非线性空间嵌入（embedding）

递归神经网络与LSTM

胡俊峰 2020/12/23

t-Distributed Stochastic Neighbor Embedding（t-SNE）

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_k - x_j\|^2 / 2\sigma_i^2)}$$

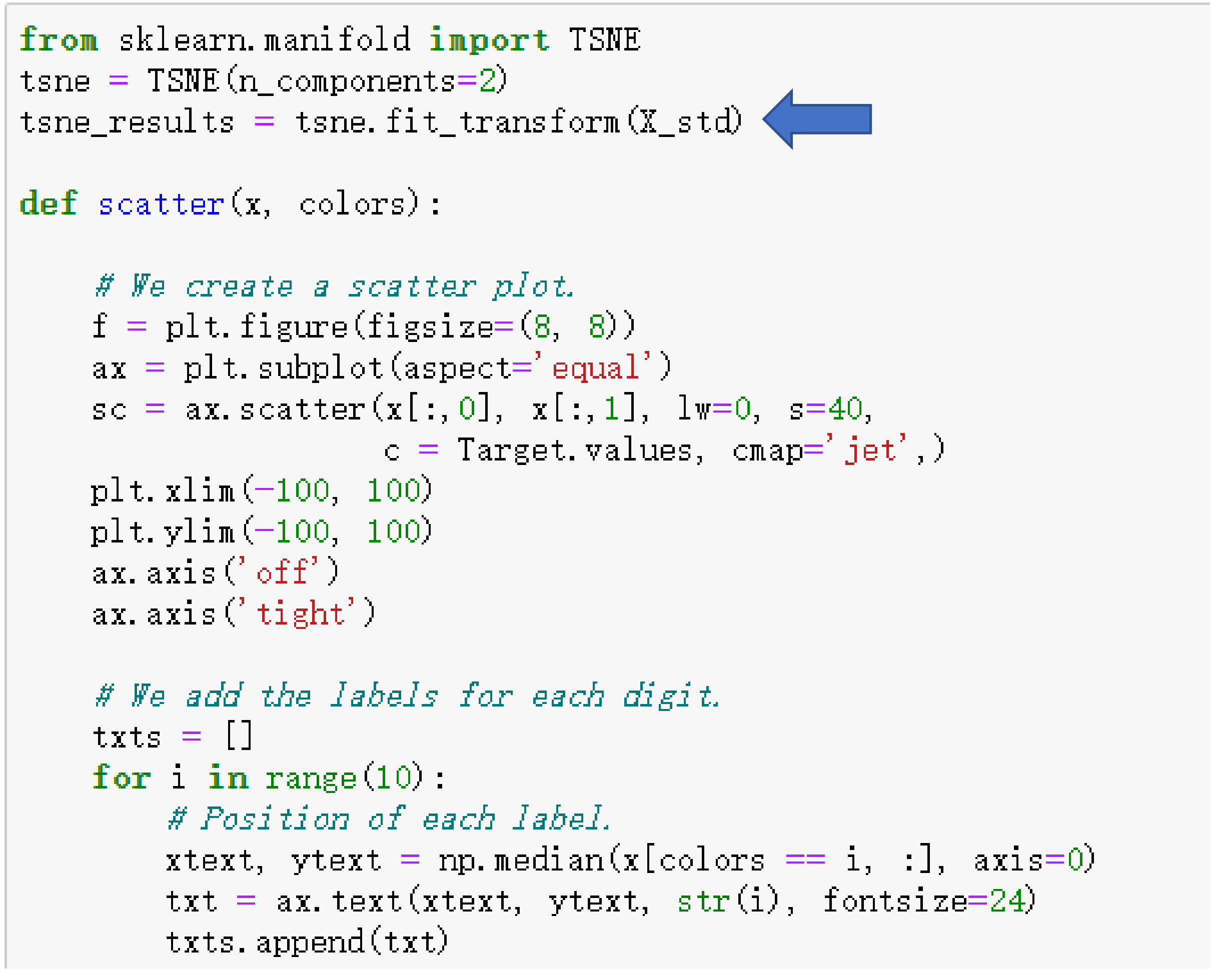
$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_k - y_j\|^2)}$$

