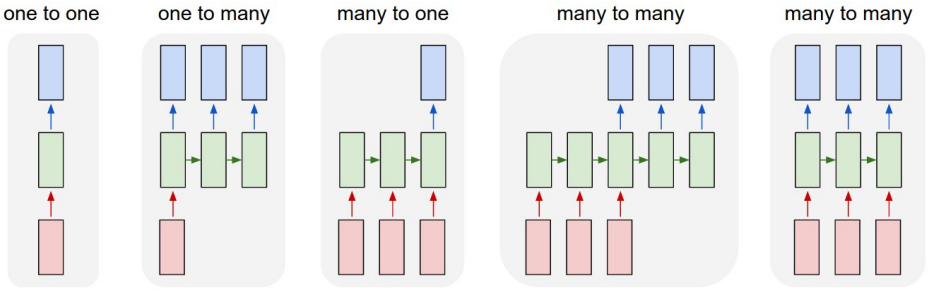
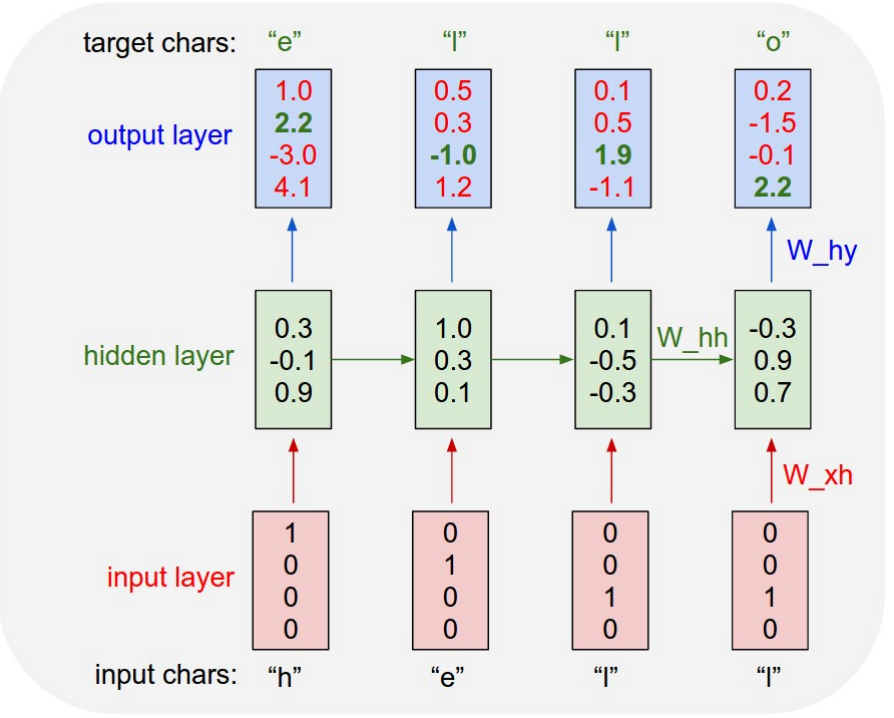
LSTM 資料

- date: 2021_0709
- title: リカレントニューラルネットワーク概説
- author: 浅川伸一





source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

単純再帰型ニューラルネットワークSRN

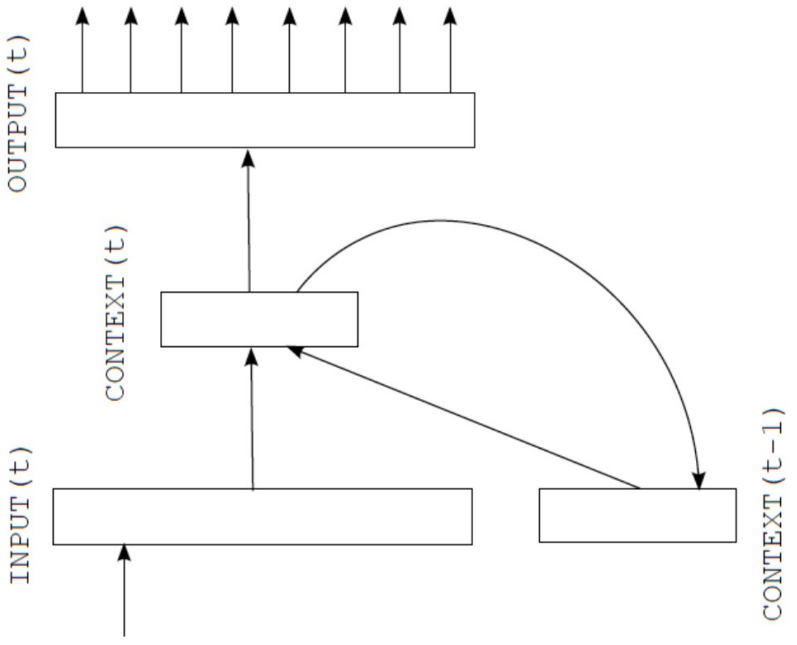


Figure 1: *Simple recurrent neural network.*

Mikolov (2010) Fig. 1

単純再帰型ニューラルネットワークSRN

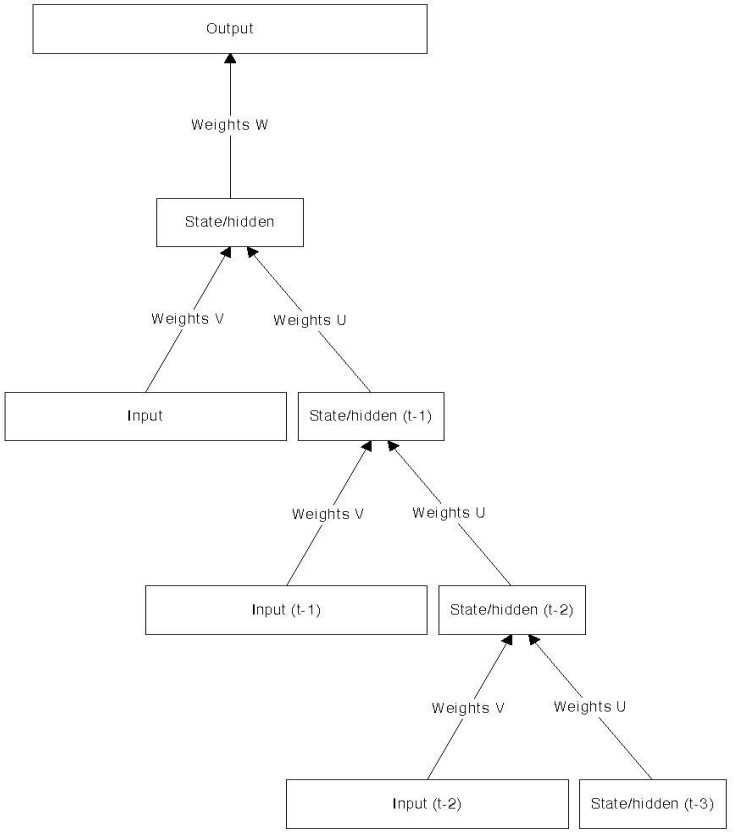
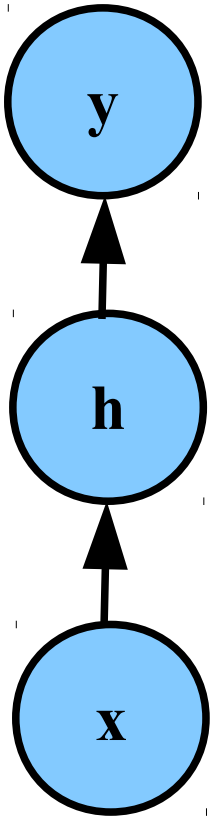


Figure 5: The effect of unfolding a network for BPTT ($\tau = 3$).

Booden (2001) Fig. 5

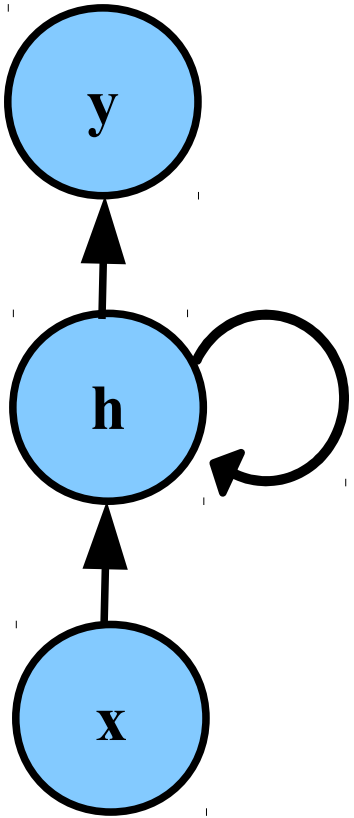
表記と基本グラフ



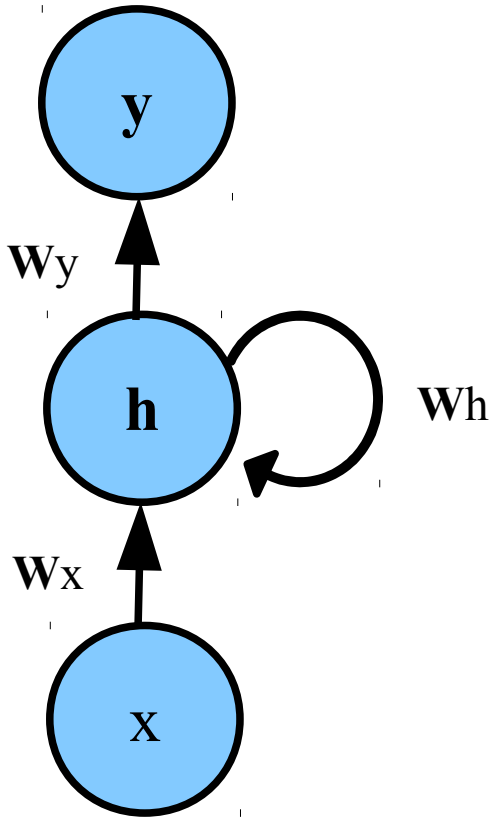
y: 出力層ニューロン

h: 中間層ニューロン

x: 入力層ニューロン



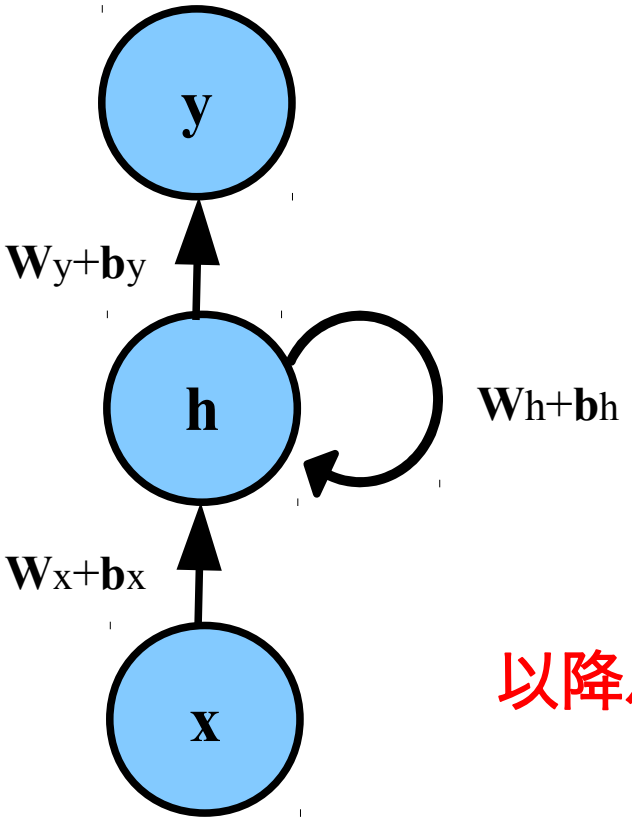
再帰結合（recurrent connections)



Wy:結合係数行列(中間から出力)

Wh:結合係数行列(再帰結合)

Wx:結合係数行列(入力から中間)

