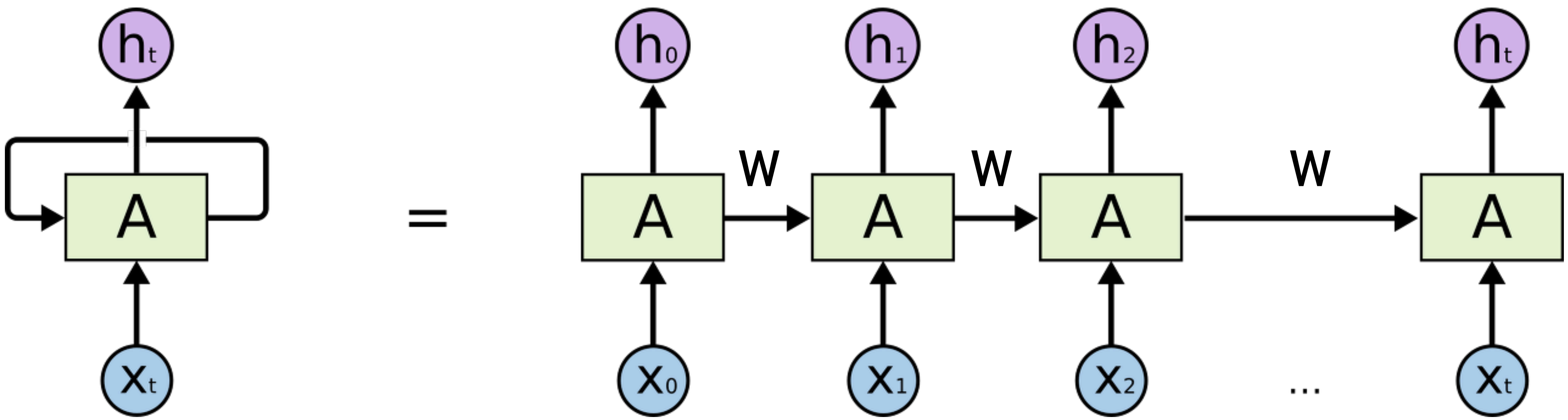
AI ROBOTICS KR : NLP STUDY

INHWAN LEE

LSTM

목차

RNN STRUCTURE

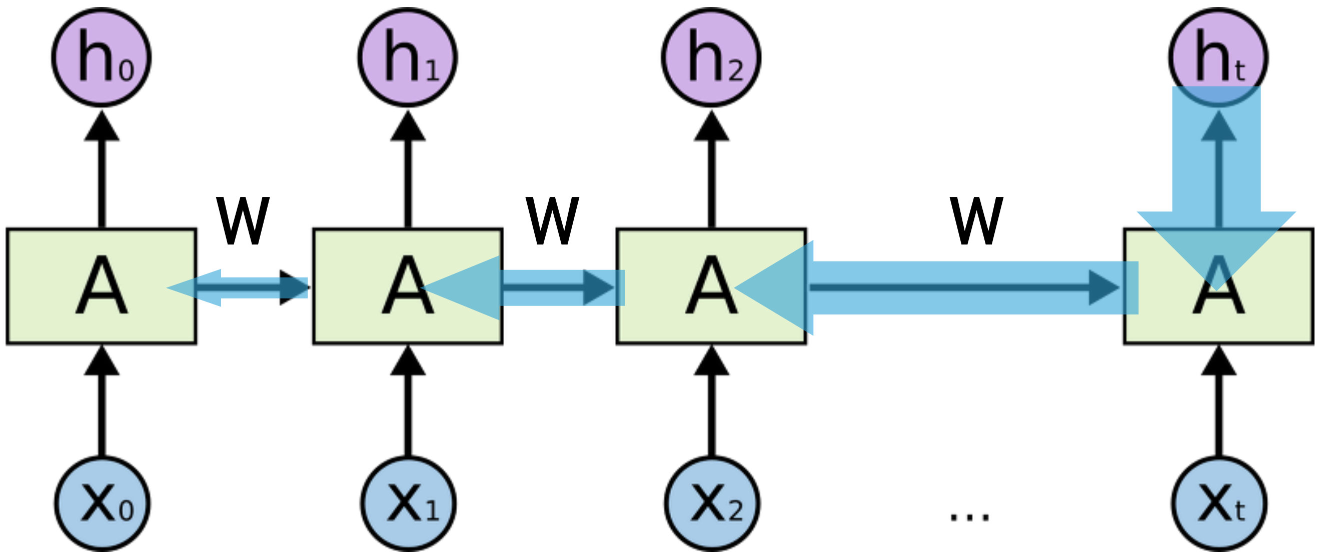


$$h_t = \tanh(W_h \cdot [h_{t-1}, x_t] + b_h)$$

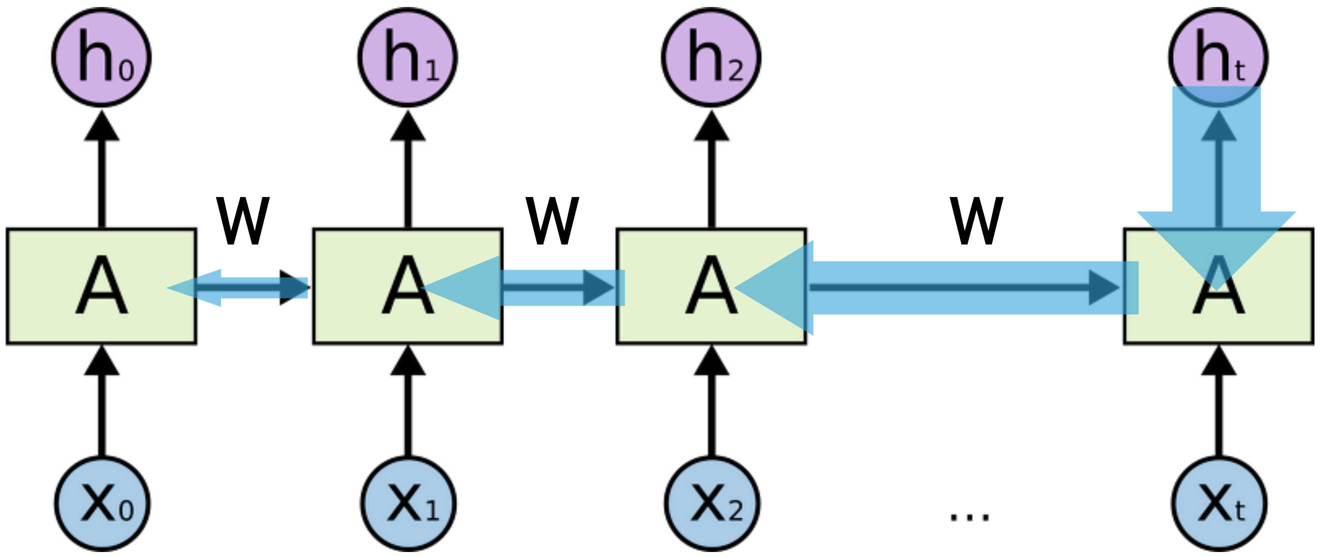
$$y_t = W_y \cdot h_t + b_y$$

(image source : https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

VANISHING GRADIENT 문제



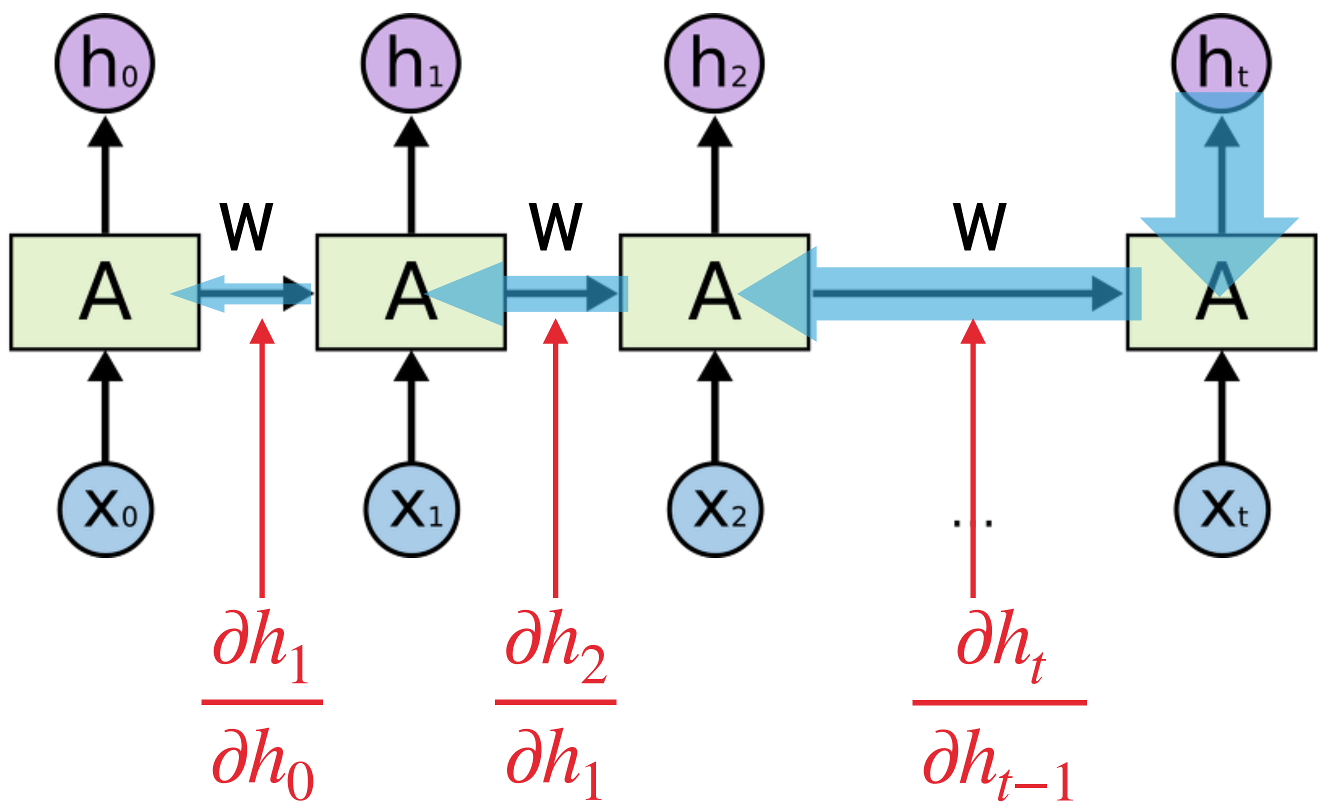
VANISHING GRADIENT 문제



역전파를 수행할 때 각 단계마다 gradient가 감소할 가능성이 크다!

따라서 거리가 벌어지면 정보(gradient)가 전달되지 않는다.

