IN5550: Neural Methods in Natural Language Processing
Sub-lecture 3.1
Multi-layered neural networks

Andrey Kutuzov

University of Oslo

7 February 2023

Contents

Obligatory assignment

Time to get your hands dirty!

- Obligatory assignment 1 is published.
- Will work on related topics at the group sessions February 8 and 15.

IN5550 – Spring 2023

1. Bag of Words Document Classification

UiO Language Technology Group

Deadline: 16 February, at 21:59 (Devilry)

Goals

1. Learn how to use the Fox cluster to train deep learning models.
2. Get familiar with the *PyTorch* library
3. Learn how to use *PyTorch* to train and evaluate neural classifiers in NLP tasks.

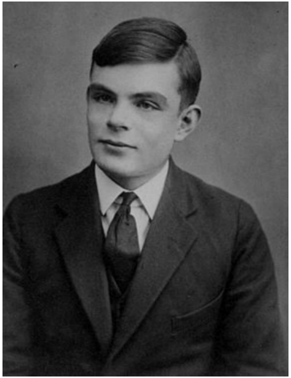
Contents

Multi-layered neural networks

'Machines of this character can behave in a very complicated manner when the number of units is large.'

Alan Turing, 'Intelligent Machinery'

[Turing, 1948]



Brain metaphor

- Why 'artificial neural networks'?

Multi-layered neural networks

Brain metaphor

- Why 'artificial neural networks'?
- ...because it seems neurons in human brain act similarly:

Multi-layered neural networks

Brain metaphor

- Why ‘artificial neural networks’?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,

Brain metaphor

- Why ‘artificial neural networks’?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,

Brain metaphor

- Why ‘artificial neural networks’?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,
  - processes (weighs) them differently, depending on their source,

Multi-layered neural networks

Brain metaphor

- Why 'artificial neural networks'?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,
  - processes (weighs) them differently, depending on their source,
  - and then passes own electrical signals (outputs) to other neurons;

Multi-layered neural networks

Brain metaphor

- Why 'artificial neural networks'?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,
  - processes (weighs) them differently, depending on their source,
  - and then passes own electrical signals (outputs) to other neurons;
  - depending on input signals, a neuron can be more or less activated,

Brain metaphor

- Why 'artificial neural networks'?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,
  - processes (weighs) them differently, depending on their source,
  - and then passes own electrical signals (outputs) to other neurons;
  - depending on input signals, a neuron can be more or less activated,
  - ... or completely relaxed ('silent');

Brain metaphor

- Why ‘artificial neural networks’?
- ...because it seems neurons in human brain act similarly:
  - Connected into a large network,
  - each neuron accepts electrical signals (inputs) from other neurons,
  - processes (weighs) them differently, depending on their source,
  - and then passes own electrical signals (outputs) to other neurons;
  - depending on input signals, a neuron can be more or less activated,
  - ... or completely relaxed (‘silent’);
  - the whole system is distributed across many neurons.

Multi-layered neural networks

...But brain metaphor is not optimal

- Although the first neural networks were based loosely on knowledge of neural activation at the time, we have long abandoned any real connection to neuroscience.

...But brain metaphor is not optimal

- Although the first neural networks were based loosely on knowledge of neural activation at the time, we have long abandoned any real connection to neuroscience.
- Thus, it's less cool but more convenient to think about artificial neural networks in terms of linear algebra concepts:

...But brain metaphor is not optimal

- Although the first neural networks were based loosely on knowledge of neural activation at the time, we have long abandoned any real connection to neuroscience.
- Thus, it's less cool but more convenient to think about artificial neural networks in terms of linear algebra concepts:
  - vectors,
  - matrices,
  - sequential algebraic operations on them.

The most basic neuron...

