软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2022,33(5):1893−1906 [doi: 10.13328/j.cnki.jos.006195]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

深度学习在分组密码差分区分器上的研究应用*

侯泽洲[1], 陈少真[1,2], 任炯炯[1]

[1](战略支援部队信息工程大学, 河南 郑州 450001)

[2](密码科学技术国家重点实验室, 北京 100878)

通信作者: 侯泽洲, E-mail: zezhouhou1998@163.com

摘　要: 差分分析在分组密码分析领域是一种重要的研究方法, 针对分组密码的差分分析的重点在于找到一个轮数或者概率更大的差分区分器. 首先描述了通过深度学习技术构造差分区分器时所需要的数据集的构造方法, 并且分别基于卷积神经网络 (convolutional neural networks, CNN) 和残差神经网络 (residual neural network, ResNet) 训练了两种轻量级分组密码算法 SIMON32 与 SPECK32 的差分区分器, 并对两种模型得到的差分区分器进行了比较, 发现综合考虑时间花销与精度的前提下, 在 SIMON32 的差分区分器构造上, ResNet 训练得到的模型表现更好, 而 CNN 则在 SPECK32 的模型训练上表现的更好; 其次, 研究了网络模型中卷积运算个数对模型精度的影响, 发现在原有模型基础上增加 CNN 模型的卷积层数和 ResNet 模型的残差块数, 都会导致模型精度的下降. 最后, 给出在进行基于深度学习的差分区分器构造时的模型及参数选择建议, 即, 应该首要考虑低卷积层数的 CNN 模型和低残差块数的 ResNet 模型.

关键词: 深度学习; 卷积神经网络; 残差神经网络; 分组密码; 差分区分器

中图法分类号: TP393

中文引用格式: 侯泽洲, 陈少真, 任炯炯. 深度学习在分组密码差分区分器上的研究应用. 软件学报, 2022, 33(5): 1893–1906. http://www.jos.org.cn/1000-9825/6195.htm

英文引用格式: Hou ZZ, Chen SZ, Ren JJ. Research and Application of Deep Learning on Differential Distinguisher of Block Cipher. Ruan Jian Xue Bao/Journal of Software, 2022, 33(5): 1893–1906 (in Chinese). http://www.jos.org.cn/1000-9825/6195.htm

Research and Application of Deep Learning on Differential Distinguisher of Block Cipher

HOU Ze-Zhou[1], CHEN Shao-Zhen[1,2], REN Jiong-Jiong[1]

[1](Information Engineering University, Zhengzhou 450001, China)

[2](State Key Laboratory of Cryptology, Beijing 100878, China)

Abstract: Differential cryptanalysis is an important method in the field of block cipher. The key point of differential cryptanalysis is to find a differential distinguisher with longer rounds or higher probability. Firstly, the method of generating data set is described which is used to train a differential distinguisher based on deep learning. At the same time, the differential distinguisher of two kinds of lightweight block cipher is trained, SIMON32 and SPECK32, based on convolutional neural networks (CNN) and residual neural network (ResNet). In addition, two differential distinguishers are compared and it is found that ResNet is good at differential distinguisher of SIMON32, CNN is good at SPECK32 when considering time and accuracy. Next, the influence of the number of convolution operations of the network model is studied on the accuracy of the neural distinguisher, and it is found that adding the number of convolution layers of the CNN and the number of residual blocks of the ResNet model will cause the accuracy decrease compared with original networks. Finally, some suggestions are given to select networks and parameters when constructing a differential distinguisher based on deep learning, i.e., the CNN with low convolutional layers and the ResNet with low residual blocks should be considered as the first choose.

Key words: deep learning; convolutional neural networks (CNN); residual neural network (ResNet); block cipher; differential distinguisher

---

* 基金项目: 数学工程与先进计算国家重点实验室开放基金 (2018A03); 国家密码发展基金 (MMJJ20180203); 信息保障技术重点实验室开放基金 (KJ-17-002)

收稿时间: 2020-05-03; 修改时间: 2020-06-28, 2020-09-27, 2020-10-13; 采用时间: 2020-10-28

得益于互联网及计算机硬件的发展, 人工智能技术广泛应用于生活的各个场景之中. 作为人工智能得以迅速发展的一个重要原因, 深度学习技术作为人工智能的重点研究领域之一, 推动了人工智能从幕后到台前. 目前应用于生活的人脸识别、智慧农业等的人工智能均离不开深度学习技术的推动. 随着深度学习算法的不断发展, 深度学习在人工智能领域一些最困难的问题上取得了重大突破, 并且被应用于图像识别、自动驾驶、自然语言处理等各大领域[1].

深度学习技术是神经网络发展到一定时期的产物. 1943 年, McCulloch 和 Pitts[2]提出了 MP 神经元模型, 即一个按照生物神经元的结构和工作原理构造出来的抽象和简化了的模型, 实际上就是对单个神经元的一种建模. 其大致模仿了自然神经元的工作原理, 开启了模拟神经网络的先河, 但是其权重设置及调整大量依赖于手工, 十分不利于研究. 1958 年, Rosenblatt[3]在 MP 神经元模型的基础上提出了第一代神经网络单层感知器, 其能够区分三角形、正方形等基本形状. 1986 年, Rumelhart 等人[4]提出了第二代神经网络, 将第一代神经网络中单一固定的特征层改为多个隐藏层, 使用 Sigmoid 函数作为激活函数, 并利用反向传播算法 (back propagation, BP) 逆向传播的思想对网络参数进行学习, 这有效地解决了第一代神经网络只能处理线性分类问题的困难. 卷积神经网络 (convolutional neural networks, CNN)[5]和循环神经网络 (recurrent neural networks, RNN)[6]等神经网络模型也得到了一定的发展. 步入 21 世纪, 伴随着计算芯片等硬件设备的提升, 神经网络的研究也得以快速发展. 2006 年, Hinton 等人[7]首次提出了深度学习的概念, 并指出: 可以通过逐层初始化的方法抑制梯度消失, 从而解决深度神经网络在训练上的难题. 2011 年, Glorot 等人[8]提出了 ReLU 激活函数, 能有效地抑制梯度消失问题.

伴随着深度学习基础理论的发展和各型神经网络模型的提出, 以及专用计算芯片的研发与大量部署使用, 深度学习广泛应用于生活的各个领域. 2016 年, AlphaGo 击败围棋冠军李世石就离不开深度学习技术的助力. 受深度学习技术蓬勃发展的影响, 密码分析领域也逐步使用深度学习技术以帮助减少密码算法分析的复杂度. 在 2019 年美密会 (Crypto2019) 上, Gohr[9]提出了使用深度学习技术对轻量级分组密码算法 SPECK[10]的密钥恢复攻击. 在文献 [9] 中, Gohr 描述了可以通过深度学习得到区分器, 并且说明了通过深度学习技术得到的区分器要优于通过传统分析方法得到的区分器. 同时, Gohr 也刻画了针对 SPECK 算法的密钥恢复攻击, 其通过将贝叶斯搜索方法与深度学习区分器结合对 SPECK32 算法的两轮子密钥进行了恢复, 并说明了使用深度学习技术的密钥恢复攻击的复杂度要远远低于传统密钥恢复攻击的复杂度. 但是, Gohr 并未说明其选择所用神经网络模型的原因. 深度学习技术领域的专家为应对多种数据的处理难题提出了多种神经网络模型, 针对不同的数据类型, 应该选择不同的模型才能保证模型的精度. 因此, 研究不同网络模型对轻量级分组密码算法的差分区分器的影响是有必要的. 同时, Gohr 构造的神经网络区分器的训练时间开销是巨大的, 这不利于后续的密码分析研究, 不利于快速找到适合不同分组算法的神经网络模型. 因此, 在保证模型精度不影响后续差分分析的前提下, 研究神经网络模型中超参数的设置以降低训练开销是有重要意义的.

本文第 1 节描述本文研究的两个轻量级分组密码算法 SIMON[10]与 SPECK[10]. 在第 2 节描述基于深度学习的差分区分器的构造方法, 使用两种典型的神经网络模型构造差分区分器, 并对这两种神经网络模型构造的差分区分器进行对比. 结合第 2 节的内容, 我们在第 3 节说明通过深度学习构造差分区分器的超参数的选择. 最后一节, 我们对目前的研究工作进行总结.

