Entity Hierarchy Embedding

Zhiting Hu, Poyao Huang, Yuntian Deng, Yingkai Gao, Eric P. Xing
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{zhitingh,poyaoh,yuntiand,yingkaig,epxing}@cs.cmu.edu

Abstract

Existing distributed representations are limited in utilizing structured knowledge to improve semantic relatedness modeling. We propose a principled framework of embedding entities that integrates hierarchical information from large-scale knowledge bases. The novel embedding model associates each category node of the hierarchy with a distance metric. To capture structured semantics, the entity similarity of context prediction are measured under the aggregated metrics of relevant categories along all inter-entity paths. We show that both the entity vectors and category distance metrics encode meaningful semantics. Experiments in entity linking and entity search show superiority of the proposed method.

1 Introduction

There has been a growing interest in distributed representation that learns compact vectors (a.k.a embedding) for words (Mikolov et al., 2013a), phrases (Passos et al., 2014), and concepts (Hill and Korhonen, 2014), etc. The induced vectors are expected to capture semantic relatedness of the linguistic items, and are widely used in sentiment analysis (Tang et al., 2014), machine translation (Zhang et al., 2014), and information retrieval (Clinchant and Perronnin, 2013), to name a few.

Despite the impressive success, existing work is still limited in utilizing structured knowledge to enhance the representation. For instance, word and phrase embeddings are largely induced from plain text. Though recent knowledge graph embeddings (Lin et al., 2015; Wang et al., 2014) integrate the relational structure among entities, they primarily target at link prediction and lack an explicit relatedness measure.

In this paper, we propose to improve the distributed representations of entities by integrating hierarchical information from large-scale knowledge bases (KBs). An entity hierarchy groups entities into categories which are further organized to form a taxonomy. It provides rich structured knowledge on entity relatedness (Resnik, 1995). Our work goes beyond the previous heuristic use of entity hierarchy which relies on hand-crafted features (Kaptein and Kamps, 2013; Ponzetto and Strube, 2007), and develops a principled optimization-based framework. We learn a *distance metric* for each category node, and measure entity-context similarity under the aggregated metrics of all relevant categories. The metric aggregation encodes the hierarchical property that nearby entities tend to share common semantic features. We further provide a highly-efficient implementation in order to handle large complex hierarchies.

We train a distributed representation for the whole entity hierarchy of Wikipedia. Both the entity vectors and the category distance metrics capture meaningful semantics. We deploy the embedding in both entity linking (Han and Sun, 2012) and entity search (Demartini et al., 2010) tasks. Hierarchy embedding significantly outperforms that without structural knowledge. Our methods also show superiority over existing competitors.

To the best of our knowledge, this is the first work to learn distributed representations that incorporates hierarchical knowledge in a principled framework. Our model that encodes hierarchy by distance metric learning and aggregation provides a potentially important and general scheme for utilizing hierarchical knowledge.

The rest of the paper is organized as follows: §2 describes the proposed embedding model; §3 presents the application of the learned embedding; §4 evaluates the approach; §5 reviews related work; and finally, §6 concludes the paper.

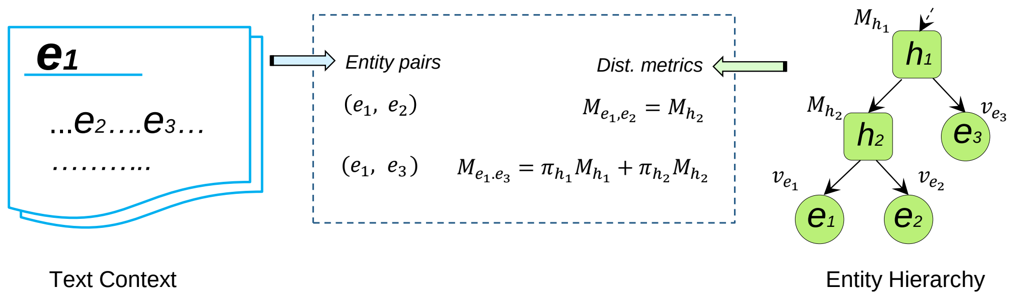

Figure 1: The model architecture. The text context of an entity is based on its KB encyclopedia article. The entity hierarchical structure is incorporated through distance metric learning and aggregation.

2 Entity Hierarchy Embedding

The objective of the embedding model is to find a representation for each entity that is useful for predicting other entities occurring in its *context*. We build entity's context upon KB encyclopedia articles, where entity annotations are readily available. We further incorporate the entity hierarchical structure in the context prediction through distance metric learning and aggregation, which encodes the rich structured knowledge in the induced representations. Our method is flexible and efficient to model large complex DAG-structured hierarchies. Figure 1 shows an overview of the model architecture.

2.1 Model Architecture

Our architecture builds on the skip-gram word embedding framework (Mikolov et al., 2013b). In the skip-gram model, a set of (target, context) word pairs are extracted by sliding a fixed-length context window over a text corpus, and the word vectors are learned such that the similarity of the target- and context-word vectors is maximized. We generalize both the context definition and the similarity measure for entity hierarchy embedding.

Unlike words that can be directly extracted from plain text, entities are hidden semantics underlying their surface forms. In order to avoid manual annotation cost, we exploit the text corpora from KBs where the referent entities of surface text are readily annotated. Moreover, since a KB encyclopedia article typically focuses on describing one entity, we naturally extend the entity's context as its whole article, and obtain a set of entity pairs $\mathcal{D} = \{(e_T, e_C)\}$, where e_T denotes the target-entity and e_C denotes the context-entity occurring in entity e_T's context.

Let $\mathcal{E}$ be the set of entities. For each entity $e \in \mathcal{E}$, the model learns both a "target vector" $v_e \in \mathbb{R}^n$ and "context vector" $\bar{v}_e \in \mathbb{R}^n$, by maximizing the training objective

$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum\nolimits_{(e_T,e_C)\in\mathcal{D}} \log p(e_C|e_T), \qquad (1)$$

where the prediction probability is defined as a softmax:

$$p(e_C|e_T) = \frac{\exp\{-d(e_T, e_C)\}}{\sum_{e\in\mathcal{E}} \exp\{-d(e_T, e)\}}. \qquad (2)$$

Here $d(e, e')$ is the distance between the target vector of e (i.e., v_e) and the context vector of e' (i.e., $\bar{v}_{e'}$). We present the design in the following.

2.2 Hierarchical Extension

An entity hierarchy takes entities as leaf nodes and categories as internal nodes, which provides key knowledge sources on semantic relatedness that 1) far-away entities in the hierarchy tend to be semantically distant, and 2) nearby entities tend to share common semantic features. We aim to encode this knowledge in our representations. As KB hierarchies are large complex DAG structures, we develop a highly-efficient scheme to enable practical training.

