Data Science and Pattern Recognition
Ubiquitous International
 ISSN 2520-4165
Volume **4**, Number **1**, April **2020**

Approximate Error Estimation based Incremental Word Representation Learning

Hao Peng

School of Cyber Science and Technology
Beihang University, Beijing
penghao@act.buaa.edu.cn

Lin Liu

School of Computer Science and Engineering
Beihang University, Beijing
liulin2018@buaa.edu.cn

Liya Ma

National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing
mly@cert.org.cn

Weiqin Zhao

School of Computer Science and Engineering
Beihang University, Beijing
zhaoweiqin@buaa.edu.cn

Hongyuan Ma

National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing
mahongyuan@foxmail.com

Long Yuntao

Institutes of Science and Development
Chinese Academy of Sciences, Beijing
yuntaol@casipm.ac.cn

Abstract. In recent years, neural network based language representation learning model and word embedding technology have been successfully applied in variants of natural language mining tasks. In this paper, we present an incremental hierarchical probabilistic neural language learning algorithm to train evolving word vectors based on hierarchical softmax approximation. We split the incremental word representation learning objective function to reserved term and updated term separately, and factorize the incremental objective function into the hierarchical softmax function. A novel stochastic gradient based approximately incremental method is proposed to update all the inherited word vectors and inherited parameter vectors, to reduce vector errors of inheritance. Theoretical analysis of the bound and convergence is also provided for the approximate incremental objective function. Extensive experiments show that the proposed approximately incremental word embedding method can save a lot of time, and even the maximum acceleration ratio is 30 times. Both word similarity/relatedness tasks, medical domain entity and relation extraction tasks, and temporal word evolution are evaluated as benchmarks for the word correctness and efficiency of the incremental learning word vectors.
Keywords: Incremental learning, Word representation, Hierarchical softmax, Error bound.

1. **Introduction.** Neural network based language models (NNLM) and word embedding technologies [2, 3, 4, 11, 12, 15, 16, 18, 19, 32, 34], as the most important innovation in unsupervised representation learning era, has deeply promoted the development of natural language processing and understanding technologies. Neural network based word representation learning technologies have been used to so much natural language mining tasks, including spam detection word similarity/relatedness and word analogy [15, 16, 20, 26], and working as raw features for part-of-speech tagging, chunking, named entity recognition, text classification, etc. [4, 9, 10, 21, 22, 24, 31, 32, 33].

However, most of existing approaches are either not suitable for scalable incremental learning [1, 8, 25], or are not obvious in time speedup and theoretical guarantee for the complexity of evolving algorithms [13, 14, 20]. There are many real-time text mining applications, such as online social topic and users text processing, and incremental or evolving word representations are required. New word embedding can reflect the evolving word semantic. Therefore, we present an incremental hierarchical probabilistic neural language learning algorithm to train evolving word vectors based on hierarchical softmax approximation.

Given streaming corpora, we consider to design approximately incremental word representation learning function. Different the latest Bert [5], we employ the popular word2vec model due to its simplicity and high efficiency, and is the implementation of mainstream neural network based word representation learning models [15, 16]. In the word2vec tool, there are two neural network frameworks, including Continuous Bag-of-Words (CBOW) based models and Skip-gram based models. While CBOW averages each word's context to represent the word, the Skip-gram employs each word to predict its context. Both models have the problem of indexing and querying the representation of context of a word. To speedup the training, word2vec used two key techniques called hierarchical softmax and negative sampling [15, 16]. Hierarchical softmax function was first proposed by Mnih and Hinton [18] to speedup the training cost from $O(n^2)$ to $O(nlog(n))$, where a word frequency based hierarchical tree is constructed. While the negative sampling is designed from noise contrastive estimation technology [7], randomly negative samples of the words are sampled to help to distinguish the positive sample. It's widely accepted that hierarchical softmax function performs better for infrequent words while negative sampling performs better for frequent words [16], since it can speed up the training cost from $O(n^2)$ to $O(n)$. Hierarchical softmax function needs to build a Huffman tree over the whole vocabulary in a corpus, and the leaf nodes representing rare words will inevitably inherit their ancestors' vector in the tree, which can be affected by other frequent words in the updated corpus. So, the hierarchical softmax is choose due to rare words representation learning, which can benefit the further detecting semantic shifts and usage of words over incremental text data.

When designing hierarchical softmax to streaming word embeddings, there are preprocessing steps to count word frequencies and the determine two hierarchical binary tree of words. The re-built hierarchical binary tree should reflect the change of words distribution. In hierarchical softmax based word representation learning models, its use the word frequency to construct the binary tree over the vocabulary since Huffman tree is the shortest code to index words [7], and reduces the computational complexity of calculating likelihood function from $\mathcal{O}(n^2)$ to $\mathcal{O}(nlog(n))$. When streaming corpora are increasing, word frequencies change in increment scene, and huffman tree structure will be sensitively changed [23]. The structure of Huffman tree directly determines the objective function of hierarchical softmax model based Word2vec [16, 17]. In order to reduce the impact of word frequency change on binary tree structure, we propose a compatible weighted contextual frequency aggregated (WCFA) hierarchical binary tree to implement the NNLMs

and our incremental framework. Conceptually, the WCFA tree structure can keep more stable of the invariance by contextual distribution. For both Huffman and WCFA tree based incremental models, the paths from root to leaves of the old tree are retained, and then a prefix matching between the old tree and the new tree are performed to inherit parameter vectors. When updating the vectors for the matched ancestors of a leaf, the original algorithm of CBOW or skip-gram in word embedding based on stochastic gradient ascent of log-likelihood are running. When updating the vectors for the different ancestors, we update the old tree with stochastic gradient descent, while update the new tree with stochastic gradient ascent to reduce the inherited vectors error. In this way, we only modify all the nodes that need to be updated while retaining all the other nodes as the same as the ones trained based on the old corpus. So, we provide a thorough theoretical analysis to demonstrate its validity. Since the update training is independent for all the internal nodes, the proposed framework is parallel.

Extensive experiments are performed to verify that the proposed framework can achieve almost the same experimental performance of CBOW and Skip-gram models as fully restarted ones. We also check both mean square error of individual vectors, word similarity/relatedness and word evolution, and medical entity and relation extraction to prove the correctness of our models.

The contributions of this work are highlighted as follows:

• We propose an incremental hierarchical neural language learning framework which can be applied in the common CBOW and Skip-gram models.

• We formulate the unsupervised incremental objective function and give the detail definition of symbols.

• We give the bound and convergence analysis for the objective difference.

• The incremental training time can be up to 30 times speedup, and experiments on word quality, similarity/relatedness, word evolution, and medical entity and relation extraction show that our approach performs fair or better in comparison with the time-consuming global learning.