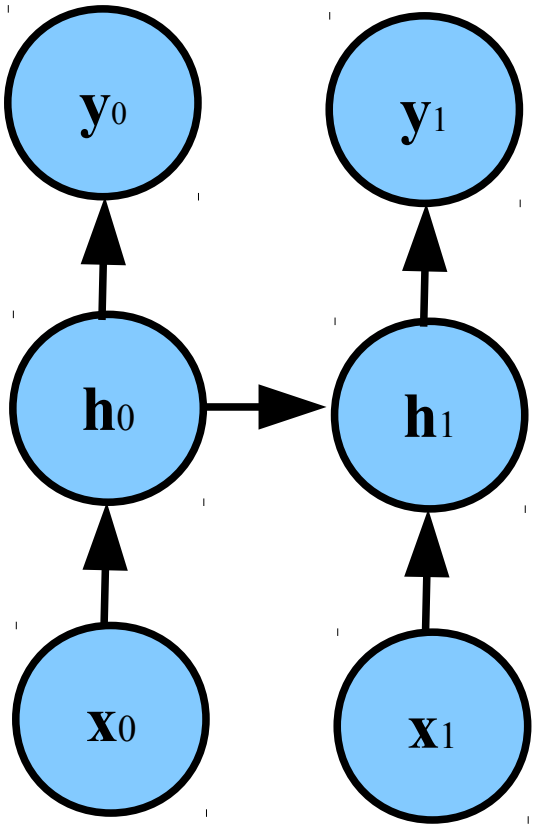


b_y:バイアス（中間から出力）

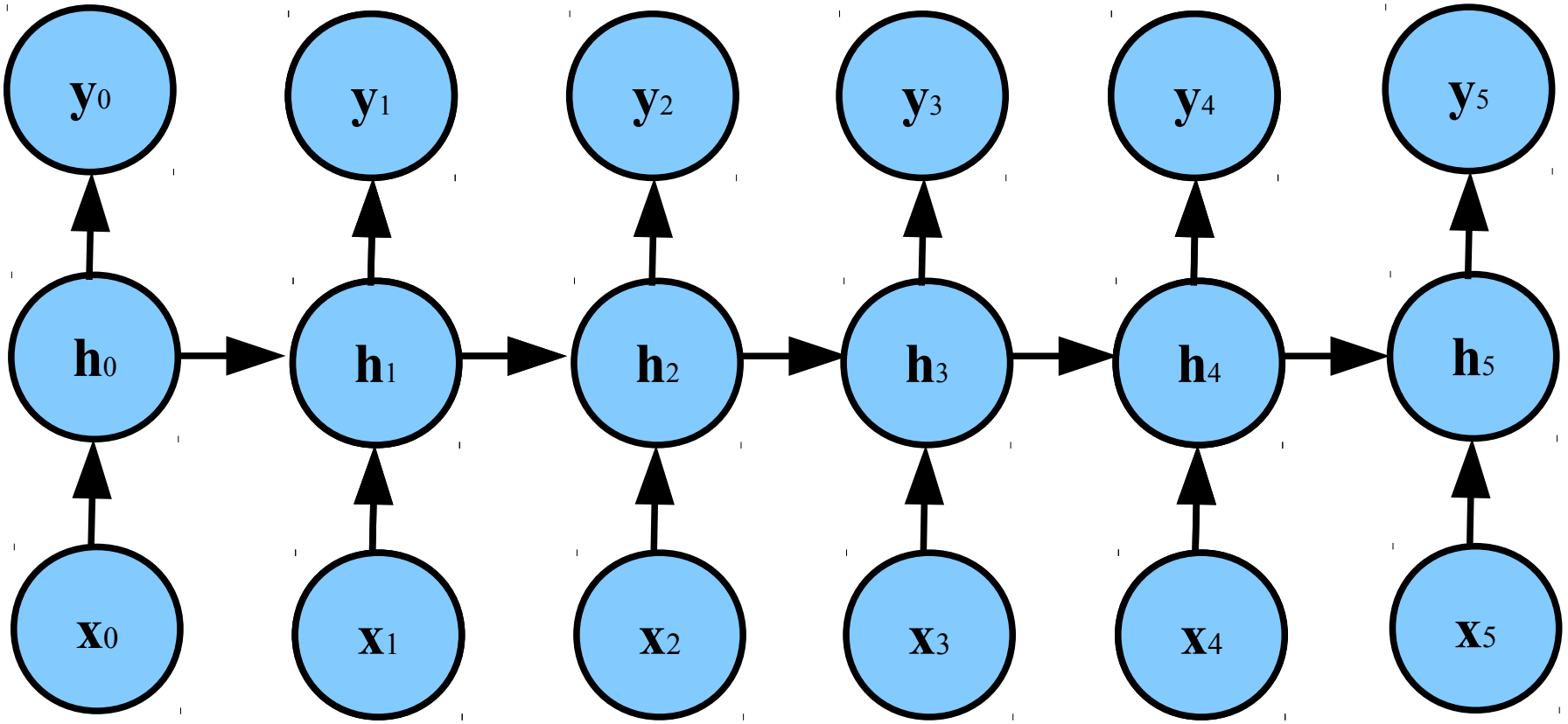
b_h:バイアス（再帰結合）

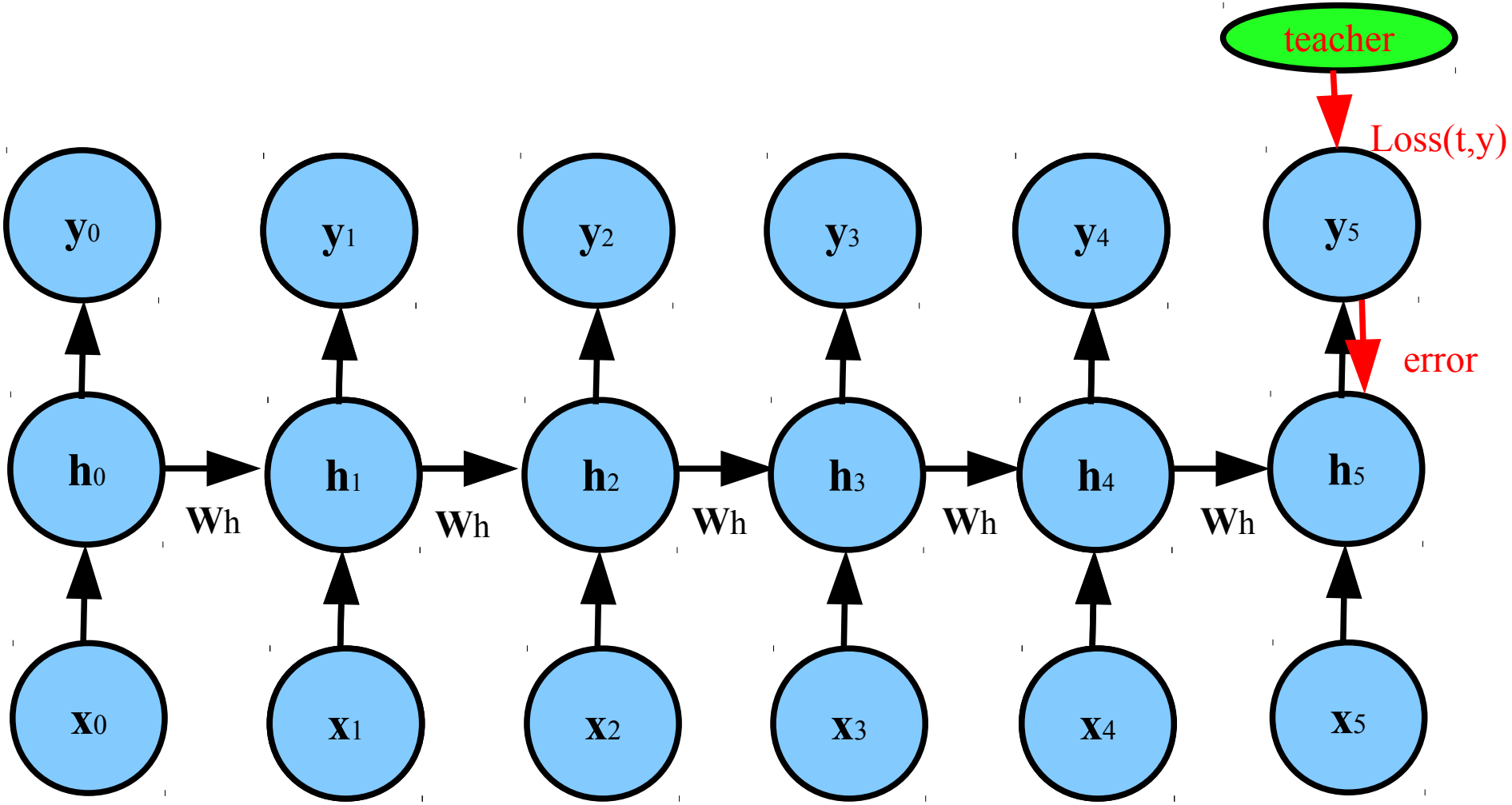
b_x:バイアス（入力から中間）

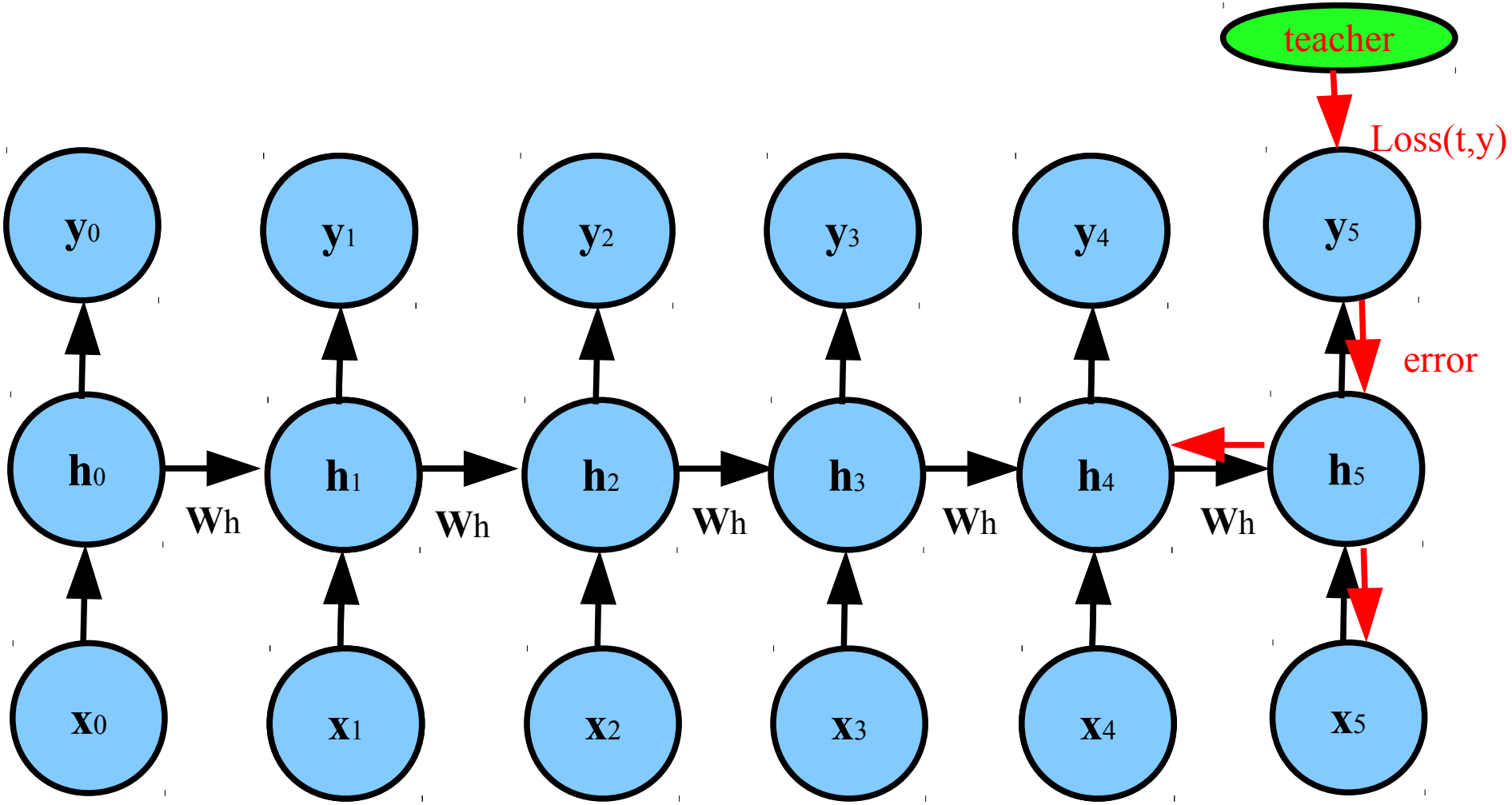
以降バイアス項は省略

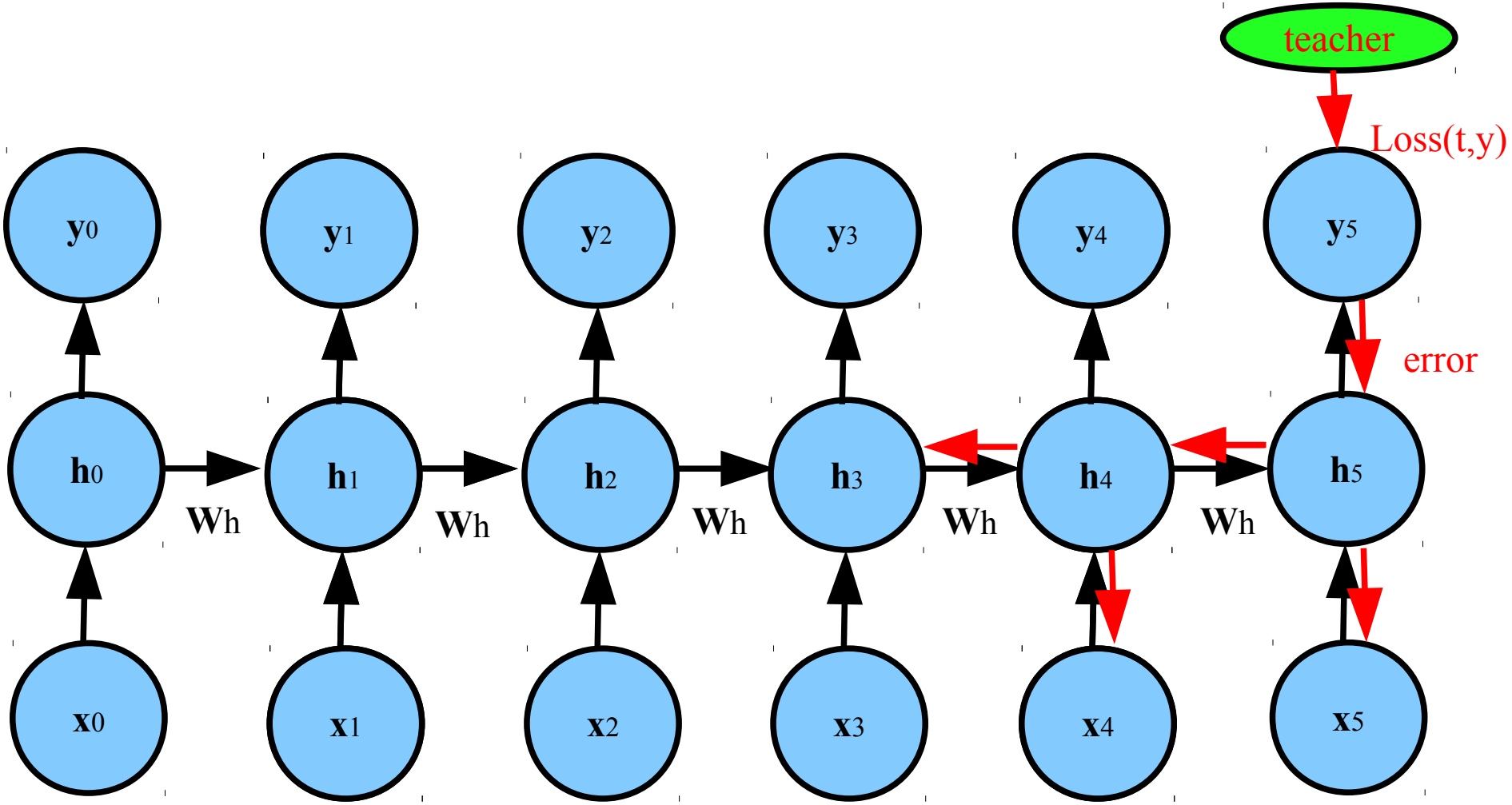


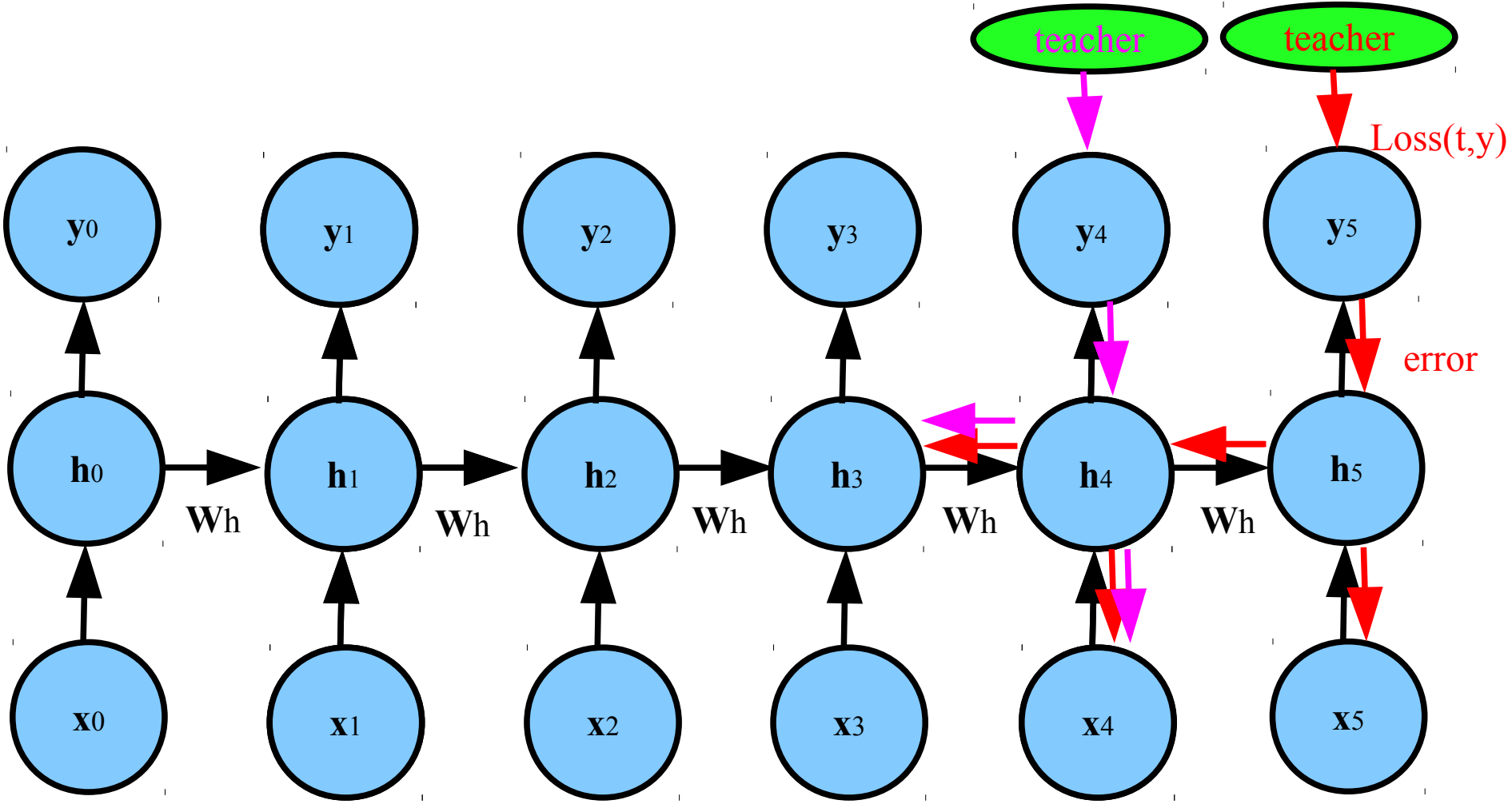
Digits subscripted indicate time
τ:= 0...
下付き添字は時刻を表す。
カッコで表記する流儀もある
(e.g. x(t))

