Topical Word Embedding

AAAI Conference on Artificial Intelligence, 2015

Yang Liu[1] Zhiyuan Liu[1] Tat-Seng Chuan[2] Maosong Sun[1,3]

[1]Department of Computer Science and Technology
State Key Lab on Intelligent Technology and Systems
National Lab for Information Science and Technology
Tsinghua University
Beijing 100084, China

[2]School of Computing, National University of Singapore
Singapore

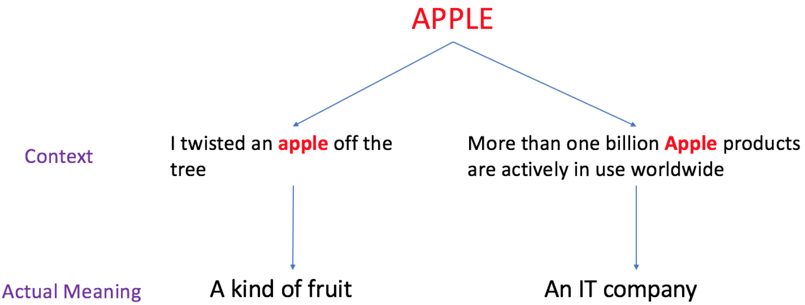
[3]Jiangsu Collaborative Innovation Center for Language Competence
Jiangsu 221009, China

Presented by Zejun, Lin

Plan

1. Introduction
   - Motivation
   - Related Work
2. Model
   - Topical Word Embeddings
3. Experiments
   - Contextual Word Similarity
   - Text Classification
   - Characteristics of TWE
4. Discussion

Motivation



- Most existing word embedding models :
  - Represent a word with a **single** vector
  - Indiscriminative for ubiquitous **homonymy** and **polysemy**

Motivation

- It is **problematic** that one word owns a **unique** vector for tasks like **Text Classification**
- Because many words have multiple senses
- To conceive a model that can **discriminate** word senses and generate multi-embeddings for each word

Related Work

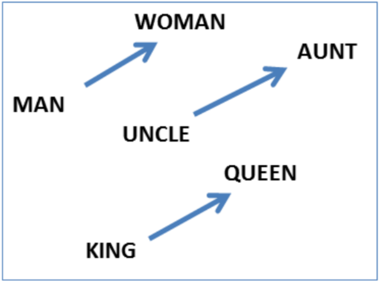
- Skip-Gram model
- Multi-prototype vector model

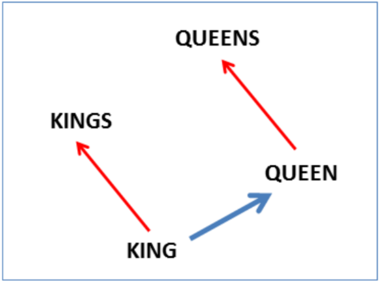
Skip-Gram

Main idea :

- A well-known framework for learning word vectors
- Represent each word as a low-dimensional, dense vector via its context words
- If two words co-occur more frequently, then their word vectors are more similar, which is estimated by the cosine similarity of their word vectors

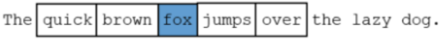
Linguistic Regularities in Skip-Gram

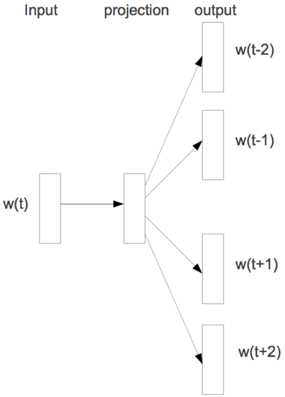




- Examples like – The famous "King - Man + Woman = Queen"

Skip-Gram Architecture





- Aim to predict context words given a target word in a sliding window

Multi-prototype Vector Model

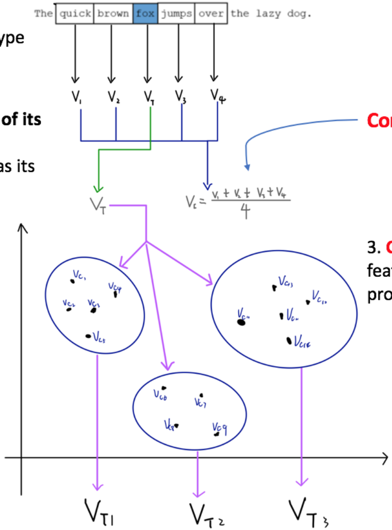
- **Cluster contexts** of a word into groups
- Generate a **distinct** prototype vector for each cluster

