Content to Node: Self-Translation Network Embedding

Jie Liu
Nankai University
Tianjin, China
jliu@nankai.edu.cn

Zhicheng He
Nankai University
Tianjin, China
hezhicheng@mail.nankai.edu.cn

Lai Wei
Nankai University
Tianjin, China
future@mail.nankai.edu.cn

Yalou Huang
Nankai University
Tianjin, China
ylhuang@nankai.edu.cn

KDD '18, August 19–23, 2018, London, United Kingdom
© 2018 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery.
ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3219988

Motivation

1.Real-world networks ————rich content information,

Previous NE methods tend to learn separated content and structure representations for each node.

The empirical and simple combination strategies often make the final vector suboptimal.

2.Existing NE methods———the structure information

Considering short and fixed neighborhood scope, such as the first- and/or the second- order proximities.

Decide the scope of the neighborhood when facing a complex problem.

$G = (V, E)$

v^i is the identity of the vertex v,

v^c is the content associated with v.

Each edge $e_{u,v} \in E$ represents the relation between two vertices (u,v).

Definition 2: Parallel Sequences. Let $S = \{v_1, v_2, \cdots, v_T\}$ be a sequence of vertices sampled from a network using random walk, the vertex identity sequence $S^i = \{v_1^i, v_2^i, \cdots, v_T^i\}$ and the corresponding content sequence $S^c = \{v_1^c, v_2^c, \cdots, v_T^c\}$ are a pair of parallel sequences.

Definition 3: Content-to-node Self-translation. Given a set of parallel sequences $S = \{(S_n^i, S_n^c)_1^N\}$, content-to-node self-translation is to learn a mapping function $f_\theta : S_n^c \mapsto S_n^i$ for each $S_n \in S$.

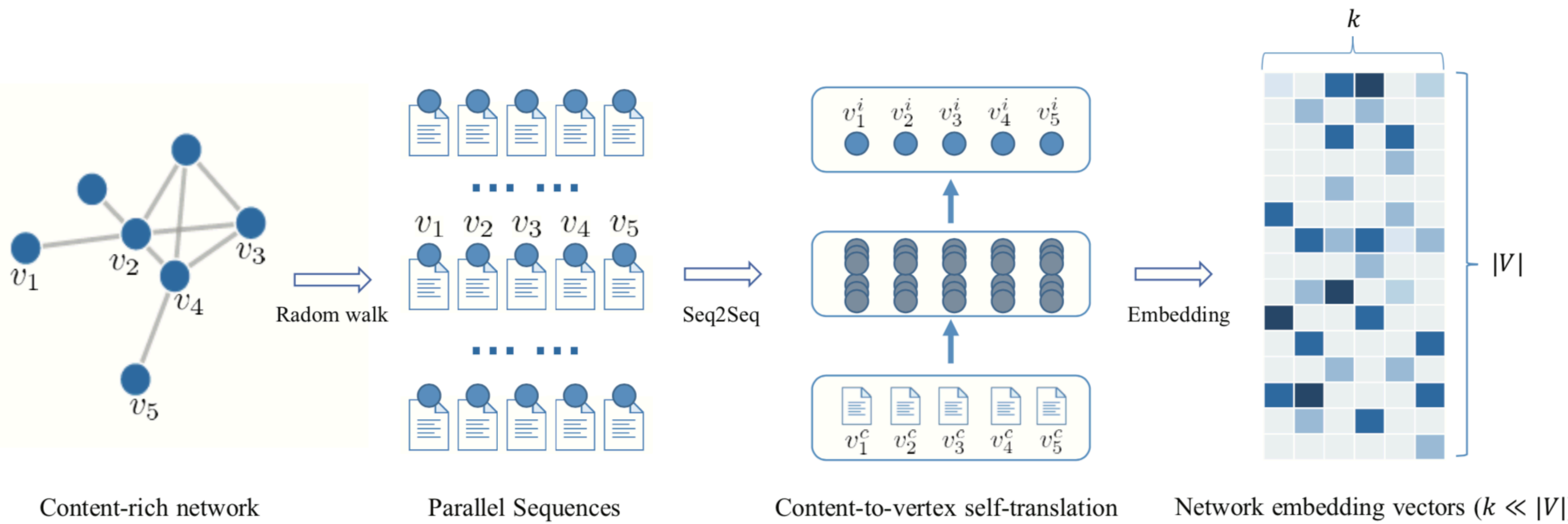


Figure 1: The framework of Self-Translation Network Embedding.

