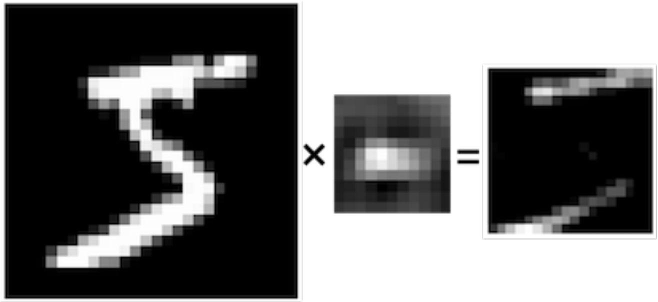
Introduction to Deep Learning

Recurrent Neural Networks I



Andreas Krug, M.Sc.
ankrug@uni-potsdam.de

19. November 2018

Orga

- max. 17 participants based on submissions
- Last day to withdraw from the course/
  I will admit those with regular submissions on PULS
  (one person in PULS would not be admitted)

- I'll provide optional programming exercises
- We'll focus on the small course projects

Last time on IDL & open questions

CNN papers

Group exercise

RNN architectures

Group Work Instructions

1. Match architectures with captions and circuit diagrams!
2. Draw missing circuit diagrams!
3. Annotate architectures with statements from next slide! (multiple matches possible)
4. *Find possible mistakes in the architecture figures!
5. *Map out relationships between architectures!

time for task: 35 min

- 7 architectures
- 5 min on average per architecture
- (some) enumerated snippets can be used multiple times

1. output at each time step
2. output after full input sequence has been read
3. input x serves as constant context or/and to initialize hidden state

4. recurrent connections between hidden units
5. recurrent connections from previous output
6. + optional output-to-hidden connections

7. encoder (reader): read input sequence, generate hidden state
8. decoder (writer): generate output sequence from hidden state
9. encoder-decoder
10. h(t) relevant summary of past (forward), g(t) relevant summary of future (backward)

11. trainable with “teacher forcing”
12. training can be parallelized

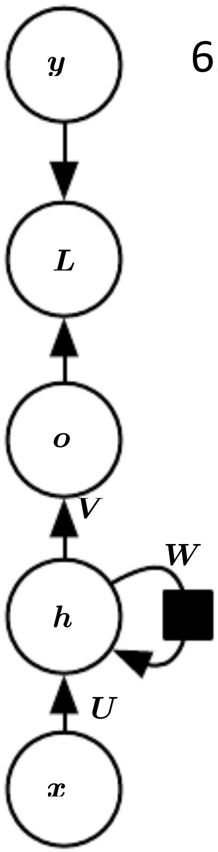
13. can compute any function computable by a Turing machine
14. can model arbitrary distribution over sequences of y given sequences of x
15. can model dependencies on both the past and the future
16. lacks important information from past unless o is very high-dimensional & rich

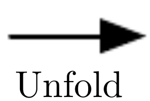
sequence to sequence (same length)

