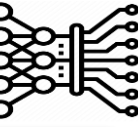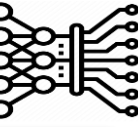# Deep Neural Networks

**(Recurrent Neural Network, Long Short-term Memory)**

Time-series Data

❖ Applications

- Prediction or forecasting

  - Stock price forecast for tomorrow

  - Weather forecast for tomorrow

  - Prediction of machine failure

  - Wind speed and wind direction prediction

  - Agricultural price/demand forecast, etc.

- Apply to language translation

- Apply to speech recognition

- Apply to neural engineering (e.g., digital healthcare)

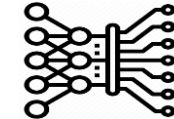- Apply to generative models (e.g., view pictures and generate descriptive sentences)

❖ Examples

- Number and electrocardiogram

x=100766555541707700

(a) 온라인 숫자

(b) 심전도 신호(3채널)

Representation of Time-series Data
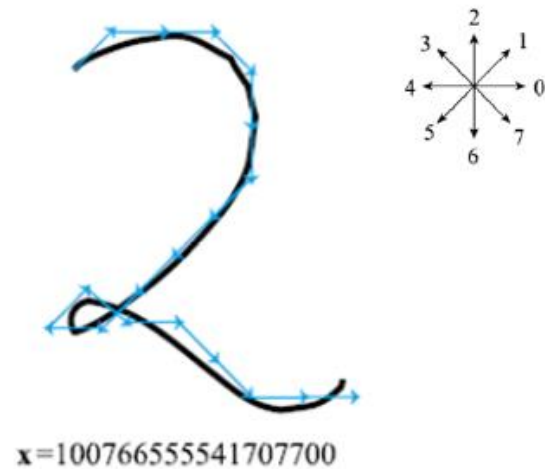
❖ Characteristics

- The order of the elements is important
  - e. g., "세상에는 시계열 데이터가 참 많다 " 를 "시계열 참 데이터가 많다 세상에는 " 으로 바꾸면 의미 훼손
- Sample length is different
  - e. g., 짧은 발음 "인공지능 " 과 긴 발음 "인~공~~지~능 "
- Context dependence
  - e. g., "시계열은 앞에서 말한 바와 … 특성이 있다 " 에서 "시계열은 " 과 "특성이 있다 " 는 밀접한 관련성
- Seasonality
  - e. g., 상추 판매량, 미세먼지 수치, 항공권 판매량 등

Representation of Time-series Data
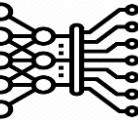
❖ Notation

  ▪ Vector of vector

$$\mathbf{x} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(T)}\right)^{\mathrm{T}}$$
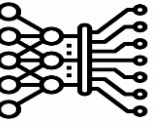
  ▪ E.g., , electroencephalogram signals:

  (If the data were recorded for 100 times per second and were measure it for 2 minutes, the length is T=12,000)

$$\mathbf{x} = \left(\begin{pmatrix} 0.3 \\ 0.1 \\ 0.2 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.6 \\ 0.4 \end{pmatrix}, \cdots \cdots \right)^{\mathrm{T}}$$

❖ Training set

$$\mathbf{x} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(T)}\right)^{\mathrm{T}}, \quad \mathbf{y} = \left(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(L)}\right)^{\mathrm{T}}$$
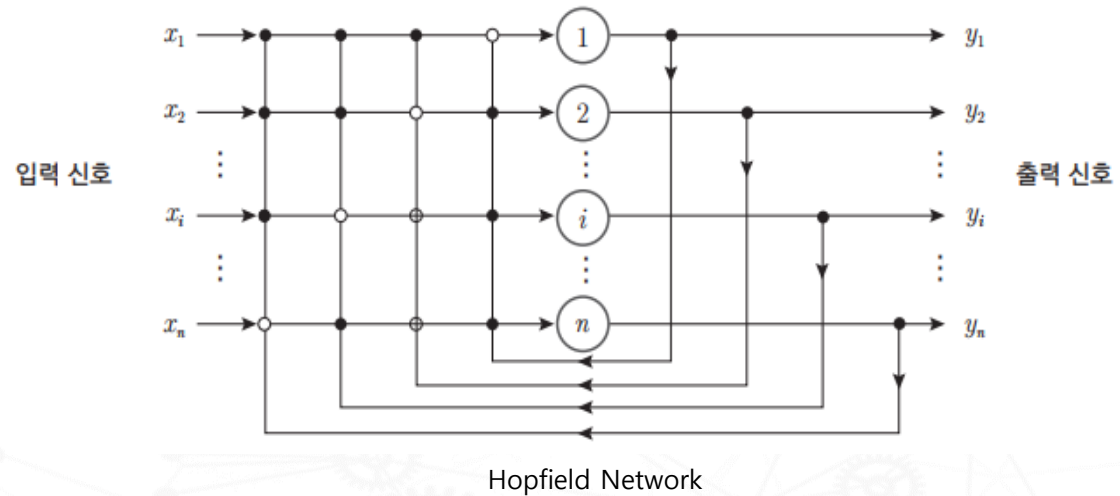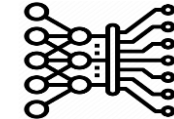
❖ Neural network model with memory

- Associated with Hopfield network

  - John Hopfield proposed a Hopfield network in 1982 with memory storage and associated memory

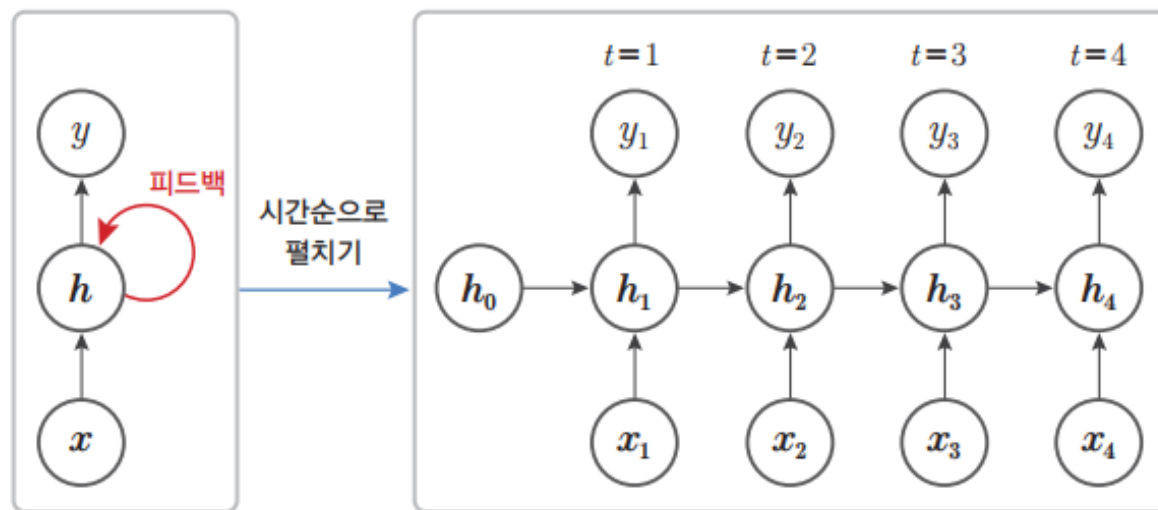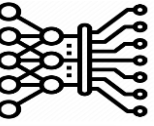 - Hopfield network with feedback structure

John Hopfield

입력 신호

출력 신호

Hopfield Network

Recurrent Neural Network

❖ Essential features of a recurrent neural network (RNN)

- 시간성: 특징을 순서대로 한 번에 하나씩 입력해야 한다.

- 가변 길이: 길이가 T인 샘플을 처리하려면 은닉층이 T번 나타나야 한다. T는 가변적이다.
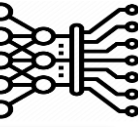
- 문맥 의존성: 이전 특징 내용을 기억하고 있다가 적절한 순간에 활용해야 한다.