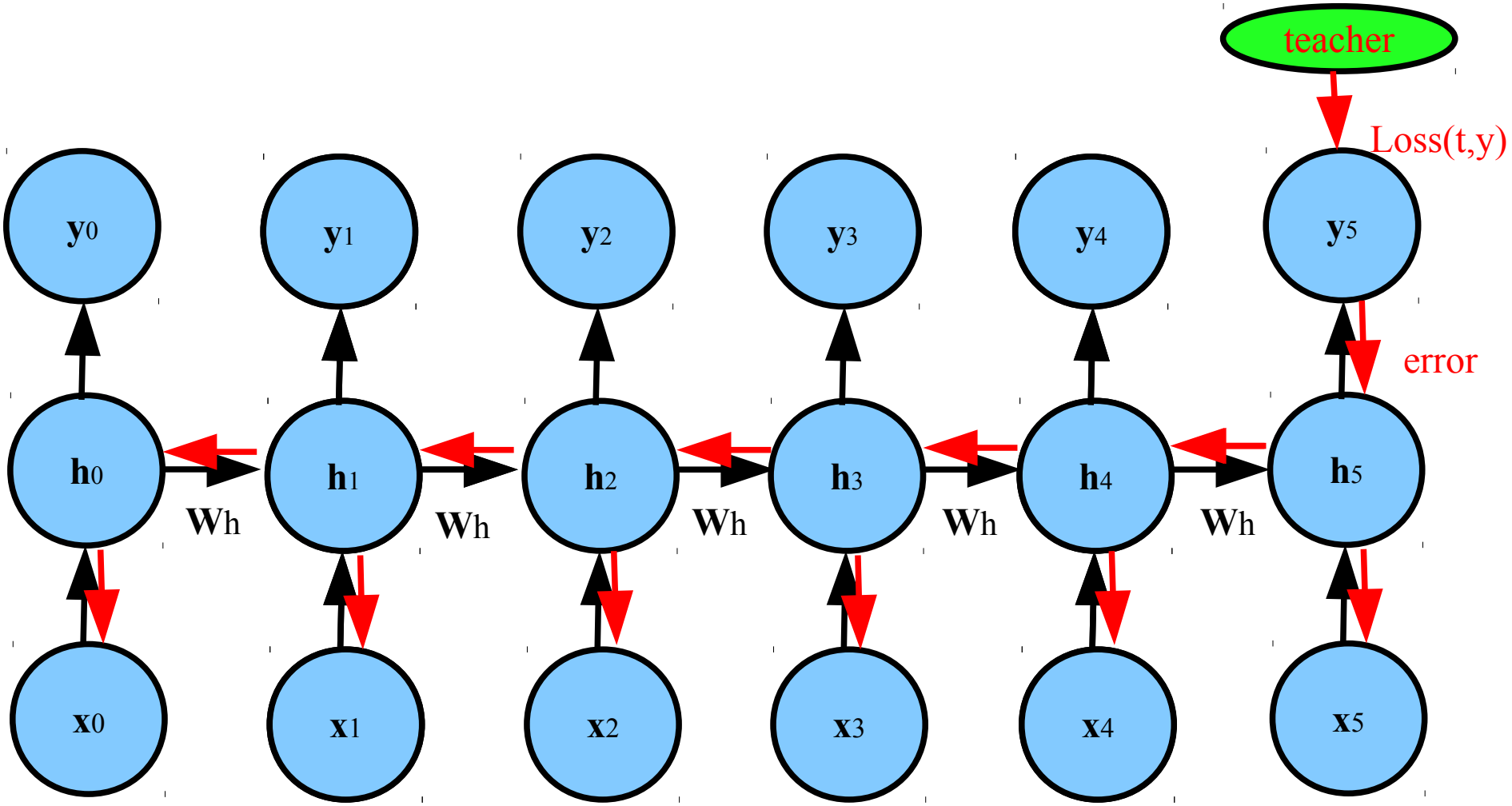

y0
y1
y2
y3
y4
y5
h0
h1
h2
h3
h4
h5
x0
x1
x2
x3
x4
x5


teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5

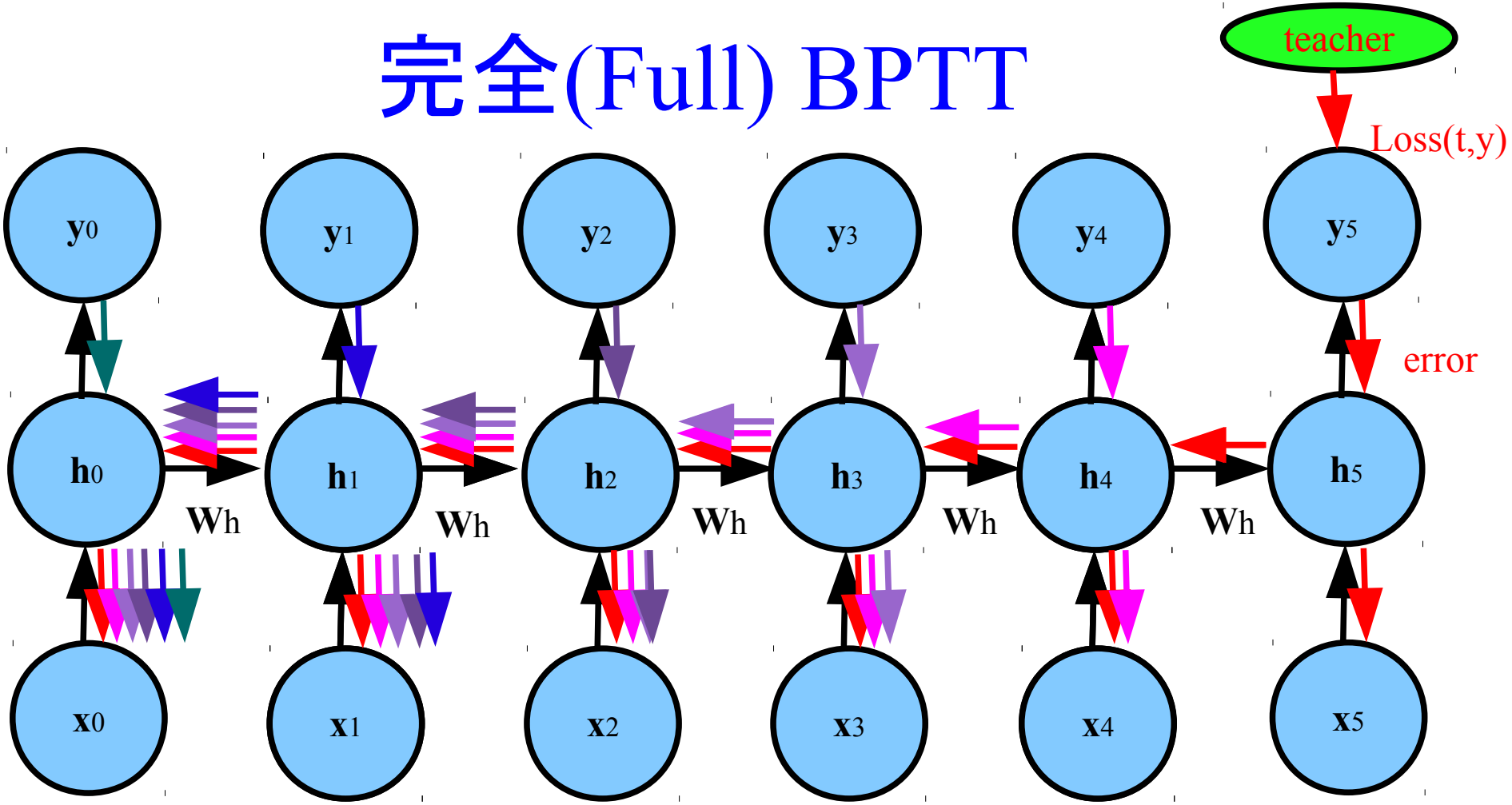

teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5


teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5

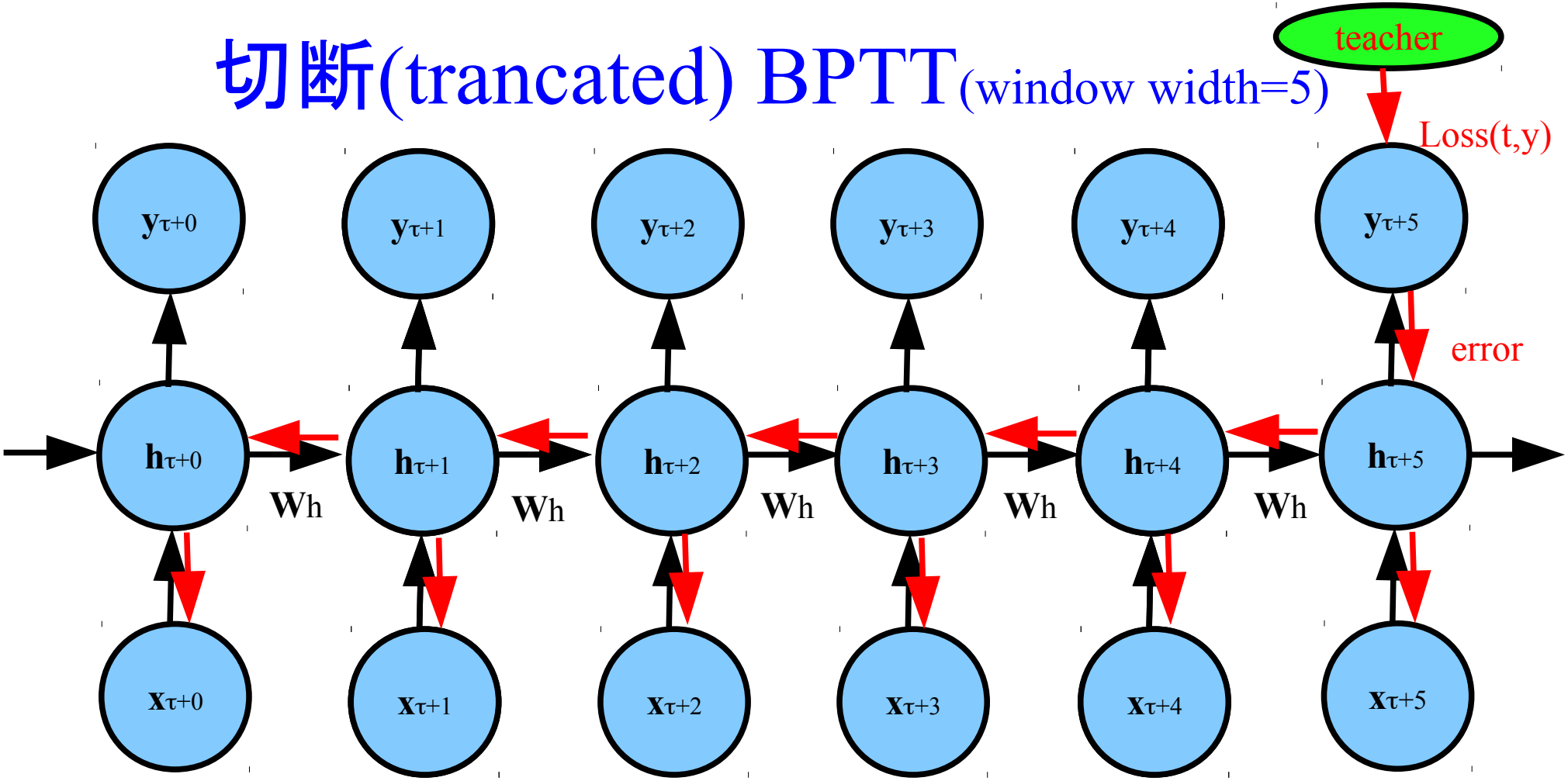

teacher
teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5


teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5

完全(Full) BPTT

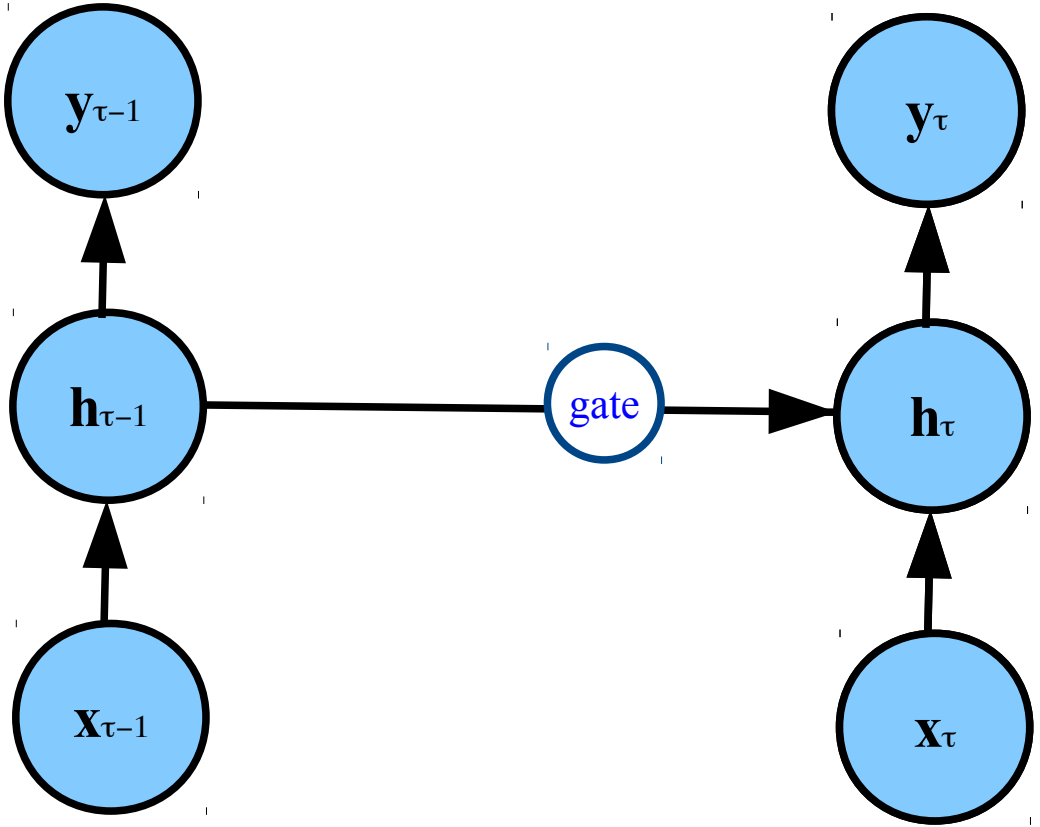

teacher
Loss(t,y)
y0
y1
y2
y3
y4
y5
error
h0
h1
h2
h3
h4
h5
Wh
Wh
Wh
Wh
Wh
x0
x1
x2
x3
x4
x5

切断(trancated) BPTT(window width=5)

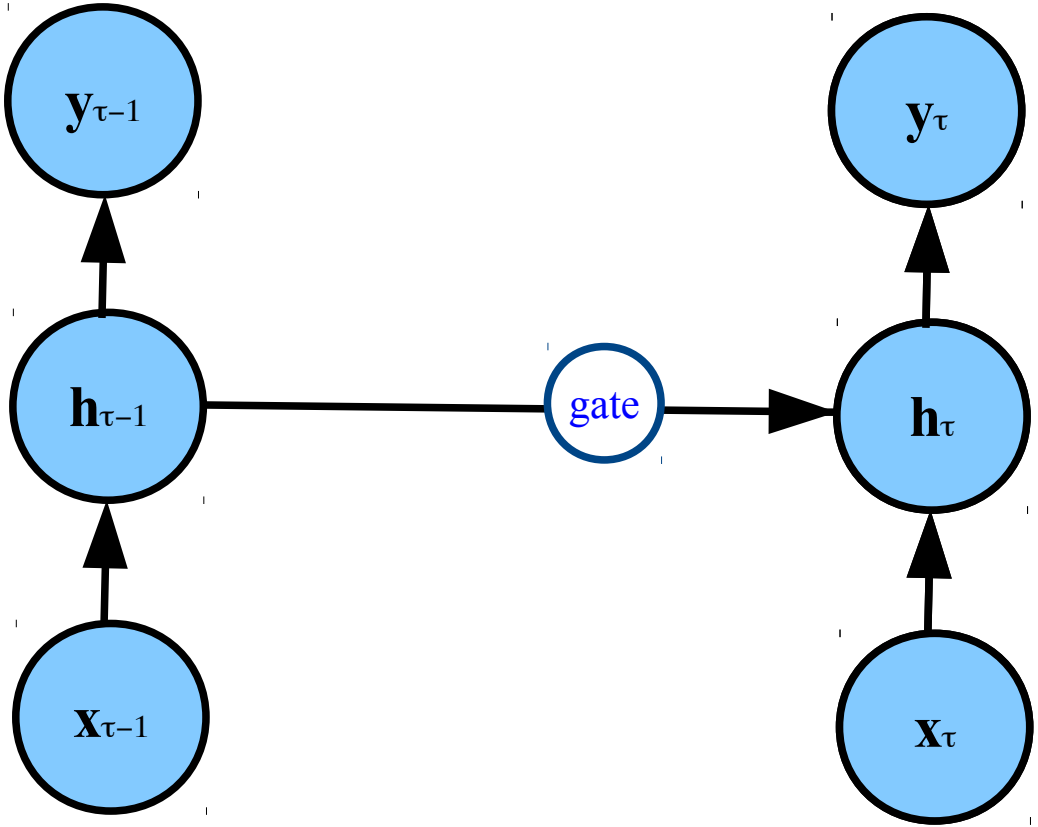


改良可能？
Can we improve?

ゲートの導入 introducing gates to control hidden state

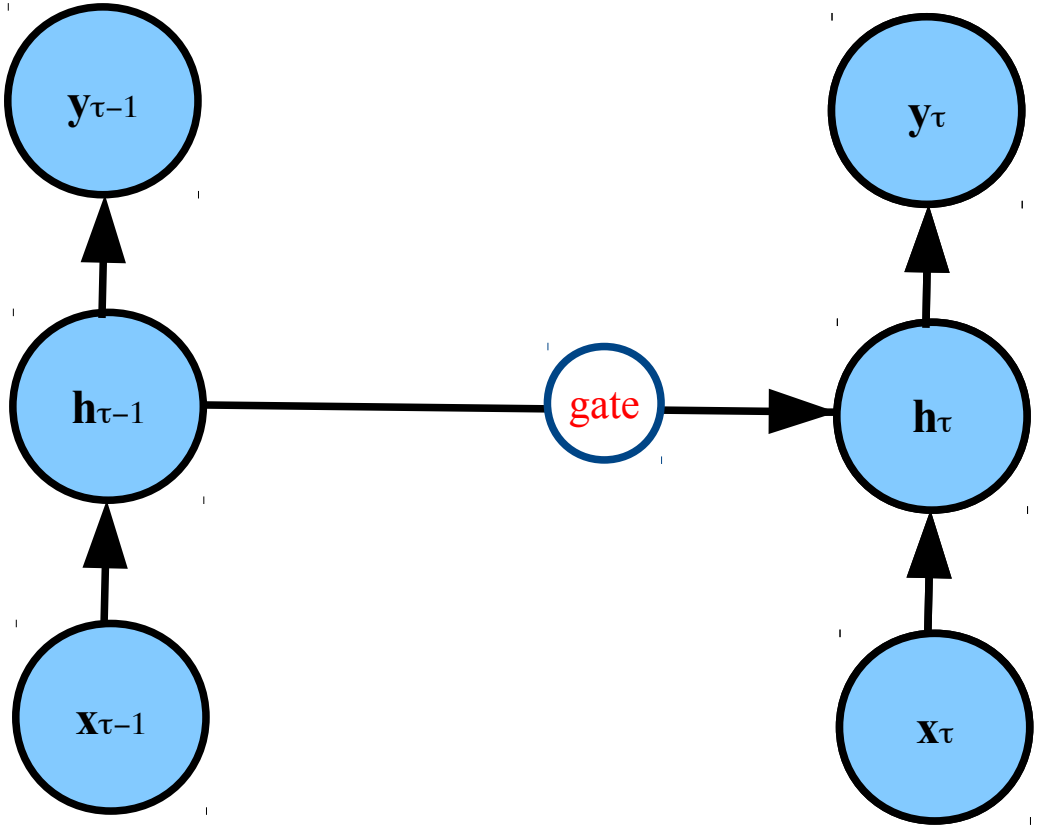


ゲートの導入 introducing gates to control hidden state



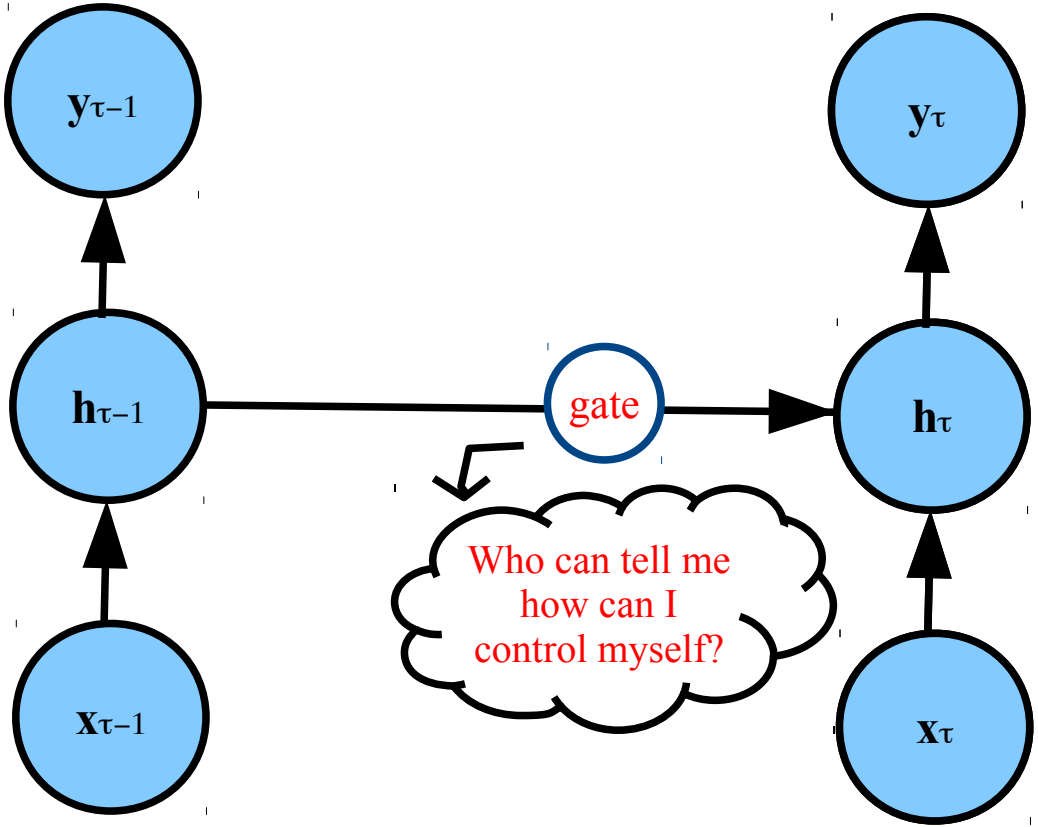
でも,
なぜゲート?
Why gates?

