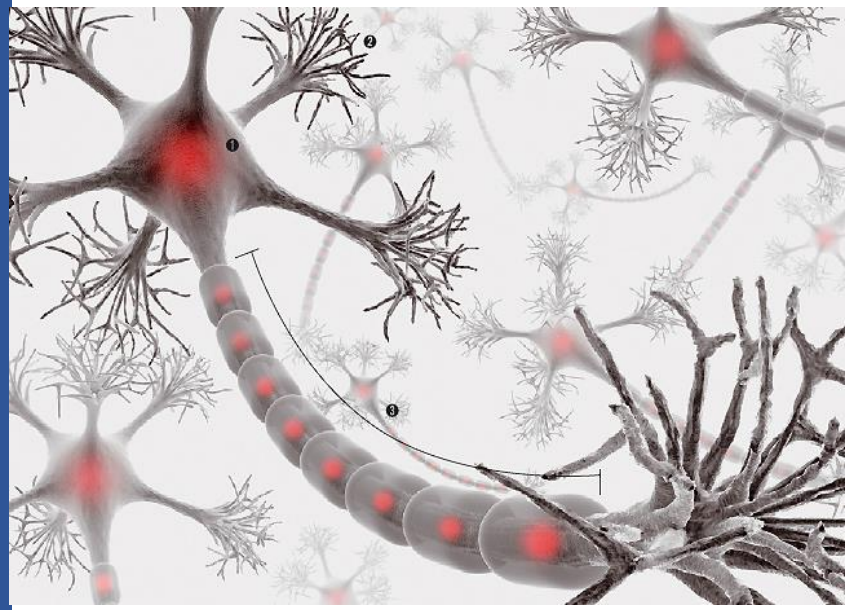
순환 신경망
(Recurrent Neural Network)

학습 목표

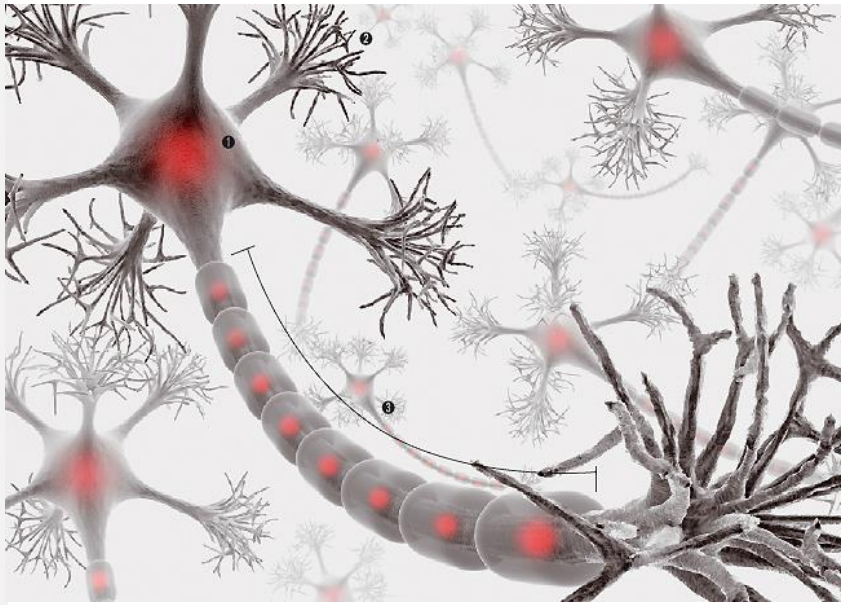
- Recurrent Neural Network의 작동 원리를 이해한다.

주요 내용

1. RNN 개요
2. RNN 주요 모델
3. BPTT
4. RNN 예제 - Language Model
5. RNN 예제 – Image Captioning
6. LSTM/GRU



1 RNN 개요



Sequence Data

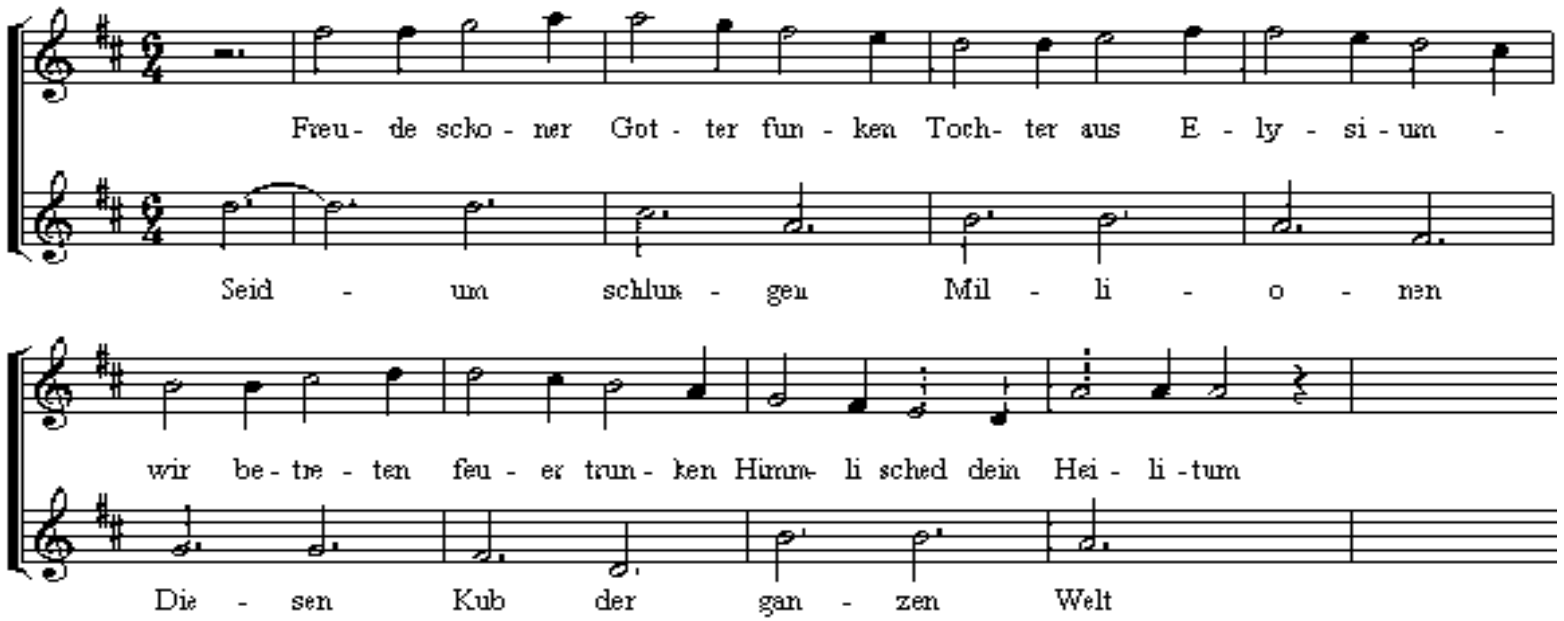
시간적, 공간적 순서 관계에 의해 Context를 갖는 특성이 있다.

I want to have an apple.

문맥이 형성하는 주변의 단어들을 함께 살펴봐야 판단할 수 있다.

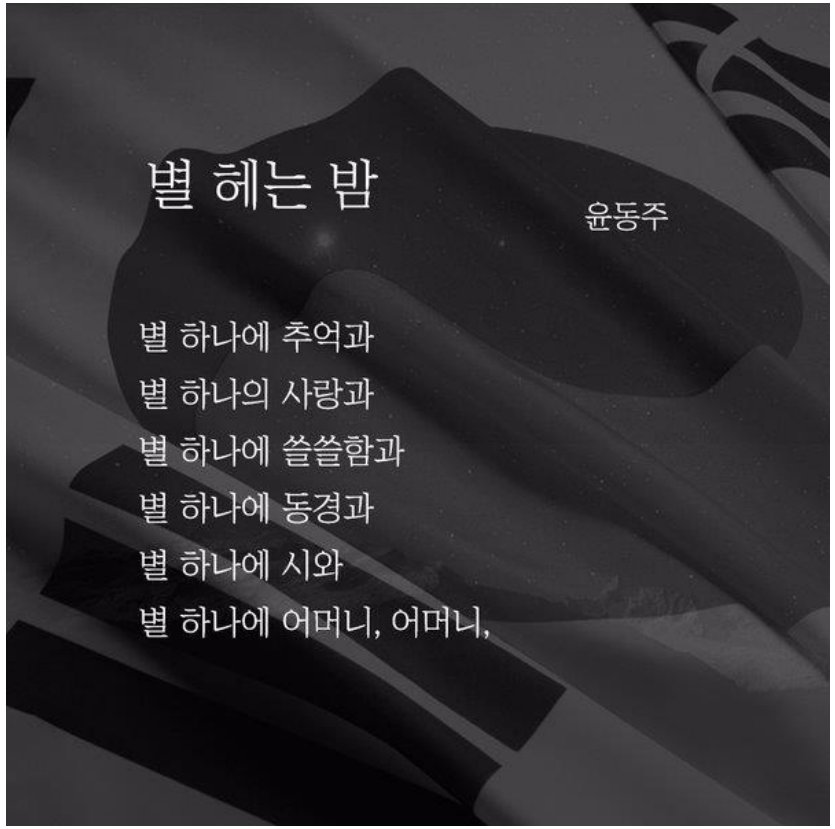
Sequence Data

음악/소리 (Sound)



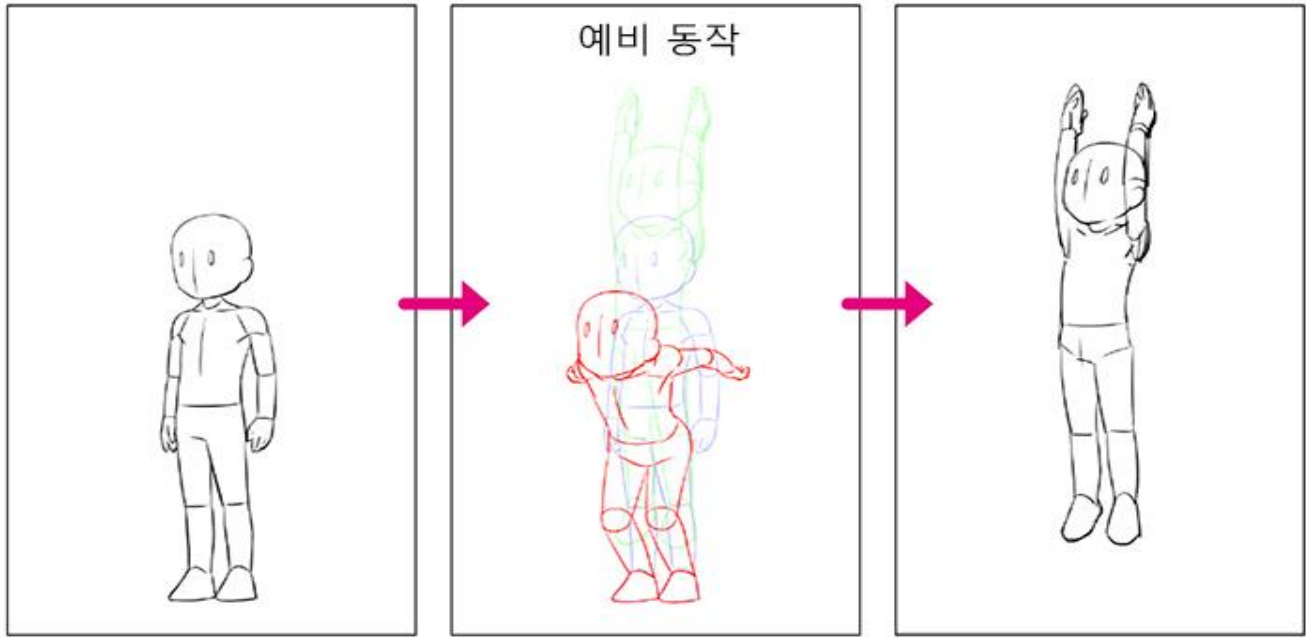
Sequence Data

문자 언어/소리 언어 (Language)



Sequence Data

동영상/애니메이션 (Video)



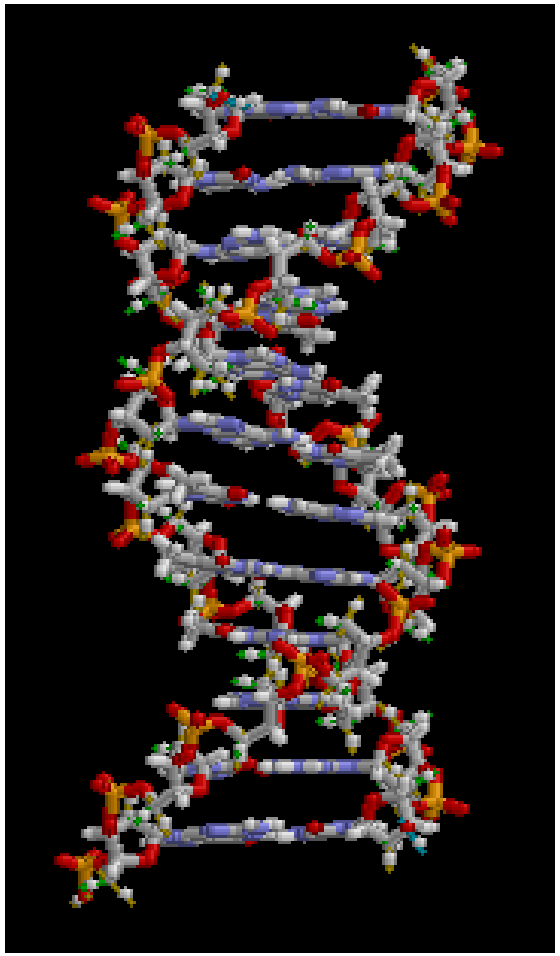
Sequence Data

운동 (Locomotion)

https://simtk-confluence.stanford.edu:8443/display/OpenSim/Getting+Started+with+Inverse+Kinematics

Sequence Data

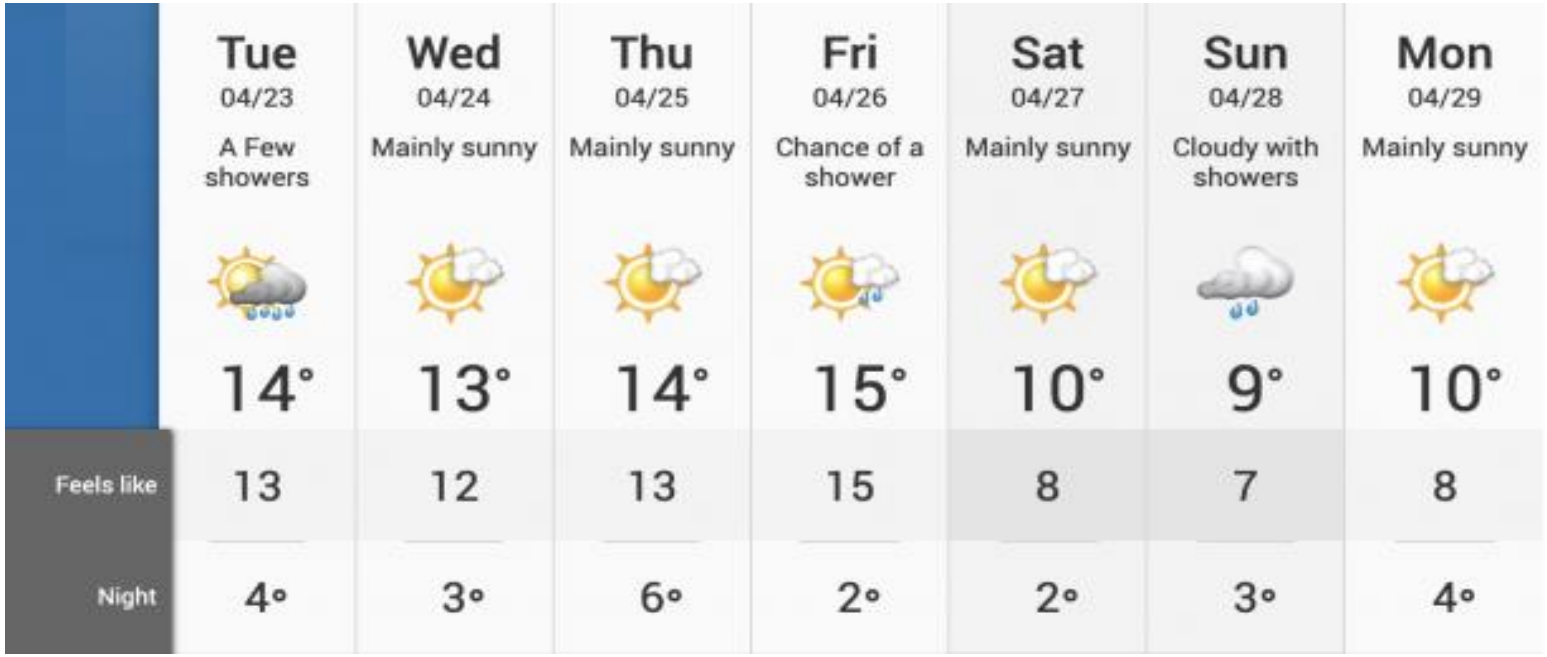
DNA 염기 서열



Sequence Data

Sequence Data

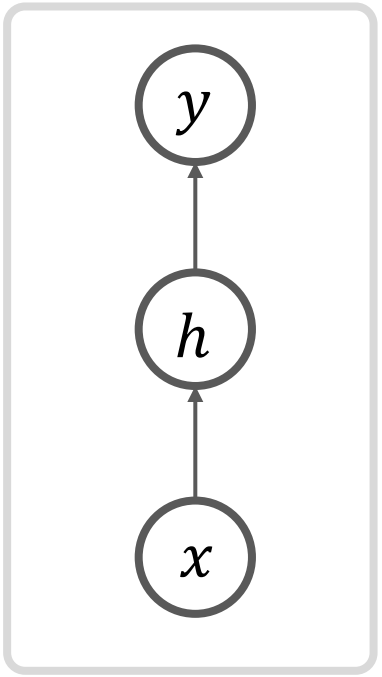
자연 현상 (Natural Phenomenon)



	Tue 04/23	Wed 04/24	Thu 04/25	Fri 04/26	Sat 04/27	Sun 04/28	Mon 04/29
	A Few showers	Mainly sunny	Mainly sunny	Chance of a shower	Mainly sunny	Cloudy with showers	Mainly sunny
	14°	13°	14°	15°	10°	9°	10°
Feels like	13	12	13	15	8	7	8
Night	4°	3°	6°	2°	2°	3°	4°

기억을 갖는 신경망

Feedforward Neural Network



One-to-One Mapping

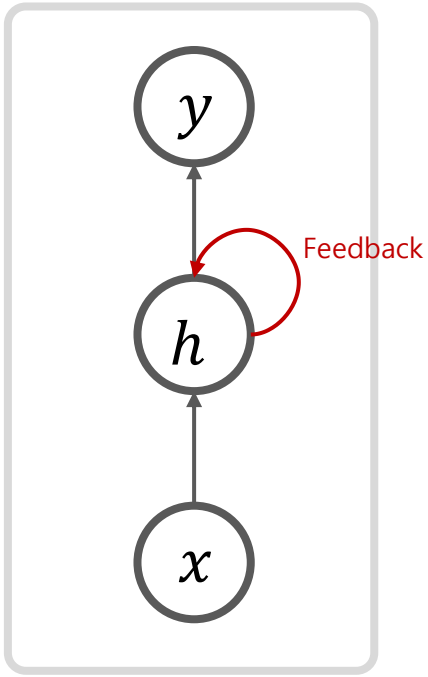
$$y = f(x)$$

- 현재의 입력만 출력에 반영됨

이전에 봤던 데이터를 기억하는 신경망을 만들 수 있을까?

기억을 갖는 신경망

RNN (Recurrent Neural Network)



Many-to-Many Mapping

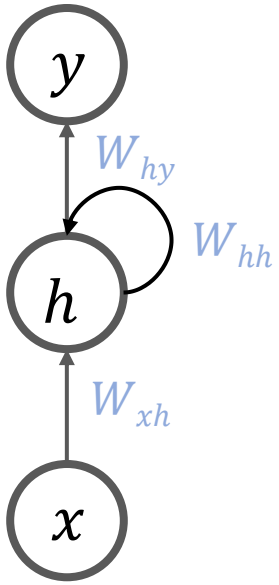
$$h_t = f_W(h_{t-1}, x_t)$$

- Hidden State에 이전 입력에 대한 기억을 저장
- 새로운 입력이 들어올 때마다 기억을 수정

Feedback 연결을 통해 과거의 기억을 전달해보자!