Specifically, we associate a separate *distance metric* $M_h \in \mathbb{R}^{n\times n}$ with each category h in the hierarchy. A distance metric is a positive semidefinite (PSD) matrix. We then measure the distance between two entities under some *aggregated* distance metric as detailed below. The local metrics thus not only serve to capture the characteristics of individual categories, but also make it possible to share the representation across entities through metric aggregation of relevant categories.

Metric aggregation Given two entities e and e', let $\mathcal{P}_{e,e'}$ be the path between them. One obvious way to define the aggregated metric $M_{e,e'} \in \mathbb{R}^{n\times n}$ is through a combination of the metrics on the

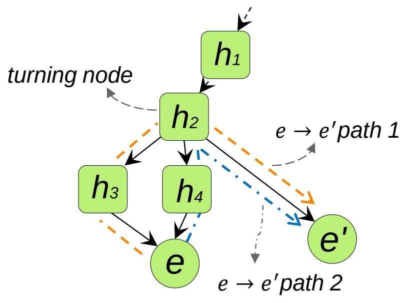

Figure 2: Paths in a DAG-structured hierarchy. A path P is defined as a sequence of non-duplicated nodes with the property that there exists a *turning node* $t \in P$ such that any two consecutive nodes before t are (child, parent) pairs, while consecutive nodes after t are (parent, child) pairs. Thus a turning node is necessarily a common ancestor.

path: $\sum_{h \in \mathcal{P}_{e,e'}} M_h$, leading to a nice property that the more nodes a path has, the more distant the entities tend to be (as M_h is PSD). This simple strategy, however, can be problematic when the hierarchy has a complex DAG structure, in that there can be multiple paths between two entities (Figure 2). Though the shortest path can be selected, it ignores other related category nodes and loses rich information. In contrast, an ideal scheme should not only mirror the distance in the hierarchy, but also take into account all possible paths in order to capture the full aspects of relatedness.

However, hierarchies in large KBs can be complex and contains combinationally many paths between any two entities. We propose an efficient approach that avoids enumerating paths and instead models the underlying nodes directly. In particular, we extend $\mathcal{P}_{e,e'}$ as the set of all category nodes included in any of the $e \to e'$ paths, and define the aggregate metric as

$$M_{e,e'} = \gamma_{e,e'} \sum_{h \in \mathcal{P}_{e,e'}} \pi_{ee',h} M_h, \qquad (3)$$

where $\{\pi_{ee',h}\}$ are the relative weights of the categories such that $\sum_{h \in \mathcal{P}_{e,e'}} \pi_{ee',h} = 1$. This serves to balance the size of $\mathcal{P}$ across different entity pairs. We set $\pi_{ee',h} \propto (\frac{1}{s_{h \downarrow e}} + \frac{1}{s_{h \downarrow e'}})$ with $s_{h \downarrow e}$ being the average #steps going down from node h to node e in the hierarchy (infinite if h is not an ancestor of e). This implements the intuition that an entity (e.g., "Iphone") is more relevant to its immediate categories (e.g., "Mobile phones") than to farther and more generic ancestors (e.g., "Technology"). The scaling factor $\gamma_{e,e'}$ encodes the distance of the entities in the hierarchy and can be of various choices. We set $\gamma_{e,e'} = \min_h \{s_{h \downarrow e} + s_{h \downarrow e'}\}$ to mirror the least common ancestor.

In Figure 2, $\mathcal{P}_{e,e'} = \{h_2, h_3, h_4\}$, and the relative weights of the categories are $\pi_{ee',h_2} \propto 3/2$ and $\pi_{ee',h_3} = \pi_{ee',h_4} \propto 1$. Category h_2 is the least common ancestor and $\gamma_{e,e'} = 3$.

Based on the aggregated metric, the distance between a target entity e_T and a context entity e_C can then be measured as

$$d(e_T, e_C) = (v_{e_T} - \tilde{v}_{e_C})^\top M_{e_T,e_C} (v_{e_T} - \tilde{v}_{e_C}). \qquad (4)$$

Note that nearby entities in the hierarchy tend to share a large proportion of local metrics in Eq 3, and hence can exhibit common semantic features when measuring distance with others.

Complexity of aggregation As computing distance is a frequent operation in both training and application stages, a highly efficient aggregation algorithm is necessary in order to handle complex large entity hierarchies (with millions of nodes). Our formulation (Eq 3) avoids exhaustive enumeration over all paths by modeling the relevant nodes directly. We show that this allows linear complexity in *the number of children of two entities' common ancestors*, which is efficient in practice.

The most costly operation is to find $\mathcal{P}_{e,e'}$, i.e., the set of all category nodes that can occur in any of $e \to e'$ paths. We use a two-step procedure that (1) finds all common ancestors of entity e and e' that are *turning nodes* of any $e \to e'$ paths (e.g., h_2 in Figure 2), denoted as $\mathcal{Q}_{e,e'}$; (2) expands from $\mathcal{Q}_{e,e'}$ to construct the full $\mathcal{P}_{e,e'}$. For the first step, the following theorem shows each common ancestor can be efficiently assessed by testing only its children nodes. For the second step, it is straightforward to see that $\mathcal{P}_{e,e'}$ can be constructed by expanding $\mathcal{Q}_{e,e'}$ with its descendants that are ancestors of either e or e'. Other parameters ($\boldsymbol{\pi}_{ee'}$ and $\gamma_{e,e'}$) of aggregation can be computed during the above process.

We next provide the theorem for the first step. Let $\mathcal{A}_e$ be the ancestor nodes of entity e (including e itself). For a node $h \in \mathcal{A}_e \cup \mathcal{A}_{e'}$, we define its *critical* node t_h as the nearest (w.r.t the length of the shortest path) descendant of h (including h itself) that is in $\mathcal{Q}_{e,e'} \cup \{e, e'\}$. E.g., in Figure 2, $t_{h_1} = h_2; t_{h_2} = h_2; t_{h_3} = e$. Let $\mathcal{C}_h$ be the set of immediate child nodes of h.

Theorem 1. *$\forall h \in \mathcal{A}_e \cap \mathcal{A}_{e'}$, $h \in \mathcal{Q}_{e,e'}$ iff it satisfies the two conditions: (1) $|\mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})| \geq 2$; (2) $\exists a, b \in \mathcal{C}_h$ s.t. $t_a \neq t_b$.*

Proof. We outline the proof here, and provide the details in the appendix.