Content Embedding

v_t^c is the raw content of node vt

preprocessed into a vector

$$\mathbf{v}_t^c = \mathrm{Emb}(v_t^c).$$

Emb()————fully connected layer, convolution layer, etc.

Content Sequence Encoder

$$\mathbf{i}_t = \sigma(\mathbf{W}_{vi}\mathbf{v}_t^c + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i), \tag{4}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{vf}\mathbf{v}_t^c + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f), \tag{5}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{vo}\mathbf{v}_t^c + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_{t-1} + \mathbf{b}_o), \tag{6}$$

$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \tanh(\mathbf{W}_{vc}\mathbf{v}_t^c + \mathbf{W}_{hc}h_{t-1+\mathbf{b}_c}), \tag{7}$$

$$\mathbf{h}_t = \mathbf{o}_t \otimes \tanh(\mathbf{c}_t), \tag{8}$$

ct is the cell memory vector

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

To model both the forward and backward context information along random walks, Adopt a bi-directional LSTM (Bi-LSTM) encoder layer:

$$\overrightarrow{\mathbf{h}_t} = \mathcal{H}^{fw}(v_t^c, \overrightarrow{\mathbf{h}_{t-1}}), \qquad \overleftarrow{\mathbf{h}_t} = \mathcal{H}^{bw}(v_t^c, \overleftarrow{\mathbf{h}_{t+1}}).$$

Q(·) function concatenates the last hidden state vectors of the forward and backward LSTM:

$$\mathbf{w} = Q(\{\overrightarrow{\mathbf{h}_1}, \ldots, \overrightarrow{\mathbf{h}_T}, \overleftarrow{\mathbf{h}_1}, \ldots, \overleftarrow{\mathbf{h}_T}\}) = [\overrightarrow{\mathbf{h}_T};\ \overleftarrow{\mathbf{h}_1}].$$

