Predictive Text Embedding through Large-scale Heterogeneous Text Networks

Jian Tang[1] Meng Qu[2] Qiaozhu Mei[3]

[1]Microsoft Research Asia
jiatang@microsoft.com

[2]Peking University
mnqu@pku.edu.cn

[3]University of Michigan
qmei@umich.edu

Made by Zejun Lin, 2017

Plan

Introduction

To learn a meaningful and effective **representation of text**, there are two kinds of methods :

1. Unsupervised method :
    - Skip-gram
    - Paragraph vector
2. Supervised method :
    - CNN

Unsupervised Method

Advantages :

1. Simple
2. Scalable
3. Effective
4. Easy to tune and accommodate unlabeled data

Unsupervised method

PROBLEM

It yield inferior results (Compared to sophisticated deep learning architectures like CNN)

Reasons :

1. Learn in a unsupervised way
2. Not leverage the labeled information
3. Embeddings learned are not particularly tuned for any task

Supervised Method

- Deep neural networks like CNN & RNN
- Disadvantages :
  - Computational
  - Need large amount of labeled examples
  - Exhaustive tuning of many parameters

Predictive Text Embedding

Characteristics :

- Semi-supervised
- Utilize both labeled & unlabeled data

Predictive text embedding

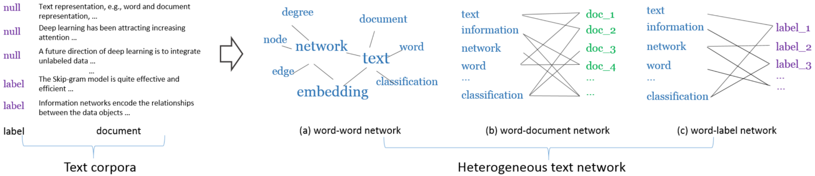
Process :

1. Represent the text as **a heterogeneous text network**
2. Embed the network into **a low dimensional space** using an algorithm

Characteristics of the network :

1. Preserve the semantic closeness of words & documents
2. Have a strong predictive power for the particular task

Bipartite Text Network

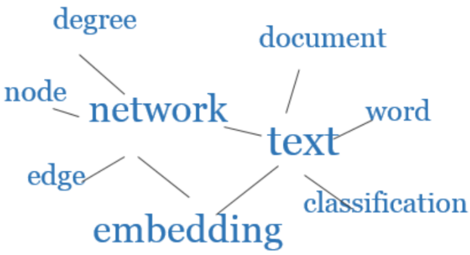


Three types of *bipartite networks*

1. Word-Word Network
2. Word-Document Network
3. Word-Label Network

Heterogeneous text network is the combination of the above networks

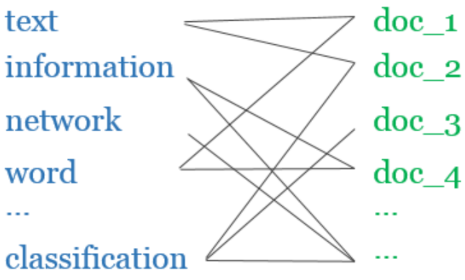
Word-Word Network



(a) word-word network

- **Vertex :** A word
- **Edge :** The two words co-occur in a context window
- **Value of Edge :** Number of times of co-occurrence

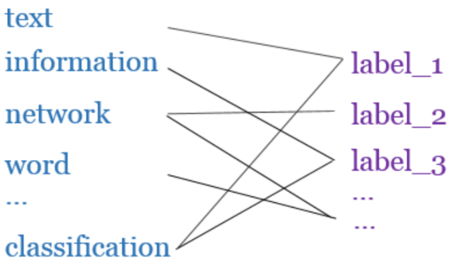
Word-Document Network



(b) word-document network

- **Vertex :** A word or a document
- **Edge :** The word occur in the document
- **Value of Edge :** Number of times of occurrence

Word-Label Network



(c) word-label network

- **Vertex :** A word or a label
- **Edge :** The word occur in the label
- **Value of Edge :** Number of times of occurrence