Sufficiency: Note that $e, e' \notin Q_{e,e'}$. We prove the sufficiency by enumerating possible situations: (i) $t_a = e, t_b = e'$; (ii) $t_a = e, t_b \in Q_{e,e'}$; (iii) $t_a, t_b \in Q_{e,e'}$. For (i): as $t_a = e$, there exists a path $e \to \cdots \to a \to h$ where any two consecutive nodes is a (child, parent) pair. Similarly, there is a path $h \to b \to \cdots \to e'$ where any two consecutive nodes is a (parent, child) pair. It is provable that the two paths intersect only at h, and thus can be combined to form an $e \to e'$ path: $e \to \cdots \to a \to h \to b \to \cdots \to e'$, yielding h as a turning node. The cases (ii) and (iii) can be proved similarly.

Necessity: We prove by contradiction. Suppose that $\forall a, b \in \mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})$ we have $t_a = t_b$. W.l.o.g. we consider two cases: (i) $t_a = t_b = e$, and (ii) $t_a = t_b \in Q_{e,e'}$. It is provable that both cases will lead to contradiction. $\square$

Therefore, by checking common ancestors from the bottom up, we can construct $\mathcal{Q}_{e,e'}$ with time complexity linear to the number of all ancestors' children.

2.3 Learning

For efficiency, we use negative sampling to reformulate the training objective, which is then optimized through coordinate gradient ascent.

Specifically, given the training data $\{(e_T, e_C)\}$ extracted from KB corpora, the representation learning is formulated as maximizing the objective in Eq 1, subject to PSD constraints on distance metrics $M_h \succeq 0$, and $\|v_e\|_2 = \|\bar{v}_e\|_2 = 1$ to avoid scale ambiguity.

The likelihood of each data sample is defined as a softmax in Eq 2, which iterates over all entities in the denominator and is thus computationally prohibitive. We apply the negative sampling technique as in conventional skip-gram model, by replacing each log probability $\log p(e_C|e_T)$ with

$$\log \sigma(-d(e_T, e_C)) + \sum\nolimits_{i=1}^{k} \mathbb{E}_{e_i \sim P(e)} \left[\log \sigma(-d(e_T, e_i))\right],$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function; and for each data sample we draw k negative samples from the noise distribution $P(e) \propto U(e)^{3/4}$ with $U(e)$ being the unigram distribution (Mikolov et al., 2013b).

The negative sampling objective is optimized using coordinate gradient ascent, as shown in Algorithm 1. To avoid overfitting and improve efficiency, in practice we restrict the distance metrics M_h to be diagonal (Xing et al., 2002). Thus the PSD project of M_h (line 17) is simply taking the positive part for each diagonal elements.

Algorithm 1 Entity Hierarchy Embedding

```
Input: The training data $\mathcal{D} = \{(e_T, e_C)\}$,
       Entity hierarchy,
       Parameters: $n$ – dimension of the embedding
                   $k$ – number of negative samples
                   $\eta$ – gradient learning rate
                   $B$ – minibatch size
 1: Initialize $\boldsymbol{v}, \bar{\boldsymbol{v}}, \boldsymbol{M}$ randomly such that $\|v\|_2 = \|\bar{v}\|_2 = 1$
    and $M \succeq 0$.
 2: repeat
 3:   Sample a batch $\mathcal{B} = \{(e_T, e_C)_i\}_{i=1}^{B}$ from $\mathcal{D}$
 4:   for all $(e_T, e_C) \in \mathcal{B}$ do
 5:     Compute $\{\mathcal{P}, \boldsymbol{\pi}, \gamma\}_{e_T, e_C}$ for metric aggregation
 6:     Sample negative pairs $\{(e_T, e_i)\}_{i=1}^{k}$
 7:     Compute $\{\{\mathcal{P}, \boldsymbol{\pi}, \gamma\}_{e_T, e_i}\}_{i=1}^{k}$ for metric aggrega-
        tion
 8:   end for
 9:   repeat
10:     for all $e \in \mathcal{E}$ included in $\mathcal{B}$ do
11:       $v_e = v_e + \eta \frac{\partial \mathcal{L}}{\partial v_e}$
12:       $\bar{v}_e = \bar{v}_e + \eta \frac{\partial \mathcal{L}}{\partial \bar{v}_e}$
13:       $v_e, \bar{v}_e$ = Project_to_unit_sphere$(v_e, \bar{v}_e)$
14:     end for
15:   until convergence
16:   repeat
17:     for all $h$ included in $\mathcal{B}$ do
18:       $M_h = M_h + \eta \frac{\partial \mathcal{L}}{\partial M_h}$
19:       $M_h$ = Project_to_PSD$(M_h)$
20:     end for
21:   until convergence
22: until convergence
Output: Entity vectors $\boldsymbol{v}, \bar{\boldsymbol{v}}$, and category dist. metrics $\boldsymbol{M}$
```

3 Applications

One primary goal of learning semantic embedding is to improve NLP tasks. The compact representations are easy to work with because they enable efficient computation of semantic relatedness. Compared to word embedding, entity embedding is particularly suitable for various language understanding applications that extract underlying semantics of surface text. Incorporating entity hierarchies further enriches the embedding with structured knowledge.

In this section, we demonstrate how the learned entity hierarchy embedding can be utilized in two important tasks, i.e., *entity linking* and *entity search*. In both tasks, we measure the semantic relatedness between entities as the reciprocal distance defined in Eq 4. This greatly simplifies previous methods which have used various hand-crafted features, and leads to improved performance as shown in our experiments.

3.1 Entity Linking

The entity linking task is to link surface forms (mentions) of entities in a document to entities in a reference KB. It is an essential first step for downstream tasks such as semantic search and KB construction. The quality of entity relatedness measure is critical for entity linking performance, because of the key observation that entities in a document tend to be semantically coherent. For example, in sentence "Apple released an operating system Lion", The mentions "Apple" and "Lion" refer to Apple Inc. and Mac OS X Lion, respectively, as is more coherent than other configurations like (fruit apple, animal lion).

