دوفصل‌نامه علمی منادی امنیت فضای تولید و تبادل اطلاعات (افتا)
سال ۱۴۰۲، جلد ۲۱، شماره ۱ (۶۶-۹۱)
https://monadi.isc.org.ir

بررسی جامع رویکردهای یادگیری عمیق در تحلیل تفاضلی رمزهای قالبی سبک‌وزن

ایمان میرزاعلی مازندرانی[۱]، نصور باقری*[۱،۲] و صادق صادقی[۲]

[۱] دانشکده مهندسی برق، گروه مهندسی مخابرات، دانشگاه تربیت دبیر شهید رجایی، تهران، ایران
[۲] دانشکده علوم ریاضی، دانشگاه تحصیلات تکمیلی علوم پایه زنجان، زنجان، ایران

اطلاعات مقاله

تاریخچه مقاله:
تاریخ دریافت: ۲۴ آذر ۱۴۰۲
تاریخ پذیرش: ۲۸ بهمن ۱۴۰۲
انتشار آنلاین: ۷ فروردین ۱۴۰۳

کلمات کلیدی:
رمز قالبی
تحلیل رمز
تمایزگر عصبی
حمله بازیابی کلید
یادگیری عمیق
شبکه عصبی

نوع مقاله: مروری

چکیده

با گسترش روزافزون استفاده از یادگیری عمیق و شبکه‌های عصبی در علوم مختلف و موفقیت‌های حاصل از آن، در سال ۲۰۱۹ شبکه‌های عصبی عمیق برای تحلیل رمز تفاضلی اتخاذ شدند و پس از آن توجه فزاینده‌ای به این زمینه از تحقیقات جلب شد. اکثر تحقیقات انجام شده بر روی بهبود و به‌کارگیری تمایزگرهای عصبی متمرکز هستند و مطالعات محدودی نیز در رابطه با اصول ذاتی و ویژگی‌های یادگرفته شده توسط تمایزگرهای عصبی صورت گرفته است. در این مطالعه با تمرکز بر روی سه رمز قالبی Speck، Simon و Simeck، به بررسی فرایند و روش تحلیل رمزهای قالبی با کمک یادگیری عمیق خواهیم پرداخت. در این میان، عوامل مؤثر و مؤلفه‌های موجود در جهت دسترسی به عملکرد بهتر، واکاوی و مقایسه خواهند شد. همچنین با تشریح حملات و مقایسه نتایج، به این سوال پاسخ خواهیم داد که آیا از شبکه‌های عصبی و یادگیری عمیق می‌توان به عنوان یک ابزار کارا برای تحلیل رمزهای قالبی استفاده نمود یا خیر.

۱ مقدمه

رمزنگاری شاخه‌ای از رمزشناسی و یک علم و فناوری حیاتی در عصر ارتباطات مدرن است. رمزنگاری اطلاعات حساس و مهم را به شیوه‌ای تبدیل می‌کند که تنها افراد مجاز به آن دسترسی دارند و افراد غیرمجاز نمی‌توانند آن را تفسیر کنند. از اهداف اصلی رمزنگاری می‌توان به حفظ محرمانگی، حریم خصوصی و امنیت اطلاعات اشاره کرد. در این فرایند، اطلاعات اصلی با استفاده از یک الگوریتم (الگوریتم رمزگذاری) و یک کلید (کلید رمزگذاری) تبدیل می‌شوند. تنها افرادی که دارای کلید رمزگشایی معتبر هستند، می‌توانند اطلاعات را به حالت اصلی بازگردانند. در رمزنگاری حرفه‌ای، نه‌تنها متن اصلی محافظت می‌شود، بلکه کلید رمزگذاری و رمزگشایی نیز مورد محافظت قرار می‌گیرد. این اصول امنیتی باعث می‌شوند که اطلاعات مهم در مقابل دسترسی غیرمجاز و حملات امنیتی محافظت شوند.