❖ Structures

■ Structure of MLP

- Input, hidden, output layers

■ The hidden layer has a recurrent edge

- Handle all timeliness, variable length, and context dependencies

- Recurrent edges are responsible for transmitting information that occurs at the t-1 moment to the t
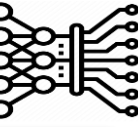
    moment

$$y_t = W_{hy} h_t$$

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

은닉 상태   이전 은닉 상태   입력 데이터
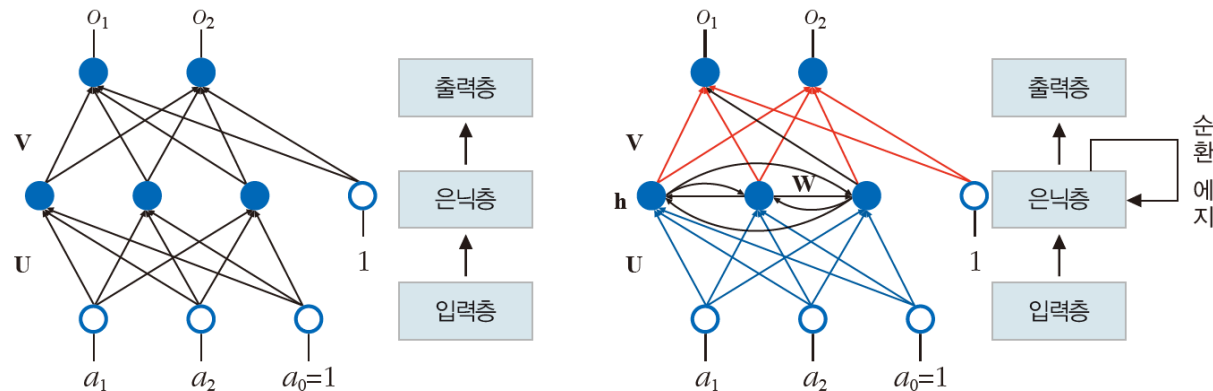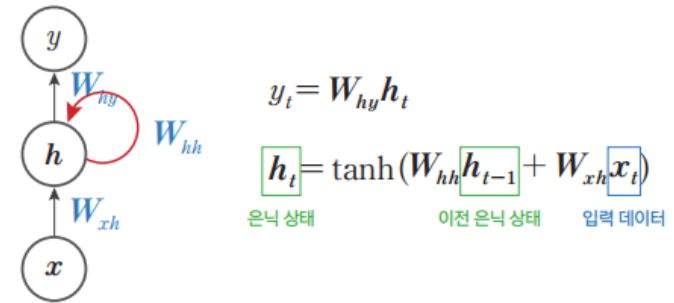
❖ Equation is,

$$\mathbf{h}^{(t)} = f\left(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \Theta\right)$$

- Calculate at t = 1 moment, calculate at t = 2 moment with the result, calculate at t = 3 moment with the result, ..., t = repeat until n moment

- At the t moment, the hidden layer value (state) $\mathbf{h}^{(t-1)}$ at the t-1 moment and the input $\mathbf{x}^{(t)}$ at the t moment are received and switched to $\mathbf{h}^{(t)}$

- $\Theta \stackrel{\llcorner}{=}$ a parameter of a recurrent neural network

- The learning algorithm of is called backpropagation through time (BRTT)

Recurrent Neural Network
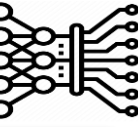
❖ Overall flow,
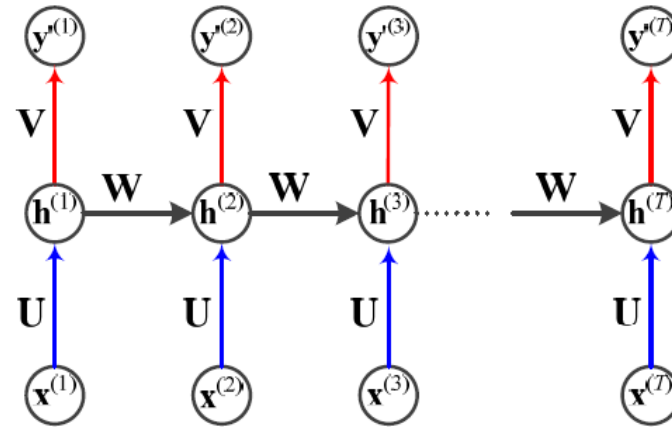
To find out the optimal {**U**, **V**, **W**} in training
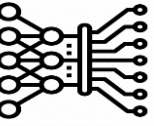
$$\mathbf{h}^{(T)} = f(\mathbf{h}^{(T-1)}, \mathbf{x}^{(T)}; \Theta)$$
$$= f(f(\mathbf{h}^{(T-2)}, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta)$$
$$\vdots$$
$$= f(f(\cdots f(\mathbf{h}^{(1)}, \mathbf{x}^{(2)}; \Theta), \cdots, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta)$$
$$= f(f(\cdots f(f(\mathbf{h}^{(0)}, \mathbf{x}^{(1)}; \Theta), \mathbf{x}^{(2)}; \Theta), \cdots, \mathbf{x}^{(T-1)}; \Theta), \mathbf{x}^{(T)}; \Theta)$$

❖ Output representation

- If x_1 is given, the conditional probability $p(y_1|x_1)$ of y_1

- Express the conditional probability $p(y_2|x_1, x_2)$ of y_2 given the inputs x_1 and x_2 because x_1 was passed to h_2 through h_1

- Therefore, the output of step t expresses the conditional $p(y_2|x_1, x_2, \ldots, x_t)$ of y_t given x_1, x_2 , …, x_t
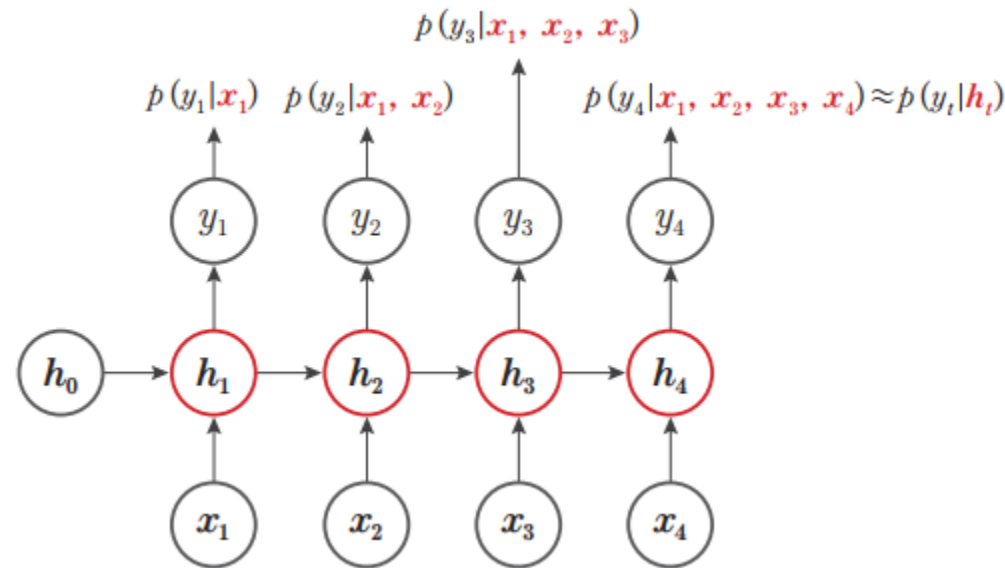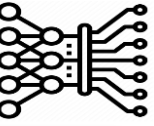
❖ Weight sharing effects of RNN