Bipartite Network Embedding

- Given a bipartite network $G = (\mathcal{V}_\mathcal{A} \bigcup \mathcal{V}_\mathcal{B}, E)$,
- The **probability** of $v_i(v_i \in \mathcal{V}_\mathcal{A})$ **generated** by $v_j(v_j \in \mathcal{V}_\mathcal{B})$ is the softmax result of their cos similarity :

$$p(v_i|v_j) = \frac{e^{\vec{u_i}^T \cdot \vec{u_j}}}{\sum_{i' \in A} e^{\vec{u_{i'}}^T \cdot \vec{u_j}}}$$

Concepts in Network Embedding

First-order proximity

- Captured by the **observed links** in the networks
- Most existing algorithms like IsoMap, preserve it
- Many **legitimate links** are actually **not observed** in a real-world network.

Second-order proximity

- Determined through the **shared neighborhood** structures of the vertices
- That is, nodes sharing similar neighbors should be similar

Concepts in Network Embedding

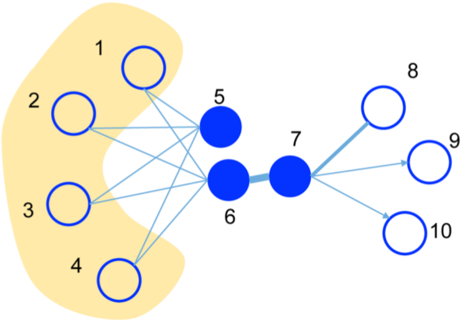


Figure 1: A toy example of information network. Edges can be undirected, directed, and/or weighted. Vertex 6 and 7 should be placed closely in the low-dimensional space as they are connected through a strong tie. Vertex 5 and 6 should also be placed closely as they share similar neighbors.

Bipartite Network Embedding

Coming back here...

- So, here we get a vertex's conditional distribution $p(\cdot|v_j)$
- To preserve second-order proximity, make it close to its empirical distribution :

$$O = \sum_{j \in B} \lambda_j d(\hat{p}(\cdot|v_j), p(\cdot|v_j))$$

- Where,
  - λ_j – importance of the vertex – estimated by the **degree**
  - $\hat{p}(v_i|v_j)$ – estimated by $\frac{w_{ij}}{deg_j}$
- Finally, omitting constants, we get :

$$O = - \sum_{(i,j) \in E} w_{ij} \log p(v_j|v_i)$$

Approach

- Optimized with **SGD**, using **edge sampling** & **negative sampling**
- Steps :
  - Sample a edge according to its proportion of weight
  - Sample K negative edges from a noise distribution $p_n(j)$
  - Optimize it

Heterogeneous Text Network Embedding

- Combine the three bipartite networks, we get a **heterogeneous text network**
- It encodes word co-occurrences at different levels :
  - Local Context Level
  - Document Level
  - Label Level

Objective Function

- An intuitive objective function is combining the three objective function :

$$O_{pte} = O_{ww} + O_{wd} + O_{wl}$$

Two Ways for Training

1. **Joint Training**
   - Use labeled & Unlabeled data simultaneously
2. **Pre-training + Fine-tuning**
   - Learn the embeddings with unlabeled data first
   - Fine-tune the embeddings with word-label network

Attention

Due to the **incomparability** of edges from different types of network, we **alternatively** sample from three sets of edges

Text Embedding

- Represent a piece of text $d = w_1, w_2, ..., w_n$ by averaging the word vectors :

$$\vec{d} = \frac{1}{n}\sum_{i=1}^{n} \vec{u}_i$$

- Which is actually the solution to minimize the **Euclidean distance** between words and texts

Data Sets

Table 1: Statistics of the Data Sets

	Long Documents							Short Documents		
Name	20NG	WIKI	IMDB	CORPORATE	ECONOMICS	GOVERNMENT	MARKET	DBLP	MR	TWITTER
Train	11,314	1,911,617*	25,000	245,650	77,242	138,990	132,040	61,479	7,108	800,000
Test	7,532	21,000	25,000	122,827	38,623	69,496	66,020	20,000	3,554	400,000
$\|V\|$	89,039	913,881	71,381	141,740	65,254	139,960	64,049	22,270	17,376	405,994
Doc. length	305.77	672.56	231.65	102.23	145.10	169.07	119.83	9.51	22.02	14.36
#classes	20	7	2	18	10	23	4	6	2	2

*In the WIKI data set, only 42,000 documents are labeled.

