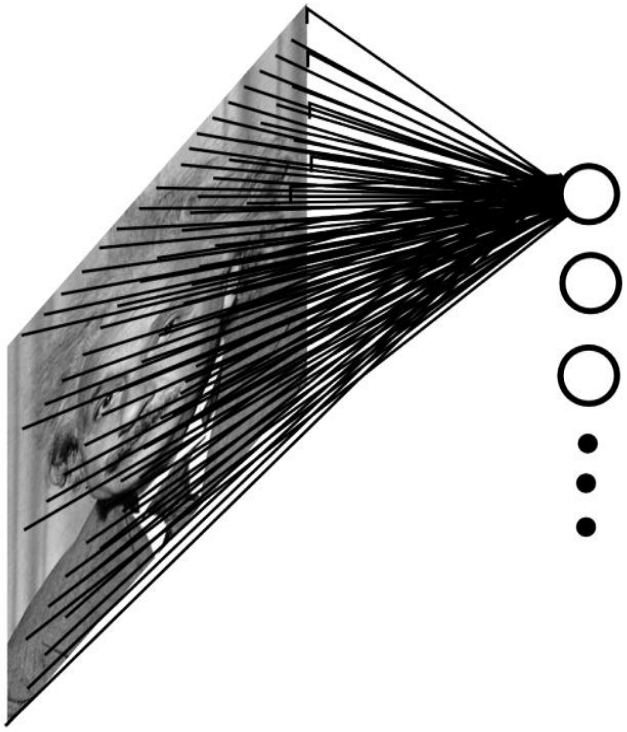
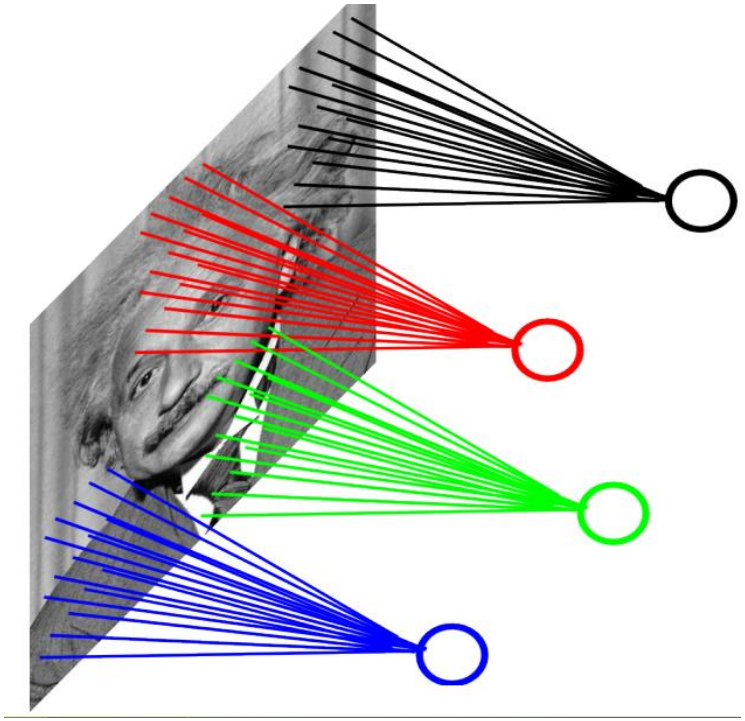
합성곱 신경망

CNN(Convolutional Neural Network)

- A **fully connected layer**:
  - Example:
    - 100x100 images
    - 1000 units in the input
  - Problems:
    - 10^7 edges!
    - Spatial correlations lost!
    - Variables sized inputs.



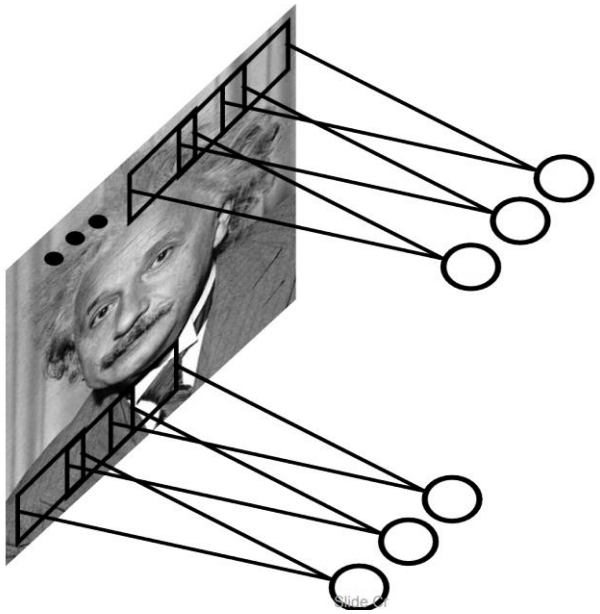
- Consider a task with image inputs:
- A **locally connected layer**:
  - Example:
    - 100x100 images
    - 1000 units in the input
    - Filter size: 10x10
  - Local correlations preserved!
  - Problems:
    - 10^5 edges
    - This parameterization is good when input image is registered (e.g., face recognition).
    - Variable sized inputs, again.



