Machine Learning

Neural Networks 3

Hao Tang

February 28, 2025

Commonly used neural architectures

- Feed-forward networks, fully-connected layers
  - MLP
  - ReLU networks
- Convolution neural networks
  - LeNet, AlexNet, VGG, ResNet
- Recurrent neural networks
  - RNN, GRU, LSTM
- Sequence-to-sequence models
- Transformers

Building blocks

- Affine transformation
- Nonlinearity (also known as activation functions)
- Normalization
- Convolution
- Pooling
- Skip connection
- Gating
- Attention

Affine transformation

- The operation

$$f(x) = Wx + b \tag{1}$$

  is called affine transformation, where W and b are trainable parameters.

- In pytorch, this is unfortunately called `torch.nn.Linear`.

Nonlinearity: Sigmoid function

- The operation

$$\sigma(x) = \begin{bmatrix} \frac{1}{1+\exp(-x_1)} \\ \frac{1}{1+\exp(-x_2)} \\ \vdots \\ \frac{1}{1+\exp(-x_d)} \end{bmatrix} \tag{2}$$

is called the sigmoid nonlinearity.

Nonlinearity: Sigmoid function

- The operation

$$\sigma(x) = \begin{bmatrix} \frac{1}{1+\exp(-x_1)} \\ \frac{1}{1+\exp(-x_2)} \\ \vdots \\ \frac{1}{1+\exp(-x_d)} \end{bmatrix} \tag{2}$$

  is called the sigmoid nonlinearity.

- The output range of sigmoid is $[0, 1]^d$.

Nonlinearity: Sigmoid function

- The operation

$$\sigma(x) = \begin{bmatrix} \frac{1}{1+\exp(-x_1)} \\ \frac{1}{1+\exp(-x_2)} \\ \vdots \\ \frac{1}{1+\exp(-x_d)} \end{bmatrix} \tag{2}$$

  is called the sigmoid nonlinearity.

- The output range of sigmoid is $[0, 1]^d$.
- This is unfortunately often written as $\sigma(x) = \frac{1}{1+\exp(-x)}$.

Nonlinearity: hyperbolic tangent

- The operation

$$f(x) = \begin{bmatrix} \tanh(x_1) \\ \tanh(x_2) \\ \vdots \\ \tanh(x_d) \end{bmatrix} \tag{3}$$

  is called the hyperbolic tangent nonlinearity, where

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{4}$$

Nonlinearity: hyperbolic tangent

- The operation

$$f(x) = \begin{bmatrix} \tanh(x_1) \\ \tanh(x_2) \\ \vdots \\ \tanh(x_d) \end{bmatrix} \tag{3}$$

  is called the hyperbolic tangent nonlinearity, where

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{4}$$

- The output range of sigmoid is $[-1, 1]^d$.

Nonlinearity: hyperbolic tangent

- The operation

$$f(x) = \begin{bmatrix} \tanh(x_1) \\ \tanh(x_2) \\ \vdots \\ \tanh(x_d) \end{bmatrix} \tag{3}$$

is called the hyperbolic tangent nonlinearity, where

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{4}$$

- The output range of sigmoid is $[-1, 1]^d$.
- This is unfortunately often written as $\tanh(x)$.

Nonlinearity: Rectified linear units (ReLU)

- The operation

$$\mathrm{ReLU}(x) = \begin{bmatrix} \max(0, x_1) \\ \max(0, x_2) \\ \vdots \\ \max(0, x_d) \end{bmatrix} \tag{5}$$

is called the rectified nonlinear unit (ReLU) (Nair and Hinton, 2010).

Nonlinearity: Rectified linear units (ReLU)

- The operation

$$\mathrm{ReLU}(x) = \begin{bmatrix} \max(0, x_1) \\ \max(0, x_2) \\ \vdots \\ \max(0, x_d) \end{bmatrix} \tag{5}$$

is called the rectified nonlinear unit (ReLU) (Nair and Hinton, 2010).

- The output range of sigmoid is $\mathbb{R}^d_{\geq 0}$.

Nonlinearity: Rectified linear units (ReLU)

- The operation

$$\text{ReLU}(x) = \begin{bmatrix} \max(0, x_1) \\ \max(0, x_2) \\ \vdots \\ \max(0, x_d) \end{bmatrix} \tag{5}$$

  is called the rectified nonlinear unit (ReLU) (Nair and Hinton, 2010).