VANISHING GRADIENT 문제



$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial h_t} \cdot \boxed{\prod_{i=1}^{t} \frac{\partial h_i}{\partial h_{i-1}}} \cdot \frac{\partial h_t}{\partial W} \cdot \star$$

: 0으로 가까워지며
업데이트가 일어나지 않음

$$\left(W \Leftarrow W - \alpha \cdot \sum_{t} \frac{\partial L_t}{\partial W} \right)$$

RNN 언어 모델 예시

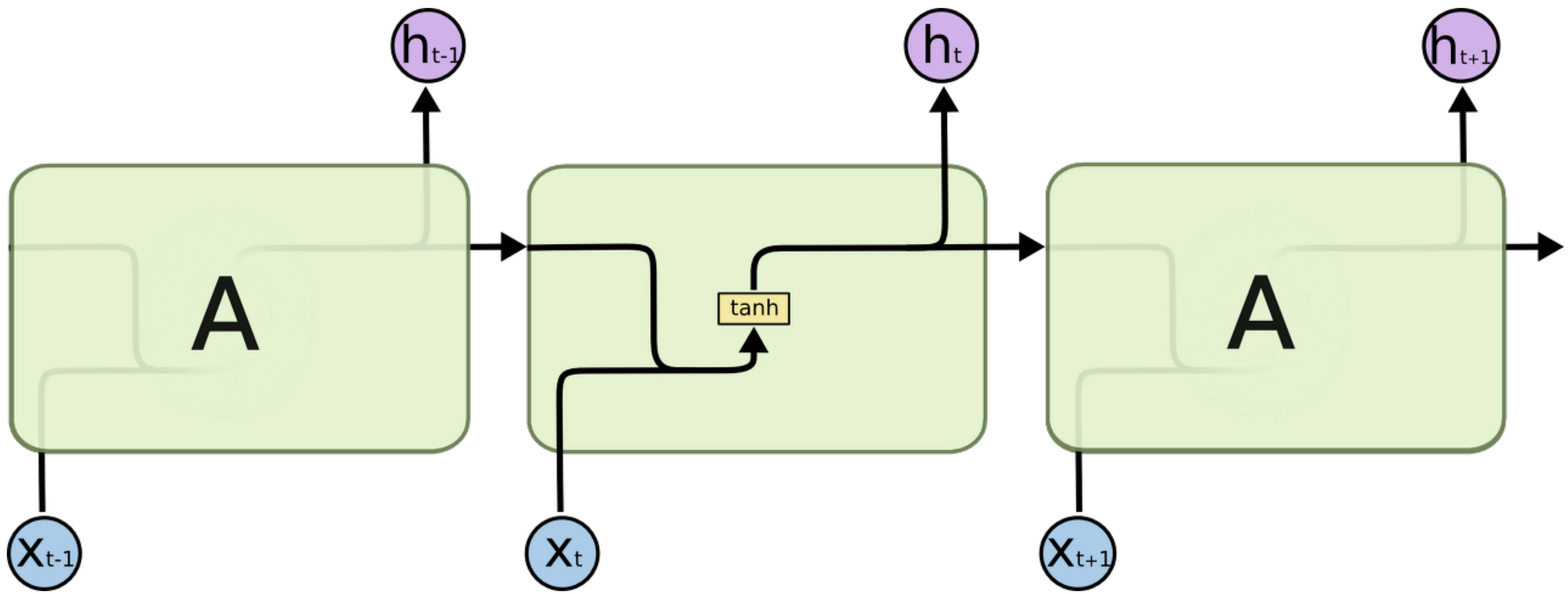
그녀가 티켓을 출력하려고 했을 때, 막 프린터의 토너가 떨어졌다.
어쩔 수 없이 비싼 돈을 주고 토너를 사와서 설치한 이후에야
그녀는 ___을 인쇄할 수 있었다.

RNN 언어 모델 예시

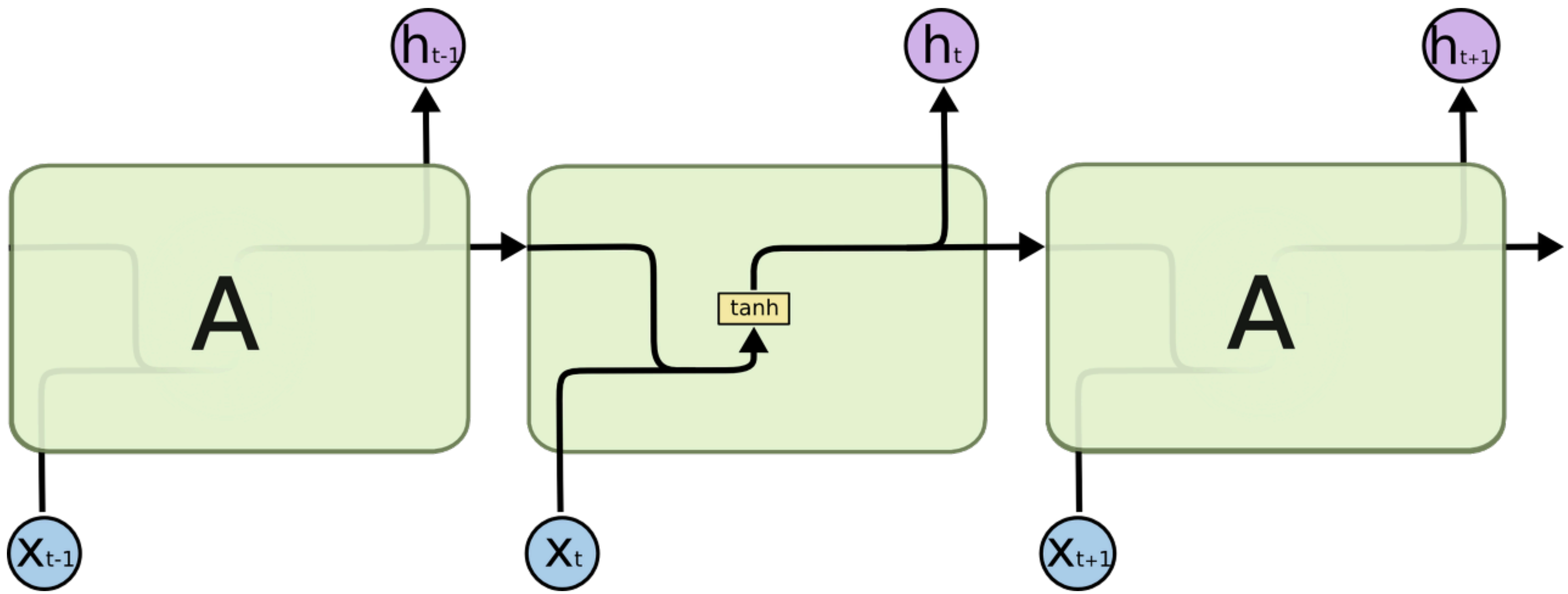
그녀가 티켓을 출력하려고 했을 때, 막 프린터의 토너가 떨어졌다.
어쩔 수 없이 비싼 돈을 주고 토너를 사와서 설치한 이후에야
그녀는 **티켓**을 인쇄할 수 있었다.

- 이 예시를 학습하기 위해서는 처음의 '티켓'과 마지막 '티켓' 사이의 관계를 모델링 해야하지만...
- gradient가 작은 경우 거리가 먼 관계성은 학습할 수 없다.

LSTM : MOTIVATION

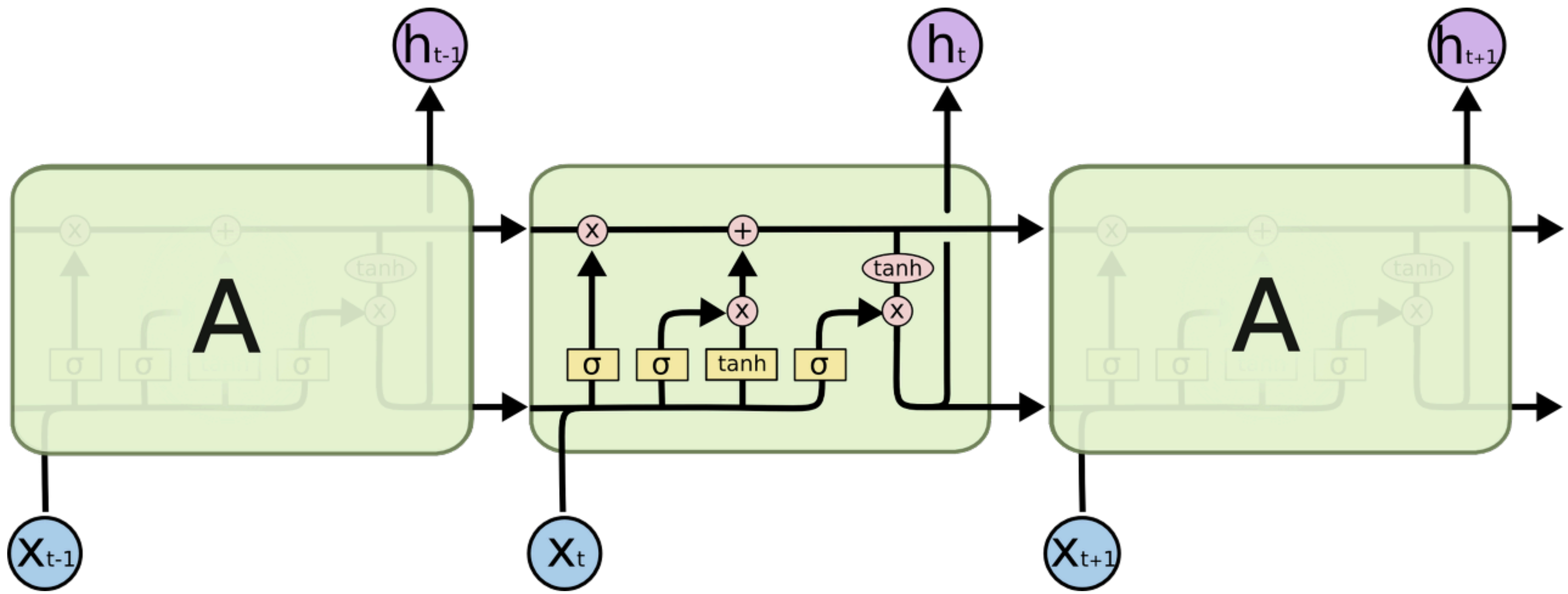


LSTM : MOTIVATION

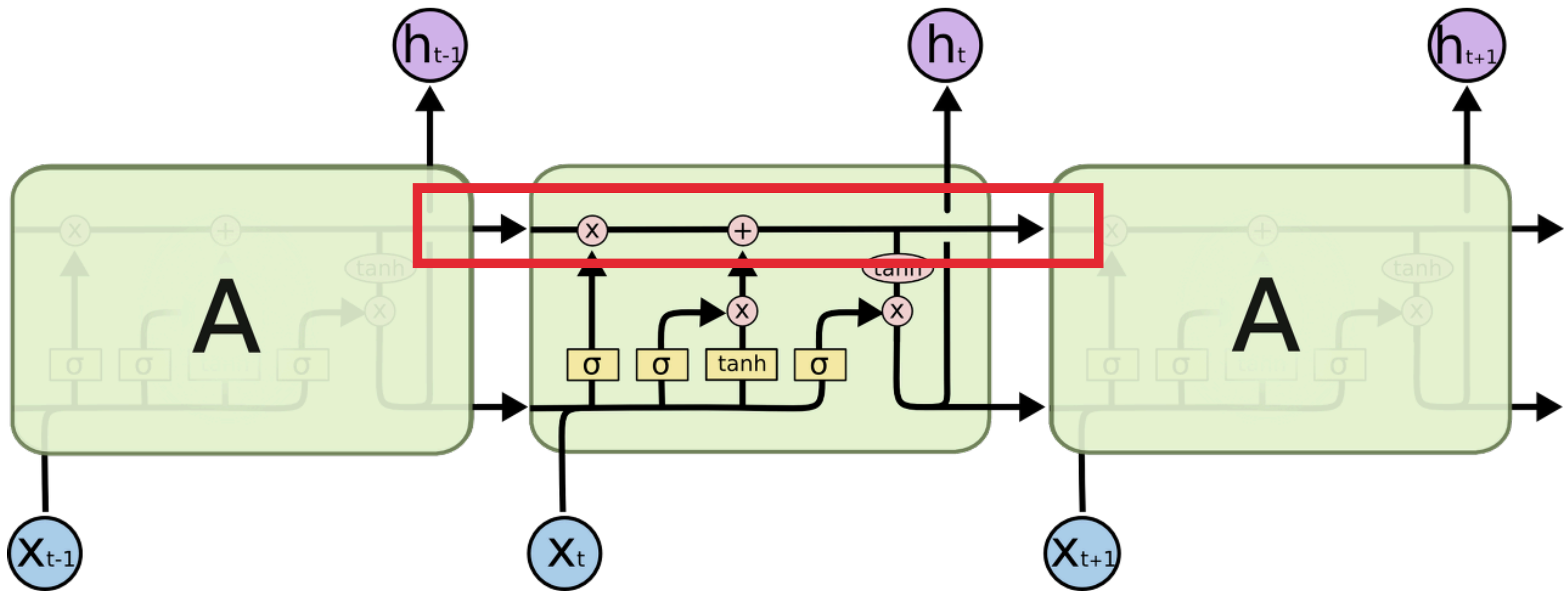


Q: RNN의 문제점은 각 층별로 전달되는 정보가 너무 적은 것이 아닐까?

