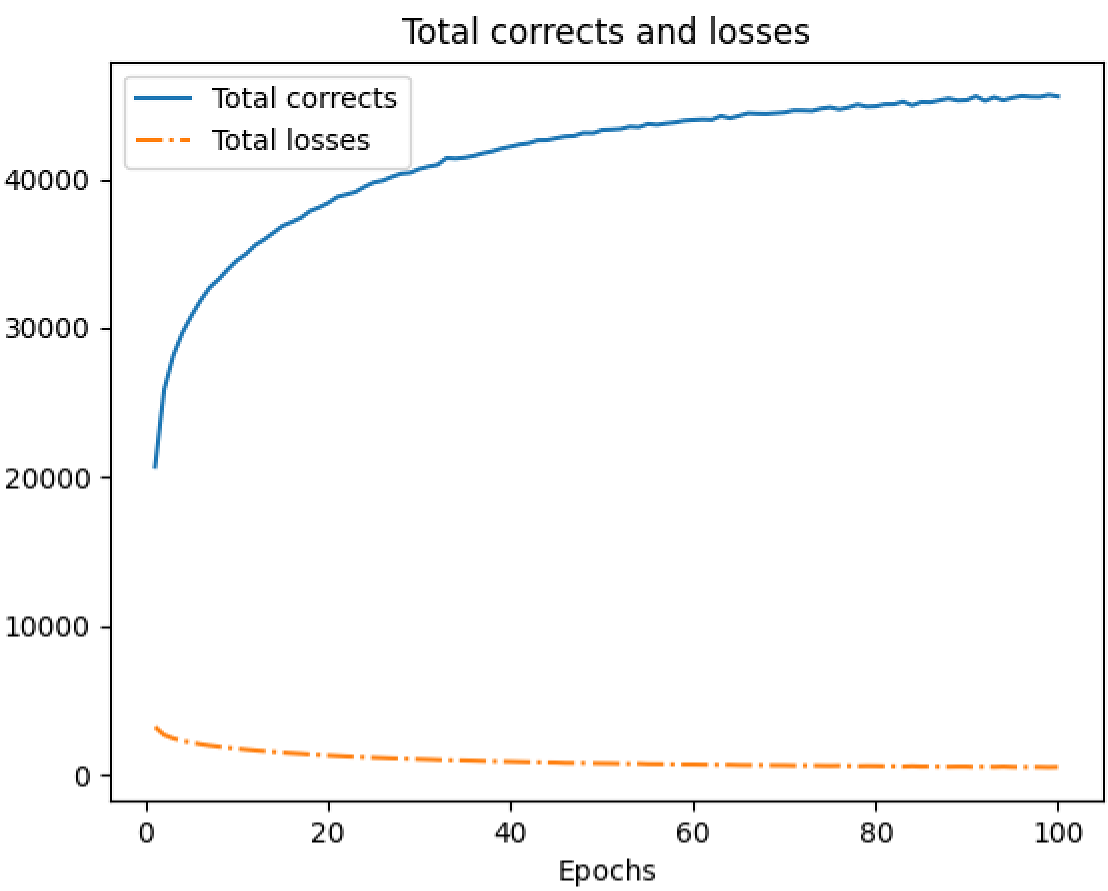
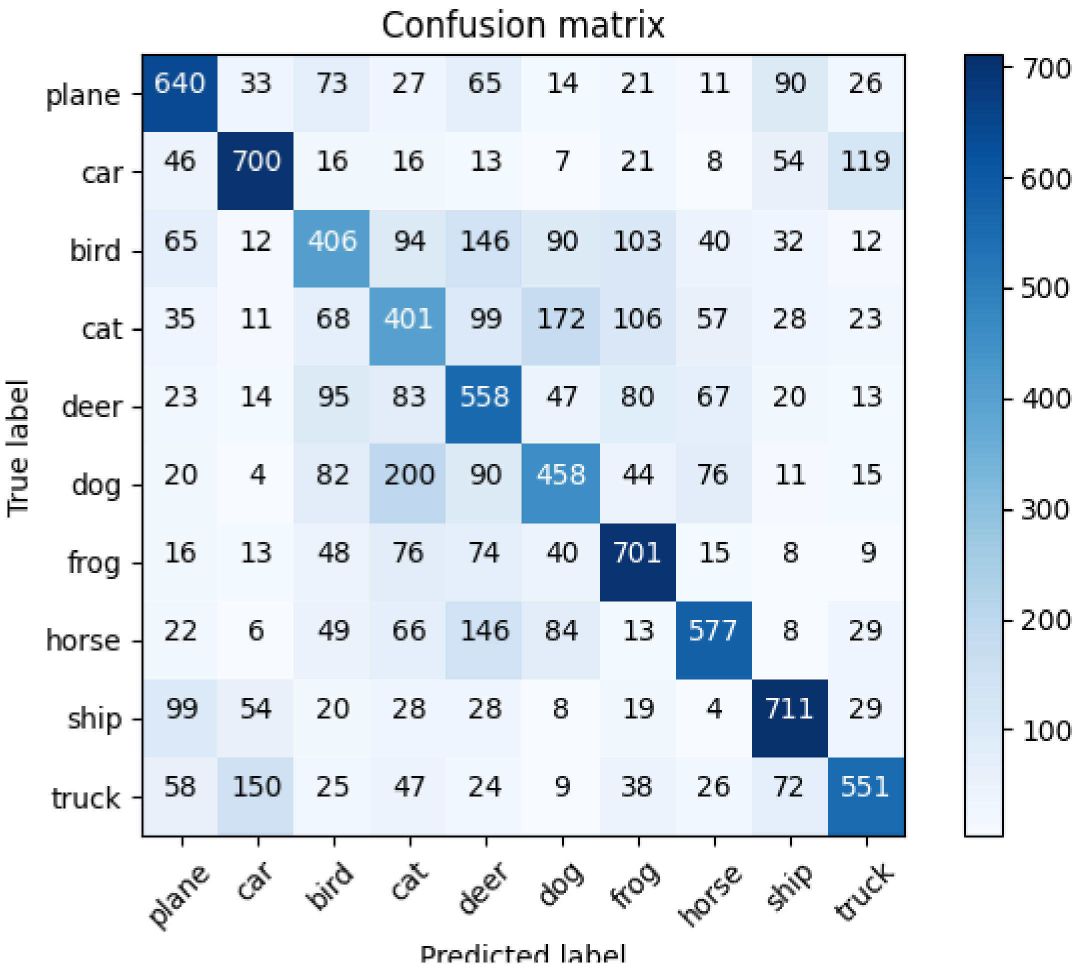
Best model's parameters

