



UPPSALA
UNIVERSITET

Machine learning, big data and artificial intelligence – Block 4

Måns Magnusson
Department of Statistics, Uppsala University

November 2020

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization



UPPSALA
UNIVERSITET

Evaluation assignment 2

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Took too much time (roughly 26h) - how to solve this?
Hints? remove subtasks?
- More teaching on code
- The bugs...
- Minor comments:
 - xgboost video
 - bigger diff between RF and bagging
 - more focus on the assignment on the lecture



UPPSALA
UNIVERSITET

Grading of assignment 1

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Why is SGD important?
- Difference between unsupervised and supervised learning. Task or experience?



UPPSALA
UNIVERSITET

Masters thesis proposals

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

1. Evaluation of probabilistic programming frameworks
2. Predicting introductions in Swedish parliamentary protocols using BERT
3. Topic model inference: (Stochastic) variational inference and Gibbs sampling
4. Will Svenska akademins ordlista (SAOL) improve Swedish word embeddings?
5. Fine-tune a language model (BERT) on EDGAR-CORPUS
6. OCR-error detection using image and text classification



UPPSALA
UNIVERSITET

This week's lecture

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

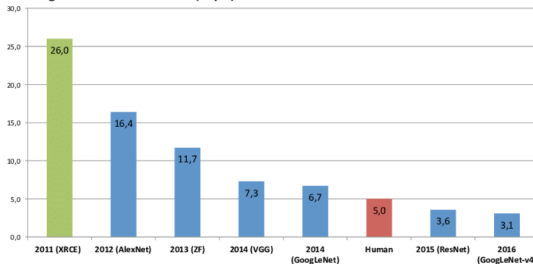
- Regularization

- Feed-Forward Neural Networks



Figure: ImageNet performance (Roessler, 2019)

ImageNet Classification Error (Top 5)



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization



UPPSALA
UNIVERSITET

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

The Hype: Speech Recognition

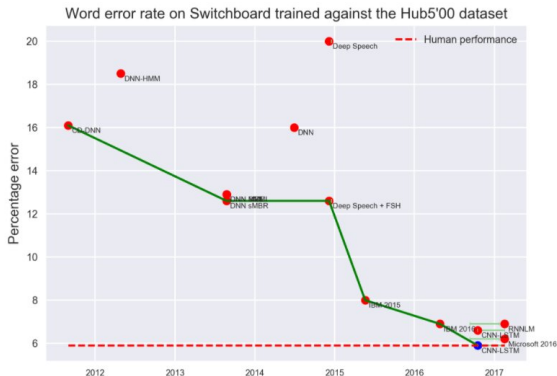


Figure: Speech recognition performance (source: <https://eff.org/ai/metrics>)



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

The Hype: Natural Language Processing

GLUE scores evolution over 2018-2019

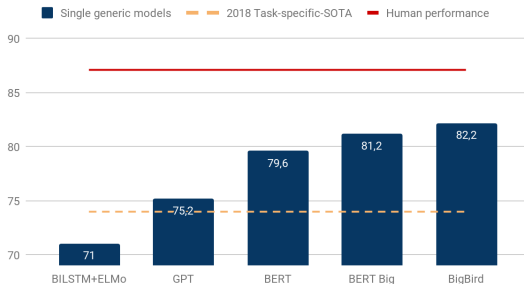


Figure: General Language Understanding (source: <https://www.programmersought.com/article/4251948498/>)

Work is very much ongoing:

<https://gluebenchmark.com/leaderboard>



UPPSALA
UNIVERSITET

The Hype

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Although - Neural Networks is not a silver bullet



UPPSALA
UNIVERSITET

The Hype

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Although - Neural Networks is not a silver bullet
- Remember the **Bayes error**



UPPSALA
UNIVERSITET

The Hype

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Although - Neural Networks is not a silver bullet
- Remember the **Bayes error**
- Some times a linear regression (or Random Forest) is enough



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

The Feed-Forward Network

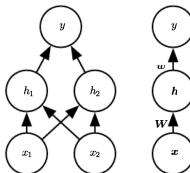


Figure 6.2: An example of a feedforward network, drawn in two different styles. Specifically, this is the feedforward network we use to solve the XOR example. It has a single hidden layer containing two units. (*Left*) In this style, we draw every unit as a node in the graph. This style is very explicit and unambiguous but for networks larger than this example it can consume too much space. (*Right*) In this style, we draw a node in the graph for each entire vector representing a layer's activations. This style is much more compact.