忘却ゲートの導入



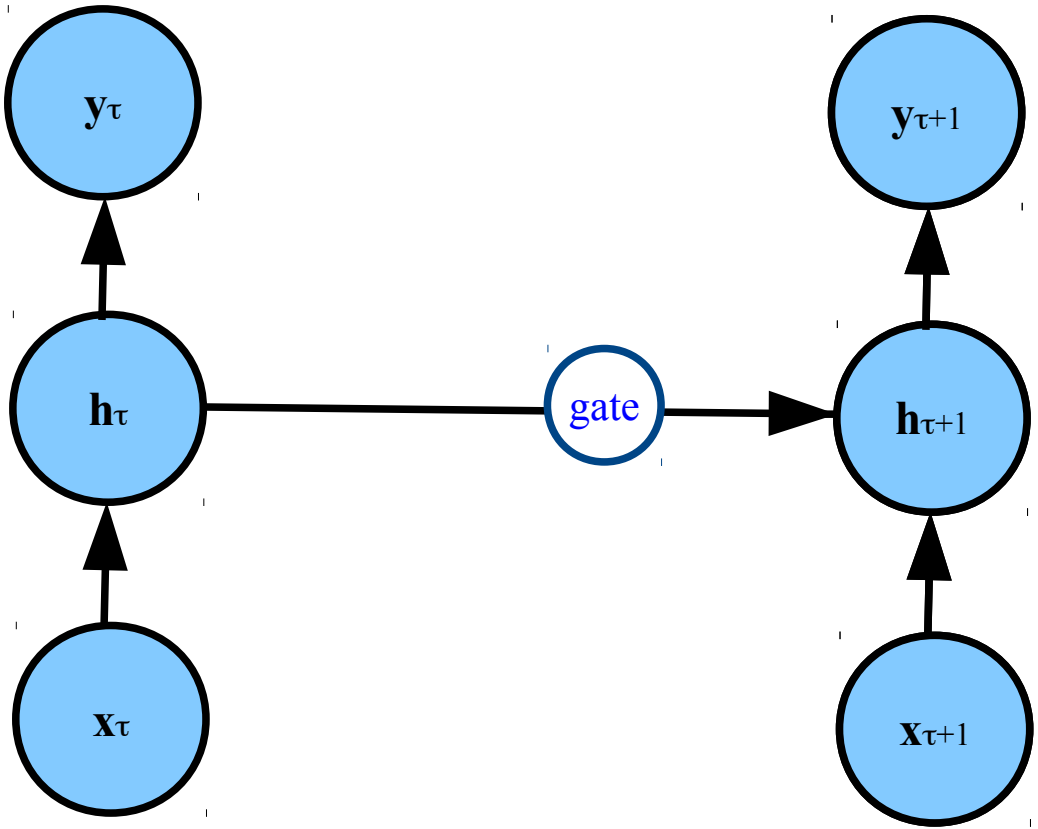
Who can control gates?
誰がどうやってゲート制御？

忘却ゲートの導入



Who can control gates?
誰がどうやってゲート制御？

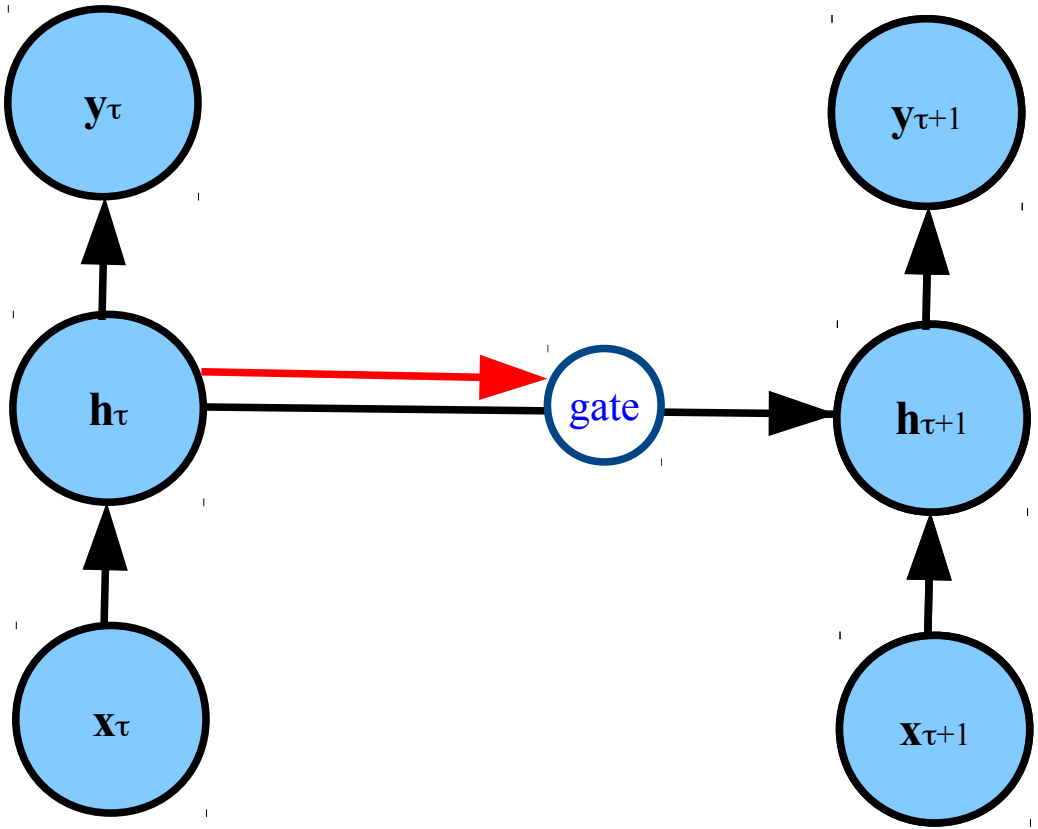
忘却ゲートの導入



who can control gates?
誰がどうやって
ゲートを制御？

３つ候補

忘却ゲートの導入

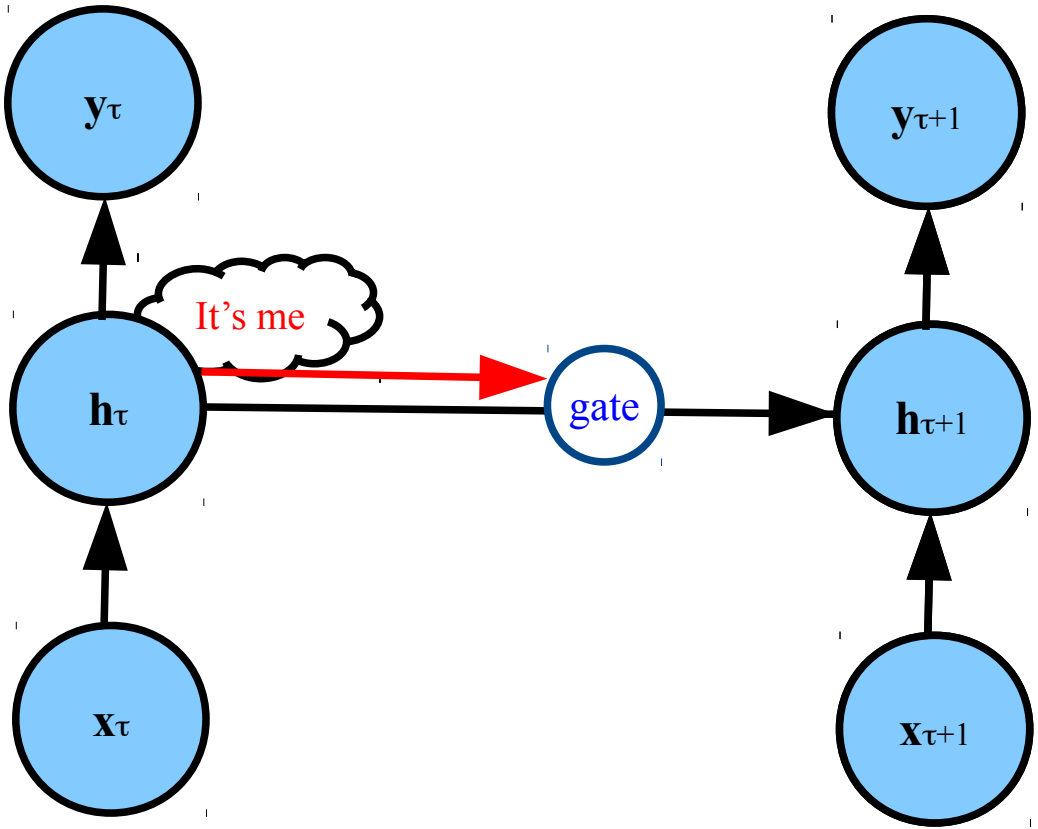


who can control gates?
誰がどうやって
ゲートを制御？

３つ候補

1. hτ

忘却ゲートの導入

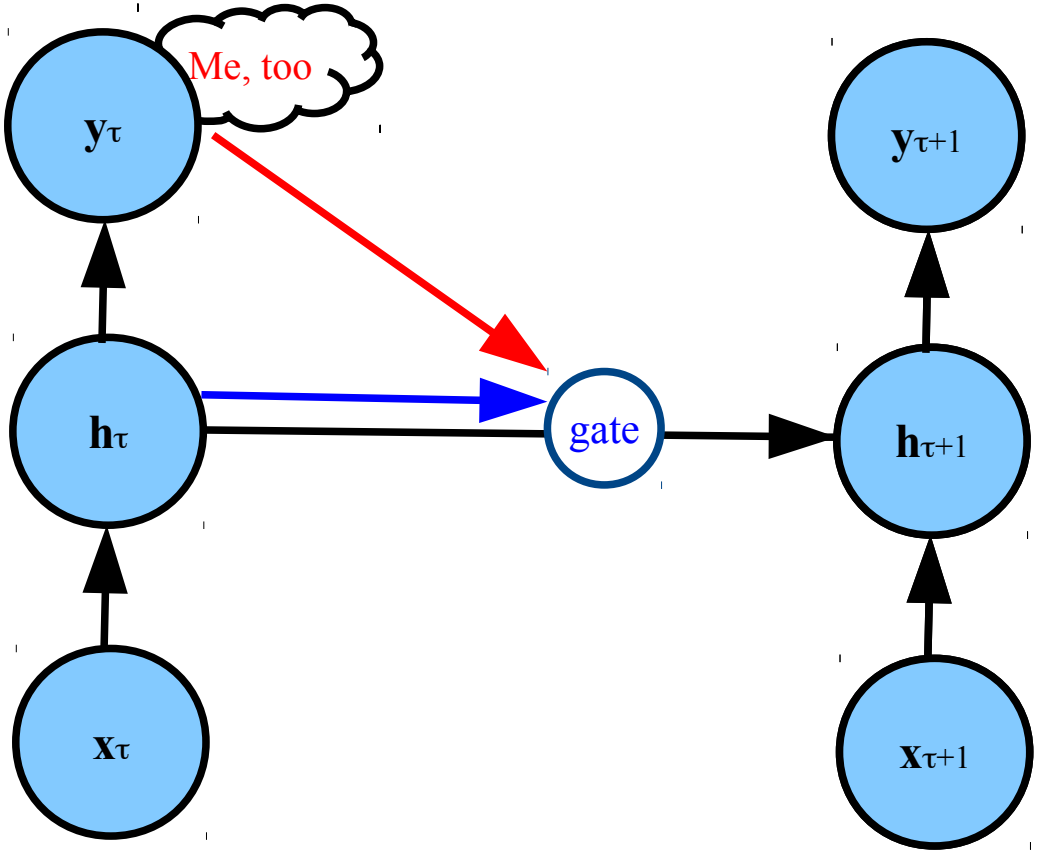


who can control gates?
誰がどうやって
ゲートを制御？

３つ候補

1. hτ

忘却ゲートの導入

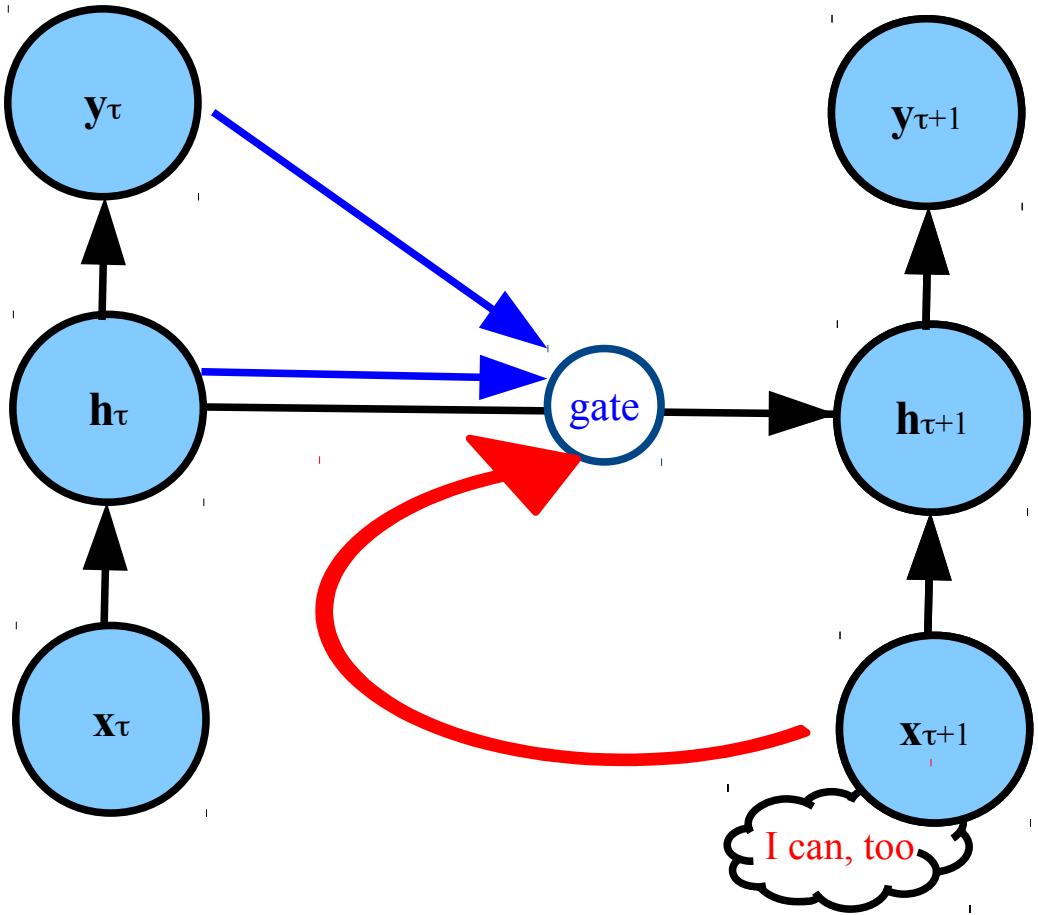


who can control gates?
誰がどうやって
ゲートを制御？

３つ候補

1. hτ
2. yτ

忘却ゲートの導入

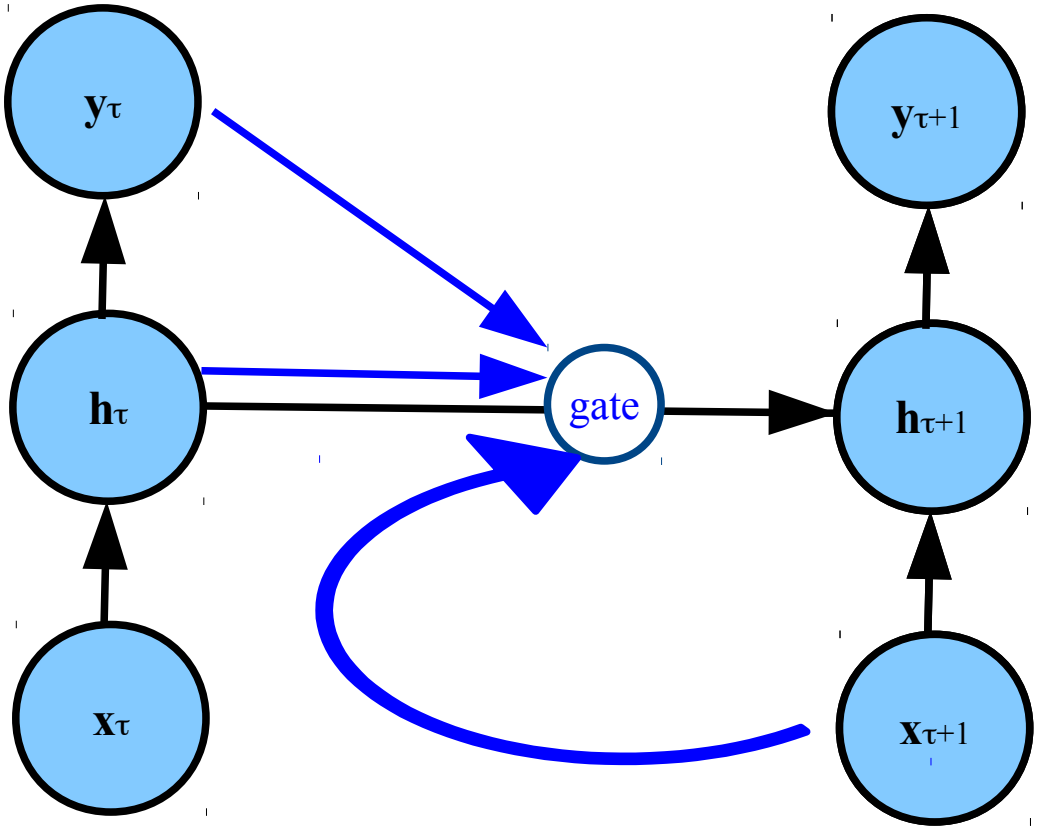


who can control gates?
誰がどうやって
ゲートを制御？

３つ候補

1. hτ
2. yτ
3. Xτ+1

忘却ゲートの導入

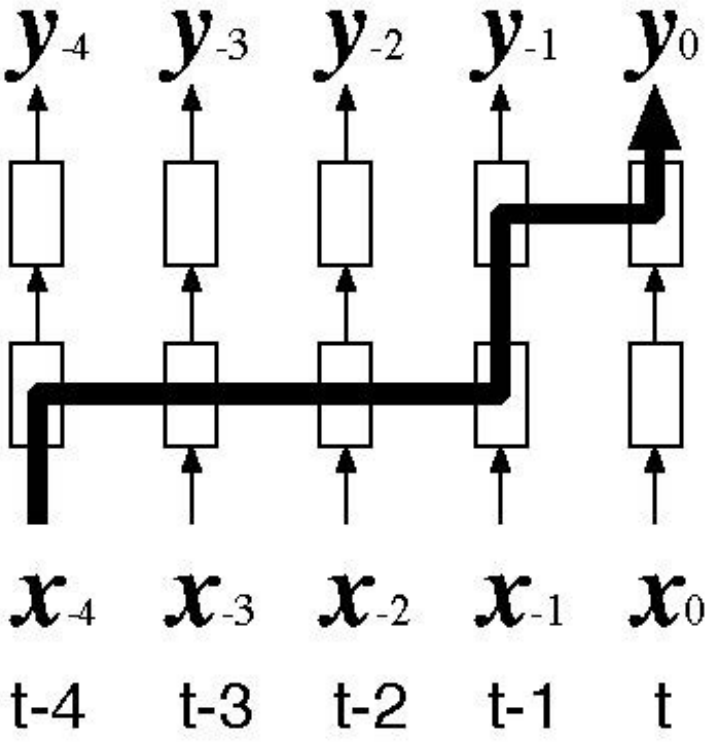


1. h_τ
2. y_τ　　ゲート制御
3. $x_{\tau+1}$

$$h_{\tau+1} = h_\tau \ \sigma(x)$$

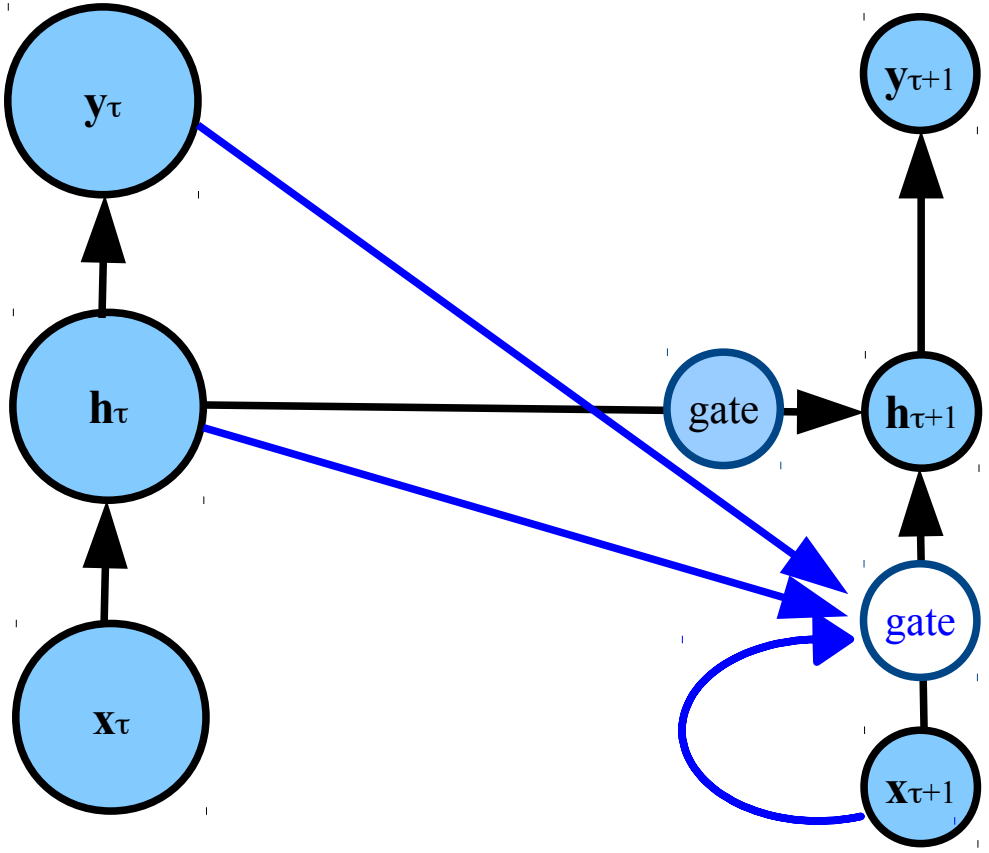
- $\sigma(x) = (1+e^{-x})^{-1}$
- $x = \mathbf{W}_f (y_\tau + h_\tau + x_{\tau+1})$

ゲートによって長距離依存LTDを解消可能



もっと改良可能？
Can we improve more?

入力ゲートの導入

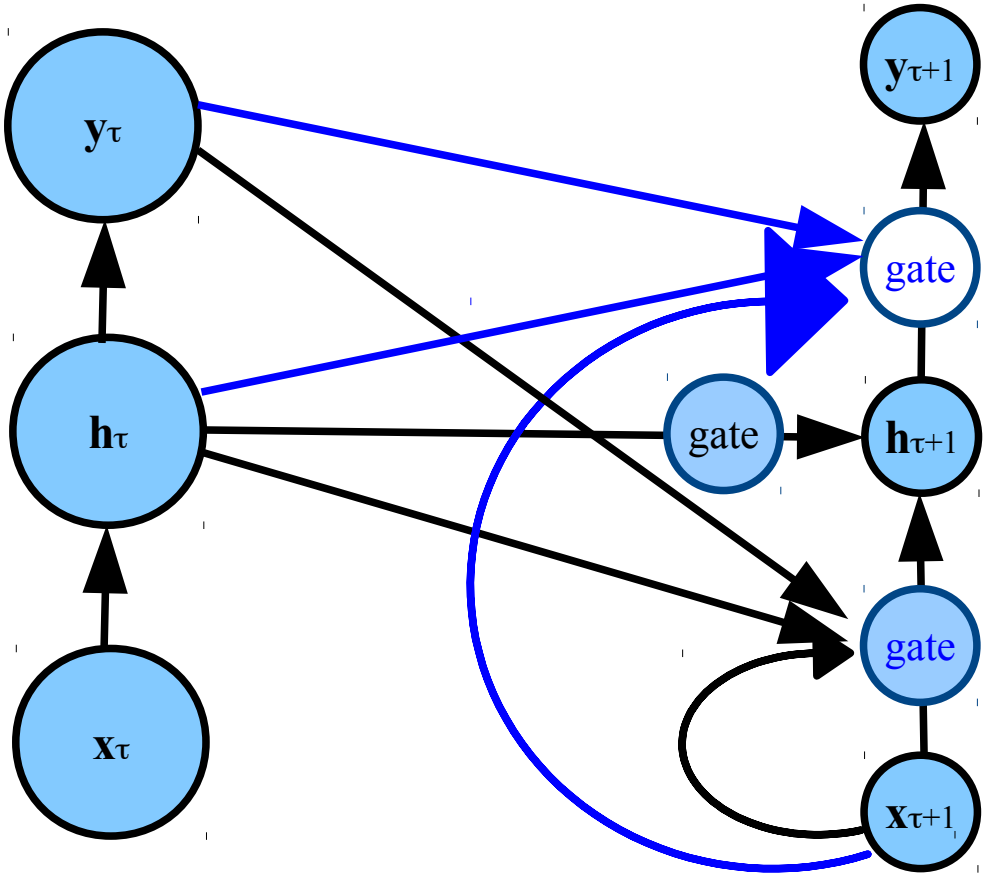


