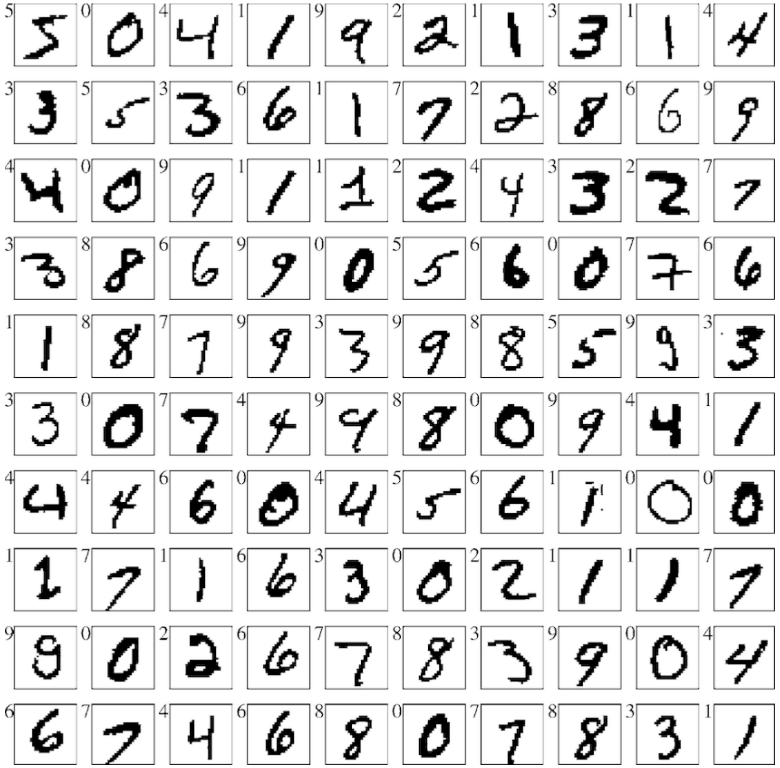
Neural Networks: Basics

Shusen Wang

Revisit Softmax Classifier

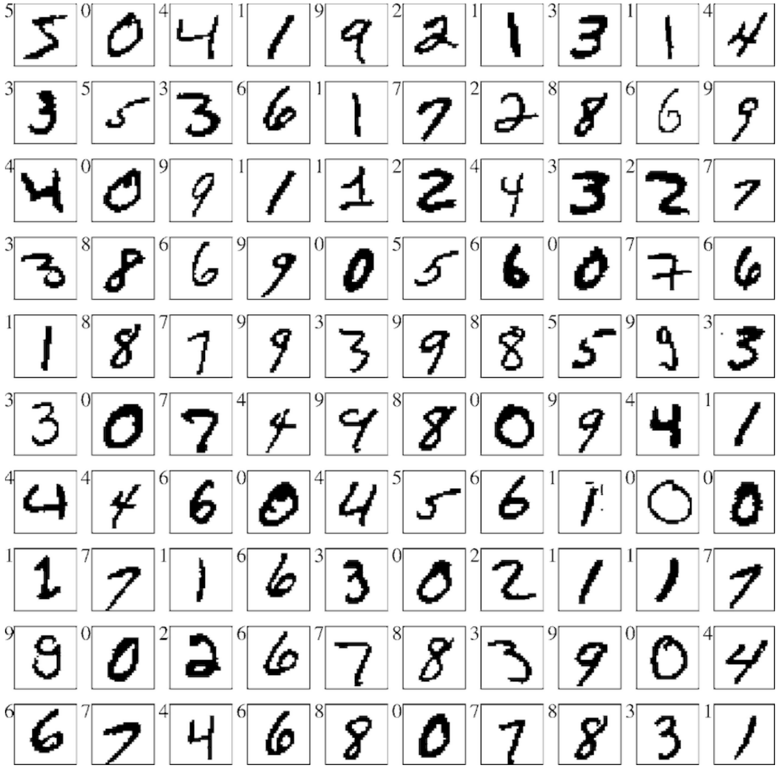
Train a Softmax Classifier



The MNIST Dataset

- $n = 60{,}000$ training samples $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)$.
- Each $\mathbf{x}_j$ is a 28×28 image.
- Each y_j is an integer in $\{0, 1, 2, \cdots, 9\}$.

Train a Softmax Classifier



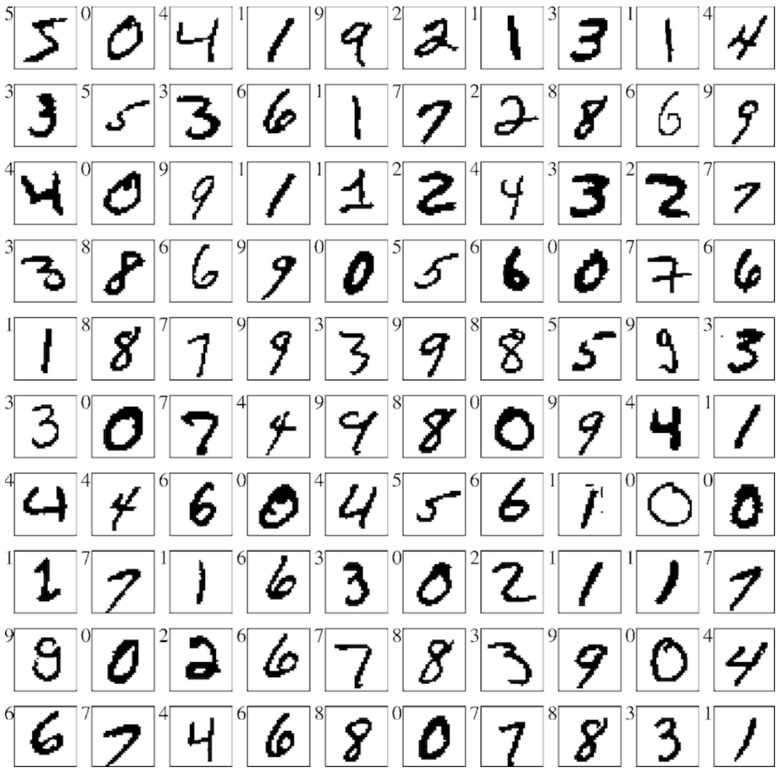
The MNIST Dataset

- $n = 60{,}000$ training samples $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)$.
- Each $\mathbf{x}_j$ is a 28×28 image.
- Each y_j is an integer in $\{0, 1, 2, \cdots, 9\}$.

Task: multi-class classification

- Given a 28×28 image, predict the digit.
- Learn a function $\mathbf{f}$: $\mathbb{R}^{28 \times 28} \mapsto \mathbb{R}^{10}$.
- The i-th entry of $\mathbf{f}(\mathbf{x})$ indicates how likely the image $\mathbf{x}$ is the digit i.

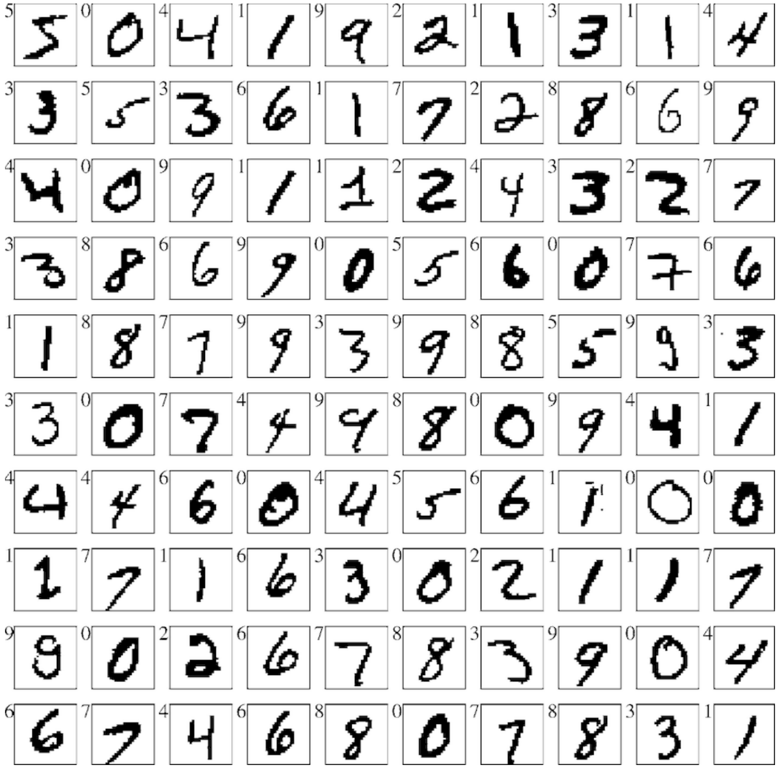
Train a Softmax Classifier



Linear model: softmax classifier

- Vectorize each 28×28 image to a 784-dim vector.
- Add a feature of all ones. (So $\mathbf{x}$ becomes 785-dim.)