LSTM : MOTIVATION

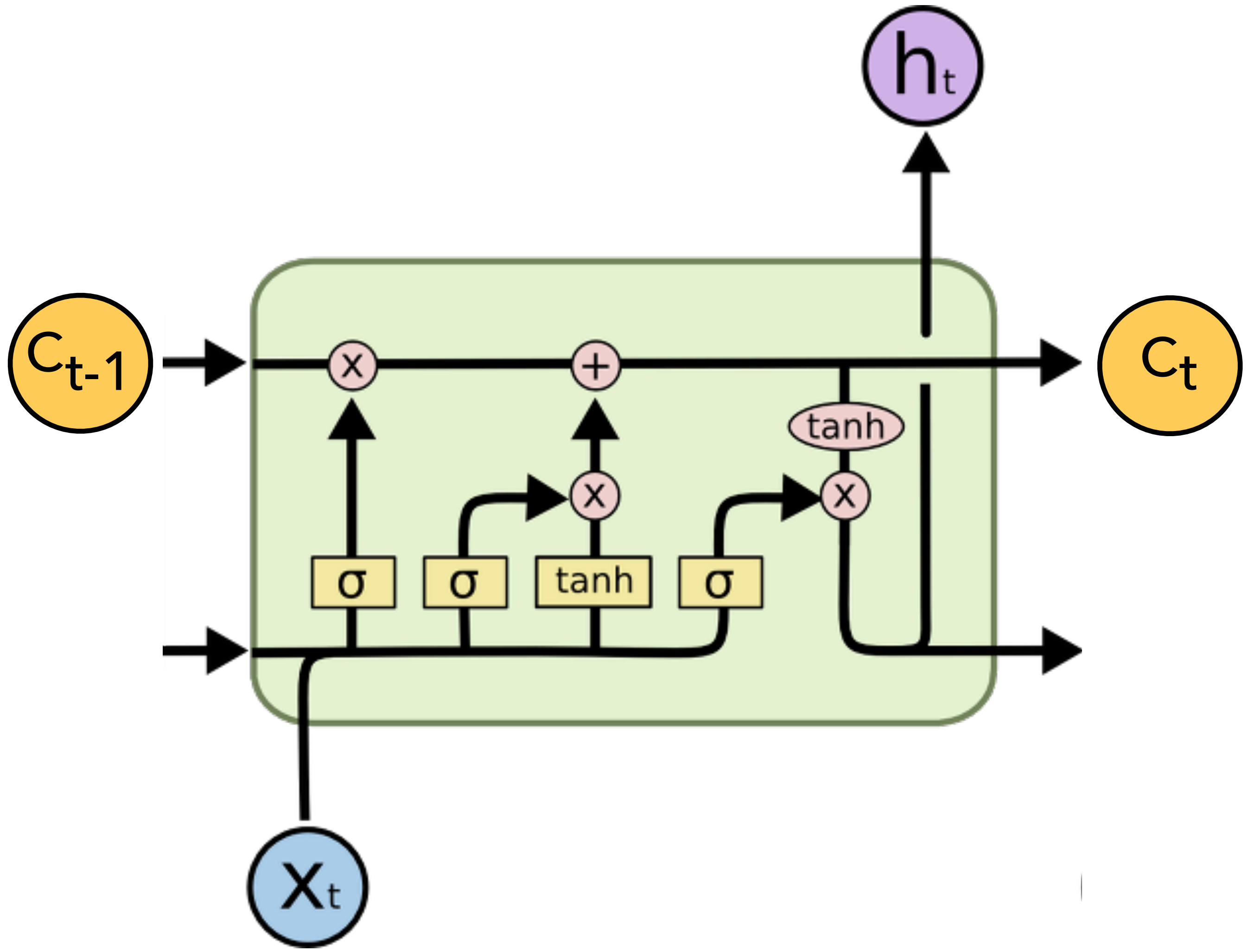


LSTM : MOTIVATION



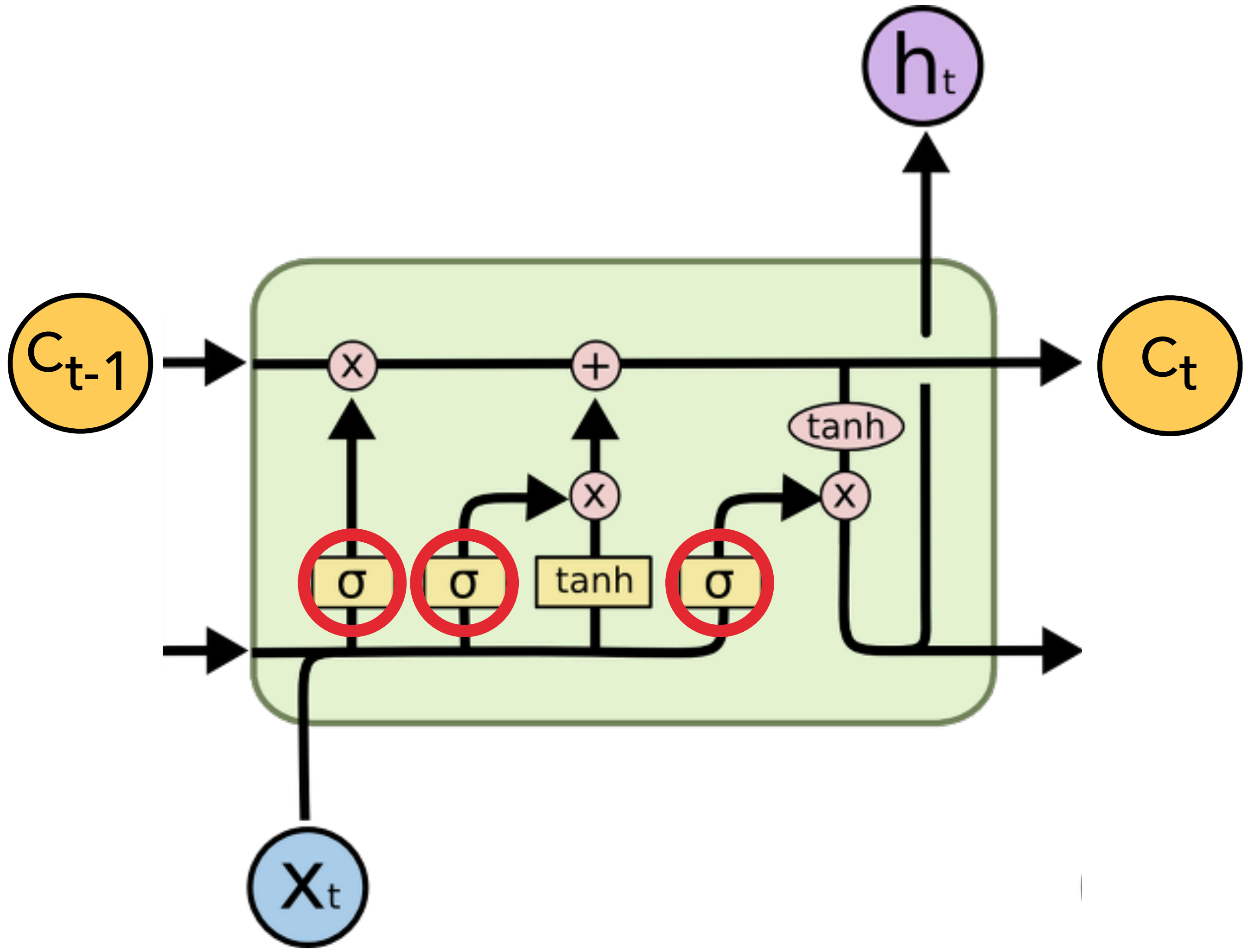
A : 추가적인 정보를 전달하는 라인을 만들자!
(Hochreiter and Schmidhuber, 1997)

LSTM : STRUCTURE



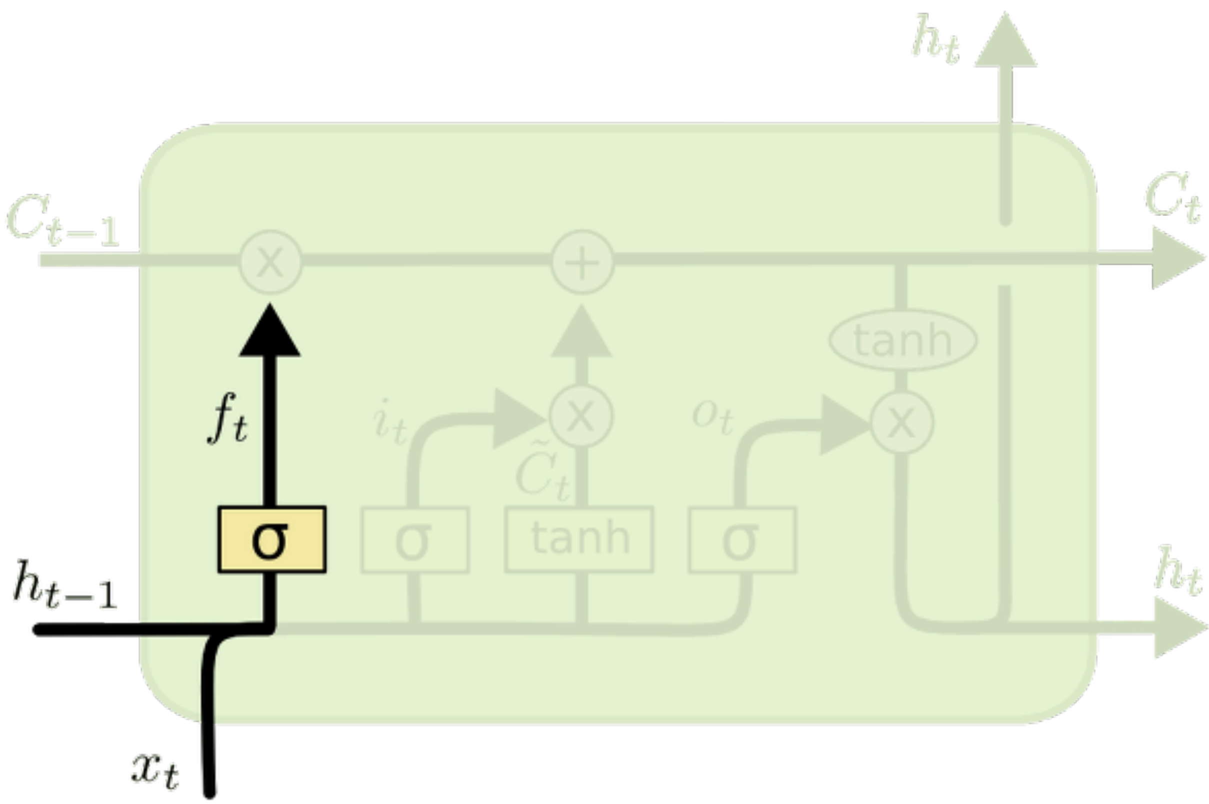
- 매 층마다 hidden state h_t와 cell state C_t를 계산
- cell state는 장기간 보존되는 정보를 저장
  (현재 문장의 주어, 화자의 성별 등)

LSTM : STRUCTURE



- LSTM의 흐름은 3개의 gate에 의해 제어된다.
- 각 gate는 sigmoid 함수에 의해 0~1 사이의 숫자를 내보내며, 얼마나 정보를 전달해야 하는지를 결정한다.

