Article

Concatenated Vector Representation with the Asymmetric GloVe Model

Junfeng Shi* and Jihong Li

School of Modern Educational Technology, Shanxi University, Taiyuan 030006, China
* Correspondence: sjf@sxu.edu.cn

Received: 7 November 2023
Accepted: 31 December 2023
Published: 25 March 2025

Abstract: The GloVe model is a widely used model for word vector representation learning. The word vector trained by the model can encode some semantic and syntactic information, and the conventional GloVe model trains the word vector representation by collecting the context word within a symmetric window for a given target word. Obviously, such collection does not obtain the left/right side information between the context word and the target word, which is linguistically critical information for learning a word representation of syntactic information. Therefore, the word vector trained by the GloVe model performs poorly in syntax-based tasks such as the part-of-speech tagging task (abbreviated as the POS task) and the chunking task. In order to solve this problem, a concatenated vector representation is proposed with the asymmetric GloVe model, which distinguishes left contexts from right contexts of the target word and exhibits more syntactic similarity than the original GloVe vector representation in looking for the target word's neighbor words. By using the syntactic test set, the concatenated vector representation performs well for the word analogy task, and the syntax-based tasks such as the POS task and the chunking task. At the same time, the dimension of the concatenated vector representation is the half dimension of the original GloVe vector representation, reducing the running time greatly.

Keywords: GloVe model; symmetric window; asymmetric window; concatenated vector representation; syntax-based tasks

1. Introduction

In recent years, many pre-training language models have been proposed in the field of natural language processing, including the CBOW model, the skip-gram model [1], the GloVe model [2], the Elmo model [3], the Bert model [4], the GPT model [5−7] and so on. Among them, the GloVe model is effective and easy-to-implement, and performs well in many downstream tasks such as sentiment classification [8], multilingual retrieval [9], and relation extraction [10].

The GloVe model is built based on the global co-occurrence information between any two words. Therefore, the learned word vector encodes the word's semantic and syntactic information, but the syntactic information encoded by the GloVe model is less. The reason is that the GloVe model does not consider whether the context words are on the left or right side of the target word. The GloVe model has two ways to set a context window. The first way is to set a symmetric context window, which is a context window that extends to the left and right of a target word, and the other way is to set an asymmetric context window that extends only to the left of a target word. The symmetrical context window is used to sum the frequency of the context before and after the target word, and the left/right side information is ignored between the context words and the target word, resulting in less syntactic information represented by the word vector. The asymmetric context window preserves the left/right side information between the context words and the target word, however, it is unreasonable to sum up the word vector and the context vector (of the target word) as the output vector. This is because the word vector and the context vector are trained based on the contexts on the left side and right side of a word, respectively. This method will lead to the loss of the left/right side information between the context words and the target word, as well as the partially loss of the syntactic information represented by the word vector.

Specifically, in the following sentence, there is a word "basketball" before and after the word "play", but only the word "basketball" in back is the common collocation of the word "play". The word "basketball" in the front has no syntactic and semantic relationship with the word "play". When counting the co-occurrence matrix in the way of the symmetrical context window of the GloVe model, the frequency of "basketball" before and after "play" will be summed up, so that the syntactic information of "play" cannot be encoded clearly. When outputting a word vector in the way of the asymmetrical context window of the GloVe model, the word vector and context vector of the target word are summed up, but the word vector and the context vector of the target word are trained based on the contexts on the left side and the right side of the target word, respectively. In this way, the syntactic information of "play" cannot be encoded clearly. Therefore, it is not effective to use the word vector trained by the GloVe model as the initial vector for syntax-based tasks.

I like basketball, so I often play basketball.

Word vectors trained by the GloVe model are not very efficient as initial vectors for syntax-based tasks. For example, the POS task [11] is to mark the corresponding part of each word in a speech. Word vectors that lack syntactic information are not beneficial to correctly determine the part of the speech in the POS task. The chunking task [12], also known as shallow semantic analysis, divides the text into syntactically related parts, and marks each part of a sentence with syntactic components such as nouns or verb phrases (NP or VP). Word vectors that lack syntactic information are not beneficial to correctly determine the syntactically related parts in the chunking task. Therefore, it is necessary to study how to train word vectors to encode syntactic information.