رمزنگاری متقارن رایج‌ترین روش برای تضمین محرمانه بودن یک پیام یا اطلاعات است. هنگامی که سیستم رمزنگاری از یک کلید یکسان برای رمزگذاری و رمزگشایی استفاده می‌کند، متقارن نامیده می‌شود. در بیشتر موارد، رمزگذاری داده‌ها با استفاده از الگوریتم‌های متقارن توصیه می‌شود. رمزنگاری متقارن به شرطی که طول کلید رمزگذاری به اندازه کافی بزرگ باشد، ایمن و سریع است. کلید رمزگذاری از طریق یک کانال امن یا روش‌های رمزنگاری نامتقارن مبادله می‌شود. رمزهای متقارن به طور کلی به دو زیردسته تقسیم می‌شوند: رمزهای جریانی و رمزهای قالبی. رمزهای قالبی متن اصلی را به دنباله‌هایی با اندازه ثابت از بیت‌ها به نام بلوک یا قالب تقسیم می‌کنند و به طور بالقوه مقداری لایه‌گذاری را اعمال می‌کنند. سپس روی هر بلوک روش رمزگذاری را اجرا می‌کنند.

ارزیابی امنیت یک الگوریتم رمزنگاری یک مسئله بسیار مهم است. در رمزنگاری متقارن، این امر معمولاً با ارزیابی مقاومت و انعطاف‌پذیری

*نویسنده مسئول
آدرس‌های رایانامه: iman.mirzaali@sru.ac.ir (ایمان میرزاعلی مازندرانی)، Nbagheri@sru.ac.ir (نصور باقری)، s.sadeghi@iasbs.ac.ir (صادق صادقی)

در برابر انواع حملات شناخته شده انجام می‌شود. حملات تفاضلی [۱] و خطی [۲] از اولین و مهم‌ترین حملات برای تحلیل رمزهای متقارن هستند. امنیت هر الگوریتم رمز طراحی‌شده در مرحله اول در برابر این حملات مورد ارزیابی قرار می‌گیرد.

یادگیری ماشین یک زیرشاخه از هوش مصنوعی است که در آن، مدل‌ها و الگوریتم‌ها طراحی می‌شوند تا کامپیوترها بتوانند از داده‌ها یاد بگیرند و بدون نیاز به برنامه‌ریزی صریح، وظایف و مسائل خاص را انجام دهند. هدف اصلی یادگیری ماشین، استخراج الگوها و اطلاعات مفهومی از داده‌ها است به گونه‌ای که مدل‌ها قادر به تعمیم آن‌ها به داده‌های جدید باشند و پیش‌بینی‌ها و تصمیم‌گیری‌های دقیقی انجام دهند. برای دستیابی به این هدف، مراحلی شامل جمع‌آوری داده‌ها، پیش‌پردازش داده‌ها، انتخاب مدل، آموزش مدل، ارزیابی مدل و استفاده از مدل صورت می‌گیرد. یادگیری عمیق یک شاخه از یادگیری ماشین است که از شبکه‌های عصبی عمیق استفاده می‌کند و در زمینه‌های مختلفی مانند طبقه‌بندی تصویر و ترجمه ماشینی به کار گرفته شده است. هدف این است که داده‌ها را بر اساس برچسب‌هایشان که در چندین کلاس قرار دارند، طبقه‌بندی کند.

یادگیری ماشین و به‌خصوص یادگیری عمیق به‌تازگی به دلیل پیشرفت‌های چشم‌گیر در حوزه‌های مهمی مانند بینایی ماشین، تشخیص گفتار و غیره، توجه زیادی را به خود جلب کرده‌اند. تحلیل رمز و یادگیری ماشین، روش‌ها و ویژگی‌های مشترک بسیاری دارند. در برخی جنبه‌ها، پیداکردن کلید یک الگوریتم رمزنگاری معادل پیداکردن مجموعه مناسبی از وزن‌های شبکه‌های عصبی در یک الگوریتم یادگیری عمیق است. اما با وجود این شباهت‌ها و پیشرفت‌های اخیر در ابزارها و مدل‌های یادگیری عمیق، پژوهشگران در کاربرد تکنیک‌های یادگیری ماشین در تحلیل رمز پیشرفت چندانی نداشته‌اند.

چندی پیش در سال ۲۰۱۹، یک نوع حمله جدید با کمک یادگیری ماشین به رمز قالبی Speck پیشنهاد شد. با داشتن مبانی آن در تحلیل رمز تفاضلی، ایده این است که زوج‌های متن رمزی را که به یک تفاضل متن اصلی ثابت تعلق دارند از زوج‌های تصادفی تمایز داد. برای این کار، از شبکه‌های عصبی استفاده می‌شود. سپس با استفاده از تمایزگر به دست آمده و ترکیب آن با روش‌های تحلیل رمز کلاسیک، حمله بازیابی کلید به الگوریتم رمز اعمال می‌شود [۳].