Figure: A simple feed-forward network (Goodfellow et al, 2017)

Important concepts:

Layers, neurons, input, output, weights, bias, architecture



UPPSALA
UNIVERSITET

Different Architectures for Different Purposes

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Different networks for different purposes
 - **Convolutional Neural Networks:** Computer Vision



UPPSALA
UNIVERSITET

Different Architectures for Different Purposes

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Different networks for different purposes
 - **Convolutional Neural Networks**: Computer Vision
 - **Recurrent Neural Networks**: Speech Audio (?)



Different Architectures for Different Purposes

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Different networks for different purposes

- **Convolutional Neural Networks**: Computer Vision
- **Recurrent Neural Networks**: Speech Audio (?)
- **Transformers/Attention**: Textual data

- The Neural Network Zoo: [https:](https://www.asimovinstitute.org/neural-network-zoo/)

[//www.asimovinstitute.org/neural-network-zoo/](https://www.asimovinstitute.org/neural-network-zoo/)



UPPSALA
UNIVERSITET

Areas of Use: All fields

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Supervised learning
- Unsupervised learning
- Reinforcement learning



UPPSALA
UNIVERSITET

Why and when neural nets?

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Learning feature representations
- Needs a lot of data to learn complex representations
- Good for sensor data (high-dimensional)



UPPSALA
UNIVERSITET

Why and when neural nets?

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Learning feature representations
- Needs a lot of data to learn complex representations
- Good for sensor data (high-dimensional)
- When should we not use neural networks?



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

Learning Representations

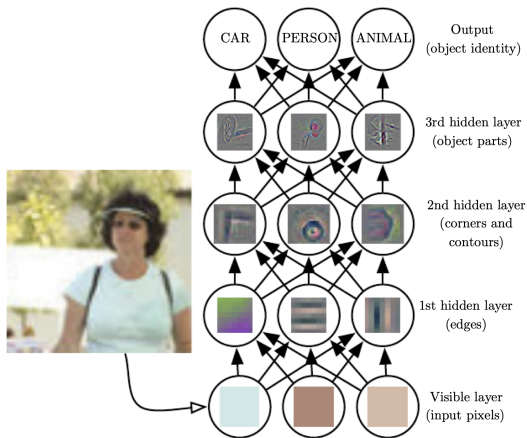


Figure: Learning representations can be crucial (Goodfellow et al, 2017, Fig. 1.2)



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

The Feed-Forward Network

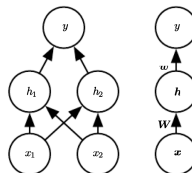


Figure 6.2: An example of a feedforward network, drawn in two different styles. Specifically, this is the feedforward network we use to solve the XOR example. It has a single hidden layer containing two units. (*Left*) In this style, we draw every unit as a node in the graph. This style is very explicit and unambiguous but for networks larger than this example it can consume too much space. (*Right*) In this style, we draw a node in the graph for each entire vector representing a layer's activations. This style is much more compact.

Figure: A simple feed-forward network (Goodfellow et al, 2017, Fig. 6.2)

In mathematical notation:

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$



$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- **Feed-Forward Neural Networks**
- Hyper-parameters

- Optimization

- Regularization

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = (0)$$

$$g(z) = \text{ReLU}(z) = \max(0, z), x_i = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

$$y_i = \begin{pmatrix} 1 \\ -2 \end{pmatrix}^T g\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \end{pmatrix}\right) + (0)$$

$$y_i = \begin{pmatrix} 1 \\ -2 \end{pmatrix}^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (0) = 1$$



The Feed-Forward Network

A feed-forward network for one observation (x_i).

