Word Embedding based Heterogeneous Entity Matching on Web of Things

Xingsi Xue*
Fujian Provincial Key Laboratory of Big Data Mining and Applications, Fujian University of Technology
Fuzhou, Fujian, China
Intelligent Information Processing Research Center, Fujian University of Technology
Fuzhou, Fujian, China
jack8375@gmail.com

Jianhua Guo
college of electrical and power engineering, Taiyuan University of Technology
Taiyuan, Shanxi, China
Guojianhua1@hotmail.com

ABSTRACT

Web of Things (WoT) is capable of promoting the knowledge discovery and address interoperability problems of diverse Internet of Things (IoT) applications. However, due to the dynamic and diverse features of data entities on WoT, the heterogeneous entity matching has become arguably the greatest "new frontier" for WoT advancements. Currently, the data entities and the corresponding knowledge on WoT are generally modelled with the ontology, and therefore, matching heterogeneous data entities on WoT can be converted to the problem of matching ontologies. Ontology matching is a complex cognitive process, it is usually initially done manually by domain experts. To effectively distinguish the heterogeneous entities and determine high-quality ontology alignment, this work proposes a word embedding based matching technique. Our approach models the word's semantic in the vector space, and use two vectors' cosine angle to measure the corresponding words' similarity. In addition, the word embedding approach does not depend on a specific knowledge base and retain the rich semantic information of words, which makes our proposal more robust. The experiment uses Ontology Alignment Evaluation Initiative (OAEI)'s benchmark for testing, and the experimental results show that our approach outperforms other advanced matching methods.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; *Lexical semantics*.

KEYWORDS

Web of Things, Word Embedding, Ontology Matching

ACM Reference Format:
Xingsi Xue and Jianhua Guo. 2022. Word Embedding based Heterogeneous Entity Matching on Web of Things. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3487553.3524704

*Corresponding Author

WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9130-6/22/04…$15.00
https://doi.org/10.1145/3487553.3524704

1 INTRODUCTION

Web of Things (WoT) is capable of promoting the knowledge discovery and address interoperability problems of diverse Internet of Things (IoT) applications [16]. In the wake of advancement of WoT, it is necessary to describe a variety of semantically identical knowledge with uniform representation. Ontology matching is a task to establish corresponding relations between semantically related elements of different ontologies so as to solve the ontology heterogeneity problem. Different WoTs can also be regarded as different ontologies, and then ontology matching technology is used to solve the problem of IoT interoperability. Ontology matching is widely involved in medical, biology and other aspects. Moreover, it is difficult in dealing with large-scale ontology matching and complex ontology matching. And Ontology matching also has some challenges in processing some semantically rich words or sentences.

In recent years, the rapid development of ontology matching technology, especially with the rise of deep learning, has added another direction for ontology matching. Ontology matching technology based on deep learning is becoming more and more mature. There are also some good articles, such as Matching sensor ontologies through Siamese neural networks without using reference alignment [21] and VeeAlign: Multifaceted Context Representation Using Dual Attention for Ontology Alignment [8]. However, in the above paper and most neural networks rely on character embedding. In order to be more intuitive and efficient, this paper adopts word embedding to deal with the ontology matching problem.

The main contributions of this paper are, therefore, as follows: (1) we propose a word embedding technology, which converts words into vectors and can effectively distinguish heterogeneous entities, making the matching of two ontologies more efficient, fast, intuitive and accurate. (2) Our method does not need to use a specific knowledge base, which makes it more robust; (3) We propose a sentence preprocessing strategy for the problem, which not only meets the conditions of word embedding but also enables similarity calculation between sentences.

The rest of the paper is organized as follows: Section "RELATED WORK" provides the related work on state-of-the-art ontology matching techniques for deep learning. Section "PRELIMINARY" presents the definition of ontology and ontology matching, and the alignment's evaluation metrics, and introduces the principle of word2vec. Section "METHODOLOGY" gives the concrete implementation of word embedding based on ontology matching, including training of the model, matching process, and giving the pseudo code. Section "EXPERIMENT" shows the experimental results. Finally, Section "CONCLUSION" draws the conclusion.