Multi-layered neural networks

Recall the equation for a linear classifier:

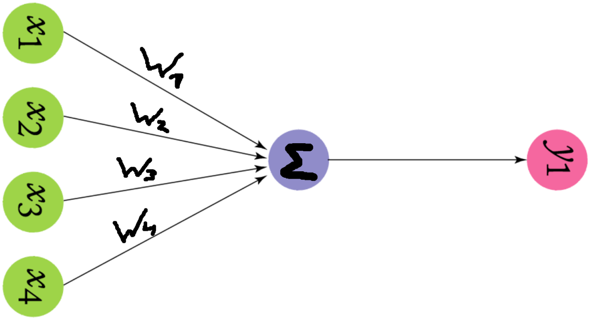
$$\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W} (+\boldsymbol{b}) \tag{1}$$

Multi-layered neural networks

Recall the equation for a linear classifier:

$$\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W}(+\boldsymbol{b}) \tag{1}$$

Suppose we have 4 features, binary classification task and no bias term $\boldsymbol{b}$:

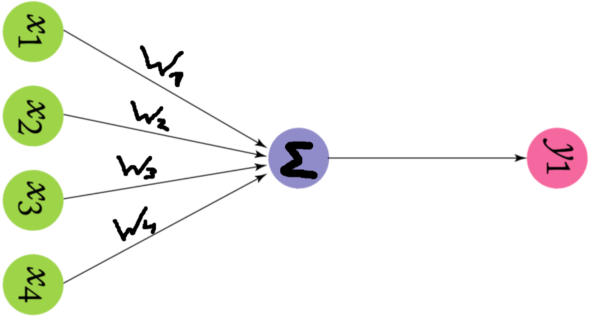


Multi-layered neural networks

Recall the equation for a linear classifier:

$$\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W}(+\boldsymbol{b}) \tag{1}$$

Suppose we have 4 features, binary classification task and no bias term $\boldsymbol{b}$:



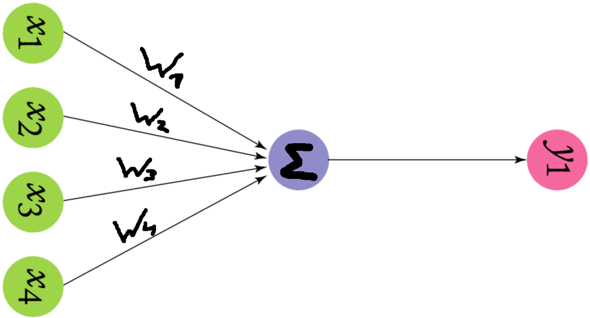
- input feature vector $\boldsymbol{x}$ is multiplied by the weights $\boldsymbol{W}$;

Multi-layered neural networks

Recall the equation for a linear classifier:

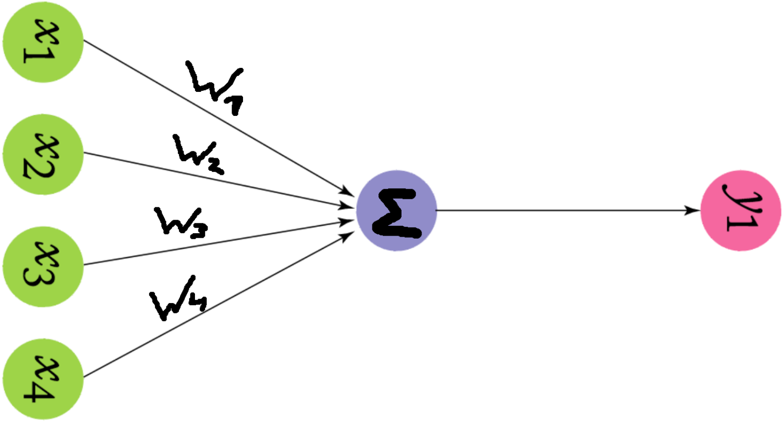
$$\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W}(+\boldsymbol{b}) \tag{1}$$

Suppose we have 4 features, binary classification task and no bias term $\boldsymbol{b}$:

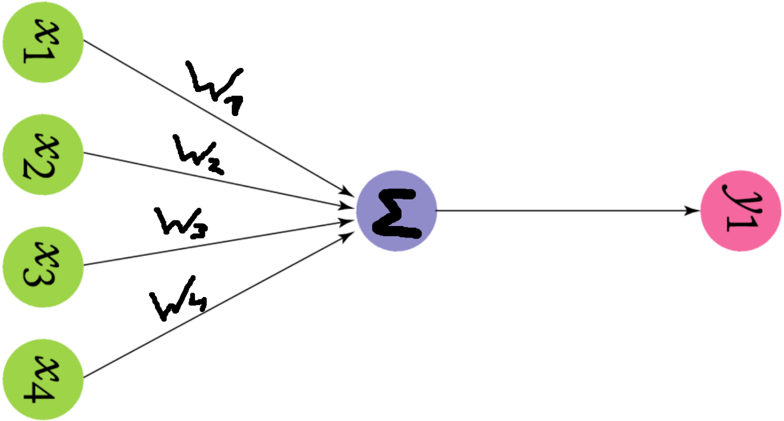


- input feature vector $\boldsymbol{x}$ is multiplied by the weights $\boldsymbol{W}$;
- here, $\boldsymbol{W}$ is a vector, so the result is a scalar value $\hat{y}_1$.