Data Sets

- Long Document Corpora :
  - 20NG – 20newsgroup
  - WIKI – a snapshot of Wikipedia corpus in April 2010
  - IMDB – a data set for sentiment classification
  - RCV1 – a large benchmark corpus for text classification
- Short Document Corpora :
  - DBLP – titles of papers from the computer science bibliography
  - MR – a movie review data set
  - TWITTER – a corpus of tweets for sentiment classification

Compared Algorithms

- BOW
- Skip-gram
- PVDBOW
- PVDM
- LINE
- CNN
- PTE
  - PTE(G_{wl}) –> word-label network only
  - PTE(pretrain)
  - PTE(joint)

Classification

Procedure :

- Representation of documents
- Classification process
    - use the one-vs-rest logistic regression model in the LibLinear package

Results on Long Documents

Table 2: Results of text classification on **long** documents.

		20NG		Wikipedia		IMDB	
Type	Algorithm	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Word	BOW	80.88	79.30	79.95	80.03	86.54	86.54
Unsupervised Embedding	Skip-gram	70.62	68.99	75.80	75.77	85.34	85.34
	PVDBOW	75.13	73.48	76.68	76.75	86.76	86.76
	PVDM	61.03	56.46	72.96	72.76	82.33	82.33
	LINE(G_{ww})	72.78	70.95	77.72	77.72	86.16	86.16
	LINE(G_{wd})	79.73	78.40	80.14	80.13	89.14	89.14
	LINE($G_{ww} + G_{wd}$)	78.74	77.39	79.91	79.94	89.07	89.07
Predictive Embedding	CNN	78.85	78.29	79.72	79.77	86.15	86.15
	CNN(pretrain)	80.15	79.43	79.25	79.32	89.00	89.00
	PTE(G_{wl})	82.70	81.97	79.00	79.02	85.98	85.98
	PTE($G_{ww} + G_{wl}$)	83.90	83.11	81.65	81.62	89.14	89.14
	PTE($G_{wd} + G_{wl}$)	**84.39**	**83.64**	82.29	82.27	89.76	89.76
	PTE(pretrain)	82.86	82.12	79.18	79.21	86.28	86.28
	PTE(joint)	84.20	83.39	**82.51**	**82.49**	**89.80**	**89.80**

Results on Long Documents

- PTE jointly trained with G_{wl} > unsupervised of PTE (e.g. G_{ww})
  - Power of supervision
- $PTE_{joint} > PTE_{G_{wl}}$
  - Power of unlabeled information
- $PTE_{joint} > PTE_{pretrain}$
- $PTE_{joint} > CNN$

Results on Long Documents

- $CNN_{pretrain} \geq CNN$
  - Useful to pretrain CNN with LINE's embedding
- $CNN_{pretrain} < PTE_{joint}$
  - CNN can only utilize the data separately
- $Time_{CNN} > 10 \cdot Time_{PTE_{joint}}$
- $Time_{CNN_{pretrain}} > 5 \cdot Time_{PTE_{joint}}$

Results on Long Documents – RCV1 data sets

Table 3: Results of text classification on **long** documents (RCV1 data sets).

	Corporate		Economics		Government		Market	
Algorithm	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
BOW	78.45	63.80	86.18	81.67	77.43	62.38	95.55	94.09
PVDBOW	65.87	45.78	79.63	74.82	70.74	54.08	91.81	88.88
LINE(G_{wd})	76.76	60.30	85.55	81.46	77.82	63.34	95.66	93.90
PTE(G_{wl})	76.69	60.48	84.88	80.02	78.26	63.69	95.58	93.84
PTE(pretrain)	77.03	61.03	84.95	80.63	78.48	64.50	95.54	93.79
PTE(joint)	**79.20**	**64.29**	**87.05**	**83.01**	**79.63**	**66.15**	**96.19**	**94.58**

Results on Short Documents

Table 4: Results of text classification on **short** documents.