2 RELATED WORK

2.1 Deep Learning Based On Ontology Matching Technique

An Ontology matching is a complex cognitive process, it is usually initially done manually by domain experts. However, in dynamic environment (such as Web environment), facing the matching requirements between large ontologies, manual method has been far from meeting the requirements. So semi-automatic or even automatic matching method was put forward. In the early stage, there are many researchers and academic groups at home and abroad dedicated to the research of ontology matching technology and the construction of ontology matching system, and achieved results. For example, the matching methods of LMO and GMO of falcon-ao system [9], the matching method based on semantic subgraph [11], and the ontology matching of COMA++ multi-strategy [7], etc. With the rise of deep learning, ontology matching based on deep learning has made some achievements. For example, Lucy Wang's unsupervised word vector matching [15] and Prodromos Kolyvakis's ontology matching research in the biomedical field [10], etc. But they are highly dependent on specific knowledge bases. Vivek Iyer proposed veealign system [8], which solved this dependency problem. However, the embedding technology it uses is character embedding technology, which is the same as that used by most deep learning nowadays, which reduces the semantic correlation between words to a certain extent. Yinfei Yang used the general sentence encoder in his research on matching, which is also in the form of character embedding at the bottom. Finally, the sentence is integrated into a vector with a specified dimension [4], but the matching effect is not very ideal.

2.2 Word Embedding Technique

This paper is based on the framework of Deeplearning4j to realize the vectorization modeling of words and sentences. Both text and sentences are made up of words. Some experts have created the word bag model in the NLP field [6], that is, when we represent the Bag of Words with a long vector. Each index position in the vector represents a word in the text. Then the specific value of each position can be determined according to the situation. We can directly use 0/1 to indicate whether the word appears or not, or fill it with the value of TF-IDF [5], [14]. Bag of Words model is a classical method of text feature representation. Up to now, feature representation of long text based on word bag model can still achieve very good results. Its presentation is simple, direct and easy to understand. However, its disadvantages are also very prominent, that is, the vector dimension is very high, and it is easy to ignore the context information. Word embedding [1] can represent words or text from another angle, that is, each word is represented by a dense vector. This dense vector is no longer the sparse expression of one-hot, but a vector trained by neural network that can represent the physical meaning of the word itself.

In order to further improve the matching effect, I put forward ontology matching technology based on word embedding, which not only greatly reduces manpower but also does not depend on specific knowledge base. In this paper, the word vector is implemented by word2vec. As long as the word is given to learn, it can be matched according to the semantic information of the word. At the same time, because of the use of word embedding technology, its matching efficiency is more efficient. In order to make full use of the advantages of word embedding, sentences are preprocessed, that is, words with actual semantic and significant features are extracted from sentences for further training. We also conducted sensitivity tests for similarity thresholds to ensure optimal matching results.

3 PRELIMINARIES

3.1 Ontology And Ontology Matching

Definition1 An ontology O is a triple $<C, P, I>$ [25], [26].
Where C is the set of classes, i.e., the set of concepts that populates the domain of interest; P is the set of properties, i.e., the set of relations existing between the concepts of domain; I is the set of individuals, i.e., the set of objects of the real world representing the instances of a concept.

Different domains have different emphases and each domain can be conceptualized, ontologies created in this way may cause semantic heterogeneity by making different terms mean the same thing or the same term mean different things. To solve this problem, the ontology matching process must be completed. Specifically, this procedure detects match between two ontologies, O_1 and O_2, and produces A so-called alignment A' in the output.
Definition2 An ontology alignment is a set of correspondences, and a correspondence is a quad $<e, e', H, R>$ [23], [19], [24].
Among them, e and e' are the same type entities from different ontologies. H represents the Confidence Degree of establishing a match, which is usually represented by a real number [0,1]. R represents the semantic relationship between two entities in different ontologies, which usually has four types: inclusion, generalization, incompatible and equivalence.
Definition3 The process of ontology matching is a function A'=f(O_1, O_2, A, r, p) [20], [18].
where A' is the final alignment; O_1 and O_2 are two to-be-matched ontologies; A is the reference alignment (optional); r is the utilized resources; and p is the utilized parameters. The matching process is shown in Figure 1.