در حالی که واضح است که یادگیری ماشین به طور کامل جایگزین تحلیل رمز کلاسیک نخواهد شد، به ویژه از آنجایی که به تفاوت کاملاً واضحی در توزیع مسئله یادگیری نیاز دارد، این سوال مطرح می‌شود که چقدر این رویکرد عمومی است و تا چه حد می‌تواند کار یک تحلیل‌گر رمز را تکمیل کند. به عبارت دیگر، آیا می‌توانیم یادگیری ماشین را به عنوان ابزاری که به تحلیل رمز کمک می‌کند، شبیه به روشی که حل‌کننده‌های SAT [۱] [۴] و MILP [۲] [۵] انجام می‌دهند در نظر گرفت؟

در ادامه مقاله، مفاهیم و روش کار تحلیل تفاضلی (شامل تمایزگر عصبی و حمله بازیابی کلید) به کمک شبکه‌های عصبی و یادگیری عمیق روی رمزهای قالبی Speck، Simon و Simeck شرح داده خواهد شد و پاسخ سؤال و فرضیه‌های فوق روشن خواهد شد. همچنین واضح است که این مبحث از دو قسمت اصلی تحلیل رمز و یادگیری عمیق تشکیل شده و در این پژوهش تلاش بر این است که هر دو قسمت مورد بررسی قرار گیرند. در پایان نیز جمع‌بندی و پیشنهادهایی برای کارهای آتی ارائه می‌شود.

۲ رمزهای قالبی مورد بررسی و حمله تفاضلی

در این قسمت سه رمز قالبی که در ادامه بررسی روی آن‌ها صورت خواهد گرفت به صورت خلاصه معرفی خواهند شد و پس از معرفی تمایزگر تفاضلی، به فرایند تولید داده برای مدل یادگیری عمیق و یافتن تمایزگر عصبی به کمک آن خواهیم پرداخت.

۱.۲ شرح مختصر رمزهای قالبی Speck، Simon و Simeck

در سال ۲۰۱۳، آژانس امنیت ملی ایالات متحده (NSA[3]) با هدف تضمین امنیت در دستگاه‌های دارای محدودیت منابع، دو خانواده از رمزهای قالبی به نام‌های Speck و Simon را ارائه کرد [۶]. چند سال بعد از آن در سال ۲۰۱۵، برای توسعه رمزهای قالبی کارآمدتر با الهام از رمزهای Speck و Simon، خانواده رمز قالبی Simeck طراحی شد [۷]. در این مقاله از نسخه‌های با طول کلمه‌ی ۳۲ بیتی و طول کلید ۶۴ بیتی (با علامت گذاری Speck32/64) برای هر سه رمز مذکور استفاده خواهد شد.

تابع دور هر یک از این سه رمز قالبی، دو کلمه ۳۲ بیتی (x_i, y_i) و یک زیرکلید ۱۶ بیتی k_i که با استفاده از فرانمای کلید شبیه تابع دور از یک شاه‌کلید ۶۴ بیتی ساخته می‌شود را به عنوان ورودی می‌گیرد. حالت دور بعدی (x_{i+1}, y_{i+1}) برای رمزهای Speck، Simon و Simeck به ترتیب با روابط ۱، ۲ و ۳ محاسبه خواهند شد. $\oplus$ بیانگر عملگر یای انحصاری، $\&$ نشان‌دهنده عملگر AND، $\boxplus$ نمایانگر جمع پیمانه‌ای و دو نماد $\ggg$ و $\lll$ به ترتیب نشان‌دهنده چرخش بیتی به سمت راست و چپ هستند.

$$x_{i+1} = ((x_i \ggg 7) \boxplus y_i) \oplus k_i,$$
$$y_{i+1} = (y_i \lll 2) \oplus x_{i+1} \quad (1)$$

$$x_{i+1} = (x_i \lll 1) \& (x_i \lll 8) \oplus (x_i \lll 2),$$
$$y_{i+1} = x_i \quad (2)$$

$$x_{i+1} = (x_i \lll 5) \& x_i \oplus (x_i \lll 1),$$
$$y_{i+1} = x_i \quad (3)$$