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

$$\underbrace{\mathbf{h}_1}_{1 \times k_1} = g_1(\underbrace{\mathbf{x}^T}_{1 \times p} \underbrace{\mathbf{W}_1}_{p \times k_1} + \underbrace{\mathbf{b}_1}_{1 \times k_1})$$

\vdots

$$\underbrace{\mathbf{h}_l}_{1 \times k_l} = g_l(\underbrace{\mathbf{h}_{l-1}^T}_{1 \times k_{l-1}} \underbrace{\mathbf{W}_l}_{k_{l-1} \times k_l} + \underbrace{\mathbf{b}_l}_{1 \times k_l})$$

\vdots

$$\underbrace{\hat{\mathbf{y}}}_{1 \times m} = g_L(\underbrace{\mathbf{h}_{L-1}^T}_{1 \times k_{L-1}} \underbrace{\mathbf{W}_L}_{k_{L-1} \times m} + \underbrace{\mathbf{b}_L}_{1 \times m})$$

$$\hat{y} = f_L(f_{L-1}(\dots f_1(x)\dots))$$



UPPSALA
UNIVERSITET

Activation functions (g_l)

- Sometimes use notation σ as in $\sigma(Wh + b)$

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization



Activation functions (g_l)

- Sometimes use notation σ as in $\sigma(Wh + b)$
- Historically $g(z)$ has been the sigmoid or or hyperbolic tangent (tanh)

$$g_{\text{sigmoid}}(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}}$$

$$g_{\text{tanh}}(z) = \frac{\sinh z}{\cosh z} = \frac{e^{2z} - 1}{e^{2z} + 1}$$

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization



Activation functions (g_I)

- Sometimes use notation σ as in $\sigma(Wh + b)$
- Historically $g(z)$ has been the sigmoid or or hyperbolic tangent (\tanh)

$$g_{\text{sigmoid}}(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}}$$

$$g_{\text{tanh}}(z) = \frac{\sinh z}{\cosh z} = \frac{e^{2z} - 1}{e^{2z} + 1}$$

- Now, usually variants of Rectified linear unit (ReLU)

$$g_{\text{ReLU}}(z) = \max(0, z)$$

- Easier to estimate with SGD
 - Easier for deep models
- Last activation is the output function g_L , usually a softmax (if classification)

$$f(z_i) = \frac{e^{z_i}}{\sum_{j=1}^J e^{z_j}}$$



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Activation functions (g_l)

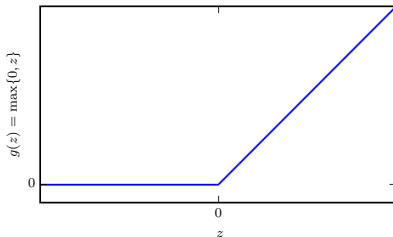


Figure: Rectified Linear Unit (Goodfellow et al, 2017, Fig. 6.3)



"A feed-forward neural network with a linear output layer and at least one hidden layer with any 'squashing' activation function can approximate any Borel measurable function from one finite-dimensional space to another with any desired non-zero amount of error, provided that the network is given enough hidden units." (Goodfellow et al. 2017, p. 198)

- Also holds for ReLU
- No guarantee we can learn the network
- No guarantee that it will generalize
- No indication of how large the network need to be



UPPSALA
UNIVERSITET

Hyper-parameters in feed-forward networks

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- The number of layers
- The number of neurons
- Activation functions
- The type of layers (CNN, MaxPooling, Multi-head attention)



UPPSALA
UNIVERSITET

How to choose parameters

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Trial and error on validation sets
- Art rather than science
- Specialized approaches (Bayesian Optimization)



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Trial and error on validation sets
- Art rather than science
- Specialized approaches (Bayesian Optimization)
- Grid search (combinatorial explosion)
 - Really bad with many parameters with less effects
 - If we have 5 irrelevant parameters we try 3 values for: 125 training per relevant run
 - Instead use...