Convolutional Layer

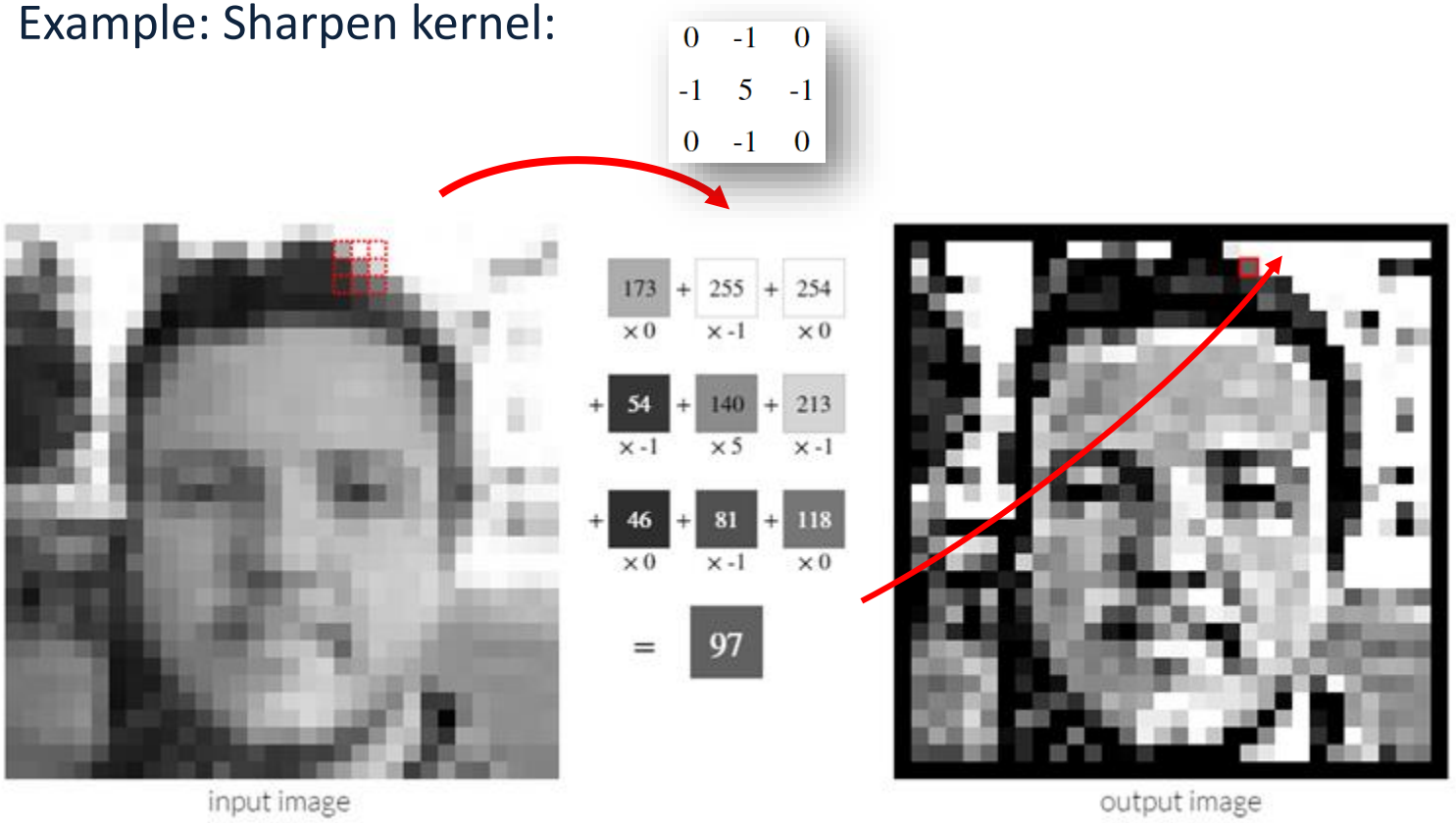
- **A solution:**
  - **Filters** to capture different patterns in the input space.
    - **Share** parameters across different locations (assuming input is stationary)
    - **Convolutions** with learned filters
  - Filters will be **learned** during training.
  - The issue of variable-sized inputs will be resolved with a **pooling** layer.

So what is a convolution?



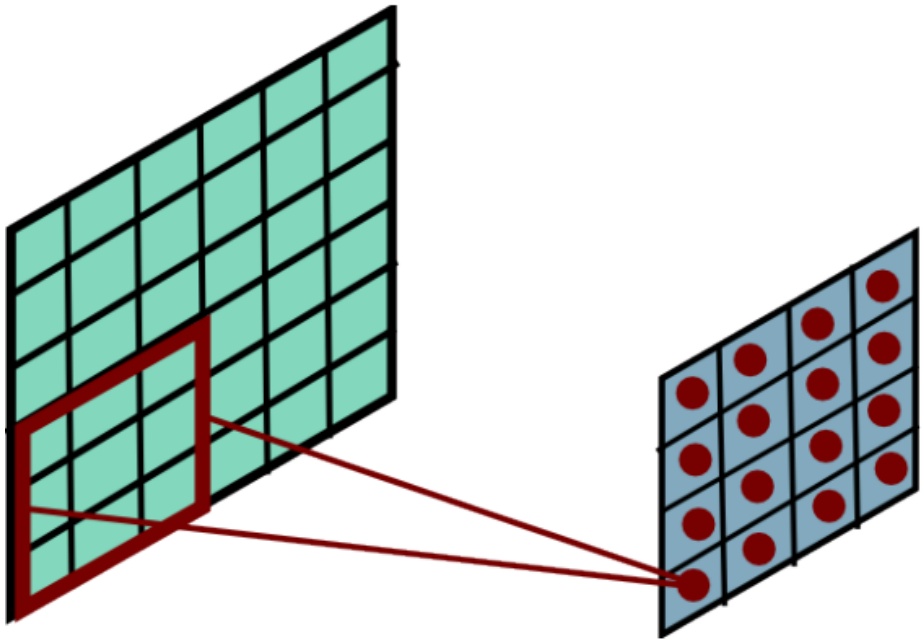
Convolution Operator

- Convolution in two dimension:
  - The same idea: flip one matrix and slide it on the other matrix
  - Example: Sharpen kernel:



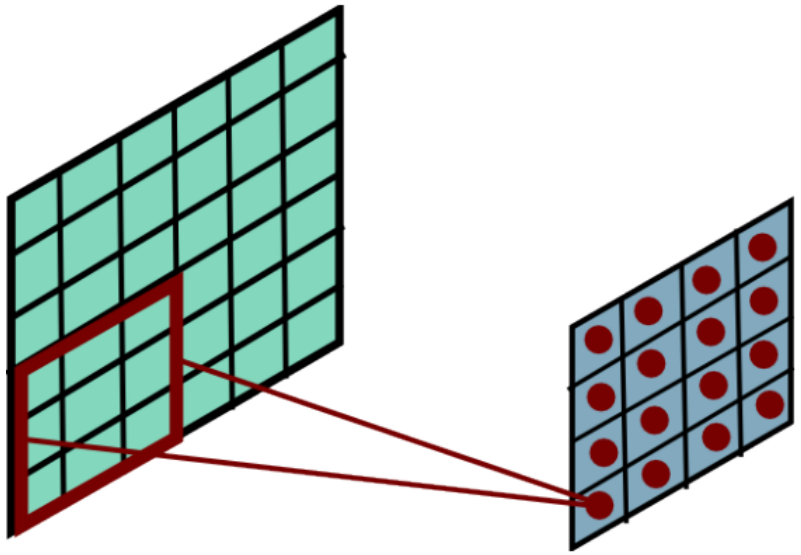
Convolution Operator

- Convolution in two dimension:



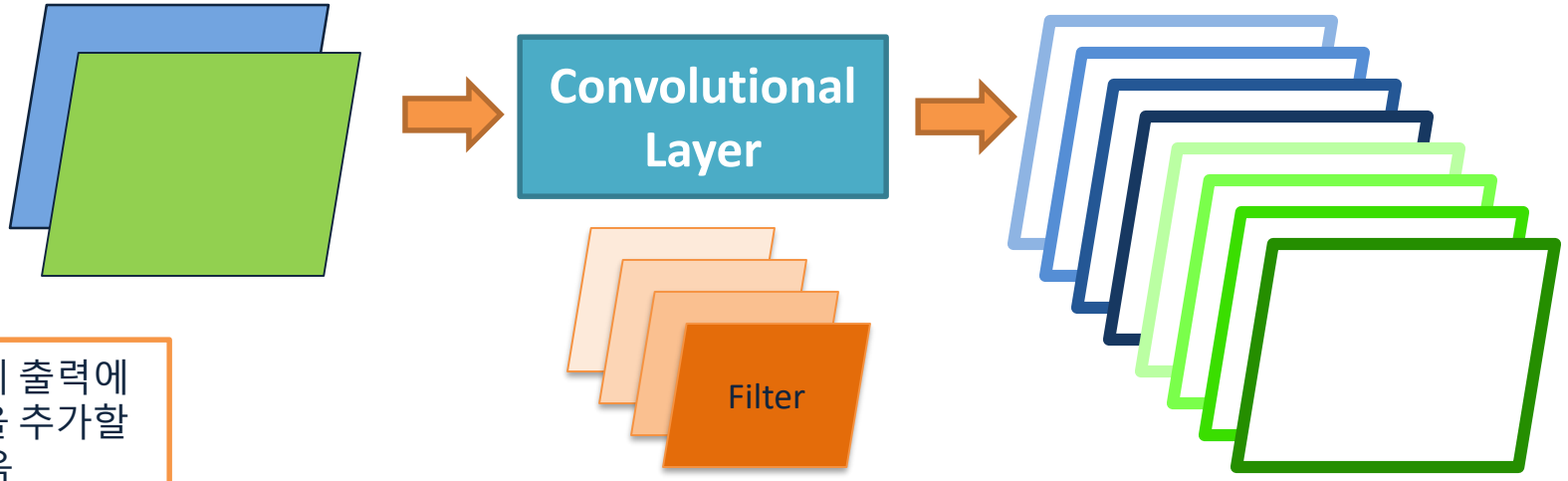
Complexity of Convolution

- Complexity of convolution operator is $n\log(n)$, for n inputs.
- For two-dimension, each convolution takes $MN\log(MN)$ time, where the size of input is MN.



Convolutional Layer

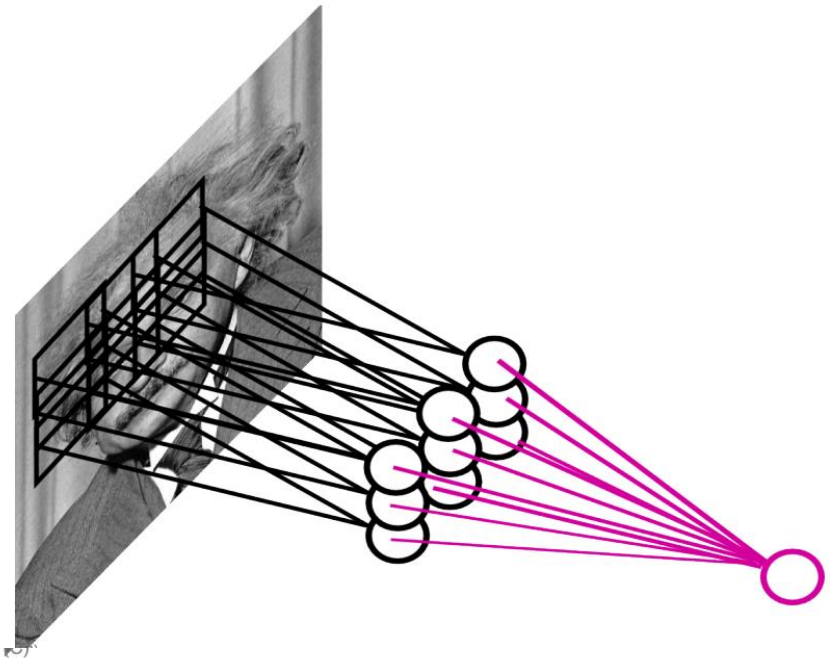
- 입력 (벡터/매트릭스)과 가중치 (벡터/매트릭스)에 컨벌루션 연산을 적용하여 최종 벡터/매트릭스를 생성
- 각 컨벌루션 계층 별로 여러 개의 필터를 사용하여 각 출력 값 생성.
- 중간 층에서는 여러 개의 입력을 가질 수 있음



