Mock Exam – WS 25/26

Thang Vu

Task 1

Exercise 1: Basics in Machine Learning [/11 points]

(a) [2 points] What is the main difference between a regression and a classification task? Name one regression task and one classification task in speech or natural language processing.

Supervised Learning

- Supervised learning: predict target y from input x
  - *Training data x is given with its corresponding label, y*
  - Regression: y is a real-valued number
  - Classification: y represents a category or class

Task 1

(c) [5 points] Mark whether the following statements are True or False.

Statement	**True**	**False**
The goal of preventing overfitting is to better generalize to unseen data.	✖	
Regularization typically increases the error on the development set.		✖
Support vector machine can only be used for binary classification.		✖
K-means is a partitional clustering method.	✖	
Hyperparameters are tuned using the test set.		✖

Task 2

(d) [5 points] Mark whether the following statements are True or False.

Statement	**True**	**False**
Each layer in a NN consists of a linear transformation and a non-linear activation.	✖	
Cross-entropy loss is often used for multi-class, multi-label classification.		✖
For the backward-pass, we always first need to compute δ^l for the first layer.		✖
Neural networks typically consist of multiple layers.	✖	
$\mathrm{ReLU}(z) = \min(0, z)$		✖

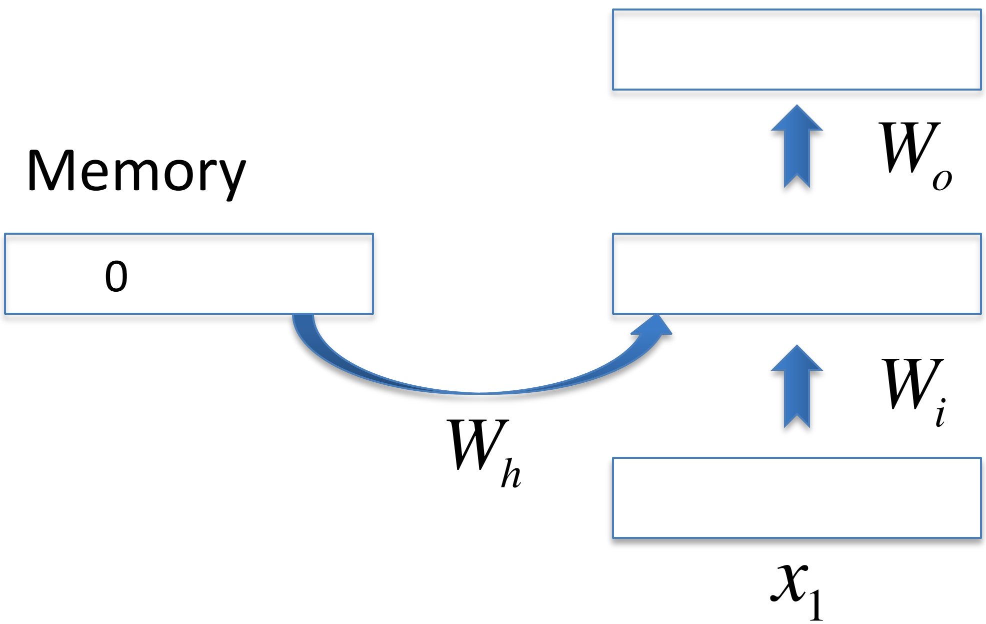
Task 3

(a) [2 points] What is the difference of RNNs to common (feed-forward) neural networks? What type of inputs do RNNs typically handle?

