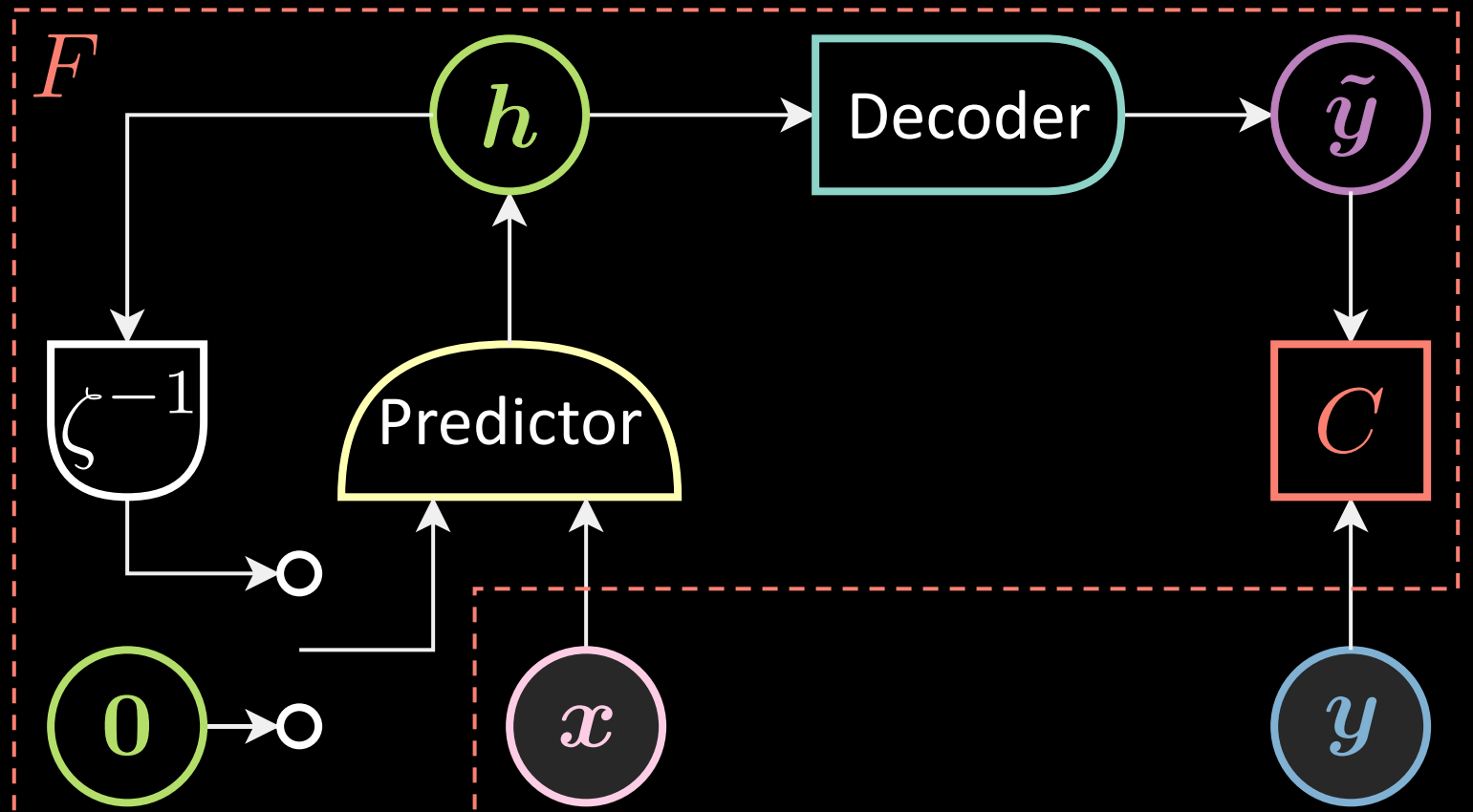
Recurrent Neural Nets

Handling sequential data

RNN diagram

$$F(\boldsymbol{x}, \boldsymbol{y}) =$$

$$\sum_{t=1}^{T} C(\boldsymbol{y}[t], \tilde{\boldsymbol{y}}[t])$$

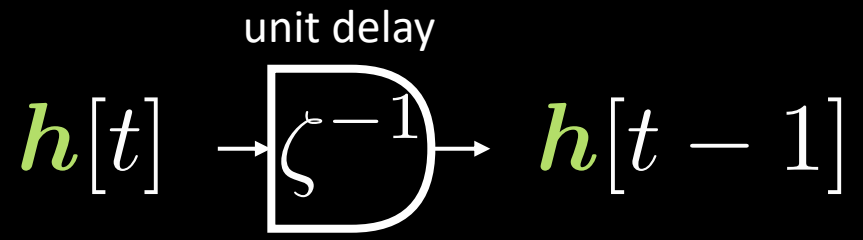


RNN equations

$$\boldsymbol{h}[0] \doteq \mathbf{0}$$

$$\boldsymbol{h}[t] = \mathrm{Pred}(\boldsymbol{h}[t-1], \boldsymbol{x}[t])$$

$$\tilde{\boldsymbol{y}}[t] = \mathrm{Dec}(\boldsymbol{h}[t])$$



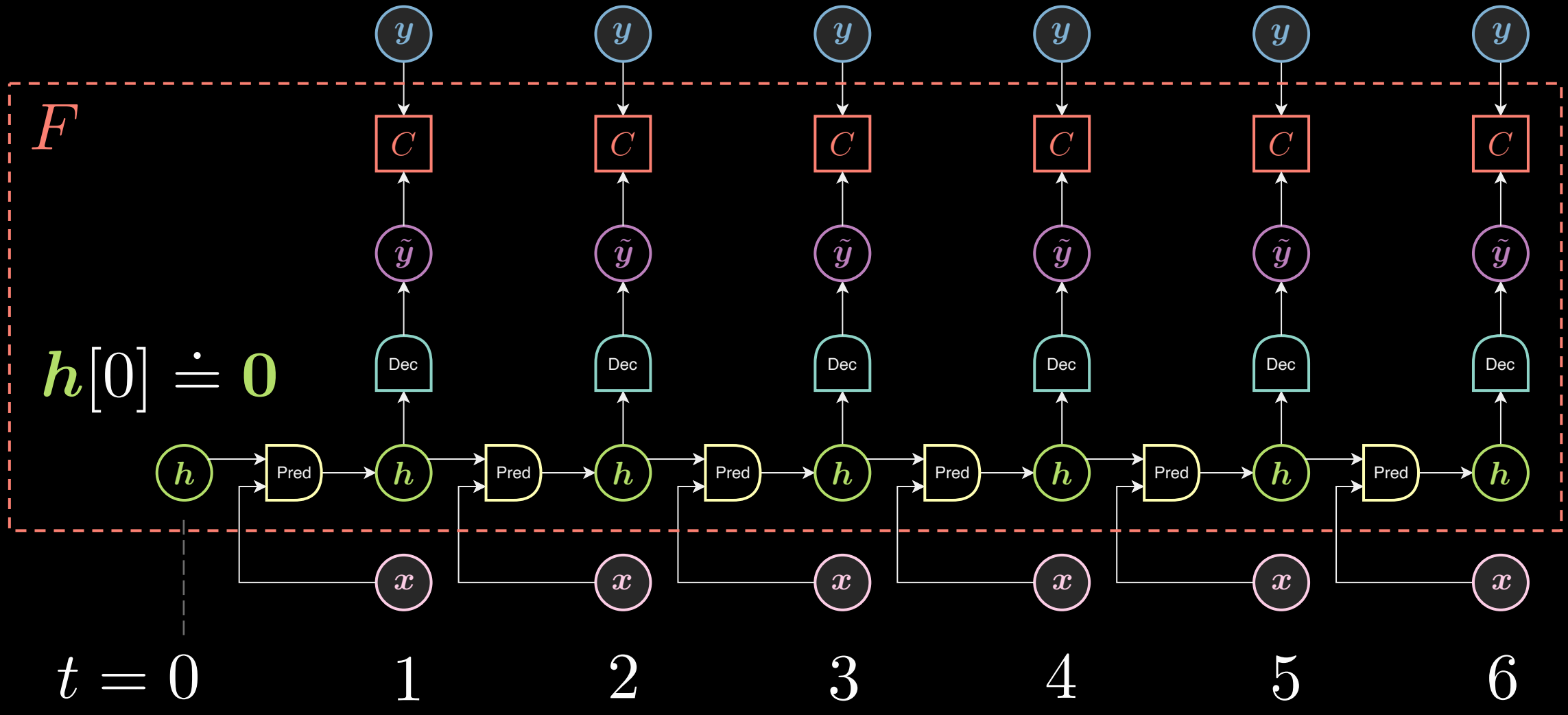
RNN training

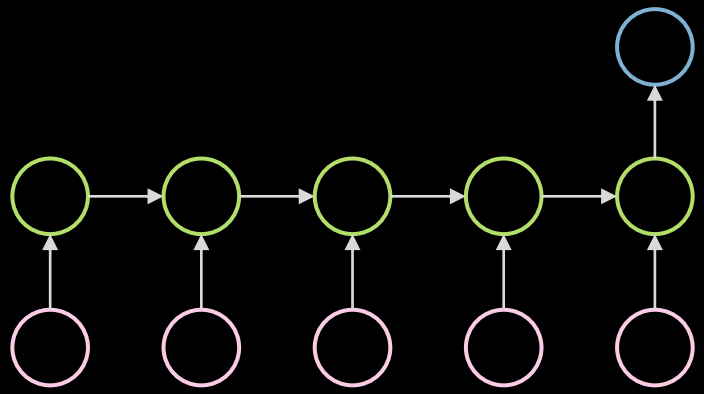
- backprop through time
- SGD wrt model's params to match ***x*** and ***y***