Multi-prototype Vector Model

1. Train single-prototype word vector

2. For **each word** with **one of its contexts**, average its neighboring word vectors as its **context feature vector**

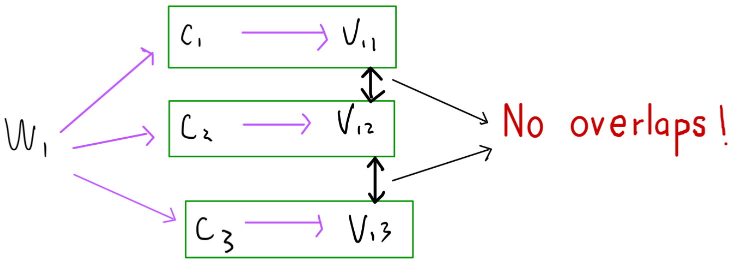
Context Feature Vector



3. **Cluster** its context feature vectors into several prototypes(classes)

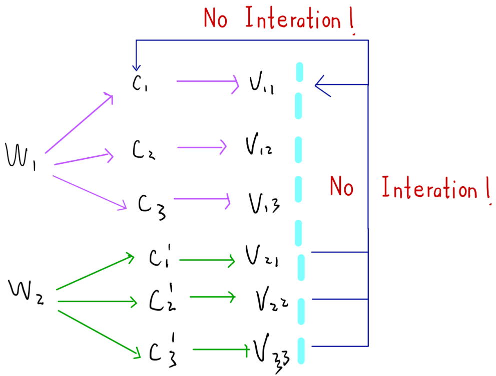
4. For each class, train one independent embedding for this word isolately

Disadvantages of Multi-prototype Vector Model



- Contexts of a word are divided into clusters with **no overlaps**
  - A word's several senses may **correlate** with each other – No clear semantic boundary
  - Like "**Play** the guitar" and "**Play** basketball"

Disadvantages of Multi-prototype Vector Model



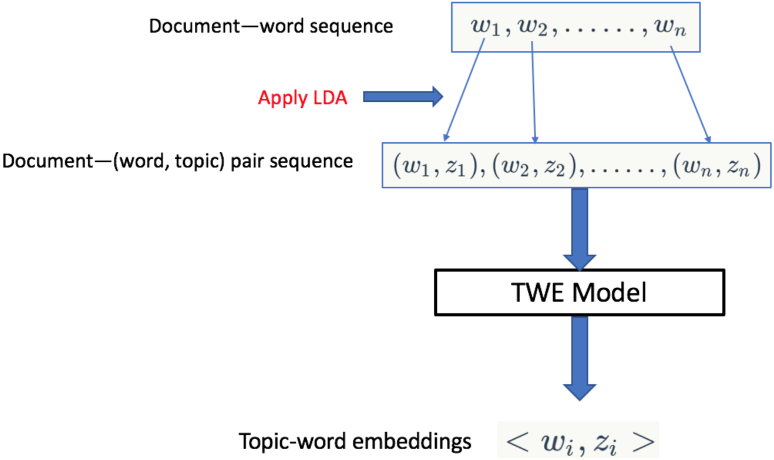
- Generate multi-prototype vectors for each word in **isolation**
  - Ignore complicated correlations among words & their contexts

Topical Word Embeddings

- Basic idea is to allow each word to have **different embeddings under different topics**
- An example – for (word, topic) pairs :
  - (apple, food) indicates a fruit while (apple, IT) indicates an IT company

Topical Word Embeddings

Main process :



Skip-Gram

- Given a sequence D = $w_1, ..., w_M$, maximize the average log probability (objective function)

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=1}^{M} \sum_{-k \leq c \leq k, c \neq 0} \log Pr(w_{i+c} | w_i)$$

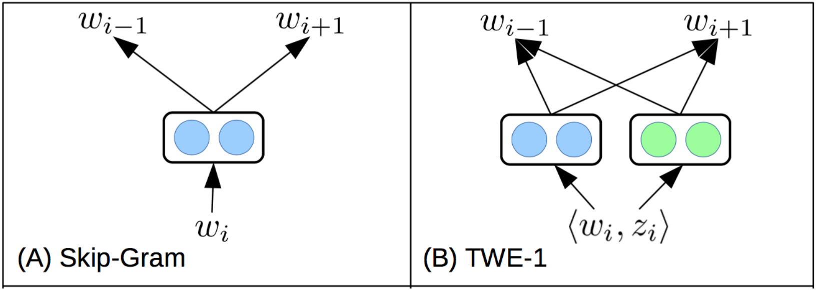
- Where the probability function is actually a softmax function as follows

$$Pr(w_c | w_i) = \frac{e^{\vec{w}_c \cdot \vec{w}_i}}{\sum_{w_i \in W} e^{\vec{w}_c \cdot \vec{w}_i}}$$