3.2 The Principle Of Word2vec

Word2vec [12], [13], proposed by Tomas Mikolov et al., is a major breakthrough in the field of natural language processing by neural networks. In fact, word2vec represents the semantic information of words by learning the word vector. The result of word2vec is to obtain word vector, which is also called word embedding. word

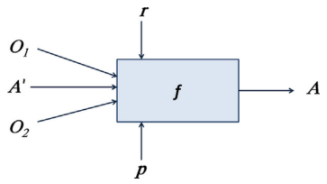

Figure 1: The Flowchart of Ontology Matching Process.

embedding is a mapping that maps words from the original space to a new multi-dimensional space, that is, the original space of words is embedded into a new space. word2vec training itself is unsupervised, and because of this, its value to some extent represents the physical meaning of words better than the word vector acquired based on supervised learning. For its use, the most direct is to measure the similarity between words, essentially using cosine similarity measurement technology. Some semantically similar words can be recalled through the model. word2vec itself can be done in both CBOW model and Skip-gram model, each of which can be implemented in Hierarchial Softmax or Negative Sampling frameworks. The difference between the two approaches is that CBOW predicts words by context, while Skip-gram does the opposite by predicting the context with the central word. Figure 2 and Figure 3 are CBOW+Hierarchial Softmax and Skip-gram + Negative Sampling respectively. This paper adopts Skip-gram + Negative Sampling structure.

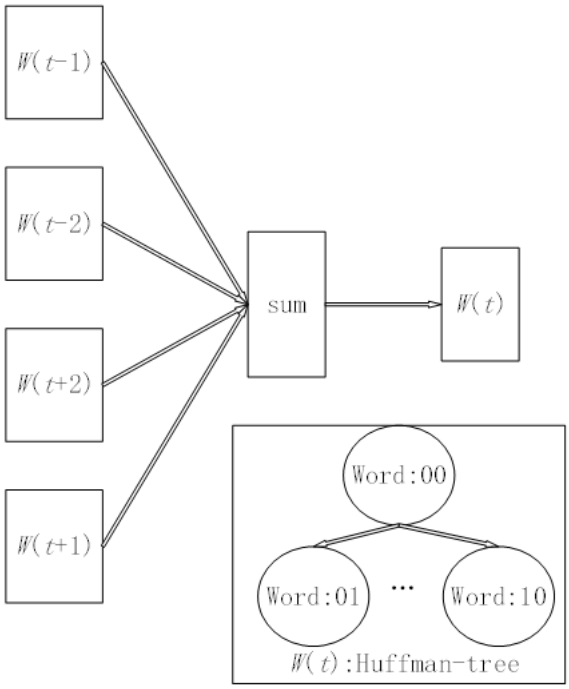

Figure 2: The structure of CBOW+Hierarchial Softmax.

Figure 2 is the implementation of word2vec via CBOW+Hierarchial Softmax, while Figure 3 is the implementation of word2vec via Skip-gram + Negative Sampling (word2vec is typed as one-hot encoding). The CBOW input is the context word set for that word, while the Skip-gram input is the word. Hierarchial Softmax is used for the former, so the output is a Huffman tree, where each leaf node represents a specific word in the word set. This gives each word a unique binary code. Hierarchial Softmax's purpose is to replace the default Softmax with a tree structure, with a Logistic dichotomy for each internal structure in the path from the root node to the leaf node, so the original Softmax approximates multiple Logistic regression. But Skip-gram + Negative Sampling is mainly adopted in this paper.

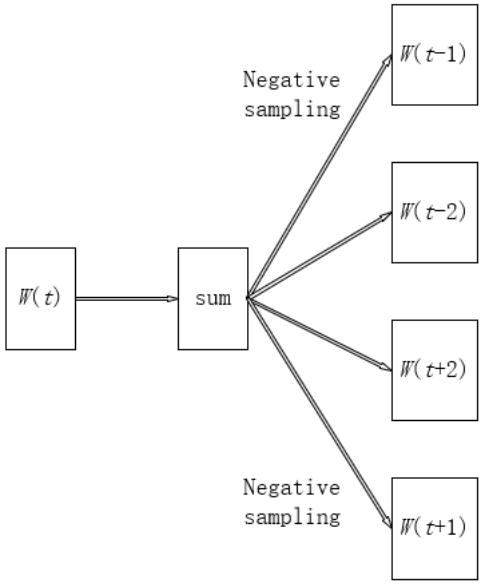

Figure 3: The structure of Skip-gram+Negative Sampling.