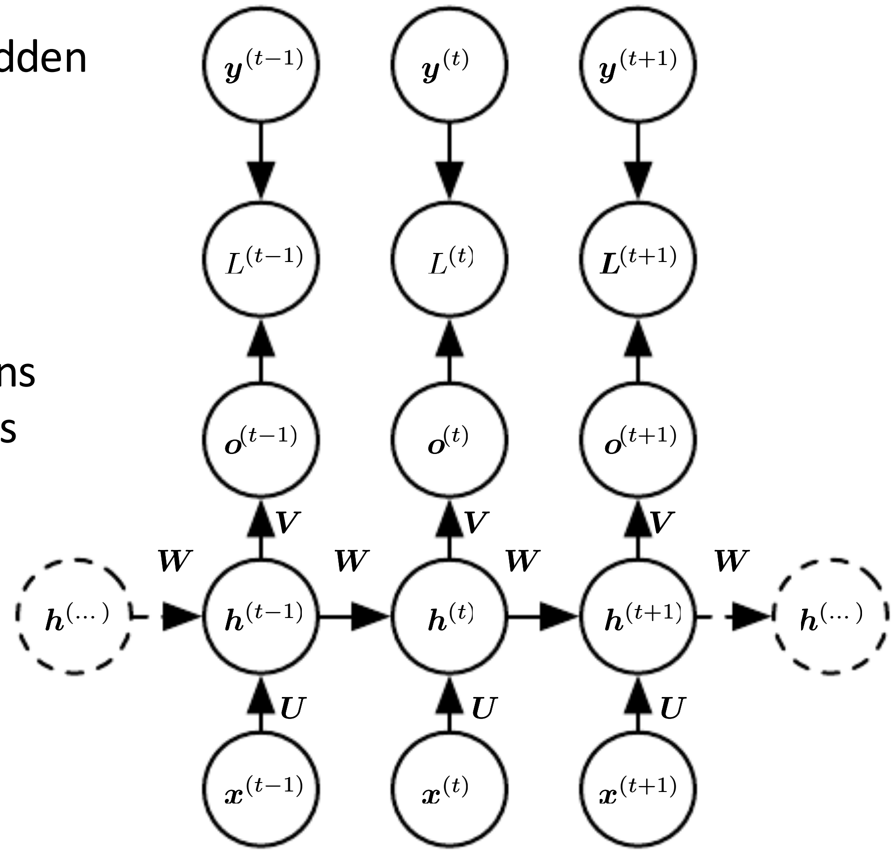
1. output at each time step

6. + optional output-to-hidden connections

4. recurrent connections between hidden units



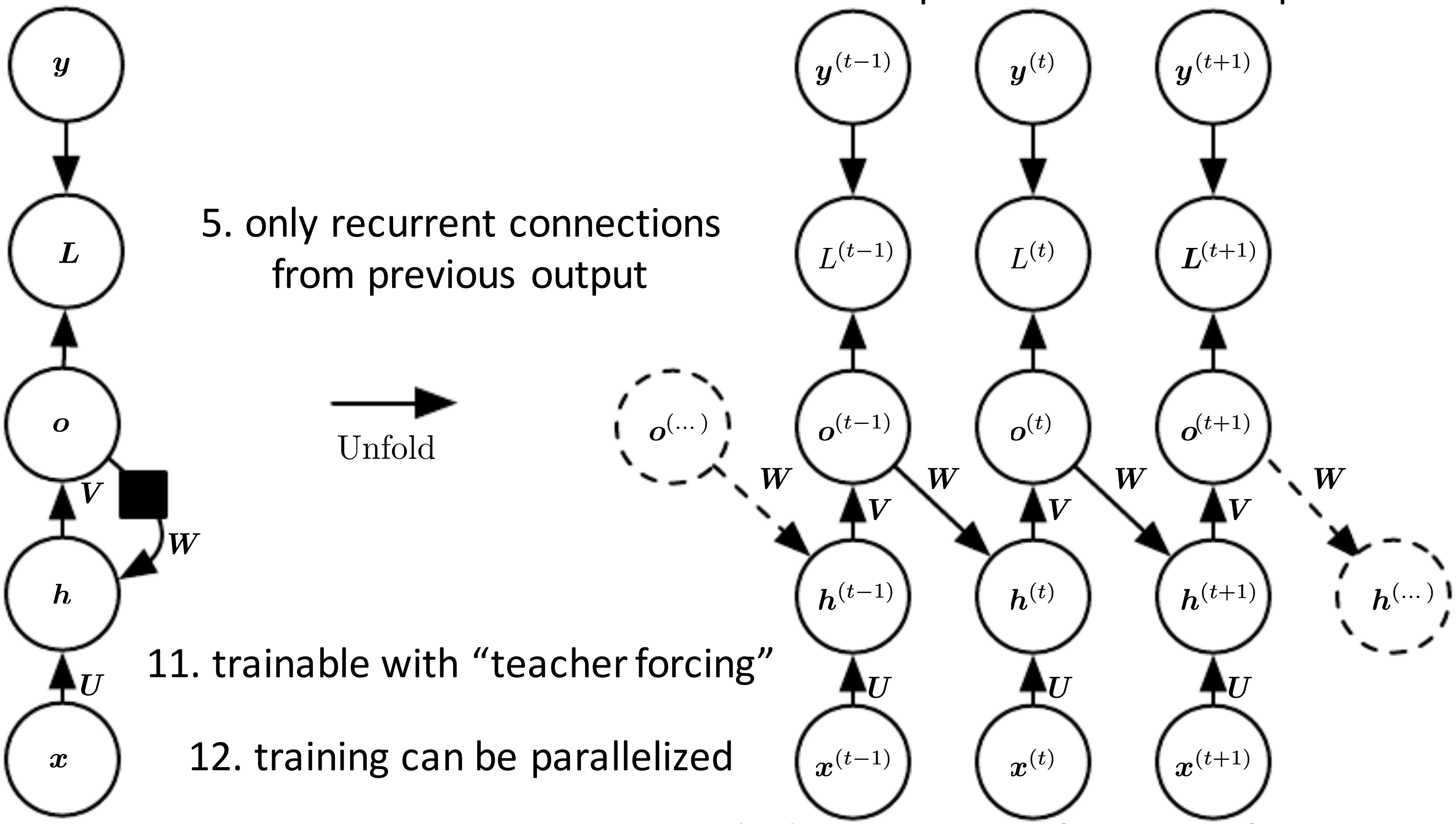




13. can compute any function computable by a Turing machine (universal function approximator)

sequence to sequence (same length)

1. output at each time step