Three TWE Models

- TWE-1
- TWE-2
- TWE-3

TWE-1



- Regard each topic as a **pseudo word**
- Learn topic embeddings and word embeddings **separately**
- **Concatenate** them to build a topical word embedding

TWE-1

- Regard each topic as a pseudo word
- Objective function :

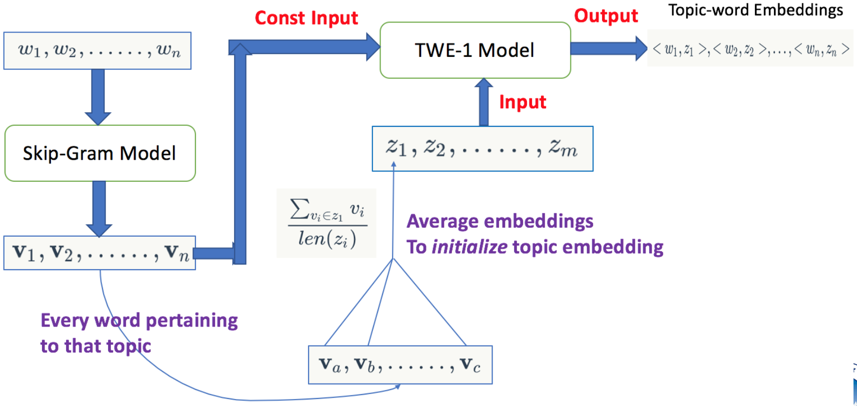
$$\mathcal{L}(D) = \frac{1}{M}\sum_{i=1}^{M}\sum_{-k\leq c\leq k, c\neq 0}\log Pr(w_{i+c}|w_i) + \log Pr(w_{i+c}|z_i)$$

- Concatenate the embedding of w & z to get the topical word embedding :

$$\vec{w}^z = \vec{w} \oplus \vec{z}$$

TWE-1

Main Process :

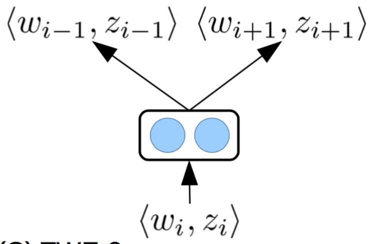


TWE-1

Disadvantage

- Not consider the **immediate interaction** between a word and its assigned topic for learning

TWE-2



(C) TWE-2

- Consider each word-pair <w_i, z_i> as a **pseudo word**
- Learn topical-word embeddings directly
- For word **apple** under topics **food** and **IT**, we consider <**apple**, **food**> is **a word** and <**apple**, **IT**> is **another word**

TWE-2

- Objective Function :

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=1}^{M} \sum_{-k \leq c \leq k, c \neq 0} \log Pr(< w_{i+c}, z_{i+c} > \mid < w_i, z_i >)$$

- Where Pr is also a softmax function

$$Pr(w_c|w_i) = \frac{e^{\vec{w}_c^{z_c} \cdot \vec{w}_i^{z_i}}}{\sum_{<w_c, z_c> \in <W, T>} e^{\vec{w}_c^{z_c} \cdot \vec{w}_i^{z_i}}}$$

TWE-2

Main Process :

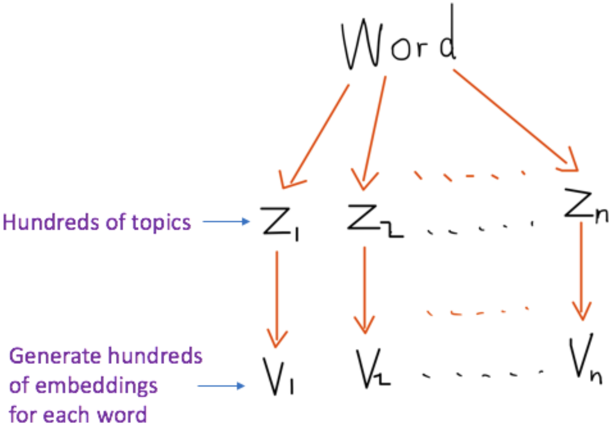
- Initialize each topic-word pair using Skip-Gram
- Learn TWE models