Vanilla RNN

RNN Formula



$$y_t = W_{hy} \cdot h_t$$

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

new state / old state / input

W를 파라미터로 갖는 함수

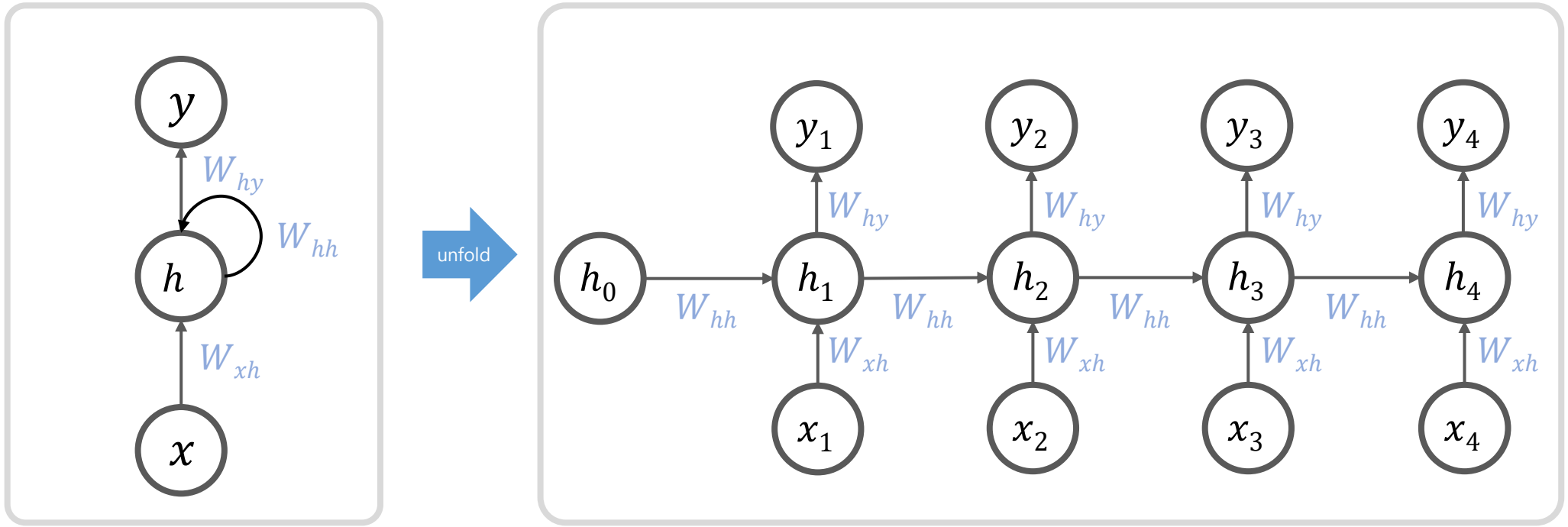
Vanilla RNN

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

모든 단계에서 파라미터를 공유

Vanilla RNN

Recurrent Neural Network Unfolding



시간에 따라 펼치면 피드포워드 네트워크와 동일

Vanilla RNN

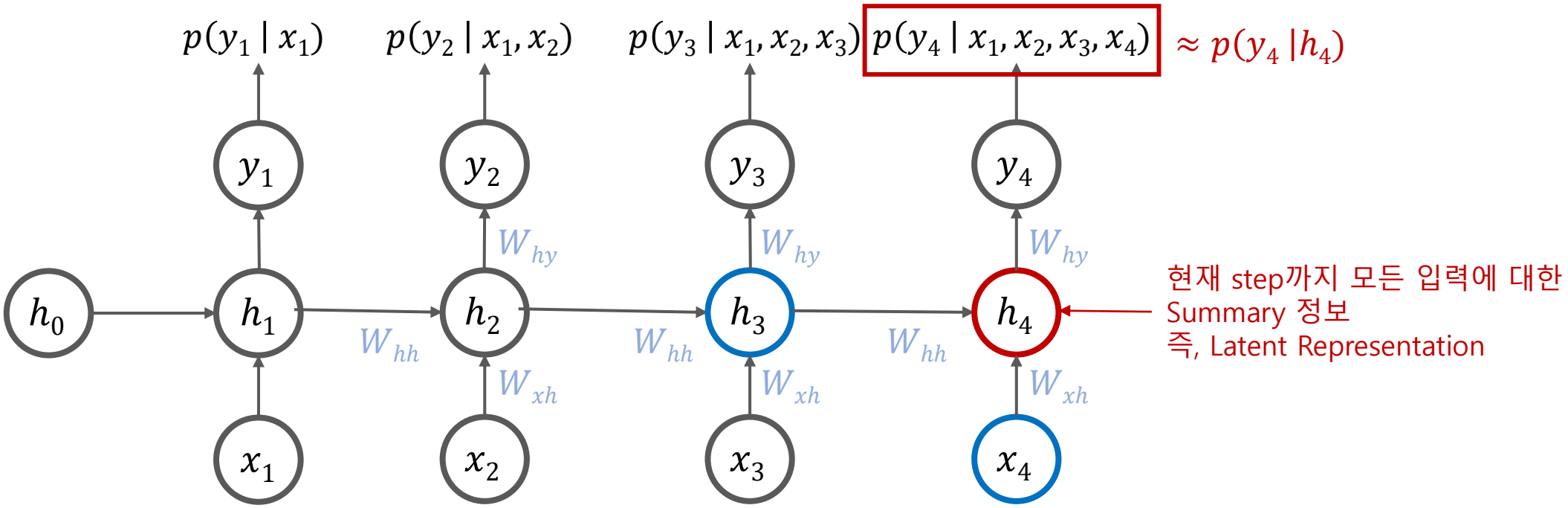
모든 단계에서 파라미터를 공유하면서 생기는 장점은 무엇이 있을까?

1. 순서 관계가 있는 데이터 패턴을 포착할 수 있다.

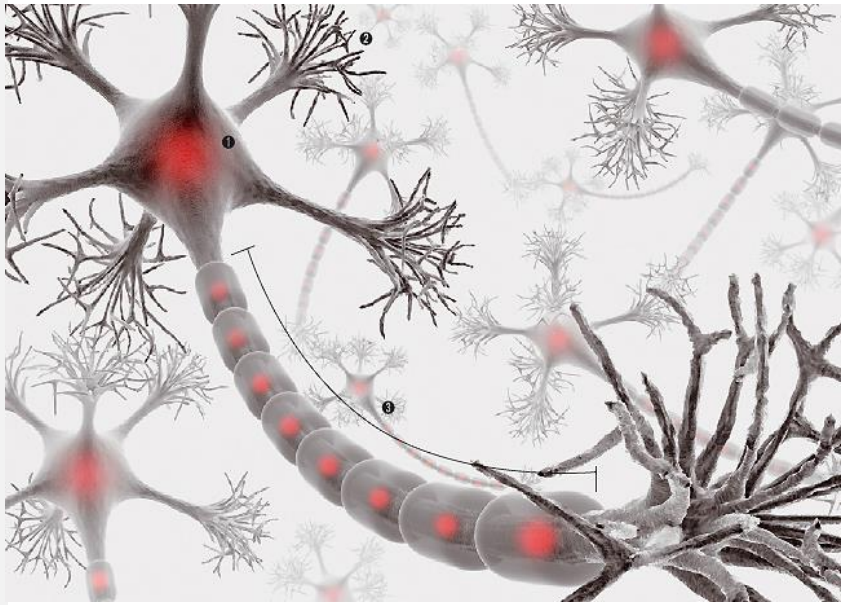
2. 가변 길이 데이터를 처리할 수 있다.

2. 파라미터 수가 절약되고 정규화 효과가 생긴다.

기억의 형태

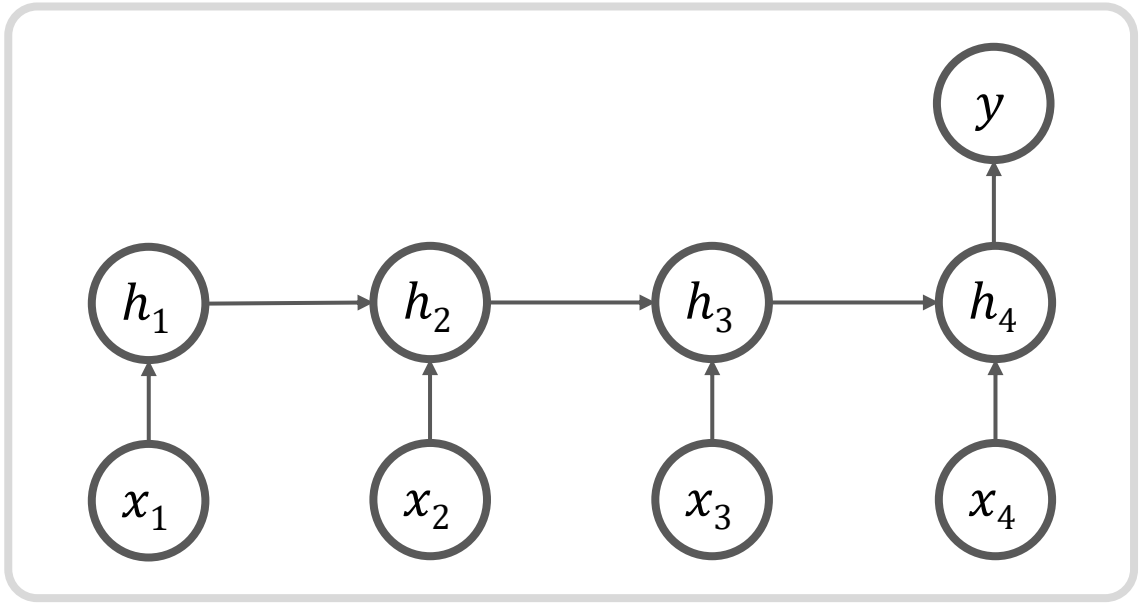


2 RNN 주요 모델



Many-to-One

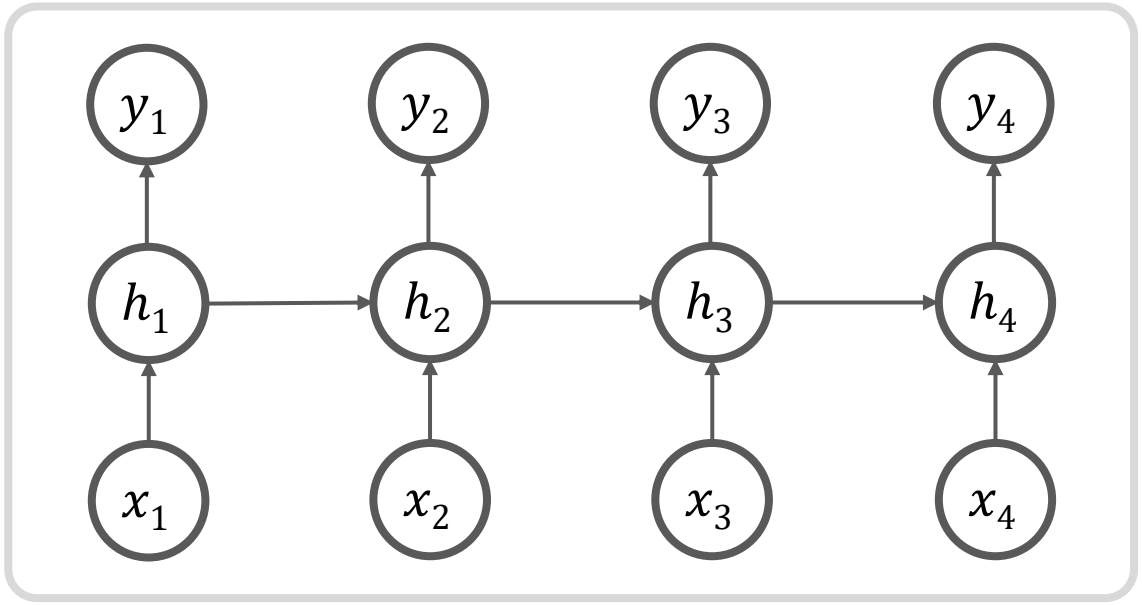
Many-to-One 모델



- 감성 분석 : sequence of words -> sentiment

Many-to-Many

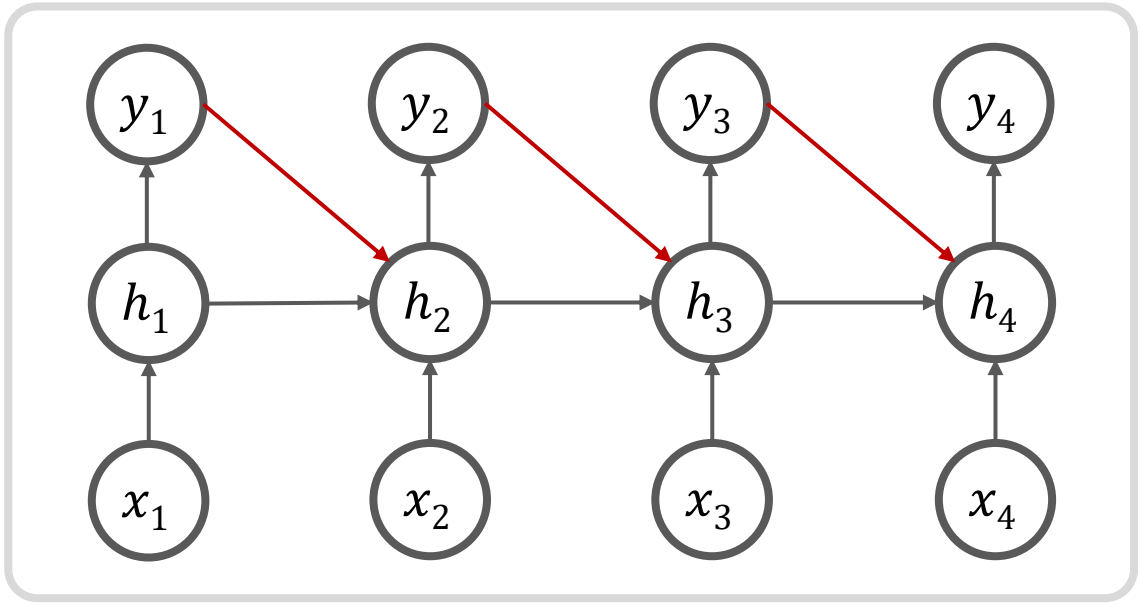
Many-to-Many 모델



- 프레임 별 비디오 분류

Many-to-Many

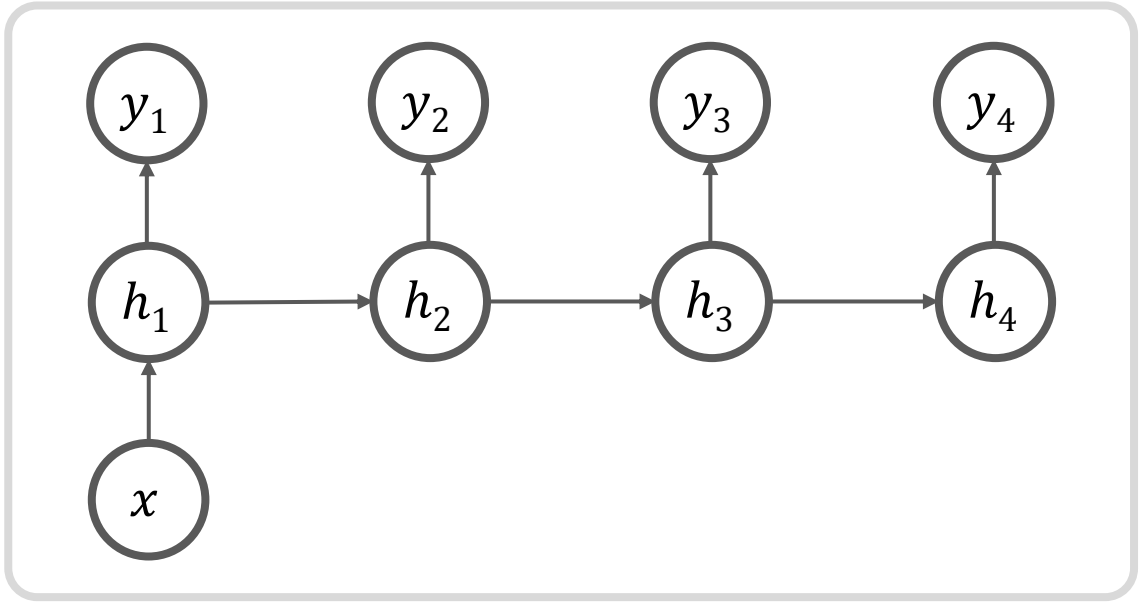
Teacher Forcing



- 학생이 한 문제를 풀 때마다 교사가 바로 정답을 제시함으로써 다음 문제를 순조롭게 풀 수 있도록 지도하는 방식
- Output을 다음 단계의 input으로 사용하는 훈련 기법으로 늦은 수렴 혹은 불안정한 훈련을 개선
- 훈련 시에는 output 대신 teacher signal로 ground truth를 사용
- Sequence prediction : machine translation, caption generation, text summarization