Some studies have been done to improve the syntactic information of the word vector representation by distinguishing the left and right contexts of target words. Literature [13] has enlarged projection layers of both CBOW model and skip-gram model to retain the left/right side information between the context words and the target word. Literature [14] has improved the CBOW model and assigned different weights to the context words based on the type and the relative position (the distance to the left/right) of the target word. Literature [15] has proposed the DSG (direct skip-gram) model, where a special direction vector has been introduced to indicate whether the context word is on the target word's left or right side. The performance of the above methods has been improved in syntax-based tasks considering the left/right side information between the context words and the target word.

How to modify the GloVe model so that the word vector trained can reflect more syntactic information? Intuitively, concatenating the left and right vectors of words can better reflect the syntactic information of words. Therefore, the left and right vectors are learned with the half size of the original dimension based on the asymmetric GloVe model, and concatenated as the output word vectors. Experiments show that, on the syntactic test set, the concatenated vector representation has better performance for syntax-based tasks including the word analogy task, the POS task and the chunking task. At the same time, more syntactically similar words can be found through the proposed word vector representation. Moreover, this method can significantly reduce the running time as only half-dimension word vectors need to be trained.

The contributions of this paper are as follows:

(1) The construction method of the left-side co-occurrence matrix, right-side co-occurrence matrix and symmetric co-occurrence matrix are given.

(2) It is proved that the target word vector and the context word vector (based on the left co-occurrence matrix training) encode the information of the left context and right context of the word, respectively.

(3) The concatenated word vector representation is proposed with the asymmetric GloVe model and its training method. Extensive experiments show the effectiveness of the proposed word vector representation.

2. GloVe model

The GloVe model [2] can, respectively, train low-dimensional word vectors with symmetric and asymmetric context windows.

2.1. GloVe Model with Symmetric Context Window

By using the GloVe model, the steps of training a low-dimensional word vector are as follows:

(1) Count the vocabulary from the corpus and rank the vocabulary according to the frequency of words, where n is the size of the vocabulary, and $c_i \left(1 \leqslant i \leqslant n\right)$ represents the ith word in the vocabulary.

(2) Set the fixed window size as w, traverse the words in the corpus, count the frequency of words in the fixed windows on both sides of the target words, and generate a symmetric co-occurrence matrix which is expressed as X^S. The size of the matrix is $n \times n$. Use X^S_{ij} to represent the elements of row i and column j of the symmetric co-occur-

rence matrix.

(3) Shuffle the co-occurrence matrix.

(4) Use v^S to represent the low-dimensional word vector based on training the symmetric co-occurrence matrix. The objective function of training v^S is as follows:

$$J^S = \sum\nolimits_{i,j=1}^{n} f\left(X_{ij}^S\right)\left(\left(v_i^S\right)^T \tilde{v}_j^S + b_i^S + b_j^S - logX_{ij}^S\right)^2 \tag{1}$$

where v_i^S is the symmetrical low-dimensional word vector of the word c_i, $\tilde{v}_j^S$ is the symmetrical low-dimensional context vector of the word c_j, b_i^S and b_j^S are the offset terms corresponding to v_i^S and $\tilde{v}_j^S$, and f is the weight function which is defined by formula (2). x^{max} and α are hyperparameters set to be 100 and 3/4, respectively, which are verified as the optimal values via experiments in literature [2].

$$f(x) = \begin{cases} \left(x/x^{max}\right)^{\alpha} & x < 100 \\ 1 & otherwise \end{cases} \tag{2}$$

(5) Sum up v_i^S and$\tilde{v}_j^S$ as the output word vector.

