Foundations of DL

Deep Learning

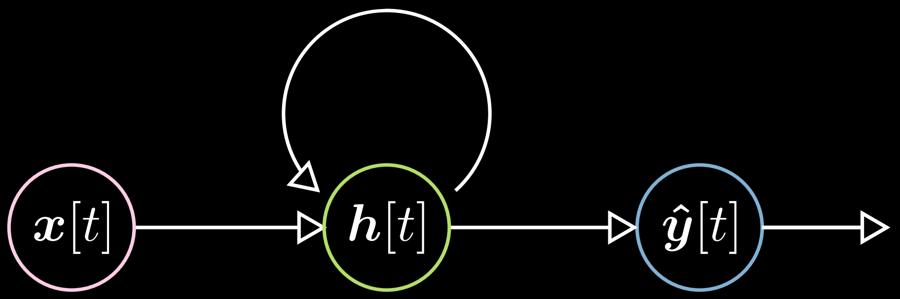
ALF

Alfredo Canziani, Ritchie Ng
@alfcnz, @RitchieNg

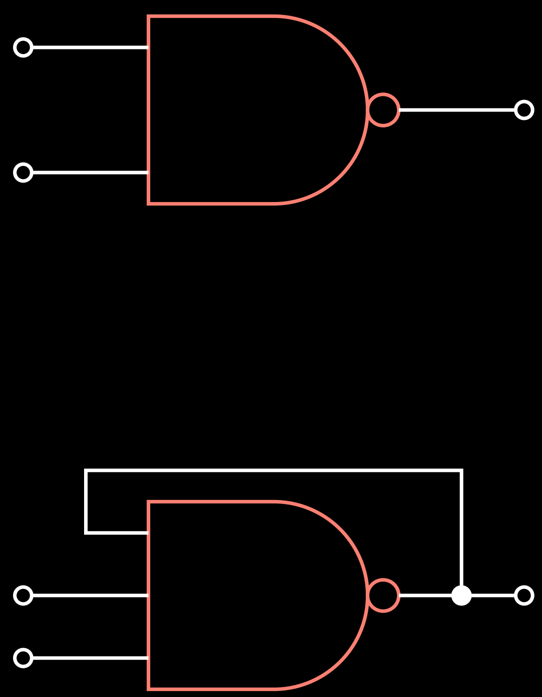
Recurrent Neural Nets

Handling sequential data

Vanilla and Recurrent NN

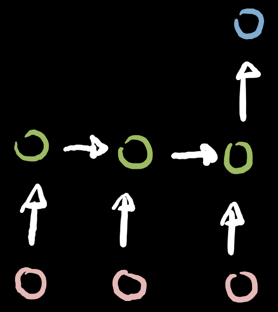


Combinatorial logic



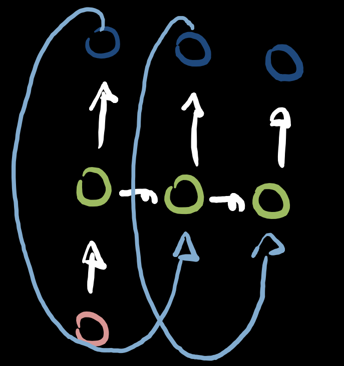
Sequential logic

Rationale



$\underline{x}[t] \longmapsto \hat{\underline{y}}[T]$

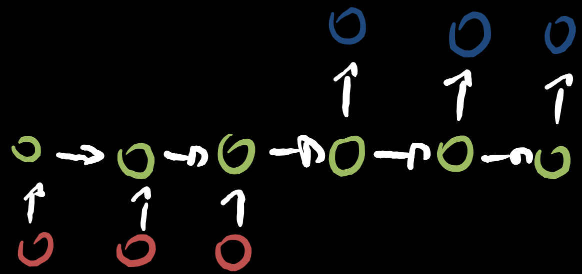
seq $\longmapsto$ vec

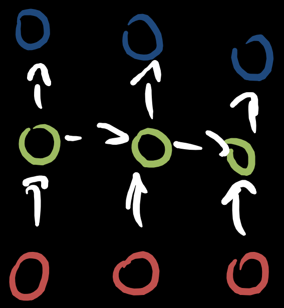


$\underline{x}[0] \longmapsto \hat{\underline{y}}[t]$

vec $\longmapsto$ seq

seq $\longmapsto$ seq





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))
```

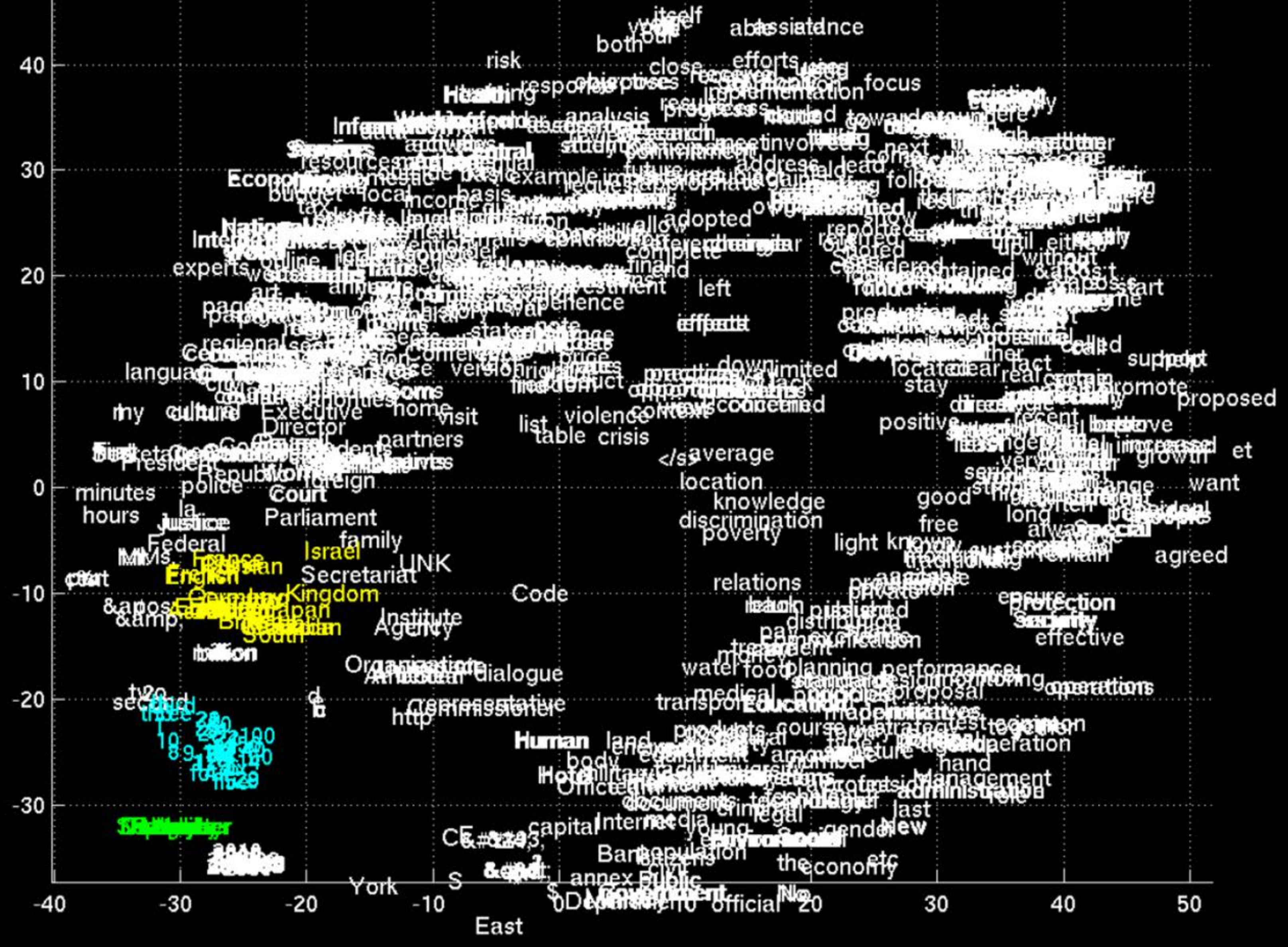
- **Target:** 25011.

- **Input:**

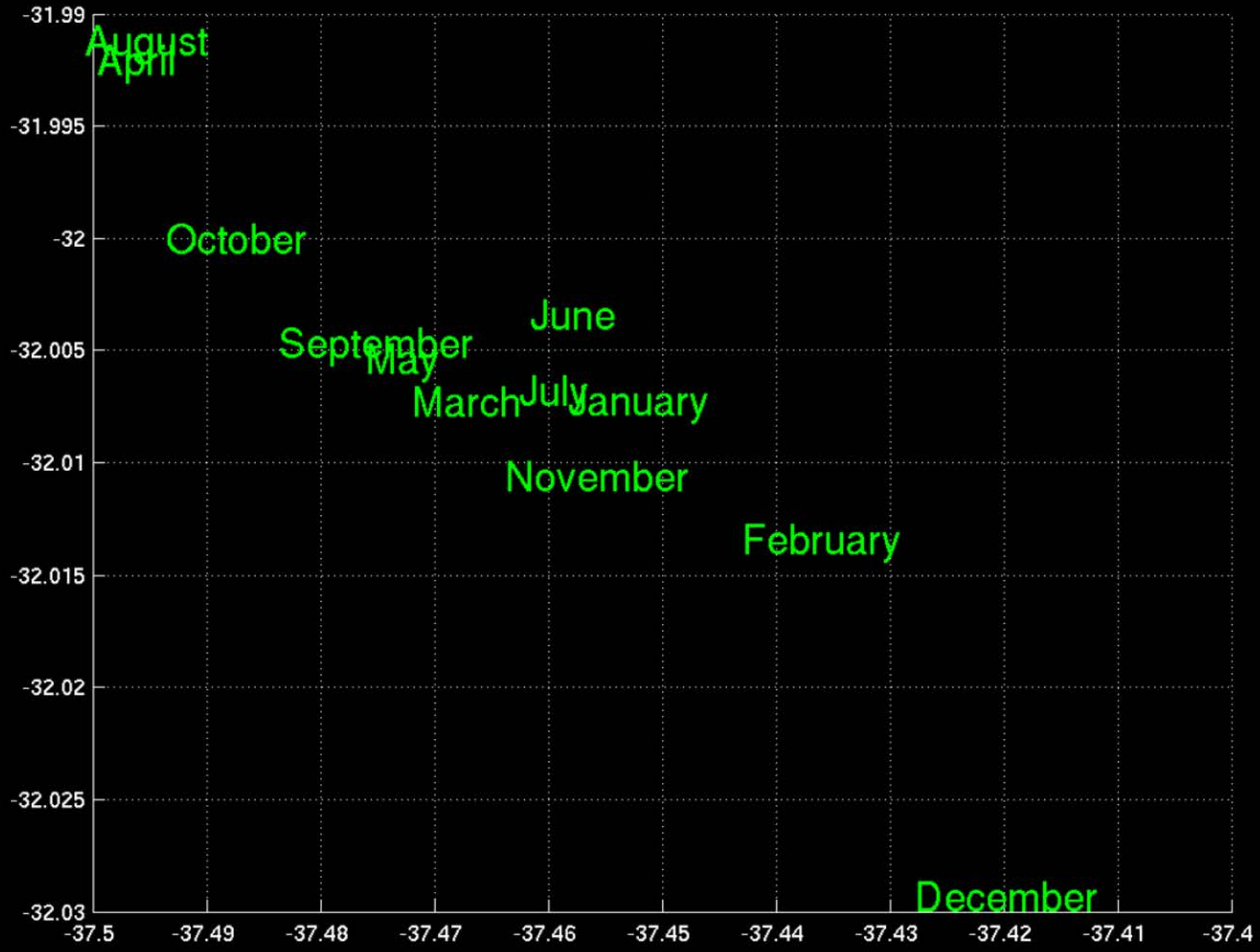
