

# neural image codec

Прошян Гарри Арменович М4150

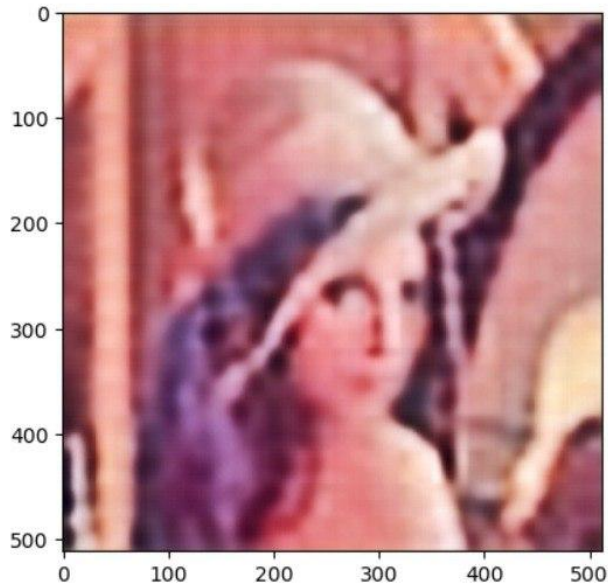
# Модель

- Энкодер - все сверточные слои resnet18
- Декодер - “зеркальная версия” энкодера, состоящая из последовательности residual блоков вида:

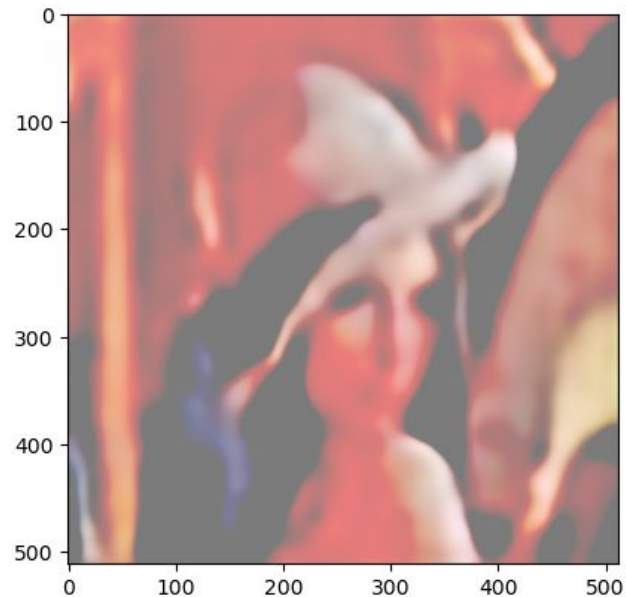
```
def forward(self, x):  
    skip_connection = self.conv_to_match_dims(x)  
    out = self.upscale(x)  
    out = self.double_conv(out)  
    out = out + skip_connection  
    return out
```

- double\_conv = (Conv2d -> Batchnorm -> Relu) x2
- upscale = upsample
- conv\_to\_match\_dims - transpose convolution

# Прирост качества при residual декодере\*



residual decoder



sequential decoder

\* Оба автоэнкодера с данного слайда обучались 5 эпох

# Квантование (только инференс)

Энкодер возвращает значения от 0 до 1 (ФА сигмоида)

Данные значения квантуются в целые числа из интервала  $[0 \text{ до } 2^B]$

```
def quantize(encoder_out: torch.Tensor, B: int):  
    quantized = torch.round(encoder_out * 2**B)  
    return quantized.type(torch.int8)
```

Функция потерь - MSE

$$MSE = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - \hat{f}(x, y)]^2$$

# Класс NeuralImageCompressor

```
class NeuralImageCompressor(nn.Module):
    def __init__(self,
                  encoder: Encoder,
                  decoder: nn.Module,
                  B: int = 1):
        super().__init__()
        self.encoder = encoder
        self.decoder = decoder
        self.B = B

    def _get_quantization_error(self, shape: Tuple[int, ...]):
        mean = torch.full(shape, -0.5)
        std = torch.full(shape, 0.5)
        quan_err = 0.5**self.B * torch.normal(mean = mean, std = std)
        return quan_err

    def forward(self, x):
        out = self.encoder(x)
        quant_err = self._get_quantization_error(out.shape).to(out.device)
        out = out + quant_err
        out = self.decoder(out)
        return out
```

При обучении модели к выходу энкодера прибавлялся шум, соизмеримый по амплитуде с ошибкой квантования

# LooselessCompressor

Для энтропийного кодирования используется объект наследника абстрактного класса LooselessCompressor. Данный абстрактный класс имеет 5 чисто виртуальных методов:

- инициализация состояния алгоритма сжатия из файла
- инициализация состояния алгоритма сжатия по последовательности
- кодирование
- декодирование
- сохранение состояния в файл

# Huffman

Реализует алгоритм Хаффмана, является наследником `LooselessCompressor`.