1 SIMON 与 SPECK 算法简介

SIMON 与 SPECK 算法是由 NSA (national security agency) 提出的轻量级分组密码算法[10], 两种算法按照分组长度及密钥长度均有多个版本. 虽然两种算法的运算结构是轻量级的, 但是其仍然可以在多个平台上提供良好的安全性. 两种加密算法可满足多平台的要求, 其在多平台上部署是足够灵活的. 其中, SIMON 算法在硬件上表现优异, 而 SPECK 算法在软件上表现优异. 由于 SIMON 算法与 SPECK 算法在运算上有诸多相似性, 因此我们将两种算法的运算符一并介绍. SIMON 算法与 SPECK 算法的运算过程主要涉及以下的 4 个运算操作.

- $\oplus$: 比特异或操作 XOR;
- &: 比特与操作 AND;

• +: 模 2^n 加操作;

• S^j: 比特循环左移位 j 比特.

1.1 SIMON 算法

为应对不同平台对通信安全性的不同需求, SIMON 算法设计者给出了多个版本的 SIMON 算法以供不同平台的选择. 每个版本的具体参数见表 1.

针对不同版本的 SIMON 算法, 其对应的参数如表 1 所示. 表 1 给出了不同版本的 SIMON 算法的参数, 其中, 固定参数序列 z_i 用于密钥扩展中. 对于特定版本的 SIMON 算法, 通常使用 SIMON2n/mn 指定. 比如说, 对于分组长度为 32 比特、密钥长度为 64 比特的 SIMON 算法, 通常用 SIMON32/64 指定. 在不考虑分组长度时, 一般也直接用 SIMON2n 指定.

设 $GF(2)$ 代表二元有限域, $GF(2)^n$ 代表 $GF(2)$ 上的 n 维向量空间, 那么对于 $k_i \in GF(2)^n$, SIMON2n 的轮函数可以定义为 Feistel 映射 $R_{k_i} GF(2)^n \times GF(2)^n \to GF(2)^n \times GF(2)^n$:

$$R_{k_i}(x_{i+1}, x_i) = (x_i \oplus f(x_{i+1}) \oplus k_i, x_{i+1}),$$

其中, $f(x_{i+1})=(S^1 x_{i+1} \& S^8 x_{i+1}) \oplus S^2 x_{i+1}$, k_i 是轮子密钥. 轮函数的逆为 $R_{k_i}^{-1}(x_{i+2}, x_{i+1}) = (x_{i+1}, x_{i+2} \oplus f(x_{i+1}) \oplus k_i)$, 这通常被用作解密操作. 其加密流程如图 1 所示.

表 1 SIMON 算法参数

分组长度	密钥长度	字节长度	密钥字节数	固定参数序列	轮数
32	64	16	4	z_0	32
48	72	24	3	z_0	36
48	96	24	4	z_1	36
64	96	32	3	z_2	42
64	128	32	4	z_3	44
96	96	48	2	z_2	52
96	144	48	3	z_3	54
128	128	64	2	z_2	68
128	192	64	3	z_3	69
128	256	64	4	z_4	72

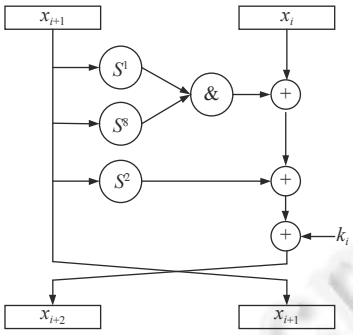

图 1 SIMON 算法的轮函数

1.2 SPECK 算法

同样的, SPECK 为应对不同平台对通信安全性的不同需求, 其算法设计者也给出了多个版本的 SPECK 算法以供用户选择. 每个版本的具体参数见表 2.

针对不同版本的 SPECK 算法, 其对应的参数如表 2 所示. 表 2 给出了不同版本的 SPECK 算法的参数, 其中, 循环参数 α 与 β 用于轮函数运算中. 对于特定版本的 SPECK 算法, 通常使用 SPECK2n/mn 指定. 比如说, 对于分组长度为 32 比特、密钥长度为 64 比特的 SPECK 算法, 通常用 SPECK32/64 指定. 在不考虑分组长度时, 一般也直接用 SPECK2n 指定.

对于 $k_i \in GF(2)^n$, SPECK2n 的轮函数可以定义为映射 $R_{k_i} GF(2)^n \times GF(2)^n \to GF(2)^n \times GF(2)^n$:

$$R_{k_i}(x_{2i+1}, x_{2i}) = ((S^{-\alpha} x_{2i+1} + x_{2i}) \oplus k_i, S^{\beta} x_{2i} \oplus (S^{-\alpha} x_{2i+1} + x_{2i}) \oplus k_i),$$

其中, α 与 β 为循环参数, k_i 是轮子密钥. 轮函数的逆为:

$$R_{k_i}^{-1}(x_{2i+3}, x_{2i+2}) = (S^{\alpha}((x_{2i+3} \oplus k_i) - S^{-\beta}(x_{2i+3} \oplus x_{2i+2})), S^{-\beta}(x_{2i+3} \oplus x_{2i+2})).$$

这通常被用作解密操作. 其加密流程如图 2 所示.

2 神经网络差分区分器的构造及选择

作为一种有效的方法, 差分分析[11]被广泛用于分组密码分析之中. 在差分分析中, 差分区分器是必不可少的,

其经常用于对明文或者密文的区分, 从而辅助密码分析. 在传统的差分分析中, 首要的是找到一条高概率差分特征, 而后通过该高概率差分特征构造差分区分器. 一条高概率差分特征需要输入差分、输出差分以及该输入差分和输出差分构成的差分对的概率. 其中, 输入差分是指输入对的差分值, 输出差分是指输入对经过多轮加密后的输出对的差分值. 差分对的概率也即差分概率, 是指给定一对差分对 (α,β), 当输入对的差分为 α 时, 其对应的输出对的输出差分为 β 的概率. 一个良好的可以区分更长轮数的区分器, 对于分组密码的分析具有极佳的辅助作用. 在传统差分分析中, 构造差分区分器更多地依赖于算法本身结构可能存在的缺陷, 而且构造过程中大量依赖手工推导, 大大迟缓了密码分析的过程. 近几年, 依赖自动化搜索技术[12]寻找差分区分器, 已经逐步成为差分区分器构造的主流方法.

表 2 SPECK 算法参数

分组长度	密钥长度	字节长度	密钥字节数	循环参数α	循环参数β	轮数
32	64	16	4	7	2	22
48	72	24	3	8	3	22
48	96	24	4	8	3	23
64	96	32	3	8	3	26
64	128	32	4	8	3	27
96	96	48	2	8	3	28
96	144	48	3	8	3	29
128	128	64	2	8	3	32
128	192	64	3	8	3	33
128	256	64	4	8	3	34

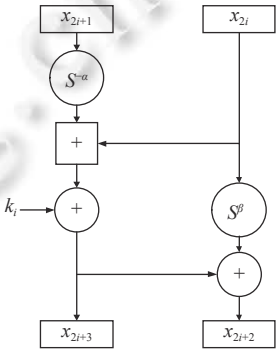

图 2 SPECK 算法的轮函数

差分区分器对明文或者密文的区分, 从本质上说就是对明文数据集或者密文数据集的分类. 而深度学习技术在图像、语音识别等领域被广泛应用于分类问题, 这与差分区分器的功能是相似的. 因此, 深度学习技术在理论上也可用于辅助差分区分器的构造或者直接用于构造差分区分器. 目前, 已经有多名学者针对基于深度学习技术构造区分器进行了研究. Gohr 最早提出了将深度学习技术应用于差分区分器的构造及密钥恢复攻击之中, 在文献 [9] 中, Gohr 对比了传统的差分区分器和基于深度学习技术的差分区分器, 无论是在区分的准确率或者应用于密钥恢复攻击上的复杂度, 基于深度学习的差分区分器都明显优于传统的差分区分器. 同样的, 在文献 [13] 中, Anubhab 等人构造了基于深度学习的 GIMLI 算法的差分区分器, 相比于传统的差分区分器,其所需要的数据量更低. 相比于只使用一条或多条差分路径的传统的差分区分器, 基于深度学习的差分区分器可以通过对已知的密文数据进行学习, 完成对未知密文数据的判别. 从基于深度学习技术的差分区分器的构造方法上看, 基于深度学习技术的差分区分器可以通过已知密文数据学到密文对的输入差分的特征, 也就是说, 通过深度学习技术可以学到更多的差分路径的特征, 这也就是基于深度学习技术的差分区分器在区分效果上优于传统的差分区分器的一个原因.