- Allows capturing sequential structures

- Easy to process variable length data

- Number of parameters saved and normalized

Recurrent Neural Network
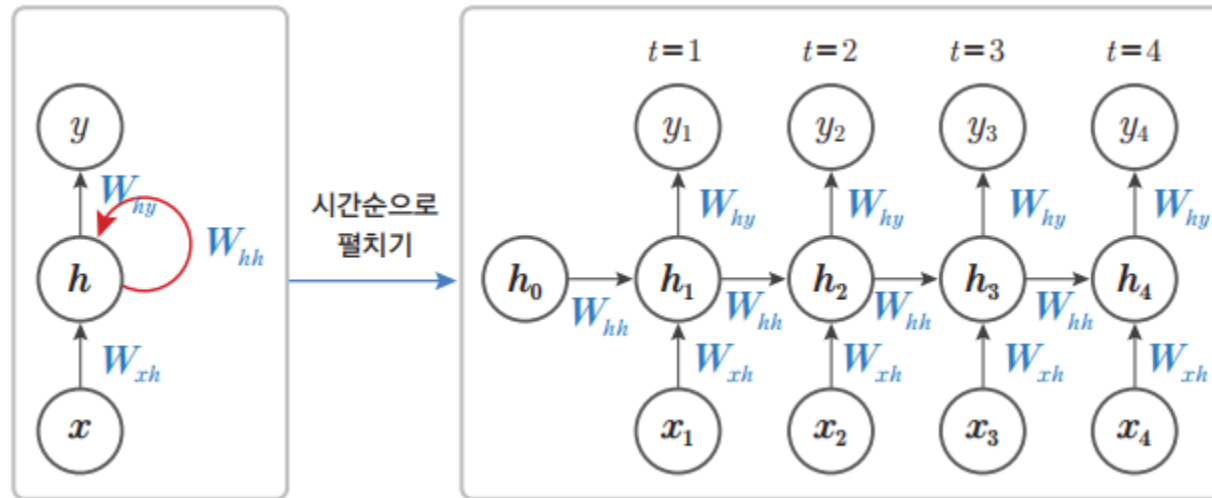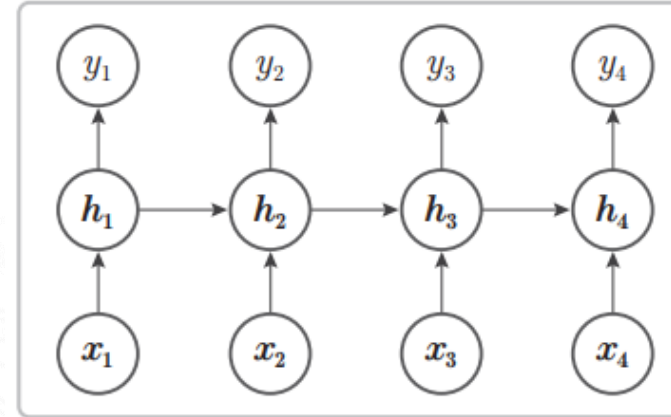
❖ A variety of architecture

- Many-to-One model

  - he input is a sequential column but the output is not a sequential column; all steps receive the input but only the output is in the last step

- Many-to-Many model

  - Used when input and output are sequential columns of the same length, receives input at all stages and outputs at all stages
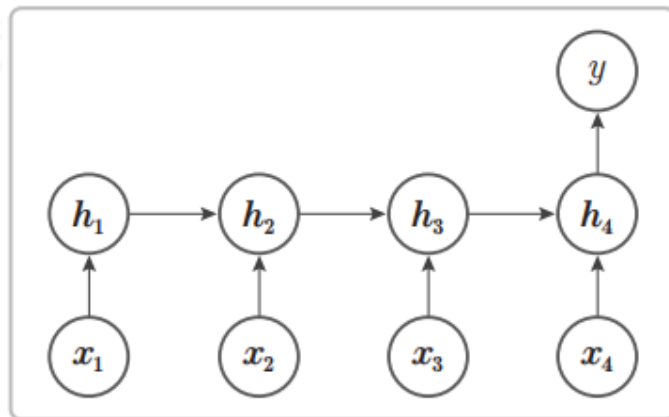
Many-to-One model   Many-to-Many model

❖ A variety of architecture

- Bidirectional model

  - Both ways at the input

  - Data generated in chronological order depends only on the past in the present, depending on the causal relationship, viewed only in the direction of the passage of time

  - Relative order is important for data with spatial order relationships, so it is more accurate to examine and judge in both directions
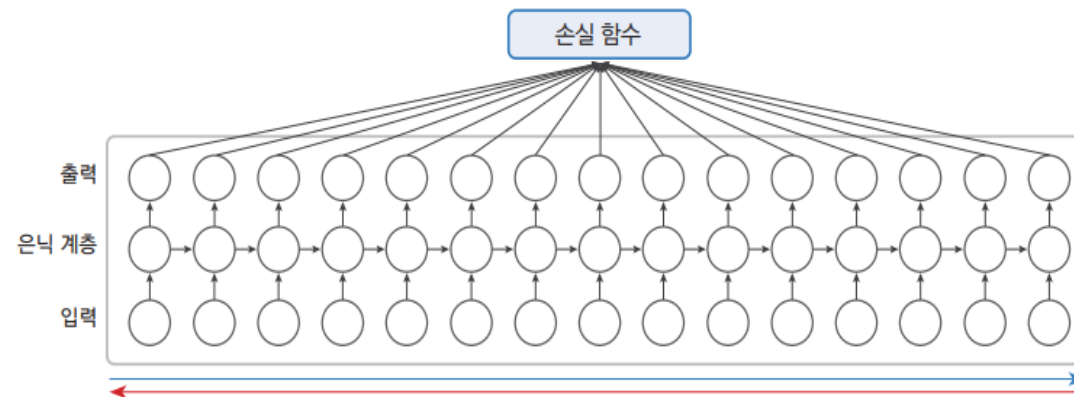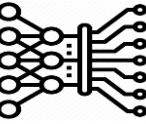

양방향 모델

Recurrent Neural Network
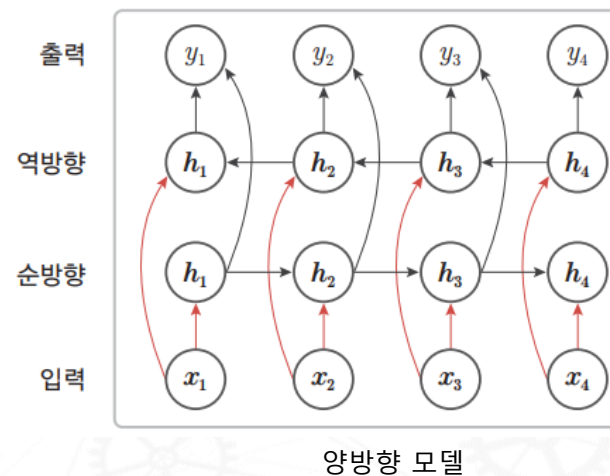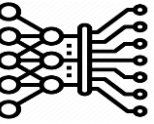
❖ Limitation of Backpropagation

- Backpropagation starts in the loss function and proceeds to all stages simultaneously in order