The remainder of the paper is organized as follows. We first give the words and parameters vector's initialization and inheritance, and formulate our incremental framework in incremental training section. In theoretical analysis section, we provide the mathematical details of incremental objective difference, corresponding bound analysis of first and second order moments. Then we show our experiments on how our approach is correct and efficient on the training and downstream tasks in experiments section. In conclusion section, we give the conclusion of this paper.

2. **Incremental Training.**

In incremental scenes, we should re-build the hierarchical tree, e.g., the Huffman tree or WCFA tree, based on the updated corpus, as shown in Figure 1.

2.1. **Notations and Definitions.**

In this section, we first give the detailed notations and definitions of basic concepts for incremental neural language models for word representations.

Basic Concepts. Conceptually, given a original hierarchical binary tree (Huffman or WCFA tree) $\mathcal{T}$, we denote the updated hierarchical binary tree as $\mathcal{T}'$. The incremental hierarchical neural language models problem is to rebuild renewed hierarchical binary tree, and retain all word vectors and much parameter vectors as much as possible. More specific, all symbols (functions) and explanations are summarized in Table 1.

TABLE 1. Definition of main Symbols

Symbol	Definitions
$\mathcal{J}_{\text{CBOW}}$	The objective function of CBOW model for the original corpus
$\mathcal{J}_{\text{SG}}$	The objective function of Skip-gram model for the original corpus
$\mathcal{J}'_{\text{CBOW}}$	The incremental objective function of CBOW model by our algorithm
$\mathcal{J}'_{\text{SG}}$	The incremental objective function of Skip-gram model by our algorithm
$\mathcal{J}*_{\text{CBOW}}$	The incremental objective function of CBOW model in theory
$\mathcal{J}*_{\text{SG}}$	The incremental objective function of Skip-gram model in theory
$v(w)$	The word vector trained by original corpus
θ_i	The i-th parameter vector trained by original corpus
$v'(w)$	The word vector of word w in increment
θ'_i	The i-th parameter vector in increment
$\theta_i^{w_j}$	The i-th parameter vector for word w_j at i's position
$\Omega(w)$	The internal vectors set of word w in tree $\mathcal{T}$
$\Omega'(w)$	The internal vectors set of word w in tree $\mathcal{T}'$
Θ	The common prefix Huffman matched substring set of parameter vectors
$\Theta(w)$	The common prefix Huffman matched substring set of parameter vectors for any word w
Υ	The inherited parameter vectors set having not matched the prefix Huffman
$\Upsilon(w)$	The inherited parameter vectors set having not matched the prefix Huffman for any word w
Φ	The inherited parameter vectors set from old tree
$\Phi(w)$	The inherited parameter vectors set from old tree for any word w
Ψ	The unmatched prefix Huffman parameter vectors set from old tree in tree $\mathcal{T}'$
$\Psi(w)$	The unmatched prefix Huffman parameter vectors set from old tree for any word w in tree $\mathcal{T}'$
L^w	The Huffman code length of any word w in $\mathcal{T}$
L'^w	The Huffman code length of any word w in $\mathcal{T}'$
L'^w_C	The common Huffman code length of any word w between $\mathcal{T}$ and $\mathcal{T}'$
L'^w_I	The length of inherited internal parameter vectors of any word w in $\mathcal{T}$

2.2. **Node Initialization and Inheritance.** Suppose we have the old corpus $\mathcal{W}$ and the new corpus $\mathcal{W}' = \mathcal{W} \cup \Delta\mathcal{W}$. We can build the binary trees $\mathcal{T}$ and $\mathcal{T}'$ from both corpora respectively, as shown in Figure 1.

For the leaf nodes, if the word has been observed in the old corpus, we simply initialize the vector as the vector that has been trained. If the word is a new word, we randomly initialize it as a random vector:

$$v'(w) = \begin{cases} v(w), & w \in \mathcal{W} \\ \text{random}, & w \notin \mathcal{W} \end{cases}, \tag{1}$$

where $v(w)$ and $v'(w)$ are the vectors of word w for old and new trees respectively.

For the internal nodes, the Huffman code change of a word may affect only partial change of the path for that word. Along the path, each internal node owns one parameter vector. We distinguish the parameter vector $\theta_i^{w_1}$'s for word w_1 at i's position and $\theta_i^{w_2}$'s

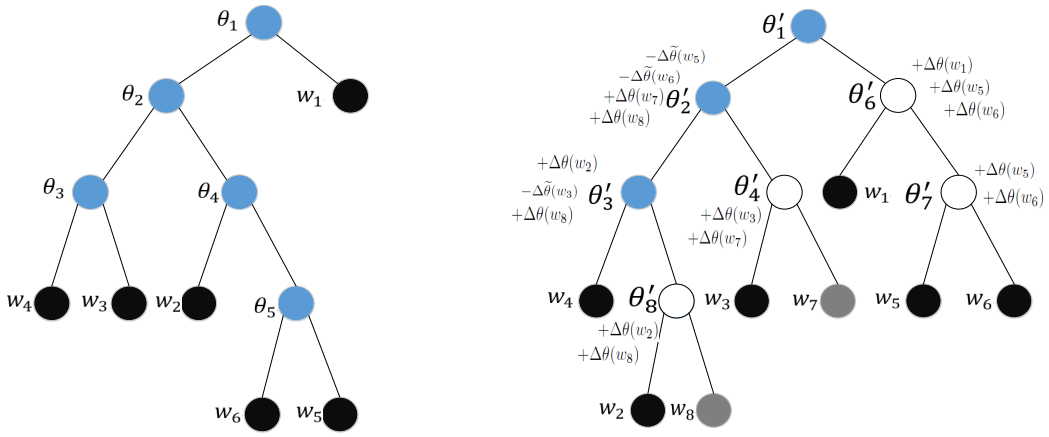

FIGURE 1. Original and Updated Hierarchical Binary Tree

for word w_2 at i's position. When they are at the same i's position in the tree, then

$$\theta_i = \theta_i^{w_1} = \theta_i^{w_2} \quad , \tag{2}$$

For example, in the left figure of Figure 1, $\theta_3^{w_2} = \theta_3^{w_6} = \theta_4$. Moreover, a word w_2 encoded as "010" in the old tree $\mathcal{T}$ may be changed as "0010" in the new tree $\mathcal{T}'$. In this case, the matched prefix "0" remains the same, and corresponding matched internal nodes θ_1, θ'_1 and θ_2, θ'_2 share the same structure in the new tree $\mathcal{T}'$ as the old tree $\mathcal{T}$. Although θ_3 and θ'_3 are not matched in word w_2, there is a word w_4 which ensures all corresponding internal nodes matching. So, $\theta'_1 = \theta_1, \theta'_2 = \theta_2 \quad and \quad \theta'_3 = \theta_3$. To make the prefix explicit, we denote L^w and L'^w as lengths of the code of word w in old and new trees respectively, e.g., $L^{w_2} = 3$ in $\mathcal{T}$ and $L'^{w_2} = 4$ in $\mathcal{T}'$ in Figure 1. We gather all internal vectors of leaf node w as set $\Omega(w) = \{\theta_i^w | i = 1, \cdots, L^w\}$ in tree $\mathcal{T}$. For the matched prefix, we use the existing parameter vector θ_i^w as the initialization for the new tree, while for the mismatched codes, we initialize them as zero vectors, as following:

$$\theta'_i = \begin{cases} \theta_i, & d'^w_i = d^w_i \\ 0, & \text{otherwise} \end{cases} \quad , \tag{3}$$

where d'^w_i and d^w_i are the Huffman codes of internal nodes in the new and old trees respectively. Thus, the inherited internal nodes of leaf node w can be divided into common prefix matched substring $\Theta(w) = \{\theta^w_{i-1} | i = 2, \cdots, L'^w_C + 1\}$ and other nodes $\Upsilon(w) = \{\theta^w_{i-1} | i = L'^w_C + 2, \cdots, L'^w_I\}$ in $\mathcal{T}$, where L'^w_C is length of common prefix matched between old and new trees, and $L'^w_I\}$ is length of inherited internal parameter vectors of any word w in $\mathcal{T}$. For any word w in $\mathcal{T}$, the set of inherited internal parameter vectors can be formulation as:

$$\Phi(w) = \Theta(w) + \Upsilon(w), \tag{4}$$

Figure 1 also shows examples of inherited nodes and new nodes. More specifically, for any word w in $\mathcal{T}'$, the set of internal parameter vectors can be formulation as following:

$$\Omega'(w) = \Theta(w) + \Psi(w), \tag{5}$$

where $\Psi(w) = \{\theta^w_{i-1} | i = L'^w_C + 2, \cdots, L'^w\}$. Such as, $\Phi(w_3) = \{\theta_1, \theta_2, \theta_3\}$, $\Theta(w_3) = \{\theta_1, \theta_2\}$, $\Upsilon(w_3) = \{\theta_3\}$, $\Omega'(w) = \{\theta_1, \theta_2, \theta'_4\}$, $\Psi(w_3) = \{\theta'_4\}$ and $\Phi(w_2) = \{\theta_1, \theta_2\}$, $\Theta(w_2) = \{\theta_1, \theta_2\}$, $\Upsilon(w_2) = \emptyset$, $\Omega'(w_2) = \{\theta'_1, \theta'_2, \theta'_3, \theta'_8\}$, $\Psi(w_2) = \{\theta'_3, \theta'_8\}$ in Figure 1.

2.3. **Model Updates.** Given the inherited nodes and the new nodes by comparing the old and new trees, we also decompose the log-likelihood functions for CBOW and Skip-gram models based on the prefix matching results.

For CBOW model, we consider to factorize the log-likelihood function by aggregating the cost term $\ell(w, i)$.

$$\mathcal{J}'_{\text{CBOW}} = \sum_{w\in\mathcal{W}} \{ \sum_{i=2}^{L'^w_C+1} - \sum_{i=L'^w_C+2}^{L'^w_I} + \sum_{i=L'^w_C+2}^{L'^w} \} \ell(w, i) + \sum_{w\in\Delta\mathcal{W}} \sum_{i=2}^{L'^w} \ell(w, i), \tag{6}$$

Here we first split the training data to be $\mathcal{W}' = \mathcal{W} \cup \Delta\mathcal{W}$. For the words in $\mathcal{W}$, we factorize it based on the common Huffman codes and distinct codes. $\sum_{i=2}^{L'^w_C+1}$ sums the codes that share the prefix with the old tree by word w. $\sum_{i=L'^w_C+2}^{L'^w_I}$ sums inherited internal vector codes by other words in the new tree. $\sum_{i=L'^w_C+2}^{L'^w}$ sums the zero initialization internal vector codes in the new tree. For the words in $\Delta\mathcal{W}$, we follow the original objective function of CBOW model.

Similarly, for Skip-gram model, the objective is:

$$\begin{aligned} \mathcal{J}'_{\text{SG}} = & \sum_{w\in\mathcal{W}} \sum_{u\in\mathcal{C}(w)} \{ \sum_{j=2}^{L'^u_C+1} - \sum_{j=L'^u_C+2}^{L'^u_I} + \sum_{j=L'^u_C+2}^{L'^u} \} \ell(w, u, j) \\ & + \sum_{w\in\Delta\mathcal{W}} \sum_{u\in\mathcal{C}(w)} \sum_{j=2}^{L'^u} \ell(w, u, j), \end{aligned} \tag{7}$$

To train a new set of word vectors, originally we need to re-scan and re-train the whole corpus $\mathcal{W}' = \mathcal{W} \cup \Delta\mathcal{W}$ based on stochastic gradient ascent method. Given the above factorization analysis of the objective function, we found that for the old corpus $\mathcal{W}$, we can apply the following trick to save a lot of training time.

Our goal is to find a new set of (local) optimal internal node vectors θ'^w_i and word vectors $v'(w)$ to approximate re-training. We first make an assumption that all the word vectors $v(w)$ are already (local) optimal and then further calibrate them. Then we perform stochastic gradient based optimization based on $\mathcal{W}$ and $\mathcal{W}'$ respectively. Moreover, the learning rate can be updated as

$$\eta' = \eta_0 (1 - \frac{\kappa}{\phi' + 1}), \tag{8}$$

where ϕ' represents the total number of tokens in the new corpus $\mathcal{W}'$.

When scanning the old corpus $\mathcal{W}$, we can update all the parameters while fixing the word vectors.[1] Denote all the inherited parameters related to w as $\Theta(w) \cup \Upsilon(w)$. We can see that for the $\Theta(w)$, the training process is the same as the original CBOW and Skip-gram models. For $\Upsilon(w)$, since the tree structure has changed, for a certain internal node, some of the leaves (words) are still under it while the others has moved out. For example, in Figure 1, the word w_6 is now under θ'_6 and θ'_7 but moved out of θ_2, θ_4, and θ_5. To recover the parameters in the new tree so that the incremental training is as similar as the fully re-trained model when seeing w_6, we need to subtract the inherited gradients of θ'_2 related to w_6, and add the gradients to θ_6 and θ_7 (here θ'_4 is initialized as zero and θ'_5 is not inherited). Formally, for a word w, the CBOW update rule for the parameters in the new path from root to this word is as follows.

[1]We can also update the word vectors in the meantime, however, it will introduce more computational cost.