컨벌루션 층의 출력에 비선형 계층을 추가할 수 있음

Pooling Layer

- 풀링: 입력 이미지 요약, 및 입력 크기 조정



Pooling Layer

- 입력 크기 조정
  - 크기가 다른 입력을 고정 크기로 축소

- 입력값 요약 함수
  - Max pooling

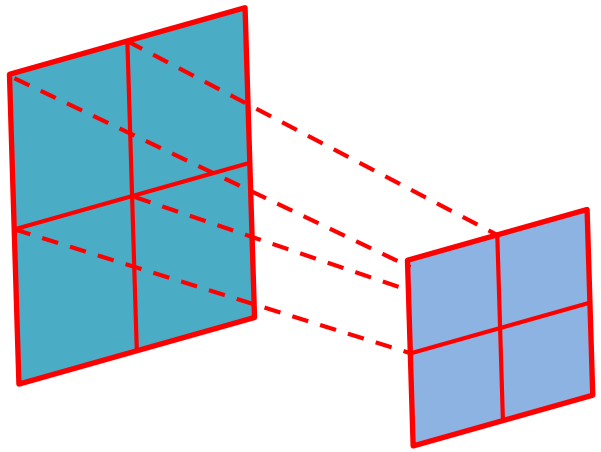
$$h_i[n] = \max_{i \in N(n)} \tilde{h}\,[i]$$

  - Average pooling

$$h_i[n] = \frac{1}{n} \sum_{i \in N(n)} \tilde{h}\,[i]$$

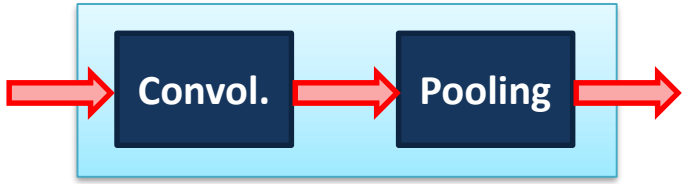
  - L2-pooling

$$h_i[n] = \frac{1}{n} \sqrt{\sum_{i \in N(n)} \tilde{h}^2\,[i]}$$

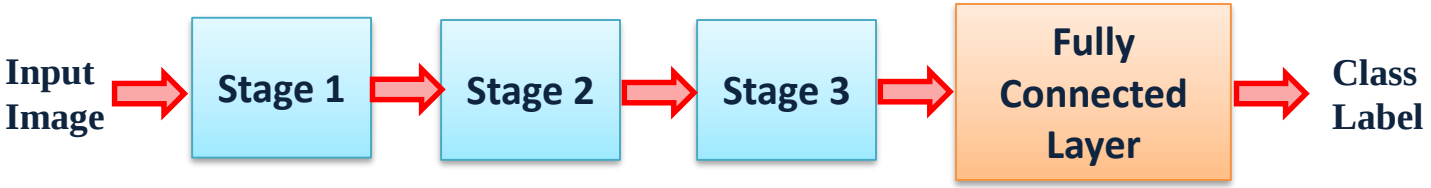


신경망 구조

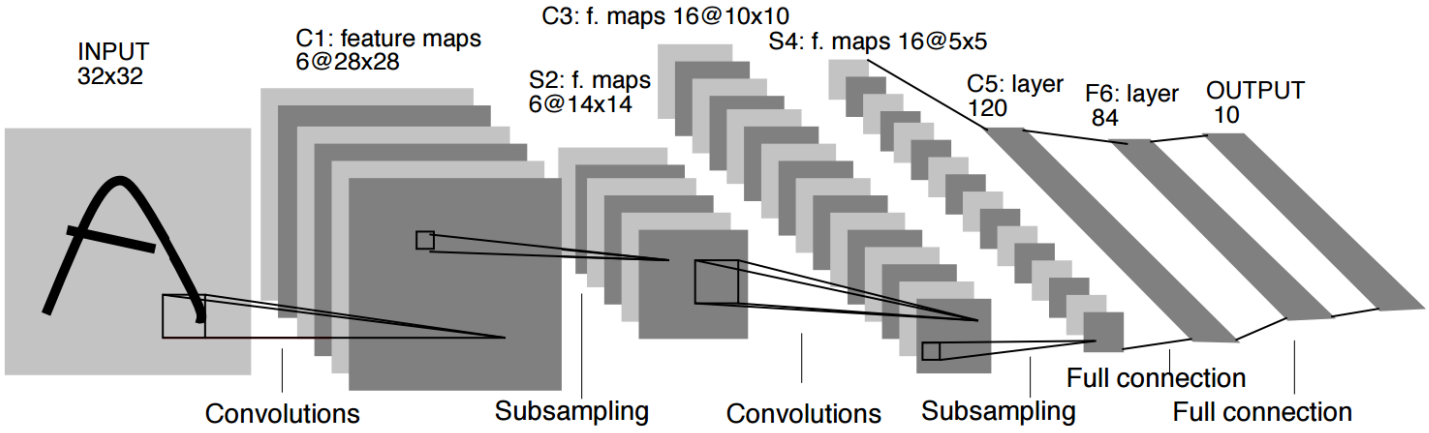
- 단위 구조:



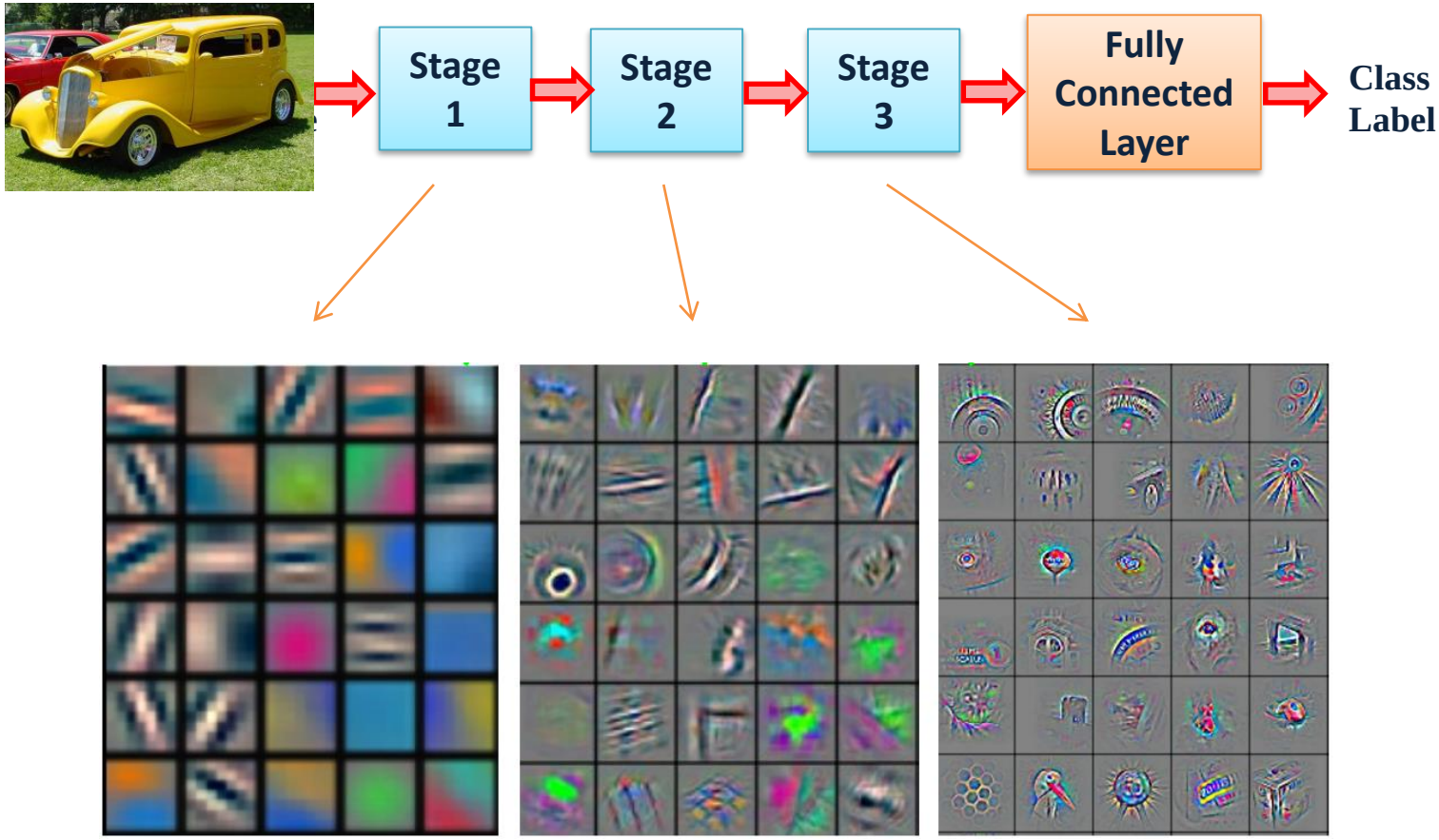
- 전체 시스템:



LeNet 구조:

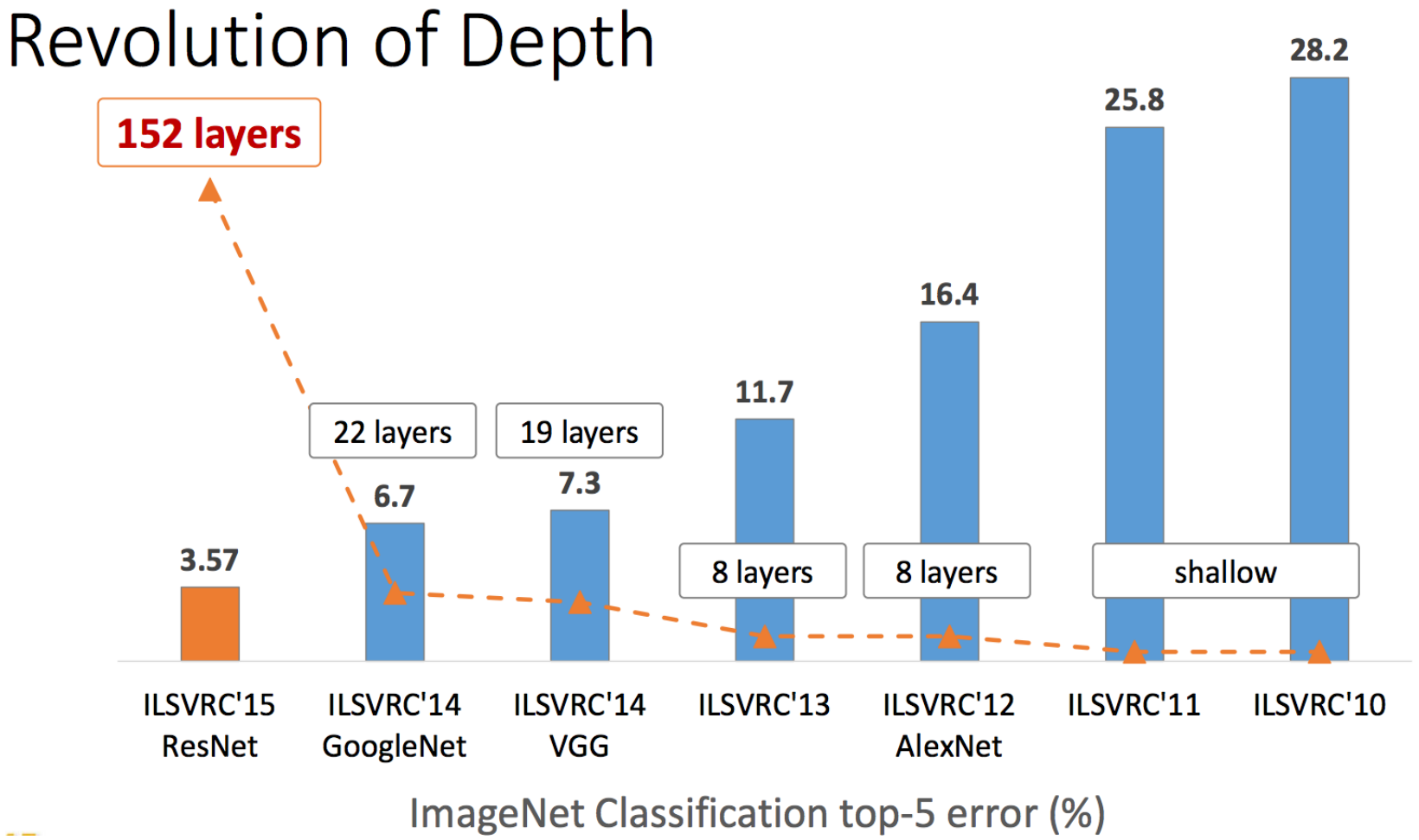


신경망 원리



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

모델 깊이와 성능


Revolution of Depth
152 layers
22 layers
19 layers
8 layers
8 layers
shallow
3.57
6.7
7.3
11.7
16.4
25.8
28.2
ILSVRC'15 ResNet
ILSVRC'14 GoogleNet
ILSVRC'14 VGG
ILSVRC'13
ILSVRC'12 AlexNet
ILSVRC'11
ILSVRC'10
ImageNet Classification top-5 error (%)

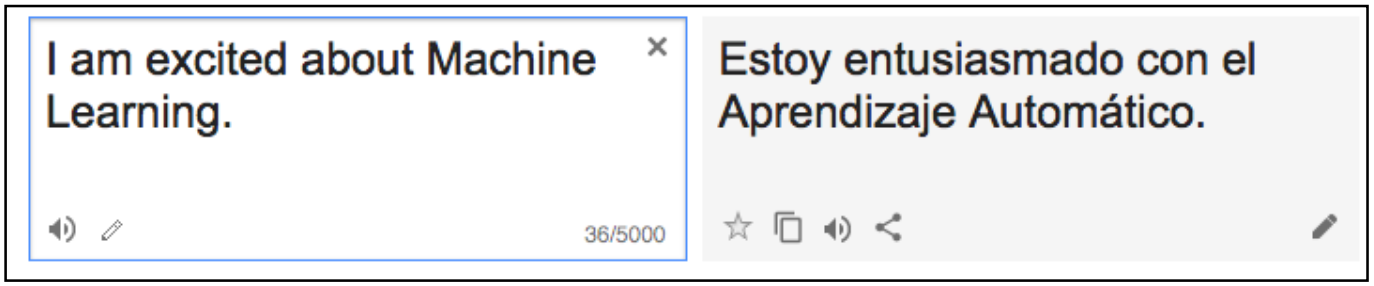
순환 신경망

RNN(Recurrent Neural Network)

Motivation

실시간으로 변화하는 데이터를 모델링

기계 번역



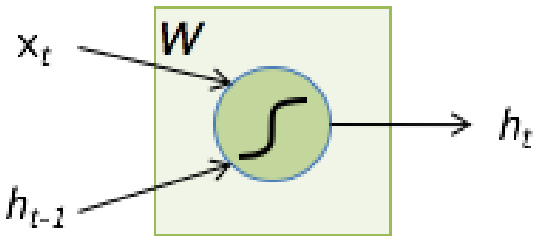
음성 인식 : translating spoken sentence to written sentence

이슈: 충분히 큰 Window를 선택하는 것이 어렵거나 불가능
(항상 새로운 문장이 올 수 있음)

Recurrent Neural Networks

- 순환 신경망(Recurrent Neural Networks)은 입력 시퀀스의 각 요소를 하나씩 입력 받음
- 순환 신경망은 이전 출력이나 은닉 상태를 입력으로 받음
- 시간 t에서의 복합 입력은 시간 T < t에서 일어난 사건에 대한 일부 역사적 정보를 저장
- 과거 입력에 대한 모든 정보를 중간 상태에서 저장하여 다음 단계로 전달

RNN cell



Input at time t is $\mathbf{x_t}$

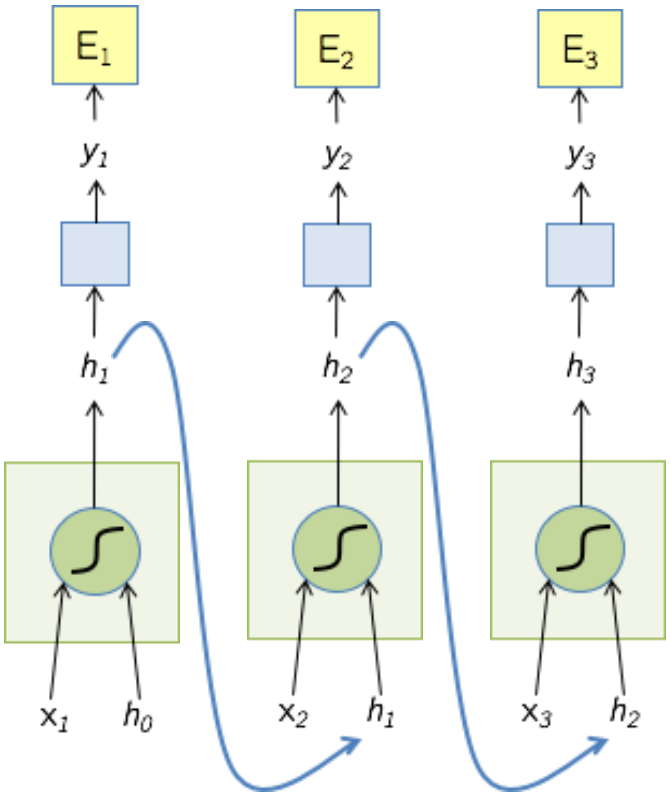
State at the end of time (t-1) is $\mathbf{h_{t-1}}$

State at the end of time t is $\mathbf{h_t}$

$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

You can obviously replace **tanh** by your favorite non-linear differentiable function

