深度学习讨论班

第四节
Recurrent Neural Networks
(递归神经网络)

黄雷

2016-12-27

上一节主要内容

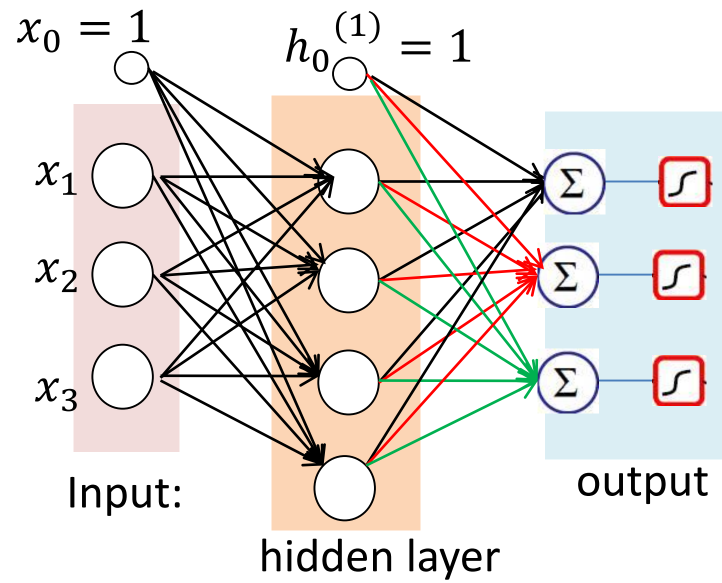
- Modeling of CNN
  - Module-wise architecture模块化结构
- Convolutional layer (module)
  - Convolution in general
  - Filters
  - Convolution module
- Pooling layer (module)

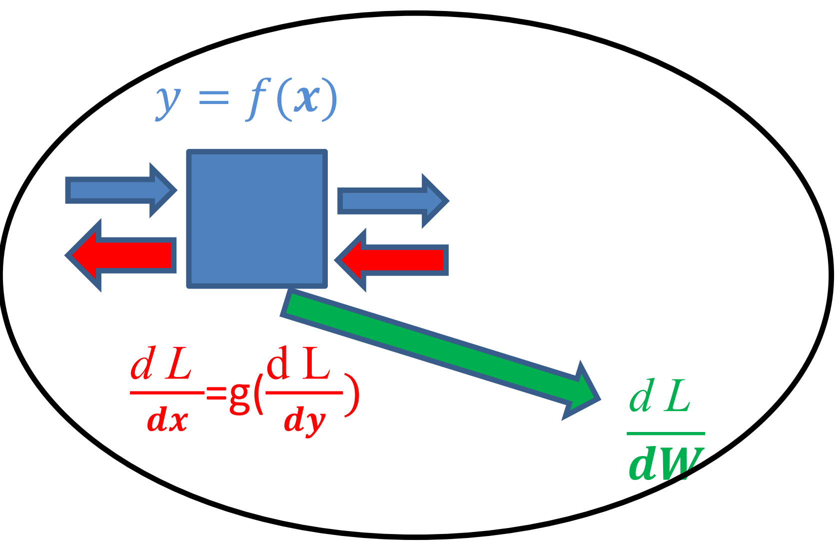
Module-wise architecture

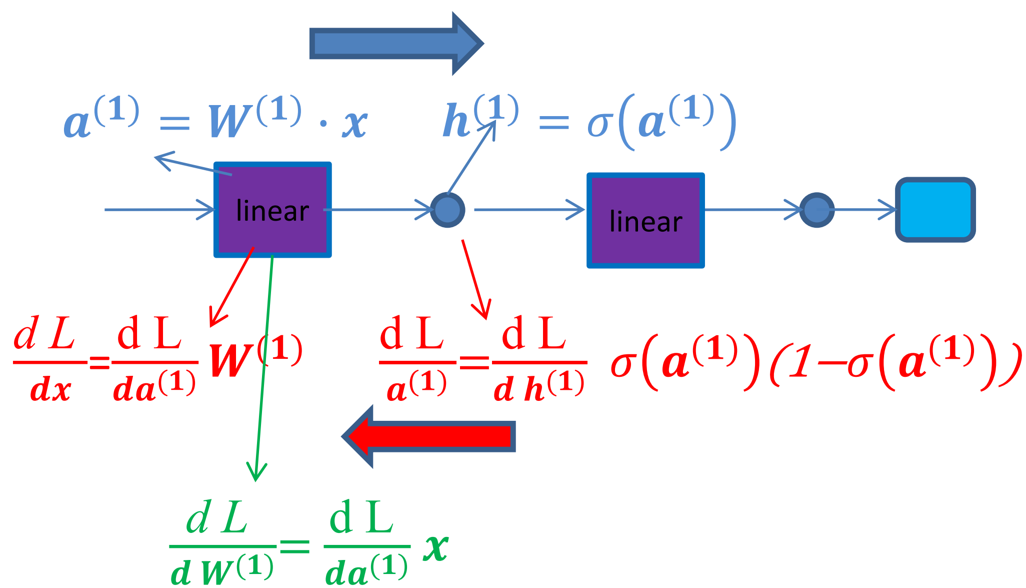
- Torch 平台

- **Training per iteration:**

```
-- forward
 outputs = model:forward(X)
 loss = criterion:forward(outputs, Y)
-- backward
 dloss_doutput = criterion:backward(outputs, Y)
 model:backward(X, dloss_doutput)
```

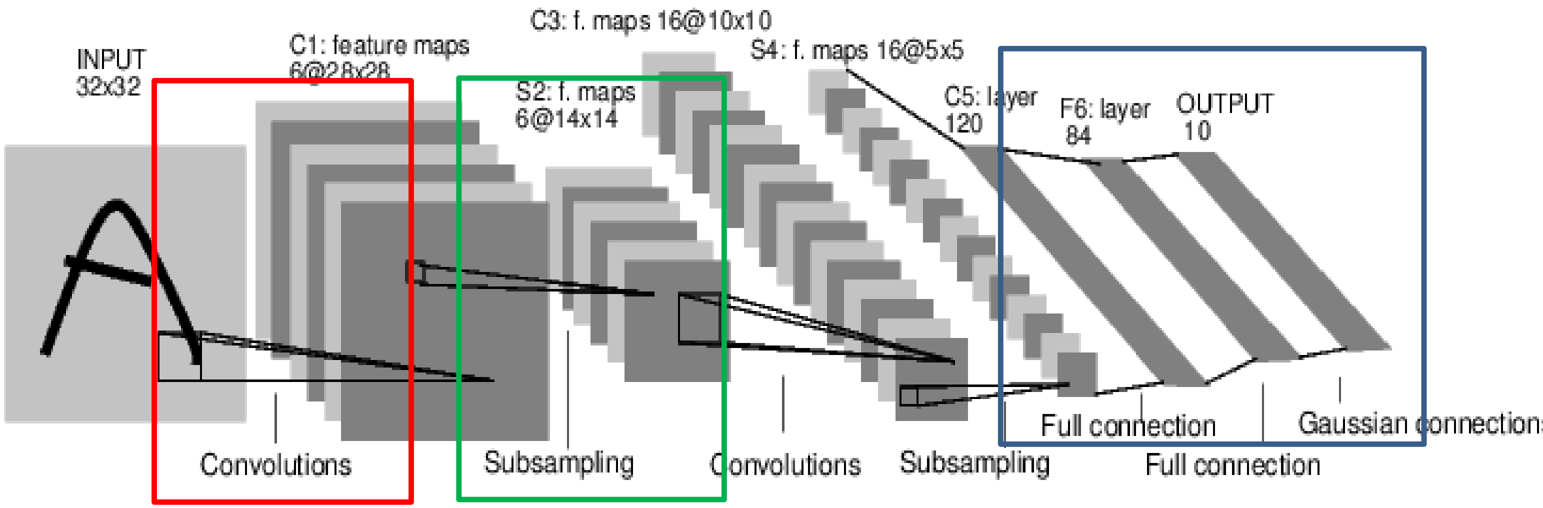






Convolution Neural Network

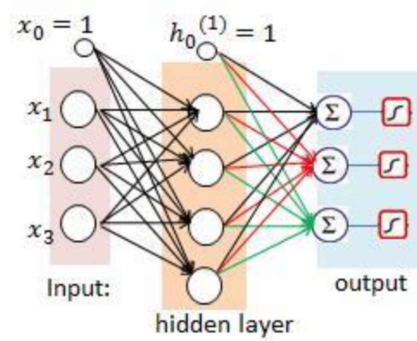
- Lenet-5



Convolution (卷积)

Pooling (池化)

全连接层



Convolution

- 离散空间卷积：

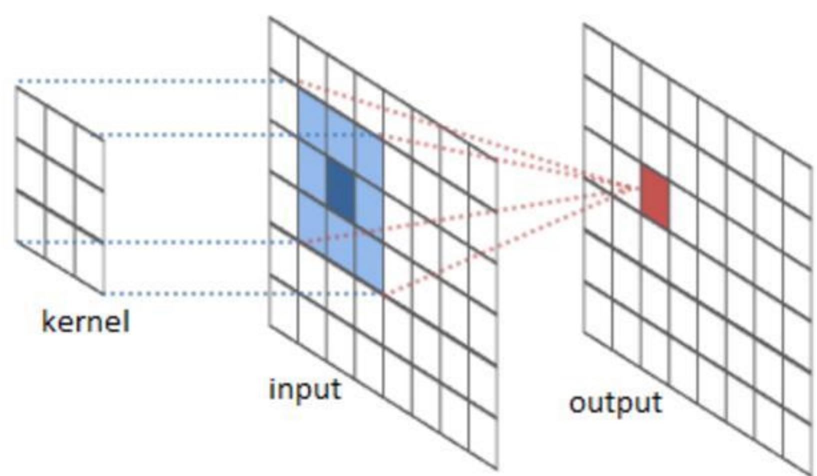
$$y(n) = x(n) * w(n) = \sum_{i=-\infty}^{i=+\infty} x(i)w(n-i)$$

- 连续空间的卷积:

$$y(t) = x(t) * h(t) = \int_{-\infty}^{+\infty} x(s)h(t-s)\,ds$$

- 图像卷积是二维离散卷积

$$g(i,j) = \sum_{k,l} f(k,l)\,w(i-k,j-l)$$

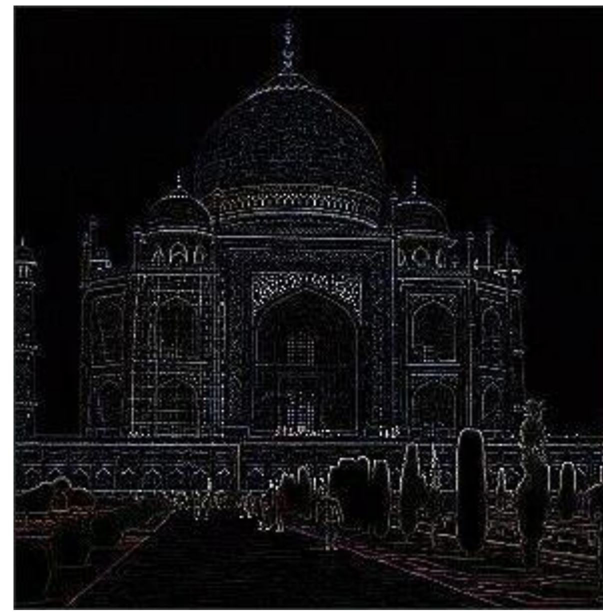


Practice with linear filters

Original

0	1	0
1	-4	1
0	1	0

Filter



Output Image

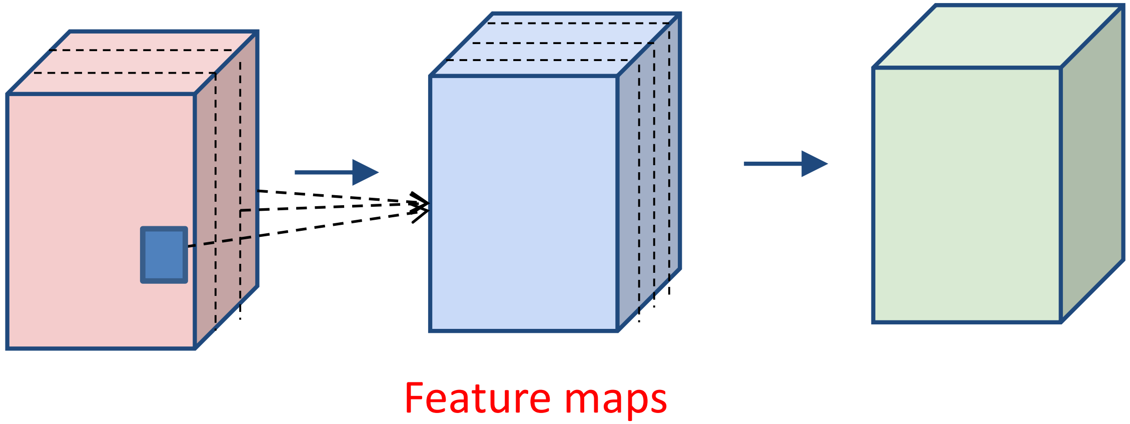
Edge detect （边缘检测）

Convolution Layer (卷积层)

