基于循环神经网络和生成式对抗网络的口令猜测模型研究

汪 定[1),2)] 邹云开[1),2)] 陶 义[3)] 王 彬[3)]

[1)](南开大学网络空间安全学院 天津 300350)

[2)](天津市网络与数据安全技术重点实验室(南开大学) 天津 300350)

[3)](北京大学信息科学技术学院 北京 100871)

摘 要 深度学习技术的进展为提高口令猜测效率提供了潜在的新途径.目前,已有研究将循环神经网络(Recurrent Neural Network,RNN)、生成式对抗网络(Generative Adversarial Network,GAN)等深度学习模型运用于设计口令猜测模型.本文基于 RNN 模型、概率上下文无关文法(Probabilistic Context-Free Grammar,PCFG)与长短期记忆网络(Long Short-Term Memory,LSTM)的混合模型(简称 PL 模型),提出采用 RNN 来代替 PL 模型中的 LSTM 的思想,将 PCFG 与 RNN 在模型层面进行融合,设计了 PR 模型.为降低猜测模型对大训练样本的依赖,进一步提出了 PR+模型,即采用 RNN 来生成字母序列,实现对口令字母段的填充.基于 4 个大规模真实口令数据集的实验结果显示,PR 模型的破解率略高于 PL 模型,且始终显著高于传统的 PCFG(10^7 量级猜测数下)和 Markov 模型(10^6 量级猜测数下),并且 PR 模型的训练效率远优于 PL 模型.鉴于不同口令模型生成口令猜测的特性不同,将不同模型生成的猜测集组合来生成新的口令猜测集,并基于 4 个大规模真实口令数据集对不同组合方法进行了对比.尽作者所知,我们首次证实了在相同猜测数下($10^4\sim10^8$ 量级猜测数),组合不同类型模型所生成口令猜测集的破解率通常高于单一猜测集.本文研究显示,GAN 模型在猜测数为 3.6×10^8 时,破解率仅为 31.41%,这表明 GAN 模型的口令破解效率劣于传统基于概率统计的模型(如 PCFG 模型和 Markov 模型)和基于 RNN 的口令猜测模型,并进一步指出了 GAN 模型表现不佳的原因.

关键词 口令;猜测攻击;深度学习;循环神经网络;生成式对抗网络

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2021.01519

Password Guessing Based on Recurrent Neural Networks and Generative Adversarial Networks

WANG Ding[1),2)] ZOU Yun-Kai[1),2)] TAO Yi[3)] WANG Bin[3)]

[1)](*College of Cyber Science, Nankai University, Tianjin* 300350)

[2)](*Tianjin Key Laboratory of Network and Data Security Technology (Nankai University), Tianjin* 300350)

[3)](*School of Electronics Engineering and Computer Science, Peking University, Beijing* 100871)

Abstract The progress of deep learning technology provides a potential way to improve the efficiency of password cracking. At present, there have been researches on applying deep learning models such as Recurrent Neural Networks(RNN) and Generative Adversarial Networks (GAN) to password guessing. Based on the implementation of password guessing algorithms such as RNN, PL(the combination of Probabilistic Context Free Grammar(PCFG) and Long Short-Term Memory(LSTM) at the model level) and GAN, this paper uses RNN instead of LSTM in the PL model and proposes the PR model (the combination of PCFG and RNN). To reduce the dependence of the guessing model on large training samples, we use the RNN network to generate the filling

收稿日期:2020-12-29;在线发布日期:2021-04-16.本课题得到国家自然科学基金(61802006)和国家重点研发计划"前沿科技创新"重点专项课题资助.**汪 定**(通信作者),博士,教授,博士生导师,主要研究领域为口令安全、密码协议.E-mail:wangding@nankai.edu.cn.**邹云开**,博士研究生,主要研究方向为口令安全与深度学习.**陶 义**,学士,研究方向为口令安全.**王 彬**,学士,研究方向为口令安全.

set of the letter segment of password, and propose the PR+model. In the experiments, we use 4 different data sets to test the cracking ability of different models. The results show that PR model is slightly higher than PL model and significantly higher than traditional main-stream models, i.e., PCFG ($<10^7$ guesses) and Markov ($<10^6$ guesses), in most data sets. At the same time, the training efficiency of PR model is far better than that of PL model. Due to the different characteristics of password samples generated by different models, we further adopt combinations of different guess sets to perform the same test process based on 4 real large-scale password datasets. To the best of our knowledge, we have confirmed for the first time that the combined guess set of different models is higher than that of the single guess set under the same guess number (10^7-10^8 guesses). While for the GAN model, when the guess number is 3.6×10^8, the cracking rate is only 31.41%. This indicates that the cracking rates of GAN is inferior to traditional statistics-based methods (such as PCFG and Markov) and RNN-based models, and we further explain the reason.

Keywords password; guessing attack; deep learning; recursive neural network; generative adversarial network

1 引 言

随着大数据、物联网、云计算等技术的不断发展,人们的日常生活与各种数字化网络服务的联系变得越来越紧密.一方面,越来越多的网络服务需要口令保护,另一方面,人类的大脑记忆十分有限,只能记忆 5～7 个口令[1],这将导致用户在设置口令时,不可避免地采用低信息熵的弱口令[2],或在不同服务中使用相同口令[3],这些行为在多起口令数据库的泄漏事件中都得到了印证[4-5],并且带来了严重的安全威胁.尽管口令存在一些可用性缺陷或安全性问题,而且近年来还不断有新型的认证技术被提出,但是考虑到口令易于实现,便于更改,不需要搭载特殊的软硬件,并且用户和开发者对口令都很熟悉[6],因而在可预见的未来,口令仍将是最主要的身份认证方式之一[7-8].与此同时,新的口令猜测方法陆续出现,攻击者的计算能力也在不断提高,这导致口令面临的安全威胁不断加剧.因此,研究用户口令的安全性具有十分重要的现实需求,而口令猜测技术则是研究口令安全性的核心方向之一.

1.1 相关工作

常见的口令猜测方法主要分为三类.第一类是基于字典的方法[9-10],该类方法将字典中的词汇直接或通过规则变换作为口令猜测集.著名的口令破解工具,如 HashCat[9] 和 John the Ripper[10],会对字典中的单词执行相应的规则变换来扩展猜测样本,比如将“password”变为“pas5w0rd”等,但是此类方法需要由人工去制定变换规则,因而猜测效果在一定程度上受限于个人经验.第二类是基于统计学规律的方法[11-14],典型的代表有 Markov 模型[11]和概率上下文无关文法(Probabilistic Context-Free Grammar,PCFG)模型[12],它们分别根据口令的前后字符依赖关系和口令的结构组成来进行建模,具有一定的系统性和理论性.在此基础上,许多研究者陆续提出了一系列改进方法,如对 Markov 模型进行正规化和平滑处理[5];按概率递减顺序枚举生成的口令[13];在 PCFG 的基础上加入键盘词模式[14];针对长口令做出适应性改进[15];将 PCFG 和 Markov 相结合形成混合攻击模型等[16].此外,在文献[17-19]中,作者利用与攻击对象相关的个人信息,分别提出了 Targeted-Markov,Personal-PCFG 以及 TarGuess 猜测框架(包括 7 个算法),增强了口令猜测的针对性.2017 年,Wang 等人[20]通过大规模真实数据集证实了口令分布服从 Zipf 定律,并指出 Markov 和 PCFG 模型生成的口令猜测集都遵循这一规律[21],这为基于概率的口令攻击方法提供了有效的理论支持.

近年来,深度学习技术的发展为改进口令猜测技术、提高口令破解效率提供了潜在的新途径,逐渐形成了口令破解的第三类方法.2016 年,Melicher 等人[22]首次使用循环神经网络(Recurrent Neural Network,RNN)进行口令破解.RNN 可以根据某个字符串前缀预测下一位的字符,如根据一个口令的前 5 位预测口令的第 6 位.Xu 等人[23]使用 RNN 的一个变种,即 LSTM 进行口令破解,在生成 3.35×10^9

个猜测样本的情况下，能够得到比 Markov 模型和 PCFG 模型更高的破解率. 2018 年，Zhou 等人[24]将个人信息与 RNN 相结合，提出了定向口令猜测模型 TPGXNN，猜测成功率高于同样场景下的 PCFG 和 Markov 模型.

2019 年，Hitaj 等人[25]首次提出利用生成式对抗网络(Generative Adversarial Networks，GAN)来破解口令，他们将模型命名为 PassGAN. GAN 同时训练两个神经网络 G(Generator)和 D(Discriminator)，G 通过学习真实数据的分布产生新的数据，D 判断数据是来源于 G 还是来源于真实数据. 两个网络以相互对抗的方式共同进步，直到 G 产生的数据和真实的数据难以区分[26]. 虽然 GAN 在图像生成方面卓有成效，但是在文本生成方面效果一直不理想. 针对这一问题，Gulrajani 等人[27]和 Rajeswar 等人[28]分别提出了使 GAN 在文本生成方面效果更好的方案. 在文献[27]提出的网络架构中，G 和 D 均由卷积神经网络(Convolutional Neural Networks，CNN)构成，而文献[28]和[29]分别探索了 G 和 D 均为 RNN 和 GRU 的情况.

总体看来，GAN 模型的应用为口令破解提供了新的技术途径. 其优势包括：不需要人工设置规则；可以生成无限多的口令；随着猜测数的增加，破解率一直在稳步上升等. 不过相比传统的口令破解方法，GAN 的效果并不理想，这与 GAN 自身在文本数据上的局限性有关. 具体来说，在生成相同数目的猜测样本(441 357 719 个口令)时，PassGAN 的破解率比自动化口令猜测工具 HashCat best64 规则低 36.7%. 本文实验结果显示，Markov 模型和 PCFG 在猜测效率方面也优于 PassGAN. 本文尝试对 PassGAN 进行改进并做出相应的分析，以进一步探究 GAN 模型在口令破解方面的可行性.