在本节中, 我们探究了 SPECK32 和 SIMON32 的基于深度学习技术的差分区分器的构造, 以及两种典型神经网络模型 CNN 和残差神经网络 (residual neural network, ResNet)[14]对基于深度学习构造的差分区分器的精度的影响.

2.1 数据集的构造

在深度学习中, 数据集的选取影响着神经网络模型对未知数据判断的准确率. 因此, 我们在构造神经网络差分区分器时, 选取的数据集应尽可能随机并覆盖所有可能的情况. 在使用传统的差分分析方法对 SIMON32 与 SPECK32 进行密钥恢复研究时, 首要的步骤是寻找到一个轮数尽可能长、概率尽可能大的差分区分器, 这离不开找到一个良好的输入差分[15]. 与传统的差分区分器相同, 神经网络差分区分器也需要给定输入差分. 我们选择固定的输入差分作为神经网络差分区分器的输入差分.

我们将给定是输入差分记为 *diff*, 下面给出构造数据集的方法.

(1) 随机生成 3 个包含元素个数为 n 的集合分别为 $X1$, $X2$ 和 Y. 其中, $X1$ 与 $X2$ 中的元素为长度不超过 32 比

特的非负整数, Y 中元素为 0 或者 1. 将这 3 个集合中的元素一一对应;

(2) 若 Y 中某一元素为 1, 则将 $X2$ 中的对应位置的元素的值替换为 $X1$ 中对应元素与输入差分 *diff* 异或后的值, 将进行变换后的 $X2$ 序列记为 $X3$;

(3) 随机生成包含元素个数为 n 的集合 KEY, 与 $X1$, $X3$ 和 Y 中的元素一一对应, 其中, KEY 中的元素为长度不超过 64 比特的非负整数;

(4) 将 $X1$ 与 $X3$ 中的元素看作明文, 使用对应位置的 KEY 中的元素作为密钥进行加密, 得到的密文序列记为 $X4$ 与 $X5$;

(5) 将 $X4$ 与 $X5$ 中对应位置的元素拼接为新的长度不超过的 64 比特的新元素, 并将其作为数据, Y 中对应位置的元素作为该数据的标签.

图 3 给出了详细的构造流程.

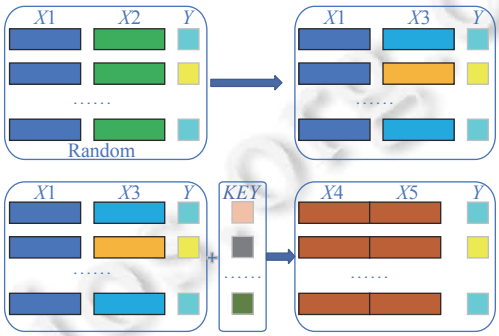

图 3 数据集构造流程

作为深度学习技术的一个重要的分支——有监督学习, 其训练的数据集中的每一个实例均对应着一个标签, 该标签作为深度学习技术的一个期望输出, 被用于对神经网络参数的调整. 如图 3 所示, 我们将 $X4$ 与 $X5$ 拼接而成的 64 比特元素作为一个实例, 与其对应的标签共同构成一个集合, 并作为进行后续深度学习训练的数据集.

深度学习技术是对已知数据集的特征进行提取, 并利用这些提取的特征对未知数据进行计算, 从而得出分类的结果. 在构造不同算法的基于深度学习技术的差分区分器时, 通过不同加密算法随机生成的数据集对于神经网络模型是一致的, 因此对于 SIMON32 和 SPECK32 的区分器构造来说, 其数据集的构造方法是相同的, 仅仅在输入差分的选择中有所不同.

2.2 基于 CNN 的差分区分器的构造与训练

CNN[5]是一类包含卷积计算且具有深度结构的前馈神经网络 (feedforward neural networks), 是深度学习的代表算法之一. CNN 的基本运算包含卷积运算和池化运算等运算, 其中, 卷积运算用于提取输入的不同特征, 而池化运算可以将卷积运算提取到的特征个数以提取主要特征的形式进一步缩减. 而且卷积运算对数据的不同位置使用相同的卷积核以及池化运算每次仅仅在局部进行运算, 也就是 CNN 采用了局部连接和权值共享的方式, 这减少了权值的数量使得网络易于优化, 同时也降低了模型的复杂度, 也就是减小了过拟合的风险. 卷积神经网络的隐藏层中一般包含卷积层、全连接层和池化层. 为便于研究, 我们构造较为简单的卷积神经网络模型, 其隐藏层仅由卷积层和全连接层构造所得. 由于卷积神经网络中输出层的上游通常是全连接层, 因此其结构和工作原理与传统前馈神经网络中的输出层相同. 使用全连接层作为输出层, 使用逻辑函数 (Sigmoid) 输出模型结果. 为便于理解, 我们绘制所构造的模型中的卷积层.

在图 4 中, 我们描述了我们构造的卷积神经网络中的所有卷积操作. 输入层将原始数据进行格式化后传递给第一个卷积层 Conv1D. 在经过第一个卷积层的卷积计算后, 数据被传递给 5 个相同的卷积层, 被这 5 个相同的卷积层依次计算, 并使用 BatchNormalization 进行数据的归一化. 在数据经过 5 轮相同的卷积操作后, 被传输到全连接层和输出层输出最后的结果. 我们使用构造的卷积神经网络模型做 SPECK32 和 SIMON32 的神经网络差分区

分器的训练. 运算在配置为 i7-8750H 和 GTX1080 的电脑上完成, 深度学习训练过程中的运算主要在显卡上进行以提高训练的效率.

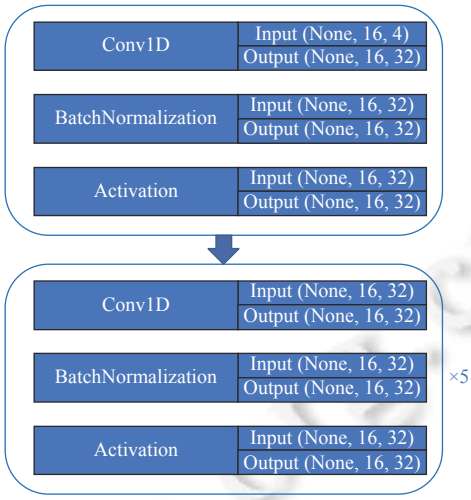

图 4 卷积层

2.2.1 基于 CNN 的 SIMON32 的神经网络区分器

选择文献 [15] 附录中所展示的 SIMON32 的差分路径的输入差分 (0x0, 0x200) 作为 SIMON32 的神经网络区分器的差分. 在训练过程中的训练数据集的数量为 10^7, 验证集的大小为 10^5. 在训练 7 轮神经网络区分器时, 数据集为加密 7 轮所得到的密文对集; 而对于 8 轮神经网络区分器, 则加密 8 轮. 为了可以更好地研究训练过程中模型在验证集上的精度和损失等因素的变化情况, 对数据集进行 200 次迭代. 将训练过程中验证集的准确率与损失变化情况绘制图像. 如图 5 所示, 其中, 横坐标表示训练过程中的迭代次数, 纵坐标表示精度及损失. 图中的 4 条折线表示模型训练过程中验证集的精度与损失. 从图 5 中我们可以看到: 使用 CNN 训练的 SIMON32 的 7 轮区分器模型的验证集精度在 94% 左右, 8 轮区分器模型的验证集精度在 75% 左右.