RNN

- Input data: $[x_1, x_2, x_3, \ldots, x_n]$

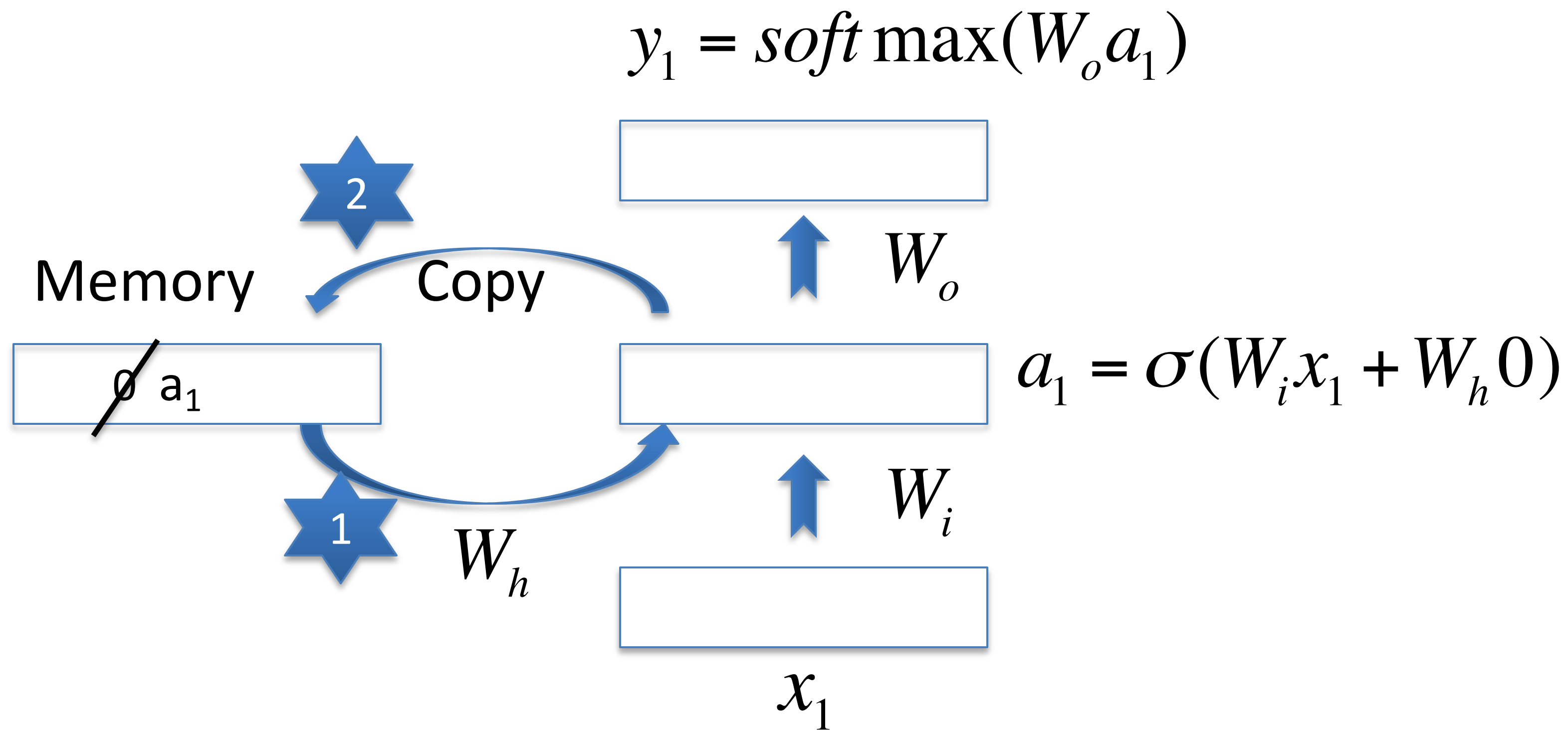
$$y_1 = softmax(W_o a_1)$$



$$a_1 = \sigma(W_i x_1 + W_h 0)$$

RNN

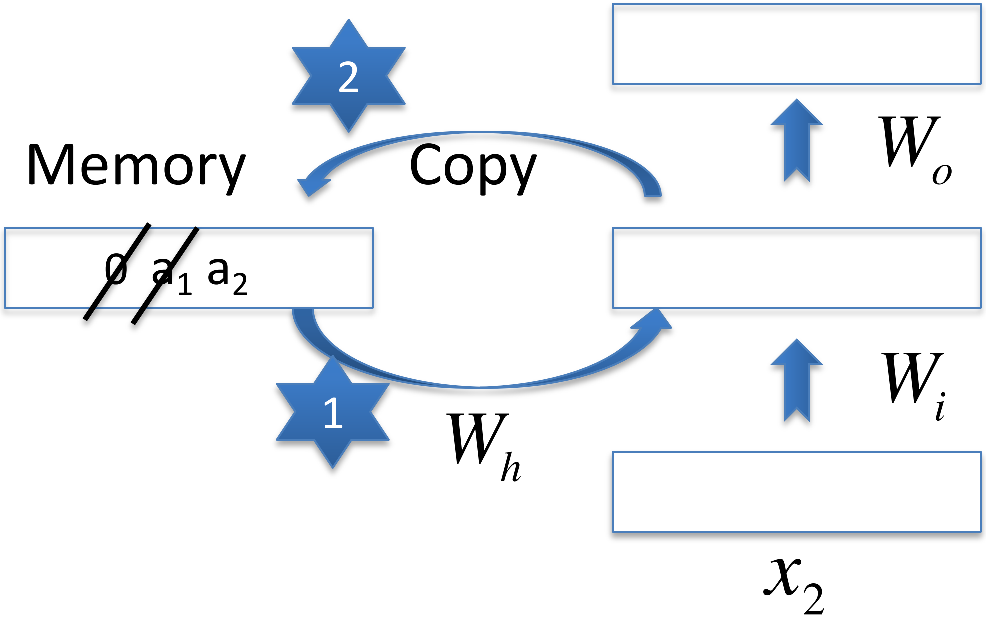
- Input data: $\left[x_1, x_2, x_3, \ldots, x_n\right]$



RNN

- Input data: $\left[x_1, x_2, x_3, \ldots, x_n\right]$

$$y_2 = softmax(W_o a_2)$$

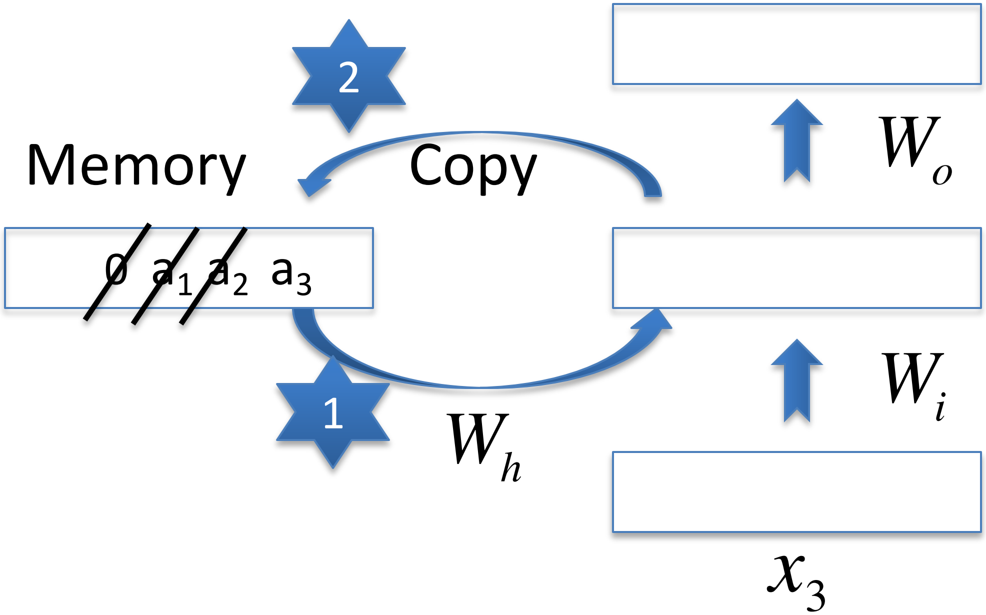


$$a_2 = \sigma(W_i x_2 + W_h a_1)$$

RNN

- Input data: $[x_1, x_2, x_3, \ldots, x_n]$

$$y_3 = softmax(W_o a_3)$$

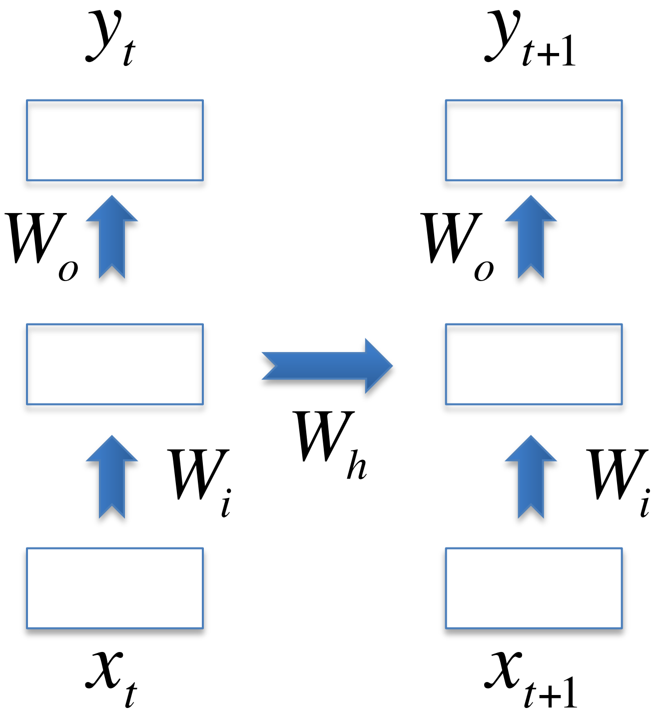


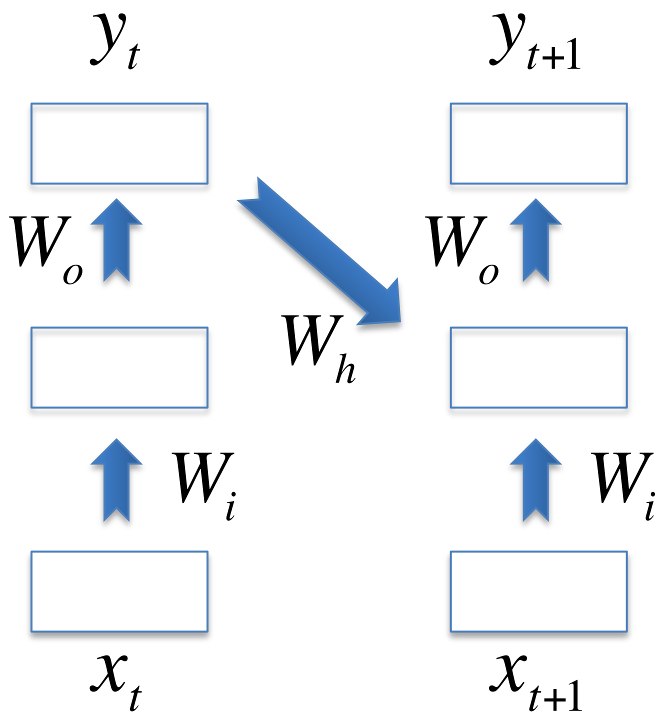
$$a_3 = \sigma(W_i x_3 + W_h a_2)$$

Task 3

(b) [2 points] How does an Elman RNN compute the hidden state at timestep t? Provide the formula and define the variables.

Elman Network & Jordan Network





Task 3

(d) [4 points] Mark whether the following statements are True or False.

Statement	**True**	**False**
RNNs are trained using backpropagation through time.	X	
LSTMs suffer from vanishing gradients.		X
The last hidden state of an RNN can contain information from the whole input sequence.	X	
When employing RNNs, we always need to pad the inputs.		X

Task 4

(a) [2 points] What is the general motivation for CNNs? Name two reasons.

Challenges

- Challenge 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

Challenges

- Challenge 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

→

0
2
4
3
1
3
0
4

...

