How to determine the number of trainable parameters in a convolutional neural network

Created by Aldo Zaimi

March 28, 2019

Use Case: a simple CNN for MNIST digit classification.

- **Consider the following network architecture:**
  - Input layer that takes grayscale digit images of size 28×28.
  - Convolutional layer 1: 5×5 filters and 4 feature maps.
  - Max pooling layer 1: size 2×2.
  - Convolutional layer 2: 3×3 filters and 8 feature maps.
  - Max pooling layer 2: size 2×2.
  - Convolutional layer 3: 3×3 filters and 16 feature maps.
  - Flattening before the fully connected layer.
  - Fully connected layer: 16 units.
  - Output layer: 10 units (for digits 0 to 9).
  - No padding is used and all convolutions use a stride of 1.

1. **Determine the shape of the tensor after the first max pooling layer.**

   The input layer has a shape of $28 \times 28 \times 1$. After the first convolutional layer, the shape of the output is $24 \times 24 \times 4$ (the 5×5 filters reduce the size of the input image by 2 at the borders and we have 4 feature maps). The first max pooling layer downsamples the output of the previous layer by a factor of 2, resulting in a tensor of shape $12 \times 12 \times 4$.

2. **Determine the shape of the tensor after the third convolutional layer.**

   The shape of the tensor after the second convolutional layer is $10 \times 10 \times 8$ (we have 8 feature maps and the the 3×3 filters reduce the size of the tensor by 1 at the borders). The second max pooling layer downsamples the output of the previous layer by a factor of 2, resulting in a tensor of shape $5 \times 5 \times 8$. Finally, the shape of the tensor after the third convolutional layer is $3 \times 3 \times 16$ (we have 16 feature maps and the the 3×3 filters reduce the size of the tensor by 1 at the borders).

3. **How many units do we get when we flatten the output of the third convolutional layer?**

   The shape of the tensor after the third convolutional layer (just before the flattening) is $3 \times 3 \times 16$, so we get $(3 \times 3 \times 16)$ units $= 144$ units.

4. **Determine the number of trainable parameters of the network.**

   The total number of trainable parameters in the network is the sum of the trainable parameters of all the layers.

   - The input layers, the pooling layers and the dropout layers have no trainable parameters.
   - For the convolutional layers, the number of trainable parameters can be computed by $(n \times m \times l + 1) \times k$, where $(n \times m)$ is the filter size, l is the number of input feature maps and k is the number of output feature maps. The *+1* term in the equation takes into account the bias terms.
   - For the fully connected layers, the number of trainable parameters can be computed by $(n + 1) \times m$, where n is the number of input units and m is the number of output units. The *+1* term in the equation takes into account the bias terms.

   Let's count the number of trainable parameters in each layer:

   - Conv. layer 1: $(5 \times 5 \times 1 + 1) \times 4 = 104$ trainable parameters.
   - Conv. layer 2: $(3 \times 3 \times 4 + 1) \times 8 = 296$ trainable parameters.
   - Conv. layer 3: $(3 \times 3 \times 8 + 1) \times 16 = 1168$ trainable parameters.
   - Dense layer: $(144 + 1) \times 16 = 2320$ trainable parameters.
   - Output layer: $(16 + 1) \times 10 = 170$ trainable parameters.

   The network has a total of $104 + 296 + 1168 + 2320 + 170 = 4058$ trainable parameters.