Input: X∈ $R^{d_{in} \times h \times w}$

weight: W∈ $R^{d_{out} \times d_{in} \times F_h \times F_w}$

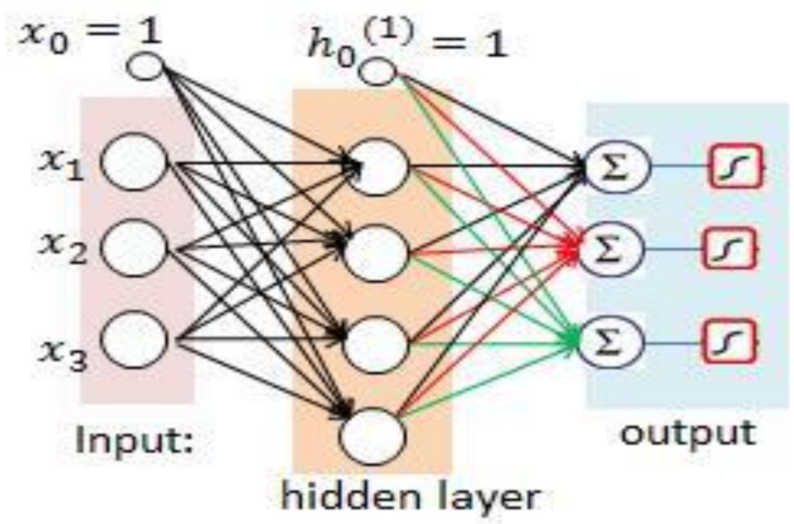
output: Y∈ $R^{d_{out} \times h \times w}$



Input: x∈ $R^{d_{in}}$

weight: W∈ $R^{d_{out} \times d_{in}}$

output: y∈ $R^{d_{out}}$

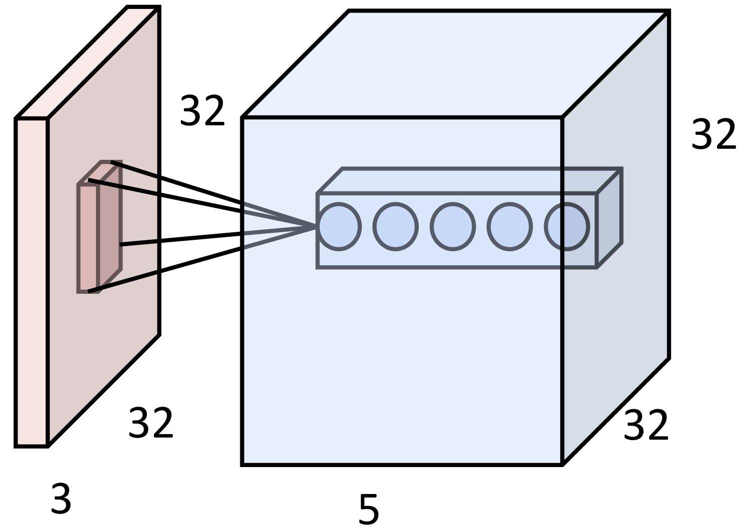


Forward (前向过程)

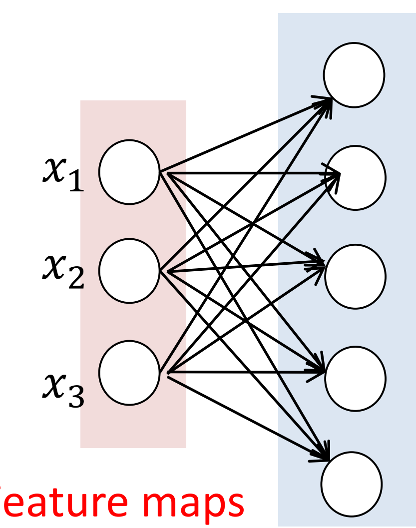
Input: X∈ $\boldsymbol{R^{d_{in}\times h\times w}}$

weight: W∈ $\boldsymbol{R^{d_{out}\times d_{in}\times F_h\times F_w}}$

output: Y∈ $\boldsymbol{R^{d_{out}\times h\times w}}$

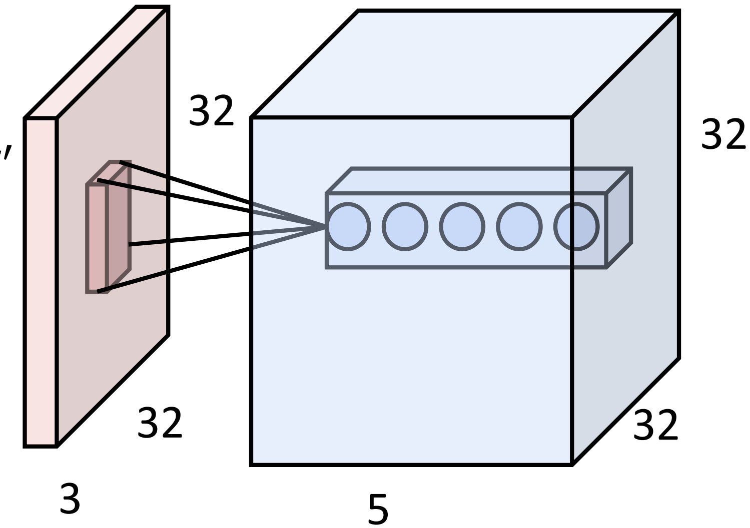


$$y_{f',i',j'} = \sum_{f=1}^{d_{in}} \sum_{i=-\frac{F_h}{2}}^{\frac{F_h}{2}} \sum_{j=-\frac{F_w}{2}}^{\frac{F_w}{2}} x_{f,i'+i-1,j'+j-1}\, w_{f',f,i,j} + b_{f'}$$



Back-propagation

$$y_{f',i',j'} = \sum_{f=1}^{d_{in}} \sum_{i=-\frac{F_h}{2}}^{\frac{F_h}{2}} \sum_{j=-\frac{F_w}{2}}^{\frac{F_w}{2}} x_{f,i'+i-1,j'+j-1} \, w_{f',f,i,j} + b_{f'}$$



$$\frac{dL}{dx_{f,i,j}} = \sum_{f'=1}^{d_{out}} \sum_{i'=1}^{h} \sum_{j'=1}^{w} \frac{dL}{dy_{f',i',j'}} \frac{dy_{f',i',j'}}{dx_{f,i,j}}$$

$$\sum_{f'=1}^{d_{out}} \sum_{i'=1}^{h} \sum_{j'=1}^{w} \frac{dL}{dy_{f',i',j'}} w_{f',f,i-i'+1,j-j'+1}$$

$$\frac{dL}{w_{f',f,i,j}} = \sum_{f'=1}^{d_{out}} \sum_{i'=1}^{h} \sum_{j'=1}^{w} \frac{dL}{dy_{f',i',j'}} \frac{dy_{f',i',j'}}{w_{f',f,i,j}}$$

$$\sum_{f'=1}^{d_{out}} \sum_{i'=1}^{h} \sum_{j'=1}^{w} \frac{dL}{dy_{f',i',j'}} x_{f,i'+i-1,j'+j-1}$$

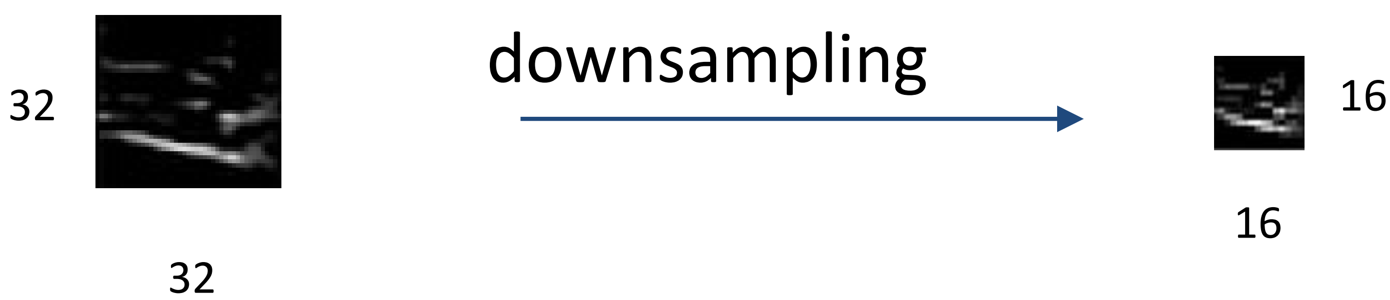
Input: X∈ $R^{d_{in} \times h \times w}$

weight: W∈ $R^{d_{out} \times d_{in} \times F_h \times F_w}$

output: Y∈ $R^{d_{out} \times h \times w}$

POOLING Layer

- In ConvNet architectures, **Conv** layers are often followed by **Pooling** layers
  - makes the representations smaller and more manageable without losing too much information.
  - Invariant in region.

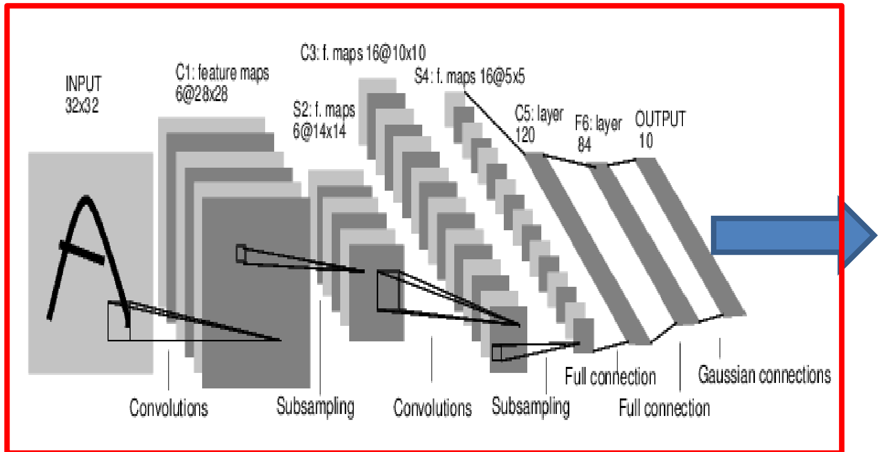


Source: Stanford CS231n, Andrej Karpathy & Fei-Fei Li

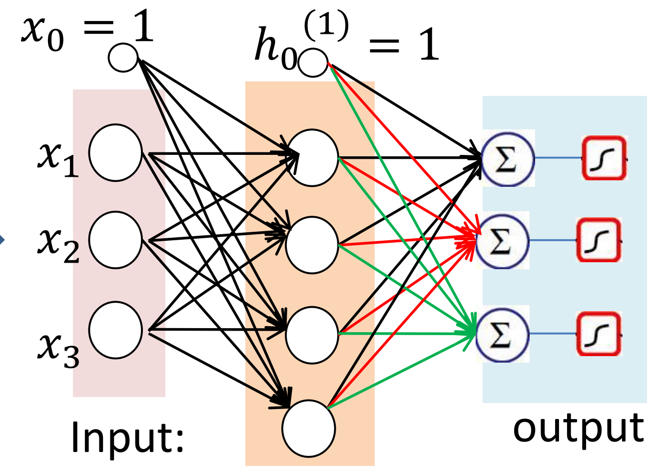
outline

- Recurrent Neural Network
  - Modeling
  - Training
- Long Short Term Memory (LSTM)
  - Motivation
  - Modeling
- Application
  - Generate article

Classification: MLP and CNN

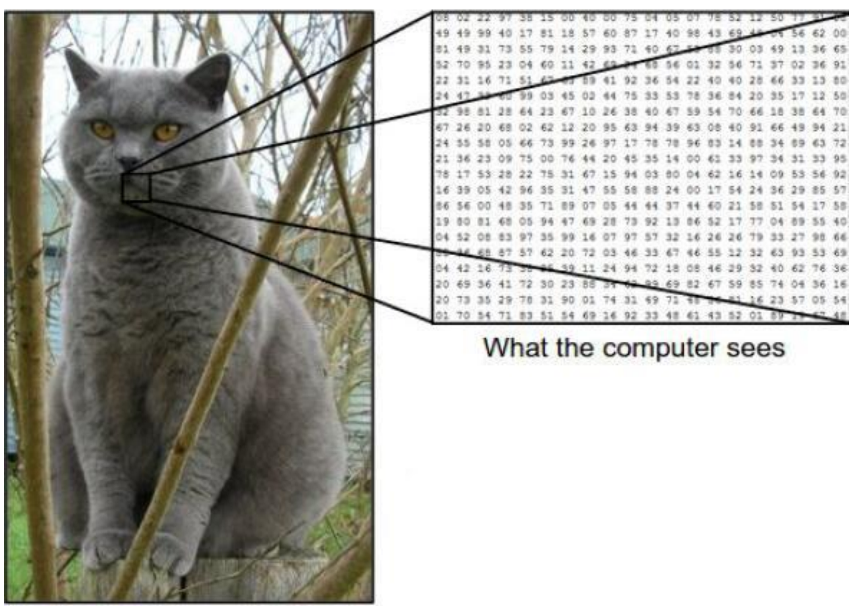


Convolution (卷积)