Our algorithm finds the optimal configuration for the mentions of a document by maximizing the overall relatedness among assigned entities, together with the local mention-to-entity compatibility. Specifically, we first construct a mention-to-entity dictionary based on Wikipedia annotations. For each mention m, the dictionary contains a set of candidate entities and for each candidate entity e a compatibility score $P(e|m)$ which is proportional to the frequency that m refers to e. For efficiency we only consider the top-5 candidate entities according to $P(e|m)$. Given a set of mentions $\mathcal{M} = \{m_i\}_{i=1}^{M}$ in a document, let $\mathcal{A} = \{e_{m_i}\}_{i=1}^{M}$ be a configuration of its entity assignments. The score of $\mathcal{A}$ is formulated as probability

$$P(\mathcal{A}|\mathcal{M}) \propto \prod\nolimits_{i=1}^{M} P(e_{m_i}|m_i) \sum\nolimits_{\substack{j=1 \\ j\neq i}}^{M} \frac{1}{d\left(e_{m_i}, e_{m_j}\right) + \epsilon},$$

where for each entity assignment we define its global relatedness to other entity assignments as the sum of the reciprocal distances ($\epsilon = 0.01$ is a constant used to avoid divide-by-zero). Direct enumeration of all potential configurations is computationally prohibitive, we therefore use simulated annealing to search for an optimal solution.

3.2 Entity Search

Entity search has attracted a growing interest (Chen et al., 2014b; Balog et al., 2011). Unlike conventional web search that finds unorganized web pages, entity search retrieves knowledge directly by generating a list of relevant entities in response to a search request. The input of the entity search task is a natural language question $\mathcal{Q}$ along with one or more desired entity categories $\mathcal{C}$. For example, a query can be $\mathcal{Q}$ ="films directed by Akira Kurosawa" and $\mathcal{C}$ ={Japanese films}.

Previous methods typically score candidate entities by measuring both the similarity between entity content and the query question $\mathcal{Q}$ (text matching), and the similarity between categories of entities and the query categories $\mathcal{C}$ (category matching).

We apply a similar category matching strategy as in previous work (Chen et al., 2014b) that assesses lexical (e.g., head words) similarity between category names, while replacing the text matching with entity relatedness measure. Specifically, we first extract the underlying entities mentioned in $\mathcal{Q}$ through entity linking, then score each candidate entity by its average relatedness to the entities in $\mathcal{Q}$. For instance, the entity Rashomon will obtain a high score in the above example as it is highly related with the entity Akira Kurosawa in the query. This scheme not only avoids complex document processing (e.g., topic modeling) in text matching, but also implicitly augments the short query text with background knowledge, and thus improves the accuracy and robustness.

4 Experiments

We validate the quality of our entity representation by evaluating its applications of entity linking and entity search on public benchmarks. In the entity linking task, our approach improves the F1 score by 10% over state-of-the-art results. We also validate the advantage of incorporating hierarchical structure. In the entity search task, our simple algorithm shows competitive performance. We further qualitatively analyze the entity vectors and category metrics, both of which capture meaningful semantics, and can potentially open up a wide rage of other applications.

Knowledge base We use the Wikipedia snapshot from Jan 12nd, 2015 as our training data and KB. After pruning administrative information we obtain an entity hierarchy including about 4.1M entities and 0.8M categories organized into 12 layers. Loops in the original hierarchy are removed by deleting bottom-up edges, yielding a DAG structure. We extract a set of 87.6M entity pairs from the wiki links on Wikipedia articles.

We train 100-dimensional vector representations for the entities and distance metrics (100×100 diagonal matrixes) for the categories (we would study the impact of dimensionality in the future). We set the batch size $B = 500$, the initial learning rate $\eta = 0.1$ and decrease it by a factor of 5 whenever the objective value does not increase, and the negative sample size $k = 5$. The model is trained on a Linux machine with 128G RAM and 16 cores. It takes 5 days to converge.

4.1 Entity Linking

4.1.1 Setup

Dataset As our entities based on English Wikipedia include not only named entities (e.g., persons, organizations) but also general concepts (e.g., "computer" and "human"), we use a standard entity linking dataset IITB[1] where mentions of Wikipedia entities are manually annotated exhaustively. The dataset contains about 100 documents and 17K mentions in total. As in the baseline work, we use only the mentions whose referent entities are contained in Wikipedia.

Criteria We adopt the common criteria, *precision*, *recall*, and *F1*. Let $\mathcal{A}^*$ be the golden standard entity annotations, and $\mathcal{A}$ be the annotations by entity linking model, then

$$precision = \frac{|\mathcal{A}^* \cap \mathcal{A}|}{|\mathcal{A}|} \qquad recall = \frac{|\mathcal{A}^* \cap \mathcal{A}|}{|\mathcal{A}^*|}.$$

The F1 score is then computed based on the average precision and recall across all documents.

Baselines We compare our algorithm with the following approaches. All the competitors are designed to be able to link general concept mentions to Wikipedia.

CSAW (Kulkarni et al., 2009) has a similar framework as our algorithm. It measures entity relatedness using a variation of Jaccard similarity on Wikipedia page incoming links.

Entity-TM (Han and Sun, 2012) models an entity as a distribution over mentions and words, and sets up a probabilistic generative process for the observed text.

Ours-NoH. To validate the advantage of incorporating hierarchical structure, we design a baseline that relies on entity embedding without entity hierarchy. That is, we obtain entity vectors by fixing the distance metric in Eq 4 as an identity matrix.

4.1.2 Results

Table 1 shows the performance of the competitors. Our algorithm using the entity hierarchy embedding gets 21% to 10% improvement in F1, and over 6% and 14% improvements in Precision and Recall, respectively. The CSAW model devises a set of entity features based on text content and link structures of Wikipedia pages, and combines them to measure relatedness. Compared to these hand-crafted features which are essentially heuristic and hard to verify, our embedding model induces semantic representations by optimizing a single well-defined objective. Note that the embedding actually also encodes the Wikipedia inter-page network, as we train on the entity-context pairs which are extracted from wiki links.

Methods	Precision	Recall	F1
CSAW	0.65	0.74	0.69
Entity-TM	0.81	0.80	0.80
Ours-NoH	0.78	0.85	0.81
Ours	**0.87**	**0.94**	**0.90**

Table 1: Entity linking performance

The Entity-TM model learns a representation for each entity as a word distribution. However, as noted in (Baroni et al., 2014), the counting-based distributional model usually shows inferior performance than context-predicting methods as ours. Moreover, in addition to the text context, our model integrates the entity hierarchical structure which provides rich knowledge of semantic relatedness. The comparison between *Ours* and *Ours-NoH* further reveals the effect of integrating the hierarchy in learning entity vectors. With entity hierarchy, we obtain more semantically meaningful representations that achieve 9% F1 improvement over entity vectors without hierarchical knowledge.

4.2 Entity Search

4.2.1 Setup

Dataset We use the dataset from INEX 2009 entity ranking track[2], which contains 55 queries. The golden standard results of each query contains a set of relevant entities each of which corresponds to a Wikipedia page.

