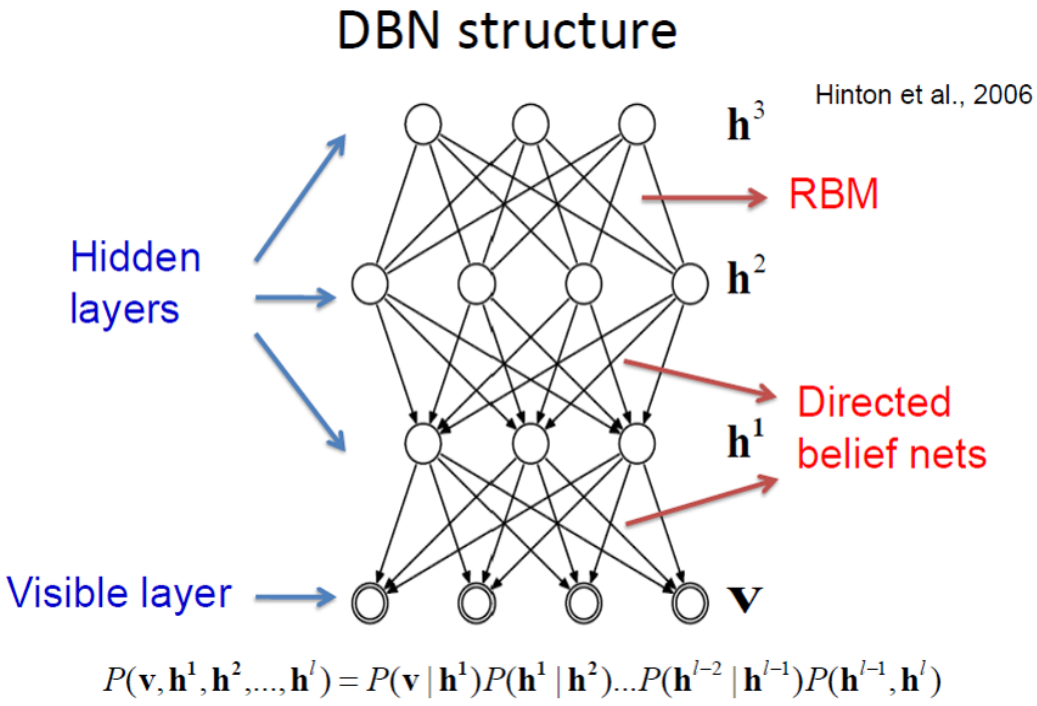
추천알고리즘 2-숙제 Open Ploblems

김용대[1]

서울대학교 통계학과[1]

16. 연속형 변수를 위한 deep belief network

Deep Belief Network



$$P(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2,...,\mathbf{h}^l) = P(\mathbf{v}\,|\,\mathbf{h}^1)P(\mathbf{h}^1\,|\,\mathbf{h}^2)...P(\mathbf{h}^{l-2}\,|\,\mathbf{h}^{l-1})P(\mathbf{h}^{l-1},\mathbf{h}^l)$$

16. 연속형 변수를 위한 deep belief network

Deep Belief Network with real valued input

- $\mathbf{v}$ 는 real-valued, $\mathbf{h}^k, k = 1, \ldots, l$ 이 binary 값을 같는다고 가정하면 모형은 다음과 같다.

$$
\begin{aligned}
P(\mathbf{v}, \mathbf{h}^1, \ldots, \mathbf{h}^l) &= P(\mathbf{v}|\mathbf{h}^1)P(\mathbf{h}^1|\mathbf{h}^2)\cdots P(\mathbf{h}^{l-2}|\mathbf{h}^{l-1})P(\mathbf{h}^{l-1}, \mathbf{h}^l) \\
P(\mathbf{v}|\mathbf{h}^1) &= \mathcal{N}\left(\mathbf{v}; b_i^1 + \mathbf{w}^{1T}\mathbf{h}^1, \Sigma\right) \\
P(\mathbf{h}^k|\mathbf{h}^{k+1}) &= \prod_i P(h_i^k|\mathbf{h}^{k+1}), k = 1, \ldots, l-2 \\
P(\mathbf{h}^{l-1}, \mathbf{h}^l) &= \frac{1}{Z} e^{-E(\mathbf{h}^{l-1}, \mathbf{h}^l)} \quad : RBM
\end{aligned}
$$

where $\mathcal{N}(\cdot)$ denotes a Gaussian distribution and $\Sigma = diag\{\sigma_i^2\}$. σ_i^2 is the sample covariances given the training samples.