Train a Softmax Classifier

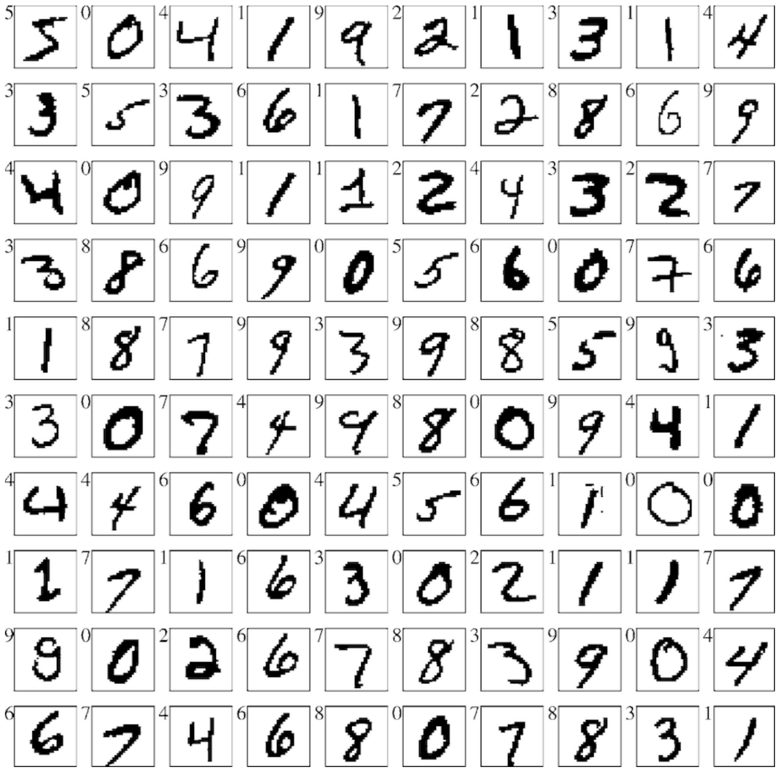


Linear model: softmax classifier

- Vectorize each 28×28 image to a 784-dim vector.
- Add a feature of all ones. (So $\mathbf{x}$ becomes 785-dim.)
- Let $\mathbf{W} \in \mathbb{R}^{10\times 785}$ contain the parameters.
- Let $\mathbf{z} = \mathbf{W}\mathbf{x} \in \mathbb{R}^{10}$.
- Output a 10-dim vector:

$$\mathbf{f}(\mathbf{x}) = \mathrm{SoftMax}(\mathbf{z}).$$

Train a Softmax Classifier



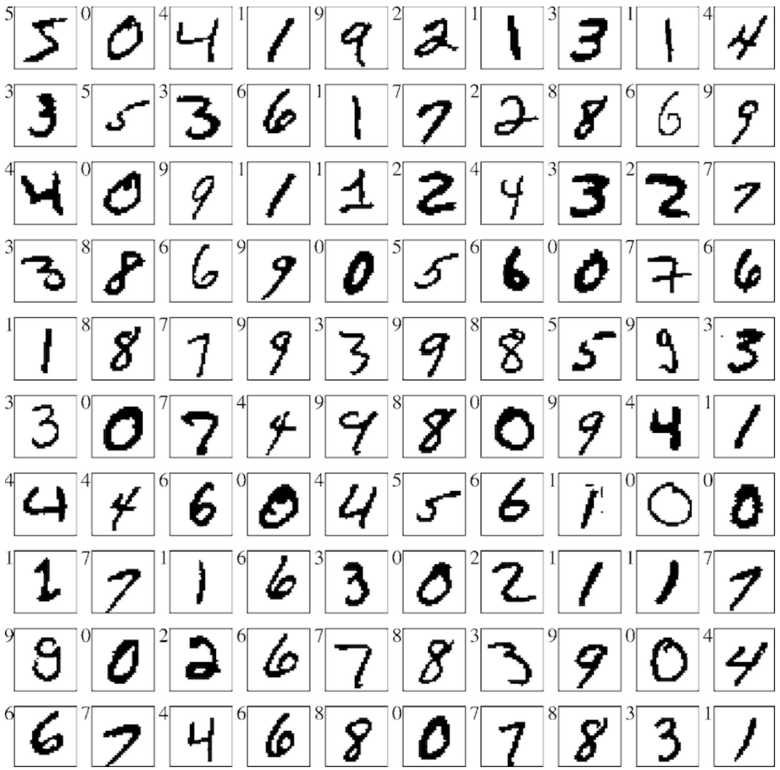
Linear model: softmax classifier

- Vectorize each 28×28 image to a 784-dim vector.
- Add a feature of all ones. (So $\mathbf{x}$ becomes 785-dim.)
- Let $\mathbf{W} \in \mathbb{R}^{10\times 785}$ contain the parameters.
- Let $\mathbf{z} = \mathbf{W}\mathbf{x} \in \mathbb{R}^{10}$.
- Output a 10-dim vector:

$$\mathbf{f}(\mathbf{x}) = \mathrm{SoftMax}(\mathbf{z}).$$

$$\mathrm{SoftMax}(\mathbf{z}) = \frac{1}{\sum_{i=0}^{9} \exp(z_i)} [\exp(z_0), \cdots, \exp(z_9)]$$

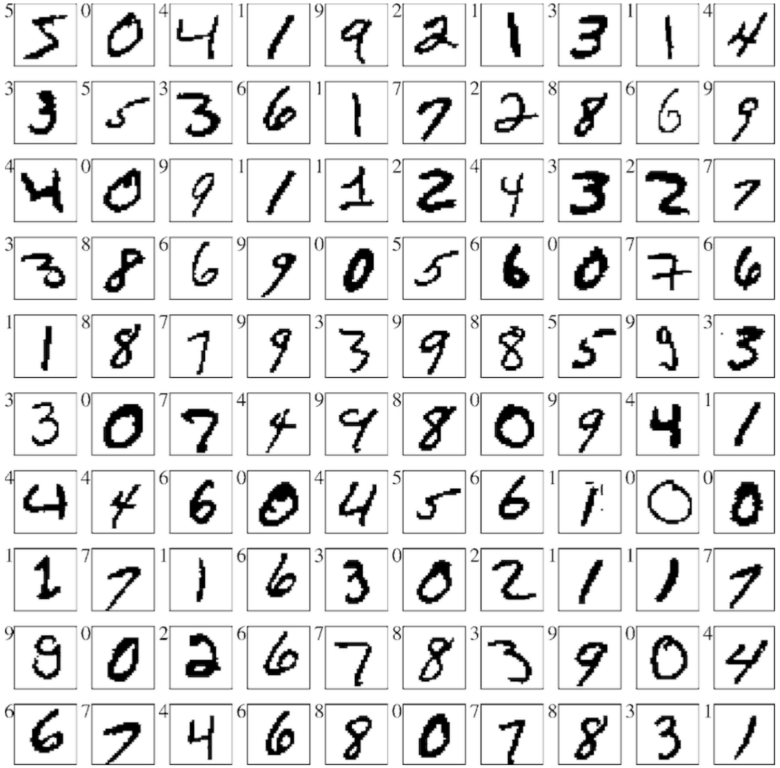
Train a Softmax Classifier



Learn W $\in \mathbb{R}^{10\times 785}$ from the training data

- One-hot encode of the labels
  - Originally, a label is a scalar in $\{0, 1, 2, \cdots, 9\}$.
  - The one-hot encode $\mathbf{y}$ is a 10-dim vector $\{0,1\}^{10}$.
  - E.g., the one-hot encode of 2 is $[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]$.