Criteria We use the common criteria of precision@k, i.e., the percentage of relevant entities in the top-k results (we set $k = 10$), as well as precision@R where R is the number of golden standard entities for a query.

[1]http://www.cse.iitb.ac.in/soumen/doc/CSAW/Annot

[2]http://www.inex.otago.ac.nz/tracks/entity-ranking/entity-ranking.asp

Baselines We compare our algorithm with the following recent competitors.

Balog (Balog et al., 2011) develops a probabilistic generative model which represents entities, as well as the query, as distributions over both words and categories. Entities are then ranked based on the KL-divergence between the distributions.

K&K (Kaptein and Kamps, 2013) exploits Wikipedia entity hierarchy to derive the content of each category, which is in turn used to measure relatedness with the query categories. It further incorporates inter-entity links for relevance propagation.

Chen (Chen et al., 2014b) creates for each entity a context profile leveraging both the whole document (long-range) and sentences around entity (short-range) context, and models query text by a generative model. Categories are weighted based on the head words and other features. Our algorithm exploits a similar method for category matching.

Methods	Precision@10	Precision@R
Balog	0.18	0.16
K&K	0.31	0.28
Chen	0.55	0.42
Ours	**0.57**	**0.46**

Table 2: Entity search performance.

4.2.2 Results

Table 2 lists the entity search results of the competitors. Our algorithm shows superiority over the previous best performing methods. Balog constructs representations for each entity merely by counting (and smoothing) its co-occurrence between words and categories, which is inadequate to capture relatedness accurately. K&K leverages the rich resources in Wikipedia such as text, hierarchy, and link structures. However, the handcrafted features are still suboptimal compared with our learned representations.

Chen performs well by combining both long- and short-context of entities, as well as category lexical similarity. Our algorithm replaces its text matching component with a semantic enrichment step, i.e., grounding entity mentions in the query text onto KB entities. This augments the short query with rich background knowledge, facilitating accurate relatedness measure based on our high-quality entity embedding.

4.3 Qualitative Analysis

We qualitatively inspect the learned representations of the entity hierarchy. The results show that both the entity vectors and the category distance metrics capture meaningful semantics, and can potentially boost a wide range of applications such as recommendation and knowledge base completion.

Entity vectors Table 3 shows a list of target entities, and their top-4 nearest entities in the whole entity set or subsets belonging to given categories. Measuring under the whole set (column 2) results in nearest neighbors that are strongly related with the target entity. For instance, the nearest entities for “black hole” are “faster-than-light”, “event horizon”, “white hole”, and “time dilation”, all of which are concepts from physical cosmology and the theory of relativity. Similar results can be observed from other 3 examples.

Even more interesting is to specify a category and search for the most related entities under the category. The third column of Table 3 shows several examples. E.g., our model found that the most related Chinese websites to Youtube are “Tudou”, “56.com”, “Youku” (three top video hosting services in China), and “YinYueTai” (a major MV sharing site in China). The high-quality results show that our embedding model is able to discover meaningful relationships between entities from the complex entity hierarchy and plain text. This can be a useful feature in a wide range of applications such as semantic search (e.g., looking for movies about black hole), recommendation (e.g., suggesting TV series of specific genre for kids), and knowledge base completion (e.g., extracting relations between persons), to name a few.

Target entity	Most related entities	
black hole	**overall:** faster-than-light event horizon white hole time dilation	**American films:** Hidden Universe 3D Hubble (film) Quantum Quest Particle Fever
Youtube	**overall:** Instagram Twitter Facebook Dipdive	**Chinese websites:** Tudou 56.com Youku YinYueTai
Harvard University	**overall:** Yale University University of Pennsylvania Princeton University Swarthmore College	**businesspeople in software:** Jack Dangermond Bill Gates Scott McNealy Marc Chardon
X-Men: Days of Future Past (film)	**overall:** Marvel Studios X-Men: The Last Stand X2 (film) Man of Steel (film)	**children's television series:** Ben 10: Race Against Time Kim Possible: A Sitch in Time Ben 10: Alien Force Star Wars: The Clone Wars

Table 3: Most related entities under specific categories. “Overall” represents the most general category that includes all the entities.

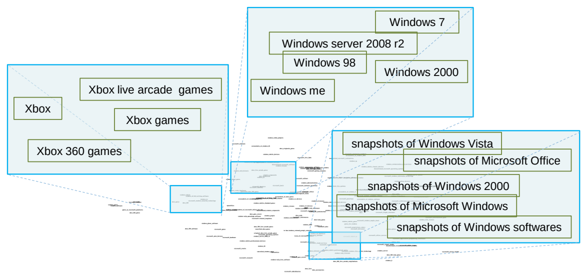

Figure 3: Distance metric visualization for the subcategories of the category "Microsoft". The t-SNE (Van der Maaten and Hinton, 2008) algorithm is used to map the high-dimensional (diagonal) matrixes into the 2D space.

Category distance metrics In addition to learning vector representations of entities, we also associate with each category a local distance metric to capture the features of individual category. As we restrict the distance metrics to be diagonal matrixes, the magnitude of each diagonal value can be viewed as how much a category is characterized by the corresponding dimension. Categories with close semantic meanings are expected to have similar metrics.

Figure 3 visualizes the metrics of all subcategories under the category "Microsoft", where we amplify some parts of the figure to showcase the clustering of semantically relevant categories. For instance, the categories of Microsoft Windows operating systems, and those of the Xbox games, are embedded close to each other, respectively. The results validate that our hierarchy embedding model can not only encode relatedness between leaf entities, but also capture semantic similarity of the internal categories. This can be helpful in taxonomy refinement and relation discovery.

5 Related Work

Distributed representation There has been a growing interest in distributed representation of words. Skip-gram model (Mikolov et al., 2013a) is one of the most popular methods to learn word representations. The model aims to find a representation for each word that is useful for predicting its context words. Word-context similarity is measured by simple inner product. A set of recent works generalizing the basic skip-gram to incorporate dependency context (Levy and Goldberg, 2014), word senses (Chen et al., 2014a), and multi-modal data (Hill and Korhonen, 2014). However, these work leverages limited structured knowledge. Our proposed method goes beyond skip-gram significantly such that we measures entity-context similarity under aggregated distance metrics of hierarchical category nodes. This effectively captures the structured knowledge. Another research line learn knowledge graph embedding (Lin et al., 2015; Wang et al., 2014; Bordes et al., 2013), which models entities as vectors and relations as some operations on the vector space (e.g., translation). These works aim at relation prediction for knowledge graph completion, and can be viewed as a supplement to the above that extracts semantics from plain text.