$(1, 0, 0)^T$

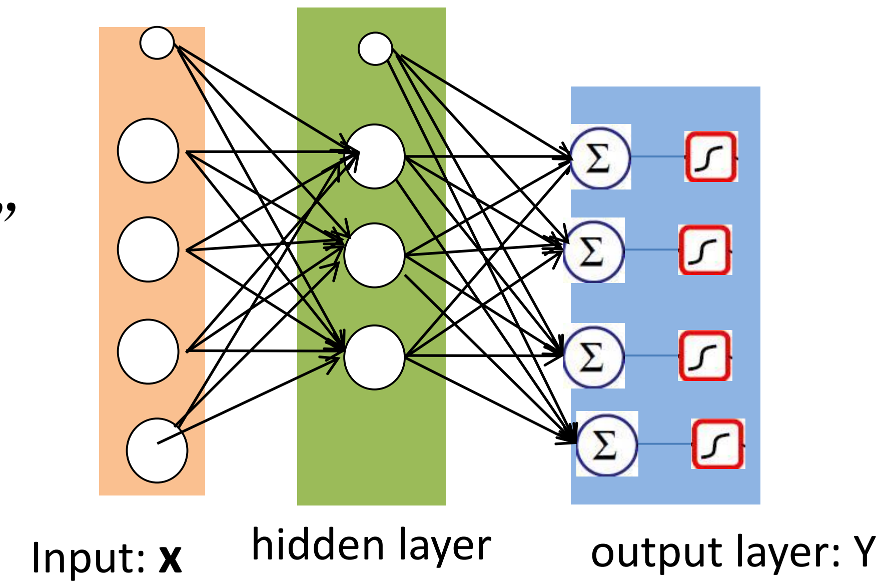
(x_1, x_2, x_3)



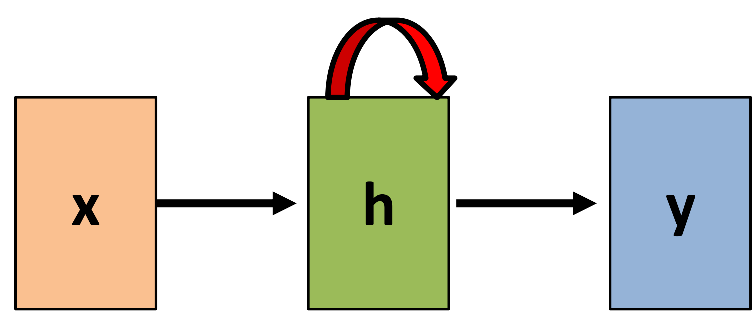
I go to cinema, and
I book a ticket

One example-modeling: motivation

- **Task**: **Character-level language model**
  - **example**
    - Vocabulary [h,e,l,o]
    - Training sequence “hello”

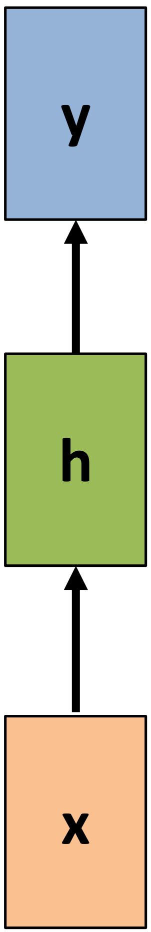


- Data representation：
  - X: {h, e, l, l}
  - Y: {e, l , l, o}



Modeling

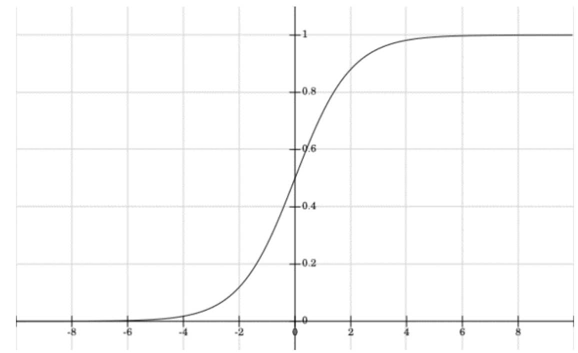
- MLP → RNN



$$\mathbf{y} = \boldsymbol{W}_{yh}\boldsymbol{h}$$

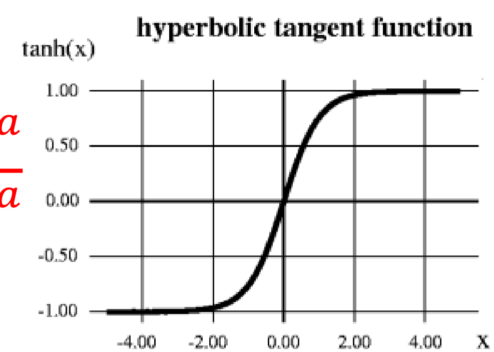
$$\boldsymbol{y} = \sigma(\boldsymbol{W}_{yh}\boldsymbol{h})$$

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$



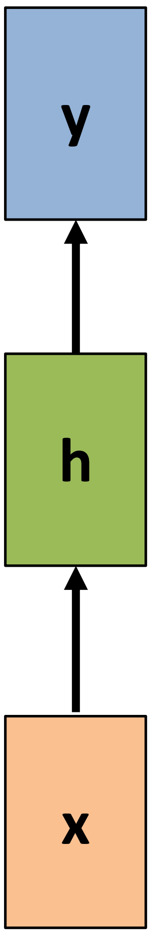
$$\boldsymbol{h} = \sigma(\boldsymbol{W}_{hx}\boldsymbol{x})$$

$$\tanh(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}}$$



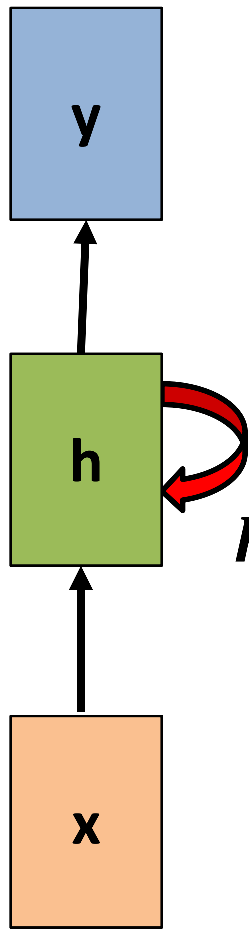
Modeling

- MLP → RNN



$$\mathbf{y} = \boldsymbol{W}_{\boldsymbol{yh}}\boldsymbol{h}$$

$$\boldsymbol{h} = tanh(\boldsymbol{W}_{\boldsymbol{hx}}\boldsymbol{x})$$

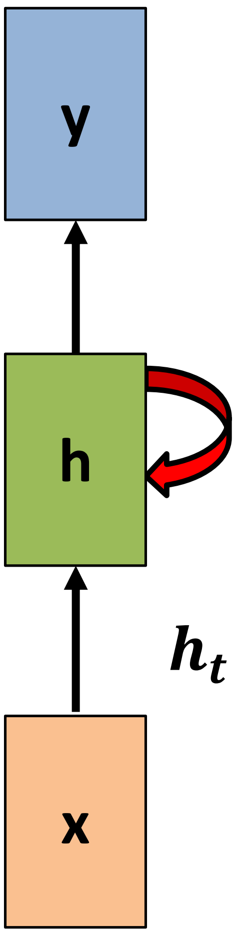


$$\boldsymbol{y}_{\boldsymbol{t}} = \boldsymbol{W}_{\boldsymbol{yh}}\boldsymbol{h}_{\boldsymbol{t}}$$

$$\boldsymbol{h}_{\boldsymbol{t}} = tanh(\boldsymbol{W}_{\boldsymbol{hx}}\boldsymbol{x} + \boldsymbol{W}_{\boldsymbol{hh}}\boldsymbol{h}_{\boldsymbol{t-1}})$$

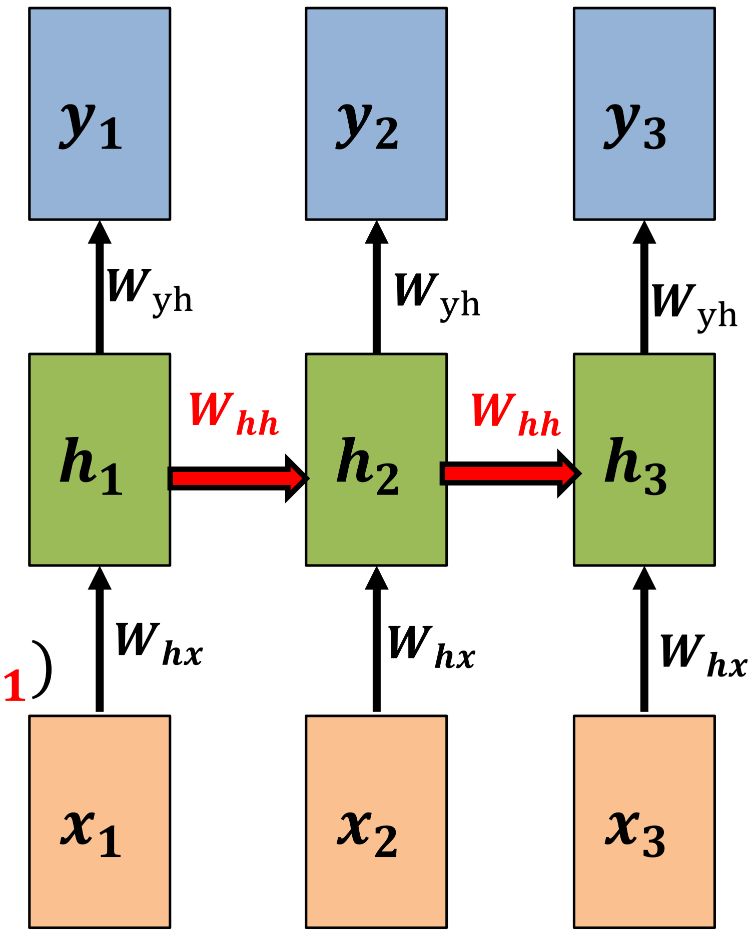
Modeling

- RNN-unrolling(展开)



$$y_t = W_{yh} h_t$$

$$h_t = tanh(W_{hx} x + W_{hh} h_{t-1})$$



Example

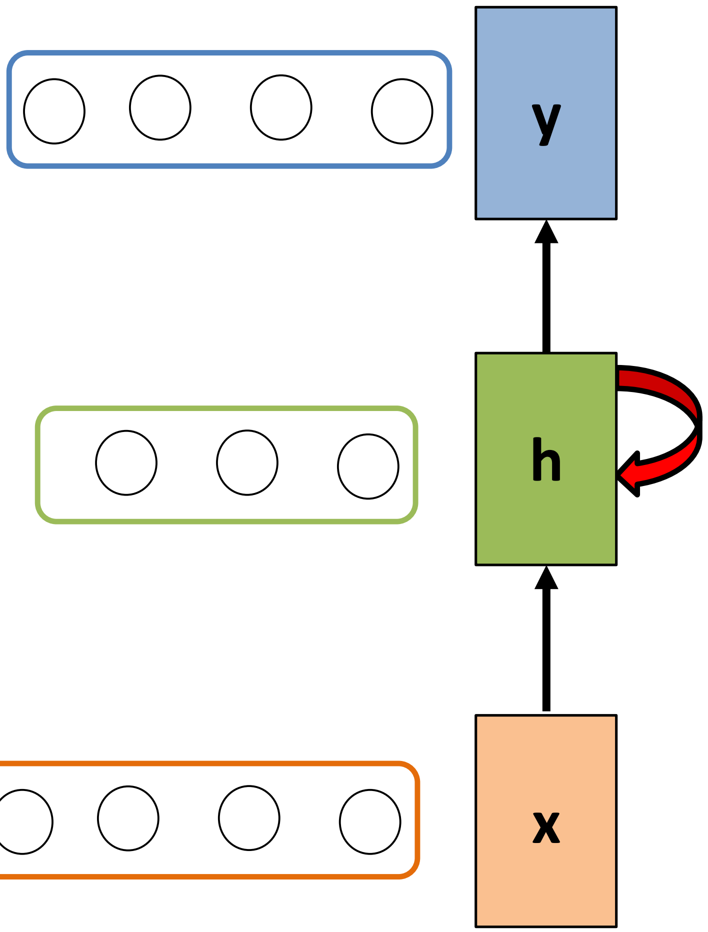
- Character-level language model

  – Training sequence:
    - “Hello”

  – Presentation:

  X: {h, e, l, l}

  Y: {e, l , l, o}



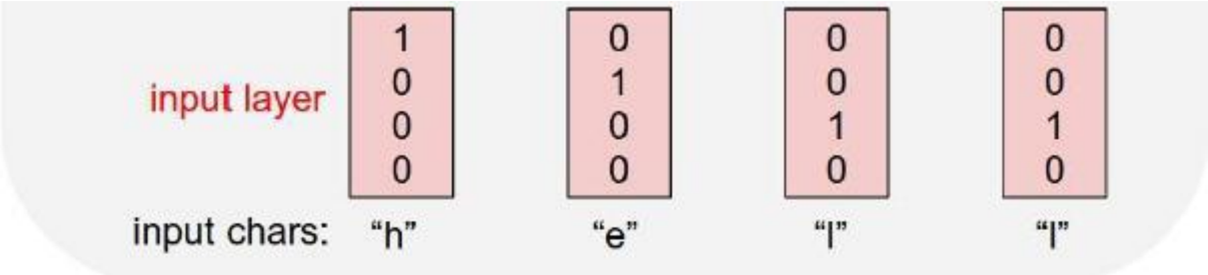
One-hot aka
one-of-K encoding

Example

- Examples:
  - Training sequence:
    - “Hello”
  - Presentation:

  X: {h, e, l, l}

  Y: {e, l , l, o}

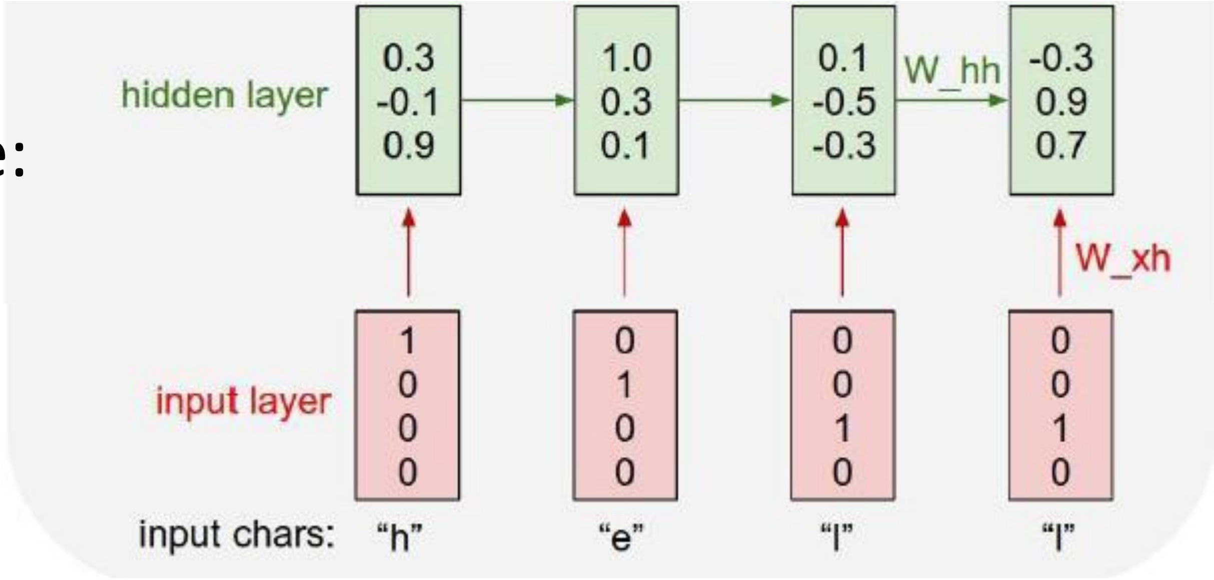


Example

- Examples:
  - Training sequence:
    - “Hello”
  - Presentation:

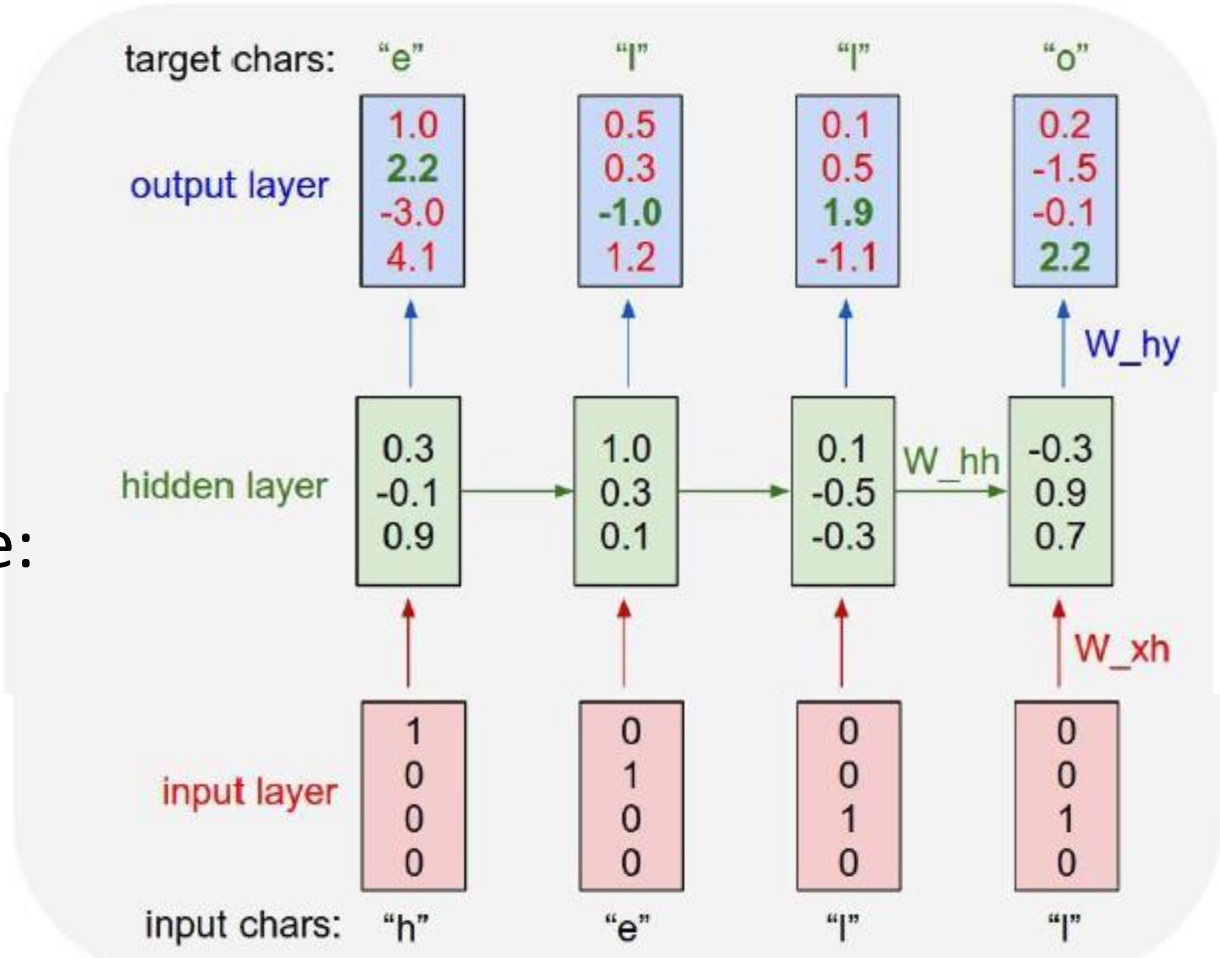
  X: {h, e, l, l}

  Y: {e, l , l, o}



Example

- Examples:



  - Training sequence:
    - "Hello"
  - Presentation:

X: {h, e, l, l}

Y: {e, l , l, o}

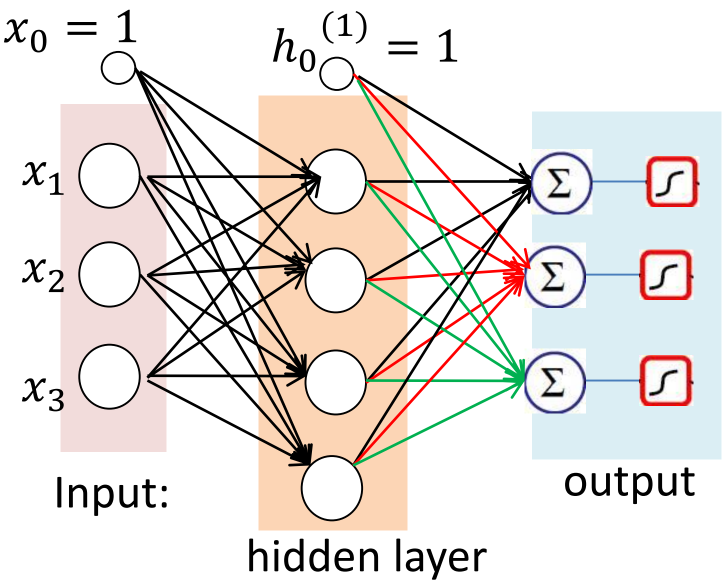
Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

outline

- Recurrent Neural Network
  - Modeling
  - Training
- Long Short Term Memory (LSTM)
  - Motivation
  - Modeling
- Application
  - Generate article

Neural Network

- Training Algorithm



$(1, 0, 0)^T$

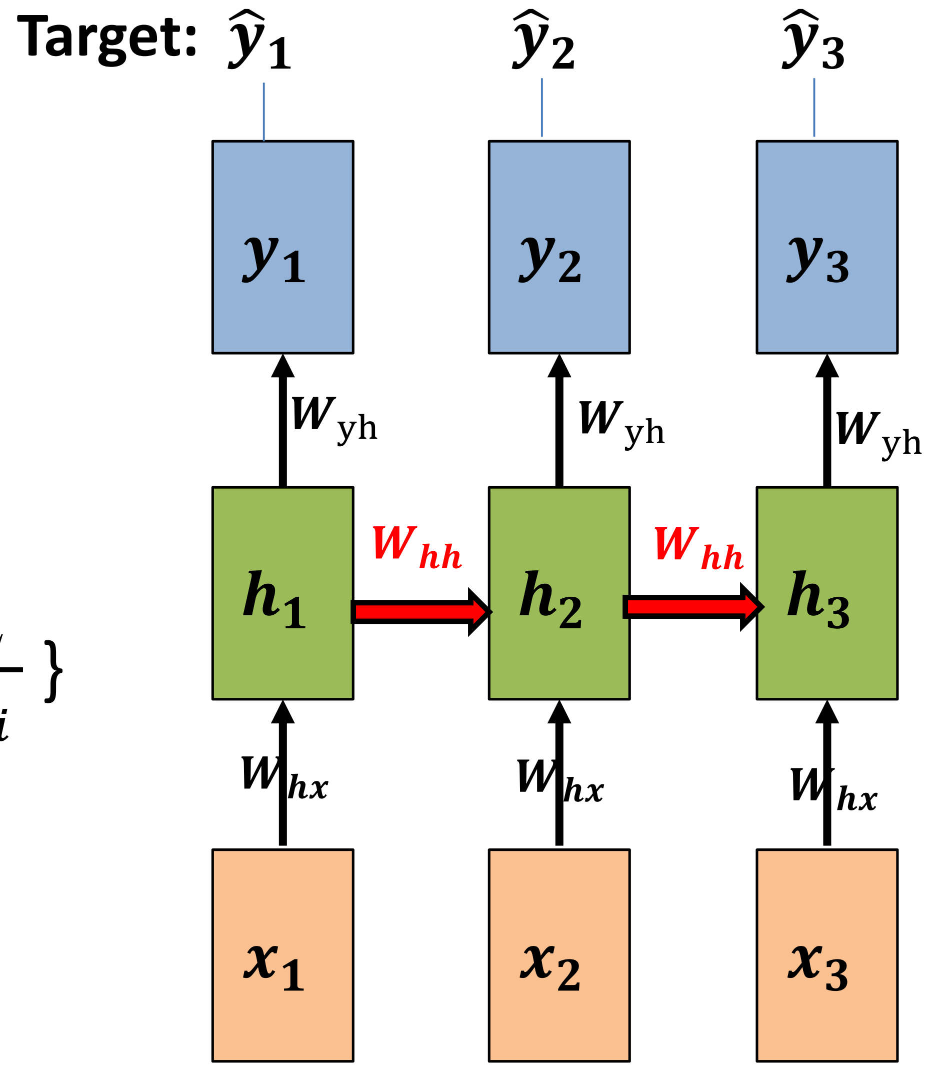
- 0.初始化权重 $\boldsymbol{W}^{(0)}$
- 1. 前向过程：
  - 1.1根据输入$\boldsymbol{x}$，计算输出值 $\boldsymbol{y}$
  - 1.2.计算损失函数值L($\mathbf{y}, \hat{\boldsymbol{y}}$)。
- 2.后向传播
  - 计算$\frac{\mathrm{d\,L}}{\mathbf{y}}$
  - 后向传播直到计算$\frac{\mathrm{d\,L}}{\mathbf{x}}$
- 3.计算梯度$\frac{d\,L}{\boldsymbol{d\,W}}$
- 4.更新梯度

$$\boldsymbol{W}^{(t+1)} = \boldsymbol{W}^{(t)} - \eta \frac{d\,\mathrm{L}}{\boldsymbol{d\,W}^{(t)}}$$

Training

- learning
  - Sequence length=3
- Back-propagation

$$-\{\frac{\mathrm{d\,L}}{d\,y_i} \Rightarrow \frac{\mathrm{d\,L}}{d\,h_i} \Rightarrow \frac{\mathrm{d\,L}}{d\,x_i}\}$$