One-to-Many

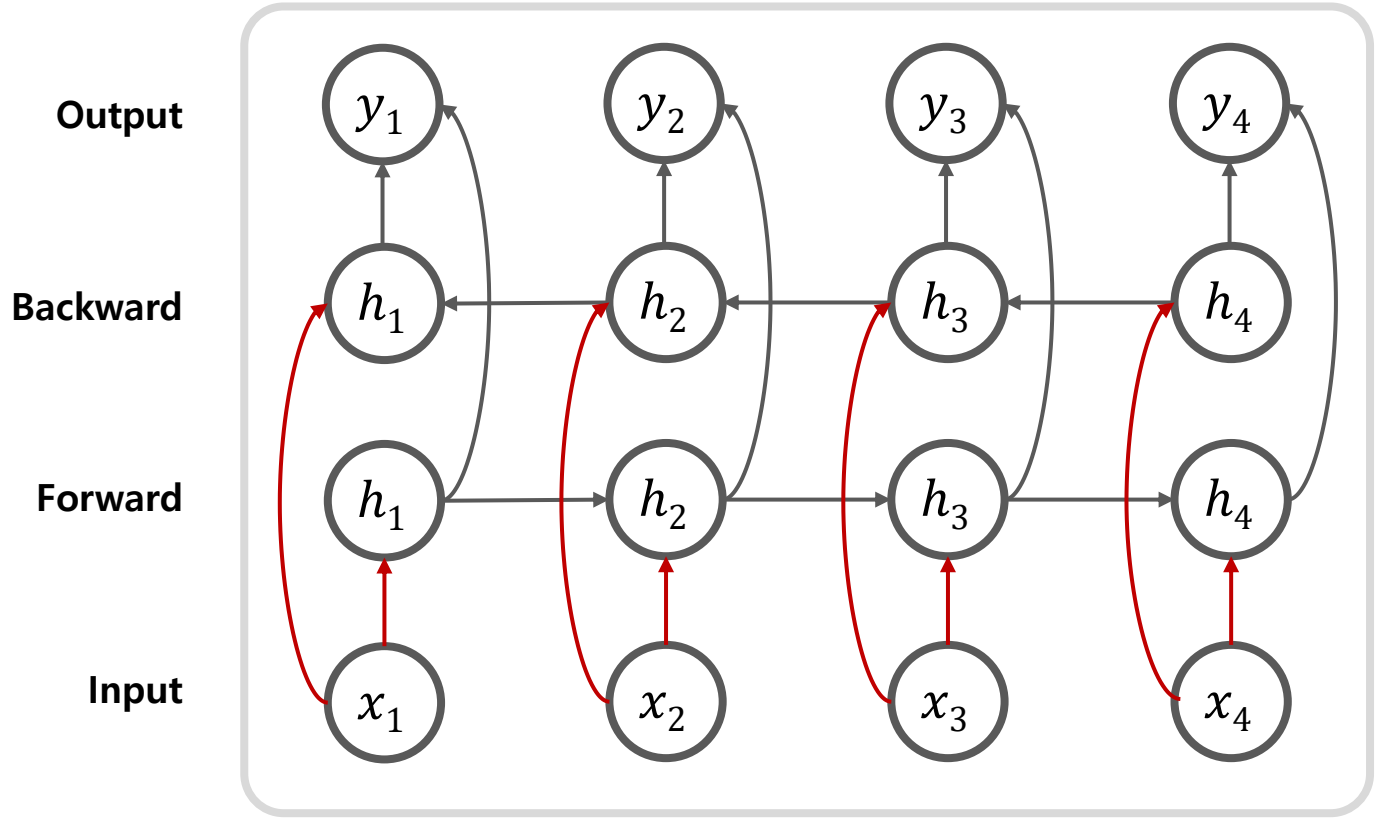
One-to-Many 모델



- Image Captioning : image -> sequence of words

Bidirectional

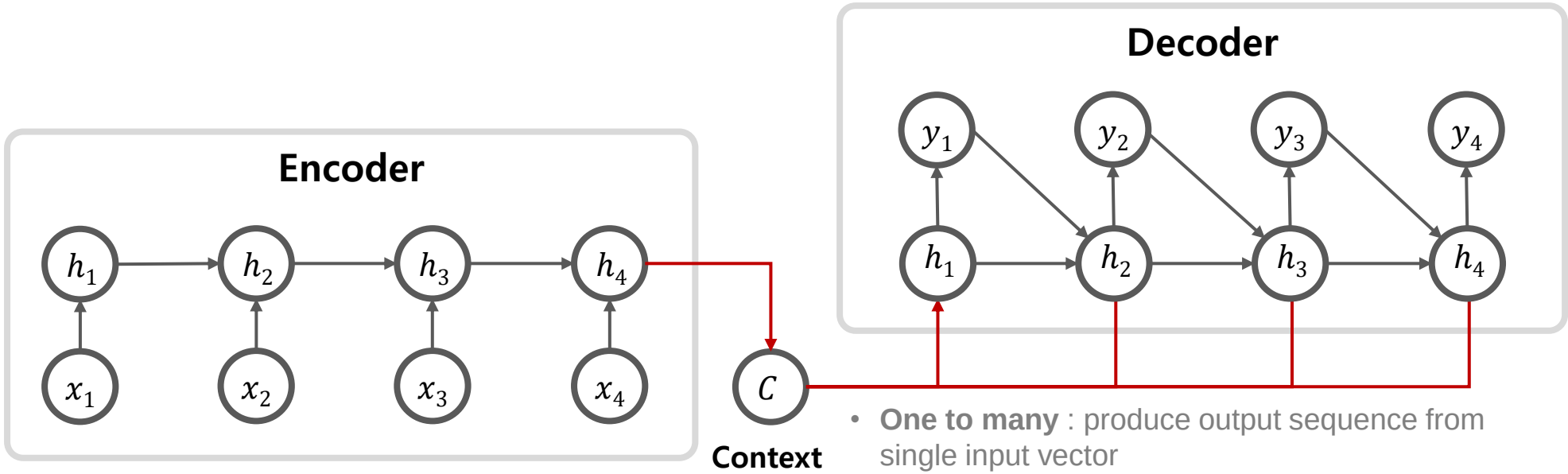
Bidirectional 모델



- 양쪽 방향으로 sequence를 살펴보는 방식
- Machine Translation : seq of words -> seq of words

Encoder-Decoder

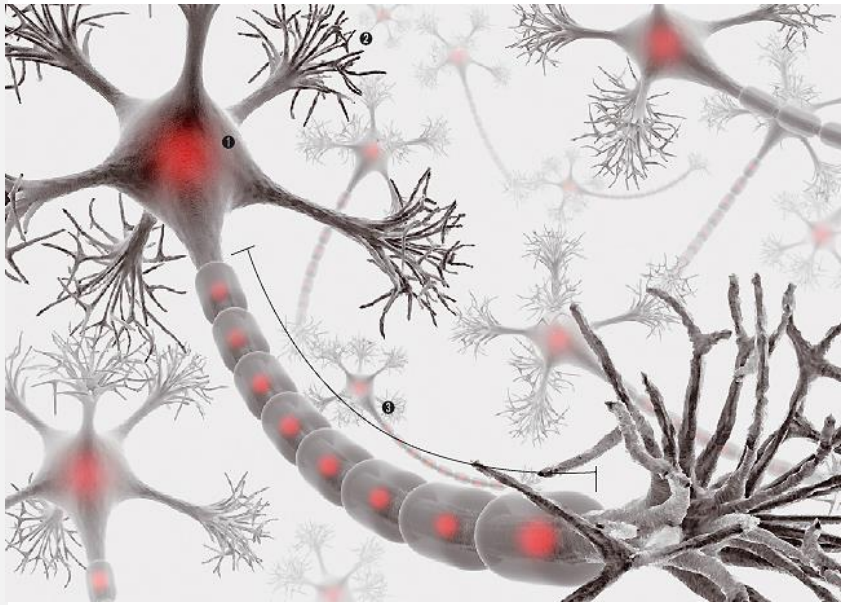
Encoder-Decoder 모델 * sequence-to-sequence이라고도 함



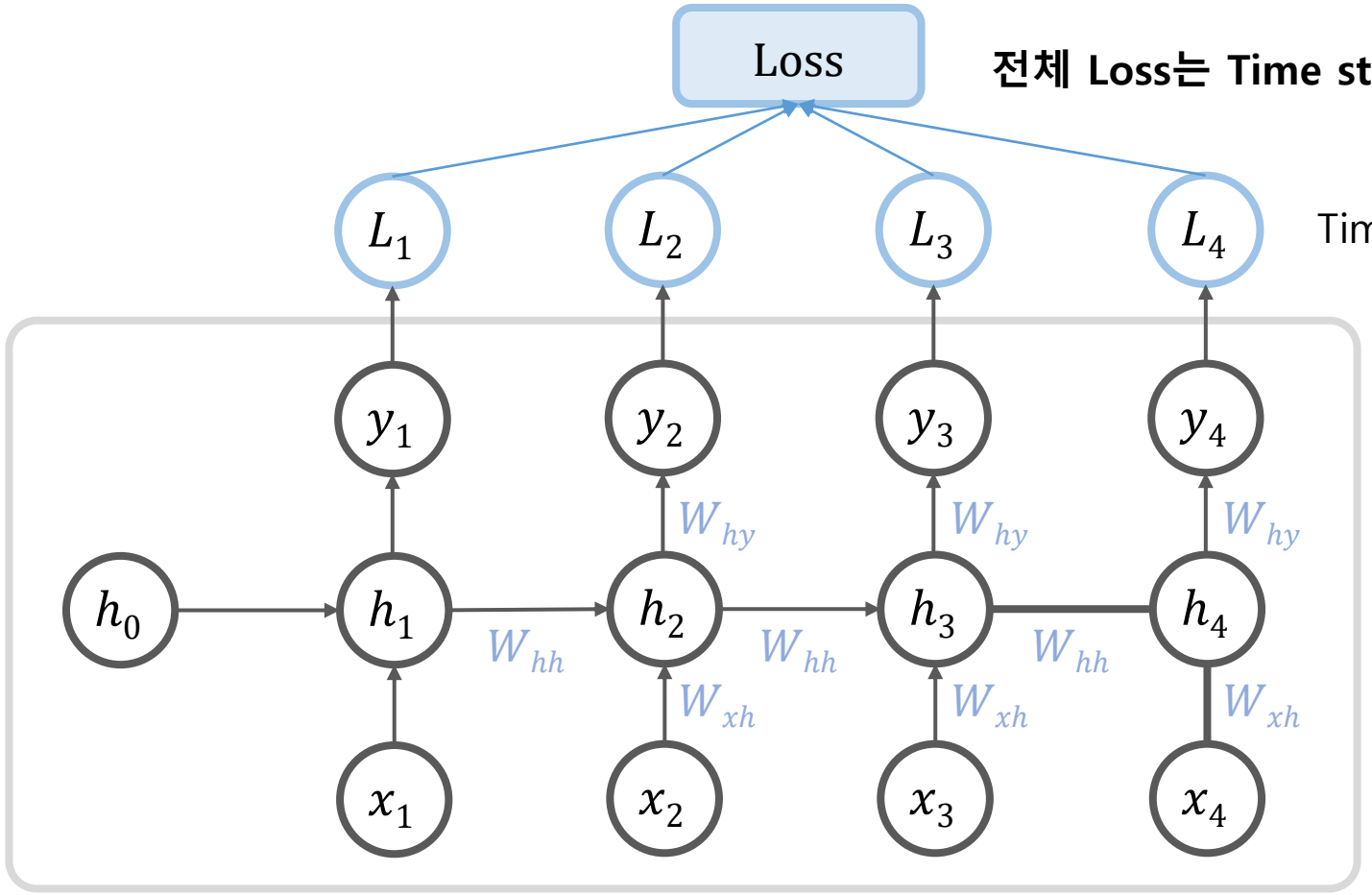
- **Many to one** : encode input sequence in a single vector
- **One to many** : produce output sequence from single input vector

- 모델 입력과 출력의 길이가 다를 때 사용하는 모델
- Encoder는 context variable c를 출력하며 이는 decoder에 입력으로 사용됨
- Machine Translation : seq of words -> seq of words

3 Backpropagation Through Time



Loss 계산

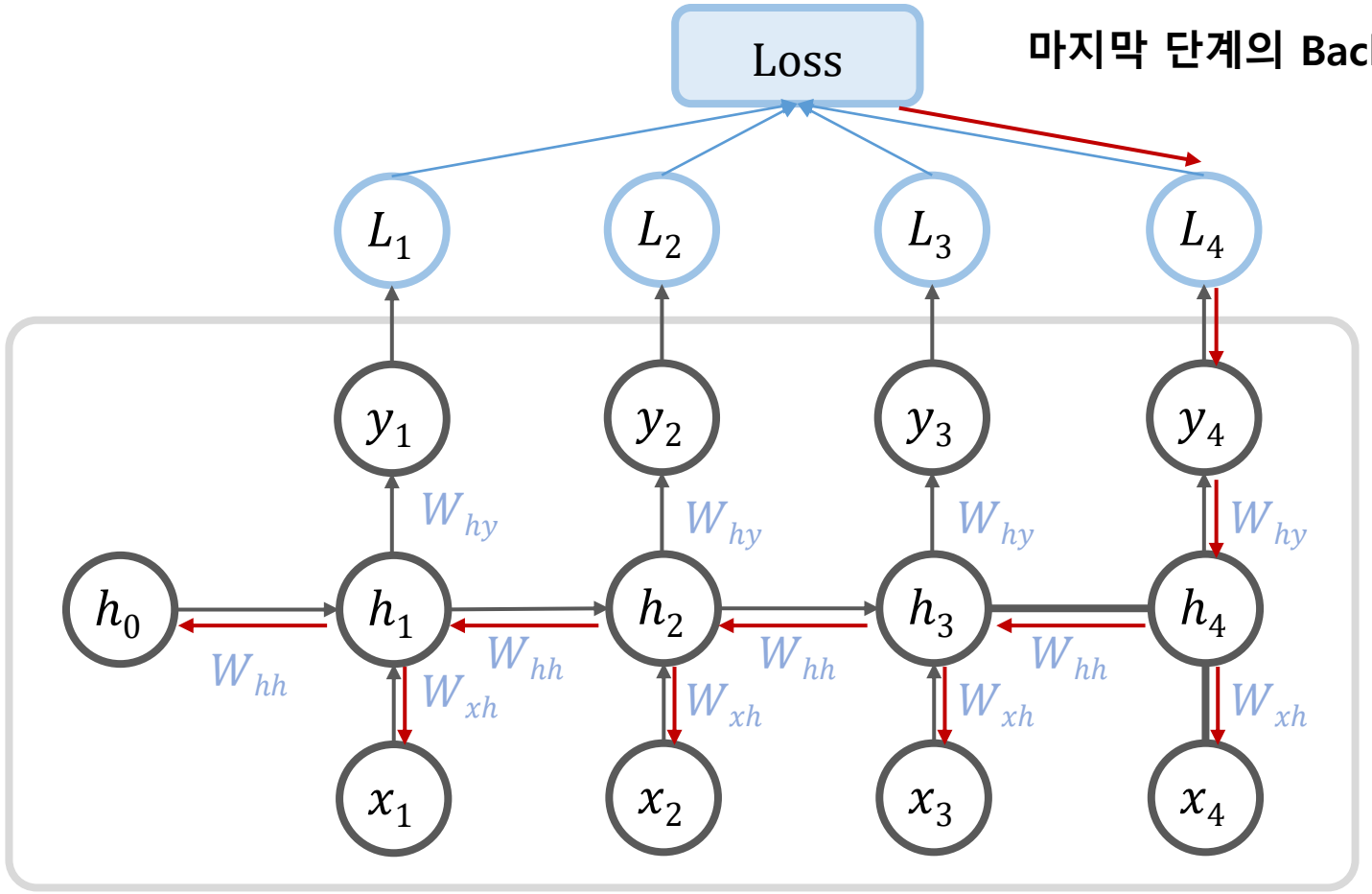


전체 Loss는 Time step 별 Loss를 더한 값

Time step에 별로 backpropagation 실행

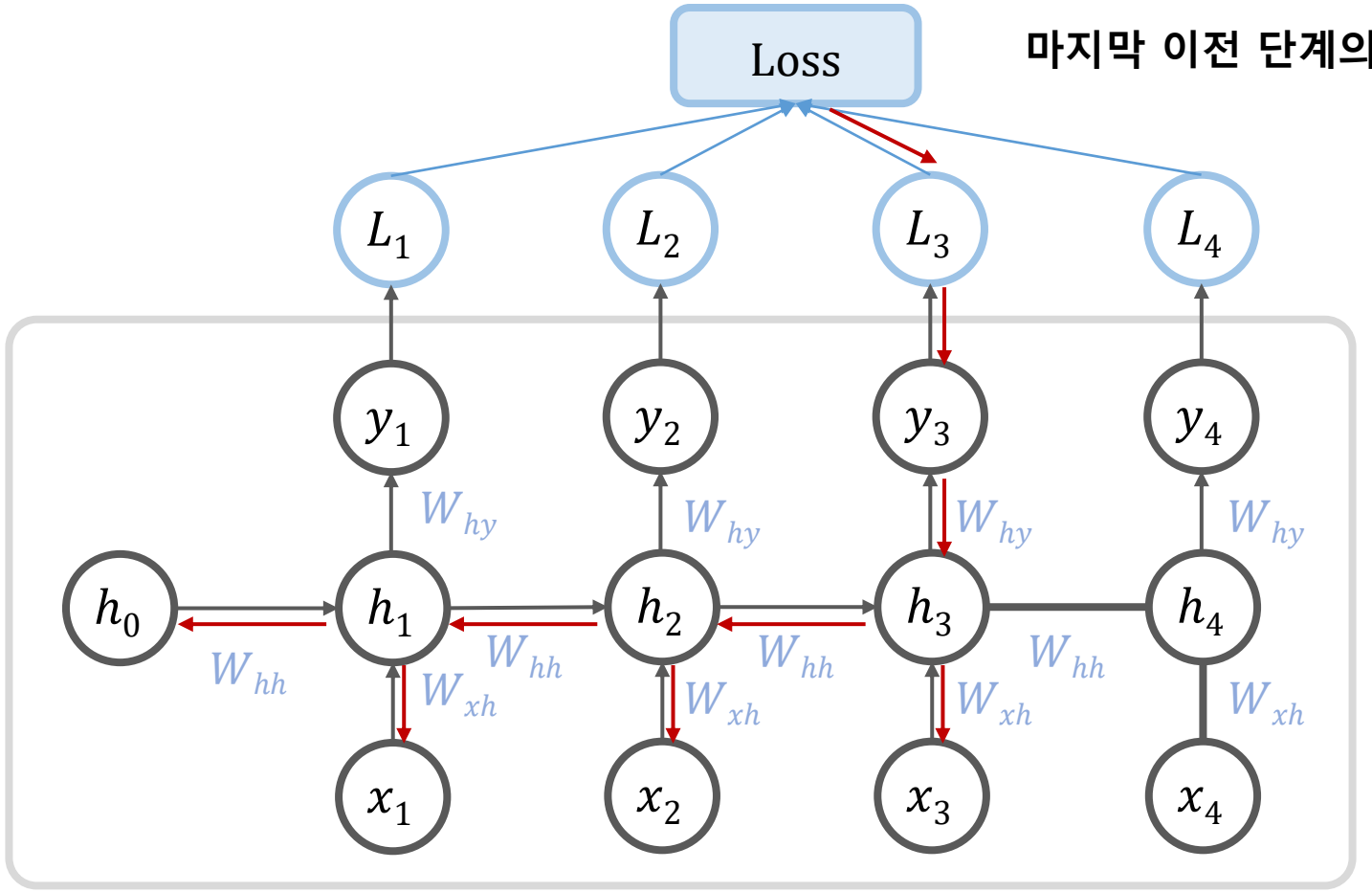
- 각 time step 별로 W에 대한 Local Gradient 계산
- Local Gradient를 모두 더한 값이 W에 대한 최종 Gradient가 됨