If $w \in \mathcal{T}', \theta'^{w}_{i-1} \in \Theta(w), i \in \{2, \dots L'^{w}_{C} + 1\}$, we have:

$$\theta'^{w}_{i-1} := \theta'^{w}_{i-1}, \tag{9}$$

If $w \in \mathcal{T}', \theta'^{w}_{i-1} \in \Psi(w), i \in \{L'^{w}_{C} + 2, \dots, L'^{w}\}$, we have:

$$\theta'^{w}_{i-1} := \theta'^{w}_{i-1} + \eta'[1 - d^{w}_{i} - \sigma(X^{T}_{w}\theta'^{w}_{i-1})]X_{w}, \tag{10}$$

If $w \in \mathcal{T}, \theta'^{w}_{i-1} \in \Upsilon(w), i \in \{L'^{w}_{C} + 2, \dots, L'^{w}_{I}\}$, we have:

$$\theta'^{w}_{i-1} := \theta'^{w}_{i-1} - \eta'[1 - d^{w}_{i} - \sigma(X^{T}_{w}\theta'^{w}_{i-1})]X_{w}, \tag{11}$$

Here retain the common prefix nodes, perform stochastic gradient ascent to the new nodes, and perform approximately stochastic gradient descent to the old sub-tree path related to the word w. More precisely, we replace the activation $\sigma(X^{T}_{w}\theta'^{w}_{i-1})$ in $\Upsilon(w)$ by $\sigma(X^{T}_{w}\theta'^{w}_{i-1})$ in $\Psi(w)$. η' is the new learning rate.

For the update corpus $\Delta\mathcal{W}$, we simply perform the stochastic gradient ascent for both parameters and word vectors (e.g., in Eq. (6)). Thus, we can see that the most computational cost is saved by not updating word vectors in old corpus, and partially saved by adjusting (partially not updating) the parameters in old corpus. An illustration is shown in Figure 1. From the figure we can see that, we adjust the internal node to approximate the process of complete re-training.

Similarly for Skip-gram, the update rule for the parameters is as follows.

If $u \in \mathcal{T}', \theta'^{u}_{j-1} \in \Theta(u), j \in \{2, \dots, L'^{u}_{C} + 1\}$, we have:

$$\theta'^{u}_{j-1} := \theta'^{u}_{j-1}, \tag{12}$$

If $u \in \mathcal{T}', \theta'^{u}_{j-1} \in \Psi(u), j \in \{L'^{u}_{C} + 2, \dots, L'^{u}\}$, we have:

$$\theta'^{u}_{j-1} := \theta'^{u}_{j-1} + \eta'[1 - d^{u}_{j} - \sigma(v(w)^{T}\theta'^{u}_{j-1})]v(w), \tag{13}$$

If $u \in \mathcal{T}, \theta^{u}_{j-1} \in \Upsilon(u), j \in \{L'^{u}_{C} + 2, \dots, L'^{v}_{I}\}$, we have:

$$\theta'^{u}_{j-1} := \theta'^{u}_{j-1} - \eta'[1 - d^{u}_{j} - \sigma(v(w)^{T}\theta'^{u}_{j-1})]v(w), \tag{14}$$

For above equations, $u \in \mathcal{C}(w)$.

3. Theoretical Analysis.

In this section, we present the theoretical analysis of our incremental learning algorithm using CBOW and Skip-gram models. To make sure whether incremental CBOW or Skip-gram models can learn word embeddings as well as the global rebuilding batch counterpart.

The theoretical analysis begins by examining the difference between the objectives optimized by approximately incremental and global rebuilding batch of CBOW and Skip-gram models. Then probabilistic properties of their difference are investigated to demonstrate the relationship between incremental and global learning models.

3.1. Objective Difference and Bound Analysis.

As discussed in above, the difference between two objectives can be given as:

$$\Delta\mathcal{J}_{\text{CBOW}} = \sum_{w \in \mathcal{W}} \sum_{i=L'^{w}_{C}+2}^{L'^{w}_{I}} [\ell(w, i) - \ell(\widetilde{w}, i)], \tag{15}$$

where the initialization of w is zero, and $\sum_{w \in \mathcal{W}} \sum_{i=L'^{w}_{C}+2}^{L'^{w}_{I}} (\ell(w, i))$ represents the original training process of word and parameter vectors. While the initialization of $\widetilde{w}$ is inherited word vector, and $\sum_{w \in \mathcal{W}} \sum_{i=L'^{\widetilde{w}}_{C}+2}^{L'^{\widetilde{w}}_{I}} (\ell(\widetilde{w}, i))$ represents the approximately stochastic gradient descent process for reducing the inheritable error in $\Upsilon(w)$. Since the approximation

shares the same $1-d_j^w$ in $\ell(\widetilde{w},i)$ and $\ell(w,i)$, the difference between two objectives can be updated as:

$$\Delta\mathcal{J}_{\text{CBOW}} = \sum_{w\in\mathcal{W}}\sum_{i=L'^w_C+2}^{L'^w_I}\{(1-d_i^w)\cdot log\frac{[\sigma(x_w^T\theta_{i-1}^w)]}{[\sigma(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w)]} + d_i^w\cdot log\frac{[1-\sigma(x_w^T\theta_{i-1}^w)]}{[1-\sigma(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w)]}\}, \tag{16}$$

Since $\sigma(x_w^T\theta_{i-1}^w) = 1/(1+\exp(-x_w^T\theta_{i-1}^w))$ and $x_w^T\theta_{i-1}^w, \widetilde{x}_w^T\widetilde{\theta}_{i-1}^w \in [-6,6]^2$. In (w,i), the difference between two objectives can be given as

$$\Delta\mathcal{J}_{\text{CBOW}}(w,i) = \{(1-d_i^w)\cdot log\frac{1+e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w}}{1+e^{-x_w^T\theta_{i-1}^w}} + d_i^w\cdot log\frac{e^{-x_w^T\theta_{i-1}^w}(1+e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w})}{e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w}(1+e^{-x_w^T\theta_{i-1}^w})}\} \tag{17}$$

We assume that the difference $\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w$ and $x_w^T\theta_{i-1}^w$ can be bounded by ϵ, and can be formalized as

$$|\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w - x_w^T\theta_{i-1}^w| < \epsilon, \tag{18}$$

In Eqs. (17), according to the Lagrange's Mean Value Theorem, we have $\frac{1+e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w}}{1+e^{-x_w^T\theta_{i-1}^w}} - 1 = \frac{e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w} - e^{-x_w^T\theta_{i-1}^w}}{1+e^{-x_w^T\theta_{i-1}^w}} < \frac{-e^{-\overline{x}_w^T\overline{\theta}_{i-1}^w}(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w - x_w^T\theta_{i-1}^w)}{1+e^{-6}}$, where the $\overline{x}_w^T\overline{\theta}_{i-1}^w$ is a solution meeting $e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w} - e^{-x_w^T\theta_{i-1}^w} = -e^{-\overline{x}_w^T\overline{\theta}_{i-1}^w}(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w - x_w^T\theta_{i-1}^w)$.