```
i=8827
c=(i-5347)
print((c+8704) if
2641<8500 else 5308)
```

- **Target:** 12184.

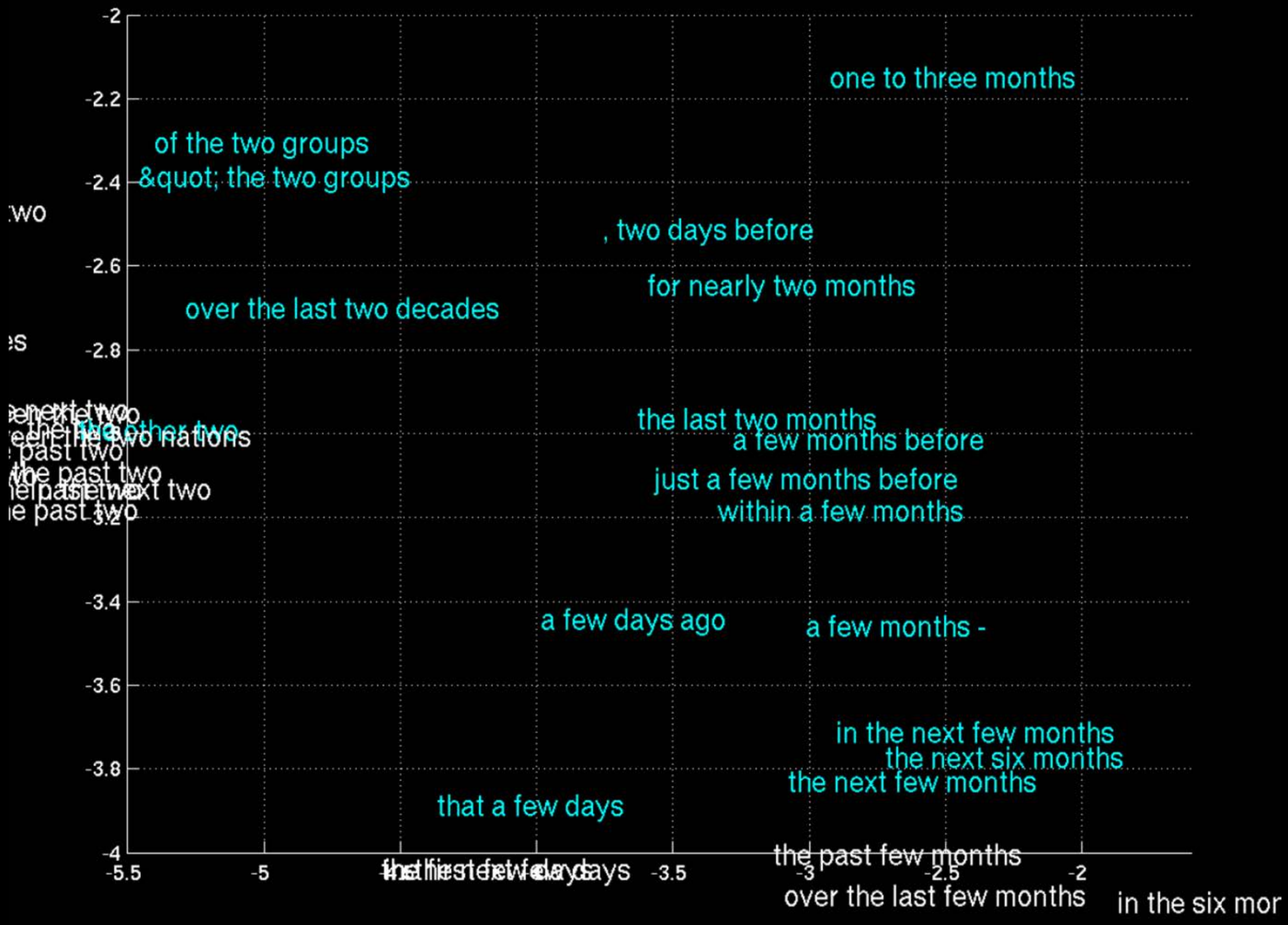
Zaremba & Sutskever (2015) Learning to Execute



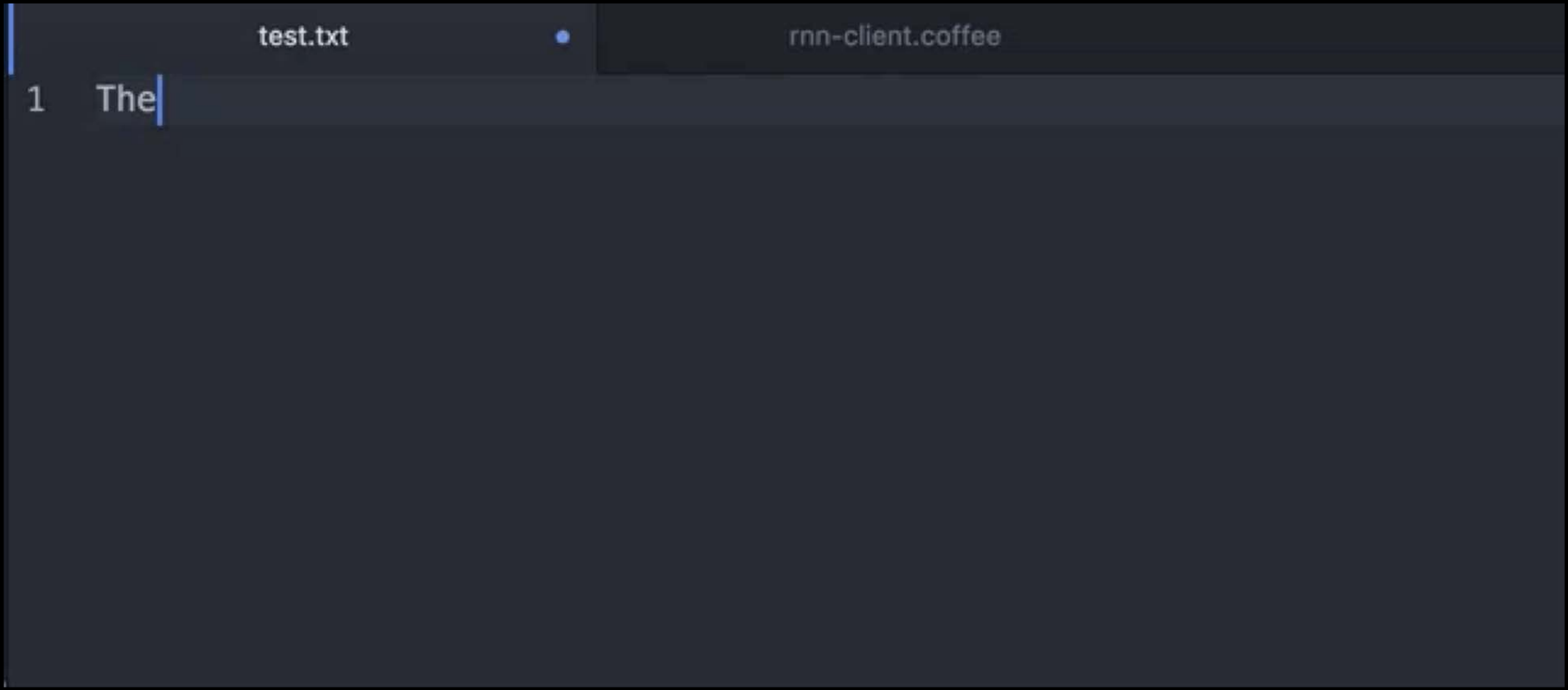
Cho et al. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation



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



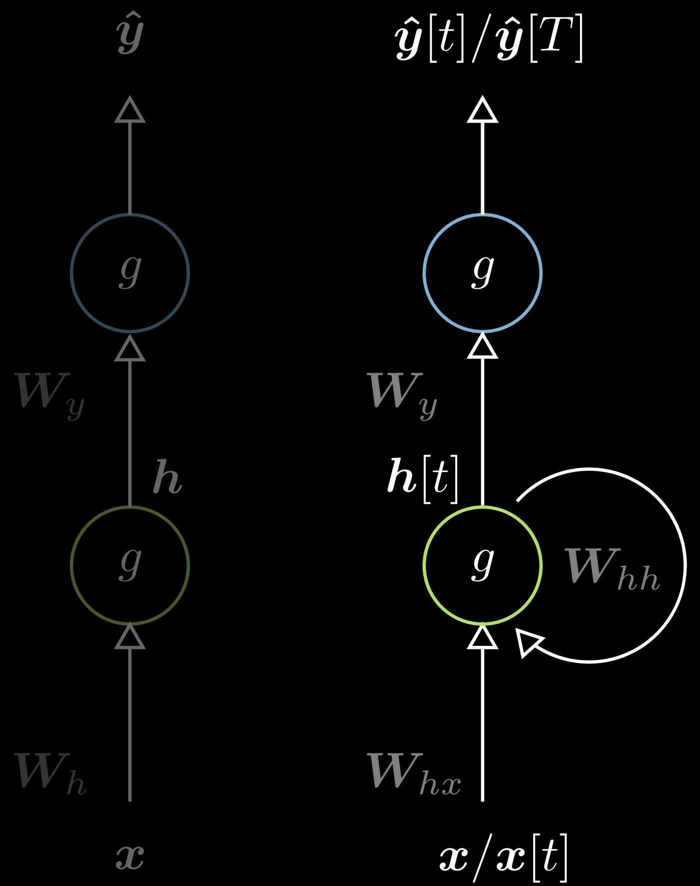
Cho et al. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation



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

RNN training

Back propagation through time (BPTT)

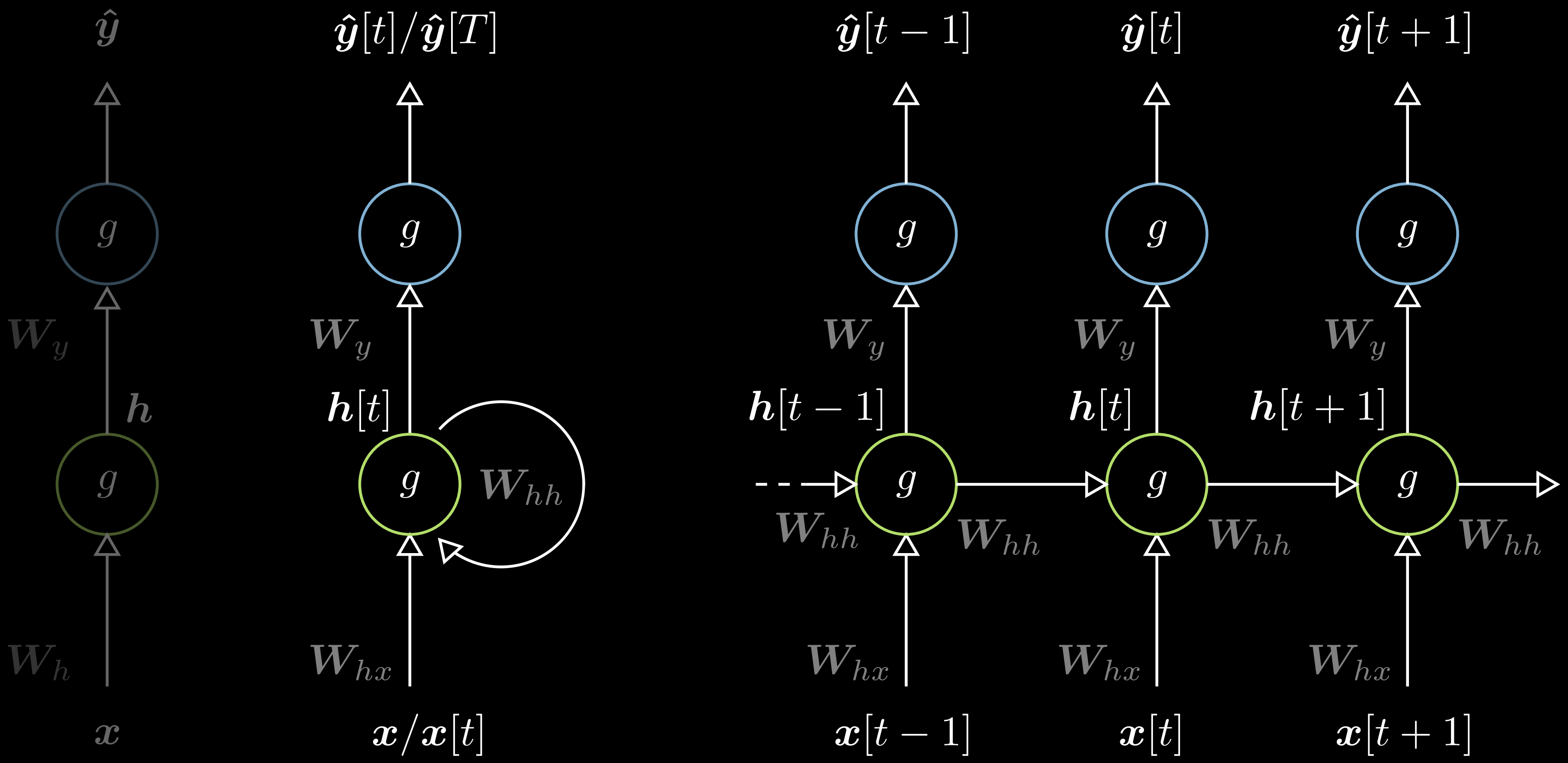


$$\boldsymbol{x} \in \mathbb{R}^q,\ \boldsymbol{h} \in \mathbb{R}^r,\ \boldsymbol{y} \in \mathbb{R}^s$$

$$\boldsymbol{h} = g(\boldsymbol{W}_h \boldsymbol{x} + \boldsymbol{b}_h)$$
$$\hat{\boldsymbol{y}} = g(\boldsymbol{W}_y \boldsymbol{h} + \boldsymbol{b}_y)$$