9

$$C = \sum_{i,j} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

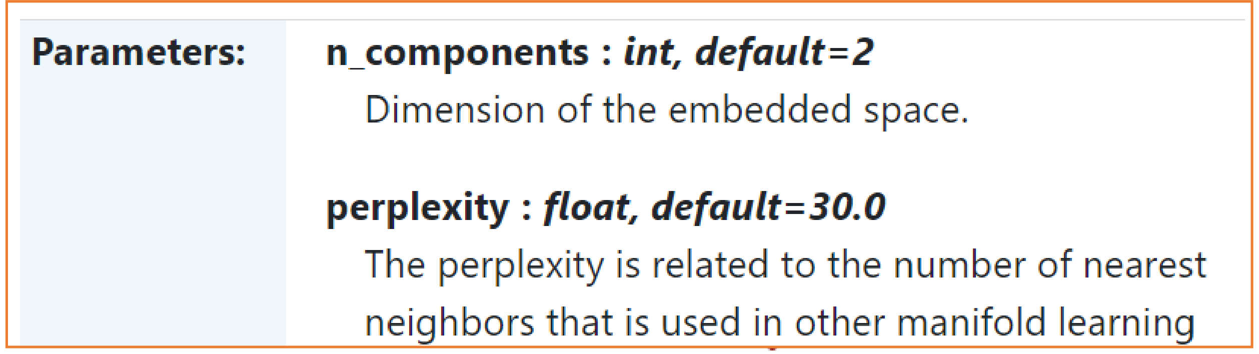
图片： https://huwang.blog.csdn.net/

T-SNE的使用

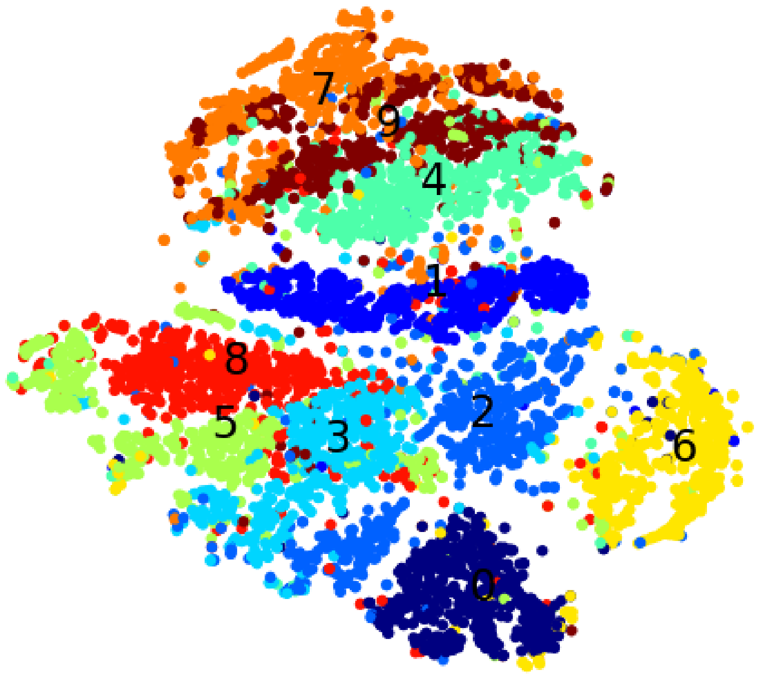


```
from sklearn.manifold import TSNE
tsne = TSNE(n_components=2)
tsne_results = tsne.fit_transform(X_std)

def scatter(x, colors):

    # We create a scatter plot.
    f = plt.figure(figsize=(8, 8))
    ax = plt.subplot(aspect='equal')
    sc = ax.scatter(x[:,0], x[:,1], lw=0, s=40,
                    c = Target.values, cmap='jet',)
    plt.xlim(-100, 100)
    plt.ylim(-100, 100)
    ax.axis('off')
    ax.axis('tight')

    # We add the labels for each digit.
    txts = []
    for i in range(10):
        # Position of each label.
        xtext, ytext = np.median(x[colors == i, :], axis=0)
        txt = ax.text(xtext, ytext, str(i), fontsize=24)
        txts.append(txt)
```

It is highly recommended to use another dimensionality reduction method (e.g. PCA for dense data or TruncatedSVD for sparse data) to reduce the number of dimensions to a reasonable amount (e.g. 50) if the number of features is very high. This will suppress some noise



Parameters:	**n_components** : ***int, default*=*2*** Dimension of the embedded space. **perplexity** : ***float, default*=*30.0*** The perplexity is related to the number of nearest neighbors that is used in other manifold learning



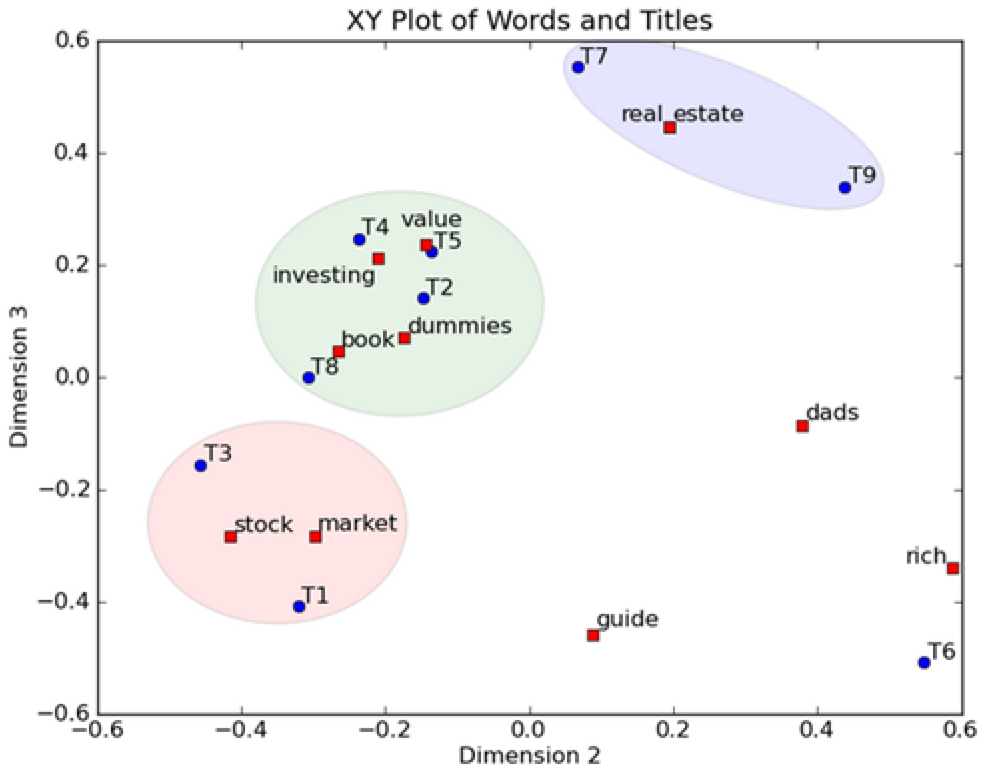
词义的空间嵌入（word embedding）（一）One-Hot 向量空间模型