1.2　主要贡献

本文的主要贡献有三：

(1) 将 PCFG 和 RNN 相结合，提出新的 PR(PCFG+RNN)模型(模型层面的组合). PR 模型借鉴了 2018 年 Liu 等人[30]提出的 PL(PCFG+LSTM)模型. 结果显示，PR 模型在多数数据集上的破解率略高于 PL 模型，且训练效率远高于后者. 此外，这两个模型的破解率通常高于传统的 PCFG 和 Markov 模型. 为降低猜测模型对训练样本的依赖，在 PR 模型的基础上，本文额外使用 RNN 用于生成口令字母段的填充内容，提出了 PR+模型. PR 模型和 PR+模型为口令猜测提供了新的技术途径.

(2) 由于不同口令模型生成的口令具有不同特性，本文还将各类基本模型所产生的口令进行组合来产生新的猜测样本集合(即“RNN+PCFG”、“RNN+Markov”、“RNN+PR”、“LSTM+PCFG”、“LSTM+Markov”和“LSTM+PR”六种组合方式)，基于大规模真实口令数据集实验结果，首次证实了组合不同类型模型所生成口令猜测集的破解率通常高于单一猜测集，进一步提升了口令的破解效率.

(3) 对 Hitaj 等人[25]提出的 PassGAN 模型进行了研究，指出其存在输出定长和输出冗余这两个缺陷. 进一步研究了网络层数对 PassGAN[25]破解率的影响，发现层数较深时，破解率会降低. 此外，PassGAN 的破解率始终在现有口令猜测模型中表现最差，比如在猜测数为 3.6×10^8 时，破解率仅为 31.41%. 最后，分析了 GAN 模型在用于口令破解时存在的根本性问题.

1.3　组织结构

本文第 2 节简要介绍 RNN 和 GAN；第 3 节详细给出基于 RNN(LSTM)的口令猜测模型；第 4 节提出 PR 模型和 PR+模型，并对不同模型生成的口令进行组合，进行全面的对比实验；第 5 节探索 GAN 在口令破解上的应用；第 6 节总结全文，并指出下一步的研究方向.

2　循环神经网络和生成式对抗网络

2.1　循环神经网络

循环神经网络(Recurrent Neural Networks，RNN)[31]是一种特殊的神经网络，一般用于与时间序列相关的任务，它是由一系列结构相同的神经元构成，该神经元在每个时间步骤中重复使用. RNN 的神经元内部有一个记忆状态，在处理序列数据时，输入不仅仅有序列数据，还有上一个时刻的记忆状态，并向下一个时刻输出当前的记忆状态. 记忆状态不断的作为输入和输出，其结构在理论上可以看作是一个运算单元无限循环的结果. 如图 1 所示，为 RNN 在时间上的展开.

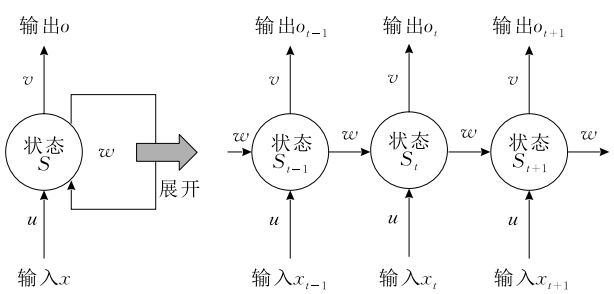


图 1　循环神经网络示意图

将图 1 左侧的 RNN 按照时间步骤展开，就得到图 1 右侧的神经网络. 其中，x 为输入，o 为输出，S 为隐藏状态，下标 t 为不同的时间步，u、v、w 分别为对应的权重.

2.2 生成式对抗网络

生成式对抗网络(Generative Adversarial Networks，GAN)[26]是一种训练和评估生成式模型的框架. 在该框架下，同时训练两个模型：一个生成式模型 G 和一个判别式模型 D. G 致力于学习真实数据的分布，以此生成新的数据，而 D 致力于分辨输入来源于真实数据还是 G 生成的数据.

在生成式对抗网络中，G 的输入是噪声 z，输出是 $G(z;\varnothing_g)$，其中 $\varnothing_g$ 为 G 的参数. D 的输入是样本 x，该样本来源于真实数据，或来源于 G；D 的输出记为标量 $D(x;\varnothing_d)$，表示 x 来自真实数据的概率，$\varnothing_d$ 为 D 的参数. G 的目标是使 $D(G(z))$ 尽可能大，而 D 的目标是使它的输出尽可能准确.

生成式对抗网络 GAN 表现出了较大的潜力，但是训练起来比较困难，尤其需要注意交替训练 G 和 D. Martin 等人[32]提出 Wasserstein GAN(简称 WGAN)，将需要优化的函数从 GAN 中的 Jenson-Shannon(JS)距离替换成 Earth-Mover(EM)距离，并对 D 的参数加以约束，使得在每次迭代中一次性把 D 训练到最优，然后再训练 G. Gulrajani 等人[27]进一步对 WGAN 进行了改进，得到了基于梯度惩罚(gradient penalty)的 WGAN 模型(WGAN-GP)，提高了训练的稳定性，而 PassGAN[25]的网络框架即为 WGAN-GP.

3 基于 RNN 的口令猜测模型

3.1 数据集与预处理

本文使用了 4 个大规模真实口令集，它们来自不同服务类型和规模的网站，用户的语言与文化背景也各异，这将充分显示本文所提模型的普适性. 4 个口令集的基本信息详见表 1.

表 1 本文所使用口令集的基本信息

口令集	服务类型	地域	语言	口令总数
CSDN	程序员论坛	中国	中文	6 428 287
Rockyou	游戏	美国	英文	32 603 388
Tianya	社交网站	中国	中文	30 233 633
Yahoo	互联网门户	美国	英文	5 626 485

以 CSDN 数据集为例，按照 8∶1∶1 的比例划分为训练集、验证集和测试集. 训练集包含 5.1×10^6 个口令，验证集和测试集均包含 6.4×10^5 个口令. 训练集用于对模型进行训练，验证集用于对模型的效果进行分析并修改模型参数，最后使用模型对测试集进行测试，并输出测试结果.

模型的输入为真实口令，因此可以将其作为字符序列来进行处理. 对口令中每个字符进行编码，将其转化为向量用于网络训练. 口令中的字符通常都包含在 95 个可打印字符中，基于此，采用独热编码(One-Hot Encoding)来对 95 个可打印字符进行编码. 对于每个口令，都可以转换为一个矩阵. 独热编码后的矩阵具有很好的稀疏性，在这里将以一个简单的例子来进行说明.

假设字符表为(a，b，c)，字符的编码分别为 a=[1，0，0]，b=[0，1，0]，c=[0，0，1]. 那么口令“abccba”的编码为[[1，0，0]，[0，1，0]，[0，0，1]，[0，0，1]，[0，1，0]，[1，0，0]]. 独热编码可以很好地反应不同字符之间的距离，这样便于模型对不同的字符进行分类.

在实际的编码中，由于绝大多数口令中不包含空格，所以不把空格放入字符集中，同时添加终结符. 对于长度为 n 的口令，编码后被转换为 $n\times95$ 的矩阵.

3.2 模型架构

基于 RNN[31]的口令猜测模型将学习字符串中每个字符与它前面的字符的关系. 生成样本集时，RNN 模型根据前面的字符依次预测字符串中的下一个字符. 如图 2 所示为本文所使用的 RNN 模型[31]的基本架构.

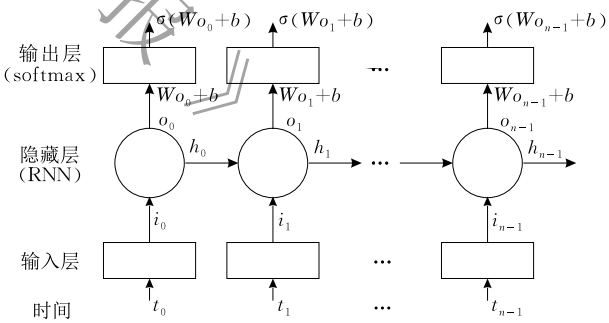


图 2 RNN 模型

本文的网络结构包括输入层、隐藏层、全连接层和 softmax 层. 输入层接收输入信号(值)并将其传递至下一层，但不对输入信号(值)执行任何运算. 在本文的网络中，输入层接收口令编码后的矩阵并向后传递给隐藏层. 隐藏层对输入进行计算，并输出结果. 可以将该计算过程理解为，每次网络接收一个字符，经过计算后会输出一个字符. 在实际的计算过程中，每个字符以长度为 95 的一维向量表示，输出结果的长度为隐藏层的维数，接着在后面两层中对隐藏层的输出做处理. 全连接层用来将隐藏层的输出

长度转换为预期长度.

期望输出向量的长度为 95，这里采用 linear 函数进行转换，linear 函数的计算如下：

$$y=\boldsymbol{W}\boldsymbol{A}^{\mathrm{T}}+b \tag{1}$$

其中 $\boldsymbol{A}$ 表示输入，W 为权重，b 为偏置，y 表示输出. 该函数通过矩阵乘法进行矩阵大小的转换，在训练过程会对这些参数进行更新.