RNN training

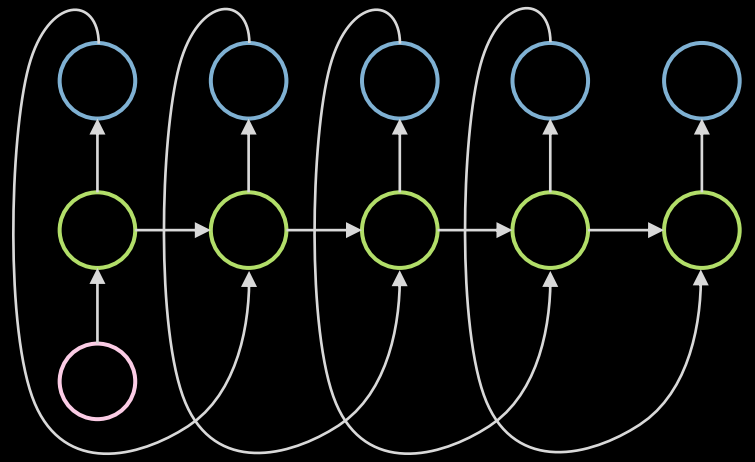
$$\boldsymbol{h}[t] = \mathrm{Pred}(\boldsymbol{h}[t-1], \boldsymbol{x}[t])$$

$$\tilde{\boldsymbol{y}}[t] = \mathrm{Dec}(\boldsymbol{h}[t])$$

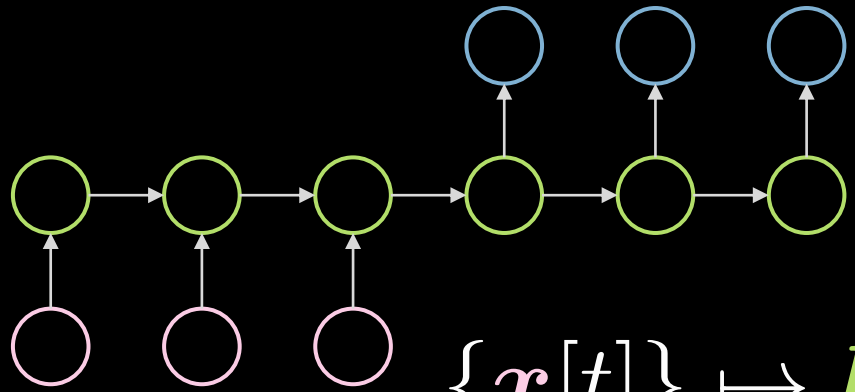




$\{\boldsymbol{x}[t]\} \mapsto \boldsymbol{y}[T]$ seq $\mapsto$ vec

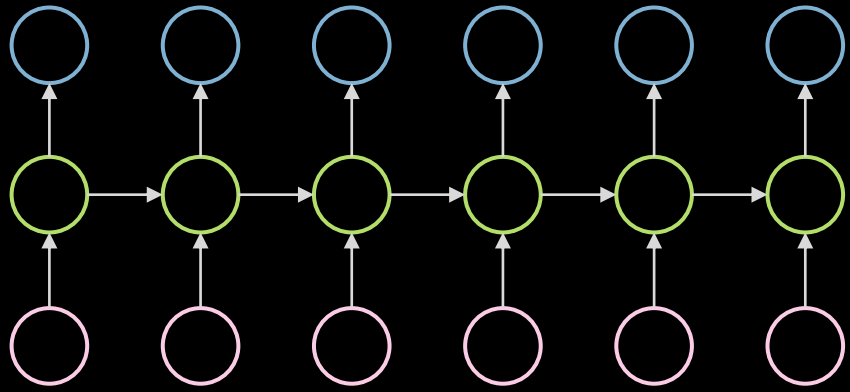


$\boldsymbol{x}[1] \mapsto \{\boldsymbol{y}[t]\}$ vec $\mapsto$ seq



$\{\boldsymbol{x}[t]\} \mapsto \boldsymbol{h} \mapsto \{\boldsymbol{y}[t]\}$

seq $\mapsto$ vec $\mapsto$ seq



$\{\boldsymbol{x}[t]\} \mapsto \{\boldsymbol{y}[t]\}$ seq $\mapsto$ seq

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

A skateboarder does a trick on a ramp.

A dog is jumping to catch a frisbee.

A group of young people playing a game of frisbee.

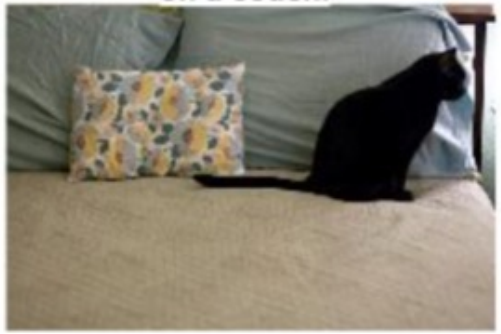
Two hockey players are fighting over the puck.

A little girl in a pink hat is blowing bubbles.

A refrigerator filled with lots of food and drinks.

A herd of elephants walking across a dry grass field.

A close up of a cat laying on a couch.



A red motorcycle parked on the side of the road.

A yellow school bus parked in a parking lot.

Describes without errors	Describes with minor errors	Somewhat related to the image	Unrelated to the image

Vinyals *et al.* (2016) Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge

Learning to execute

- Input:

```
j=8584
for x in range(8):
    j+=920
b=(1500+j)
print((b+7567))
```

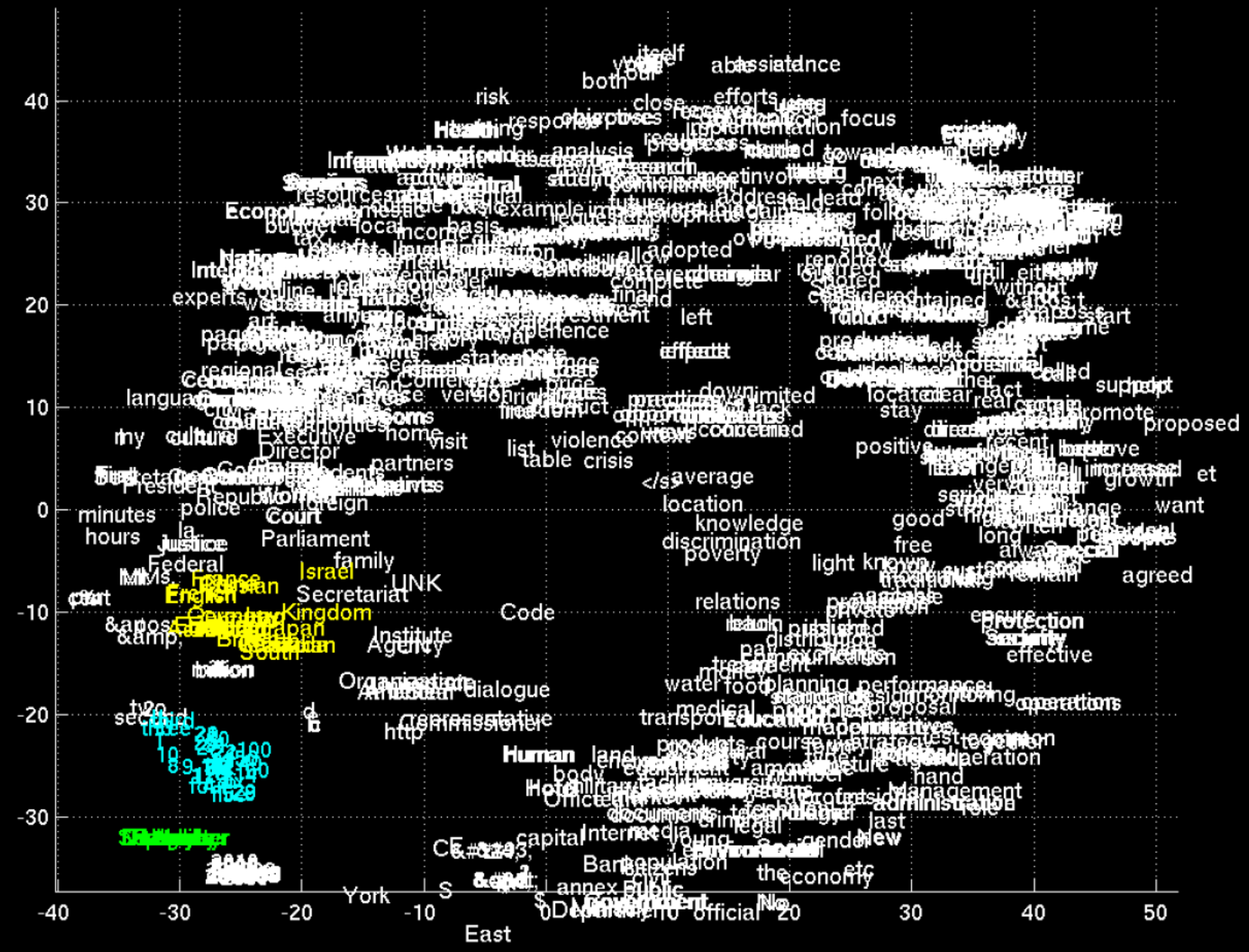
- Target: 25011.