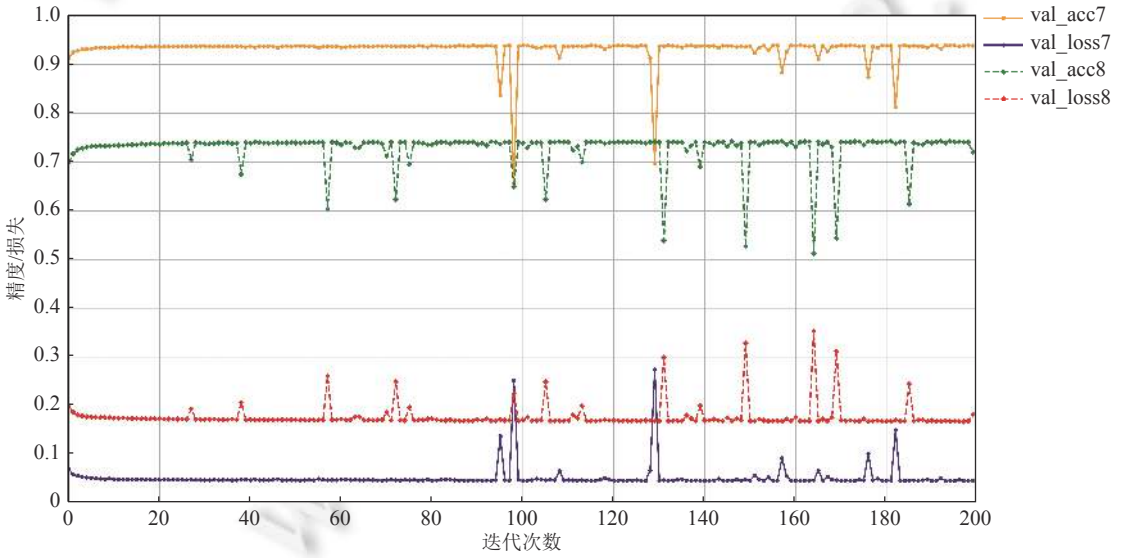

图 5 SIMON32 的 CNN 训练

通过图 4 所示的神经网络结构以及图 5 所示的训练, 我们可以得到训练后的神经网络模型. 对于使用 7 轮加密数据进行训练得到的模型, 我们称这个网络模型为 SIMON32 的基于 CNN 的 7 轮差分区分器. 通过这个区分器, 我们可以对一个经过 7 轮加密的密文对进行分析, 并以一定概率判断该密文对是否由输入差分为 (0x0,0x200) 的明文对加密生成. 同样的, 对于由 8 轮加密数据训练得到的模型, 我们也可使用其对经过 8 轮加密的密文对进行判别. 通过此种方式, 我们便可以将减轮的加密算法与随机置换进行区分.

为了衡量模型的精度, 我们使用函数 $acc=\frac{N_1}{N}$ 对模型进行评估. 其中, N 为使用模型进行判断的数据个数, N_1 为使用模型对数据进行判断时判断结果与真实结果相同的数据的个数, acc 为模型的精度. Keras[16]中的 evaluate 函数即使用此方法对模型的精度进行评估. 我们按照图 3 所示方法生成数量为 10^5 的新数据集, 并使用 Keras 中的 evaluate 函数对模型进行评估, 模型的精度与验证集的精度相当. 这已经满足了密码分析中对区分器的基本要求, 即精度超过 50%.

同时, 研究训练过程中精度与损失的变化, 我们发现在训练过程中曲线会出现波动, 这是由于优化器设置导致的. 在本文中, 我们选择 Keras 中的 Adam 作为优化器, 且使用默认的学习率进行训练, 这就使得学习率无法变化, 从而导致图像出现轻微波动. 学习率是深度学习中的一个重要的超参数, 如何调整学习率, 是训练出好模型的关键要素之一. 在通过随机梯度下降求解问题的极小值时, 梯度不能太大, 也不能太小: 梯度太大容易出现超调现象, 即在极值点两端不断发散, 或是剧烈震荡; 太小会导致无法快速找到好的下降方向, 随着迭代次数增大损失基本不变. 学习率越小, 损失梯度下降的速度越慢, 收敛的时间更长. 在设置学习率时, 可在训练初期设置较大的学习率, 而后在训练过程中学习率逐渐降低, 这样可有效地避免出现波动. 在深度学习领域, 调节超参数是没有成文规则的, 只能通过有限的工具[17]来调节超参数来优化模型. 由于我们关注的是模型对通过固定输入差分得到的密文对的判断的准确率, 在模型训练过程中, 我们保存了模型对验证集准确率最高时的模型参数, 以便后续的研究与使用. 因此, 由于学习率导致的准确率和损失的轻微波动并不影响我们对模型的使用. 但若模型训练过程中准确率和损失波动过大, 则需要对学习率进行调整, 以求模型尽可能训练到最佳位置. 最后, 从验证集精度的变化曲线可以看出: 在进行模型训练过程中, 可能并不需要多代训练, 通过前几代训练便可以得到一个较好的差分区分器模型. 这有助于降低训练的时间开销, 以实现快速的对分组算法的分析.

2.2.2 基于 CNN 的 SPECK32 的神经网络区分器

与 SIMON32 的神经网络差分区分器的训练过程相似, 选择差分 (0x40,0x0) 作为 SPECK32 的神经网络区分器的差分. 在训练过程中, 数据集的数量为 10^7, 验证集的大小为 10^5. 训练 SPECK32 的 6 轮与 7 轮神经网络区分器时, 数据集均通过 2.1 节的方法构造. 与在 SIMON32 上的训练相同, 我们对数据集进行 200 次迭代. 我们将 SPECK32 的 6 轮与 7 轮区分器的训练过程验证集的精度及损失绘制如图 6 所示, 其中, 横坐标表示训练过程中的迭代次数, 纵坐标表示精度及损失. 从图 6 中我们可以看到: 使用 CNN 训练的 SPECK32 的 6 轮区分器模型的验证集精度在 78% 左右, 7 轮区分器模型的验证集精度在 61% 左右. 对模型进行评估, 模型的精度与模型在验证集上的精度相当. 我们可以看到: 使用 CNN 模型对 SPECK32 进行 6 轮与 7 轮区分器训练时, 其损失与精度在经过最初的几次的迭代后便趋于平缓. 这对于我们构造与训练神经网络差分区分器是有所启示的, 我们对于模型的训练可能并不需要经过多次迭代也可取得一个良好的模型.

2.3 基于 ResNet 的差分区分器的构造与训练

作为卷积神经网络的一种优化版本, ResNet[14]在 2015 年一经提出便被应用到 ImageNet 比赛中, 并取得了第一名的成绩. 由于神经网络训练过程中的反向传播, 很容易出现梯度消失的问题. 这使得随着网络层级的不断增加, 模型的精度会不断得到提升. 而当网络层级增加到一定的数目后, 训练精度和测试精度会迅速下降, 从而导致当网络变得很深以后, 深度网络就变得更加难以训练. 也就是说, 随着网络深度的增加, 准确率达到饱和然后迅速退化. ResNet 引入了残差网络结构, 也就是在卷积神经网络中叠加了恒等映射, 这可以让网络随着深度增加而不退化, 这也从侧面反映了多层非线性网络无法逼近恒等映射网络. 通过这种残差网络结构可以增加网络层数, 同时使用了较少的池化层, 大量使用降采样, 提高了传播效率, 最终的分类效果会表现得非常好. 为便于直观了解我们

所构造的模型, 我们将所构造的残差神经网络中的残差块绘制如图 7 所示.

我们构造的残差模型的隐藏层中共包含 5 个图 7 所示的残差块. 为保证硬件的统一, 我们选用与 CNN 训练过程中相同的硬件设备进行训练. 与 CNN 中对原始密文数据集的处理方式相同, 我们将原始数据进行格式处理, 并通过 Conv1D 卷积层对数据进行计算和通过 BatchNormalization 对数据进行归一化, 而后传输到图 7 所示的残差块中进行计算, 最后通过输出层输出最终的结果.