$$\phi : \mathbb{R}^q \to \mathbb{R}^r \to \mathbb{R}^s, r \gg q, s$$

$$\boldsymbol{h}[t] = g\left(\boldsymbol{W}_h \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_h\right)$$
$$\boldsymbol{h}[0] \doteq \boldsymbol{0}, \boldsymbol{W}_h \doteq \begin{bmatrix} \boldsymbol{W}_{hx} & \boldsymbol{W}_{hh} \end{bmatrix}$$
$$\hat{\boldsymbol{y}}[t] = g(\boldsymbol{W}_y \boldsymbol{h}[t] + \boldsymbol{b}_y)$$



$\hat{\boldsymbol{y}}$
$\boldsymbol{W}_y$
$\boldsymbol{h}$
g
g
$\boldsymbol{W}_h$
$\boldsymbol{x}$
$\hat{\boldsymbol{y}}[t]/\hat{\boldsymbol{y}}[T]$
$\boldsymbol{W}_y$
$\boldsymbol{h}[t]$
$\boldsymbol{W}_{hh}$
$\boldsymbol{W}_{hx}$
$\boldsymbol{x}/\boldsymbol{x}[t]$
$\hat{\boldsymbol{y}}[t-1]$
$\hat{\boldsymbol{y}}[t]$
$\hat{\boldsymbol{y}}[t+1]$
$\boldsymbol{W}_y$