2.2. GloVe Model with Asymmetric Context Window

The GloVe Model with asymmetric context windows is actually based on the left-side co-occurrence matrix. Literature [2] regards the context of the left co-occurrence matrix as the asymmetric context. Compared with the GloVe model with a symmetric context window, the difference lies in the construction of the co-occurrence matrix. Set the fixed window size as w, traverse the words in the corpus, count the frequency of the words in the fixed window on the left side of the target word, and generate the left side co-occurrence matrix which is expressed as X^L. Use X_{ij}^L to represent the elements of row i and column j of the asymmetric co-occurrence matrix. Use v^L to represent the low-dimensional word vector based on training the left-side co-occurrence matrix. The objective function of training v^L is as follows:

$$J^L = \sum\nolimits_{i,j=1}^{n} f\left(X_{ij}^L\right)\left(\left(v_i^L\right)^T \tilde{v}_j^L + b_i^L + b_j^L - logX_{ij}^L\right)^2 \tag{3}$$

where v_i^L is the left low-dimensional word vector of the word c_i, $\tilde{v}_j^L$ is the left low-dimensional context vector of the word c_j, b_i^L and b_j^L are offset terms corresponding to v_i^L and $\tilde{v}_j^L$, and f is the same weight function as formula (2).

3. Concatenated Word Vector Representation with the Asymmetric GloVe Model

Based on the left co-occurrence matrix, two kinds of word vectors are obtained, i.e., the target word vector and the context word vector. In fact, the context word vector encodes the right-side information of the word, and the argument is given as follows.

Use X^R to represent the co-occurrence matrix based on the right window of the target word, use X_{ji}^R to represent the elements of row j and column i of X^R and use v^R to represent the low-dimensional word vector based on training the left-side co-occurrence matrix. The objective function of training v^R is as follows:

$$J^R = \sum\nolimits_{j,i=1}^{n} f\left(X_{ji}^R\right)\left(\left(v_j^R\right)^T \tilde{v}_i^R + b_i^R + b_j^R - logX_{ji}^R\right)^2 \tag{4}$$

Obviously, $X_{ij}^L = X_{ji}^R$. For example, a corpus contains one sentence, that is, “I like basketball”. The vocabulary is shown in Table 1. Set the fixed window size as 1. The co-occurrence matrix is shown in Table 2 based on the left window of the target word, and the co-occurrence matrix is shown in Table 3 based on the right window of the target word. It can be seen that $X_{ij}^L = X_{ji}^R \left(1 \leqslant i \leqslant 3, 1 \leqslant j \leqslant 3\right)$, and X^L is the transpose of X^R.

Table 1 the vocabulary of the sample dataset

word	frequency
I	1
like	1
basketball	1

Table 2 the left-side co-occurrence matrix X^L

	I	like	basketball
I	0	0	0
like	1	0	0
basketball	0	1	0

Table 3 the right-side co-occurrence matrix X^R

	I	**like**	**basketball**
I	0	1	0
like	0	0	1
basketball	0	0	0

Ignoring b_i^L , b_j^L , b_i^R and b_j^R, we have $\left(v_i^L\right)^T \tilde{v}_j^L = \left(v_j^R\right)^T \tilde{v}_i^R$. The objective function actually uses the dot product of the *i*th word vector and the *j*th word vector to fit the co-occurrence frequency of words *i* and *j*. Therefore, we can treat $\tilde{v}_j^L$ as v_j^R, and rewrite the objective function of training v^L in the following form:

$$J^L = \sum_{i,j=1}^{n} f\left(X_{ij}^L\right)\left(\left(v_i^L\right)^T v_j^R + b_i^L + b_j^R - logX_{ij}^L\right)^2 \tag{5}$$

It can be seen from formula (5) that while training the left low-dimensional word vector, the right low-dimensional word vector is obtained. Intuitively, the word vectors representing the information on both sides of the word should be concatenated to form the final vector representation of the word. Therefore, the concatenated vector representation is proposed with the asymmetric GloVe model. We train half-dimension asymmetric word vectors, namely, v^L and v^R, and then concatenate them. The specific steps are as follows. The flowchart is shown in Figure 1.