Multi-layered neural networks

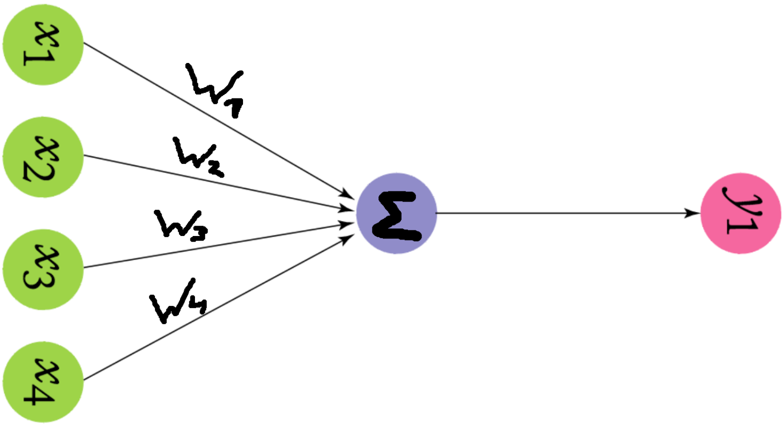


Multi-layered neural networks



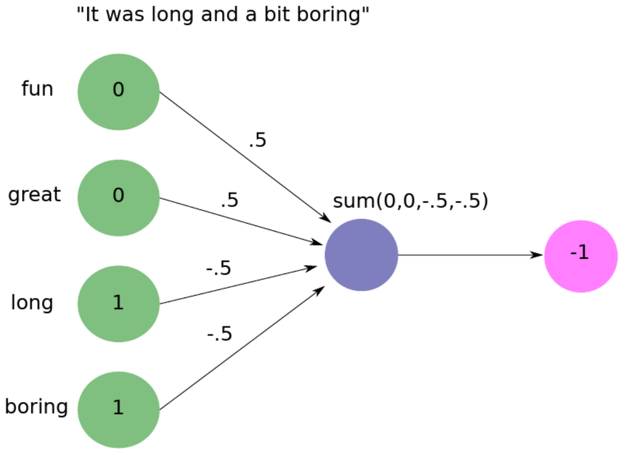
- In fact this is already a simple neural network…
- with one neuron Σ as a computational unit.

Multi-layered neural networks



- In fact this is already a simple neural network...
- with one neuron Σ as a computational unit.
- Σ takes 4 input values and returns their weighted sum as output value.

Multi-layered neural networks



Sentiment analysis with only one neuron and binary bag-of-words.

Multi-layered neural networks

Stack of linear classifiers

- This is a linear architecture
  $\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W}$
- We can make this classifier predict vectors instead of scalars...

Stack of linear classifiers

- This is a linear architecture
  $\hat{\boldsymbol{y}} = \boldsymbol{x} \cdot \boldsymbol{W}$
- We can make this classifier predict vectors instead of scalars...
  - ...by making $\boldsymbol{W}$ a matrix;
  - ...and thus using a row of several neurons, instead of only one.

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:
- $\hat{\boldsymbol{y}} = (\boldsymbol{x} \cdot \boldsymbol{W}^{\mathbf{0}}) \cdot \boldsymbol{W}^{\mathbf{1}}$

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:
- $\hat{\boldsymbol{y}} = (\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}$
- $\hat{\boldsymbol{y}} = ((\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}) \cdot \boldsymbol{W^2}$

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:
- $\hat{\boldsymbol{y}} = (\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}$
- $\hat{\boldsymbol{y}} = ((\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}) \cdot \boldsymbol{W^2}$
- ...etc. Can be arbitrarily deep

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:
- $\hat{y} = (\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}$
- $\hat{y} = ((\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}) \cdot \boldsymbol{W^2}$
- ...etc. Can be arbitrarily deep
- each subsequent row of neurons (layer) is a vector;

Multi-layered neural networks

Stack of linear classifiers

Additionally we can stack classifiers...