So, we have

$$\frac{1+e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w}}{1+e^{-x_w^T\theta_{i-1}^w}} < 1 - \frac{e^6}{1+e^{-6}}\epsilon, \tag{19}$$

We assume that the $f(X) = \frac{e^{-X}}{1+e^{-X}}$. Since $f'(X) = -\frac{e^{-X}}{(1+e^{-X})^2} < 0$ and belongs to uniformly continuous function, the minimum value $f_{max}(X) = f(-6)$. Further, we have $f''(X) = \frac{e^{-X}(1-e^{-2X})}{(1+e^{-X})^4}$, and $f'_{max}(X) = f'(0)$. In Eqs. (17), we have $\frac{e^{-x_w^T\theta_{i-1}^w}(1+e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w})}{e^{-\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w}(1+e^{-x_w^T\theta_{i-1}^w})} = \frac{f(x_w^T\theta_{i-1}^w)}{f(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w)}$. Similarly, according to the Lagrange's Mean Value Theorem, we have

$$\frac{f(x_w^T\theta_{i-1}^w)}{f(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w)} \le 1 + \frac{1}{f(-6)}f'(\dot{x}_w^T\dot{\theta}_{i-1}^w)(x_w^T\theta_{i-1}^w - \overline{x}_w^T\overline{\theta}_{i-1}^w) \quad \le 1 + \frac{f'(0)}{f(-6)}\epsilon \le 1 - \frac{e^6+1}{4e^6}\epsilon, \tag{20}$$

where the $\dot{x}_w^T\dot{\theta}_{i-1}^w$ is a solution that meets $f(x_w^T\theta_{i-1}^w) - f(\widetilde{x}_w^T\widetilde{\theta}_{i-1}^w) = f'(\dot{x}_w^T\dot{\theta}_{i-1}^w)(x_w^T\theta_{i-1}^w - \overline{x}_w^T\overline{\theta}_{i-1}^w)$.

As we fix the inherited word vector and iterate the parameter vector, we have

$$\widetilde{x}_w^T = x_w^T \quad and \quad \widetilde{\theta}_{i-1}^w - \theta_{i-1}^w < \frac{\epsilon}{x_w^T}, \tag{21}$$

so the difference between two objectives can be bounded as

$$\Delta\mathcal{J}_{\text{CBOW}} < \sum_{w\in\mathcal{W}}\sum_{i=L'^w_C+2}^{L'^w_I}(1-d_i^w)\cdot log(1-\frac{e^6}{1+e^{-6}}\epsilon) + d_i^w\cdot log(1-\frac{e^6+1}{4e^6}\epsilon), \tag{22}$$

Similarly for Skip-gram model, the difference between two objectives can be given and bounded as

$$\Delta\mathcal{J}_{\text{SG}} = \sum_{w\in\mathcal{W}}\sum_{u\in\mathcal{C}(w)}\{\sum_{j=L'^u_C+2}^{L'^u_I}\}[\ell(w,u,j) - \ell(\widetilde{w},u,j)], \tag{23}$$

[2]There is a piecewise optimized process for $\sigma(x)$ in [15][11][17], and the independent variables are limited to [-6,6] in activation function.

$$\Delta\mathcal{J}_{\mathrm{SG}} < \sum_{w\in\mathcal{W}}\sum_{u\in\mathcal{C}(w)}\{\sum_{j=L'^{u}_{C}+2}^{L'^{u}_{I}}\} log[(1-d^{u}_{j})\cdot log(1-\frac{e^{6}}{1+e^{-6}}\epsilon)+d^{u}_{j}\cdot log(1-\frac{e^{6}+1}{4e^{6}}\epsilon)], \tag{24}$$

3.2. **Convergence of First and Second Order Moments of $\Delta\mathcal{J}$.** The first order moment of $\Delta\mathcal{J}_{\mathrm{CBOW}}$ can be given as

$$E[\Delta\mathcal{J}_{\mathrm{CBOW}}] = \sum_{w\in\mathcal{W}}\sum_{i=L'^{w}_{C}+2}^{L'^{w}_{I}}\{(1-d^{w}_{i})log\frac{[\sigma(x^{T}_{w}\theta^{w}_{i-1})]}{[E(\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1})])}+d^{w}_{i}log\frac{[1-\sigma(x^{T}_{w}\theta^{w}_{i-1})]}{[E(1-\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]}\}, \tag{25}$$

We assume that $L_{+}+L_{-}=\sum_{w\in\mathcal{W}}L'^{w}_{I}-L'^{w}_{C}-1$ represents the count of child-nodes of left and right for all words $w\in\mathcal{W}$, and $C=L_{+}+L_{-}$ represents the total depth in $\Upsilon(w)$ of word $w\in\mathcal{W}$. The first order moment of $\Delta\mathcal{J}_{\mathrm{CBOW}}$ can be updated as:

$$E[\Delta\mathcal{J}_{\mathrm{CBOW}}] = \sum_{w\in\mathcal{W}} log\frac{[\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{+}}}{[E(\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{+}}}+log\frac{[1-\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{-}}}, \tag{26}$$

For $x^{T}_{w}\theta^{w}_{i-1}-\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}<\epsilon$, we have $E[\Delta\mathcal{J}_{\mathrm{CBOW}}]=\mathcal{O}log(1+\frac{1}{n})$, and thus converges to zero in the limit of infinity:

$$lim_{n\to\infty}E[\Delta\mathcal{J}_{\mathrm{CBOW}}]=0. \tag{27}$$

The second-order moment of $\Delta\mathcal{J}_{\mathrm{CBOW}}$ can be given as

$$\begin{aligned}E[(\Delta\mathcal{J}_{\mathrm{CBOW}})^{2}] \le \sum_{w\in\mathcal{W}}\{&[log\frac{[\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{+}}}{[E(\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{+}}}]^{2}+[log\frac{[1-\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{-}}}]^{2}\\ &+2log\frac{[\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{+}}}{[E(\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{+}}}\cdot log\frac{[1-\sigma(x^{T}_{w}\theta^{w}_{i-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{x}^{T}_{w}\widetilde{\theta}^{w}_{i-1}))]^{L_{-}}}\},\end{aligned} \tag{28}$$

