Deep Neural Networks

Petra Galuscakova

petra.galuscakova@uis.no

University of Stavanger

Information Access & Artificial Intelligence

Department of Electrical Engineering and Computer Science
University of Stavanger

January 8, 2026

Overview

Deep Neural Networks

Building DNNs

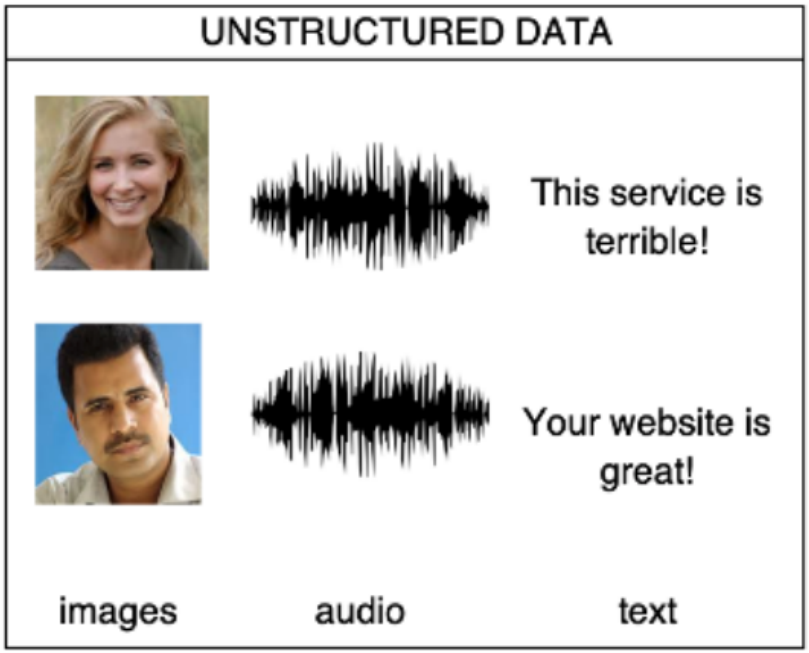
Deep Learning

- Deep learning is a class of machine learning algorithms that uses multiple stacked layers of processing units (multilayered perceptron, i.e. MLP) to learn high-level representations from unstructured data.

Structured and Unstructured Data

STRUCTURED DATA

id	age	gender	height (cm)	location
0001	54	M	186	London
0002	35	F	166	New York
0003	62	F	170	Amsterdam
0004	23	M	164	London
0005	25	M	180	Cairo
0006	29	F	181	Beijing
0007	46	M	172	Chicago



- Structured: tabular data, arranged into columns of features that describe each observation (e.g. age, income, and number of website visits)
- Unstructured data refers to any data that is not naturally arranged into columns of features, such as images, audio, and text.

Structured and Unstructured Data (cont.)