Negative Sampling can improve the training speed and the quality of the obtained word vector. Instead of updating all the weights for each training sample, Negative Sampling only updates a small part of the weights for each training sample, which reduces the amount of calculation in the process of gradient descent. If we train the sample, because we're using one-hot coding for words, we expect the output neuron for the word to produce 1, and all the remaining output neurons to produce 0, and we call all the output

neurons that produce 1 for the word "positive", all the neurons that output 0 correspond to the word "negative". Using the Negative Sampling method, instead of updating the weights corresponding to all the output neurons, we just randomly select a few "negative" words and update their corresponding weights. Of course, we'll also update the weights of "positive". The probability of a word being selected as "negative word" depends on how often it appears, with the more frequently it appears the easier it is to use as "negative word".

3.3 Matching Evaluation Metrics

Generally, precision(P), recall(R), and F-measure(F) are utilized to test the matching results' quality[22].

$$R = \frac{correctfoundcorrespondences}{allpossiblecorrespondences} \tag{1}$$

$$P = \frac{correctfoundcorrespondences}{allfoundcorrespondences} \tag{2}$$

$$F = \frac{2 \times R \times P}{R + P} \tag{3}$$

where P and R respectively indicate the accuracy and completeness of the results. P equals 1 denoting all found correspondences are correct, while R equals 1 representing that all correct correspondences are found; F is the harmonic mean of P and R to balance them.

4 METHODOLOGY

Deep learning-based ontology matching technology generally requires a specific external knowledge base to train neural networks, and the ontology matching technology using character embedding lacks the semantic information of words, and also abandons the plug and play pre-trained word vector ecosystem. In order to solve the above problems, we propose word embedding matching technology based on deep learning. This is a technique for converting words directly into vectors by training. It does not require a specific external knowledge base, as long as enough words are given, the next step can be matched. At the same time, the ontology matching technology proposed by us is based on word2vec tool for neural network modeling.

Ontology matching matches not only matched words, but also matched sentences. However, our model only trained words. In order to use the word embedding model, we preprocessed sentences to make maximum use of our model, so as to obtain good matching results. The preprocessing stage is divided into two parts. The first part is to remove special symbols and convert case to case. The second part processes sentences into words and removes meaningless words, such as: 'is', 'a', etc. In order to improve the matching result, we change the words into their original word form.

Our model converts words into vectors. In order to calculate the similarity between vectors, cosine similarity is used for measurement. In order to optimize the threshold value of similarity matrix in the matching process, the best matching result is obtained. We test the sensitivity of the threshold value of the matched similarity matrix. The parameter configuration of model training and sensitivity experiments on thresholds will be given in the experiment section. The Pseudo code of ontology matching algorithm based on deep learning is shown in Algorithm 1.

Algorithm1:Word Embedding Based Ontology Matching Technique

1:**Preprocess**: Extract the elements of each line of text
2:**Input**: Each line of text files
3:**Output**: F-measure
4:**if** The input is a word **then**
5: Remove special symbols and case conversion operations;
6: Training model;
7: The text documents of two ontologies O_1 and O_2 are converted into sets;
8: The similarity measurement of words is calculated;
9: The similarity measurement matrix is obtained;
10:**else**
11: Cut sentences into words, remove words with no actual meaning and restore each word to its original form;
12: Remove special symbols and case conversion operations;
13: Training model;
14: The similarity measurement of ontology O_1 and O_2 word sets is calculated;
15: The similarity measure matrix is obtained;
16:**end if**
17: The similarity matrix is merged into a matrix;
18: Remove every element less than the threshold value in the matrix;
19: F-measure is used to obtain the output results;

As shown in Algorithm 1. First, we extract the elements of each line from the text. Secondly, when we input words, we convert all words in the text to lowercase, and then provide these words to the neural network model for training. The purpose of converting text to a collection is to manipulate words in the text. The next step is to calculate the similarity between two ontologies using cosine similarity measurement technique and get the similarity matrix. If the input is sentence text, the difference is that it needs to be processed. At this time, the model trains all words except meaningless words in the sentence. At the same time, in the similarity calculation, since a sentence becomes a word set, two word sets are used to measure the similarity and finally a similarity value is obtained.