- Input:

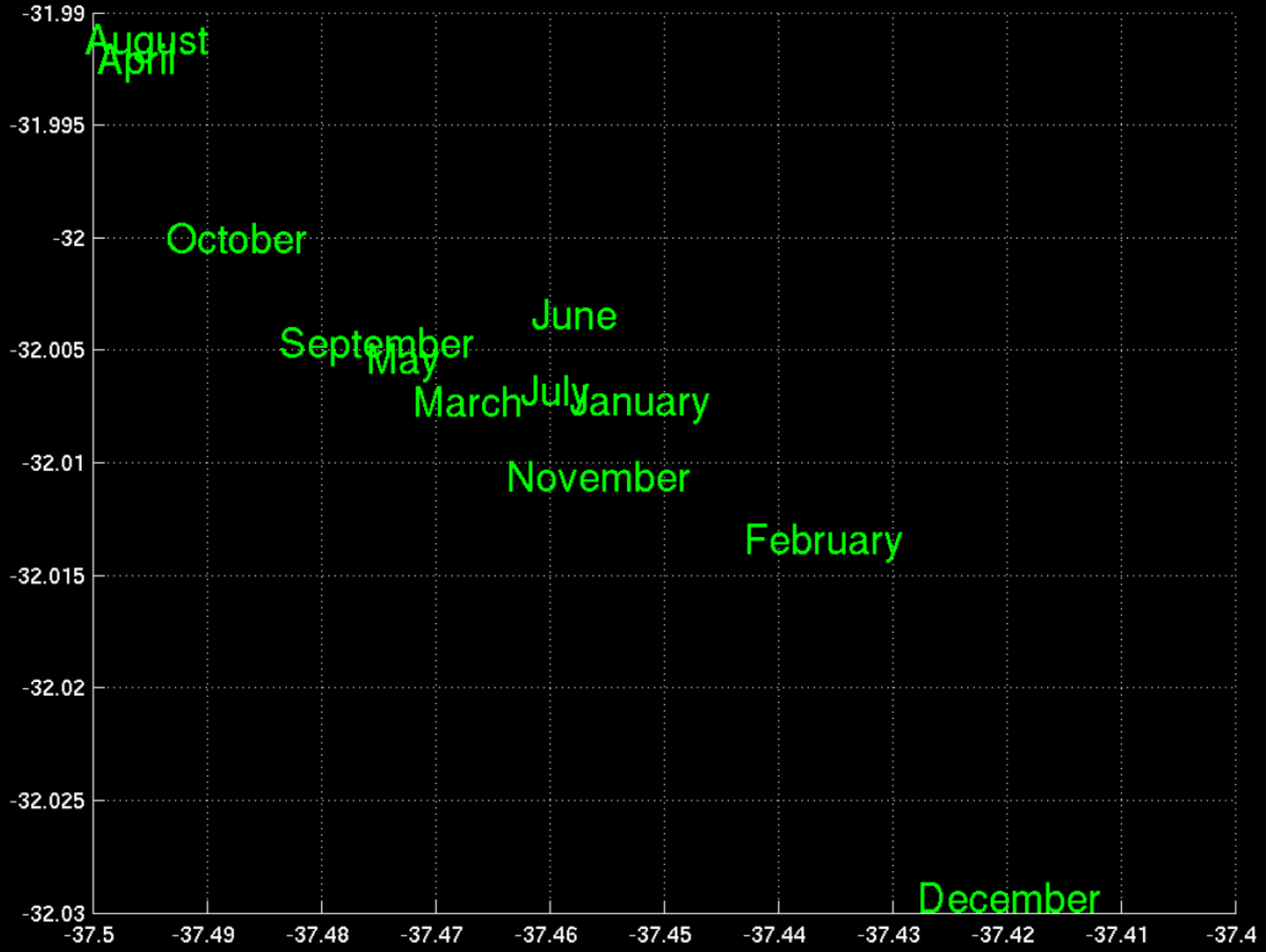
```
i=8827
c=(i-5347)
print((c+8704) if
2641<8500 else 5308)
```

- Target: 12184.

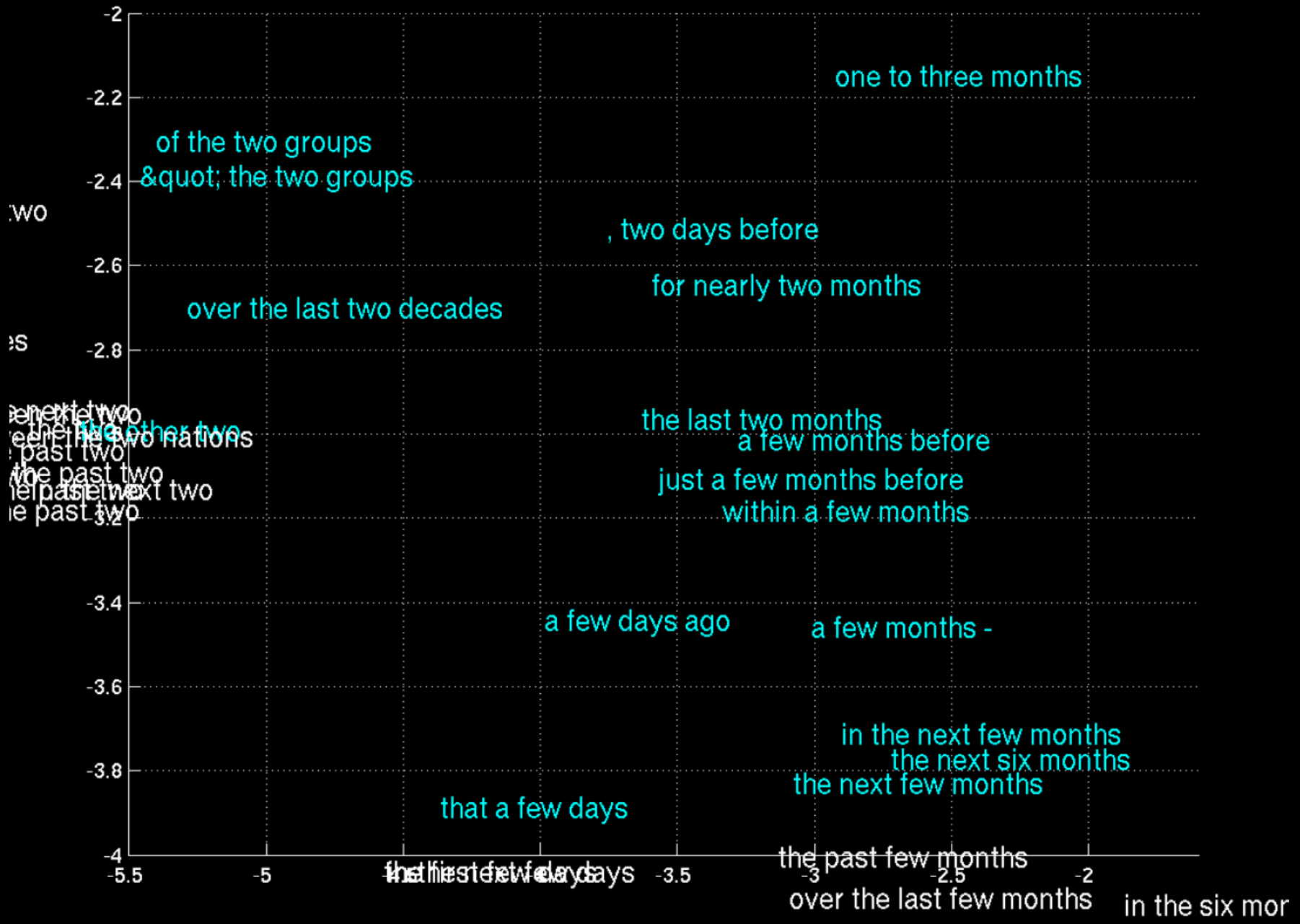
Zaremba & Sutskever (2015) Learning to Execute



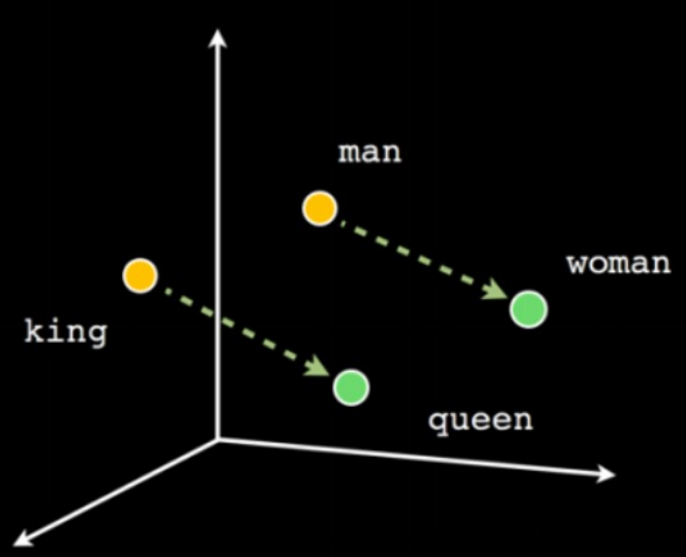
Cho et al. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation



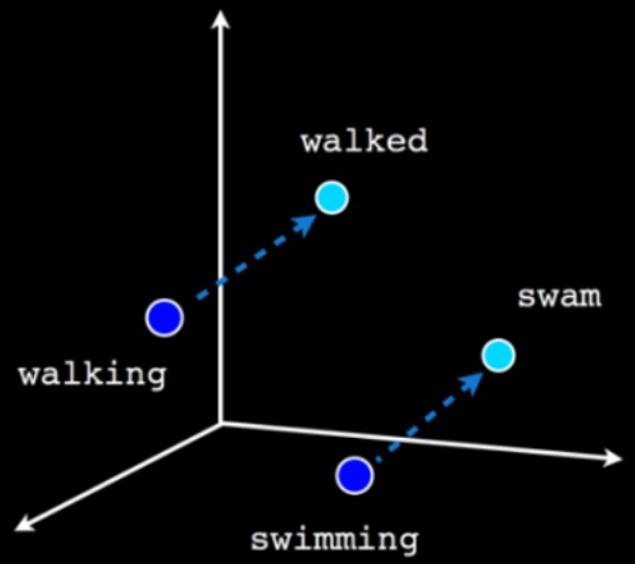
Cho et al. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation



Cho et al. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation

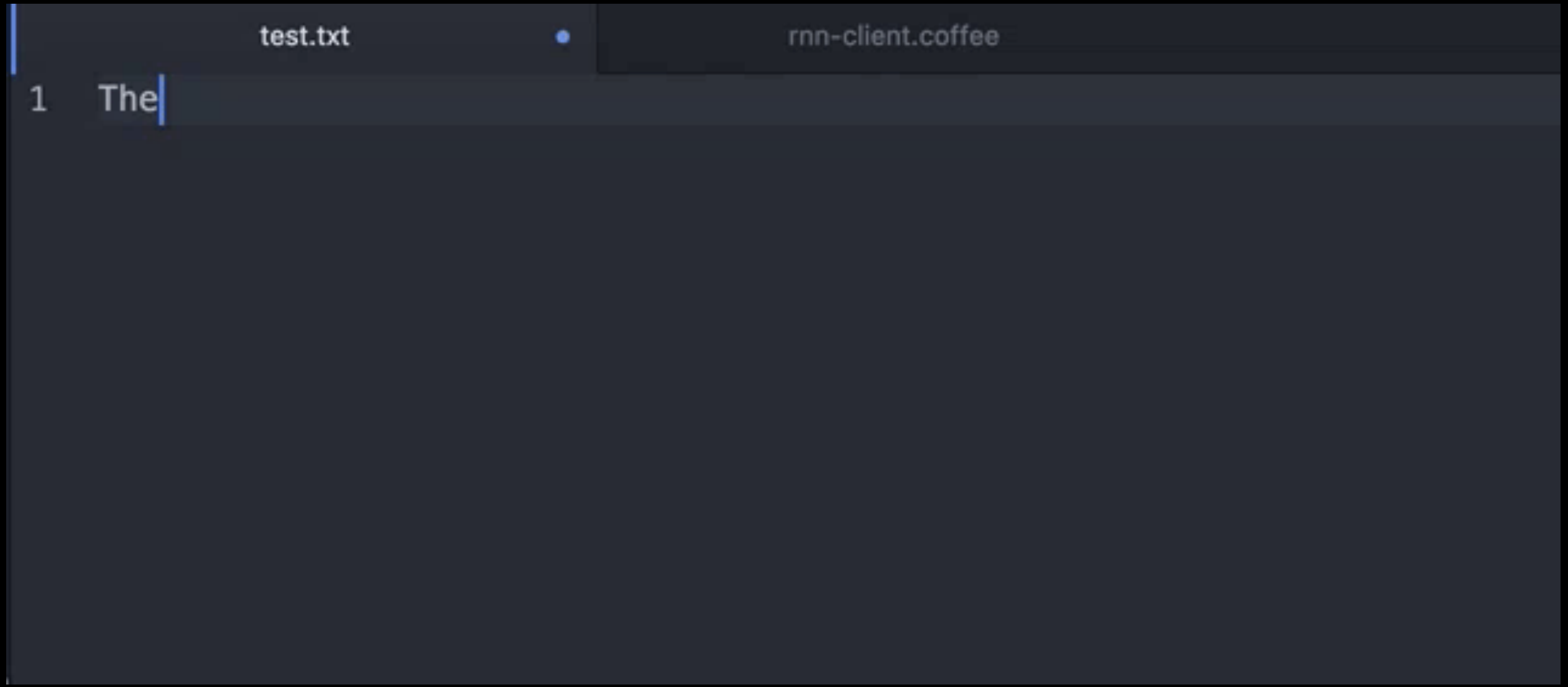


Male-Female



Verb tense

Country-Capital



Sloan (2016) rnn-writer – github.com/robinsloan/rnn-writer

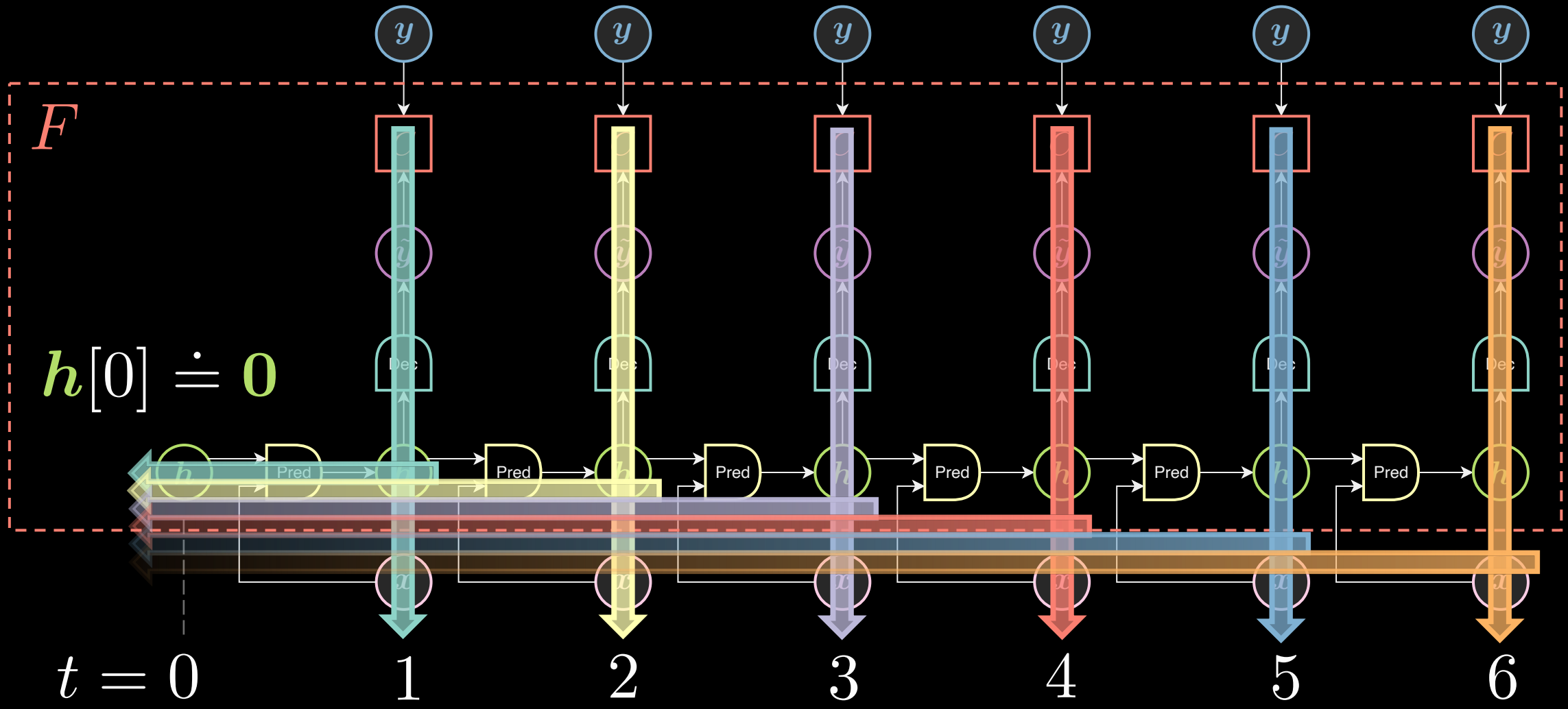
RNN training

Back propagation through time (BPTT)