- 最基本的方法
- 每个单词赋予一个向量，向量维度为当前词库大小，只有当前单词对应维度上为1，其余为0

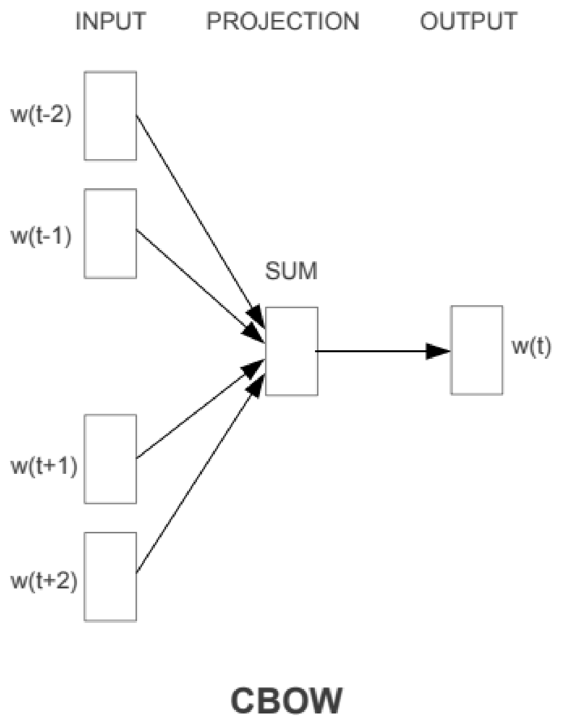
- 例：
- 北京大学: [1,0,0,0,…]
- 清华大学: [0,1,0,0,…]

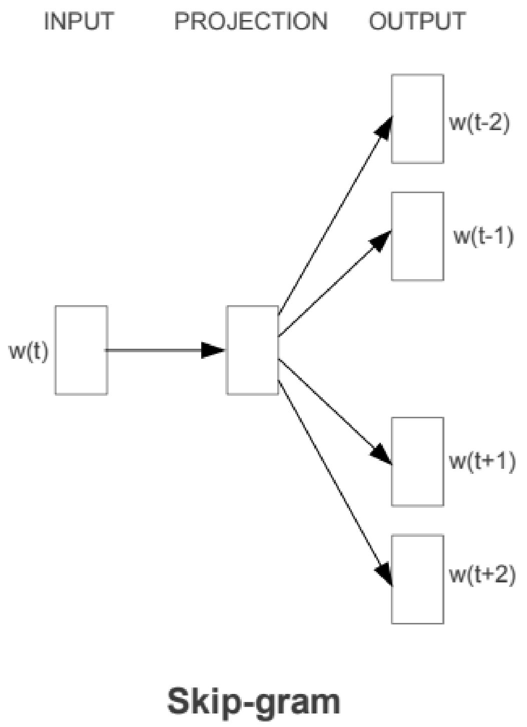
词义-文档的隐含语义空间（LSI）（二）词义分布与SVD

- 基于大规模语料中 词义-文档 分布或 词-词 分布
- 最小化对原矩阵的全局最小二乘损失（MSE）



文本的向量表示
（三）词向量(Word2Vec)





https://arxiv.org/pdf/1301.3781.pdf

文本的向量表示
（三）词向量(Word2Vec-CBOW)