在全连接层得到长度为 95 的输出结果后，采用 softmax 函数对其做最后处理，使输出向量表示各个字符的概率. softmax 函数的数学运算如下：

$$\mathrm{softmax}(x_i)=\frac{e^{x_i}}{\sum_{j=1}^{n}e^{x_j}} \tag{2}$$

在进行处理后，用输出结果向量中的第 n 个数字表示第 n 个字符的概率. 假设字符集为(a,b,c,d)，一个结果向量为(0.1,0.2,0.3,0.4)，那么网络预测的下一个字符为 a 的概率为 0.1，为 b 的概率为 0.2，为 c 的概率为 0.3，为 d 的概率为 0.4.

3.3　网络训练

如 3.2 节中描述，期望所设计的网络可以预测一个字符串的下一个字符. 假设网络的输入依次为(B,h,e,l,l,o)，那么网络的输出依次为(h,e,l,l,o,E)，其中 B 为起始符，E 为终结符. 对于一个长度为 n 的口令，在口令的头部加上起始符作为网络的输入，在口令的尾部加上终结符作为口令的标签，这个标签将用于和网络输出结果进行比较得出损失. 对于标签中的字符，首先将字符用数字 0～94 标记，然后将标签中的字符转换为相应的数字. 可以把数字理解为字符的类别，一共有 95 种字符，标签中的每个数字则对应其中的一种.

输入经过网络得到输出后，将结果与标签比较，通过交叉熵(cross entropy)计算损失. 交叉熵损失的表达式可以描述为

$$loss(\boldsymbol{x},i)=-\log\left(\frac{e^{x_i}}{\sum_{j=1}^{N}e^{x_j}}\right) \tag{3}$$

其中 $\boldsymbol{x}$ 为结果向量，i 为标签索引. 依然以输入(h,e,l,l,o)为例，网络读入 h 时，假设输出的结果向量为 $\boldsymbol{x}=(0.1,0.2,0.3,0.3,0.1)$，此时标签为 e，也就是 i 为 1. 将 $\boldsymbol{x}$ 和 i 代入上述公式即可得到损失.

在得到损失后，采用随机梯度下降(Stochastic Gradient Descent，SGD)对网络的参数进行优化. 这里举一个简单的例子对 SGD 进行说明. 假设网络要拟合的函数为一条直线：

$$h(k)=kx \tag{4}$$

其中 k 为斜率，即需要优化的参数. 假设有直线上的一个点(1,2)作为训练数据，损失函数为

$$J(k)=(h_k(x)-y)^2 \tag{5}$$

其中 x 和 y 分别为训练数据中点的横纵坐标，$h_k(x)$ 为网络的输出. 目的是让损失函数 $J(k)$ 的值最小，根据梯度下降法，首先要用 $J(k)$ 对 k 求偏导：

$$\frac{\partial J(k)}{\partial k}=2x(h_k(x)-y) \tag{6}$$

将 x 和 y 的值代入，就可以求出梯度的数值. 由于目标是要使损失函数最小化，所以参数 k 按照其负梯度的方向进行更新(这里不妨设学习率为 1)，即：

$$k'=k-\frac{\partial J(k)}{k}=k-2x(h_k(x)-y) \tag{7}$$

这样，就可以使得网络的参数按照指定的方向进行更新. 随机梯度下降算法是基于梯度下降法的，它的特点是每次迭代更新网络参数时，随机采用输入样本中的一组，而不是考虑输入的所有样本. 仍以输入(h,e,l,l,o)为例，输入包括五个字符，即五组样本，那么在迭代时，会从这五组中随机选择一组作为参数更新的依据. 在本文的网络中，输入层不执行运算，softmax 层是固定的函数运算，需要进行参数优化的是 RNN 层和全连接层.

3.4　样本生成

采用算法 1 来进行独立口令样本的生成. 算法的输入为概率阈值 *threshold*，输出为猜测集 *guess_set*. 维护的数据结构有：*prefixes*，它保存当前已生成的口令前缀；LUT(Look-Up Table)，即查找表，是当前已生成的口令前缀到概率的映射. 利用已训练完成的神经网络，将口令前缀作为输入，即可得到下一个字符的概率分布 *next_char_prob*，把字符添加到口令前缀后端便可以生成新的口令前缀，如果新生成的口令前缀概率大于 *threshold*，则加入到 *prefixes* 中，用于后续的口令生成，否则便舍弃. 如果添加的字符为终结符则说明生成了一个新口令，将其添加到 *guess_set* 中. 算法运行的初始阶段，*prefixes* 中仅包含一个口令前缀，其内容为起始字符，对应的概率为 1. 在执行的过程中，会不断的新增和消耗 *prefixes* 中的口令前缀. 算法运行到最后，*prefixes* 将变为空，此时 *guess_set* 中包含了所有概率大于 *threshold* 的口令.

算法 1.　口令生成算法.

```
输入：threshold 概率阈值
输出：guess_set 猜测集
FUNCTION GUESS(threshold)
  char_set={PrintableCharacters}∪
  {BeginCharacter,EndCharacter}
  Prefixes.push(BeginCharacter.to_string())
  LUT[Beginharacter.to_string()]=1
```

```
WHILE prefixes not empty DO
  content_prefix=prefixes.pop()
  next_char_prob=NeuralNetwork(current_prefix)
  FOR c in char_set DO
    next_prob=LUT[current_prefix]*
              next_char_prob[c]
    IF next_prob>threshold THEN
      IF c==EndCharacter THEN
        guess_set.append(current_prefix)
      END IF
    ELSE
      IF c≠BeginCharacter THEN
        prefixes.push(current_prefix+c)
        LUT[current_prefix+c]=new_prob
      END IF
    END IF
  END FOR
END WHILE
RETURN guess_set
END FUNCTION
```

3.5 基于 LSTM 的口令猜测模型

长短期记忆网络(Long Short-Term Memory, LSTM)[33]是一种特殊的 RNN,其输入输出与 RNN 相同.同样可以使用在 RNN 模型中描述的训练方法训练网络,以及样本生成算法来生成口令样本.这里只对结构单元的不同之处进行说明.

LSTM 模型同样包含输入层、隐藏层、全连接层和 softmax 层.与 RNN 不同之处在于,其隐藏层采用基于 LSTM 单元的网络结构.LSTM 可以依靠门结构进行删除或者增加信息,门是由 sigmoid 层和点乘单元组成.sigmoid 层的输出介于 0 和 1 之间,0 表示信息完全不通过,1 表示信息全部通过.

LSTM 单元通过遗忘门、输入门和输出门来决定下一个单元状态以及输出.假设上一时刻的单元状态和输出分别为 C_{t-1} 和 h_{t-1},则当前时刻的单元状态和输出分别为 C_t 和 h_t,输入为 x_t,对应权重和偏置分别用 W 和 b 表示.LSTM 的第一步是决定删除哪些信息,该过程在遗忘门中进行,运算如下:

$$f_t=\sigma(W_f*[h_{t-1},x_t]+b_f) \tag{8}$$

f_t是一个介于 0 和 1 之间的值,它会作用于 C_{t-1},1 表示完全保留,0 表示完全遗忘,σ 表示激活函数.

下一步决定哪些新的信息将被加入到单元状态中.该步骤由两部分构成,一是由 sigmoid 层构成的输入门,它用来决定哪些值要更新,二是 tanh 层构成的新候选值的向量生成器,它用于创造新的候选值向量,具体计算过程如下:

$$i_t=\sigma(W_i*[h_{t-1},x_t]+b_i) \tag{9}$$

$$\widetilde{C}_t=\tanh(W_C*[h_{t-1},x_t]+b_C) \tag{10}$$

$$C_t=f_t*C_{t-1}+i_t*\widetilde{C}_t \tag{11}$$

其中 i_t和 $\widetilde{C}_t$的乘积是需要新加入的信息.在决定了删除和增加的信息之后,将两者相加,就可以得出当前的单元状态,用 C_t表示.

最后决定要输出的信息,它是在 C_t的基础上,通过滤波得到,可描述为:首先使用 sigmoid 层决定输出的信息 o_t,同时将求出的 C_t输入到 tanh 层,然后乘 sigmoid 门的输出,即可得到最后的输出.

$$o_t=\sigma(W_o*[h_{t-1},x_t]+b_o) \tag{12}$$

$$h_t=o_t*\tanh(C_t) \tag{13}$$

其中 h_t为当前时刻产生的输出.

LSTM 单元这样的设计是为了解决传统 RNN 存在的长期依赖问题.但是对于口令数据来说,一个口令的长度最长一般不超过 16,并不会存在明显的长期依赖问题,所以由 RNN 和 LSTM 组成的模型在口令破解上的差异不大.

4 基于 PL 的口令猜测模型

PL 模型[30]是 PCFG[12]和 LSTM[23]的结合,该模型的核心思想是把口令按照 PCFG 中的规则分割成段的形式,再用 LSTM 对口令结构段进行训练,例如,口令“abcd123”对应结构段为 L_4D_3,网络的输入依次为(B,L_4,D_3),网络的输出依次为(L_4,D_3,E),其中 B 为起始符号,E 为终结符.随后通过训练好的 LSTM 生成结构化形式的口令(即 $L_3D_4S_1$这种形式),最后对结构化形式的口令中的段进行填充生成口令.如图 3 所示,为 PL 模型示意图.

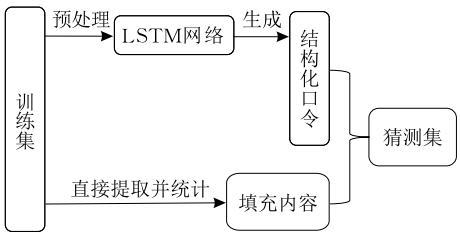


图 3 PL 模型

4.1 数据预处理

在数据的预处理阶段,基于 PCFG 的规则,把口令划分成为字母段 L、数字段 D 和特殊字符段 S,比如口令“abc1234!”按照规则就会被划分成为 $L_3D_4S_1$ 的结构.因为预处理后的口令需要经过神经网络训练,所以需要对其进行编码.对于预处理后的口令,可以将其看成是段的序列,因为不同类型的段之间的关系是相对独立的,所以本文依然采取独热编码的方式来对段进行编码.假设需要被编码的段

有(L_1,D_1,S_1),那么编码后 L_1 的向量形式为[1,0,0],D_1 的向量形式为[0,1,0],S_1 的向量形式为[0,0,1],L_1D_1 的矩阵形式为[[1,0,0],[0,1,0]].