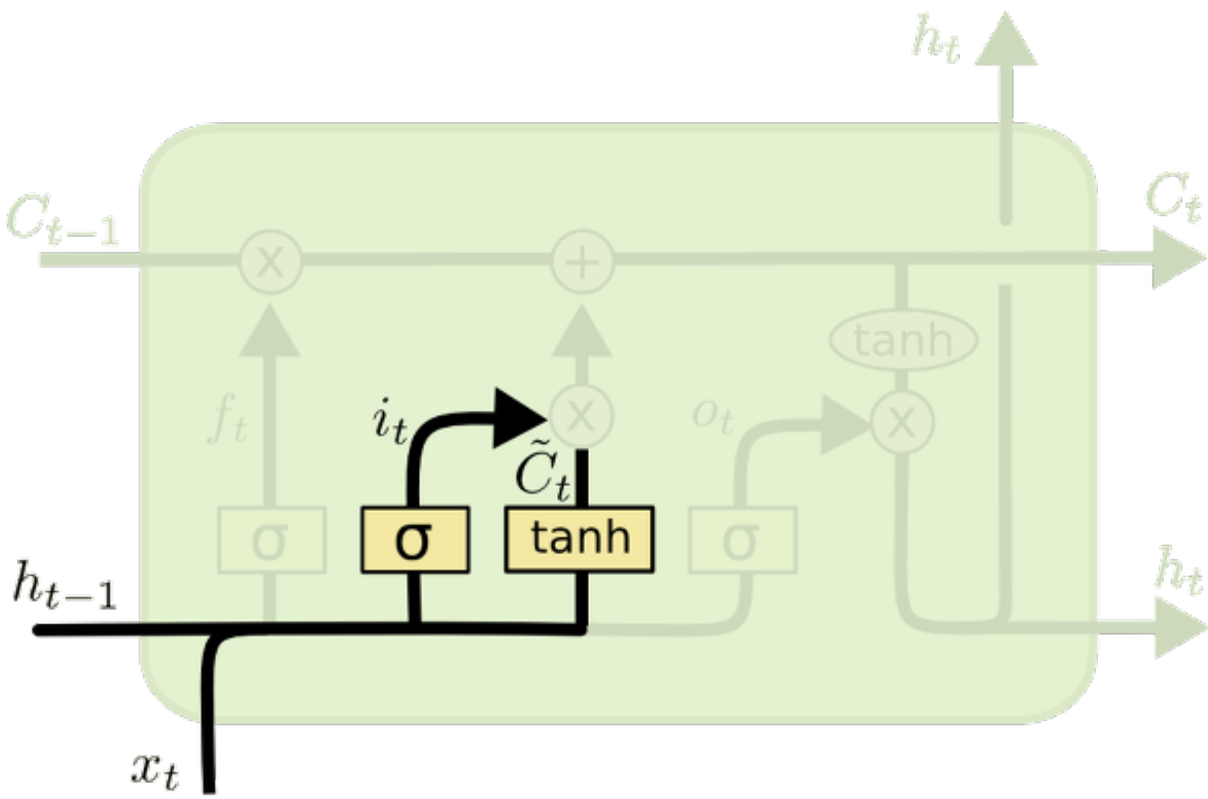
LSTM : FORGET GATE



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

- forget gate f_t는 이전 cell state C_{t-1} 의 어떤 부분을 보존할지 결정한다.

LSTM : INPUT GATE

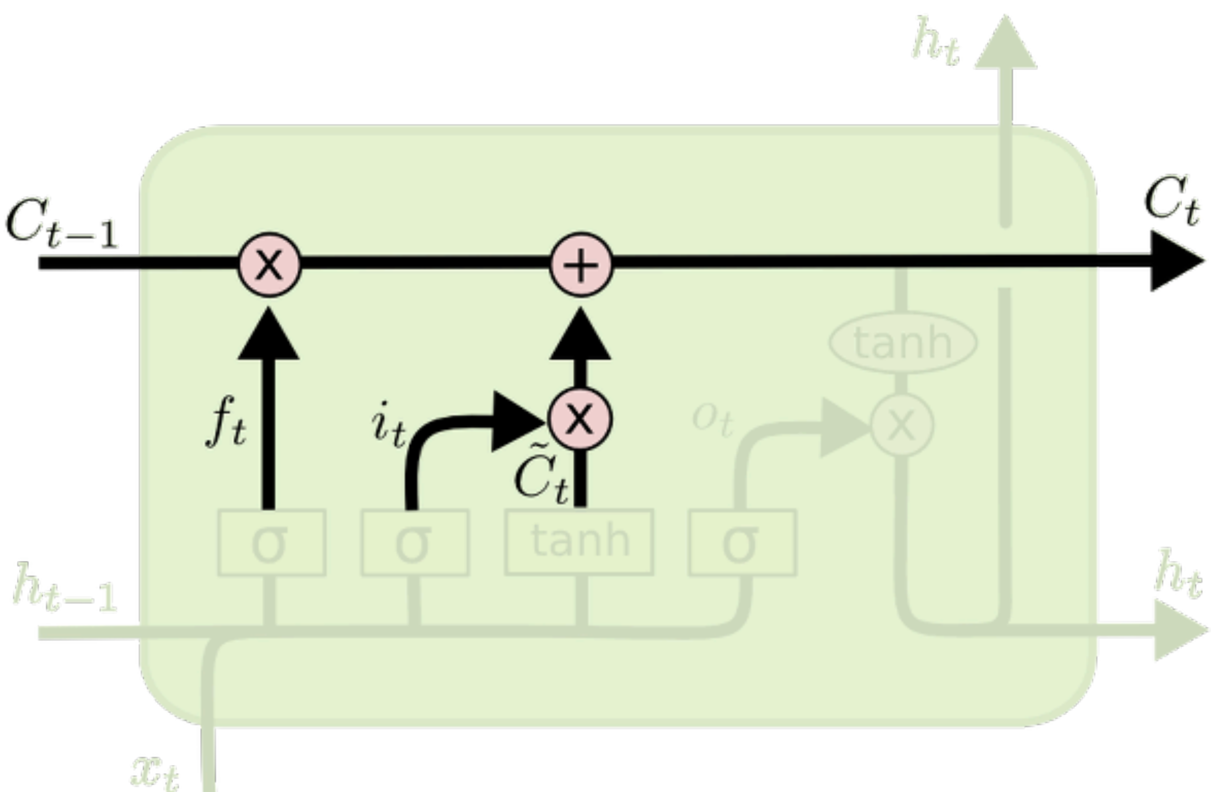


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

- input gate i_t는 입력 x_t를 토대로 cell state를 얼마나 갱신할지 결정한다.
- 직전 hidden state h_{t-1} 과 입력 x_t를 이용하여 이번 층의 새로운 갱신값인 $\tilde{C}_t$를 생성한다.

LSTM : UPDATE CELL STATE

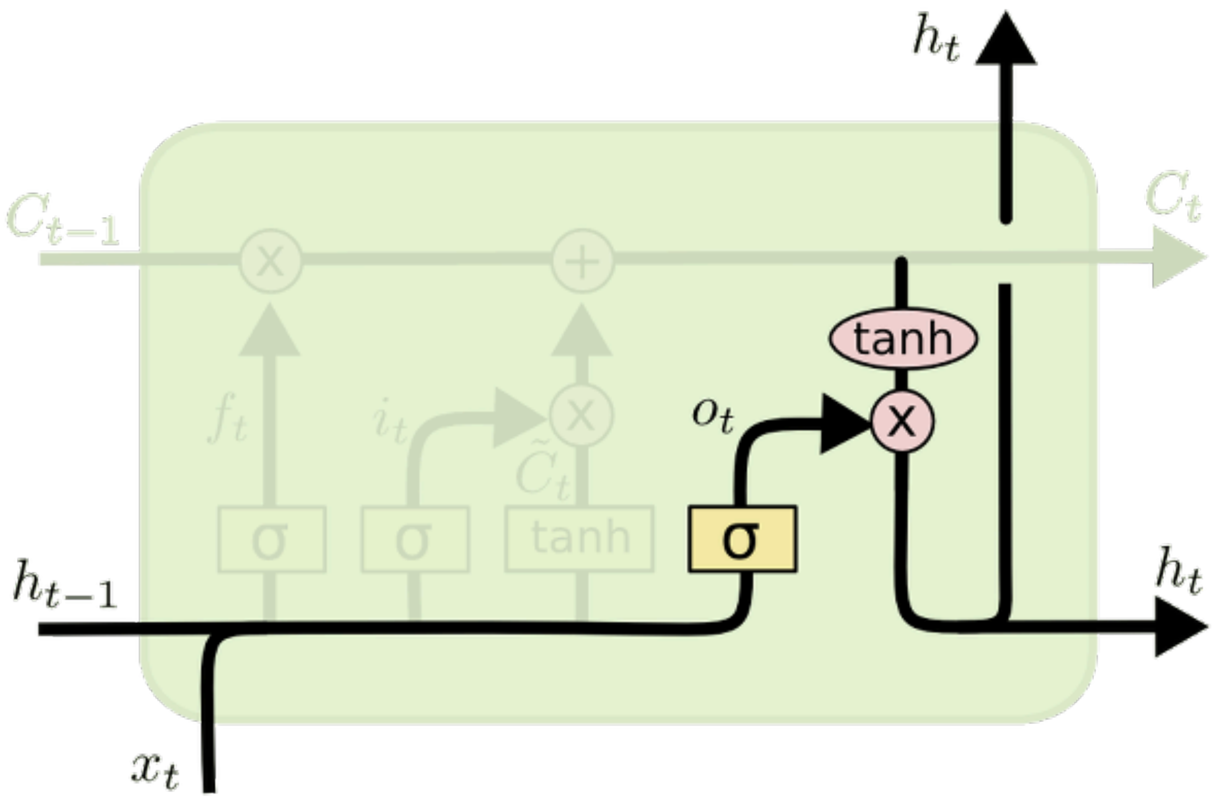


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

(*는 원소별 곱셈)

- forget gate f_t를 이용하여 지난 cell state의 정보를 지우고 input gate i_t를 이용하여 cell state를 갱신한다.