$\boldsymbol{h}[t-1]$
$\boldsymbol{h}[t]$
$\boldsymbol{h}[t+1]$
$\boldsymbol{W}_{hh}$
$\boldsymbol{W}_{hx}$
$\boldsymbol{x}[t-1]$
$\boldsymbol{x}[t]$
$\boldsymbol{x}[t+1]$

Training example

Language modelling

Batch-ification

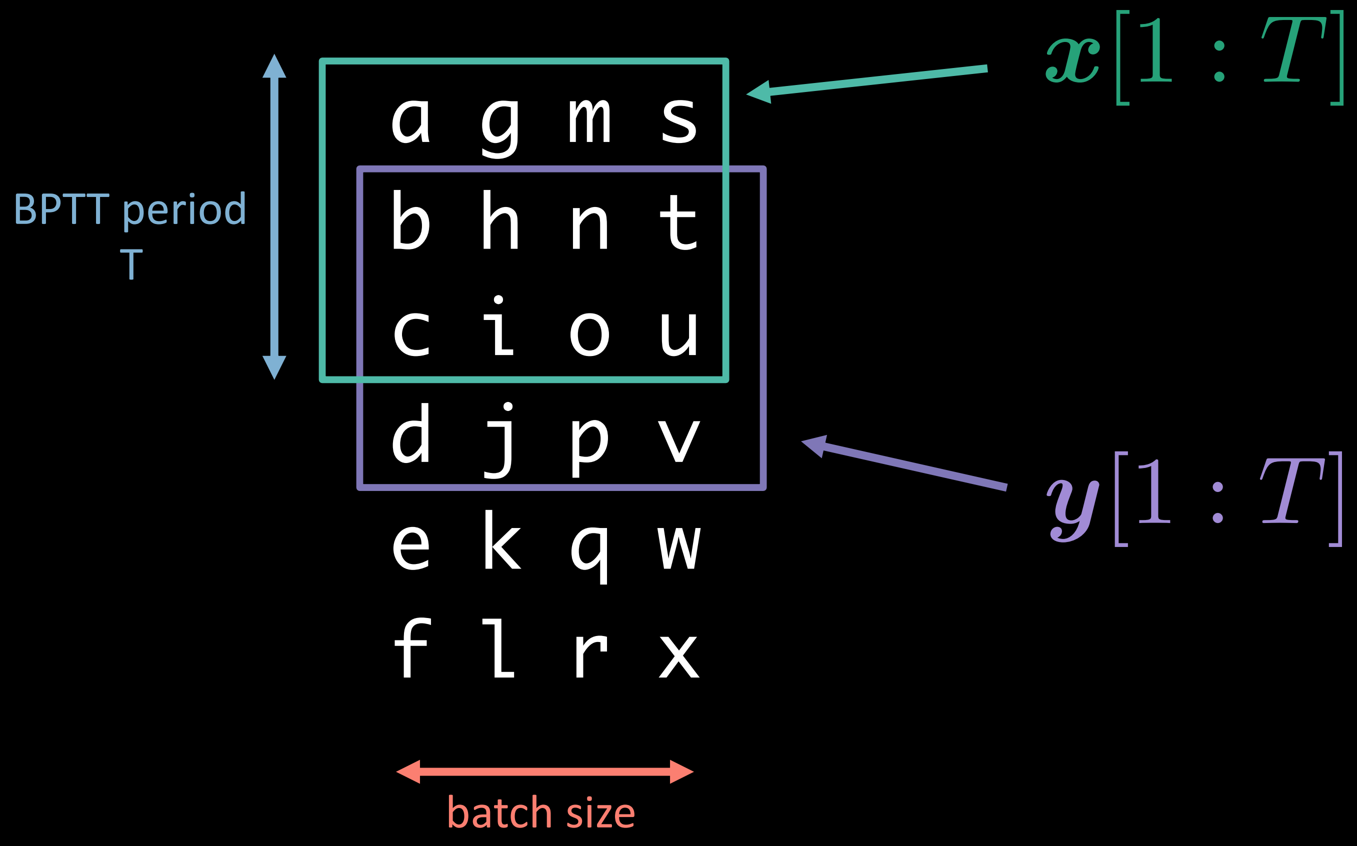
abcdefghijklmnopqrstuvwxyz

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

batch size

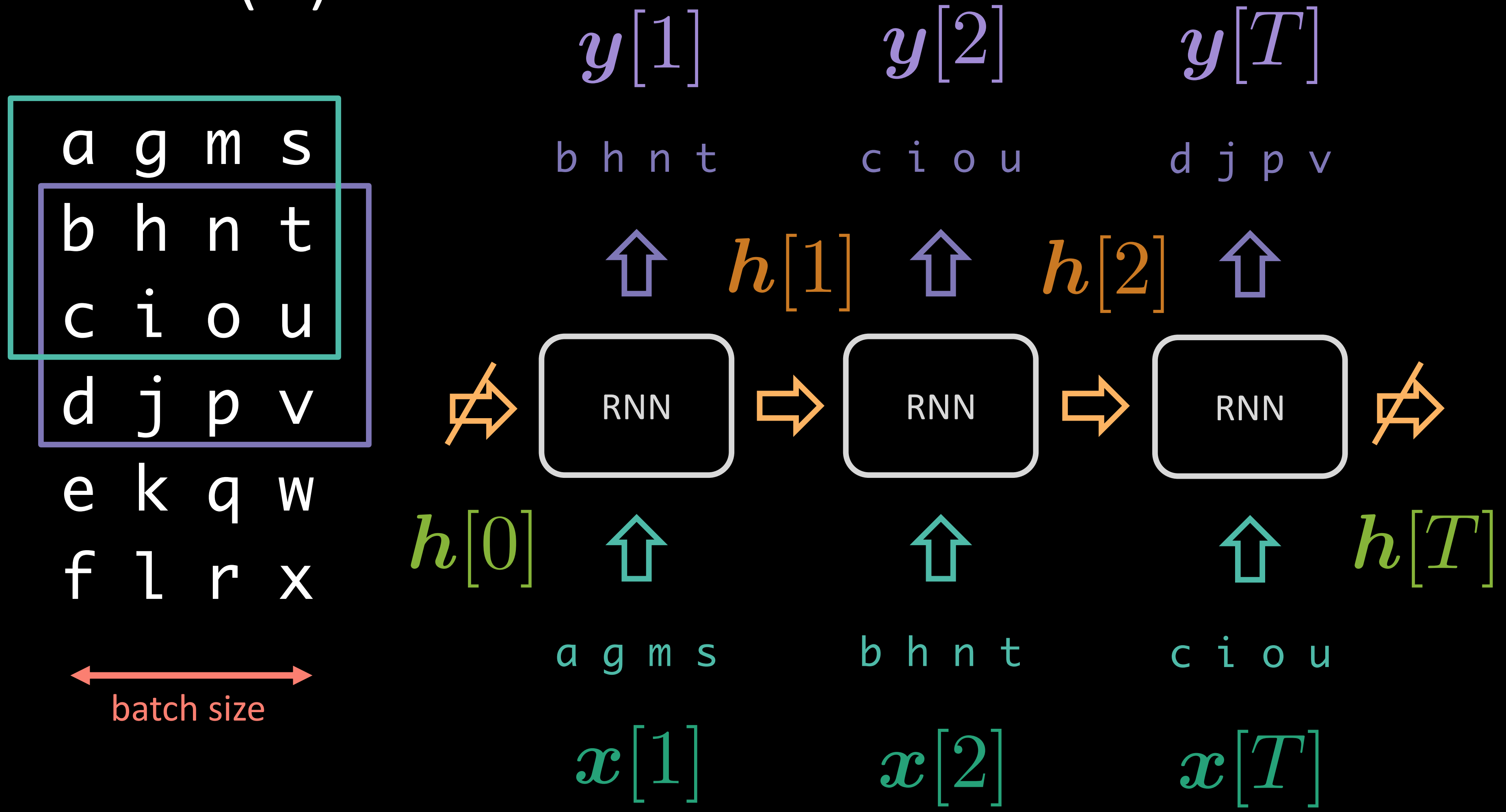
Check `word_language_model` @ `github.com/pytorch/examples/`

Get batch (I)



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

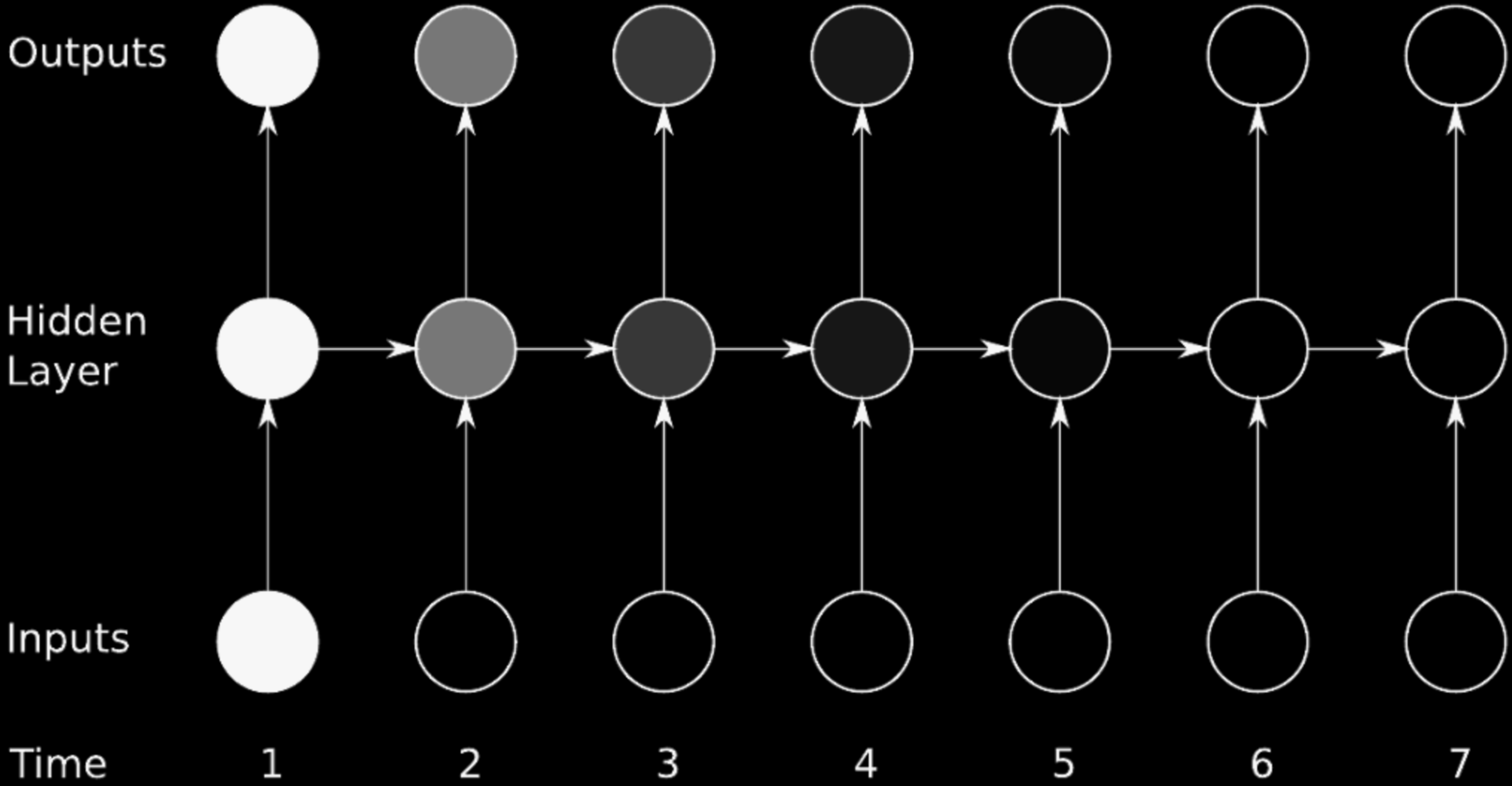
Get batch (II)



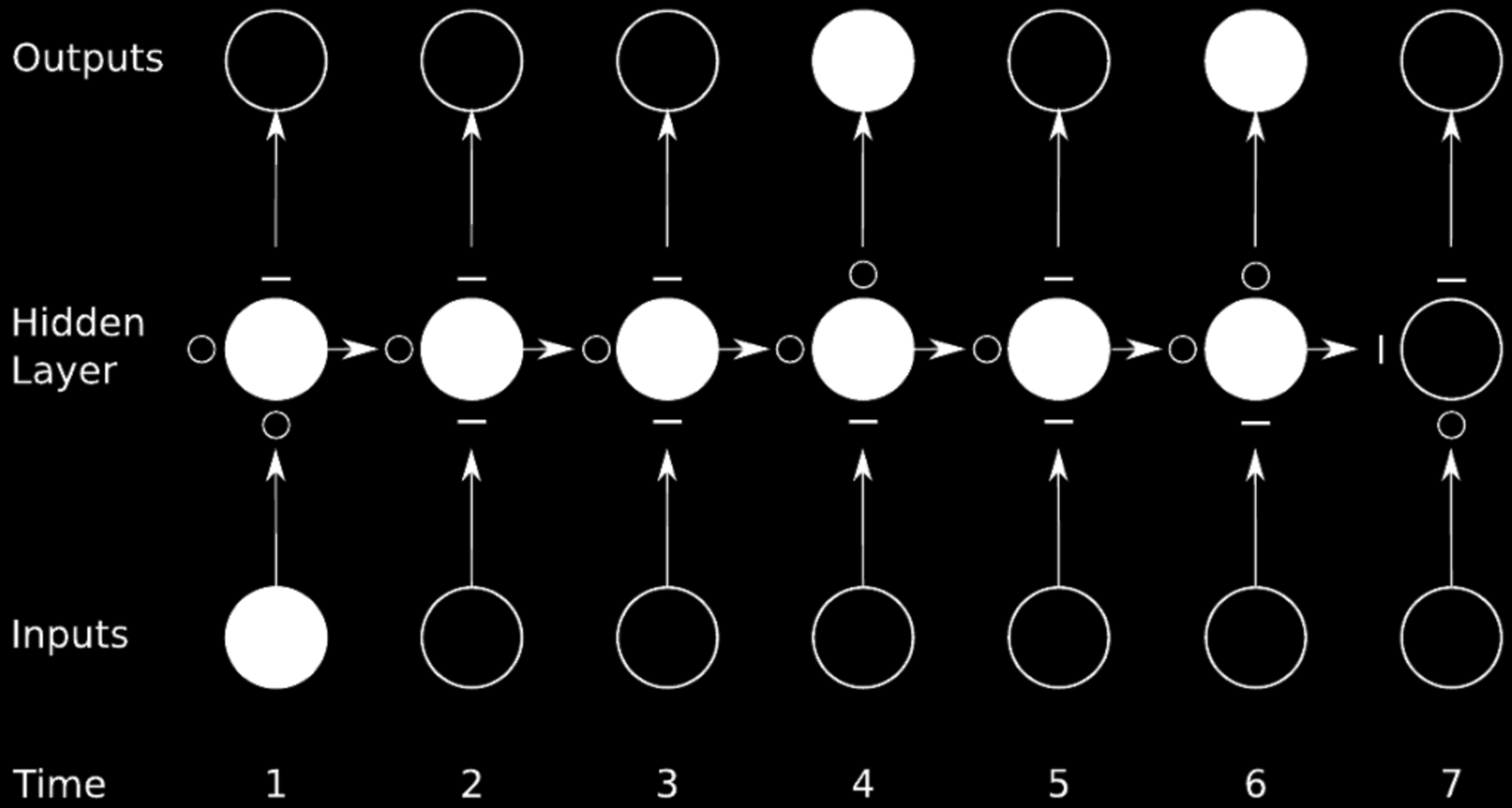
Check word_language_model @ github.com/pytorch/examples/

Vanishing & exploding gradients

Limitations of temporally deep nets



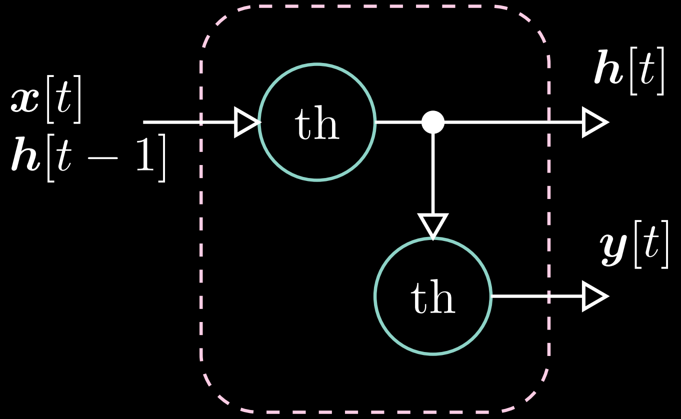
Graves (2012) Supervised sequence labelling



