Sequence Modeling

Oct 20th, 2016

Applied Deep Learning

Yun-Nung (Vivian) Chen WWW.CSIE.NTU.EDU.TW/~YVCHEN/F105-ADL

National Taiwan University

Slide credit from Richard Socher

How to Frame the Learning Problem?

The learning algorithm f is to map the input domain X into the output domain Y

$$f : X \to Y$$

Input domain: word, word sequence, audio signal, click logs

Output domain: single label, sequence tags, tree structure, probability distribution

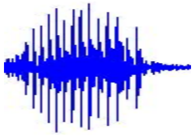
Output Domain – Classification

Sentiment Analysis

"這規格有誠意!" → +

"太爛了吧~" → -

Speech Phoneme Recognition

 → /h/

Handwritten Recognition

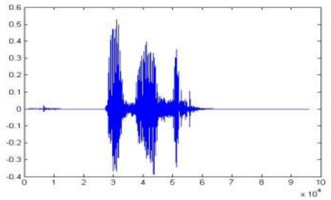
 → 2

Output Domain – Sequence Prediction

POS Tagging

“推薦我台大後門的餐廳” ⟶ 推薦/VV 我/PN 台大/NR 後門/NN 的/DEG 餐廳/NN

Speech Recognition



⟶ “大家好”

Machine Translation

“How are you doing today?” ⟶ “你好嗎?”

Learning tasks are decided by the output domains

Input Domain –
How to Aggregate Information

Input: word sequence, image pixels, audio signal, click logs

Property: continuity, temporal, importance distribution

Example

- CNN (convolutional neural network): local connections, shared weights, pooling
  - AlexNet, VGGNet, etc.
- RNN (recurrent neural network): temporal information
- RvNN (recursive neural network): compositionality

Network architectures should consider the input domain properties

How to Frame the Learning Problem?

The learning algorithm f is to map the input domain X into the output domain Y

$$f : X \rightarrow Y$$

Input domain: word, word sequence, audio signal, click logs

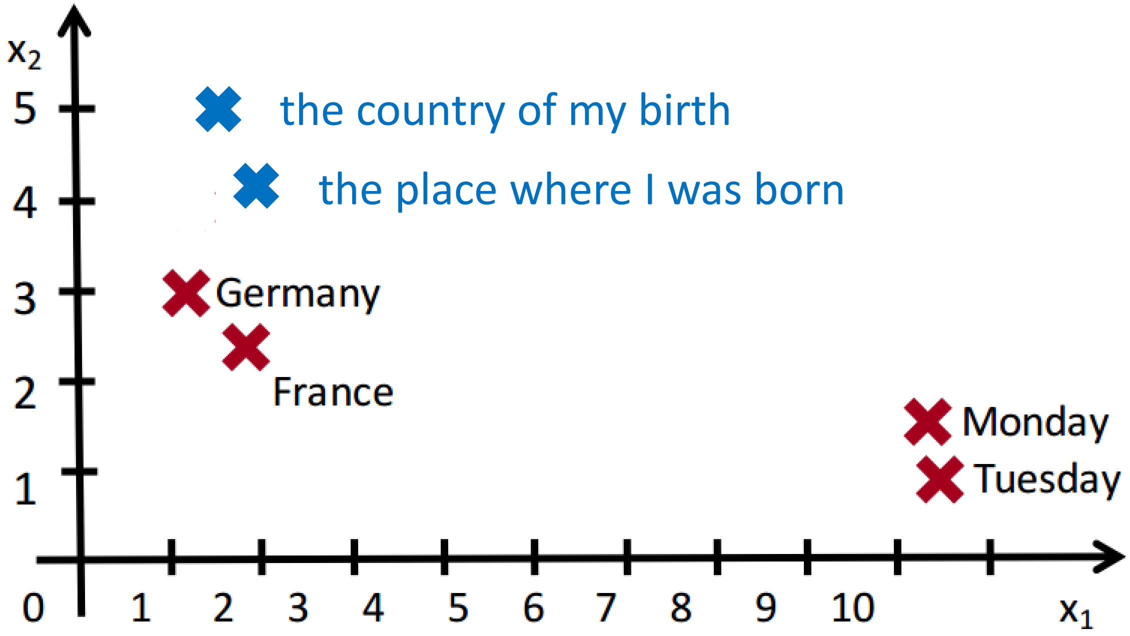
Output domain: single label, sequence tags, tree structure, probability distribution

Network design should leverage input and output domain properties

Review

Word Vector Space

The words can be represented as vectors in the high-dim space



How can we represent the meaning of longer phrases?