Backpropagation



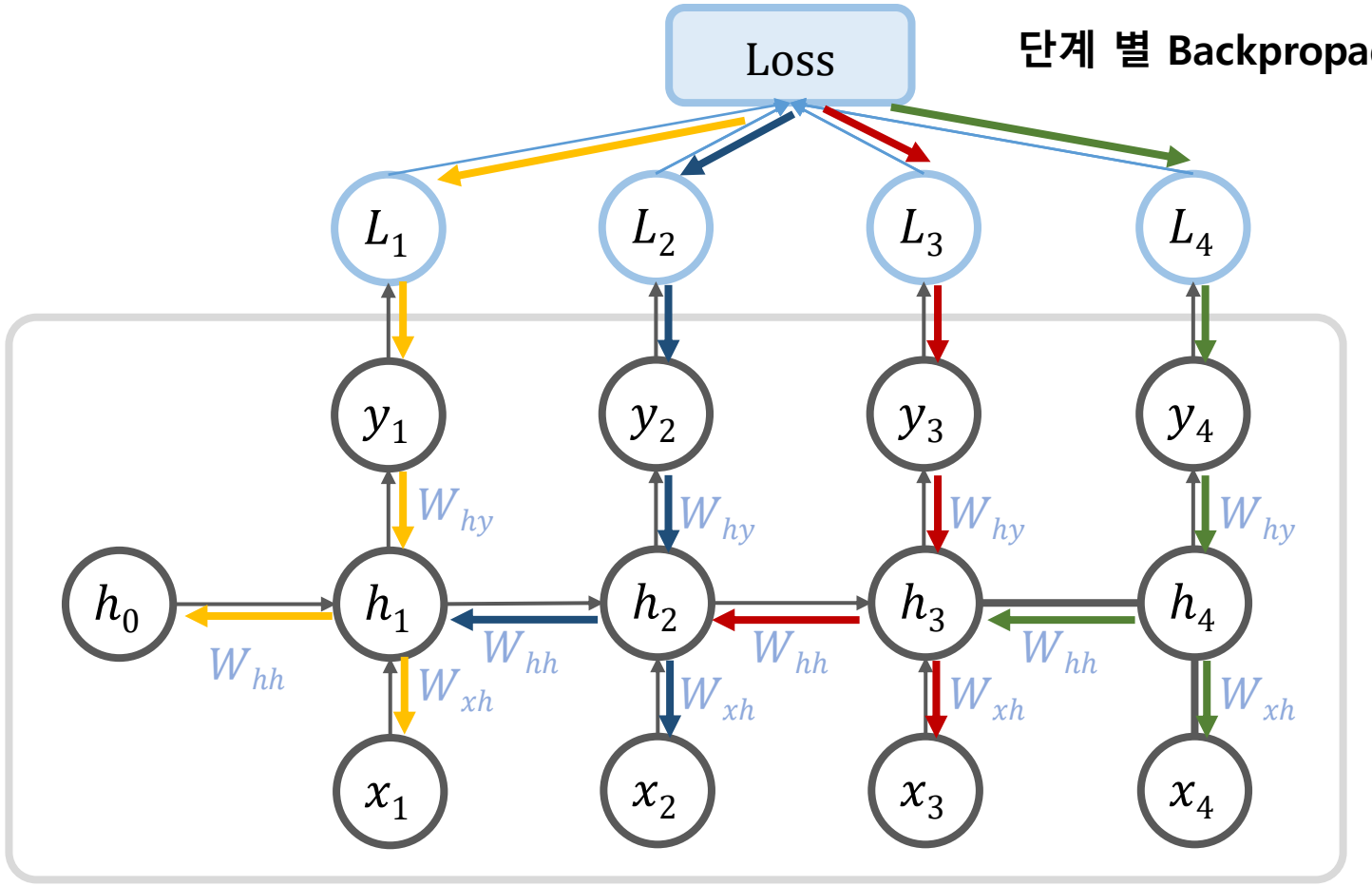
마지막 단계의 Backpropagation 경로

Backpropagation



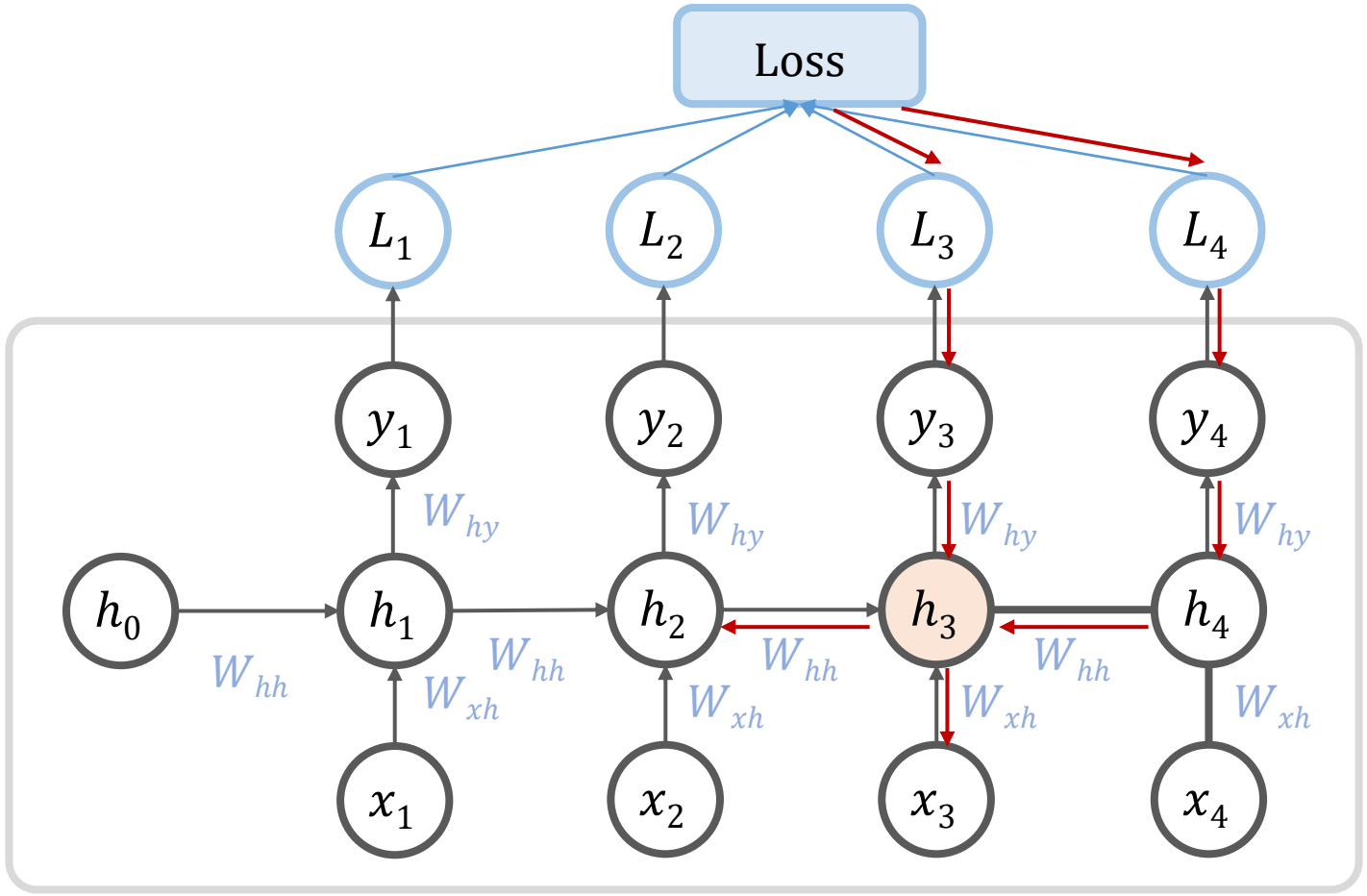
마지막 이전 단계의 Backpropagation 경로

Backpropagation



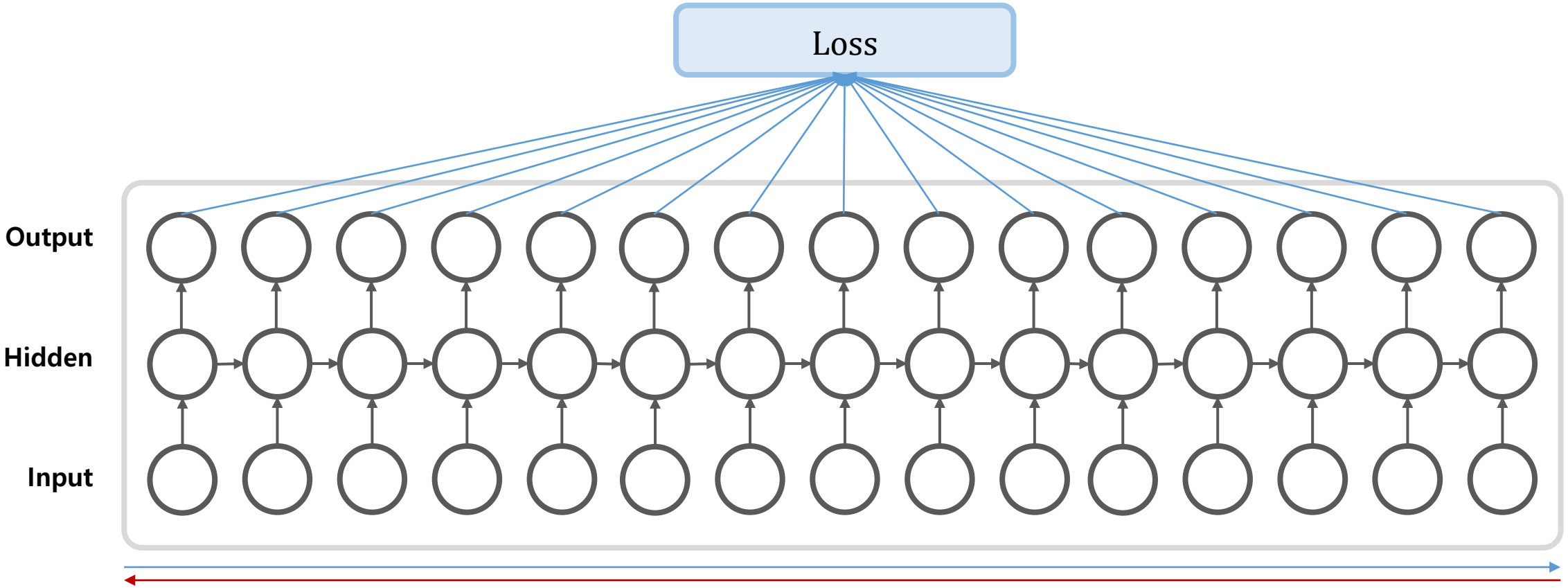
- 실제 시간 Step 별로 역순으로 Backpropagation이 실행됨

Backpropagation



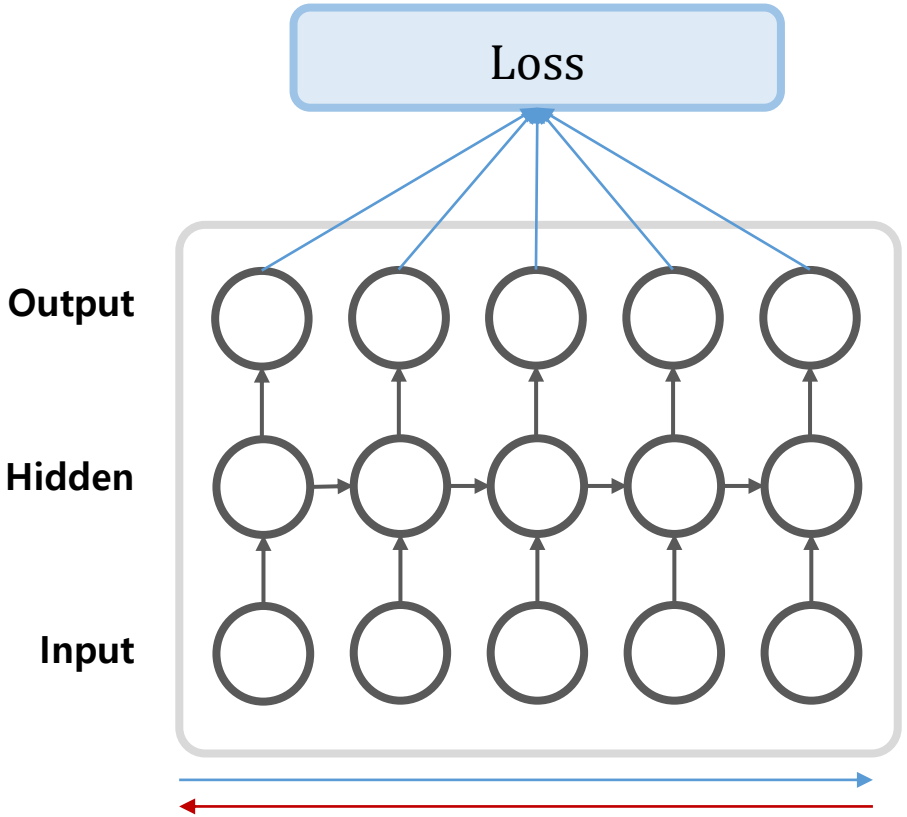
- Hidden Node는 다음 단계에서 받은 Gradient와 출력에서 받은 Gradient를 더해서 사용

BPTT : Backpropagation Through Time



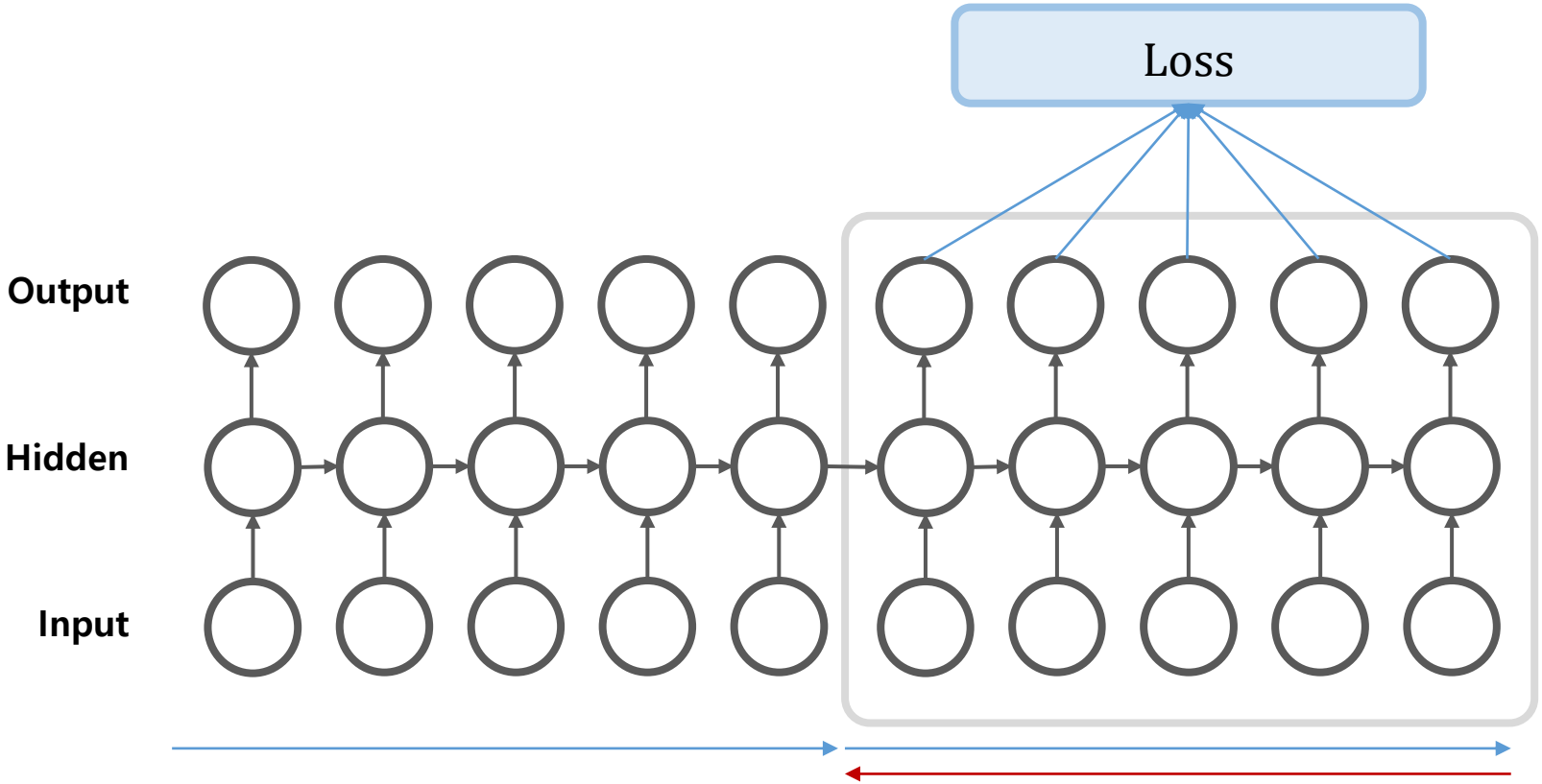
Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient

Truncated BPTT



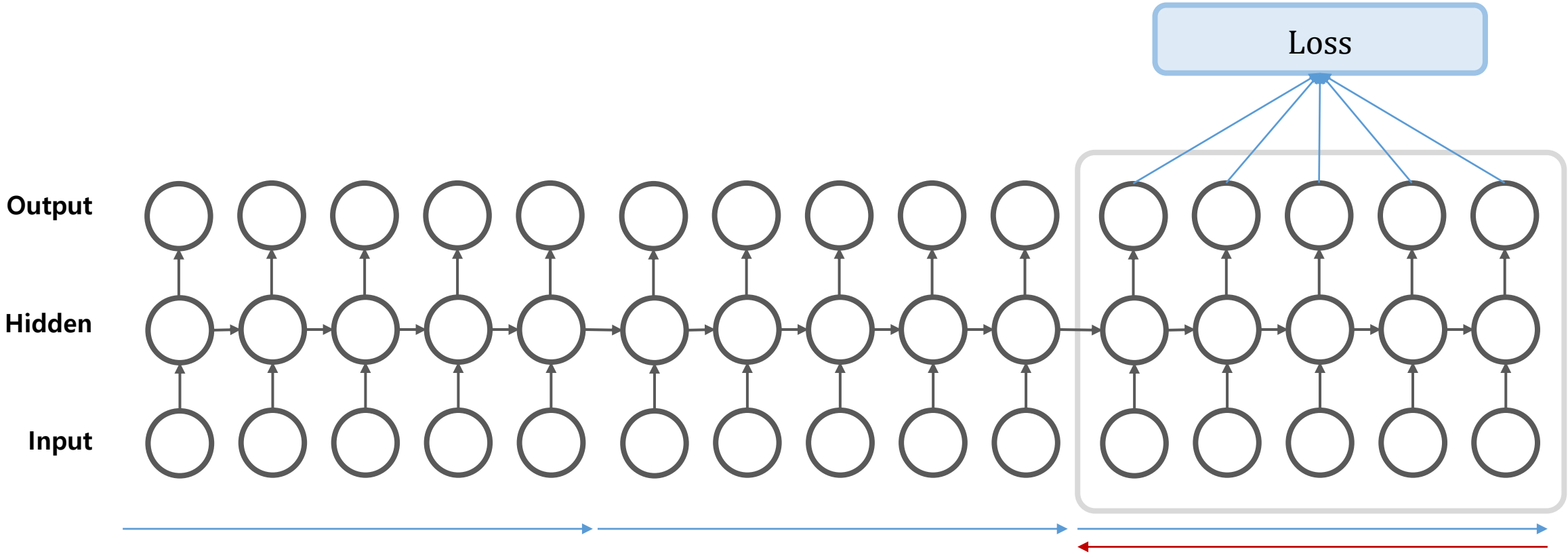
Run forward and backward through chunks of the sequence instead of whole sequence

Truncated BPTT

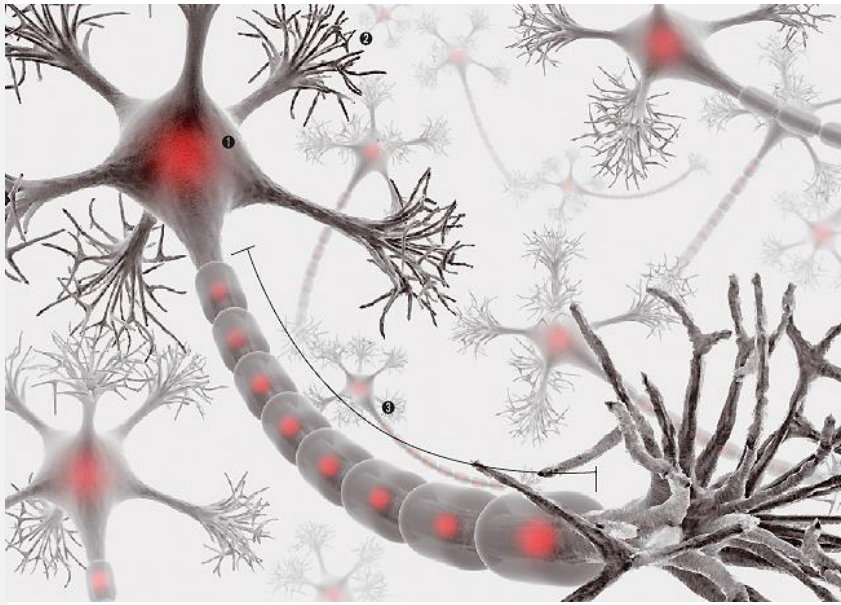


Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps

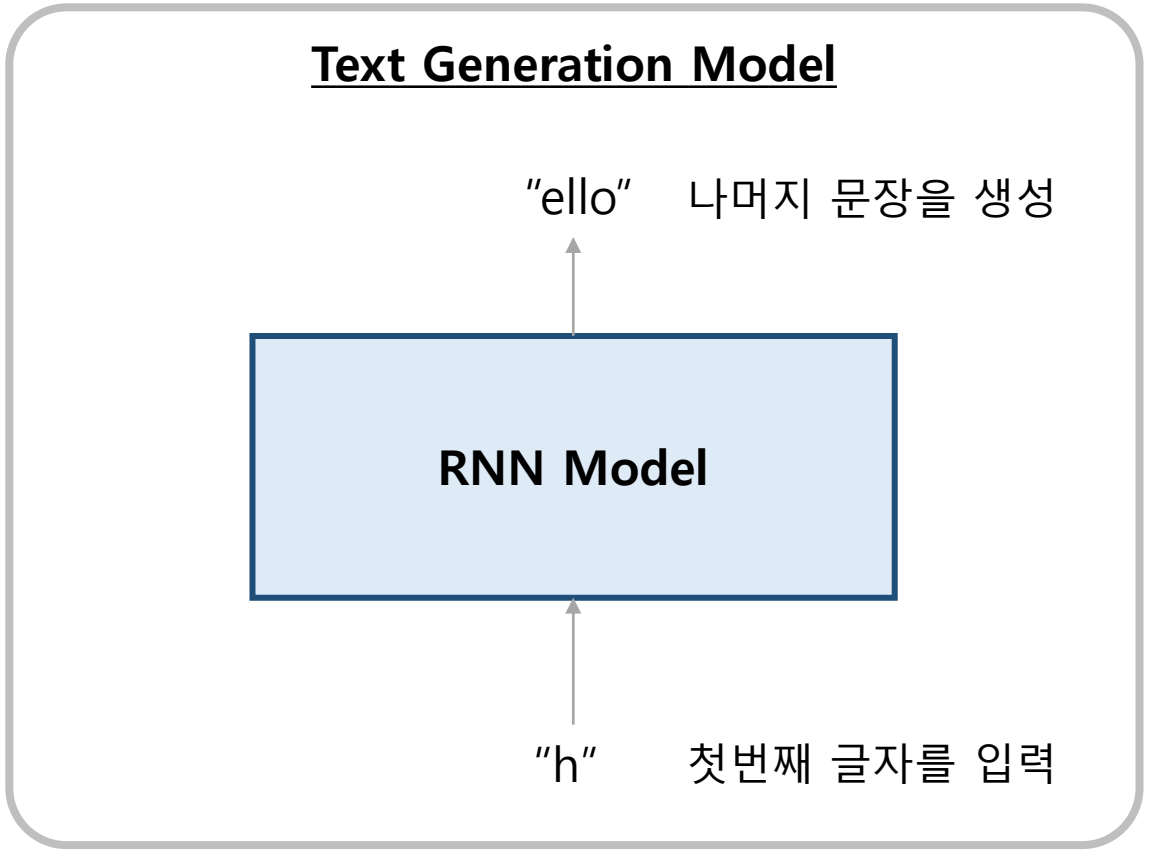
Truncated BPTT



4 RNN 예제 - Language Model



Character-level Language Model



훈련 데이터셋과 같은 스타일의 문자열을 생성하는 모델로 글자 단위로 훈련

Example training sequence: "hello"

Vocabulary: [h,e,l,o]