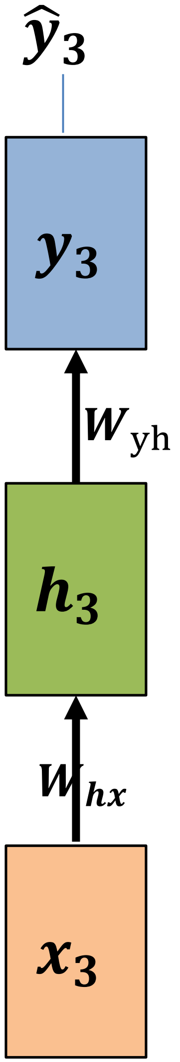


Training

- Back-propagation

Target:

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_3} = \frac{\mathrm{d\,L}}{\boldsymbol{d}\,\boldsymbol{y}_3}\,\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}$$



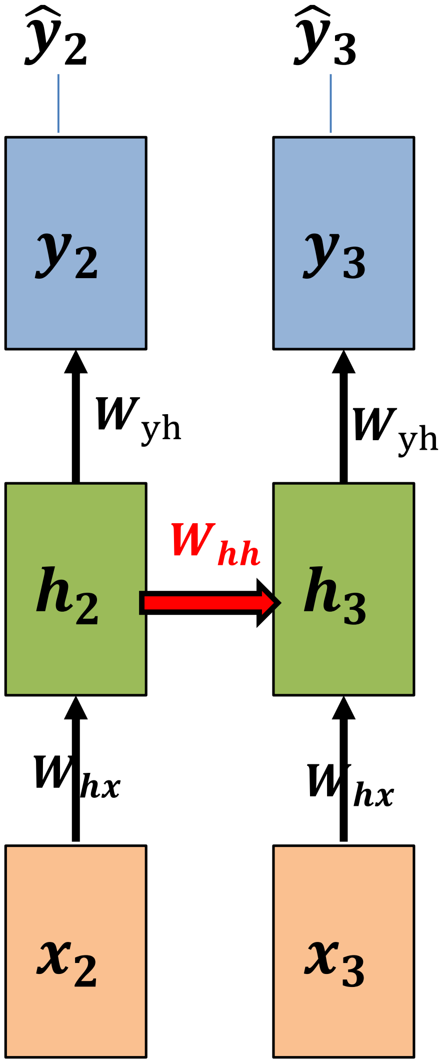
Training

- Back-propagation

Target:

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_3} = \frac{\mathrm{d\,L}}{\boldsymbol{d}\,\boldsymbol{y}_3}\,\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}$$

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_2} = \frac{\mathrm{d\,L}}{\boldsymbol{d}\,\boldsymbol{y}_2}\,\frac{\mathrm{d}\boldsymbol{y}_2}{d\boldsymbol{h}_2} + \frac{\mathrm{d\,L}}{\boldsymbol{d}\,\boldsymbol{y}_3}\,\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}\,\frac{\mathrm{d}\boldsymbol{h}_3}{d\boldsymbol{h}_2}$$



Training

- Back-propagation

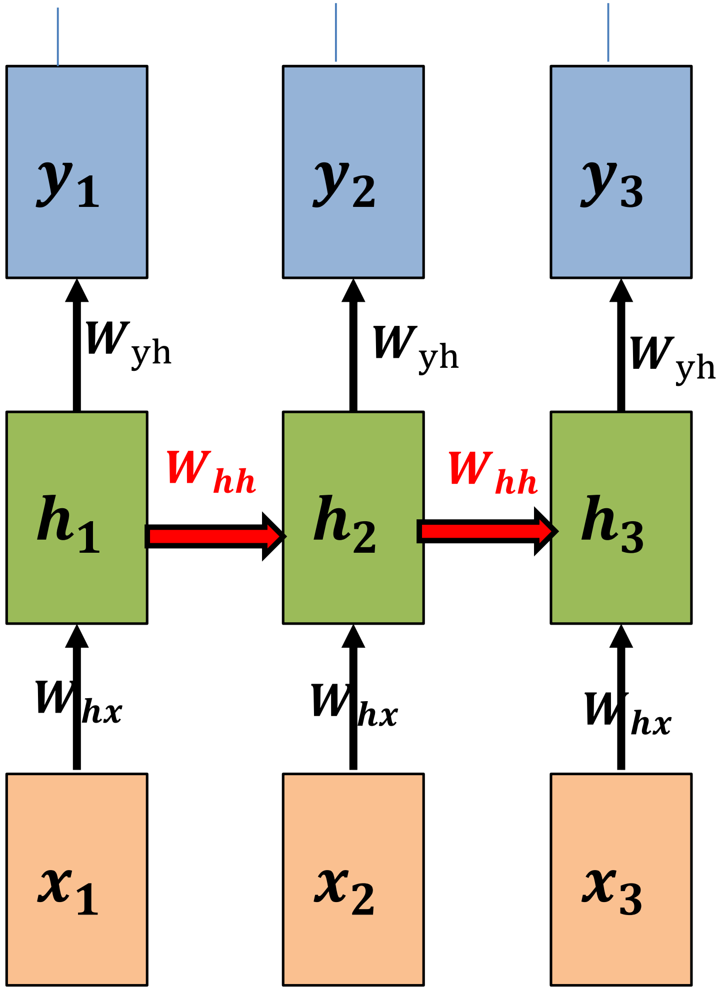
$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_3} = \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_3}\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}$$

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_2} = \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_2}\frac{\mathrm{d}\boldsymbol{y}_2}{d\boldsymbol{h}_2} + \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_3}\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}\frac{\mathrm{d}\boldsymbol{h}_3}{d\boldsymbol{h}_2}$$

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_1} = \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_1}\frac{\mathrm{d}\boldsymbol{y}_1}{d\boldsymbol{h}_1} + \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_2}\frac{\mathrm{d}\boldsymbol{y}_2}{d\boldsymbol{h}_2}\frac{\mathrm{d}\boldsymbol{h}_2}{d\boldsymbol{h}_1} + \frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_3}\frac{\mathrm{d}\boldsymbol{y}_3}{d\boldsymbol{h}_3}\frac{\mathrm{d}\boldsymbol{h}_3}{d\boldsymbol{h}_2}\frac{\mathrm{d}\boldsymbol{h}_2}{d\boldsymbol{h}_1}$$

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_t} = \sum_{s=t}^{T=3}\frac{\mathrm{d\,L}}{\boldsymbol{d\,y}_s}\frac{\mathrm{d}\boldsymbol{y}_s}{d\boldsymbol{h}_s}\frac{\mathrm{d}\boldsymbol{h}_s}{d\boldsymbol{h}_t}$$

Target: $\hat{\boldsymbol{y}}_1$ $\hat{\boldsymbol{y}}_2$ $\hat{\boldsymbol{y}}_3$



Training

- Gradient r.t Weight

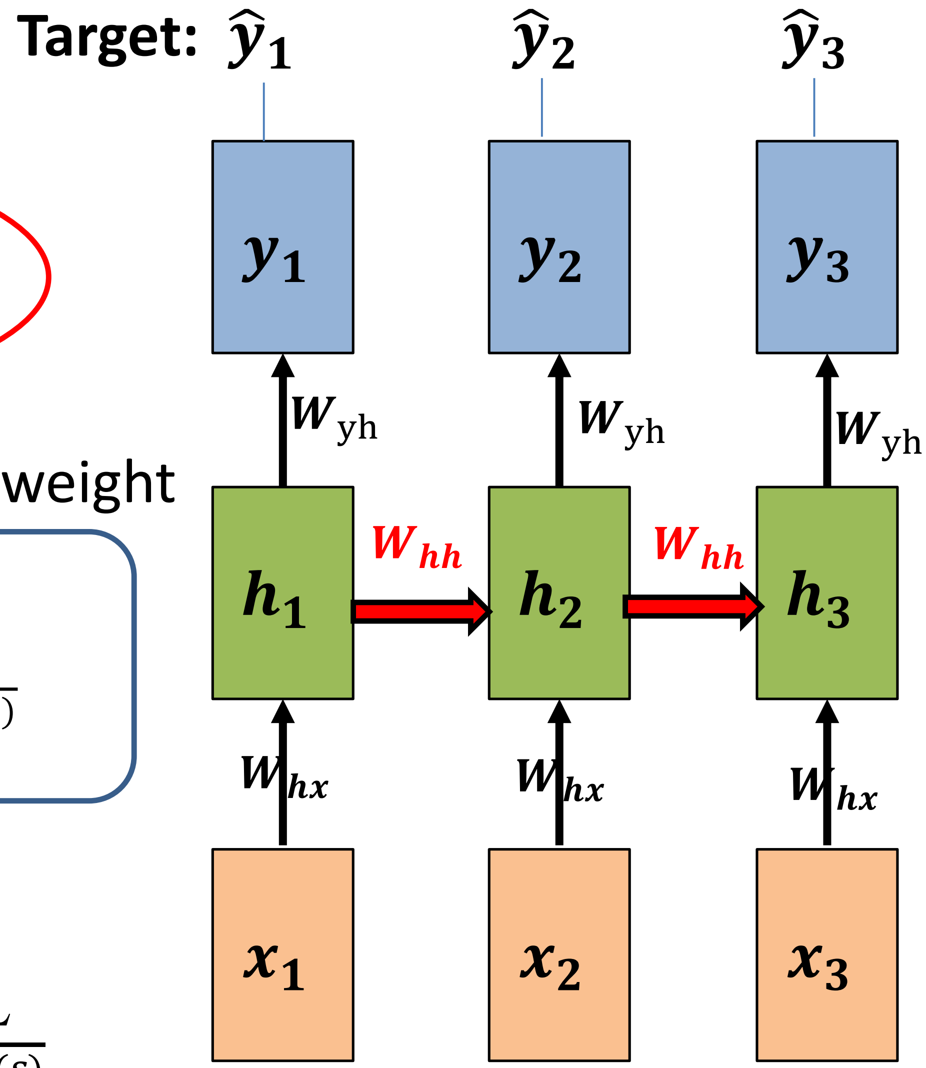
$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_t} = \sum_{s=t}^{T} \frac{\mathrm{d\,L}}{\boldsymbol{d}\;\boldsymbol{y}_s} \frac{\mathrm{d}\boldsymbol{y}_s}{d\boldsymbol{h}_s} \frac{\mathrm{d}\boldsymbol{h}_s}{d\boldsymbol{h}_t}$$

➢ Calculate gradient respect to weight

$$\frac{\mathrm{d\,L}}{d\mathbf{W}_{\mathrm{yh}}{}^{(\mathrm{i})}} \qquad \frac{\mathrm{d\,L}}{d\mathbf{W}_{\mathrm{hh}}{}^{(\mathrm{i})}} \qquad \frac{\mathrm{d\,L}}{d\mathbf{W}_{\mathrm{hx}}{}^{(\mathrm{i})}}$$

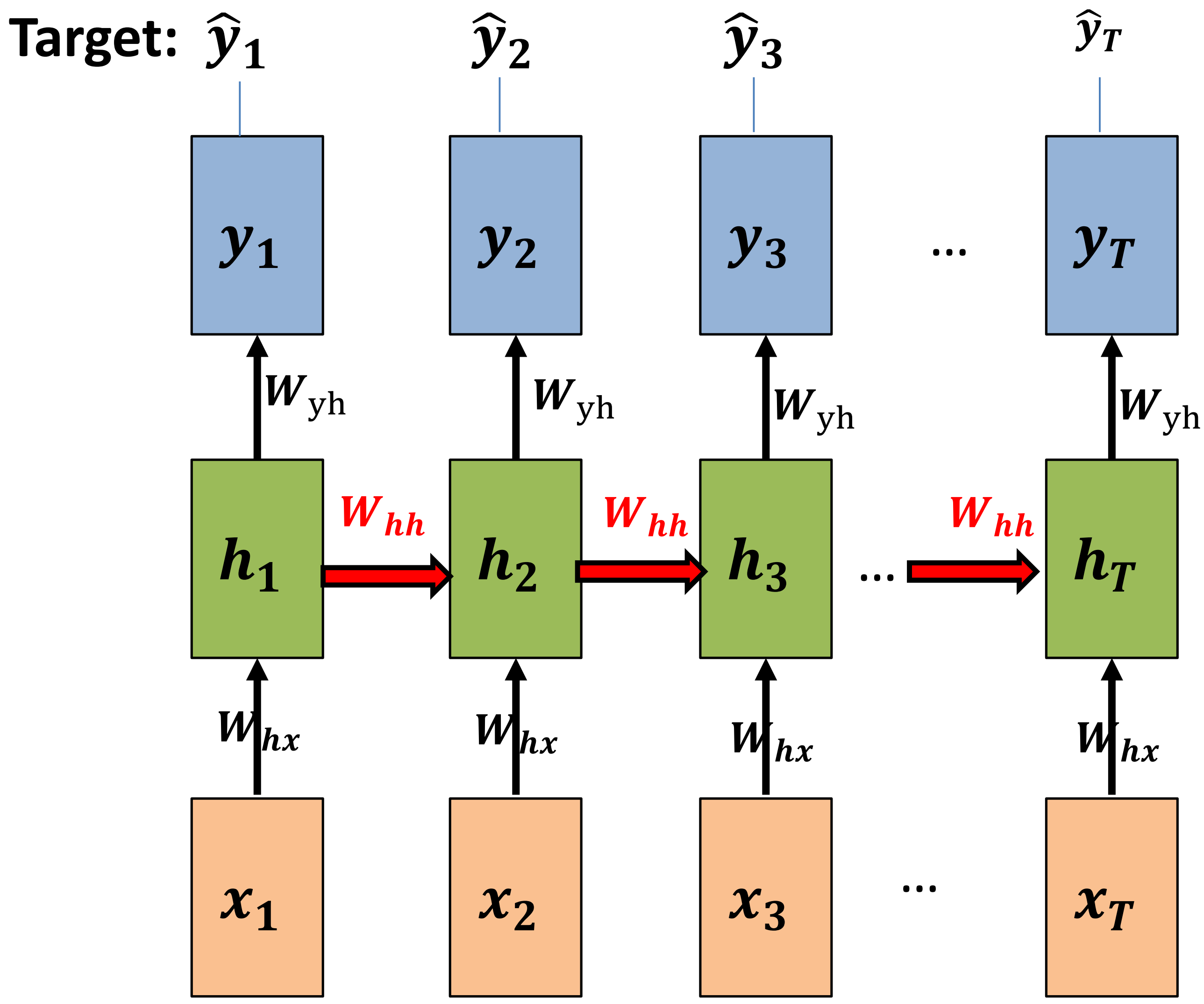
➢ Update weight

$$\boldsymbol{W}^{(t+1)} = \boldsymbol{W}^{(t)} - \eta \sum_{s=1}^{T} \frac{d\,\mathrm{L}}{\boldsymbol{d}\,\boldsymbol{W}^{(s)}}$$