- The output range of sigmoid is $\mathbb{R}^d_{\geq 0}$.
- This is unfortunately often written as $\text{ReLU}(x) = \max(0, x)$.

Feed-forward networks

- A neural network of the form

$$F(x) = W_\ell(\cdots \sigma(W_2 \sigma(W_1 x + b_1) + b_2)) + b_\ell \tag{6}$$

  is called a multi-layer perceptron (MLP).
- A neural network of the form

$$F(x) = W_\ell(\cdots \mathrm{ReLU}(W_2 \mathrm{ReLU}(W_1 x + b_1) + b_2)) + b_\ell \tag{7}$$

  is called a ReLU network.

Feed-forward networks

- A neural network of the form

$$F(x) = W_\ell(\cdots \sigma(W_2\sigma(W_1 x + b_1) + b_2)) + b_\ell \tag{6}$$

  is called a multi-layer perceptron (MLP).
- A neural network of the form

$$F(x) = W_\ell(\cdots \mathrm{ReLU}(W_2\mathrm{ReLU}(W_1 x + b_1) + b_2)) + b_\ell \tag{7}$$

  is called a ReLU network.
- They are unfortunately often called feed-forward networks (FFNs).
- An affine transformationn with a nonlinearity is unfortunately often called a fully-connected layer (FC layer).

Normalization: Batch normalization

- Standardization (i.e., z normalization) of the input to the network typically brings optimization benefits.

Normalization: Batch normalization

- Standardization (i.e., z normalization) of the input to the network typically brings optimization benefits.
- Batch normalization (Ioffe and Szegedy, 2015) is defined as

$$f(x) = \frac{x - \mu}{\sqrt{\sigma^2}} \quad \text{where} \quad \mu = \frac{1}{B}\sum_{i=1}^{B} x_i, \quad \sigma^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2 - \mu^2 \tag{8}$$

where x_i is the i-th sample in a batch of size B.

Normalization: Layer normalization

- Layer normalization (Ba *et al.*, 2016) is defined as

$$f(x) = \begin{bmatrix} \frac{[x]_1 - \mu}{\sqrt{\sigma^2}} \\ \frac{[x]_2 - \mu}{\sqrt{\sigma^2}} \\ \vdots \\ \frac{[x]_d - \mu}{\sqrt{\sigma^2}} \end{bmatrix} \quad \text{where} \quad \mu = \frac{1}{d}\sum_{i=1}^{d}[x]_i, \quad \sigma^2 = \frac{1}{d}\sum_{i=1}^{d}[x]_i^2 - \mu^2 \tag{9}$$

where $[x]_i$ is the i-th coordinate of the vector x.

Convolution

- The 1D convolution of x and w is defined as

$$y_t = \sum_{i=1}^{d} x_i w_{t-i}, \tag{10}$$

where w is the learnable parameter (often called a filter).

Convolution

- The 1D convolution of x and w is defined as

$$y_t = \sum_{i=1}^{d} x_i w_{t-i}, \tag{10}$$

where w is the learnable parameter (often called a filter).

- The 1D cross correlation of x and w is defined as

$$y_t = \sum_{i=1}^{d} x_i w_{t+i}. \tag{11}$$

Convolution

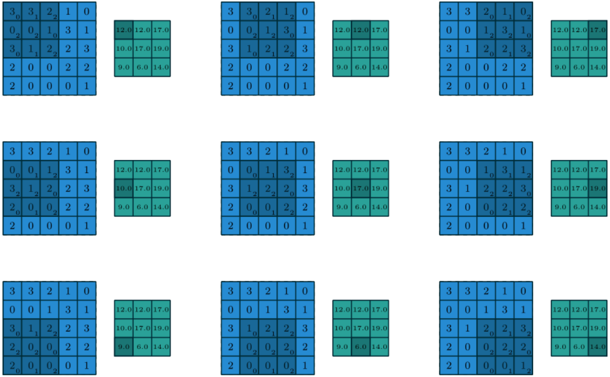


Figure 1.1: Computing the output values of a discrete convolution.