Feeding a sequence to an RNN



$$h_t = \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

You can output **y_t** at each time step (based on your problem), based on **h_t**

$$y_t = F(h_t)$$

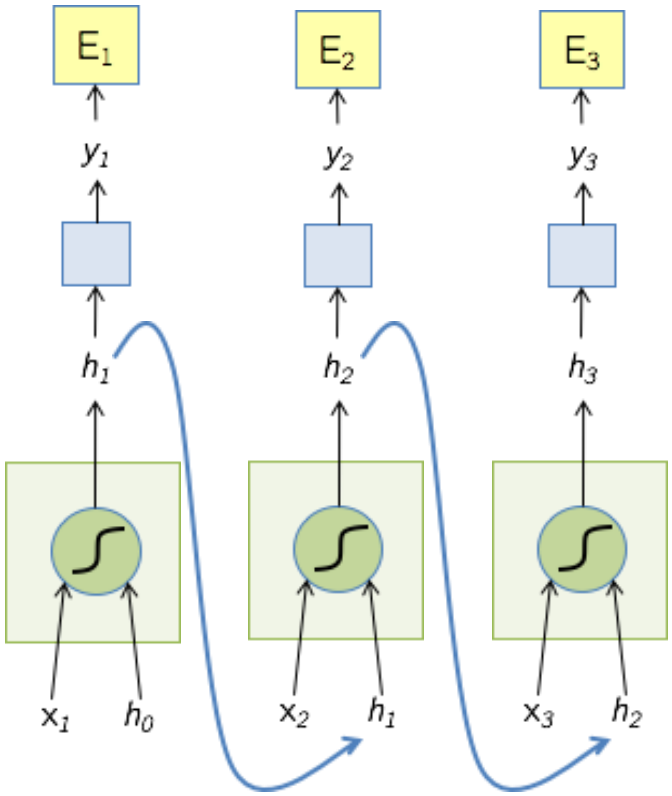
h_t의 선형함수

Finally define error **E_t** based on **y_t** and the target output at time **t**

$$E_t = Loss(y_t, GT_t)$$

GT_t is target at time **t**

RNNs



- Note that the weights are shared over time
- Essentially, copies of the RNN cell are made over time (unrolling/unfolding), with different inputs at different time steps

Sentiment Classification

식당, 영화 등에 대한 리뷰를 분석하여 선호도 분류

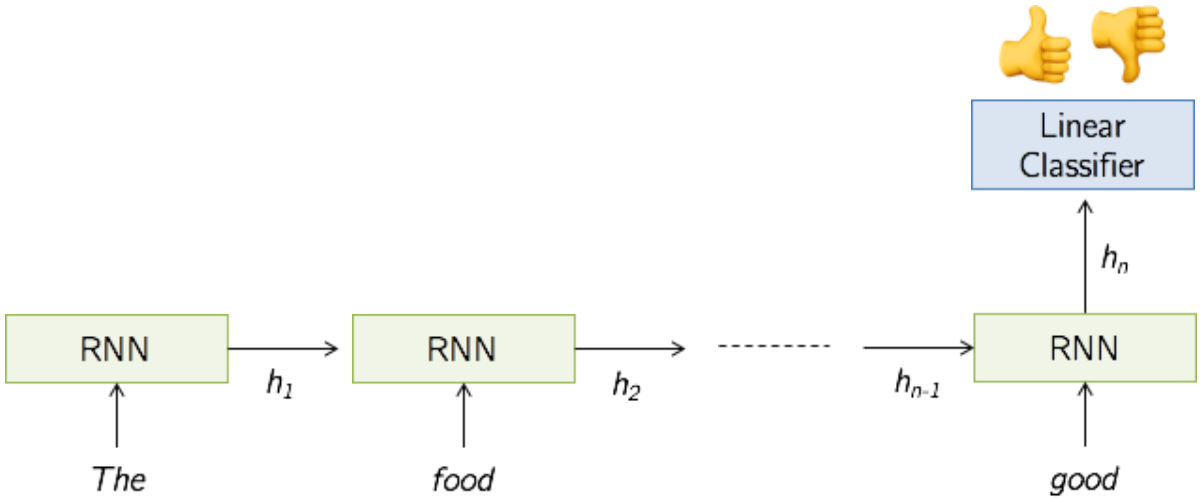
Inputs : Multiple words, one or more sentences

Outputs : Positive or negative classification

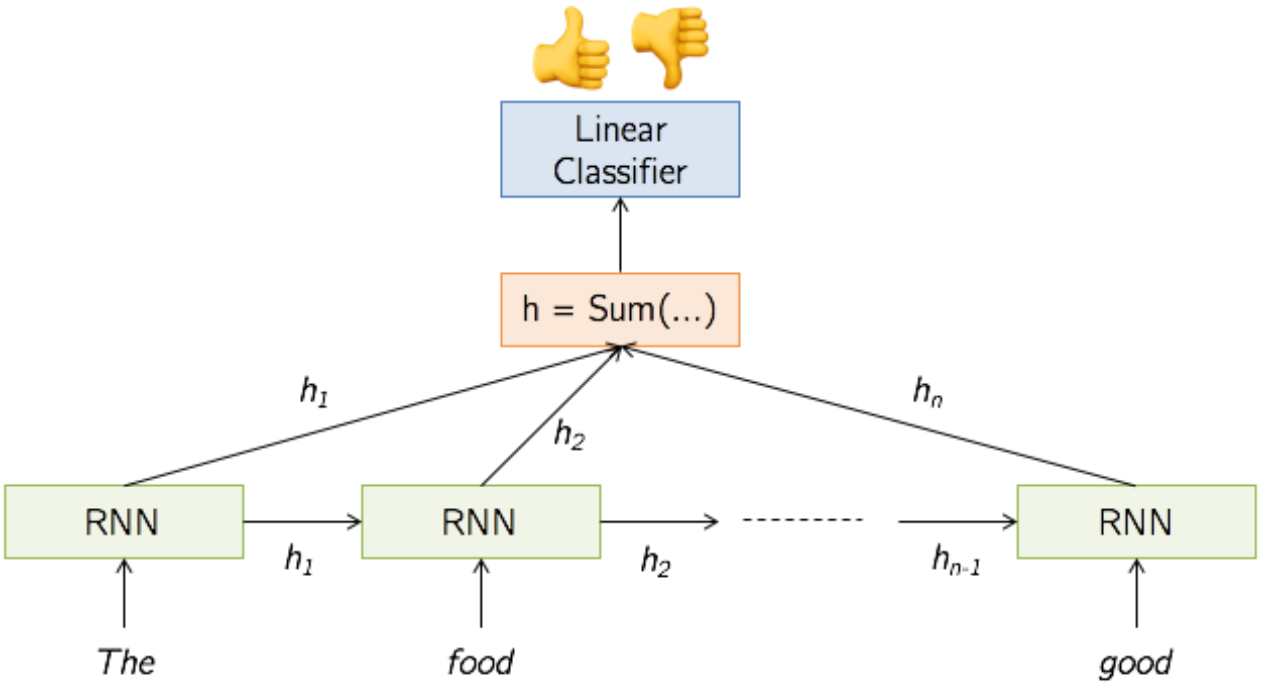
“The food was really good”