5. only recurrent connections from previous output

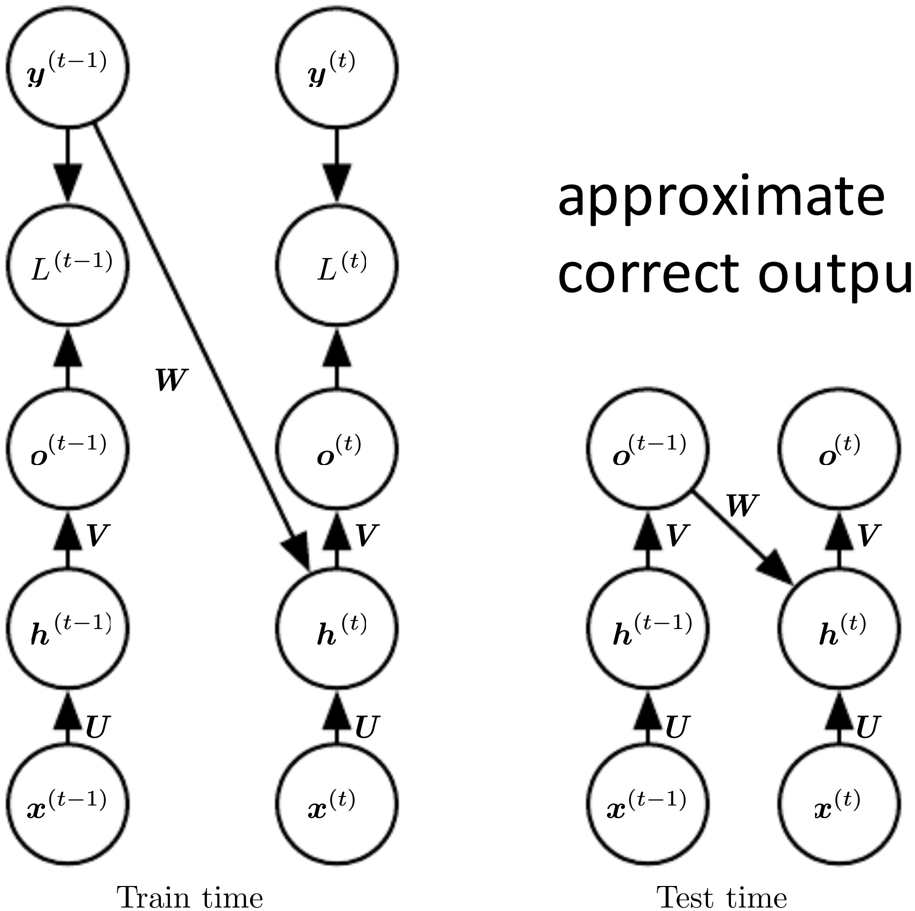
11. trainable with “teacher forcing”

12. training can be parallelized

16. lacks important information from past unless $\boldsymbol{o}$ is very high-dimensional & rich

Teacher Forcing

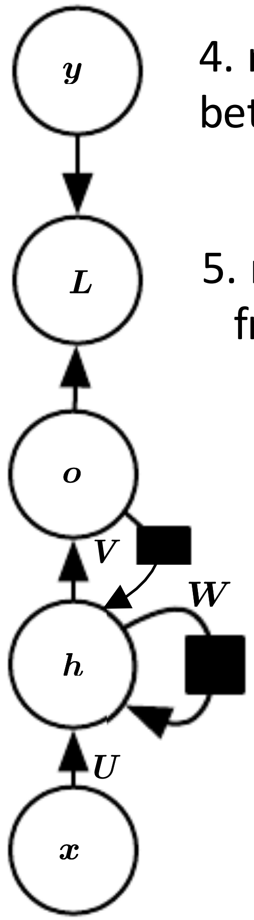
- use targets as prior outputs
- time steps decoupled
- training parallelizable