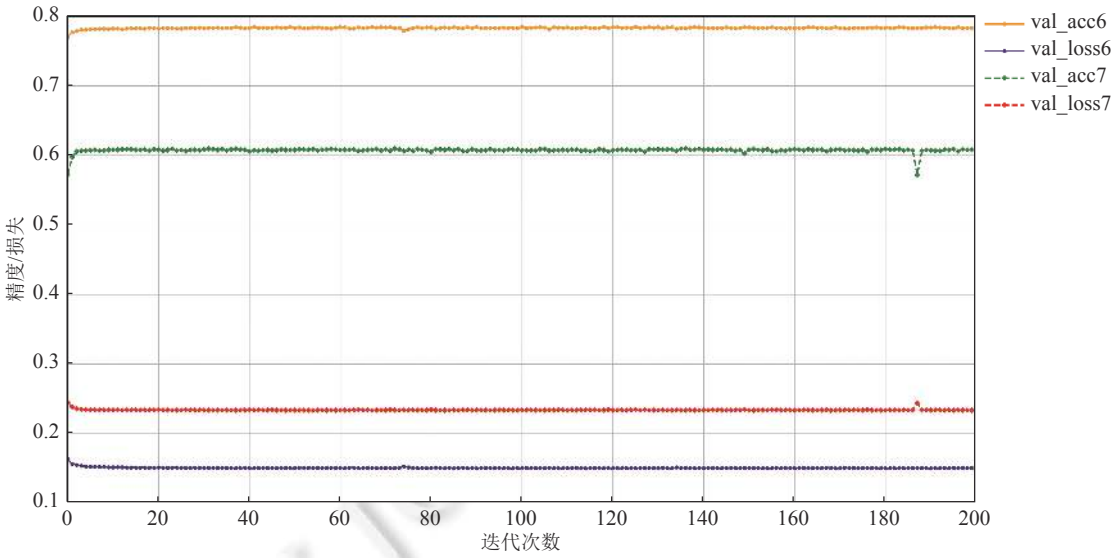

图 6 SPECK32 的 CNN 训练

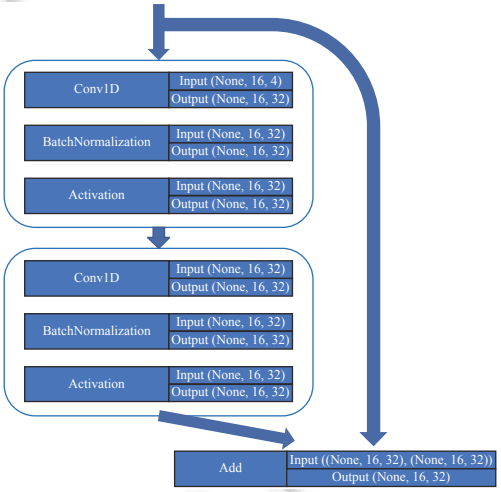

图 7 残差块

2.3.1 基于 ResNet 的 SIMON32 的神经网络区分器

与 CNN 训练中选择的参数相同, 选择输入差分为 (0x0, 0x200) 作为 SIMON32 的神经网络区分器的差分. 在训练过程中, 训练数据集的数量为 10^7, 验证集的大小为 10^5, 且在训练中进行 200 次迭代. 在训练 7 轮与 8 轮神经网络区分器时, 其数据集的获得方式与基于 CNN 的 SIMON32 神经网络区分器训练时的数据集获取方式一致. 我们将训练过程中验证集的准确率与损失变化情况绘制图像, 如图 8 所示, 其中, 横坐标表示训练过程中的迭代次数, 纵坐标表示精度及损失. 从图 8 可以看到: 使用 ResNet 训练的 SIMON32 的 7 轮区分器模型的验证集精度在

97% 左右, 8 轮区分器模型的验证集精度在 75% 左右. 对模型进行评估, 其在新的数据集上的精度与验证集上的精度相当.

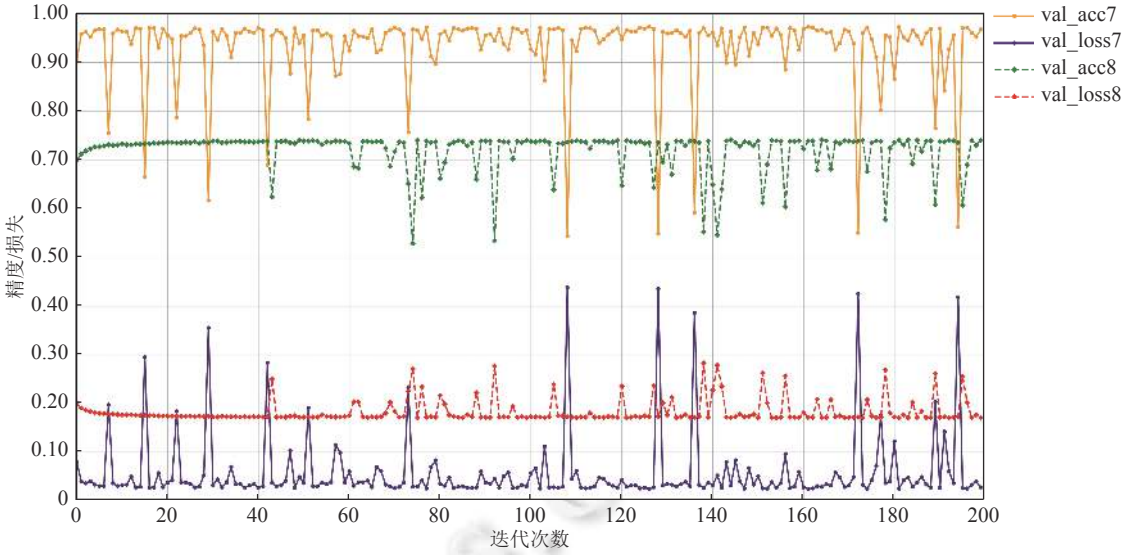

图 8 SIMON32 的 ResNet 训练

2.3.2 基于 ResNet 的 SPECK32 的神经网络区分器

与 SPECK32 的基于 CNN 模型的神经网络差分区分器的训练过程相似, 我们选择差分 (0x40, 0x0) 作为 SPECK32 的基于 ResNet 的神经网络区分器的差分. 在训练过程中, 数据集的数量为 10^7, 验证集的大小为 10^5, 且在训练中进行 200 次迭代. 通过第 2.1 节可知, 在训练 6 轮和 7 轮神经网络区分器时, 其数据集分别通过 7 轮与 8 轮 SPECK32 加密得到. 我们将训练过程中验证集的准确率与损失变化情况绘制图像, 如图 9 所示, 其中, 横坐标代表训练过程中的迭代次数, 纵坐标表示精度及损失. 从图 9 中我们可以看到: 使用 ResNet 训练的 SPECK32 的 6 轮区分器模型的验证集精度在 78% 左右, 7 轮区分器模型的验证集精度在 61% 左右. 对模型进行评估, 模型的精度与验证集的精度相当, 在 6 轮和 7 轮上的精度分别是 78. 3% 和 61%.