(Dumoulin and Visin, 2018)

Convolution

- Cross correlation is linear in w, and it can be implemented with an linear transformation.

Convolution

- Cross correlation is linear in w, and it can be implemented with an linear transformation.
- In pytorch, convolution (e.g., `torch.nn.Conv1d`) is unfortunately implemented with as cross correlation and with affine transformation.

Pooling

- Max pooling

$$y_t = \max_{i=1,\dots,d} x_{t+i}. \tag{12}$$

- Mean pooling

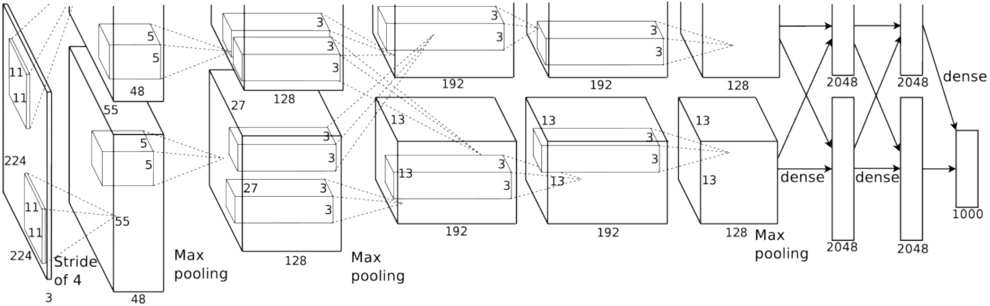
$$y_t = \frac{1}{d} \sum_{i=1}^{d} x_{t+i}. \tag{13}$$

- Pooling is useful for learning whether there exists something.

Convolutional neural networks (CNNs)

- A convolutional neural network is a stack of convolutions and ReLUs.
- Depending on the task, there might also be max pooling.

Convolutional neural networks (CNNs)



(Krizhevsky *et al.*, 2012)

Skip connections

- A skip connection (He *et al.*, 2016) is of the form

$$f(x) = x + T(x) \tag{14}$$

for some other transformation T.

Skip connections

- A skip connection (He *et al.*, 2016) is of the form

$$f(x) = x + T(x) \tag{14}$$

  for some other transformation T.

- Due to the form,

$$\frac{\partial f}{\partial x} = \mathbf{1}_d + \frac{\partial T}{\partial x}, \tag{15}$$

  where $\mathbf{1}_d$ is a d-dimensional all-one vector. There is always some gradient after adding the skip connection.

Skip connections

- A skip connection (He *et al.*, 2016) is of the form

$$f(x) = x + T(x) \tag{14}$$

  for some other transformation T.

- Due to the form,

$$\frac{\partial f}{\partial x} = \mathbf{1}_d + \frac{\partial T}{\partial x}, \tag{15}$$

  where $\mathbf{1}_d$ is a d-dimensional all-one vector. There is always some gradient after adding the skip connection.

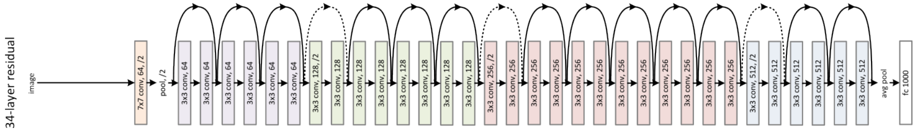
- Since $T(x) = f(x) - x$, the transformation T whatever there is other than the identity x. A skip connection is also called a residual connection.

ResNet

- A ResNet (He *et al.*, 2016) is a regular CNN with skip connections every several layers.

ResNet

- A ResNet (He *et al.*, 2016) is a regular CNN with skip connections every several layers.



(He *et al.*, 2016)

Gating

- The sigmoid activation can also be used as a gating function.
- The output of the sigmoid can be thought of as the probability of the gate being open.

Gating

- The sigmoid activation can also be used as a gating function.
- The output of the sigmoid can be thought of as the probability of the gate being open.
- For example, the gated linear unit (Dauphin *et al.*, 2017) is defined as

$$f(x) = \alpha T(x) \quad \text{where} \quad \alpha = \sigma(Wx + b) \tag{16}$$

for some other transformation T.