The ontology matching technology based on word embedding proposed by us essentially uses neural network for training. Compared with the traditional ontology matching technology, it does not need a lot of manpower and avoids human-caused errors. Compared with the ontology matching of other neural networks mentioned above, our method does not need specific knowledge base and can make the matching result better. In view of the ontology matching technology based on machine learning proposed recently, such as: Bento et al. [3] used convolutional neural networks to carry out the ontology matching which shows good performance. Khoudja Et al. [2] Integrated the state-of-the art ontology matchers through the neural network to improve the results quality. Jiang et al. [17]

proposed a Long Short-Term Memory Networks based ontology matching technique to matching biomedical ontologies. The methods proposed in this three articles all require reference alignment, while our approach does not.

The proposed ontology matching technique based on word embedding not only generally outperforms some traditional ontology matching techniques in experimental results, but also runs fast. The calculation of time complexity is divided into two serial parts. We calculate the time complexity of Skip-gram model and the space-time complexity of matching process. When Skip-gram is used for prediction, the number of predictions is almost equal to the number of words in the text, so the time complexity should be O(n). Since the similarity between entities of different ontologies needs to be calculated, the space-time complexity of this matching process is O(n^3). After practical experiments, the whole running time should be about 6 seconds. In general, the performance of ontology matching based on word embedding is quite good.

5 EXPERIMENT

5.1 Experimental Configuration

To evaluate the proposed ontology matching method of word embedding, we use OAEI benchmark data set. It has a total of 54 test cases. We will test some typical cases. These benchmark test cases can be divided into five categories. Test cases 101 through 104 have the same label and hierarchical ontology. Ontology of 201 - 210 test cases has similar hierarchical structure. Test cases 221 through 247 have similar label representations. In the test cases 248 through 266, the label representation and hierarchy are different. Test cases 301 through 304 are real-time cases from different organizations.

In the configuration of word2vec neural network, we set the seed number to 42, iteration number to 10 and context window length to 1 according to previous experience. The vector dimension was set to 3 by sensitivity test, and the threshold used to filter the similarity value in the similarity matrix was set to 0.8 by sensitivity test.

Table1 shows the experimental results of our proposed method, and table2 shows the comparison of f-measure values between our method and current advanced methods.

5.2 Sensitivity Testing

When conducting sensitivity experiments, we first set the threshold to 0.8, and conducted sensitivity tests for vector dimensions, as shown in Figure 4. In Figure 4, x is the vector dimension, and y is the average value of F-measure in the test set. The vector dimension was set as 3-dimensional when sensitivity test was carried out for thresholds. When threshold x was 0.7, 0.75, 0.8, 0.85 and 0.9 respectively, the average value of F-measure measured in the test set was y, as shown in Figure 5.

It can be seen from FIG. 4 that the average value of F-measure is 0.965 at the maximum in dimension 3, and then with the increase of vector dimension, the mean value of F-measure on the whole test set gradually decreases. Meanwhile, it can also be seen from Figure 5 that F-measure reaches the peak value of 0.965 when the

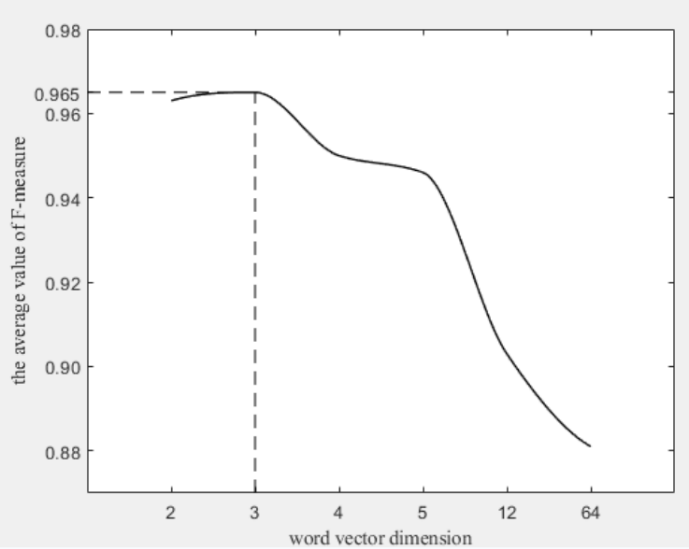

Figure 4: Sensitivity testing on vector dimensions.