TWE-2

Disadvantage

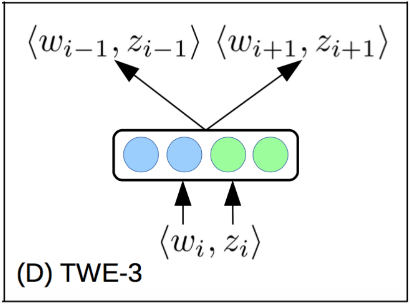
- Although it considers the inner interaction of a word-topic pair,
- It suffers from the **sparsity issue**

TWE-2



- A word is actually **separated** into hundreds of parts

TWE-3



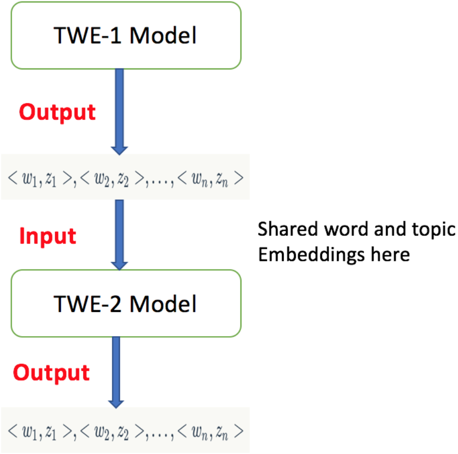
- Concatenate the vector $\vec{w}$ and $\vec{z}$ to build $\vec{w}^z$, with

$$|\vec{w}^z| = |\vec{w}| + |\vec{z}|$$

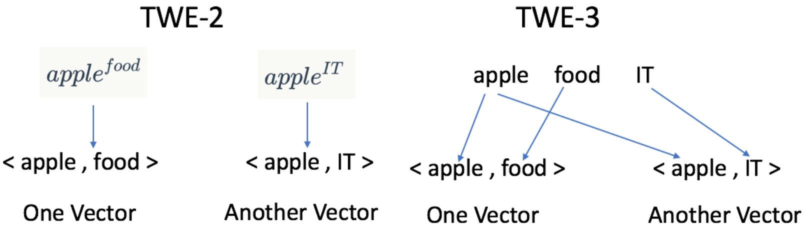
- The objective of TWE-3 and probability function is identical to TWE-2, as shown above

TWE-3

Main Process :



TWE-3



ATTENTION

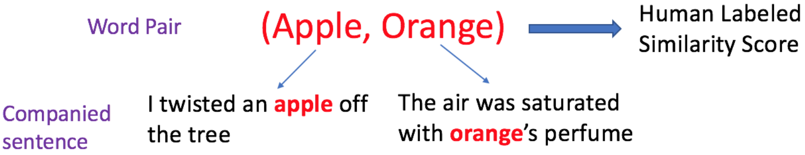
- In TWE-3, the parameter of each word embedding $\vec{w}$ and topic embedding $\vec{z}$ is **shared** over all word-topic <w, z>
- Example : In TWE-2, two word-topic pair <w, z> and <w, z'> will have **distinct** parameters, while in TWE-3 they share the **same** word embedding

TWE-3

Disadvantage

- Although TWE-3 provide a trade-off between discrimination and sparsity
- It makes those words in the same topic **less discriminative,** because during the learning process, topic embeddings will influence the corresponding word embeddings

Contextual Word Similarity



Dataset :

- Our Dataset – SCWS
  - contain 2003 pairs of words and sentences containing these words
  - human labeled word similarity scores based on the word meaning in the context

Method :

- Compute **Spearman Correlation** between the human labeled result and the result from the model

Contextual Word Embedding

- For each word with its context c, infer the topic distribution by

$$Pr(z|w,c) \propto Pr(w|z)Pr(z|c)$$

- Further obtain the contextual word embedding as

$$\vec{w}^c = \sum_{z_i \in T} Pr(z_i|w,c)\vec{w}^z$$

Contextual Word Embedding

- AvgSimC method :

$$S(w_i, c_i, w_j, c_j) = \vec{w}_i^{c_i} \cdot \vec{w}_j^{c_j}$$
$$= \sum_{z \in T} \sum_{z' \in T} Pr(z|w_i, c_i) Pr(z'|w_j, c_j) S(\vec{w}^z, \vec{w}^{z'})$$

- MaxSimC method :
  - $\vec{w}^c = \vec{w}^z, z = arg\max_z Pr(z|w, c)$
  - $S(w_i, c_i, w_j, c_j) = \vec{w}_i^z \cdot \vec{w}_j^{z'}$

Contextual Word Similarity

Compared Methods :

- Single-prototype model
    - C&W (Collobert and Weston 2008)
    - TFIDF & Pruned TFIDF
    - LDA-S
    - LDA-C
    - Skip-Gram
- Multi-prototype model
    - Pruned TFIDF-M
    - Huang's model (Huang et al. 2012)
    - Tian's model (Tian et al. 2014)

Contextual Word Similarity

Table 2: Spearman correlation $\rho \times 100$ of contextual word similarity on the SCWS data set.