- learning rate - 0.001
- epochs - 100
- layers
    - conv1.weight torch.Size([6, 3, 5, 5])
    - conv1.bias torch.Size([6])
    - conv2.weight torch.Size([16, 6, 5, 5])
    - conv2.bias torch.Size([16])
    - fc1.weight torch.Size([120, 400])
    - fc1.bias torch.Size([120])
    - fc2.weight torch.Size([84, 120])
    - fc2.bias torch.Size([84])
    - out.weight torch.Size([10, 84])
    - out.bias torch.Size([10])
- Process of learning



- Confusion matrix



**ENet - A deep neural architecture for real-time semantic segmentation**

- **Overview**
  ENet (Efficient Neural Network) gives the ability to perform pixel-wise semantic segmentation in real-time. ENet is upto 18x faster, requires 75x less FLOPs, has 79x less parameters and provides similar or better accuracy to existing models. Tested on CamVid, CityScapes and SUN datasets.
- architecture in the pictures below

Table 1: ENet architecture. Output sizes are given for an example input of $512 \times 512$.

| Name | Type | Output size |
|---|---|---|
| initial | | $16 \times 256 \times 256$ |
| bottleneck1.0 | downsampling | $64 \times 128 \times 128$ |
| $4\times$ bottleneck1.x | | $64 \times 128 \times 128$ |
| bottleneck2.0 | downsampling | $128 \times 64 \times 64$ |
| bottleneck2.1 | | $128 \times 64 \times 64$ |
| bottleneck2.2 | dilated 2 | $128 \times 64 \times 64$ |
| bottleneck2.3 | asymmetric 5 | $128 \times 64 \times 64$ |
| bottleneck2.4 | dilated 4 | $128 \times 64 \times 64$ |
| bottleneck2.5 | | $128 \times 64 \times 64$ |
| bottleneck2.6 | dilated 8 | $128 \times 64 \times 64$ |
| bottleneck2.7 | asymmetric 5 | $128 \times 64 \times 64$ |
| bottleneck2.8 | dilated 16 | $128 \times 64 \times 64$ |
| *Repeat section 2, without bottleneck2.0* | | |
| bottleneck4.0 | upsampling | $64 \times 128 \times 128$ |
| bottleneck4.1 | | $64 \times 128 \times 128$ |
| bottleneck4.2 | | $64 \times 128 \times 128$ |
| bottleneck5.0 | upsampling | $16 \times 256 \times 256$ |
| bottleneck5.1 | | $16 \times 256 \times 256$ |
| fullconv | | $C \times 512 \times 512$ |