“Don’t try the pizza, its so good you will come back everyday. I hate this place. ”

Sentiment Classification



Sentiment Classification



Input Output Scenario

Single - Single

Feed-forward network

Single - Multiple

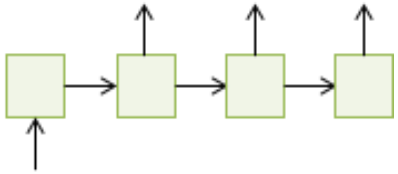
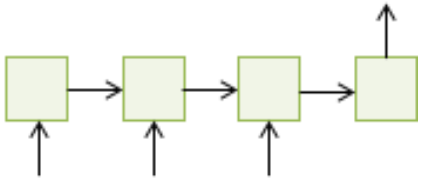


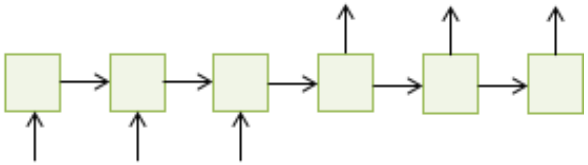
Image captioning

Multiple - Single



Sentiment classification

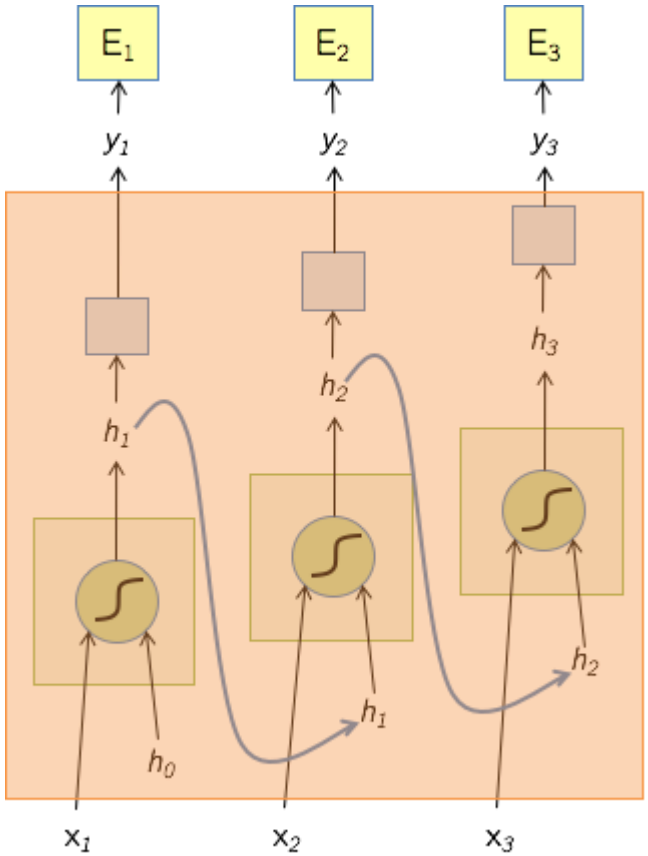
Multiple - Multiple



Translation

Backpropagation through Time (BPTT)

시간 역전파



- RNN 구조를 시간축으로 펼쳐서 하나의 큰 피드포워드 네트워크로 처리
- 펼쳐진 네트워크는 전체 시계열을 입력으로 받음
- 펼쳐진 네트워크의 각 복사본에 대해 가중치 업데이트가 계산되고, 그런 다음 합산(또는 평균)되어 RNN 가중치에 적용

RNN 이슈

역전파는 각 변수에 대한 기울기 계산 필요. RNN의 경우, $\mathbf{h_1}$ 에 대한 $\mathbf{E_t}$ 의 기울기를 계산 하기 위해 긴 체인 필요

$$\frac{\delta E_t}{h_1} = \left(\frac{\delta E_t}{\delta y_t}\right)\left(\frac{\delta y_t}{\delta h_1}\right)$$

$$= \left(\frac{\delta E_t}{\delta y_t}\right)\left(\frac{\delta y_t}{\delta h_t}\right)\left(\frac{\delta h_t}{\delta h_{t-1}}\right)\cdots\left(\frac{\delta h_2}{\delta h_1}\right)$$

기울기 소실 현상으로 인해 0으로 소멸하거나 기울기 폭주 현상으로 무한대로 증가하는 항이 발생. 기울기 폭주는 최대값을 설정하는 방식으로 해결. 기울기 소실 문제는 ht와 ht-1의 미분값 간의 비율을 지정하는 방식으로 해결

$$\left(\frac{\delta h_t}{\delta h_{t-1}}\right) = 1$$

기울기 소실 문제를 완화시킨 Long Short Term Memory (LSTM) 방식이 있으나 완전 해결은 어려운 문제이고 장거리 의존성을 포착하는 것은 도적 과제임.