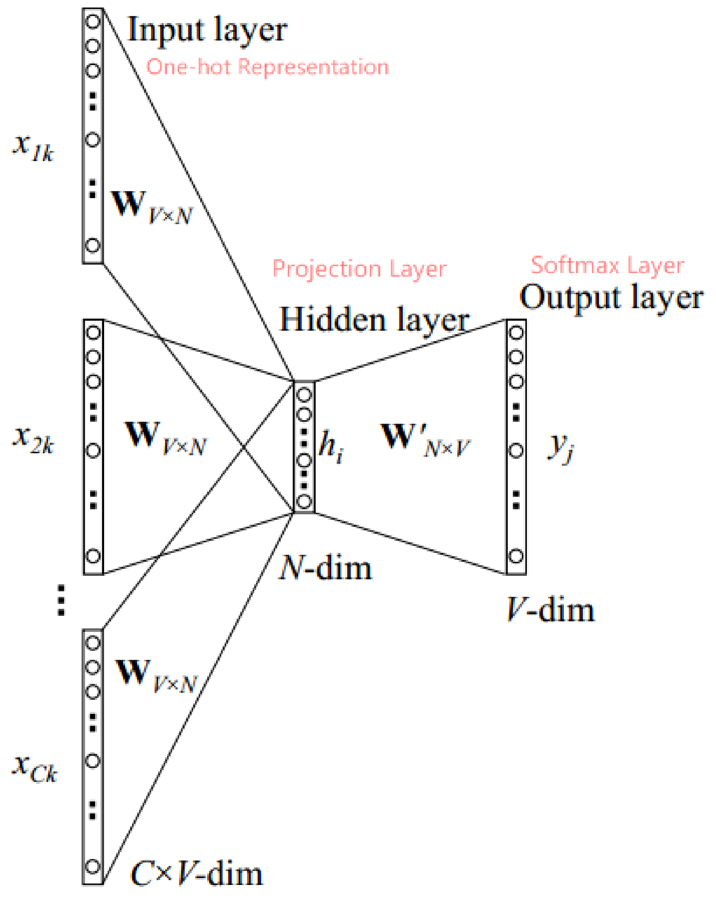


Figure 2: Continuous bag-of-word model

- CBOW通过上下文预测中心词概率
- Skip-gram模型则通过中心词预测上下文的概率

CBOW：

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,j\neq 0}\log p(w_{t+j}|w_t)$$

- 词汇总数V，目标词向量维度N，上下文参考词个数C
- X_i 第i个词的one-hot向量，通过词向量矩阵W得到v_i，计算平均值：

$$h_t = \frac{1}{C} * W^T \sum_{i=1}^{C} x_i = \frac{1}{C} * \sum_{i=1}^{C} v_i$$

- 最后映射回原本词汇维度，softmax计算预测词语概率概率

文本的向量表示（三）词向量-训练与使用

- Gensim 库：

```
>>> from gensim.models import Word2Vec
>>> sentences = [["cat", "say", "meow"],
["dog", "say", "woof"]]
>>> model = Word2Vec(sentences, min_count=1)
>>> vector  =  model.wv["cat"]
```