Graves (2012) Supervised sequence labelling

Long Short-Term Memory

Gated RNN



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

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

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

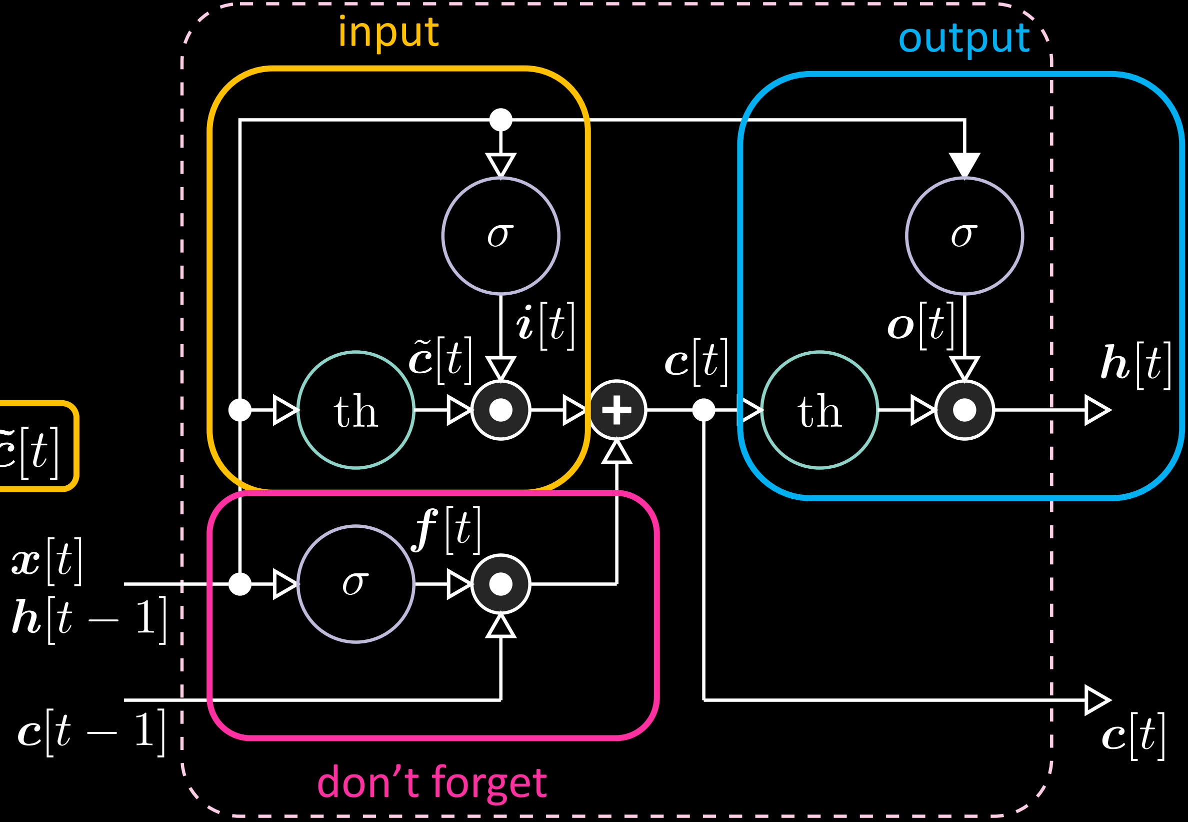
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix} \boldsymbol{x}[t] \\ \boldsymbol{h}[t-1] \end{bmatrix} + \boldsymbol{b}_f\right)$$

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

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

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

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



Controlling the $\boxed{\text{output}}$ - OFF

Saturated sigmoid: green = 1, red = 0

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

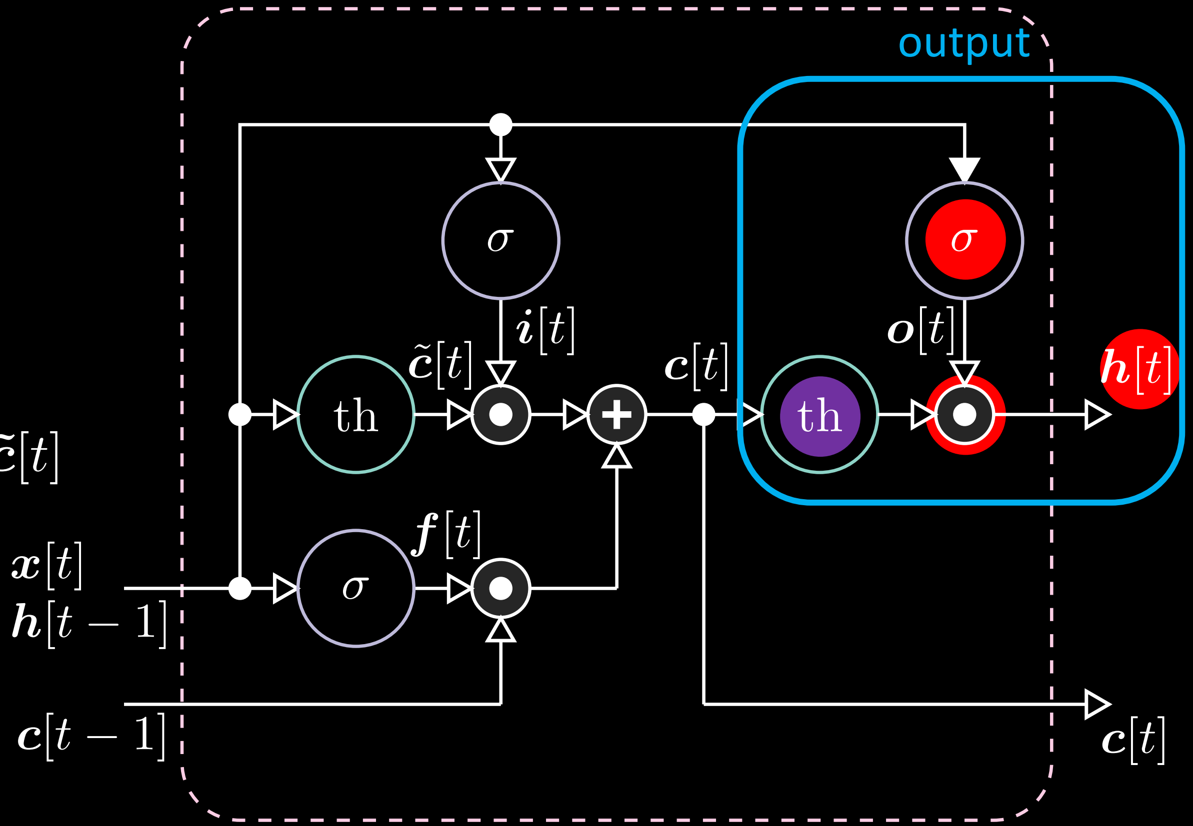