Challenges

- Challenge 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

→

0
2
4
3
1
3
0
4

…

- #parameters ↗

Challenges

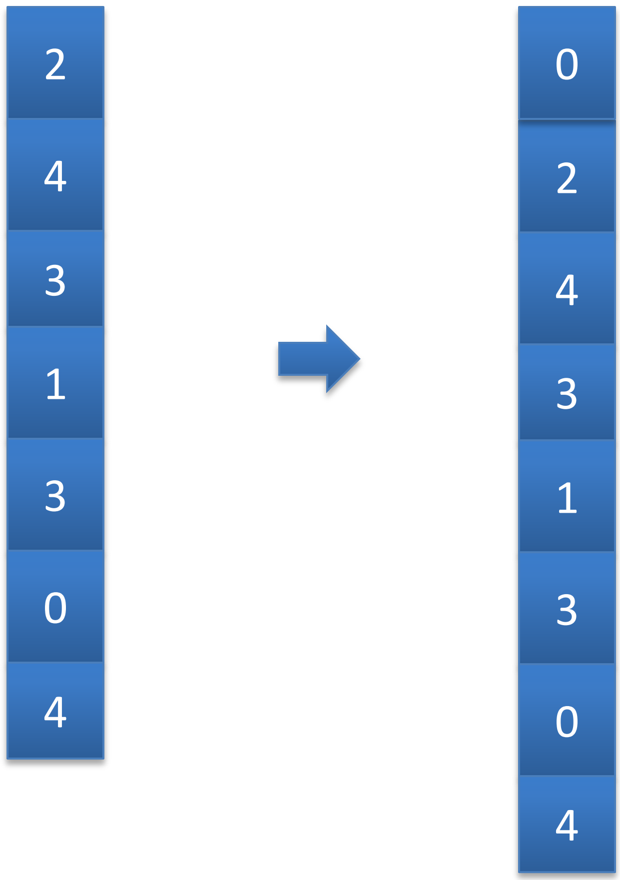
- Challenge 1: What if the input is a matrix?

0	1	0	0	1
2	3	3	3	4
4	0	1	0	1
3	4	2	3	3

- Don't want to blow up the number of parameters
  - If possible, reduce the number of parameters

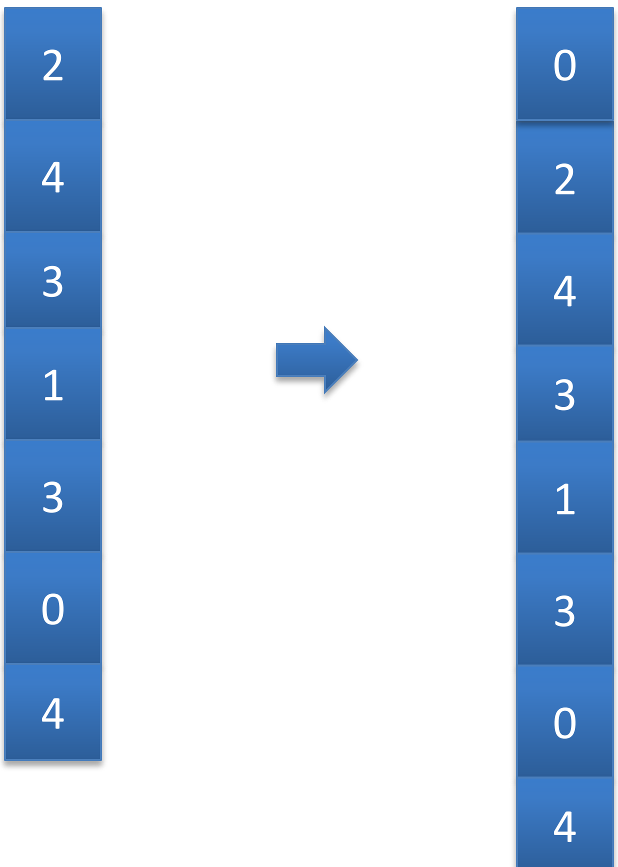
Challenges

- Challenge 2: What if the input is shifted?



Challenges

- Challenge 2: What if the input is shifted?

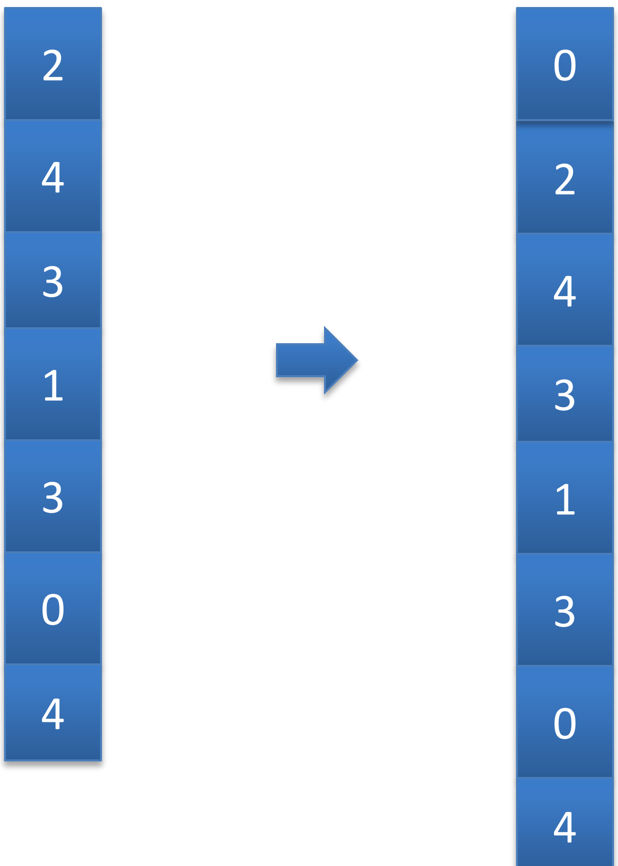


Real-world example 1:

- Language processing:
  - Input 1: „I am happy today“
  - Input 2: „Today I am happy“

Challenges

- Challenge 2: What if the input is shifted?

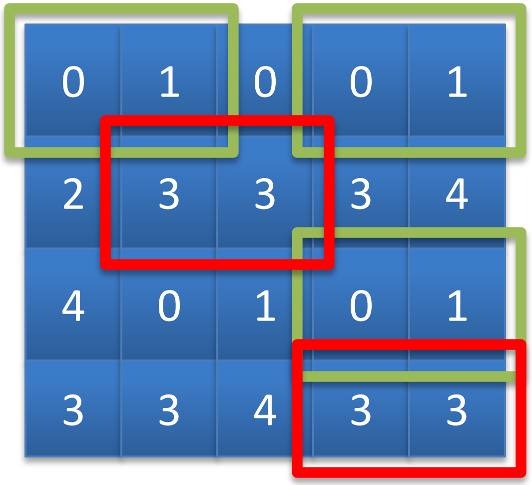


Real-world example 2:

- Computer vision:

Challenges

- Challenge 2: What if the input is shifted?



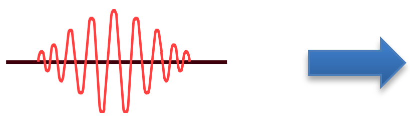
- Wish to recognize patterns
  - Position independent

Task 4

(b) [2 points] Give one example for a speech task and one for a text task of how the inputs to a CNN can be represented.

CNN for Speech Emotion Recognition [Neumann, Vu, 2017]