(may also be applied to RNNs with additional hidden-to-hidden connections)

sequence to sequence (same length)

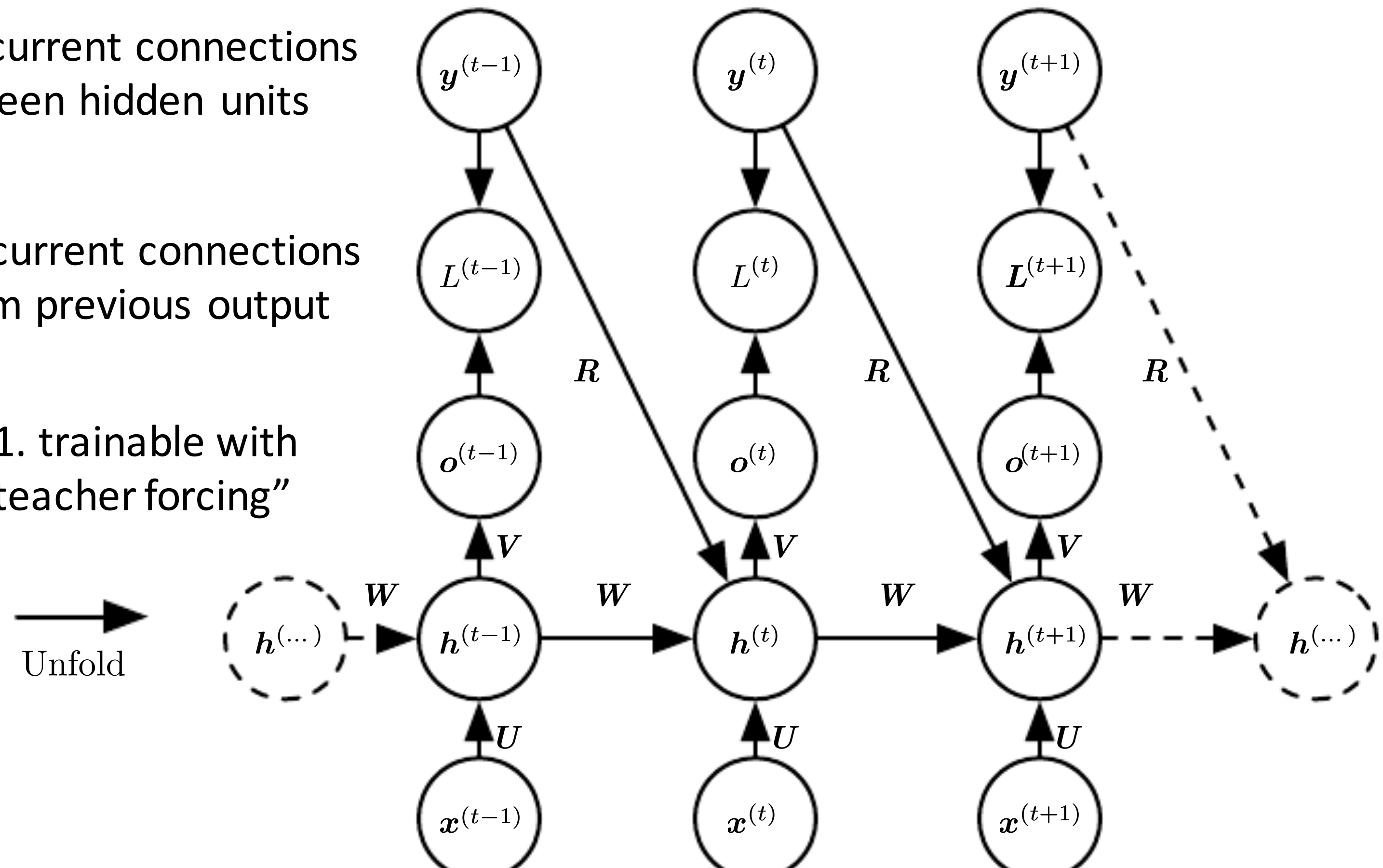
1. output at each time step



4. recurrent connections between hidden units

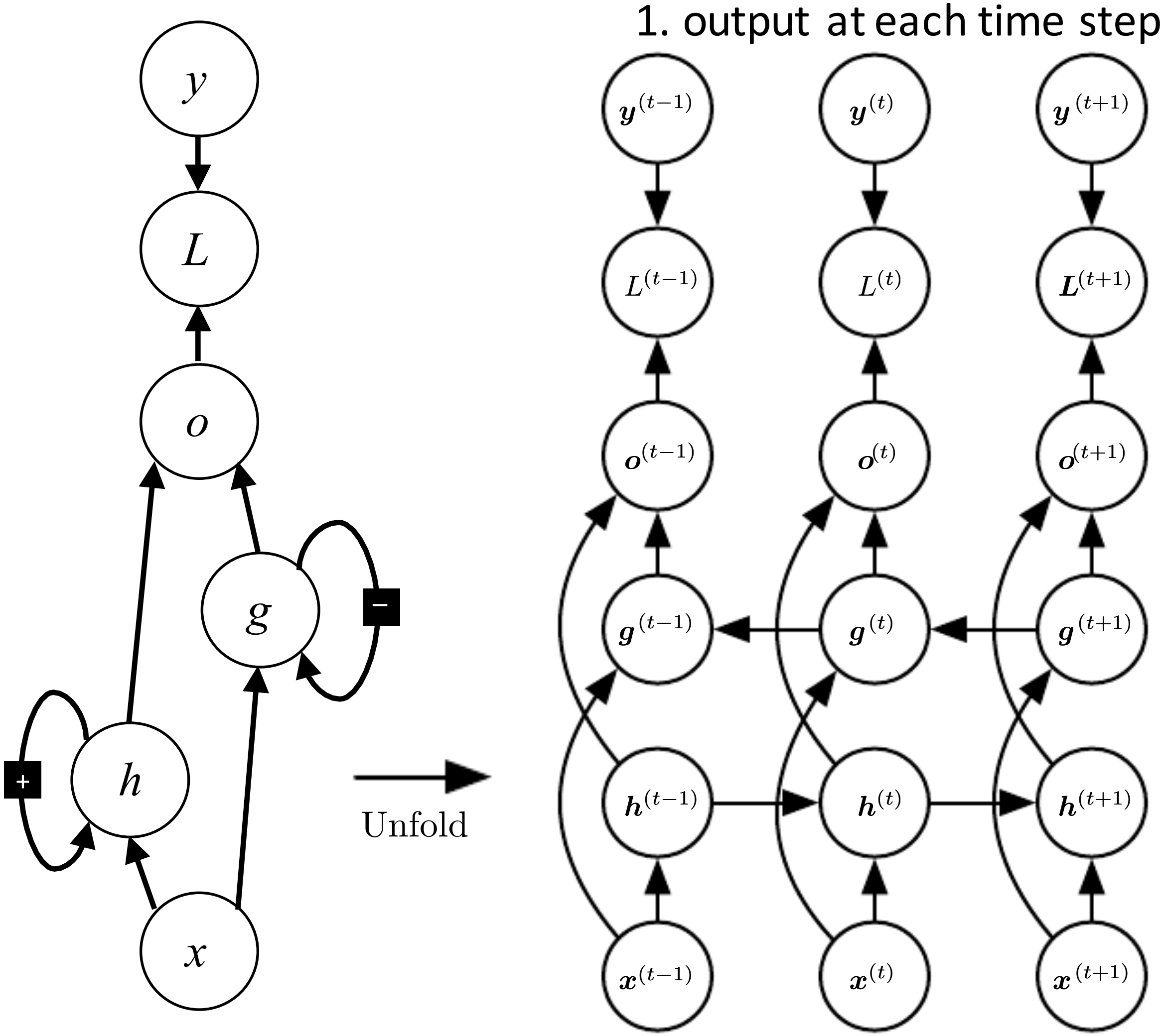
5. recurrent connections from previous output

11. trainable with “teacher forcing”



14. can model arbitrary distribution over sequences of y given sequences of x

bi-directional sequence to sequence (same length)



1. output at each time step

(extendable to 2D inputs)

4. recurrent connections between hidden units

6. + optional output-to-hidden connections

(in that case 11. trainable with “teacher forcing”)

10. g(t) relevant summary of future (backward)