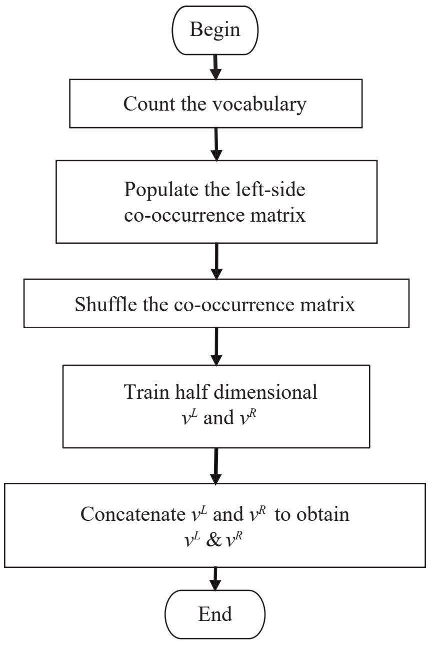
Begin
Count the vocabulary
Populate the left-side co-occurrence matrix
Shuffle the co-occurrence matrix
Train half dimensional v^L and v^R
Concatenate v^L and v^R to obtain v^L & v^R
End

Figure 1. The flowchart of the concatenated vector representation with the asymmetric GloVe model.

1) Count the vocabulary from the corpus and rank the vocabulary according to the frequency of words, where n is the size of the vocabulary.

2) Set the fixed window size as *w*, traverse the words in the corpus, and calculate the left-side co-occurrence matrix expressed as X^L. The size of the matrix is $n \times n$.

3) Shuffle the co-occurrence matrix.

4) Set the dimension to be the half dimension of the GloVe model, and train v^L and v^R with formula (5).

5) Concatenate v^L and v^R as a low-dimensional word vector expressed as $v^L \& v^R$.

4. Experiment

4.1. Experimental configuration

The English Wikipedia corpus containing 200 million tokens is used as the experimental corpus. Other parameters are shown in Table 4. In each experiment, this set of hyperparameters is used unless otherwise stated. In the table, “vector-size” represents the vector size of the word representation, ”window-size” represents the size of the sliding window for scanning the corpus, ”max-vocab” represents the upper limit of the frequency in the vocabulary, ”min-count” represents the lower limit of the frequency in the vocabulary, and “iteration” represents the maximum number of iterations of the training word vector.

Table 4 The parameters of training word vector

Parameters	vector-size	window-size	max-vocab	min-count	iteration
Value	200	10	100000	10	50

$v^S + \tilde{v}^S$ and $v^L + \tilde{v}^L$ are trained using the GloVe model with a symmetric context window and an asymmetric context window, respectively. Half-dimensional v^L and v^R are trained with the asymmetric Glove model and concatenated to generate $v^L \& v^R$. The performances of the three kinds of word vectors are compared.

4.2. Evaluation Task

Quality evaluation. The quality of the word vector representation is evaluated by looking for the target word's neighbor words. In this study, the cosine similarity of word vectors is used to measure the degree of word adjacency.

Word analogy task. Suppose that there are two pairs of similar words, and the task of word analogy [1] is to find the fourth word by assuming that three words' vectors are known. For example, the word vector of "king", "queen" and "woman" is known to find "man". The fourth word is found by finding the nearest word vector to the vector of "king+queen+woman". The similarity of the word vector is calculated by cosine similarity. The word analogy task's test set includes a semantic test set (8869 questions) and a syntactic test set (10675 questions). We compare the accuracy by using the syntactic test set. The performances of $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ are compared in the word analogy task. Four corpora are used to verify the validity of the proposed word vector representation including the English Wikipedia corpus (200 million tokens, 500 million tokens) and the 1B Word Benchmark corpus (200 million tokens, 500 million tokens) [16]. The hyperparameters for the Wikipedia corpus (200 million tokens) are carefully selected to be vector-size 500, window-size 14, max-vocab 10000, min-count 10, and iteration 50. The hyperparameters for the other three corpora are set to be vector-size 600, window-size 10, max-vocab 100000, min-count 10, and iteration 50 based on expert experience.