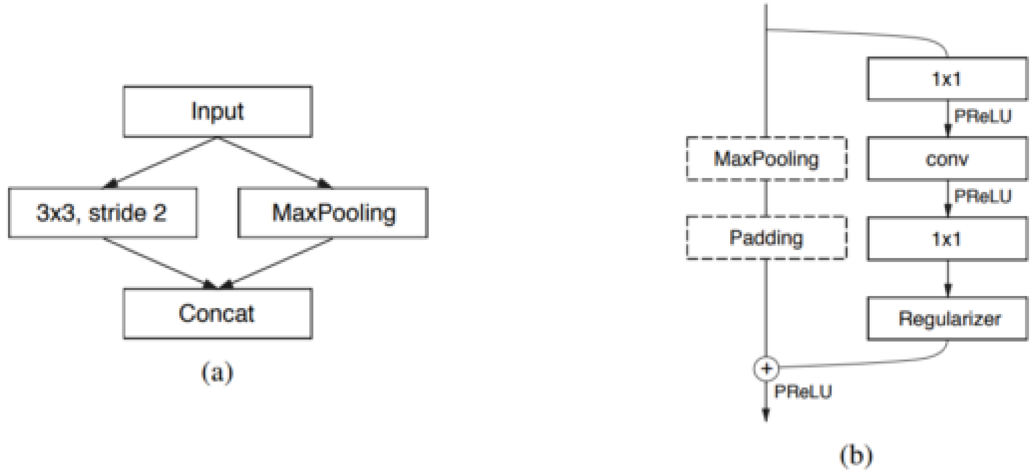



Figure 2: (a) ENet initial block. MaxPooling is performed with non-overlapping $2 \times 2$ windows, and the convolution has 13 filters, which sums up to 16 feature maps after concatenation. This is heavily inspired by [28]. (b) ENet bottleneck module. conv is either a regular, dilated, or full convolution (also known as deconvolution) with $3 \times 3$ filters, or a $5 \times 5$ convolution decomposed into two asymmetric ones.

The visual representation of:

- The initial Block is the one shown in (a)
- And the bottleneck blocks are shown in (b)

Each bottleneck module consists of:

- 1x1 projection that reduces the dimensionality
- A main convolution layer (conv) (either - regular, dilated or full) (3x3)
- 1x1 expansion
- and they place Batch Normalization and PReLU between all convolutional layers.

- If the bottleneck is downsampling, a max pooling layer is added to the main branch. Also, the first 1x1 projection is replaced with 2x2 convolution with stride=2.

- The zero pad the activations to match the number of feature maps.

- The conv is sometimes asymmetric convolution i.e. a sequence of 5 * 1 and 1 * 5 convolutions.

- For the regularizer they use Spatial Dropout
    - with p = 0.01 before bottleneck2.0
    - with p = 0.1 afterwards

- Stage 1, 2, 3 - the encoder - consists of 5 bottleneck blocks (with exception that Stage 3 doesn't downsample).
- Stage 4, 5- the decoder - Stage 4 contains 3 bottlenecks and Stage 5 contains 2 bottlenecks

- Followed by a fullconv which outputs the final output with dimension - C * 512 * 512 , where C is the number of filters.
- A few more facts
    - They didn’t use bias terms in any of the projections
    - Between each convolutional layer and activation, they use Batch Normalization
    - In decoder,MaxPooling is replaced with MaxUnpooling
    - In decoder, Padding is replaced with Spatial Convolution without bias
    - No use of pooling indices in the last(5.0) upsampling module
    - The last module of the network is a bare full convolution, which alone takes up a sizeable portion of the decoder of processing time.
    - Each side branch has a Spatial Dropout with p = 0.01 for Stage 1 and p = 0.1 for stages afterwards.