Gating

- The sigmoid activation can also be used as a gating function.
- The output of the sigmoid can be thought of as the probability of the gate being open.
- For example, the gated linear unit (Dauphin *et al.*, 2017) is defined as

$$f(x) = \alpha T(x) \quad \text{where} \quad \alpha = \sigma(Wx + b) \tag{16}$$

  for some other transformation T.
- We can also have a softer skip connection

$$f(x) = (1 - \alpha)x + \alpha T(x) \quad \text{where} \quad \alpha = \sigma(Wx + b) \tag{17}$$

  for some other transformation T.

Recurrent neural networks (RNNs)

- Some data, e.g., text and speech, comes in varying lengths.
- The input is a sequence $x_1, \dots, x_T$ and the output is a sequence $y_1, \dots, y_T$.

Recurrent neural networks (RNNs)

- Some data, e.g., text and speech, comes in varying lengths.
- The input is a sequence $x_1, \ldots, x_T$ and the output is a sequence $y_1, \ldots, y_T$.
- An Elman network (Elman, 1990) has the form

$$h_t = \sigma(Vh_{t-1} + Ux_t + b_1) \tag{18}$$

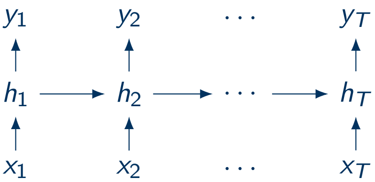
$$y_t = Wh_t + b_2 \tag{19}$$

Recurrent neural networks (RNNs)

- Some data, e.g., text and speech, comes in varying lengths.
- The input is a sequence $x_1, \ldots, x_T$ and the output is a sequence $y_1, \ldots, y_T$.
- An Elman network (Elman, 1990) has the form

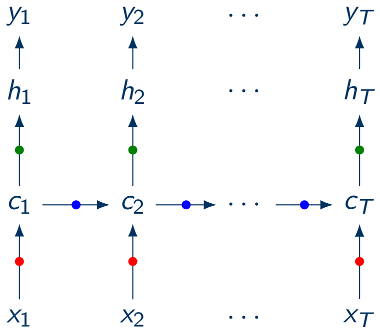
$$h_t = \sigma(V h_{t-1} + U x_t + b_1) \quad (18)$$

$$y_t = W h_t + b_2 \quad (19)$$



Recurrent neural networks (RNNs)

- The recurrence is gated in long short-term memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997).
  - input gate (red dot)
  - forget gate (blue dot)
  - output gate (green dot)



Attention mechanism

- Given a query q, keys $k_1, \ldots, k_T$, and values $v_1, \ldots, v_T$, the attention mechanism (Bahdanau *et al.*, 2015) is defined as

$$f(q, k_1, \ldots, k_T, v_1, \ldots, v_T) = \sum_{i=1}^{T} \frac{\exp(k_i^\top q)}{\sum_{j=1}^{T} \exp(k_j^\top q)} v_i \tag{20}$$

$$\tag{21}$$

Attention mechanism

- Given a query q, keys $k_1, \ldots, k_T$, and values $v_1, \ldots, v_T$, the attention mechanism (Bahdanau *et al.*, 2015) is defined as

$$
\begin{aligned}
f(q, k_1, \ldots, k_T, v_1, \ldots, v_T) &= \sum_{i=1}^{T} \frac{\exp(k_i^\top q)}{\sum_{j=1}^{T} \exp(k_j^\top q)} v_i && (20) \\
&= \mathrm{softmax}(qK^\top)V, && (21)
\end{aligned}
$$

where

$$
K = \begin{bmatrix} — & k_1 & — \\ — & k_2 & — \\ & \vdots & \\ — & k_T & — \end{bmatrix} \quad V = \begin{bmatrix} — & v_1 & — \\ — & v_2 & — \\ & \vdots & \\ — & v_T & — \end{bmatrix}. \qquad (22)
$$

Attention mechanism

- The term $\text{softmax}(qK)$ is sometimes called the attention weights.