23. ResNet (He, Kaiming, et al., 2016)

- Advanced CNN
  - 이전의 CNN 모형은 일정 정도 이상으로 layer를 깊게 쌓으면 Vanishing gradient 문제와 Overfitting 문제로 성능이 안좋아짐.
  - Residual Learning 방법은 깊이를 깊게 하면서 학습효과가 좋음.
- Idea : $\mathcal{H}(\mathbf{x})$ 가 target function이면 target function과 input의 차이인 $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$ 를 학습.
  - 만약 $\mathcal{H}$ 가 identity 함수라면 $\mathcal{F}(\mathbf{x}) \approx 0$으로 학습시킨다.
  - 나머지(residual) 을 학습한다는 관점에서 Residual learing이라 한다.

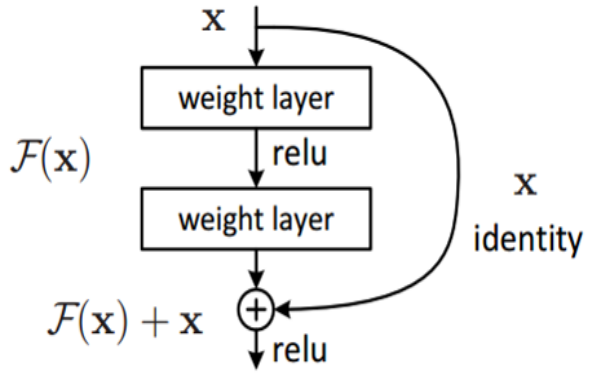


Figure 2. Residual learning: a building block.

23. ResNet (He, Kaiming, et al., 2016)

- **x**를 block의 input, **y**를 block의 output vector라 하면, building block은 다음과 같다.

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}$$

  그림2에서는 $\mathcal{F} = W_2\sigma(W_1\mathbf{x})$이다. 여기서 σ 는 ReLU이고, bias는 편의상 생략함.

- 1개 또는 그 이상의 층을 모수없이 바로 skip하여 연결.

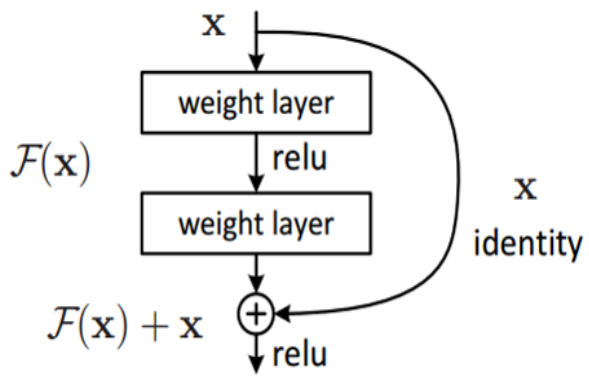
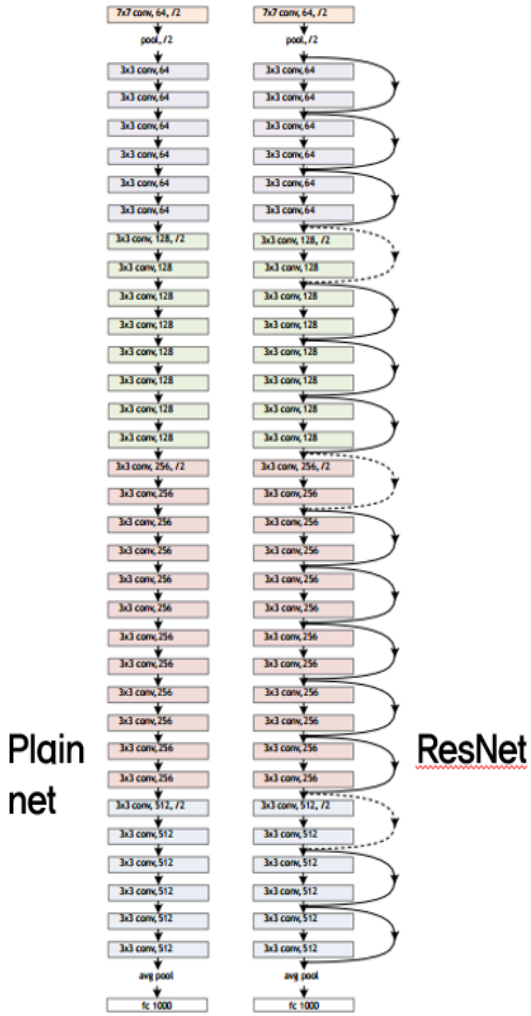


Figure 2. Residual learning: a building block.