Syntactic task. To verify whether the concatenated word vector can reflect more syntactic information, the POS task and chunking task are selected for verification because these tasks are based on syntax information.

• The POS task is to mark the corresponding part of a speech for each word. The NCRF++ model [17] is used to complete the POS task, which builds an LSTM-CRF framework together with the CNN to encode character sequences. The ARK dataset [18] is used which includes training sets (1000tweets), validation sets (327tweets), and test sets (500tweets). The parameter settings of the model are shown in Table 5. $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ are used as the initial word vectors of the NCRF + + model, respectively.

Table 5 parameters of NCRF++ model

parameter	value
word emb dim	200
char emb dim	100
use_crf	True
use_char	True
word_seq feature	LSTM
char_seq_feature	CNN
optimizer	SGD
iteration	100
batch size	10
cnn_layer	4
char hidden dim	100
hidden dim	400
lstm layer	1
learning_rate	0.015
lr decay	0.05
momentum	0
l2	1e-8
dropout	0.5

• The chunking task, also known as shallow semantic analysis, divides the text into syntactically related parts and marks each part of a sentence with syntactic components such as nouns or verb phrases (NP or VP). Chunking is usually evaluated using CoNLL2000 shared tasks [12], the training data is the Wall Street Journal corpus (WSJ) from sections 15-18 (211727 tokens), and the test data is WSJ from section 20 (47377 tokens). The NCRF++ model [17] is used to complete the chunking task. The other parameter settings of the model are shown in Table 5 except the parameter "iteration" which is set to be 50. $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ are used as the initial word vector of the NCRF + + model, respectively.

4.3. Result

Table 6, Table 7 and Table 8 list the top ten neighbors of "established", "providing", and "worse" under the word vectors $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$, respectively.

Table 6 The top ten neighbors of "established" under $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$

$v^S + \tilde{v}^S$	$v^L + \tilde{v}^L$	$v^L \& v^R$
founded	**incorporated**	**founded**
incorporated	**founded**	**incorporated**
establishments	*establish*	**organized**
establish	**settled**	**formed**
establishing	*date*	**created**
area	*establishments*	**recogonized**
formed	**formed**	**settled**
leader	*area*	**adopted**
establishment	**populated**	*establish*
organizations	*establishing*	**known**

Table 7 The top ten neighbors of "providing" under $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$

$v^S + \tilde{v}^S$	$v^L + \tilde{v}^L$	$v^L \& v^R$
provide	*provide*	*provide*
provided	*provides*	*provides*
provides	*provided*	*provided*
additional	*additional*	**offering**
services	*services*	**creating**
access	**offering**	**giving**
allowing	**allowing**	**requiring**
support	**creating**	**maintaining**
creating	*access*	*offers*
assistance	*offers*	**allowing**

Table 8 The top ten neighbors of "worse" under $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$

$v^S + \tilde{v}^S$	$v^L + \tilde{v}^L$	$v^L \& v^R$
better	**better**	**better**
bigger	**stronger**	**stronger**
getting	**happier**	**harder**
harder	*even*	**easier**
fared	*getting*	**weaker**
matters	*things*	**faster**
louder	**harder**	**bigger**
situation	**quicker**	**happier**
progressivly	**bigger**	*happen*
nothing	**tougher**	**clearer**

In Tables 6–8, bold words are syntactic neighbors, and italic words are semantic neighbors. It can be seen that from $v^S + \tilde{v}^S$ to $v^L + \tilde{v}^L$and then to $v^L \& v^R$, the numbers of semantic neighbors decrease, and the numbers of syntactic neighbors increase. For example, inTable 6, for $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$, the numbers of semantic adjacency words of "established" are 6,4, 0, respectively, while the syntactic adjacency words of "established" are 3,5,9, respectively. The same is true for the neighbors of "providing" and "worse". Due to the limited space, more words could not be listed here. In sum, under $v^L \& v^R$, the adjacency words include more words with similar syntax. It is shown that $v^L \& v^R$ represents more syntactic information. Therefore, $v^L \& v^R$ can perform well in syntax-based tasks.