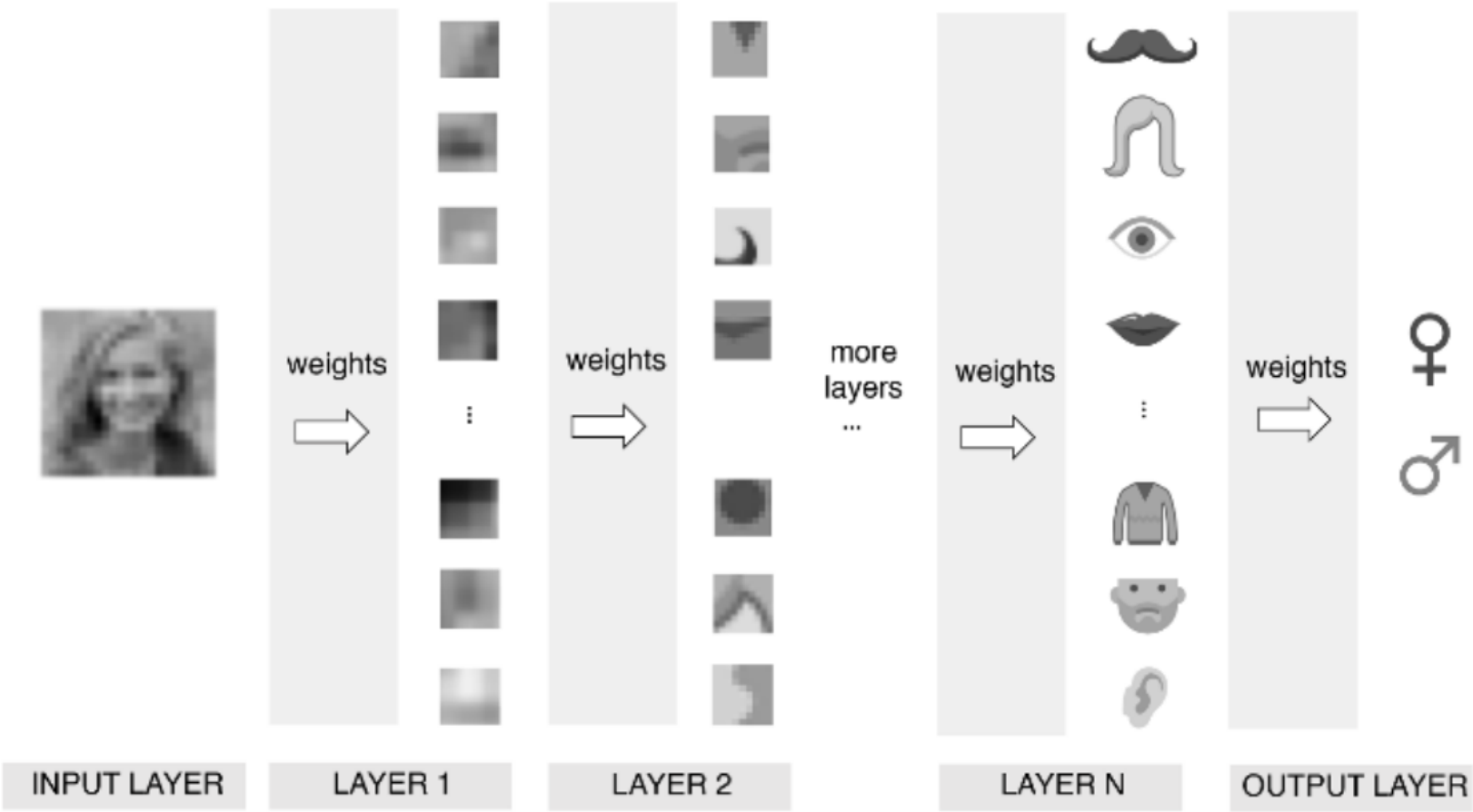
- Deep learning can be applied to structured data, but its real power, especially with regard to generative modeling, comes from its ability to work with unstructured data.
- A deep learning model, can learn how to build high-level informative features by itself, directly from the unstructured data.

Deep Neural Networks

- The majority of deep learning systems are artificial neural networks with multiple stacked hidden layers.
- A deep neural network consists of a series of stacked layers.
- Each layer contains units, that are connected to the previous layer's units through a set of weights.
- One of the most common is the dense layer that connects all units in the layer directly to every unit in the previous layer.
- Deep neural networks can have any number of middle or hidden layers.

Deep Neural Networks (cont.)

- By stacking layers, the units in each subsequent layer can represent increasingly sophisticated aspects of the original input.



Deep Neural Networks (cont.)

- The final output layer is the culmination of this process, where the network outputs a set of numbers that can be converted into probabilities, to represent the chance that the original input belongs to one of n categories.
- During the training process, batches of images are passed through the network and the output is compared to the ground truth.
- Backpropagation: The error in the prediction is then propagated backward through the network, adjusting each set of weights.