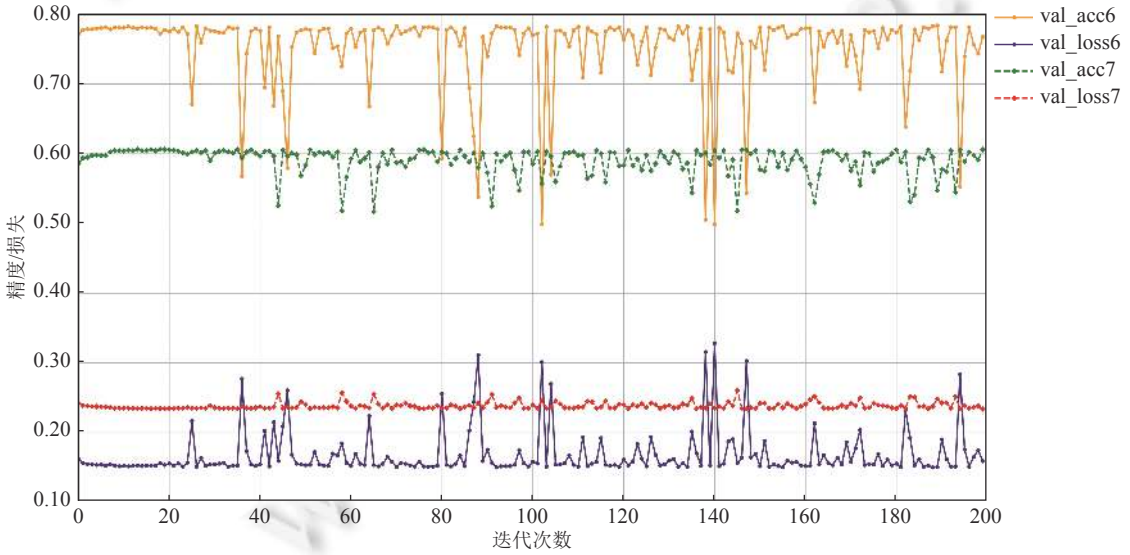

图 9 SPECK32 的 ResNet 训练

2.4 两种区分器的对比

2.4.1 传统区分器与神经网络区分器对比

差分分析属于选择明文的攻击方法, 其探究选择的明文对差分在加密过程中的概率传播特性, 将分组密码与随机置换进行区分, 并依赖于此进行密钥恢复攻击. 在传统的差分区分器构造过程中, 其依赖一对或几对差分对, 而这离不开对差分路径的寻找. 目前, 主流的寻找差分路径的方式为自动化搜索, 包括基于 SAT 的自动化搜索和基于 MILP 的自动化搜索. 为更好地利用差分对, 目前常用的技术是固定输入差分和输出差分, 使用自动化搜索技术差分路径, 而后统计并计算差分对的概率[18]. 而对于基于深度学习的差分区分器, 其利用一个给定的输入差分, 将密文对作为训练集进行模型的训练构造区分器. 而由于这些密文对的差分并不固定, 因此, 通过深度学习训练的差分区分器可以视为一个多合一的区分器[13]. 从这一角度来看, 基于深度学习的差分区分器可以更有效地利用输入差分.

在 Crypto2019 中, Gohr 已经对 SPECK32 算法的神经网络区分器强于传统的差分区分器进行了对比说明, 但目前仍未有文献对 SIMON32 的神经网络区分器和传统区分器进行对比说明. 因此, 我们着重对 SIMON32 算法的传统差分区分器和基于深度学习的差分区分器进行对比说明. 在第 2.2 节与第 2.3 节中, 我们构造了 SIMON32 的 7 轮和 8 轮的神经网络区分器. 为进行对比, 我们构造 7 轮和 8 轮 SIMON32 的传统差分区分器, 即基于差分路径的差分区分器. 基于文献 [18] 中的 SIMON 算法的 SAT 模型, 使用 Z3 求解器[19]搜索得到了 SIMON32 算法的 8 轮差分路径.

如表 3 所示, 基于 SAT 对 SIMON32 的差分路径进行了自动化搜索, 其在 8 轮中最佳的差分路径的概率为 2^{-25}. 也就是说, 使用传统的差分区分器至少需要 2^{25} 对明文对[20]才具有区分效果. 而在第 2.2 节和第 2.3 节中, 构造基于深度学习的差分区分器只需要 10^7 对密文对即可完成对神经网络模型的训练.

但是传统的差分区分器在轮数较低时, 其传播概率较大, 所需要的选择明文量也更少. 我们使用 Z3 求解器搜索得到了 SIMON32 算法的最佳 7 轮差分路径, 其概率为 2^{-20}.

通过表 4 发现: 对于 SIMON32 的 7 轮区分器, 其依赖于传统的差分区分器所需要的选择明文量要低于基于深度学习技术构造差分区分器所需要的明文量. 我们可以发现, 传统的差分区分器所需要的选择明文量与其轮数有关: 当轮数增加时, 即使选择固定轮数, 其传播概率也会降低, 从而使得选择明文量增加; 而对于基于深度学习的差分区分器, 其由于需要对差分特征进行学习, 因此其选择的明文量越大, 其准确率一般也会增大.

表 3 SIMON32 的 8 轮差分路径

Round	ΔL^i	ΔR^i	$\log_2(p)$
0	0x2800	0xaaa0	–
1	0xaa0	0x2800	−3
2	0x280	0xaa0	−5
3	0xa0	0x280	−3
4	0x0	0xa0	−3
5	0xa0	0x0	0
6	0x280	0xa0	−3
7	0xaa0	0x280	−3
8	0x2800	0xaa0	−5

表 4 SIMON32 的 7 轮差分路径

Round	ΔL^i	ΔR^i	$\log_2(p)$
0	0x0	0x80	–
1	0x80	0x0	0
2	0x200	0x80	−2
3	0x880	0x200	−2
4	0x2000	0x880	−4
5	0x8880	0x2000	−2
6	0x202	0x8880	−6
7	0x8088	0x202	−4

对于相同的密码算法, 由于算法的扩散性和混淆性, 当轮数增加时, 密文的伪随机性也在增加, 这导致了对于基于深度学习得到的差分区分器其识别效果将会下降. 而对于通过差分路径构造的差分区分器, 其轮数的增加会导致其差分路径的概率下降, 从而使得其区分效果也是下降的. 从这一点来看, 两种方法得到的差分区分器是相同的. 同时, 在构造基于深度学习的差分区分器时, 其需要给定输入差分, 当该输入差分对应的输出差分分布较为均匀时, 神经网络区分器的区分效果是较差的. 而对于传统的差分区分器, 当输入差分对应的输出差分分布较为均匀时, 其差分路径的概率接近 $\frac{1}{2^n}$, 其中, n 为分组长度. 这样得到的差分区分器对加密算法与随机置换的区分效果也是较差的. 即:

通过差分路径的寻找可以帮助筛选得到好的输入差分, 以使得基于深度学习的差分区分器的区分效果更好.

2.4.2 两种神经网络模型的区分器对比

我们对两种模型进行对比, 以便选择更佳的模型进行分组密码的分析. 表 5 给出了其中的对比.

在表 5 中, 训练时间是指在配置为 i7-8750H 和 GTX1080 的 PC 上进行一代 (epoch) 运算的时间, 精度是指使用新数据集对模型进行评估的精度. 正如表 5 所展示的: 对于 SIMON32 的差分区分器来说, ResNet 模型在单考虑精度的前提下是要优于 CNN 模型. 因此, 在进行 SIMON32 的差分区分器训练过程中, 在硬件设备可以满足及时间可以允许的前提下, 选择 ResNet 作为神经网络区分器模型对于区分器的精度的提升是有效的. 而对于 SPECK32 的差分区分器, CNN 模型在考虑时间与模型精度上均胜于 ResNet, 也即对于 SPECK32 的差分区分器来说, 首要应该选择 CNN 模型. 同时, 根据第 2.2 节与第 2.3 节中差分区分器的训练过程中验证集的精度的变化, 我们发现 CNN 模型与 ResNet 模型并未出现过拟合的现象, 而且精度在最初 30 次迭代中已经达到最高点. 这也启示我们, 可通过减少迭代来降低训练的时间花销.

表 5 CNN 与 ResNet 的对比

算法	区分器轮数	网络模型	训练时间 (s/epoch)	精度 (%)
SIMON32	7	CNN	68	93.8
SIMON32	7	ResNet	110	97.2
SIMON32	8	CNN	68	74.1
SIMON32	8	ResNet	110	73.7
SPECK32	6	CNN	68	78.3
SPECK32	6	ResNet	110	78.3
SPECK32	7	CNN	68	60.8
SPECK32	7	ResNet	110	60.7

3 神经网络差分区分器的参数选择建议

不同的神经网络模型针对相同的数据集训练出的模型的精度会有所不同, 而在相同的神经网络模型中,由于网络超参数的设置, 也会导致训练出的模型精度有所不同. 在卷积神经网络模型及其各种优化版本中, 卷积层的数量会直接影响模型的精度. 一般认为: 在没有过拟合现象出现时, 对数据集进行越多的运算, 数据集的特征被模型学到的越多, 也就意味着模型的精度越高. 在深度学习中, 有时会通过增加卷积层等运算层的个数提高模型的精度.