The experimental results of the analogy task are given by Table 9. It can be seen that the accuracy of the word analogy task is high for different sizes and different kinds of corpora on the syntactic test set under $v^L \& v^R$.

Table 9 Results on word analogy tasks with the syntactic test set on different corpora

word rep.	**Wikipedia corpus (200 million tokens)**	**Wikipedia corpus (500 million tokens)**	**1B Word Benchmark (200 million tokens)**	**1B Word Benchmark (500 million tokens)**
$v^S + \tilde{v}^S$	50.19%	56.94%	41.54%	51.06%
$v^L + \tilde{v}^L$	49.82%	56.41%	42.85%	50.28%
$v^L \& v^R$	**54.70%**	**60.01%**	**51.53%**	**56.70%**

The accuracy comparison of $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ for POS task is shown in Table 10. The accuracy with the initial word vector $v^L \& v^R$ has the best performance, followed by $v^L + \tilde{v}^L$ and $v^S + \tilde{v}^S$.

Table 10 Accuracy comparison on POS task

word rep.	**Dev**	**Test**
$v^S + \tilde{v}^S$	88.41%	88.52%
$v^L + \tilde{v}^L$	88.76%	89.01%
$v^L \& v^R$	**89.22%**	**89.49%**

The F1 score comparison of $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ for the chunking task is shown in Table 11. The F1 score of the chunking task with the initial word vector $v^L \& v^R$ has the best performance, followed by $v^L + \tilde{v}^L$ and $v^S + \tilde{v}^S$.

Table 11 F1 score comparison on Chunking task

word rep.	Test
$v^S + \tilde{v}^S$	94.37%
$v^L + \tilde{v}^L$	96.11%
$v^L \& v^R$	**96.14%**

The above experimental results show that $v^L \& v^R$ can encode syntactic information better and is suitable to be the initial word vector for syntax-based tasks.

4.4. Ablation study

The performance of $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ in the word analogy task with the syntactic test set is compared under different context window sizes or under different word vector lengths. In Figure 2(a) and (b), experimental results are shown for different context windows or different vector lengths. As can be seen from Figure 2(a), under various window sizes, $v^L \& v^R$ has the best performance. As can be seen from Figure 2(b), under various vector lengths, $v^L \& v^R$ has the best performance, and the larger the dimension, the more obvious the advantage.

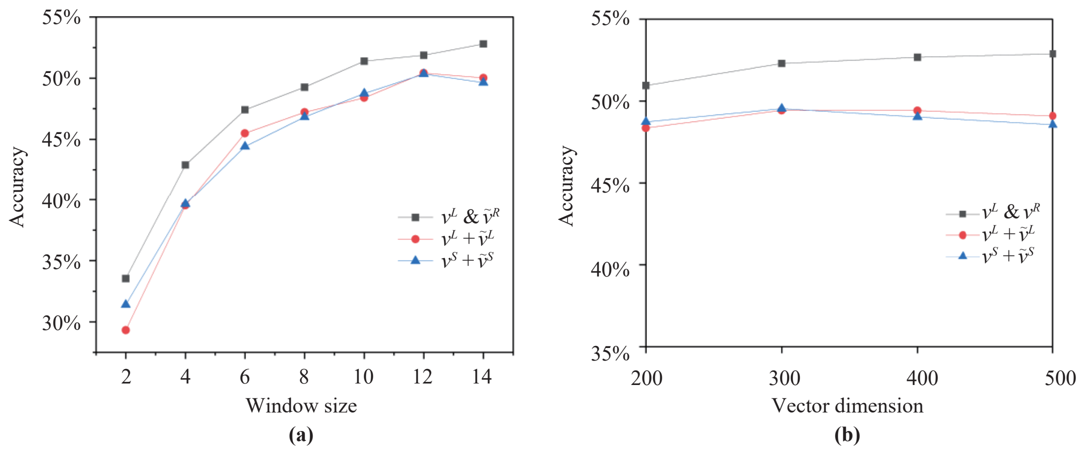

Figure 2. **(a)** shows the accuracy for the analogy task with the syntactic test set varies with the context window size, and **(b)** shows the accuracy for the analogy task with the syntactic test set varies with the vector size.