由于绝大多数的口令长度都小于 16,因此本文只对长度不大于 16 的段进行编码,为了便于后续生成结构化形式的口令以及概率的归一化,还加入了起始段和终结段.按照以上描述,共对 50 个段进行编码,其中 L_1～L_{16} 占第 1 维到第 16 维,D_1～D_{16} 占第 17 维到第 32 维,S_1～S_{16} 占第 33 维到第 48 维,起始段占第 49 维,终结段占第 50 维.

4.2　网络结构、训练和结构化口令生成

在 RNN[31] 和 LSTM[23] 模型中,神经网络的训练输入和最终输出都是字符序列,在 PL 模型中,神经网络训练输入和最终输出的都是段序列,而在经过编码后,段序列和字符序列都使用向量表示.因此,在 PL 模型中,神经网络的训练方式、生成方式和网络结构与第 3 节介绍的相同.

由于在生成最终猜测的口令时,需要使用结构化口令的概率,所以在结构化口令的生成阶段,除结构化口令外,还要把计算后的概率输出.值得注意的是,在结构段生成过程中,有小概率会生成类似"D_4D_4"的结构,只需在结构段生成后设置对应的条件,进行简单的过滤即可.

在生成结构化口令及其概率后,只需对其中的段进行内容填充即可生成口令.在 PL 模型中,填充的内容是从训练集中提取的,对应的概率通过统计得出,比如口令"abcd123"出现在训练集中,那么"abcd"用于填充 L_4 段,"123"用于填充 D_3 段,在计算概率前,需要统计出段出现的次数和填充内容出现的次数,进而计算出对应的概率.

按照上述方式获得了填充内容及其概率之后,便可以开始生成口令.需要维护的数据结构为 *segs_list*,它是结构化口令的列表,由神经网络生成的结构化口令进行初始化.从 *segs_list* 中取出一个段序列 *segs*,然后获得其下一个需要填充的段 *segment*,再从基于训练集提取统计的 *contents_list* 来获得可以填充当前 *segment* 的内容列表 *contents*,最后再遍历 *contents* 中的内容进行填充,直到遇到终结段,详细过程如算法 2 所示.

算法 2.　口令生成算法.

```
输入:threshold 概率阈值
输出:gues_set 猜测集
FUNCTION GUESS(threshold)
  segs_list=NeuralNetworkGenerate()
  contents_list=PCFG(train_set)
  WHILE segs_list not empty DO
    segs=segs_list.pop()
    segment=segs.nextseg()
    IF segment==EndSegment THEN
      guess_set.append(segs.content)
    ELSE
      contents=contents_list[segment]
      FOR content in contents DO
        current_prob=segs.prob*content.prob
        IF current_prob>threshold THEN
          new_segs=segs.copy()
          new_segs.content+=content
          new_segs.prob=current_prob
          segs_list.push(new_segs)
        END IF
      END FOR
    END IF
  END WHILE
  RETURN guess_set
END FUNCTION
```

4.3　PR 模型

我们提出的 PR 模型是 PCFG[12] 和 RNN[31] 的结合.LSTM 通过增加遗忘门、输入门和输出门来解决长期依赖问题,但相应地也增加了网络的复杂度,需要进行更多的计算.在现实情况中,用户设置的口令绝大多数都小于 16.在口令长度本身就不长的条件下,将口令按照 PCFG 规则处理后生成的段序列只会更短.例如,口令"abcdefg123456789"本身长度为 16 位,对应结构段为 L_7D_9,长度仅为 2.经统计,4 个数据集结构段序列的平均长度均小于 2.以泄露的 CSDN 用户口令为例,98.8%的口令预处理成段序列后,其长度不超过 4.

基于上述分析,可以认为在对结构化口令的训练中不会出现长期依赖问题.本文将通过实验证明,由 RNN 构成的 PR 模型可以在绝大多数数据集上取得略优于 PL 模型的口令破解效果,同时训练效率远远优于后者.因此将 PL 模型中的 LSTM 替换成普通 RNN,即变成 PR 模型.PR 模型相比于 PL 模型只改变了神经网络,其训练过程和最终口令的生成方式与 PL 模型没有区别,因此在这里不再进行详细介绍.

4.4　PR+模型

在 PCFG[12]、PR 和 PL[30] 模型中,当模型处于最后的口令生成阶段时,向段中填充的内容将从训练集中提取,而训练集是有限的.因此,对于一个部分内容不存在于训练集中的口令,PCFG、PL 和 PR 模型是不可能生成的.为了解决这个缺陷,本文在 PR 模型的基础上提出了 PR+模型.

在 PR+模型中，向段中填充的内容是通过神经网络生成的.实验过程中，通过从训练集中提取出口令字母段的内容，让神经网络学习这些内容的特征，进而生成出更加符合真实分布的填充内容.

如图 4 所示，训练和生成填充内容是通过 RNN 完成.因为填充的内容属于口令的字母段，和完整口令的形式类似，所以对填充内容的处理可以使用基于 RNN 的口令猜测模型.

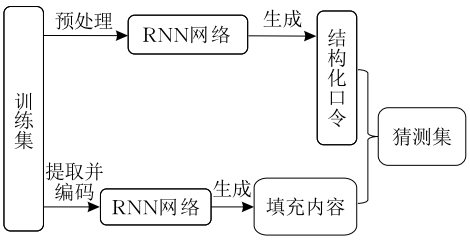


图 4 PR+模型架构

在最后的口令生成阶段，和 PL 模型相同，同样需要使用算法 2，依靠结构化口令和填充内容来生成最终用于猜测的口令即可.

4.5 实验结果与分析

对于 RNN[31]、LSTM[23]、PL[30]、PR 和 PR+模型，网络的层数为 2，隐藏层的神经元个数为 64，训练时的学习率为 0.005.模型训练完成后，将概率阈值设置为 1×10^{-9}，生成概率大于该阈值的口令.

使用相同的训练集对 PCFG 模型以及 Markov 模型进行训练.其中 Markov 模型采用效果最好的四阶模型，并采用 Laplace 平滑和 End-Symbol 正规化技术.对于 PCFG、Markov、PL、PR 和 PR+模型，均生成 1×10^{8} 个口令.取五个模型生成的口令中概率最高的前 5×10^{7} 个口令，在同一测试集上进行测试，结果如表 2 所示(以 CSDN 数据集为例).

表 2 各口令猜测模型破解率对比(基于 CSDN 数据集)

模型	测试集	猜测集	破解率/%
RNN[22]	6.4×10^{5}	5×10^{7}	40.34
LSTM[23]	6.4×10^{5}	5×10^{7}	38.73
PCFG[12]	6.4×10^{5}	5×10^{7}	40.90
Markov[11]	6.4×10^{5}	5×10^{7}	40.81
PL[30]	6.4×10^{5}	5×10^{7}	**42.48**
PR	6.4×10^{5}	5×10^{7}	**42.66**
PR+	6.4×10^{5}	5×10^{7}	40.90

其中，PR 模型与 PL 模型破解效果最好，比 PCFG 和 Markov 模型多破解出约 1100 条口令.相比于传统的口令破解算法(PCFG 和 Markov)，将深度学习与传统方法结合的模型(PR 和 PL)表现出了一定的优势，这说明将深度学习应用在口令破解上是可行且有效的.为进一步探究基于深度学习模型在口令破解方面的效果，本文在不同猜测集大小下测试了不同模型的破解率，结果如图 5 所示.

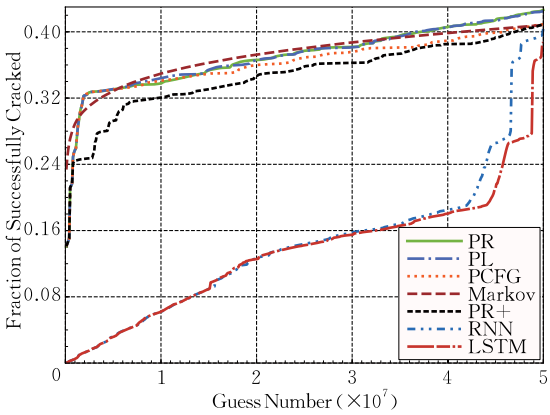


图 5 各模型破解率比较

从图 5 可以看出，在破解率的增长上，五个模型(除 RNN、LSTM 之外)都有先迅速增长再缓慢增长的趋势，在猜测集大小小于 5×10^{6} 时，破解率几乎直线上升，但随后迅速放缓.对于 RNN 和 LSTM，虽然在猜测集较小时的破解率不如传统方法，但是可以看出在猜测集大小到达 5×10^{7} 时，依然保持了比较可观的增长率.

从结果来看，在猜测集大小小于 5×10^{7} 时，PCFG、Markov、PL、PR 和 PR+模型都表现出了优势.但是从破解率的增长率来看，这五个模型和 RNN 以及 LSTM 之间的差距在不断缩小.可以预计，在猜测集足够大后，RNN 模型和 LSTM 模型会明显优于传统方法.

PR 模型相比于 PCFG，其对于结构化口令特征的提取上有优势，神经网络可以更好模拟结构化口令的分布，因此在破解效果上有着更好的表现.