- Reverse propagation after running the last step, so it cannot be applied to an endless or very long sequential column
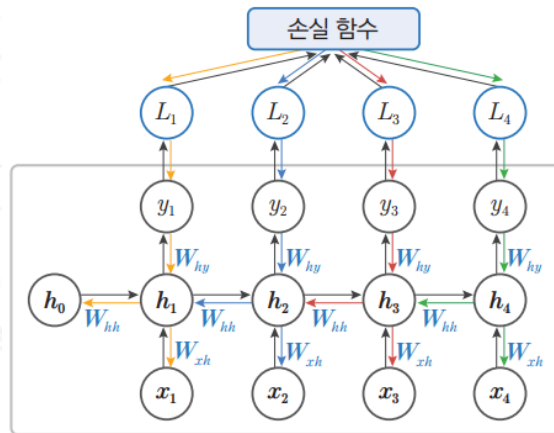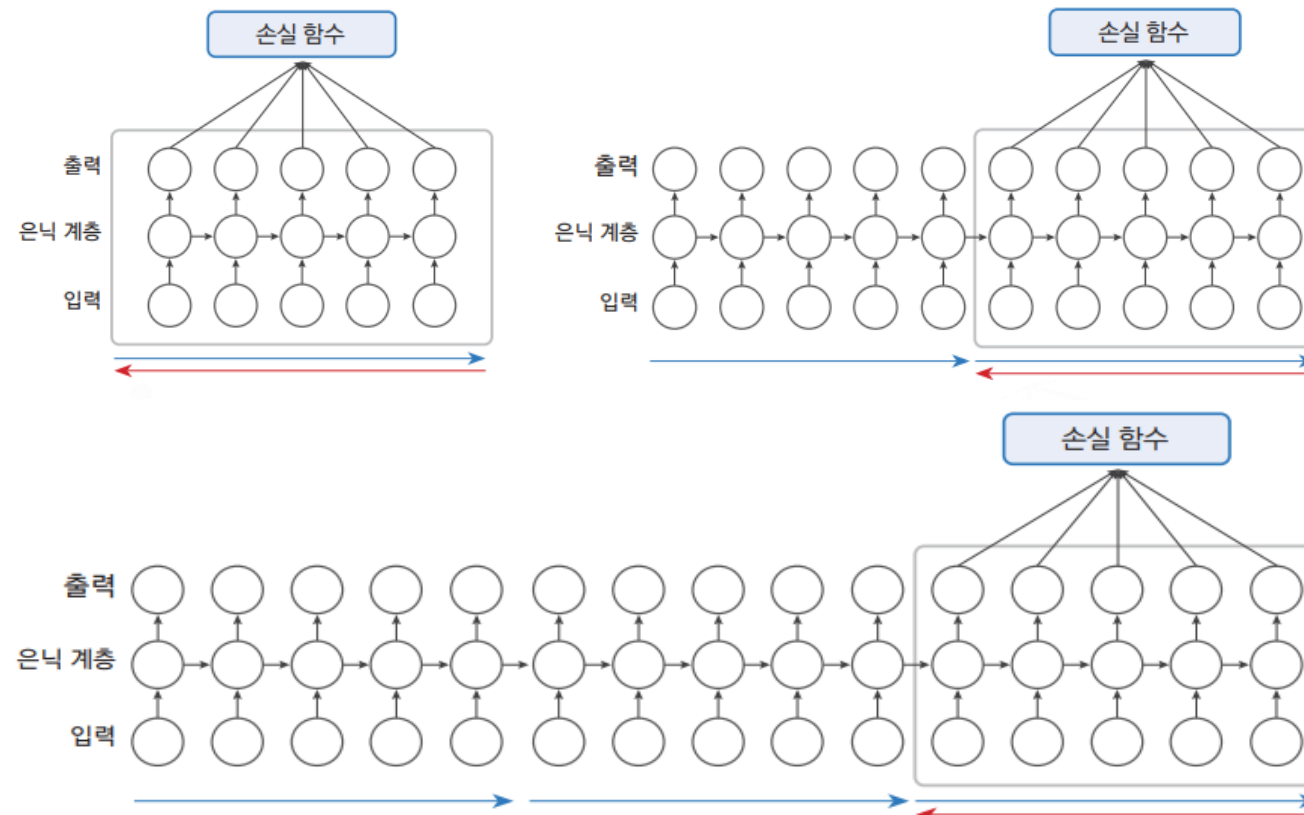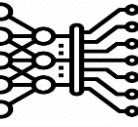

Principle of BRTT

Limitation of BRTT

Recurrent Neural Network

❖ Answer of backpropagation problem

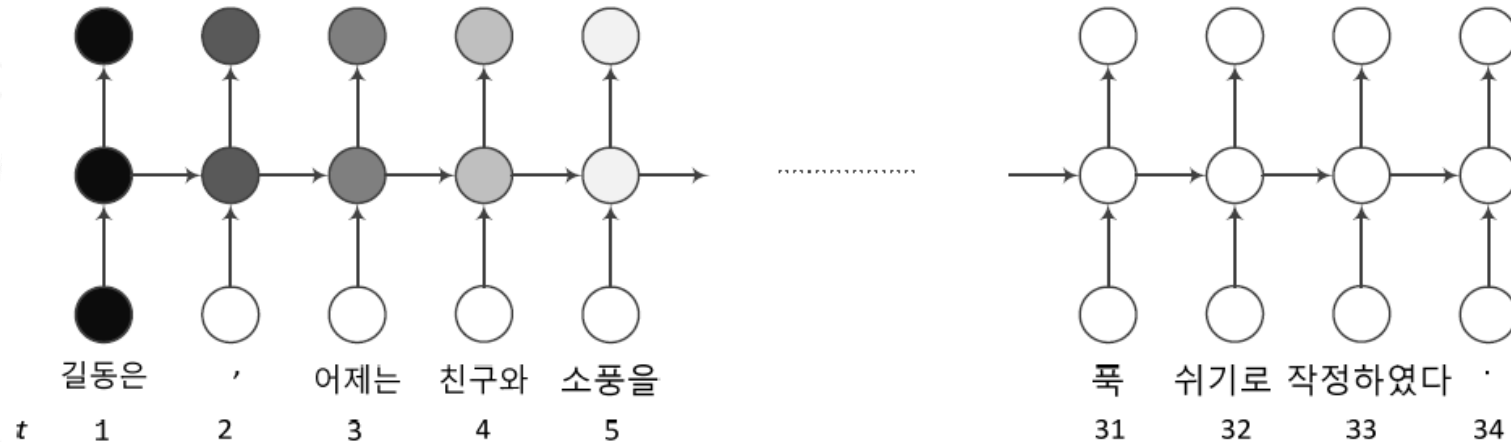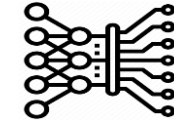- To perform forward training and reverse training by grouping certain steps

❖ Problem

- A phenomenon in which the influence of input is increasingly lost despite the dependence on distant input when the context is wide

  - e.g., 아래 문장에서 t=1의 '길동은'과 t=32의 '쉬기로'는 아주 밀접한 관련


Long term dependency

Long Short-Term Memory^(LSTM)

❖ Modeling of long-term and short-term memory

- Remember the past with the ability to pass the hidden layer state to the next moment

- However, the limitations of not properly dealing with long-term context dependencies

    (a phenomenon where distant elements interact closely)

- Memory loss due to the influence of continuous incoming input

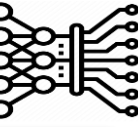- People retain their memories of a long time ago with selective memory

Structure of LSTM
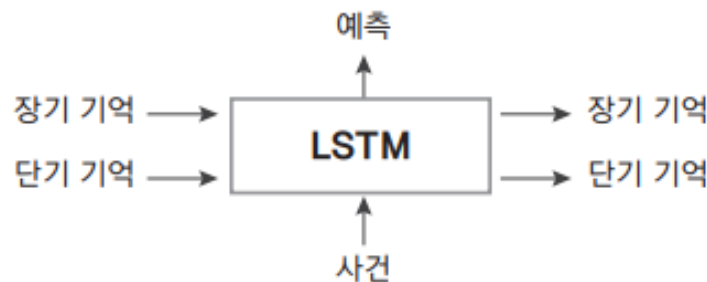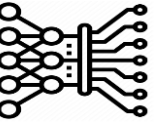
❖ Gate structure for selecting memory

- Four types of gates in the LSTM that are involved in shaping memory

- Gate is the role of choosing whether to keep or forget memories

  - Remember to keep up with the current context and frequently used memories, but forget those that don't

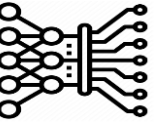- Forget gate, input gate, memory gate, output gate in LSTM cell

