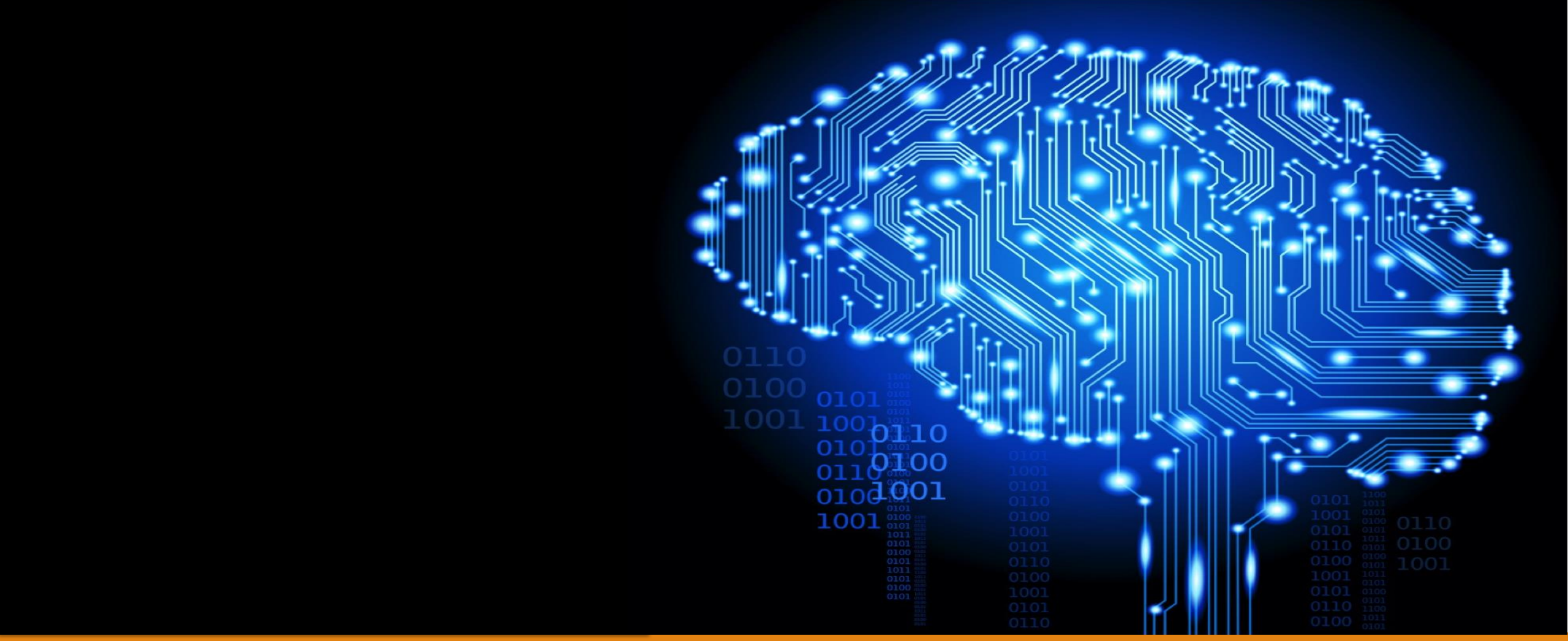


Lecture 6: Recurrent Neural Networks

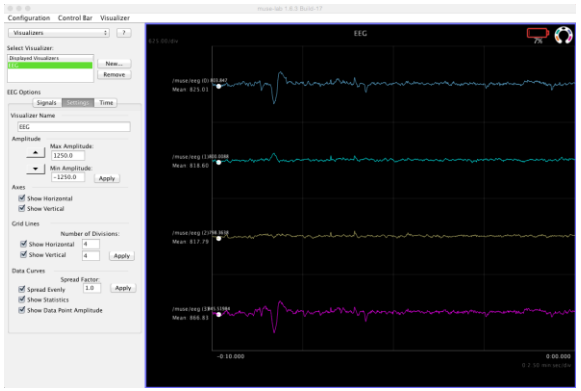
Efstratios Gavves

Lecture overview

- Inductive bias: what makes sequences special?
- Backpropagation through time
- LSTMs and variants
- Attention, Transformers
- Applications of sequence models

Example of sequential data and time series

- Videos
- Stock exchange
- Biological measurements
- Climate measurements
- Market analysis
- Speech/Music
- User behavior in websites
- Computer games



What makes sequences special?