Attention mechanism

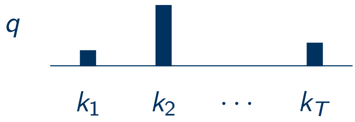
- The term softmax(qK) is sometimes called the attention weights.

q

$k_1 \quad k_2 \quad \cdots \quad k_T$

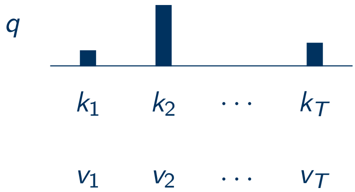
Attention mechanism

- The term softmax(qK) is sometimes called the attention weights.



Attention mechanism

- The term softmax(qK) is sometimes called the attention weights.



Sequence-to-sequence model

- The input is a sequence $x_1, \ldots, x_T$ and the output is a sequence $y_1, \ldots, y_T$.

Sequence-to-sequence model

- The input is a sequence $x_1, \ldots, x_T$ and the output is a sequence $y_1, \ldots, y_T$.
- A sequence-to-sequence model (or seq2seq) (Sutskever *et al.*, 2014) is a loop

$$q_i = \text{update}(q_{i-1}, y_{i-1}, x_1, \ldots, x_T) \tag{23}$$

$$y_i = f(q_i) \tag{24}$$

Sequence-to-sequence model

- The input is a sequence $x_1, \dots, x_T$ and the output is a sequence $y_1, \dots, y_T$.
- A sequence-to-sequence model (or seq2seq) (Sutskever *et al.*, 2014) is a loop

$$q_i = \text{update}(q_{i-1}, y_{i-1}, x_1, \dots, x_T) \tag{23}$$

$$y_i = f(q_i) \tag{24}$$

- A seq2seq is often equipped with an attention mechanism and the loop becomes

$$q_i = \text{update}(q_{i-1}, y_{i-1}) \tag{25}$$

$$z_i = \text{attend}(q_i, h_1, \dots, h_T, h_1, \dots, h_T) \tag{26}$$

$$y_i = f(z_i) \tag{27}$$

where $h_1, \dots, h_T = \text{encode}(x_1, \dots, x_T)$.

Sequence-to-sequence model

- The input is a sequence $x_1, \ldots, x_T$ and the output is a sequence $y_1, \ldots, y_T$.
- A sequence-to-sequence model (or seq2seq) (Sutskever *et al.*, 2014) is a loop

$$q_i = \text{update}(q_{i-1}, y_{i-1}, x_1, \ldots, x_T) \tag{23}$$
$$y_i = f(q_i) \tag{24}$$

- A seq2seq is often equipped with an attention mechanism and the loop becomes

$$q_i = \text{update}(q_{i-1}, y_{i-1}) \tag{25}$$
$$z_i = \text{attend}(q_i, h_1, \ldots, h_T, h_1, \ldots, h_T) \tag{26}$$
$$y_i = f(z_i) \tag{27}$$

  where $h_1, \ldots, h_T = \text{encode}(x_1, \ldots, x_T)$.
- The loop is often refer to as the decoder, and seq2seq is unfortunately often called a encoder-decoder model.

Self-attention

- When applying attention on

$$q_t = W_1 x_t + b_1 \tag{28}$$
$$k_t = W_2 x_t + b_2 \tag{29}$$
$$v_t = W_3 x_t + b_3 \tag{30}$$

  we are using the transformation of the input to attend to itself; hence the name self-attention (Vaswani *et al.*, 2017).

- To contrast with self-attention, regular attention is often called cross attention.

Transformer

- A Transformer block is defined as

$$\mathrm{FC}(\mathrm{FC}(H)) + H \tag{31}$$

where $H = \mathrm{attend}(W_1 X, W_2 X, W_3 X) + X$.

Transformer

- A Transformer block is defined as

$$\text{FC}(\text{FC}(H)) + H \tag{31}$$

  where $H = \text{attend}(W_1 X, W_2 X, W_3 X) + X$.

- A Transformer (Vaswani *et al.*, 2017) is a seq2seq where both the encoder and the decoder consist of a sequence of Transformer blocks.

Questions to think about

- If a simple MLP is already an universal approximator, why do we need convolution, recurrence, and attention?
- Even though some layers have intuitions attached, after training, are they learning what is intended?
- If we want to solve a new problem, how do we know what layer types to use?
- If there are differences among different layers, why are people using Transformers more than other model architectures these days?
- Are there things that are easy to learn for one layer type but hard to learn for another?