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix}\boldsymbol{x}[t] \\ \boldsymbol{h}[t-1]\end{bmatrix} + \boldsymbol{b}_f\right)$$

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

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

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

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



Controlling the $\boxed{\text{output}}$ - ON

Saturated sigmoid: green = 1, red = 0

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

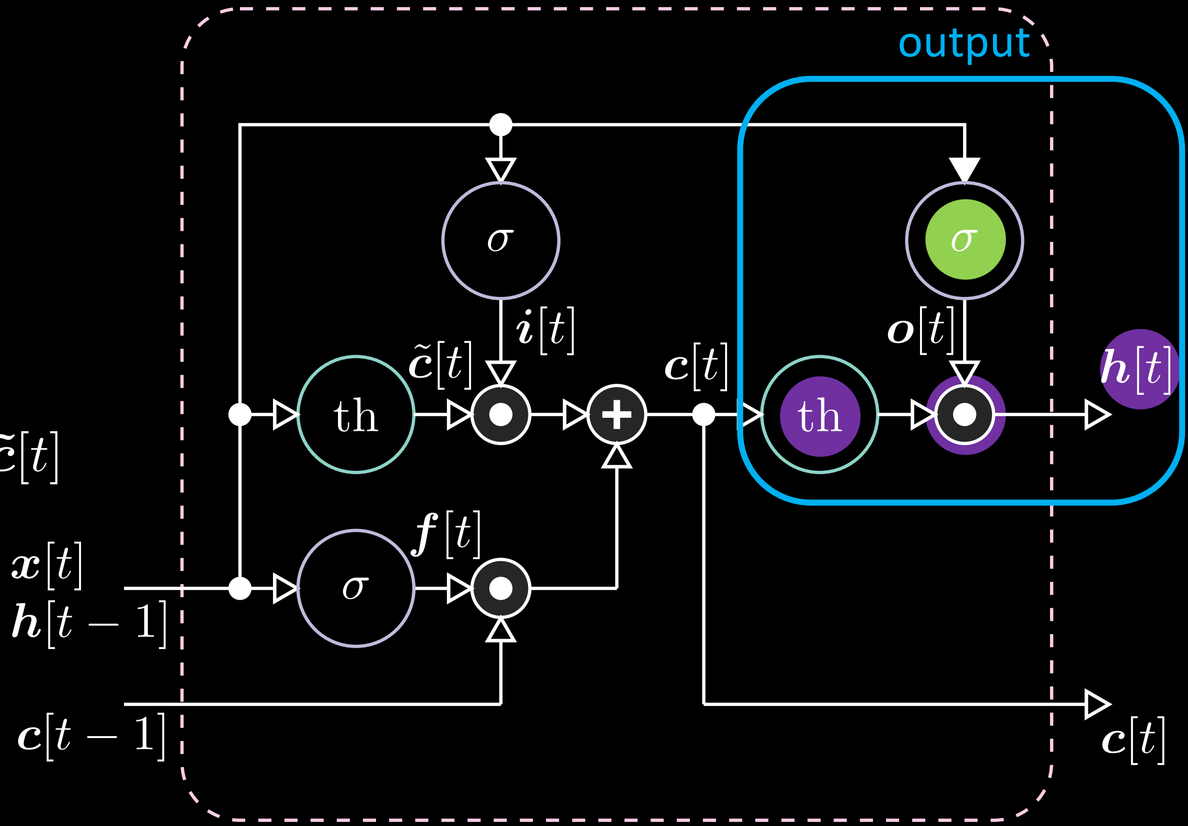
$$\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix}\boldsymbol{x}[t]\\ \boldsymbol{h}[t-1]\end{bmatrix} + \boldsymbol{b}_f\right)$$

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

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