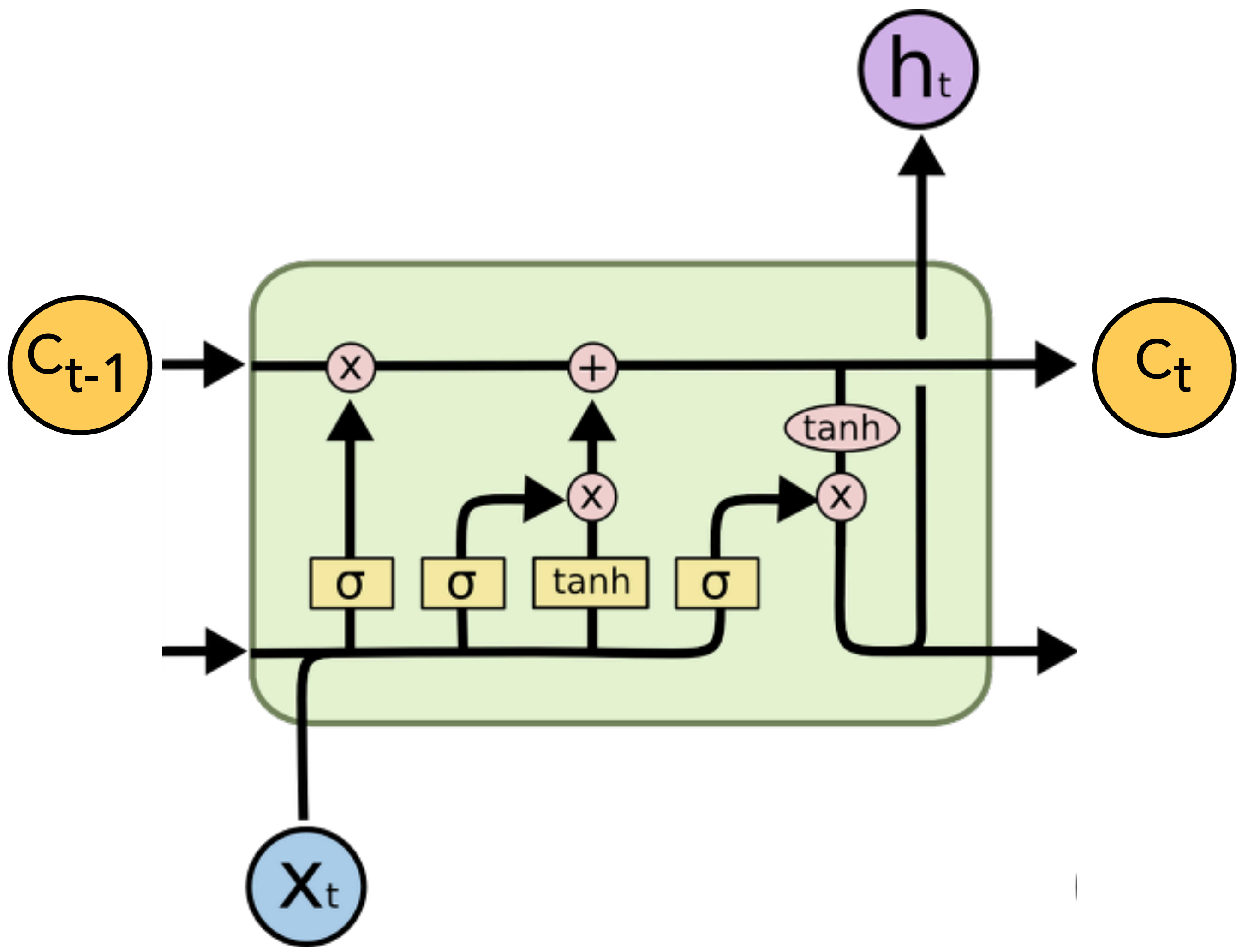
LSTM : OUTPUT GATE



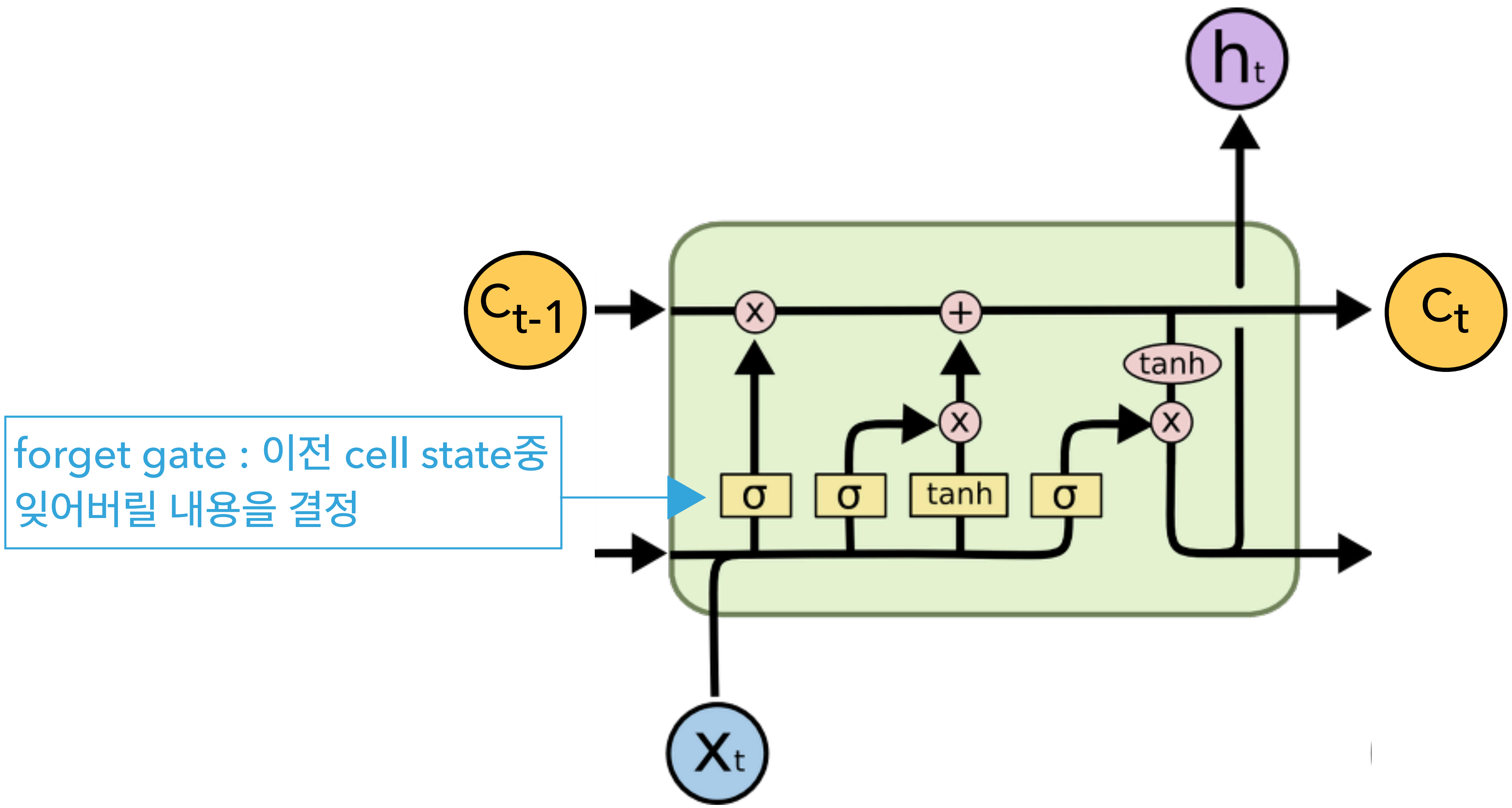
$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

- cell state에 tanh 함수를 적용한 후 output gate를 이용하여 어떤 부분을 hidden state에 반영할 지 결정한다.