- by feeding the prediction of one classifier to another one:
- $\hat{\boldsymbol{y}} = (\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}$
- $\hat{\boldsymbol{y}} = ((\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}) \cdot \boldsymbol{W^2}$
- ...etc. Can be arbitrarily deep
- each subsequent row of neurons (layer) is a vector;
- it is multiplied by the next weight matrix (shapes must match).

Multi-layered neural networks

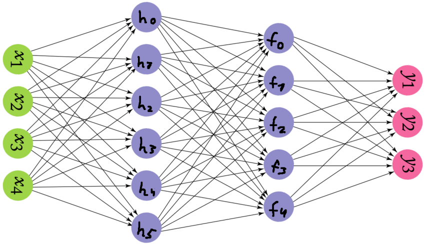
Stack of linear classifiers

Additionally we can stack classifiers...

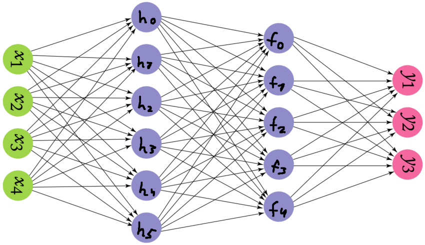
- by feeding the prediction of one classifier to another one:
- $\hat{\boldsymbol{y}} = (\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}$
- $\hat{\boldsymbol{y}} = ((\boldsymbol{x} \cdot \boldsymbol{W^0}) \cdot \boldsymbol{W^1}) \cdot \boldsymbol{W^2}$
- ...etc. Can be arbitrarily deep
- each subsequent row of neurons (layer) is a vector;
- it is multiplied by the next weight matrix (shapes must match).

Can even use both multiple neurons and multiple layers.

Multi-layered neural networks



Multi-layered neural networks



Looks much like a 'deep neural network'

Stacked linear classifier with multiple computational units at each layer:

$$h = \boldsymbol{x} \cdot \boldsymbol{W^0} \tag{2}$$

$$f = \boldsymbol{h} \cdot \boldsymbol{W^1} \tag{3}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{f} \cdot \boldsymbol{W^2} \tag{4}$$

Multi-layered neural networks

- But any stack of linear classifiers is still a linear classifier :-(

Multi-layered neural networks

- But any stack of linear classifiers is still a linear classifier :-(
- Still can't handle XOR and other non-linear problems.

Multi-layered neural networks

Non-linear transformations of the input data make the desired difference.

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.
- They help induce good input data representations.

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.
- They help induce good input data representations.
- These representations are processed with a linear classifier in the end.

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.
- They help induce good input data representations.
- These representations are processed with a linear classifier in the end.

Popular activation functions

- Sigmoid (logistic function):
  $\sigma(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{x}}} \rightarrow [0, 1]$

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.
- They help induce good input data representations.
- These representations are processed with a linear classifier in the end.

Popular activation functions

- Sigmoid (logistic function):
  $\sigma(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{x}}} \rightarrow [0, 1]$
- REctified Linear Unit (**ReLU**) [Glorot et al., 2011]:

$$relu(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}, & \text{if } \boldsymbol{x} \geq 0 \\ 0, & \text{otherwise} \end{cases} \rightarrow [0, \infty]$$

Multi-layered neural networks

- Differentiable non-linear transformations (activations) are the core reason of deep learning's substantial results, including in NLP.
- They help induce good input data representations.
- These representations are processed with a linear classifier in the end.

Popular activation functions

- Sigmoid (logistic function):
  $\sigma(x) = \frac{1}{1+e^{-x}} \rightarrow [0,1]$
- REctified Linear Unit (**ReLU**) [Glorot et al., 2011]:

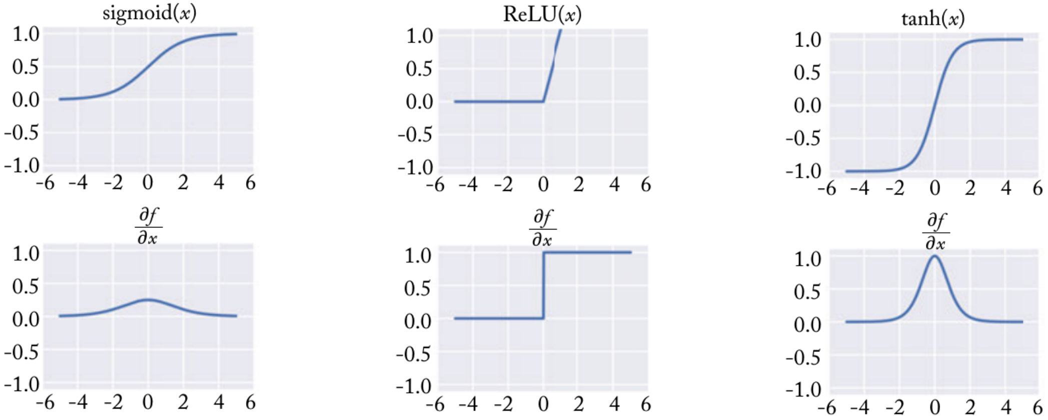
$$relu(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \rightarrow [0, \infty]$$

- Hyperbolic tangent:

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \rightarrow [-1, 1]$$

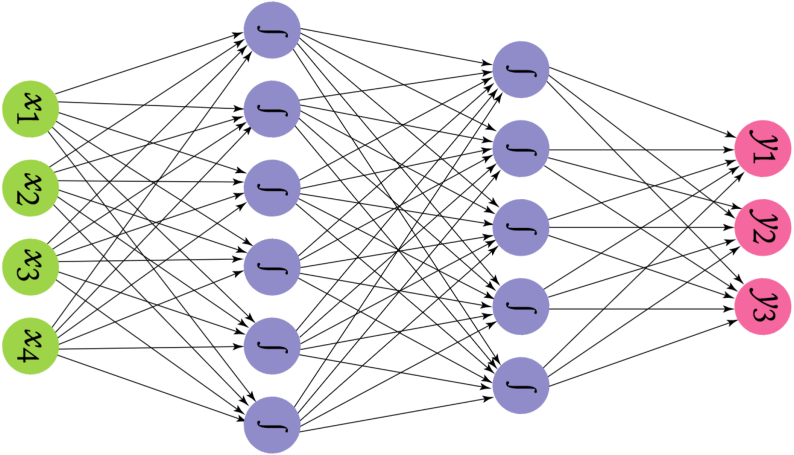
Multi-layered neural networks

Popular activation functions...



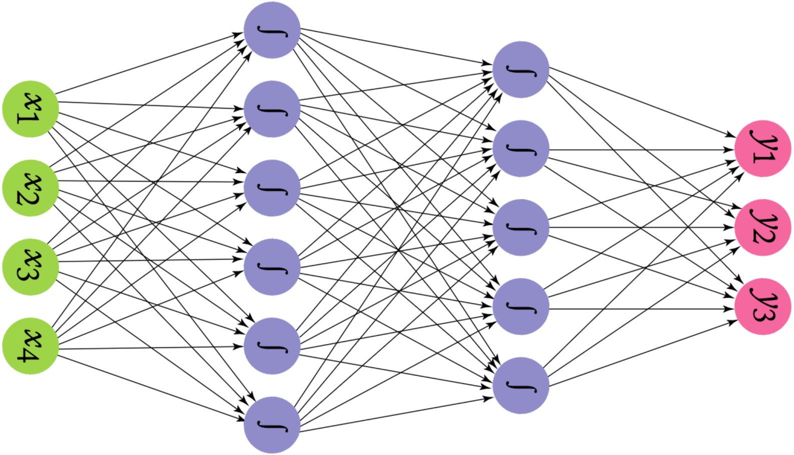
...and their derivatives.

Multi-layered neural networks



The only difference between this deep neural network and a stack of linear classifiers: non-linearities ($\int$) between linear transformations.

Multi-layered neural networks



The only difference between this deep neural network and a stack of linear classifiers: non-linearities ($\int$) between linear transformations.

In the next sub-lecture 3.2: why 'feed-forward' and 'fully connected'?

References I

Glorot, X., Bordes, A., and Bengio, Y. (2011).
Deep sparse rectifier neural networks.
In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pages 315–323.

Turing, A. (1948).
Intelligent machinery.
Technical report, National Physical Laboratory.