- Sequence → Data are no more i.i.d.
  - Each data point on the previous ones and their distribution changes over time
  - *How to model temporal dependencies, especially in high-dim complex data?*
- Sequences might be of arbitrary length
  - *How to model dependencies at different temporal scales?*
  - *How to make a model that works for 10 sec and 10 min sequences?*
- Cycles in our computational graph
  - *How to learn with cycles?*

What makes sequences special?

- Unclear how to assign credit to past time steps
  - *Which of the infinite past time steps is responsible for this observed present?*
- Chaotic behavior
  - *How do we account for all possible effects of small changes in past sequence?*
- Temporal redundancy
  - *In temporally dense series there is lots of redundancy. Curse or blessing?*

Inductive bias for sequences (usually)

Challenges	Inductive bias
Non iid data	State models, chain rule of probabilities
Arbitrary lengths	Sharing weights
Credit assignment problem	Backpropagation through time
Chaotic behavior	LSTM, GRU
Temporal redundancy	Spiking neural nets, slow feature learning

A sequence of probabilities

- Sequences are by definition non iid

$p(\texttt{Donald Trump is an eloquent speaker!}) =$

$p(\texttt{Donald}) \cdot p(\texttt{Trump}|\texttt{Donald}) \cdot p(\texttt{is}|\texttt{Donald Trump}) \cdot \ldots \cdot p(\texttt{an eloquent speaker}!|\texttt{Donald Trump is})$

$p(\texttt{Barack Obama is an eloquent speaker!}) =$

$p(\texttt{Barack}) \cdot p(\texttt{Obama}|\texttt{Barack}) \cdot p(\texttt{is}|\texttt{Barack Obama}) \cdot \ldots \cdot p(\texttt{an eloquent speaker}!|\texttt{Barack Obama is})$

- Compute the likelihood by decomposing with chain rule of probabilities

$$p(x) = \prod_i p(x_i|x_1, \ldots, x_{i-1})$$

- Model each term $p(x_i|x_1, \ldots, x_{i-1})$ separately

Keeping a (memory) state

- To have the past influence present we must keep a summary of the past
  - A state, a memory, a representation of the past

$$p(\texttt{Donald Trump is an eloquent speaker!}) =$$
$$p(\texttt{Donald}) \cdot p(\texttt{Trump}|\texttt{Donald}) \cdot p(\texttt{is}|\texttt{Donald Trump}) \cdot \ldots \cdot p(\texttt{an eloquent speaker}!|\texttt{Donald Trump is})$$

- Memory can take all forms of shapes as long as it encodes the past
  - Otherwise it is hard to keep track of distributional shifts

Memory

- At timestep t project all previous information $1, \dots, t$ onto a state space s_t
  - Memory controlled by a neural network h with shared parameters θ
- Then, at timestep $t+1$ re-use the parameters θ_t and the previous $\boldsymbol{s}_t$

$$\boldsymbol{s}_{t+1} = h_{t+1}(\boldsymbol{x}_{t+1}, \boldsymbol{s}_t)$$

- Recursive operation, often with Markov Chain assumption
  - Any new state depends on the previous time step only

$$\boldsymbol{s}_t = h_t(\boldsymbol{x}_t, \boldsymbol{s}_{t-1})$$
$$\boldsymbol{s}_{t-1} = h_{t-1}(\boldsymbol{x}_{t-1}, \boldsymbol{s}_{t-2})$$
$$\dots$$