Train a Softmax Classifier

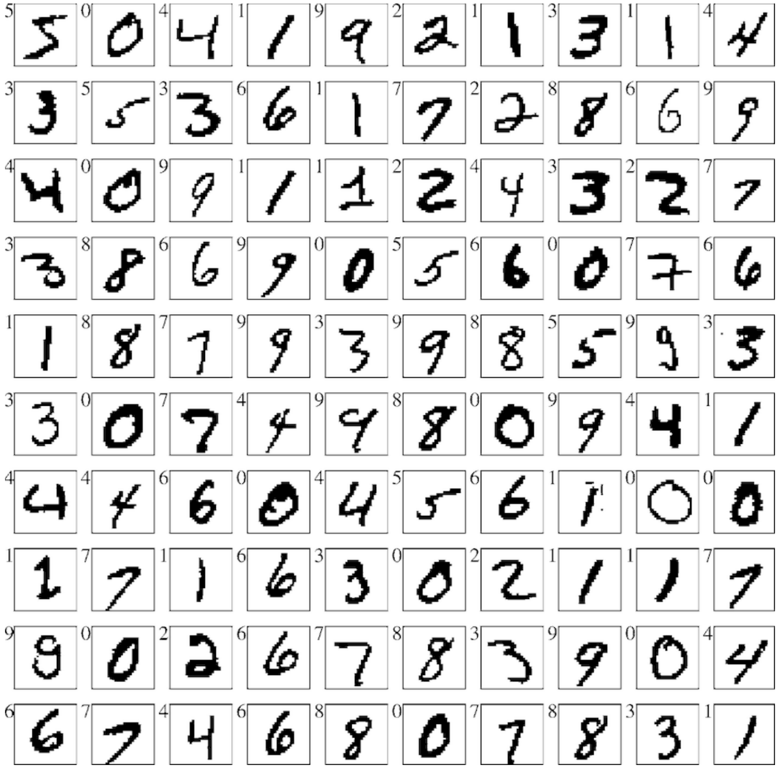


Learn W $\in \mathbb{R}^{10\times 785}$ from the training data

- One-hot encode of the labels
  - Originally, a label is a scalar in $\{0, 1, 2, \cdots, 9\}$.
  - The one-hot encode $\mathbf{y}$ is a 10-dim vector $\{0,1\}^{10}$.
  - E.g., the one-hot encode of 2 is $[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]$.
- Cross-entropy loss:

$$\text{CrossEntropy}(\mathbf{y}, \mathbf{f}) = -\sum_{i=0}^{9} y_i \cdot \log(f_i).$$

Train a Softmax Classifier



Learn W $\in \mathbb{R}^{10\times 785}$ from the training data

- One-hot encode of the labels
  - Originally, a label is a scalar in $\{0, 1, 2, \cdots, 9\}$.
  - The one-hot encode $\mathbf{y}$ is a 10-dim vector $\{0,1\}^{10}$.
  - E.g., the one-hot encode of 2 is $[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]$.
- Cross-entropy loss:

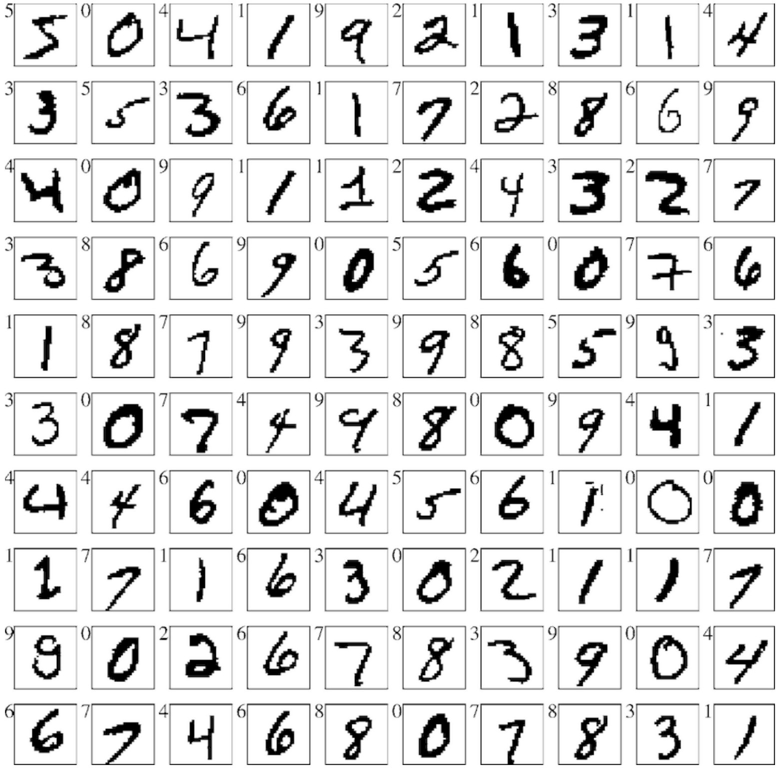
$$\text{CrossEntropy}(\mathbf{y}, \mathbf{f}) = -\sum_{i=0}^{9} y_i \cdot \log(f_i).$$

- Solve the optimization model:

$$\mathbf{W}^{\star} = \underset{\mathbf{W}}{\text{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \text{CrossEntropy}\Big(\mathbf{y}_j, \mathbf{f}\big(\mathbf{x}_j\big)\Big) \right\}.$$

W is the parameter of $\mathbf{f}$

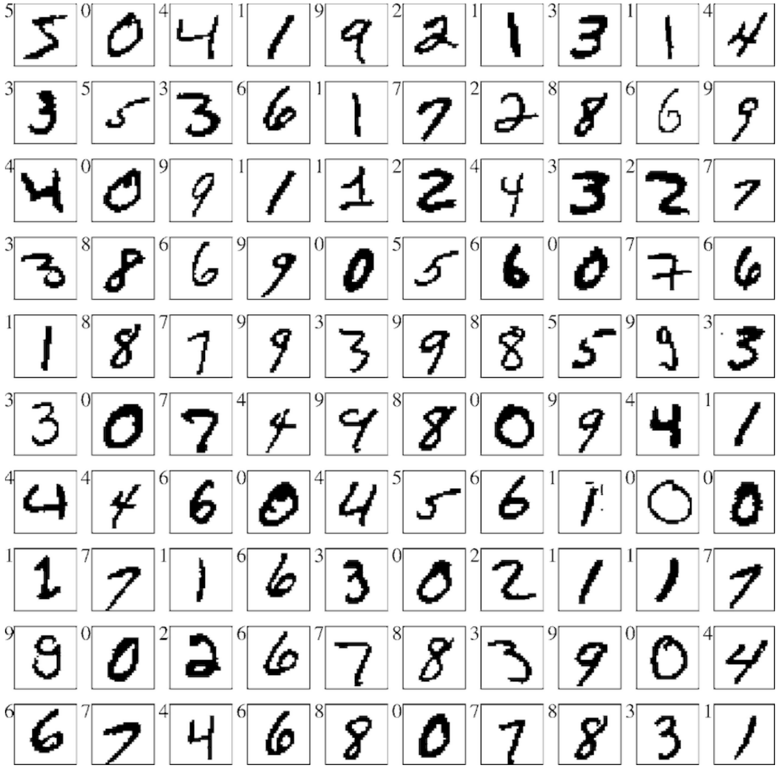
Train a Softmax Classifier



Make prediction for a test sample $\mathbf{x}'$

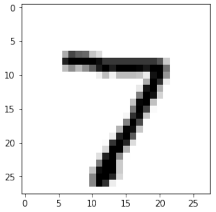
- Now we have $\mathbf{W}^{\star} \in \mathbb{R}^{10\times 785}$.
- For a test sample $\mathbf{x}'$, compute $\mathbf{z} = \mathbf{W}^{\star}\mathbf{x}' \in \mathbb{R}^{10}$.
- Make prediction by $\operatorname{argmax} \mathbf{z}$.
  - If the 7-th entry of $\mathbf{z}$ is the largest, then the model thinks the image is digit "7".