- 参考： https://radimrehurek.com/gensim/models/word2vec.html

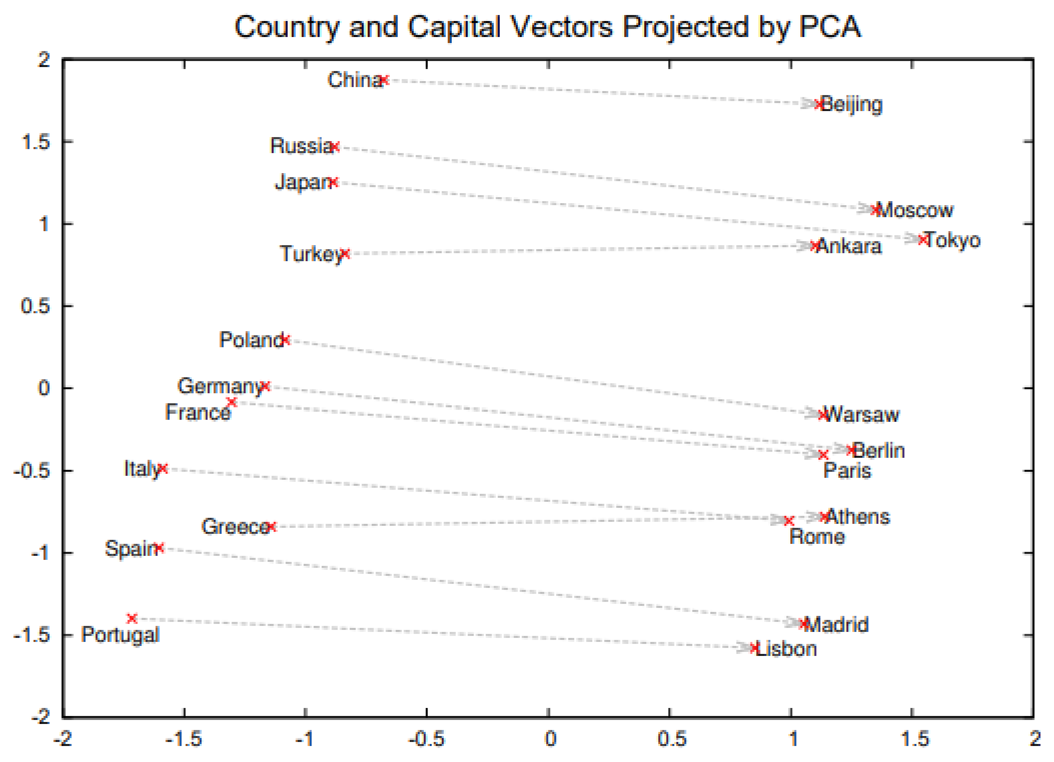
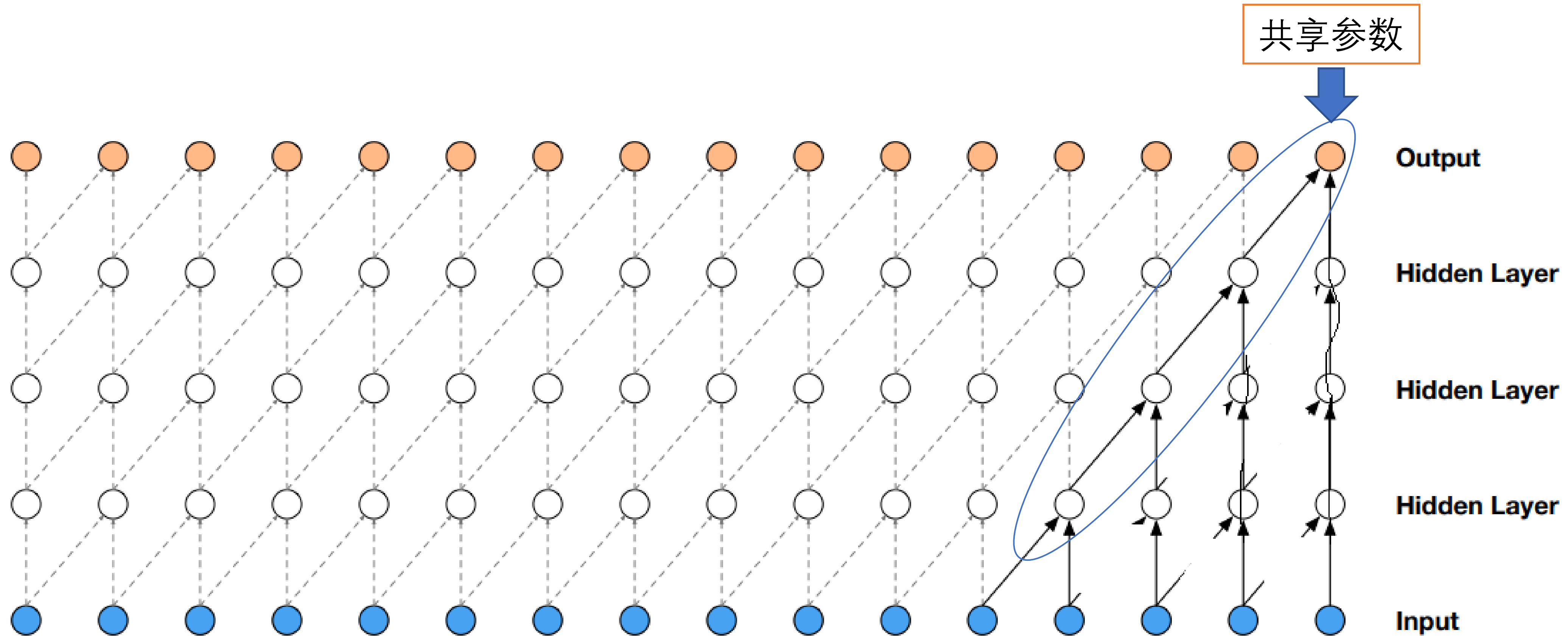