글자 or 단어 사전

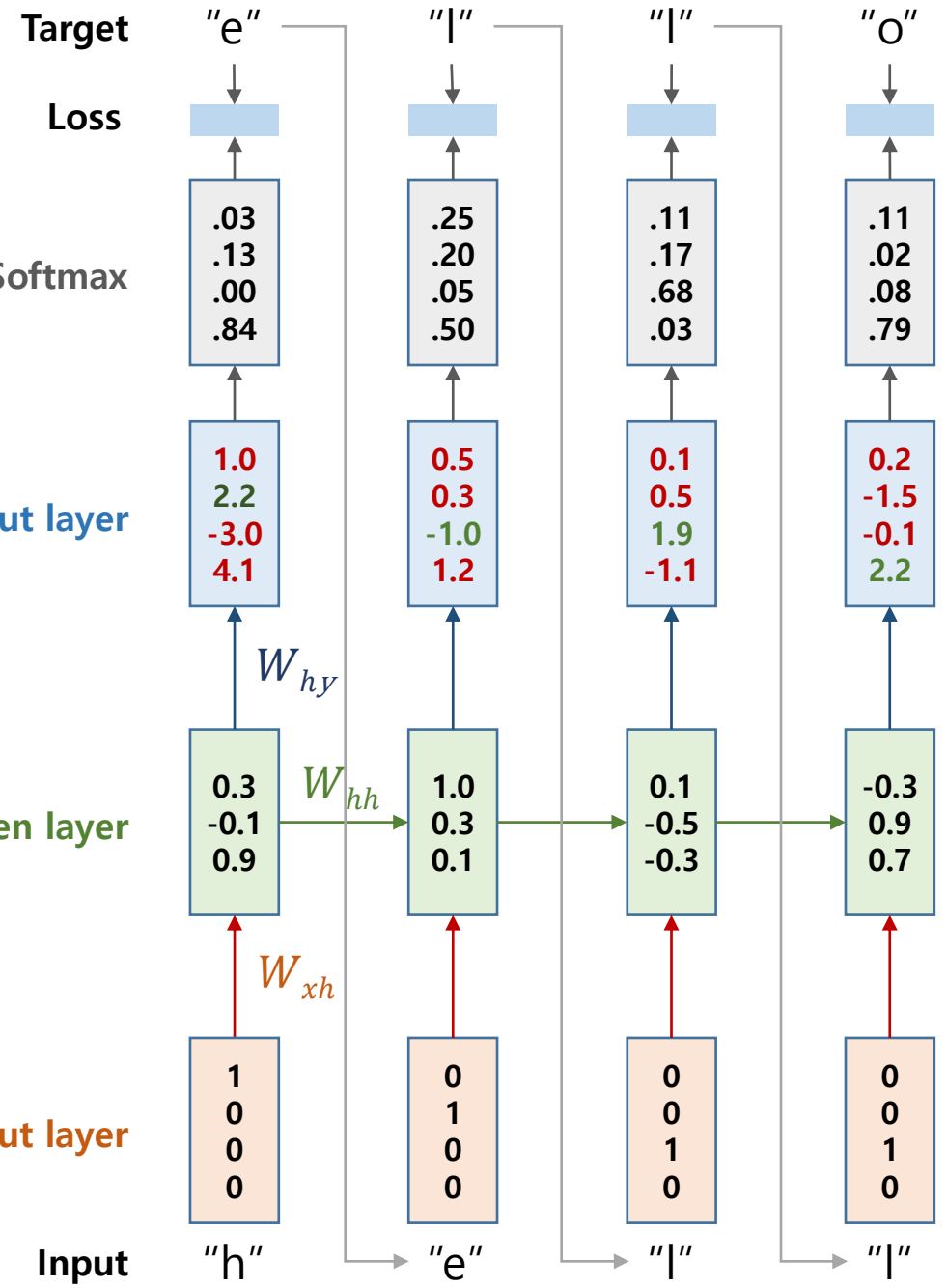
Training Time

Example training sequence: “hello”
Vocabulary: [h,e,l,o]

첫번째 글자 “h”에 대해서 “e”가 예측되어야 하지만 “o”가 예측되었음

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

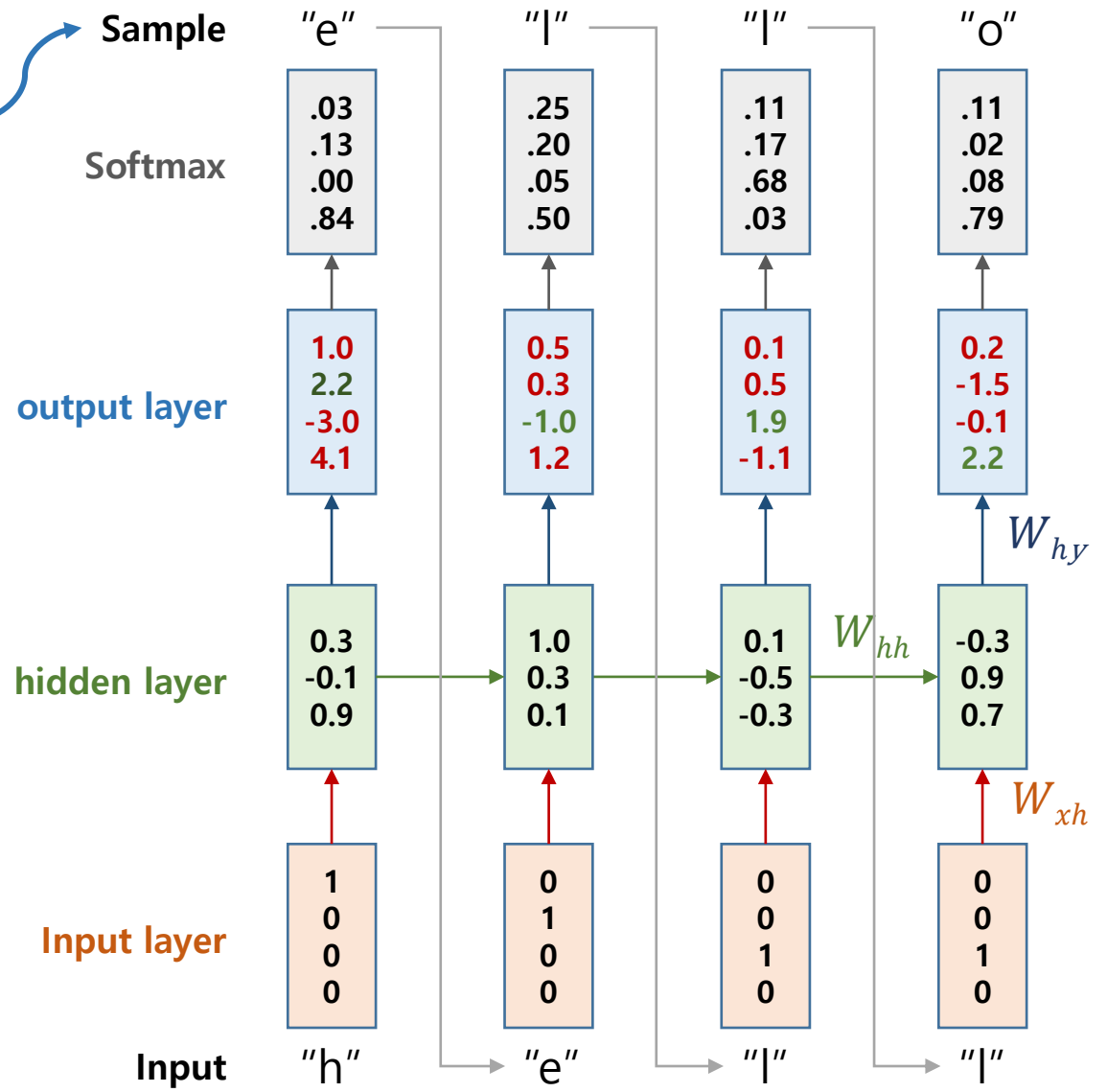
네 개의 vocabulary [h,e,l,o]에 대해 각각 one-hot coding 형태로 변환



Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Test Time

- 테스트 시에는 점수를 확률로 변환하여 샘플링
- 샘플링을 함으로서 모델의 diversity를 만듬

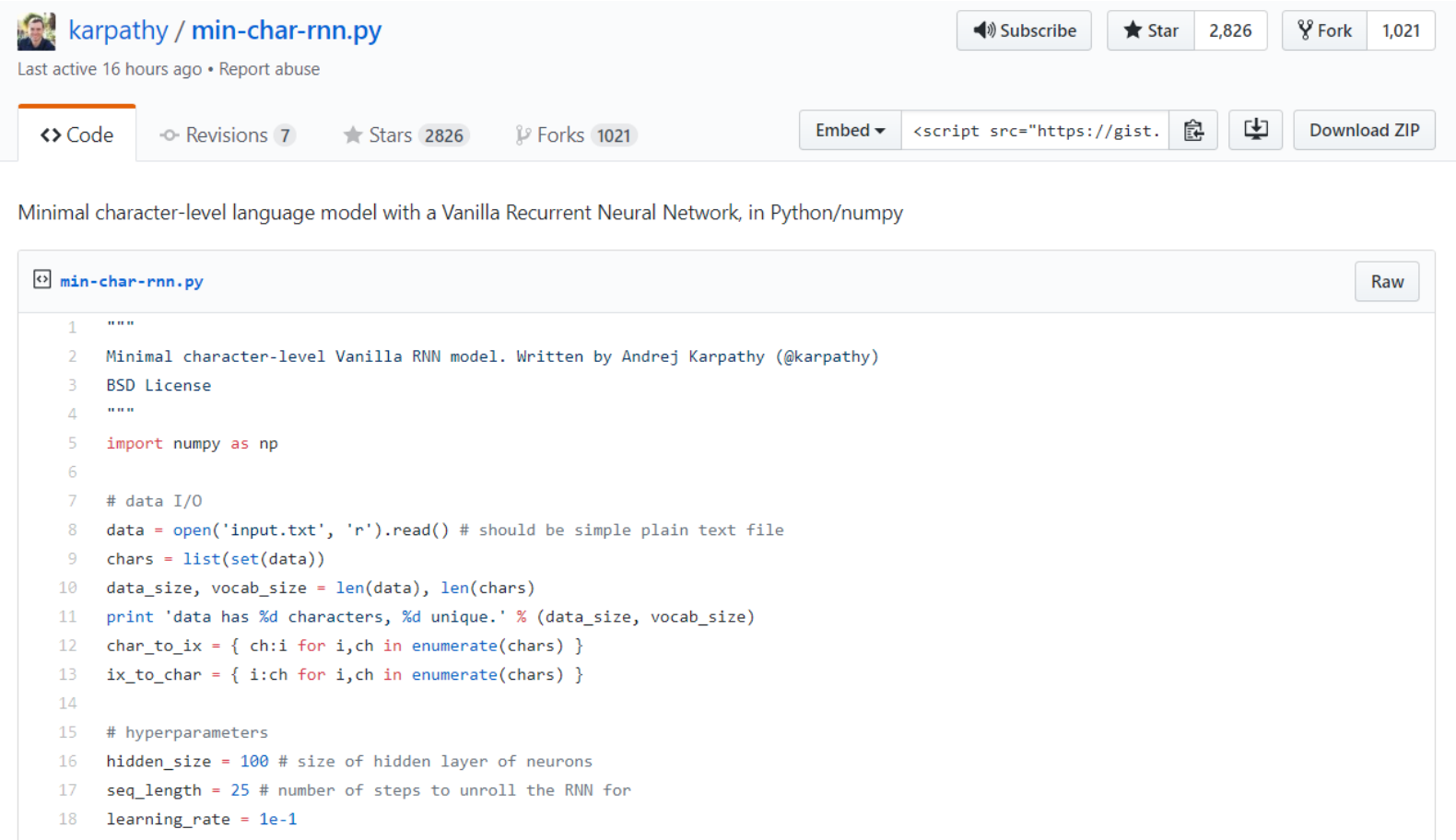


Example training sequence: "hello"
Vocabulary: [h,e,l,o]

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Python 코드

min-char-rnn.py gist: 112 lines of Python



karpathy / min-char-rnn.py

Last active 16 hours ago • Report abuse

Subscribe | Star 2,826 | Fork 1,021

Code | Revisions 7 | Stars 2826 | Forks 1021

Embed <script src="https://gist. | Download ZIP

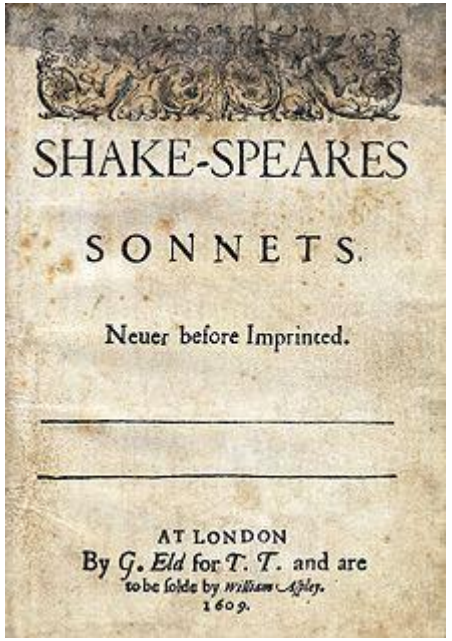
Minimal character-level language model with a Vanilla Recurrent Neural Network, in Python/numpy

min-char-rnn.py — Raw

```
"""
Minimal character-level Vanilla RNN model. Written by Andrej Karpathy (@karpathy)
BSD License
"""
import numpy as np

# data I/O
data = open('input.txt', 'r').read() # should be simple plain text file
chars = list(set(data))
data_size, vocab_size = len(data), len(chars)
print 'data has %d characters, %d unique.' % (data_size, vocab_size)
char_to_ix = { ch:i for i,ch in enumerate(chars) }
ix_to_char = { i:ch for i,ch in enumerate(chars) }

# hyperparameters
hidden_size = 100 # size of hidden layer of neurons
seq_length = 25 # number of steps to unroll the RNN for
learning_rate = 1e-1
```

https://gist.github.com/karpathy/d4dee566867f8291f086

THE SONNETS



by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou, contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fuel,
Making a famine where abundance lies,
Thyself thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
  Pity the world, or else this glutton be,
  To eat the world's due, by the grave and thee.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's proud livery so gazed on now,
Will be a tatter'd weed of small worth held:
Then being asked, where all thy beauty lies,
Where all the treasure of thy lusty days;
To say, within thine own deep sunken eyes,
Were an all-eating shame, and thriftless praise.
How much more praise deserv'd thy beauty's use,
If thou couldst answer 'This fair child of mine
Shall sum my count, and make my old excuse,'
Proving his beauty by succession thine!
  This were to be new made when thou art old,
  And see thy blood warm when thou feel'st it cold.

- 유럽의 정형시
- 소네트(10개의 음절로 구성되는 시행 14개가 일정한 운율로 이어지는 14행시)

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

THE SONNETS

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.
```

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

THE SONNETS

```
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.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

```
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.
```

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

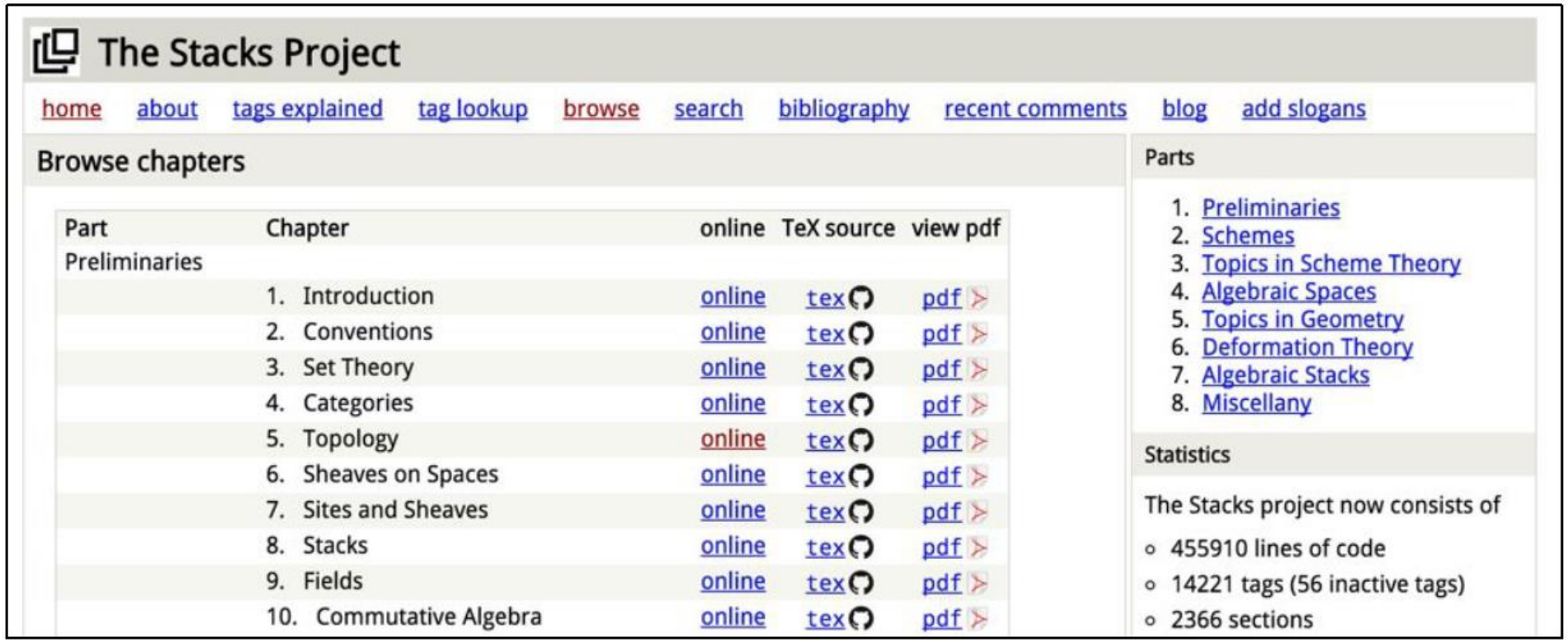
THE SONNETS

```
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.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

```
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.
```

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

The Stacks Project: open source algebraic geometry textbook



Training Set

http://stacks.math.columbia.edu/
The stacks project is licensed under the GNU Free Documentation License

The Stacks Project

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariciy in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M^\bullet} = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ??. It may replace S by $X_{spaces,étale}$ which gives an open subspace of X and T equal to S_{Zar}, see Descent, Lemma ??. Namely, by Lemma ?? we see that R is geometrically regular over S.

Lemma 0.1. *Assume (3) and (3) by the construction in the description.*

Suppose $X = \lim |X|$ *(by the formal open covering* X *and a single map* $\underline{Proj}_X(\mathcal{A}) =$ $\mathrm{Spec}(B)$ *over* U *compatible with the complex*

$$Set(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{Z/X}$ *is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If* T *is surjective we may assume that* T *is connected with residue fields of* S*. Moreover there exists a closed subspace* $Z \subset X$ *of* X *where* U *in* X' *is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem*

(1) f *is locally of finite type. Since* $S = \mathrm{Spec}(R)$ *and* $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\ldots,n} U_i$ be the scheme X over S at the schemes $X_i \to X$ and $U = \lim_i X_i$. □

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\ldots,0}$.

Lemma 0.2. *Let* X *be a locally Noetherian scheme over* S, $E = \mathcal{F}_{X/S}$. *Set* $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. *Since* $\mathcal{I}^n \subset \mathcal{I}^n$ *are nonzero over* $i_0 \leq \mathfrak{p}$ *is a subset of* $\mathcal{J}_{n,0} \circ \overline{A}_2$ *works.*

Lemma 0.3. *In Situation ??. Hence we may assume* $\mathfrak{q}' = 0$.

Proof. We will use the property we see that $\mathfrak{p}$ is the mext functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F-algebra where δ_{n+1} is a scheme over S. □

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

The Stacks Project

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram

$$\begin{array}{ccc} \mathcal{S} & \longrightarrow & \\ \downarrow & & \\ \xi & \longrightarrow & \mathcal{O}_{X'} \\ & & \uparrow \\ \text{gor}_s & & \end{array} \qquad \begin{array}{ccc} = \alpha' & \longrightarrow & \\ \updownarrow & & \\ = \alpha' & \longrightarrow & \alpha \end{array} \qquad \begin{array}{c} X \\ \downarrow \end{array}$$

$$\text{Spec}(K_\psi) \qquad \text{Mor}_{Sets} \quad \text{d}(\mathcal{O}_{\mathcal{X}_{Xfk}}, \mathcal{G})$$

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \text{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

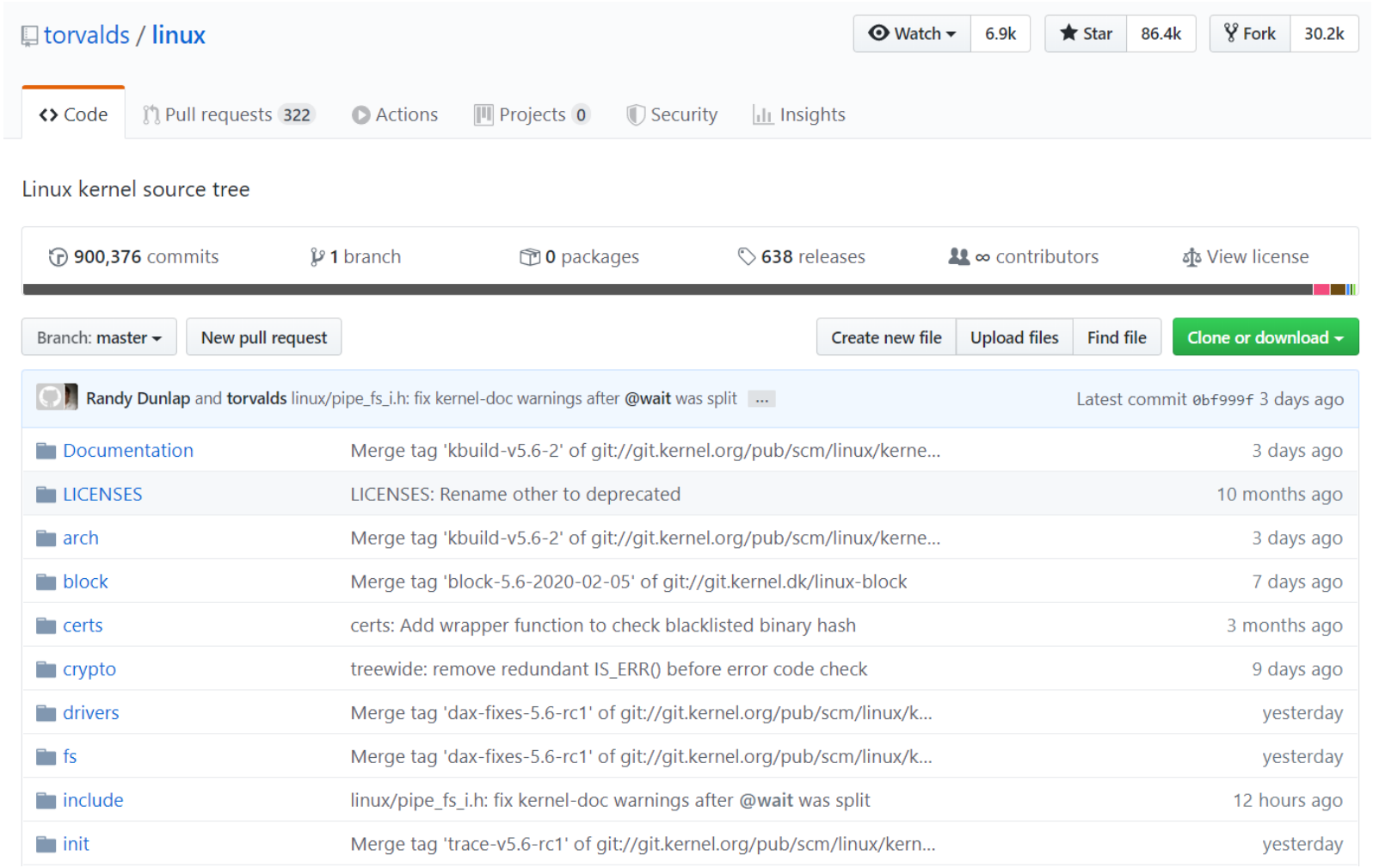
$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad -1(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_t}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\mathbb{V}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Linux Kernel



https://github.com/torvalds/linux

Linux Kernel

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 & 0x00000000ffffff8) & 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 ");
}
```

