COMPUTER VISION
MEI/1

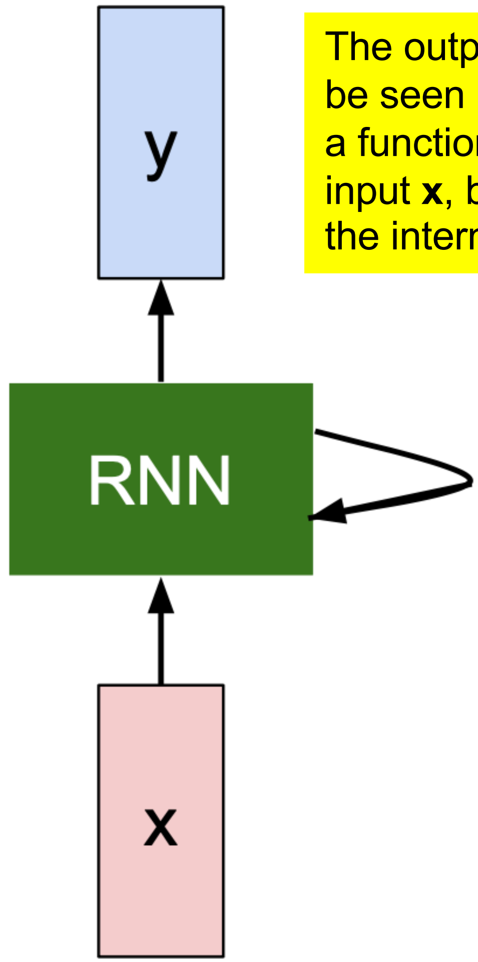
University of Beira Interior, Department of Informatics

Hugo Pedro Proença

hugomcp@di.ubi.pt, **2023/24**

Recurrent Neural Networks

- Recurrent Neural Networks (**RNNs**) are deep learning model typically used to process and convert a **sequential data input** into a sequential data **output**.
- **Sequential data**—such as words, sentences, or time-series— have interrelated sequential components, based on complex semantics and syntax rules.
- The key idea in RNNs is to use (apart the classical "weights") an **internal state** that is updated as a sequence is processed



The output **y** can be seen not only as a function of the input **x**, but also of the internal state **h**

Recurrent Neural Networks

- The forward step of RNNs is divided into two phases:

  - **Step 1:** Obtain the hidden state at time "t" ($\boldsymbol{h}_t$), given the input at time "t" ($\boldsymbol{x}_t$), and the previous state ($\boldsymbol{h}_{t-1}$).

$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

old state

input vector at some time step

some function with parameters W

  - **Step 2:** Then, obtain the output at time "t" ($\boldsymbol{y}_t$), using the recently updated state ($\boldsymbol{h}_t$).