LSTM의 장기기억과 단기기억

(a) 선별 기억 능력이 없는 순환 신경망

(b) 선별 기억 능력을 지닌 LSTM
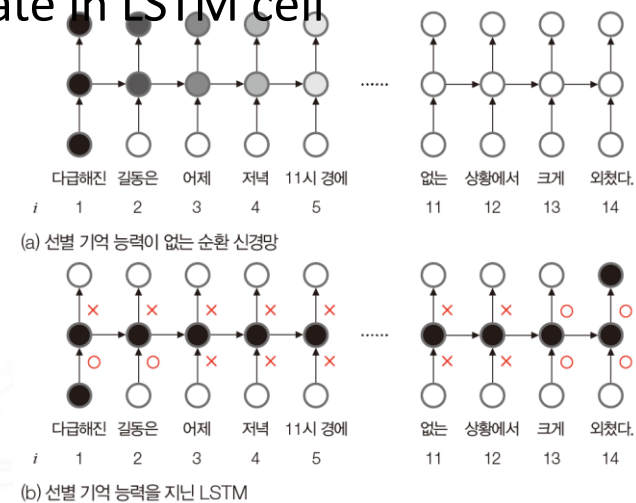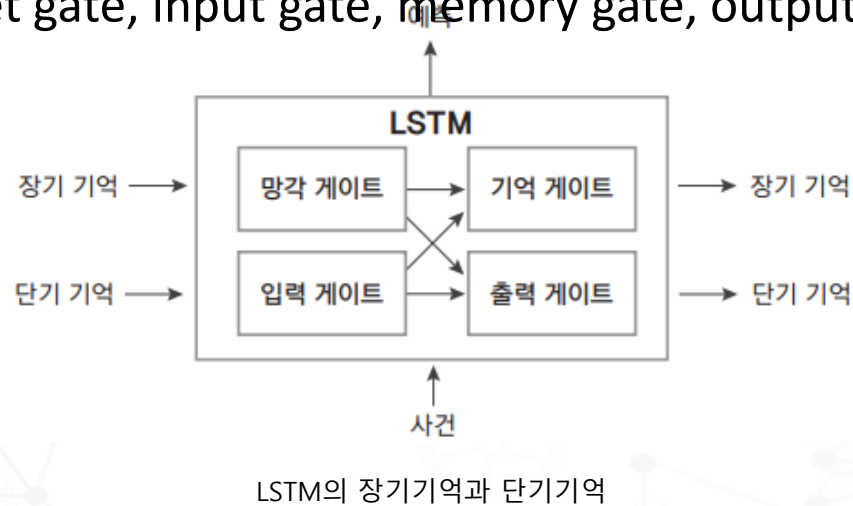
❖ Flow of memory and the input, output, and forget gates

- LSTM maintains or eliminates newly formed memory through gate
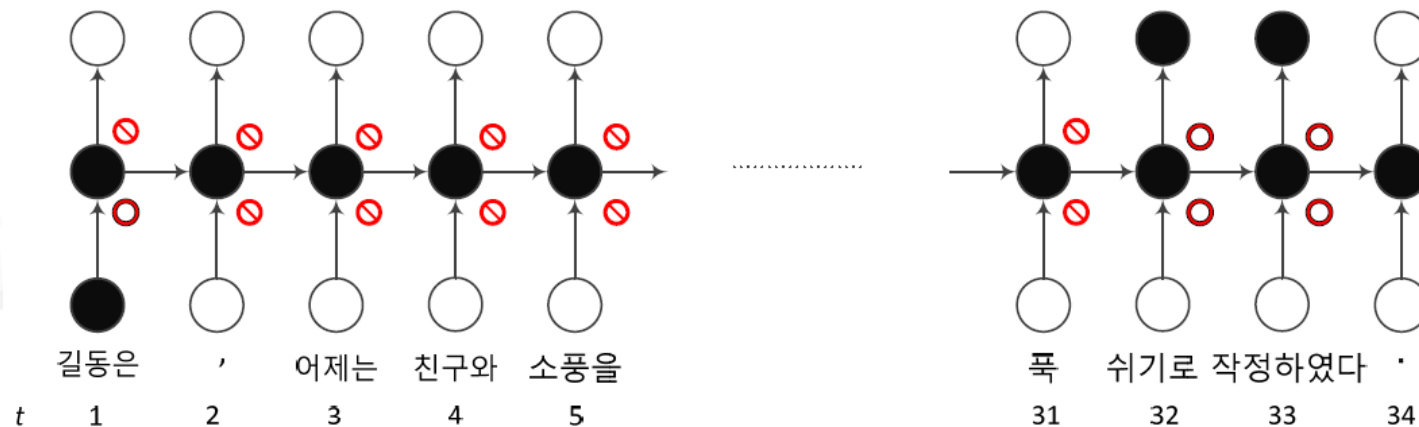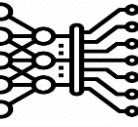

장기 기억의 전달

Structure of LSTM
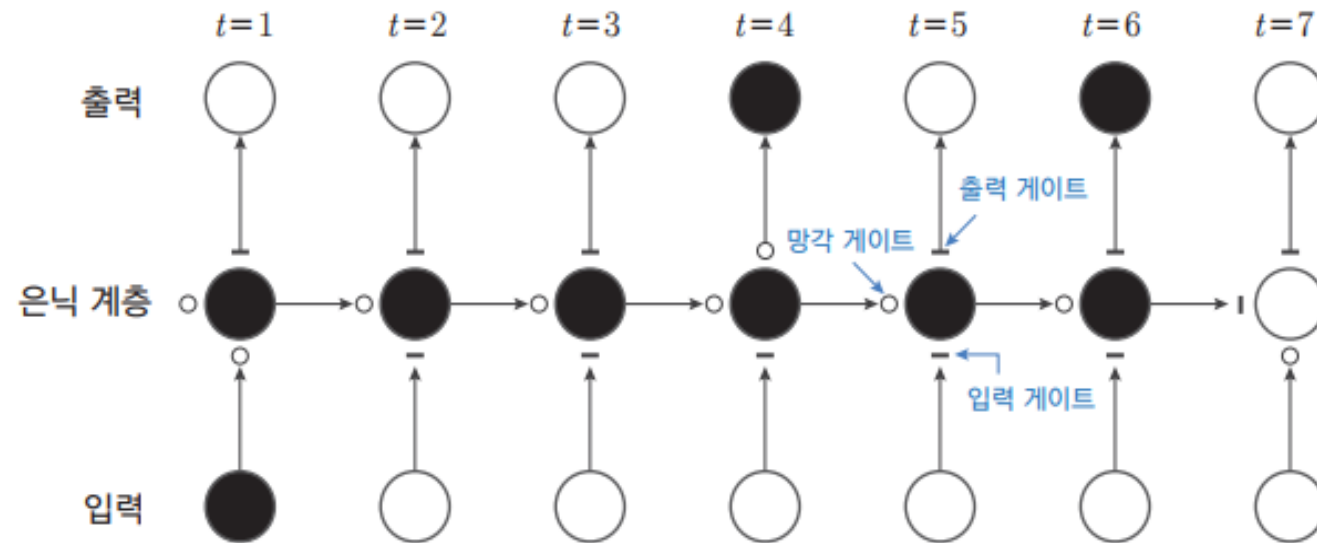
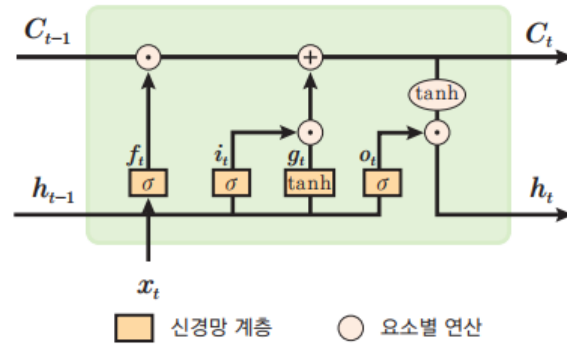❖ Input gate and output gate

- A signal flows when the gate is opened (⭕) and is blocked when the gate is closed (⊘)

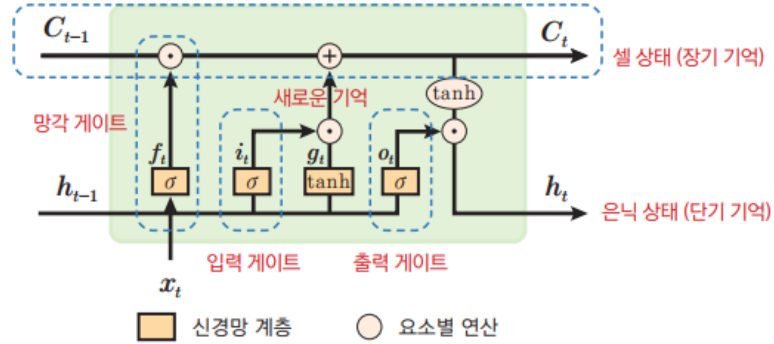 - Only input was opened at t=1, and both input and output were opened at t =32 and t =33
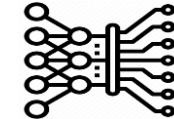