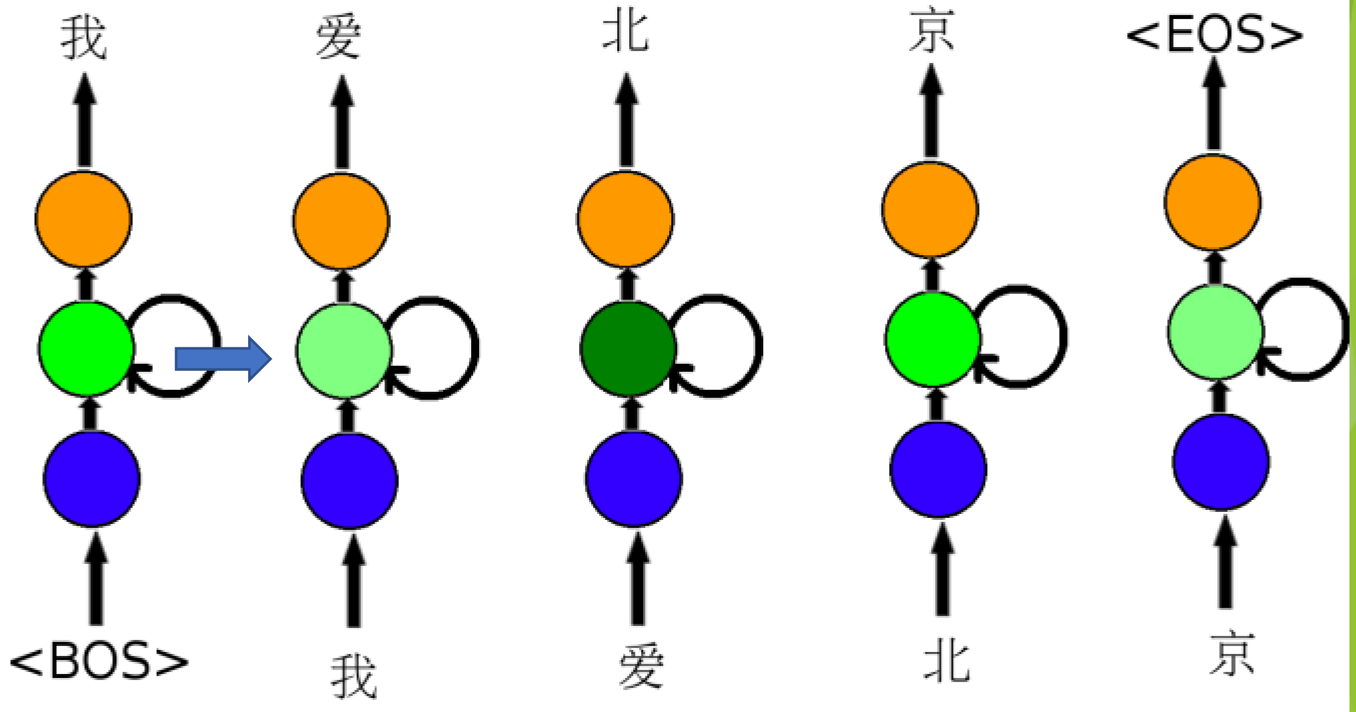
Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised information about what a capital city means.

时序数据处理与递归神经网络



后续的输入直接与前面层的输出进行计算，形成了递归神经网络架构。

- Recurrent Neural Network 循环神经网络
- 用法
  - 序列映射



带上下文记忆机制的时序网络

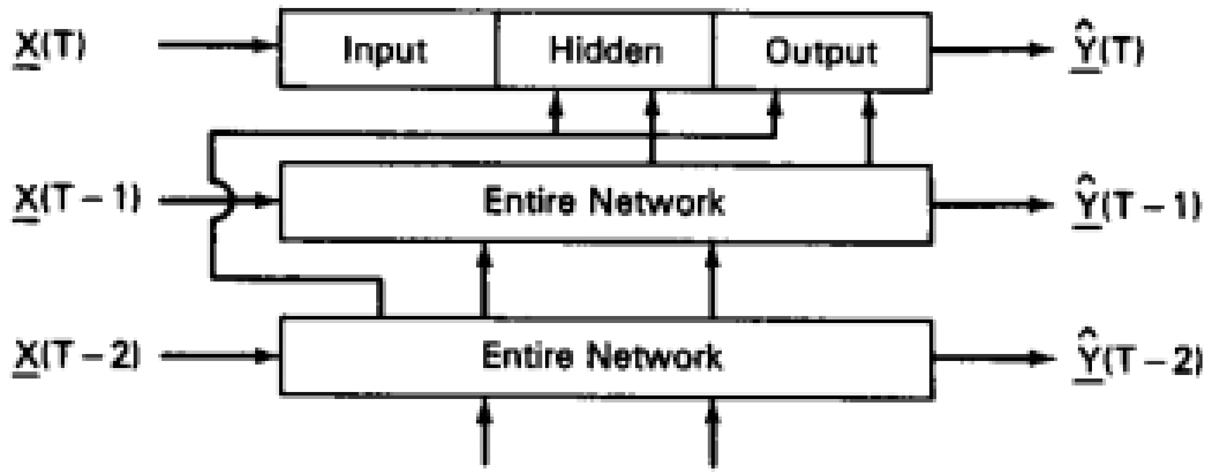


Fig. 5. Generalized network design with time lags.

Backpropagation Through Time: What It Does and How to Do It（1991）

简介

▶ 常见LSTM结构

$$\bar{\mathbf{z}}^t = \mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1}$$

$$\mathbf{z}^t = g(\bar{\mathbf{z}}^t) \qquad \textit{block input}$$

$$\bar{\mathbf{i}}^t = \mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1}$$

$$\mathbf{i}^t = \sigma(\bar{\mathbf{i}}^t) \qquad \textit{input gate}$$

$$\bar{\mathbf{f}}^t = \mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1}$$

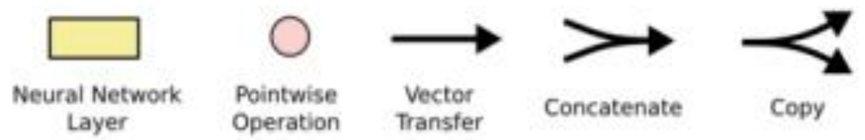
$$\mathbf{f}^t = \sigma(\bar{\mathbf{f}}^t) \qquad \textit{forget gate}$$