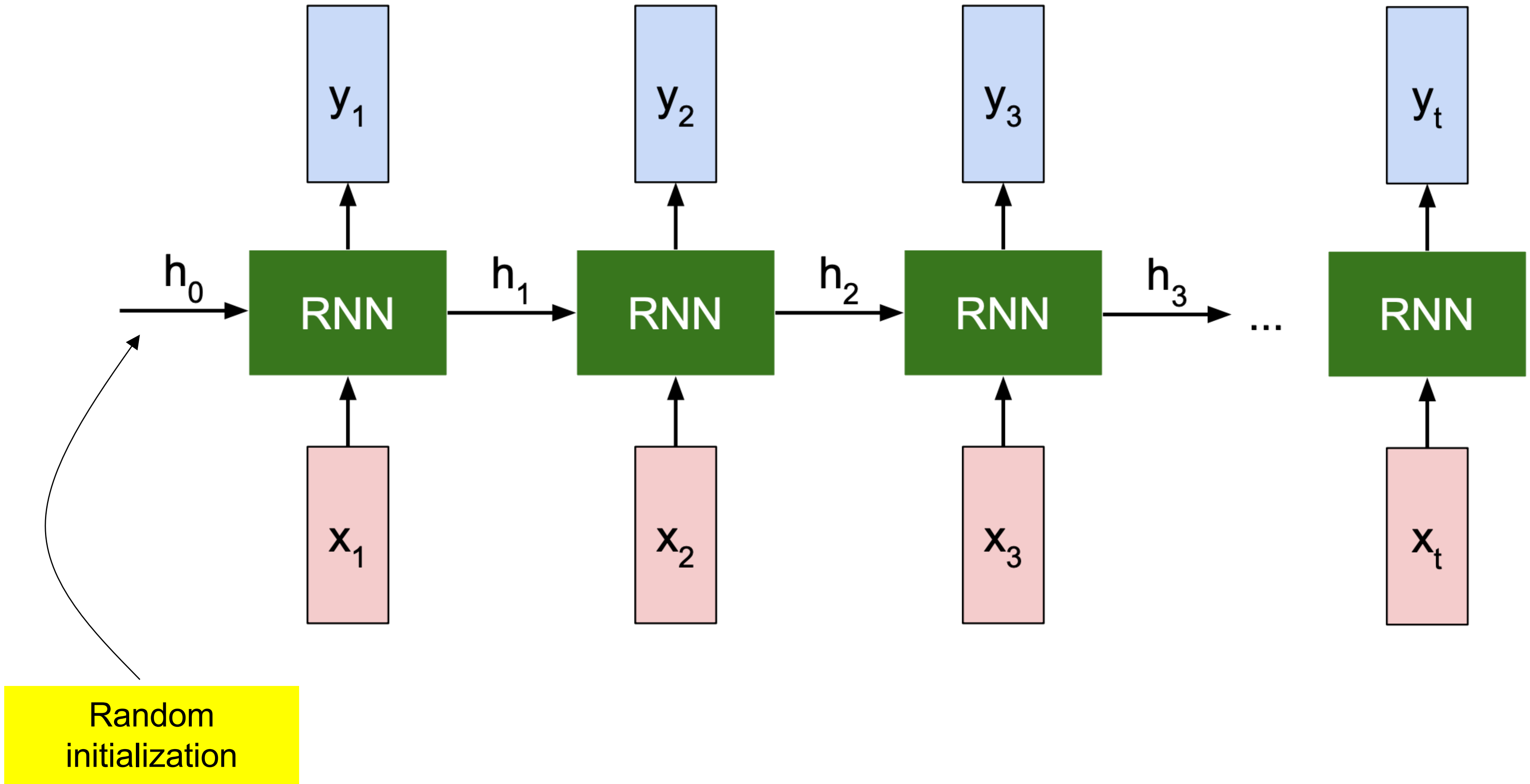
$$\boxed{y_t} = \boxed{f_{W_{hy}}}(\boxed{h_t})$$

output

new state

another function with parameters W_o

Recurrent Neural Networks



Recurrent Neural Networks

- **Step 1.** To obtain the hidden state at time "t" ($\boldsymbol{h}_t$) , we process a set of inputs ($\boldsymbol{x}_i$) , using the same function $\boldsymbol{f}_W$ at every step.
- In practice, this is due to the fact that backpropagation (weights update) is only done after a batch of steps.

$$h_t = f_W(h_{t-1}, x_t)$$

- The pioneer architecture (Vanilla RNN) assumes that the state ($\boldsymbol{h}_t$) is a single hidden vector in the network.
  - "s" is the dimension of the input/output space, and "d" is a hyper-parameter of the RNN.

[d x 1] vector

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

[s x 1] vector

[d x d] Matrix

[d x 1] vector

[d x s] Matrix

Recurrent Neural Networks

- **Step 2.** Once $\boldsymbol{h}_t$ is found, the output at time “t” $(\boldsymbol{y}_t)$, can also be obtained