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

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



Controlling the memory - reset

Saturated sigmoid = 1

= 0

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

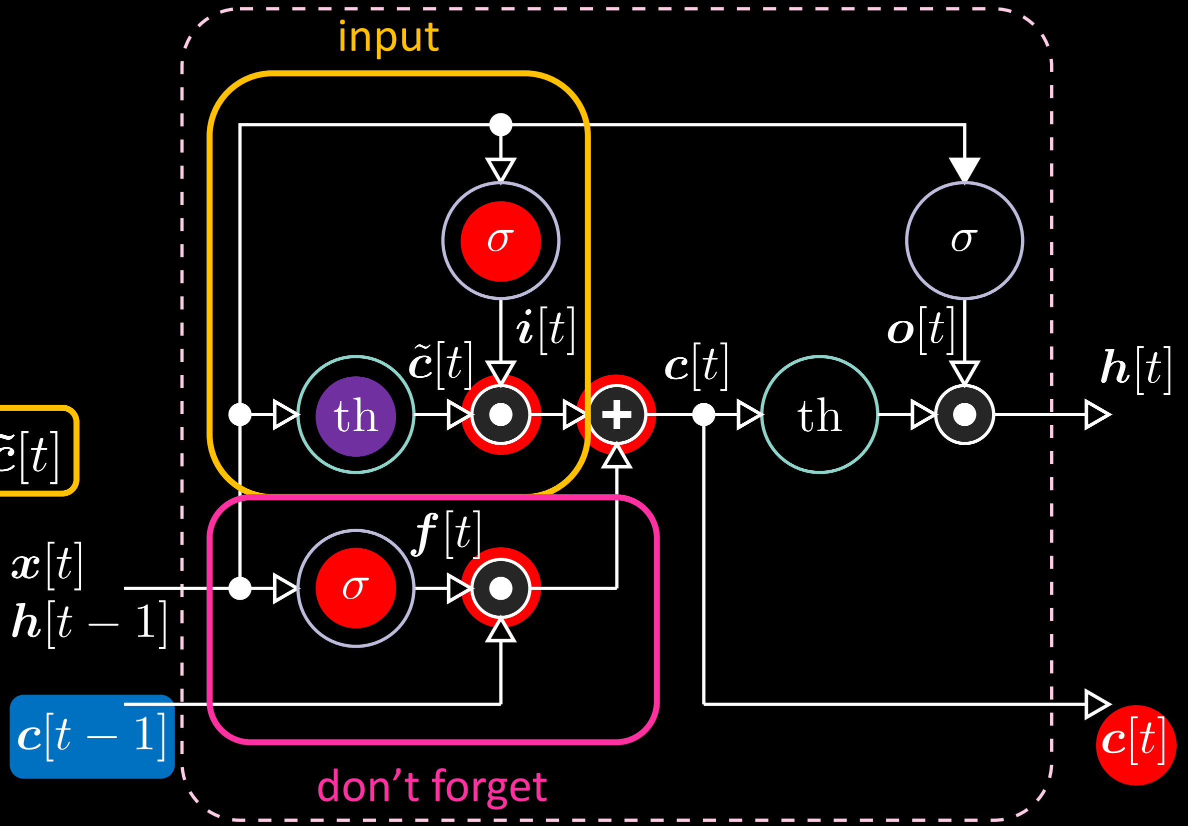
$$\boxed{\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix}\boldsymbol{x}[t]\\ \boldsymbol{h}[t-1]\end{bmatrix} + \boldsymbol{b}_f\right)}$$

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

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

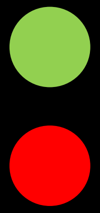
$$\boldsymbol{c}[t] = \boxed{\boldsymbol{f}[t] \odot \boldsymbol{c}[t-1]} + \boxed{\boldsymbol{i}[t] \odot \tilde{\boldsymbol{c}}[t]}$$

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



Controlling the memory - keep

Saturated sigmoid



= 1

= 0

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