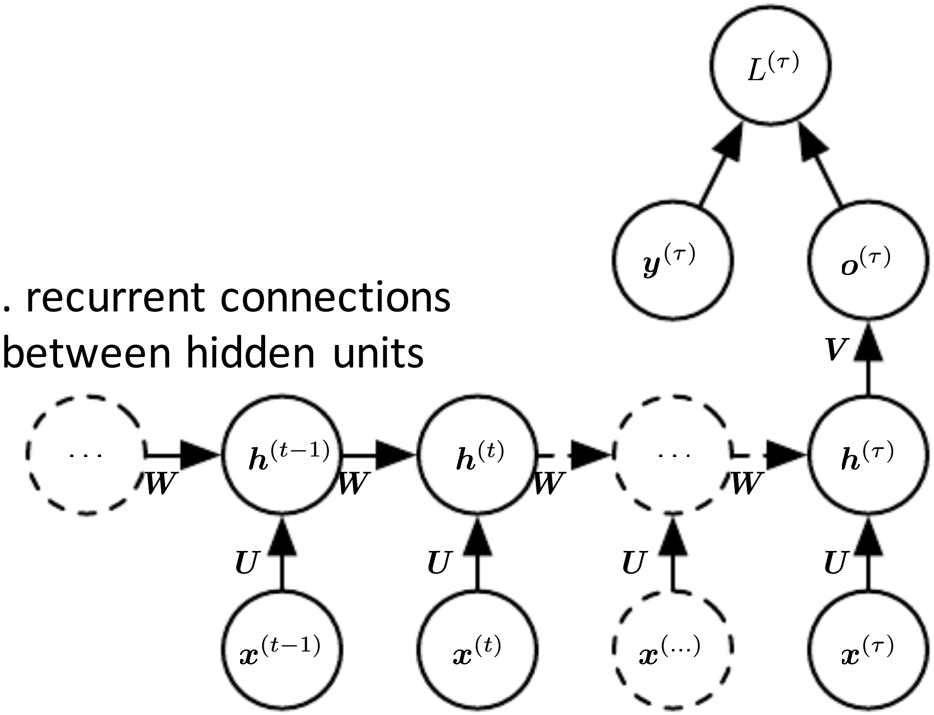
10. h(t) relevant summary of past (forward)

15. can model dependencies on both the past and the future

sequence to fixed-size vector

2. output after full input sequence has been read

4. recurrent connections between hidden units



7. encoder (reader): read input sequence, generate hidden state (= encoder part of encoder-decoder architecture)

fixed-size ("context") vector to sequence

strange indexing (stressing prediction of next output)

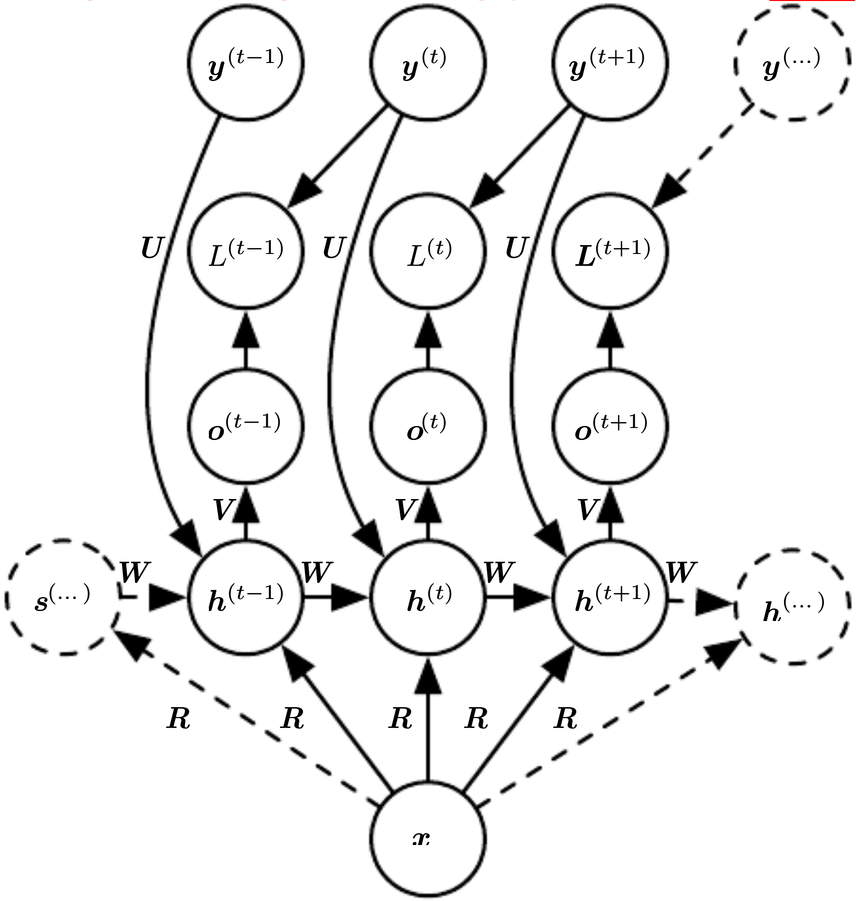
(needs to determine end of sequence)