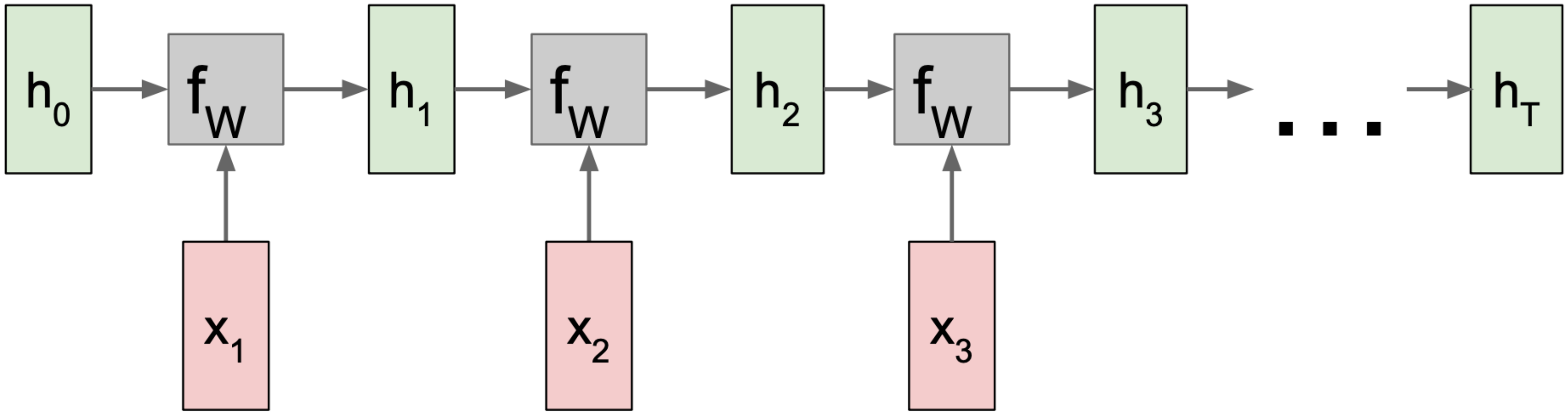
$$y_t = W_{hy} h_t$$

[s x 1] vector

[s x d] Matrix

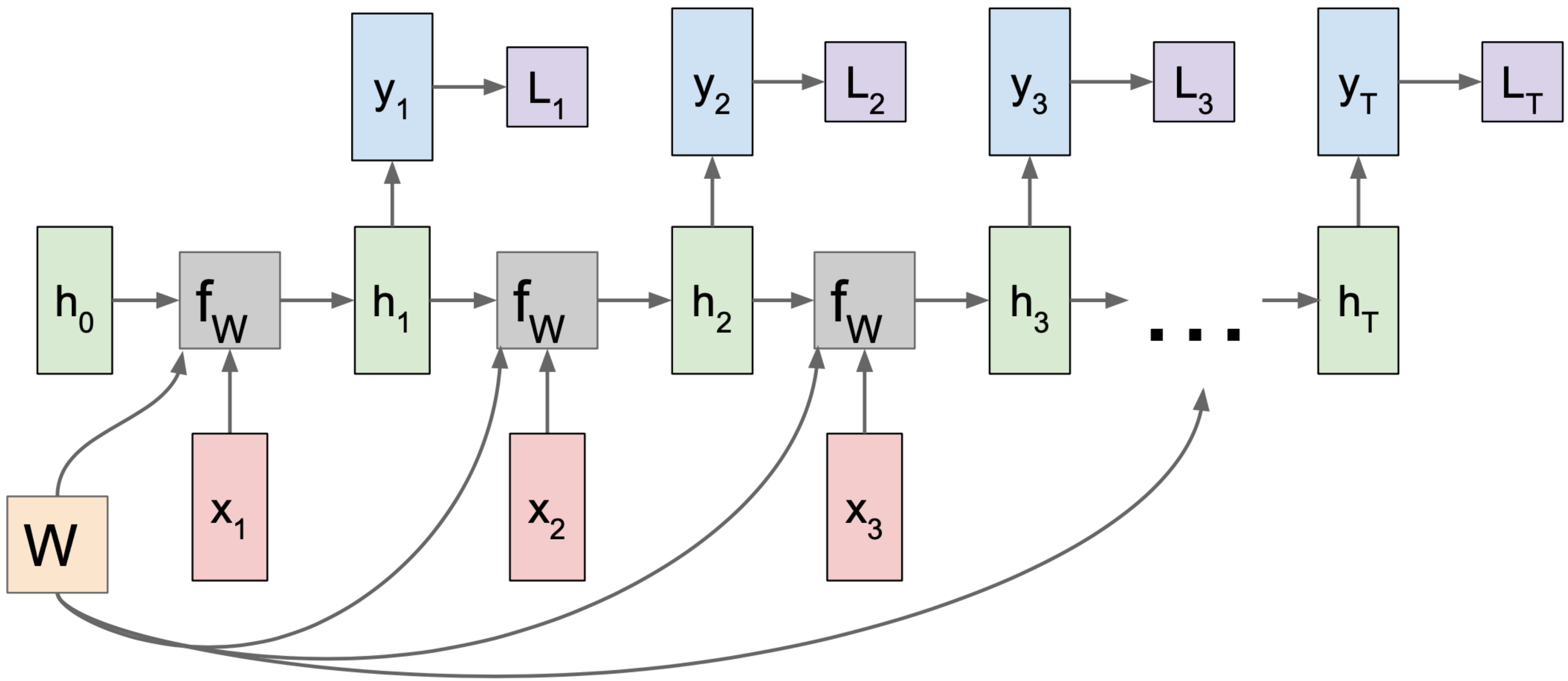
[d x 1] vector

- Hence, the first step of the corresponding computational graph is given by:



Recurrent Neural Networks

- Only at the second step, the outputs $(\boldsymbol{y}_t)$ are obtained and the partial losses found.
- Such partial loss values are then used to obtain the final loss $\mathcal{L}$ that will be used in backpropagation.



Recurrent Neural Networks: Example

- Text Generation. Consider a single training sequence (“hello”).
- The vocabulary is a set of four symbols: {“h”, “e”, “l”, “o”}
- We start by obtaining a latent representation of each element in the training set. The simplest one is the hot-one encoding.
