Deep Learning and Temporal Data Processing

3 - Recurrent Neural Networks

Andrea Palazzi

June 21th, 2017

University of Modena and Reggio Emilia

Agenda

Introduction

Credits

Introduction

test [1]

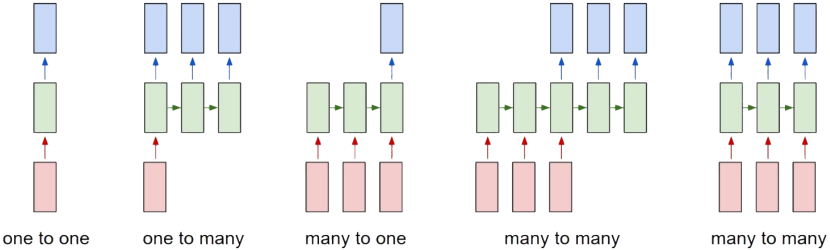
In **feedforward neural network** computation flows directly from input $\boldsymbol{x}$ through intermediate layers $\boldsymbol{h}$ to output $\boldsymbol{y}$.

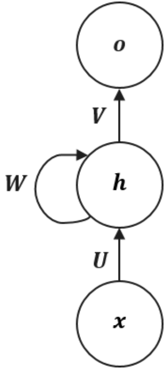
Conversely, some networks topology feature feedback connections, in other words model outputs are fed back into the model itself.

The term **recurrent neural networks** defines this family of models.

Recurrent neural networks (RNN) are **specialized for processing sequences**. Similarly, we saw that convolutional neural networks feature specialized architecture for processing images.

RNNs boast a **much wider API with respect to feedforward neural networks**. Indeed, these models can deal with *sequences* in the input, in the output or even both.





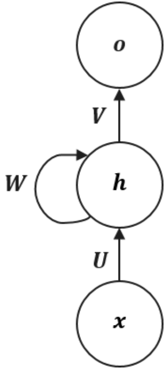
The vanilla RNN is provided with three sets of parameters:

- $\boldsymbol{U}$ maps inputs to the hidden state
- $\boldsymbol{W}$ parametrizes hidden state transition
- $\boldsymbol{V}$ maps hidden state to output

System dynamics is as simple as:

$$\begin{cases} \boldsymbol{h}^{(t)} = \phi(\boldsymbol{W}\,\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\,\boldsymbol{x}^{(t)}) \\ \boldsymbol{o}^{(t)} = \boldsymbol{V}\,\boldsymbol{h}^{(t)} \end{cases} \tag{1}$$

Intuition about Hidden State

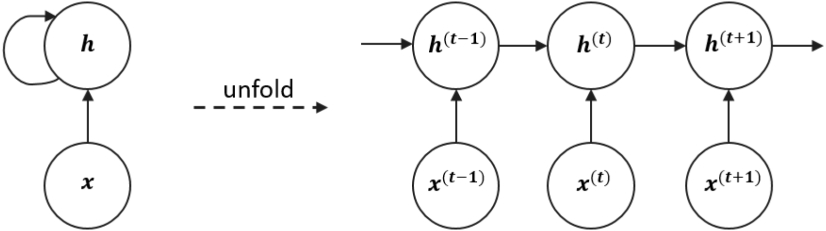


The hidden state $\boldsymbol{h}^{(t)}$ can be intuitively viewed as a *lossy* summary of the sequence of past inputs fed to the network, in which are stored the main task-relevant aspects of the past sequence of inputs up to time t.

Since the an input sequence of arbitrary length $(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, ..., \boldsymbol{x}^{(t)})$ is mapped into a fixed size vector $\boldsymbol{h}^{(t)}$, this summary is necessarily lossy.

A recurrent computational graph can be unfolded into a sequential computational graph with a repetitive structure.

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{t-1}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta})$$



how to unroll a recursive graph

Vanishing and exploding gradient problem.

Vanishing and exploding gradient problem.

Notes

No matter how's the network topology, during backpropagation the network is unfolded in a DAG, so there are no loops.

Credits

Credits

These slides heavily borrow from the following Stanford course:

- http://cs231n.stanford.edu/

if you want to deepen your knowledge of these concepts, I'd really suggest you to start from here!

Also, nice convolution animations are taken from here:

- https://github.com/vdumoulin/conv_arithmetic

[1] G. Cybenko.
Approximation by superpositions of a sigmoidal function.
Mathematics of Control, Signals, and Systems (MCSS), 2(4):303–314, 1989.