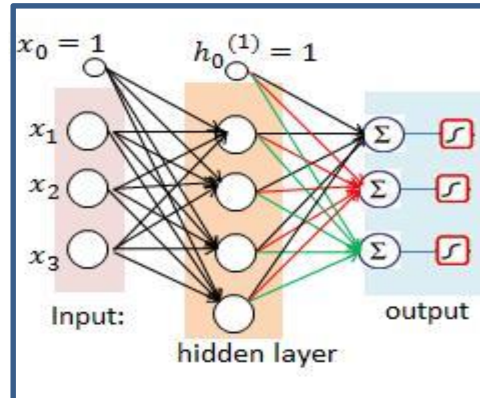
RNN

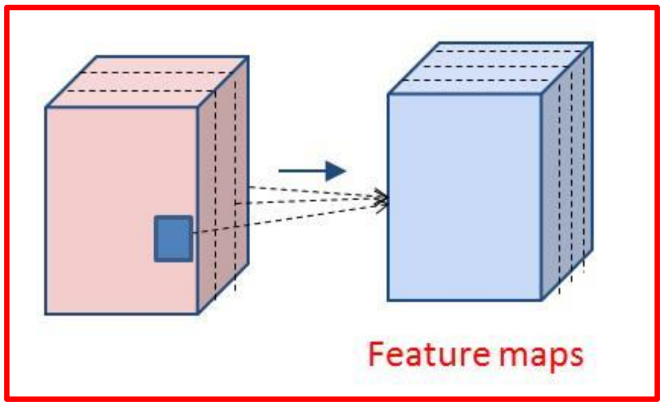
- longer
- deeper



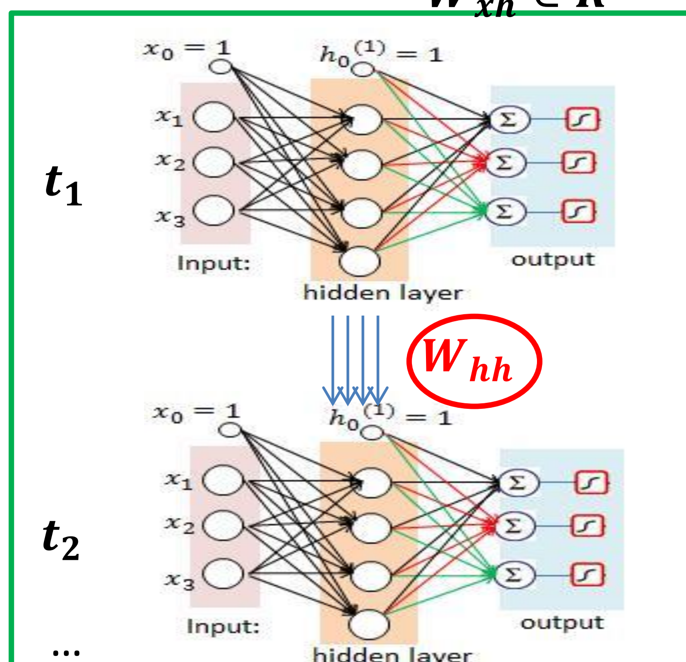
Relation: MLP, CNN and RNN

x$\in R^{d_{in}}$ $W \in R^{d_{out} \times d_{in}}$ y$\in R^{d_{out}}$





$W_{xh} \in R^{d_{out} \times d_{in}}$



X$\in R^{d_{in} \times h \times w}$

W$\in R^{d_{out} \times d_{in} \times F_h \times F_w}$

Y$\in R^{d_{out} \times h \times w}$

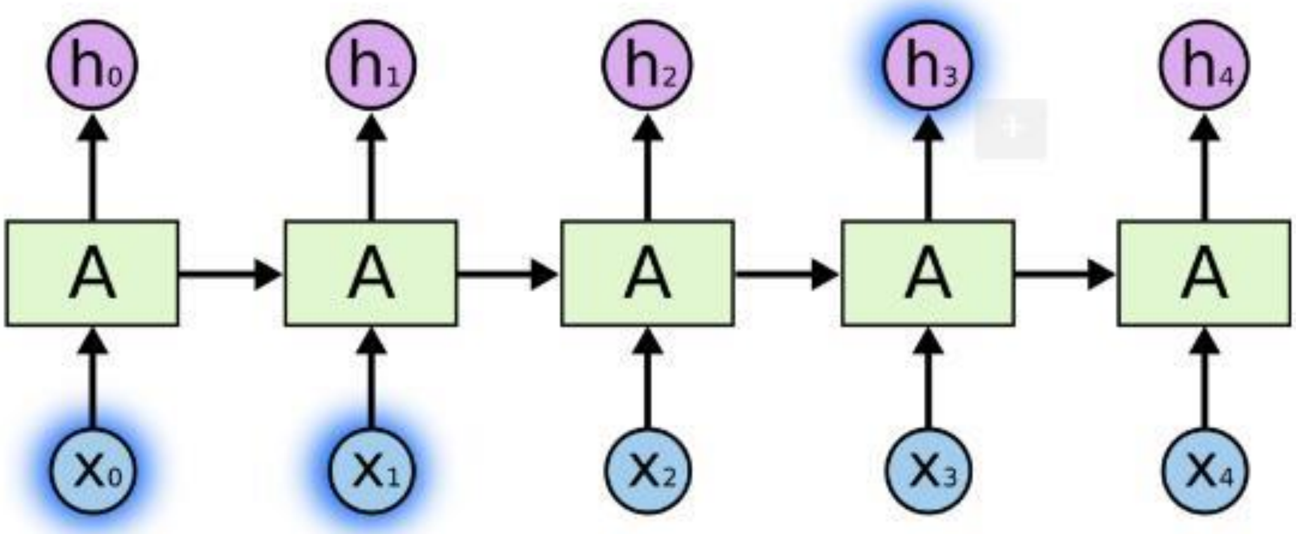
outline

- Recurrent Neural Network
  - Modeling
  - Training
- Long Short Term Memory (LSTM)
  - Motivation
  - Modeling
- Application
  - Generate article

Long Short Term Memory (LSTM)

- Motivation

"the clouds are in the *sky*,"

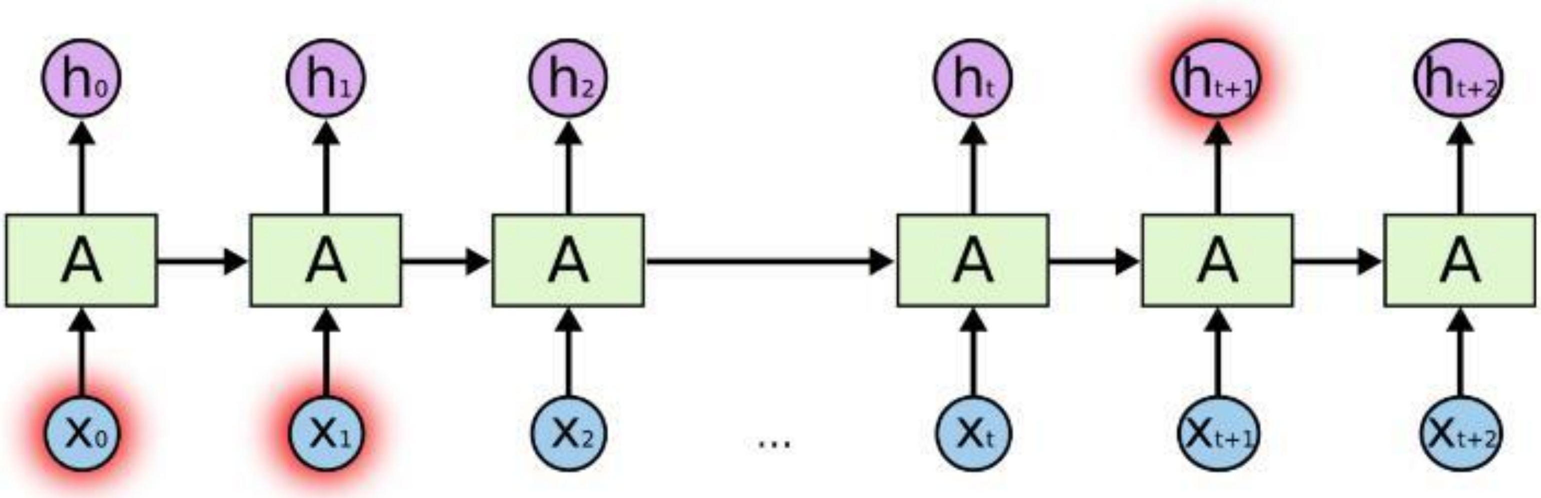


Source: Understanding LSTM Networks
Christopher Olah

Long Short Term Memory (LSTM)

- Motivation

"I grew up in France… I speak fluent *French*."

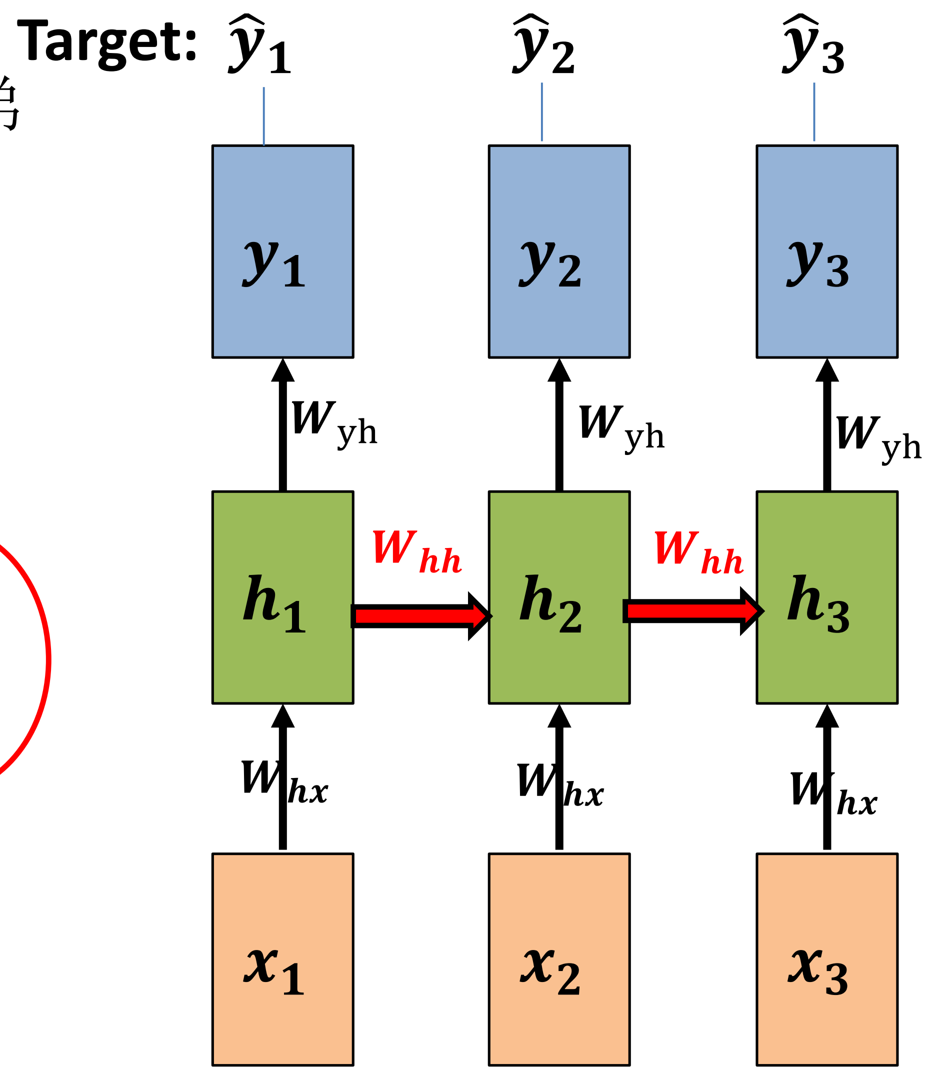


Source: Understanding LSTM Networks
Christopher Olah

LSTM

- Motivation
  - Gradient explosion (梯度爆炸)
  - Gradient vanishing(梯度弥散)

