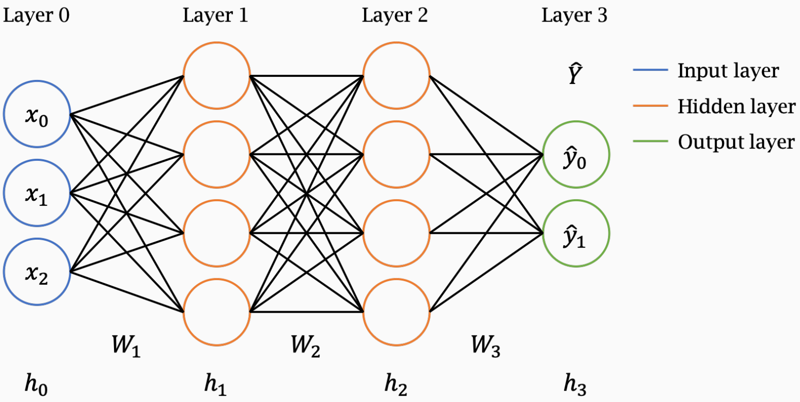
Day 2: Neural Networks

Guillem & Roderic Guigo Corominas

24-08-2021

SMTB

Neural Networks

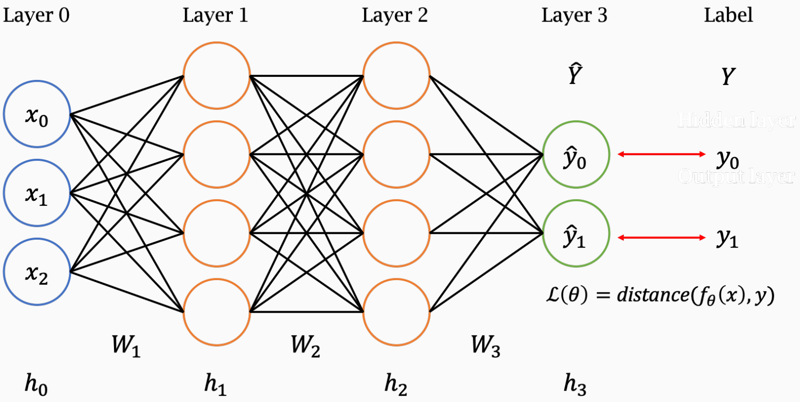


$h_0 = x$

$h^{(t)} = g(W^{(t)} h^{(t-1)} + b^t)$

$f_\theta(x) = h^{(L)}$

$\theta \to W, b$



$h_0 = x$

$h^{(t)} = g(W^{(t)} h^{(t-1)} + b^t)$

$f_\theta(x) = h^{(L)}$

$\theta \rightarrow W, b$

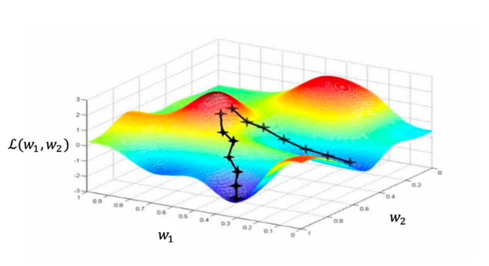
Loss function

- Mean Squared Error

$$\mathcal{L}(\theta) = MSE = \frac{1}{n}\sum_i (y_i - f_\theta(x_i))^2$$

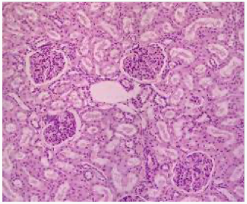
- Cross Entropy

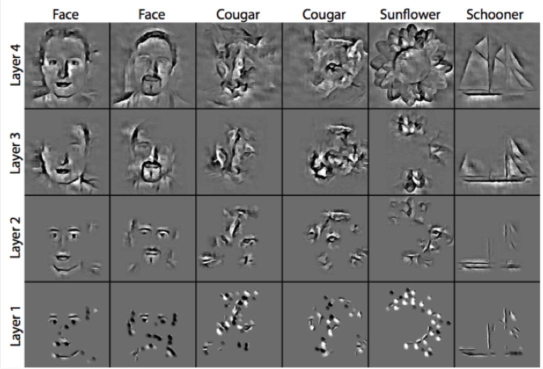
$$\mathcal{L}(\theta) = H = -\sum_i y_i \log(f_\theta(x_i))$$