[1] Satisfiability [2] Mixed-Integer linear programming

[3] National security agency

۲.۲ تحلیل رمز تفاضلی

در تحلیل رمز تفاضلی، هدف تحلیل‌گر ایجاد تمایز بین الگوریتم رمز از تابع جایگشت تصادفی با مطالعه دقیق خواص انتشار تفاضل متن اصلی است. برای تحلیل رمز تفاضلی، رمزنگاران بر جستجوی مسیر تفاضل $(\alpha \to \beta)$ با بهترین احتمال تفاضلی $DP(\alpha \to \beta)$ ممکن تمرکز می‌کنند. برای یک تابع رمزگذاری $E : F_2^n \leftarrow F_2^n$ داریم:

$$DP(\alpha \to \beta) = \frac{\#\{x \mid E(x \oplus \alpha) \oplus E(x) = \beta\}}{2^n} \quad (۴)$$

که $\#\{S\}$ در رابطه ۴ نشان‌دهنده تعداد اعضا در مجموعه S است. تابع رمز از تبدیل تفاضل $(\alpha \to \beta)$ با احتمال $DP(\alpha \to \beta)$ پیروی می‌کند، در حالی که این احتمال برای یک تابع جایگشت تصادفی 2^{-n} است. این ویژگی به مهاجم اجازه می‌دهد تا یک تمایزگر تفاضلی براساس احتمال تبدیل تفاضل $(\alpha \to \beta)$ بسازد. بر این اساس، هر زمان در اثر تفاضل ورودی مشخص، تفاضل خروجی به سمت خاصی گرایش داشته باشد، این گرایش می‌تواند طی فرآیند بازیابی کلید منجر به آشکار شدن برخی از بیت‌های کلید شود.

در مقاله [۸] روشی به نام تحلیل رمز تفاضلی چندگانه [۱] معرفی شد که برای انجام این کار از مجموعه تفاضل‌های Δ استفاده می‌نماید. برخلاف تحلیل رمز تفاضلی کلاسیک که بر بهترین مسیر تفاضلی متمرکز می‌شود، این روش هر مسیر تفاضلی موجود در $(\alpha_i \to \beta_j)$ که $\alpha_i, \beta_j \in \Delta$ هستند را در نظر می‌گیرد. تمایزگر تفاضلی خالص یک نوع از تحلیل رمز تفاضلی چندگانه است که هر تفاضل متن رمزی $\beta_j \in F_2^n$ را از یک تفاضل متن اصلی α در نظر می‌گیرد و برای یک جفت تفاضل (α, β_j)، تمایزگر آن را براساس رابطه ۵ طبقه‌بندی می‌کند.

$$\text{طبقه‌بندی} = \begin{cases} \text{تابع رمزنگاری} & \text{اگر } DP(\alpha_i \to \beta_j) > 2^{-n} \\ \text{تابع جایگشت تصادفی} & \text{در غیر این صورت} \end{cases} \quad (۵)$$

۳ تمایزگر عصبی

همان طور که بیان شد یک تمایزگر قادر است بین داده‌هایی که توسط یک الگوریتم ساختاردهی شده‌اند و داده‌های تصادفی تمایز ایجاد کند. در مقاله [۳] برای اولین بار نشان داده شد که می‌توان تمایزگرهایی بر اساس شبکه‌های عصبی ساخت. این تمایزگرها اغلب می‌توانند اطلاعات بیشتری را نسبت به تمایزگرهای بر پایه جدول توزیع تفاضل (DDT [۲]) کسب کنند و در مواردی از تمایزگرهای کلاسیک کارآمدتر باشند. برای این منظور از یک شبکه عصبی مصنوعی مدل‌سازی شده با الگوریتم یادگیری عمیق به نام تمایزگر عصبی (ND [۳]) استفاده شده است. این تمایزگر عصبی مبتنی بر حمله تفاضلی، اولین مدل یادگیری عمیق شناخته شده است که با موفقیت تحلیل رمز تفاضلی را بر روی رمزهای مدرن (فرای کاربرد در حملات کانال جانبی [۹، ۱۰]) انجام داد.

[1]Multiple differential cryptanalysis [2]Difference distribution table [3]Neural distinguisher

داده‌های ورودی به مدل پایه تمایزگر عصبی، زوج‌های متن رمزشده‌ای هستند که با اعمال r دور از الگوریتم رمز مورد نظر به زوج‌های متن اصلی به دست آمده‌اند. در این میان متن‌های رمزشده‌ای که حاصل از یک تفاضل متن اصلی (ورودی) معین و ثابت هستند به عنوان کلاس توزیع حقیقی با برچسب ۱ و آن‌هایی که حاصل از تفاضل ورودی تصادفی هستند به عنوان کلاس توزیع تصادفی با برچسب ۰ برای تولید مجموعه داده در نظر گرفته می‌شوند. هدف مدل ایجاد تمایزگر بین این دو توزیع، تبدیل مسئله تمایز به طبقه‌بندی و سپس استفاده از آن تمایزگر برای حمله بازیابی کلید است. بر این اساس تمایزگر عصبی ND، یک مدل یادگیری نظارت‌شده [۴] است.