Model	$\rho \times 100$	
C&W	57.0	
TFIDF	26.3	
Pruned TFIDF	62.5	
LDA-S	56.9	
LDA-C	50.4	
Skip-Gram	65.7	
	AvgSimC	MaxSimC
Pruned TFIDF-M	60.5	60.4
Tian	65.4	63.6
Huang	65.3	58.6
TWE-1	**68.1**	**67.3**
TWE-2	67.9	63.6
TWE-3	67.1	65.5

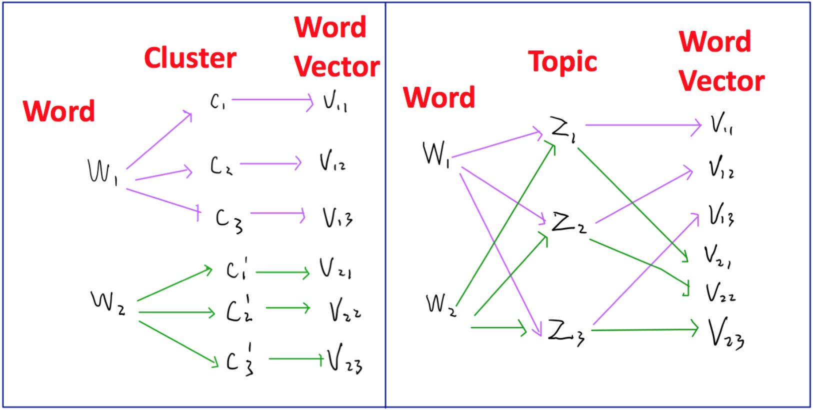
Analysis

TWE performs better than other multi-prototype models

- More topics than clusters, more details of word semantics (Topics=400, cluster=8)
- Most multi-prototype models build multi-prototypes for each word **separately**, ignoring rich **interactions** between words as well as their contexts
- LDA provides a more **principle** way to select the most **appropriate** topical word embedding for a word under specific context

Analysis

Multi-prototype Model **TWE Model**



Text Classification

Data Set :

- 20NewsGroup
  - 20, 000 documents from 20 different newsgroup

Method :

- Use Liblinear Classification
- Report macro-averaging precision, recall and F-measure for comparison

Document Embedding

- Represent the semantics of a document by aggregating over all topical word embeddings :

$$d = \sum_{w \in d} Pr(w|d)\vec{w}^z$$

- Where Pr(w|d) can be weighted with TFIDF scores of words in d

Text Classification

Table 3: Evaluation results of multi-class text classification.

Model	Accuracy	Precision	Recall	F-measure
BOW	79.7	79.5	79.0	79.0
LDA	72.2	70.8	70.7	70.0
Skip-Gram	75.4	75.1	74.3	74.2
PV-DM	72.4	72.1	71.5	71.5
PV-DBOW	75.4	74.9	74.3	74.3
TWE-1	**81.5**	**81.2**	**80.6**	**80.6**
TWE-2	79.0	78.6	77.9	77.9
TWE-3	77.4	77.2	76.2	76.1

Analysis

TWE-1 achieves the best performance

- The reason may be the **independent assumption** in TWE-1
- The size of 20NewsGroup is to some extent small
  - TWE-2 and TWE-3 may achieve better performance given more data for training

Characteristics of TWE

Table 4: Nearest neighbor words by TWE-2 and Skip-Gram.

Words	Similar Words
bank	citibank, investment, river
bank#1	insurance, stock, investor
bank#2	river, edge, coast
left	right, leave, quit
left#1	moved, arrived, leave
left#2	right, bottom, hand
apple	macintosh, ios, juice
apple#1	peach, juice, strawberry
apple#2	mac, ipod, android

- Find the most similar words of several example words in different topics

Discussion

1. Explore **non-parametric** topic models
   - In LDA, the topic number must be pre-defined
   - Use CV to find the topic number but are **time-consuming** and **impractical** for large-scale data
2. Is there any other **better embedding model** utilizing **topics**
   - The three TWE models somewhat have some disadvantages
   - And the last two models did not perform better as expected

Question ?

Thank you !