Utilizing hierarchical knowledge Semantic hierarchies are key sources of knowledge. Previous works (Ponzetto and Strube, 2007; Leacock and Chodorow, 1998) use KB hierarchies to define relatedness between concepts, typically based on path-length measure. Recent works (Yogatama et al., 2015; Zhao et al., 2011) learn representations through hierarchical sparse coding that enforces similar sparse patterns between nearby nodes. Category hierarchies have also been widely used in classification (Xiao et al., 2011; Weinberger and Chapelle, 2009). E.g., in (Verma et al., 2012) category nodes are endowed with discriminative power by learning distance metrics. Our approach differs in terms of entity vector learning and metric aggregation on DAG hierarchy.

6 Conclusion

In this paper, we proposed to learn entity hierarchy embedding to boost semantic NLP tasks. A principled framework was developed to incorporate both text context and entity hierarchical structure from large-scale knowledge bases. We learn a distance metric for each category node, and measure entity vector similarity under aggregated metrics. A flexible and efficient metric aggregation scheme was also developed to model large-scale hierarchies. Experiments in both entity linking and entity search tasks show superiority of our approach.

The qualitative analysis indicates that our model can be potentially useful in a wide range of other applications such as knowledge base completion and ontology refinement. Another interesting aspect of future work is to incorporate other sources of knowledge to further enrich the semantics.

Acknowledgments

This research is supported by NSF IIS-1218282, IIS-12511827, and IIS-1450545.

References

Krisztian Balog, Marc Bron, and Maarten De Rijke. 2011. Query modeling for entity search based on terms, categories, and examples. *TOIS*, 29(4):22.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Dont count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proc. of ACL*, volume 1, pages 238–247.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proc. of NIPS*, pages 2787–2795.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014a. A unified model for word sense representation and disambiguation. In *Proc. of EMNLP*, pages 1025–1035.

Yueguo Chen, Lexi Gao, Shuming Shi, Xiaoyong Du, and Ji-Rong Wen. 2014b. Improving context and category matching for entity search. In *Proc. of AAAI*.

Stéphane Clinchant and Florent Perronnin. 2013. Aggregating continuous word embeddings for information retrieval. page 100.

Gianluca Demartini, Tereza Iofciu, and Arjen P De Vries. 2010. Overview of the inex 2009 entity ranking track. In *Focused Retrieval and Evaluation*, pages 254–264. Springer.

Xianpei Han and Le Sun. 2012. An entity-topic model for entity linking. In *Proc. of EMNLP*, pages 105–115. Association for Computational Linguistics.

Felix Hill and Anna Korhonen. 2014. Learning abstract concept embeddings from multi-modal data: Since you probably can't see what i mean. In *Proc. of EMNLP*, pages 255–265.

Rianne Kaptein and Jaap Kamps. 2013. Exploiting the category structure of wikipedia for entity ranking. *Artificial Intelligence*, 194:111–129.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proc. of KDD*, pages 457–466. ACM.

Claudia Leacock and Martin Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proc. of ACL*, volume 2, pages 302–308.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proc. of AAAI*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*, pages 3111–3119.

Alexandre Passos, Vineet Kumar, and Andrew McCallum, 2014. *Lexicon Infused Phrase Embeddings for Named Entity Resolution*, pages 78–86. Association for Computational Linguistics.

Simone Paolo Ponzetto and Michael Strube. 2007. Knowledge derived from wikipedia for computing semantic relatedness. *JAIR*, 30(1):181–212.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proc. of IJCAI*, pages 448–453.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proc. of ACL*, pages 1555–1565.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *JMLR*, 9(2579-2605):85.

Nakul Verma, Dhruv Mahajan, Sundararajan Sellamanickam, and Vinod Nair. 2012. Learning hierarchical similarity metrics. In *Proc. of CVPR*, pages 2280–2287. IEEE.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. In *Proc. of EMNLP*.

Kilian Q Weinberger and Olivier Chapelle. 2009. Large margin taxonomy embedding for document categorization. In *Proc. of NIPS*, pages 1737–1744.

Lin Xiao, Dengyong Zhou, and Mingrui Wu. 2011. Hierarchical classification via orthogonal transfer. In *Proc. of ICML*, pages 801–808.

Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Y Ng. 2002. Distance metric learning with application to clustering with side-information. In *Proc. of NIPS*, pages 505–512.

Dani Yogatama, Manaal Faruqui, Chris Dyer, and Noah Smith. 2015. Learning word representations with hierarchical sparse coding. *Proc. of ICML*.

Jiajun Zhang, Shujie Liu, Mu Li, Ming Zhou, and Chengqing Zong. 2014. Bilingually-constrained phrase embeddings for machine translation. In *Proc. of ACL*.

Bin Zhao, Fei Li, and Eric P Xing. 2011. Large-scale category structure aware image categorization. In *Proc. of NIPS*, pages 1251–1259.

Entity Hierarchy Embedding: Supplementary Material

1 Proof of Theorem 1

In this section we prove Theorem 1 (Section 2.2):

Theorem 1. *$\forall h \in \mathcal{A}_e \cap \mathcal{A}_{e'}$, $h \in \mathcal{Q}_{e,e'}$ iff it satisfies the two conditions: (1) $|\mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})| \geq 2$; (2) $\exists a, b \in \mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})$ s.t. $t_a \neq t_b$.*

Recall that $\mathcal{Q}_{e,e'}$ is the set of common ancestors of entity e and e' that are turning nodes of any $e \to e'$ paths; $\mathcal{A}_e$ is the ancestor nodes of entity e (including e itself); for a node $h \in \mathcal{A}_e \cup \mathcal{A}_{e'}$, its critical node t_h is the nearest (w.r.t the length of the shortest path) descendant of h (including h itself) that is in $\mathcal{Q}_{e,e'} \cup \{e, e'\}$; $\mathcal{C}_h$ be the set of immediate child nodes of h.

Lemma 2. *$\forall h \in \mathcal{A}_e \cap \mathcal{A}_{e'}$, $t_h \in \mathcal{Q}_{e,e'}$.*

Proof. $h \in \mathcal{A}_e \cap \mathcal{A}_{e'} \Rightarrow (h \in \mathcal{A}_e) \wedge (h \in \mathcal{A}_{e'})$.

As $h \in \mathcal{A}_e$, there's path $e \to \cdots \to h$ where the consecutive nodes are (child, parent) pairs. Similarly, there exists path $h \to \cdots \to e'$ where the consecutive nodes are (parent, child) pairs. Denote the set of intersections of the two paths as $\mathcal{I}$. Because the two paths intersects at h, $\mathcal{I} \neq \phi$.

