相关工作

2.1 车辆CAN 入侵检测的方法

2.1.1 传统方法

The CAN frame is generally unable to support Message Authentication Code (MAC) [46] and other methods of securing communication. Some researchers have attempted to either create new protocols or spread MAC across multiple transmissions. in [28], Tashiro et al. for tampering detection can be conducted for both individual frames and entire sections, they propose a protocol that provides protection against replay, masquerading and injection attacks by sending a partial MAC in each frame. Nowdehi et al. [25] examine many of these altered protocols in light of five criteria for potential CAN message authentication solutions from an industry perspective. They found that no solutions met all the criteria. VatiCAN takes the approach of utilizing maintenance support, “sufficient implementation details” and no excessive overhead, while WooAuth alters the extended CAN protocol to allow more space for authentication codes. Finally, the authors suggest that the CAN bus might “be fundamentally unsuited for secure communication”[25]

2.1.2 机器学习方法

Intrusion detection system (IDS) can combine machine learning to train itself to identify abnormal behavior, which can be used as an alternative or supplement to Mac. IDS can prevent spoofing, injection, bus shutdown and denial of service attacks. In [18], Choi et al. Introduced a method called voltage IDS, which uses the inconsistency of ECU signals, first conducts training and testing, identifies the signal characteristics at this stage, and then uses the training data to verify whether the ECU has been damaged. Voltage IDS can detect camouflage attacks by using multi-class classifiers, one of which corresponds to an ECU. It predicts the most likely sender and compares this information with the actual can ID of the message. If they are different, a camouflage attack is detected. In [], song et al, by converting the ID part of the CAN frame into binary. Used the changed ResNet model to extract the features of the binary text, and learned the features of the intrusion message and the normal message for intrusion detection. Their experimental results show that compared with the traditional machine learning algorithm, this algorithm has lower false positive rate and false positive rate. In [], Kang et al. Proposed using deep learning methods for intrusion detection, because they are generated directly from the bit stream on the network, the execution efficiency of these functions is high and the complexity is low. This technology monitors the exchange packets in the vehicle network while training the characteristics offline, and provides a real-time response to the attack with a significantly high detection rate in their experiment.

2.2 遗传算法

2.2.1 EA 架构搜索

neural architecture search (NAS) aims to automatically design network architecture, which is essentially an optimization problem of ﬁnding an architecture with the best performance in speciﬁc search space with constrained resources [1], [2]. Sun et al. [19] used EA with variable coding length to automatically evolve the architecture of CNN. William et al. [41] introduced an evolutionary NAS coding strategy based on directed acyclic graphs (DAG), which has better performance than the randomly generated CNN architecture. Real et al. propose Amoebanet [42], which uses improved tournament selection to evolve network groups, and achieves better results on Imagenet than the handmade model. Wang et al. [43] designed an effective evolutionary algorithm to optimize the generator within the framework of GANs. This method can effectively improve the generation performance and training stability of GAN model. Sun et al. [44] proposed a variable length coding, which can represent different numbers of building blocks and layers to search for the best depth convolutional neural network. Nemo [45] uses evolutionary multi-objective method to design CNN architecture, which uses NSGA-II to maximize classification performance and minimize network reasoning time. Elsken et al. [46] described NAS as a bi-objective optimization problem, in which two objectives are to maximize performance and minimize computing resources. Lu et al. [47] proposed nsganet, which can automatically design the network, maximize the model performance and minimize floating point operations (flops).

2.2.2 代理模型

A major disadvantage of EvoNAS is that in the process of evolutionary optimization, each new candidate neural network needs to be trained on the training data set and then evaluated on the validation data set to avoid over-fitting. Therefore, if the network is large and the training dataset is large, the architecture evaluation in EvoNAS may take several hours. Because EAS is a kind of group based search methods, they usually need a lot of fitness evaluation, which makes EvoNAS computationally difficult to implement. For example, on CIFAR10 and CIFAR100 datasets, CNN-GA [50] consumes 35 GPU days and 40 GPU days respectively, genetic CNN method [14] consumes 17 GPU days, and large-scale evolutionary algorithm [16] consumes 2750 GPU days. Therefore, in the case of limited computing resources, the agent model can accelerate the fitness evaluation in EvoNAS.

Agents are divided into high-level agents and low-level agents. The high-level agent and low-level agent represent the architecture level and the parameter level in the architecture respectively. High level agent representation predicts the accuracy of different neural networks by parameterizing the neural network architecture. However, the low-level agent solves the complexity of using SGD optimization from scratch for each architecture after searching multiple architectures. The low-level agent is given a trained hypernetwork and neural network structure including all sub architectures. The weight of the neural network architecture inherits the weight from the hypernetwork. In the search process, the accuracy of using the weight inherited from the hypernetwork becomes the standard for selecting the architecture. However, the correlation between the accuracy of prediction architecture and the final accuracy of neural architecture through weight sharing is not close. The neural architecture reference MSuNAS we searched is not only sharing the weight of hypernetwork, but also fine-tuning through training again.

MetaQNN [1] uses the agent model to predict the final accuracy of candidate architectures (as time series prediction) from the first 25% learning curve of SGD training. PNAs [19] uses an alternative model to predict the accuracy of the structure, adding an additional branch to the unit structure, which is repeatedly stacked together. Both methods use the agent method to evaluate the performance of neural architecture. However, the correlation between the prediction accuracy of this method and the actual accuracy of the model is relatively low. Onceforall [5] also uses an agent model to predict the accuracy of architecture coding. However, the agent model is trained offline for the whole search space, so it needs a large number of samples to learn. ChamNet [10] trains many architectures through complete low-level optimization, and selects only 300 high-precision samples with different efficiency (trigger, delay, energy) to train alternative models offline. Our model only conducts online learning on samples close to Pareto frontier, which significantly improves the efficiency of architecture search. Our model evaluation method draws lessons from the idea of msunas.