فرایند تولید تمایزگر عصبی شامل دو مرحله است: مرحله تولید داده و مرحله آموزش. مرحله تولید داده شامل ایجاد نمونه‌های داده برای تمایزگرها می‌شود؛ در حالی که مرحله آموزش بر روی آموزش و یادگیری تمایزگر عصبی با استفاده از نمونه‌های تولید شده متمرکز است. در ادامه به ذکر جزییات دو مرحله‌ی مذکور خواهیم پرداخت.

۱.۳ تولید داده

مرحله تولید داده و شکل داده ورودی شبکه عصبی، یکی از مراحل مهم و تاثیرگذار در دقت و عملکرد تمایزگر عصبی است. در تمایزگرهای عصبی داده‌های ورودی به مدل یادگیری عمیق با استفاده از رمز کردن (متن رمزی) تعداد زیادی متن اصلی به دست می‌آیند (معمولاً 10^7 نمونه). بر این اساس از نگاه کلی تولید داده را می‌توان به دو دسته تقسیم‌بندی نمود. در دسته اول هر نمونه با استفاده از یک زوج متن رمزی و در دسته دوم هر نمونه با استفاده از k زوج متن رمزی تولید خواهد شد. در ادامه تمایزگر عصبی حاصل از به کارگیری روش تولید داده اول با نام تک–زوج [۵] (ND_1^{cp}) و دسته دوم با نام چند–زوج [۶] (ND_k^{cp}) ذکر خواهند شد. درنمادگذاری انجام‌شده، منظور از cp [۷] زوج متن رمزی و k نشان‌دهنده تعداد زوج متن رمزی به کار گرفته شده برای تولید یک نمونه از مجموعه‌داده هستند؛ بنابراین دسته اول به نوعی حالت خاصی از دسته دوم ($k = 1$) است.

برای تمایزگر ND_k^{cp} این تعبیر را می‌توان داشت که مدل یادگیری عمیق به دلیل دیدن و در نظر گرفتن چند زوج متن رمزی در کنار هم به عنوان یک نمونه، می‌تواند ویژگی‌های بیشتری از هر نمونه مجموعه داده را استخراج کند. این رویکرد توانایی به‌کارگیری وابستگی‌های بین زوج‌های متن رمزی که با استفاده از یک کلید مشابه تولید شده‌اند را به مدل یادگیری می‌دهد. بنابراین، اطلاعاتی که ND_k^{cp} می‌تواند از نمونه‌های مجموعه داده ورودی خود استخراج کند، ممکن است تفاضل بین k زوج متن رمزی و ارتباطات بین خود متن‌های رمزی یا زوج‌های متن رمزی باشد؛ در نتیجه استفاده از این نوع تولید داده مطلوب‌تر است و باعث عملکرد بهتر تمایزگر عصبی می‌شود.

[4]Supervised model [5]Single-pair [6]Multi-pair [7]Ciphertext pair

در مرحله تولید داده، ابتدا N زوج متن اصلی $(P_{i,۰}, P_{i,۱}), i \in [۰, N-۱]$ تولید می‌شوند. نحوه تولید زوج‌های متن اصلی به این صورت است که متن اول یا همان $P_{i,۰}$ به طور تصادفی تولید شده و سپس برای نیمی از زوج‌ها، $P_{i,۱}$ با اعمال عملگر XOR به $P_{i,۰}$ متناظر و تفاضل ورودی معین α مطابق رابطه ۶ به دست می‌آید؛

$$P_{i,۱} = P_{i,۰} \oplus \alpha, \quad i \in \left[۰, \frac{N-۱}{۲}\right] \qquad (۶)$$

