applied sciences

MDPI

Article

ADLBiLSTM: A Semantic Generation Algorithm for Multi-Grammar Network Access Control Policies

Jing Zhang and Xiaoyan Liang *

School of Cyber Security and Computer, Hebei University, Baoding 071002, China; zhangjing@stumail.hbu.edu.cn
* Correspondence: liangxiaoyan@hbu.edu.cn

Abstract: Semantic generation of network access control policies can help network administrators accurately implement policies to achieve desired security objectives. Current semantic generation research mainly focuses on semantic generation of single grammar and lacks work on automatically generating semantics for different grammatical strategies. Generating semantics for different grammars is a tedious, inefficient, and non-scalable task. Inspired by sequence labeling in the field of natural language processing, this article models automatic semantic generation as a sequence labeling task. We propose a semantic generation algorithm named ADLBiLSTM. The algorithm uses a self-attention mechanism and double-layer BiLSTM to extract the features of security policies from different aspects, so that the algorithm can flexibly adapt to policies of different complexity without frequent modification. Experimental results showed that the algorithm has good performance and can achieve high accuracy in semantic generation of access control list (ACL) and firewall data and can accurately understand and generate the semantics of network access control policies.

Keywords: double-layer BiLSTM; self-attention mechanism; network access control policies; semantic generation

check for updates

Citation: Zhang, J.; Liang, X. ADLBiLSTM: A Semantic Generation Algorithm for Multi-Grammar Network Access Control Policies. *Appl. Sci.* **2024**, *14*, 4555. https://doi.org/10.3390/app14114555

Academic Editor: Luis Javier García Villalba

Received: 25 April 2024
Revised: 23 May 2024
Accepted: 24 May 2024
Published: 25 May 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Network access control policies are a commonly used method to implement organizational security policies [1]. They define a series of rules and actions to match and process network packets. These rules are set based on various factors such as source address, destination address, port number, protocol type, etc., to ensure that only packets that meet specific conditions can pass through the network. However, in practical applications, the configuration and management of network access control policies face many challenges. Traditional network access control configuration methods mainly rely on manual operations, but due to the low readability of policies [2–4], manual configuration is inefficient and prone to errors and when new security threats emerge, manual modification and updating of policies are required, resulting in poor scalability [5–8], which can lead to serious security issues in the network [9]. At the same time, the lack of unified standards and specifications increases the difficulty of configuring network access control policies with different grammars. Therefore, the automatic semantic generation of security policies is particularly important. Automatic semantic generation can effectively parse security policies with different syntaxes, realize the automation of network access control configuration, and thus improve configuration efficiency and reduce errors. However, there are currently few studies on the automation of security policy semantic generation. In the early stage, semantic generation mainly relied on manual work to transform security policies into machine-understandable languages. Later, compilers were used to semantically generate security policies, but it requires manual settings beforehand and only adapts to fixed security policies. Nowadays, the compilation technique proposed by researchers [10] can work without human intervention, but it is still only suitable for monolingual grammar and cannot cope with multilingual environments, which has become a bottleneck for the

automation of network access control configuration. Regarding the current limitation of security policy semantic generation techniques in accommodating multi-grammar environments, this paper aimed to implement an algorithm that can automatically generate security policy semantics for multiple grammars. Inspired by the advancements in natural language processing, this paper abstracted the task of automatic security policy semantic generation as a sequence labeling task in natural language processing and proposes a network security policy semantic generation algorithm named ADLBiLSTM, which is used to generate semantics for different types and grammatical structures of security policies. This model combines a self-attention mechanism and double-layer BiLSTM for automatic feature extraction, which is used to generate semantics for security policies with different grammars and assist in the management of network security policies. Our contributions in this work can be summarized as follows.

(1) We propose an algorithm for semantic generation of network security policies, which combines the self-attention mechanism with a double-layer BiLSTM to automatically extract features of network security policies from different levels and generate semantics accordingly.

(2) Through experiments, we verify that the model performs excellently in automatically generating semantics for network security policies of different types and grammars.

This paper is organized as follows. Section 2 reviews related works. We propose the semantic generation algorithm in Section 3. We provide the evaluating experiments in Section 4. Section 5 concludes the paper.

2. Related Work

Early security policies primarily relied on static, predefined rule sets [11], which were typically configured by security experts based on the specific needs and environments of the organization. In this case, semantic generation primarily involves converting rules into machine-readable formats for implementation in network devices or security systems, which relies heavily on manual operations. Increasingly complex security needs have gradually transformed simple rule sets into more advanced formal methods. Elfaki et al. [12] using first-order logic (FOL) to formalize firewall rules. Hamilton et al. [13] proposed an algorithm that converts firewall rules into conjunctive normal form (CNF) or disjunctive normal form (DNF) Boolean expressions. To represent and understand security policies more effectively, semantic models such as firewall decision diagram models [14] and ordered binary decision diagram (OBDD)-based models [15] have been introduced. These models not only enhance the expressive power of policies but also improve the flexibility and complexity of policy evaluation. However, manual optimization is still required during policy semantic generation to ensure that it accurately reflects the organization's security intentions. To address this issue, [10] designed a verified stateless firewall policy compiler, but this compiler can only perform semantic generation for specific security policies and requires reprocessing of data when dealing with security policies of other syntaxes. With the development of big data and artificial intelligence technologies, security policies are gradually relying on these technologies for optimization and automatic generation [16]. The accuracy of semantic generation is crucial when generating semantics for many different types of security policies. Deep learning has demonstrated its unique advantages in this scenario, as it can extract useful information and patterns from vast datasets for in-depth learning.