입력 게이트와 출력 게이트를 이용한 입출력 제어

❖ LSTM Cell

LSTM Cell

LSTM 게이트 위치

❖ Gate values that depend on the gate location and context in the LSTM cell

- The complexity of the LSTM cell structure separates each gate from the memory area

Training of LSTM

❖ Calculate i_t, f_t, o_t, g_t

$$
\begin{array}{l}
\text{입력 게이트} \\
\text{망각 게이트} \\
\text{출력 게이트} \\
\text{새로운 기억}
\end{array}
\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix}
=
\begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix}
W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}, \quad
W = \begin{bmatrix} W_{hi} & W_{xi} \\ W_{hf} & W_{xf} \\ W_{ho} & W_{xo} \\ W_{hg} & W_{xg} \end{bmatrix}
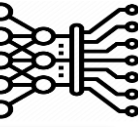$$

이전 단기 기억 → h_{t-1} 새로운 사건 → x_t

❖ Calculate C_t with respect to long term memory

이전 장기 기억   새로운 기억
망각 게이트   입력 게이트

$$\text{장기 기억} \quad C_t = f_t \odot C_{t-1} + i_t \odot g_t$$

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_{xf}^T \cdot \mathbf{x}_t + \mathbf{W}_{hf}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_f\right)$$
$$\mathbf{i}_t = \sigma\left(\mathbf{W}_{xi}^T \cdot \mathbf{x}_t + \mathbf{W}_{hi}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_i\right)$$
$$\mathbf{o}_t = \sigma\left(\mathbf{W}_{xo}^T \cdot \mathbf{x}_t + \mathbf{W}_{ho}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_o\right)$$
$$\mathbf{g}_t = \tanh\left(\mathbf{W}_{xg}^T \cdot \mathbf{x}_t + \mathbf{W}_{hg}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_g\right)$$
$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \mathbf{g}_t$$

❖ Calculate $i_t, \ f_t, \ o_t, \ g_t$

$$
\begin{array}{l}
\text{입력 게이트} \\
\text{망각 게이트} \\
\text{출력 게이트} \\
\text{새로운 기억}
\end{array}
\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix}
=
\begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix}
W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}, \quad
W = \begin{bmatrix} W_{hi} & W_{xi} \\ W_{hf} & W_{xf} \\ W_{ho} & W_{xo} \\ W_{hg} & W_{xg} \end{bmatrix}
$$

이전 단기 기억

새로운 사건

❖ Calculate h_t with respect to short term memory

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

출력 게이트   장기 기억

LSTM Model
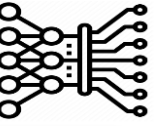
❖ A variety structure

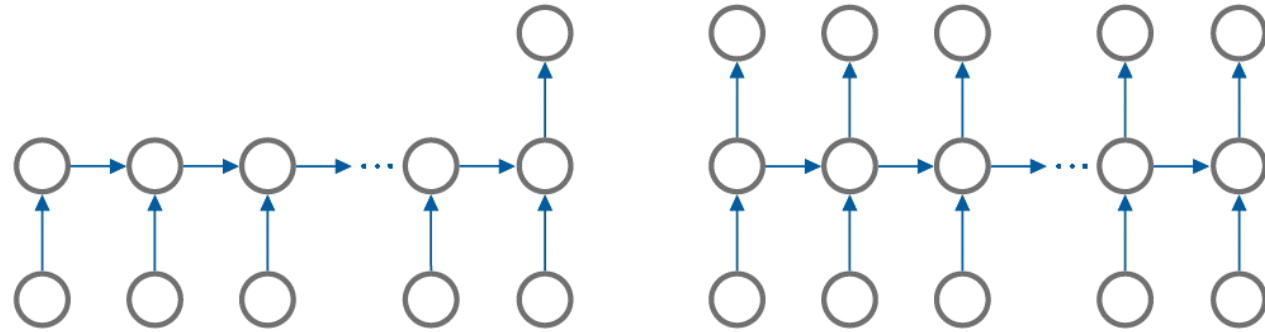- Use bi-directional LSTM if context needs to be examined in both directions
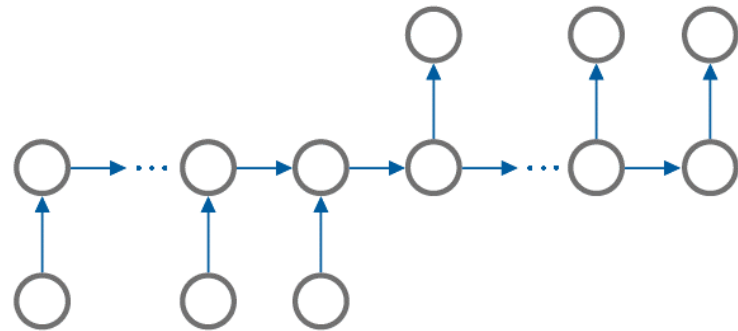

(a) 적층 LSTM

(b) 양방향 LSTM

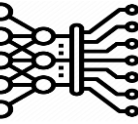# LSTM Model
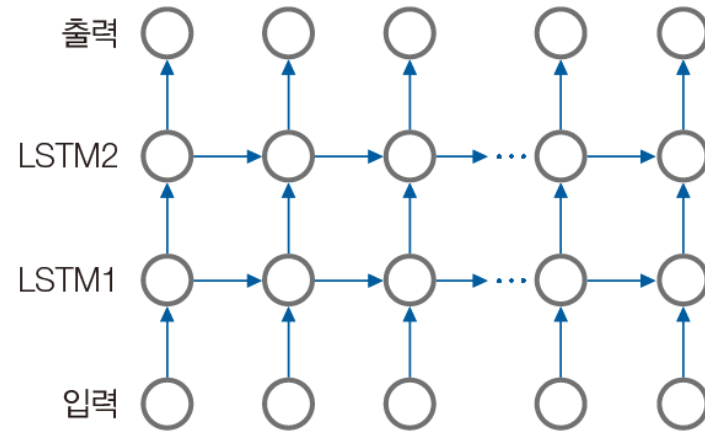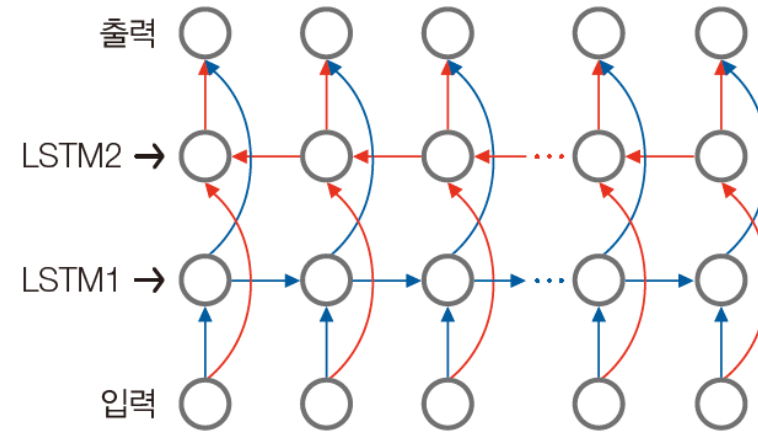
❖ A variety structure according to each problem

(a) 분류 또는 예측 문제

(b) 비디오 프레임 분류

(c) 언어 번역

❖ Learning Phrase Representations using RNN Encoder–Decoder for Statistical

Machine Translation

Gated Recurrent Unit

❖ Structure of GRU

- Recurrent neural networks that simplify gate structure while maintaining the advantages of LSTM