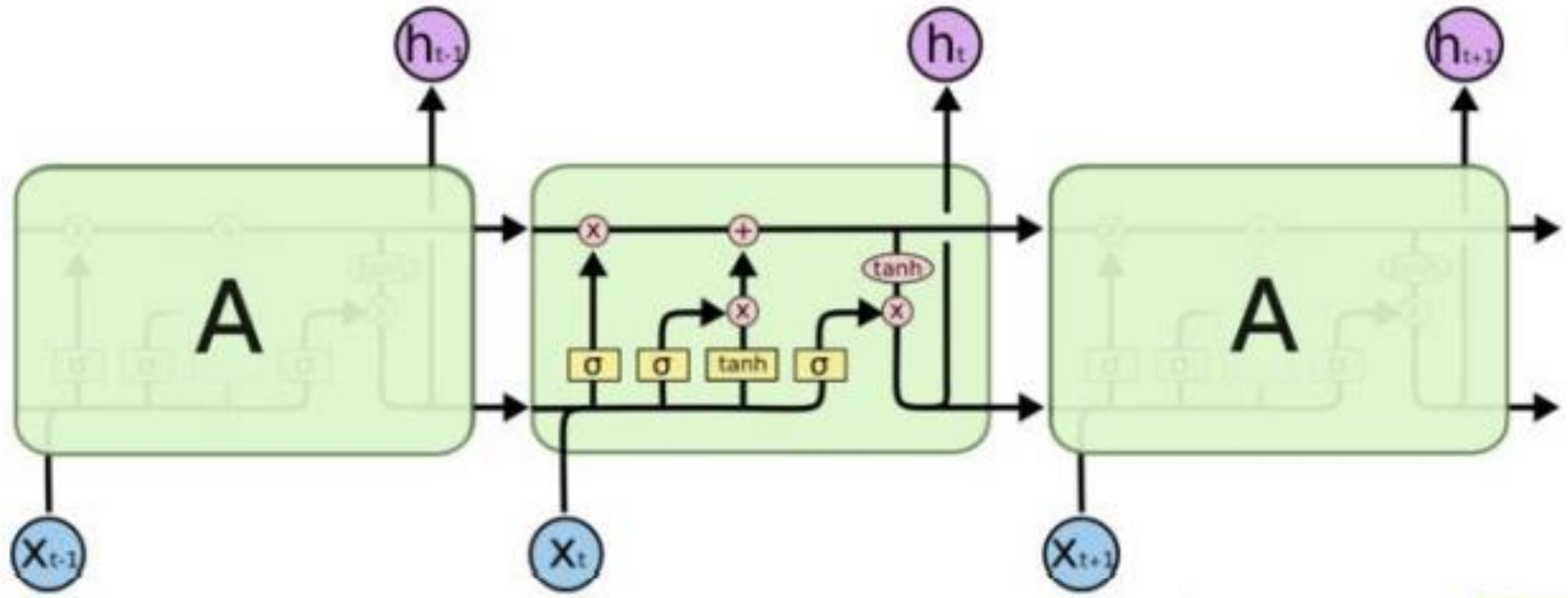
$$\mathbf{c}^t = \mathbf{z}^t \odot \mathbf{i}^t + \mathbf{c}^{t-1} \odot \mathbf{f}^t \qquad \textit{cell}$$

$$\bar{\mathbf{o}}^t = \mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1}$$

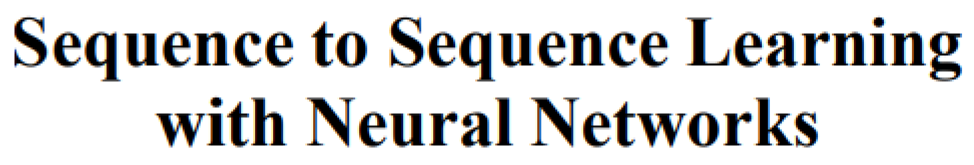
$$\mathbf{o}^t = \sigma(\bar{\mathbf{o}}^t) \qquad \textit{output gate}$$

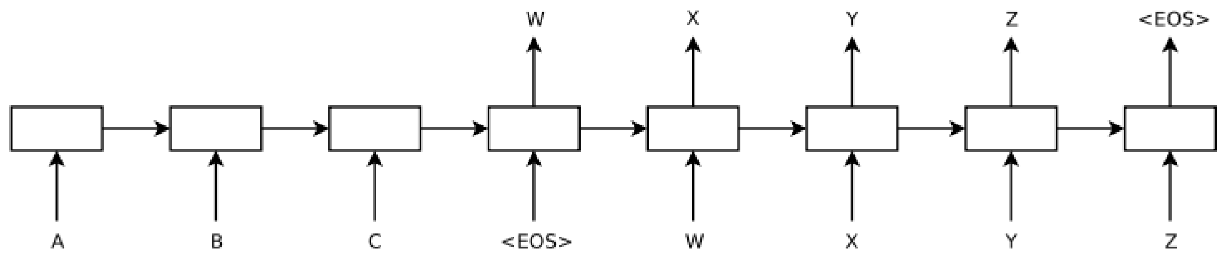
$$\mathbf{y}^t = h(\mathbf{c}^t) \odot \mathbf{o}^t \qquad \textit{block output}$$