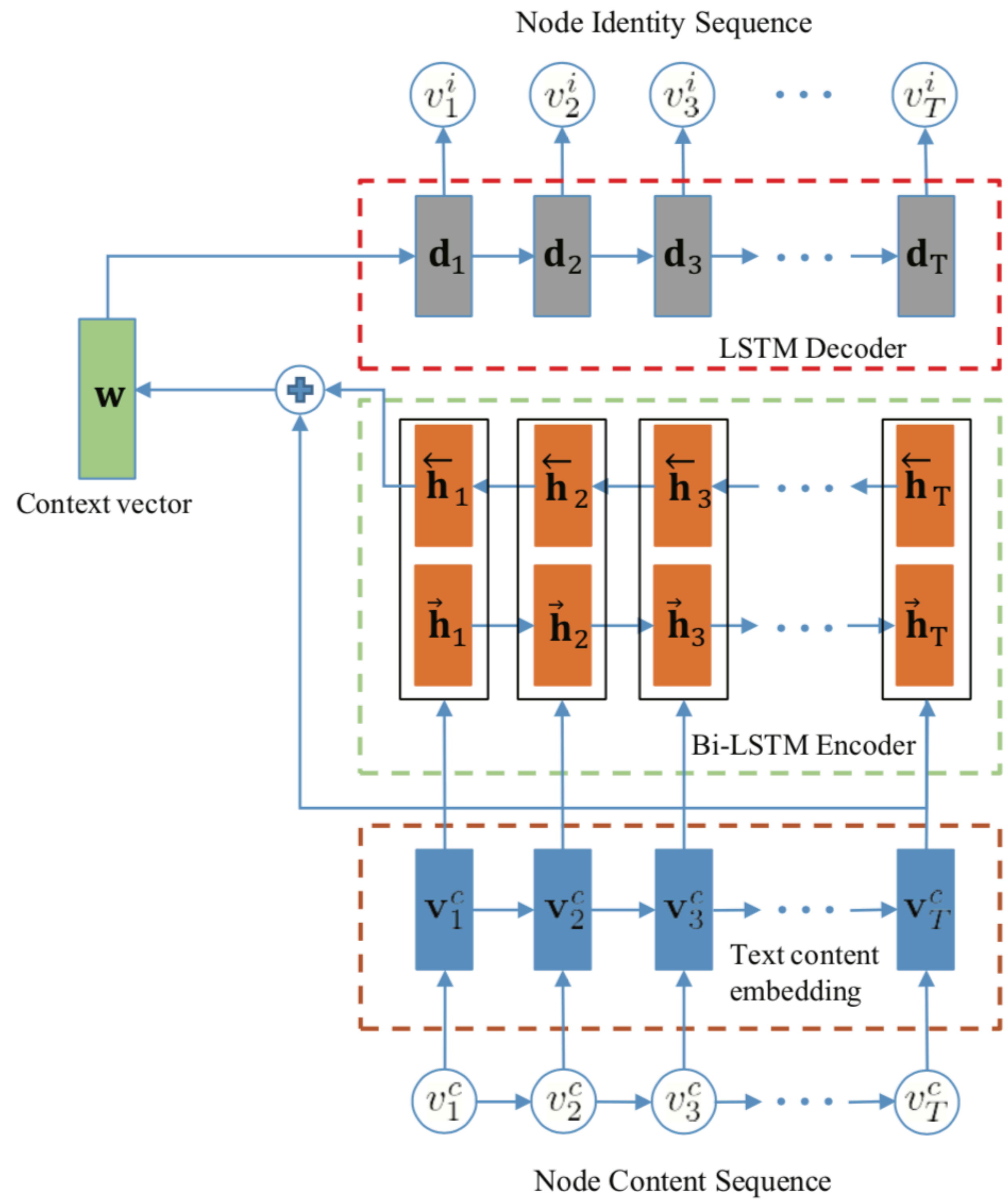
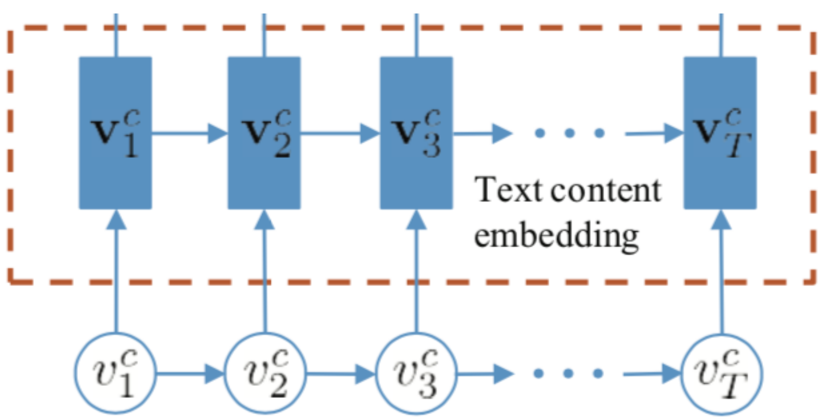
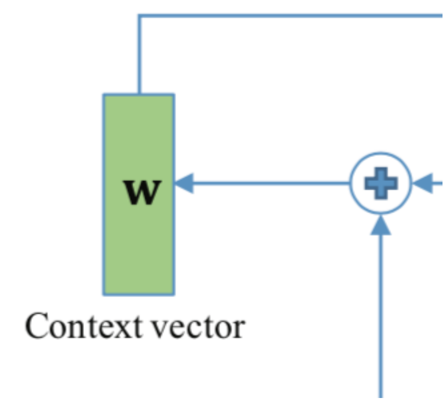


Figure 2: STNE for network embedding.

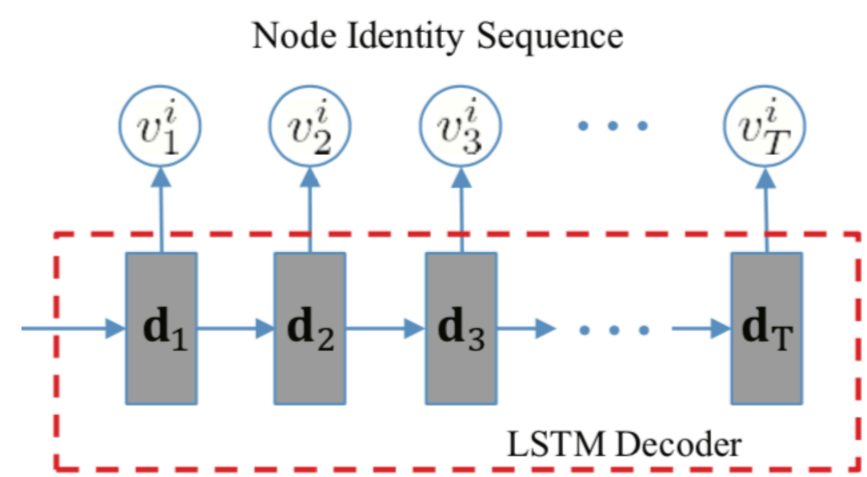
Node Sequence Generation

content sequence S_n^c has been compressed into the context vector representation w.