- Extract emotion from speech



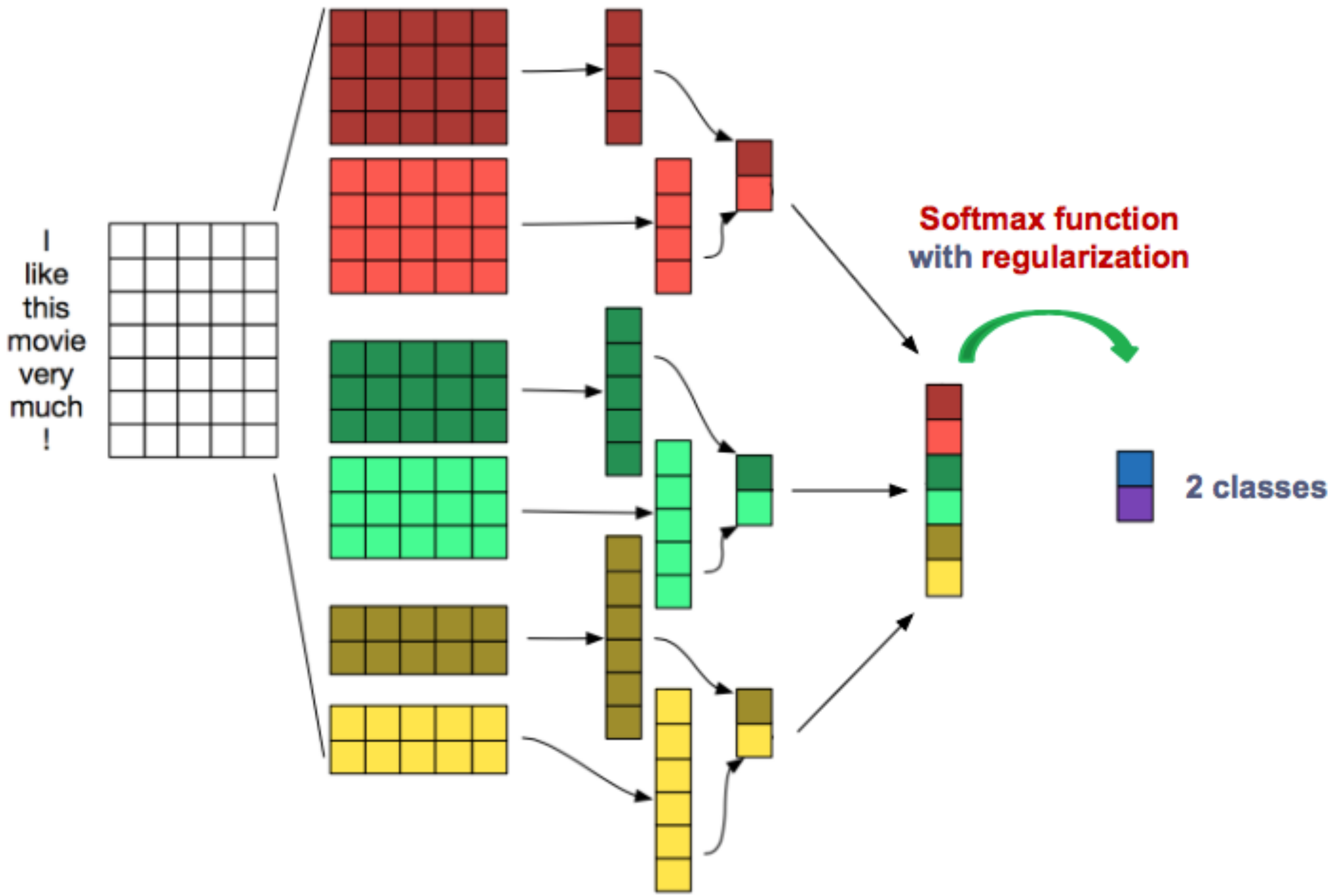
Which emotion? (happy, anger, sad or neutral)

CNN for Speech Emotion Recognition [Neumann, Vu, 2017]

- Extract emotion from speech

happy,
anger
sad
neutral

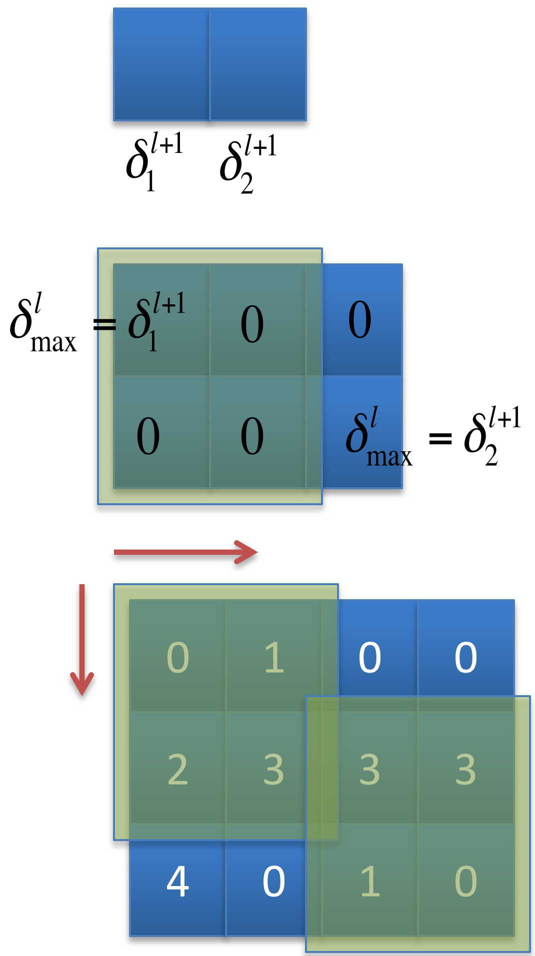
CNN for Natural Language Processing



Task 4

(e) [1 point] How do we compute the derivative of a max pooling layer?

Gradient Computation for CNN



- Gradient of the convolution is a sum over all gradients of the shared parameters
- Gradient of the max pooling layer depends on the indices of the largest value
  - Need to store the indices of the largest value

Task 4

(f) [4 points] Mark whether the following statements are True or False.

Statement	**True**	**False**
The number of filters is a hyper-parameter.	X	
The size of the filters are parameters.		X
CNNs are a special case of feed-forward neural networks.	X	
The filter weights are parameters.	X	

Task 5

(a) [2 points] Name a property of the tasks on which sequence-to-sequence models can be applied. Also name a task for which sequence-to-sequence models can be applied.

Sequence to sequence (Seq2seq)

- The first paper:
  Sequence to sequence learning with neural networks – Sutskever, Vinyals, Le - 2014

$$y^* = \arg\max_y p(y|x)$$

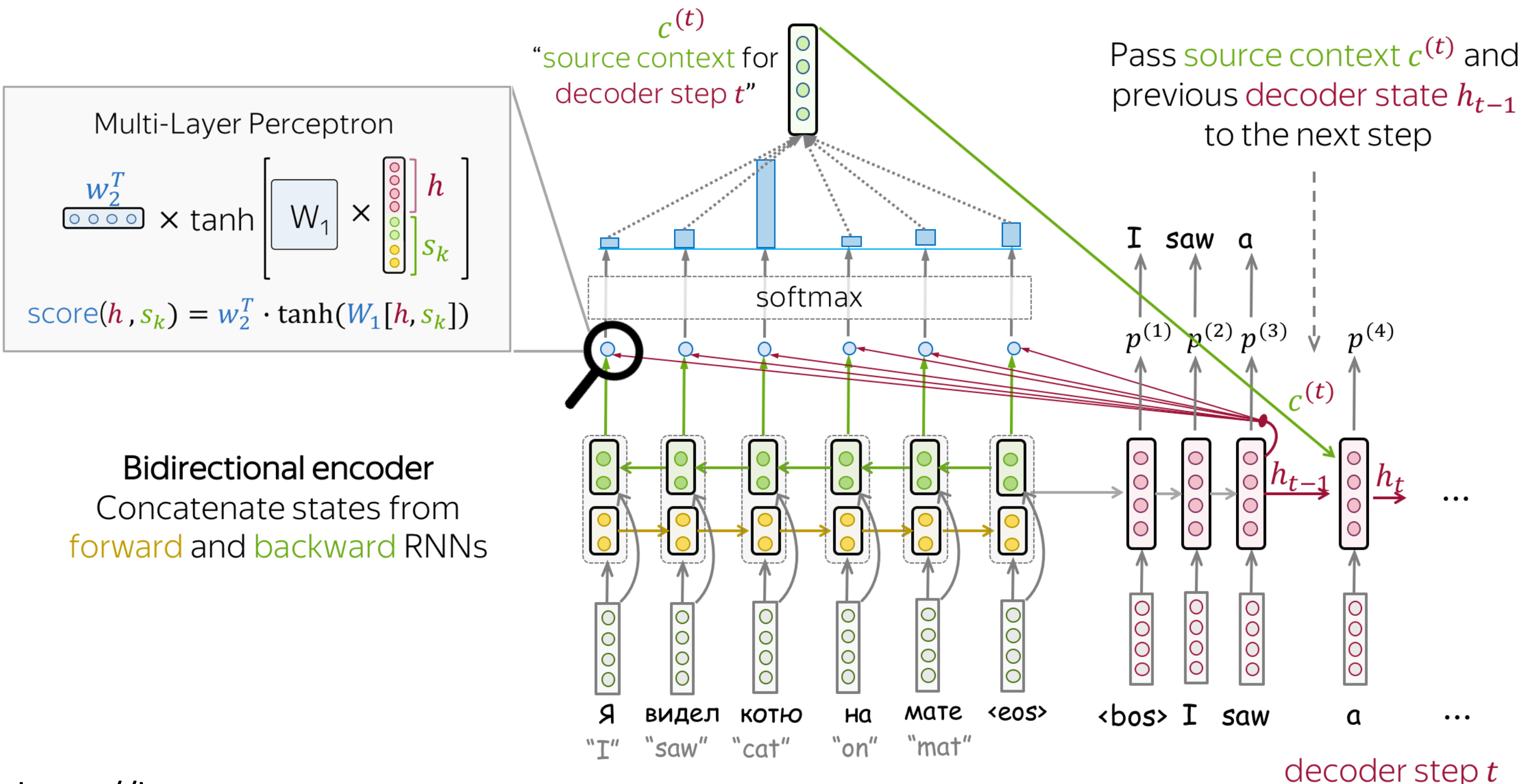
Sequence to sequence (Seq2seq)