Layer abstraction

- Different types of layers: Fully connected, convolutional, recurrent
- **Convolutional neural networks (CNN)**, recurrent neural networks (RNN), autoencoders, GANs, ...
- Layers of different types can be stacked to obtain new functionalities





Why do we need multiple layers?

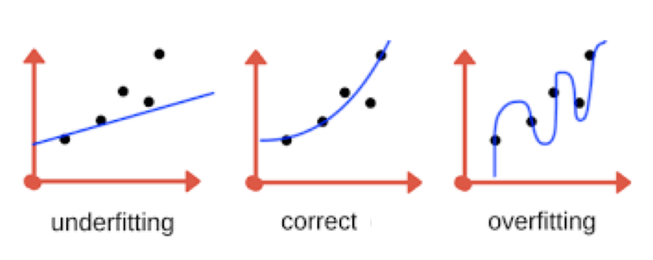
- Universal approximation theorem: The standard multilayer feed-forward network with a single hidden layer, which contains finite number of hidden neurons, is a universal approximator among continuous functions on compact subsets of $\mathbb{R}^n$, under mild assumptions on the activation function
- We might need an extremely large number of neurons in the hidden layer to model our function
- Deep neural networks are easier to train

Train, test & validation sets

- Dataset with 80.000 labeled images
- Splits: 70% train/20% validation/10% test: 56.000 in training set, 16.000 in validation set & 8.000 in test set
- The training set is the data that we use to fit our model
- The validation set is used to adjust the hyperparameters of the model
- The test set is used to test the performance of our model in unseen data

Overfitting & Underfitting

- The number of parameters in our model is related to the dimensionality of our data and the number of examples in our training set
- Overfitting occurs when our model fits "too well" the training set, so it doesn't capture the true trend in our data



Bias/Variance tradeoff

- A highly biased model will likely have low variance, but it will not be able to fit the data properly
- If the bias is too low our model can overfit the training data and it will not do well on new data
- Ideally, we want low bias & low variance
- Regularization

Regularization

- Regularization techniques are used to avoid overfitting and improve generalization of the model
- A common technique is L2 regularization or weight decay. It prevents weights in the neural network from becoming too large by adding a penalty to the loss function

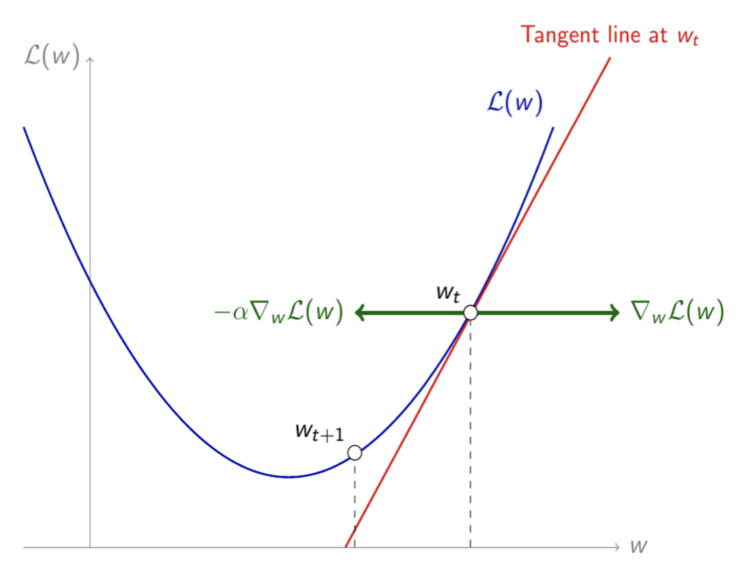
$$E = (\theta) + \lambda \sum_{j}^{p} \theta_j$$