23. ResNet (He, Kaiming, et al., 2016)

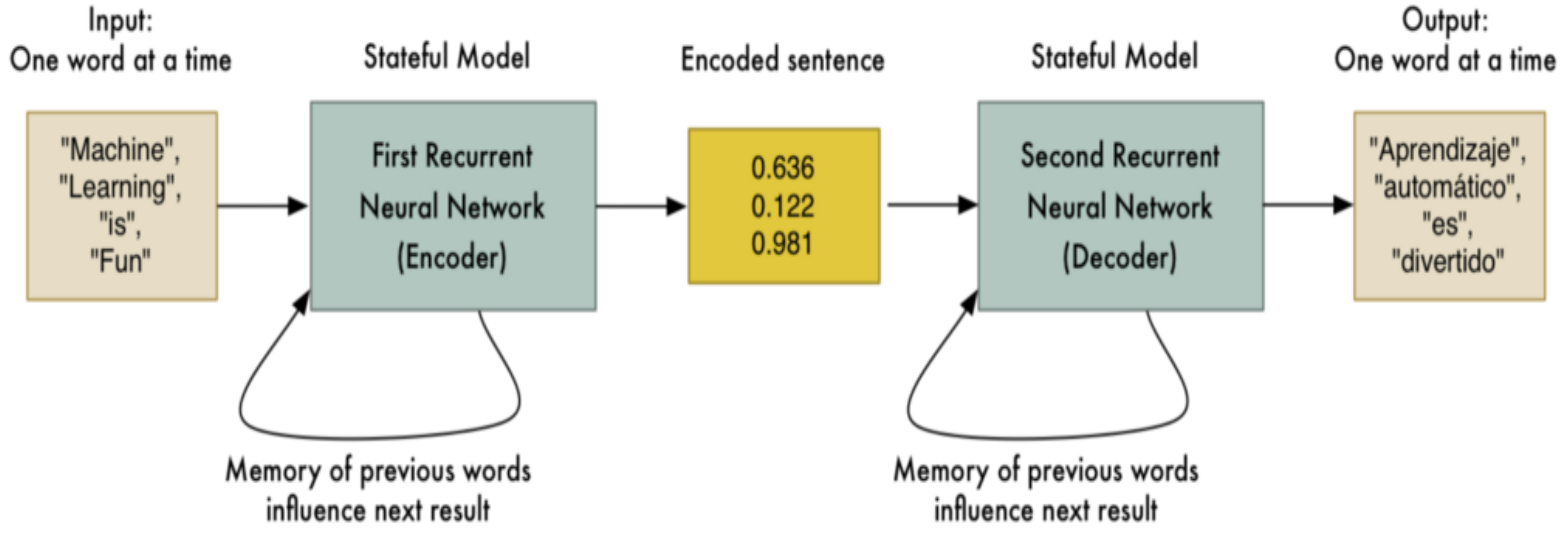
- 총 152개의 중간층들을 사용. Drop-out 사용하지 않음.



28. Machine translation 딥러닝 architecture

RNN Encoder-Decoder(Cho, Kyunghyun, et al., 2014)

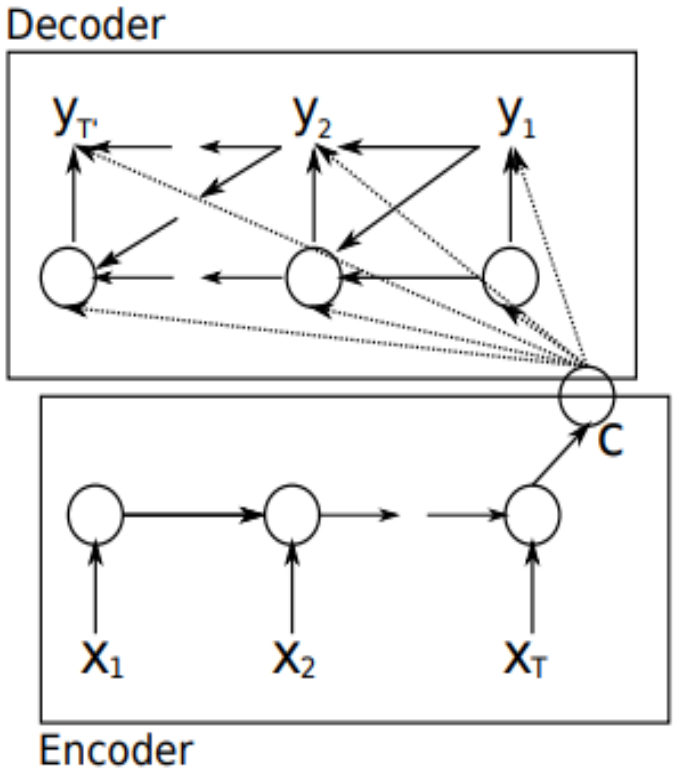
- seq2seq 모형이라고도 함.
- 2개의 RNN 모형으로 구성.
  - Encoder : 하나의 sequence가 입력으로 들어가면, 일정 길이의 vector representation으로 인코딩(encoding)
  - Decoder : 인코딩(encoding)된 실수 벡터로 해당되는 다른 언어의 sequence 생성.