LSTM : SUMMARY



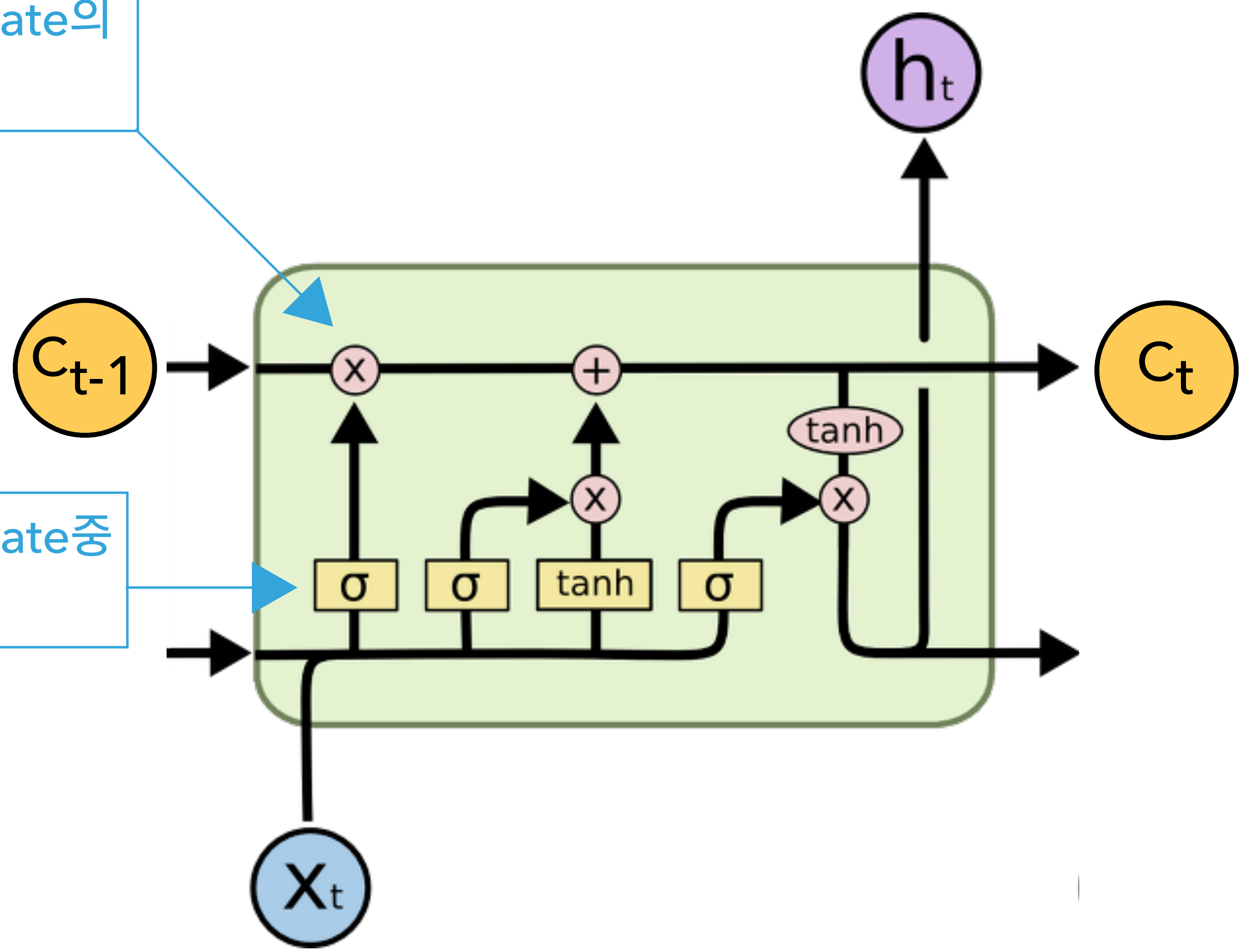
LSTM : SUMMARY



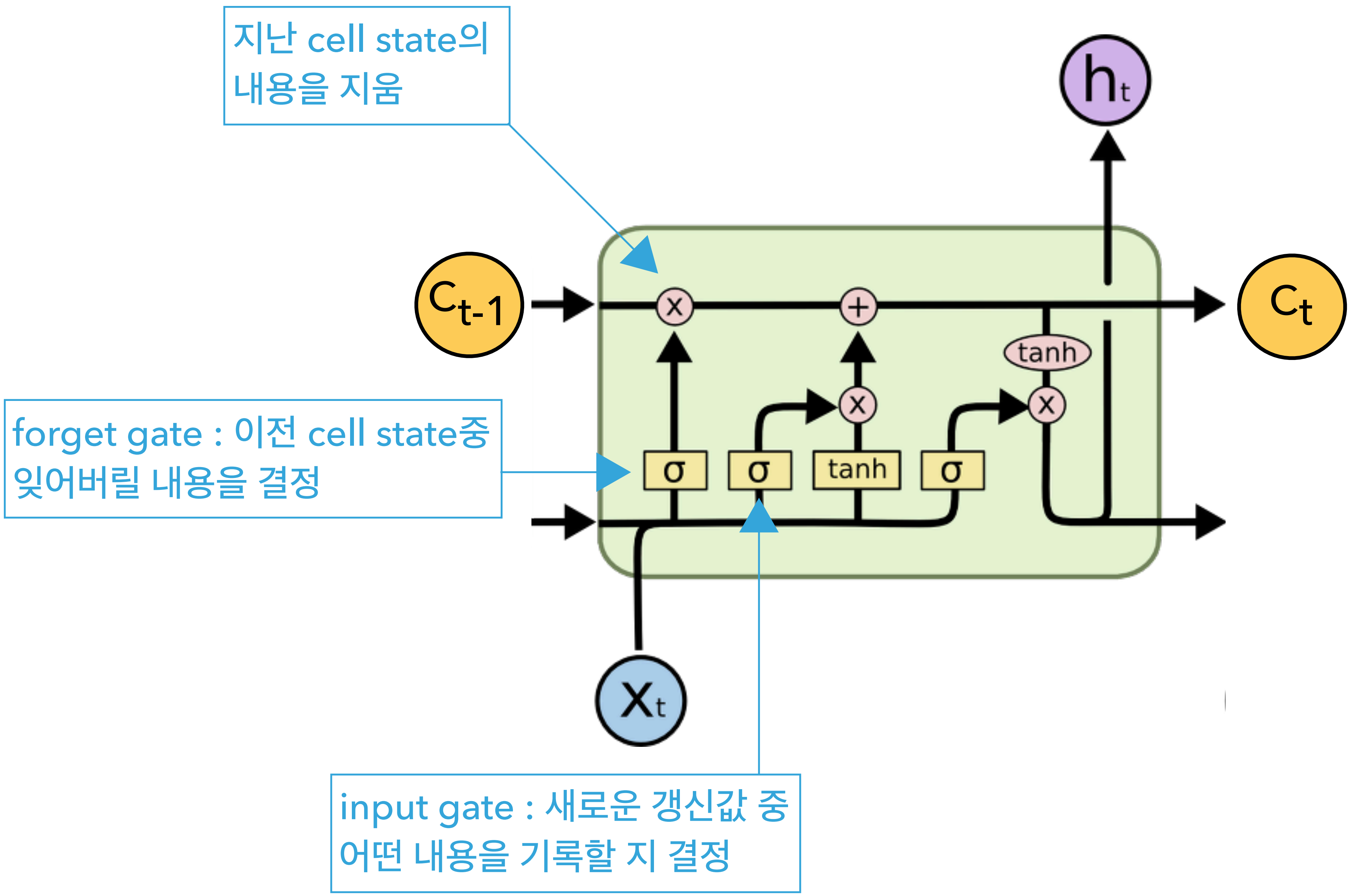
LSTM : SUMMARY

지난 cell state의 내용을 지움

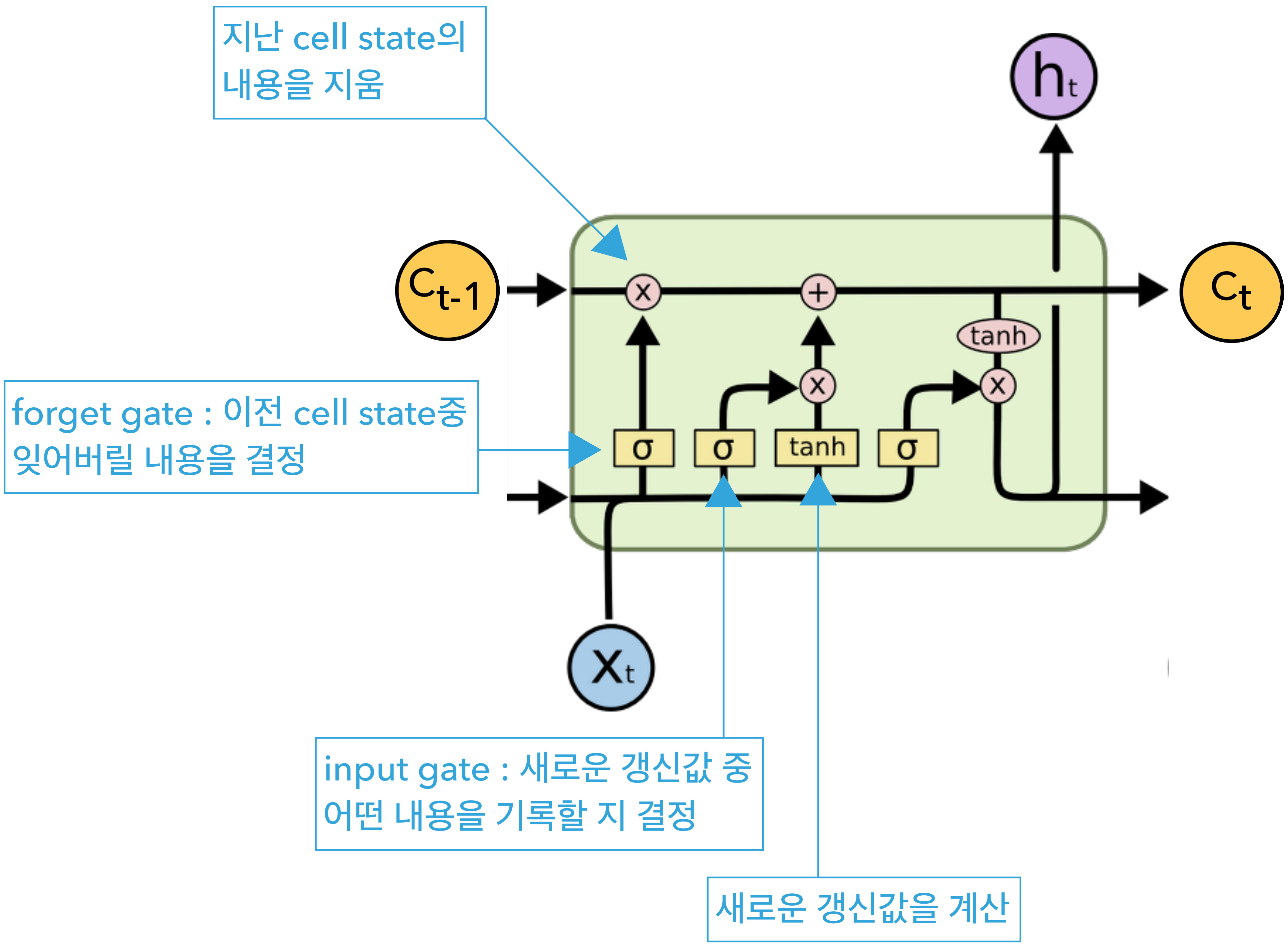
forget gate : 이전 cell state중 잊어버릴 내용을 결정