GNU License Generation

```c
/*
 *  Copyright (c) 2006-2010, Intel Mobile Communications.  All rights reserved.
 *
 *   This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License version 2 as published by
 * the Free Software Foundation.
 *
 *        This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 *   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *
 *  GNU General Public License for more details.
 *
 *   You should have received a copy of the GNU General Public License
 *    along with this program; if not, write to the Free Software Foundation,
 *  Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

#include <linux/kexec.h>
#include <linux/errno.h>
#include <linux/io.h>
#include <linux/platform_device.h>
#include <linux/multi.h>
#include <linux/ckevent.h>

#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>
```

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Linux Kernel

```
#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;
}
```

Generated
C code

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

RNN Interpretability

VISUALIZING AND UNDERSTANDING RECURRENT NETWORKS

Andrej Karpathy* **Justin Johnson*** **Li Fei-Fei**
Department of Computer Science, Stanford University
{karpathy, jcjohns, feifeili}@cs.stanford.edu

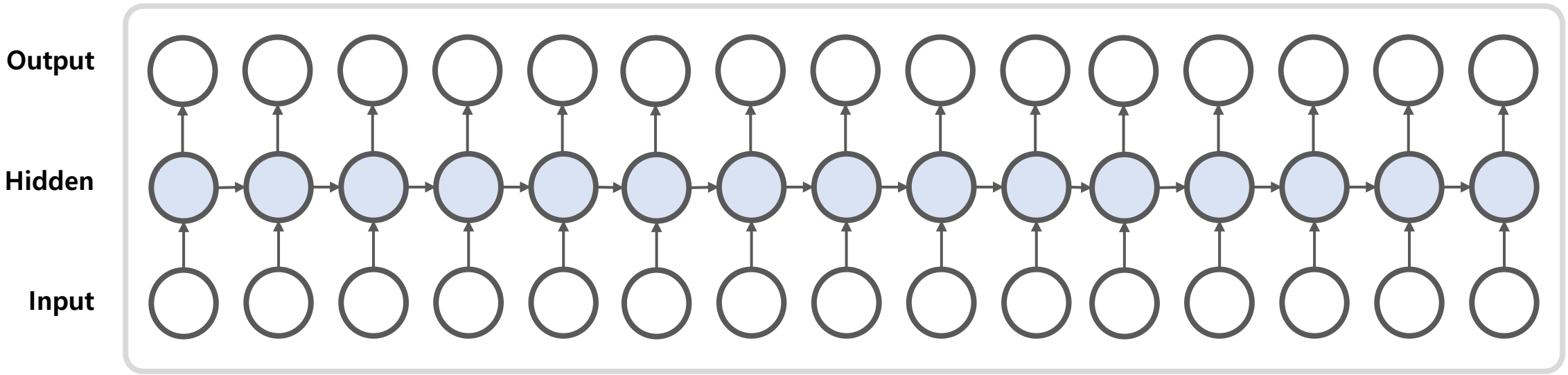
ABSTRACT

Recurrent Neural Networks (RNNs), and specifically a variant with Long Short-Term Memory (LSTM), are enjoying renewed interest as a result of successful applications in a wide range of machine learning problems that involve sequential data. However, while LSTMs provide exceptional results in practice, the source of their performance and their limitations remain rather poorly understood. Using character-level language models as an interpretable testbed, we aim to bridge this gap by providing an analysis of their representations, predictions and error types. In particular, our experiments reveal the existence of interpretable cells that keep track of long-range dependencies such as line lengths, quotes and brackets. Moreover, our comparative analysis with finite horizon n-gram models traces the source of the LSTM improvements to long-range structural dependencies. Finally, we provide analysis of the remaining errors and suggests areas for further study.

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

RNN Interpretability

RNN의 Hidden Vector가 입력 데이터에 대한 정보를 갖고 있는지 확인



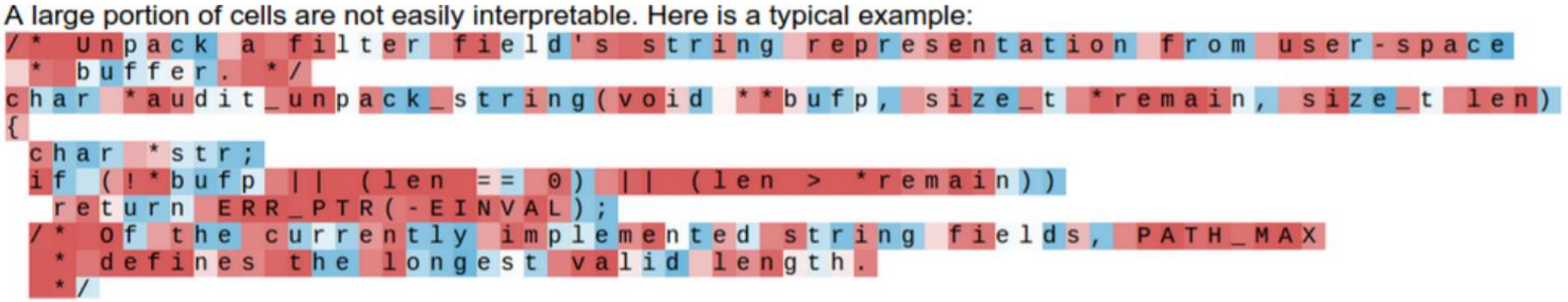
- Character Level 모델로 실험
- Hidden Vector의 각 Element가 Sequence를 처리할 때 각 Step 별로 반응하는 크기를 측정
- 각 Hidden state가 무엇을 보는지 그리고 문법적으로 해석가능한 의미를 갖는지 확인
- 데이터셋 : War and Peace 3,258,246 문자, Linux Kernel 6,206,996 문자

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

RNN Interpretability

이 경우 랜덤하게 반응하기 때문에 의미를 부여할 수 없음

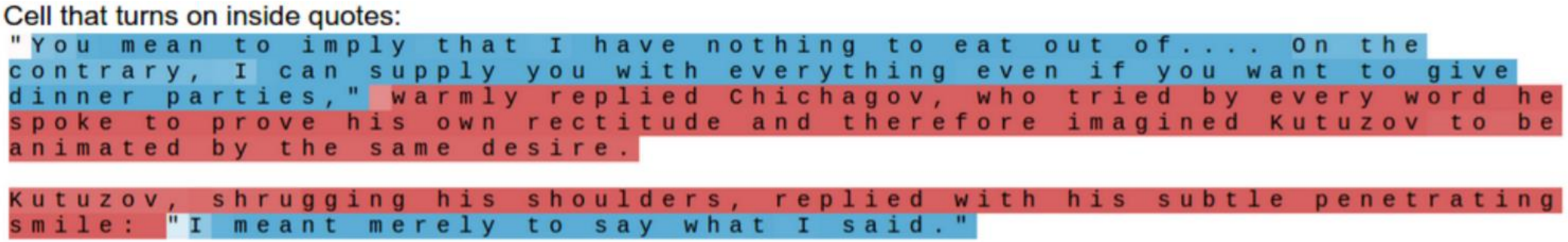


- Several examples of cells with interpretable activations discovered in our best Linux Kernel and War and Peace LSTMs.
- Text color corresponds to tanh(c), where -1 is red and +1 is blue

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

RNN Interpretability

인용문을 탐지하는 셀

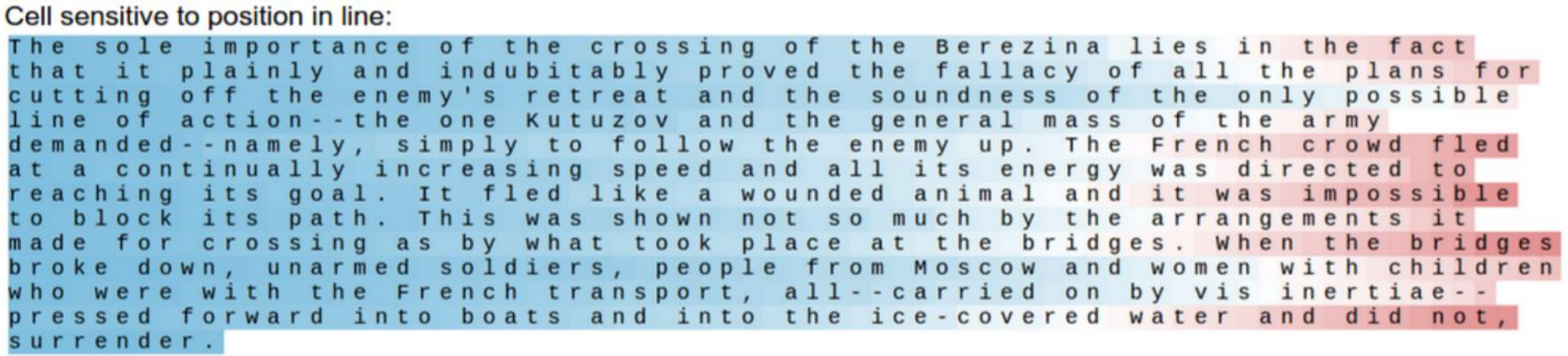


- "가 나오면 켜지고 두번째 "가 나오면 꺼짐

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

RNN Interpretability

라인 길이를 추적하는 셀



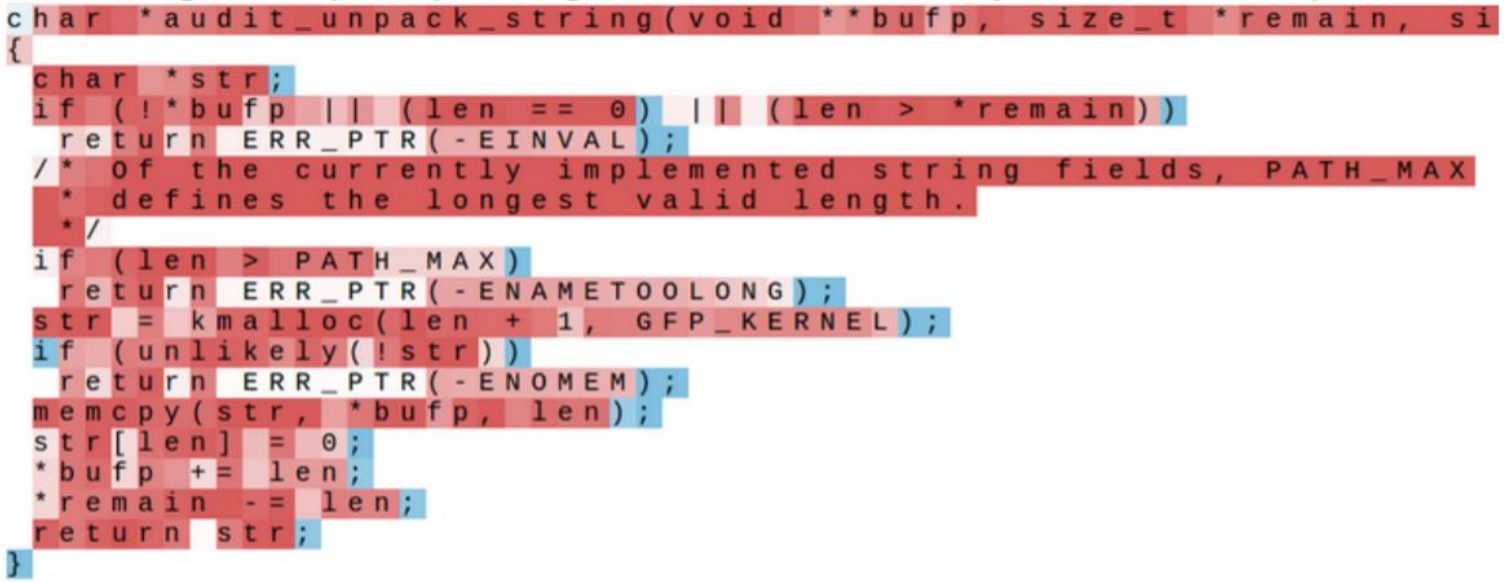
- 처음에 높은 값으로 시작해서 새로운 라인이 나올 때까지 값이 점점 감소하면서 빨강색으로 변함

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

RNN Interpretability

새로운 라인을 예측 하는데 도움이 될 셀

Cell that might be helpful in predicting a new line. Note that it only turns on for some ")":



```
char *audit_unpack_string(void **bufp, size_t *remain, si
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
  if (len > PATH_MAX)
    return ERR_PTR(-ENAMETOOLONG);
  str = kmalloc(len + 1, GFP_KERNEL);
  if (unlikely(!str))
    return ERR_PTR(-ENOMEM);
  memcpy(str, *bufp, len);
  str[len] = 0;
  *bufp += len;
  *remain -= len;
  return str;
}
```

-), ;, }을 만나면 켜짐

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

RNN Interpretability

If statement의 조건을 탐지하는 셀

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
        siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

RNN Interpretability

주석을 탐지하는 셀

Cell that turns on inside comments and quotes:

```
/*  Duplicate LSM field information.  The lsm_rule is opaque, so
 *  re-initialized.  */
static inline int audit_dupe_lsm_field(struct audit_field *df,
           struct audit_field *sf)
{
  int ret = 0;
  char *lsm_str;
  /*  our own copy of lsm_str */
  lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
  if (unlikely(!lsm_str))
    return -ENOMEM;
  df->lsm_str = lsm_str;
  /*  our own (refreshed) copy of lsm_rule */
  ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
                 (void **)&df->lsm_rule);
  /*  Keep currently invalid fields around in case they
   *  become valid after a policy reload.  */
  if (ret == -EINVAL) {
    pr_warn("audit rule for LSM \'%s\' is invalid\n",
      df->lsm_str);
    ret = 0;
  }
  return ret;
}
```

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

RNN Interpretability

Indentation Level을 세는 셀

Cell that is sensitive to the depth of an expression:

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

Visualizing and Understanding Recurrent Network, Andrej Karpathy, Justin Johnson, Li Fei-Fei

5 RNN 예제 - Image Captioning

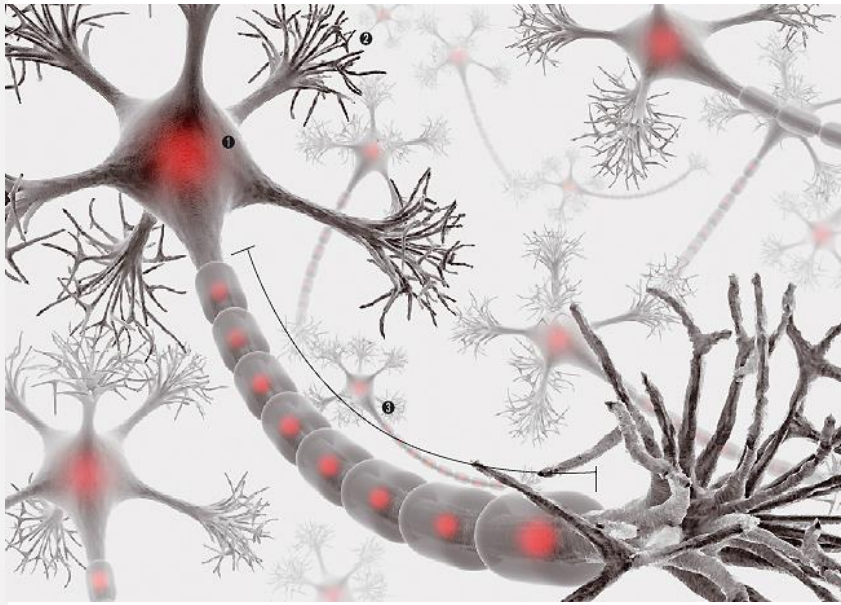
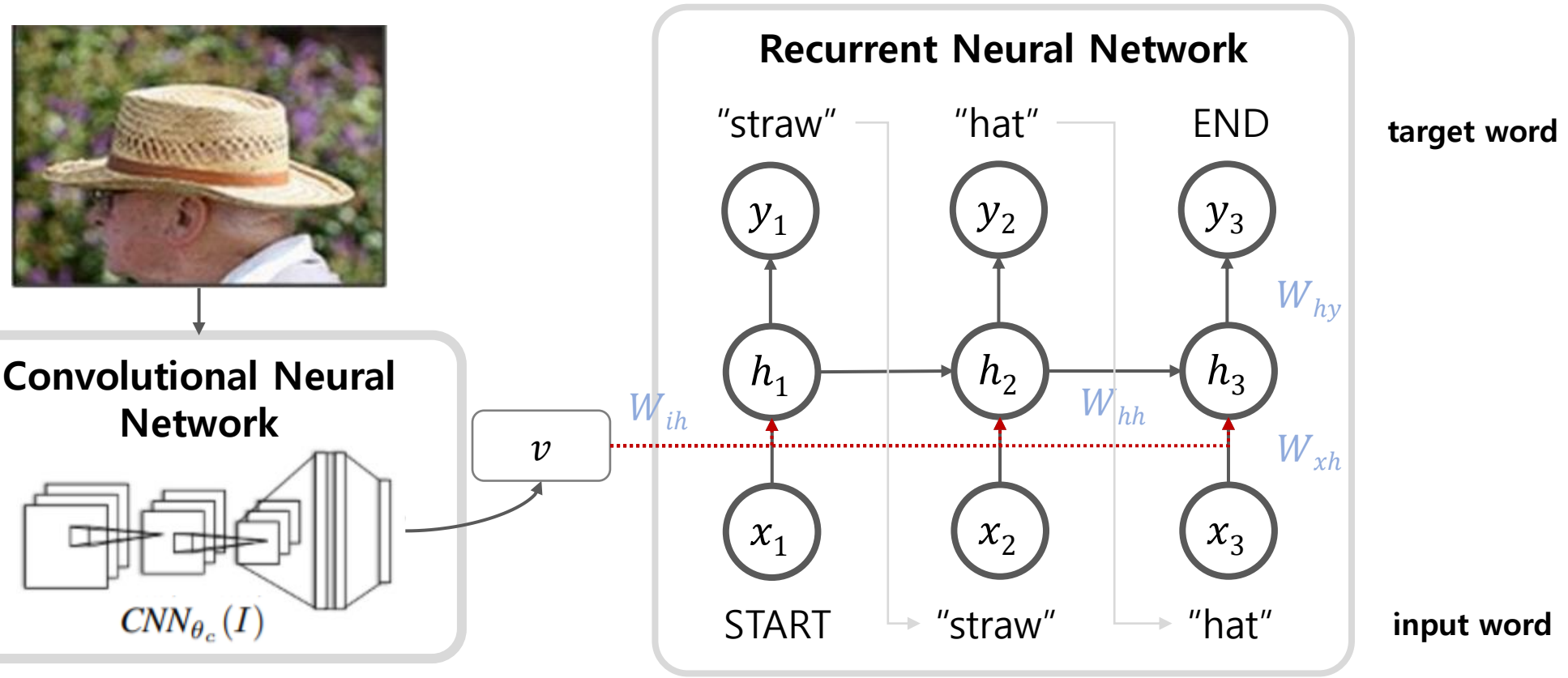