$$\frac{\mathrm{d\,L}}{d\,\boldsymbol{h}_t} = \sum_{s=t}^{T} \frac{\mathrm{d\,L}}{\boldsymbol{d}\,\boldsymbol{y}_s} \frac{\mathrm{d}\boldsymbol{y}_s}{d\boldsymbol{h}_s} \frac{\mathrm{d}\boldsymbol{h}_s}{d\boldsymbol{h}_t}$$



outline

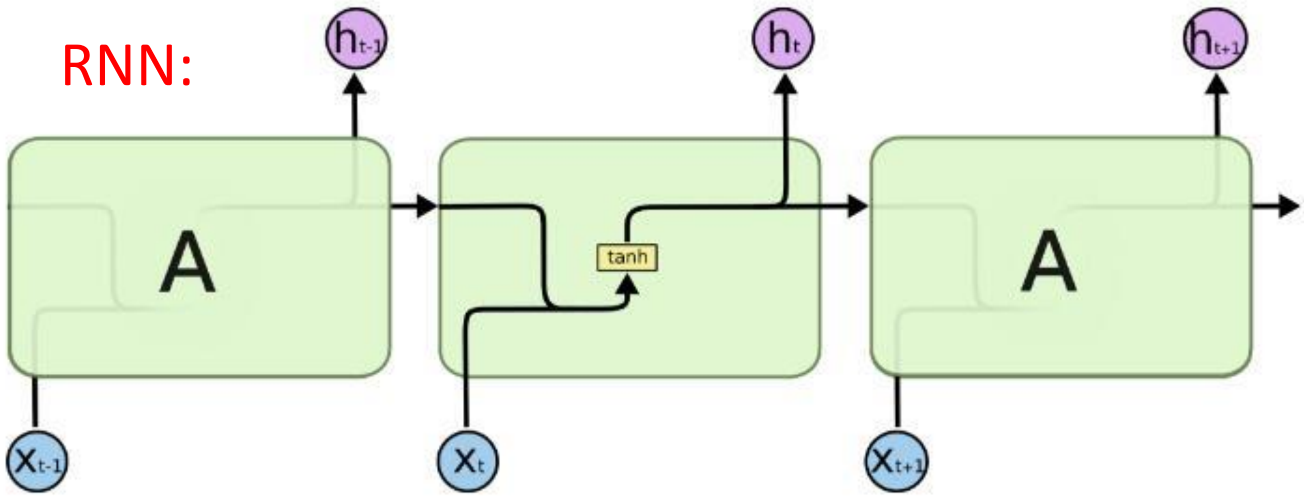
- Recurrent Neural Network
  - Modeling
  - Training
- Long Short Term Memory (LSTM)
  - Motivation
  - Modeling
- Application
  - Generate article

LSTM

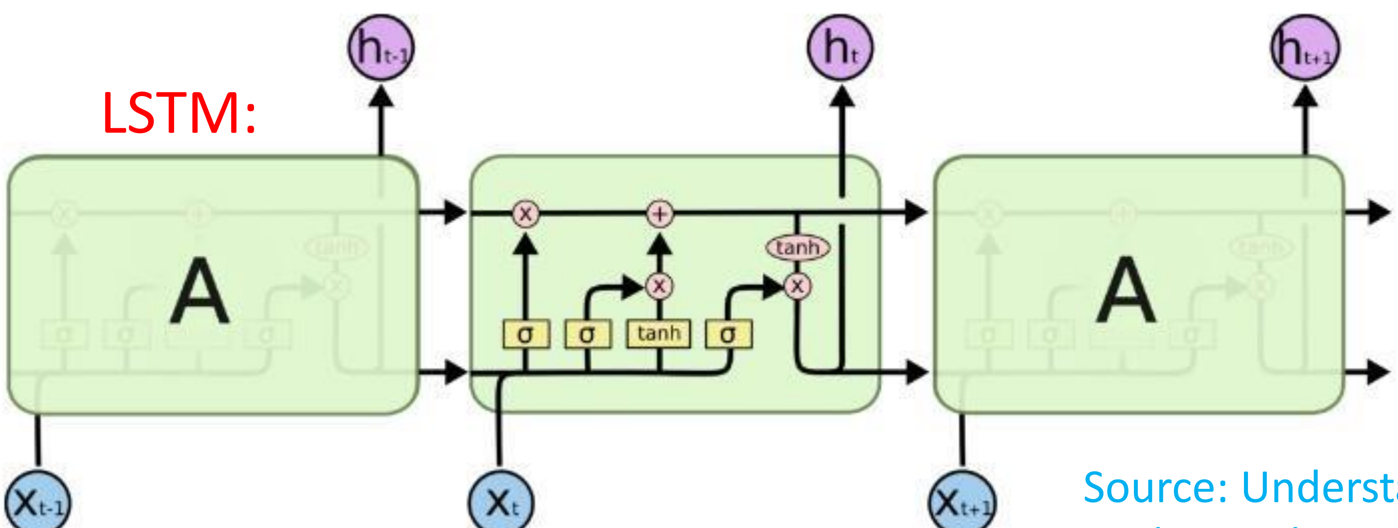
- Modeling

$$h_t = tanh(W_{hx}x + W_{hh}h_{t-1})$$

RNN:



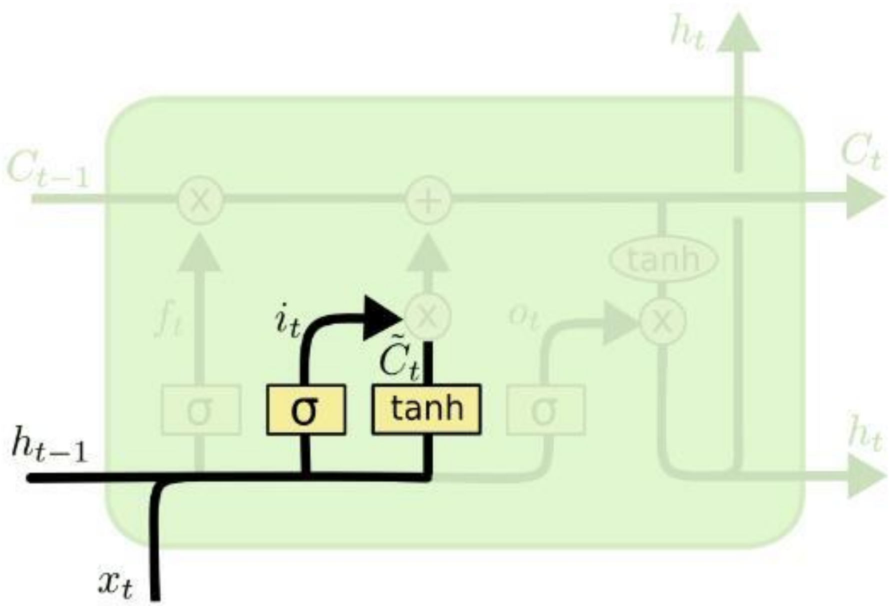
LSTM:



Source: Understanding LSTM Networks
Christopher Olah

LSTM

- Modeling
  - Input gate
  - Input information



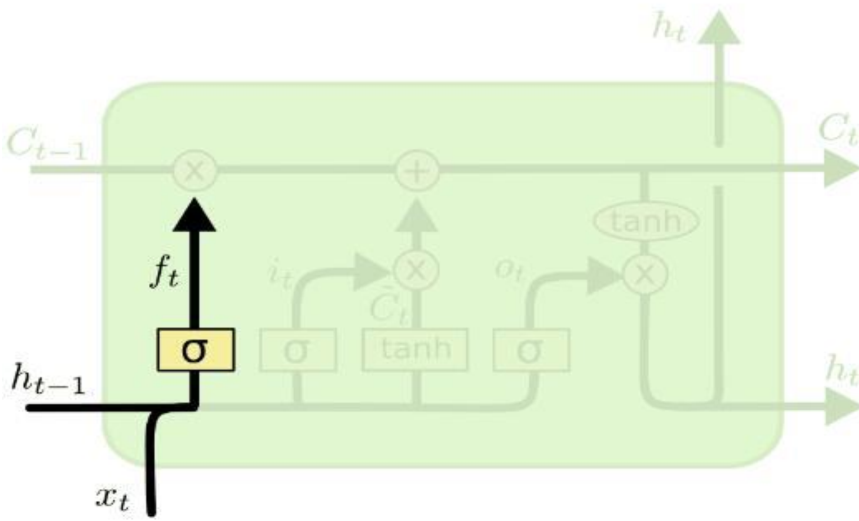
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

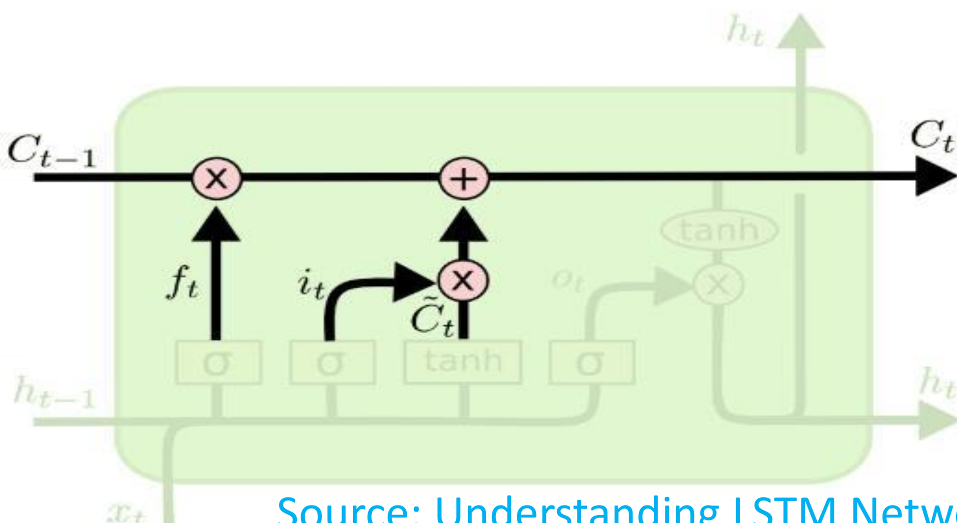
Source: Understanding LSTM Networks
Christopher Olah