- $h \rightarrow [1, 0, 0, 0]^T$; $e \rightarrow [0, 1, 0, 0]^T$; $l \rightarrow [0, 0, 1, 0]^T$; $\mathrm{o} \rightarrow [0, 0, 0, 1]^T$
- More sophisticated content generation techniques (e.g., Chat GPT) obtain richer representations, which elements lie in topological spaces (i.e., neighbor representations are related or are alike).
- It is reported that these representations play a very important role in the final effectiveness of the model.
- In this example, we are working at the character level. However, “word” or even “small sentence” levels can also be considered.
- “cat” $\rightarrow [1, 0, \dots, 0, 0]^T$; $dog \rightarrow [0, 1, \dots, 0, 0]^T$;

Recurrent Neural Networks: Example

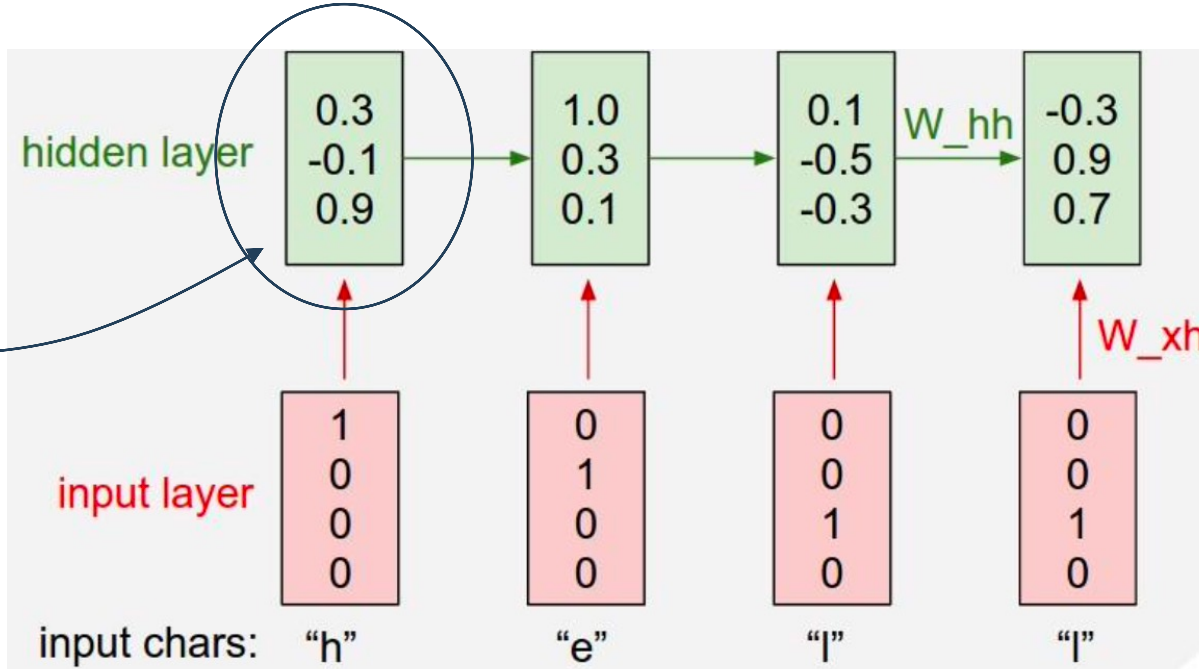
- **Step 1**. Obtain the hidden state representations ($\boldsymbol{h}_t$) for the training sequence (“hell”).