Sequences of arbitrary lengths

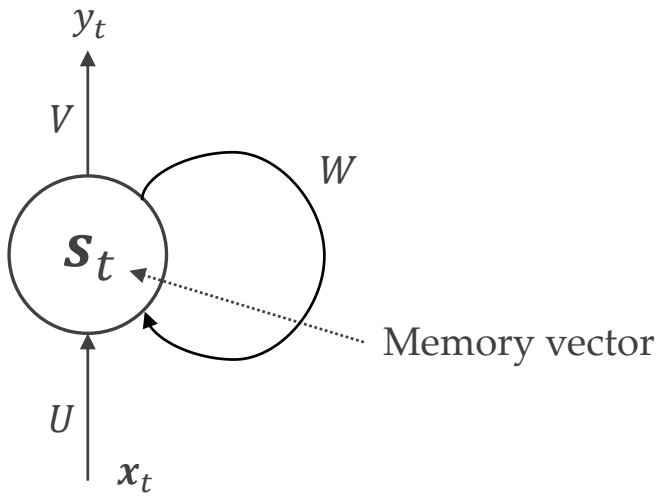
- We cannot, and ideally should not, have a constraint over sequence length
- The model should have no problem with
  - varying
  - unknown
  - or even infinite

  sequence lengths
- One logical solution: sharing parameters through time

$$\theta_t = \theta_{t-1} = \cdots = \theta_0$$

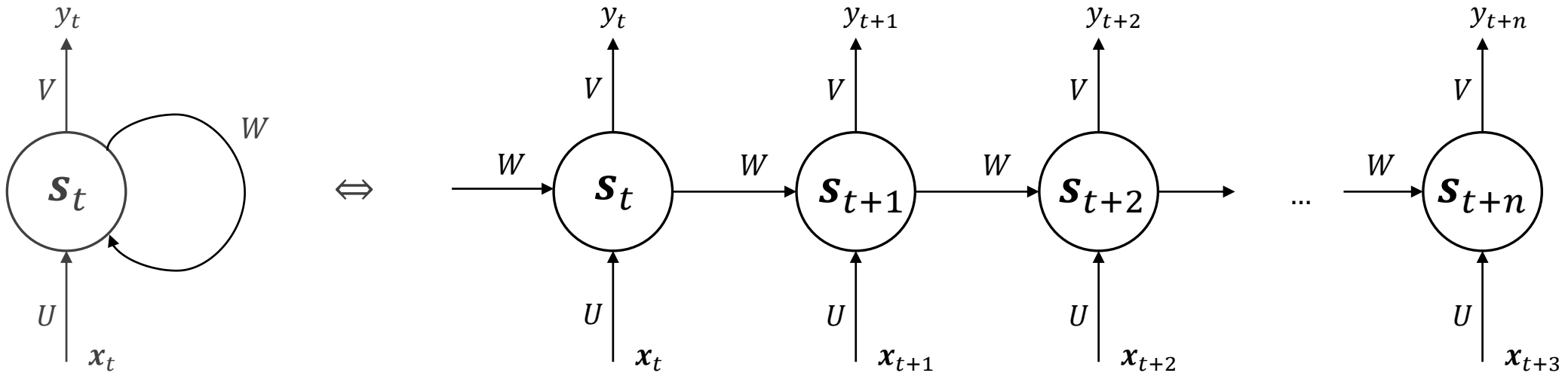
A graphical representation of memory

- In the simplest case with fully connected layers: $\theta = \{U, V, W\}$
- The matrix U transforms a vector from the input space to the state space
- The matrix V transforms a vector from the the state space to the output space
- The matrix W transforms the past state and input to the new state



Removing cycles by 'unfolding' the graph

- Write down each time step explicitly → no cycles anymore
- In theory, a problem with infinite time steps back
  - Can we do backpropagation with infinite time steps?
- In practice, we cut the sequence short after a while
  - After all, does the 'very distant' past have influence to the present?



Recurrent Neural Networks

- Putting things together

$$\boldsymbol{s}_t = \tanh(U \cdot \boldsymbol{x}_t + W \cdot \boldsymbol{s}_{t-1})$$
$$y_t = \text{softmax}(V \cdot \boldsymbol{s}_t)$$
$$\mathcal{L} = \sum_t \mathcal{L}_t(y_t, l_t)$$