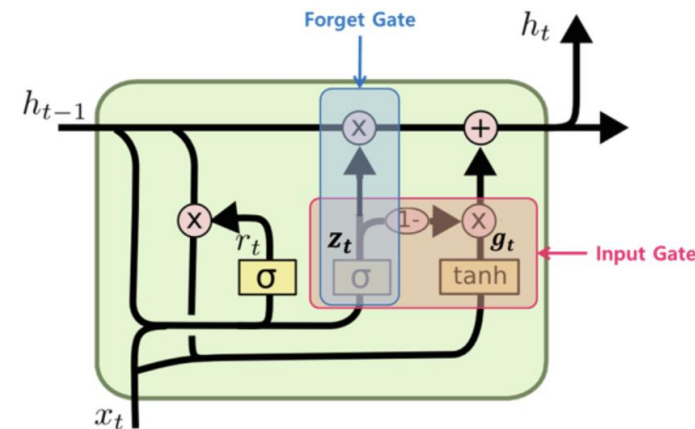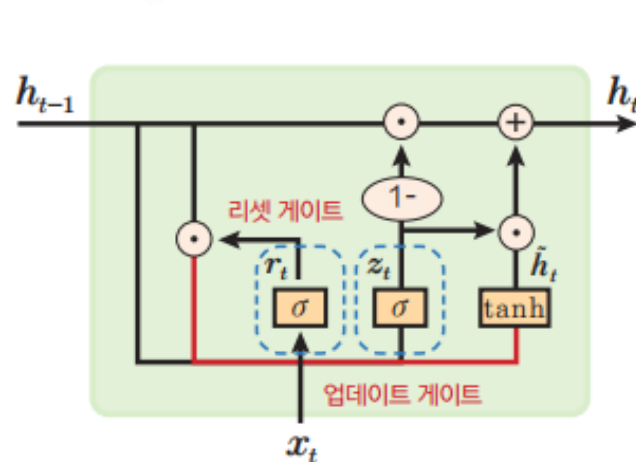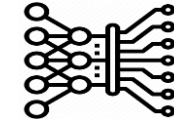- Remove cell state C_t and again hide state h_t remember both long-term and short-term memory

- How the path branches into two branches and then merges again within the GRU cell between h_{t-1} and h_t

❖ Calculate reset gate r_t and update gate z_t

- Process for calculating each gate and hidden status inside the GRU

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

❖ Calculate a newborn memory $\widetilde{h}_t$

$$\text{리셋 게이트} \quad \text{과거의 기억}$$

$$\text{새로운 기억} \quad \tilde{h}_t = \tanh\left((W_{hh} \quad W_{xh}) \binom{r_t \odot h_{t-1}}{x_t} \right)$$

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_{xr}^T \cdot \mathbf{x}_t + \mathbf{W}_{hr}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_r \right)$$

$$\mathbf{z}_t = \sigma\left(\mathbf{W}_{xz}^T \cdot \mathbf{x}_t + \mathbf{W}_{hz}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_z \right)$$

$$\mathbf{g}_t = \tanh\left(\mathbf{W}_{xg}^T \cdot \mathbf{x}_t + \mathbf{W}_{hg}^T \cdot (\mathbf{r}_t \otimes \mathbf{h}_{t-1}) + \mathbf{b}_g \right)$$

$$\mathbf{h}_t = \mathbf{z}_t \otimes \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \otimes \mathbf{g}_t$$

❖ Calculate Update memory h_t

- The updated memory, the hidden state h_t , is calculated as the weighted average of the previous state h_{t-1} and the memory $\widetilde{h}_t$ fo

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

$$\text{업데이트 기억} \quad \text{업데이트 기억}$$

$$\text{과거의 기억} \quad \text{새로운 기억}$$
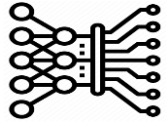
❖ Musical composition

Polyphonic music generation with style transitions using recurrent neural networks

Mathieu De Coster

Supervisors: Bart Dhoedt, Cedric De Boom, Steven Bohez, Sam Leroux

Abstract—Computer aided music generation is a problem that has occupied researchers' minds for several decades. While composition is typically seen as a human skill or an art form, requiring creativity, having an algorithm generate music is something that has merit in certain situations. Consider for example a video game, in which scenes or player actions are often unpredictable. Having music that adapts to the situation could improve player immersion. This master dissertation is not solely about music generation, however. It also considers style transitions, where style is represented as the composer of a piano piece, in this case Bach, Haydn, Beethoven or Chopin. The proposed methods can also be applied to other interpretations of style, such as musical genres or the mood of the music.

A novel data representation is presented. A recurrent neural network is trained on such data and conditioned on composer information. The quality of the compositions created by the neural network is rated closely to the quality of real compositions by a panel of volunteers. The accuracy of composer classifications by this panel is also similar to that of real compositions.

A method for generating polyphonic music with style transitions is presented and forms a basis for future work.

Keywords— Music generation, Style transitions, Recurrent neural networks

I. INTRODUCTION

To be able to compose new music solely with a computer is an interesting and challenging goal. Such a composer algorithm could be used to create new music on the fly, or to

However, traditional RNNs are unable to learn very long time dependencies due to the vanishing and exploding gradient problem [1]. For this reason, the Long-Short Term Memory (LSTM) unit is often used instead of the hyperbolic tangent activation function found in traditional RNNs. Another alternative is the Gated Recurrent Unit (GRU). Both have been shown to perform better than traditional RNNs [2].

Previous research has used both text based methods and methods based on piano rolls. A piano roll is a binary matrix $\mathbf{P} = [P_{i,j}]$, where $P_{i,j}$ is 1 if the note with pitch i is to be played at time j and 0 otherwise. A text based approach was taken by Choi et al. to generate jazz chord progressions [3]. Eck and Schmidhuber have used piano rolls to generate chord progressions from a limited set of chords [4]. Liu et al. also used piano rolls, to generate Bach chorales [5]. Both Liu et al. and Eck and Schmidhuber reported that in piano rolls, it is not possible to distinguish between held notes and repeated notes and that this is a drawback of this data representation.

Style transitions for music generation have not yet been considered in literature. WaveNet however performs conditioning on speaker identity for a text to speech model [6], which is

TABLE I
NETWORK HYPER-PARAMETERS.

Hyper-parameter	Value
Batch size	64
Sequence length	100
Optimiser	Adam
Learning rate	10^{-5}

Output (130)

identity (2)   softmax (128)

Dense (512)

GRU (256)

GRU (256)

Embedding (64), Input (130)

Input (4)
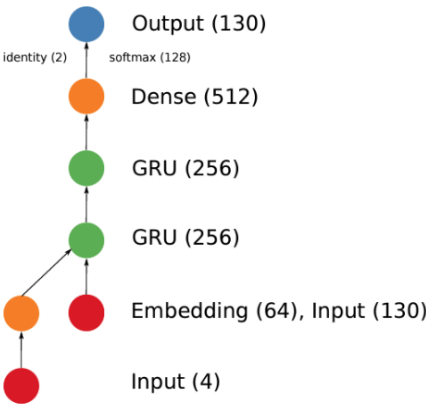
Fig. 1. The network architecture.

https://www.youtube.com/watch?v=j60J1cGINX4
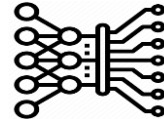
❖ To restore the sound of a video

Visually Indicated Sounds

Andrew Owens[1] Phillip Isola[2,1] Josh McDermott[1]
Antonio Torralba[1] Edward H. Adelson[1] William T. Freeman[1,3]

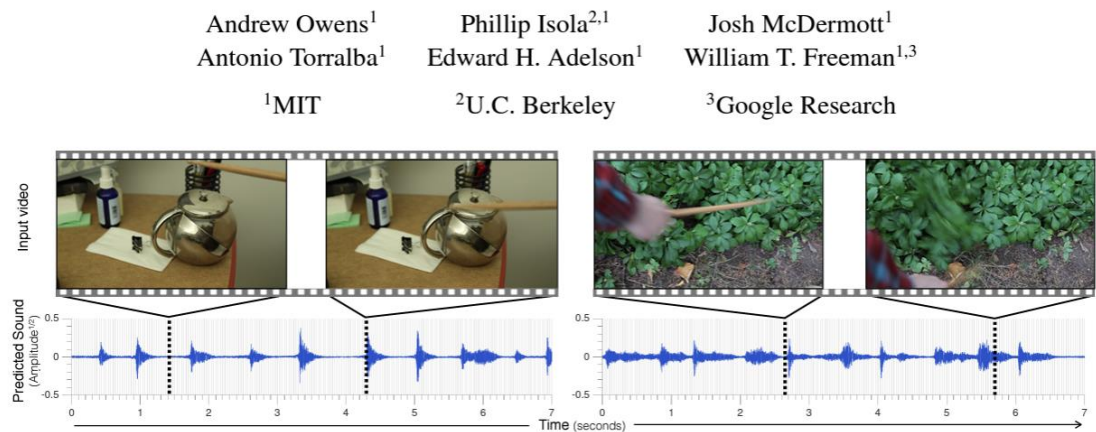[1]MIT [2]U.C. Berkeley [3]Google Research

Figure 1: We train a model to synthesize plausible impact sounds from silent videos, a task that requires implicit knowledge of material properties and physical interactions. In each video, someone probes the scene with a drumstick, hitting and scratching different objects. We show frames from two videos and below them the predicted audio tracks. The locations of these sampled frames are indicated by the dotted lines on the audio track. The predicted audio tracks show seven seconds of sound, corresponding to multiple hits in the videos.