Why isn’t the complete set considered?

- Suppose that ($\boldsymbol{W}_{hh}$) and ($\boldsymbol{W}_{xh}$) were initialized randomly.

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

($\boldsymbol{h}_t$)

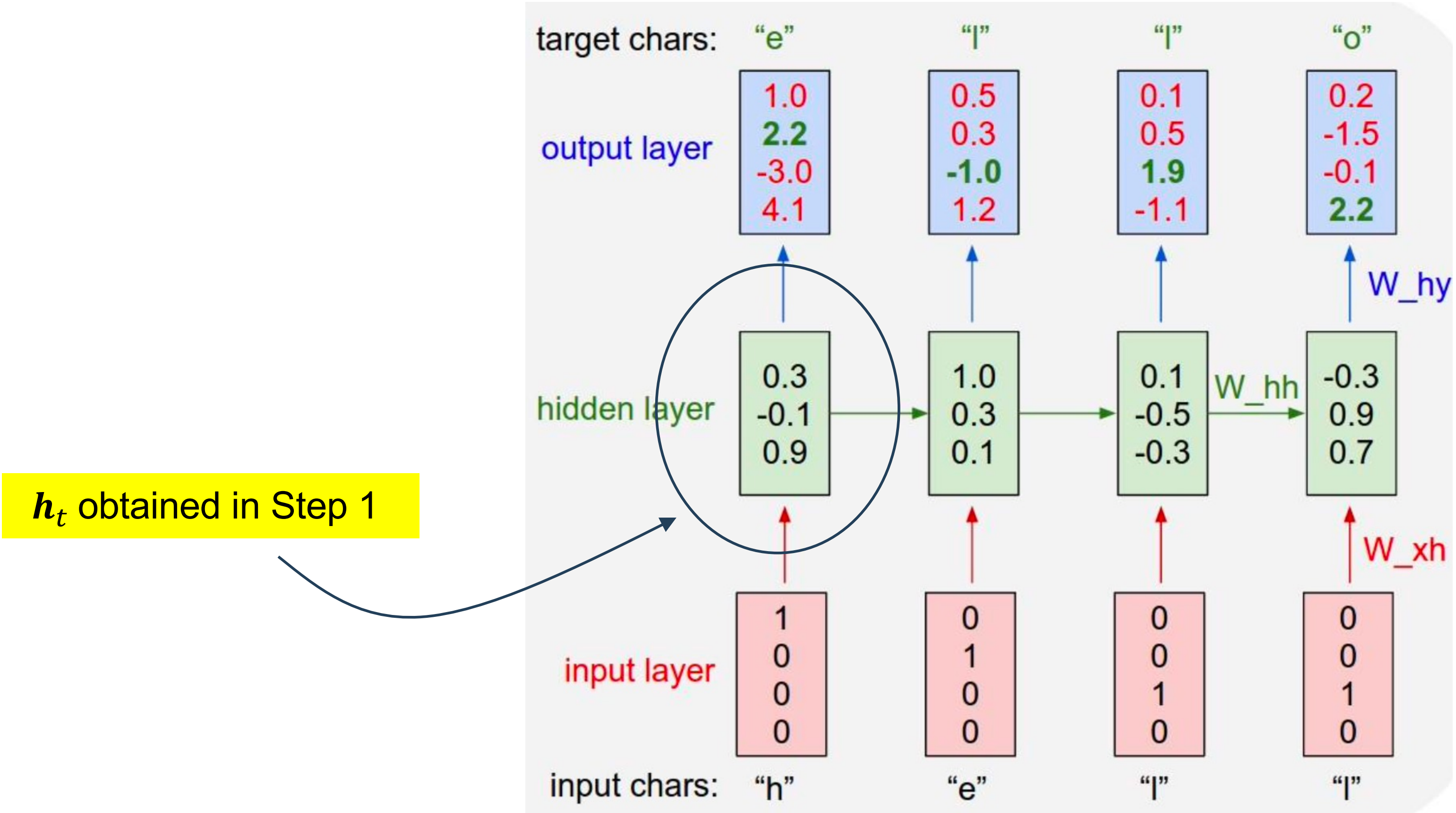


Recurrent Neural Networks: Example

- **Step 2**. Next, we can obtain the predicted elements at each time.

$$y_t = W_{hy} h_t$$

