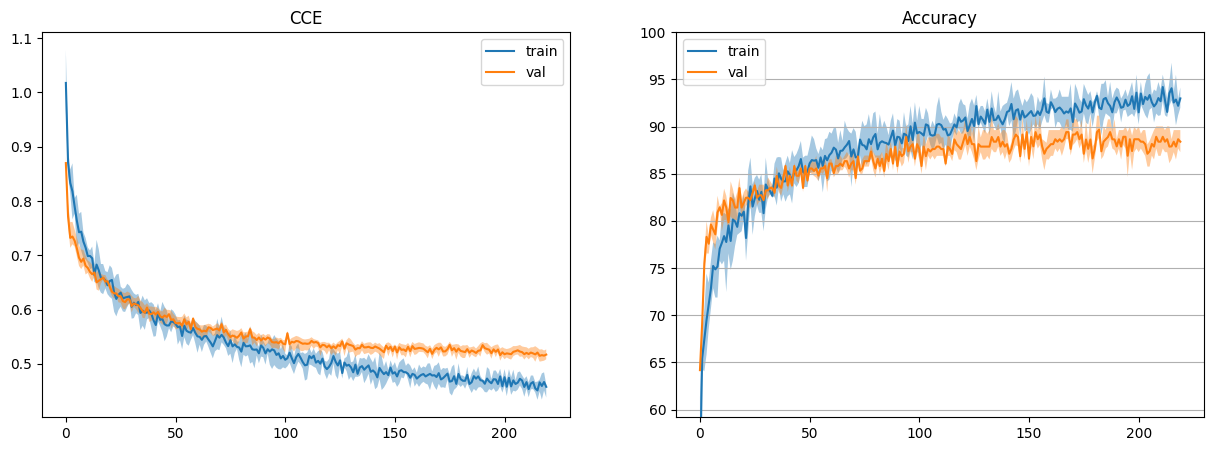


Рисунок 20 - График точности для конфигурации ЛР2 с dropout = 0.3 и 0.4.

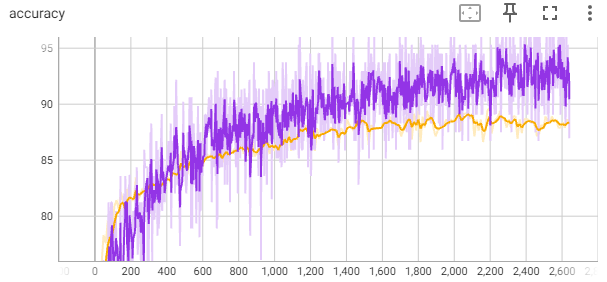


Рисунок 21 - График точности для конфигурации ЛР2 с dropout = 0.3 и 0.4.

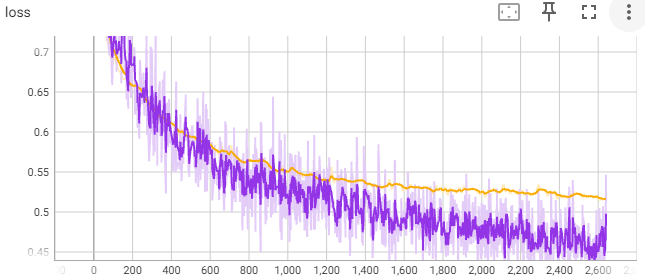


Рисунок 22 - График потерь для конфигурации ЛР2 с dropout = 0.3 и 0.4

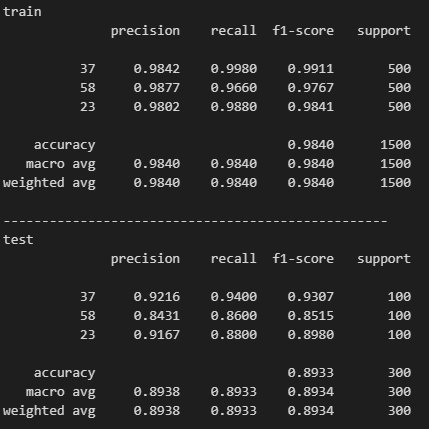


Рисунок 23 - Показатели модели ЛР2 с dropout= 0.3 и 0.4.

# Часть 4. Использование Weight Decay.

Попробуем изменять weight decay. Для результата 1e-5 результат был выше, изменим weight decay и представим результаты в сводной таблице ниже.

Таблица 1 - Результаты смены weight decay

|  |  |
| --- | --- |
| **Weight decay** | **Точность** |
| 1e-5 | 90,3% |
| 1e-4 | 88,7% |
| 1e-3 | 90,3% |
| 1e-2 | 90% |
| 1e-1 | 84,3% |

Наилучшим результатом является weight\_decay = 1e-5

# Часть 5. Сравнение аугментаций.

Сначала попробуем изменить аугментацию, отвечающую за расположение картинки.