Note that the nodes in the intersection set are also in the path $h \to \cdots \to e'$, so we can sort the nodes in $\mathcal{I}$ according to the topological order in path $h \to \cdots \to e'$. Denote the topologically lowest node in $\mathcal{I}$ as t. As t is in the intersection set of two paths, there exists path $e \to \cdots \to t$ where the consecutive nodes are (child, parent) pairs and path $t \to \cdots \to e'$ where the consecutive nodes are (parent, child) pairs. If the two paths $e \to \cdots \to t$ and $t \to \cdots \to e'$ have any intersections except for t, then the intersection will be topologically lower than t, which contradicts the definition of t. So paths $e \to \cdots \to t$ and $t \to \cdots \to e'$ have intersection only at t, so t is a turning node. So $Q_{e,e'} \neq \phi$. According to the construction of t, t is a descendant of h, therefore $t_h \in Q_{e,e'}$. $\square$

We next prove Theorem 1.

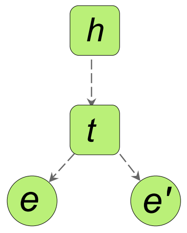

Figure 1: Illustration for Lemma 2. The topologically lowest intersection node is a turning node, which is also a descendant of h.

Proof. **Sufficiency**: Note that $e, e' \notin Q_{e,e'}$, we prove by enumerating possible situations: (i) $t_a = e, t_b = e'$, (ii) $t_a = e, t_b \in Q_{e,e'}$, (iii) $t_a, t_b \in Q_{e,e'}$. Case $t_a = e, t_b = e'$ is equivalent to case (i) if we swap e and e', and the cases $t_a = e', t_b \in Q_{e,e'}$, $t_a \in Q_{e,e'}, t_b = e(e')$ are equivalent to case (ii) if we swap the notations for variables a, b, e, e' properly. So the proof for cases (i), (ii) and (iii) is sufficient. An illustration of the cases is provided in Figure 2.

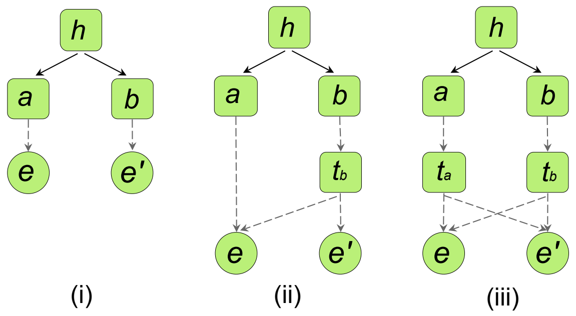

Figure 2: Three cases: (i) $t_a = e, t_b = e'$; (ii) $t_a = e, t_b \in Q_{e,e'}$; (iii) $t_a, t_b \in Q_{e,e'}$.

(i) $t_a = e, t_b = e'$:
As $t_a = e$, there's a path $e \to \cdots \to a \to h$ where the consecutive nodes are (child, parent) pairs. Similarly, there's a path $h \to b \to \cdots \to e'$ where the consecutive nodes are (parent, child) pairs. The above two paths only intersect at h, otherwise as a is the topologically highest node in path $e \to \cdots \to a \to h$ except for h, and e' is the topologically lowest node in path $h \to b \to \cdots \to e'$, e' would be a descendant of a. According to Lemma 2, $t_a \in Q_{e,e'}$, which contradicts $t_a = e$. So the two paths only intersect at h, and we can combine the two paths to construct a valid path $e \to \cdots \to a \to h \to b \to \cdots \to e'$, yielding h as a turning node.

(ii) $t_a = e, t_b \in Q_{e,e'}$:
$t_a = e \Rightarrow \exists e \to \cdots \to a \to h$ where the consecutive nodes are (child, parent) pairs. As $t_b \in Q_{e,e'}$, there exists path $h \to b \to \cdots \to t_b \to \cdots \to e'$ where the consecutive nodes are (parent, child) pairs. If the two paths $e \to \cdots \to a \to h$ and $h \to b \to \cdots \to t_b \to \cdots \to e'$ has any intersections except for h, then e' will be a descendant of a, thus $a \in \mathcal{A}_e \cup \mathcal{A}_{e'}$. According to Lemma 2, $t_a \in \mathcal{Q}_{e,e'}$, which contradicts the assumption that $t_a = e \notin \mathcal{Q}_{e,e'}$. So path $e \to \cdots \to a \to h \to b \to \cdots \to t_b \to \cdots \to e'$ is a valid path, yielding h as a turning node.

(iii) $t_a, t_b \in Q_{e,e'}$:
First of all, we prove that there exists path $e(e') \to \cdots \to t_a$ where the consecutive nodes are (child, parent) pairs and path $t_b \to \cdots \to e'(e)$ where the consecutive nodes are (parent, child) pairs and the two paths do not intersect with each other. If $t_b \to \cdots \to e'$ does not intersect with $e \to \cdots \to t_a$ (the existence of the paths is due to the definition of turning node), we've already got the construction. Otherwise, if $t_b \to \cdots \to e'$ intersects with $t_a \to \cdots \to e'$ at x before it intersects with $e \to \cdots \to t_a$, the path $e \to \cdots \to t_a$ and path $t_b \to \cdots \to x \to \cdots \to e'$ where the part $x \to \cdots \to e'$ is subpath of $t_a \to \cdots \to e'$ satisfies the above requirements. Similarly, if $t_b \to \cdots \to e'$ intersects with $e \to \cdots \to t_a$ at x before it intersects with $t_a \to \cdots \to e'$, the path $e' \to \cdots \to t_a$ and path $t_b \to \cdots \to x \to \cdots \to e$ where the part $x \to \cdots \to e$ is subpath of $t_a \to \cdots \to e$ satisfies the above requirements.

Using the above conclusion, if path $t_a \to \cdots \to a \to h$ (we choose the shortest path in the part $t_a \to \cdots \to a$ if there are multiple paths) intersects with $h \to b \to \cdots \to t_b$ (similarly, we choose the shortest path in the part $b \to \cdots \to t_b$) at any node except for h, we denote the topologically lowest one (w.r.t. path $h \to b \to \cdots \to t_b$) as x, then $t_a \to \cdots \to x$ has no intersection with $x \to \cdots \to t_b$ except for x, as any such intersection will be lower than x. So the path $e(e') \to \cdots \to t_a \to \cdots \to x \to \cdots \to t_b \to \cdots \to e'$ is a valid path, making x a turning node. As $t_a \neq t_b$, we have $(x \neq t_a) \vee (x \neq t_b)$. If $x \neq t_a$, x is closer to a as we've chosen the shortest path in part $t_a \to \cdots \to a$, contradicting the definition of t_a. Similarly, it is also impossible that $x \neq t_b$. So the two paths $t_a \to \cdots \to a \to h$ and $h \to b \to \cdots \to t_b$ do not intersect with each other.