Image Captioning



- CNN은 Image Summary 정보를 생성해서 RNN에 전달
- RNN은 이전 time step의 context와 단어를 입력 받고 다음 단어에 대한 분포를 정의
- START와 END는 special token

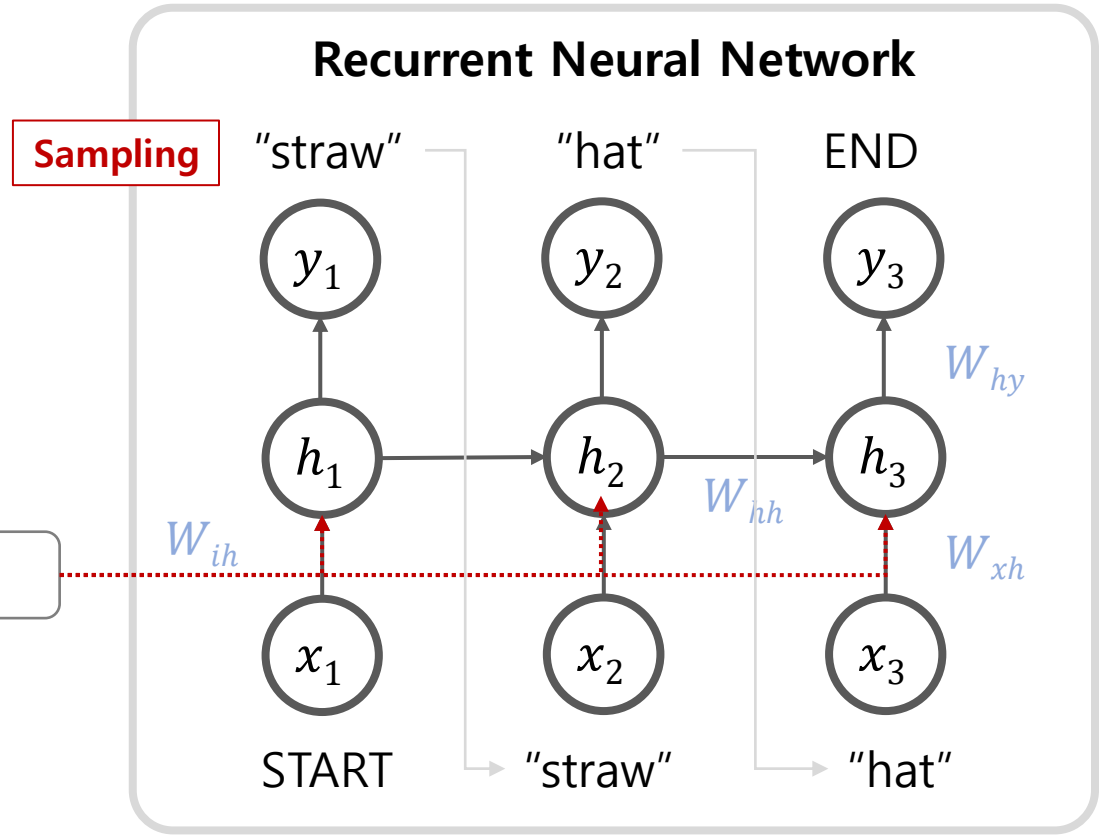
Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Test Time

Test Image

FC-4094 Layer에서
Image 정보 추출

v



<END> token을
샘플링하면 종료

$$h_t = \tanh(W_{hx} \cdot x_t + W_{hh} \cdot h_{t-1} + W_{ih} \cdot v)$$

$$y_t = \text{Softmax}(W_{hy} \cdot h_t + b_o)$$

Stanford CS231n: Convolutional Neural Networks for Visual Recognition

Example 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

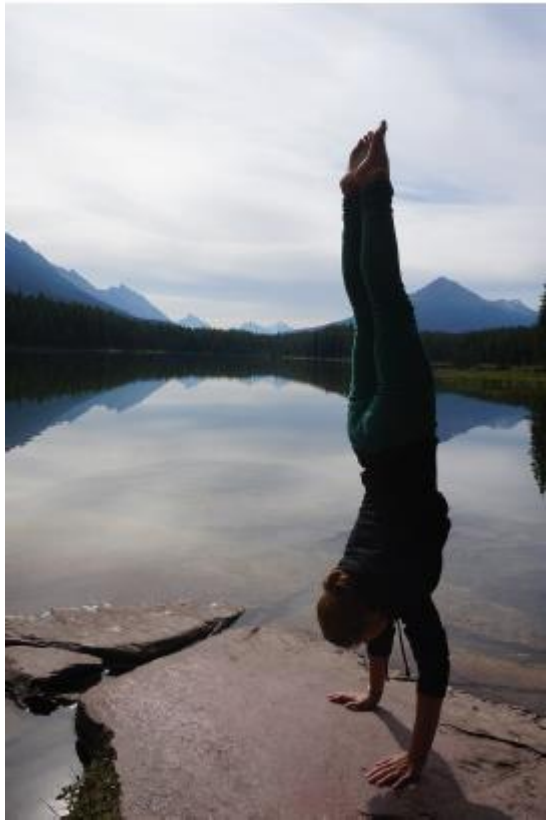
Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Failure Cases

A woman is holding a cat in her hand

A person holding a computer mouse on a desk

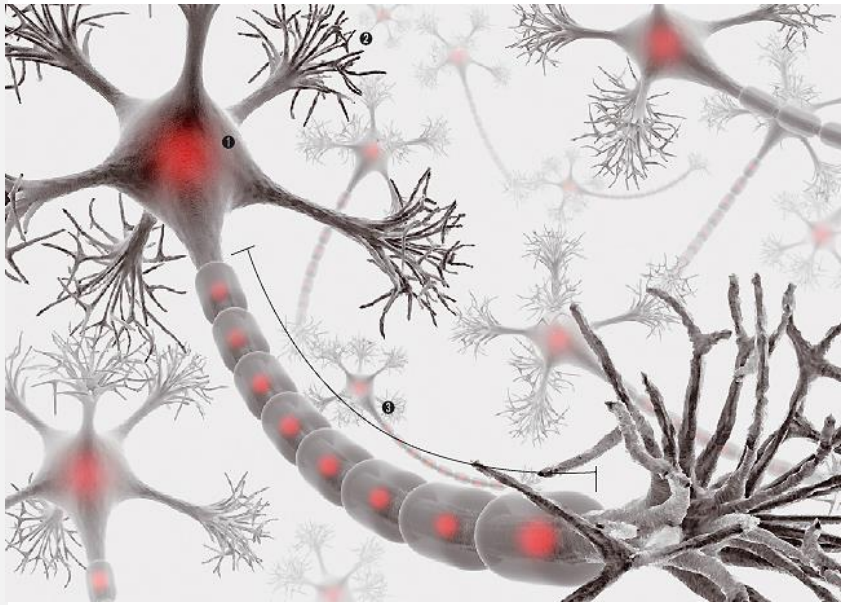


A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

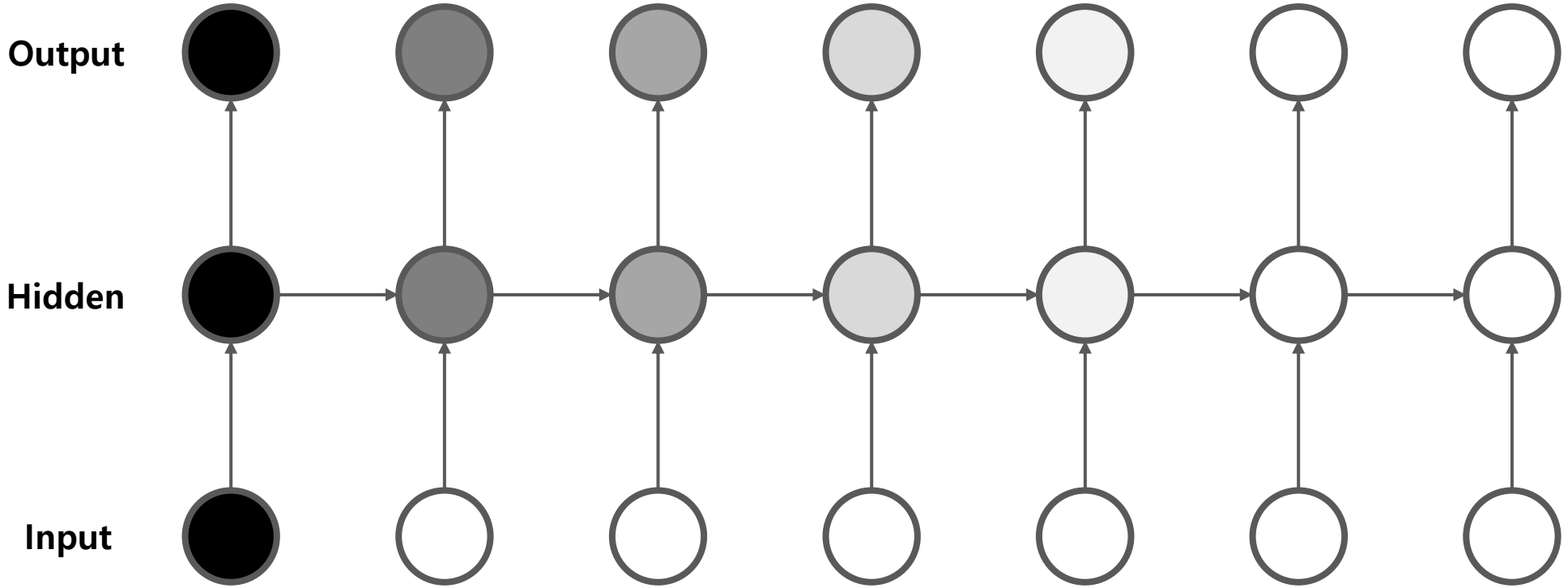
A man in a baseball uniform throwing a ball

6 LSTM/GRU



RNN Issue Long-Term Dependency

긴 문맥이 필요한 경우 멀리 떨어진 입력에 의존해야 하는데 값이 사리지는 현상 발생

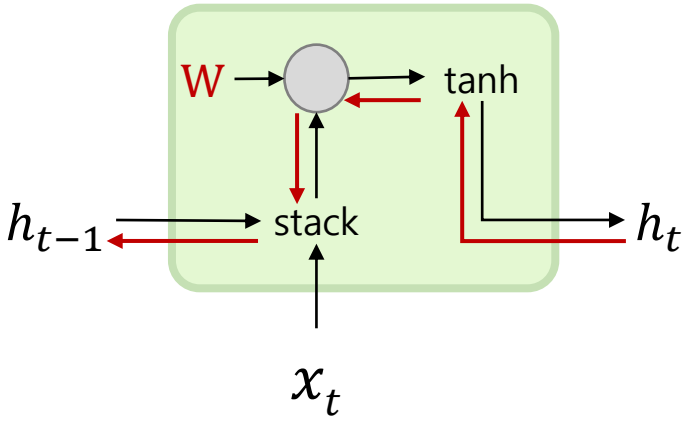


- Time step이 지나면서 입력 값의 영향이 점점 감소
- Sequence 길이가 길어질 경우 오래 전 입력 값의 정보를 제대로 반영하지 못함

RNN Issue

Gradient Vanishing/Exploding

Vanilla RNN



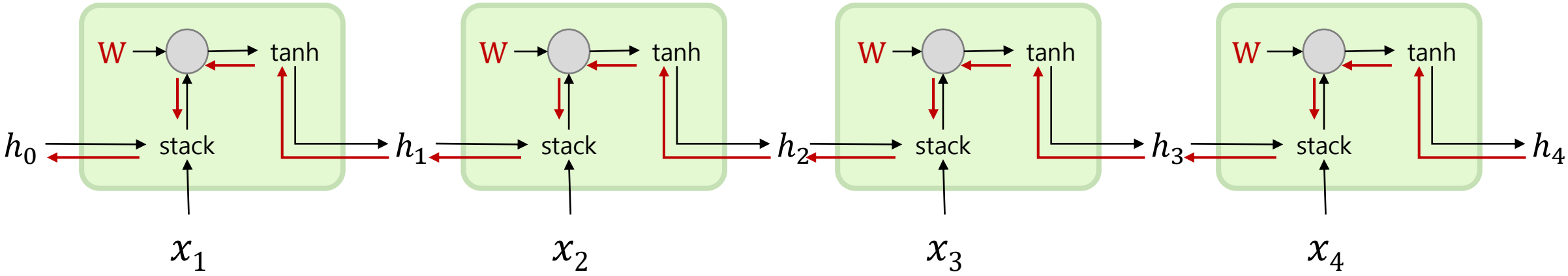
$$h_t = \tanh(W_{hh} \cdot h_{t-1} + W_{xh} \cdot x_t)$$

$$= \tanh\left((W_{hh} \quad W_{xh}) \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

$$= \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

h_t 에서 h_{t-1}로 Backpropagation 할 때 W가 곱해짐

RNN Issue Gradient Vanishing/Exploding



Vanilla RNN

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

tanh와 x_t를 생략하고 단순화

Forward Pass

$$h_t = W_{hh}^{\top} \cdot h_{t-1}$$

$$= W_{hh}^{\top} \cdot (W_{hh}^{\top} \cdot h_{t-2})$$

$$\cdots$$

$$= (W_{hh}^{t})^{\top} \cdot h_0$$

Backward Pass

$\frac{\partial h_t}{\partial h_{t-1}} = W_{hh}$ 이므로

$$\frac{\partial h_t}{\partial h_0} = \frac{\partial h_t}{\partial h_{t-1}} \cdot \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdot \cdots \cdot \frac{\partial h_1}{\partial h_0}$$

$$= W_{hh} \cdot W_{hh} \cdot \cdots \cdot W_{hh}$$

$$= W_{hh}^{t}$$

W가 반복적으로 곱해짐

RNN Issue Gradient Vanishing/Exploding

Gradient Vanishing과 Exploding이 왜 일어나는가?

Backward Pass

$W_{hh} = Q \wedge Q^{\intercal}$: eigenvalue decomposition (Q는 직교 행렬이고 $\wedge$는 대각 행렬)

$$\frac{\partial h_t}{\partial h_0} = {W_{hh}}^t$$

$$= (Q \wedge Q^{\intercal})^t$$

$$= Q^{\intercal} \wedge^t Q$$

sigular value의 t 제곱승

σ : Largest singular value

$$\begin{cases} \sigma > 1 & \text{Gradient Exploding} \\ \sigma < 1 & \text{Gradient Vanishing} \end{cases}$$

Gradient Exploding

Gradient Exploding이 일어나면 어떻게 해야 하는가?

σ : Largest singular value

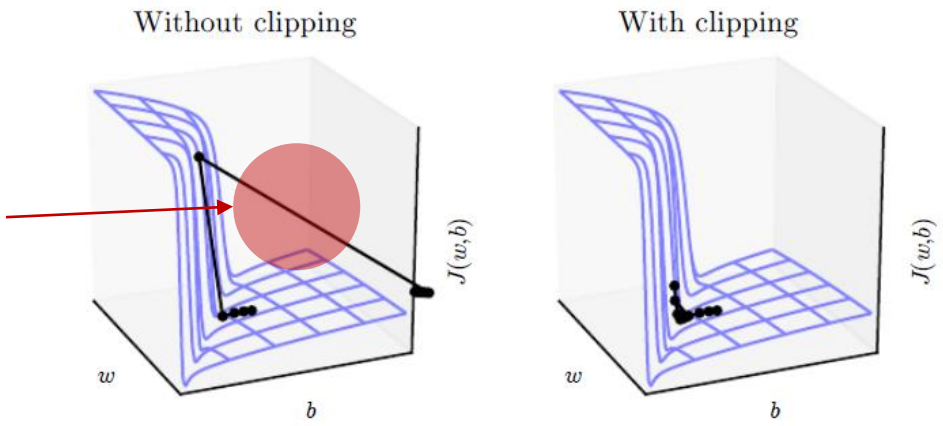
$$\begin{cases} \sigma > 1 & \textbf{Gradient Exploding} \\ \sigma < 1 & \text{Gradient Vanishing} \end{cases}$$

Gradient Clipping

- Gradient의 크기가 임계치 이상으로 커지면 그 미만이 되도록 Scaling 해 줌

$$\text{if } ||\boldsymbol{g}|| > v$$
$$\boldsymbol{g} \leftarrow \frac{\boldsymbol{g}v}{||\boldsymbol{g}||}$$

가파른 절벽을 만나서 너무 큰 gradient가 생겨서 parameter가 overshooting됨



Gradient Vanishing

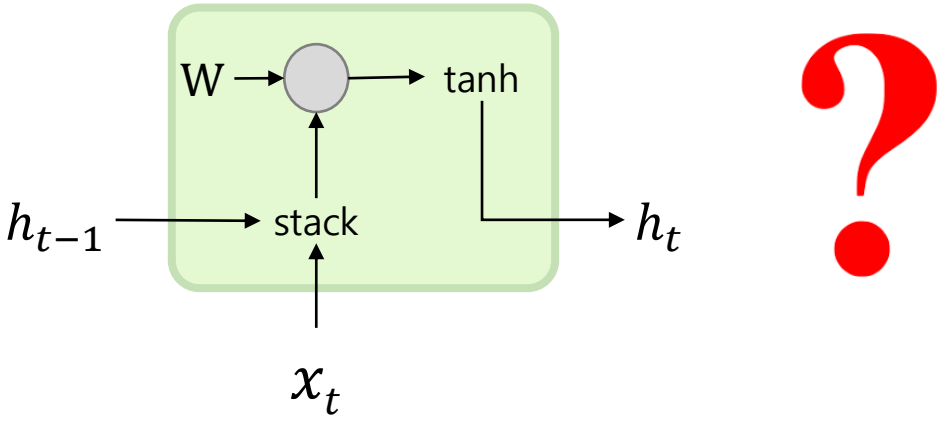
Gradient Vanishing이 일어나면 어떻게 해야 하는가?

σ : Largest singular value

$\sigma > 1$ Gradient Exploding

$\sigma < 1$ **Gradient Vanishing**

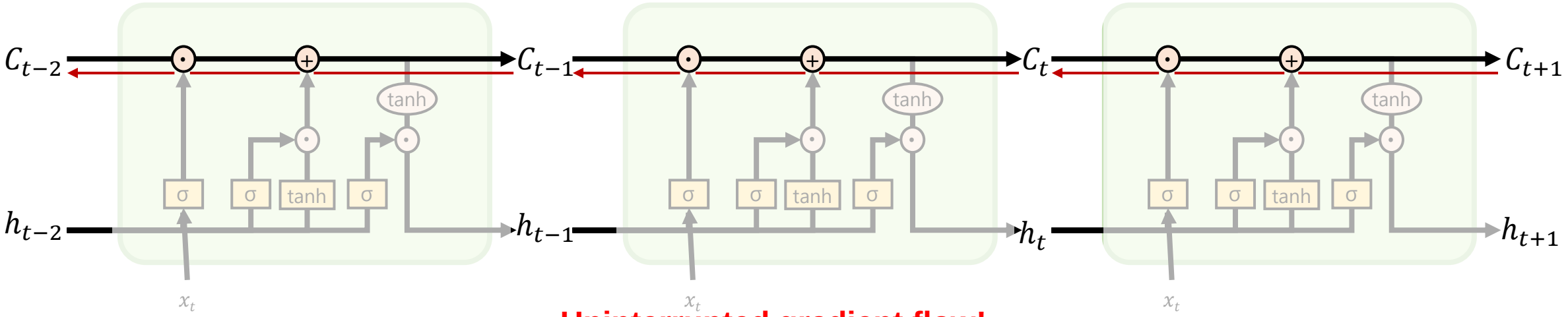
Vanilla RNN



RNN 구조를 바꿔보자!