The advancements in the field of natural language processing (NLP) [17] have provided technical support for the semantic generation of security policies, with sequence labeling techniques being one of its key components. This technology possesses robust capabilities in information extraction, text understanding, and semantic analysis, enabling it to identify and label crucial concepts and entities from unstructured text, providing a solid foundation for automatic semantic generation. Long short-term memory neural networks (LSTMs), which are a form of recurrent neural networks (RNNs), are widely used in NLP [18–21]. RNNs face issues such as gradient vanishing and exploding when

dealing with long sequences, making it difficult to capture long-term dependencies. LSTMs alleviate these problems through their gating mechanism and memory cells, but standard LSTMs are unidirectional, limiting their ability to capture contextual information. Therefore, researchers have begun to explore more effective network structures. Huang et al. [22] first proposed the BiLSTM-CRF structure, which uses a bidirectional LSTM (BiLSTM) plus a conditional random field (CRF) to solve the problem of text tagging. In the article, they compared long short-term memory networks (LSTMs), bidirectional long short-term memory networks (BiLSTMs), and BiLSTM-CRF for natural language tagging. This network structure is still one of the mainstream methods today. Subsequently, Bahdanau et al. [23] first introduced the attention mechanism into NLP tasks, an innovation that enabled the model to better allocate attention when processing long sentences, improving processing effectiveness. However, the structure of these models is relatively simple, and the feature extraction for complex data is not comprehensive. Therefore, Lin et al. [24] proposed an innovative model that adopts a hierarchical feature extraction method to extract more information from both the character level and the word level, such as suffix information, and applies attention mechanisms at both levels to distinguish the importance of information. This addresses the limitations and one-sidedness that exist when extracting contextual features solely from characters or words in current settings, resulting in an improved ability for the model to extract features from data. However, for non-natural language data, the importance of character-level features is lower than that of word-level features, thus requiring accurate capture of features between words. Liu et al. [25] proposed a novel unified architecture for text classification, which includes BiLSTM, an attention mechanism, and convolutional layers. The model performs well in text classification tasks, but for sequence labeling tasks, ensuring the legality of labels is crucial. Wang et al. [26] proposed a character-level entity recognition model based on multi-features, which adopts a double-layer BiLSTM-CRF structure and uses BERT for character vector embedding to identify named entities in unstructured data of daily scheduling in the power system. This model performs well in feature extraction, and compared with traditional models, the F1 value has been improved. Wu et al. [27] proposed an attention-based CNN-LSTM-BiLSTM model for short-term power load forecasting in integrated energy systems. These models perform well in non-natural language fields, but the types of data they handle are relatively stable. When the types of data change significantly, the model needs to be able to extract features from the data more effectively.

3. The Semantic Generation Algorithm

Current semantic generation methods mainly rely on compilers, which are typically only suitable for a single grammar or single vendor and require manual configuration. Therefore, they are difficult to adapt to environments with multiple grammar configurations. However, natural language processing has made significant progress in dealing with unstructured data, as it can convert unstructured text into structured data, enabling further utilization of these data. This is consistent with the goals we need to achieve in our task. Therefore, we can consider utilizing natural language processing techniques to design improved semantic generation methods and better adapt them to environments with multiple grammar configurations.

In this section, we present our proposed model, named ADLBiLSTM. The proposed model is shown in Figure 1 and can be divided into three layers based on the bottom-up function, which are the self-attention layer, the DL-BiLSTM layer and the CRF layer. Firstly, the self-attention layer is used to capture the correlation between the input vectors a_i (a_i is obtained from input after word embedding) and weights the original inputs; the output x_i is a rule that combines the importance of the whole utterance, and x_i serves as the input to the next DL-BiLSTM layer. Then, the DL-BiLSTM layer obtains the contextual features from the input vector x_i and computes the score matrix of the labels input to the next layer. Finally, the score matrix is used as an input to the CRF layer for joint training with the

feature scores of the CRF layer to restrict the label range to obtain the final label output. The final output of the model is the semantics of the input security policy.

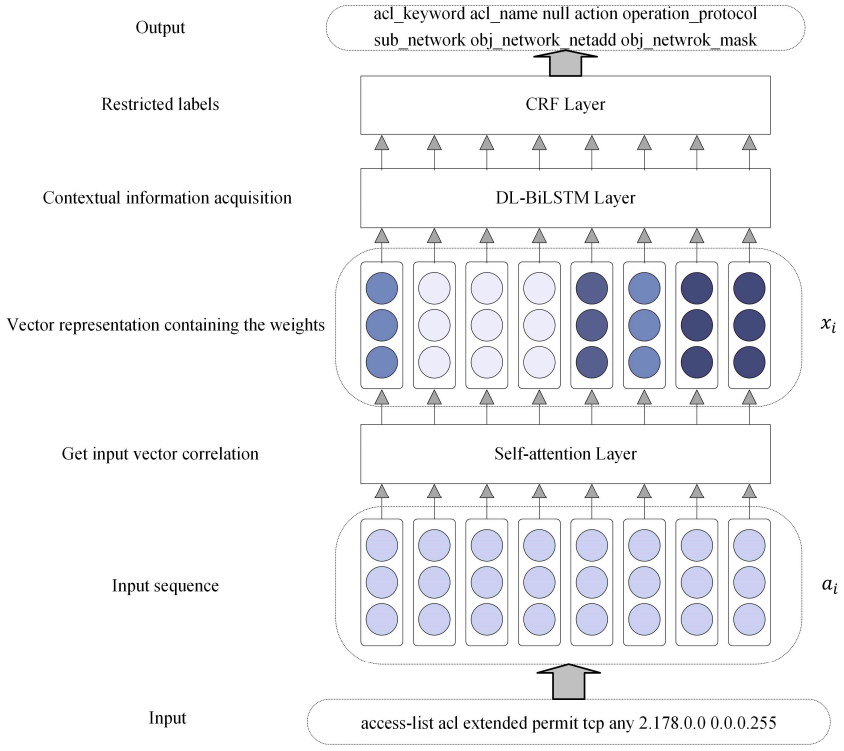


Figure 1. Network structure.