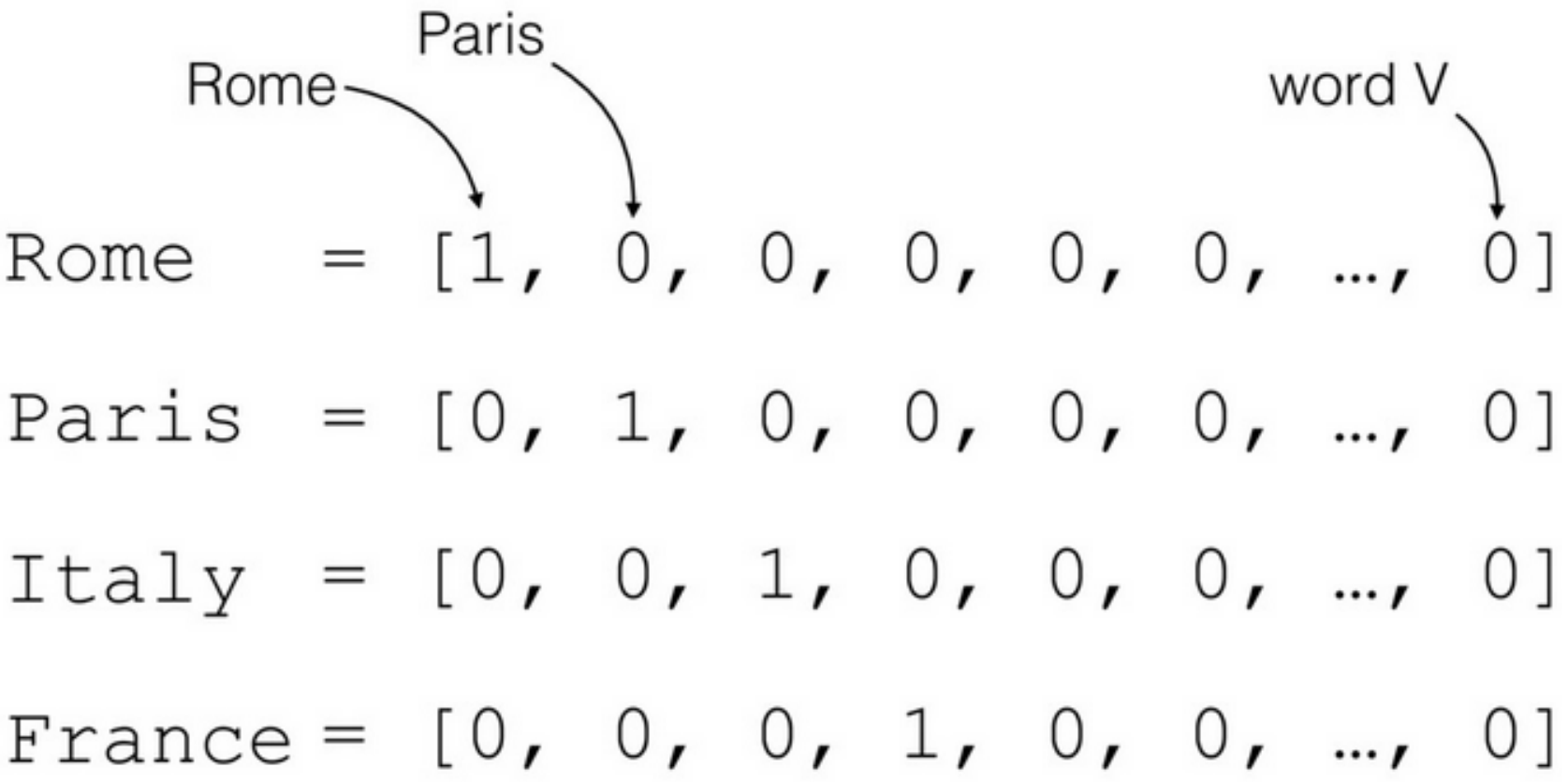
DNN Libaries

- TensorFlow is an open-source Python library for machine learning, developed by Google.
- It is now one of the most utilized frameworks for building machine learning solutions, with particular emphasis on the manipulation of tensors.
- Within the context of deep learning, tensors are simply multidimensional arrays that store the data as it flows through the network.
- PyTorch is generally preferred for its flexibility and ease of use in research and rapid prototyping, while TensorFlow excels in large-scale production deployments.
- Keras uses low-level array operations required to train neural networks