PCFG 模型和 Markov 模型是基于严密的统计学理论进行设计的，因此可以对其结果进行科学的分析且具有良好的可解释性.此类方法通常依赖于人在设置口令时具有的一些可统计的共性，比如使用英文单词后加“123”这样的口令构造方式.但是到目前为止，还没有比较严谨的分析深度学习神经网络行为的方法.本文进行了一些探索性实验来对网络的行为进行分析，如果在网络中输入“boo”，深度学习模型(这里特指 RNN 和 LSTM)预测下一个字符为“k”的概率非常高，这点和 Markov 模型具有一定相似性.

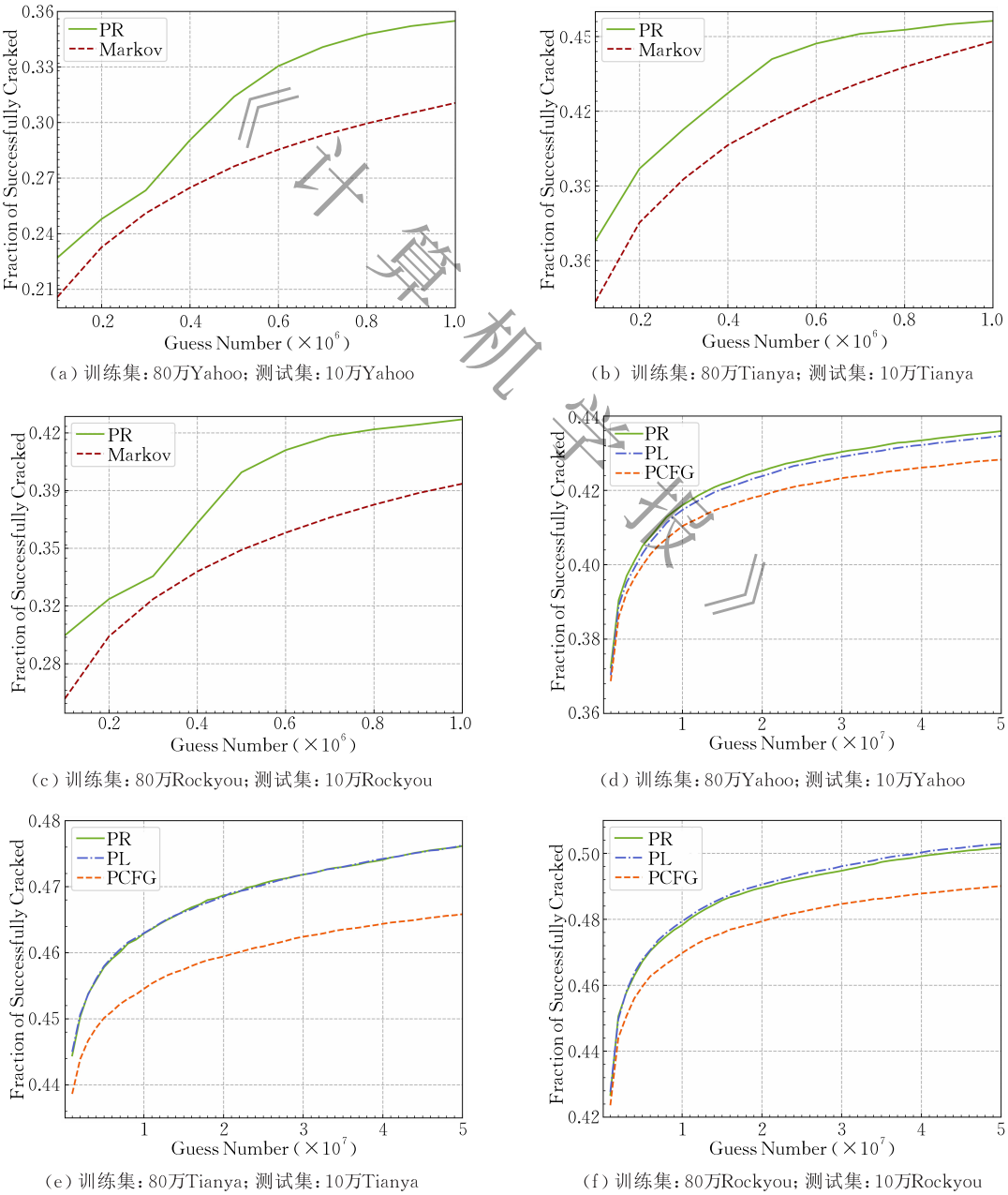
同时，本文还发现了 RNN 和 LSTM 生成的口令中有大量类似于“zzh19950201”这样，中文名缩写加生日的口令结构，这点和 PCFG 模型具有一定相似性.在分析了深度学习模型生成的具体口令结构后，可以发现深度学习模型同时具有 PCFG 模型和 Markov 模型的特性.而在口令集中，统计学可统计的共性应当不止这两种，因此，我们有理由相信深度

学习模型同样可以学习到口令的其他特性，即深度学习模型在生成口令的多样性上是优于传统方法的.

如果仅对比 RNN 模型和 LSTM 模型，从图中可以看出，两个深度学习模型破解率的增长曲线非常相似. 虽然 RNN 模型的破解率略高于 LSTM 模型，但是两者的差距并不明显.

4.6　PR 模型与其他模型对比分析

为了进一步验证不同模型的猜测能力，本文分别从 Tianya、Rockyou 以及 Yahoo 数据集中随机选取 100 万数据进行相关实验，训练集与测试集的划分比例与 CSDN 数据集相同（由于 RNN 模型[31]与 LSTM 模型[23]的生成时间远远大于其它方法，且在 10^7 数量级的猜测数下猜测效果显著低于其他方法，因而将这两种方法排除在外）. 实验结果表明，PR 模型的破解率在小猜测数下（$<10^6$ 量级）显著高于 Markov 模型[5]，与 PL 模型[30]破解率相当，具体来说，PR 在 CSDN、Yahoo 数据集上破解率略高于 PL，在 Tianya 数据集上破解率与 PL 相当，PL 模型的破解率仅在 Rockyou 数据集上略高于 PR 模型. 为此，我们统计了所有数据集结构段长度的变化情况，发现 Rockyou 数据集方差最小，Tianya 数据集次之，因此，我们推测：结构段长度变化较大时，PR 模型能更好的学习到数据的变化规律. 此外，PR 模型与 PL 模型的破解率均高于 PCFG 模型（10^7 量级猜测数）.



（a）训练集：80万Yahoo；测试集：10万Yahoo

（b）训练集：80万Tianya；测试集：10万Tianya

（c）训练集：80万Rockyou；测试集：10万Rockyou

（d）训练集：80万Yahoo；测试集：10万Yahoo

（e）训练集：80万Tianya；测试集：10万Tianya

（f）训练集：80万Rockyou；测试集：10万Rockyou

图 6　不同模型在小猜测数和大猜测次数下的破解率对比

如图 6 所示，(a)～(c)为 PR 模型与 Markov 模型在小猜测数下的破解率对比图，(d)～(f)为 PR 模型与 PL 模型以及 PCFG 模型在较大猜测数下的破解率对比图.

对于 Markov 模型，主要存在以下两个问题：(1)概率的计算很大程度上反映的是训练集的特征，而不是字符串本身的特征，因而在阶数过大或训练集过小时容易出现过拟合的问题；(2)平滑技术的引入并不能完全解决频数为 0 的问题，平滑后概率的计算仍然反映的是训练集的特征，而不是频数为 0 的字符串本身的特征.

对于 PCFG，它与 PR 模型的区别在于 PR 模型中结构化口令是通过 RNN 生成的. 实验结果表明，PR 模型在 10^6 猜测数以后，破解率稳定高于 PCFG 模型.

如表 3 所示，为两模型概率最高的 10 个结构化口令. 通过结构化口令的对比可以发现，两个模型中的结构化口令 top10 的组成几乎是相同的，但概率分布有差异，即通过 RNN 生成的结构化口令的概率分布比通过统计得到的概率分布更加合理. 对于 PL 模型，其训练效率远劣于 PR 模型，这是 RNN 与 LSTM 单元在结构组成上的差异所决定的. 图 7 对比了 RNN 与 LSTM 内部结构. 可以看出，RNN 单元结构较为简单，训练参数少，而 LSTM 单元相比于 RNN，增加了遗忘门、输入门和输出门等结构来改善长期依赖问题，但在口令经过结构化处理后，长度较短，因而该问题并不存在.

表 3 结构化口令 Top10 比较(基于 CSDN 数据集)

排名	PR	PCFG
1	D_8(0.2408)	D_8(0.2155)
2	D_9(0.1273)	D_9(0.1122)
3	L_8(0.0752)	L_8(0.0488)
4	D_{11}(0.0555)	D_{11}(0.0428)
5	D_{10}(0.0423)	D_{10}(0.0374)
6	L_9(0.0298)	L_3D_6(0.0293)
7	L_3D_6(0.0241)	L_9(0.0266)
8	D_{12}(0.0215)	L_2D_6(0.0219)
9	L_{10}(0.0209)	L_{10}(0.0205)
10	L_3D_7(0.0153)	D_{12}(0.0165)

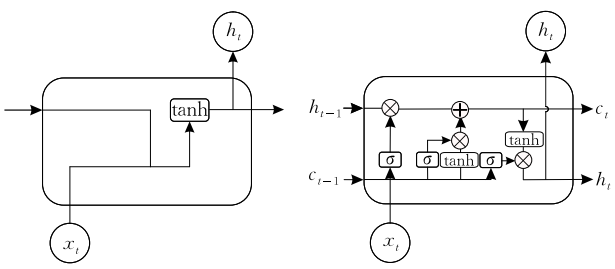


图 7 RNN 与 LSTM 内部结构对比图

对于本文所使用的网络结构，PR 模型需要训练的参数有 18994 个，而 PL 模型需要训练的参数有 66226 个，超出前者两倍以上；在训练时间上，PL 模型所需时间比 PR 模型平均多出 15%(在 CPU 为 Intel Core i77700HQ，GPU 为 NVIDA GeForce RTX 1070 环境下).

