# Encoding Structural Texts

Yan Song

# Outline

- Context2vec
- TreeLSTM
- TransE
- Node2vec
- Hyperdoc2vec

# Look Back

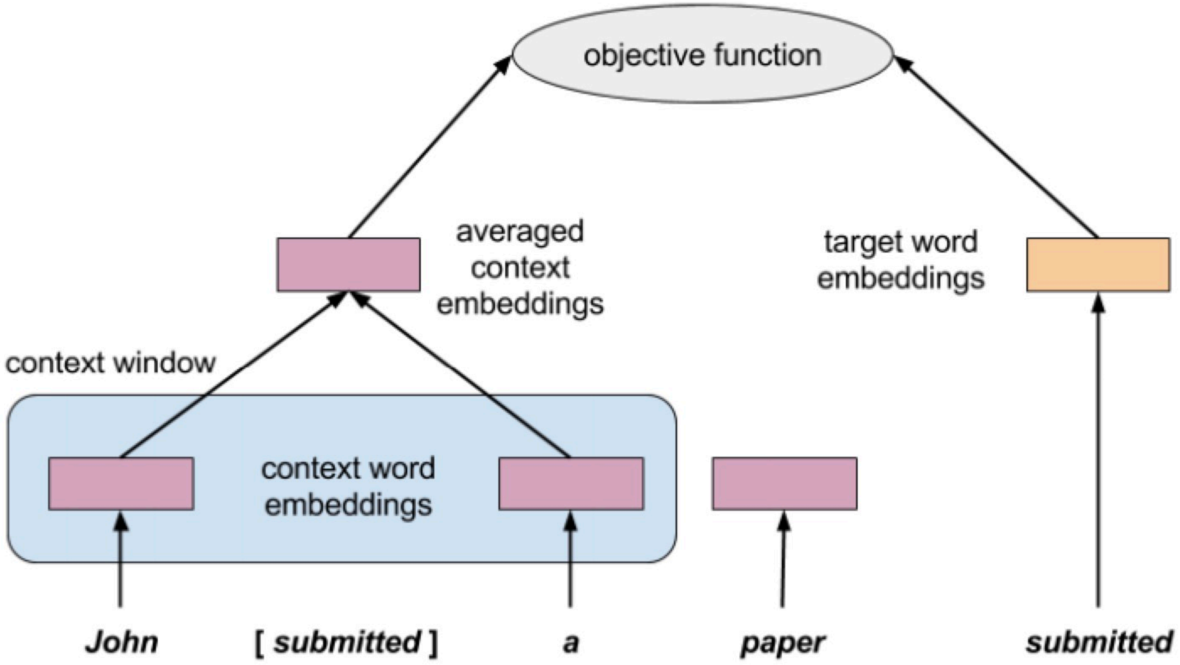
- So far the embeddings are learned from unstructured texts
- Weak signals are leveraged
    - Word context window
    - Sentence sequences
- Structural models are always exploited
    - Such as LSTM, GRU
- Data determines model performance
    - Esp. supervised encoder

# Model? Data?

- Learning from plain text is not the only way out
  - For many cases, data themselves are structured
  - Anything can be leveraged from conventional embeddings?
- Maybe we don’t need a complex model, but a better way to handle data structures
  - Sentence-word relations
  - Knowledge concepts and relations
  - Hyper-documents