- Sequence-to-sequence tasks are everywhere:
  - Speech Recognition
  - Machine translation
  - Text summarization
  - Conversational modeling
  - Image captioning

Task 5

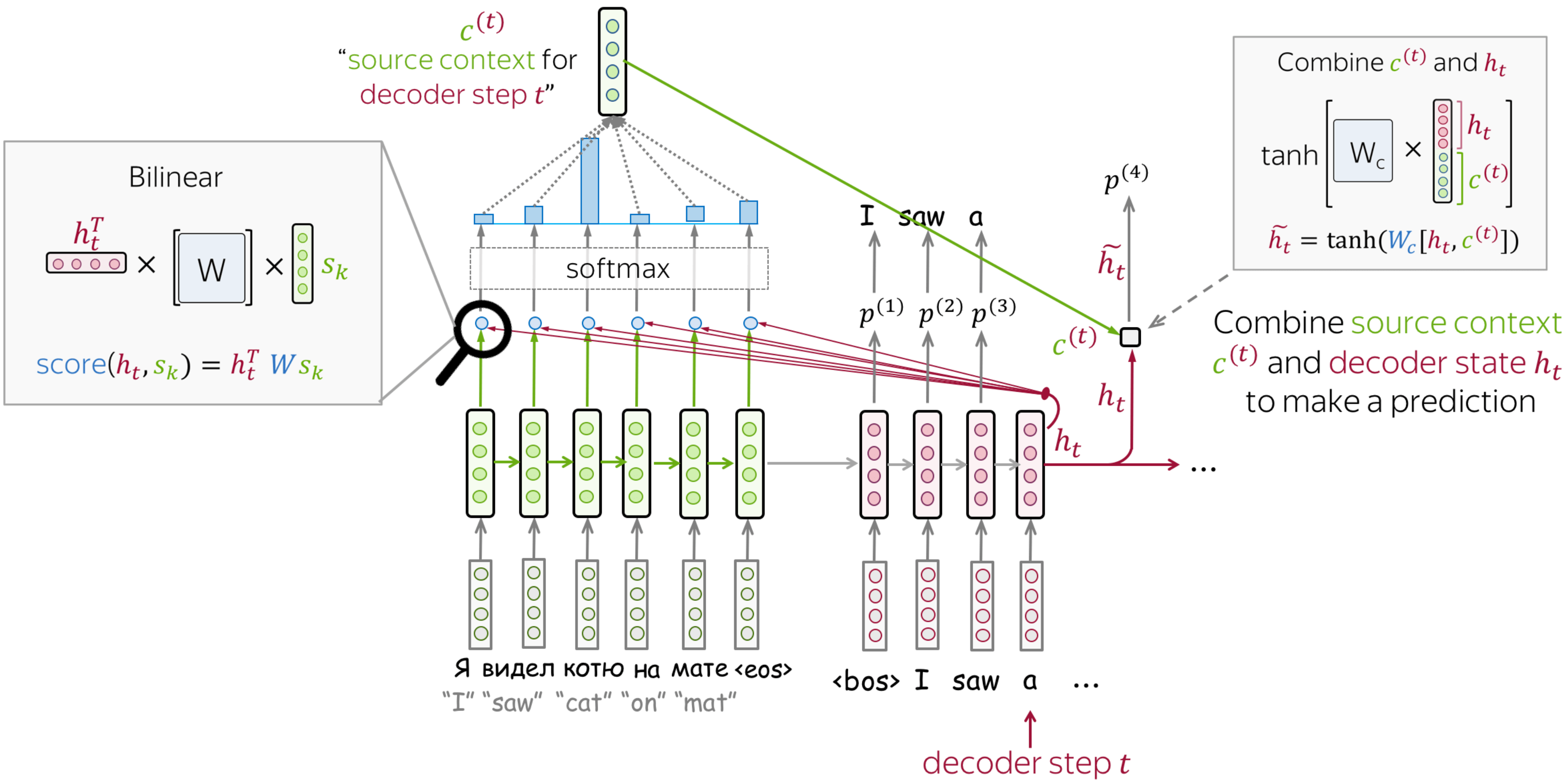
(b) [2 points] For two attention scoring methods, explain how the hidden states of an encoder model are used by the decoder model. Both, the encoder and decoder are recurrent neural networks.

Bahdanau et al 2014



https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Luong et al 2015



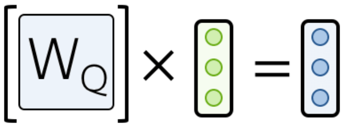
https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Task 5

(c) [2 points] How are the queries, keys and values computed in self-attention? What are the learnable weights in this operation?

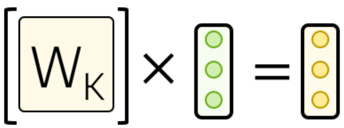
Self-Attention in Details

Each vector receives three representations ("roles")



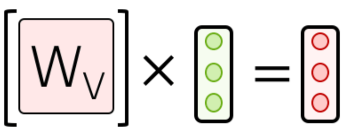
Query: vector **from** which the attention is looking

"Hey there, do you have this information?"



Key: vector **at** which the query looks to compute weights

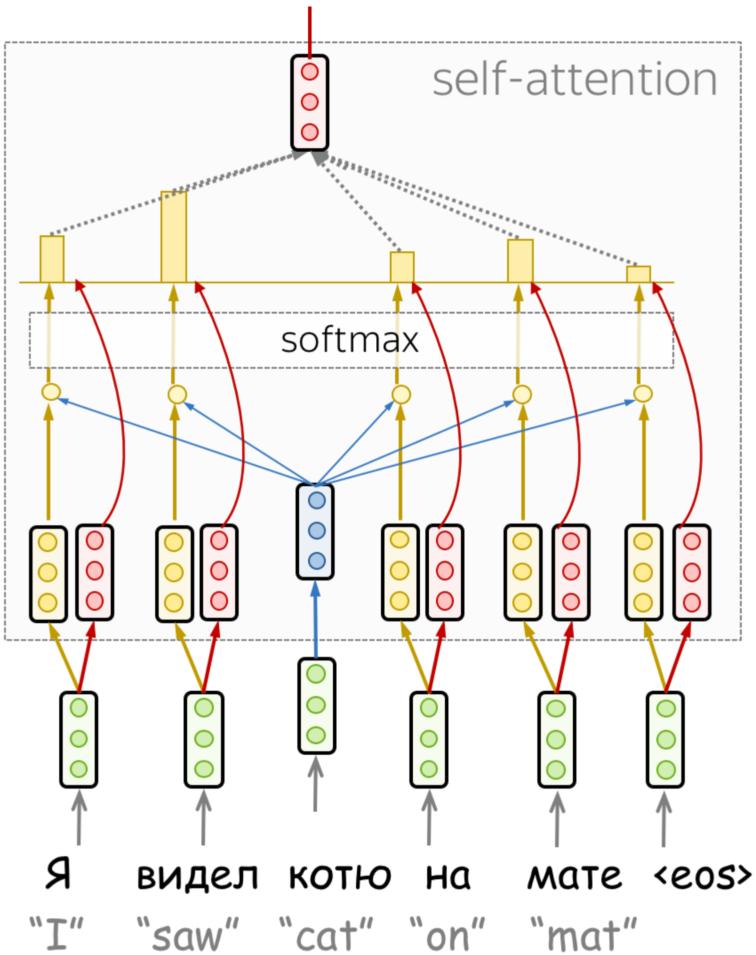
"Hi, I have this information – give me a large weight!"



Value: their weighted sum is attention output

"Here's the information I have!"