RNN training

$$\boldsymbol{h}[t] = \mathrm{Pred}(\boldsymbol{h}[t-1], \boldsymbol{x}[t])$$

$$\tilde{\boldsymbol{y}}[t] = \mathrm{Dec}(\boldsymbol{h}[t])$$

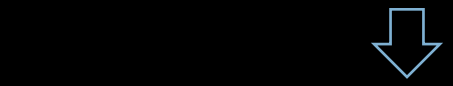


Training example

Language modelling

Batch-ification

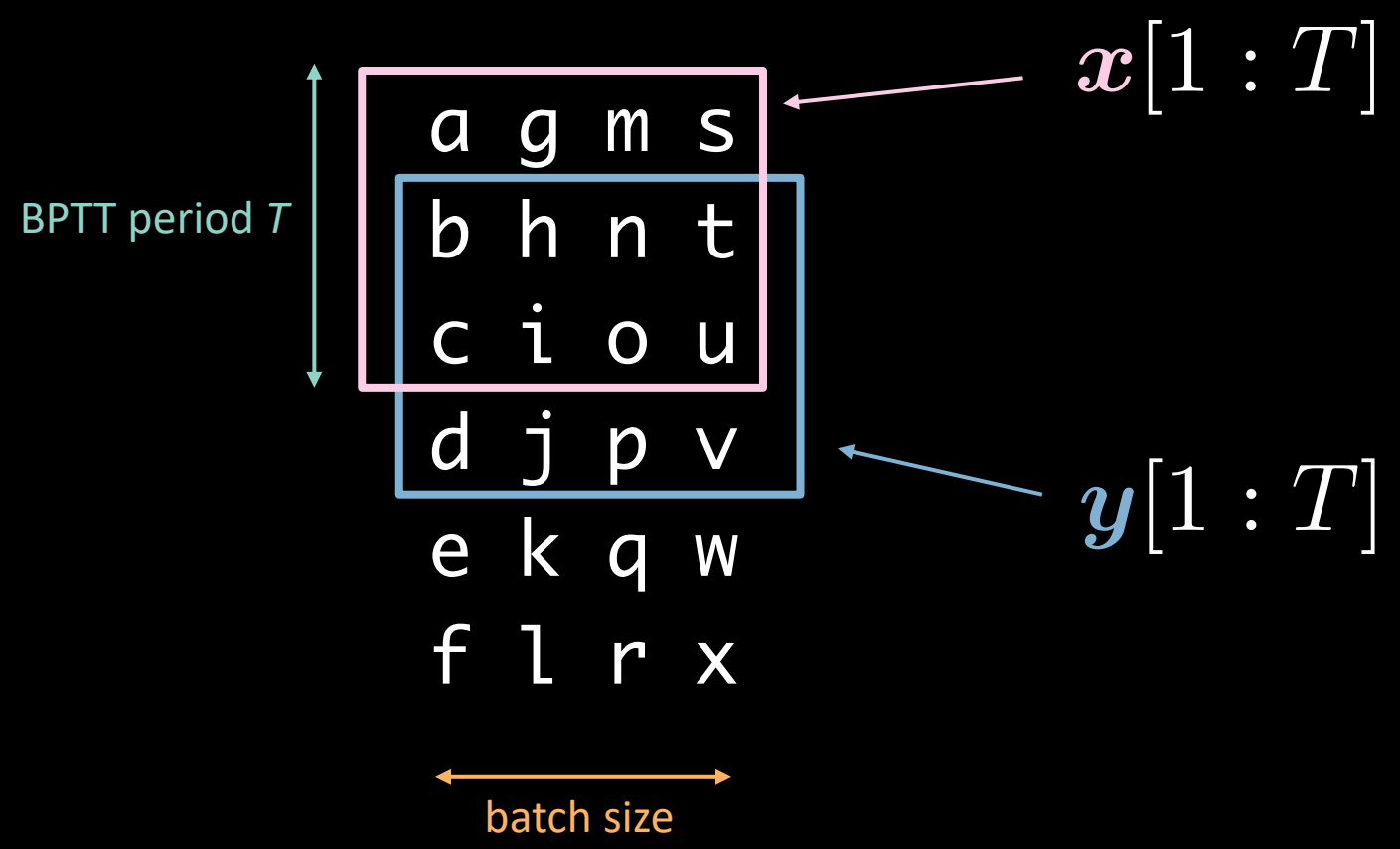
abcdefghijklmnopqrstuvwxyz



a	g	m	s
b	h	n	t
c	i	o	u
d	j	p	v
e	k	q	w
f	l	r	x

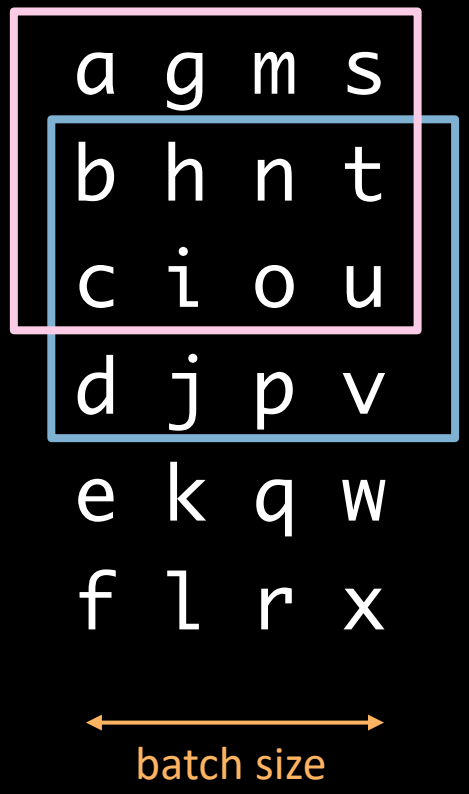
Check word_language_model @ github.com/pytorch/examples/

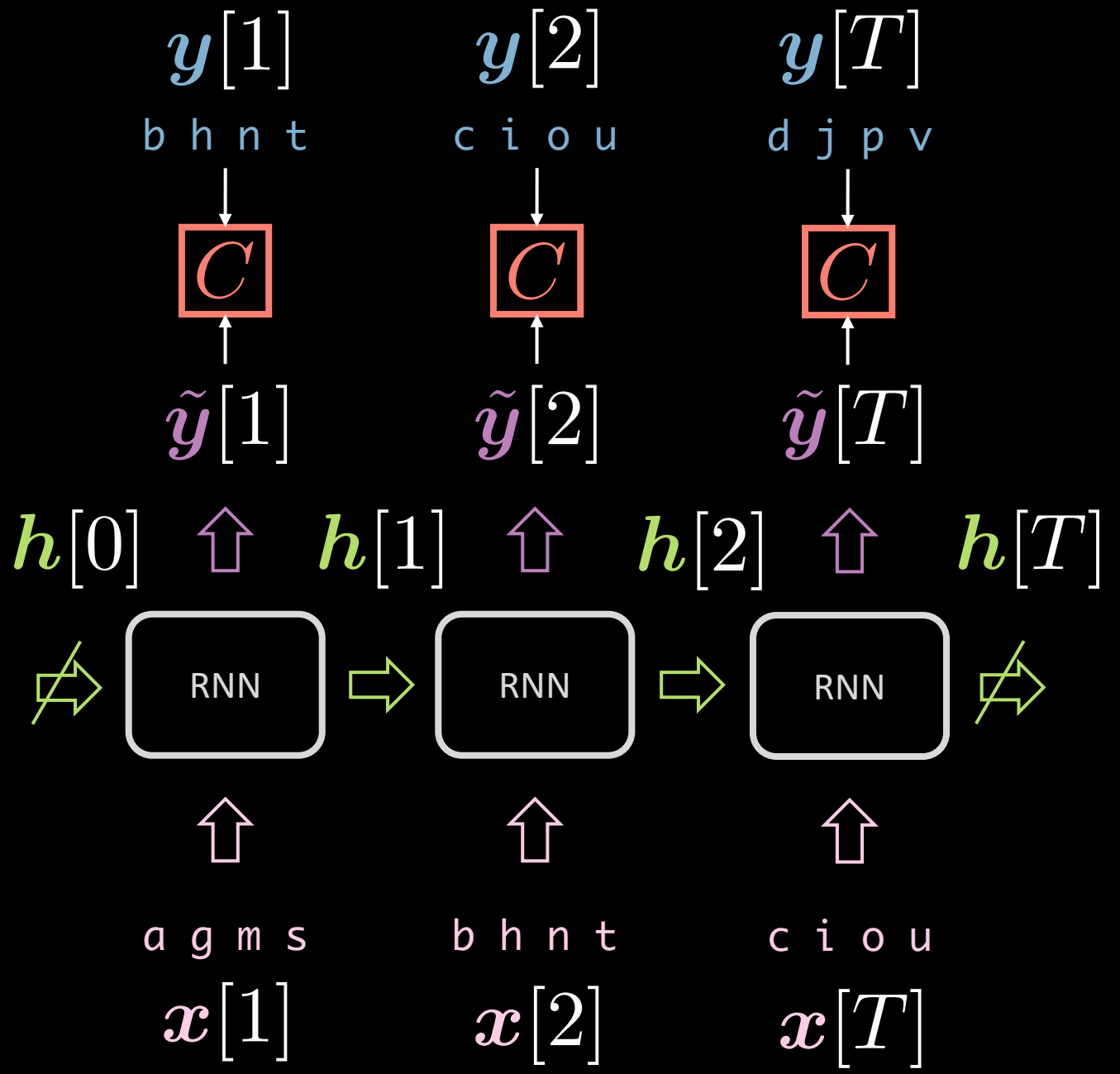
Get batch (I)



Check `word_language_model` @ `github.com/pytorch/examples/`

Get batch (II)

