A Data-Driven Method for Diagnosing ATS Architecture by Anomaly Detection

Aimin Zhou[1], Shaowu Cheng[1](✉), Xiantong Li[1], Kui Li[1], Linlin You[2], and Ming Cai[2]

[1] School of Transportation Science and Engineering, Harbin Institute of Technology, Harbin 150090, China
csw_h@hit.edu.cn

[2] School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou 510006, China

Abstract. Autonomous Transport System (ATS) architectures enable a wide range of new applications and bring significant benefits to transport systems. However, during the design stage, errors of the architecture can have an impact on the smooth implementation of the ATS, which will endanger the normal operation of the transport systems. To ensure a high autonomy of the ATS architecture, i.e., "functionally evolvable, logically reconfigurable and physically configurable", the detection of ATS architecture design errors is essential. This paper aims to fill the research gap in the existing research on diagnosing or evaluating ATS architectures. Inspired by word embedding models in natural language processing communities, we propose a data-driven approach to diagnose ATS architectures without prior knowledge or rules. We use an architecture embedding model to generate vector representations of ATS architectures, then train the model through negative sampling of the training dataset to identify the features of abnormal ATS architecture. Finally, we employ the trained model to classify structural errors of the test dataset generated from the ATS architecture. The experimental results show that the proposed method gains a relatively good effect of classifying with an average accuracy of 79.3%, demonstrating the effectiveness of the method.

Keywords: Autonomous Transport System · Architecture embedding model · Triple classification · Vector computation

1 Introduction

With the development of technologies such as self-driving cars and cooperative vehicles infrastructure system, existing transport systems are evolving from intelligent transport systems to autonomous transport systems. To reduce human intervention, autonomous transport systems transport passengers and goods through self-organized operations and autonomous services. All these systems have one common character: Inside the system, independent components have to communicate with others to avoid incomplete or unclear communication mechanisms and inconsistent information quality that complicates the introduction of new services and the involvement of new stakeholders and even blocks this process, it is essential to introduce a unified ATS architecture. An ATS architecture integrates transport services, communications networks, vehicles, transport infrastructure, with traffic participants to provide a steady, trustworthy, secure, and privacy-friendly environment for users. The design process of an ATS architecture involves a great deal of repetitive and specialized work that requires a wealth of knowledge and careful reasoning ability, and analytical skills. The components and interactions of the ATS architecture are in continuous evolution and iteration. The above process may generate new and potential errors in the ATS architecture. Therefore, a scientific diagnosis approach is required to ensure the validity and reliability of the ATS architecture throughout the whole process of architecture evolution and iteration. By diagnosing the ATS architecture designed for a given city or region, potential errors in the architecture can be detected and fixed before implementing the ATS. It is good practice to implement such assurance procedures in the development of designing or modifying an ATS architecture. The diagnosis characteristics can be stored in the computer and reusable. However, no feasible methods that can be applied to the diagnosis process without prior knowledge or rules. In this paper, we propose a data-driven approach that represents the structural features of the ATS architecture through vector representation generated by the ATS architecture embedding model. Then the potential errors in the ATS architecture will be detected by vector computation without additional rules. The method simplifies the anomaly detection process into vector computation, which is relatively suitable for the continuously updated ATS architecture.

The paper is structured as follows: Sect. 2 describes the ATS architecture diagnosis problem and provides an overview of the proposed method. Then our diagnosis approach is presented in Sect. 3. In Sect. 4, we give numerical examples based on dataset from the national ITS reference architecture to demonstrate the effectiveness of the proposed method, and finally, Sect. 5 summarizes this paper.

2 Problem Description and Methodology Overview

Autonomous Transport System transports equipment, traffic participants, goods, information, or resources from one point to another with minimal human intervention [1, 2]. ATS exists in a variety of transportation modes such as trucks, buses, rail, ships, and even aircraft. At the early stage they are typically deployed in controlled industrial areas but are expected to be deployed soon in public areas with various degrees of autonomy. Unlike Autonomous Vehicles (AVs) [3], which offer excellent services to individual passengers, ATS integrates vehicles, freight, traffic participants, infrastructure, and information into a system to meet particular needs. When it comes to ATS architecture, the focus is on the interactions between components in a complex system to ensure the efficiency and the stable operation of ATS.

The ATS architecture is a static framework for giving macro guidance for the transport system [4]. Designing an ATS architecture involves lots of repeatable and technical work that requires fund of knowledge and careful reasoning ability and analytical skills. With continuous evolution and iteration of the ATS architecture, the components and interactions can change accordingly. While unforeknown errors may be generated in the ATS architecture. Detecting potential errors at the architecture design stage can help to reduce cost and improve productivity [5]. The architecture diagnosis checks over the internal logical relations, providing a basis for tracing and solving problems of the ATS architecture. It is vital in ensuring the valid deployment of an ATS through the diagnosis techniques.

With the rise of artificial intelligent technology, diagnosis methods that focus on sensors monitoring and signal processing as core tasks have gradually transitioned to diagnosis approaches based on knowledge [6–8]. The diagnosis knowledge can be stored in the computer and reusable. Therefore, the theoretical ATS architecture will relatively match up with the knowledge-based diagnosis approaches.

Currently, knowledge-based diagnosis methods [9–12] can be broadly classified into two categories: symbolic logic inference methods and representation learning methods. Traditional symbolic logic approaches focus on deterministic deductive reasoning and achieve inference diagnosis by defining ontological axioms or logical rules, which have the advantage of being precise and interpretable. While the main disadvantage is that they require the manual definition of logically strict inference rules, so their coverage is narrow and not easily extended. Besides, they cannot handle implicit or uncertain knowledge. To avoid the manual definition of rules, another kind of symbolic inference method uses statistical models of inductive reasoning to automatically learn rules from a large amount of facts. It can generalize abstract logical rules by learning the common features of tagged cases. The main advantage is that it reduces the workload of manually defining rules, but rule learning consumes too much computing resource and cannot represent implicit or uncertain knowledge either.

The representation learning diagnosis method transforms both entities and relations into the vector space and completes inferential diagnosis by vector computation [13], using low-dimensional dense vectors to represent entities and relations. The parameterized vector is an approximate representation in the vector space based on the existing knowledge in the knowledge graph as a sub-supervised signal. On the other hand, vector-based or neural network based inference computing is an approximate inference result obtained through differentiable representation learning [14]. Therefore, the inference process and result are also uncertain. Thus, representation learning methods and neural network approaches are easier to represent uncertain knowledge and implement uncertain reasoning than logical inference and symbolic rule-based approaches. In addition, the inference is more efficient as the inference process is simplified to a vector computation, eliminating the need for symbolic matching and rule search.

The ATS architecture is a flexible and future-proof architecture that guides the evolution and iteration of the transport system. Therefore, ATS architecture diagnosis also accompanies the whole process of architecture evolution and iteration. The above process may cause unpredictable and artificial errors in the ATS architecture. These changes in architecture caused by the evolution and iteration require updating diagnosis knowledge frequently to accommodate them. Since the dynamic problem is not well solved by traditional logical reference methods, we need a scientific diagnosis approach to ensure validity and reliability. Inspired by the word embedding model in the natural language processing community [15], this paper proposes a data-driven approach to diagnose potential errors without extra knowledge or rules in ATS architectures through our architecture embedding model and vector computation. The method simplifies the inference process into vector computation, overcoming the shortcomings of traditional diagnosis methods that rely on expert knowledge and rules. The method consists of two key components: architecture embedding model and anomaly detection.

3 Architecture Embedding Model

An ATS architecture consists of requisite structural knowledge and constraint rules that provide a guide to establish a relationship with infrastructure, vehicles, traffic participants and system-level functional objects in a transport system. This knowledge can be represented through classical knowledge graph methods, stored in the form of triples as (head entity, relationship, tail entity). Here a single example (roadside devices, provisioning, road network health status detection) means that roadside devices are responsible for provisioning road network health status. While triples are powerful in representing structured data, the symbolic characteristic of such triples makes knowledge graph difficult to handle, especially on a large scale. Due to the above reason and ATS architecture embedding model is proposed, inspired by word embedding models from the natural language processing (NLP) community. The key idea is to vectorize structural details into the continuous vector space, simplifying the difficulty and workload while maintaining the inherent structure of the knowledge graph. The vector representation embedded contains semantic information that can be used in rich downstream applications of NLP such as link prediction and triple classification. The embedding model utilizes rich semantic information about entities and relations, which can significantly improve knowledge acquisition and reasoning ability. The vector representation makes it possible to check whether the triple is correct by vector computation. The criterion of correct triples is defined by the truth value, which is calculated as the Euclidean distance. Architecture anomalies are then identified based on the truth value of the triples. The rationality of the architecture is measured by computing the mean value of all triples in the same vector space. This section consists of two parts: knowledge representation and model training.

3.1 Knowledge Representation in ATS Architecture

The embedding model generates the vector representation of the architectural content based on the co-occurrence distribution of the architectural content in the training dataset. Inspired by the word2vec model proposed by Google in 2013 [13], the architecture embedding model in this paper also uses negative sampling to increase the training speed of the model and improve the quality of the vector representation. Negative sampling training means that the model will be trained on both positive and negative data features during the training process. The architecture embedding model is a three-layer neural network, as shown in Fig. 1, with input, output and hidden layers. It is trained on both positive samples (true connections) and negative samples (incorrect functional and physical object connections). The weights of the hidden layer are a vector representation of a word. To achieve the best effect of the embedding model, this paper defines a loss function to which increases the differentiation between the positive and negative samples by minimization of loss. The trained neural network shows that words that appear in similar contexts will have similar vector representations. In the architecture embedding model, this means that entities co-occurring under the same semantic conditions have similar vector representations.

We obtain an ATS architecture $K = \{(\boldsymbol{e_i}, \boldsymbol{r_k}, \boldsymbol{e_j})\}$ containing triples, each containing two entities $\boldsymbol{e_i}, \boldsymbol{e_j} \in E$ and the relation $\boldsymbol{r_k} \in R$ between them, where E is the set of entities and R is the set of relations, respectively. Modeling triples we use the TransE model [16], based on its simplicity and effectiveness in achieving state-of-the-art predictive performance. Given a triple instance ($\boldsymbol{e_i}$, $\boldsymbol{r_k}$, $\boldsymbol{e_j}$), the relation $\boldsymbol{r_k}$ in the modeled triple instance is used as a vector transformation from $\boldsymbol{e_i}$ to $\boldsymbol{e_j}$, i.e. when the triple holds, we want to have the effect that $\boldsymbol{e_i} + \boldsymbol{r_k} \approx \boldsymbol{e_j}$. We then score each triple based on $\|\boldsymbol{e_i} + \boldsymbol{r_k}$ - $\boldsymbol{e_j}\|_1$, defining the true value of the triple as:

$$P(\boldsymbol{e_i}, \boldsymbol{r_k}, \boldsymbol{e_j}) = 1 - \frac{1}{\sqrt{3}d} \|\boldsymbol{e_i} + \boldsymbol{r_k} - \boldsymbol{e_j}\|_1 \tag{1}$$

Where d is the dimension of the embedding space. It is clear to see that $P\left(e_i, r_k, e_j\right) \in [0, 1]$ because of $\|e_i\|_2 \le 1$, $\|e_j\|_2 \le 1$, $\|r_k\|_2 \le 1$.

Note that $0 \leqslant \|\boldsymbol{e_i} + \boldsymbol{r_k} - \boldsymbol{e_j}\|_1 \leqslant \|\boldsymbol{e_i}\|_2 + \|\boldsymbol{r_k}\|_2 + \|\boldsymbol{e_j}\|_2 \leqslant 3\sqrt{d}$, where the last inequality holds because $\|\boldsymbol{x}\|_1 = \Sigma_i |x_i| \le \sqrt{d\,\Sigma_i x_i^2} = \sqrt{d}\|\boldsymbol{x}\|_2$, according to the Cauchy-Schwarz inequality.

3.2 Training Model with Negative Sampling

The architectural embedding model is trained with positive and negative samples. We first generate a training set containing all positive samples, and then we make the model work best by defining a minimizing global loss function.

$$\begin{gathered} \min_{\{e\},\{r\}} \sum\nolimits_{f^+ \in \mathcal{F}} \sum\nolimits_{f^- \in \mathcal{N}_{f^+}} \left[\gamma - P\left(f^+\right) + P\left(f^-\right)\right]_+, \\ s.t. \|e\|_2 \le 1, \forall e \in E;\ \|r\|_2 \le 1, \forall r \in R. \end{gathered} \tag{2}$$

Here $f^+ \in \mathcal{F}$ is a positive sample, $f^- \in \mathcal{N}_{f^+}$ is a negative sample constructed from a positive sample, and γ is the boundary condition for determining positive and negative samples. For a triple ($\boldsymbol{e_i}$, $\boldsymbol{r_k}$, $\boldsymbol{e_j}$), we achieve minimization by replacing $\boldsymbol{e_i}$ or $\boldsymbol{e_j}$ randomly with an arbitrary entity $\boldsymbol{e}$ in the entity set, which ensures the triple after the replacement does not exist in the original training set. We believe that the vast majority of triple instances generated in this way are negative samples, using small-batch stochastic gradient descent. The effect of model training is shown in Fig. 2.

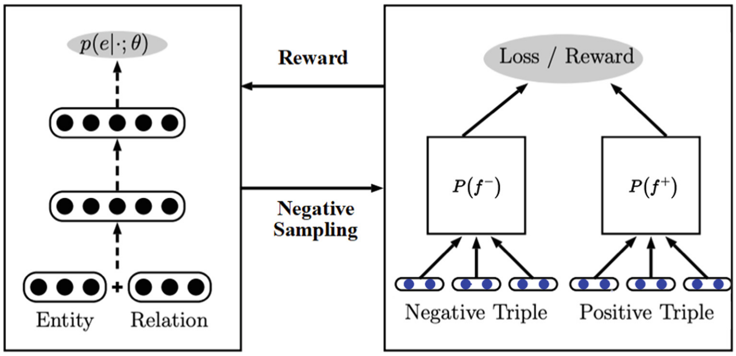

Fig. 1. General structure of the ATS embedding model

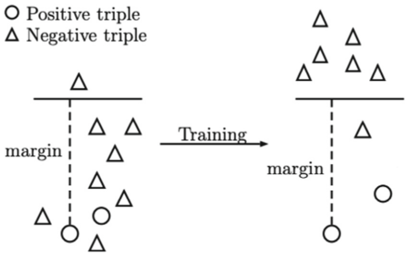

Fig. 2. The effect of training model by negative sampling

4 Experiments

4.1 Preparing Dataset

The classification results are obtained by triple classification to find the incorrect part of the ATS architecture. This task is to verify if any unobserved triples ($\boldsymbol{e_i}$, $\boldsymbol{r_k}$, $\boldsymbol{e_j}$) are correct. The dataset comes from the National ITS Reference Architecture (ARC-IT 9.0)[1] developed by the U.S. Department of Transportation. The part we use is the exchange process of information flow with a physical view of the ATS architecture, including the source and destination physical objects and the information flow. The physical source object acts as the head entity $\boldsymbol{e_i}$, the destination physical object acts as the tail entity $\boldsymbol{e_j}$, and we take the information flow as the relation $\boldsymbol{r_k}$. The dataset is respectively divided into a training dataset, a validation dataset, and a test dataset for model training, parameter tuning, and evaluation. The training dataset contains 1896 positive triples, 154 entities and 905 relations. We constructed 14 negative samples for every positive triple (Table 1).

[1] The Architecture Reference for Cooperative and Intelligent Transportation (ARC-IT) provides a common framework for planning, defining, and integrating intelligent transportation systems. ARC-IT 9.0 includes all views of the National ITS Reference Architecture - Enterprise, Functional, Physical and Communications views. The information is available at https://www.arc-it.net/index.html.

Table 1. A sample for training the architecture embedding model

Sources (head entity)	Flow (relation)	Destination (tail entity)
Alerting and advisory system	Alerts and advisories	Emergency management center
Basic vehicle	Driver input information	Vehicle OBE
Basic vehicle	Host vehicle status	Vehicle OBE
Connected vehicle roadside equipment	Intersection status	Commercial vehicle OBE
Connected vehicle roadside equipment	Signal priority status	Commercial vehicle OBE
Connected vehicle roadside equipment	Data provision	Data distribution system
Connected vehicle roadside equipment	Data query	Data distribution system
…	…	…

4.2 Evaluation Results

We set up an evaluation scheme similar to the TransE model [17]. We first generate test data for evaluation. For each positive triple in the test or validation set, we construct 10 negative triples by randomly replacing entities, five in the head position and the other five in the tail position. To make the evaluation process as accurate as possible, we use only the entities that occurred in that position to replace the corresponding position, and further ensure that the replaced triples do not exist in the training, validation or test datasets. We only use every triple's truth value as the classifying criterion. Triples with large truth values are often predicted to be correct. F_1 is used here to measure the accuracy of the triple classification task. As shown in Table 2, let TP be the triples that our method correctly predicts to hold, let FP be the triples that our method incorrectly predicts to hold, and FN be the triples that our method incorrectly predicts not to hold. The F_1-score formula is established as follows:

Table 2. Results of triple classification

Predicted	True triple	
	Positive	Negative
Positive	TP	FP
Negative	FN	TN

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

$$F_1 = 2 \times \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

We set up multiple training sets with different proportions of triples, using 30%, 50% and 75% of the triples as training sets to calculate the accuracy of the triple classification task (Table 3).

Table 3. The F_1-score accuracy comparison results of triple classification task (in percentage)

Training dataset	Precision(%)	Recall(%)	F_1(%)
30%	74.8	71.3	73.0
50%	76.2	73.5	74.8
75%	80.4	78.2	79.3

The experimental results show that the model gains relatively good results with different sizes of training sets, and the classification accuracy can reach 79.3% as the training set increases, which is positive for separating the incorrect triples from the ATS architecture.

5 Conclusions and Future Work

In this paper, we propose a new method to achieve the purpose of the ATS architecture abnormality detection. We generate the vector representation of the ATS architecture structural features through an embedding model, and classify features of the incorrect type from the ATS architecture by vector computation and triple classification. The experimental results show that the method can diagnose most of the structural errors and achieve good detection accuracy. The method can diagnose ATS architecture without prior knowledge or rules. It has potential for further applications such as tracing and solving problems of the ATS architecture.

For future work, we would like to consider expanding the diagnosing scope of ATS architecture and adding more features of the ATS architecture as embedding content, such as the logic of collaboration between functional objects, the hierarchical relationship between transport services and functional objects, etc. The relationship of "1-to-many" and "many-to-many" between them, grows the complexity of the embedding model and the difficulty of model training. We will also improve the diagnosis framework by integrating rules with the embedding model, and use the improved embedding model to obtain the vector representation of rules, which may enhance the accuracy or efficiency of detection.

Acknowledgments. This study was supported by the National Key R&D Program of China (No. 2020YFB1600400).

References

1. Zhang, J., Wang, F., Wang, K., Lin, W., Xu, X., Chen, C.: Data-driven intelligent transportation systems: a survey. IEEE Trans. Intell. Transp. Syst. **12**(4), 1624–1639 (2011)
2. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet Things J. **4**(5), 1125–1142 (2017)
3. Duan, X., Jiang, H., Tian, D., Zou, T., Zhou, J., Cao, Y.: V2I based environment perception for autonomous vehicles at intersections. China Commun. **18**(7), 1–12 (2021)
4. Fünfrocken, M., Otte, A., Vogt, J., Wolniak, N., Wieker, H.: Assessment of ITS architectures. IET Intel. Transport Syst. **12**(9), 1096–1102 (2018)
5. Vogt, T., et al.: A comprehensive risk management approach to information security in intelligent transport systems. SAE Int. J. Transp. Cyber. Privacy **4**(1), 39–58 (2021)
6. Gao, Z., Cecati, C., Ding, S.X.: A survey of fault diagnosis and fault-tolerant techniques—part I: Fault diagnosis with model-based and signal-based approaches. IEEE Trans. Industr. Electron. **62**(6), 3757–3767 (2015)
7. Li, W., Li, H., Gu, S., Chen, T.: Process fault diagnosis with model- and knowledge-based approaches: advances and opportunities. Control. Eng. Pract. **105**, 104637 (2020)
8. Xiaojun, C., Shengbin, J., Yang, X.: A review: Knowledge reasoning over knowledge graph. Expert Syst. Appl. **141**, 112948 (2020)
9. Chen, G., Liu, M., Kong, Z.: Temporal-Logic-Based semantic fault diagnosis with time-series data from industrial internet of things. IEEE Trans. Industr. Electron. **68**(5), 4393–4403 (2021)
10. Guang, C., Tonghai, J., Meng, W., Xinyu, T., Wenfei, J.: Modeling and reasoning of IoT architecture in semantic ontology dimension. Comput. Commun. **153**, 580–594 (2020)
11. Omran, P.G., Wang, K., Wang, Z.: An embedding-based approach to rule learning in knowledge graphs. IEEE Trans. Knowl. Data Eng. **33**(4), 1348–1359 (2021)
12. Shihong, E.H., Yiheng, F., Henry, X.L.: A data-driven method for falsified vehicle trajectory identification by anomaly detection. Transp. Res. Part C Emerg. Technol. **128**, 103196 (2021)
13. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS 2013), vol. 2, pp. 2787–2795. Curran Associates Inc., Red Hook (2013)
14. S Suresh, S., Neville, J.: A hybrid model for learning embeddings and logical rules simultaneously from knowledge graphs. In: 2020 IEEE International Conference on Data Mining (ICDM), 17–20 November 2020, pp. 1280–1285 (2020)
15. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: a survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)
16. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp. 192–202 (2016)
17. Lv, X., Hou, L., Li, J., Liu, Z.: Differentiating concepts and instances for knowledge graph embedding. arXiv preprint arXiv:1811.04588 (2018)