对于 PR＋模型，在多数数据集下，它的效果远不如 PR 模型. 我们以 CSDN 数据集为例，探究了这两者在不同概率阈值下的猜测集大小及猜测效果，如表 4 所示. 可以发现，在相同概率阈值下，PR 模型和 PR＋模型生成的猜测集大小是相近的，但 PR 模型的猜测效果始终优于 PR＋模型. 进一步观察发现，PR＋模型以 10^{-10} 为阈值的猜测效果和 PR 模型以 10^{-9} 为阈值的猜测效果相近，PR＋模型以 10^{-11} 为阈值的猜测效果和 PR 模型以 10^{-10} 为阈值的猜测效果相近，这是因为口令的概率计算与填充内容的概率相关，在猜测次数相对较大的情况下，主导口令生成的都是低概率填充内容. 以低概率填充内容“gjgty”为例，其在 PR 模型中的概率为 3.24×10^{-7}，而在 PR＋模型中的概率为 1.05×10^{-8}. PR＋模型中的 RNN 通过降低一部分填充内容的概率来泛化出一批新的填充内容. 因此，同一个口令在 PR＋模型中的概率低于在 PR 模型中的概率，造成攻破相同数量的口令，PR＋模型所需设置的概率阈值小于 PR 模型，即猜测数应大于 PR 模型. 此外，随着猜测数的增加，PR＋模型与 PR 模型的差距也逐渐变小. 由此我们推测，在足够大猜测数下，这种概率阈值不同所造成的差异将不复存在.

表 4 PR 与 PR＋模型对比(基于 CSDN 数据集)

模型	概率阈值	猜测数	破解率/%
PR	10^{-9}	20350821	42.28
	10^{-10}	230950044	46.92
	10^{-11}	2381390500	51.60
PR＋	10^{-9}	21201295	38.26
	10^{-10}	249345682	42.43
	10^{-11}	2433935699	48.34

考虑到生成较大猜测集所带来的空间存储问题(10^{16} 猜测集的大小约占 10^5 TB 存储空间)，借鉴研究[22,34-35]中的方法，我们采用蒙特卡洛模拟方法探究 PR 和 PR＋模型在大猜测数下的破解率，并且蒙特卡洛模拟方法的准确性已得到证明[35]. 如图 8 所示，为 PR 和 PR＋模型在大猜测数下(10^{25} 猜测量级)的对比图(仍以 CSDN 数据集为例)，实验结果表明，在大猜测数下，PR＋模型的猜测效果始终不低于 PR 模型.

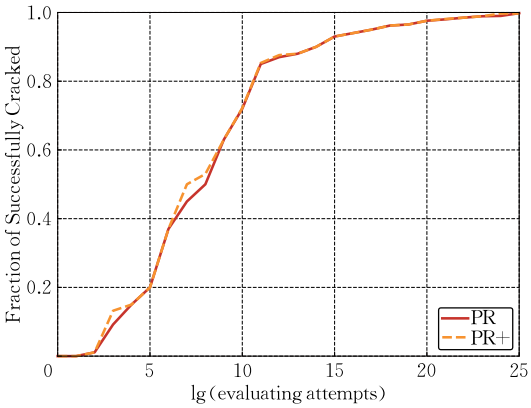


图 8　PR 和 PR+模型在大猜测数下的对比图

4.7　不同模型口令样本组合

在对不同模型生成的口令样本进行了分析后，发现不同模型生成口令样本的特性有所不同，所以在本节中，我们尝试对不同模型生成的口令样本进行组合，探究了“RNN+PCFG”、“RNN+Markov”、“RNN+PR”、“LSTM+PCFG”、“LSTM+Markov”和“LSTM+PR”这六种组合在同一测试集下的破解率. 以 CSDN 数据集为例，分别取组合中两种模型生成的 5×10^7 个口令，合并得到 1×10^8 大小的猜测集. 合并后的猜测集中可能存在重复口令，因此需要在去重后再进行测试. 各个模型在 CSDN 数据集下的详细实验结果如表 5 所示.

表 5　不同模型组合后的破解率比较(基于 CSDN 数据集)

模型	测试集	猜测集	破解率/%
RNN+PCFG	6.4×10^5	9.2×10^7	48.90
RNN+Markov	6.4×10^5	8.8×10^7	44.70
RNN+PR	6.4×10^5	9.2×10^7	49.40
LSTM+PCFG	6.4×10^5	9.3×10^7	49.10
LSTM+Markov	6.4×10^5	8.8×10^7	45.04
LSTM+PR	6.4×10^5	9.2×10^7	49.61
PR	6.4×10^5	1.0×10^8	44.55
PCFG[12]	6.4×10^5	1.0×10^8	44.84
Markov[11]	6.4×10^5	1.0×10^8	42.96

组合模型的优点在于，可以将具有不同特性的口令样本进行结合. 单一模型是很难覆盖所有口令类型的，虽然本次实验的模型组合也没有做到全部覆盖，但是这样的组合尝试是有效果的. 实验结果表明，组合不同类型模型的破解率显著高于单个模型，如表 5 中“LSTM+PR”的组合比相同猜测数下的 PR 高 5.06%. 同时从表 5 中还可以观察到，RNN 模型[31]和 PCFG 模型[12]的重合度以及 LSTM 模型[23]和 PCFG 模型[12]的重合度很低，仅有 4%，而和 Markov 模型[5]的重合度较高，为 24%. 这与模型生成口令的方式有关，基于 RNN(LSTM)的口令猜测模型和 Markov 模型在基本思想和生成样本的方式上具有相似性，都是考虑口令中字符的前后依赖关系，通过一定阶数的字符串来预测其之后的字符，这点在其它实验中也得到了印证，即“RNN(LSTM)+Markov”的组合方式与 Markov 模型所生成的猜测集在相同猜测数下($10^7\sim10^8$ 量级)猜测效果差别不大. RNN(LSTM)和 PR 的重合口令数为 8×10^6，这意味着深度学习模型具有一定模拟 PCFG 模型的能力，而与 Markov 模型的重合口令数更多，则表明了它对 Markov 模型的模拟能力更强. 我们在其它 3 个大规模真实口令集上(Rockyou、Yahoo 以及 Tianya 数据集)也进行了实验，得到了类似的结果，即相同猜测数下($10^7\sim10^8$ 量级)，将不同类型(不同类型是指基本思想和生成口令方式具有显著差异，如 RNN 与 PCFG)模型生成的猜测集组合，其破解率通常高于单一模型的猜测结果. 在研究组合模型的过程中，我们注意到，Parish 等人[36]也对不同模型口令样本的组合进行了研究，但该项研究与本文的切入角度不同. 在文献[36]中，作者发现，恒等猜测者(Identity Guesser，即直接将已有的口令数据集去重，再按频率进行排列)的猜测效果经常高于一些最新的口令猜测方法，因而作者以恒等猜测者为基线模型，尝试将不同模型生成的口令集与其组合，探究组合攻击的猜测效率与猜测效果. 本节以不同模型生成口令样本的特性不同为切入点，基于大规模真实口令数据的实验结果，首次证实了在相同猜测数下，将不同类型模型生成的猜测集组合，其破解率通常高于单一模型的猜测结果. 根据本节得到的结论，在进行组合时，应优先考虑将口令生成方式不同的模型所产生的猜测集进行组合，这在一定程度上能提高口令猜测效率.

本节与文献[36]都没有测试这几种组合在不同大小猜测集下的破解率. 这是由于不同模型对于生成口令概率的定义有差别，因而难以根据概率对猜测集进行排列，从而得到一定大小的合理猜测集. 另外，RNN 模型[31]与 LSTM 模型[23]在生成口令时，耗时较长. 具体来说，在 CPU 为 Intel Core i79700F，GPU 为 NVIDA GeForce RTX 2070 SUPER 的环境下，生成 5×10^7 个猜测，需要 72 h 左右. 因此，本文不再对这两个模型在 10^8 量级猜测数下进行相关实验. 在后续研究中，可以尝试继续改进此类模型的生成算法，提升口令生成效率，以探究它们在更大猜测数下的口令破解效果.

5 基于 GAN 的口令猜测模型

5.1 模型架构

基于生成式对抗网络(GAN)的口令猜测模型[25]学习整个字符串的分布,它由生成式模型 G 和判别式模型 D 组成. G 接受一个随机噪音向量作为输入,根据它所学习到的分布,将噪音向量转换为一个口令样本. 在训练完成后,只需要不断为 G 提供噪音向量,即可产生口令样本,直到口令样本集达到所需大小.

该模型的超参数包括:

批尺寸:每次优化中,通过网络的训练数据数目;

序列长度:输入序列的最大长度;

G 的神经元数目:G 的每层网络神经元的数目;

G 的神经网络层数:G 的网络层数;

D 的神经元数目:D 的每层网络神经元的数目;

D 的神经网络层数:D 的网络层数;

G 的神经网络层数:G 的网络层数;

D/G 训练次数比:每训练一次 G,训练 D 的次数.

具体参数如表 6 所示.