Building DNN

- https://github.com/dat560-2026/info → Deep learning recap → MLP → Notebook

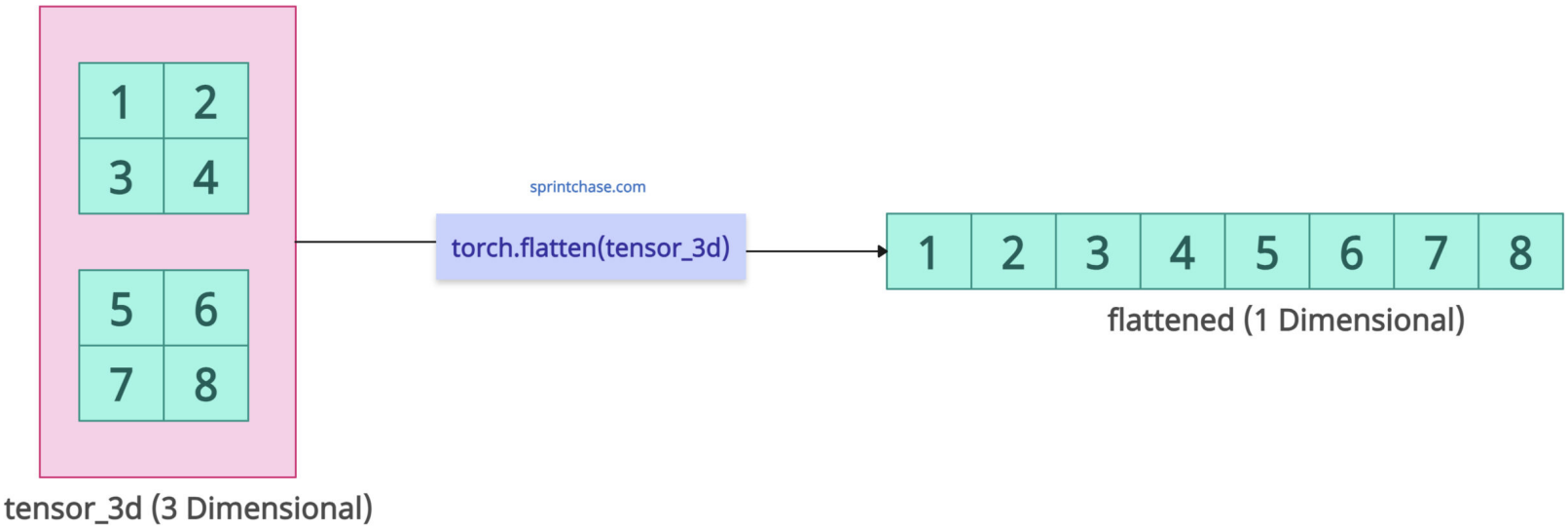
One hot encoding



1

[1] https://medium.com/intelligentmachines/word-embedding-and-one-hot-encoding-ad17b4bbe111