28. Machine translation 딥러닝 architecture

RNN Encoder-Decoder 구조

$X = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$: input 단어열, $Y = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{T'})$: target 단어열, $\mathbf{x}_i(\mathbf{y}_i)$: input(target) 단어열의 i번째 단어의 one-hot vector 일때,



$$P(y_{t,j}|\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}, \mathbf{c}) = g(\mathbf{h}_t, \mathbf{y}_{t-1}, \mathbf{c})$$

$$\mathbf{h}'_t = f'(\mathbf{h}'_{t-1}, \mathbf{y}_{t-1}, \mathbf{c})$$

$$\mathbf{c} = tanh(\mathbf{V}\mathbf{h}_T)$$

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t)$$

$g(\cdot)$, $f(\cdot)$은 nonlinear activation function, 특히 $f(\cdot)$은 GRU 구조 사용(뒷장참조).

28. Machine translation 딥러닝 architecture

GRU in hidden unit

- Encoder

$$\begin{aligned}
\mathbf{z} &= \sigma(\mathbf{W}^z\mathbf{x}_t + \mathbf{U}^z\mathbf{h}_{t-1}) : \text{update gate} \\
\mathbf{r} &= \sigma(\mathbf{W}^r\mathbf{x}_t + \mathbf{U}^r\mathbf{h}_{t-1}) : \text{reset gate} \\
\tilde{\mathbf{h}_t} &= tanh\,(\mathbf{W}\mathbf{x}_t + \mathbf{U}(\mathbf{h}_{t-1} \circ \mathbf{r})) \\
\mathbf{h}_t &= (1 - \mathbf{z}) \circ \tilde{\mathbf{h}_t} + \mathbf{z} \circ \mathbf{h}_{t-1}
\end{aligned}$$

  여기서, $\circ$ 는 element-wise 곱

- Decoder
  Initializing the hidden state with $\mathbf{h}'_0 = tanh(\mathbf{V}'\mathbf{c})$,

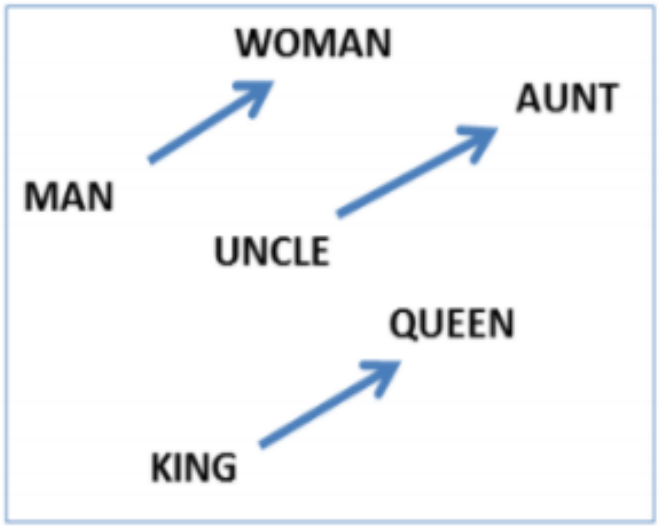
$$\begin{aligned}
\mathbf{z}' &= \sigma(\mathbf{W}^{z\prime}\mathbf{y}_{t-1} + \mathbf{U}^{z\prime}\mathbf{h}'_{t-1} + \mathbf{C}^z\mathbf{c}) : \text{update gate} \\
\mathbf{r}' &= \sigma(\mathbf{W}^{r\prime}\mathbf{y}_{t-1} + \mathbf{U}^{r\prime}\mathbf{h}'_{t-1} + \mathbf{C}^{\mathbf{r}}\mathbf{c}) : \text{reset gate} \\
\tilde{\mathbf{h}_t}' &= tanh\left(\mathbf{W}'\mathbf{y}_{t-1} + \mathbf{U}'(\mathbf{h}'_{t-1} + \mathbf{C}\mathbf{c}) \circ \mathbf{r}'\right) \\
\mathbf{h}'_t &= (1 - \mathbf{z}') \circ \tilde{\mathbf{h}}'_t + \mathbf{z}' \circ \mathbf{h}'_{t-1}
\end{aligned}$$