# Context2vec

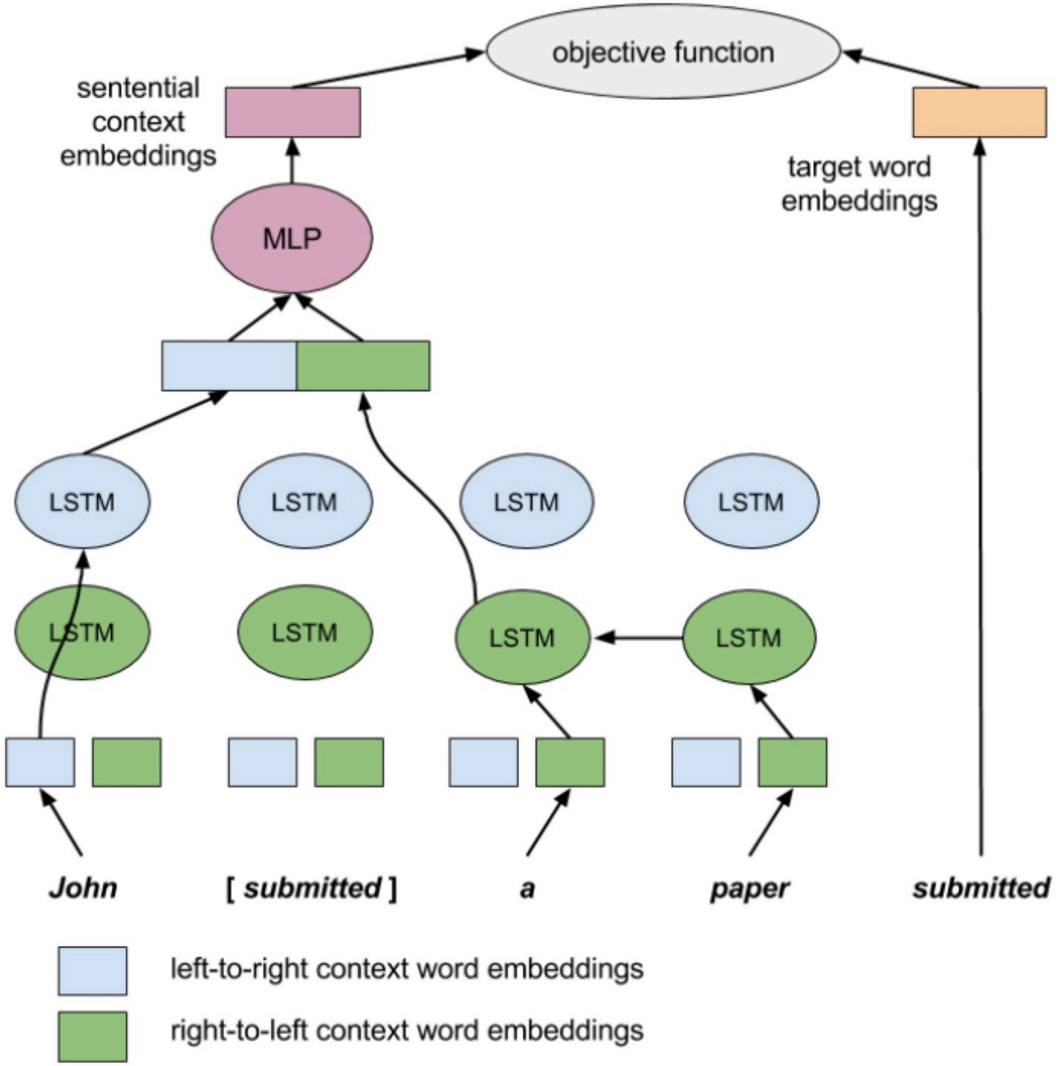
- Look at the CBOW model



(a) word2vec *CBOW*

# Context2vec

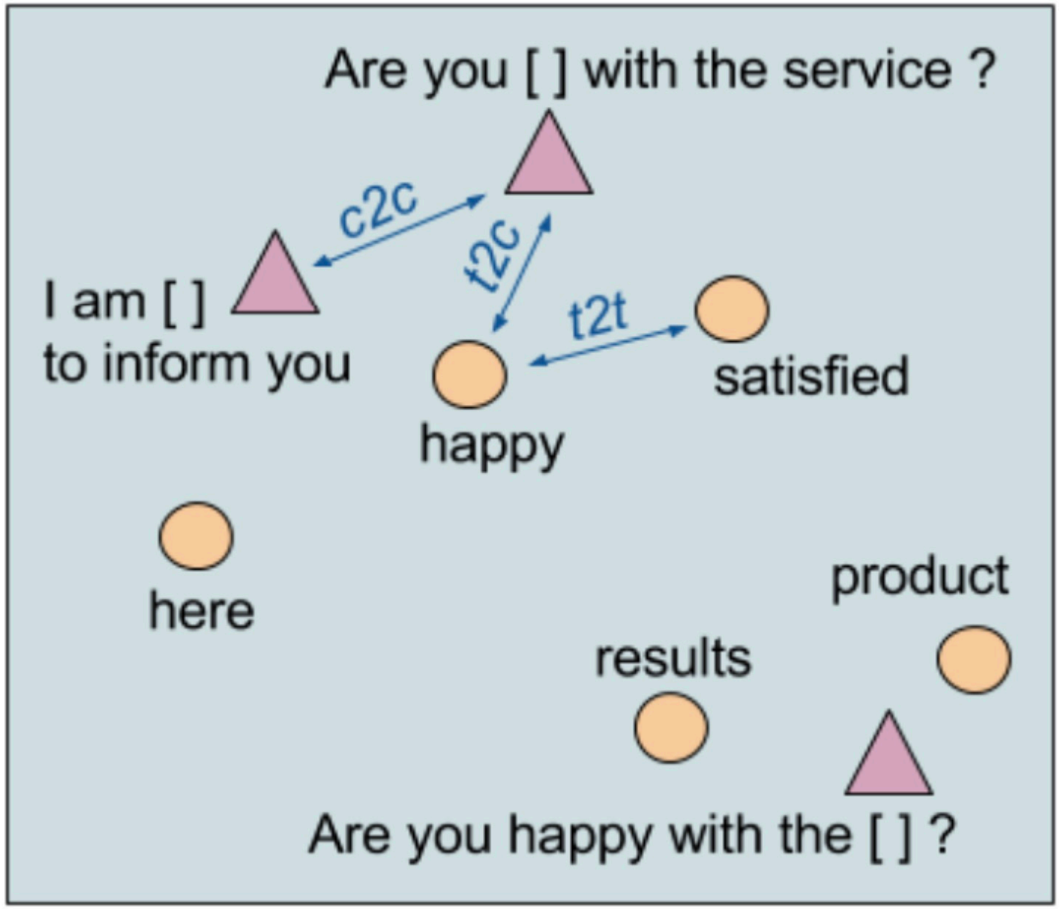
- One can add structure into context
- A simple but effective adaptation of CBOW
- Which implies that word embeddings can be learned with sentences



(b) *context2vec*

# Context2vec

- It is interesting to investigate the sentential embeddings and word embeddings learned by context2vec
- In this way, sentence and word can be represented in one vector space