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

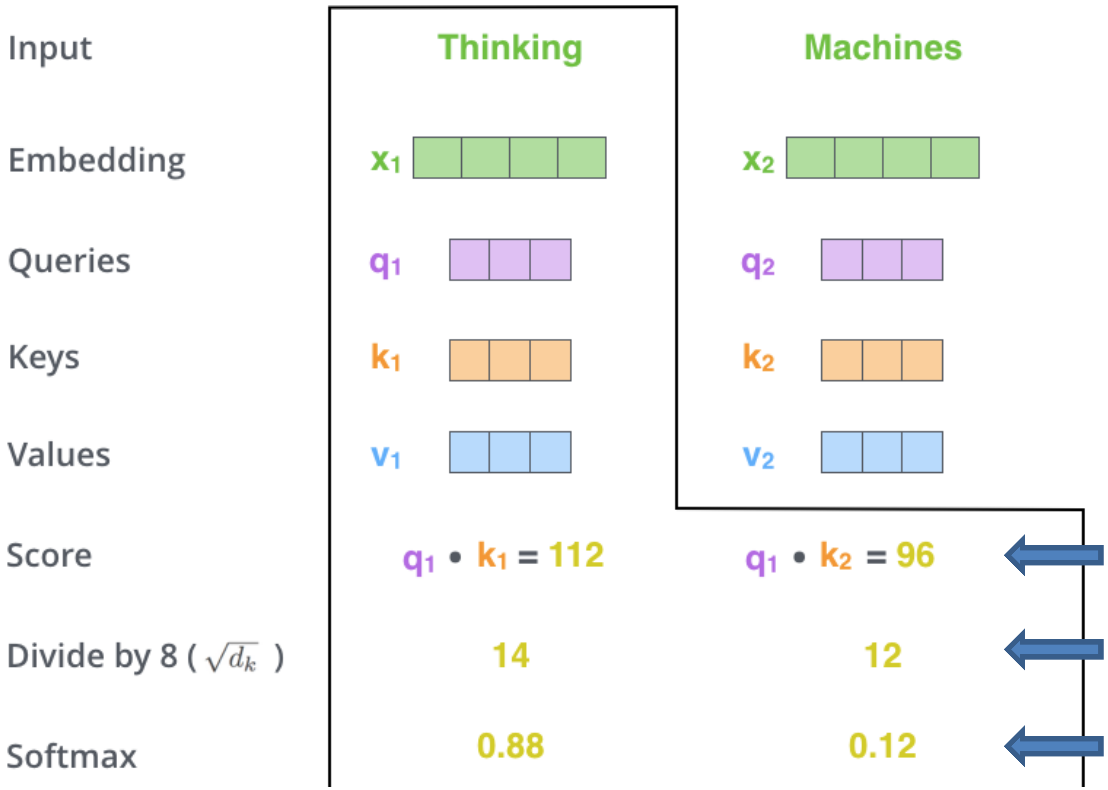


Task 5

(d) [4 points] In the following figure on self-attention operations[1], shortly describe steps 1, 3 and 4. What is the purpose of step 2?

Self Attention in Details

- Step 2: Calculate scores for each word with others



scoring function

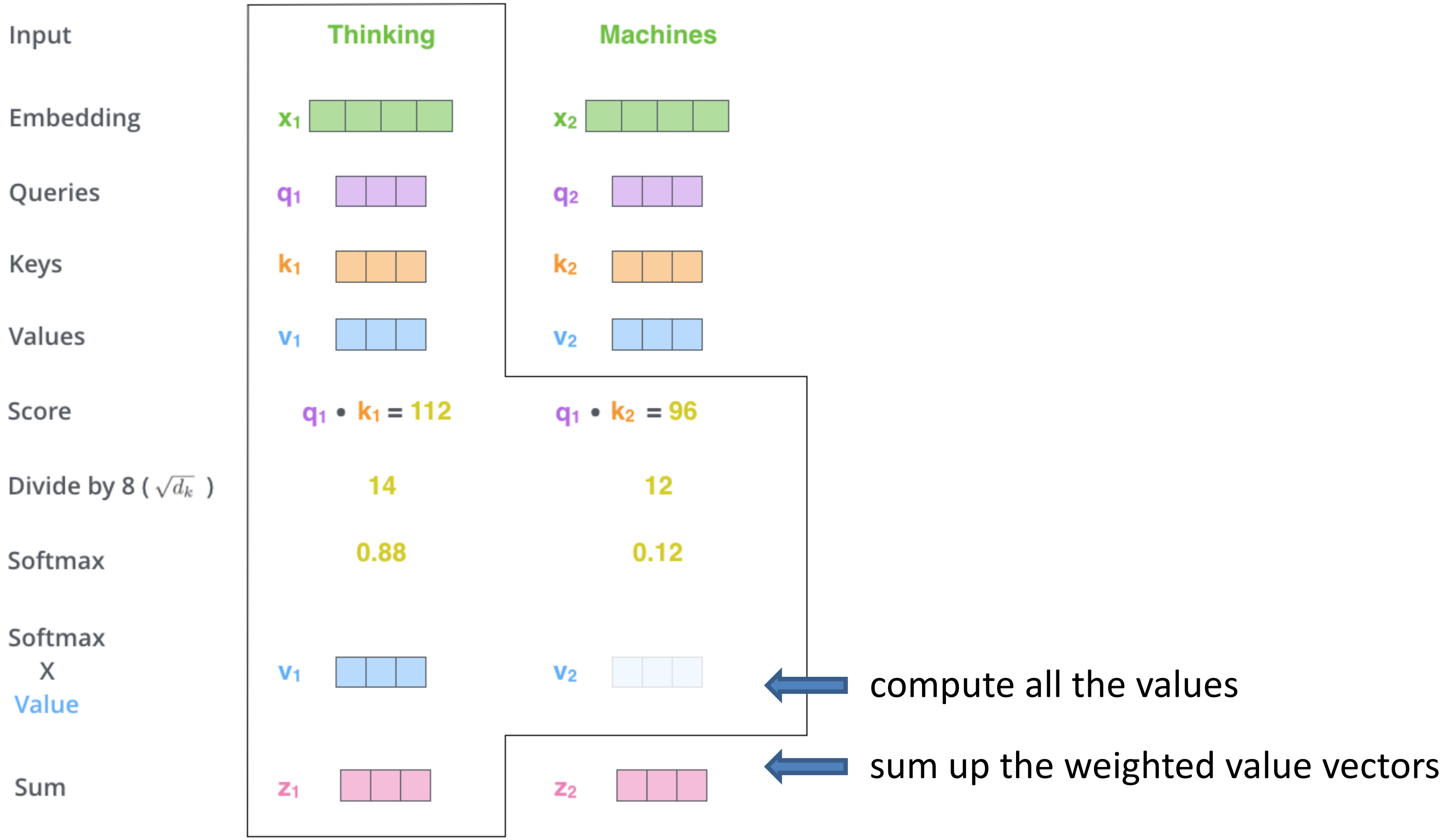
trick to have more stable gradients

normalization step

http://jalammar.github.io/illustrated-transformer/

Self Attention in Details

- Step 3: Compute the values and the output



http://jalammar.github.io/illustrated-transformer/

Task 6

(a) [1 point] Explain the main difference between Stochastic Gradient Descent and Mini-Batch Gradient Descent.