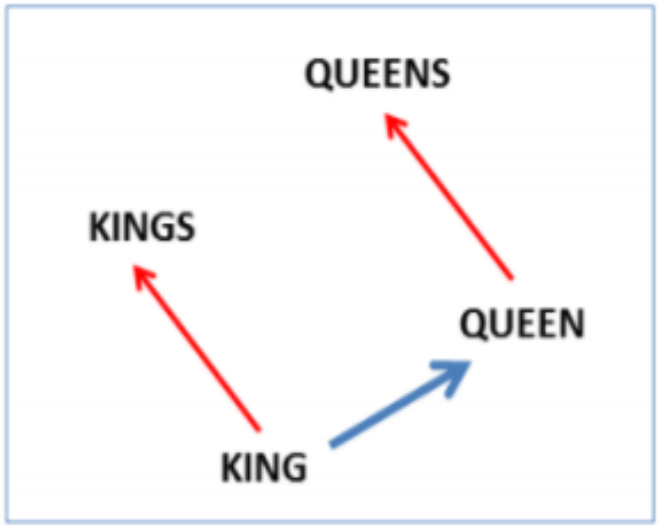
30. Word2vec

Word2vec (T.Mikolov, et al., 2013)

- Word Embedding : 문자로 이루어진 단어를 숫자 벡터로 변환하는 것. 의미 자체가 벡터로 수치화되기 때문에, 벡터 연산을 통해 추론을 할 수 있다.

- Vector Representation을 통한 추론의 예시

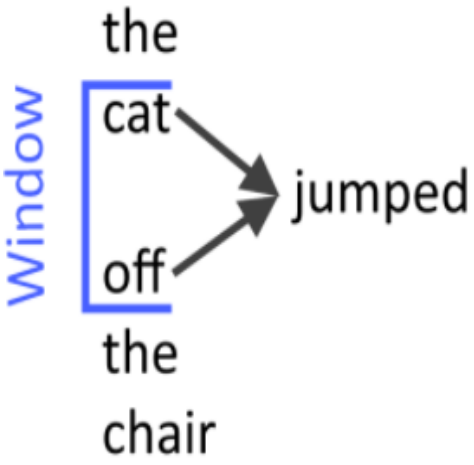


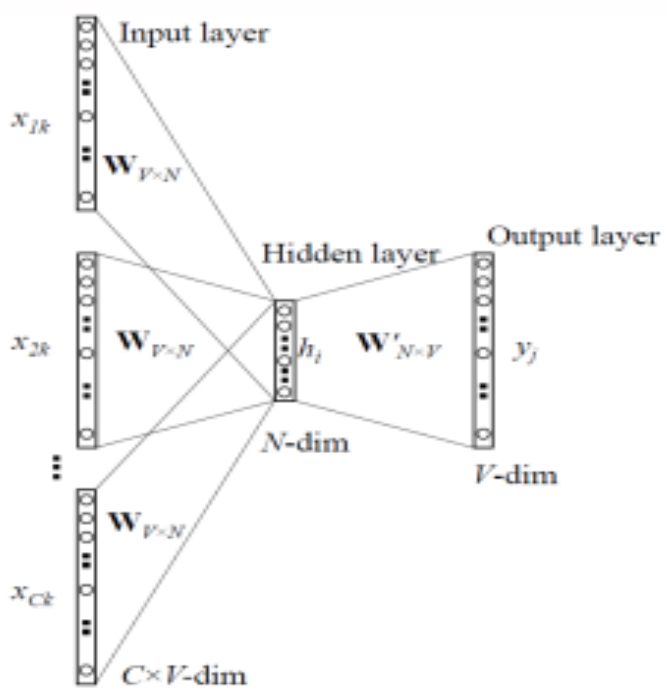


30. Word2vec

Word2Vec 의 두가지 모형 : CBOW 와 Skip-gram

- CBOW(Continuous bag-of words) : 주변 단어를 이용하여 주어진 단어를 맞추기 위한 네트워크
  - 중간층의 학습된 weight matrix $\mathbf{W}_{V\times N}$ 를 Word vector representation으로 사용.
  - window size: 네트워크에 사용하는 주어진 단어 앞뒤 주변단어의 수.





30. Word2vec

Word2Vec 의 두가지 모형 : CBOW 와 Skip-gram

- Skip-gram : CBOW와 반대로 주어진 단어를 가지고 주위에 등장하는 몇개의 단어를 맞추기 위한 네트워크
  - 마찬가지로, 중간층의 학습된 weight matrix $\mathbf{W}_{V\times N}$ 를 Word vector representation으로 사용.