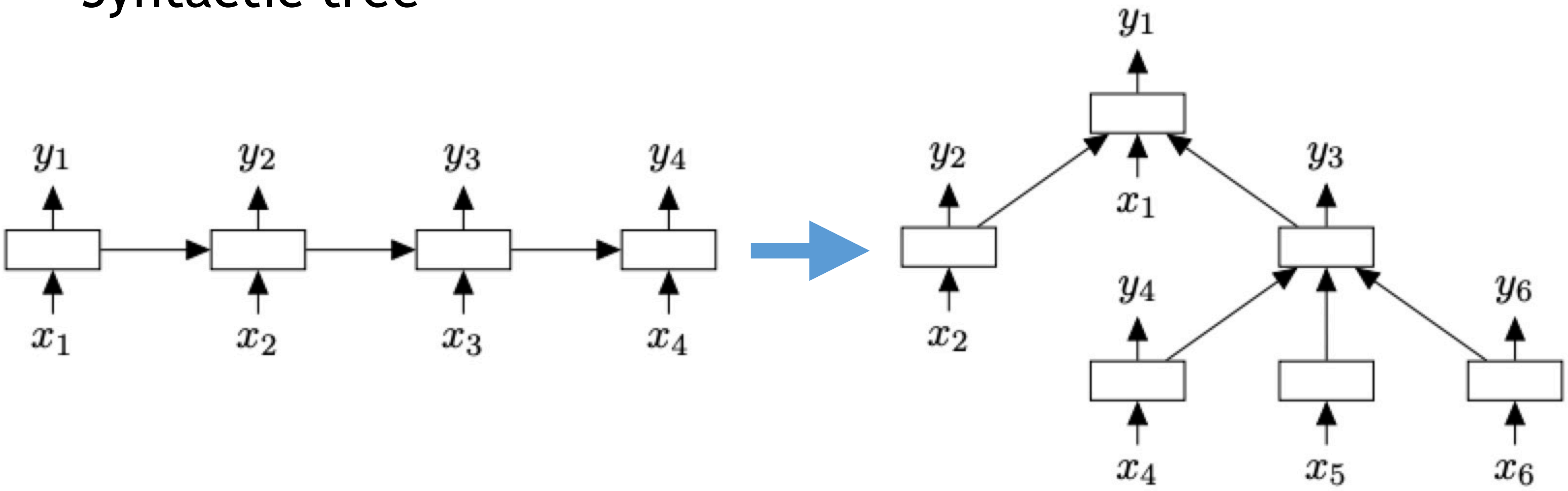


# Context2vec

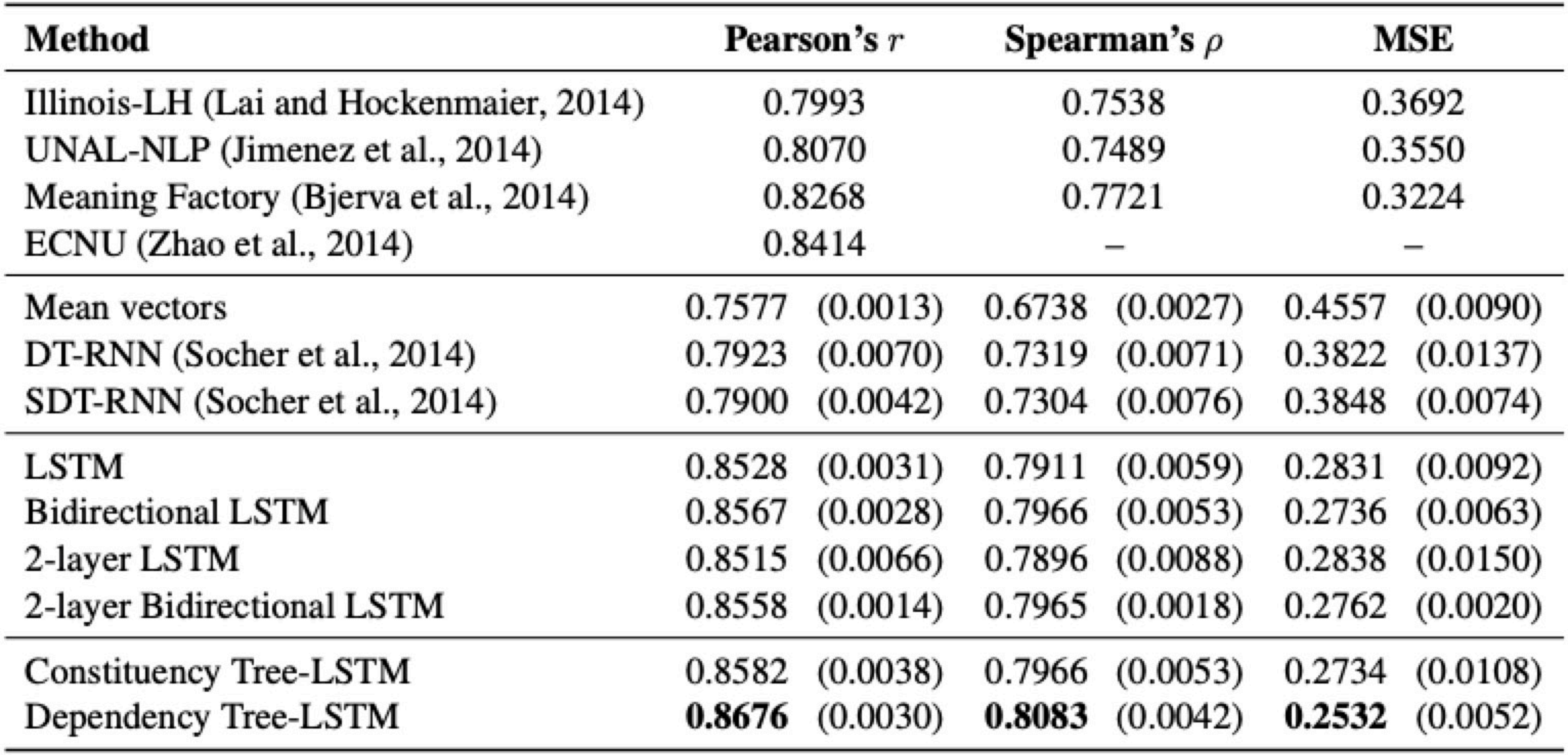
- We already covered enhancement to word2vec and sentence encoders in previous class, again, context2vec proves that structure info could significantly help representing texts
- This study implicitly build the connections between representing word and other text granularities, which enlightens other research in the similar vein
- A good guidance for you to learn how to do a similar research and implement it. https://github.com/orenmel/context2vec

# TreeLSTM

- Instead of sequential order of words in a sentence, what other structures we can leverage?
  - Syntactic tree



# TreeLSTM



| **Method** | **Pearson's $r$** | | **Spearman's $\rho$** | | **MSE** | |
|---|---|---|---|---|---|---|
| Illinois-LH (Lai and Hockenmaier, 2014) | 0.7993 | | 0.7538 | | 0.3692 | |
| UNAL-NLP (Jimenez et al., 2014) | 0.8070 | | 0.7489 | | 0.3550 | |
| Meaning Factory (Bjerva et al., 2014) | 0.8268 | | 0.7721 | | 0.3224 | |
| ECNU (Zhao et al., 2014) | 0.8414 | | – | | – | |
| Mean vectors | 0.7577 | (0.0013) | 0.6738 | (0.0027) | 0.4557 | (0.0090) |
| DT-RNN (Socher et al., 2014) | 0.7923 | (0.0070) | 0.7319 | (0.0071) | 0.3822 | (0.0137) |
| SDT-RNN (Socher et al., 2014) | 0.7900 | (0.0042) | 0.7304 | (0.0076) | 0.3848 | (0.0074) |
| LSTM | 0.8528 | (0.0031) | 0.7911 | (0.0059) | 0.2831 | (0.0092) |
| Bidirectional LSTM | 0.8567 | (0.0028) | 0.7966 | (0.0053) | 0.2736 | (0.0063) |
| 2-layer LSTM | 0.8515 | (0.0066) | 0.7896 | (0.0088) | 0.2838 | (0.0150) |
| 2-layer Bidirectional LSTM | 0.8558 | (0.0014) | 0.7965 | (0.0018) | 0.2762 | (0.0020) |
| Constituency Tree-LSTM | 0.8582 | (0.0038) | 0.7966 | (0.0053) | 0.2734 | (0.0108) |
| Dependency Tree-LSTM | **0.8676** | (0.0030) | **0.8083** | (0.0042) | **0.2532** | (0.0052) |

Performance comparison on the SICK data.