We have $E[(\Delta\mathcal{J}_{\mathrm{CBOW}})^{2}]=\mathcal{O}log^{2}(1+\frac{1}{n})$, and thus converges to zero in the limit of infinity:

$$lim_{n\to\infty}E[(\Delta\mathcal{J}_{\mathrm{CBOW}})^{2}]=0. \tag{29}$$

Similarly for Skip-gram model, the first and second order of moment of $\Delta\mathcal{J}_{\mathrm{SG}}$ can be given as

$$E[\Delta\mathcal{J}_{\mathrm{SG}}] = \sum_{w\in\mathcal{W}}\sum_{u\in\mathcal{C}(w)} log\frac{[\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{+}}}{[E(\sigma(\widetilde{v}(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{+}}}+log\frac{[1-\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{v}(v(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{-}}}, \tag{30}$$

$$\begin{aligned}E[(\Delta\mathcal{J}_{\mathrm{SG}})^{2}] \le \sum_{w\in\mathcal{W}}\sum_{u\in\mathcal{C}(w)}\{&[log\frac{[\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{+}}}{[E(\sigma(\widetilde{v}(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{+}}}]^{2}+[log\frac{[1-\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{v}(v(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{-}}}]^{2}\\ &+2log\frac{[\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{+}}}{[E(\sigma(\widetilde{v}(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{+}}}\cdot log\frac{[1-\sigma(v(w)^{T}\theta^{u}_{j-1})]^{L_{-}}}{[E(1-\sigma(\widetilde{v}(v(w)^{T}\widetilde{\theta}^{u}_{j-1}))]^{L_{-}}}\},\end{aligned} \tag{31}$$

We have $E[\Delta\mathcal{J}_{\mathrm{CBOW}}]=\mathcal{O}log(1+\frac{1}{n})$ and $E[(\Delta\mathcal{J}_{\mathrm{SG}})^{2}]=\mathcal{O}log^{2}(1+\frac{1}{n})$, and thus converges to zero in the limit of infinity:

$$lim_{n\to\infty}E[\Delta\mathcal{J}_{\mathrm{SG}}]=0 \quad and \quad lim_{n\to\infty}E[(\Delta\mathcal{J}_{\mathrm{SG}})^{2}]=0. \tag{32}$$

4. **Experiments.** In this section, we present rich experiments to verify both effectiveness and efficiency of the approximately incremental hierarchical softmax function based word embeddings. We first evaluate the time and quality of the vectors by comparing global training and incremental training, then use two natural language processing tasks, i.e., word similarity/relatedness and medical entity identification, to evaluate the vectors. Finally, we automatically detect words that undergo significant semantic changes over time and visualize the word semantic evolution.

4.1. **Training Time and Quality.** The popular English Wikipedia is used as the source to train the incremental word vectors. The Wikipedia data is split into several sets. The 2GB original texts, containing 474,746,098 tokens and 100,278 unique words, is choose as the initial training corpus, which is the old corpus as presented in previous sections. Then, 10KB, 100KB, 1MB, 10MB, 100MB, and 1GB new texts is choose as new update corpora to compare the performance of the approximately incremental models. For the contrasting incremental hierarchical training, we employ the weighted contextual frequency aggregated (WCFA) to build the binary tree, where the context window size is 10. The number of threads is 10, and the word vector dimensions is 300 in the all comparative experiments.

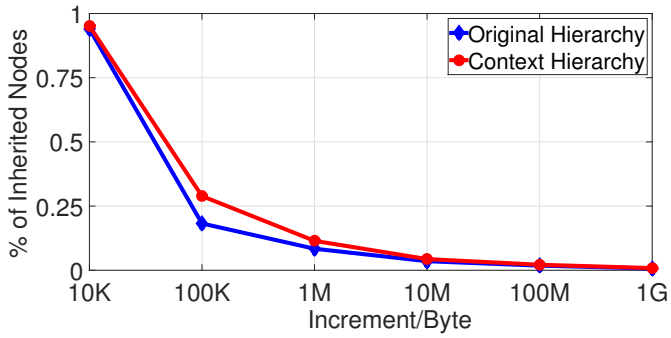

FIGURE 2. Leaf and internal nodes change in incremental scenes.

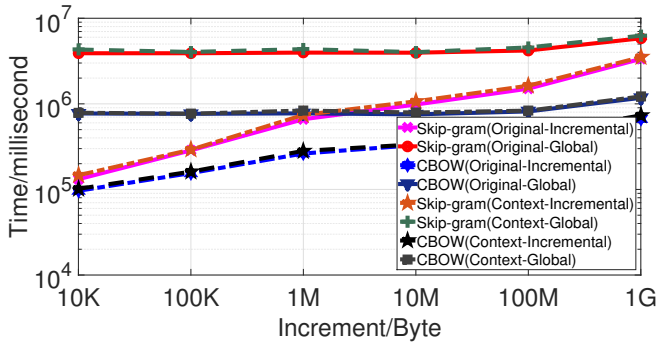

FIGURE 3. Training time.

First, we count the change of Huffman and WCFA trees in the incremental scene, as shown in Figure 2. We have found that the WCFA tree can keep more internal nodes and inherit more parameter vectors. Intuitively, the weighted contextual frequency is more stable than original word frequency. Then we compare the training time and speedup using our incremental training algorithm in hierarchical Huffman and WCFA trees, as

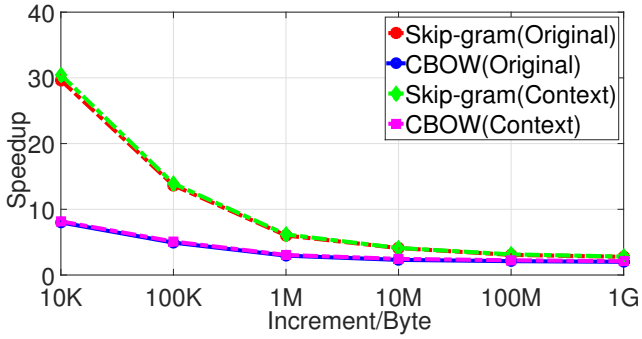

FIGURE 4. Speedup.

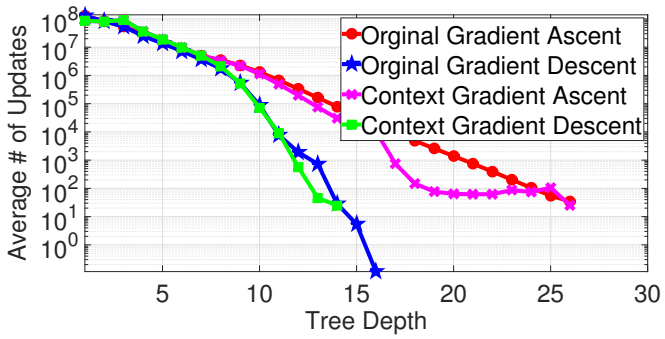

FIGURE 5. Average # of gradient updates (2GB+1GB).

shown in Figure 3. As shown in Figure 4, the maximum speed up is up to 30 in Skip-gram model. We can see that the WCFA tree based models have slightly better than natural Huffman based models. To further understand the approximately stochastic gradient descent process for reducing the inheritable error, we show the average number of gradient updates at each level of the Huffman tree in Figure 5.