Train a Softmax Classifier



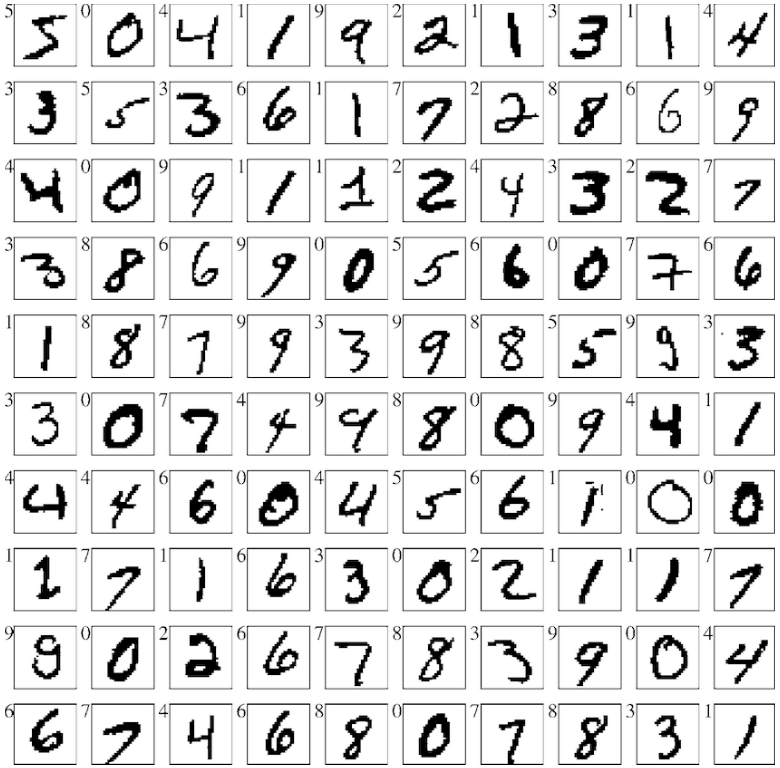
Make prediction for a test sample $\mathbf{x}'$

- Now we have $\mathbf{W}^{\star} \in \mathbb{R}^{10\times 785}$.
- For a test sample $\mathbf{x}'$, compute $\mathbf{z} = \mathbf{W}^{\star}\mathbf{x}' \in \mathbb{R}^{10}$.
- Make prediction by $\operatorname{argmax} \mathbf{z}$.
  - If the 7-th entry of $\mathbf{z}$ is the largest, then the model thinks the image is digit "7".



$$\mathbf{z} = [\,-55.7, -141.4\,, 18.1\,, 188.3\,, -91.3\,, -26.8\,, -183.6\,, \boxed{411.2}\,, -142.1\,, 96.2]$$

Train a Softmax Classifier



Results

- The training set has $60{,}000$ samples.
- The test set has $10{,}000$ samples.

- The accuracy on the training set is 84.64%.
- The accuracy on the test set is 83.58%.

- Not too bad!
- The accuracy of a random guess is merely 10%.

Train a Softmax Classifier: Re-cap

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input**: vector $\mathbf{x} \in \mathbb{R}^{785}$.
- $\mathbf{z} = \mathbf{W}\,\mathbf{x} \in \mathbb{R}^{10}$.
- **Output**: $\mathbf{f}(\mathbf{x}) = \mathrm{SoftMax}(\mathbf{z})$.

Trainable parameters: $\mathbf{W} \in \mathbb{R}^{10\times 785}$

Train a Softmax Classifier: Re-cap

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input**: vector $\mathbf{x} \in \mathbb{R}^{785}$.
- $\mathbf{z} = \mathbf{W}\,\mathbf{x} \in \mathbb{R}^{10}$.
- **Output**: $\mathbf{f}(\mathbf{x}) = \mathrm{SoftMax}(\mathbf{z})$.

Trainable parameters: $\mathbf{W} \in \mathbb{R}^{10 \times 785}$

Train the function by empirical risk minimization (ERM):

- **Training set:** $(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n) \in \mathbb{R}^{785} \times \mathbb{R}^{10}$.
- **Loss function:** $\mathrm{CrossEntropy}(\mathbf{y}, \mathbf{f}) = -\sum_{i=1}^{10} y_i \cdot \log(\mathbf{f}(\mathbf{x})_i)$.
- **Solve ERM:** $\underset{\mathbf{W}}{\mathrm{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \mathrm{CrossEntropy}\left(\mathbf{y}_j, \mathbf{f}\left(\mathbf{x}_j\right)\right) \right\}$.

Train a Softmax Classifier: Re-cap

- **How to solve** $\underset{\mathbf{W}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \operatorname{CrossEntropy}\left(\mathbf{y}_j, \mathbf{f}(\mathbf{x}_j) \right) \right\}$?

- **Stochastic gradient descent (SGD) with momentum** repeats:
  1. Randomly pick j from $\{1, 2, \cdots, n\}$.
  2. Evaluate the gradient $\mathbf{G}_j = \frac{\partial \operatorname{CrossEntropy}\left(\mathbf{y}_j, \mathbf{f}(\mathbf{x}_j) \right)}{\partial \, \mathbf{W}} \Big|_{\mathbf{W}=\mathbf{W}_{\text{old}}}$.
  3. Update the momentum: $\mathbf{V}_{\text{new}} = \beta \mathbf{V}_{\text{old}} + \mathbf{G}_j$.
  4. Update $\mathbf{W}$ by $\mathbf{W}_{\text{new}} \leftarrow \mathbf{W}_{\text{old}} - \alpha \, \mathbf{V}_{\text{new}}$.

Fully-Connected Neural Network
(Multi-layer Perceptron)

Softmax Classifier

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input**: vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(1)} = \mathrm{SoftMax}\left(\mathbf{z}^{(1)}\right) \in \mathbb{R}^{d_1}$.
- **Output**: $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(1)}$.

Trainable parameter:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{10 \times 785}$.

From Linear Model to Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Trainable parameters:

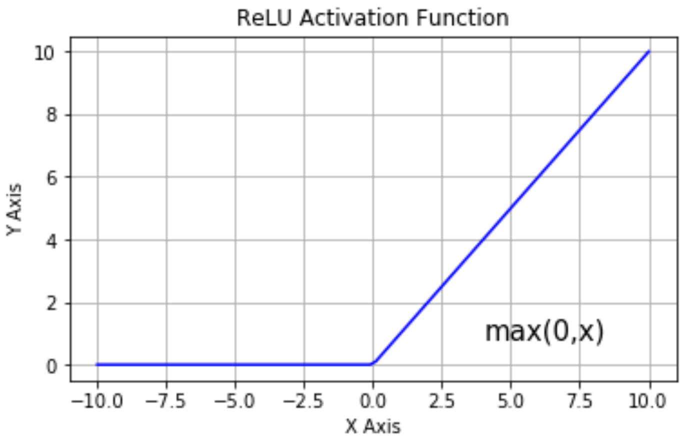
- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,

From Linear Model to Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \boxed{\max\{\mathbf{0}, \ \mathbf{z}^{(1)}\}} \in \mathbb{R}^{d_1}$.

ReLU (activation function)



Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,

From Linear Model to Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.

- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Hidden Layer 1

Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,

From Linear Model to Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Hidden Layer 1

- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.

It should be $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} + \mathbf{b}^{(1)}$ in practice.
I leave out $\mathbf{b} \in \mathbb{R}^{d_2}$ in the slides for simplicity.

Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,
- $\mathbf{W}^{(1)} \in \mathbb{R}^{d_2 \times d_1}$,

From Linear Model to Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input**: vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.

- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Hidden Layer 1

- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.

Hidden Layer 2

Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,
- $\mathbf{W}^{(1)} \in \mathbb{R}^{d_2 \times d_1}$,

From Linear Model to Neural Network

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.

Hidden Layer 1

- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Hidden Layer 2

- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \ \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.

Output Layer

- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \text{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.

Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,
- $\mathbf{W}^{(1)} \in \mathbb{R}^{d_2 \times d_1}$,
- $\mathbf{W}^{(2)} \in \mathbb{R}^{10 \times d_2}$.

From Linear Model to Neural Network

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.

- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.

Hidden Layer 1

- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.

Hidden Layer 2

- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.

Output Layer

- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

Trainable parameters:

- $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,
- $\mathbf{W}^{(1)} \in \mathbb{R}^{d_2 \times d_1}$,
- $\mathbf{W}^{(2)} \in \mathbb{R}^{10 \times d_2}$.

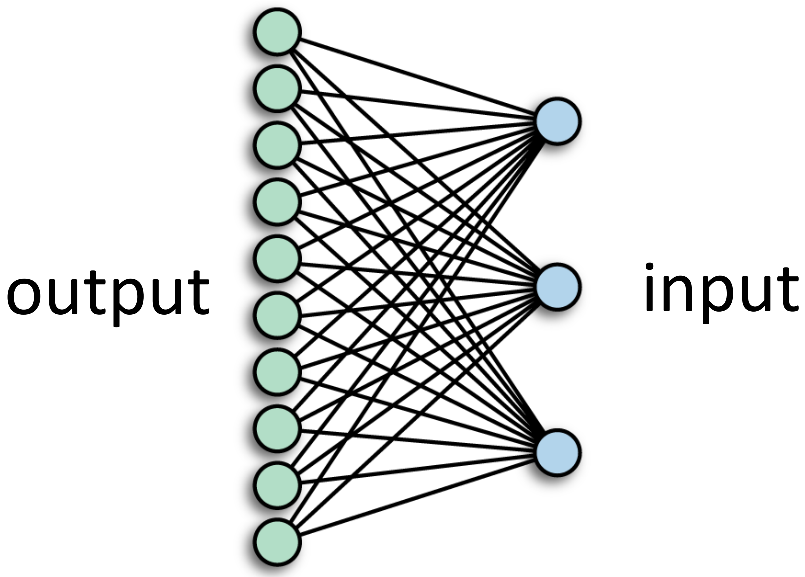
Fully-Connected Layer

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathrm{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

"Fully-Connected" or "Dense" Layer

Each entry of $\mathbf{z}^{(1)}$ depends on (i.e., connected to) all the entries of $\mathbf{x}^{(0)}$.