# TreeLSTM

- Why TreeLSTM works better than conventional LSTM?
    - Captures salient words in a sentence
    - Each path is encoded with more semantics
    - Better way to represent long distance dependencies
- Still, limitation?
    - Requires a parser to produce the structure
    - Not a simple model, with carefully designed implementation

# TransE

- Knowledge graph
  - OpenCyc, WordNet, Freebase, DBpedia
- Basic units
  - Triplets: (head, relation, tail)
    (Barack Obama, place of birth, Hawai)
    (Albert Einstein, follows diet, Veganism)
    (San Francisco, contains, Telegraph Hill)

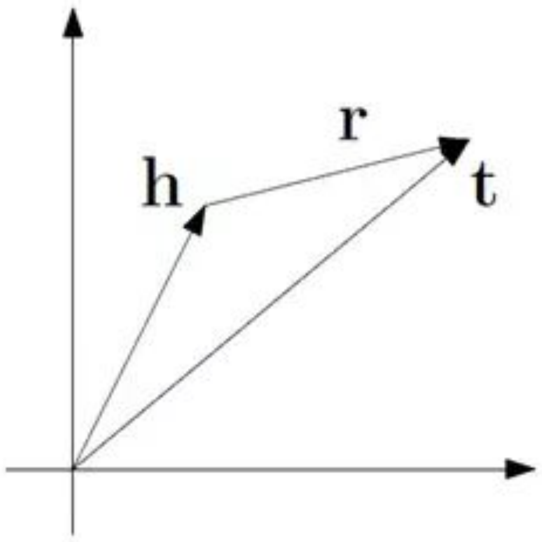
# TransE

- It is natural to think about using embeddings to represent knowledge graph

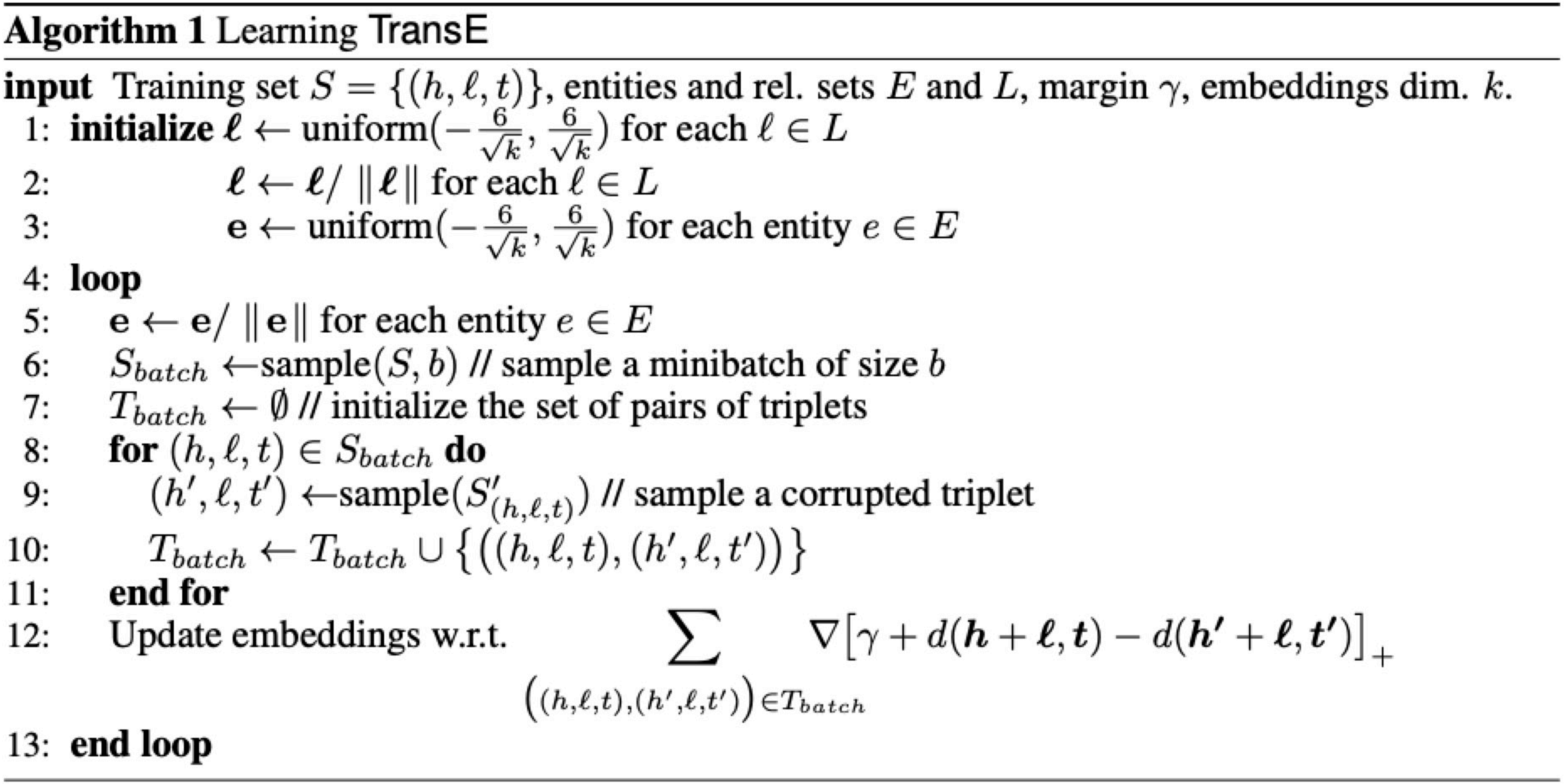
- Two questions:
  - How to vectorize a knowledge graph?
  - What is the most effective way to represent it?

# TransE

- Leverage triplets for a case-by-case learning
- To “translate” head into tail with respect to the relation
  - h -> (r) -> t    or    h + r = t
  - Initialize and learn embeddings for both entities and relations
- Analogy to word2vec
  - h in the context of r, predict t



# TransE



**Algorithm 1** Learning TransE

```
input Training set $S = \{(h, \ell, t)\}$, entities and rel. sets $E$ and $L$, margin $\gamma$, embeddings dim. $k$.
initialize $\boldsymbol{\ell} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$
           $\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} / \|\boldsymbol{\ell}\|$ for each $\ell \in L$
           $\mathbf{e} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$
loop
    $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$
    $S_{batch} \leftarrow \text{sample}(S, b)$ // sample a minibatch of size $b$
    $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets
    for $(h, \ell, t) \in S_{batch}$ do
        $(h', \ell, t') \leftarrow \text{sample}(S'_{(h,\ell,t)})$ // sample a corrupted triplet
        $T_{batch} \leftarrow T_{batch} \cup \left\{\left((h, \ell, t), (h', \ell, t')\right)\right\}$
    end for
    Update embeddings w.r.t. $\sum_{\left((h,\ell,t),(h',\ell,t')\right) \in T_{batch}} \nabla\left[\gamma + d(\boldsymbol{h} + \boldsymbol{\ell}, \boldsymbol{t}) - d(\boldsymbol{h'} + \boldsymbol{\ell}, \boldsymbol{t'})\right]_+$
end loop
```

# TransE

- Pros and Cons
  - A straightforward way to model knowledge graph
  - Efficient use of data structure
  - Restricted to one-to-one relation
    (space needle, location, Seattle)
    (UW, location, Seattle)

# TransE

- Extensions
    - TransH (2014): one-to-many, many-to-one relations
    - TransR (2015): separate relation space
    - TransD (2015): distinguish translation matrices, reduce parameter numbers in TransR

# Node2vec

- Is there a solution to represent generic networks
  - Social networking
  - Advertisement nets
  - Query-session associations
  - …
- Two questions:
  - What to model for a network?
  - How to model a network with an efficient way?