表 6 GAN 模型超参数

超参数	数值
批尺寸	64
序列长度	20
G 的神经元数目	128
D 的神经元数目	128
D 的神经网络层数	5
G 的神经网络层数	5
D/G 训练次数比	10

训练阶段,在每一轮迭代过程中,交替训练 1 次生成式模型 G 和 k 次判别式模型 D,其中 k 是超参数,即 D/G 训练次数比. 训练 G 时,其输入 $\boldsymbol{z}$ 是一个随机噪音向量,输出 $G(\boldsymbol{z};\phi_g)$ 是一个概率向量,该概率向量将直接作为 D 的输入. D 的输出是 $D(G(\boldsymbol{z};\phi_g);\phi_d)$,它是一个[0,1]范围内的标量,表示 D 的输入来自真实数据的概率. 如果 D 的输出为 1,则表示 D 的输入来自真实数据. 在优化 G 的时候,目标是使 G 的输出尽可能逼真,即期望目标函数 $\mathrm{Loss}(G)=-D(G(\boldsymbol{z};\phi_g);\phi_d)$ 尽可能的小.

训练 D 时,D 的一个输入是 $G(\boldsymbol{z};\phi_g)$,另一个输入是来自真实数据的样本 x_{real}. 在优化 D 的时候,目标是使 D 的判断尽可能准确,即期望目标函数 $\mathrm{Loss}(D)=D(G(\boldsymbol{z};\phi_g);\phi_d)-D(x_{\mathrm{real}};\phi_d)$ 尽可能小.

由于传统的 GAN 训练起来比较困难,Martin 等人[32]提出 WGAN,将判别式模型 D 的参数限制在一个范围内. Gulrajani 等人[27]在这个基础上提出 WGAN-GP,使用另一种方法限制了 D 的参数. 本文的模型就是基于 WGAN-GP 的,如图 9 所示,这里不再对其网络结构进行介绍.

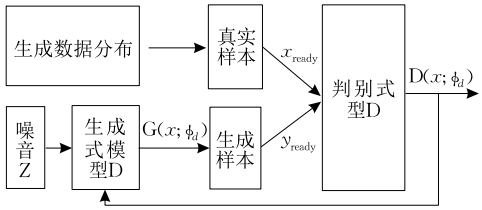


图 9 GAN 模型

5.2 样本生成

训练完成后,可以得到一个生成式模型 G 和一个判别式模型 D. 其中 G 可以从噪音向量出发模拟生成一个口令,D 可以判别一个字符串是否为真实的口令. 在样本生成时,只需要利用 G,输入一定数量的噪音向量,就可以得到一定数量的口令样本. 然而 GAN 模型在生成样本时存在两个问题:

(1) GAN 模型输出口令的长度是固定的,而我们期望生成的口令长度是变长的;

(2) 对于不同的输入,GAN 模型的输出有可能是相同的.

对于第一个问题,本文通过对输出进行剪切来解决. 在数据预处理的过程中,在每个口令后添加了若干终结符以保证定长的输入. 例如,对于口令“Hello”来说,假设模型设置的序列长度为 10,则在训练前将其补全为“Hello⊥⊥⊥⊥⊥”. 在生成的口令中,假设生成了两个口令,分别为“123456⊥⊥⊥⊥”和“1234567⊥⊥⊥”,则将口令中出现的第一个终结符之后的字符串剪切掉,即变为“123456”和“1234567”,从而使得定长的输出序列变为变长的口令.

对于第二个问题,通过生成尽可能多的口令,再做去重处理,以得到不包含重复口令的样本集.

5.3 实验结果与分析

首先探究不同网络层数对于 GAN 模型破解率的影响. 在 PassGAN[22]的网络结构中,G 和 D 都是包含 5 层残差块的卷积神经网络. 在本文的实验中,将残差块的层数分别设置为 5 层、6 层及 7 层,观察该参数对模型破解率的影响. 对于每种网络结构,分别生成了 10^8 个口令,具体结果如表 7 所示.

表 7　网络层数对 GAN 模型破解率的影响(CSDN 数据集)

网络层数	测试集	猜测集	独立口令	破解率/%
5 层	6.4×10^5	1×10^8	69 788 687	**26.31**
6 层	6.4×10^5	1×10^8	69 697 680	26.23
7 层	6.4×10^5	1×10^8	66 662 809	25.33

从表 7 可以看出，在生成同样大小猜测集的情况下，随着网络层数的增加，GAN 模型破解率逐渐下降，同时，模型生成独立口令的能力也逐渐下降. 另外，同 4.5 节中几种口令破解模型(PCFG 模型，Markov 模型，RNN 模型，LSTM 模型，PR 模型，PL 模型等)相比，在相同猜测数时，GAN 模型的破解率是最低的.

图 10 为不同网络层数在不同猜测数下的破解率比较图，本文采用 5 层网络结构的 GAN 模型尝试生成更多的口令，探究其在较大猜测数下的破解率，具体结果如表 8 所示.

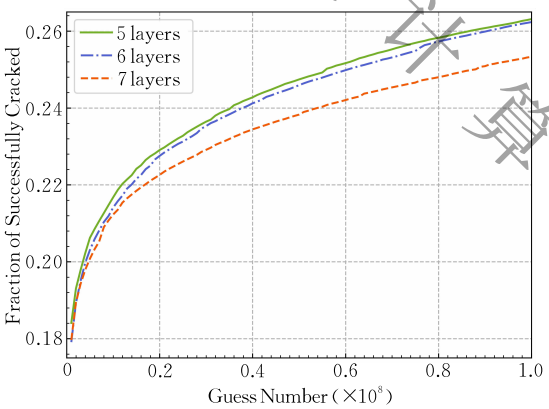


图 10　不同网络层数 GAN 模型破解率比较

表 8　GAN 模型的破解率(CSDN 数据集)

测试集	猜测集	独立口令	破解率/%
6.4×10^5	1×10^8	69 788 687	26.31
6.4×10^5	2×10^8	132 923 038	28.16
6.4×10^5	3×10^8	192 983 042	29.31
6.4×10^5	4×10^8	250 969 594	30.18
6.4×10^5	5×10^8	307 388 166	30.84
6.4×10^5	6×10^8	362 513 849	**31.41**

图 11 为破解率的变化趋势(猜测集中包含重复口令)，可以看出随着猜测数的增长，GAN 模型的破解率具有明显的提高，但是增长率逐渐变缓. 按照这个趋势，即使猜测数很大，破解率也很难达到 40%. 这表明 GAN 模型在口令破解上具有一定效果，但不如传统方法和基于 RNN 的模型.

对于 GAN 模型表现不佳的问题，本文做了以下分析：

(1) GAN 模型不同于概率模型，它无法给出生成口令的概率，也就无法将猜测集按照概率大小进行排列. 一般认为，以较大概率生成的口令更容易命中测试集，所以在猜测数较小时，如果采用按照概率降序排列的猜测集，效果要更好.

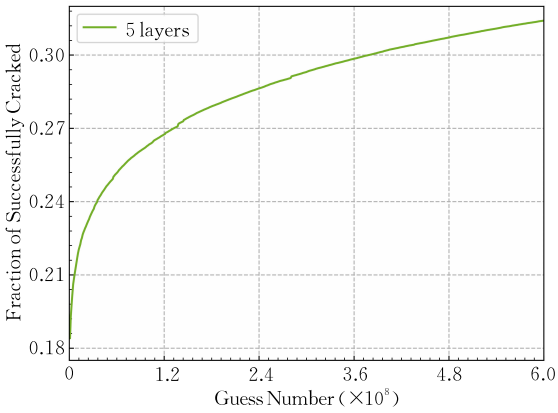


图 11　GAN 模型的破解率

(2) GAN 模型大多用于图像生成. 对于图像生成，其结果并不需要非常精细，一些偏差并不会明显影响最终的结果. 但是在口令破解上，如果存在偏差，其生成的口令就无法和测试集匹配.

在开展本研究的同时，我们注意到 Pasquini 等人[37]对 PassGAN 进行了改进. 在文献[37]中，作者通过对口令字符独热编码时加入噪声，再将添加噪声后的字符分布归一化的方式来改善 GAN 在训练时的模式坍塌问题[38]，使得 GAN 在大猜测数下(10^{10}量级)的破解率得到了显著提升，但该研究仍没有从根本上解决 GAN 模型存在的上述两个局限性.

总之，将 GAN 模型直接应用在口令猜测领域，其破解效果在目前仍劣于 PCFG、Markov、RNN 等基线模型，在未来我们将尝试解决上述两个问题.

6　结束语

本文分别重现了 RNN 模型[31]、LSTM 模型[23]、PL 模型[30]和 GAN 模型[25]，提出了 PR 模型和 PR+模型. 其中 PR 模型破解率要高于传统的 PCFG 模型[12](10^7量级猜测数)和 Markov 模型[5](10^6量级猜测数). 同时，本文还探究了不同模型进行组合之后的破解率，在相同猜测数下，组合不同类型模型的破解率通常高于单一模型. 在 GAN 模型破解率的探究实验中，本文首先研究了网络层数对破解率的影响，发现层数较深时，破解率会略有降低；然后，基于 GAN 模型生成了大规模的口令猜测，在猜测集

大小为 3.6×10^8 时(独立口令数量),GAN 模型在 CSDN 测试集上的破解率仅为 31.41%.在未来的工作中,有以下几点可以改进或者进行进一步探索:

(1)对于基于 RNN 的口令破解模型,可以尝试不同的训练方法和口令生成算法.当前设计的算法与 Markov 模型具有一定的相似性,转变思路或许可以取得更好的结果,有待进一步研究.

(2)GAN 模型破解口令的效率问题有待进一步改善.从实验中可以看出,GAN 模型在口令破解上具有很大的潜力,但和传统方法及 RNN 模型相比,在猜测数较小时,会有明显的劣势.

(3)GAN 的提出虽然仅有几年时间,但是已经有了非常多的变种,在后续研究中,可以尝试使用不同的 GAN 结构来进行模型设计和实验,如采用文献[28-29]中的结构,将 G 和 D 的组成单元替换为 RNN 单元及其变种,并在此基础上,尝试添加卷积层以更好地捕获口令的局部特征.