$$h_{\tau+1} = h_\tau \ \sigma(\mathbf{w}(h_\tau + x_{\tau+1}))$$

- $\sigma(x) = (1+e^{-x})^{-1}$
- $x = y_\tau + h_\tau + x_{\tau+1}$

もっともっと可能？
You need more?

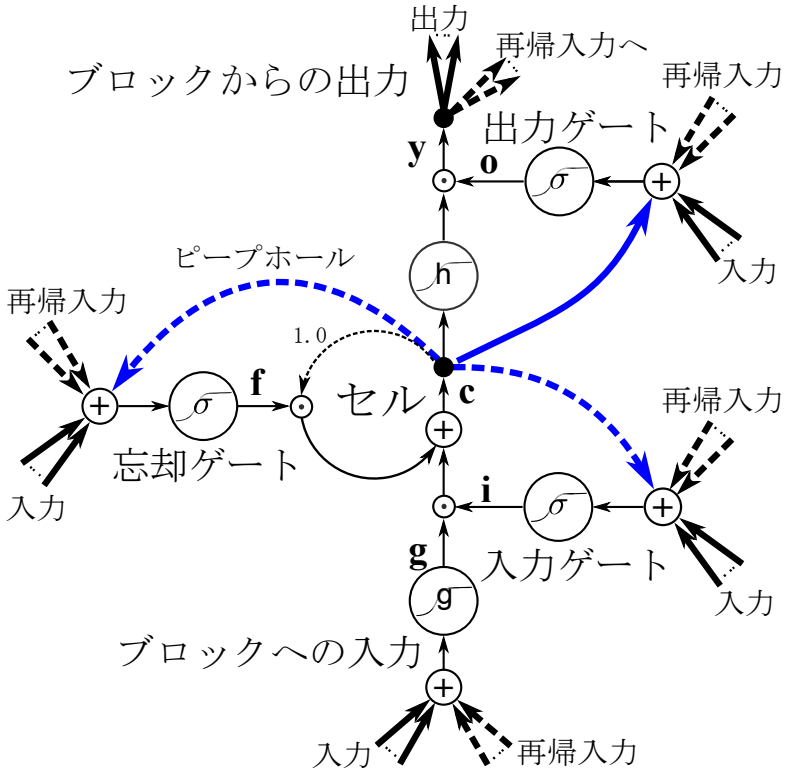
出力ゲートの導入



$$h_{\tau+1} = h_\tau \ \sigma\left(\mathbf{W}(\mathbf{h}_\tau + \mathbf{x}_{\tau+1} + \mathbf{y}_{\tau+1})\right)$$

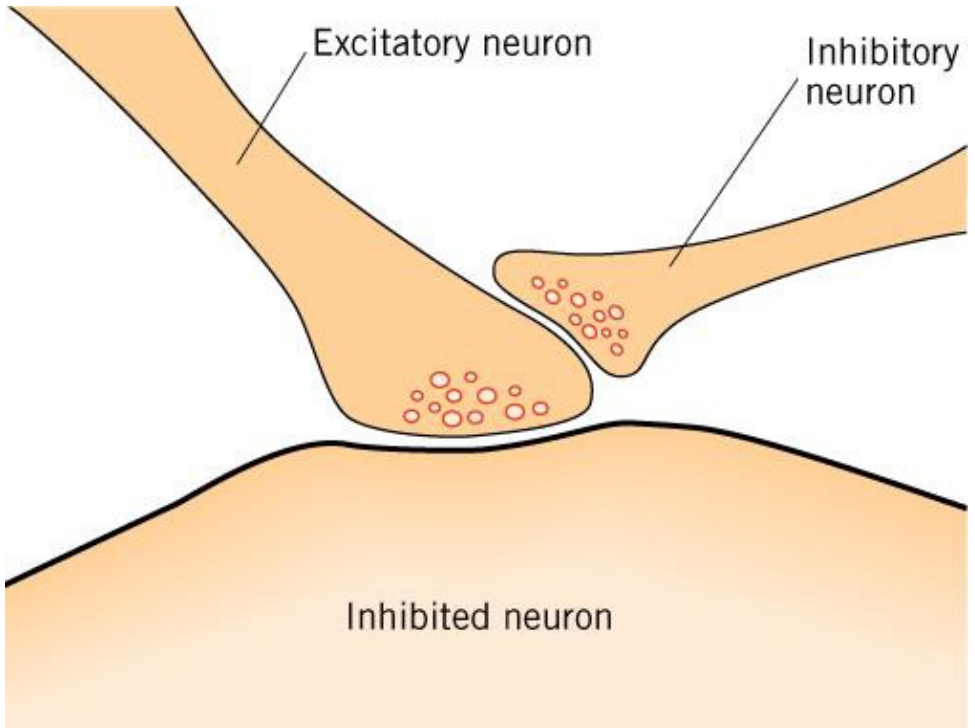
- $\sigma(x) = (1+e^{-x})^{-1}$
- $x = y_\tau + h_\tau + x_{\tau+1}$

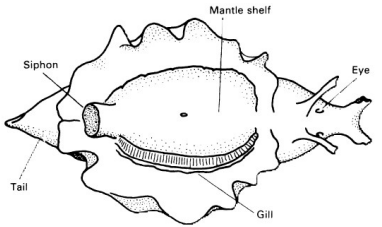
LSTM

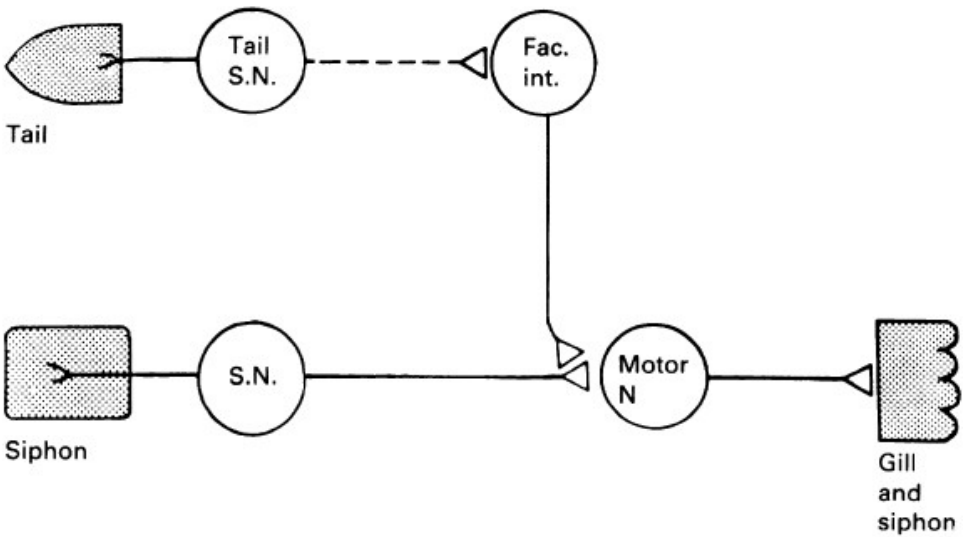


$$
\begin{aligned}
i_t &= \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right), \\
f_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right), \\
o_t &= \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right), \\
g_t &= \phi\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right), \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t, \\
y_t &= o_t \odot \phi\left(c_t\right)
\end{aligned}
$$

LSTMの生理学的対応物







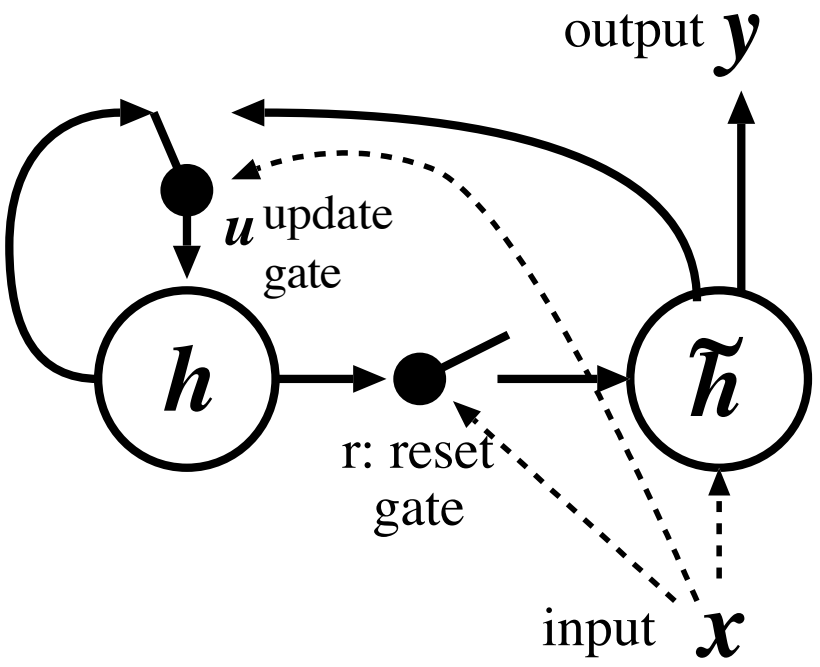
http://kybele.psych.cornell.edu/~edelman/Psych-2140/week-2-2.html

How does LSTM work?