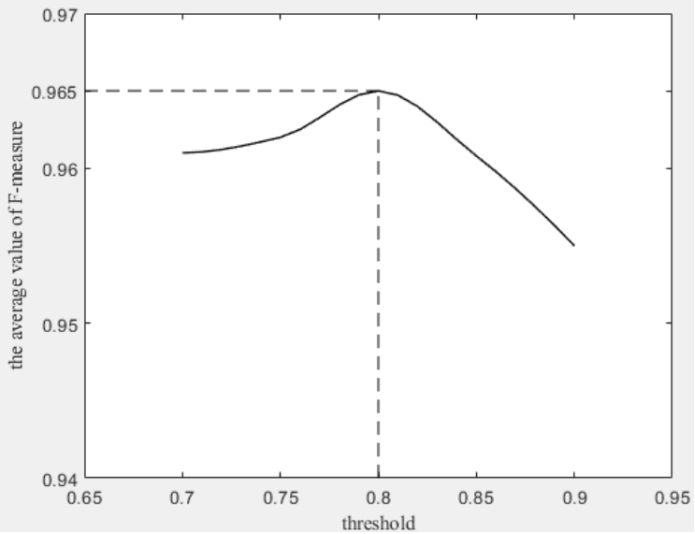

Figure 5: Sensitivity testing on the threshold.

threshold value is 0.8. By combining the two figures, we select the threshold value of 0.8 and the vector dimension of 3 to conduct experiments on the whole OAEI benchmark test set.

5.3 Experimental Results

We refer to the data set of the comparison between the ontology matching methods proposed by other outstanding scholars and the advanced ontology matching methods [3], [21], and compare our method with the current advanced ontology matching technology. Table1 shows the recall, precision and f-measure values of our approach on the test experiment set. Table2 is the comparison of F-measure values between our method and some current advanced methods. It can be seen from table 1 that the f-measure values of

Table 1: The quality of alignments obtained by our approach.

No.	F-measure	Recall	Precision	No.	F-measure	Recall	Precision
101	1.00	1.00	1.00	230	1.00	1.00	1.00
103	1.00	1.00	1.00	231	1.00	1.00	1.00
104	1.00	1.00	1.00	232	1.00	1.00	1.00
201	0.74	0.60	0.97	233	1.00	1.00	1.00
204	0.95	0.91	1.00	236	1.00	1.00	1.00
205	0.85	0.76	0.96	237	1.00	1.00	1.00
206	0.82	0.72	0.96	238	1.00	1.00	1.00
207	0.79	0.67	0.96	239	1.00	1.00	1.00
221	1.00	1.00	1.00	240	1.00	1.00	1.00
222	1.00	1.00	1.00	241	1.00	1.00	1.00
223	1.00	1.00	1.00	246	1.00	1.00	1.00
224	1.00	1.00	1.00	247	1.00	1.00	1.00
225	1.00	1.00	1.00	301	0.96	0.93	1.00
228	1.00	1.00	1.00	302	0.95	1.00	0.90

Table 2: Comparison with OAEI's participants in terms of F-measure.

No.	Edna	AgrMaker	AROMA	ASMOV	CODI	Ef2Match	Falcon	GeRMeSMB	MapPSO	RiMOM	SOBOM	TaxoMap	SNN-OM	Our proposal
101	1.00	0.99	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
103	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
104	1.00	0.99	0.99	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
201	0.04	0.92	0.95	1.00	0.13	0.77	0.97	0.94	0.42	1.00	0.95	0.51	0.97	0.74
204	0.93	0.97	0.97	1.00	0.74	0.99	0.96	0.98	0.98	1.00	0.99	0.51	0.99	0.95
205	0.34	0.92	0.95	0.99	0.28	0.84	0.97	0.99	0.73	0.99	0.96	0.51	0.98	0.85
206	0.54	0.93	0.95	0.99	0.39	0.87	0.94	0.92	0.85	0.99	0.96	0.51	0.96	0.82
207	0.54	0.93	0.95	0.99	0.42	0.87	0.96	0.96	0.81	0.99	0.96	0.51	0.89	0.79
221	1.00	0.97	0.99	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
222	0.98	0.98	0.99	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	0.46	1.00	1.00
223	1.00	0.95	0.93	1.00	1.00	1.00	1.00	0.96	0.98	0.98	0.99	0.45	1.00	1.00
224	1.00	0.99	0.97	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	0.51	1.00	1.00
225	1.00	0.99	0.99	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
228	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
230	0.85	0.90	0.93	0.97	0.98	0.97	0.97	0.94	0.98	0.97	0.97	0.49	1.00	1.00
231	1.00	0.99	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.51	1.00	1.00
232	1.00	0.97	0.97	1.00	0.97	1.00	0.99	1.00	1.00	1.00	1.00	0.51	1.00	1.00
233	1.00	1.00	1.00	1.00	0.94	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00
236	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
237	0.98	0.98	0.97	1.00	0.99	1.00	0.99	1.00	0.99	1.00	1.00	0.46	1.00	1.00
238	1.00	0.94	0.92	1.00	0.99	1.00	0.99	0.96	0.97	0.98	0.98	0.45	1.00	1.00
239	0.50	0.98	0.98	0.98	0.98	0.98	1.00	0.98	0.98	0.98	0.98	0.94	0.98	1.00
240	0.55	0.91	0.83	0.98	0.95	0.98	1.00	0.85	0.92	0.94	0.98	0.88	1.00	1.00
241	1.00	1.00	0.98	1.00	0.94	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00
246	0.50	0.98	0.97	0.98	0.98	0.98	1.00	0.98	0.98	0.98	0.95	0.94	0.98	1.00
247	0.55	0.88	0.80	0.98	0.98	0.98	1.00	0.91	0.89	0.94	0.98	0.88	1.00	1.00
301	0.59	0.59	0.73	0.86	0.38	0.71	0.78	0.71	0.64	0.73	0.84	0.43	0.86	0.96
302	0.43	0.32	0.35	0.73	0.59	0.71	0.71	0.41	0.04	0.73	0.74	0.40	0.75	0.95

this method are above 0.95 except that the results of 201, 205, 206, 207 test sets are not particularly high. As can be seen from table 2, the experimental results of this method are generally higher than those of other methods. SNN-OM is an ontology matching technology based on siamese neural network, which is proposed by Xue et al. [21]. In addition, this technology combines Alignment refinement technology of matched ontology in this paper to achieve high quality of final matching results. It can be seen from the table that although our matching results in 20X test cases was slightly worse, the matching results in 30X test cases was stronger than SNN-OM. And in the whole test set is not weaker than the more advanced SNN-OM in this field.

6 CONCLUSION

Web of Things(WoT) can facilitate knowledge discovery and solve heterogeneous problems between different ontologies, and with the rise of deep learning, ontology matching technology based on deep learning is developing gradually. However, most matching techniques proposed by experts rely on specific external knowledge base and cannot be extended. In addition, some neural networks are embedded with characters and lack semantic correlation between words. To solve the above problems, we propose a deep learning ontology matching technology based on word embedding, which makes the whole match result more efficient, fast, intuitive and accurate. And because our approach does not rely on a specific knowledge base, the results are even robust. By translating words into vector space, the problem is solved cleverly, especially in the 30X test sets, the f-measure value is above 0.95, and compared with other advanced methods, our proposed method is generally superior to other methods.

Although the word embedding method has great potential, for example, it can be combined with other neural networks to aggregate different similarity measures. But there are problems with this approach. Like this approach requires a large number of words to

train the model. Meanwhile, the model is limited in terms of the number of words that can be processed. There are also problems in dealing with spelling mistakes and rare words.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 62172095), the Natural Science Foundation of Fujian Province (No. 2020J01875), the Scientific Research Foundation of Fujian University of Technology (No. GY-Z17162).

REFERENCES

[1] Reina Akama, Kento Watanabe, Sho Yokoi, Sosuke Kobayashi, and Kentaro Inui. 2018. Unsupervised learning of style-sensitive word vectors. *arXiv preprint arXiv:1805.05581* (2018).

[2] Meriem Ali Khoudja, Messaouda Fareh, and Hafida Bouarfa. 2018. A new supervised learning based ontology matching approach using neural networks. In *International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning*. Springer, 542–551.

[3] Alexandre Bento, Amal Zouaq, and Michel Gagnon. 2020. Ontology matching using convolutional neural networks. In *Proceedings of the 12th language resources and evaluation conference*. 5648–5653.

[4] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* (2018).

[5] Ondrej Chum, James Philbin, Andrew Zisserman, et al. 2008. Near duplicate image detection: Min-hash and TF-IDF weighting.. In *Bmvc*, Vol. 810. 812–815.

[6] Gerard Clement and Jacqueline Stern. 1986. Constant chromomagnetic field in the soliton bag model. *Physical Review D* 34, 5 (1986), 1581.

[7] Hong-Hai Do and Erhard Rahm. 2007. Matching large schemas: Approaches and evaluation. *Information Systems* 32, 6 (2007), 857–885.

[8] Vivek Iyer, Arvind Agarwal, and Harshit Kumar. 2020. Multifaceted context representation using dual attention for ontology alignment. *arXiv preprint arXiv:2010.11721* (2020).

[9] Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu. 2005. Falcon-ao: Aligning ontologies with falcon. In *Proceedings of K-CAP Workshop on Integrating Ontologies*. 85–91.

[10] Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2018. Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 787–798.

[11] Zhang Lili and Jinghua Ding. 2014. Semantic Ontology Method of Learning Resource based on the Approximate Subgraph Isomorphism. *Journal of Multimedia* 9, 2 (2014).

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[14] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. Citeseer, 29–48.

[15] Lucy Lu Wang, Chandra Bhagavatula, Mark Neumann, Kyle Lo, Chris Wilhelm, and Waleed Ammar. 2018. Ontology alignment in the biomedical domain using entity definitions and context. *arXiv preprint arXiv:1806.07976* (2018).

[16] Zhenyu Wu, Yuan Xu, Yunong Yang, Chunhong Zhang, Xinning Zhu, and Yang Ji. 2017. Towards a semantic web of things: a hybrid semantic annotation, extraction, and reasoning framework for cyber-physical system. *Sensors* 17, 2 (2017), 403.

[17] Xingsi Xue and Junfeng Chen. 2019. Optimizing ontology alignment through hybrid population-based incremental learning algorithm. *Memetic Computing* 11, 2, 209–217.

[18] Xingsi Xue and Junfeng Chen. 2021. Matching biomedical ontologies through compact differential evolution algorithm with compact adaption schemes on control parameters. *Neurocomputing* 458 (2021), 526–534.

[19] Xingsi Xue, Junfeng Chen, and Xin Yao. 2018. Efficient user involvement in semiautomatic ontology matching. *IEEE Transactions on Emerging Topics in Computational Intelligence* 5, 2 (2018), 214–224.

[20] Xingsi Xue and Qihan Huang. 2022. Generative adversarial learning for optimizing ontology alignment. *Expert Systems* (2022), e12936.

[21] Xingsi Xue and Chao Jiang. 2021. Matching Sensor Ontologies with Multi-Context Similarity Measure and Parallel Compact Differential Evolution Algorithm. *IEEE Sensors Journal* 21, 21 (2021), 24570–24578.

[22] Xingsi Xue, Jianhua Liu, Pei-Wei Tsai, Xianyin Zhan, and Aihong Ren. 2015. Optimizing ontology alignment by using compact genetic algorithm. (2015), 231–234.

[23] Xingsi Xue and Wenyu Liu. 2022. Integrating Heterogeneous Ontologies in Asian Languages Through Compact Genetic Algorithm with Annealing Re-sample Inheritance Mechanism. *Transactions on Asian and Low-Resource Language Information Processing* (2022), 1–21.

[24] Xingsi Xue, Jiawei Lu, and Junfeng Chen. 2019. Using NSGA-III for optimising biomedical ontology alignment. *CAAI Transactions on Intelligence Technology* 4, 3 (2019), 135–141.

[25] Xingsi Xue and Jeng-Shyang Pan. 2017. A segment-based approach for large-scale ontology matching. *Knowledge and Information Systems* 52, 2 (2017), 467–484.

[26] Xingsi Xue and Jie Zhang. 2021. Matching large-scale biomedical ontologies with central concept based partitioning algorithm and adaptive compact evolutionary algorithm. *Applied Soft Computing* 106 (2021), 107343.