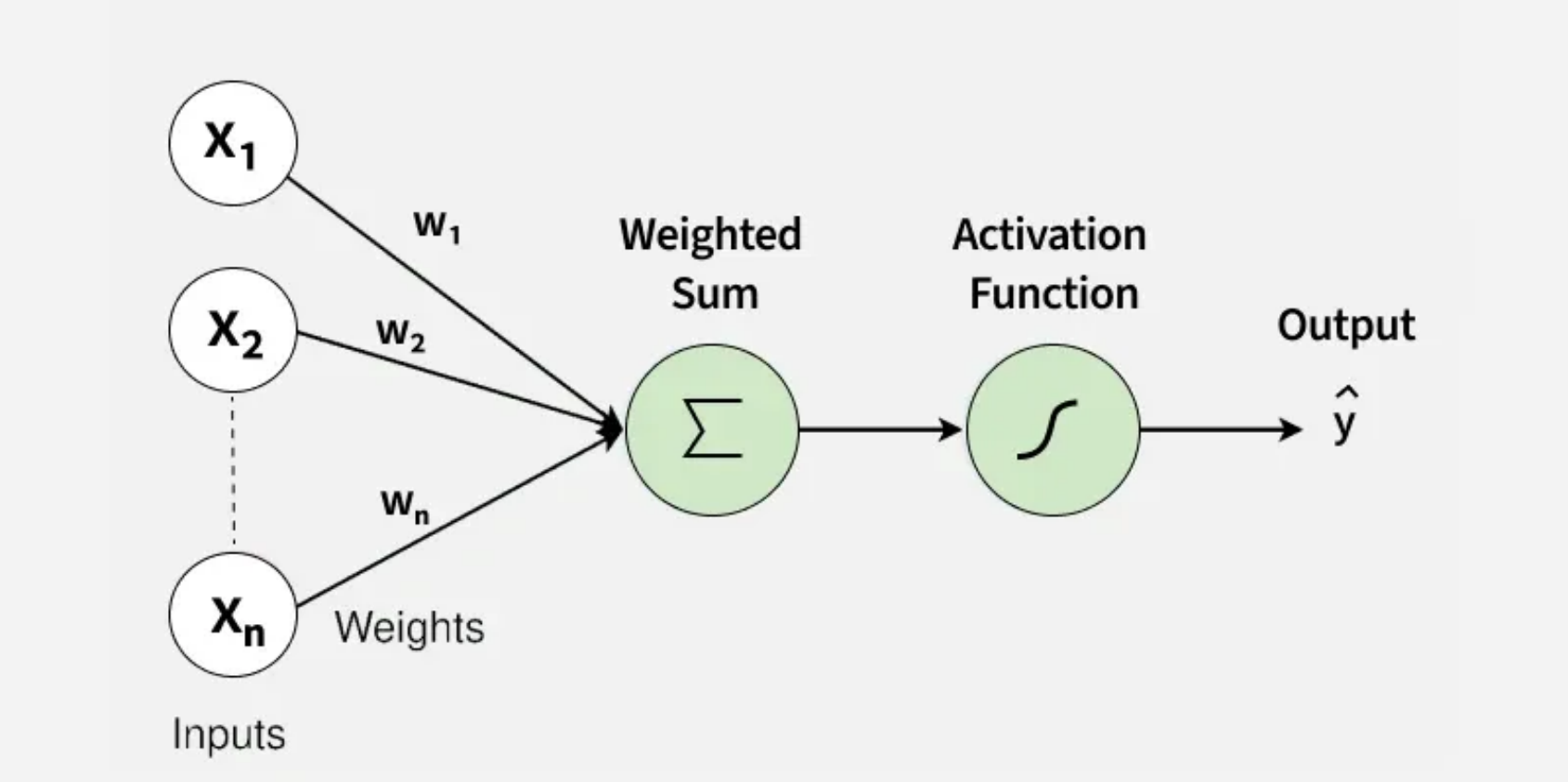
Vector flattening



2

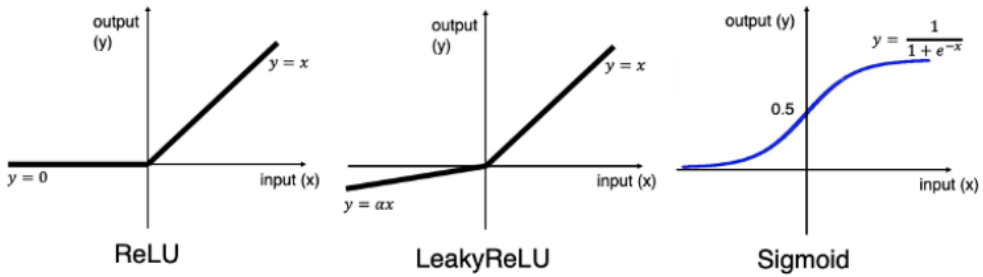
[2] https://sprintchase.com/torch-flatten/

Perceptron



Activation functions are crucial in neural networks because they introduce non-linearity, allowing the network to learn complex patterns beyond simple straight lines.