- Dropout $(0 < p < 1)$

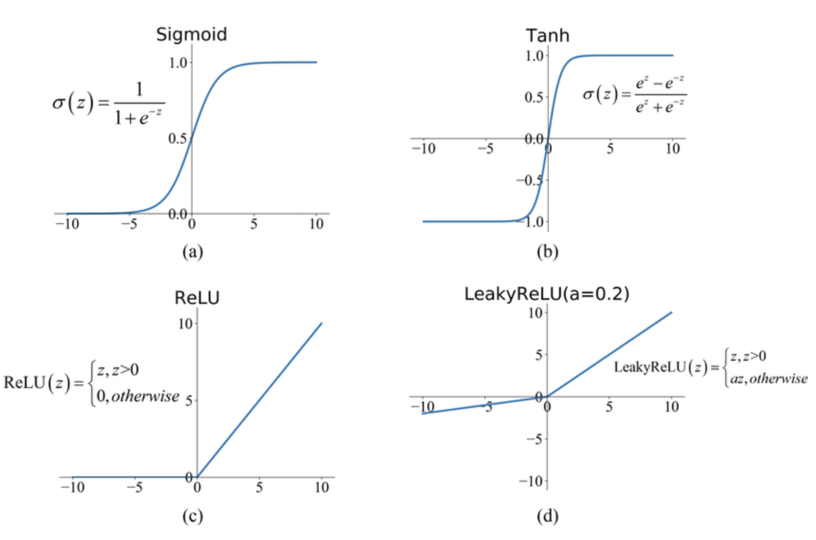
Hyperparameters

- Number of hidden layers
- Number of nodes in each layer
- Loss function
- Weight initialization
- Weight decay (λ)
- Dropout
- Learning rate (α)
- Activation functions
- Batch size
- Number of epochs

Learning rate



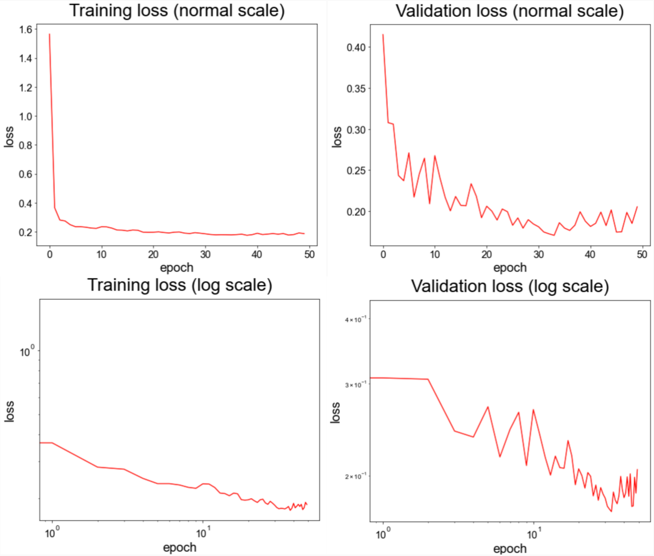
Activation functions



Batch size

- We (usually) don't train with the entire dataset at once, nor with a single example at a time
- Instead, we select a subset of examples in our training set and we compute the gradient after we calculate the error of all of the examples in the subset. This subset is called a batch
- Batch size: 1, 16, 64, 256...

Epochs


Training loss (normal scale)
loss
epoch
Validation loss (normal scale)
loss
epoch
Training loss (log scale)
loss
epoch
Validation loss (log scale)
loss
epoch

Hyperparameters

- Number of hidden layers
- Number of nodes in each layer
- Loss function
- Weight initialization
- Weight decay (λ)
- Dropout
- Learning rate (α)
- Activation functions
- Batch size
- Number of epochs

Next Class

Train a Neural Network