为了研究神经网络在构造差分区分器中超参数选取对模型精度的影响, 我们分别增加了 CNN 模型与 ResNet 模型的卷积层数与残差块数. 将图 4 中相同卷积层的层数增加到 10 层, 将图 7 所示的残差块的个数增加到 10 个, 并对 7 轮 SPECK32 差分区分器和 8 轮 SIMON32 差分区分器进行训练.

3.1 改变网络结构对区分器精度的影响

首先针对 SPECK32 的 7 轮差分区分器进行研究. 通过改变 CNN 和 ResNet 模型中卷积层和残差块的个数训练差分区分器, 将训练过程中模型对验证集判断的精度变化绘制如图 10. 其中, 横坐标依然代表着训练过程的迭代次数, 纵坐标表示精度. 图例中, CNN5, CNN10 分别代表只有 5 层相同卷积层的 CNN 模型与 10 层相同卷积层的 CNN 模型, 而 ResNet5, ResNet10 分别代表 5 个残差块的 ResNet 模型与 10 个残差块的 ResNet 模型. 我们从图 10 中可以看出: 对于 SPECK32 的 7 轮差分区分器, 虽然增加了模型的深度对数据进行了更多的计算去提取特征, 但是模型在验证集上的精度并没有因此而上升, 反而有所下降. 当我们使用 Keras 中的 evaluate 函数评估模型的精度时, 当使用 5 层卷积层时, 其精度大约为 60.8%; 但是当使用 10 层卷积层时, 其精度只能保持在 50% 左右. 而构造区分器的问题本质上是二分类问题, 因此, 使用 10 层卷积层构造的 CNN 模型所训练得到的区分器在分类问题上是失败的, 其无法有效地对数据进行区分. 同时, 考虑 ResNet5 与 ResNet10 的精度, 我们发现相较于 ResNet5 的精度, ResNet10 的精度也有所下降.

同样基于 CNN5, CNN10, ResNet5 和 ResNet10 这 4 种模型训练了 8 轮的 SIMON32 的差分区分器, 发现: 类

似于这 4 种模型对 SPECK32 的差分区分器的影响, 增加卷积层层数和残差块个数都会使得区分器的精度有所下降.

如图 11, 横坐标依然代表着训练过程的迭代次数, 纵坐标表示模型对于验证集判断的精度. 图例中 CNN5, CNN10, ResNet5, ResNet10 分别代表只有 5 层相同卷积层的 CNN 模型与 10 层相同卷积层的 CNN 模型以及有 5 个残差块的 ResNet 模型与 10 个残差块的 ResNet 模型. 从图 11 中可以看出, CNN5 与 ResNet5 面对验证集的精度都远远高于 CNN10 和 ResNet10. 使用 Keras 的 evaluate 函数对模型进行评估, 我们发现对于未知数据, 选择 CNN5 和 ResNet5 进行判别的精度大约为 74%, 而 ResNet10 的精度大约为 73%, CNN10 的精度不足 70%.

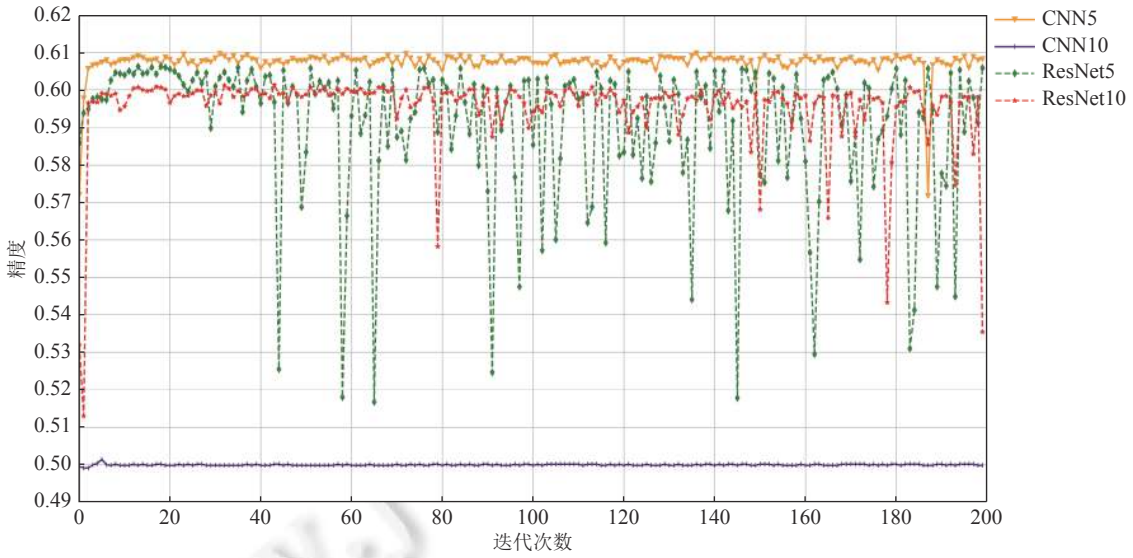

图 10 SPECK32 不同参数的训练

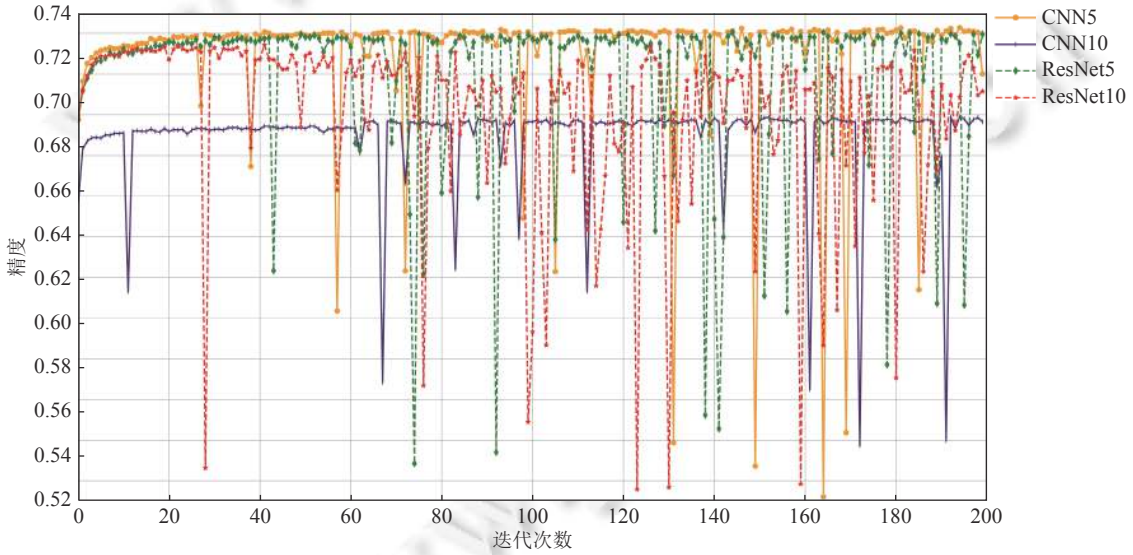

图 11 SIMON32 不同参数的训练

同时, 在深度学习训练过程中, 增加卷积层的个数会增加训练的时间花销. 我们在 GTX1080 的显卡上进行差分区分器的训练时, 在 CNN 模型中, CNN5 的时间花销大约为 CNN10 的一半; 在 ResNet 模型中也有相似的结论, ResNet5 的时间开销大约为 ResNet10 的一半. 这也就意味着: 增加卷积层数或者残差块数加大了训练过程的时间开销, 而精度却没有上升反而有所下降.

3.2 分析与建议

在密码分析中, 时间复杂度与存储复杂度都是需要考虑的内容. 有时需要使用时空折中技术, 通过以时间换空间或者以空间换时间完成对密码的分析. 在通过深度学习技术构造差分区分器时, 我们需要考虑的内容是模型的精度与训练时间开销的折中. 根据不同的研究目标, 我们通常会选择增加运算内容, 加大时间开销, 从而提高模型的精度. 或者在精度要求较低时, 通过减少训练中的计算从而降低时间开销. 在基于 CNN 模型或 ResNet 模型构造差分区分器时, 我们已经发现: 增加卷积层数和残差块数虽然增加了时间开销, 但是对于精度的提升是无效的. 我们在通过深度学习技术构造差分区分器时, 需要考虑训练的花销及模型精度对后续密码分析的影响. 通过 CNN5 模型构造的 SIMON32 的区分器与 SPECK32 的区分器与 ResNet5 构造的区分器的精度相差在 4% 以内, CNN5 时间开销在 GTX1080 上仅需要 68 s, 而 ResNet5 的时间花销却大约为 110 s.