Stochastic Gradient Descent

- Gradient Descent:

$$\theta_i \leftarrow \theta_{i-1} - \eta \nabla C(\theta_{i-1})$$

  - In which

$$\nabla C(\theta_{i-1}) = \frac{1}{R}\sum_r \nabla C^r(\theta)$$

- Stochastic Gradient Descent:
  - Pick an example x^r

$$\theta_i \leftarrow \theta_{i-1} - \eta \nabla C^r(\theta_{i-1})$$

Stochastic Gradient Descent

- *What is an epoch?*
- Training data: $(x_1, y_1), (x_2, y_2), \ldots, (x_R, y_R)$
- When using the stochastic gradient descent:
  - Starting with θ_0
  - Pick (x_1, y_1) $\quad \theta_1 \leftarrow \theta_0 - \eta \nabla C^1(\theta_0)$

    (x_2, y_2) $\quad \theta_2 \leftarrow \theta_1 - \eta \nabla C^2(\theta_1)$

    (x_R, y_R) $\quad \theta_R \leftarrow \theta_{R-1} - \eta \nabla C^R(\theta_{R-1})$

Seen all the training data

One epoch

(x_1, y_1) $\quad \theta_{R+1} \leftarrow \theta_R - \eta \nabla C^{R+1}(\theta_R)$

Stochastic Gradient Descent

- Mini-batch Gradient Descent:
  - Pick B examples as a batch b
  - B is the batch size

$$\theta_i \leftarrow \theta_{i-1} - \eta \frac{1}{B} \sum_{x_r \in b} \nabla C^r(\theta_{i-1})$$

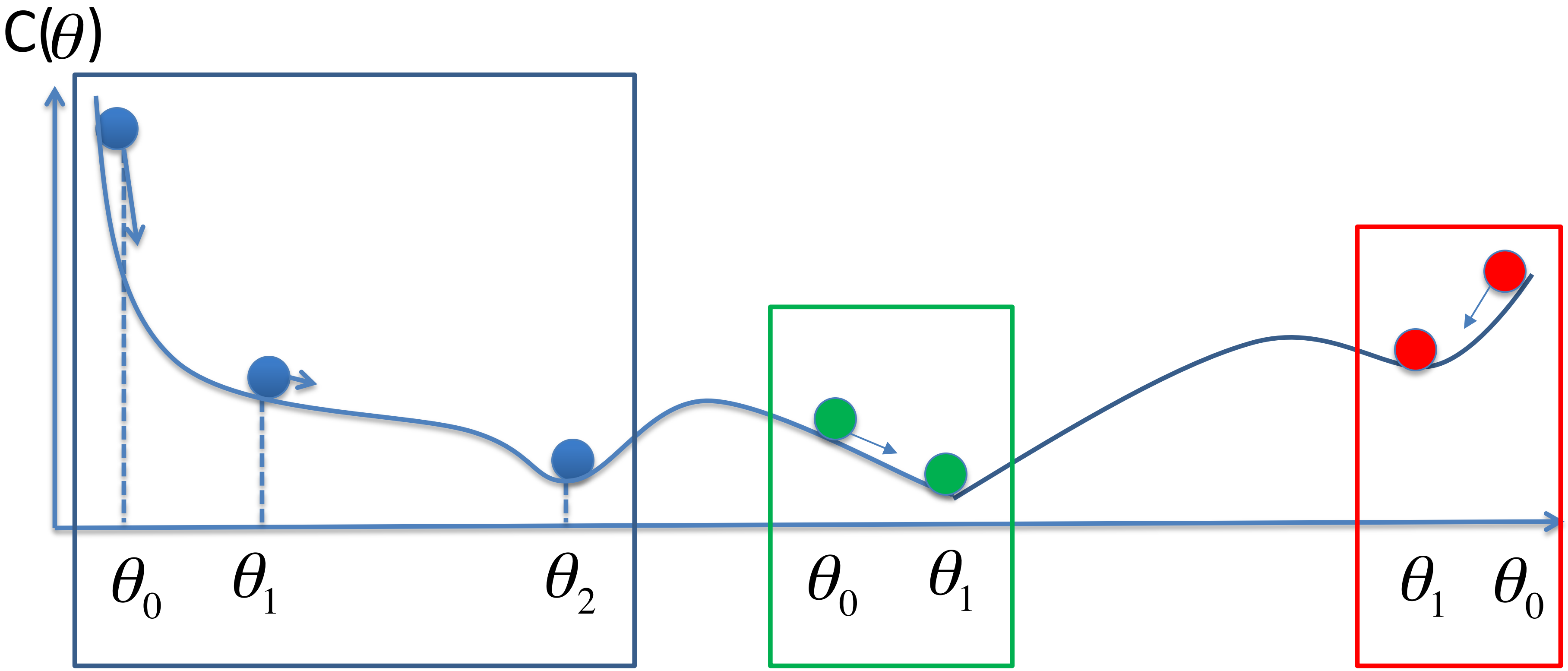
- Mini-batch Gradient Descent is faster than Stochastic Gradient Descent
  - Less updates
  - Better parallelization
- <u>Important:</u> Shuffle the data after each epoch

Task 6

(b) [3 points] Why does parameter initialization matter? Name and describe two methods.

Challenges of Gradient Descent

- Depending on the initialization points, we will obtain different models and, therefore different results



Uniform or Normal Distribution

- Uniform
  - Values are drawn from a uniform distribution U(a,b)
  - a is lower bound, e.g. 0 and
    b is the upper bound, e.g. 1
- Normal
  - Values are drawn from a normal distribution
    N(mean, std^2)
  - mean is the mean value, e.g. 0
    and std is the standard deviation, e.g. 1

Xavier or Glorot Methods

- Understanding the difficulty of training deep feedforward neural networks, Xavier Glorot and Yoshua Bengio, 2010
- Uniform:
  - Values are drawn from a uniform distribution U(-a,a)

$$a = gain \, . \sqrt{\frac{6}{fan_{in} + fan_{out}}}$$

- Normal:
  - Values are drawn from a normal distribution N(0, std^2)

$$std \ = gain \, . \sqrt{\frac{2}{fan_{in} + fan_{out}}}$$

Kaiming or He Methods

- Delving Deep into Rectifiers:
  Surpassing Human-Level Performance on ImageNet Classification, Kaiming He et al 2015
- Uniform:
  - Values are drawn from a uniform distribution U(-b,b)

$$b = gain\ .\sqrt{\frac{3}{fan_{mode}}}$$

mode could be either in or out

- Normal:
  - Values are drawn from a normal distribution N(0, std^2)

$$std\ = gain\ .\sqrt{\frac{2}{fan_{mode}}}$$

Task 6

(c) [2 points] What is weight decay and what is its purpose?

Weight Decay

- It is also well known as ‚Regularization' (L2)
- New cost function to be minimized

$$C'(\theta) = C(\theta) + \lambda \frac{1}{2} \|\theta\|^2$$

Regularization term

$$\theta = W^1, W^2, \ldots$$

- Original cost to minimize

  (e.g. cross entropy)

Weight Decay

- It is also well known as ‚Regularization‘ (L2)
- New cost function to be minimized

$$C'(\theta) = C(\theta) + \lambda \frac{1}{2} \|\theta\|^2$$ Gradient: $$\frac{\partial C'}{\partial w} = \frac{\partial C}{\partial w} + \lambda w$$

- Update:

$$w^{t+1} \leftarrow w^t - \eta \frac{\partial C'}{\partial w} = w^t - \eta \left(\frac{\partial C}{\partial w} + \lambda w^t \right)$$

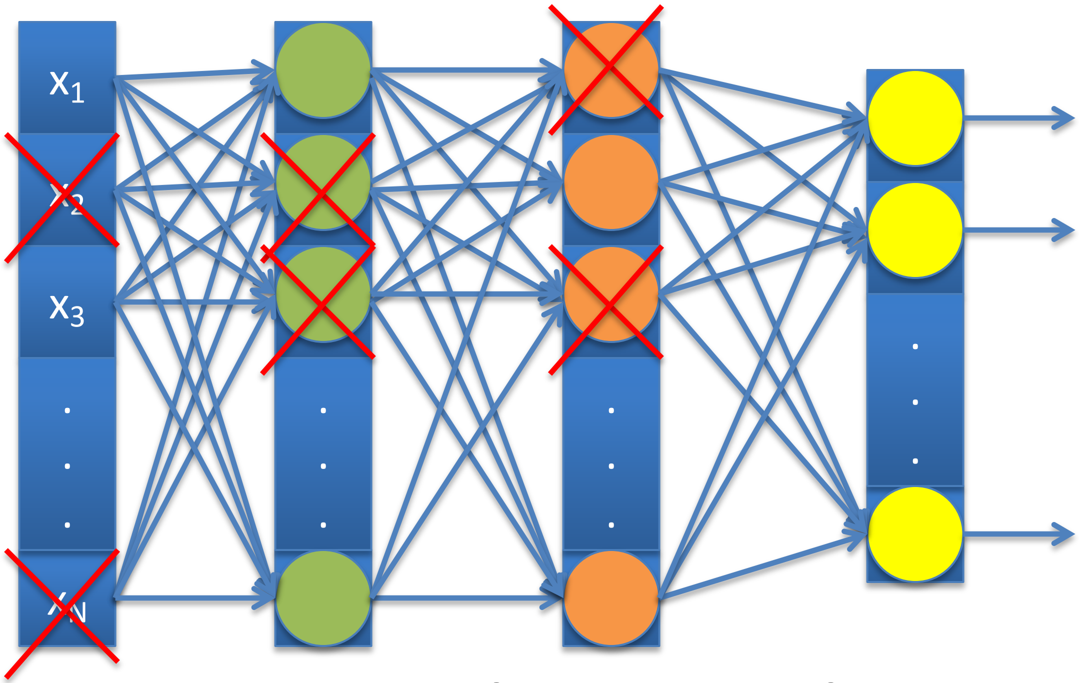
$$= \underbrace{(1 - \eta\lambda) w^t}_{\text{smaller and smaller}} - \eta \frac{\partial C}{\partial w}$$

Task 6

(d) [2 points] How do the outputs of neurons using dropout have to be scaled during testing if we do not want to scale during training?