4.5. Running Time

The running time of the GloVe model mainly depends on the time of populating co-occurrence matrices and the time of training word vectors. The more sparse the co-occurrence matrix is, the shorter the time is needed to construct the co-occurrence matrix. The lower the dimension of the word vector is, the shorter the time is needed to train the word vector.

The left-side co-occurrence matrix is more sparse than the symmetric co-occurrence matrix. Therefore, the time of the populating co-occurrence matrix for $v^L \& v^R$ is shorter than that of $v^S + \tilde{v}^S$ and is the same as that of $v^L + \tilde{v}^L$.

The dimension of $v^L \& v^R$ is the half of $v^S + \tilde{v}^S$ and $v^L + \tilde{v}^L$. Therefore, the training time of $v^L \& v^R$ is the half of $v^L + \tilde{v}^L$ and less than the half of $v^S + \tilde{v}^S$.

The running time of 200-dimensional $v^S + \tilde{v}^S$, $v^L + \tilde{v}^L$ and $v^L \& v^R$ is recorded. The parameter settings are shown in Table 4. The running time comparison is shown in Table 12. It can be seen that $v^L \& v^R$ has the shortest running time, followed by $v^L + \tilde{v}^L$ and $v^S + \tilde{v}^S$. This is the same result as shown by our analysis.

Table 12 Running time

Word rep.	Running time
$v^S + \tilde{v}^S$	2 hours and 36 minutes
$v^L + \tilde{v}^L$	1 hours and 44 minutes
$v^L \& v^R$	1 hours and 3 minutes

5. Analysis of Experimental Results

The experimental results show that the performance of the concatenated vector representation (with the asymmetric Glove model) is better than the traditional word vector representation for syntactic tasks. This is because concatenated word vectors encode more syntactic information.

The proposed word vector of the training is based on the left side co-occurrence matrix, and the word frequencies of the left side and right side of the target word are not added together. Therefore, the proposed word vector can encode the left side and right side information of the word, that is, it can better reflect the syntactic information.

In addition, it can be theoretically deduced that the target word vector and the context word vector (based on the left co-occurrence matrix training) embody the left context information and the right context information of the word, respectively. Therefore, concatenating these two word vectors as the final word vector can better reflect the syntactic information of the word.

6. Conclusion

In this paper, the concatenated vector representation has been proposed with the asymmetric Glove model. The word vector has been trained based on the left side co-occurrence matrix, and the left word vector and the right word vector are concatenated as the output vector. Compared with the word vector trained by the GloVe model with symmetric context windows and asymmetric context windows, this vector can aggregate more words with the same syntax, and has higher performance on the syntactic test set for the word analogy task, POS task and chunking task. These experiments have shown that the proposed word vector representation encodes more syntactic information and is suitable to be the initial word vector for syntax-based tasks. In addition, the experiment has shown that the time of learning the concatenated vector representation is significantly reduced. In the next step, we will study how to get a co-occurrence matrix that fully reflects multiple levels of information, effectively integrates multiple word representation vectors, and greatly improves the performance of representation learning.