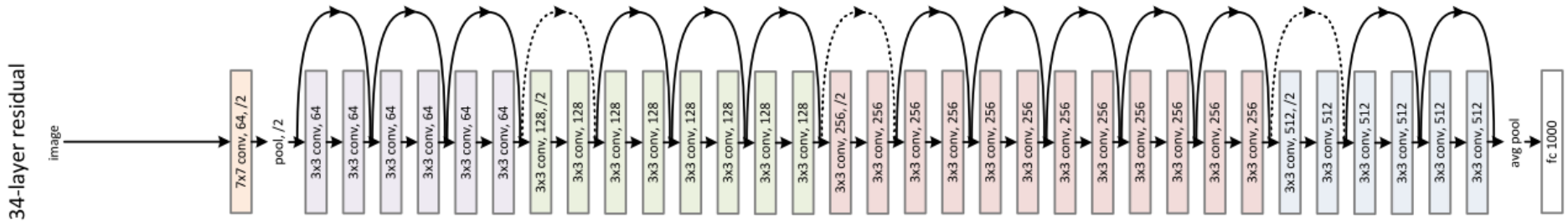
LSTM : Long Short Term Memory

Gradient Flow에 W가 곱해지지 않도록 구조 변경!



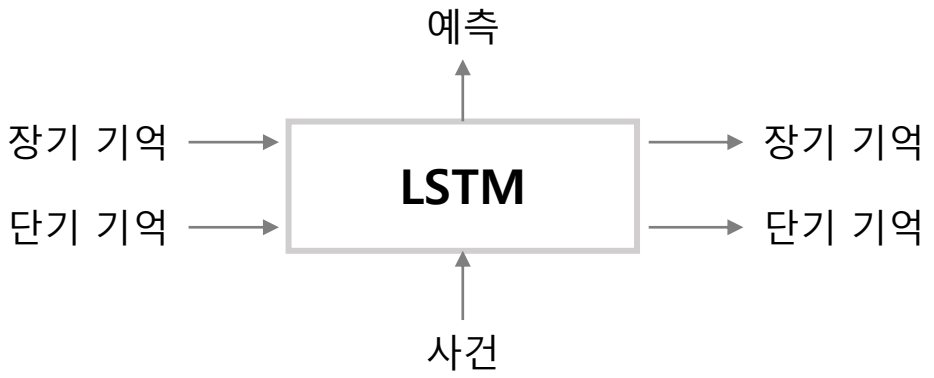
Uninterrupted gradient flow!

Gradient를 원활히 잘 흐르게 하는 방식이 ResNet의 Residual Connection과도 유사



LSTM : Long Short Term Memory

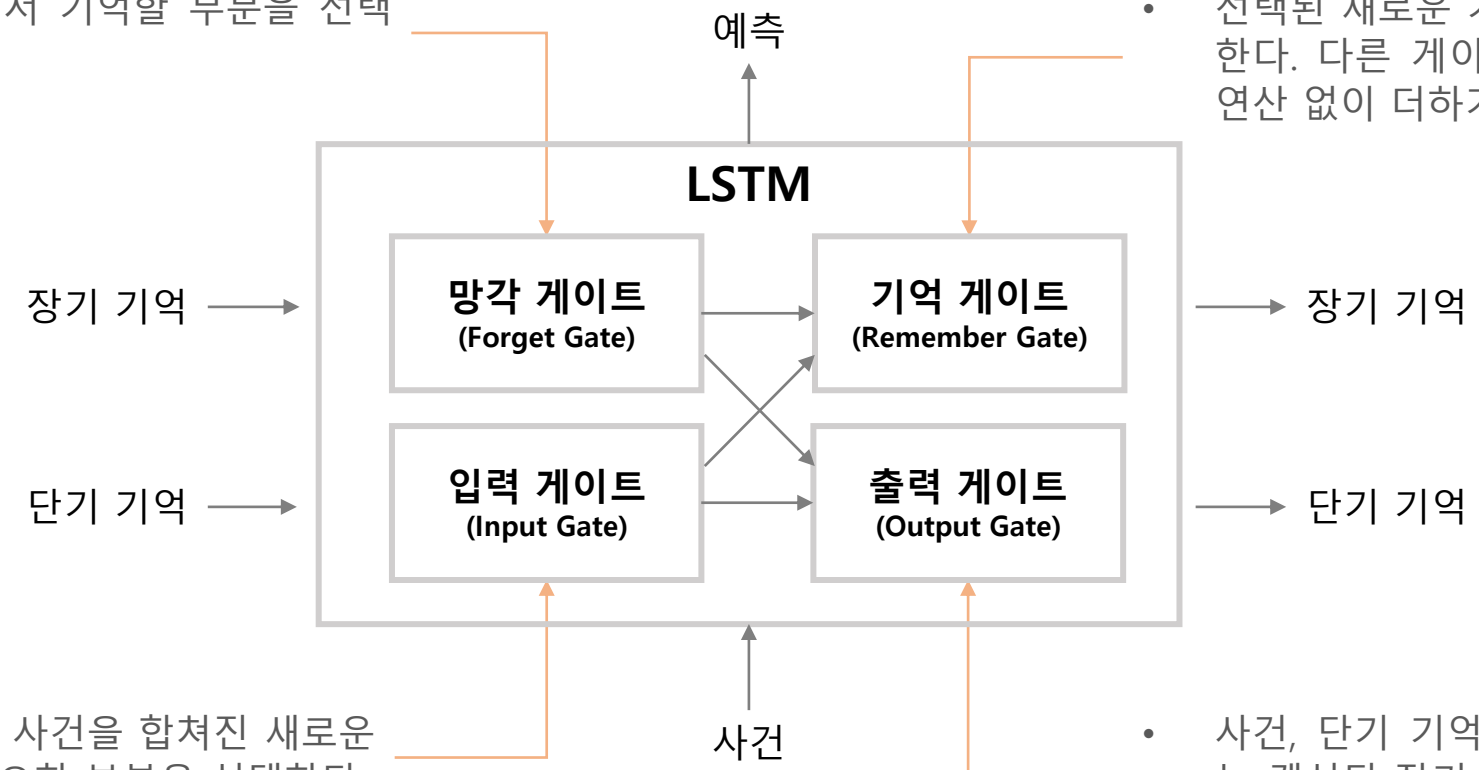
장기 기억과 단기 기억이 새로운 이벤트와 합쳐져서 갱신되는 방식



- 장기 기억은 오래 지속되도록, 단기 기억은 최근 사건을 중심으로 기억하도록 기억이 형성되는 과정이 분리된 구조를 갖는다.
- 장기 기억은 Cell State, 단기 기억은 Hidden State

LSTM : Long Short Term Memory

- 장기 기억 중 셀에서 기억할 부분을 선택한다.
- 선택된 새로운 기억으로 장기 기억을 갱신한다. 다른 게이트와 달리 별도의 게이트 연산 없이 더하기로 구현된다.

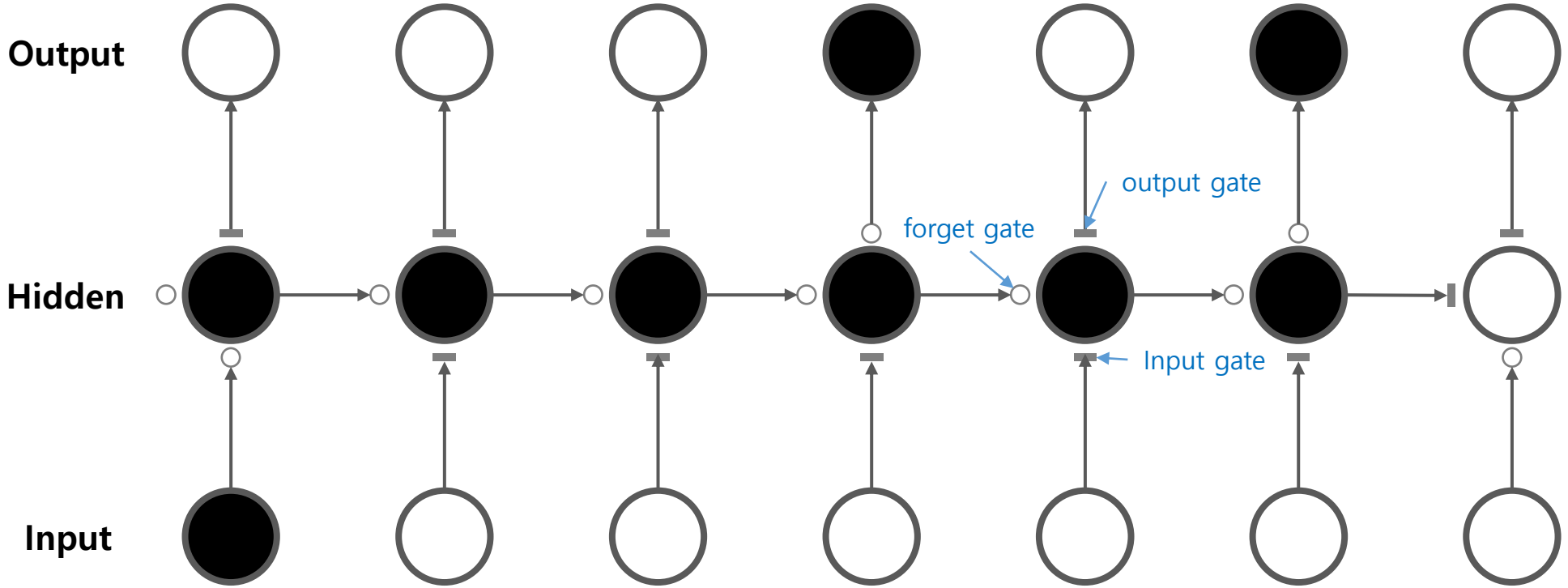


- 단기 기억과 새로운 사건을 합쳐진 새로운 기억에서 예측에 필요한 부분을 선택한다
- 사건, 단기 기억, 장기 기억이 연합되어 있는 갱신된 장기 기억에서 예측에 필요한 부분을 선택한다.

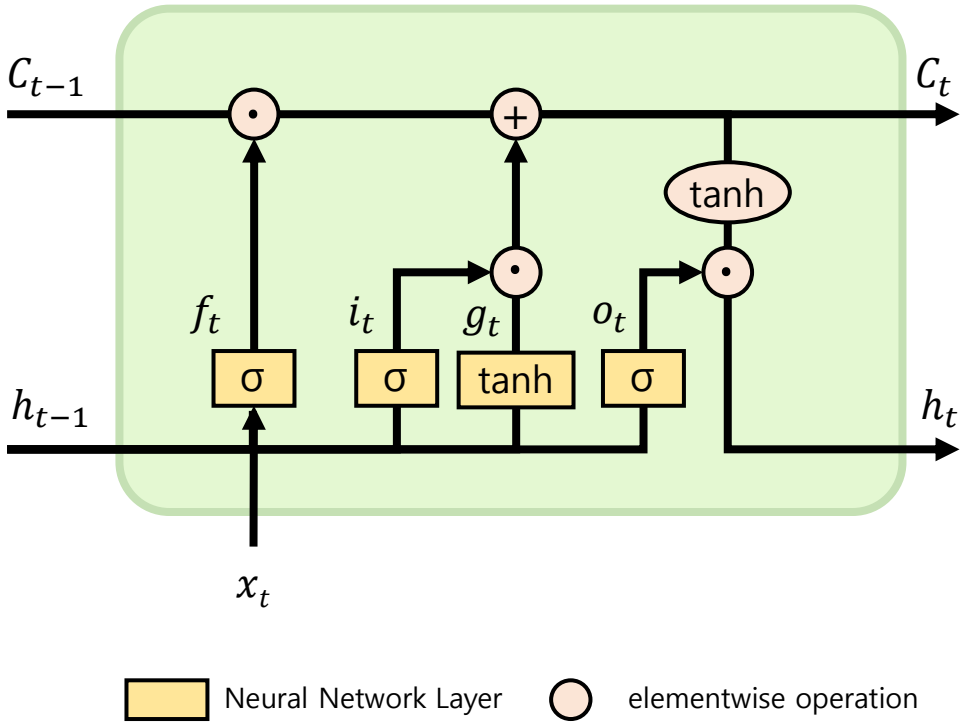
사건과 단기 기억, 장기 기억이 어느 정도 예측에 관여하는지는 LSTM의 게이트 구조로 조절

LSTM : Long Short Term Memory

장기 기억을 오래 기억할 수 있고 어느 부분을 기억할지를 선택할 수 있다!



LSTM : Long Short Term Memory

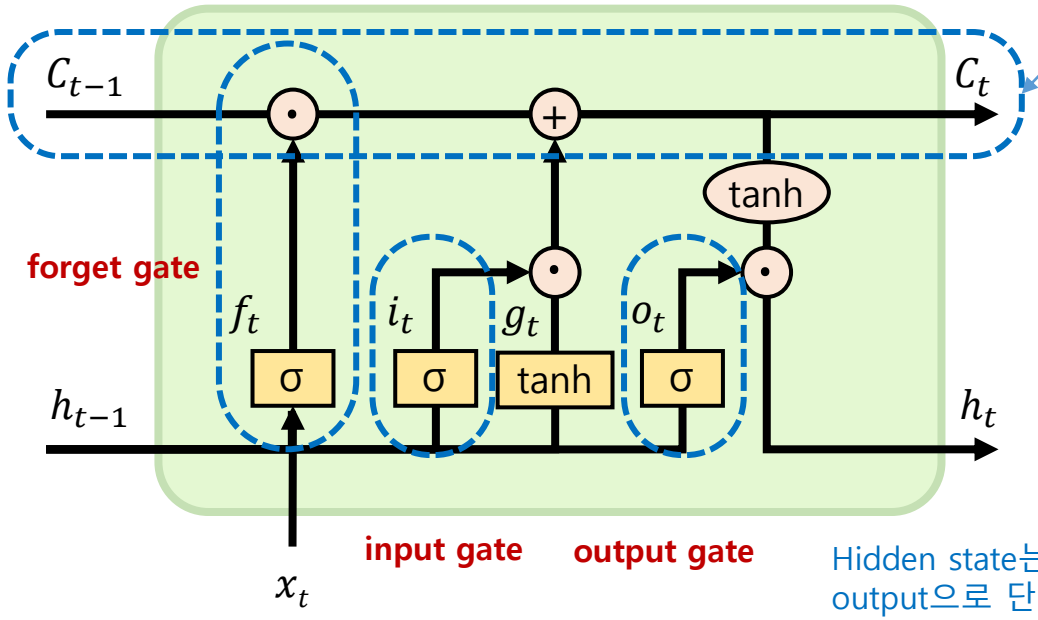


$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}, \qquad W = \begin{bmatrix} W_{hi} & W_{xi} \\ W_{hf} & W_{xf} \\ W_{ho} & W_{xo} \\ W_{hg} & W_{xg} \end{bmatrix}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \tanh(C_t)$$

LSTM : Long Short Term Memory



Cell State 간에 linear interaction 형태로 구성하여 Gradient Flow 지름길을 생성한 것이 핵심 Idea

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}, \quad W = \begin{bmatrix} W_{hi} & W_{xi} \\ W_{hf} & W_{xf} \\ W_{ho} & W_{xo} \\ W_{hg} & W_{xg} \end{bmatrix}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot g_t$$

Cell state가 1씩 증가 or 감소
→ Element 별로 integer counter

$$h_t = o_t \odot \tanh(C_t)$$

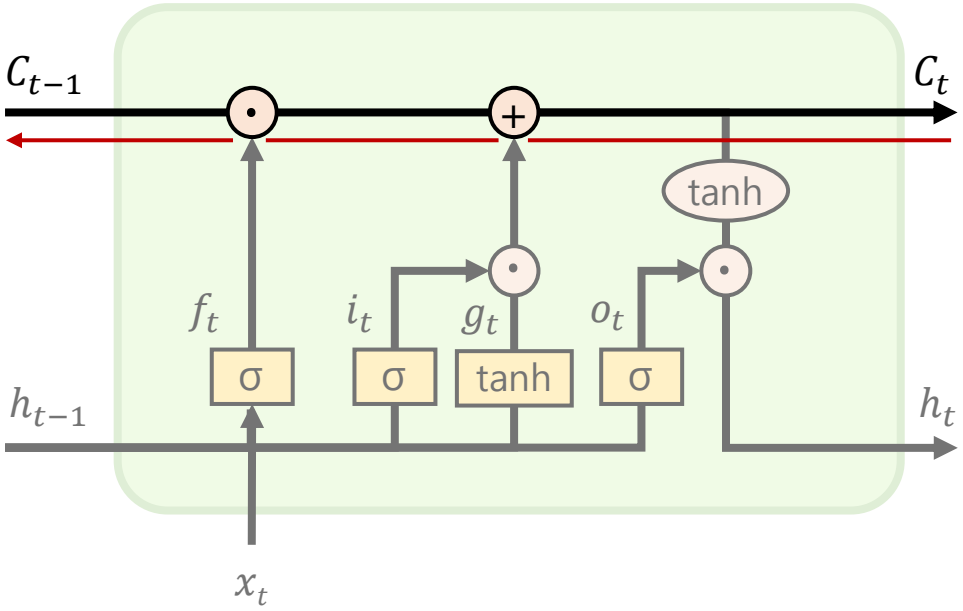
Counter 값을 [-1, 1] 범위로 squashing

Gate 종류

- $g_t \in$ **(-1,1)** : 셀에서 형성된 새로운 기억
- $i_t \in$ **(0,1) :** 입력 게이트 (input gate)
- $f_t \in$**(0,1) :** 망각 게이트 (forget gate)
- $o_t \in$ **(0,1) :** 출력 게이트 (output gate)

LSTM Gradient Flow

Gradient Flow에 W가 곱해지지 않도록 구조 변경!



C_t 에서 C_{t-1} 사이 Gradient 연산에서 W 곱은 완전히 사라짐

f_t와의 요소 곱 ⊙이 있기 때문에 local gradient는 forget gate 값이 됨

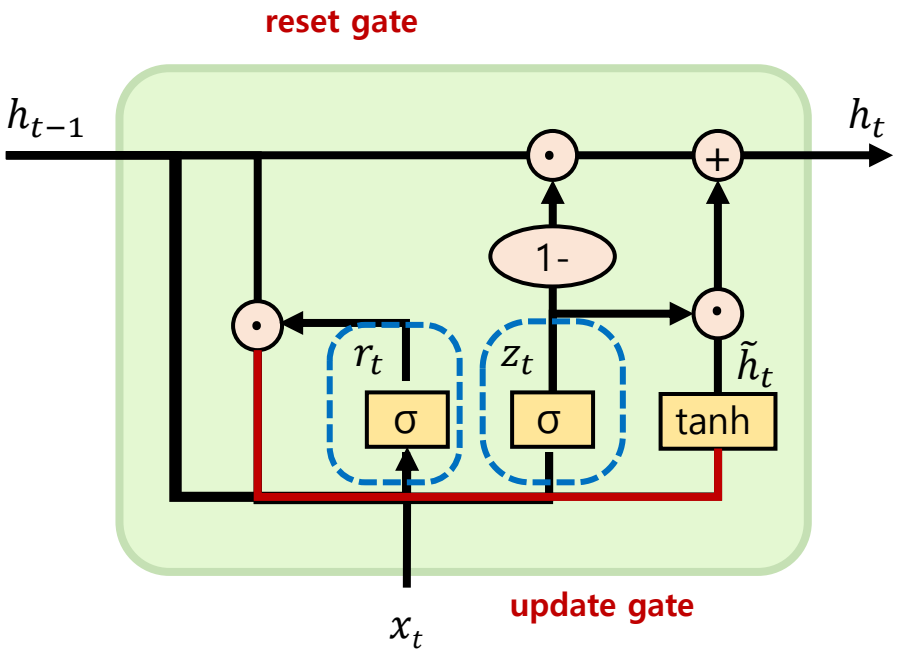
LSTM의 Gradient Flow가 좋은 이유

- f_t 값이 매번 바뀌기 때문에 같은 값이 반복해서 곱해지지 않음
- forget gate 값이 (0, 1) 범위에서 존재하기 때문에 gradient exploding은 일어나지 않음
- Final hidden state에서 first cell state까지 backward path에는 tanh를 1번만 나타남 (즉, 반복적인 tanh의 곱셈이 사라짐)

Tip : forget gate가 1보다 작기 때문에 gradient vanishing이 일어날 수도 있는데 이를 방지하기 위해 forget gate의 bias를 1로 초기화함

GRU : Gated Recurrent Unit

LSTM의 장점은 유지하되 게이트 구조를 단순하게!



$$\begin{pmatrix} r_t \\ z_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \quad , \quad W = \begin{bmatrix} W_{hr} & W_{xr} \\ W_{hz} & W_{xz} \end{bmatrix}$$

$$\tilde{h}_t = \tanh\left((W_{hh} \quad W_{xh}) \begin{pmatrix} r_t \odot h_{t-1} \\ x_t \end{pmatrix} \right)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

- Cell State를 없애고 h가 그 역할을 겸하도록 수정
- $\boldsymbol{r_t}$ **reset gates :** 다음 상태 $\tilde{h}_t$를 계산하기 위해 이전 상태 h_{t-1} 값에서 사용할 부분을 제어
- $\boldsymbol{z_t}$ **update gates :** 이전 상태 h_{t-1}를 유지할 지 새로운 상태 $\tilde{h}_t$로 대체할지를 결정

LSTM의 forget gate와 output gate역할을 update gate $\boldsymbol{z}$로 합침

Thank you!