- Again, suppose that ($\boldsymbol{W}_{hy}$) was initialized randomly.

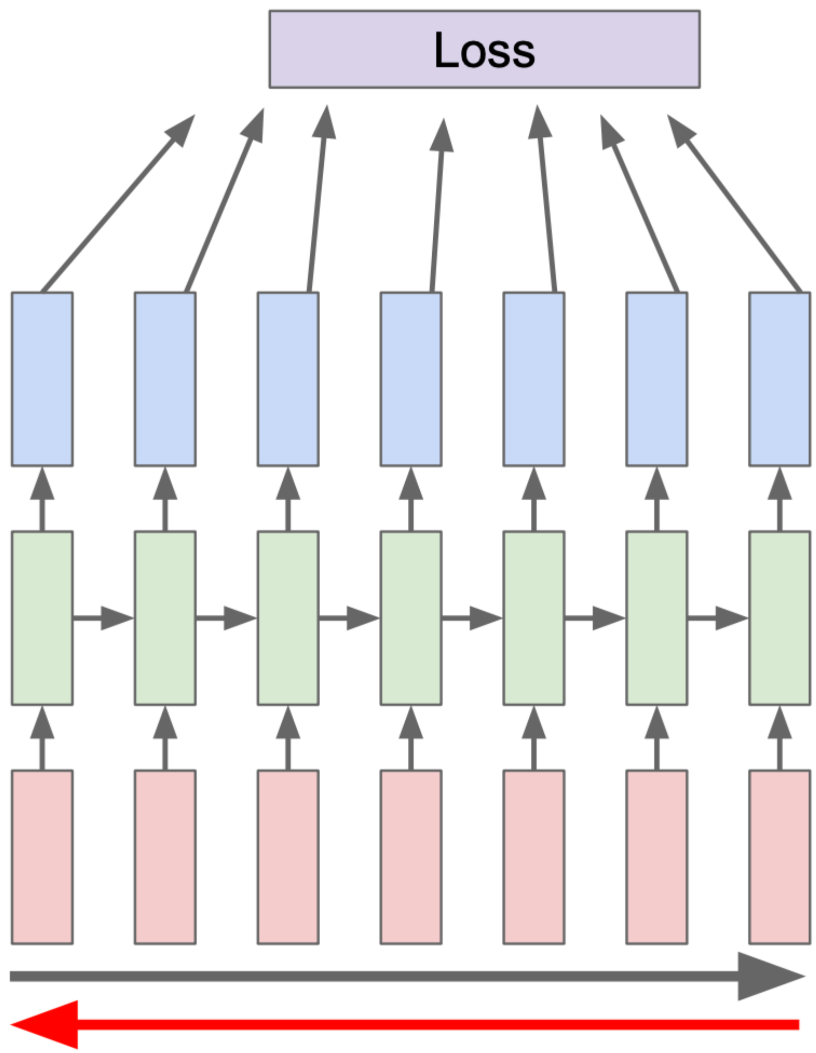


Recurrent Neural Networks: Example

- During training, we forward during the entire sequence to obtain the loss, and then backpropagate to obtain the gradientes and adjust weights.

- However, in practice, we run forward/backward through "**chunks**" instead of the whole sequence.