Abstract

Objects make distinctive sounds when they are hit or scratched. These sounds reveal aspects of an object's material properties, as well as the actions that produced them. In this paper, we propose the task of predicting what sound by the physical interaction being depicted: you *see* what is making the sound.

We call these events *visually indicated sounds*, and we propose the task of predicting sound from videos as a way to study physical interactions within a visual scene (Figure 1). To accurately predict a video's held-out soundtrack,
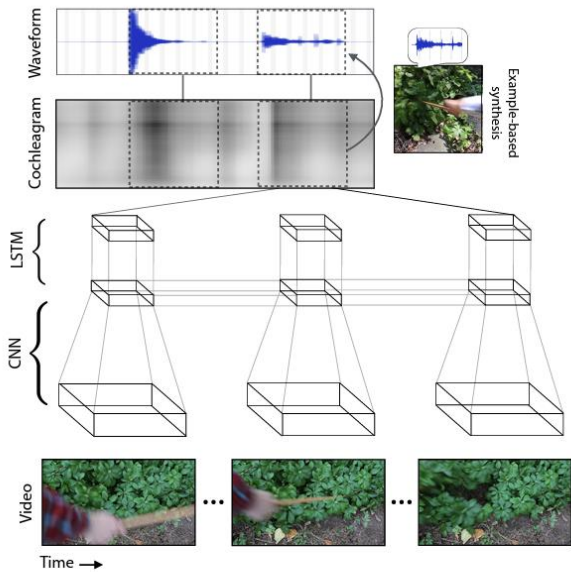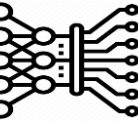
Figure 4: We train a neural network to map video sequences to sound features. These sound features are subsequently converted into a waveform using either parametric or example-based synthesis. We represent the images using a convolutional network, and the time series using a recurrent neural network. We show a subsequence of images corresponding to one impact.

https://www.youtube.com/watch?v=0FW99AQmMc8

https://arxiv.org/abs/1512.08512

❖ Generating handwriting

Generating Sequences With
Recurrent Neural Networks

Alex Graves
Department of Computer Science
University of Toronto
graves@cs.toronto.edu

Abstract

This paper shows how Long Short-term Memory recurrent neural networks can be used to generate complex sequences with long-range structure, simply by predicting one data point at a time. The approach is demonstrated for text (where the data are discrete) and online handwriting (where the data are real-valued). It is then extended to handwriting synthesis by allowing the network to condition its predictions on a text sequence. The resulting system is able to generate highly realistic cursive handwriting in a wide variety of styles.

1   Introduction

Recurrent neural networks (RNNs) are a rich class of dynamic models that have been used to generate sequences in domains as diverse as music [6, 4], text [30] and motion capture data [29]. RNNs can be trained for sequence generation by

https://arxiv.org/abs/1308.0850

31

❖ Stock prediction

Applications

❖ Stock prediction

❖ Text generation (seq2seq model)
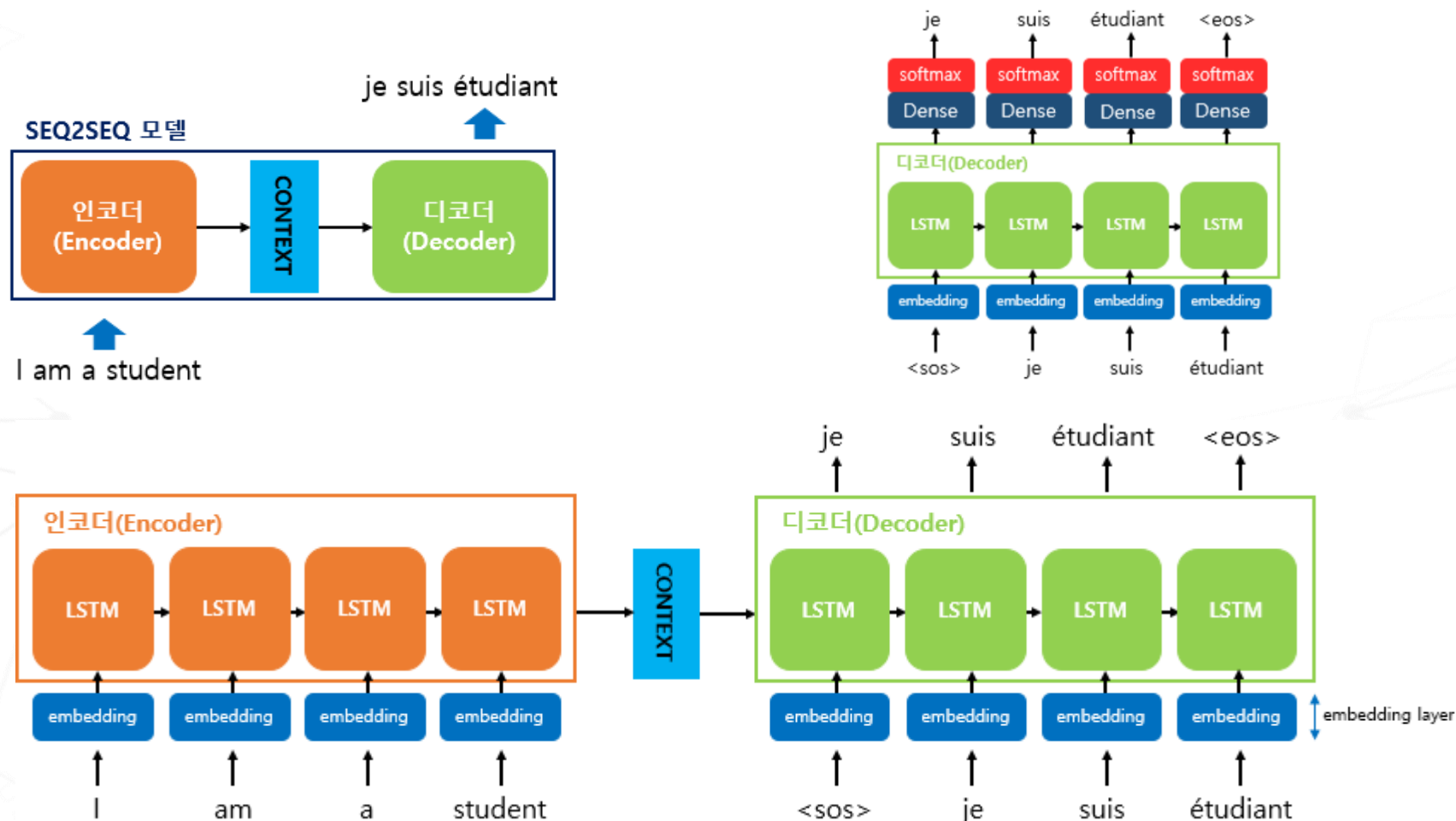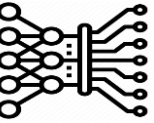
❖ Neurophysiological signal decoding

❖ Mental state detection