Activation Functions

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\boxed{\mathbf{x}^{(1)} = \max\{\mathbf{0},\ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}.}$ ReLU
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\boxed{\mathbf{x}^{(2)} = \max\{\mathbf{0},\ \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}.}$ ReLU
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\boxed{\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}.}$ SoftMax
- **Output:** $\mathrm{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

Activation Functions

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\boxed{\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}.}$
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

SoftMax

- Use SoftMax because this is a **multi-class classification** problem.

Activation Functions

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

SoftMax

- Use Sigmoid or tanh function for **binary classification problem.**

- **For regression**:
  - No activation function, if the labels are in $\mathbb{R}$.
  - Use ReLU if the labels are positive.

- Use SoftMax because this is a **multi-class classification** problem.

Activation Functions

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$. ReLU
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$. ReLU
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

Question: Why bothering using ReLU?

- Without the two activation functions, $\mathbf{z}^{(3)}$ would be a linear function of $\mathbf{x}^{(0)}$.
- A linear function can be represented by $\mathbf{z}^{(3)} = \mathbf{W}\mathbf{x}^{(0)}$.
- The neural network would be equally expressive as a linear model!!!

Gradient and Backpropagation

Train the Neural Network

Define a function f: $\mathbb{R}^{785} \mapsto \mathbb{R}^{10}$**:**

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

Train the Neural Network

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

Build an optimization model:

$$\underset{\mathbf{W}^{(0)},\mathbf{W}^{(1)},\mathbf{W}^{(2)}}{\mathrm{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \boxed{\mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j \right)} \right\}$$

E.g., the cross-entropy loss

Train the Neural Network

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to solve

$$\underset{\mathbf{W}^{(0)},\mathbf{W}^{(1)},\mathbf{W}^{(2)}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \mathrm{Loss}\big(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j \big) \right\} ?$$

Stochastic gradient descent (SGD)

Train the Neural Network

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \ \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to solve

$$\underset{\mathbf{W}^{(0)},\mathbf{W}^{(1)},\mathbf{W}^{(2)}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{n} \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \ \mathbf{y}_j \right) \right\} ?$$

Stochastic gradient descent (SGD):

- Randomly pick j from $\{1, 2, \cdots, n\}$.
- Compute the stochastic gradient w.r.t. $\mathbf{W}^{(0)}$ at the current iteration $\mathbf{W}_{\text{old}}^{(0)}$:

$$\mathbf{g}_j^{(0)} = \frac{\partial \, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j \right)}{\partial \mathbf{W}^{(0)}} \Bigg|_{\mathbf{W}^{(0)} = \mathbf{W}_{\text{old}}^{(0)}} .$$

- Update $\mathbf{W}^{(0)}$: $\mathbf{W}_{\text{new}}^{(0)} = \mathbf{W}_{\text{old}}^{(0)} - \alpha \, \mathbf{g}_j^{(0)}$.
- Do the same for $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial \, \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)$.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

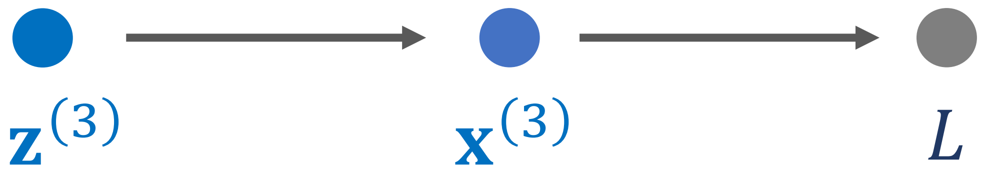
- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\right)$.
- Compute $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}}$.

10-dim vector



Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\right)$.
- Compute $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}}$.

$\mathbf{z}^{(3)}$ is a function of $\mathbf{z}^{(2)}$ and $\mathbf{W}^{(2)}$.

Apply the chain rule.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to compute $\frac{\partial \, \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)$.
- Compute $\boxed{\frac{\partial L}{\partial \mathbf{z}^{(3)}}}$.
- $\frac{\partial L}{\partial \mathbf{z}^{(2)}} = \frac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \boxed{\frac{\partial L}{\partial \mathbf{z}^{(3)}}}$, $\frac{\partial L}{\partial \mathbf{W}^{(2)}} = \frac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \boxed{\frac{\partial L}{\partial \mathbf{z}^{(3)}}}$.

$\mathbf{z}^{(3)}$ is a function of $\mathbf{z}^{(2)}$ and $\mathbf{W}^{(2)}$.

Apply the chain rule.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$ **?**

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\right)$.
- Compute $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}, \qquad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}.$

Then free $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$ from memory.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$ **?**

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(2)}}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.

Use it to update $\mathbf{W}^{(2)}$ (e.g., by SGD).

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial \, \mathrm{Loss}(\, \mathbf{f}(\mathbf{x}_j), \mathbf{y}_j \,)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}(\, \mathbf{f}(\mathbf{x}_j), \; \mathbf{y}_j \,)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(2)}}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.

Then free $\dfrac{\partial L}{\partial \mathbf{W}^{(2)}}$ from memory.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0},\ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0},\ \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$ **?**

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\right)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(2)}}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(2)}}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(2)}}}$.

Apply the chain rule again.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0},\ \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0},\ \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}(\,\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\,)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}(\,\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\,)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(2)}}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$.

Free $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$ from memory.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\right)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$, $\quad \boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(1)}}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$.

Use it to update $\mathbf{W}^{(1)}$ (e.g., by SGD).

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial \, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$, $\quad \boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(1)}}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$.

Free $\dfrac{\partial L}{\partial \mathbf{W}^{(1)}}$ from memory.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}\left(\mathbf{z}^{(3)}\right) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}\left(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\right)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(1)}}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$.
- $\dfrac{\partial L}{\partial \mathbf{W}^{(0)}} = \dfrac{\partial \mathbf{z}^{(1)}}{\partial \mathbf{W}^{(0)}} \boxed{\dfrac{\partial L}{\partial \mathbf{z}^{(1)}}}$.

Apply the chain rule again.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}(\,\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j\,)}{\partial \mathbf{W}^{(k)}}$?

Backpropagation:

- Denote $L = \mathrm{Loss}(\,\mathbf{f}(\mathbf{x}_j),\ \mathbf{y}_j\,)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$,

Free $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}}$ from memory.

- $\boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(0)}}} = \dfrac{\partial \mathbf{z}^{(1)}}{\partial \mathbf{W}^{(0)}} \dfrac{\partial L}{\partial \mathbf{z}^{(1)}}$.

Backpropagation

Define a function $\mathbf{f}: \mathbb{R}^{785} \mapsto \mathbb{R}^{10}$:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- $\mathbf{x}^{(3)} = \mathrm{SoftMax}(\mathbf{z}^{(3)}) \in \mathbb{R}^{10}$.
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathbf{x}^{(3)}$.

How to compute $\dfrac{\partial\, \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(k)}}$ **?**

Backpropagation:

- Denote $L = \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)$.
- Compute $\dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{z}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(2)}} = \dfrac{\partial \mathbf{z}^{(3)}}{\partial \mathbf{W}^{(2)}} \dfrac{\partial L}{\partial \mathbf{z}^{(3)}}$.
- $\dfrac{\partial L}{\partial \mathbf{z}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{z}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$, $\quad \dfrac{\partial L}{\partial \mathbf{W}^{(1)}} = \dfrac{\partial \mathbf{z}^{(2)}}{\partial \mathbf{W}^{(1)}} \dfrac{\partial L}{\partial \mathbf{z}^{(2)}}$.
- $\boxed{\dfrac{\partial L}{\partial \mathbf{W}^{(0)}}} = \dfrac{\partial \mathbf{z}^{(1)}}{\partial \mathbf{W}^{(0)}} \dfrac{\partial L}{\partial \mathbf{z}^{(1)}}$. Use it to update $\mathbf{W}^{(0)}$.

Backpropagation: Example

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input**: scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, \ z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, \ z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output**: $f\left(x^{(0)}\right) = z^{(3)}$.

Random sampling:

- Randomly sample j from $\{1, 2, \cdots, n\}$.