- This is the equivalent to the notion "**batch**" in classical CNNs architectures

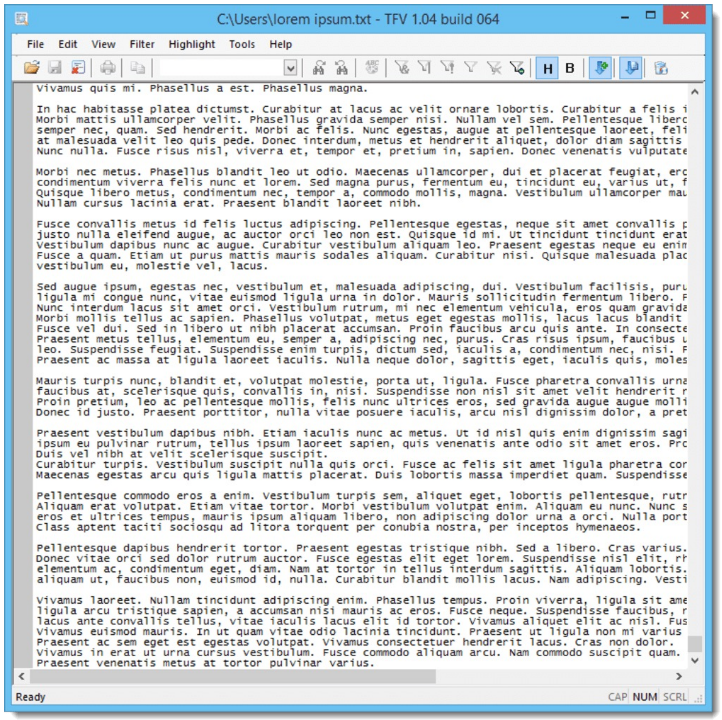


Recurrent Neural Networks: Example

- A minimal example (in 112 lines of Python) is available at the web page of this course. It contains a “Vanila” RNN learning process, depending exclusively of “numpy” library.

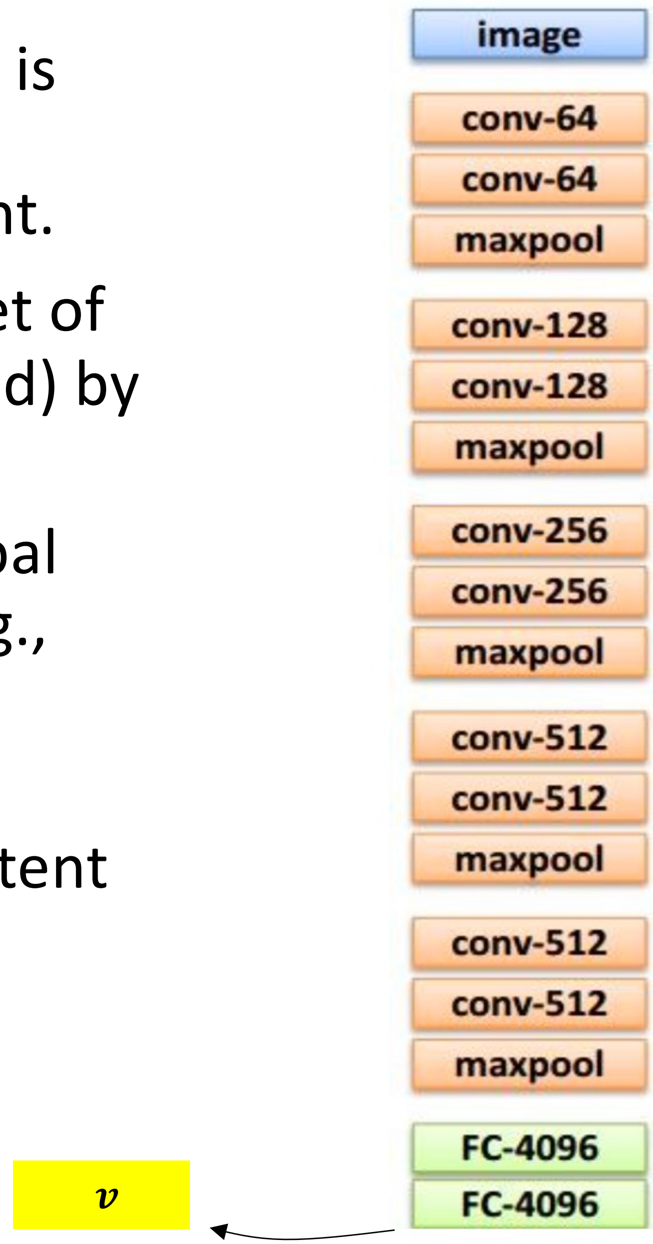
Credits: Andrej Karpathy

- Based in a simple plain text file (input.txt”) it learns to generate text.