1. LSTM replaces logistic or tanh hidden units with "memory cells" that can store an analog value.
2. Each memory cell has its own input and output gates that control.
3. There is a forget gate which the analog value stored in the memory cell decays.
4. For periods when the input and output gates are off and the forget gate is not causing decay, a memory cell simply holds its value over time.

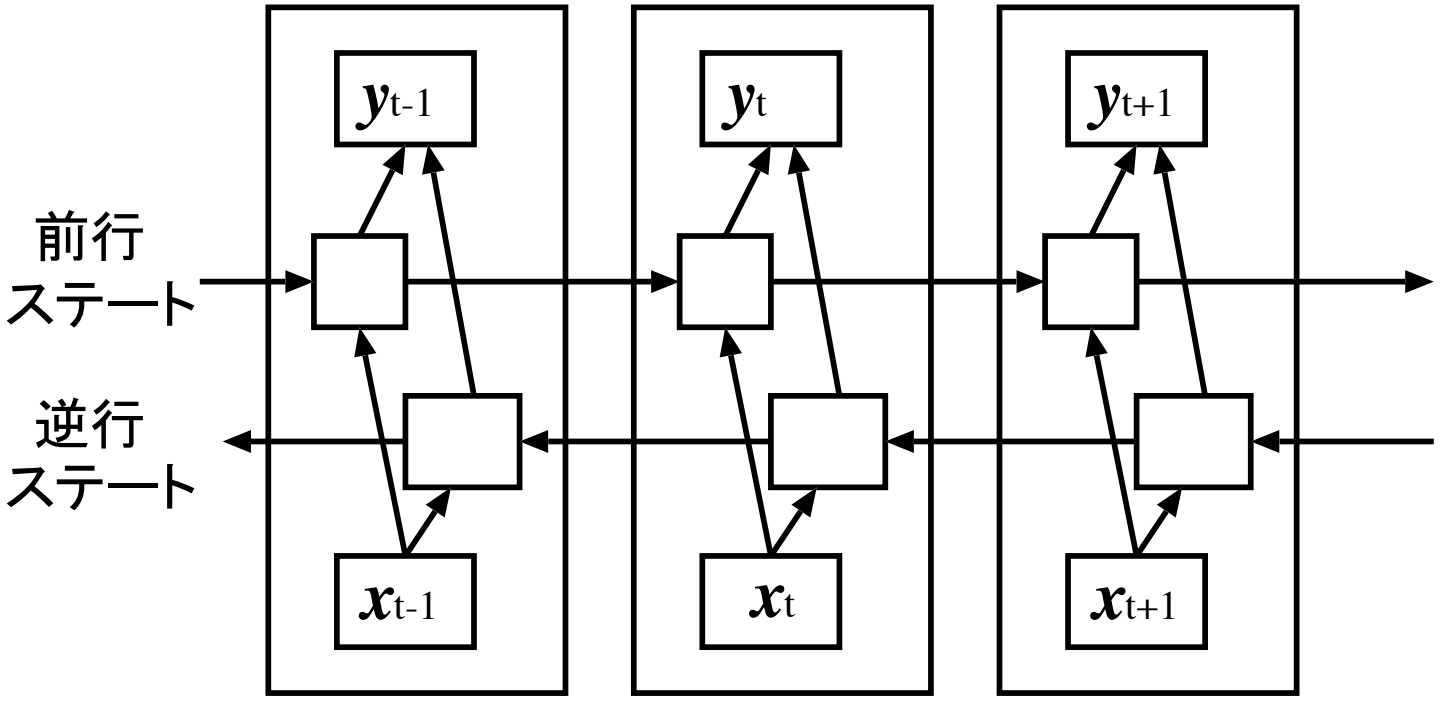
Le, Jaitly, & Hinton (2015)

別モデル GRU（An alternative of the LSTM)

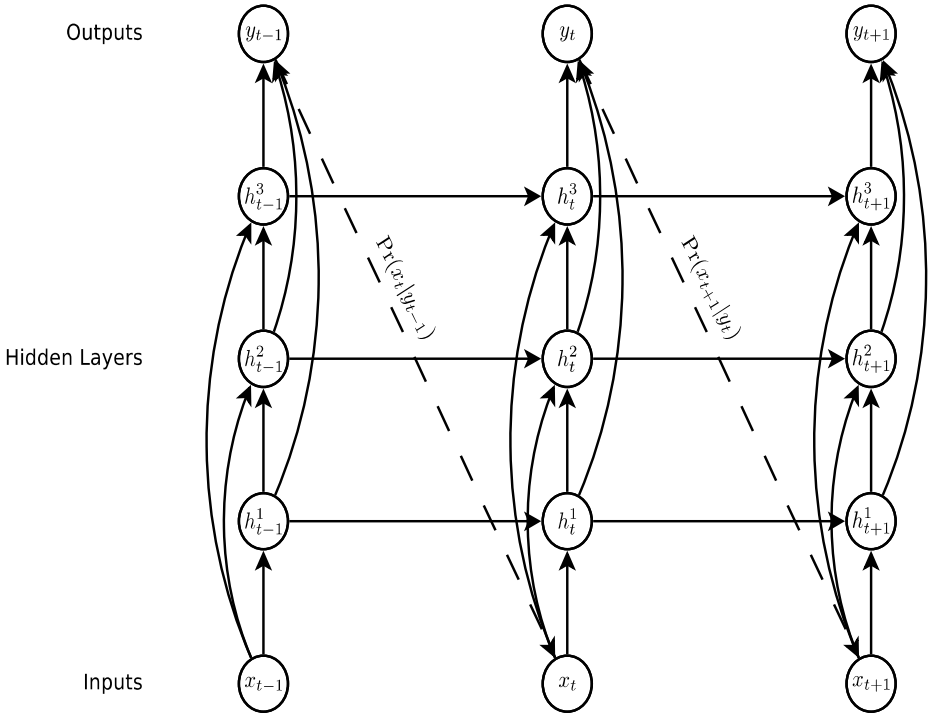


$$
\begin{aligned}
u_t &= \sigma\left(W_u x_t + U_u h_{t-1}\right), \\
h_t &= \phi\left(W x_t + U_h\left(u_t \odot h_{t-1}\right)\right), \\
r_t &= \sigma\left(W_r x_t + U_r h_{t-1}\right), \\
\tilde{h}_t &= \left(1 - r_t\right) h_t + r_t \tilde{h}_{t-1}, \\
y_t &= W_y \tilde{h}_t
\end{aligned}
$$

双方向RNN (Bidirectional RNN)



グレーブス (Graves, 2013)の生成 LSTM



7–1 リカレントニューラルネットワーク

Recurrent Neural Network

はじめに

リカレントニューラルネットワークとは，フィードバック結合（feedback connection）を有するネットワークである．ネットワーク内に再帰結合が存在すると，現時刻における情報を処理する際に過去の情報を再帰結合から受け取り利用する．これにより，リカレントニューラルネットワークは時系列情報を扱うことが可能となる．フィードバック結合の時間遅れを変動項と考えることで，種々の実際のニューロンの活動をモデル化することが可能である．リカレントニューラルネットワークは離散時間を仮定した場合と連続時間を仮定する場合とがあるが，ここでは主に前者を取り上げる．図 1 (a) はリカレントニューラルネットワークの簡単な例である．再帰結合行列を $\mathbf{W}$ とする．この時間発展を考えると，再帰結合行列を共有する多層ニューラルネットワーク（図 1 (b)）となる．

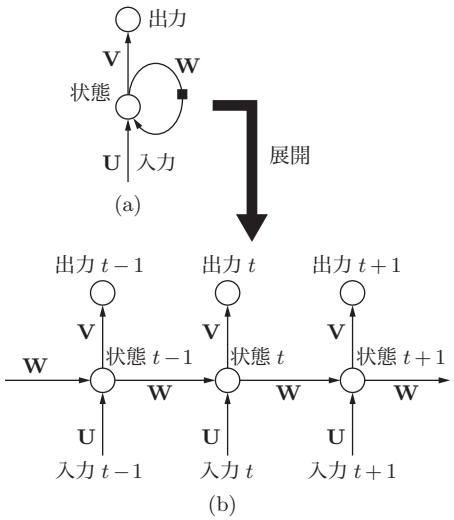


図 1 リカレントニューラルネットワーク (a) とその時間発展 (b)

後者は力学系（dynamical system）として記述される [1]．歴史的には，各時刻に単位時間幅の入力情報を与え，再帰結合を仮定しないモデルであるネットトーク（NETtalk）[2] が始まりである．時刻 $t+1$ におけるユニット i の出力 $x_i(t+1)$ は，バイアス項を無視すれば，x の生起確率を p として，

$$x_i(t+1) = \operatorname{argmax} p\left(x_i(t+1) \,|x_t(t), c(t); \theta\right) \quad (1)$$

$$= f_i\left(\sum_{j \in U} w_{ij} x_j(t)\right) \quad (2)$$

と表記される．ここで，f_i は任意の出力関数，w_{ij} は i, j 間の結合係数，$x_j(t)$ は y_i への入力信号である．$c(t)$ は文脈層と呼び，1 時刻前の状態を保持する．θ はニューラルネットワークを定めるパラメータ集合であり，結合係数やバイアス項を含む．

初期のモデルとしては，ジョーダンネットワーク（Jordan network）[3]，エルマンネットワーク（Elman network）[4] がある．両モデルとも 1 時刻前の状態を文脈層に保存し，通常のバックプロパゲーション法によって学習を行う．一方，時系列情報を一定の時間窓の期間維持し，時間窓内の情報について学習を行うアルゴリズムとして，BPTT (back-propagation through time) [5], RTRL (real time recurrent learning; 実時間リカレント学習) [6] が提案された．BPTT と RTRL は後の節で解説する．2010 年代以降，リカレントニューラルネットワークは時系列情報処理 [7] の発展として，チューリングマシン（Turing machine）との相同性 [8]，擬似プログラムコードの生成 [9]，英語からフランス語へ自動翻訳 [10]，写真からの脚注の生成 [11]，など，応用が盛んな領域となっている．類似した用語で表記されるモデルに，リカーシブニューラルネットワーク（recursive neural network）[12] があるが，これは自然言語処理に特化したモデルである．

単純再帰型ニューラルネットワーク

図 2 (a) にジョーダンネットワーク，(b) にエルマンネットワークを示す．

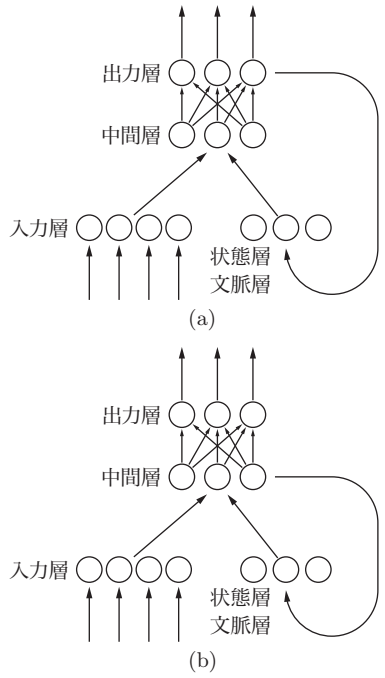


図 2　(a) ジョーダンネットワークと，(b) エルマンネットワーク

両者を併せて単純再帰型ニューラルネットワーク (simple recurrent neural network) と呼ぶ．ジョーダンネットワークは 1 時刻前の出力層の情報を，またエルマンネットワークは 1 時刻前の中間層の内容を保持しておき，現時刻における入力情報と文脈層情報は中間層で計算される．文脈層（context layer）とは，1 時刻前の状態を保持しておく層である．1 時刻前の状態をコピーするのであるから，ジョーダンネットワークにおける出力層ユニット数と文脈層ユニット数と同数であり，同様にエルマンネットワークの中間層ユニット数と文脈層ユニット数は同数である．ジョーダンネットワークの文脈層の信号が出力層へと帰還する図が描かれている文献が存在するが，原典のジョーダンネットワーク [3] では，出力層からの文脈層情報は中間層への帰還信号である．ジョーダンネットワークとエルマンネットワークの相違は，文脈情報として出力表象を用いるか，内部表象を用いるかの違いである．すなわち，1 時刻前の状態を利用する際に出力表象を用いたほうが制御に有用であるロボットアーム，姿勢，運動などの動作制御の場合にはジョーダンネットワークが用いられ，一方，1 時刻前の内部状態を扱ったほうが有利な言語情報処理，文法判断などではエルマンネットワークが用いられることが多かった．

学習については，ジョーダンネットワークとエルマンネットワークとも 1 時刻前の状態を文脈層にコピーするので，中間層から見ると入力情報が増えたこと以外に，再帰結合のない通常のフィードフォワード型のニューラルネットワークの学習との間で相違はない．任意の目標関数を各結合係数で微分したバックプロパゲーションによる学習も行われる．

BPTT と RTRL

BPTT [13][6] は back-propagation through time の略，RTRL [5] は real time recurrent learning の略である．上記のジョーダンネットワークとエルマンネットワークの両モデルは，再帰（帰還信号）が出力層あるいは中間層から戻ってくることを区別したモデルである．この両者を区別せず，一般にリカレントニューラルネットワークを構成することを考える．このとき，学習について時間をどのように考慮するかについて BPTT と RTRL の二つのモデルが提案された．BPTT と RTRL とを統一して表記することも可能である [14]．RTRL は “real time” との命名から連続時間を考慮したモデルと誤解される場合もあるが，学習時に「即時」に学習できるという意味であり，離散時間を仮定したモデルである．一方，“through” を用いる BPTT は状態変動の履歴を全時刻について保持し，時刻を “through” すなわち串刺しにして計算するため，このような命名で区別されている．したがって，BPTT は学習時に全時刻の全状態を保持しておく必要がある．このため，扱う系列が長くなると，学習のために必要となる記憶容量や学習時間が問題となる．実際には，一定の時間幅で切断したり，過去の影響を時間幅に応じて減衰させたりする実装もある．

ある時刻における出力は，入力と 1 時刻前の状態に依存する．しかし，1 時刻前の状態は，そのときの入力とさらに 1 時刻前の状態とに依存するので RTRL も時間情報を扱いうる．換言すれば，学習時に考慮する時間幅を h とすると，RTRL は $\mathrm{BPTT}_{h=1}$ でもある．さらに，$\mathrm{BPTT}_{h=1}$，かつ，帰還信号の発生源に制約を設けたモデルは，ジョーダンネットワークやエルマンネットワークと見なしうる．時刻 t における出力層ユニット k の誤差を，d_k を教師信号として以下のように定義する．

$$e_k = d_k(t) - y_k(t) \tag{3}$$

ここで，$y_k(t)$ は出力信号を表す．システムの 2 乗誤差の総和 $J(t)$ を

$$J(t) = -\frac{1}{2}\sum_{i\in U}[e_i(t)]^2 \tag{4}$$

とすれば，時間幅 $(t', t]$ についての総誤差 $J^{\text{total}}(t', t)$ は，以下のように表記できる．

$$J^{\text{total}}(t', t) = \sum_{\tau=t'+1}^{t} J(\tau) \tag{5}$$

学習則は，学習係数を η として，

$$\nabla_w J^{\text{total}}(t', t) = -\sum_{\tau=t'+1}^{t} \nabla_w J(\tau) \tag{6}$$

$$\Delta w_{ij} = \eta \frac{\partial J^{\text{total}}(t', t)}{\partial w_{ij}} \tag{7}$$

と書くことができる．

BPTT

BPTT では過去の状態を記憶バッファに保持し，保持された記憶状態への結合についてもバックプロパゲーション法を適用して結合係数を更新する．時刻 t における誤差 $J(t)$ の勾配を計算するために

$$\delta_i(t) = f_i' \left(\sum_{j \in U} w_{ij} x_j \right) e_i(t) \tag{8}$$

$$e_i(t-1) = \sum_{j \in U} w_{ji} e_i(t) \tag{9}$$

とし，上式に従って時間を遡り $t_0 + 1$ まで達すれば

$$\frac{\partial J(t)}{\partial w_{ij}} = \sum_{\tau=t_0+1}^{t} e(\tau) x_j(\tau - 1) \tag{10}$$

として誤差を計算できる．上記をまとめると，

1. 現在の状態を入力とを履歴バッファに保存する
2. 現在の誤差をバックプロパゲーション法によって計算する
3. 全時刻における誤差の総和を計算する
4. 重みを更新する