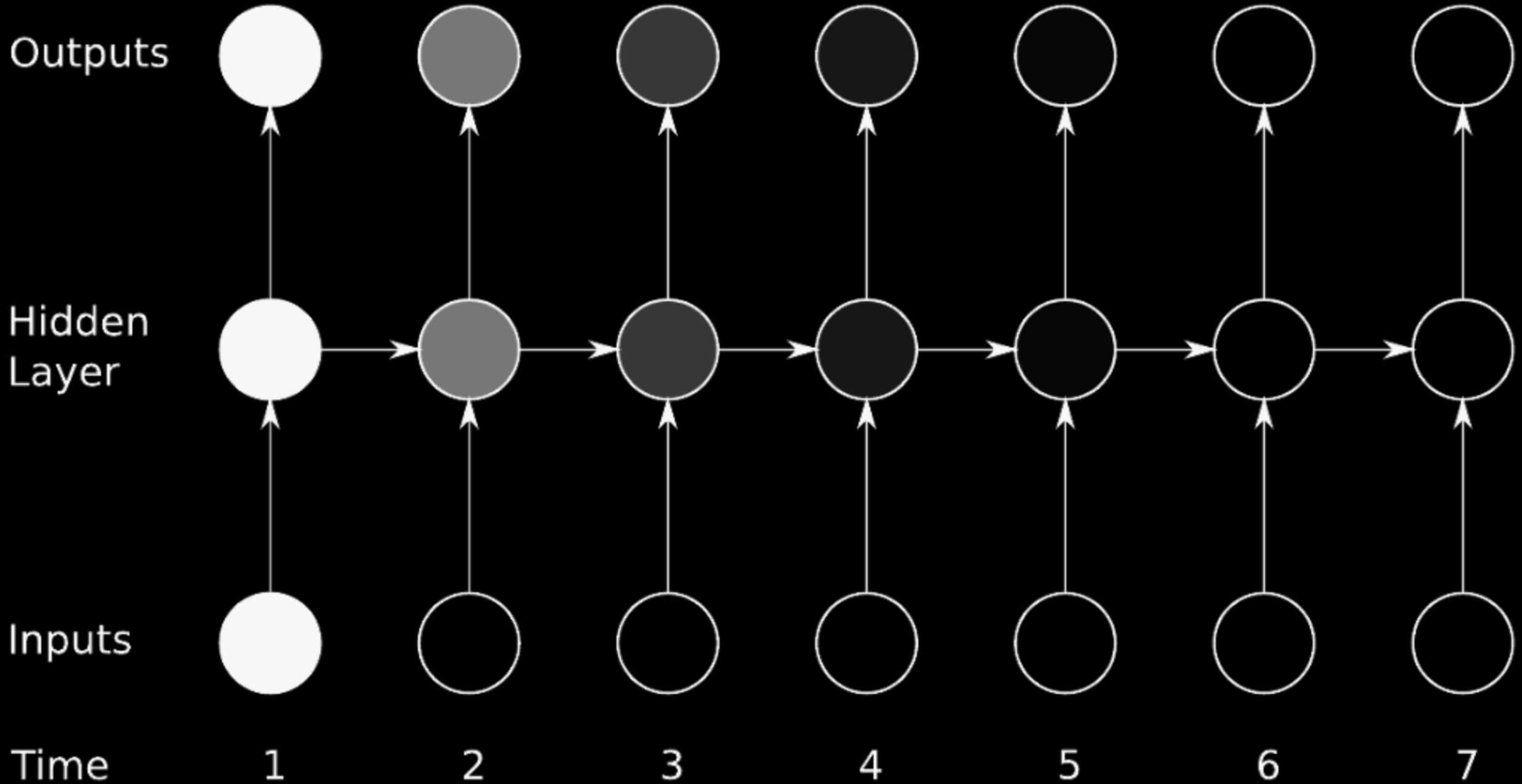




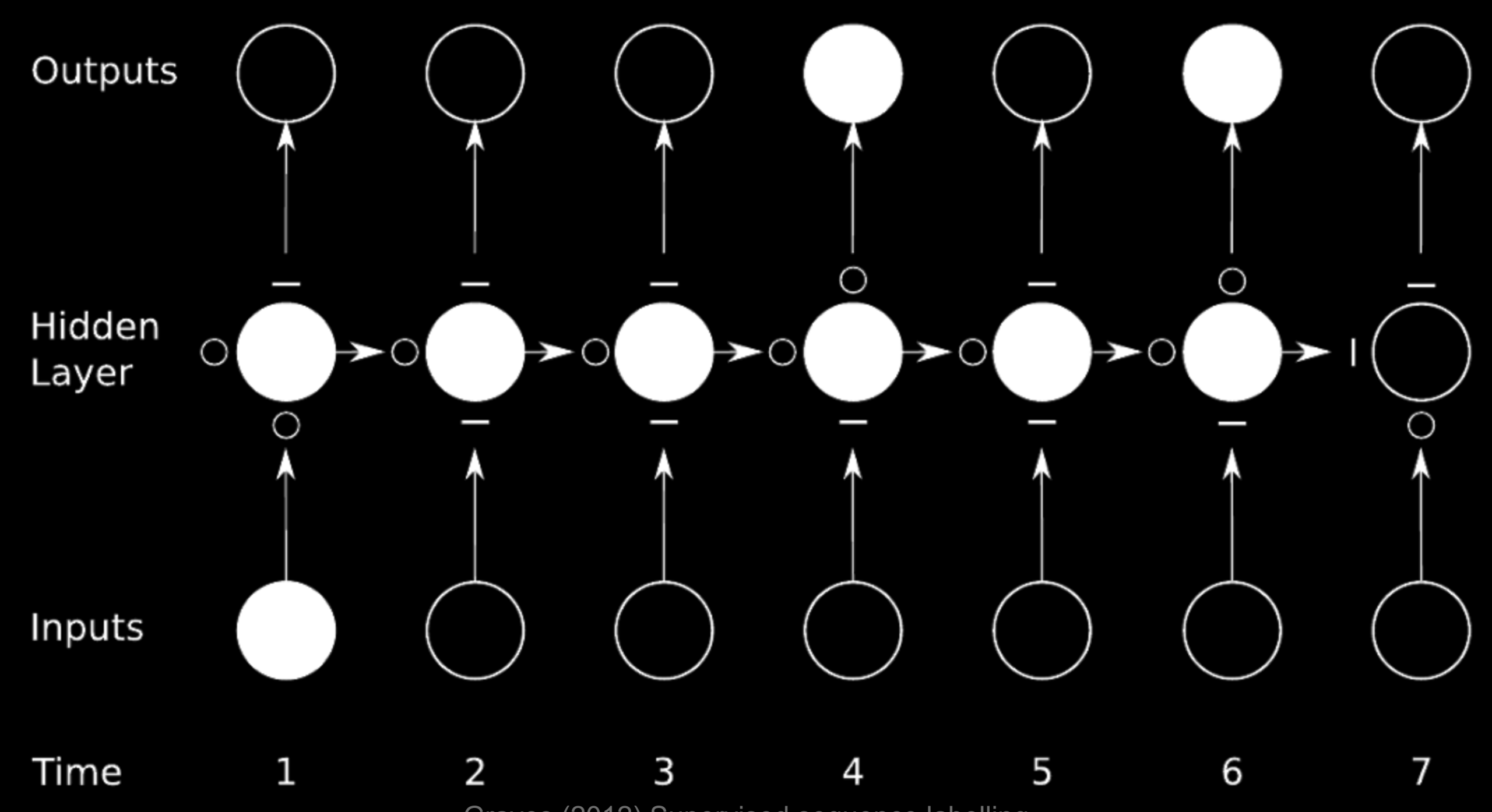
Check word_language_model @ github.com/pytorch/examples/

Vanishing & exploding gradients

Limitations of temporally deep nets



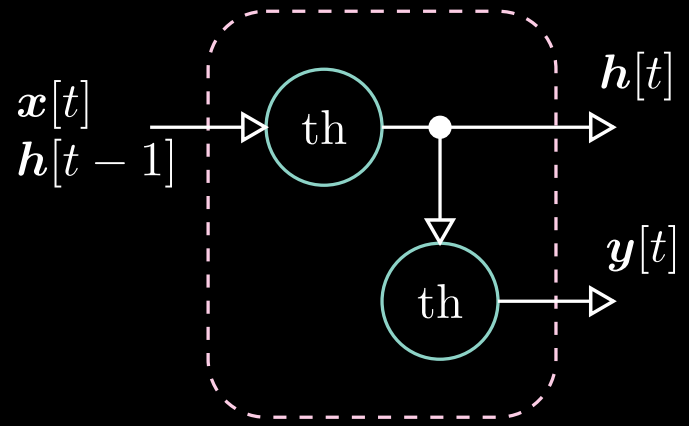
Graves (2012) Supervised sequence labelling



Graves (2012) Supervised sequence labelling

Long Short-Term Memory

Gated RNN



$$\boldsymbol{h}[t] = g(\boldsymbol{W}_h \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_h)$$

$$\hat{\boldsymbol{y}}[t] = g(\boldsymbol{W}_y \boldsymbol{h}[t] + \boldsymbol{b}_y)$$

$$\boldsymbol{i}[t] = \sigma(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i)$$