To illustrate more specifically how this model can be applied to the semantic generation of security policies, we can use the ACL statement "access-list acl extended permit tcp any 2.178.0.0 0.0.0.255" as an example. This statement defines an access control rule that permits TCP traffic from any source address ("any") to the IP address range 2.178.0.0/24 (defined by the IP address 2.178.0.0 and subnet mask 0.0.0.255).

When we input this ACL statement into the ADLBiLSTM model, the model first converts the text into vector representations through word embeddings. Then, the self-attention layer captures the correlations between these vectors (e.g., the association between "permit tcp any 2.178.0.0"), and weights the original input, where in this case, permit and tcp have higher importance, followed by "any 2.178.0.0 0.0.0.255". Next, the DL-BiLSTM layer extracts contextual features from the weighted inputs and calculates a score matrix for labels, which will compute the label scores for each word in the security policy. Finally, the CRF layer performs joint training based on the score matrix and its own feature scores, outputting the final label sequence. This layer considers not only the final semantic generation result but also whether the semantics are reasonable. For example, in this case, it learns that the statement is valid when "any" represents the source address (sub_network) and 2.178.0.0 represents the destination address (obj_network_netadd).

These three layers are described in detail below.

3.1. Self-Attention Layer

At the bottom, we apply the self-attention mechanism to fuse the relevance of words into the original input. The self-attention mechanism is at the bottom of the network and processes the raw inputs directly, weighting the raw inputs as the output of the next layer. A rule, after word embedding, can be represented as $X^* = [a_1, a_2, \ldots, a_n]$, where a_n denotes the nth word in the rule. The input vectors are dotted with the coefficient matrices

w_q, w_k, and w_v to obtain $Q = [q_1, q_2 \ldots q_n]$, $K = [k_1, k_2 \ldots k_n]$, and $V = [v_1, v_2 \ldots v_n]$, as in Equations (1)–(3).

$$q_n = w_q \cdot a_n \tag{1}$$

$$k_n = w_k \cdot a_n \tag{2}$$

$$v_n = w_v \cdot a_n \tag{3}$$

The q_n values of each input word are dotted with the k_n^T values of the other input words (including itself) to obtain the attention scores so that the correlation between the words can be obtained, and then the attention scores are normalized using the softmax to form the attention score matrix. Then, each v_n is multiplied by the attention score matrix to obtain its weight vector, and then all the vectors are weighted and summed to obtain the final output $X = [x_1, x_2 \ldots x_n]$. This weighted processing method makes the vital information more fully used, which improves the effect of semantic generation. Equation (4) describes the entire process, where d_k denotes the dimensionality of the word vectors, divided by $\sqrt{d_k}$ to prevent computational overflow caused by too large a value of QK^T.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4}$$

The self-attention mechanism considers the relationship between each element and other elements, processing sequential data in a disorderly manner to better understand the context. In rules such as those for routers and firewalls, the same address may have different labels due to factors such as location and specific connecting words. The self-attention mechanism can ignore non-critical parts of a rule and find the word with the highest correlation to the obfuscated word for judgment. In addition, security policies from the same vendor may adjust the original syntax due to changes in versions or operational requirements. Therefore, employing the self-attention mechanism to capture the relevance of each word in each rule can solve the problem of label inconsistency and syntactic heterogeneity in the semantic generation of security policies.

However, the self-attention mechanism does not directly include positional information, thus requiring additional mechanisms to capture sequential information.

3.2. DL-BiLSTM Layer

Double-layer BiLSTM (DL-BiLSTM) consists of two layers of BiLSTM, which we utilize for context feature extraction. The given input $X = [x_1, x_2 \ldots x_n]$ is a set of inputs that can embody long sequential information. Inputting it into the first BiLSTM procures a context vector for each time step t. The hidden state of the first BiLSTM layer is $h_{1,t}$, which performs forward and backward operations on the input X, denoted as the contextual feature vector for the tth tag, i.e., at the tth time point, the procured feature data contains both past and future information, as in Equation (5). The output c_t of the first BiLSTM layer is used as an input to the second BiLSTM layer.

$$h_{1,t} = [\overrightarrow{h_{1,t}} : \overleftarrow{h_{1,t}}] \tag{5}$$

The second BiLSTM is recursively executed to perform label probability prediction, and then the prediction vectors are input to the dense layer to procure the score matrix. The state of the hidden layer of the second BiLSTM layer at moment t is $h_{2,t}$, which is used as the output of the probabilistic prediction vector p_n, p_n composition of the predicted score matrix as an input to the next layer. The process is shown in Figure 2.