При использовании в `encoder_pipeline` сохраняет свое состояние в **отдельный** `.json` файл. JSON был выбран для человеко-читаемости, ясно, что закодировать дерево в бинарном виде было бы экономнее

При использовании в `decoder_pipeline` состояние считывается из переданного файла

В файл с сжатым изображением не входит информация о состоянии алгоритма энтропийного кодирования

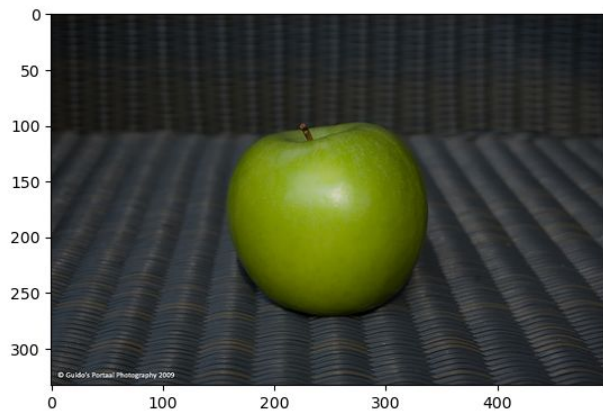
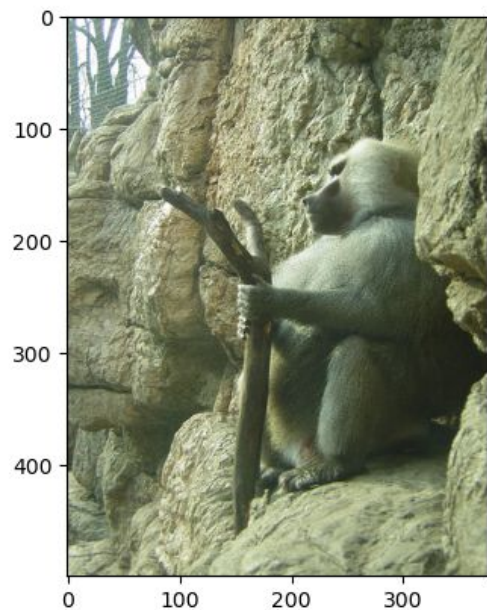


# Датасет

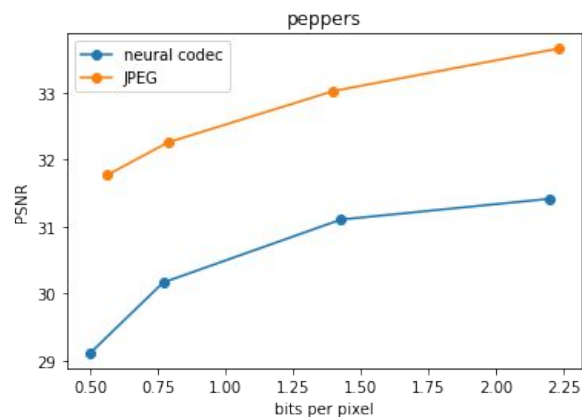
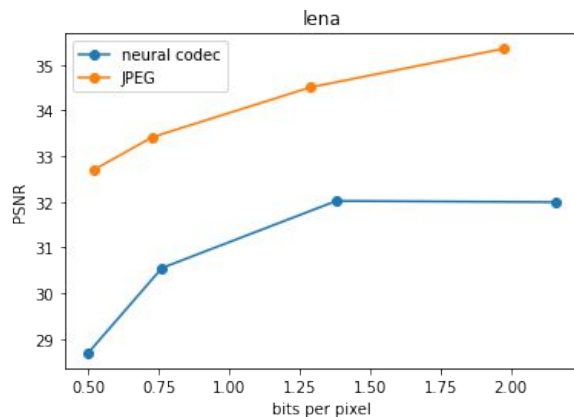
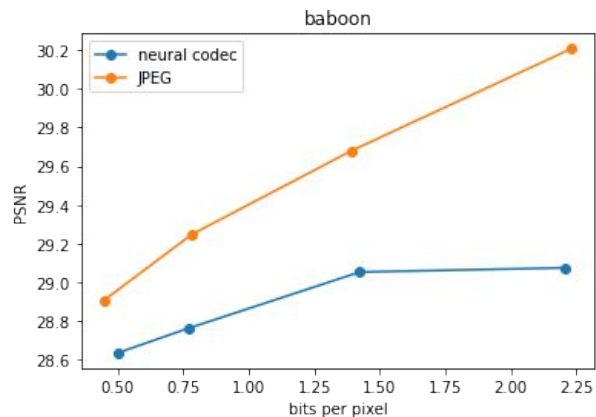
17'000 изображений из 13 классов из Imagenet:

```
"n02486410": "baboon",  
"n07720875": "bell pepper",  
"n03124170": "cowboy hat, ten-gallon hat",  
  
"n02493509": "titi, titi monkey",  
"n02110627": "affenpinscher, monkey pinscher, monkey dog",  
"n02493793": "spider monkey, Ateles geoffroyi",  
"n02480855": "gorilla, Gorilla gorilla",  
  
"n07742313": "Granny Smith",  
"n03724870": "mask",  
"n03379051": "football helmet",  
"n04356056": "sunglasses, dark glasses, shades",  
"n04591157": "Windsor tie",  
"n02906734": "broom"
```