Таблица 2 - Результаты смены расположения картинки

|  |  |
| --- | --- |
| **Аугментация** | **Точность** |
| degrees = 15, translate = (0.1, 0.1), scale = (0.8, 1.2), shear = 5 | 90.3% |
| degrees=20, translate=(0.3, 0.3), scale=(0.7, 1.3), shear=8 | 90.3% |
| degrees = 25, translate = (0.2, 0.2), scale = (0.7, 1.3), shear = 10 | 90.3% |

Теперь попробуем изменить настройки цвета

Таблица 3 - Результаты смены настроек цвета

|  |  |
| --- | --- |
| **Аугментация** | **Точность** |
| brightness = 0.1, contrast = 0.1, saturation = 0.2, hue = 0.0 | 90.3% |
| brightness = 0.05, contrast = 0.05, saturation = 0.1, hue = 0.0 | 89.3% |
| brightness=0.4, contrast=0.6, saturation=0.6, hue=0.0 | 89% |

Таблица 4 - Результаты смены оттенков

В конце будем менять оттенки.

|  |  |
| --- | --- |
| **Аугментация** | **Точность** |
| brightness = 0.1, contrast = 0.1, saturation = 0.2, hue = 0.0 | 90.3% |
| brightness = 0.1, contrast = 0.1, saturation = 0.2, hue = 0.2 | 90.3% |
| brightness = 0.1, contrast = 0.1, saturation = 0.2, hue = 0.4 | 94.5% |
| brightness = 2, contrast = 2, saturation = 2, hue = 0.5 | 89.7% |

# Часть 6. Улучшение итоговой модели.

Попробуем увеличить число эпох, найти момент переобучения, затем изменить размер батча, скорость обучения. В рамках улучшения были попытки изменить шаг обучения, число эпох, размер батча, однако это не дало существенных изменений.

Таблица 5 - Результаты улучшения итоговой модели

| **Конфигурация сети** | **Гиперпараметры** | **Точность** | **Комментарии** |
| --- | --- | --- | --- |
| Conv2d(3, 32, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Conv2d(32, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0) | lr = 0.001, batch\_size = 128, epochs = 500 | test = 88.4%, train = 96% | Базовый вариант из ЛР2 |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.1, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.2, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 500,  momentum = 0.9 | test = 90%,  train = 99,5% | Дропаут в первом слое с вероятностью 0.1, во втором – 0.2. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.2, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  AvgPool2d(kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.3, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 500,  momentum = 0.9 | test = 90.3%, train = 98.5% | Дропаут в первом слое с вероятностью 0.2, во втором – 0.3. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.3, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.4, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 200,  momentum = 0.9 | test = 89.3%, train = 98.4% | Дропаут в первом слое с вероятностью 0.3, во втором – 0.4. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.2, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.3, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 500,  momentum = 0.9,  weight\_decay = 1e-5 | test = 90.3%, train = 92% | Выбрали модель с дропаутами -0.2 и 0.3. Добавили модели штраф за сложность 1е-5. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.1, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.2, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 500,  momentum = 0.9,  weight\_decay = 1e-4 | test = 88,7%, train = 89.7% | Поменяли штраф за сложность на 1е-4. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.1, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.2, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 500,  momentum = 0.9,  weight\_decay = 1e-3 | test = 90.3%, train = 89% | Поменяли штраф за сложность на 1е-3. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.2, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.3, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 200,  momentum = 0.9,  weight\_decay = 1e-5, degrees= 15, translate=(0.1, 0.1), scale=(0.8, 1.2), shear=5 | test = 90%,  train = 96% | Добавили аугментацию к варианту модели штраф за сложность 1е-2. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.2, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.3, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 750,  momentum = 0.9,  weight\_decay = 1e-3, degrees= 25, translate=(0.2, 0.2), scale=(0.8, 1.2), shear=10 | test = 90,3%, train = 92% | Изменили degrees на 25, tanslate на (0.2,0.2) и shear на 10. |
| Conv2d(3, 64, kernel\_size=(5, 5), stride=(1, 1), padding=(2, 2))  MaxPool2d(kernel\_size=4, stride=4, padding=0, dilation=1, ceil\_mode=False)  Dropout2d(p=0.2, inplace=False)  Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))  MaxPool2d (kernel\_size=4, stride=4, padding=0)  Dropout2d(p=0.3, inplace=False) | lr = 0.001, batch\_size = 128, epochs = 750,  momentum = 0.9,  weight\_decay = 1e-3, degrees= 5, translate=(0.1, 0.1), scale=(0.9, 1.1),shear=3 | test = 90,3%, train = 93,3% | Изменили degrees на 5, tanslate на (0.1,0.1), scale на (0.9, 1.1) и shear на 3. |

# Часть 7. Сравнение CNN и СNN\_AUG.0

Изображение выглядит как снимок экрана, зеленый, диаграмма, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 25 - Результат прошлой модели при сдвиге вниз на 9 пикселей.

Изображение выглядит как снимок экрана, зеленый, диаграмма, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 26 - Результат обновлённой моделей при таком же сдвиге.

Видим, что обновленная модель смогла дать верный результат, где прошлая давала ошибку.

**Вывод:**

В данной лабораторной работе были рассмотрены различные методы борьбы с переобучением. Среди таких методов, мы использовали дропаут (выключение нейронов), штраф за сложность, аугментация данных.

Однако, их следует использовать аккуратно, применение большого dropout или большого штрафа за сложность приводит к тому, что модель недообучается. При сильной аугментации - картинки могут сильно видоизмениться, однако, в лабораторной работе это практически ни на что не повлияло. Причина этому – сильный контраст между объектами классов.

Конкретно, наихудшие результаты были получены при больших значениях dropout и weight decay. Наилучший результат был получен при использовании ненулевых значениях dropout и weight decay, чуть меньше средних из выбранных, это показывает об эффективности данных методов. В случае аугментации наилучший результат был получен при среднем коэффициенте.

Объяснить в выводе почему одинаковые результаты по дропауту и штрафу. В теории, при увеличении дропаута и штрафа, мы должны были получить недообучение, однако, на практике таких результатов не получили.