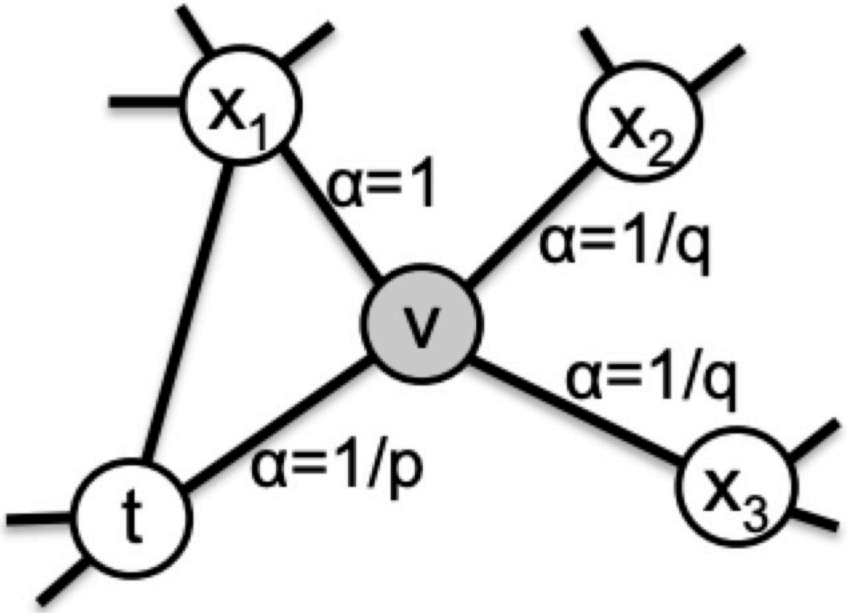
# Node2vec

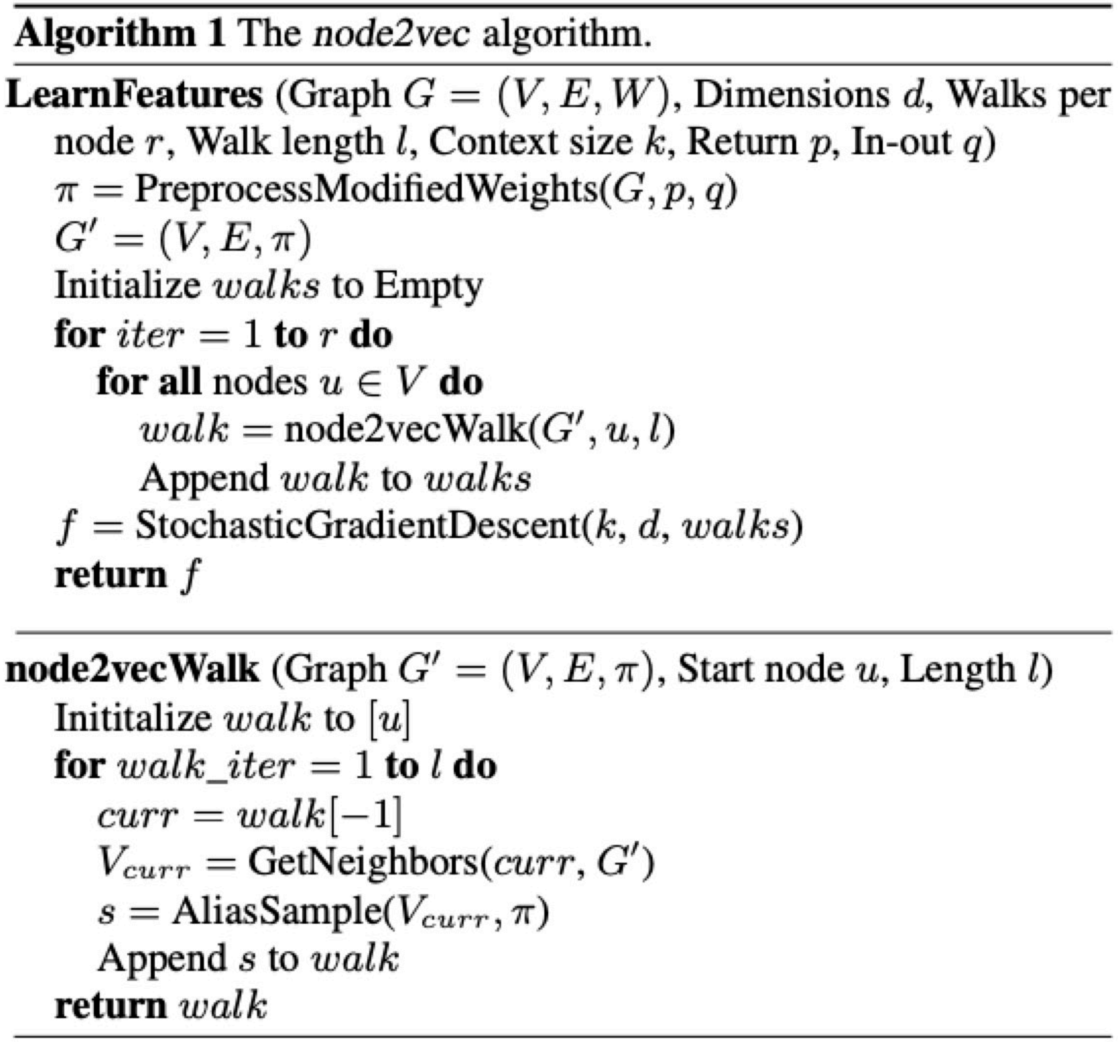
- A clustering way to model neighbor nodes:
  - Content similarity
    - Neighbor nodes should share some characteristics because they are in similar locations
  - Structure similarity
    - Not necessarily neighbor nodes, could be faraway ones sharing common features in network structures