Mapping them into same vector space!!

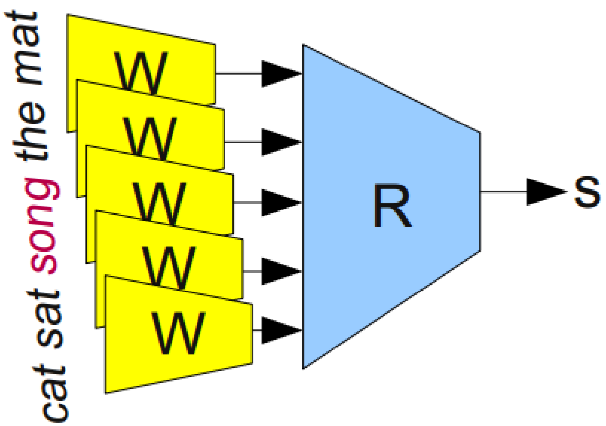
Word Embedding Benefit

Given an unlabeled training corpus, produce a vector for each word that encodes its semantic information. These vectors are useful because:

① semantic similarity between two words can be calculated as the cosine similarity between their corresponding word vectors

② word vectors as powerful features for various supervised NLP tasks since the vectors contain semantic information

③ propagate any information into them via neural networks and update during training



Target Function

Classification Task

$$f(x) = y \quad \Rightarrow \quad f: R^N \rightarrow R^M$$

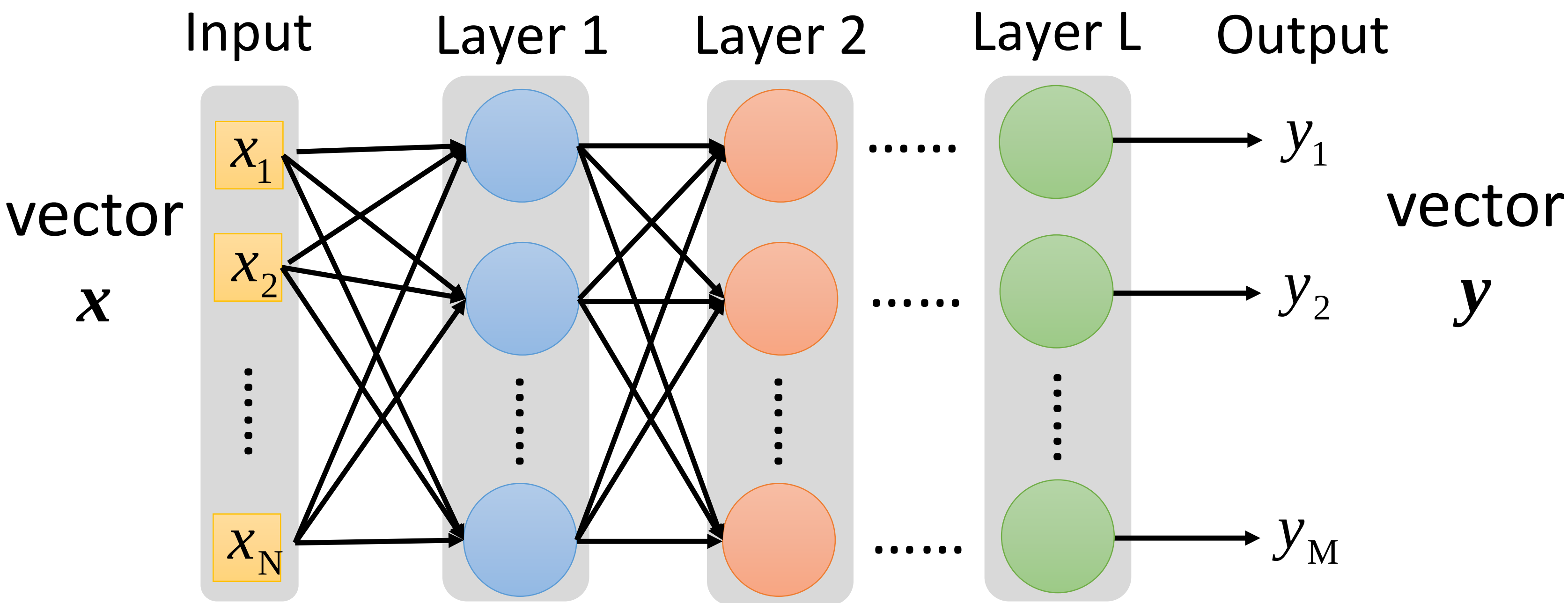
- x: input object to be classified → a N-dim vector
- y: class/label → a M-dim vector