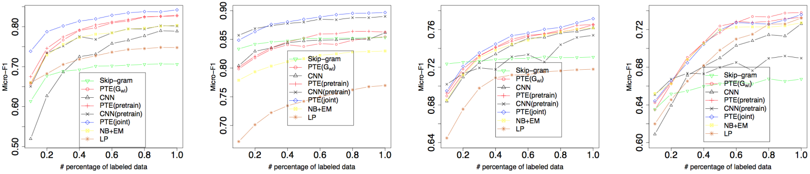
		DBLP		MR		Twitter	
Type	Algorithm	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Word	BOW	75.28	71.59	71.90	71.90	75.27	75.27
Unsupervised Embedding	Skip-gram	73.08	68.92	67.05	67.05	73.02	73.00
	PVDBOW	67.19	62.46	67.78	67.78	71.29	71.18
	PVDM	37.11	34.38	58.22	58.17	70.75	70.73
	LINE(G_{ww})	73.98	69.92	71.07	71.06	73.19	73.18
	LINE(G_{wd})	71.50	67.23	69.25	69.24	73.19	73.19
	LINE($G_{ww} + G_{wd}$)	74.22	70.12	71.13	71.12	73.84	73.84
Predictive Embedding	CNN	76.16	73.08	72.71	72.69	**75.97**	**75.96**
	CNN(pretrain)	75.39	72.28	68.96	68.87	75.92	75.92
	PTE(G_{wl})	76.45	72.74	73.44	73.42	73.92	73.91
	PTE($G_{ww} + G_{wl}$)	76.80	73.28	72.93	72.92	74.93	74.92
	PTE($G_{wd} + G_{wl}$)	**77.46**	**74.03**	73.13	73.11	75.61	75.61
	PTE(pretrain)	76.53	72.94	73.27	73.24	73.79	73.79
	PTE(joint)	77.15	73.61	**73.58**	**73.57**	75.21	75.21

Results on Short Documents

PET_{joint} not consistently outperform CNN :

- Word **sense ambiguity** (more serious on the short documents)
- CNN reduces it using **word orders** in local context

Effects of Labeled Data



(a) 20NG (b) IMDB (c) DBLP (d) MR

Figure 2: Performance w.r.t. # labeled data.

Effects of Unlabeled Data

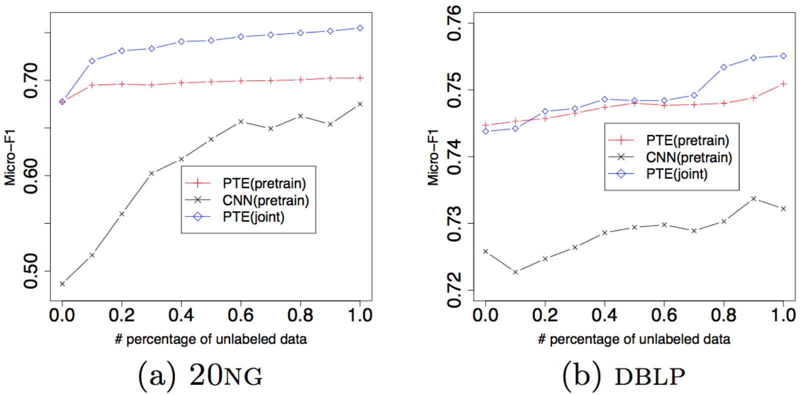


Figure 3: Performance w.r.t. # unlabeled data.

Conclusion

Unsupervised embeddings :

- For Long Document :
  - **Document-Level** word co-occurrence more useful than that of **Local-Context-Level**
  - Their combination **not** further improve the result
- For Short Document :
  - **Local-Context-Level** word co-occurrence more useful than that of **Document-Level**
  - Their combination further improve the result

Conclusion

Supervised embeddings :

- CNN handled labeled information more efficiently :
  - Especially on short document (Labeled data is sparse)
  - Because it has a more complicated structure to
    - Utilize word orders
    - Address word sense ambiguity
- PTE :
  - When labeled data is abundant, PTE is comparable or superior to CNN
  - Able to be jointly trained while CNN should be trained in an indirect way
  - Faster and easier to configure

Future Work

There is considerable room to improve PTE like :

- Considering the orders of words
- More reasonable way to represent document embeddings

Thank you !