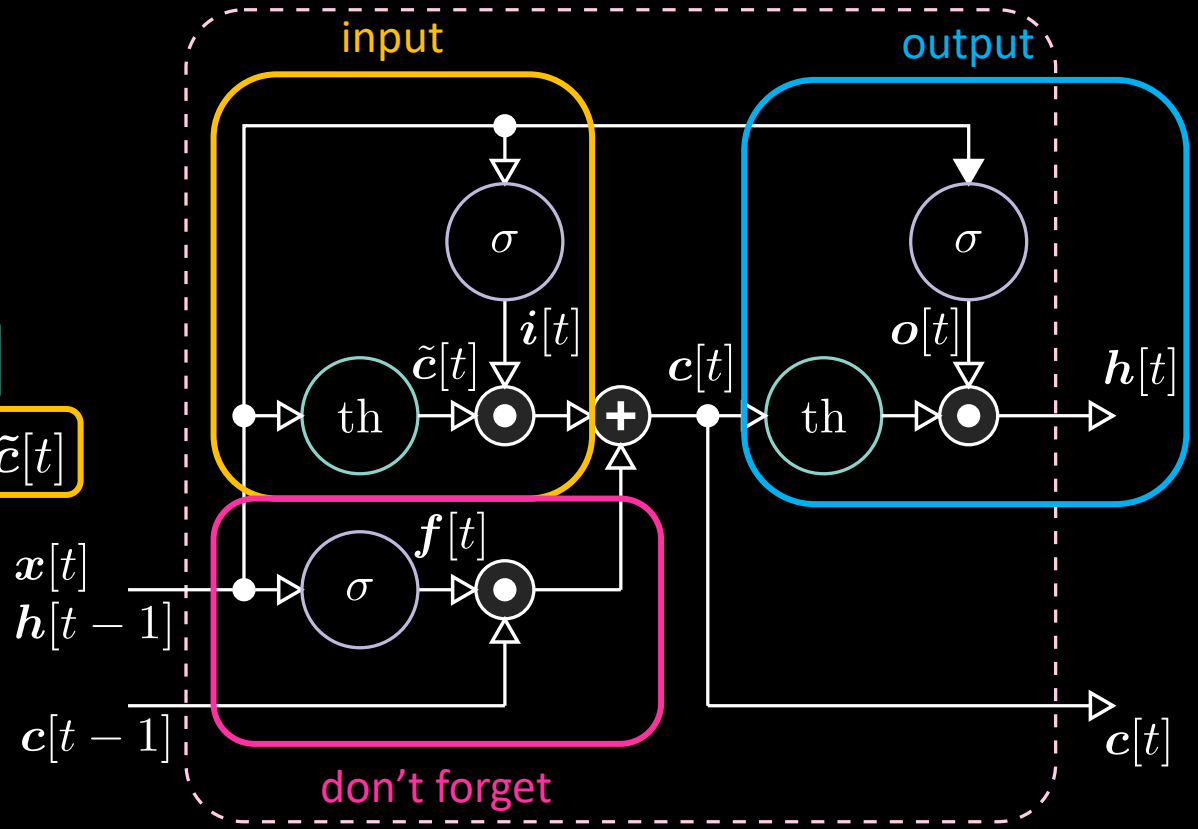
$$\boldsymbol{f}[t] = \sigma(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f)$$

$$\boldsymbol{o}[t] = \sigma(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o)$$

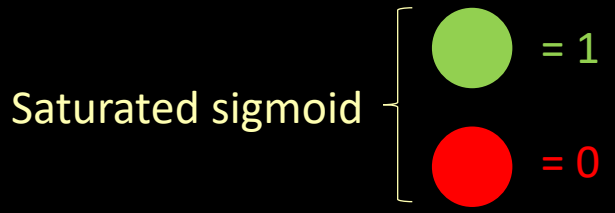
$$\tilde{\boldsymbol{c}}[t] = \tanh(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$



Controlling the output - OFF



$$\boldsymbol{i}[t] = \sigma\left(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i\right)$$

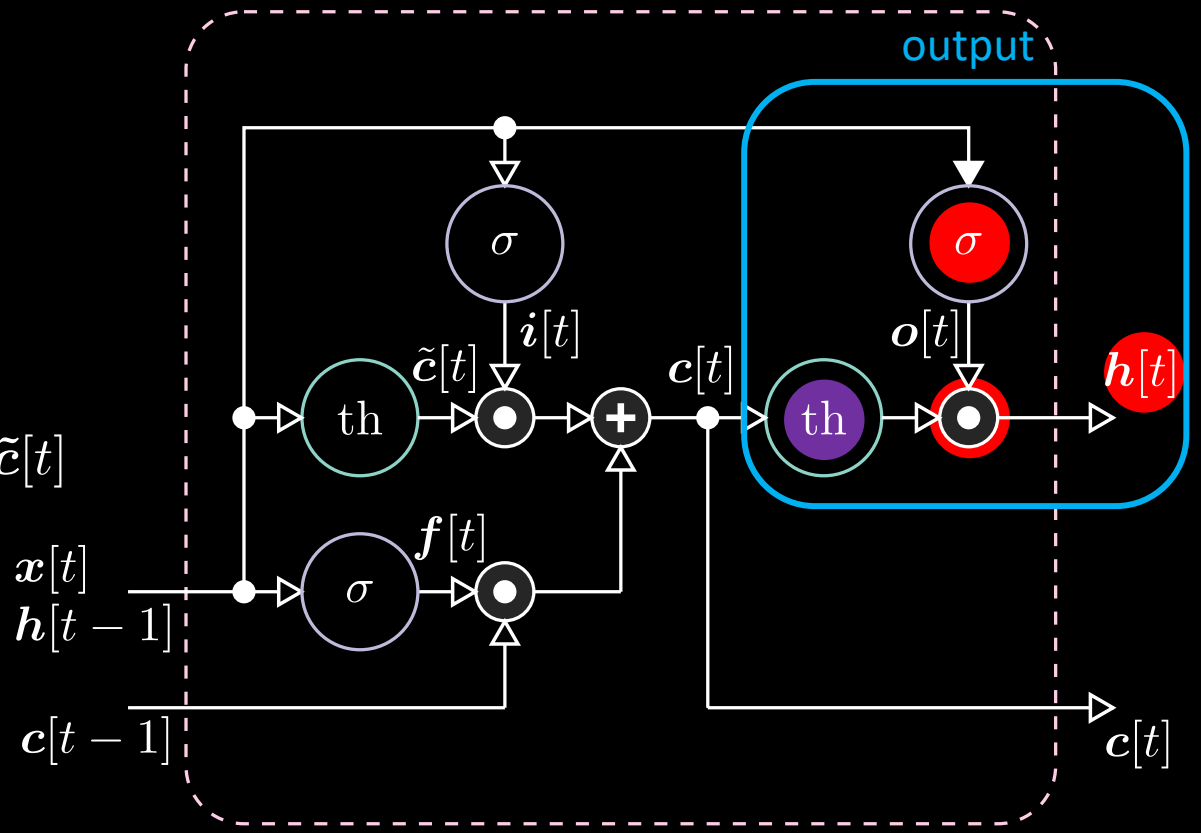
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}[t] = \sigma\left(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o\right)$$

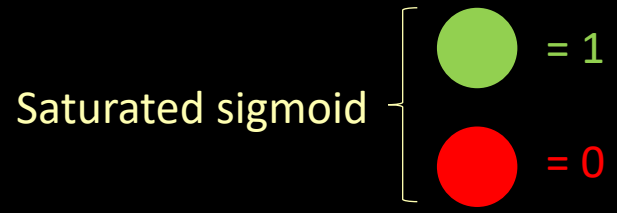
$$\tilde{\boldsymbol{c}}[t] = \tanh\left(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$



Controlling the output - ON



$$\boldsymbol{i}[t] = \sigma\left(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i\right)$$