# Node2vec

- Use skip-gram to learn representations for nodes
- One important things to consider:
    - What is the context?
        - Conventionally, DFS or BFS to sample neighbor nodes, micro- v.s. macro-view
        - Representative issue for both DFS and BFS
    - Solution: random walk, referring to DeepWalk (KDD, 2014)

# Node2vec



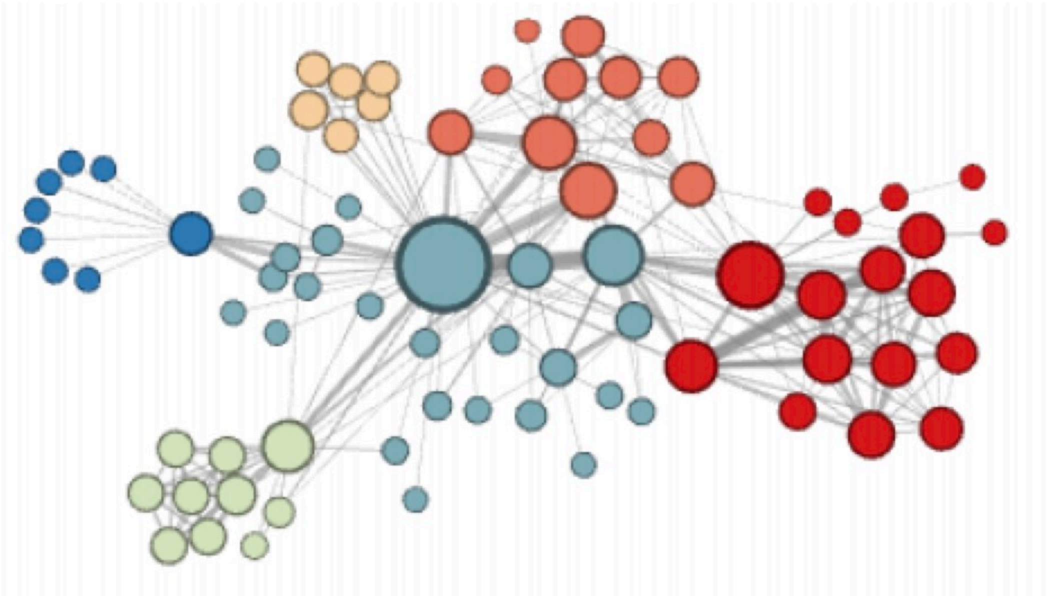


**Algorithm 1** The *node2vec* algorithm.

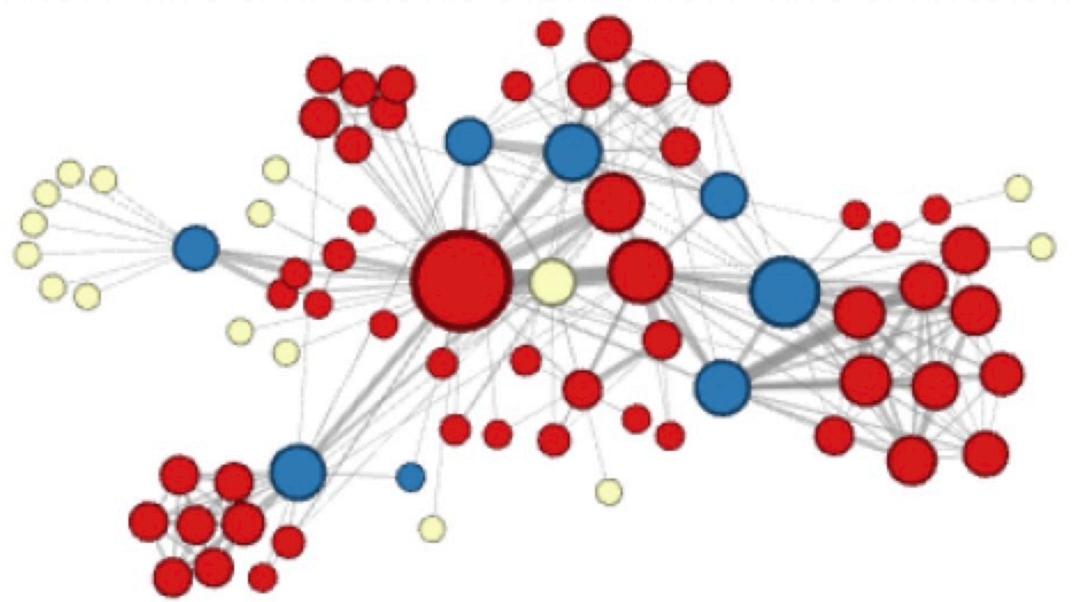
```
LearnFeatures (Graph $G = (V, E, W)$, Dimensions $d$, Walks per
  node $r$, Walk length $l$, Context size $k$, Return $p$, In-out $q$)
  $\pi$ = PreprocessModifiedWeights($G, p, q$)
  $G' = (V, E, \pi)$
  Initialize $walks$ to Empty
  for $iter = 1$ to $r$ do
    for all nodes $u \in V$ do
      $walk$ = node2vecWalk($G', u, l$)
      Append $walk$ to $walks$
  $f$ = StochasticGradientDescent($k$, $d$, $walks$)
  return $f$

node2vecWalk (Graph $G' = (V, E, \pi)$, Start node $u$, Length $l$)
  Inititalize $walk$ to $[u]$
  for $walk\_iter = 1$ to $l$ do
    $curr = walk[-1]$
    $V_{curr}$ = GetNeighbors($curr$, $G'$)
    $s$ = AliasSample($V_{curr}, \pi$)
    Append $s$ to $walk$
  return $walk$
```