LSTM : SUMMARY



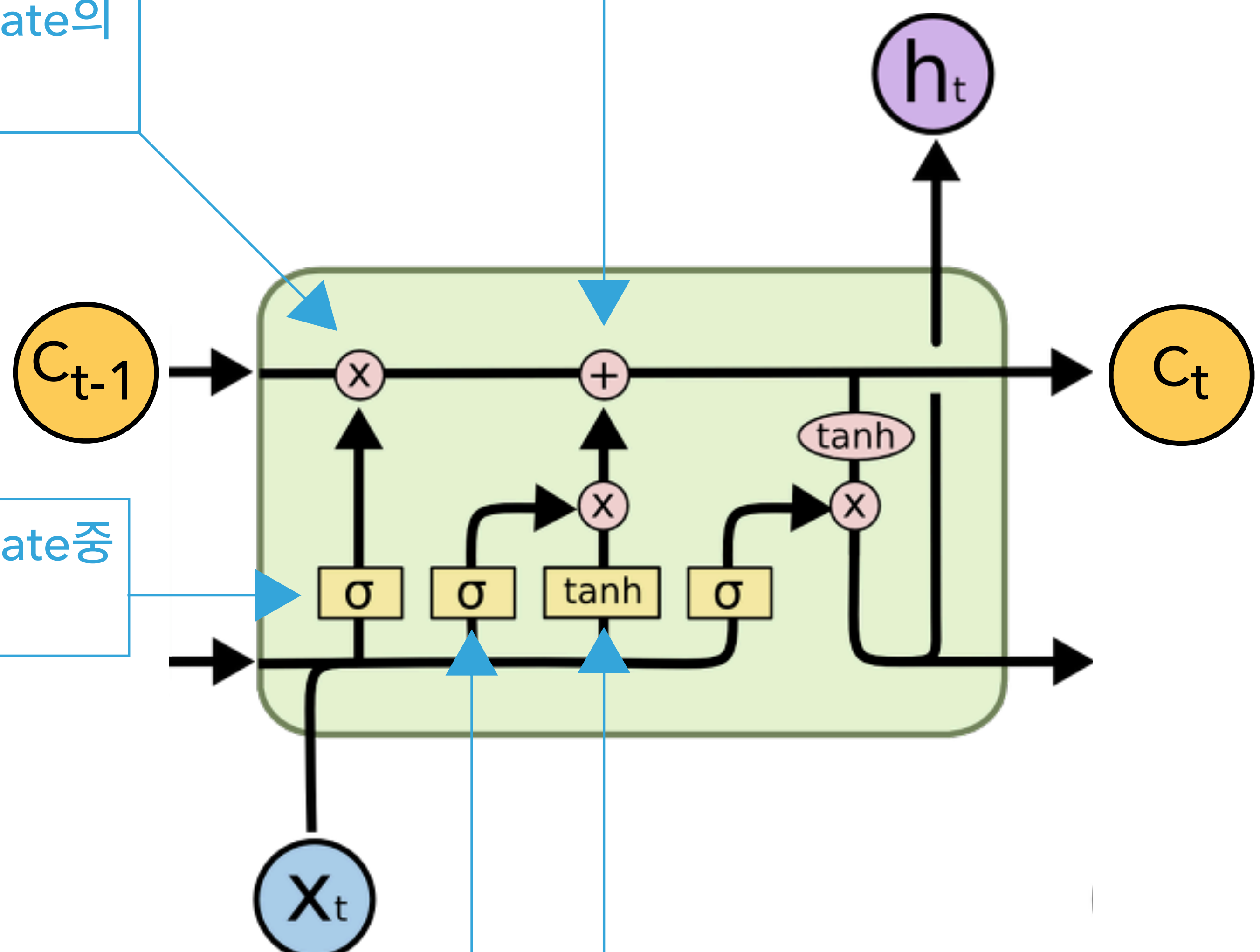
LSTM : SUMMARY



LSTM : SUMMARY

forget gate와 input gate를
이용하여 새로운 cell state 산출

지난 cell state의
내용을 지움



forget gate : 이전 cell state중
잊어버릴 내용을 결정

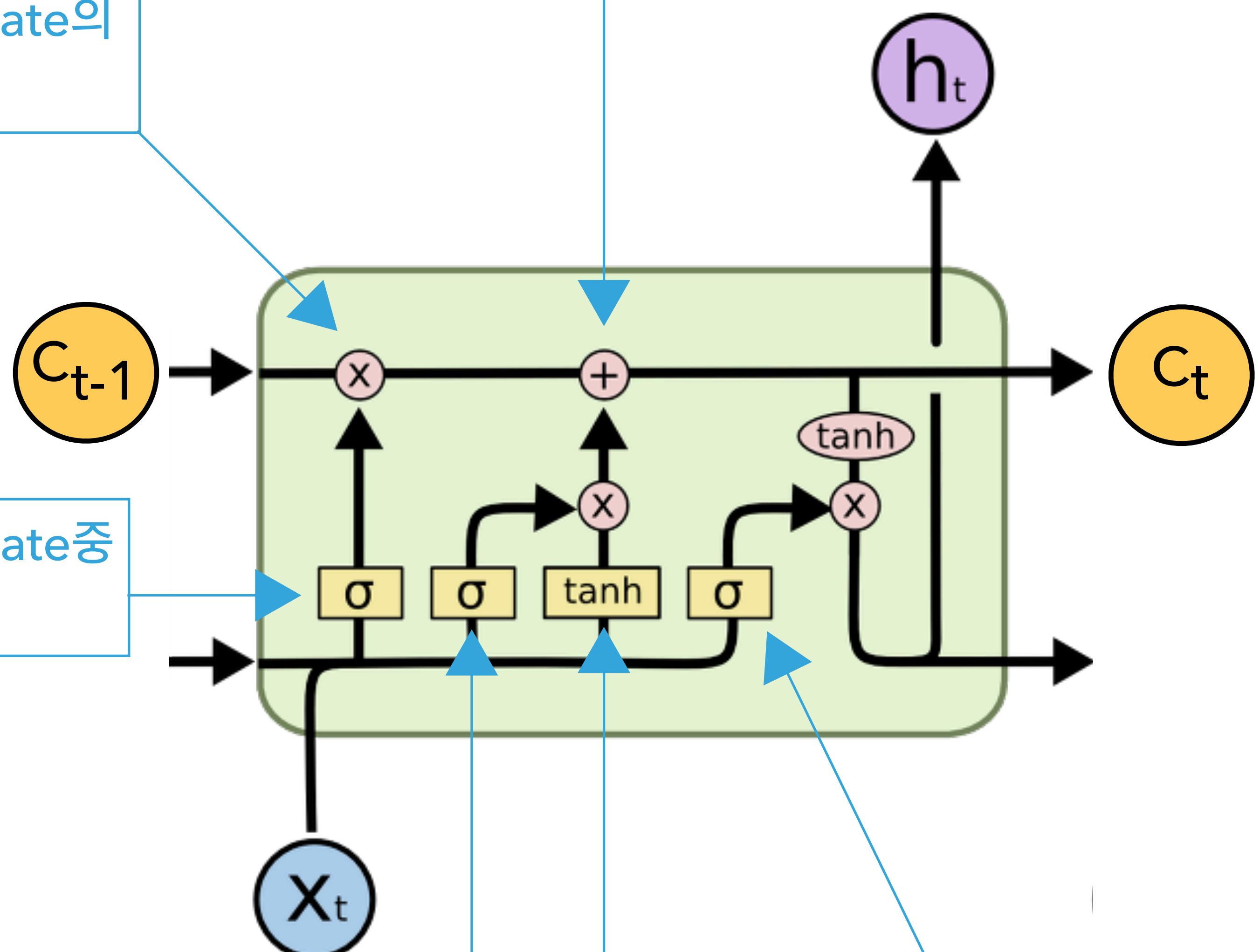
input gate : 새로운 갱신값 중
어떤 내용을 기록할 지 결정

새로운 갱신값을 계산

LSTM : SUMMARY

forget gate와 input gate를
이용하여 새로운 cell state 산출

지난 cell state의
내용을 지움



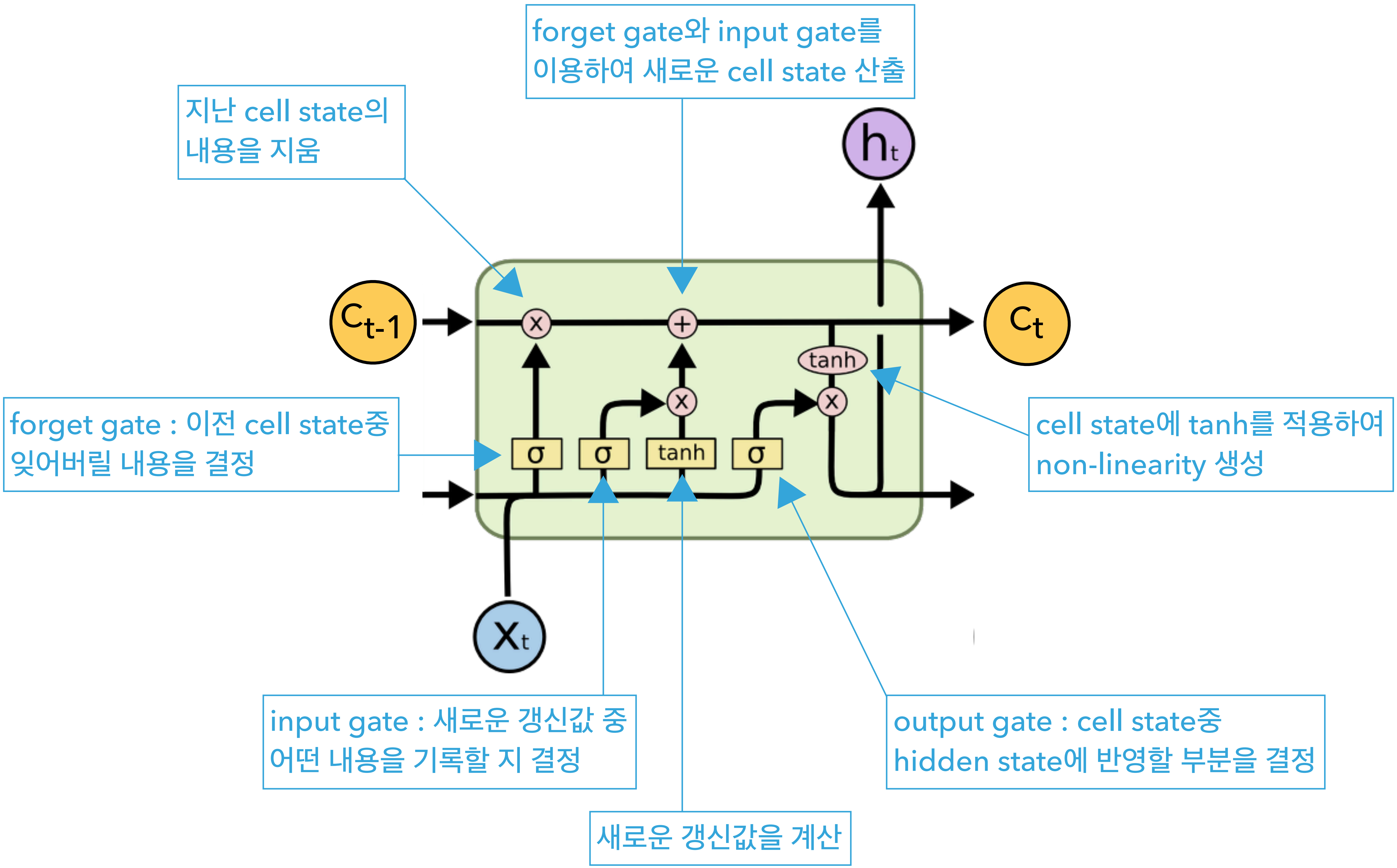
forget gate : 이전 cell state중
잊어버릴 내용을 결정

input gate : 새로운 갱신값 중
어떤 내용을 기록할 지 결정

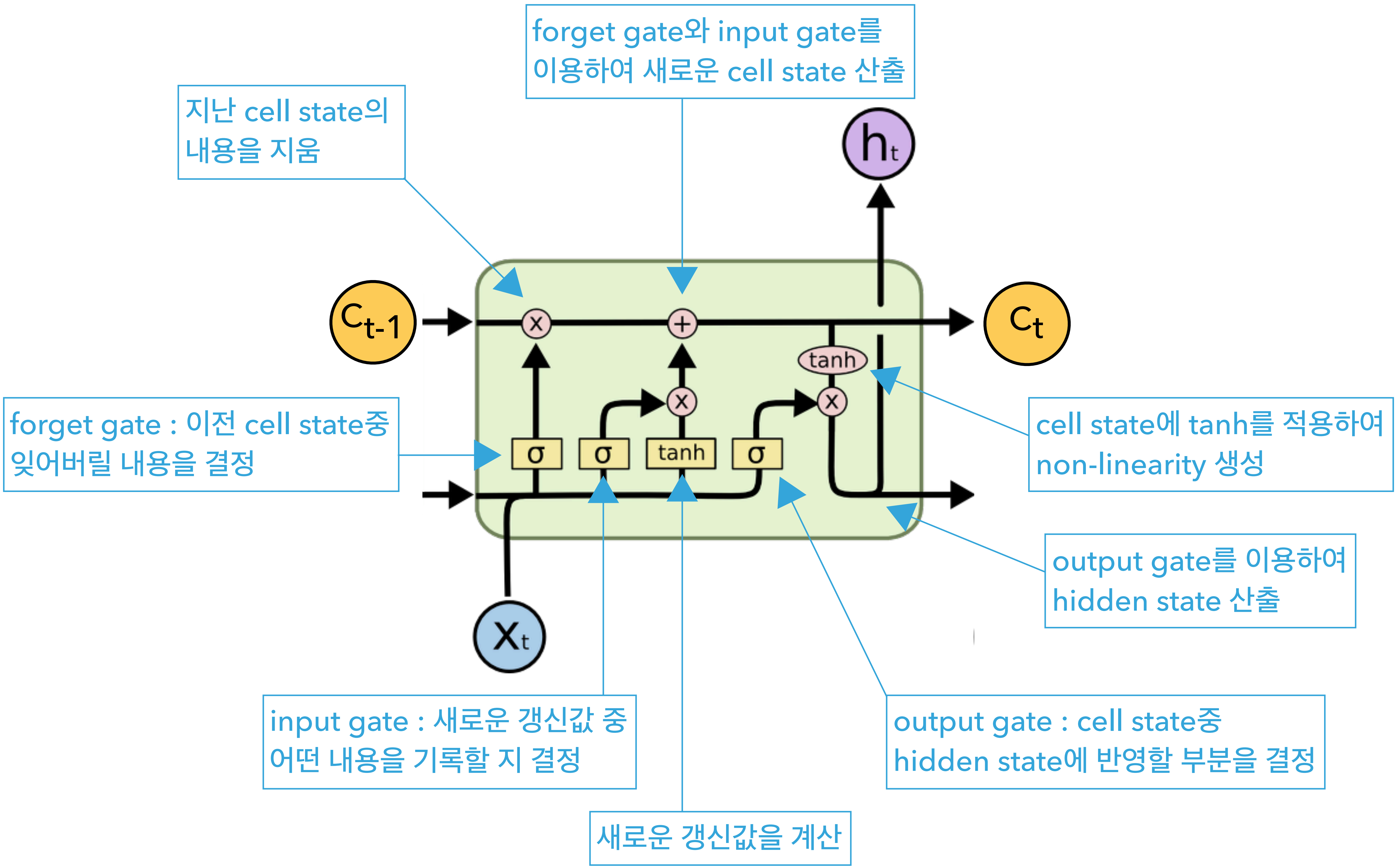
output gate : cell state중
hidden state에 반영할 부분을 결정

새로운 갱신값을 계산

LSTM : SUMMARY



LSTM : SUMMARY



LSTM : LOSS FUCTION

$$\hat{y}_t = softmax(h_t)$$

$$L_t = L(\hat{y}_t, y_t) = -\,y_t \log \hat{y}_t$$

$$L = \sum_t L_t = -\sum_t y_t \log \hat{y}_t$$

- 모든 timestep t에 대하여 cross-entropy loss를 더하여 산출
- 이후 Backpropagation을 통해 parameter update
  (Backpropagation through time : https://aikorea.org/blog/rnn-tutorial-3/)

LSTM : VANISHING GRADIENT PROBLEM?