5. recurrent connections from [previous] output (6. usually with output-to-hidden connections)

11. trainable with "teacher forcing"

4. recurrent connections between hidden units

3. input x serves as constant context or / and to initialize hidden state



8. decoder (writer): generate output sequence from hidden state (= decoder part of encoder-decoder architecture)

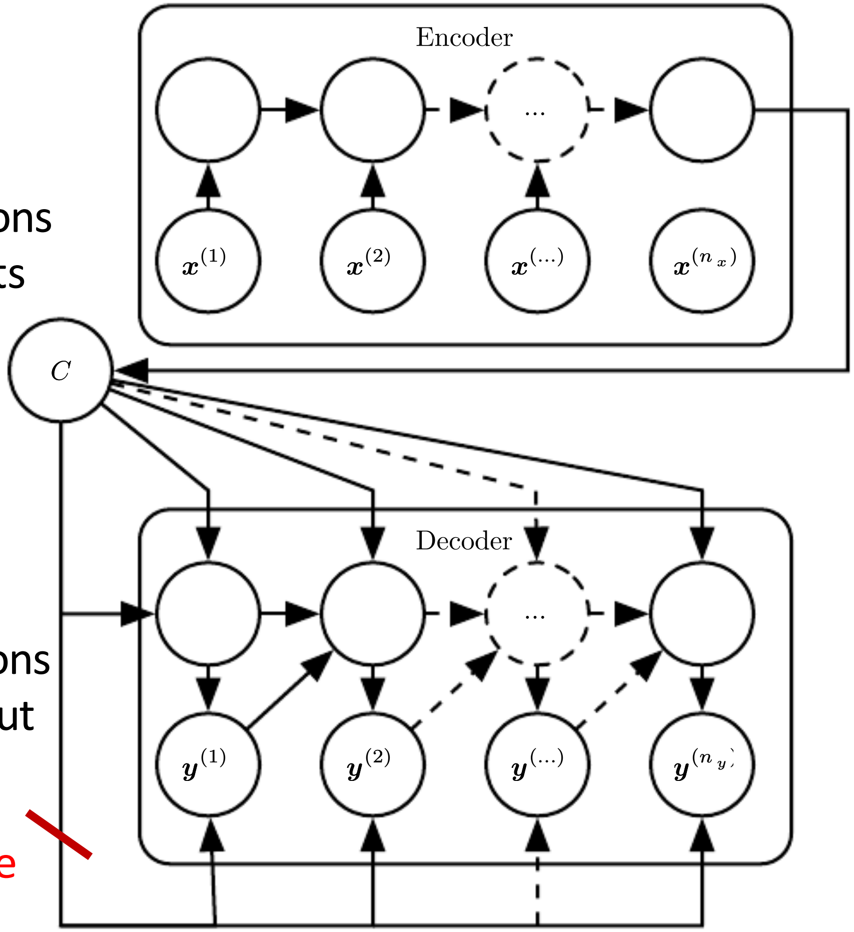
sequence to sequence (variable length)

simplified figure without state and transition labels

4. recurrent connections between hidden units

5. recurrent connections from [previous] output

does not make sense



loss not shown!

9. encoder-decoder

7. encoder (reader): read input sequence, generate hidden state

(bottleneck)

8. decoder (writer): generate output sequence from hidden state

Assignments until next week

- Responsible for recap: Edit & Ignatia
- Reading:
  Recurrent/Recursive Neural Networks part II
- Project:
  find partners and topic
  create channel on Mattermost
- Programming exercise (without submission):
  language modelling with RNN

Slides & assignments on: https://mlcogup.github.io/idl_ws18/