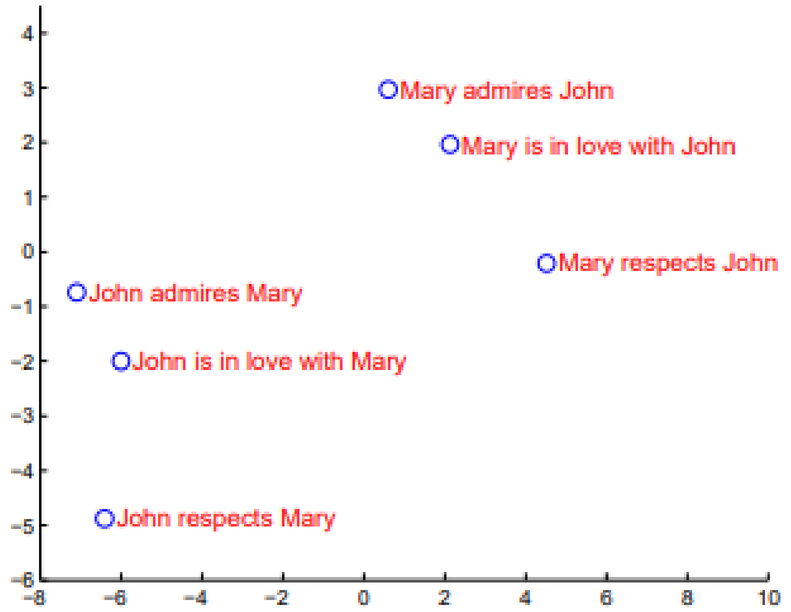


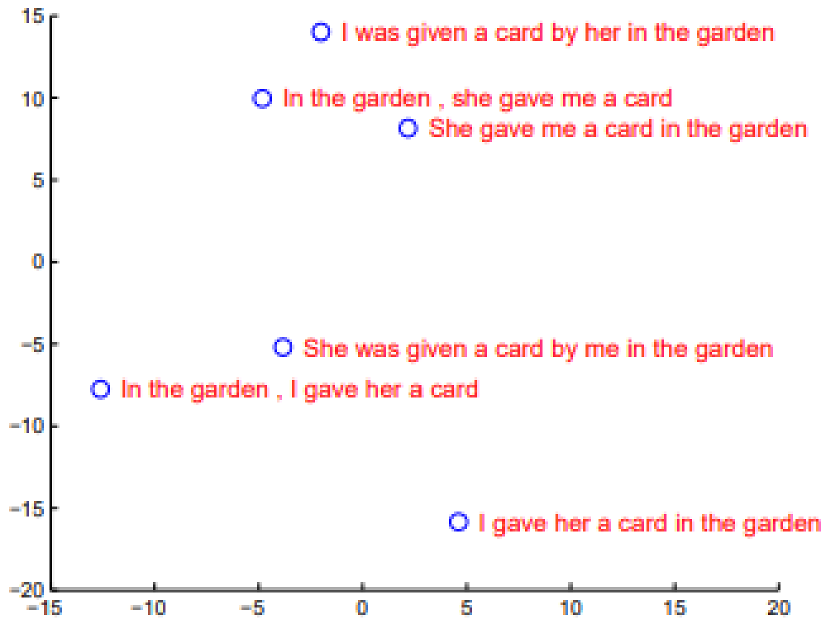


文本的向量表示 seq2seq learning


Sequence to Sequence Learning with Neural Networks







Attention机制：生成模型与判定模型

Attention Mechanism

- Bahdanau Attention
  - 用上一步的隐状态s_i询问得到context，直接让context成为RNNCell输入的一部分。
  - alignments的计算：

$$e_{ij} = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j\right),$$

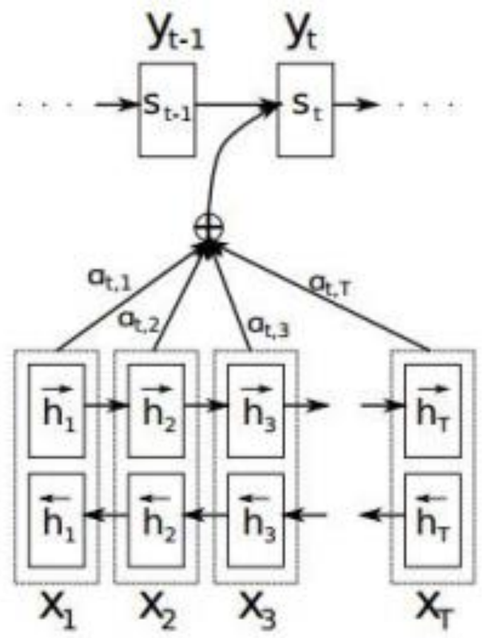


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

具体实现中的变化：

- 加入教师监督的训练过程
- Beam-search与loss

作业：