(4)在 PR 和 PR+模型中口令的划分上,目前口令的分段方式是依据 PCFG 规则,这种方式略显粗糙.另一方面,随着大数据时代的到来,用户的个人信息在不断泄露,因而在后续研究中,可以融入个人信息,对口令的分段进一步细化.比如数字段可以划分出生日和手机号码,而生日内部又可以进一步划分为年月日,字母段可以划分出完整的姓名和姓名的首字母等.也可采用文献[19]中的方式,将个人信息以类型为标准进行划分,再利用 PR 或 PR+模型进行定向口令猜测.对于长度较长且包含个人信息的口令,尝试在 TransPCFG 框架[15]的基础上,再对个人信息进行划分.

本文的实验结果与分析充分说明了将深度学习应用在口令破解上是可行的,值得进一步研究.

参 考 文 献

[1] Mark K, Benjamin S, Paul J S. The usability of passphrases for authentication: An empirical field study. International Journal of Human-Computer Studies, 2007, 65(1): 17-28

[2] Bonneau J. The science of guessing: Analyzing an anonymized corpus of 70 million passwords//Proceedings of the IEEE Symposium on Security and Privacy. Washington, USA, 2012: 538-552

[3] Das A, Bonneau J, Caesar M, et al. The tangled Web of password reuse//Proceedings of the Network and Distributed System Security Symposium. San Diego, USA, 2014: 23-26

[4] Dell'Amico M, Michiardi P, Roudier Y. Password strength: An empirical analysis//Proceedings of the IEEE International Conference on Computer Communications. Cape Town, South Africa, 2010: 1-9

[5] Ma J, Yang W, Luo M, et al. A study of probabilistic password models//Proceedings of the IEEE Symposium on Security and Privacy. Oakland, USA, 2014: 689-704

[6] Troy H. Here's Why [Insert Thing Here] Is Not a Password Killer. https://www.troyhunt.com/heres-why-insert-thing-here-is-not-a-password-killer/2018, 11, 5

[7] Chatterjee R, Athayle A, Akhawe D, et al. pASSWORD tYPOS and how to correct them securely//Proceedings of the 2016 IEEE Symposium on Security and Privacy. Oakland, USA, 2016: 799-818

[8] Wang Ping, Wang Ding, Huang Xin-Yi. Advances in password security. Journal of Computer Research and Development, 2016, 53(10): 2173-2188(in Chinese)
(王平,汪定,黄欣沂. 口令安全研究进展. 计算机研究与发展, 2016, 53(10): 2173-2188)

[9] HashCat. https://hashcat.net/2020, 7, 29

[10] John the Ripper. http://www.openwall.com/john/2019, 5, 14

[11] Narayanan A, Shmatikov V. Fast dictionary attacks on passwords using time-space tradeoff//Proceedings of the 12th ACM Conference on Computer and Communications Security. Alexandria, USA, 2005: 364-372

[12] Weir M, Aggarwal S, De Medeiros B, et al. Password cracking using probabilistic context-free grammars//Proceedings of the 30th IEEE Symposium on Security and Privacy. Oakland, USA, 2009: 91-105

[13] Dürmuth M, Angelstorf F, Castelluccia C, et al. OMEN: Faster password guessing using an ordered Markov enumerator//Proceedings of the International Symposium on Engineering Secure Software and Systems. Milan, Italy, 2015: 119-132

[14] Houshmand S, Aggarwal S, Flood R. Next gen PCFG password cracking. IEEE Transactions on Information Forensics and Security, 2015, 10(8): 1776-1791

[15] Han Wei-Li, Xu M, Zhang Jun-Jie, et al. TransPCFG: Transferring the grammars from short passwords to guess long passwords effectively. IEEE Transactions on Information Forensics and Security, 2021, 16: 451-465

[16] Zhang Meng-Li, Zhang Qi-Hui, Liu Wen-Fen, et al. A method of password attack based on structure partition and string reorganization. Chinese Journal of Computers, 2019, 42(4): 913-928(in Chinese)
(章梦礼,张启慧,刘文芬等. 一种基于结构划分及字符串重组的口令攻击方法. 计算机学报, 2019, 42(4): 913-928)

[17] Wang D, Wang P. The Emperor's new password creation policies: An evaluation of leading Web services and the effect of role in resisting against online guessing//Proceedings of the 20th European Symposium on Research in Computer Security. Vienna, Austria, 2015: 456-477

[18] Li Y, Wang H, Sun K. A study of personal information in human-chosen passwords and its security implications//Proceedings of the IEEE International Conference on Computer Communications. San Francisco, USA, 2016: 1-9

[19] Wang D, Zhang Z, Wang P, et al. Targeted online password guessing: An underestimated threat//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Vienna, Austria, 2016: 1242-1254

[20] Wang D, Wang P. On the implications of Zipf's law in passwords//Proceedings of the European Symposium on Research in Computer Security. Heraklion, Greece, 2016: 111-131

[21] Wang D, Cheng H, Wang P, et al. Zipf's law in passwords. IEEE Transactions on Information Forensics and Security, 2017, 12(11): 2776-2791

[22] Melicher W, Ur B, Segreti S M, et al. Fast, lean, and accurate: Modeling password guessability using neural networks//Proceedings of the 25th USENIX Security Symposium. Austin, USA, 2016: 175-191

[23] Xu L, Ge C, Qiu W, et al. Password guessing based on LSTM recurrent neural networks//Proceedings of the IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC). Guangzhou, China, 2017: 785-788

[24] Zhou Huan, Liu Qi-Xu, Cui Xiang, et al. Research on targeted password guessing using neural networks. Journal of Cyber Security, 2018, 3(5): 25-37(in Chinese)
(周环，刘奇旭，崔翔等. 基于神经网络的定向口令猜测研究. 信息安全学报，2018，3(5)：25-37)

[25] Hitaj B, Gasti P, Ateniese G, et al. PassGAN: A deep learning approach for password guessing//Proceedings of the International Conference on Applied Cryptography and Network Security. Bogotá, Colombia, 2019: 217-237

[26] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets//Proceedings of the Advances in Neural Information Processing Systems 27. Montréal, Canada, 2014: 2672-2680

[27] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved training of Wasserstein GANs//Proceedings of the Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 5767-5777

[28] Rajeswar S, Subramanian S, Dutil F, et al. Adversarial generation of natural language. https://arxiv.org/abs/1705.10929/ 2017, 5, 31

[29] Liu Gong-Shen, Wang Jing-Kang, Luo Yu-Tao, et al. Data mining and password generation based on large-scale real plain passwords. Journal on Communications, 2018, 39(Z2): 217-226(in Chinese)
(刘功申，王靖康，罗宇韬等. 基于海量真实口令的挖掘分析及口令生成. 通信学报，2018，39(Z2)：217-226)

[30] Liu Y, Xia Z, Yi P, et al. GENPass: A general deep learning model for password guessing with PCFG rules and adversarial generation//Proceedings of the IEEE International Conference on Communications (ICC). Missouri, USA, 2018: 1-6

[31] Schuster M, Paliwal K. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 1997, 45(11): 2673-2681

[32] Martin A, Soumith C, Léon B. Wasserstein generative adversarial networks//Proceedings of the International Conference on Machine Learning. Sydney, Australia, 2017: 214-223

[33] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735-1780

[34] Guo Y, Zhang Z. LPSE: Lightweight password-strength estimation for password meters. Computers & Security, 2018, 73: 507-518

[35] Matteo D, Maurizio F. Monte Carlo strength evaluation: Fast and reliable password checking//Proceedings of the 22nd ACM Conference on Computer and Communications Security. Denver, USA, 2015: 158-169

[36] Parish Z, Cushing C, Aggarwal S, et al. Password guessers under a microscope: An in-depth analysis to inform deployments. arXiv preprint arXiv:2008.07986, 2020

[37] Pasquini D, Gangwal A, Ateniese G, et al. Improving password guessing via representation learning//Proceedings of the 42nd IEEE Symposium on Security and Privacy. Oakland, USA, 2021: 265-282

[38] Andrew B, Jeff D, Karen S. Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv: 1809.11096, 2018

WANG Ding, Ph. D., professor, Ph. D. supervisor. His research interests focus on password security and cryptographic protocol.

ZOU Yun-Kai, Ph. D. candidate. His research interests focus on password security and deep learning.

TAO Yi, bachelor. His research interests focus on password security.

WANG Bin, bachelor. His research interests focus on password security.

Background

With the development of new computing technologies such as cloud computing, the Internet of Things, and big data, the security threats to passwords are increasing. At the same time, passwords will still be one of the most important methods of identity authentication in foreseeable future. So, studying the security of passwords is necessary and password guessing is exactly a good entry point. On the other hand, the progress of deep learning technology has provided a potential way to improve the efficiency of password guessing. Therefore, this paper focuses on the issue of password guessing from the perspective of deep learning. Specifically, we propose the PR model and achieve the best guessing result under different guessing scenarios. To reduce the dependence of the guessing model on large training samples, we further propose the PR+model. Then we highlight that the passwords generated by different models have different characteristics, so we combine the passwords generated by different models and achieve satisfactory results. Finally, we conduct an in-depth exploration of the GAN model and analyze the reason of the poor performance of this model.

Our research group has devoted a lot of efforts in password security and user authentication scheme. We have published more than 60 academic papers at prestigious venues including top international conferences like ACM CCS, USENIX Security, NDSS, DSN, ESORICS and top journals like IEEE TDSC, IEEE TIFS, ACM TCPS, etc. Our work won the First Prize of Natural Science Award, Ministry of Education of China, 2017, and the Second Prize of Science & Technology Achievement Award, Tianjin, 2020.