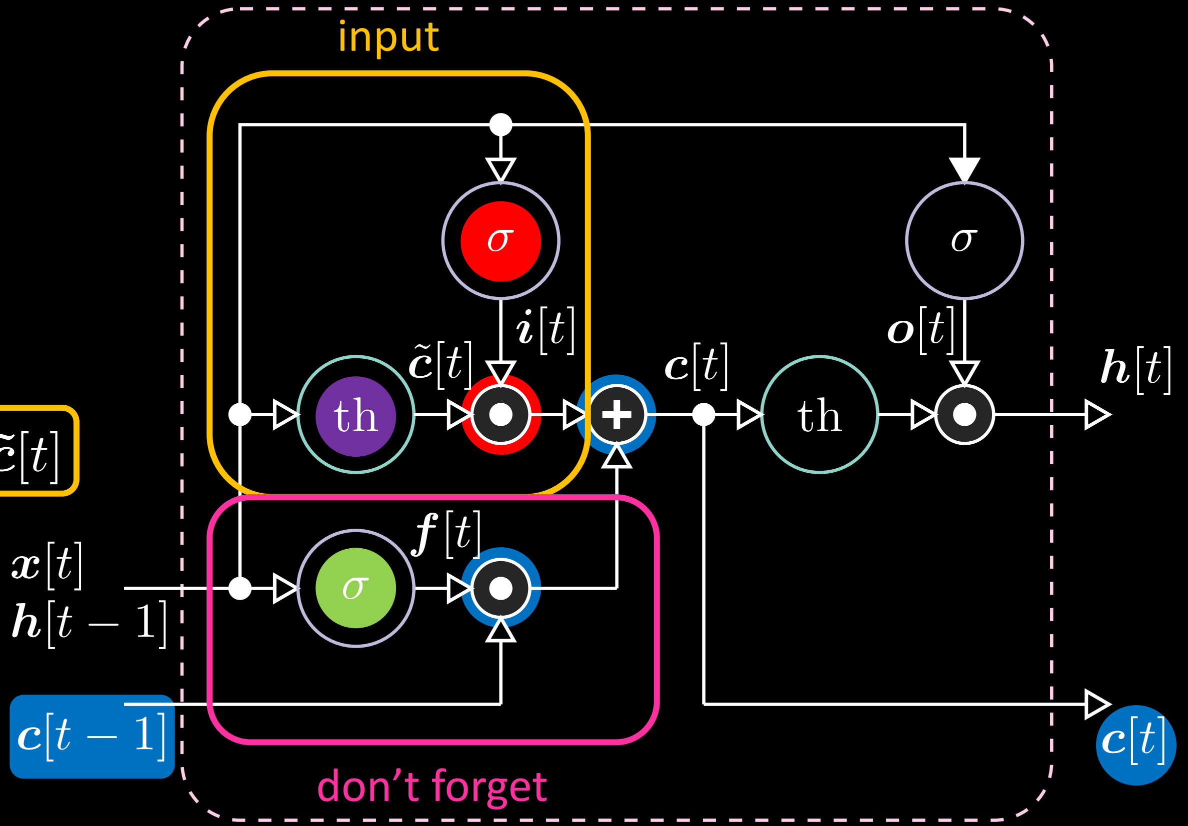
$$\boxed{\boldsymbol{f}[t] = \sigma\left(\boldsymbol{W}_f \begin{bmatrix}\boldsymbol{x}[t] \\ \boldsymbol{h}[t-1]\end{bmatrix} + \boldsymbol{b}_f\right)}$$

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

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

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

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



Controlling the memory - write

Saturated sigmoid = 1

= 0

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

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

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

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

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

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