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

$$\frac{\partial C_T}{\partial C_t} = \frac{\partial C_T}{\partial C_{T-1}} \cdot \frac{\partial C_{T-1}}{\partial C_{T-2}} \cdot \dots \cdot \frac{\partial C_{t+1}}{\partial C_t} = \boxed{\prod_{i=t+1}^{T} f_i}$$

$$\left(\frac{\partial C_t}{\partial C_{t-1}} = f_t \right)$$

- forget gate f_t 가 0에 가깝지 않는 한 기억을 보존할 수 있다!

(reference : https://curt-park.github.io/2017-04-03/why-is-lstm-strong-on-gradient-vanishing/)

LSTM : REAL-WORLD SUCCESS

- 2013-2015년 사이에 LSTM은 전성기를 누림
  - 손글씨 인식, 음성 인식, 기계 번역, 이미지 자막 처리 등에서 큰 성과를 냄
- 현재는 Transformer 등 다른 방법이 대세
  - WMT16의 결과 보고서에는 'RNN'이 44번 등장
  - 그러나 WMT18의 보고서에는 'RNN'이 9번, 'Transformer'가 63번 등장

SHAKESPEARE TEXT GENERATING

```
HAMLET :
  To be, or not to be, that is the question:
  Whether 'tis nobler in the mind to suffer
  The slings and arrows of outrageous fortune,
  Or to take arms against a sea of troubles
  And by opposing end them. To die—to sleep,
  No more; and by a sleep to say we end
  The heart-ache and the thousand natural shocks
  That flesh is heir to: 'tis a consummation
  Devoutly to be wish'd. To die, to sleep;
```

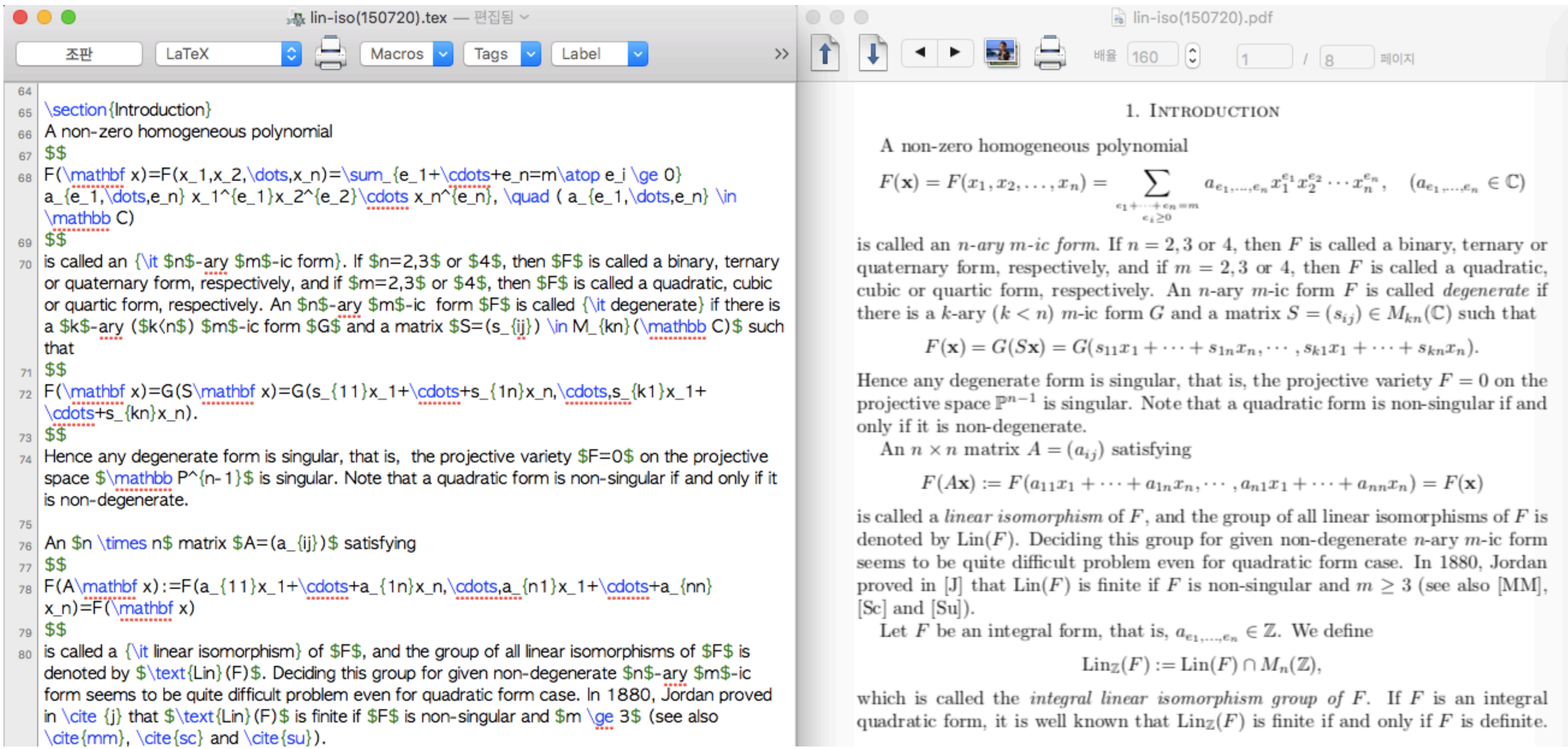
SHAKESPEARE TEXT GENERATING

```
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.

VIOLA:
I'll drink it.
```

SHAKESPEARE TEXT GENERATING

```
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.
```

LATEX : SOURCE CODE AND RESULT



LATEX GENERATING BASED ON ALG.GEOMETRY

For $\bigoplus_{n=1,\dots,m}$ 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

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

and the comparicoly 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 $\operatorname{Spec}(R') \to S$ 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

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

is a unique morphism of algebraic stacks. Note that

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

and

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

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 $\underline{Proj}_X(\mathcal{A}) = \operatorname{Spec}(B)$ 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 $S = \operatorname{Spec}(R)$ and $Y = \operatorname{Spec}(R)$.*

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\dots,n} U_i$ 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 $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\dots,0}$.

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. □

(image source : https://karpathy.github.io/2015/05/21/rnn-effectiveness/)

LATEX GENERATING BASED ON ALG.GEOMETRY

Proof. Omitted. $\square$

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 $X_{étale}$ 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. $\square$

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

Proof. See Spaces, Lemma ??. $\square$

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

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

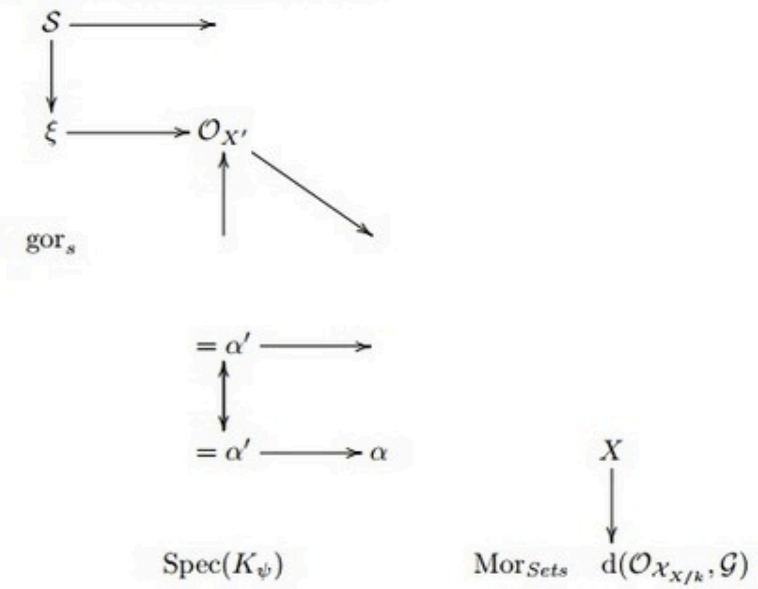
be a morphism of algebraic spaces over S and Y.

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. $\square$

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



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.

$\square$

Proof. We have see that $X = \mathrm{Spec}(R)$ 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. $\square$

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

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \text{ -1}(\mathcal{O}_{X_{étale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

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. $\square$

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.

LATEX : LONG-TERM DEPENDENCY PROBLEM

```
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
is an injective and let $\mathfrak q$ be an abelian sheaf on $X$.
Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\begin{enumerate}
\item \hyperref[setain-construction-phantom]{Lemma}
\label{lemma-characterize-quasi-finite}
Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
$\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
\item The following are equivalent
\begin{enumerate}
\item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
\end{lemma}
```

LINUX SOURCE CODE

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

LINUX SOURCE CODE

```c
/*
 * If this error is set, we will need anything right after that BSD.
 */
static void action_new_function(struct s_stat_info *wb)
{
  unsigned long flags;
  int lel_idx_bit = e->edd, *sys & ~((unsigned long) *FIRST_COMPAT);
  buf[0] = 0xFFFFFFFF & (bit << 4);
  min(inc, slist->bytes);
  printk(KERN_WARNING "Memory allocated %02x/%02x, "
    "original MLL instead\n"),
    min(min(multi_run - s->len, max) * num_data_in),
    frame_pos, sz + first_seg);
  div_u64_w(val, inb_p);
  spin_unlock(&disk->queue_lock);
  mutex_unlock(&s->sock->mutex);
  mutex_unlock(&func->mutex);
  return disassemble(info->pending_bh);
}

static void num_serial_settings(struct tty_struct *tty)
{
  if (tty == tty)
    disable_single_st_p(dev);
  pci_disable_spool(port);
  return 0;
}
```

LINUX SOURCE CODE

```
/*
 *  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 <asm/io.h>
#include <asm/prom.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)
```

한글 자동 띄어쓰기

```
wsp$predict("크리스마스는친구와함께!",best_epoc = 1)
```

```
## [1] "크리스마스는 친구와 함께 !"
```

```
wsp$predict("대표직을사퇴한그는새로운사업을시작했다.",best_epoc = 1)
```

```
## [1] "대표직을 사퇴한 그는 새로 운사업을 시작했다. "
```

```
wsp$predict("일정한조건에따르면-자유롭게-이것을재배포할수가있습니다.",best_epoc = 1
```

```
## [1] "일정한 조건에 따르면 -자유롭게 -이것을 재배포할 수가있습니다. "
```

(image source : http://freesearch.pe.kr/archives/4617)

감사합니다!