Putting the above conclusions together, we can construct a valid path $e(e') \to \cdots \to t_a \to \cdots \to a \to h \to b \to \cdots \to t_b \to \cdots \to e'$, making h a turning node. Note that we also need to prove that the path $e \to \cdots \to t_a$ does not intersect with path $h \rightarrow b \rightarrow \cdots \rightarrow t_b$, which is analogous to the proof that path $t_a \rightarrow \cdots \rightarrow a \rightarrow h$ intersects with $h \rightarrow b \rightarrow \cdots \rightarrow t_b$ only at h.

Necessity: If h was a turning node, there would be a path $e \rightarrow \cdots a \rightarrow h \rightarrow b \rightarrow \cdots \rightarrow e'$, where the consecutive nodes before h are (child, parent) pairs and (parent, child) pairs after h, and we denote the two direct children of h in the path as a and b, in which a is ascendant of e (or e itself) and b ascendant of e' (or e' itself). So $|\mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})| \geq |\{a, b\}| = 2$.

Then we prove that $\exists a, b \in \mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})$ s.t. $t_a \neq t_b$ by contradiction. Suppose that $\forall a, b \in \mathcal{C}_h \cap (\mathcal{A}_e \cup \mathcal{A}_{e'})$ we have $t_a = t_b$. Using the same notation as above, denote a, b as the direct children of h in the path $e \rightarrow \cdots a \rightarrow h \rightarrow b \rightarrow \cdots \rightarrow e'$ which makes h a turning node. W.l.o.g. we consider two cases: $t_a = t_b = e$, and $t_a = t_b \in Q_{e,e'}$. For the first case, $t_b = e \Rightarrow e$ is a descendant of b, and from the definition of b we know that e' is a descendant of b, so $b \in \mathcal{A}_{e,e'}$. From Lemma 2, $t_b \in Q_{e,e'}$, contradicts $t_b = e$.

For the second case $t_a = t_b \in Q_{e,e'}$, denote $t_{a,b} = t_a = t_b$. As h is a turning node, there exists a path $e \rightarrow \cdots a \rightarrow h \rightarrow b \rightarrow \cdots \rightarrow e'$. Then the subpaths $e \rightarrow \cdots \rightarrow a$ and $b \rightarrow \cdots \rightarrow e'$ has no common nodes according to the definition of a path. So at least one of the subpaths does not include $t_{a,b}$, w.l.o.g assume subpath $b \rightarrow \cdots \rightarrow e'$ does not include $t_{a,b}$. As $t_{a,b}$ is a descendant of b, there exists paths $b \rightarrow \cdots \rightarrow t_{a,b}$, and we pick up the shortest one. We'll prove that there's no intersection between path $b \rightarrow \cdots \rightarrow t_{a,b}$ and path $b \rightarrow \cdots \rightarrow e'$: Assume that there exists such intersections, and denote the topologically lowest intersection (w.r.t. path $b \rightarrow \cdots \rightarrow t_{a,b}$) as x, then as we've assumed that subpath $b \rightarrow \cdots \rightarrow e'$ does not include $t_{a,b}$, we have $x \neq t_{a,b}$. Then we can prove that x is a turning node: If subpath $x \rightarrow \cdots \rightarrow e'$ does not intersect with path $e \rightarrow \cdots \rightarrow t_{a,b}$, then we can construct a path $e \rightarrow \cdots \rightarrow t_{a,b} \rightarrow x \rightarrow \cdots \rightarrow e'$, yielding x as a turning node. Otherwise, if $x \rightarrow \cdots \rightarrow e'$ intersects with $t_{a,b} \rightarrow \cdots \rightarrow e'$ before it intersects with $e \rightarrow \cdots \rightarrow t_{a,b}$ or it does not intersect with $e \rightarrow \cdots \rightarrow t_{a,b}$ at all, then denote the intersection node as y, we have a valid path $e \rightarrow \cdots \rightarrow t_{a,b} \rightarrow x \rightarrow \cdots \rightarrow y \rightarrow \cdots \rightarrow e'$ in which the part $y \rightarrow \cdots \rightarrow e'$ is a subpath of $t_{a,b} \rightarrow \cdots \rightarrow e'$, yielding x as a turning node. By similar construction, we can prove that if if $x \rightarrow \cdots \rightarrow e'$ intersects with $e \rightarrow \cdots \rightarrow t_{a,b}$ before it intersects with $t_{a,b} \rightarrow \cdots \rightarrow e'$ or it does not intersect with $t_{a,b} \rightarrow \cdots \rightarrow e'$ at all, x is also a turning node. However, x is nearer to b than $t_{a,b}$, which contradicts the definition of $t_{a,b}$. So we have proved that there's no intersection between path $b \rightarrow \cdots \rightarrow t_{a,b}$ and path $b \rightarrow \cdots \rightarrow e'$. Then we can prove that $t_{a,b} = b$: If path $b \rightarrow \cdots \rightarrow e'$ does not intersect with $e \rightarrow \cdots \rightarrow t_{a,b}$, then a valid path $e \rightarrow \cdots \rightarrow t_{a,b} \rightarrow \cdots \rightarrow b \rightarrow \cdots \rightarrow e'$ will make b a turning node, so $t_{a,b} = b$. Otherwise, if $b \rightarrow \cdots \rightarrow e'$ intersects with $t_{a,b} \rightarrow \cdots \rightarrow e'$ at z before it intersects with $e \rightarrow \cdots \rightarrow t_{a,b}$, then a valid path $e \rightarrow \cdots \rightarrow t_{a,b} \rightarrow \cdots \rightarrow b \rightarrow \cdots \rightarrow z \rightarrow \cdots \rightarrow e'$ where the part $z \rightarrow \cdots \rightarrow e'$ is subpath of $t_{a,b} \rightarrow e'$ will make b a turning node. If $b \rightarrow \cdots \rightarrow e'$ intersects with $e \rightarrow \cdots \rightarrow t_{a,b}$ at z before it intersects with $t_{a,b} \rightarrow \cdots \rightarrow e'$, then through similar construction we can also prove $t_{a,b} = b$. This contradicts the assumption that subpath $b \rightarrow \cdots \rightarrow e'$ does not include $t_{a,b}$, so the second case is also impossible. $\square$