Activation Function



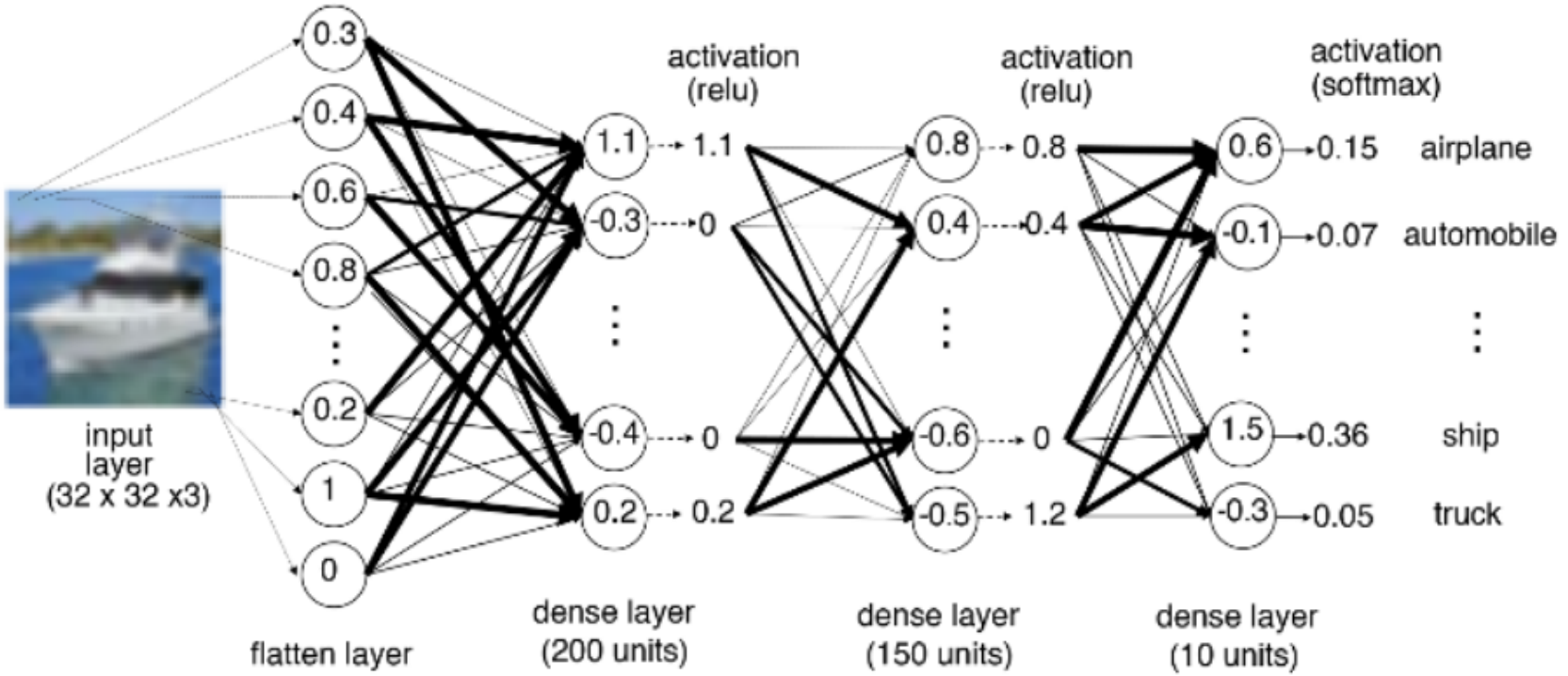
- The ReLU (rectified linear unit) activation function is defined to be zero if the input is negative and is otherwise equal to the input.
- The LeakyReLU activation function is very similar to ReLU. Whereas the ReLU activation function returns zero for input values less than zero, the LeakyReLU function returns a small negative number proportional to the input.
- The sigmoid activation is useful if you wish the output from the layer to be scaled between 0 and 1.

Softmax Activation Function

- The softmax activation is useful if you want the total sum of the output from the layer to equal 1, for example, for multiclass classification problems where each observation only belongs to exactly one class.
- $$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

  where j is the number of all classes

Full CIFAR DNN Example



Loss Functions

- The loss function is used by the neural network to compare its predicted output to the ground truth.
- If your neural network is designed to solve a regression problem (i.e., the output is continuous), then you can use the **mean squared error loss**:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  where n is number of training examples, y_i is a ground truth value and $\hat{y}_i$ is the predicted value.
- If you are working on a classification problem where each observation only belongs to one class, then use **categorical cross-entropy**:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

Training

- The optimizer is the algorithm that is used to update the weights in the neural network based on the gradient of the loss function.
- The gradient of a loss function is a vector of partial derivatives that indicates the direction and rate of steepest increase of the model's error relative to its weights and biases.
- First, the weights of the network are initialized to small random values.
- Then the network performs a series of training steps. At each training step, one batch of images is passed through the network and the errors are backpropagated to update the weights.
- This continues until all observations in the dataset have been seen once. This completes the first epoch. This process repeats until the specified number of epochs have elapsed.