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Trial and error on validation sets
- Art rather than science
- Specialized approaches (Bayesian Optimization)
- Grid search (combinatorial explosion)
 - Really bad with many parameters with less effects
 - If we have 5 irrelevant parameters we try 3 values for: 125 training per relevant run
 - Instead use...
- Random search



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Grid search vs. Random Search

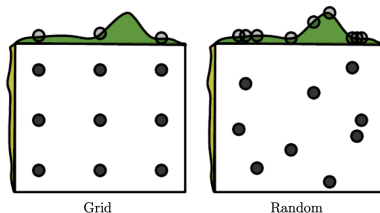


Figure: Grid search and random search (Goodfellow et al, 2017, Fig. 11.2)



UPPSALA
UNIVERSITET

Optimization of Neural Networks

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Difficult problem
- Many local minima (weight space symmetry)
- Plateaus and saddle points
 - Gradient is small - but not a minimum or maximum
 - Saddle points increase with the number of dimensions (?)
 - Large areas with small change in cost function



UPPSALA
UNIVERSITET

Optimization of Neural Networks II

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- A lot of parameters (W and b)
- Usually a lot of data



UPPSALA
UNIVERSITET

Optimization of Neural Networks II

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- A lot of parameters (W and b)
- Usually a lot of data
- Stochastic Gradient Descent, commonly
 - Adam



UPPSALA
UNIVERSITET

Optimization of Neural Networks II

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- A lot of parameters (W and b)
- Usually a lot of data
- Stochastic Gradient Descent, commonly
 - Adam
- To compute gradients: backpropagation
 - Chain-rule for derivatives



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- We need to have starting values for SGD - non-trivial
- Bad initial values might
 - Bad convergence (local optimum)
 - Numerical problems
- We want to break symmetry between layers
- Initialization can be seen as a hyperparameter
- Good practice
 - Initialize values randomly close to zero (uniform or normal)



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Neural Networks in Practice: TensorFlow and Keras

- Tensorflow
 - Framework for large-scale machine learning and Neural Networks
 - Developed by Google
 - Computational graphs
 - Handles:
 - Computing gradients for Neural Networks
 - Enable simple use of graphical processing units (GPU) and Tensor processing Units (TPU)
 - Used in both research and production
- Keras
 - Syntax for 'building' Neural Networks
 - Platform independent (ish)





UPPSALA
UNIVERSITET

Regularization of Neural Networks

- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Reduce training error but improve test/validation error



UPPSALA
UNIVERSITET

Regularization of Neural Networks

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Reduce training error but improve test/validation error
- Neural Networks are extremely flexible / high model capacity



UPPSALA
UNIVERSITET

Regularization of Neural Networks

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Reduce training error but improve test/validation error
- Neural Networks are extremely flexible / high model capacity
- Regularization is crucial for good generalizability of NN



UPPSALA
UNIVERSITET

Regularization of Neural Networks

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- Reduce training error but improve test/validation error
- Neural Networks are extremely flexible / high model capacity
- Regularization is crucial for good generalizability of NN



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

- Let

$$\tilde{J}(W, b) = J(W, b) + \alpha \Omega(W),$$

where $J(W, b)$ is the cost function and $\alpha \Omega(W)$ is the penalty for the weight matrices.

- α is the strength of the penalty.



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Weight decay / Norm penalty

- Let

$$\Omega_1(W) = \sum_i \sum_j |w|_{i,j},$$

and

$$\Omega_2(W) = \sum_i \sum_j w_{i,j}^2,$$

be the L_1 and L_2 regularization respectively.