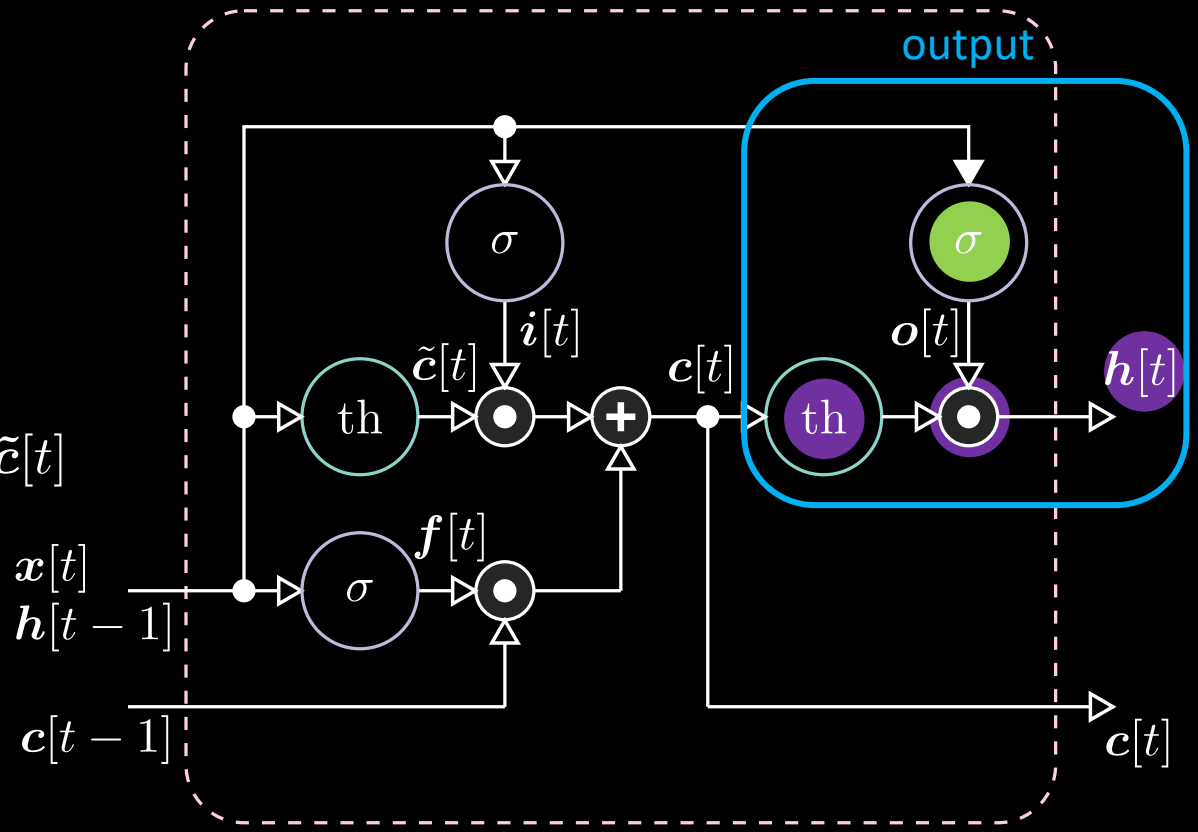
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}[t] = \sigma\left(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o\right)$$

$$\tilde{\boldsymbol{c}}[t] = \tanh\left(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$



Controlling the memory - reset

Saturated sigmoid
= 1
= 0

$$\boldsymbol{i}[t] = \sigma\left(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i\right)$$

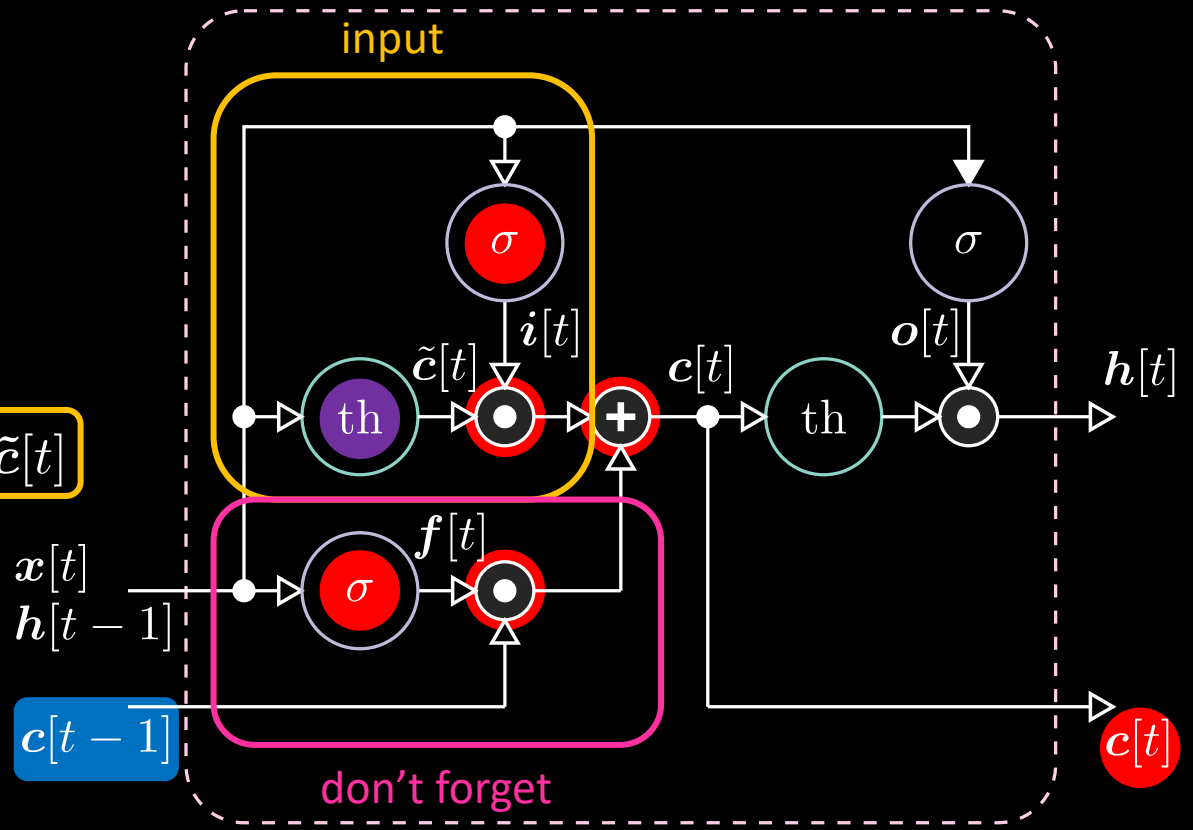
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}[t] = \sigma\left(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o\right)$$

$$\tilde{\boldsymbol{c}}[t] = \tanh\left(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$



Controlling the memory - keep

Saturated sigmoid
= 1
= 0

$$\boldsymbol{i}[t] = \sigma\left(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i\right)$$

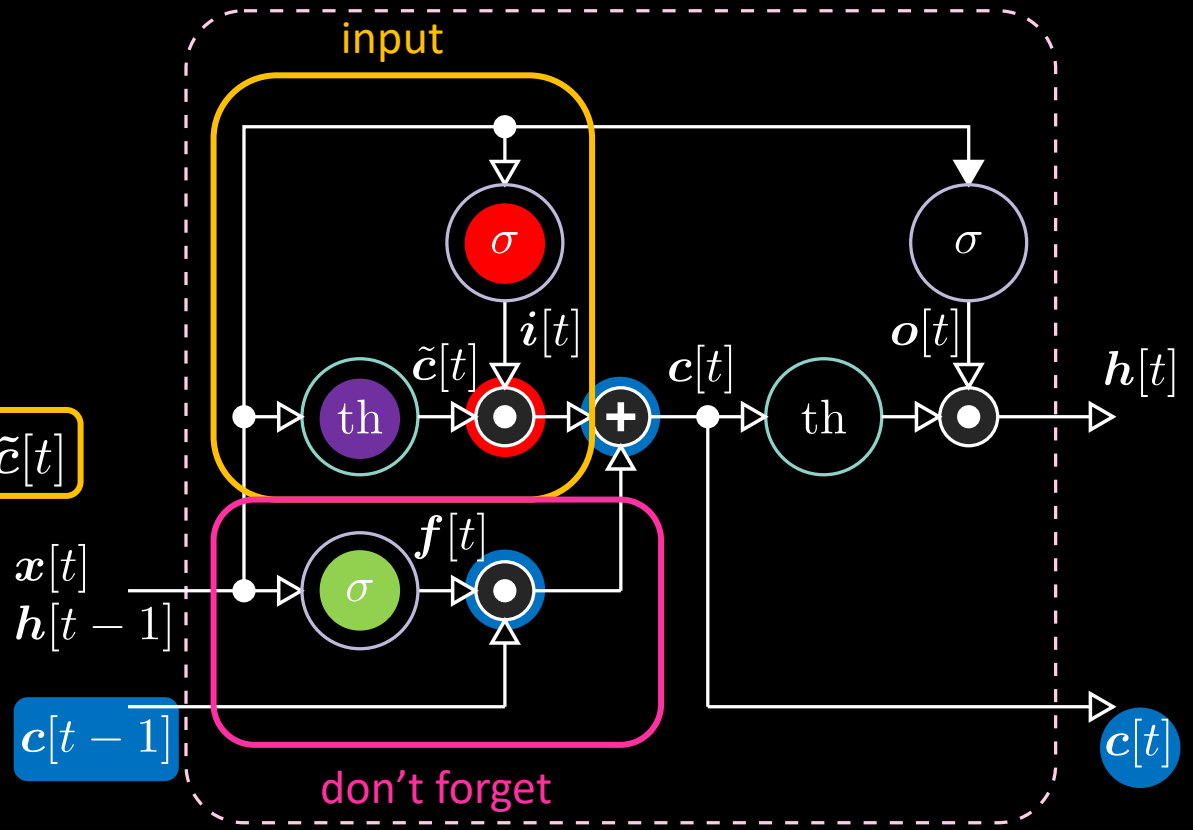
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}[t] = \sigma\left(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o\right)$$

$$\tilde{\boldsymbol{c}}[t] = \tanh\left(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$



Controlling the memory - write

Saturated sigmoid
= 1
= 0

$$\boldsymbol{i}[t] = \sigma\left(\boldsymbol{W}_i \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_i\right)$$

$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}[t] = \sigma\left(\boldsymbol{W}_o \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_o\right)$$

$$\tilde{\boldsymbol{c}}[t] = \tanh\left(\boldsymbol{W}_c \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{c}[t] = \boldsymbol{f}[t] \odot \boldsymbol{c}[t-1] + \boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]$$

$$\boldsymbol{h}[t] = \boldsymbol{o}[t] \odot \tanh(\boldsymbol{c}[t])$$