# Node2vec



Similar in content



Similar in structure

# Node2vec

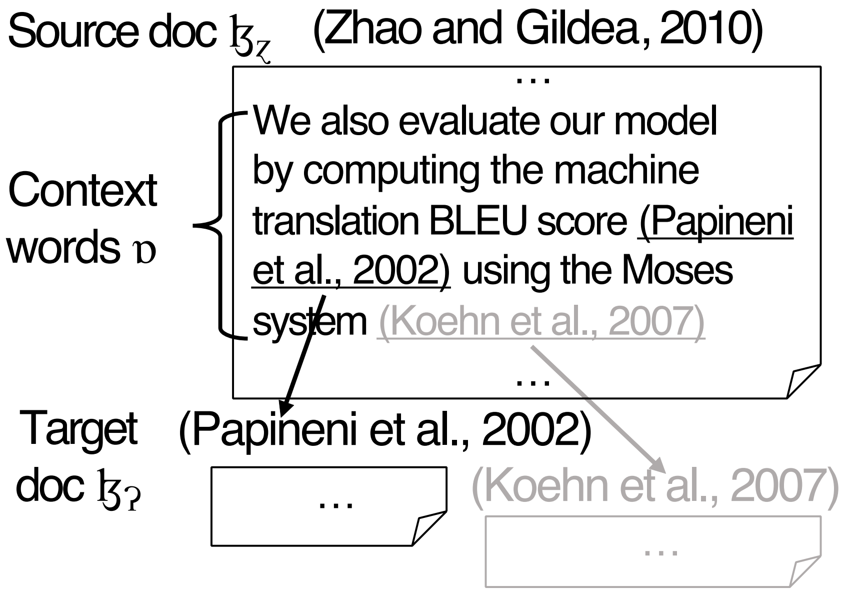
- Pros and Cons
    - Model networks in a bottom-up manner
    - Model homophily and impact of nodes in a unified framework
    - In a strained application of skip-gram
        - Neighborhood of nodes does not like context window of words

# Hyperdoc2vec

- What are the most widely used structures of text?
    - Hyper-documents, e.g., HTML, and what else?
- This type of texts and normally document-based, and has strong association among documents w.r.t. topics, class, and other clustering criteria
- For example, Wikipedia pages are grouped in topics and linked to relevant topics.

# Hyperdoc2vec

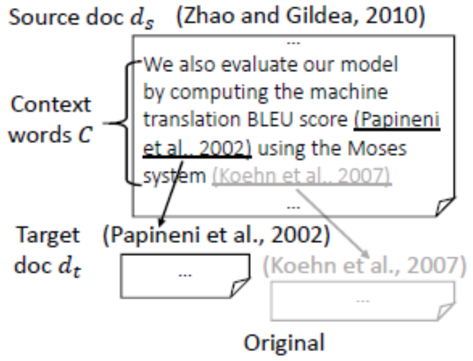
- Hyper-documents, e.g., academic papers:
  - Textual contents + hyper-links (citations)
  - Embeddings may facilitate
    - Hyper-document classification
    - Citation recommendation
    - Embedding-based entity linking
- Desired properties of approaches
  - Content awareness
  - Context awareness
  - Newcomer friendliness
  - Context intent awareness



We also evaluate our model by computing the machine translation BLEU score (Papineni et al., 2002) using the Moses system (Koehn et al., 2007)

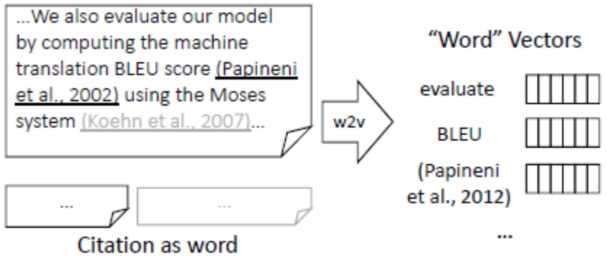
# Hyperdoc2vec

- Conventional approaches
  - word2vec  (citation as words): violates Properties 1, 3, and 4
  - doc2vec (context as content): violates Property 4
  - DeepWalk & node2vec: only encodes network structure
  - Studies considering both: task-specific and non-generalizable



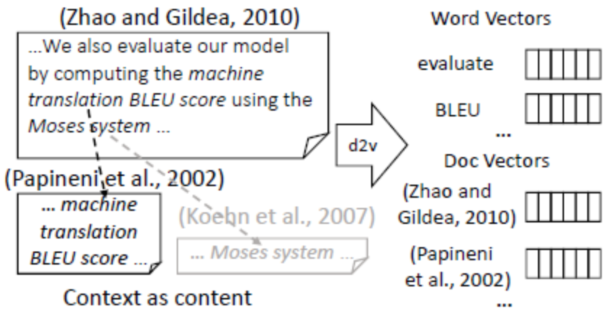
Source doc $d_s$
(Zhao and Gildea, 2010)
...
Context words $C$
We also evaluate our model by computing the machine translation BLEU score (Papineni et al., 2002) using the Moses system (Koehn et al., 2007)
...
Target doc $d_t$
(Papineni et al., 2002)
...
(Koehn et al., 2007)
...
Original

(a) Hyper-documents.

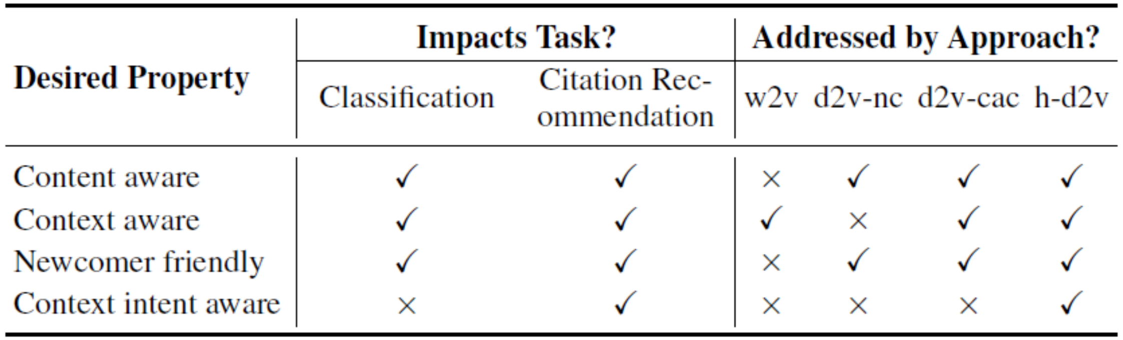


...We also evaluate our model by computing the machine translation BLEU score (Papineni et al., 2002) using the Moses system (Koehn et al., 2007)...
...
...
Citation as word
w2v
"Word" Vectors
evaluate
BLEU
(Papineni et al., 2012)
...

(b) Citation as word.