- We can then get the cost function

$$\tilde{J}(W, b) = J(W, b) + \sum_l \alpha_l \Omega_2(W_l),$$



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Weight decay / Norm penalty

- Lets define the cost function as

$$\begin{aligned}\tilde{J}(w) &= J(w) + \alpha \Omega_2(w) \\ &= J(w) + \alpha w^T w\end{aligned}$$

- Then the gradient update becomes

$$\nabla_w \tilde{J}(w) = \nabla_w J(w) + 2\alpha w$$

- To update our weights with gradient descent

$$\begin{aligned}w &\leftarrow w - \epsilon(\nabla_w J(w) + 2\alpha w) \\ w &\leftarrow (1 - 2\alpha\epsilon)w - \epsilon\nabla_w J(w)\end{aligned}$$



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Weight decay / Norm penalty

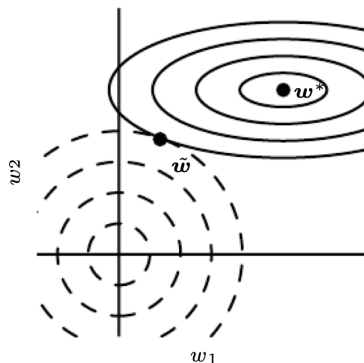


Figure: L_2 regularization (Goodfellow et al, 2017, Fig. 7.1)



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

Early Stopping

- Stop optimization early based on validation error
- Rerun to that number of epochs (hyperparameter)
- Can be shown to be equivalent (under strict assumptions) to L_2 regularization

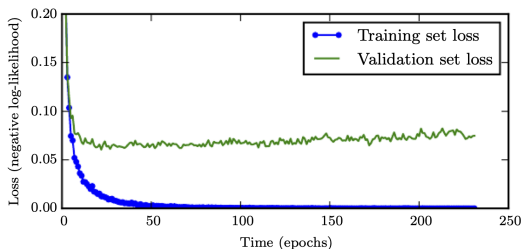


Figure: Early Stopping (Goodfellow et al, 2017, Fig. 7.3)



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

Early Stopping

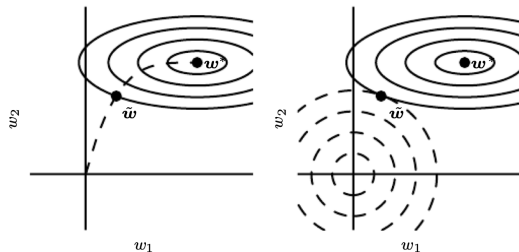


Figure: Early Stopping (Goodfellow et al, 2017, Fig. 7.4)



- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- In each iteration:
 - Sample an indicator I_i for each node i
 - Set the value h_i to 0 with probability p
- The dropout probability is typically 0.8 for input nodes and 0.5 for hidden nodes
- Forces the network to
 - not rely on individual nodes
 - spread out the weights over more nodes
- Can be seen as an ensemble method



- Introduction to Neural Networks

- Feed-Forward Neural Networks
- Feed-Forward Neural Networks
- Hyper-parameters

- Optimization

- Regularization

Dropout

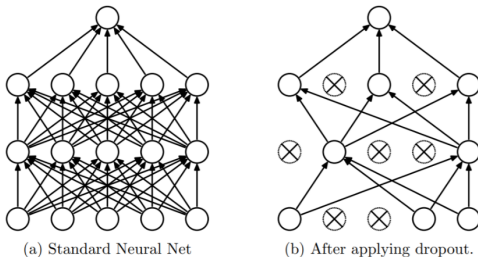


Figure 1: Dropout Neural Net Model. **Left:** A standard neural net with 2 hidden layers. **Right:** An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

Figure: Dropout (Srivastava et al, 2014)



UPPSALA
UNIVERSITET

Other regularization techniques

- Introduction to Neural Networks
 - Feed-Forward Neural Networks
 - Feed-Forward Neural Networks
 - Hyper-parameters
- Optimization
- Regularization

- In CNN: Dataset augmentation
- Get more data...