Forward pass (input ➔ output):

- Take x_j as input ($x^{(0)} = x_j$).
- Compute each layer $z^{(1)}, x^{(1)}, z^{(2)}, x^{(2)}, z^{(3)}$.

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2 = \frac{1}{2}\left(z^{(3)} - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.

1D Example

Define a function $\mathbf{f}: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = \boxed{z^{(3)}} - y_j$.

The value of $z^{(3)}$ is known (after the forward pass).

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\boxed{\frac{\partial L}{\partial z^{(3)}}} = z^{(3)} - y_j$.

The value of $z^{(3)}$ is known (after the forward pass).

Thus the value of $\frac{\partial L}{\partial z^{(3)}}$ is known.

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \boxed{\frac{\partial z^{(3)}}{\partial z^{(2)}}} \frac{\partial L}{\partial z^{(3)}}$

$$\frac{\partial z^{(3)}}{\partial x^{(2)}} = w^{(2)}, \qquad \frac{\partial x^{(2)}}{\partial z^{(2)}} = \begin{cases} 1, & \text{if } z^{(2)} > 0; \\ 0, & \text{else.} \end{cases}$$

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \boxed{\frac{\partial z^{(3)}}{\partial z^{(2)}}} \frac{\partial L}{\partial z^{(3)}}$

$$\frac{\partial z^{(3)}}{\partial x^{(2)}} = w^{(2)}, \qquad \frac{\partial x^{(2)}}{\partial z^{(2)}} = \begin{cases} 1, & \text{if } z^{(2)} > 0; \\ 0, & \text{else.} \end{cases}$$

$$\boxed{\frac{\partial z^{(3)}}{\partial z^{(2)}}} = \frac{\partial z^{(3)}}{\partial x^{(2)}} \frac{\partial x^{(2)}}{\partial z^{(2)}} = \begin{cases} w^{(2)}, & \text{if } z^{(2)} > 0; \\ 0, & \text{else.} \end{cases}$$

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\quad \frac{\partial L}{\partial w^{(2)}} = \boxed{\frac{\partial z^{(3)}}{\partial w^{(2)}}} \frac{\partial L}{\partial z^{(3)}}$.

$$\frac{\partial z^{(3)}}{\partial w^{(2)}} = x^{(2)}.$$

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.

Free $\frac{\partial L}{\partial z^{(3)}}$ from memory.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\boxed{\frac{\partial L}{\partial w^{(2)}}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.

Update $w^{(2)}$: $w^{(2)} \leftarrow w^{(2)} - \alpha \frac{\partial L}{\partial w^{(2)}}$.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.

Free $\frac{\partial z^{(3)}}{\partial w^{(2)}}$ from memory.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \boxed{\frac{\partial z^{(2)}}{\partial z^{(1)}}} \frac{\partial L}{\partial z^{(2)}}$

$$\boxed{\frac{\partial z^{(2)}}{\partial z^{(1)}}} = \frac{\partial z^{(2)}}{\partial x^{(1)}} \frac{\partial x^{(1)}}{\partial z^{(1)}} = \begin{cases} w^{(1)}, & \text{if } z^{(1)} > 0; \\ 0, & \text{else.} \end{cases}$$

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\quad \frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\quad \frac{\partial L}{\partial w^{(1)}} = \boxed{\frac{\partial z^{(2)}}{\partial w^{(1)}}} \frac{\partial L}{\partial z^{(2)}}$.

$$\frac{\partial z^{(2)}}{\partial w^{(1)}} = x^{(1)}.$$

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0,\ z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0,\ z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\frac{\partial L}{\partial w^{(1)}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.

Free $\frac{\partial L}{\partial z^{(2)}}$ from memory.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, \ z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, \ z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\boxed{\frac{\partial L}{\partial w^{(1)}}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.

Update $w^{(1)}$: $w^{(1)} \leftarrow w^{(1)} - \alpha \frac{\partial L}{\partial w^{(1)}}$.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\boxed{\frac{\partial L}{\partial w^{(1)}}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.

Free $\frac{\partial L}{\partial w^{(1)}}$ from memory.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\frac{\partial L}{\partial w^{(1)}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.
- $\frac{\partial L}{\partial w^{(0)}} = \boxed{\frac{\partial z^{(1)}}{\partial w^{(0)}}} \frac{\partial L}{\partial z^{(1)}}$.

$$\frac{\partial z^{(1)}}{\partial w^{(0)}} = x^{(0)}.$$

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f(x^{(0)}) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f(x_j) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\frac{\partial L}{\partial w^{(1)}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.
- $\frac{\partial L}{\partial w^{(0)}} = \frac{\partial z^{(1)}}{\partial w^{(0)}} \frac{\partial L}{\partial z^{(1)}}$.

Free $\frac{\partial L}{\partial z^{(1)}}$ from memory.

1D Example

Define a function f: $\mathbb{R} \mapsto \mathbb{R}$

- **Input:** scalar $x^{(0)}$.
- $z^{(1)} = w^{(0)} x^{(0)}$.
- $x^{(1)} = \max\{0, z^{(1)}\}$.
- $z^{(2)} = w^{(1)} x^{(1)}$.
- $x^{(2)} = \max\{0, z^{(2)}\}$.
- $z^{(3)} = w^{(2)} x^{(2)}$.
- **Output:** $f\left(x^{(0)}\right) = z^{(3)}$.

Backpropagation:

- Loss: $L = \frac{1}{2}\left(f\left(x_j\right) - y_j\right)^2$.
- $\frac{\partial L}{\partial z^{(3)}} = z^{(3)} - y_j$.
- $\frac{\partial L}{\partial z^{(2)}} = \frac{\partial z^{(3)}}{\partial z^{(2)}} \frac{\partial L}{\partial z^{(3)}}$, $\frac{\partial L}{\partial w^{(2)}} = \frac{\partial z^{(3)}}{\partial w^{(2)}} \frac{\partial L}{\partial z^{(3)}}$.
- $\frac{\partial L}{\partial z^{(1)}} = \frac{\partial z^{(2)}}{\partial z^{(1)}} \frac{\partial L}{\partial z^{(2)}}$, $\frac{\partial L}{\partial w^{(1)}} = \frac{\partial z^{(2)}}{\partial w^{(1)}} \frac{\partial L}{\partial z^{(2)}}$.
- $\boxed{\frac{\partial L}{\partial w^{(0)}}} = \frac{\partial z^{(1)}}{\partial w^{(0)}} \frac{\partial L}{\partial z^{(1)}}$.

Update $w^{(0)}$: $w^{(0)} \leftarrow w^{(0)} - \alpha \frac{\partial L}{\partial w^{(0)}}$.

1D Example

Define a function $f: \mathbb{R} \mapsto \mathbb{R}$

Backpropagation:

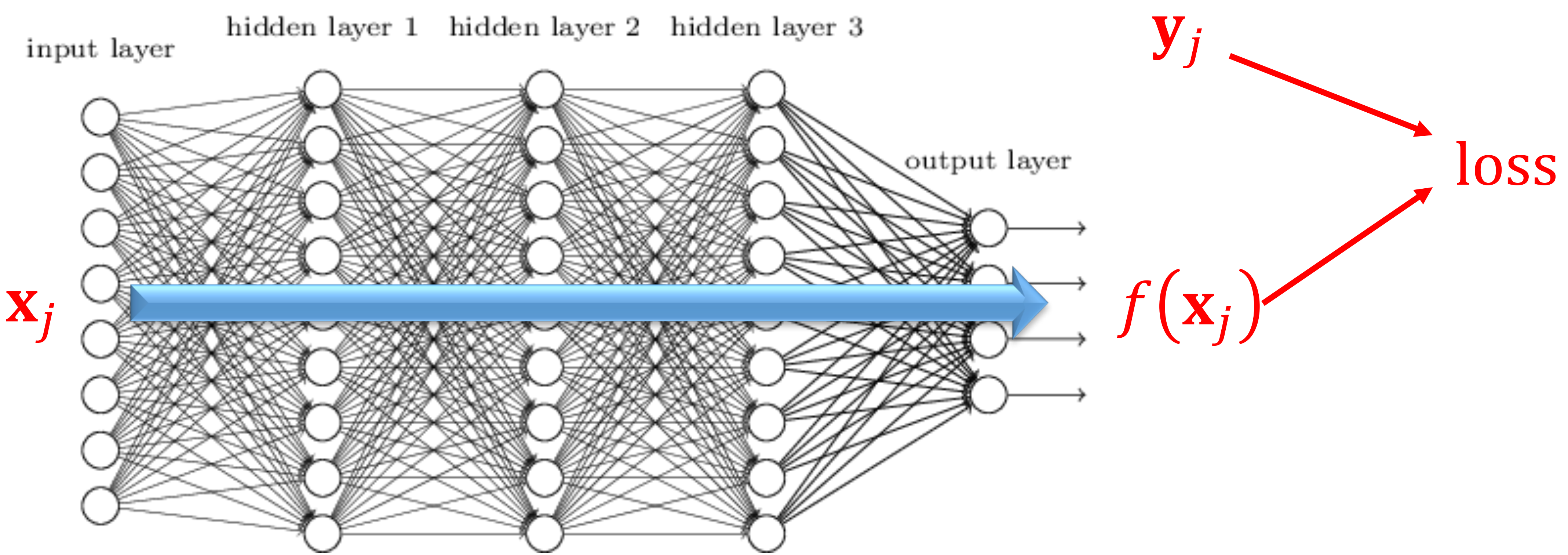
One iteration:

1. Randomly sample j from $\{1, 2, \cdots, n\}$.
2. **Forward pass**: take x_j as input ($x^{(0)} = x_j$), compute each layer $z^{(1)}, x^{(1)}, z^{(2)}, x^{(2)}, z^{(3)}$.
3. **Backward pass**:
   i. Compute the derivatives $\frac{\partial L}{\partial z^{(3)}}, \frac{\partial L}{\partial w^{(2)}}, \frac{\partial L}{\partial z^{(2)}}, \frac{\partial L}{\partial w^{(1)}}, \frac{\partial L}{\partial z^{(1)}}, \frac{\partial L}{\partial w^{(0)}}$.
   ii. Update $w^{(k)}$ using $\frac{\partial L}{\partial w^{(k)}}$.

Summary of Backpropagation

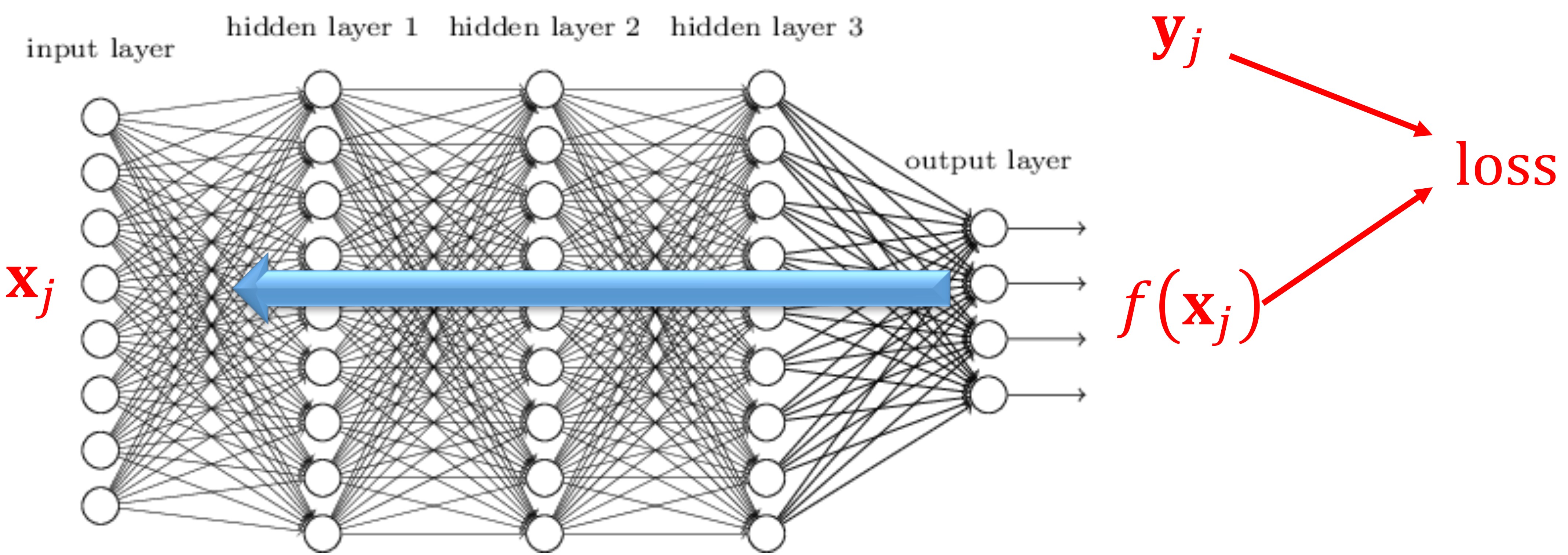
Backpropagation: Take-Home Message

1. Randomly pick a sample $(\mathbf{x}_j, \mathbf{y}_j)$.
2. Run a forward pass (from the input $\mathbf{x}^{(0)}$ to the prediction).



Backpropagation: Take-Home Message

1. Randomly pick a sample $(\mathbf{x}_j, \mathbf{y}_j)$.
2. Run a forward pass (from the input $\mathbf{x}^{(0)}$ to the prediction).
3. Run a backward pass (from the loss to $\mathbf{W}^{(0)}$).



Backpropagation: Take-Home Message

1. Randomly pick a sample $(\mathbf{x}_j, \mathbf{y}_j)$.
2. Run a forward pass (from the input $\mathbf{x}^{(0)}$ to the prediction).
3. Run a backward pass (from the loss to $\mathbf{W}^{(0)}$).

Get the derivatives (stochastic gradients):

$$\frac{\partial \text{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(2)}}, \quad \frac{\partial \text{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(1)}}, \quad \frac{\partial \text{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(0)}}.$$

Backpropagation: Take-Home Message

1. Randomly pick a sample $(\mathbf{x}_j, \mathbf{y}_j)$.
2. Run a forward pass (from the input $\mathbf{x}^{(0)}$ to the prediction).
3. Run a backward pass (from the loss to $\mathbf{W}^{(0)}$).

Get the derivatives (stochastic gradients):

$$\frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(2)}}, \quad \frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(1)}}, \quad \frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(0)}}.$$

Update $\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, \mathbf{W}^{(2)}$ using the derivatives.

Mini-Batch

1. Randomly pick a ~~sample $(\mathbf{x}_j, \mathbf{y}_j)$~~. Several random samples.
2. Run a forward pass (from the input $\mathbf{x}^{(0)}$ to the prediction).
3. Run a backward pass (from the loss to $\mathbf{W}^{(0)}$).

Get the derivatives (stochastic gradients):

$$\cancel{\frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(2)}}}, \quad \cancel{\frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(1)}}}, \quad \cancel{\frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(0)}}}.$$

$$\frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(2)}}, \quad \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(1)}}, \quad \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \frac{\partial \mathrm{Loss}(\mathbf{f}(\mathbf{x}_j), \mathbf{y}_j)}{\partial \mathbf{W}^{(0)}}.$$

Mini-batch should always be used! Set batch size $|\mathcal{J}|$ to 16, 32, 64, …

Mini-Batch

SGD: $\text{BatchSize} = 1.$

- Per-iteration cost is low.
- Lots of iterations to converge.

Mini-Batch: $\text{BatchSize} > 1.$

- Better than the other two, if BatchSize is properly set.

Full Gradient: $\text{BatchSize} = n.$

- Per-iteration cost is n times higher than SGD.
- Convex problem: less number of iterations.
- Neural network: it doesn’t work!

First-Order Optimization

- First-order optimization: update the parameters using gradient.
- Gradient descent algorithm (including SGD, mini-batch SGD, and full gradient descent, conjugate gradient) are typical 1st-order algorithms.
- Other 1st-order algorithms: SGD with momentum, AdaGrad, RMSprop…
- See the blogs:
  - http://ruder.io/optimizing-gradient-descent/
  - https://distill.pub/2017/momentum/

Summary of FC Neural Network

Build a Fully-Connected Neural Network

- Network structure

Number of layers

Width of each layer

Activation functions

Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

Example:

- **Input:** vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$. Input layer
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$. Hidden Layer 1
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$. Hidden Layer 2
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$. Output layer
- **Output:** $\mathbf{f}(\mathbf{x}^{(0)}) = \mathrm{SoftMax}(\mathbf{z}^{(2)})$.

- Three layers (2 hidden and 1 output).
  - Input layer doesn't count (no parameter).

Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

Example:

- **Input**: vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\boxed{\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}.}$
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\boxed{\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}.}$
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\boxed{\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}.}$
- **Output**: $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathrm{SoftMax}\left(\mathbf{z}^{(2)}\right)$.