TABLE 2. Similarity results

	Items	Re-start	Incremental(Original)	Incremental(Context)	T-Reserved
CBOW	MSE	6.58×10^{-3}	6.60×10^{-3}	6.65×10^{-3}	7.82×10^{-3}
	1-Cos	0.360	0.363	0.374	0.418
	E-Dis	1.410	1.414	1.441	1.545
Skip-gram	MSE	6.62×10^{-3}	6.65×10^{-3}	6.75×10^{-3}	7.89×10^{-3}
	1-Cos	0.362	0.364	0.375	0.402
	E-Dis	1.411	1.416	1.438	1.528

5,000 word vectors are selected to test the difference of the word vectors, and the mean square error (MSE) is choose to evaluate the difference between two sets of word vectors. The results is shown in Table 2. "**Global**" represents re-start training from all corpus. "**Incremental(Original)**" represents the proposed incremental learning models on Huffman structure, and "**Incremental(Context)**" represents our incremental learning models on WCFA structure. "**T-Reserved**" is the old tree reserved based incremental model with stochastic gradients on the new corpus. "T-Reserved" is worst since it

only uses the tree based on the old corpus. We can see that in general, the MSE of re-start training is better than approximately incremental word embedding. The MSE of incremental Huffman based model is better than WCFA tree based model for the same hierarchical binary tree.

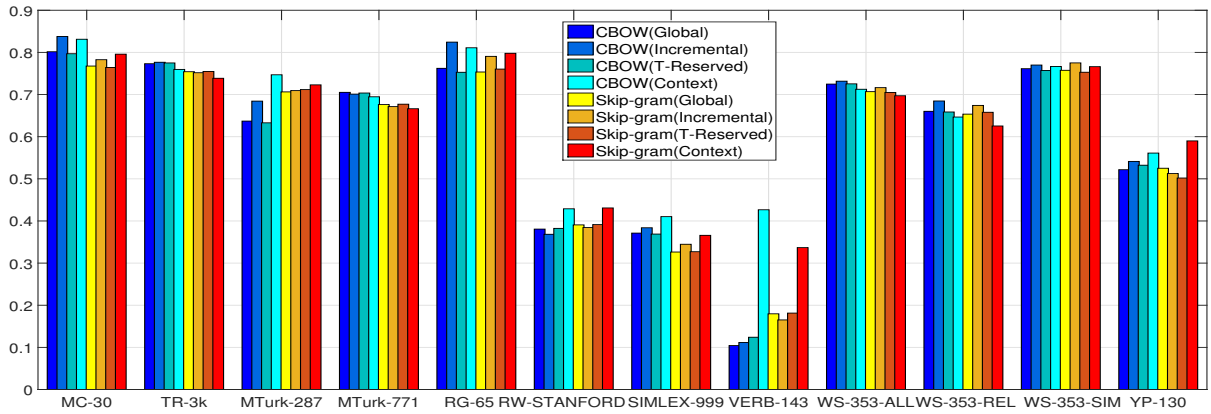

FIGURE 6. Comparison word embeddings for word similarity/relatedness benchmark datasets.

4.2. **Word Similarity/Relatedness.** Next, we use the word similarity/relatedness evaluation benchmarks to evaluate the correctness of the proposed approximately incremental word embedding. Specifically, we use the datasets including MC-30, TR-3k, MTurk-287, MTurk-771, RG-65, RW-STANFORD (RW), SIMLEX-999, VERB-143, WS-353-ALL, WS-353-REL, WS-353-SIM, and YP-130, which is collected by Faruqui and Dyer [6]. The results are shown in Figure 6, and we can see that, the incremental training results including the Huffman and WCFA based models are comparable and sometimes better than the global training results.

4.3. **Word Evolution.** To evaluate the semantic changes of individual words by following nearest-neighbor relations over time, we adopt the NYTimes corpus which contains nearly every article published in the New York Times between January 01, 1987 and June 19th, 2007. We chose 6 groups of words from different domains, such as politics, traffic and information technology, to trace the word semantic changes by cosine distance of word vectors over the years. We set different thresholds respectively, and filter some noise, as shown in Figure 7. We can track the President and the real President of the United States. There are two stages that the cosine distance of 'Bush' and 'President' are small for the real 41st and 43st Presidents, etc. We also track the decline of Detroit in modern automobile industry, and the rise of information technology, software, Internet and Chip manufacturing, etc At last, we also track the late war in Vietnam, which is getting closer and closer to the peace.

4.4. **Medical Entity and Relation Extraction.** We also test medical entity and relation extraction using different training methods. Different from the previous task, we uses word embeddings as fundamental features to build entity and relation embeddings, and train a model to predict the structural output of each sentence. Thus, it is more complicated than comparing the similarities between words. We use the distant supervision framework [28, 29] to train a model based on the word embeddings produces by our experiments. We use the **BiInfer** [27] dataset, which consists of 1,530 manually annotated

FIGURE 7. Evolution of the 6 words whose embedding vector changed the most (in cosine distance) from 1987 to 2007 (NYTimes).

biomedical paper abstracts as test data and 100k sampled PubMed paper abstracts as training corpus. Firstly, we train 2G English Wikipedia as the basic word embeddings. To emphasize the medical corpus, we adjust the adaptive learning rate:

$$\eta' = \eta_0(\frac{\kappa}{\phi' + 1}) \tag{33}$$

where ϕ' represents the total number of tokens in the new corpus $\mathcal{W}'$, and κ is the number of already trained words. Secondly, we also employ the automatically labeled the sampled training corpora NYtimes by distant supervision following the procedure in [29, 30]. We follow [29] to setup the training and testing processes, using the tools available online.[3]

[3] https://github.com/shanzhenren/CoType

Then we also use the strict, micro and macro F1 scores as our evaluation metrics to detect entity mention boundaries and predicted entity types.

TABLE 3. Performance comparison of entity recognition and typing (using strict, micro and macro metrics) on the BioInfer.

	Items	S-F1	Mi-F1	Ma-F1
Skip-gram	Normal	0.74	0.76	0.75
	Context Incremental	0.74	0.76	0.75
	Incremental	**0.74**	**0.77**	**0.76**

TABLE 4. Performance comparison on relation classification accuracy on the BioInfer.

	Items	CT-RM	CT-TS	CT
Skip-gram	Normal	0.587	0.591	0.617
	Context Incremental	0.587	0.591	0.618
	Incremental	**0.589**	**0.591**	**0.618**

The results are shown in Table 3 and Table 4. Both the entity recognition score and relation classification score are reported. We can see that, the incremental training and normal training are also similar, and it seems incremental training is a little bit better than global training for the rich medical entity and relation embeddings. This again demonstrates that incremental training is comparable to global training, and saves a lot of training time. The CBOW model and Skip-gram model perform similarly, and we choose the better results in rounds of tests. This may be because the semi-supervised learning and distant supervision models can eliminate the vector representation difference produced by different algorithms.