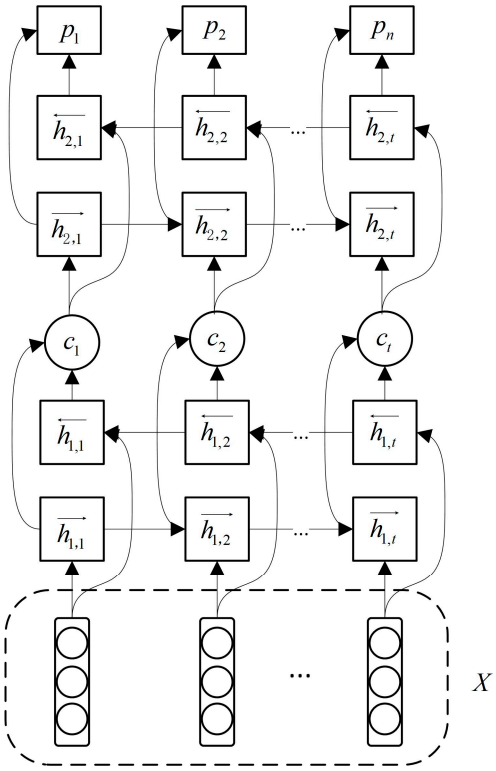


Figure 2. The inner workings of the DL-BiLSTM.

Our security policy, as long sequence data, requires the model to fully understand its context. Therefore, we chose the BiLSTM structure, which consists of two LSTMs that are responsible for capturing the before and after information of the sequence, respectively, and improves the model's expressive ability.

To further enhance the model's ability to capture temporal information, we adopted a double-layer BiLSTM structure; i.e., stacking another layer on top of the standard BiLSTM, which enables the upper BiLSTM to process the output of the lower layer, thus extracting and integrating temporal sequence information in a deeper way. This structure not only extracts information from past and future directions, but also enhances the feature extraction ability of the model for complex data by double stacking. In addition, the double-layer BiLSTM has good generalization ability to capture rich timing information, strong expressive power, and stable and reliable performance in processing new data, which is crucial for dealing with new data or scenarios in real tasks.

However, due to its depth and complexity, it may face issues such as gradient vanishing or exploding during training, which requires adjusting parameters to enable the model to converge better.

3.3. CRF Layer

After the output of the double-layer BiLSTM, we obtained the prediction score matrix, which, however, disregards the connections between labels. Therefore, we used CRF to impose certain constraints on the final prediction results to ensure the accuracy of label generation. CRF is a probabilistic graph-based model for modeling sequence labeling problems, which achieves specification of labels by learning the joint distribution between the input sequence and the output sequence, taking full account of the relationship between individual markers, and dealing with the dependencies between the markers. For a given input sequence X, its $Y = [y_1, y_2, \ldots, y_n]$. When inputting x, calculate its score to obtain the score matrix, as in Equation (6).

$$score(x, y) = \sum_i h_i[y_i] + P_{y_{i-1}, y_i} \tag{6}$$

where $h_i[y_i]$ represents the emission score, which is the score when predicting the result to be a particular label, and it comes from the output of DL-BiLSTM, and P_{y_{i-1},y_i} represents the transition score, which is the probability of the previous character's labeling result to the current character's labeling result. Considering the labeling process as a path, the true path (the correct prediction result) is the highest scoring of all the paths, and thus the formula is expressed as the sum of the emission score and the transition score over the paths. We want the correct scores to have a greater weight in the total score, and we typically apply softmax to the scores, so we defined the loss as in Equation (7):

$$loss = -\log\left(\frac{e^{score(x,y)}}{\sum_{y\prime \in Y} e^{score(x,y\prime)}}\right) \quad (7)$$

By training, we obtained the final prediction and in the process normalized the prediction labels.

As the CRF layer needs to calculate the scores for the entire sequence during training, this can lead to increased computational complexity.

4. Experiments

This section focuses on the experiments we designed and the experimental results. We designed two experiments on semantic generation. The experiments: (1) revealed the influence of different model structures on the generation of security policy semantics; and (2) verified the performance of the proposed model for generating security policy semantics with different grammars. The experimental results showed that the proposed model can generate highly accurate semantics for different security policies in different scenarios.

4.1. Dataset

There is no labeled dataset of security policies for network access control. Therefore, based on the role of each part of the security policy in the network, this research constructed a dataset of network access control policies [28]. A security policy primarily filters, allows, or discards data packets on interfaces by setting specific conditions. Therefore, we annotated the definition statements and conditions within the security policy to accurately express its semantics. In this experiment, we used two types of data, ACL data and firewall data.

ACL data contains seven datasets, among which Data_mix for training is composed of actual data used by three vendors: Cisco, Huawei, and Ruijie. The six datasets used in the experiment were simulated based on the actual usage frequency of network configurations for standard ACLs and extended ACLs from three different vendors, with each dataset containing 3000 policy rules. Firewall data included two datasets, among which Firewall_train for training is the actual data used by TOPSEC, and Firewall_test for experiments is the simulated data based on the actual configuration frequency.

Our dataset was manually annotated. The volume of various data is shown in Table 1, and the number of labels of each dataset is also shown in Table 1.

Table 1. Data used in the experiments.

	ACL						**Firewall**			
	Standard_ACL		**Extend_ACL**		**Data_Mix**		**Firewall_Test**		**Firewall_Train**	
	Num	**Labels**	**Num**	**Labels**	**Num**	**Labels**	**Num**	**Labels**	**Num**	**Labels**
Huawei	3000	6	3000	9						
Cisco	3000	4	3000	10	3437	13	2000	10	668	10
Ruijie	3000	4	3000	7						

In Table 1, the number of labels corresponding to each dataset represents the number of distinct label categories that appeared in that dataset. Taking Cisco's extended ACL dataset as an example, this dataset includes two types of extended ACLs: one that contains port information and one that does not. The total number of distinct label categories for these two types of security policies is 10.

The example datasets used in this experiment are shown in Table 2, and due to the large number of firewall definition statements, only two types are shown here.