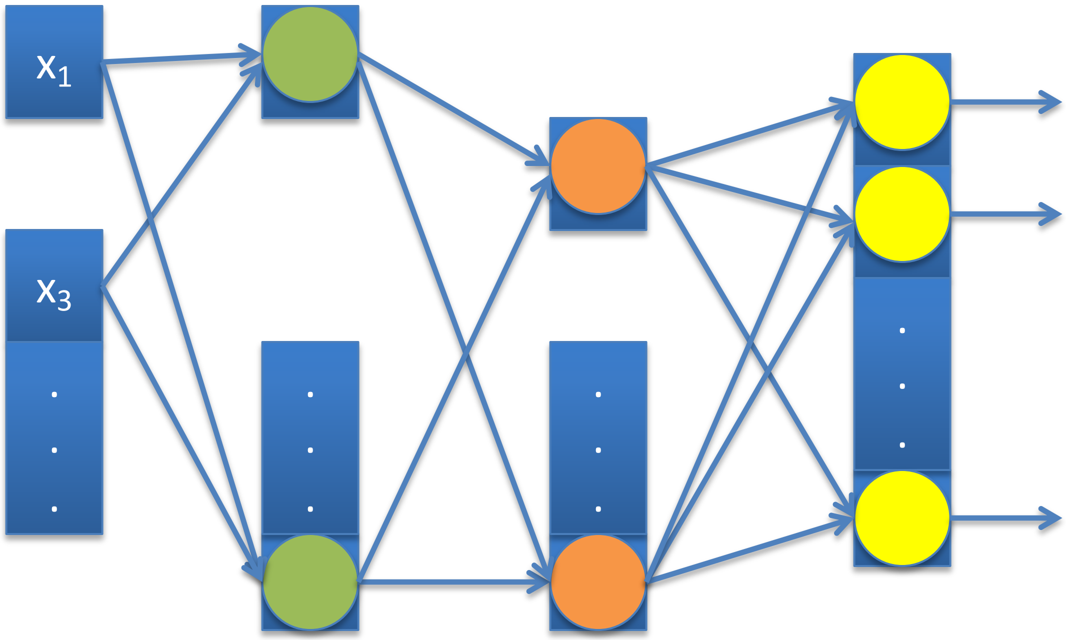
Dropout

Dropout: A Simple Way to Prevent Neural Networks from Overfitting, 2014



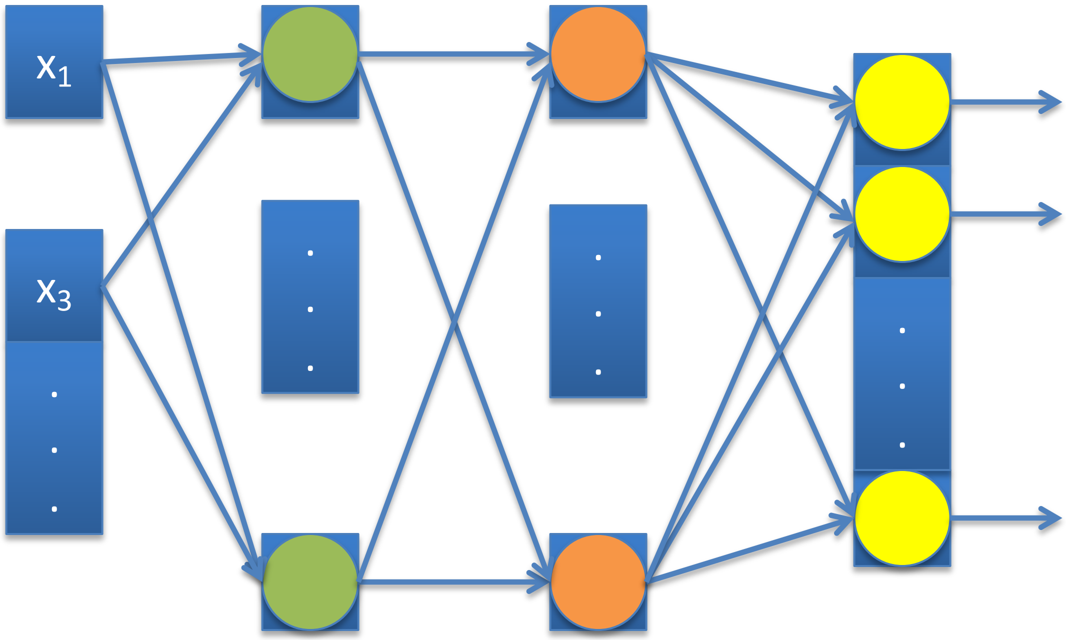
- In each iteration: Each neuron has p% to dropout during training

Dropout



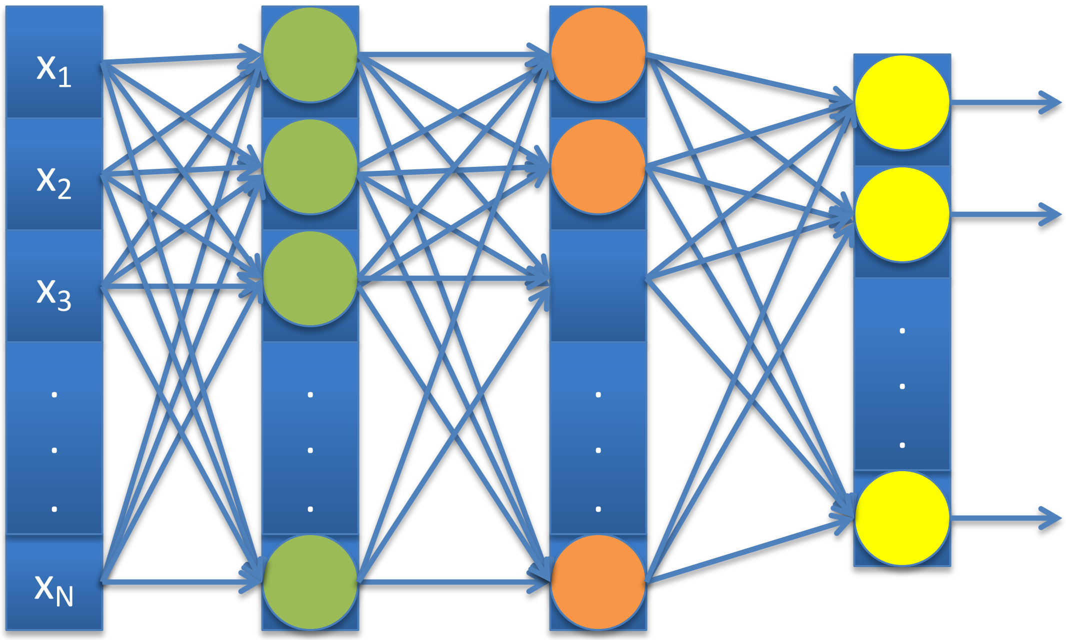
- In each iteration: Each neuron has p% to dropout during training, i.e. network structure is changed
- Only update the existing networks

Dropout



- In each iteration: Each neuron has p% to dropout during training, i.e. network structure is changed
- Only update the existing networks

Dropout



- No dropout during testing: Scale the weight with 1-p% if the dropout weight is p% during training

Thanks for listening!