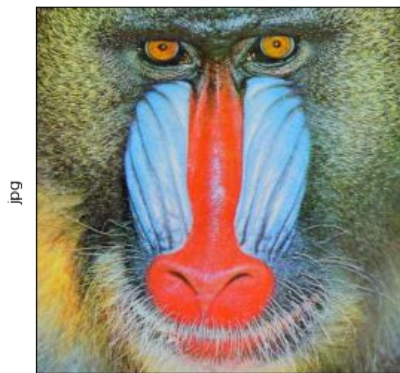
# Датасет



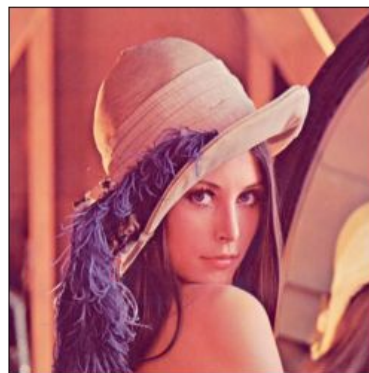
# Результаты



# B = 4 сравнение с JPEG



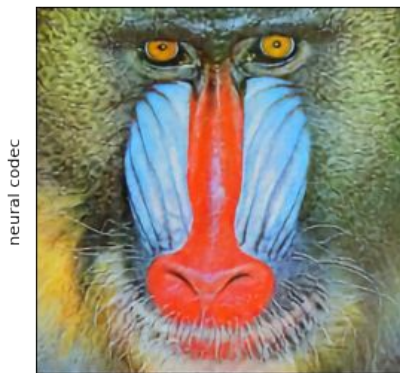
q = 45, bpp = 1.390, psnr = 29.678



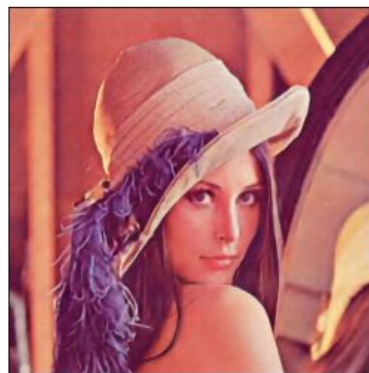
q = 80, bpp = 1.288, psnr = 34.509



q = 80, bpp = 1.398, psnr = 33.022



B = 4, bpp = 1.424, psnr = 29.194

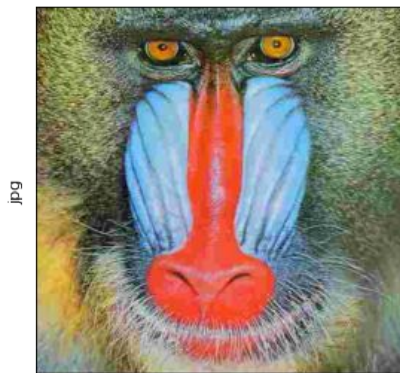


B = 4, bpp = 1.381, psnr = 32.490

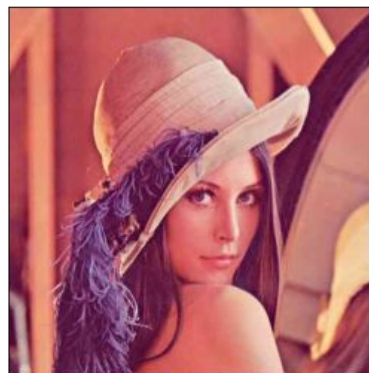


B = 4, bpp = 1.426, psnr = 31.562

# B = 1 сравнение с JPEG



q = 10, bpp = 0.448, psnr = 28.907



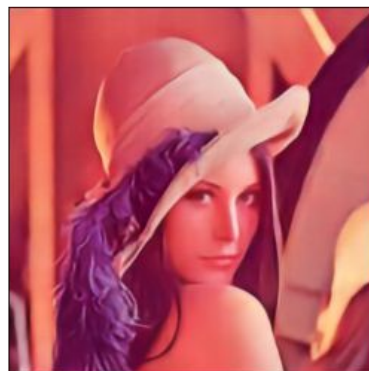
q = 30, bpp = 0.521, psnr = 32.706



q = 30, bpp = 0.562, psnr = 31.768



B = 1, bpp = 0.500, psnr = 28.677



B = 1, bpp = 0.500, psnr = 28.645



B = 1, bpp = 0.500, psnr = 29.139