を，全データで収束基準に達するまで繰り返すこととなる．BPTT の計算量のオーダーは $O(TN^2)$ となる．ここで，T は時間，N は中間層のユニット数である．BPTT は全時間について考慮するが，実際の計算においては時間幅 h を定める必要がある．これを BPTT_h と表記すれば，RTRL は BPTT の $h = 1$ なる特別な場合と見なしうる [14]．ジョーダンネットワークもエルマンネットワークも $\text{BPTT}_{h=1}$ に相当する．図 3 の実線矢印は情報の流れを，破線矢印は誤差の伝播を示している．各時刻において入出力は異なるが，各時刻における状態は保持され結合係数の更新のために用いられる．図中の数字は誤差の伝播されていく順番を示している．

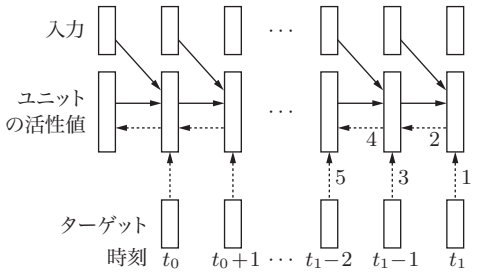


図 3 BPTT の概念図（文献 [6], p.448, Figure 4 を改変）

RTRL

BPTT は誤差勾配の計算に誤差情報の時間に関する逆伝播を用いた．これに対し，誤差勾配を順伝播させる方法が RTRL である．結合係数 w_{ij} に対するユニット k の影響を p_{ij}^k とし，すべてのユニット $(i, j, k \in U)$ について，

$$p_{ij}^k(t) = \frac{\partial y_k(t)}{\partial w_{ij}} \tag{11}$$

を定義する．時刻 t における誤差 $J(t)$ を各結合係数で微分した量を次式に従うと仮定する．

$$\frac{\partial J(t)}{\partial w_{ij}} = \sum_{k \in U} e_k(t) p_{ij}^k(t) \tag{12}$$

次の時刻の $p_{ij}^k(t+1)$ は，次式で表される．

$$p_{ij}^k(t+1) = f_k'\left(\sum(t+1)\right) \left[\sum_{l \in U} w_{kl} p_{ij}^t(t) + \delta_{ik} x_j(t) \right] \tag{13}$$

ここで，δ_{ij} はクロネッカーのデルタである．さらに，初期状態 t_0 においては，$p_{ij}^k(t_0) = \partial y_k(t_0) / \partial w_{ij} = 0$ と仮定すれば各時刻ステップにおける $p_{ij}^k(t)$ の量を計算可能である．この値と誤差との積によって誤差の勾配を求めることができる．この意味で，BPTT とは異なり RTRL は “即時” 的に計算可能である．

勾配消失問題と勾配爆発問題

バックプロパゲーション法における多層パーセプトロンの学習においては，勾配消失問題（gradient vanishing problem）により学習が進まないことが以前か

ら指摘されてきた [15]．Bengio ら [16] はこの問題を定式化した．

バックプロパゲーション法では，任意の課題における特定の出力層ユニットの誤差が下位層の全ユニットに伝播する．したがって，多層化されたニューラルネットワークで，活性化関数にシグモイド関数（$\sigma(x) = (1 + \exp -x)^{-1}$）を用いると，誤差関数を各結合係数で微分した値にシグモイド関数の微分が入る．

$$\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x)) \tag{14}$$

$0 \leq \sigma(x) \leq 1$ であるので，$d\sigma(x)/dx$ は微分するごとに，すなわち層を下るごとに小さくなってしまう．これが勾配消失問題の一因である．

一方，リカレントニューラルネットワークにおける学習の際，BPTT の時間窓が広くなると勾配が発散する場合がある．最小化すべき損失関数 L の勾配をパラメータ θ で時間窓 t_w まで計算すると次式のようになる．

$$\frac{\partial L}{\partial \theta} = \sum_{1 \leq k \leq t} \frac{\partial L}{\partial x_t} \frac{\partial x_t}{\partial x_k} \frac{\partial x_k}{\partial \theta} \tag{15}$$

ここで，$\partial x_t/\partial x_k$ は中間層 h での時刻 t における時刻 k による微分である．階層が深くなると，$\partial h_t/\partial h_{t-1}$ のヤコビアン（Jacobian）行列式の最大特異値に応じて拡大縮小が起こる．特異値が 1 より小さければ勾配消失問題となり，1 より大きければ勾配爆発問題（gradient exploding problem）となる [16]．リカレントニューラルネットワークの学習においては，式 (15) で再帰的に勾配を計算した場合，ヤコビアン（Jacobian）の最大特異値分解（singular value decomposition; SVD）に応じて指数関数的に勾配が変化することになる．

勾配正規化

勾配消失問題を回避するために，次のような勾配正則化を行うことが提案されている．誤差関数 E のパラメータ x に関する勾配計算の際に，その 1 時刻後の微分量との比 $\partial x_{k+1}/\partial x_k$ を用いて，次式のような正規化を行う [17]．

$$\sum_k \left(\frac{\left\| \frac{\partial E}{\partial x_{k+1}} \frac{\partial x_{k+1}}{\partial x_k} \right\|}{\left\| \frac{\partial E}{\partial_{k+1}} \right\|} - 1 \right)^2 \tag{16}$$

勾配クリップ

勾配爆発問題に対する対処には，勾配値を一定範囲に収める勾配クリップ（gradient clip）が提案されている．勾配クリップは，勾配爆発が局在した小領域でしか起こらないという仮定に基づき，一定以上の勾配値に達した場合，強制的にその値を抑える [16]．以下のアルゴリズムではしきい値 θ を設定し，求めた勾配の絶対値がしきい値以上であればしきい値以下に変換している [17]．文献 [18] では $\theta = 1$ が用いられた．

1. $\hat{\mathbf{g}} \leftarrow \frac{\partial \varepsilon}{\partial \theta}$
2. $\hat{\mathbf{g}} \leftarrow \frac{\theta}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$,　　If $\|\hat{\mathbf{g}}\| \geq \theta$

LSTM

2010 年以降，リカレントニューラルネットワークについては LSTM（long short-term memory）を用いた研究が数多く発表されている．単純再帰型ニューラルネットワークでは，長距離間隙（long-time lag）あるいは長距離依存（long term dependency）を学習することが難しかった（収束に要する繰り返し回数が大きかったため）．これは，直前の情報を適切に利用することが困難だったからである（CEC（constant error carousel）の呪い）．LSTM はこれを解決するために，ゲートを設定し，ゲートの開閉によって 1 時刻前のシステムの状態からの影響を制御する．

図 4 に LSTM の概念図を示す．図で，情報は下から上へと流れる．この図は図 2 と異なり，全ネットワークを示しているのではなく，一つの LSTM セルを描いている．この LSTM セルを複数個用いてニューラルネットワークの 1 層が構成される．図の中央の $\mathbf{c}$ が記憶素子である．$\mathbf{c}$ の前後に二つのゲートが配置され，それぞれ入力ゲート，出力ゲートと呼ばれる．$\mathbf{c}$ の自己フィードバックに関するゲートは忘却ゲートと呼ばれる．$\mathbf{c}$ の出力すなわち過去の状態からの再帰結合は，恒等関数 $f(x) = x, \forall x$ であるため，1.0 と表記している．$\mathbf{c}$ への入力は CEC と忘却ゲートの出力との積である．

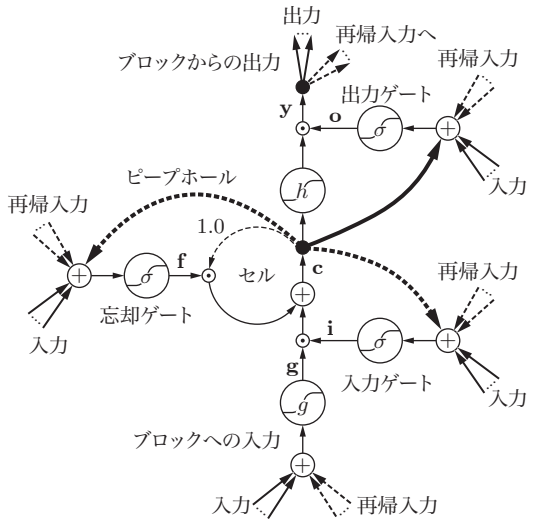


図 4 LSTM の概念図

出力ゲートと入力ゲートの役割は，入出力量をスケーリングすることでメモリセルにどの程度の誤差を学習するかを指示することである．入力ゲートによるスケーリングと出力ゲートのスケーリングにより誤差を開放するか否かの情報を，記憶ユニットが学習することになる．忘却ゲートは，メモリセルに対して，言い換えれば，1 時刻前の自身の保持している情報を利用するか否かを定めるために，ゲート開閉を行う．なお，原典の LSTM [19] においては忘却ゲートは存在せず，Gers ら [20] によって導入された．

LSTM は実用的な繰り返し回数で長期記憶（long-term memory）を制御できる．長期記憶は記憶セル $c^n_{t\in\mathcal{R}}$ のベクトルとして保持される．LSTM はネットワーク全体のアーキテクチャと活性化関数とは独立に長期記憶を保持する能力を持つ．LSTM は記憶セルを上書きし，保持し，引き出すことが可能である．

図 4 内の表記と対応させて，時刻 t における入力を $\mathbf{i}_t$，忘却ゲートを $\mathbf{f}_t$，出力ゲートを $\mathbf{o}_t$，LSTM ユニットの出力を $\mathbf{g}_t$，記憶セルを $\mathbf{c}_t$ とすれば，LSTM のユニットは以下のように表される．

$$\mathbf{i}_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right) \tag{17}$$
$$\mathbf{f}_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right) \tag{18}$$
$$\mathbf{o}_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right) \tag{19}$$
$$\mathbf{g}_t = \phi\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \tag{20}$$
$$\mathbf{c}_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{21}$$
$$\mathbf{y}_t = o_t \odot \phi\left(c_t\right) \tag{22}$$

W は関連する結合係数行列，σ はロジスティック関数

$$\sigma\left(x\right) = \frac{1}{1+\exp\left(-x\right)} \tag{23}$$

であり，ϕ はハイパータンジェント（hyper tangent; tanh）

$$\phi\left(x\right) = \tan\left(x\right) = \frac{\exp\left(x\right)-\exp\left(-x\right)}{\exp\left(x\right)+\exp\left(-x\right)} = 2\sigma\left(2x\right) - 1 \tag{24}$$

である．また，$\odot$ は要素積（アダマール積; Hadamard product）である．LSTM ユニットの学習には，BPTT もしくは RTRL が用いられる．

ゲート付き再帰ユニット（gated recurrent unit; GRU）（図 5）の動作は，以下のように記述できる．

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1}\right) \tag{25}$$
$$\tilde{h}_t = \phi\left(W x_t + U\left(r_t \odot h_{t-1}\right)\right) \tag{26}$$
$$r_t = \sigma\left(W_r x_t + U_r h_{t-1}\right) \tag{27}$$
$$h_t = \left(1 - z_t\right) h_{t-1} + z_t \tilde{h}_t \tag{28}$$
$$y_t = W_y h_t \tag{29}$$

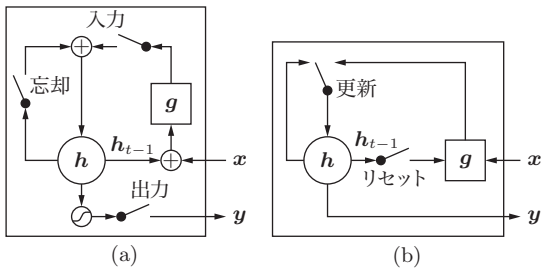


図 5 LSTM (a) とゲート付き再帰ユニット (b) の比較

2010 年代に入ってリカレントニューラルネットワーク，とりわけ LSTM を用いた研究が盛んである．写真からの脚注作成（neural image captioning）では，認識部分に深層学習（deep learning）を用い，出力に LSTM を用いる [21]．Donahue ら [22] は動画へ，Zaremba ら [23] は機械翻訳に，Graves ら [24] は発話解析に，それぞれ LSTM を応用している．

リカレントニューラルネットワークの総説論文としては，[25], [26] がある．LSTM の性能を評価した論文には，[27] を挙げることができる．リカレントニューラルネットワークを多層にした場合に，畳み込み深層学習と同じくドロップアウト（dropout）が行われる場合があるが，フィードフォワードネットワーク（feed-forward network; multi-layered network）型の畳み込

み深層学習と異なり，リカレントニューラルネットワークではドロップアウトの効果が認められないとの報告もあった．これに対し，Zaremba ら [23] はフィードフォワード結合のみをドロップアウトし，リカレント結合はドロップアウトを行わないことを提案している．

その他のリカレントニューラルネットワーク

古典的なジョーダンネットワーク，エルマンネットワークは 1 時刻前の状態と現在の入力に基づいて出力を計算する．これは 1 階差分に類比しうることを意味する．LSTM に見られるように，1 階差分の情報を断ち切って，さらに長期の記憶を呼び起こしたりする機構として，ゲートの存在が注目を集めている．すなわち，高階差分情報を活用するためにゲートの果たす役割が大切だとされ，ゲート付き再帰ユニット [28] など，新たなモデルも提案されてきた．どこまで過去の情報を保持し，どのような周期を仮定するのかについては，リカレントニューラルネットワークの持つ内部構造をランダム結合により実現したエコーステートネットワーク（echo state network; ESN）[29] がある．これは互いに疎に（sparse）結合した中間層ユニットを持つリカレントニューラルネットワークである．学習は出力ユニットについてのみ行われる．

系列データが与えられている場合，次刻の出力を予測する際に $t+\alpha$ なる未来のデータ系列も利用可能であれば，双方向リカレントニューラルネットワーク [30] と呼ばれるリカレントニューラルネットワークを用いた訓練が行われることがある．実際の予測をする際には無視すればよい．

参考文献

[1] Grossberg, S. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks*, Vol. 1, pp. 17–61, 1988.

[2] Sejnowski, T. J. and Rosenberg, C. R. Parallel Networks that Learn to Pronounce Eglish Text. *Complex Systems*, Vol. 1, pp. 145–168, 1987.

[3] Jordan, M. I. Serial Order: A Parallel Distributed Processing Approach. Technical report, University of California, San Diego, 1986.

[4] Elman, J. L. Finding structure in time. *Cognitive Science*, Vol. 14, No. 2, pp. 179–211, 1990.

[5] Williams, R. J. and Zipser, D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, Vol. 1, No. 2, pp. 270–280, 1989.

[6] Williams, R. J. and Zipser, D. Gradient-based Learning Algorithms for Recurrent Networks and Their Computational Complexity. In Chauvin, Y. and Rumelhart, D. E., editors, *Backpropagation: Theory, Architectures, and Applications*, pp. 434–486. Lawrence Erlbaum Associate, New Jersey, 1995.

[7] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and Tell: A Neural Image Caption Generator. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pp. 3156–3164, 2015.

[8] Graves, A., Wayne, G., and Danihelka, I. Neural Turing Machines. *arXiv:1410.5401*, Vol. 1410.5401, 2014.

[9] Zaremba, W. and Sutskever, I. Learning to Execute. In Bengio, Y. and LeCun, Y., editors, *Proc. the International Conference on Learning Representations (ICLR)*, 2015.

[10] Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to Sequence Learning with Neural Networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 3104–3112, 2014.

[11] Karpathy, A. and Fei-Fei, L. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *Proc. the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[12] Socher, R., Manning, C., and Ng, A. Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks. In *Proc. Advances in Neural Information Processing Systems*, 2010.

[13] Werbos, P. J. Backpropagation Through Time: What It Does and How to Do It. *Proceedings of the IEEE*, Vol. 78, No. 10, pp. 1550–1560, 1990.

[14] Hanselmann, T., Zaknich, A., and Attikiouzel, Y., authors, Mastorakis, N., editor. Connection between BPTT and RTRL. *Computational Intelligence and Applications*, Vol. 1, pp. 97–102, 1999.

[15] Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. Gradient Flow in Recurrent Nets the Difficulty of Learning Long-Term Dependencies. In Kremer, S. C. and Kolen, J. F., editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.

[16] Bengio, Y., Boulanger-Lewandowski, N., and Pascanu, R. Advances in Optimizing Recurrent Networks. *arXiv:1212.0901*, 2012.

[17] Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. *arXiv:1211.5063*, 2013.

[18] Graves, A. Generating Sequences With Recurrent Neural Networks. *arXiv:1308.0850*, 2013.

[19] Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[20] Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, Vol. 12, pp. 2451–2471, 2000.

[21] Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *arXiv:1502.03044v2*, 2015.

[22] Donahue, J., Hendricks, Anne, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. In *Proc. the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[23] Zaremba, W., Sutskever, I., and Vinyals, O. Recurrent Neural Network Regularization. In Bengio, Y. and LeCun, Y., editors, *Proc. the International Conference on Learning Representations (ICLR)*, 2015.

[24] Graves, A., Mohamed, A., and Hinton, G. Speech recognition with deep recurrent neural networks. In Ward, R. K., editor, *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6645–6649, 2013.

[25] Lipton, Z. C., Berkowitz, J., and Elkan, C. A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv:1506.00019*, 2015.

[26] Jozefowicz, R., Zaremba, W., and Sutskever, I. An Empirical Exploration of Recurrent Network Architectures. In Pineau, J., Bach, F., and Blei, D., editors, *Proc. the 32nd Annual International Conference on Machine Learning (ICML)*, 2015.

[27] Greff, K., Srivastava, R. K., Koutnìk, J., Steunebrink, B. R., and Schmidhuber, J. LSTM: A Search Space Odyssey. *arXiv*, 2015.

[28] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:1412.3555*, 2014.