LSTM

- Modeling



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$



Forget Gate

Input Gate

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

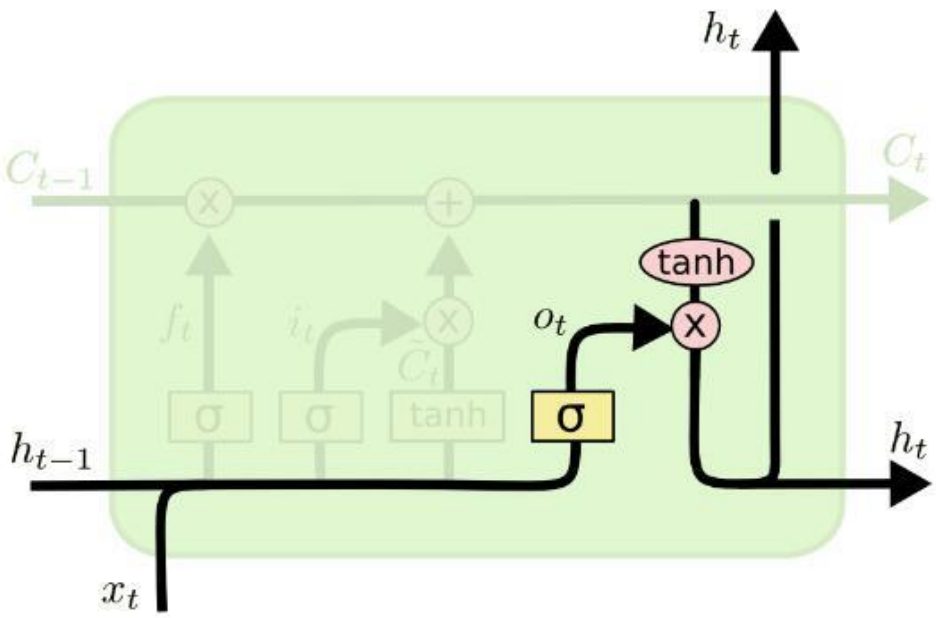
Previous Info

Input Info

Source: Understanding LSTM Networks
Christopher Olah

LSTM

- Modeling
  - Output gate
  - Output information



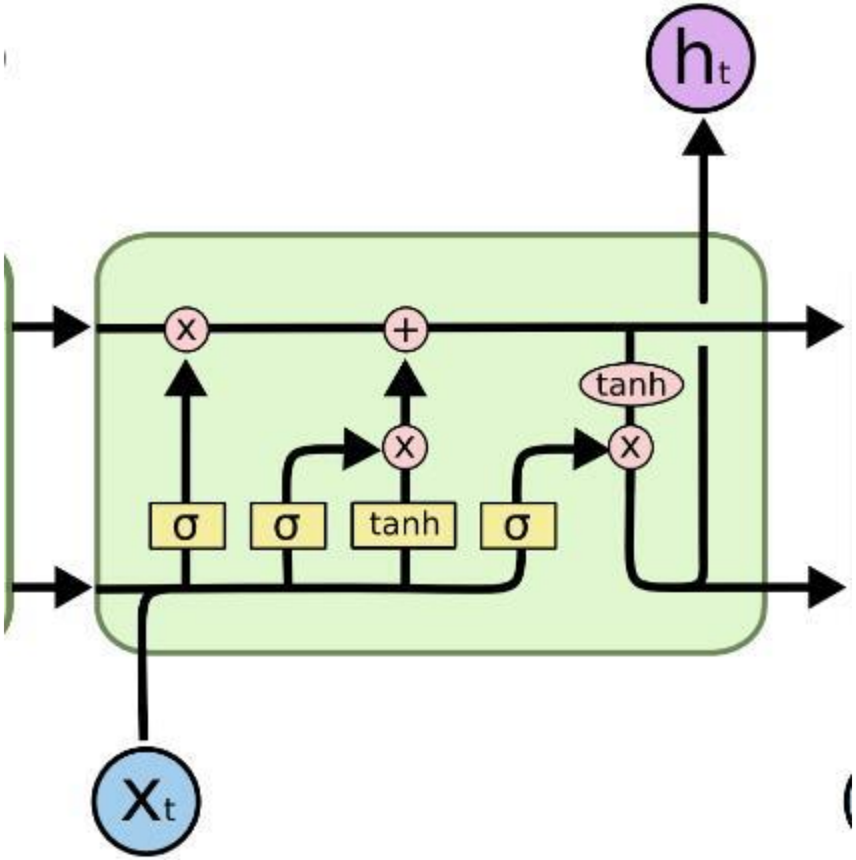
$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Source: Understanding LSTM Networks
Christopher Olah

LSTM

- Modeling



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

$$h_t = o_t * \tanh\left(C_t\right)$$

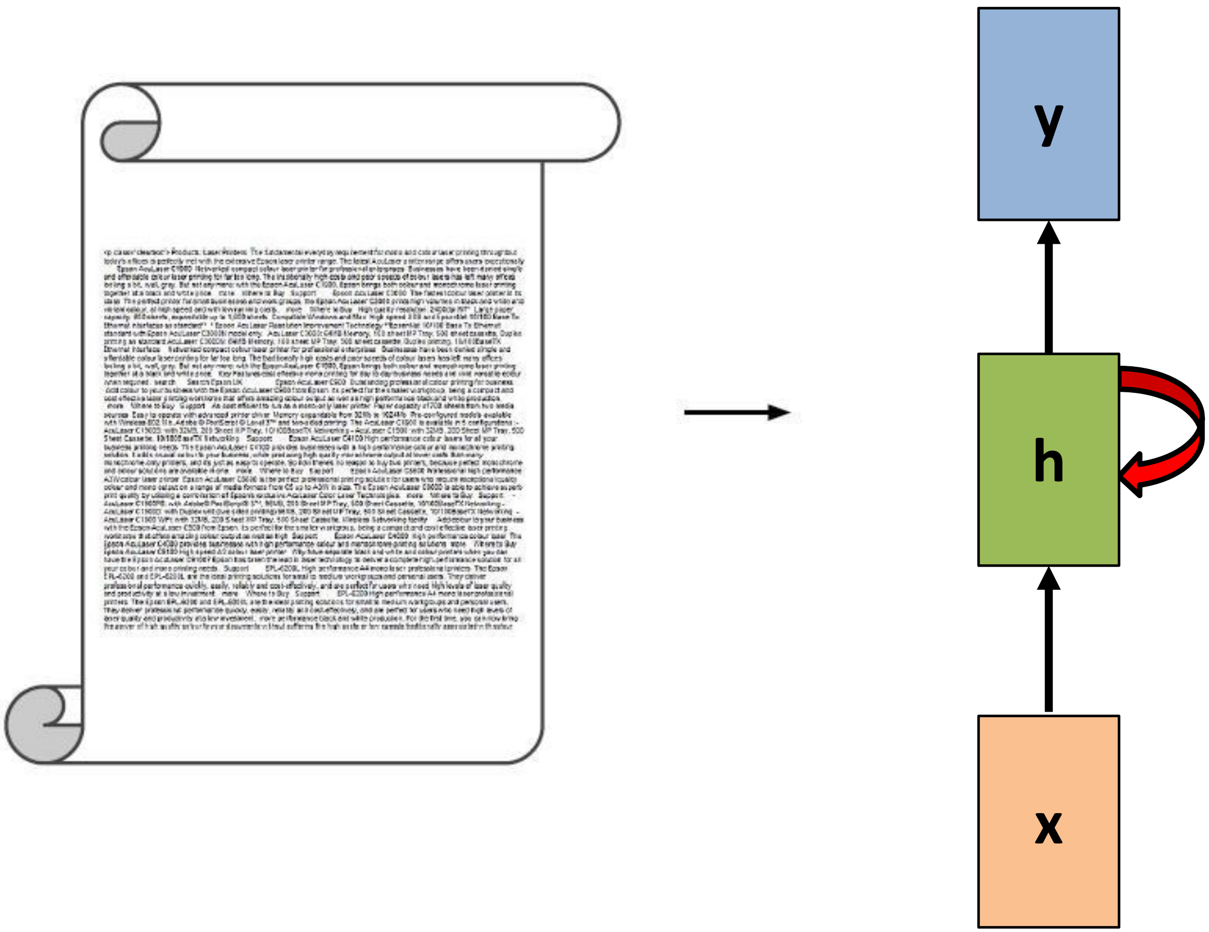
Source: Understanding LSTM Networks
Christopher Olah

outline

- Recurrent Neural Network
  - Modeling
  - Training
- Long Short Term Memory (LSTM)
  - Motivation
  - Modeling
- Application
  - Generate article

RNN-application

- Generate article



RNN-application

- Generate article

Sonnet 116 – Let me not ...

by William Shakespeare

Let me not to the marriage of true minds
        Admit impediments. Love is not love
Which alters when it alteration finds,
        Or bends with the remover to remove:
O no! it is an ever-fixed mark
        That looks on tempests and is never shaken;
It is the star to every wandering bark,
        Whose worth's unknown, although his height be taken.
Love's not Time's fool, though rosy lips and cheeks
        Within his bending sickle's compass come:
Love alters not with his brief hours and weeks,
        But bears it out even to the edge of doom.
If this be error and upon me proved,
        I never writ, nor no man ever loved.

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generate article

at first:

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

train more

```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

train more

```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generate article

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generate article

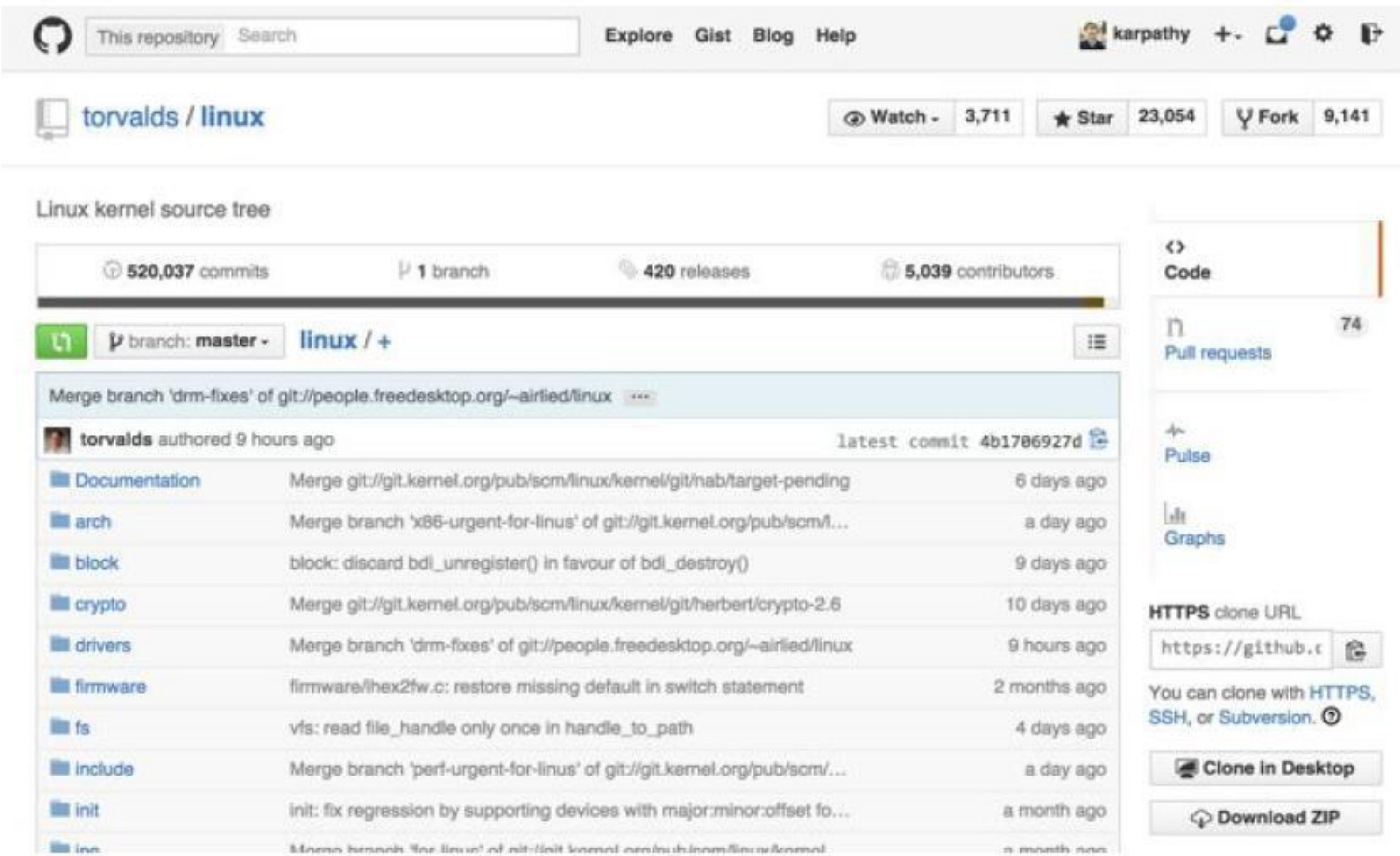
```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generate C code



Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generated C code

```c
static void do_command(struct seq_file *m, void *v)
{
  int column = 32 << (cmd[2] & 0x80);
  if (state)
    cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
  else
    seq = 1;
  for (i = 0; i < 16; i++) {
    if (k & (1 << 1))
      pipe = (in_use & UMXTHREAD_UNCCA) +
        ((count & 0x00000000fffffff8) & 0x000000f) << 8;
    if (count == 0)
      sub(pid, ppc_md.kexec_handle, 0x20000000);
    pipe_set_bytes(i, 0);
  }
  /* Free our user pages pointer to place camera if all dash */
  subsystem_info = &of_changes[PAGE_SIZE];
  rek_controls(offset, idx, &soffset);
  /* Now we want to deliberately put it to device */
  control_check_polarity(&context, val, 0);
  for (i = 0; i < COUNTER; i++)
    seq_puts(s, "policy ");
}
```

Generated
C code

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy

RNN-application

- Generated C code

```
#include <asm/io.h>
#include <asm/prom.h>
#include <asm/e820.h>
#include <asm/system_info.h>
#include <asm/setew.h>
#include <asm/pgproto.h>

#define REG_PG    vesa_slot_addr_pack
#define PFM_NOCOMP  AFSR(0, load)
#define STACK_DDR(type)     (func)

#define SWAP_ALLOCATE(nr)     (e)
#define emulate_sigs()  arch_get_unaligned_child()
#define access_rw(TST)  asm volatile("movd %%esp, %0, %3" : : "r" (0));   \
  if (__type & DO_READ)

static void stat_PC_SEC __read_mostly offsetof(struct seq_argsqueue, \
          pC>[1]);

static void
os_prefix(unsigned long sys)
{
#ifdef CONFIG_PREEMPT
  PUT_PARAM_RAID(2, sel) = get_state_state();
  set_pid_sum((unsigned long)state, current_state_str(),
           (unsigned long)-1->lr_full; low;
}
```

Source: The Unreasonable Effectiveness of Recurrent Neural Networks
Andrej Karpathy