Author Contributions: Junfeng Shi: Methodology, Investigation, Validation, Writing; **Jihong Li:** Methodology, Data collection and analysis, Conceptualization, Editing and Reviewing.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mikolov, T.; Chen, K.; Corrado, G.; *et al*. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations, Scottsdale, AZ, USA, 2–4 May 2013*; ICLR, 2013.
2. Pennington, J.; Socher, R.; Manning, C. GloVe: Global vectors for word representation. In *Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014*; Association for Computational Linguistics, 2014; pp. 1532–1543. doi:10.3115/v1/D14-1162
3. Peters, M.E.; Neumann, M.; Iyyer, M.; *et al*. Deep contextualized word representations. In *Proceedings of 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018*; Association for Computational Linguistics, 2018; pp. 2227–2237. doi:10.18653/v1/N18-1202
4. Devlin, J.; Chang, M.W.; Lee, K.; *et al*. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, Minnesota, 2–7 June 2019*; Association for Computational Linguistics, 2019; pp. 4171–4186. doi:10.18653/v1/N19-1423
5. Radford, A.; Narasimhan, K.; Salimans, T.; *et al*. Improving language understanding by generative pre-training. 2018. Available online: https://paperswithcode.com/paper/improving-language-understanding-by.
6. Radford, A.; Wu, J.; Child, R.; *et al*. Language models are unsupervised multitask learners. 2019. Available online: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
7. Brown, T.B.; Mann, B.; Ryder, N.; *et al*. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, New York, 6 December 2020*; ACM: Vancouver, 2020; p. 159. doi:10.5555/3495724.3495883
8. Ranjan, S.; Mishra, S. Perceiving university students' opinions from Google app reviews. Concurr. Comput. Pract. Exp., **2022**, *34*: e6800. doi: 10.1002/cpe.6800
9. Mahalakshmi, P.; Fatima, N.S.; Balaji, R.; *et al*. An effective multilingual retrieval with query optimization using deep learning technique. Adv. Eng. Softw., **2022**, *173*: 103244. doi: 10.1016/j.advengsoft.2022.103244
10. Chen, X.Y.; Zhang, M.; Xiong, S.W.; *et al*. On the form of parsed sentences for relation extraction. Knowl.-Based Syst., **2022**, *251*: 109184. doi: 10.1016/j.knosys.2022.109184
11. Brants, T. Part-of-speech tagging. In *Encyclopedia of Language & Linguistics*, 2nd ed.; Brown, K., Ed.; Elsevier: Oxford, 2006; pp. 221–230.
12. Sang, E.F.T.K.; Buchholz, S. Introduction to the CoNLL-2000 shared task chunking. In *Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal*; Association for Computational Linguistics, 2000; pp. 127–132.
13. Ling, W.; Dyer, C.; Black, A.W.; *et al*. Two/too simple adaptations of Word2Vec for syntax problems. In *Proceedings of 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, CO, USA, 31 May–5 June 2015*; Association for Computational Linguistics, 2015; pp. 1299–1304. doi:10.3115/v1/N15-1142

14. Ling, W.; Tsvetkov, Y.; Amir, S.; *et al*. Not all contexts are created equal: Better word representations with variable attention. In *Proceedings of 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015*; Association for Computational Linguistics, 2015; pp. 1367–1372. doi:10.18653/v1/D15-1161
15. Song, Y.; Shi, S.M.; Li, J.; *et al*. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018*; Association for Computational Linguistics, 2018, pp. 175–180. doi:10.18653/v1/N18-2028
16. Chelba, C.; Mikolov, T.; Schuster, M.; *et al*. One billion word benchmark for measuring progress in statistical language modeling. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association, Singapore, 14–18 September 2014*; ISCA: Singapore, 2014.
17. Yang, J.; Zhang, Y. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations, Melbourne, Australia, 15–20 July 2018*; Association for Computational Linguistics, 2018; pp. 74–79. doi:10.18653/v1/P18-4013
18. Gimpel, K.; Schneider, N.; O'Connor, B.; *et al*. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011*; Association for Computational Linguistics, 2011; 42–47.

Citation: Shi, J.; Li, J. Concatenated Vector Representation with the Asymmetric GloVe Model. *International Journal of Network Dynamics and Intelligence*. 2025, 4(1), 100002. doi: 10.53941/ijndi.2025.100002