LSTM decoder function D(·, ·) to generate D:



$$\mathbf{d}_t = \mathcal{D}(\mathbf{w},\ \mathbf{d}_{t-1}) = \begin{cases} \mathcal{H}(\mathbf{0},\ \mathbf{w}) & t = 1 \\ \mathcal{H}(\mathbf{0},\ \mathbf{d}_{t-1}) & t > 1 \end{cases},$$

$D = \{d1, d2, \ldots, d_T\}$

$$\mathbf{g}_t = \sigma(\mathbf{W}_{fc}\mathbf{d}_t + \mathbf{b}_{fc}).$$

Optimization

A softmax layer transforms g_t into the probabilities,

$$p_t(j) = \mathrm{softmax}(\mathbf{g}_t)_j = \frac{\exp(\mathbf{g}_t(j))}{\sum_{j'} \exp(\mathbf{g}_t(j'))}.$$

a cross-entropy loss is adopted to measure the correctness of the translation,

$$L = -\sum_{n=1}^{N} \sum_{v_t \in S_n} \sum_{j}^{|V|} \delta(v_t^i, j) p_t(j),$$

$\delta(\cdot, \cdot)$ is a binary function that outputs 1 if v_t^i equals j, otherwise 0.

To make the predicted identity sequence continuous, the predicted node v_t^i should be an eighbor of the previous node v_{t-1}^i,

$$L_t = -\sum_{n=1}^{N} \sum_{v_t \in S_n} \sum_{j \in N(v_{t-1}^i)} \delta(v_t^i, j) p_t(j),$$

Node Embedding

one node appears in multiple sequences and has multiple hidden representations.

capture different semantic aspects of a node when interacting with different neighbors.

Suppose that node v_i appears $|v_i|$ times in different sequences,

$$\mathbf{h}(v_i) = \frac{1}{|v_i|} \sum_{j=1}^{|v_i|} [\overrightarrow{\mathbf{h}_j(v_i)}; \overleftarrow{\mathbf{h}_j(v_i)}].$$

EXPERIMENTS

Table 1: Statistics of Datasets.

Datasets	Cora	Citeseer	Wiki
# Nodes	2708	3312	2405
# Edges	5429	4732	17981
Edge Density	0.074%	0.043%	31.1%
# Words	1433	3703	4973
# Avg. Words / Doc.	18	32	640
# Labels	7	6	17
Max. class size	818	701	406
Min. class size	180	248	9
Avg. class size	387	552	141

Table 3: F1-score on Cora dataset with the percentage of labeled nodes varies from 10% to 50%.

% Labeled Nodes	10%	20%	30%	40%	50%
DeepWalk	76.4	78.0	79.5	80.5	81.0
MMDW	74.9	80.8	82.8	83.7	84.7
SVD	58.3	67.4	71.1	73.3	74.0
PLSA	57.0	63.1	65.1	66.6	67.6
Naive Combination	76.5	80.4	82.3	83.3	84.1
NetPLSA	80.2	83.0	84.0	84.9	85.4
TADW	82.4	85.0	85.6	86.0	86.7
STNE	**84.2**	**86.5**	**87.0**	**86.9**	**88.2**

Table 4: F1-score on Citeseer dataset with the percentage of labeled nodes varies from 10% to 50%.

% Labeled Nodes	10%	20%	30%	40%	50%
DeepWalk	52.4	54.7	56.0	56.5	57.3
MMDW	55.6	60.1	63.2	65.1	66.9
SVD	58.3	66.4	69.2	71.2	72.2
PLSA	54.1	58.3	60.9	62.1	62.6
Naive Combination	61.0	66.7	69.1	70.8	72.0
NetPLSA	58.7	61.6	63.3	64.0	64.7
TADW	**70.6**	**71.9**	**73.3**	73.7	74.2
STNE	69.6	71.2	72.2	**74.3**	**74.8**

Table 5: F1-score on Wiki dataset with the percentage of labeled nodes varies from 10% to 50%.

% Labeled Nodes	10%	20%	30%	40%	50%
DeepWalk	59.3	64.3	66.2	68.1	68.8
MMDW	57.8	62.3	65.8	67.3	67.3
SVD	65.1	72.9	75.6	77.1	77.4
PLSA	69.0	72.5	74.7	75.5	76.0
Naive Combination	66.3	73.0	75.2	77.1	78.6
NetPLSA	67.2	70.6	71.7	71.9	72.3
TADW	72.6	77.3	79.2	79.9	80.3
STNE	**73.9**	**78.0**	**80.6**	**81.5**	**82.7**