针对 SIMON32 和 SPECK32 两种轻量级分组算法, 通过 CNN 就可构造得到模型精度符合密码分析要求的区分器, 而无须再通过 ResNet 模型构造差分区分器. 但是 ResNet 模型由于其独特的残差块设计, 可使模型的深度增加而不会使得梯度消失. 因此, ResNet 在训练深度较深的区分器时是有优势的.

综合考虑, 面对轻量级分组密码算法的差分区分器, 可预先通过 CNN 模型进行快速地训练, 若 CNN 模型训练得到的区分器的精度已经满足后续密码分析的要求, 便可快速对密码进行分析. 而若 CNN 模型并不能满足密码分析的要求, 可通过尝试 ResNet 等神经网络模型训练神经网络区分器. 在设备与时间受限的情况下, CNN 模型应是进行基于深度学习技术构造差分区分器的首要选择. 同时, 我们也发现: 由于基于深度学习技术构造差分区分器所使用的数据集较为简单, 不像图像或者语音中使用的数据那样复杂, 构造差分区分器的网络模型无须太多的层数已可得到一个较好的结果. 因此, 在使用深度学习技术构造差分区分器时, 应以少层数的模型为主, 这样也可以降低训练的时间花销.

4 总结与未来工作

本文深入研究了基于深度学习的差分区分器的构造及网络模型参数对其的影响. 首先描述了基于深度学习的差分区分器的构造方法, 而后, 基于 CNN 模型与 ResNet 模型训练了神经网络差分区分器, 并对两种方法训练的差分区分器进行了比较. 最后, 给出了训练神经网络差分区分器的参数选择建议. 我们在实验过程中发现, 不同的网络模型对密码算法的差分区分器的训练效果是不一样的. 在我们构造的 CNN 模型与 ResNet 模型中, CNN 模型的训练时间开销更少而精度与 ResNet 模型的精度相当. 由于差分区分器的构造过程在密码分析中属于预计算过程, 其一旦训练完成, 在之后的密码分析中将可以一直使用. 因此, 即使在前期花费大量的时间训练神经网络差分区分器去提高区分器的精度, 对密码分析也是有效的. 但是预训练时间过长,依然不利于对分组密码算法的研究. 同时我们发现: 增加网络模型中的卷积运算, 并不能有效提高模型的精度. 我们在实验中发现: 增加卷积层数和残差块数会导致差分区分器模型的精度下降, 从而无法对数据进行有效的区分, 最终导致实验失败. 因此, 若精度的提升不足以大幅度降低密码分析的复杂度, 通过适当地减少卷积操作从而减少训练的时间, 也将提高密码分析的效率. 我们在实验过程中也发现, 输入差分的选择将会影响神经网络差分区分器的精度. 目前, 我们正在研究输入差分对神经网络差分区分器精度的影响, 从传统的差分分析方法角度分析差分对神经网络差分区分器的影响.

References:

[1] Zhang ZK, Pang WG, Xie WJ, Lü MS, Wang Y. Deep learning for real-time applications: A survey. Ruan Jian Xue Bao/Journal of Software, 2020, 31(9): 2654−2677 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5946.htm [doi: 10.13328/j.cnki.jos.005946]

[2] McCulloch WS, Pitts W. A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics, 1943, 52(1−2): 115−133. [doi: 10.1007/BF02478259]

[3] Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 1958, 65(6): 386–408. [doi: 10.1037/h0042519]

[4] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature, 1986, 323(6088): 533−536. [doi: 10.1038/323533a0]

[5] Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: A convolutional neural-network approach. IEEE Trans. on Neural Networks, 1997, 8(1): 98–113. [doi: 10.1109/72.554195]

[6] Williams RJ, Zipser D. A learning algorithm for continually running fully recurrent neural networks. Neural Computation, 1989, 1(2): 270–280. [doi: 10.1162/neco.1989.1.2.270]

[7] Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. Neural Computation, 2006, 18(7): 1527–1554. [doi: 10.1162/neco.2006.18.7.1527]

[8] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. Journal of Machine Learning Research, 2011, 15: 315–323.

[9] Gohr A. Improving attacks on round-reduced speck32/64 using deep learning. In: Boldyreva A, Micciancio D, eds. Advances in Cryptology. Cham: Springer, 2019. 150−179. [doi: 10.1007/978-3-030-26951-7_6]

[10] Beaulieu R, Treatman-Clark S, Shors D, Weeks B, Smith J, Wingers L. The SIMON and SPECK lightweight block ciphers. In: Proc. of the 52nd ACM/EDAC/IEEE Design Automation Conf. San Francisco: IEEE, 2015. 1−6. [doi: 10.1145/2744769.2747946]

[11] Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology, 1991, 4(1): 3−72. [doi: 10.1007/BF00630563]

[12] Sun SW, Hu L, Wang P, Qiao KX, Ma XS, Song L. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar P, Iwata T, eds. Advances in Cryptology. Berlin: Springer, 2014. 158−178. [doi: 10.1007/978-3-662-45611-8_9]

[13] Baksi A, Breier J, Chen Y, Dong XY. Machine learning assisted differential distinguishers for lightweight ciphers. In: Proc. of the 2021 Design, Automation & Test in Europe Conf. & Exhibition (DATE). Grenoble: IEEE, 2021. 176−181. [doi: 10.23919/DATE51398.2021.9474092]

[14] He KM, Zhang XY, Ren SQ, Sun J. Deep residual learning for image recognition. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 770−778. [doi: 10.1109/CVPR.2016.90]

[15] Abed F, List E, Lucks S, Wenzel J. Differential cryptanalysis of round-reduced simon and speck. In: Proc. of the 21st Int'l Workshop on Fast Software Encryption. London: Springer, 2015. 525−545. [doi: 10.1007/978-3-662-46706-0_27]

[16] Ramasubramanian K, Singh A. Deep learning using keras and TensorFlow. In: Ramasubramanian K, Singh A, eds. Machine Learning Using R. Berkeley: Apress, 2019. 667−688. [doi: 10.1007/978-1-4842-4215-5_11]

[17] Bergstra J, Komer B, Eliasmith C, Yamins D, Cox DD. Hyperopt: A python library for model selection and hyperparameter optimization. Computational Science & Discovery, 2015, 8(1): 014008. [doi: 10.1088/1749-4699/8/1/014008]

[18] Kölbl S, Leander G, Tiessen T. Observations on the SIMON block cipher family. In: Gennaro R, Robshaw M, eds. Advances in Cryptology. Berlin: Springer, 2015. 161−185. [doi: 10.1007/978-3-662-47989-6_8]

[19] De Moura L, Bjørner N. Z3: An efficient SMT solver. In: Ramakrishnan CR, Rehof J, eds. Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer, 2008. 337−340. [doi: 10.1007/978-3-540-78800-3_24]

[20] Li C, Sun B, Li RL. Attack Method and Case Analysis of Block Cipher. Beijing: Science China Press, 2010. 64−72 (in Chinese).

附中文参考文献:

[1] 张政馗, 庞为光, 谢文静, 吕鸣松, 王义. 面向实时应用的深度学习研究综述. 软件学报, 2020, 31(9): 2654−2677. http://www.jos.org.cn/1000-9825/5946.htm [doi: 10.13328/j.cnki.jos.005946]

[20] 李超, 孙兵, 李瑞林. 分组密码的攻击方法与实例分析. 北京: 科学出版社, 2010. 64−72.

侯泽洲(1998－), 男, 硕士生, 主要研究领域为深度学习技术在分组密码领域的应用.

任炯炯(1995－), 男, 博士, 讲师, 主要研究领域为对称密码设计与分析.

陈少真(1967－), 女, 博士, 教授, 博士生导师, 主要研究领域为密码学, 信息安全.