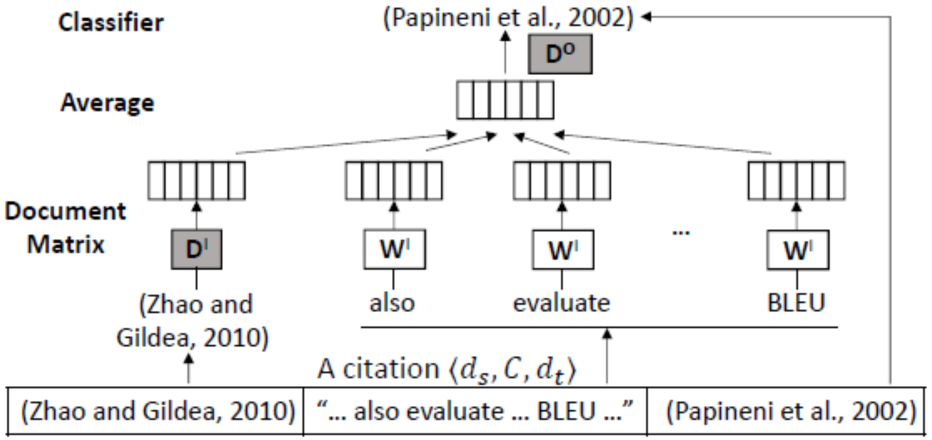
(c) Context as content.



| Desired Property | Impacts Task? | | Addressed by Approach? | | | |
|---|---|---|---|---|---|---|
| | Classification | Citation Recommendation | w2v | d2v-nc | d2v-cac | h-d2v |
| Content aware | ✓ | ✓ | × | ✓ | ✓ | ✓ |
| Context aware | ✓ | ✓ | ✓ | × | ✓ | ✓ |
| Newcomer friendly | ✓ | ✓ | × | ✓ | ✓ | ✓ |
| Context intent aware | × | ✓ | × | × | × | ✓ |

# Hyperdoc2vec

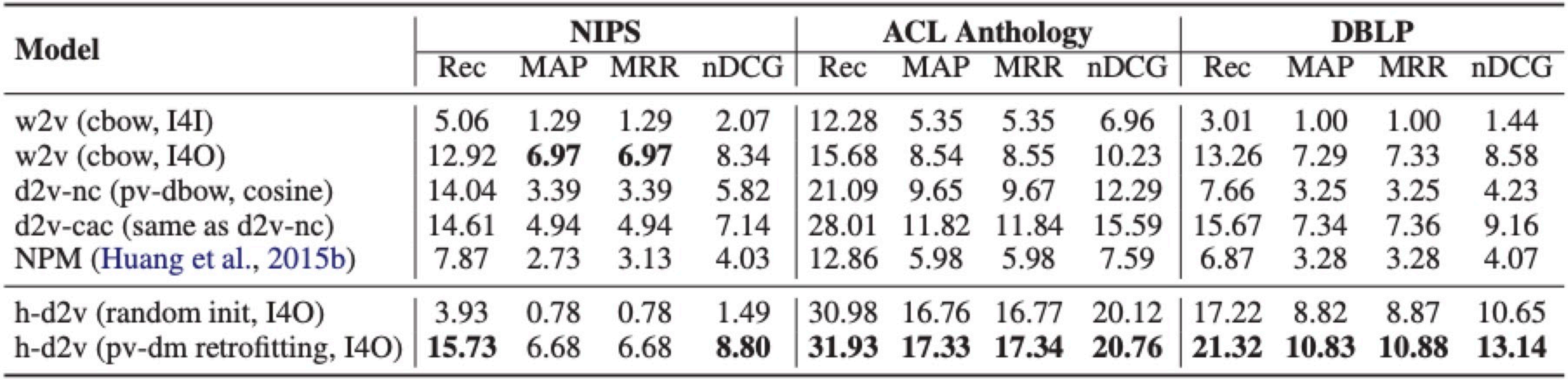
- Represents a doc with two vectors (IN/OUT)
  - IN vector encodes contents & out-links;
  - OUT vector encodes in-links & contexts of in-links.



- Satisfies all four properties
  - Content awareness: initialization by pv-dm
  - Context awareness: “BLEU” -> (Papineni et al., 2002)
  - Newcomer friendliness: newcomers have IN vectors at least.
  - Context intent awareness: (Zhao and Gildea, 2010) + “evaluate by” -> (Papineni et al., 2002)

- Task-independent and generalizable

# Hyperdoc2vec



| Model | NIPS | | | | ACL Anthology | | | | DBLP | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Rec | MAP | MRR | nDCG | Rec | MAP | MRR | nDCG | Rec | MAP | MRR | nDCG |
| w2v (cbow, I4I) | 5.06 | 1.29 | 1.29 | 2.07 | 12.28 | 5.35 | 5.35 | 6.96 | 3.01 | 1.00 | 1.00 | 1.44 |
| w2v (cbow, I4O) | 12.92 | **6.97** | **6.97** | 8.34 | 15.68 | 8.54 | 8.55 | 10.23 | 13.26 | 7.29 | 7.33 | 8.58 |
| d2v-nc (pv-dbow, cosine) | 14.04 | 3.39 | 3.39 | 5.82 | 21.09 | 9.65 | 9.67 | 12.29 | 7.66 | 3.25 | 3.25 | 4.23 |
| d2v-cac (same as d2v-nc) | 14.61 | 4.94 | 4.94 | 7.14 | 28.01 | 11.82 | 11.84 | 15.59 | 15.67 | 7.34 | 7.36 | 9.16 |
| NPM (Huang et al., 2015b) | 7.87 | 2.73 | 3.13 | 4.03 | 12.86 | 5.98 | 5.98 | 7.59 | 6.87 | 3.28 | 3.28 | 4.07 |
| h-d2v (random init, I4O) | 3.93 | 0.78 | 0.78 | 1.49 | 30.98 | 16.76 | 16.77 | 20.12 | 17.22 | 8.82 | 8.87 | 10.65 |
| h-d2v (pv-dm retrofitting, I4O) | **15.73** | 6.68 | 6.68 | **8.80** | **31.93** | **17.33** | **17.34** | **20.76** | **21.32** | **10.83** | **10.88** | **13.14** |

Citation recommendation results on three paper datasets.

# Hyperdoc2vec

- Summary
  - A simple way to model hyper-docs
  - A generic method, has the potential to perform other structural text
  - Can be enhanced on many aspects
    - Long-distance dependencies
    - Citation (link) type
    - Something to borrow from TransX?

# Hw7

- Prepare your presentations for recent pre-trained models
  - Group presentation
  - 40 mins
  - Done by May 20th, 11:59am
- Three studies to choose
  - ELMo
  - GPT
  - BERT

# Hw7

- Content
    - Motivation (why and how this model is presented)
    - Model
        - Design (architecture? why it is designed in this way)
        - How to learn it (what objectives? and why)
    - Usage
        - Performance (on what tasks? what data?)
    - Discussion