Table 2. Examples of the data used in the experiments.

Type	Example	Label
cisco_standard_ACL	access-list 1 deny 10.64.0.0	keyword id action source_network
huawei_standard_ACL	rule 1 deny source 14.45.178.0 255.255.255.192	keyword id action keyword sub_network_netadd sub_network_mask
ruijie_standard_ACL	1 permit 10.92.96.0 0.0.31.255	id action obj_network_netadd obj_network_mask
cisco_extend_ACL	access-list aaa extended deny udp any 172.22.118.248 192.168.0.0	keyword id null action operation_protocol sub_network obj_network_netadd obj_network_mask
huawei_extend_ACL	rule 1 deny tcp destination 178.23.144.0 0.0.7.255 mrdkp 99483	keyword id action operation_protocol keyword dest_host_ip dest_host_ip_mask port_operaor port_name
ruijie_extend_ACL	1 permit udp host 10.176.0.0 host 192.168.0.0	id action operation_protocol keyword sub_network_netadd keyword obj_network_netadd
Firewall_test	defaddr add 1 10.92.96.0 255.255.255.255	define operation id sub_network_netadd sub_network_mask
	defaddrgrp addmbr first 540	define operation id id

4.2. Network Parameters

Here, we describe the parameter settings used in the experiment.

Word embedding: We utilized embedding to generate a 100-dimensional vector for each word. The embedding layer was trained together with the neural network, so pre-trained embeddings were not used. The weights of the embedding were randomly initialized.

Optimization algorithm: We used small-batch stochastic gradient descent with an initial learning rate set to 0.001 and a decay rate of 0.0001. Gradient clipping was set to 0.5 to avoid gradient explosion.

The network structure: The self-attention layer was fed with 100-dimensional word embeddings; the output length was the number of labels received by the DL-BiLSTM layer, the hidden layer was also set to the number of tags for dynamic adjustment, and the output was of the same dimension as the hidden layer, which was then fed to the subsequent layers.

4.3. Model Structure and Evaluation Metrics

In this paper, four experimental groups were set up to demonstrate the superiority of the proposed model. The first group was the baseline group; that is, the BiLSTM-CRF model. The second group added another layer of BiLSTM to the baseline to confirm the performance improvement of the double-layer BiLSTM in extracting contextual features of security policies. The third group added a self-attention mechanism to the baseline to verify the performance improvement of the self-attention mechanism in generating semantic representations of security policies. The fourth group was the model proposed in this paper, ADLBiLSTM, which adds a self-attention mechanism and uses double-layer BiLSTM to extract contextual features of security policies. The specific model structure is shown in Table 3.

Table 3. Experimental group design.

Network Structure	Name
BiLSTM + CRF	Baseline
Double-layer BiLSTM + CRF	Baseline + DL-BiLSTM
Self-attention + BiLSTM + CRF	Baseline + Self-attention
Self-attention+ double-layer BiLSTM + CRF	Ours

The models in the four experimental groups were all trained using Data_mix in Table 1, and then the semantic representations of security policies from three vendors, Cisco, Huawei, and Ruijie, were generated respectively. To evaluate the performance of the models, this paper adopted common evaluation metrics, namely precision (P), recall(R), and $F1$ score($F1$). We defined TP as the number of correctly generated semantics, NP as the number of incorrectly generated semantics, and N as the total number of semantics that should be generated. Therefore, these three metrics can be defined as follows in Equations (8)–(10)

$$P = \frac{TP}{TP + NP} \tag{8}$$

$$R = \frac{TP}{N} \tag{9}$$

$$F1 = \frac{2PR}{P + R} \tag{10}$$

In the experiments, the dataset used was simulated based on usage frequency, resulting in a small amount of data for some types, which may lead to inaccurate evaluation of the model's performance. Therefore, we conducted multiple experiments and calculated the metrics to improve their stability and reliability.

4.4. Semantic Generation for ACLs

ACL policies include extended ACL and standard ACL, so this experiment separately generated the semantics of these data to verify the performance of the model.

Table 4 presents the semantic generation results of the four models when processing standard ACL security policies with different syntaxes.

The syntactical structure of standard ACL is relatively simple, and the length of the policies is also shorter. Therefore, the model proposed in this paper exhibits higher accuracy in extracting features of standard ACL. Especially for the standard ACLs of Huawei and Ruijie, the model achieved higher precision, which fully demonstrated the good performance of the model in processing security policies with clear and concise structures. This result not only verified the effectiveness of the model's feature extraction ability but also further proved the applicability of the self-attention mechanism and the double-layer BiLSTM network structure in the semantic generation task. By comparing the performance of different models on standard ACL, we found that the model combining

the self-attention mechanism and double-layer BiLSTM has significant advantages in extracting and understanding concise syntactical structures. This advantage enabled the model to capture key information more accurately and generate semantic representations that conformed to the original meaning.

Table 4. The model's semantic generation results for the standard ACL.

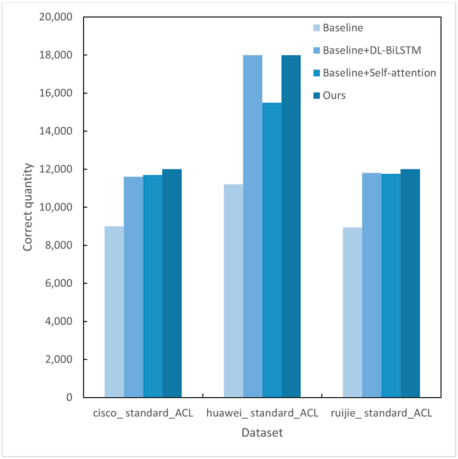
Dataset	Model	*P* (%)	*R* (%)	*F*1 (%)
cisco_standard_ACL	Baseline	74.96	76.15	75.53
	Baseline + DL-BiLSTM	99.99	99.99	99.99
	Baseline + Self-attention	99.9	99.9	99.9
	Ours	100	100	100
huawei_standard_ACL	Baseline	62.37	75.47	68.24
	Baseline + DL-BiLSTM	100	100	100
	Baseline + Self-attention	86.59	86.59	86.59
	Ours	100	100	100
ruijie_standard_ACL	Baseline	74.44	75.58	75.0
	Baseline + DL-BiLSTM	99.99	99.99	99.99
	Baseline + Self-attention	99.42	99.42	99.42
	Ours	100	100	100

Table 5 shows the detailed results of semantic generation for security policies of extended ACL with different syntaxes using the four models. Compared to standard ACL, the syntactical structure of extended ACL is more complex and the length is longer. From the table, we can observe that for extended ACL, adding BiLSTM and the self-attention mechanism to the baseline both improved the performance of semantic generation for security policies. However, the model proposed in this paper significantly enhanced the *P*, *R*, and *F*1 scores. Taking Cisco's extended ACL as an example, *P* increased from 34.92% to 99.74%, *R* increased from 45.99% to 99.74%, and *F*1 increased from 39.48% to 99.74%. Our model achieved good results when generating semantics for Huawei's extended ACL and Ruijie's ACL, and the generated results achieved high precision. The self-attention mechanism can focus on the most important parts of the original data, which is helpful for processing structured and logical texts such as ACL. However, security policies often involve not only individual rules or conditions but also the relationships between these rules or conditions and their positions and roles in the entire security policy. Therefore, adding an additional layer of BiLSTM further enhanced the ability to extract contextual features of security policies. By combining the self-attention mechanism and double-layer BiLSTM, the model not only focused on the most important information but also understood the position and role of this information in the entire security policy, further improving the performance of the model.

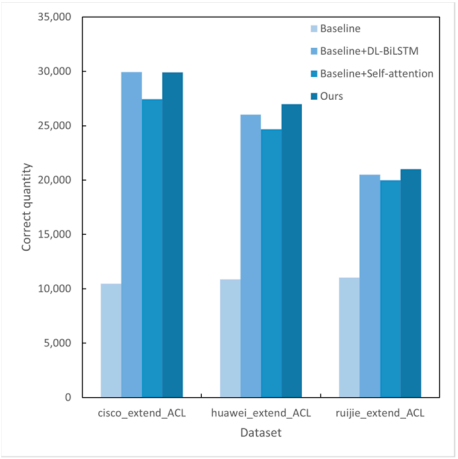
Figure 3 shows the number of correct semantic generations made by these four models for both standard ACLs and extended ACLs. As can be seen from the figure, compared to the baseline, the other three models significantly improved the number of correct semantic generations. For standard ACLs with simple grammatical structures and shorter sentences, the difference in the number of correct semantic generations among the three models was not obvious. However, for more complex extended ACLs, the gap between these three models was more pronounced. Overall, our model outperformed other models in terms of semantic generation performance.

Table 5. The model's semantic generation results for the extended ACL.

Dataset	Model	*P* (%)	*R* (%)	*F*1 (%)
cisco_extend_ACL	Baseline	34.92	45.99	39.48
	Baseline + DL-BiLSTM	99.8	99.8	99.8
	Baseline + Self-attention	91.51	91.31	91.31
	Ours	99.74	99.74	99.74
huawei_extend_ACL	Baseline	40.31	45.65	42.53
	Baseline + DL-BiLSTM	97.68	97.68	97.68
	Baseline + Self-attention	91.41	89.75	86.26
	Ours	100	100	100
ruijie_extend_ACL	Baseline	52.58	58.96	55.56
	Baseline + DL-BiLSTM	98.99	98.89	98.93
	Baseline + Self-attention	95.3	95.24	95.14
	Ours	100	100	100



(**a**) Generating the standard ACL semantics.

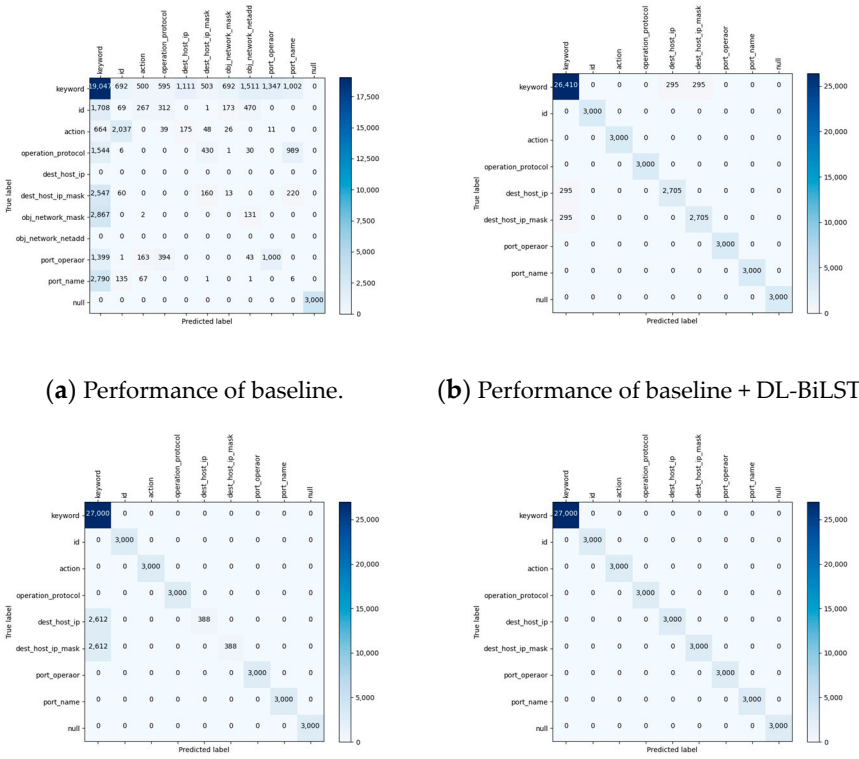


(**b**) Generating the extended ACL semantics.

Figure 3. The number of correct security policy semantics generated by four models.

To further validate the role of the self-attention mechanism and double-layer BiLSTM in semantic generation, we generated a confusion matrix using the experimental results of Huawei's extended ACL as an example to verify the impact of these two layers of the network. As shown in Figure 4, adding BiLSTM and the self-attention mechanism to the baseline improved the model's ability to extract features from security policies.

By comparing Figure 4a–c, it can be seen that "dest_host_ip" and "dest_host_ip_mask" were difficult to generate as labels. The reason is that the data for these two labels were in the form of IP addresses and very similar. Therefore, the baseline often generated these two labels as "obj_network_mask" and "obj_network_netadd". These two labels also corresponded to data in the form of IP addresses, but they represented different meanings in the security policy. The correct labels represented the address and mask of the destination host, while the incorrectly generated labels represented the address and mask of the source host. When we added a self-attention layer and an additional BiLSTM layer to the baseline, although the accuracy of other semantic generations improved, there were still errors in generating these two labels. However, our model, which combined the double-layer BiLSTM and the self-attention mechanism, achieved a 100% accuracy rate when generating these two labels, as shown in Figure 4d.



(**a**) Performance of baseline. (**b**) Performance of baseline + DL-BiLSTM.

(**c**) Performance of baseline + self-attention. (**d**) Performance of our model.

Figure 4. Comparison of confusion matrices among four models.

4.5. Semantic Generation for Firewalls

Compared to ACL data, firewall data is more cumbersome and contains many definition statements, making it more difficult to extract features from firewall data. Our model had advantages in this scenario. Table 6 demonstrates the performance of four models on the task of firewall security policy semantic generation. Through comparison, we found that the method proposed in this paper achieved significant improvements in precision (*P*), recall (*R*), and *F*1 score compared to the baseline model. *P* increased from 64.74% to 96.09%, *R* increased from 68.4% to 96.18%, and the *F*1 score increased from 66.51% to 96.13%. This was due to our model structure, and as we can see from the data in the table, the self-attention mechanism preprocessed complex data by extracting the importance of the original data, providing convenience for subsequent networks to extract contextual features. This is crucial in complex redundancy scenarios such as firewall data, as it helps the model capture key semantic information more accurately, and double-layer BiLSTM can extract better contextual features than single-layer BiLSTM, which together improve the accuracy of semantic generation.

Table 6. The model's semantic generation results for firewall.

Dataset	Model	*P* (%)	*R* (%)	*F*1 (%)
Firewall_test	Baseline	64.74	68.4	66.51
	Baseline + DL-BiLSTM	89.76	91.53	90.63
	Baseline + Self-attention	94.54	94.54	94.54
	Ours	96.09	96.18	96.13

The results of the above two experiments indicated that due to the advantages of the self-attention mechanism and the double-layer BiLSTM network structure, the model

proposed in this paper performed well in automatic semantic generation. Whether it was faced with security policies with simple grammatical structure and short length, or with complex grammatical structure and cumbersome structure, the model effectively performed automated semantic generation.

The self-attention mechanism used in the model captured long-distance dependencies in the input sequence and focus on the most relevant parts to the current task, thereby improving the model's semantic understanding ability. The double-layer BiLSTM structure fully utilized the before and after information of the sequence, encoded the input sequence bidirectionally, and further enhanced the model's context awareness.

Therefore, this structure combining the self-attention mechanism and the double-layer BiLSTM enabled our model to perform well in automatic semantic generation tasks, cope with security policies of different complexities, and generate its semantics.

5. Conclusions

In this article, we propose a semantic generation algorithm for security policies named ADLBiLSTM. The algorithm model consists of three layers, namely the self-attention layer, the DL-BiLSTM layer, and the CRF layer, which are used to extract features from security policies in different aspects. This algorithm can generate semantics for different types of security policies with varying grammatical structures. Experimental results demonstrated that our algorithm can achieve a precision rate of 100% for semantic generation of ACLs and a precision rate of 96.09% for semantic generation of firewalls.

The algorithm proposed in this paper can understand different types and grammatical structures of security policies, generating their corresponding semantic representations, and avoiding errors caused by human factors, thus improving the enforceability of the policies. In addition, this algorithm can be applied to automatic configuration of network security policies, enhancing accuracy while reducing management costs, to address the problems in configuring network security policies caused by new security threats.

Author Contributions: Conceptualization, J.Z.; methodology, J.Z. and X.L.; software, J.Z.; Validation, J.Z.; data curation, J.Z.; writing: J.Z. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Natural Science Foundation of Hebei Province of China (F2018201197).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request. The test data is publicly available and can be found at the following link: https://gitee.com/ainer4869/data/tree/master/ (accessed on 25 April 2024).

Conflicts of Interest: The authors declare that this study was conducted without any commercial or financial relationship that could be considered a potential conflict of interest.

References

1. Kovacevic, I.; Stengl, B.; Gros, S. Systematic review of automatic translation of high-level security policy into firewall rules. In Proceedings of the 2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 23–27 May 2022. [CrossRef]
2. Krumay, B.; Klar, J. Readability of Privacy Policies. In *Data and Applications Security and Privacy XXXIV, Proceedings of the 34th Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy (DBSec 2020), Regensburg, Germany, 25–26 June 2020*; Lecture Notes in Computer Science; Singhal, A., Vaidya, J., Eds.; Springer: Cham, Switzerland, 2020; Volume 12122. [CrossRef]
3. Angraini; Alias, R.A.; Okfalisa. Information Security Policy Compliance: Systematic Literature Review. *Procedia Comput. Sci.* **2019**, *161*, 1216–1224. [CrossRef]
4. Ermakova, T.; Krasnova, H.; Fabian, B. Exploring the Impact of Readability of Privacy Policies on Users' Trust. In Proceedings of the European Conference on Information Systems, Istanbul, Turkey, 12–15 June 2016; Available online: https://api.semanticscholar.org/CorpusID:6072616 (accessed on 15 September 2016).

5. Voronkov, A.; Martucci, L.A.; Lindskog, S. Measuring the usability of firewall rule sets. *IEEE Access* **2020**, *8*, 27106–27121. [CrossRef]
6. Arthur, J.K.; Kwadwo, E.; Doh, R.F.; Mantey, E.A. Firewall rule anomaly detection and resolution using particle swarm optimization algorithm. *Int. J. Comput. Appl.* **2019**, *975*, 8887.
7. Liang, X.; Xia, C.; Jiao, J.; Hu, J.; Li, X. Modeling and global conflict analysis of firewall policy. *China Commun.* **2014**, *11*, 124–135. [CrossRef]
8. Al-Shaer, E.; Hamed, H.; Boutaba, R.; Hasan, M. Conflict classification and analysis of distributed firewall policies. *IEEE J. Sel. Areas Commun.* **2005**, *23*, 2069–2084. [CrossRef]
9. Mukhtar, A.S.S.; Mukhtar, G.F.S. Deep learning powered firewall anomaly management environment using convolution and recurrent neural network. *Int. J. Res. Biosci. Agric. Technol.* **2023**, *II*, 64–66.
10. Chavanon, C.; Besson, F.; Ninet, T. PfComp: A Verified Compiler for Packet Filtering Leveraging Binary Decision Diagrams. In Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, London, UK, 15–16 January 2024.
11. Choudhary, A.R. Policy-based network management. *Bell Labs Tech. J.* **2004**, *9*, 19–29. [CrossRef]
12. Elfaki, A.O.; Aljaedi, A. Deep analysis and detection of firewall anomalies using knowledge graph. In Proceedings of the 12th International Conference on Pattern Recognition Applications and Methods, Lisbon, Portugal, 22–24 February 2023.
13. Hamilton, A.; Roughan, M.; Nguyen, G.T. Boolean expressions in firewall analysis. *arXiv* **2022**, arXiv:2205.04210.
14. Liu, A.X. Formal verification of firewall policies. In Proceedings of the 2008 IEEE International Conference on Communications, Beijing, China, 19–23 May 2008; pp. 1494–1498.
15. Hamed, H.; Al-Shaer, E.; Marrero, W. Modeling and verification of ipsec and vpn security policies. In Proceedings of the 13th IEEE International Conference on Network Protocols (ICNP'05), Boston, MA, USA, 6–9 November 2005; p. 10.
16. Heaps, J.; Krishnan, R.; Huang, Y.; Niu, J.; Sandhu, R. Access Control Policy Generation from User Stories Using Machine Learning. In Proceedings of the 35th IFIP Annual Conference on Data and Applications Security and Privacy (DBSec), Calgary, AB, Canada, 19–20 July 2021; pp. 171–188.
17. Cambria, E.; White, B. Jumping NLP Curves: A Review of Natural Language Processing Research. *IEEE Comput. Intell. Mag.* **2014**, *9*, 48–57. [CrossRef]
18. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
19. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
20. Arkhangelskaya, E.O.; Nikolenko, S.I. Deep Learning for Natural Language Processing: A Survey. *J. Math. Sci.* **2023**, *273*, 533–582. [CrossRef]
21. Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the 11th Annual Conference of the International Speech Communication Association (Interspeech 2010), Chiba, Japan, 26–30 September 2010.
22. Huang, Z.; Xu, W.; Yu, K. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv* **2015**, arXiv:1508.01991.
23. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv* **2014**. [CrossRef]
24. Lin, J.C.-W.; Shao, Y.; Djenouri, Y.; Yun, U. ASRNN: A recurrent neural network with an attention model for sequence labeling. *Knowl.-Based Syst.* **2021**, *212*, 106548. [CrossRef]
25. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338. [CrossRef]
26. Wang, M.; Zhou, T.; Wang, H.; Zhai, Y.; Dong, X. Chinese power dispatching text entity recognition based on a double-layer BiLSTM and multi-feature fusion. *Energy Rep.* **2022**, *8*, 980–987. [CrossRef]
27. Wu, K.; Wu, J.; Feng, L.; Yang, B.; Liang, R.; Yang, S.; Zhao, R. An attention-based CNN-LSTM-BiLSTM model for short-term electric load forecasting in integrated energy system. *Int. Trans. Electr. Energy Syst.* **2020**, *31*, e12637. [CrossRef]
28. Zhangjing, Data. 2024. Available online: https://gitee.com/ainer4869/data/tree/master/ (accessed on 25 April 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.