Recurrent Neural Networks: Applications

- One interesting application of RNNs is "**Image Captioning**", that regards to obtain descriptions for visual content.
- The learning set is composed of a set of images previously labeled (captioned) by humans.
- A classical CNN architecture for global image classification can be used (e.g., VGG or ResNet), removing the final classification layer.
- We use the highest-level possible latent representation



Recurrent Neural Networks: Applications

- The latent representation $\boldsymbol{v}$ is also considered by the RNN, fusing text $\boldsymbol{x}$ to visual information $\boldsymbol{v}$
- A new weights matrix $\boldsymbol{W}_{ih}$ is also required

$$h = \tanh(W_{xh} * x + W_{hh} * h + W_{ih} * v)$$

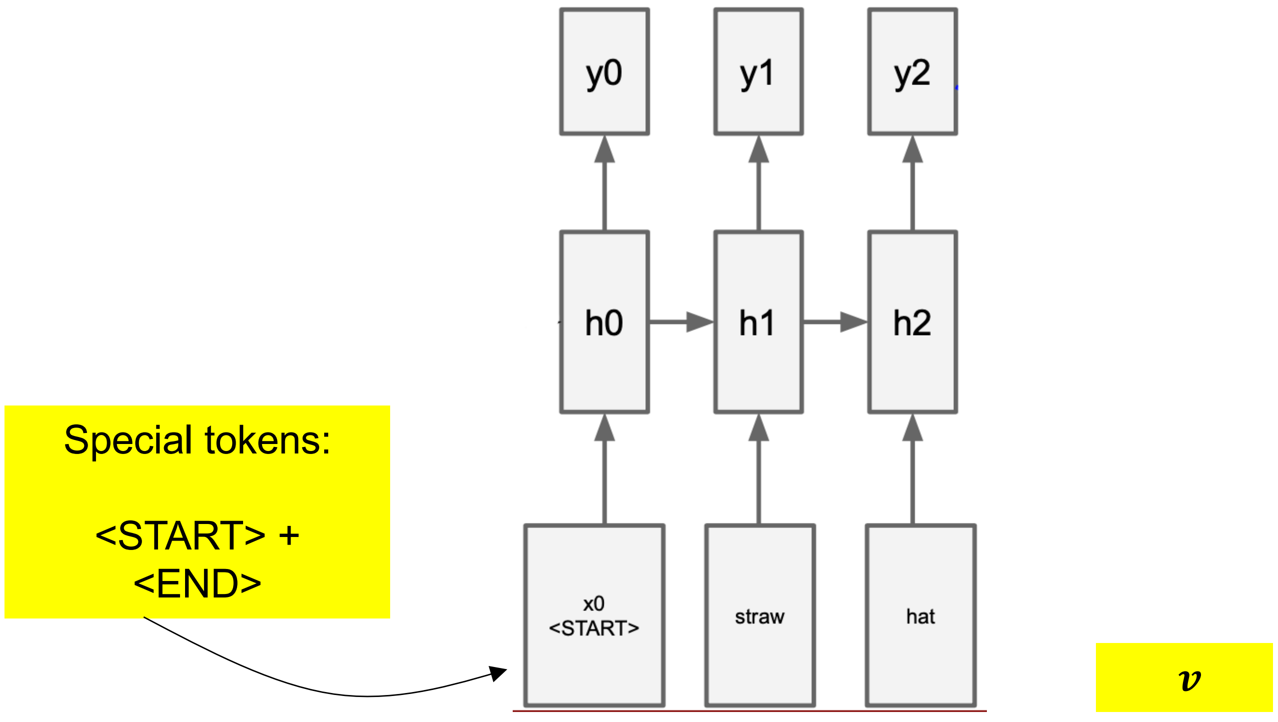


Image Captioning: Results

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Credits: Fei-Fei Li, Yunzhu Li, Ruohan Gao