IETI Transactions on Engineering Research and Practice
http://ieti.net/TERP/

2022, Volume 6, Issue 1, 29-37, DOI 10.6723/TERP.202207_6(1).0003

CTO ORDER CLUSTERING FOR ELECTRONIC PRODUCTS BASED ON WORD VECTORS

Jiaming Wang[1,2,a], Xiaolan Xie[1,2,b,*], Bangjin Wang[3,c,*]
[1]The College of Information Science and Engineering, Guilin University of Technology Guilin 541000, China
[2]The Guangxi Key Laboratory of Embedded Technology and Intelligent System, Guilin 541000, China
[3]Zhejiang Academy of Surveying and Mapping, Hangzhou 310000, China
[a]1638382633@qq.com,[b] xie_xiao_lan@foxmail.com, [c]554397037@qq.com
* Correspondence: xie_xiao_lan@foxmail.com, 554397037@qq.com

Abstract The business model of configure to order has been developed on a large scale, and orders configured according to user needs are an important way to transform producers. Effective clustering of CTO (Configure To Order) orders can effectively reduce the operational costs of producers. In order to solve the above problem, a clustering model for CTO orders is developed in the analysis of user order configurations and a vector representation based on orders is designed. First, the Skip-gram model was applied to train on the electronic product dataset to obtain word vectors; then the similarity between orders was calculated using the vector representation of orders. The experimental results show that the algorithm has better vector differentiation ability in the similarity measure of orders compared to the topic-based feature representation and the word frequency-based feature representation. The clustering metric shows that the SG-CTO algorithm has better results.

Keywords: Skip-gram model; clustering; Configure To Order; order clustering.

1. INTRODUSTION

With the advancement of the e-intelligence industry and the increasing demands of customers, there are a number of issues that need to be addressed in order to achieve personalised product customisation: CTO (Configure To Order), where orders are freely configured and generated according to user-defined order requirements. This model meets the individual needs of the consumer, but CTO creates challenges for the producer. Production costs, line allocation and order categorisation need to be taken into account. This requires the producer to be able to allocate orders to the assembly of production lines in the company under optimal conditions in order to maximise the efficiency of the company's production. For the engineering to order (ETO) model, a large-scale order-driven model is used [1]. Due to the limitations of the ETO model, it increases the production and design tasks of the producer, as well as increasing the time costs at the order delivery level.

The efficient classification of orders cannot be neglected in the production process of a company. Machine learning has been widely applied to the task of classifying orders [2-4]. Kamishima et al [5] used the k-means algorithm [6] to classify orders, which was characterised by discrete encoding of the commodity attributes in the orders. Koskosidis et al [7] abstracted the order classification problem into a capacity clustering problem, proposing a better than heuristic algorithm Chen et al. proposed an integer planning based order clustering model by analyzing the correlation values between orders [8].

The process was automated, enabling automatic allocation of delivery units and reducing operational costs [9]

Order classification can be abstracted as a scenario-specific text clustering problem [10-11]. One-hot coding has been widely used to extract the location features of words from a word corpus, which in turn forms a feature representation of the text. the method is prone to dimensional disasters and feature sparsity due to the large size of the word corpus. Salton et al. proposed the TF-IDF algorithm by means of the frequency of words in weighted representation of the frequency of occurrence in articles, which can effectively reduce the feature dimensionality of texts [12]. The clustering phase of this algorithm uses the K-means algorithm to achieve text clustering. The TF-IDF algorithm also suffers from the problems of sparse data, lack of semantic information and shortcomings in the measurement of similarity between texts. Orders cannot get a good feature representation of orders due to the semantic sparsity problem. DM et al. proposed a topic-based feature representation method based on text classification [13], which uses topic features to represent the feature words in the text, reducing the text from semantic dimension to topic dimension and effectively reducing the generated vector dimension. However, order data is different from long text data, which is rich in semantic information and good semantic information exists. The order data has scattered topic features due to the limitation of the length of a single piece of data.

In summary, the text clustering problem has received much attention from scholars, while many improvements have been proposed and applied on many fields. However, a more refined solution is needed for the CTO order clustering problem. In this paper, we use the Skip-gram model to analyse the order text and generate word vectors with semantic relationships between words, and use the Euclidean distance between the word vectors to obtain the similarity between them. The similarity of the orders is then calculated using the order representation method defined in this paper and the similarity matrix between the orders is generated using the generated word vectors. Finally, the orders are clustered using the K-means clustering algorithm to achieve a summary of CTO orders for electronic products and to provide a basis for scheduling for decision makers.

2. CTO ORDER CLUSTERING BASED ON WORD VECTORS AND STREAMWISE DISTANCES

2.1. Experimental Framework

This paper uses a word vector fusion Euclidean distance metric for metrics, designs order vectorisation algorithms and implements CTO order packaging for electronic products through clustering algorithms. Figure 1 shows the experimental framework and demonstrates the experimental process. The figure mainly includes pre-processing of data, word vector representation, and order vectorisation. The model is trained with commodity words to obtain the similarity between the word vectors. This similarity is then applied to measure the similarity relationship between orders, and finally a clustering algorithm is used to complete the clustering task.

2022, Volume 6, Issue 1, 29-37, DOI 10.6723/TERP.202207_6(1).0003

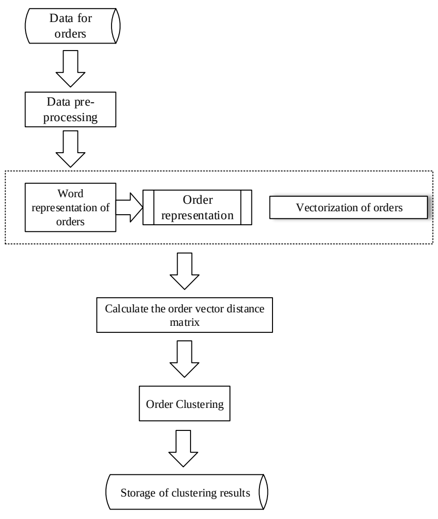
Data for orders
Data pre-processing
Word representation of orders
Order representation
Vectorization of orders
Calculate the order vector distance matrix
Order Clustering
Storage of clustering results

Figure 1. Experimental framework of CTO order clustering based on word vectors and manifold distances.

2.2. Vectorisation of Goods

The vectorisation of orders is based on the development of word vectors. The methods for generating word vectors are usually based on the words represented around the word. One-hot methods generate word vectors by counting the frequency of words, but this approach is prone to problems such as dimensional disasters. The basic idea of NNLM (neural network language model) is to encode words using a neural network model, which is also based on the representation of the words around the word, incorporating semantic information from the context [14]. The Skip-Gram model is trained by a neural network using a single word as input and the context of the word as the object to be fitted, and the model is trained by a lossy Han formula [15]. As order data is small text data, using this type of word vector to characterise order data can effectively circumvent the risk of vector sparsity. For personalised electronics CTO orders, there may be dependencies between different electronic accessories. Not only does the mere counting of word frequencies not reveal the dependencies between accessories, but it also does not allow the low-dimensional vectorisation of order data.

In this paper, we use the Skip-Gram model in the word2vec tool to vectorise personalised electronic CTO orders, which is an unsupervised learning method for learning contextual information from large-scale unlabelled text data to obtain a low-dimensional vector representation of text data. The item or sequence of items in a CTO order is input to the model by masking that item as the training target. The mechanism of the algorithm is to use a sliding window as the input to the model, allowing the generated vectors to have contextual information. The sliding window mechanism also effectively ignores words in the CTO order that do not affect understanding, enhancing the generalisation performance of the model over the CTO order. skip-Gram model training is shown in Figure 2.

2022, Volume 6, Issue 1, 29-37, DOI 10.6723/TERP.202207_6(1).0003

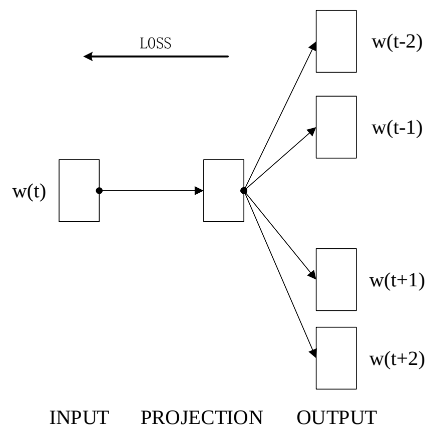

Figure 2. Skip-Gram model.

As shown in the figure, the model contains a three-layer neural network with an input layer, a mapping layer and an output layer that uses the input vocabulary to make predictions about the context. Suppose an electronic product CTO order involves *n* accessories; the order is denoted as $\{w_1,w_2\ldots\ldots w_{n-1},w_n\}$; the *ith* accessory in the order is denoted as w_i, Assuming that the current item is w_i and the size of the sliding window is *C*, the set of context words is denoted as *Context(*w_i*)*.

$$Context(w_i)=w_{i-c},w_{i-c+1}\ldots w_{i-1},w_{i+1}\ldots w_{i+c} \tag{1}$$

The objective function of the model is to maximise the log probability, defined as

$$L=\sum_{i=c}^{n-c}\log p(Context(w_i)\mid w_i) \tag{2}$$

The formula $p(Context(w_i)|w_i)$ is expressed as

$$p(context(w_i)\mid w_i)=\prod_{u\in Context(wi)} p(u\mid wi) \tag{3}$$

The model training output gives a low-dimensional dense vector for each commodity. For each of the *N* commodities $N\in R^{m*t}$ has a vector representation, where *m* represents that each commodity vector has *m* dimensions and *t* is the number of commodity types. From this, the distance matrix between each commodity can be calculated, and the similarity measure between commodities is then expressed in terms of Euclidean distance as

$$dis\tan ce(x_i,x_j)=\left\|x_i-x_j\right\|^2 \tag{4}$$

Where x_i， $x_j\in N$. The smaller the distance between the two items, the more similar the two items are in terms of structural alignment and semantics. Conversely, the larger the distance, the less similar the two items are.

2022, Volume 6, Issue 1, 29-37, DOI 10.6723/TERP.202207_6(1).0003

2.3. Order Vectorization

There are similarities and differences in the characteristics of the item listings in CTO orders and short text books. In the CTO order, the items are arranged in a more orderly manner and are generally of the same type for a given item. In contrast to the short text, the short text has a variety of grammatical collocations, and there is a risk of synonymy in the feature words in the short text, which interferes with the algorithm. A text similarity algorithm is defined as the use of a specific algorithm to measure the degree of similarity between two texts. After obtaining the similarity matrix of the goods, it is possible to expand the similarity between orders by using the similarity matrix of the goods. The similarity between item sequence 1 and item sequence 2 is calculated as shown in Figure 3. Where the similarity between items S1 and Q1 can be found by the similarity of the items, which in turn gives the similarity vector represented by the two orders.

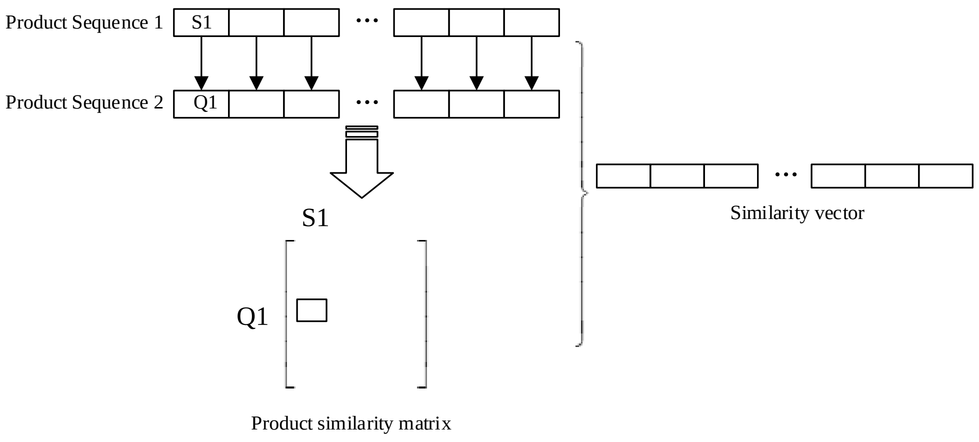

Figure 3. Schematic representation of the order similarity vector.

As orders are less distinctive compared to short texts, orders are also characterised by the presence of serialisation. This paper therefore uses the similarity vector represented by the two generated orders to sum up and obtain the similarity measure between orders *Matrxi*(m_i,m_j), where order $mi=\{w_{i1},w_{i2}\ldots\ldots,w_{in}\}$ and order $m_j=\{w_{j1},w_{j2}\ldots\ldots w_{jn}\}$. The similarity measure between the orders, *Matrxi*(m_i,m_j), is then fused with the K-means algorithm to complete the clustering of the CTO orders.

2.4. Algorithm Flow

The algorithm uses the Skip-Gram model to vectorise the items in the order and uses the shortest distance between items to measure the item similarity and form an inter-item similarity matrix. At the same time, the characteristics of the sequence of items in the CTO order are analysed and a similarity measure between orders is defined. Incorporating the above ideas, the algorithm flow is as follows.

Algorithm: CTO order clustering based on word vectors
Input: set of orders $M=\{m_1,\ m_2\ldots m_n\}$
Output: labels of order clustering results

2022, Volume 6, Issue 1, 29-37, DOI 10.6723/TERP.202207_6(1).0003

Step 1: pre-processing of each order by word separation.
Step 2: train the Skip-Gram model using the feature words obtained from the order dataset to obtain the feature word vector set $\{x_1, x_2 \ldots x_n\}$;
Step 3: calculating the similarity matrix between the feature word vectors and forming the similarity matrix between orders.
Step 4: randomly selecting orders as the centre of mass for the K-means algorithm.
Step 5: metering through the similarity matrix between orders and selecting the orders that are most similar to the orders selected in Step IV as a cluster and assigning similar labels.
Step 6: recalculation of the centre of mass.
Step 7: executing steps V and VI in a loop until the centre of mass no longer changes.
Step 8: Output the labelling results

3. EXPERIMENT AND ANALYSIS

3.1. Experimental Data and Evaluation Indexes

In this paper, we take the PC side of the computer assembly as an example, and use crawling technology to collect a total of more than 1400 computer personalization order data from the public e-commerce platform. Each order data includes screen, motherboard, CPU, graphics card, memory, hard disk, etc., forming an order sequence. The dataset selected for this paper is a Chinese dataset, with each order having a commodity dimension of 10. Dividing orders into three categories according to the price range of brain order data and assigning labels. There may be missing values in PC customisation orders, for example, a graphics card may not be required for a complete machine assembly. This paper does fill in such missing values. The algorithm in this paper runs on a platform environment of 64bit windows11, a programming environment of python3.6, a computer running on 16G of RAM and an Intel-i7 CPU.

In this paper, the dataset is validated by data cleaning and using the algorithm model on the dataset. The performance of the model was measured using the silhouette coefficient and Davies-Bouldin index respectively. The silhouette coefficient is a commonly used internal validity metric in clustering algorithms and can be used to measure the performance of clustering results. The profile coefficient has a value range of [-1,1], with the index converging to -1 indicating poor clustering performance and converging to 1 indicating good clustering results. The definition is as follows.

For a single data the contour coefficient is

$$S(x_i) = \frac{b(x_i) - a(x_i)}{\max(a(x_i), b(x_i))} \tag{5}$$

where $a(x_i)$ is defined as the average distance of data point x_i from other data points in the same cluster class, expressed as

$$a(x_i) = Average_{i \in A, j \in A, i \neq j}(dist(x_i, x_j)) \tag{6}$$

where $b(x_i)$ is defined as the minimum of the average distance values of the data points x_i from the data points of the other class clusters, defined as

$$dist(x_i, C) = average_{j \in C}(dist(x_i, x_j)), i \in A, C \neq A \tag{7}$$

$$b(x_i) = \min_{C \neq A} dist(x_i, C) \tag{8}$$

The contour coefficients for the entire data set are defined as the average of the single datacontour coefficients.

$$S = \frac{1}{N}\sum_{i=1}^{N} s(x_i) \tag{9}$$

The Davidson Boudin Index (DBI Index) is a measure of how good a clustering algorithm is, based on the distance between class clusters and the size of the clusters themselves. The metric is defined as follows.

$$DBI = \frac{1}{N}\sum_{i=1}^{N} \max_{j \neq i}\left(\frac{S_i + S_j}{\| w_i - w_j \|^2}\right) \tag{10}$$

where S_i represents the average distance from the data points in a class cluster to the centre of mass in the cluster, reflecting the closeness of the data points in the class cluster to each other. Where w_i represents the centre of mass of the Si class cluster and w_j represents the centre of mass of the S_j class cluster. For each class cluster the maximum similarity value to the other class clusters is found, and the average of the maximum similarity values for all class clusters is the Davidson Fortin index.

Where the clustering external indicator uses *ACC*, which is defined as follows.

$$ACC = \frac{\sum_{i=1}^{n} \delta(V_i, U_i)}{N} \tag{11}$$

where $\delta(x,y)$ indicates that it takes the value 1 when $x=y$ and 0 for the rest.

3.2. Analysis of Experimental Results

The algorithm requires the determination of the hyperparameters in the Skip-gram model, where the size of the sliding window C is set to 3. Also, in order to verify the performance of the proposed algorithm, this paper compares with the TF-IDF method based on word frequency statistics and the topic-based LDA method, and uses the K-means algorithm in the final clustering stage. The paper also uses contour coefficients and Davidson's Fortin index to measure the clustering results. After several experiments, the best results were selected for each algorithm as the final comparison in order to ensure the fairness of the algorithms. The metric results of each algorithm on the CTO order clustering task are shown in Table 1.

Table 1. Results of the indicator review.

datasets	TF-IDF-CTO	LDA-CTO	SG-CTO
silhouette coefficient	0.118	0.127	0.374
DBI	0.953	1.230	0.684
ACC	0.605	0.379	0.673

As can be seen from Table 1, the TF-IDF method based on word frequency statistics shows the presence of aggregation properties in the metrics compared to the algorithm proposed in this paper, which indicates the reasonableness of the order vectorisation method proposed in this paper. The TF-IDF algorithm outperforms the topic-based algorithm in comparison with the TF-IDF algorithm, indicating that the probability distribution between items does not reflect the topic-based characteristics of orders when processing CTO order data due to the small number of item characteristics of orders. The Skip-gram model uses a sliding window mechanism to effectively fuse the relational features between contextual items, maximising the constraint properties between items and better solving the task of CTO order clustering. Experiments show that the word vectors trained using the large-scale corpus are improved on the CTO order clustering task using the same clustering algorithm as the algorithms proposed in this paper.

Each algorithm uses Euclidean distance for the distance metric of the feature vectors of the feature words. This paper also explores the impact of different distance metrics on the order clustering task, using Euclidean distance and cosine distance applied to the commodity feature words respectively. The experimental results are shown in Table 2.

Table 2. Effect of different distance metrics on the results.

	silhouette coefficient			DBI		
Distance Metric	TF-IDF-CTO	LDA-CTO	SG-CTO	TF-IDF-CTO	LDA-CTO	SG-CTO
Euclidean distance	0.118	0.127	0.374	0.9531	1.2295	0.6841
Cosine distance	0.124	0.137	0.343	1.114	1.3455	0.7834

The experimental results show that the Euclidean distance metric performs better overall than the cosine distance in order classification tasks. Due to the sequential nature of the feature words in the orders, the Euclidean distance metric is better able to handle this type of data.

4. SUMMARY

In this paper, a solution to the CTO order clustering problem is proposed. The algorithm uses the Skip-gram model to represent the text feature word vectors, which solves the problem of sparse order text data on the clustering results. The vector representation of CTO orders is also proposed, which can effectively represent the similarity between orders based on word vectors and make the clustering results more distinguishable. The experimental data shows that this algorithm is more suitable for the CTO order clustering task than the methods based on word frequency statistics and topic features, and can effectively improve the clustering results.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No.61762031), The Science and Technology Major Project of Guangxi Province (NO.AA19046004), and The Natural Science Foundation of Guangxi (No.2021JJA170130).

References

[1] Chen-Ritzo C.-H., Ervolina T., Harrison T. P. and Gupta B., 2011, Component rationing for available-to-promise scheduling in configure-to-order systems, *European Journal of Operational Research*, 211(1), pp. 57-65.
[2] Mahesh B., 2020, Machine learning algorithms - a review, *International Journal of Science and Research (IJSR)*, 9, pp. 381-386.
[3] Xu D. and Tian Y., 2015, A comprehensive survey of clustering algorithms, *Annals of Data Science*, 2(2), pp. 165-193
[4] Benabdellah A. C., Benghabrit A.and Bouhaddou I., 2019, A survey of clustering algorithms for an industrial context, *Procedia computer science*, 148, pp. 291-302.
[5] Kamishima T. and Fujiki J., 2003, Clustering orders, *International Conference on Discovery Science*, pp. 194-207, Springer, Berlin, Heidelberg.
[6] MacQueen J., 1967, Some methods for classification and analysis of multivariate observations, *5th Berkeley Symp. Math. Statist. Probability*, pp. 281-297, USA.
[7] Koskosidis Y. A. and Powell W. B., 1992, Clustering algorithms for consolidation of customer orders into vehicle shipments, *Transportation Research Part B: Methodological*, 26(5), pp. 365-379.
[8] Chen M.-C. and Wu H.-P., 2005, An association-based clustering approach to order batching considering customer demand patterns, *Omega*, 33(4), pp. 333-343.
[9] Dong C., Tian'en C., Shuwen J., Chi Z., Cong W. and Mengyao L., 2020, Optimal Model of Chicken Distribution Vehicle Scheduling Based on Order Clustering, *Smart Agriculture*, 2(4), pp. 137.
[10] Abualigah L., Gandomi A. H., Elaziz M. A., Hamad H. A., Omari M., Alshinwan M. and Khasawneh A. M., 2021, Advances in meta-heuristic optimization algorithms in big data text clustering, *Electronics*, 10(2), pp. 101.
[11] Choudhary B. and Bhattacharyya P., 2002, Text clustering using semantics, *Proceedings of the 11th International World Wide Web Conference*, pp. 1-4.
[12] Ramos J., 2003, Using tf-idf to determine word relevance in document queries, *Proceedings of the first instructional conference on machine learning*, 242(1), pp. 29-48, New Jersey, USA.
[13] Blei D. M., Ng A. Y. and Jordan M. I., 2003, Latent dirichlet allocation, *Journal of machine Learning research*, 3(Jan), pp. 993-1022.
[14] Mikolov T., Kombrink S., Burget L., Černocký J. and Khudanpur S., 2011, Extensions of recurrent neural network language model, *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5528-5531, IEEE.
[15] Mikolov T., Chen K., Corrado G. and Dean J., 2013, Efficient estimation of word representations in vector space, *arXiv preprint arXiv:1301.3781*.