- Three layers (2 hidden and 1 output).
  - Input layer doesn't count (no parameter).
- Width of each layer:
  - Layer 1: d_1,
  - Layer 2: d_2,
  - Output layer: 10.

Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

Example:

- **Input**: vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}$.
- $\boxed{\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}.}$
- $\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}$.
- $\boxed{\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}.}$
- $\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}$.
- **Output**: $\boxed{\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathrm{SoftMax}\left(\mathbf{z}^{(2)}\right)}$.

- Three layers (2 hidden and 1 output).
  - Input layer doesn't count (no parameter).
- Width of each layer:
  - Layer 1: d_1,
  - Layer 2: d_2,
  - Output layer: 10.
- Activation functions:
  - Layer 1: ReLU,
  - Layer 2: ReLU,
  - Output layer: Softmax.

Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

Example:

- **Input**: vector $\mathbf{x}^{(0)} \in \mathbb{R}^{785}$.
- $\boxed{\mathbf{z}^{(1)} = \mathbf{W}^{(0)} \mathbf{x}^{(0)} \in \mathbb{R}^{d_1}.}$
- $\mathbf{x}^{(1)} = \max\{\mathbf{0}, \mathbf{z}^{(1)}\} \in \mathbb{R}^{d_1}$.
- $\boxed{\mathbf{z}^{(2)} = \mathbf{W}^{(1)} \mathbf{x}^{(1)} \in \mathbb{R}^{d_2}.}$
- $\mathbf{x}^{(2)} = \max\{\mathbf{0}, \mathbf{z}^{(2)}\} \in \mathbb{R}^{d_2}$.
- $\boxed{\mathbf{z}^{(3)} = \mathbf{W}^{(2)} \mathbf{x}^{(2)} \in \mathbb{R}^{10}.}$
- **Output**: $\mathbf{f}\left(\mathbf{x}^{(0)}\right) = \mathrm{SoftMax}\left(\mathbf{z}^{(2)}\right)$.

- Trainable parameters:
  - $\mathbf{W}^{(0)} \in \mathbb{R}^{d_1 \times 785}$,
  - $\mathbf{W}^{(1)} \in \mathbb{R}^{d_2 \times d_1}$,
  - $\mathbf{W}^{(2)} \in \mathbb{R}^{10 \times d_2}$.

- Number of parameters:
  - $785 d_1 + d_1 d_2 + 10 d_2$.

Build a Fully-Connected Neural Network

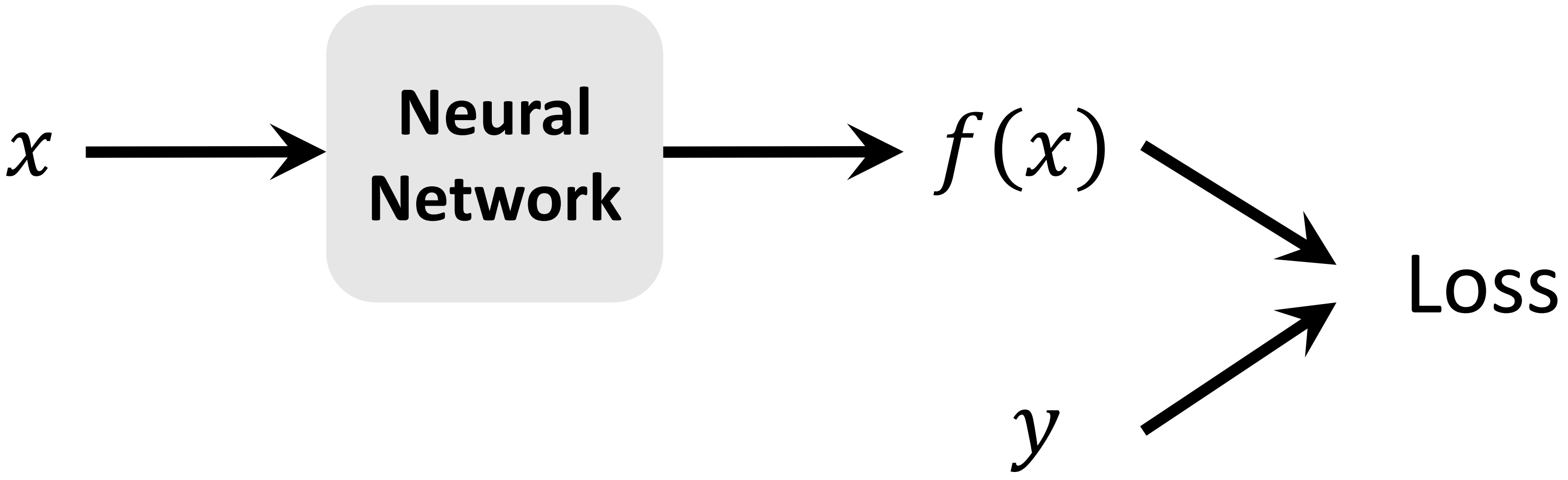
- Network structure

Number of layers | Width of each layer | Activation functions

- Loss functions

Cross-entropy for classification

L1 or squared L2 for regression (the labels are continuous)



Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

- Loss functions

Cross-entropy for classification

L1 or squared L2 for regression (the labels are continuous)

- Compute gradient: by a forward pass and a backward pass

Automatically performed by many deep learning systems | Batch size

Build a Fully-Connected Neural Network

- Network structure

Number of layers | Width of each layer | Activation functions

- Loss functions

Cross-entropy for classification

L1 or squared L2 for regression (the labels are continuous)

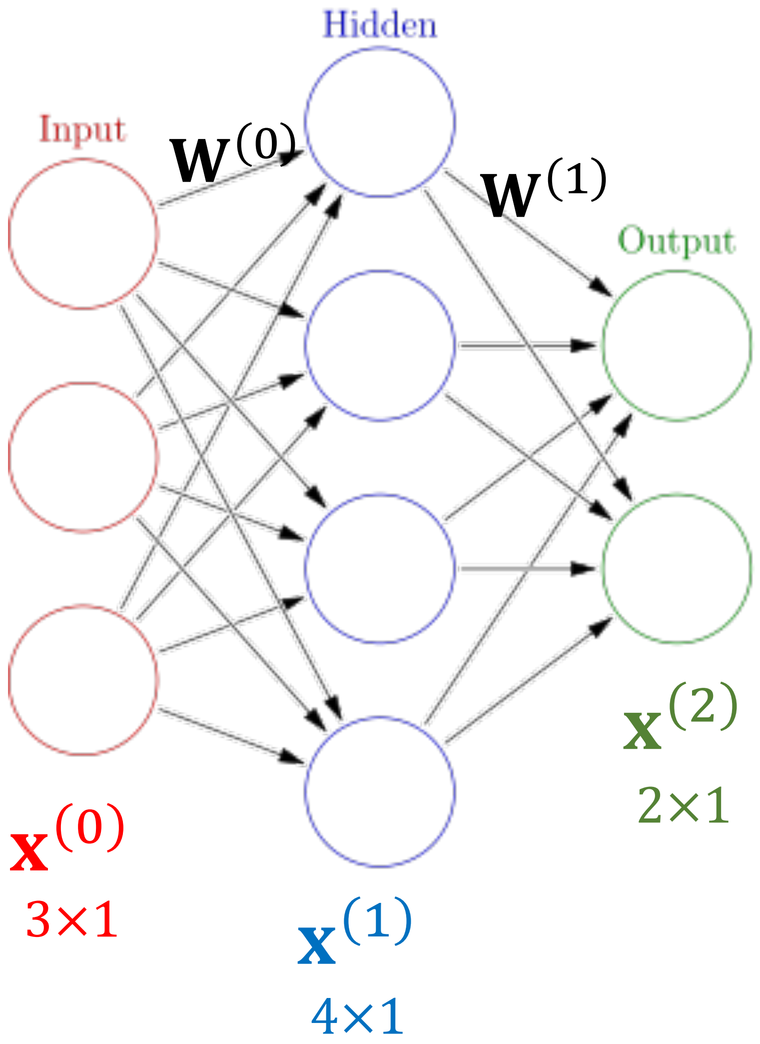
- Compute gradient: by a forward pass and a backward pass

Automatically performed by many deep learning systems | Batch size

- Choose an optimization algorithm (and tune its hyper-parameters)

SGD | SGD with momentum | AdaGrad | RMSprop

Representing a Neural Network



- A node denotes one entry of vector $\mathbf{x}$.
- A dense layer is parameterized by a matrix $\mathbf{W}$.
- An edge denotes one entry of parameter matrix $\mathbf{W}$.

Representing a Neural Network

Equivalent representation: