采用深度学习的DGA域名检测模型比较

裴兰珍[1,2] 赵英俊[1] 王 哲[1] 罗赟骞[2]
(空军工程大学防空反导学院 西安 710051)[1] (中国人民解放军95899部队 北京 100085)[2]

摘 要 针对DGA域名难以检测的问题,构建了一种面向字符的采用深度学习的DGA域名检测模型,模型由字符嵌入层、特征检测层和分类预测层组成。字符嵌入层实现对输入DGA域名的数字编码;特征检测层采用深度学习模型自动提取特征;分类预测层采用全连接网络进行分类预测。为了选取最优的特征提取模型,分析比较了采用Bidirectional机制、Stack机制和Attention机制的LSTM模型与GRU模型,CNN模型,以及将CNN模型分别与LSTM模型和GRU模型相组合的模型。结果表明,与LSTM和GRU模型相比,采用Stack机制、前向Attention机制结合Bidirectional机制的LSTM和GRU模型,CNN模型,CNN模型与LSTM和GRU相组合的模型可提升模型的检测效果,但采用CNN和Bi-GRU组合构建的DGA域名检测模型可获得最优的检测效果。

关键词 网络空间安全,深度学习,动态域名生成算法,卷积神经网络,门控循环单元,长短期记忆网络
中图法分类号 TP393.08 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.05.017

Comparison of DGA Domain Detection Models Using Deep Learning

PEI Lan-zhen[1,2] ZHAO Ying-jun[1] WANG Zhe[1] LUO Yun-qian[2]
(School of Air and Missile Defense, Air Force Engineering University, Xi'an 710051, China)[1]
(Army 95899 of PLA, Beijing 100085, China)[2]

Abstract For solving the problem of detection diffculty of the DGA domain name, this paper proposed a new DGA domain detection model from the viewpoint of character level by deep learning model. The model consisted of character embedding layer, feature detection layer and classification prediction layer. The character embedding layer realizes the digital encoding of DGA domain. The feature detection layer adopts the deep learning model to extract features automatically, and the classification prediction layer adopts neural network for classification prediction. In order to select the optimal model of feature extraction, the LSTM and GRU models using Bidirectional mechanism, Stack mechanism, Attention mechanism, CNN models and CNN models integrated respectively with LSTM and GRU model were compared. The results show that the LSTM and GRU models using Stack mechanism and Attention mechanism integrated with Bidirectional mechanism, CNN models and CNN models integrated with LSTM and GRU model can improve the detection effect. The DGA domain detection model using CNN model integrated with Bi-GRU can obtain the optimum detection effect.

Keywords Cyberspace security, Deep learning, Danamic domain generation algorithms, Convolutional neural network, Gatedrecurrent unit, Long short-term memory

1 引言

动态域名生成算法(Domain Generation Algorithms, DGA)能够有效地生成伪随机域名。恶意软件使用该域名连接其所使用的命令控制服务器(Command and Control server, C2),不仅可以有效地绕过黑名单检测,而且在命令控制服务器地址变更或部分失效时仍然可以连接服务器,提高了连接的可靠性。著名的恶意软件Zeus和Conficker等就使用了该算法[1]。该算法是实现恶意软件的关键技术,破解该技术可有效地检测恶意软件,对于提升企业、军队和国家的信息安全防护水平有着重要意义。

用于实时检测DGA域名的传统方法主要有:黑名单过滤和使用机器学习方法构建的DGA域名分类器。研究表明,黑名单难以适应动态变化的DGA域名,准确性较低[2];使用机器学习方法构建的DGA域名分类器需要人工构造特征,容易被DGA算法规避,且效率低[3-5]。深度学习方法能够自动提取特征,降低特征构造难度,并有效提高DGA域名检测的准确性[5]。文献[6]提出了一种采用LSTM的深度学

到稿日期:2018-04-17 返修日期:2018-06-29 本文受全军军事学研究生课题项目(2014JY514)资助。
裴兰珍(1982-),女,博士生,工程师,主要研究方向为装备作战运用与保障、软件测试,E-mail:peilanzhen2018@163.com;赵英俊(1966-),男,博士,教授,博士生导师,主要研究方向为装备作战运用与保障,E-mail:zhaoyingjun2018@163.com(通信作者);王 哲(1985-),男,博士生,讲师,主要研究方向为装备作战使用与保障;罗赟骞(1981-),男,博士后,工程师,CCF会员,主要研究方向为网络空间安全。

习模型来检测 DGA 域名；文献[7]提出采用集成的 CNN 模型对多种类型的恶意字符串进行检测；文献[8]使用实际域名对比了 CNN 和 LSTM 的检测效果；文献[9]分析了 LSTM 与 RNN 和传统机器学习算法的域名检测效果；文献[10]将迁移学习应用于 DGA 域名检测；文献[11]提出了一种将 GRU 和 Attention 机制相结合的 DGA 域名检测方法；文献[12]提出采用生成式对抗网络生成 DGA 域名以解决训练样本不足的问题。上述方法都是从单种角度构建 DGA 域名检测模型，而不同的深度学习方法可以从不同的角度提取特征。如果能够综合多种深度学习方法从不同的角度提取特征，则可以提升 DGA 域名的检测性能。

本文将 LSTM，GRU 和 CNN 等深度学习模型，以及综合多种机制的改进模型和组合模型应用于 DGA 域名检测，并分析比较了不同模型检测 DGA 域名的效果，从而选出最优的 DGA 检测模型。

2 应用于 DGA 域名检测的深度学习模型

深度学习可针对不同的应用场景采取不同的机制，从而有效地提升模型的性能。下面介绍本文采用的多个深度学习模型。

2.1 基于 Bidirectional 机制和 Stack 机制的深度学习模型

LSTM 和 GRU 模型由于在文本和时序数据处理上具有卓越的表现，目前已经在工业界和学术界得到了广泛应用。

LSTM 模型由多个 LSTM 单元组成。每个单元由遗忘门、输入门和输出门组成。每个单元在不同时刻完成模型的更新过程如下[13]。

(1)在时刻 t 下，计算输入层的输出值 i_t，并创建候选状态 $\tilde{c}_t$。

$$\boldsymbol{i}_t=\sigma(\boldsymbol{W}_i\boldsymbol{x}_t+\boldsymbol{U}_i\boldsymbol{h}_{t-1}+\boldsymbol{b}_i) \tag{1}$$

$$\tilde{\boldsymbol{c}}_t=\tanh(\boldsymbol{W}_c\boldsymbol{x}_t+\boldsymbol{U}_c\boldsymbol{h}_{t-1}+\boldsymbol{b}_c) \tag{2}$$

(2)计算遗忘门的输出值 f_t：

$$\boldsymbol{f}_t=\sigma(\boldsymbol{W}_f\boldsymbol{x}_t+\boldsymbol{U}_f\boldsymbol{h}_{t-1}+\boldsymbol{b}_f) \tag{3}$$

(3)在以上基础上，可以计算出 t 时刻单元的新状态量 $\boldsymbol{c}_t$。

$$\boldsymbol{c}_t=\boldsymbol{f}_t\odot\boldsymbol{c}_{t-1}+\boldsymbol{i}_t\odot\tilde{\boldsymbol{c}}_t \tag{4}$$

(4)使用新的状态量，可以计算输出门的状态以及单元的输出值。

$$\boldsymbol{o}_t=\sigma(\boldsymbol{W}_o\boldsymbol{x}_t+\boldsymbol{U}_o\boldsymbol{h}_{t-1}+\boldsymbol{b}_o) \tag{5}$$

$$\boldsymbol{h}_t=\boldsymbol{o}_t\odot\tanh(c_t) \tag{6}$$

以上公式中，x_t 为 t 时刻单元的输入值；h_t 为隐藏状态；i_t，f_t，c_t，o_t 为输入门、遗忘门、单元状态和输出门；W_i，W_f，W_c 和 W_o 为权重矩阵；b_i，b_f，b_c 和 b_o 为偏置；σ 为激活函数。

GRU 是 LSTM 的变种，其结构简单且效果好。GRU 与 LSTM 相比，只有更新门和重置门。不同时刻 GRU 模型的更新过程如下[14]。

(1)在时刻 t 下，分别计算更新门和重置门的输出值。

$$\boldsymbol{z}_t=\sigma(\boldsymbol{W}_z\boldsymbol{x}_t+\boldsymbol{U}_z\boldsymbol{h}_{t-1}+\boldsymbol{b}_z) \tag{7}$$

$$\boldsymbol{r}_t=\sigma(\boldsymbol{W}_r\boldsymbol{x}_t+\boldsymbol{U}_r\boldsymbol{h}_{t-1}+\boldsymbol{b}_r) \tag{8}$$

(2)计算时刻 t 下候选的隐藏状态。

$$\tilde{\boldsymbol{h}}_t=\tanh(\boldsymbol{W}_h\boldsymbol{x}_t+\boldsymbol{U}_h(\boldsymbol{r}_t\odot\boldsymbol{h}_{t-1})+\boldsymbol{b}_h) \tag{9}$$

(3)计算时刻 t 下的隐藏状态。

$$\boldsymbol{h}_t=(1-\boldsymbol{z}_t)\boldsymbol{h}_{t-1}+\boldsymbol{z}_t\tilde{\boldsymbol{h}}_t \tag{10}$$

以上公式中，x_t 为 t 时刻单元的输入值；h_t 为隐藏状态；z_t 和 r_t 分别为更新门和重置门；W_z，W_r 和 W_h 为权重矩阵；U_z，U_r 和 U_h 为隐藏状态的权重矩阵；b_z，b_r 和 b_h 为偏置；σ 为激活函数。

LSTM 和 GRU 可以通过增加网络中每层的单元数量或增加更多的层来扩大网络容量。层的堆叠机制(Stack)[15]可以增加网络容量，Google 翻译中就曾采用多层 LSTM 提高系统的容量。

Bidirectional 机制[15]常用于自然语言处理中，是深度学习应用于自然语言处理的一把“瑞士军刀”，它通过对输入的自然语言数据进行正向(chronological-order)和反向(reversed-order)处理，捕捉了可能仅由正向处理丢失的模式，获取了更加丰富的表现形式，进而得到了比正向处理更加有效的性能。

基于 Stack 机制和 Bidirectional 机制的 LSTM 和 GRU 模型可以提取 DGA 域名的时序特征。

2.2 采用 Attention 的深度学习模型

文献[16]给出了一种“前向”(feed-forward) Attention 机制，该机制将隐藏的状态序列向量 h_t 输入到一个可学习的函数 $a(h_t)$ 中生成一个可能向量 α，最后输出的上下文向量 c 由 α 给出的权重对 h_t 进行加权计算得到。从整个序列生成上下文向量 c 的计算公式如下：

$$\begin{cases}e_t=a(\boldsymbol{h}_t)\\ \alpha_t=\dfrac{\exp(e_t)}{\sum\limits_{k=1}^{T}\exp(e_k)}\\ \boldsymbol{c}=\sum\limits_{t=1}^{T}\alpha_t\boldsymbol{h}_t\end{cases} \tag{11}$$

a 是一个可学习的函数，如 sigmod 函数、tanh 函数，其仅仅依赖于 h_t。在式(1)中，Attention 可以被认为是通过计算状态序列 h 的自适应加权平均值，从而生成输入序列的固定长度嵌入上下文 c。使用 Attention 机制的作用是整合随着时间变化的信息。

文献[17]给出了一种上下文 Attention 机制。首先，通过一个可学习函数 a 获取字符 h_t 的一个隐藏表示 u_t；然后通过上下文向量 u_ω 与 u_t 之间的相似性度量字符的重要性，并且通过一个 softmax 函数获取归一化的重要权重 α；最后基于该权重计算整个序列的加权上下文向量 c。相关计算公式如下：

$$\begin{cases}\boldsymbol{u}_t=a(\boldsymbol{h}_t)\\ \alpha_t=\dfrac{\exp(\boldsymbol{u}_t^{\mathrm{T}}\boldsymbol{u}_s)}{\sum\limits_{t=1}^{T}\exp(\boldsymbol{u}_t^{\mathrm{T}}\boldsymbol{u}_s)}\\ \boldsymbol{c}=\sum\limits_{t=1}^{T}\alpha_t\boldsymbol{h}_t\end{cases} \tag{12}$$

u_t 为一个向量；u_s 也是随机初始化的向量，在训练过程中进行学习。与“前向”Attention 机制相比，该方式相当于增加了一层神经网络层。

2.3 CNN 模型

CNN 模型在图像识别领域取得了巨大的成功，目前在自然语言处理领域由于其效率和准确性也得到了广泛的应用。

针对 DGA 域名检测，可以在字符层面构建模型，在嵌入空间中嵌入文本字符串的字符，然后使用卷积的方法提取特征。CNN 模型可提取 DGA 域名的空间结构特征。

3　采用深度学习的 DGA 域名检测模型

本文将采用第 2 节所提到的深度学习模型构建 DGA 域名检测模型，模型主要包括 3 层：字符嵌入、特征检测和分类预测。字符嵌入层将输入的原始字符串序列编码成一个二维张量。特征检测层根据输入的张量提取输入字符序列的重要特征模式，并将这些信息聚合到固定长度的特征向量中，从而实现自动化的特征提取。特征检测模块使用第 2 节中的深度学习模型。分类预测层使用全连接神经网络分类被检测特征。

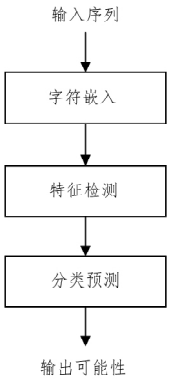

图 1　DGA 域名检测模型

Fig. 1　DGA domain detection model

3.1　字符嵌入层

字符嵌入层对一个长度为 s 的字符串序列进行处理，如果输入序列长度大于 s，则去除多余的字符，如果输入序列长度小于 s，则补齐；然后，将字符串序列嵌入到一个 $s\times d$ 的浮点数矩阵中。输入字符串序列包含非冗余的合法域名称字符（小写字母数字、句点、破折号和下划线），输出空间 d 是一个可调参数，表示嵌入，本文中 $d=128$。这是一个简单的字典查询操作，字符被映射成相应的向量，然后这些向量被合成为一个矩阵。矩阵的行表示原始字符的字符序列，矩阵的列表示嵌入空间的维数。嵌入层通过反向传播方法与剩余模型进行联合优化，优化单个字符的嵌入向量更能反映语义；这种聚合字符语义相似的方法使得其底层模型能够更好地识别字符串中的语义相似性。

3.2　特征检测层

特征检测层将根据字符嵌入层的输出结果，使用深度学习模型抽取和聚合被检测特征，其作用是将矩阵形式的输入编码为较低维度的一维向量，从而保留大多有用信息。本文将使用深度学习中 CNN 模型、LSTM 模型、GRU 模型及其变种和组合模型提取特征。在提取特征上，CNN 模型注重全局的模糊感知，而 LSTM 和 GRU 模型则注重邻近位置的重构。

3.2.1　采用 LSTM 和 GRU 模型的特征提取

由于采用 Bidirectional 机制和 Stack 机制能够有效地提高模型的性能，并且采用 Attention 机制也可以提高模型的性能，因此本文将分别使用这些机制来构建基于 LSTM 和 GRU 模型的特征检测层。

3.2.2　采用 CNN 模型的特征提取

CNN 模型根据输入的二维张量，检测在全字符序列中重要的局部序列模式，然后将这些信息聚合到一个固定长度的特征向量中。使用多核卷积 $Conv(t,k,n)$ 检测局部特征，其中过滤器 $k=\{2,3,4,5\}$，卷积核数量 $t=256$，n 为输入的张量。对原始字符嵌入矩阵计算卷积在概念上与传统的词袋方法相似，区别在于其不用直接检测 n-grams，而是对语法上相似的子字符进行近似匹配。

3.2.3　采用 CNN 和 LSTM 与 GRU 模型结合的特征提取

因为 1D 的卷积网络独立地处理输入块，所以它与 RNN 模型不同，对输入的时间顺序不敏感；尽管可以通过堆叠多层的卷积层和池化层使其能够识别长的序列模式，但其仍然不能克服对时间序列不敏感的问题。为了克服该问题，可以将 CNN 预处理后的结果输入到 LSTM 和 GRU 模型进行再次处理，这种方式可以结合 CNN 模型与 LSTM 和 GRU 模型的优点。

3.3　分类预测层

一旦抽取了特征，就可以使用标准的全连接神经网络将字符串分类为恶意的或良性的。对于 LSTM 和 GRU 模型，使用 1 层神经网络进行预测；对于 CNN 模型，使用 3 层神经网络进行预测，神经网络使用 sigmod 函数。本文使用 binary-cross 熵来衡量检测器的损失值：

$$L(\hat{\boldsymbol{y}},\boldsymbol{y})=-\frac{1}{N}\sum_{i}^{N}[_i\log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i)] \tag{13}$$

其中，$\hat{\boldsymbol{y}}$ 是对所有输入的 URL 进行预测的可能性向量，$\boldsymbol{y}$ 是所有输入的 URL 对应的真实标记值，良性为 0，恶性为 1。

4　实例分析

4.1　数据来源及嵌入层处理

本文中的数据采用文献[6]使用的公开数据集。良性域名数据集使用来自 Alexa 的前 100 万个域名；DGA 域名使用来自 OSINT DGA 库的 30 多种类型的 DGA 域名，该库中包含了大约 75 万个 DGA 实例。实验时，随机从上述数据集中各取 11 万个良性域名和 DGA 域名组成 22 万个样本用于实验。

对输入的恶意和良性域名进行处理，获取所有域名中所用的字符，建立相应的数字字符字典表，将输入域名中的字符按字典表中的数字键值进行编码。将编码后的字符串输入嵌入层，形成输入字符串向量矩阵。本文中输入字符串的最大长度为 53，每个字符由长度为 128 的向量表示。

4.2　输入特征的提取

目前可用于特征提取的方法很多，不同方法可能在特定环境下具有最优特性。为了获取最优特征提取方法，本文比较了 25 种特征提取方法，如表 1 所列。

表 1 中，CNN(2)表示一维卷积 Convolution1D，其中的 2 表示过滤器的长度，卷积核的数量为 256，feed-attention 表示前向 Attention 机制，con-attention 表示上下文 Attention 机制。模型 17 中的第一层是 4 个卷积神经网络的并列处理，第二层则将第一层的结果合成在一起。

表 1 基于深度学习的特征提取模型

Table 1 Feature extraction model based on deep learning

名称	第一层	第 2 层	第 3 层	第 4 层	第 5 层
模型 1	GRU(128)	Drop(0.5)	Dense(1)	—	—
模型 2	LSTM(128)	Drop(0.5)	Dense(1)	—	—
模型 3	Bi-GRU(128)	Drop(0.5)	Dense(1)	—	—
模型 4	Bi-LSTM(128)	Drop(0.5)	Dense(1)	—	—
模型 5	GRU(128)	Drop(0.5)	GRU(128)	Drop(0.5)	Dense(1)
模型 6	LSTM(128)	Drop(0.5)	LSTM(128)	Drop(0.5)	Dense(1)
模型 7	Bi-GRU(128)	Drop(0.5)	Bi-GRU(128)	Drop(0.5)	Dense(1)
模型 8	Bi-LSTM(128)	Drop(0.5)	Bi-LSTM(128)	Drop(0.5)	Dense(1)
模型 9	GRU(128)	Drop(0.5)	feed-attention	Dense(1)	—
模型 10	LSTM(128)	Drop(0.5)	feed-attention	Dense(1)	—
模型 11	Bi-GRU(128)	Drop(0.5)	feed-attention	Dense(1)	—
模型 12	Bi-LSTM(128)	Drop(0.5)	feed-attention	Dense(1)	—
模型 13	GRU(128)	Drop(0.5)	con-attention	Dense(1)	—
模型 14	LSTM(128)	Drop(0.5)	con-attention	Dense(1)	—
模型 15	Bi-GRU(128)	Drop(0.5)	con-attention	Dense(1)	—
模型 16	Bi-LSTM(128)	Drop(0.5)	con-attention	Dense(1)	—
模型 17	CNN(2)+drop(0.5) CNN(3)+drop(0.5) CNN(4)+drop(0.5) CNN(5)+drop(0.5)	Merge(1024)	Dense(1024)	Dense(1024)	Dense(1)
模型 18	CNN(2)	Drop(0.5)	Dense(256)	Dense(256)	Dense(1)
模型 19	CNN(3)	Drop(0.5)	Dense(256)	Dense(256)	Dense(1)
模型 20	CNN(4)	Drop(0.5)	Dense(256)	Dense(256)	Dense(1)
模型 21	CNN(5)	Drop(0.5)	Dense(256)	Dense(256)	Dense(1)
模型 22	CNN(5)	Drop(0.5)	GRU(256)	Drop(0.5)	Dense(1)
模型 23	CNN(5)	Drop(0.5)	LSTM(256)	Drop(0.5)	Dense(1)
模型 24	CNN(5)	Drop(0.5)	Bi-GRU(128)	Drop(0.5)	Dense(1)
模型 25	CNN(5)	Drop(0.5)	Bi-LSTM(128)	Drop(0.5)	Dense(1)

4.3 输出预测

本文采用 AUC(Area Under the Curve)值衡量二分类问题的机器学习泛化能力。AUC 值是 ROC(Receiver Operating Characteristic)的曲线面积,ROC 实现了对 TPR(True Positive Rate)和 FPR(False Positive Rate)的权衡度量,它通过不同的阈值对 TPR 和 FPR 进行计算:

$$TPR=\frac{\sum \text{True Positive}}{\sum \text{True Positive}+\sum \text{False Negative}} \tag{14}$$

$$FPR=\frac{\sum \text{False Positive}}{\sum \text{False Positive}+\sum \text{True Negative}} \tag{15}$$

在得到不同阈值下的 TPR 和 FPR 之后,可以采用梯形法则(Trapezoidal Rule)计算 AUC 值。梯形法则的计算方法如下[18]:

$$\int_a^b f(x)\mathrm{d}x \approx \sum_{k=1}^{N}\frac{f(x_{k-1})+f(x_k)}{2}\Delta x_k \tag{16}$$

利用$\{x_k\}$对区间$[a,b]$进行划分,使得 $a=x_0<x_1<\cdots<x_N=b$,Δx_k 是子区间 k 的长度,即 $\Delta x_k=x_k-x_{k-1}$。计算 AUC 时,$a=0$,$b=1$,$\{x_k\}$对应的是不同阈值情况下的 FPR 值,$f(x_k)$为对应的 TPR 值。

4.4 实验的结果分析

本文实验运行环境为 ubuntu 16.04 LTS,8 GB 内存,Intel® Core™ i5-6200U CPU @ 2.30 GHz×4,程序开发环境为 Anaconda4.4.0,Python 版本为 2.7.13,Keras 版本为 2.0.6。

本文将 80% 的样本数据作为模型的训练样本数据,将 20% 的数据模型作为测试样本数据。模型训练时可采用交叉验证方法和 HoldOut 检验法选择最优模型;由于计算资源有限,本文在模型训练时采用 HoldOut 检验法。选择最优模型时使用 95% 的训练样本数据构建模型,使用 5% 的训练样本数据进行模型验证。模型训练过程中,如果模型的 AUC 值连续 5 次没有发生变化,停止训练过程。不同模型各运行 3 次,每个模型的性能指标为 3 次运行结果的平均值。

为了获取最佳的 DGA 域名检测模型,本文使用表 1 中的多种深度学习方法对 DGA 域名进行检测,并比较不同模型的性能。文中主要分析模型的训练时间、训练损失值和训练 AUC 值,以及模型预测的 AUC 值,如表 2 所列。

表 2 多种检测模型的性能比较

Table 2 Performance comparison of various detection models

名称	训练时间/s	训练损失值	训练 AUC	预测 AUC
模型 1	2440	0.0526	0.9970	0.9966
模型 2	5046	0.0439	0.9965	0.9964
模型 3	4941	0.0526	0.9967	0.9966
模型 4	9964	0.0429	0.9968	0.9966
模型 5	6415	0.0471	0.9967	0.9969
模型 6	10122	0.0513	0.9970	0.9967
模型 7	11410	0.0501	0.9969	0.9970
模型 8	21957	0.0420	0.9966	0.9970
模型 9	3873	0.0580	0.9959	0.9959
模型 10	6967	0.0545	0.9960	0.9955
模型 11	7131	0.0418	0.9971	0.9970
模型 12	12694	0.0387	0.9966	0.9969
模型 13	4969	0.0549	0.9963	0.9964
模型 14	8192	0.0514	0.9962	0.9964
模型 15	8838	0.0545	0.9965	0.9964
模型 16	15134	0.0495	0.9964	0.9964
模型 17	11296	0.0579	0.9974	0.9974
模型 18	2560	0.1182	0.9952	0.9948
模型 19	1950	0.0962	0.9964	0.9963
模型 20	2717	0.0793	0.9967	0.9971
模型 21	3685	0.0699	0.9973	0.9972
模型 22	11304	0.0517	0.9977	0.9978
模型 23	14085	0.0508	0.9981	0.9978
模型 24	11820	0.0488	0.9980	0.9978
模型 25	12714	0.0529	0.9982	0.9977

本文以模型 1 和模型 2 为基本测试基线，对比分析多种在其基础上改进的模型的性能。模型 3 和模型 4 采用 Bidirectional 机制，预测性能提升不明显。模型 5 和模型 6 采用堆叠的方法提取特征，预测 *AUC* 性能有小幅提升。模型 7 和模型 8 将 Bidirectional 机制和堆叠机制相结合，预测 *AUC* 性能有小幅提升。模型 9－模型 12 采用了前向 Attention 机制，可以看到在模型 1 和模型 2 的基础上使用该机制时性能不升反降，在模型 3 和模型 4 的基础上使用该机制时性能有小幅提升。模型 13－模型 16 采用上下文 Attention 机制，基于模型 1－模型 4 使用该机制时性能基本没有变化。模型 17 采用过滤器长度为 2，3，4，5 的 4 类 CNN 并行提取特征，其预测 *AUC* 性能有较大幅度的提升。模型 18－模型 21 分别采用过滤器长度为 2，3，4，5 的 4 类 CNN 提取特征，可以看到过滤器的大小对预测性能影响较大，过滤器长度越长时模型性能越好。模型 22－模型 25 将 CNN 和模型 1－模型 4 分别进行组合，先使用 CNN 提取局部特征，然后使用 LSTM 和 GRU 提取局部特征间的时序关系，组合后的模型在预测性能上有较为明显的提升；模型 24 最好的 *AUC* 预测效果甚至能够达到 0.9980。另外可以看到，模型中的预测 *AUC* 值有明显提升时，模型的训练 *AUC* 也有明显提升，但是训练时间也随之大幅增加。

结束语　本文给出了一种基于深度学习方法检测 DGA 域名的通用框架；基于该框架，对检测 DGA 域名的多种深度学习方法进行了比较。研究结果表明：1）堆叠机制有利于提升模型的预测性能；2）前向 Attention 机制结合 Bidirectional 机制时有利于模型性能的提升，上下文 Attention 机制不利于模型性能的提升；3）采用 CNN 可以有效提升模型性能，但是模型性能对过滤器大小敏感，需要仔细选择过滤器值；4）在多种模型中，采用 CNN 和 GRU，LSTM 组合的方式可有效提升模型的检测性能；CNN 和 Bi-GRU 组合模型在所有模型中的检测效果最好。由于采用 Bidirectional 机制时，模型的训练时间将大幅度增加，未来将对 GRU 和 LSTM 进行改进，以提升模型的训练效率作为进一步的研究方向；另外，如何应对智能化的 DGA 域名生成机制也是未来的一个重要研究方向。

参 考 文 献

[1] ABAKUMOV A. DGA[EB/OL]. (2017-07-31)[2018-04-13]. https://github.com/andrewaeva/ DGA.

[2] SHA H Z, LIU Q Y, LIU T W, et al. Survey on Malicious Webpage Detection Research [J]. Chinese Journal of Computers, 2016, 39(3): 529-542. (in Chinese)
沙泓州，刘庆云，柳厅文，等. 恶意网页识别研究综述[J]. 计算机学报，2016，39(3)：529-542.

[3] ZHAO G, XU K, XU L, et al. Detecting APT Malware Infections Based on Malicious DNS and Traffic Analysis[J]. IEEE Access, 2015, 3: 1132-1142.

[4] WANG X, WU Y, LU Z G. Study on Malicious URL Detection Based on Threat Intelligence Platform[J]. Computer Science, 2018, 45(3): 124-130, 170. (in Chinese)
汪鑫，武杨，卢志刚. 基于威胁情报平台的恶意 URL 检测研究[J]. 计算机科学，2018，45(3)：124-130，170.

[5] SAHOO D, LIU C H, HOI S. Malicious URL Detection using Machine Learning: A Survey[EB/OL]. (2017-03-16)[2018-04-13]. https://arxiv.org/abs/ 1701.07179.

[6] WOODBRIDGE J, ANDERSON H, AHUJA A, et al. Predicting Domain Generation Algorithms with Long Short-Term Memory Networks[EB/OL]. (2016-11-02)[2018-04-13]. https://arxiv.org/abs/ 1611.00791.

[7] SAXE J, BERLIN K. eXpose: A Character-Level Convolutional Neural Network with Embeddings For Detecting Malicious URLs, File Paths and Registry Keys[EB/OL]. (2017-02-27)[2018-04-13]. https://arxiv.org/abs/1702.08568.

[8] YU B, GRAY D L, PAN J. Inline DGA Detection with Deep Networks [C]// 2017 IEEE International Conference on Data Mining Workshops (ICDMW). New Orleans: IEEE Press, 2017: 2375-9259.

[9] VINAYAKUMAR R, SOMAN K P, POORNACHANDRAN P. Detecting malicious domain names using deep learning approaches at scale[J]. Journal of Intelligent and Fuzzy Systems, 2018, 34(3): 1355-1367.

[10] ZENG F, CHANG S, WAN X C. Classification for DGA-Based Malicious Domain Names with Deep Learning Architectures[J]. International Journal of Intelligent Information Systems, 2017, 6(6): 67-71.

[11] 陈立皇，程华，房一泉. 基于注意力机制的 DGA 域名检测算法[EB/OL]. (2018-06-19)[2018-06-25]. http://kns.cnki.net/kcms/detail/31.1691.TQ.20180615.1620.004.html.

[12] ANDERSON H S. DeepDGA: Adversarially-Tuned Domain Generation and Detection [C]// AISec'16 Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security. New York: ACM Press, 2016: 13-21.

[13] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural Computation, 1997, 9(8): 1735-1780.

[14] CHO K, MERRIENBOER B V, GULCEHRE C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation [EB/OL]. (2014-09-03)[2018-06-13]. https://arxiv.org/abs/1406.1078.

[15] FANCOIS C. Deep Learning with Python[M]. New York: Manning Publications, 2017: 192-215.

[16] RAFFEL C, ELLIS P W. Feed-Forward Networks with Attention Can Solve Some Long-Term Memory Problems[EB/OL]. (2016-09-20)[2018-04-13]. https://arxiv.org/abs/1512.08756.

[17] YANG Z, YANGD, DYER C, et al. Hierarchical Attention Networks for Document Classification [C]// NAACL-HLT 2016: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego: Association for Computational Linguistics, 2016: 1480-1489.

[18] Wikipedia. Trapezoidal rule[EB/OL]. (2018-03-16)[2018-04-13]. https://en.wikipedia.org/wiki/ Trapezoidal_rule.