[29] Jaeger, H. A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach. Technical report, Fraunhofer Institute for Autonomous Intelligent Systems, 2002.

[30] Schuster, M. and Paliwal, K. K. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, pp. 2673–2681, 1997.

執筆者：浅川伸一

7–1 リカレントネットワークによる文法学習

Learning Grammar by Recurrent Neural Network

はじめに

リカレントニューラルネットワークで文法を扱う研究は，エルマン [1][2][3] 以来の伝統がある．論文のタイトルである「小さく始めることが重要」とは，広義にカリキュラム学習（curriculum learning）[4] の先駆けと見なせる．2010 年，Mikolov ら [5] は音声認識にリカレントニューラルネットワークを適用し，さらに自然言語処理へ適用可能であることを示した．これをリカレントニューラルネットワーク言語モデル（recurrent neural network language model; RNNLM）という．リカレントニューラルネットワーク言語モデルにおいては単語や品詞の選択という離散的意思決定問題になるので，隠れマルコフモデル（hidden Markov model; HMM）の最短パス探索や，条件付き確率場（conditional random field; CRF）における最適解の探索に比肩しうる．単語配列の生起確率を最大化する条件付き確率場と観察単語系列の荷重和とを求め，損失を最小化する意思決定がなされる．この手続きをグラフ上の各ノードに対して繰り返し適用することで，連続値の場合の確率的勾配降下法（stochastic gradient descent; SGD）に相当する．クヌース [6] が拡張文脈自由文法（augmented context free grammar; ACFG）に関してコンパイラ設計で言及したコンピュータ言語に関するものでも，自然言語処理に関するものでも，リカレントニューラルネットワークを扱う手法が用いられている．加えて，リカレントニューラルネットワークを用いて数学の演算子を文法として扱う応用も提案されている [7]．ここでは，リカレントニューラルネットワークによる文法学習として，(1) 系列予測課題，(2) リカレントニューラルネットワーク，(3) 機械翻訳，(4) 構文解析を紹介する．

系列予測と系列生成

エルマンは系列予測学習課題を用いて単純再帰型ニューラルネットワークに逐次単語を入力し，その都度次の単語を予測させる訓練を行った [3]．系列予測課題とは，時刻 $t-1$ におけるシステムの内部状態を $s(t-1)$ とし，入力を $x(t)$ とすれば条件付き確率

$$P(x(t+1)|x(t), s(t-1)) \tag{1}$$

となり，入出力 $(x_t, y_t)_{t\in T}$ を音素とすれば音韻予測，単語とすれば単語予測課題となる．

一方，入力を BOW（bag-of-word）などに固定し，文脈層の遷移によって出力系列を生成する課題を系列生成課題と呼ぶ．リカレントニューラルネットワークは系列予測および生成の能力を持つ．

> 単語予測課題： 次の例文には主格となりうる名詞が複数存在する．
> *The girls who the teacher has picked for the play which will be produced next month practice every afternoon.*

主語 “girls” と動詞 “practice” との間に中央埋め込み文（center embedded sentence）が存在する．動詞との間で数の一致が必要であるため，リカレントニューラルネットワークはこの例文の主語 “girls” が複数形であることを保持し，三人称単数現在の s を付けないことを学習する．このとき，単純再帰型ニューラルネットワークは単文をまず学習し，続いて関係代名詞節を含む複雑な文章を学習することで増分学習（incremental learning）が用いられる．すなわち，リカレントニューラルネットワークにおける文法学習とは，与えられた文章の構文解析木を返すという意味での文法を表すのではなく，時制，数，性の一致などの系列予測課題によって統語規則を正しく学習することを指していた．系列予測課題は，次の単語を予測することから，明示的に教師信号を用意しなくてもよいので，チョムスキーの指摘した刺激の貧困（poverty of stimulus），拡大して考えればプラトン問題（Plato’s problem）へのニューラルネットワークによる一つの解答だとも見なせる．

ニューラルネットワークを単語予測課題による言語モデルとして考えた場合，フィードフォワード型の Bengio [8] とリカレントニューラルネットワーク型の Mikolov ら [5] との両モデルを従来の比較参照モデルとして扱うことが多い．

リカレントニューラルネットワーク言語モデル

Mikolov ら [5] は，音声認識にエルマンネットワーク（単純再帰型ニューラルネットワーク）[1] を用い，

リカレントニューラルネットワークの自然言語処理への応用への道を開いた．x, h, y をそれぞれ中間，文脈，出力層の出力とする．時刻 t における入力語彙を $v(t)$ とすれば，

$$x(t) = f(v(t) + h(t)) \tag{2}$$

$$h_i(t) = f\left(\sum_{j\in V,H} w_{ij} x_j(t-1)\right) \tag{3}$$

$$y_k(t) = g\left(\sum_{j\in H} w_{kj} x_j(t)\right) \tag{4}$$

ここで，f はロジスティック関数

$$f(x) = \frac{1}{1+\exp(-x)} \tag{5}$$

であり，g はソフトマックス関数（softmax function）

$$g(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{6}$$

である．また，w_{qp} は p から q への結合係数である．

Mikolov らは，ニューヨークタイムズ紙の 370 万語中 64 万語（30 万文）を用いてニューラルネットワークを訓練した．学習時間は要するが，ニーザー–ネイ（Kneser-Ney）の平滑化などの従来手法より訓練語彙数が少ないにもかかわらず，単語誤認識率（word error ratio）で比較した場合の性能が向上したことが報告された．Mikolov らは低頻度語を以下のように扱った．

$$p(v_i(t+1)|w(t), s(t-1)) = \begin{cases} \frac{y_{\text{rare}}(t)}{C_{\text{rare}}}, & w_i(t+1) \text{ が低頻度語の場合} \\ y_i(t), & \text{それ以外} \end{cases} \tag{7}$$

ここで，C_{rare} は閾値以下しか発生しない単語数である．

以降，リカレントニューラルネットワークに LSTM（long short-term memory）やゲート付き再帰ユニット（gated recurrent unit）を用いた研究が増加した．

ニューラルネットワーク機械翻訳

機械翻訳は Weaver [9] や Brown [10] 以来の歴史がある．ニューラルネットワーク機械翻訳（neural network machine translation; NMT）と直接対比される概念に，SMT（statistical machine translation; 統計的機械翻訳）がある [10]．SMT で整備されてきた枠組みをリカレントニューラルネットワークに適用して，機械翻訳を実装する試みがなされている．画像処理と自然言語処理とを同一の枠組みで記述する試み [11][12][13] もある．SMT では，入力文（ソース言語）x に対応する翻訳文（ターゲット言語）y を条件付き確率 $p(y|x)$ と見なし，y の条件付き確率をベイズ則から次式のように表記できる．

$$p(y|x) \propto p(x|y)\, p(y) \tag{8}$$

上式 (8) を対数変換して

$$\log(y|x) = \log p(x|y) + \log p(y) + C \tag{9}$$

とおけば，右辺第 1 項は翻訳モデル，第 2 項は言語モデルとなる [14][15]．式 (9) は BLEU [16] などを目標関数として学習される．

ニューラルネットワークを用いた機械翻訳（NMT）[17] では，リカレントニューラルネットワークを用いてソース言語の文を目標言語の文の確率分布へと変換する．ソース言語と文脈層とを用いた

$$h_t = \sigma\left(W^{hx} x_t + W^{hh} h_{t-1}\right) \tag{10}$$

$$y_t = W^{yh} h_t \tag{11}$$

などが一般形であろう [18]．機械翻訳では，ソース言語とターゲット言語における単語が一対一に対応するとは仮定できないため，系列長も対応がとれるとは限らず，T と T' とを可変として $p(y_1, \ldots, y_{T'}|x_1, \ldots, x_T)$ を考える．LSTM を用いた言語モデル（LSTM-LM）（図 1）で考えれば，

$$p(y_1, \ldots, y_{T'}|x_1, \ldots, x_T) = \prod_{t\in T'} p(y_t|v, y_1, \ldots, y_{t-1}) \tag{12}$$

となる．上式 (12) 右辺 $p(y_t|v, y_1, \ldots, y_{t-1})$ はソフトマックスを用いる．文頭を <SOS>（start of sentence），文末を <EOS>（end of sentence）という単語として扱う場合が多い．

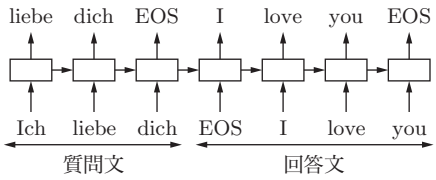


図 1　LSTM 言語モデル

この時点で最高性能のモデルの一つである Sutskever らの文献 [18] では，ソース言語とターゲット言語，すなわち入出力系列を異なる LSTM によってモデル化している．加えて，ソース言語の語順を逆転させる（ターゲット言語は逆順にしない）と成績が向上したと報告している．なぜソース言語を逆順にして入力する

とよいかの理由については言及されていない．また，ニューラルネットワーク機械翻訳（NMT）では，双方向リカレントニューラルネットワーク（bidirectional recurrent neural network; BRNN）[19] を用いる場合がある（図 2）．系列（$\tau = 0, \ldots, T$）における隠れ層の状態 h_τ を順方向 $\overrightarrow{h}_\tau$ と逆方向 $\overleftarrow{h}_\tau$ の二つの方向に情報が流れる二つの独立した中間層を用意する．両中間層を合併した $h_\tau = \left[\overrightarrow{h}_\tau, \overleftarrow{h}_\tau\right]$ を用いて，時刻 t における中間層の出力 a_t は，以下のように表記される．

$$a_t = f\left(s_t, h_t, y_{t-1}\right) \tag{13}$$

ここで，s_t は入力，y_{t-1} は 1 時刻前の出力である．

$$p\left(y_i \,|y_i, \ldots, y_{i-1}, x, \theta\right) = g\left(y_{i-1}, s_i, c_i, \theta\right) \tag{14}$$

を考える．リカレントニューラルネットワークの内部状態 s は

$$s_i = f\left(s_{i-1}, y_{i-1}, c_i\right) \tag{15}$$

である．文脈ベクトル c_i は

$$c_i = \sum_{j \in T_z} \alpha_{ij} h_j \tag{16}$$

であり，α_{ij} と BRNN の順行および逆行中間層の状態である h_j との積の総和である．各 BRNN ユニットからの出力 $\alpha_{i \in T}$ は，ソフトマックスである．

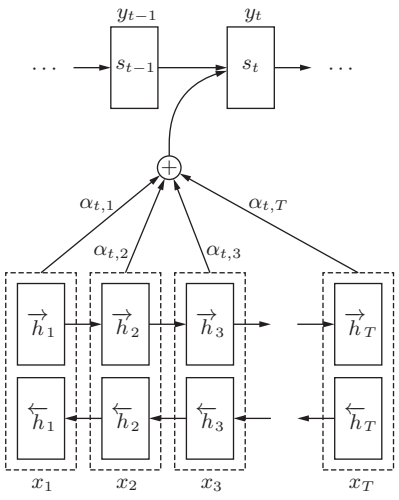


図 2 BRNN による NMT の概念図（文献 [20]，Figure 1 を改変）

ニューラルチューリングマシン（neural Turing machine）[21] の成果に基づいたニューラル翻訳機械 [22] と呼ばれるアプローチもある．これは，通常の深層リカレントニューラルネットワークが LSTM ユニットを多層に重ねるのに対して，入力層（入力メモリ）→ ヘッドから読み出し → コントローラ → ヘッドから書き出し → 次の入力層，というチューリングマシンの 1 ステップを多層化したモデルである．

Graves ら [21] は，LSTM，ニューラルチューリングマシン＋ LSTM，ニューラルチューリングマシン＋フィードフォワード制御のモデルを比較し，ニューラルチューリングマシンに LSTM の制御をつけたモデルの学習が速いことを報告している [23].

構文解析

リカレントニューラルネットワークによって構文解析木を出力するモデルがある [23]．日本語では “recurrent” と “recursive” をともに “再帰” と訳す．ニューラルネットワークの文脈では，“recursive” は同じ演算子やパラメータセットをスケールの異なる種々の状態，成分，因子，構造の計算に繰り返し適用する操作モデルを指す．一方，“recurrent” はフィードバック結合を有するニューラルネットワークモデルを指す場合に用いる．ここでは，recursive なモデルを RecNN，recurrent なモデルをリカレントニューラルネットワークと表記する．Socher ら [11] は，RecNN による構文解析を提案した．RecNN は，語や句，節など文の要素を入力として受け取り，それらの関係を出力とする．RecNN は有向非環グラフ（directed acyclic graph; DAG）であるから，構文解析木との相性は高い．図 3 は RecNN の例である．RecNN は同一の係数行列をすべての端点ノードに適用する．葉ノードは単語を表す n 次元ベクトル表現である．

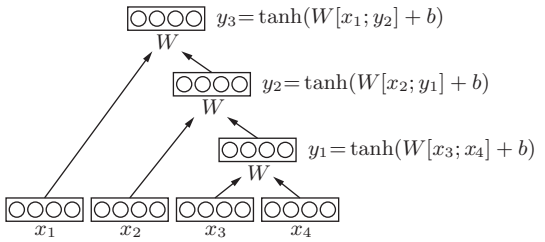


図 3 リカレントニューラルネットワーク

$$p\left(B \,|A\right) = \prod_{t \in T_B} p\left(y_{B_t} \,|A_1, \ldots, A_{T_A}, B_1, \ldots, B_{t-1}\right) \tag{17}$$

$$\equiv \prod_{t \in T_B} z\left(W_0 h_{T_{A+t}}\right)' \delta_{B_t} \tag{18}$$

ここで，$x = (A_1, \ldots, A_{T_A}, B_1, \ldots, B_{T_B})$ であり，h_t は LSTM の h 空間上で t 番目の要素である．z はソフトマックスを表す．W_0 は各記号の表象行列であり，δ はクロネッカーのデルタである．図 4 に RecNN による解析木の例を示す．

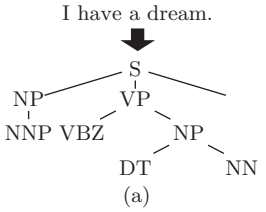


図 4　RecNN による構文解析の例．(a) 構文解析木，(b) 対応する文字列による出力 (文献 [23], Figure 2 を改変)

注意の導入

画像処理や自然言語処理において注意を導入する場合がある．Vinyals ら [23] も，Bahdanau ら [20] と同じく注意機構を導入した．注意は次式のように表記できる．

$$u_i^t = v^T \phi\left(W_1^{'} h_i + W_2^{'} d_t\right) \tag{19}$$

$$a_i^t = z\left(u_i^t\right) \tag{20}$$

$$d_t^{'} = \sum_{i \in T_A} a_i^t h_i \tag{21}$$

$v, W_1^{'}, W_2^{'}$ は学習可能なパラメータである．u_i^t は長さ T のベクトルであり，これをソフトマックス z にかけて中間層の出力となる．ソフトマックスをかけることを注意と呼んでいる．

参考文献

[1] Elman, J. L. Finding structure in time. *Cognitive Science*, Vol. 14, No. 2, pp. 179–211, 1990.

[2] Elman, J. L. Incremental learing, or The importance of starting small. Technical report, University of California, San Diego, 1991.

[3] Elman, J. L. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, Vol. 7, pp. 195–225, 1991.

[4] Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum Learning. In *Proc. the 26th Annual International Conference on Machine Learning (ICML '09)*, pp. 41–48, 2009.

[5] Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. Recurrent Neural Network Based Language Model. In *Proc. INTERSPEECH2010*, pp. 1045–1048, 2010.

[6] Knuth, D. E. Semantics of Context-Free Languages. *Mathematical Systems Theory*, Vol. 2, No. 2, pp. 127–145, 1968.

[7] Zaremba, W., Kurach, K., and Fergus, R. Learning to Discover Efficient Mathematical Identities. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, pp. 1278–1286. 2014.

[8] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, Vol. 3, pp. 1137–1155, 2003.

[9] Weaver, W. Translation. In *Machine Translation of Languages*, pp. 15–23. MIT Press, Cambridge, Massachusetts, USA, 1949.

[10] Brown, P. E., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, Vol. 19, No. 2, pp. 263–311, 1993.

[11] Socher, R., Manning, C., and Ng, A. Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks. In *Proc. Advances in Neural Information Processing Systems*, 2010.

[12] Socher, R., Lin, C. C., Ng, A. Y., and Manning, C. D. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In Getoor, L. and Scheffer, T., editors, *Proc. the 28th Annual International Conference on Machine Learning (ICML)*, 2011.

[13] Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. Parsing With Compositional Vector Grammars. In *Proc. the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013.

[14] Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In *Proc. the Empiricial Methods in Natural Language Processing (EMNLP)*, 2014.

[15] Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proc. the Tenth Machine Translation Summit*, pp. 79–86, 2005.

[16] Papineni, K., Roukos, S., Ward, T., and Zhu, W. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proc. the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 311–318, 2002.

[17] Kalchbrenner, N. and Blunsom, P. Recurrent Continuous Translation Models. In *Proc. the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[18] Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to Sequence Learning with Neural Networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 3104–3112, 2014.

[19] Schuster, M. and Paliwal, K. K. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, pp. 2673–2681, 1997.

[20] Bahdanau, D., Cho, K., and Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proc. the International Conference on Learning Representations (ICLR)*, 2015.

[21] Graves, A., Wayne, G., and Danihelka, I. Neural Turing Machines. *arXiv:1410.5401*, Vol. 1410.5401, 2014.

[22] Meng, F., Lu, Z., Tu, Z., Li, H., and Liu, Q. Neural Transformation Machine: A New Architecture for Sequence-to-Sequence Learning. *arXiv:1506.06442*, 2015.

[23] Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. Grammar as a Foreign Language. In *Proc. the International Conference on Learning Representations (ICLR)*, 2015.

執筆者：浅川伸一