5. **Conclusion.** In this paper, we present an approximately incremental neural language learning framework for the hierarchical softmax function to train evolving data and some domain independent corpus. We developed the algorithms based on CBOW and Skip-gram models, which are the most popular word embeddings. We extend the traditional hierarchical structure from Huffman tree to weighted contextual frequency aggregated tree for high time speedup. To demonstrate the effectiveness and efficiency, we performed systematic evaluation on both training time and vector quality. We also evaluate our results on word similarity/relatedness, word evolution and medical entity and relation extraction benchmarks. The results of the systematic evaluation and down-stream tasks show that our incremental training is significantly faster than global training, and has similar or better performance. Theoretical analysis also helped us better understand the bound and convergence of the incremental algorithm. The natural future work is to extend our approach to other advanced network representation learning [25].

Acknowledgment. This work is supported by the National Key R&D Program of China under Grant No.2018YFC083084.

REFERENCES

[1] R. Bamler and S. Mandt, "Dynamic word embeddings via skip-gram filtering," *arXiv preprint arXiv:1702.08359*, 2017.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.
[3] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.
[4] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 493–2537, 2011.
[5] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 4171–4186, 2019.
[6] M. Faruqui and C. Dyer, "Improving vector space word representations using multilingual correlation," *The Conference of the European Chapter of the Association for Computational Linguistics*, pp. 462–471, 2014.
[7] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *Journal of Machine Learning Research*, vol. 13, pp. 307–361, 2012.
[8] N. Kaji and H. Kobayashi, "Incremental skip-gram model with negative sampling," *The Conference on Empirical Methods in Natural Language Processing*, pp. 363–371, 2017.
[9] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.
[10] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.
[11] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," *The International Conference on Neural Information Processing Systems*, pp. 2177–2185, 2014.
[12] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015.
[13] S. Li, J. Zhu, and C. Miao, "Psdvec: A toolbox for incremental and scalable word embedding," *Neurocomputing*, vol. 237, pp. 405–409, 2017.
[14] H. Luo, Z. Liu, H.-B. Luan, and M. Sun, "Online learning of interpretable word embeddings," *The Conference on Empirical Methods in Natural Language Processing*, pp. 1687–1692, 2015.
[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *The International Conference on Learning Representations*, 2013.
[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *The International Conference on Neural Information Processing Systems*, pp. 3111–3119, 2013.
[17] T. Mikolov, W. t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," *The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751, 2013.
[18] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," *The International Conference on Neural Information Processing Systems*, pp. 1081–1088, 2008.
[19] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," *The International Conference on Artificial Intelligence and Statistics*, pp. 246–252, 2005.
[20] H. Peng, M. Bao, J. Li, M. Z. A. Bhuiyan, Y. Liu, Y. He, and E. Yang, Incremental term representation learning for social network analysis. *Future Generation Computer Systems*, vol. 86, pp. 1503–1512, 2018.
[21] H. Peng, J. Li, Q. Gong, Y. Song, Y. Ning, K. Lai, and P. S. Yu, "Fine-grained event categorization with heterogeneous graph convolutional networks," *The International Joint Conference on Artificial Intelligence*, pp. 3238–3245, 2019.
[22] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-cnn," *The World Wide Web Conference*, pp. 1063–1072, 2018.
[23] H. Peng, J. Li, Y. Song, and Y. Liu, "Incrementally learning the hierarchical softmax function for neural language models," *AAAI Conference on Artificial Intelligence*, pp. 3267–3273, 2017.
[24] H. Peng, J. Li, S. Wang, L. Wang, Q. Gong, R. Yang, B. Li, P. Yu, and L. He, "Hierarchical taxonomy-aware and attentional graph capsule rcnns for large-scale multi-label text classification," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[25] H. Peng, J. Li, H. Yan, Q. Gong, S. Wang, L. Liu, L. Wang, and X. Ren, "Dynamic network embedding via incremental skip-gram with negative sampling," *arXiv preprint arXiv:1906.03586*, 2019.
[26] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," *The Conference on Empirical Methods in Natural Language Processing*, vol. 14, pp. 1532–1543, 2014.
[27] S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, and T. Salakoski, "Bioinfer: a corpus for information extraction in the biomedical domain," *BMC Bioinformatics*, vol. 8(50), pp. 1–24, 2007.
[28] M. Qu, X. Ren, and J. Han, "Automatic synonym discovery with knowledge bases," *arXiv preprint arXiv:1706.08186*, 2017.
[29] X. Ren, Z. Wu, W. He, M. Qu, C. R. Voss, H. Ji, T. F. Abdelzaher, and J. Han, "Cotype: Joint extraction of typed entities and relations with knowledge bases," *The International Conference on World Wide Web*, pp. 1015–1024, 2017.
[30] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," *Machine Learning and Knowledge Discovery in Databases*, pp. 148–163, 2010.
[31] S. Tomori, T. Ninomiya, and S. Mori, "Domain specific named entity recognition referring to the real world by deep neural networks," *The Annual Meeting of the Association for Computational Linguistics*, pp. 236–242, 2016.
[32] J. Turian, L. A. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," *The Annual Meeting of the Association for Computational Linguistics*, pp. 384–394, 2010.
[33] H. Yan, H. Peng, C. Li, J. Li, and L. Wang, "Bibliographic name disambiguation with graph convolutional network," *The International Conference on Web Information Systems Engineering*, pp. 538–551, 2019.
[34] C. Yuwei, P. Hao, and Y. Philip S, "Multi-information source hin for medical concept embedding," *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020.

Hao Peng, is currently an assistant professor at School of Cyber Science and Technology in Beihang University. His research interests include representation learning, text mining and social network mining.

Lin Liu, is currently a Ph.D. candidate at the State Key Laboratory of Software Development Environment in Beihang University(BUAA), Beijing, China. Her research interests include data mining and blockchain.

liya Ma, is currently an engineer at National Computer Network Emergency Response Technical Team/Coordination Center of China. Her research interests include data mining and deep learning.

Weiqin Zhao, is currently a undergraduate at the School of Computer Science and Engineering in Beihang University(BUAA), Beijing, China. His research interests include data mining and deep learning.

Hongyuan Ma, is a senior engineer in National Computer Network Emergency Response Technical Team/Coordination Center of China. His current research interests include information retrieval, big data mining and analytics. Email: mahongyuan@foxmail.com.

Yuntao Long, is an engineer at Institutes of Science and Development in Chinese Academy of Sciences. Her research interests include data analysis and management.