Assume both x and y can be represented as fixed-size vectors

How to use word embeddings for the subsequent tasks

Deep Neural Networks (DNN) $f: R^N \rightarrow R^M$

Fully connected feedforward network



From input vector x to output class vector y

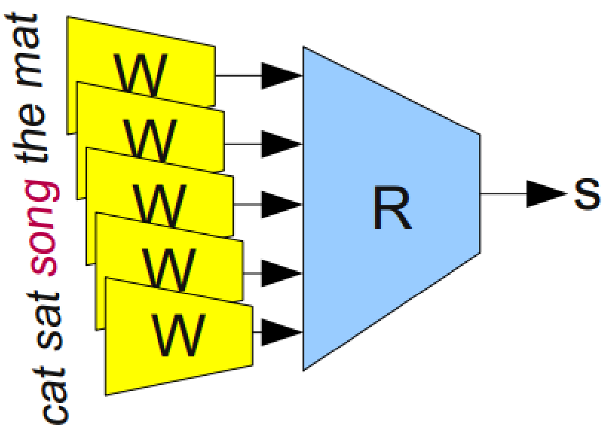
Word Sequence as a Vector

Combine word embeddings into a single input vector

“這規格有誠意!” ⟶ +

“太爛了吧~” ⟶ -

這 (this)	$[0.2\ \ 0.6\ \ 0.3\ \ \cdots\ \ 0.4]$
規格 (specification)	$[0.9\ \ 0.8\ \ 0.1\ \ \cdots\ \ 0.1]$
有 (have)	$[0.1\ \ 0.3\ \ 0.1\ \ \cdots\ \ 0.7]$
誠意 (sincerity)	$[0.5\ \ 0.0\ \ 0.6\ \ \cdots\ \ 0.4]$



$$\vec{x} = [x_1\ \ x_2\ \ x_3\ \ \cdots\ \ x_N]$$

How to compute x?

Semantic Vector Space

single word vector ⟷ document vector

Single word vector
- Distributional representation
- Useful features inside models
- Cannot capture meaning of longer phrases

Document vector
- Bag of words models
- PCA/LSA/LDA
- Great for IR, document exploration
- Ignore word ordering, no detail understanding

Vectors representing *Phrases* and *Sentences* with word order and capture semantics for NLP tasks

Sequence Modeling

Idea: aggregate the meaning from all words into a vector

→ ***Compositionality***

Method:

- Basic combination: average, sum
- Neural combination:
  - ✓Recursive neural network (RvNN)
  - ✓Recurrent neural network (RNN)
  - ✓Convolutional neural network (CNN)

$$
\begin{array}{cc}
 & \overbrace{\hphantom{[0.2\ 0.6\ 0.3\ \cdots\ 0.4]}}^{N\text{-dim}} \\
\text{這 (this)} & \begin{bmatrix} 0.2 & 0.6 & 0.3 & \cdots & 0.4 \end{bmatrix} \\
\text{規格 (specification)} & \begin{bmatrix} 0.9 & 0.8 & 0.1 & \cdots & 0.1 \end{bmatrix} \\
\text{有 (have)} & \begin{bmatrix} 0.1 & 0.3 & 0.1 & \cdots & 0.7 \end{bmatrix} \\
\text{誠意 (sincerity)} & \begin{bmatrix} 0.5 & 0.0 & 0.6 & \cdots & 0.4 \end{bmatrix}
\end{array}
$$

How to compute $\vec{x} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_N \end{bmatrix}$