در حالی که نیمه باقی‌مانده مقادیر $P_{i,۱}$ به صورت تصادفی تولید خواهند شد. این عمل تعادل بین تعداد نمونه‌های دو کلاس توزیع حقیقی و توزیع تصادفی را برای یادگیری بهتر تمایزگر عصبی حفظ می‌کند. توجه شود که در ابتدا تمامی مقادیر $P_{i,۰}$ به صورت تصادفی تولید می‌شوند. سپس N زوج متن اصلی به دست آمده براساس تفاضل‌هایشان گروه‌بندی می‌شوند که هر گروه شامل ۱ زوج متن اصلی برای تمایزگر تک-زوج $(ND_۱^{cp})$ و شامل k زوج متن اصلی برای تمایزگر چند-زوج (ND_k^{cp}) است. برای یک گروه با نمایه j که تفاضل ورودی بین زوج‌های متن اصلی آن برابر مقدار α باشد، برچسب Y_j متناظر با آن مقدار ۱ را خواهد گرفت و در غیر این صورت مقدار برچسب آن برابر با صفر می‌شود. این گروه‌بندی و برچسب‌زنی را می‌توان به صورت رابطه ۷ نوشت.

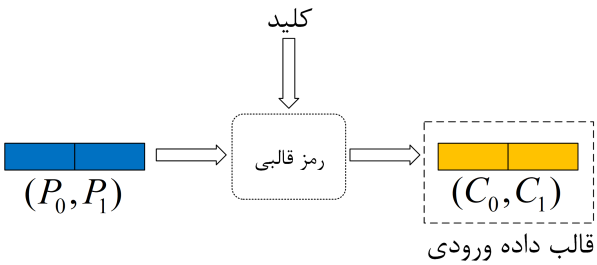
$$Y_j = \begin{cases} ۱ & \text{اگر } P_{j,۰} \oplus P_{j,۱} = \alpha, j \in [۰, k-۱] \\ ۰ & \text{در غیر این صورت} \end{cases} \qquad (۷)$$

در گام بعدی تمامی زوج‌های متن اصلی توسط الگوریتم رمزنگاری E برای به دست آوردن زوج‌های متن رمزی متناظر، به تعداد r دور طبق رابطه ۸ رمز می‌شوند.

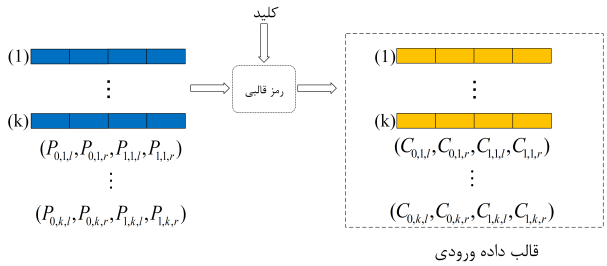
$$(C_{i,۰}, C_{i,۱}) = (E(P_{i,۰}), E(P_{i,۱})), \quad i \in [۰, N-۱] \qquad (۸)$$

در نهایت همه گروه‌های متن رمزی به همراه برچسب‌هایشان جمع‌آوری می‌شوند و مجموعه‌داده براساس آن تولید می‌شود. شکل ۱ نشان‌دهنده تولید داده ورودی در مقاله [۳] با یک زوج متن رمزی به ازای هر نمونه $(ND_۱^{cp})$ است. در این شکل هر یک از زوج‌های متن اصلی $(P_۰, P_۱)$ با یک شاه‌کلید تصادفی برای به دست آوردن یک زوج متن رمزی $(C_۰, C_۱)$ به عنوان یک نمونه آموزشی رمز می‌شوند. در هنگام آموزش شبکه عصبی، هر نمونه یک برچسب با مقادیر ۰ و یا ۱ را می‌گیرد. مقدار ۱ به این معناست که زوج‌های داده از رمز کردن $(P_۰, P_۱)$ با تفاضل ورودی α تولیده شده‌اند و مقدار ۰ به این معنی است که زوج‌های داده از زوج تصادفی تولید شده‌اند.

شکل ۲ نیز نشان‌دهنده تولید داده با چند زوج متن رمزی به ازای هر نمونه (ND_k^{cp}) است. در شکل ۲، k زوج متن اصلی $\{(P_{۰,۱}, P_{۱,۱}), \ldots, (P_{۰,k}, P_{۱,k})\}$ (که در شکل هر زوج با در نظر گرفتن ۲ واژه سمت چپ با زیروند l و سمت راست با زیروند r برای هر متن، به ۴ واژه تبدیل شده است) با یک شاه‌کلید تصادفی برای به دست آوردن k زوج متن رمزی $\{(C_{۰,۱}, C_{۱,۱}), \ldots, (C_{۰,k}, C_{۱,k})\}$ به عنوان یک نمونه آموزشی رمز می‌شوند. در این‌جا نیز برچسب ۱ به



شکل ۱. تولید داده با زوج متن رمز تکی [۱۱]

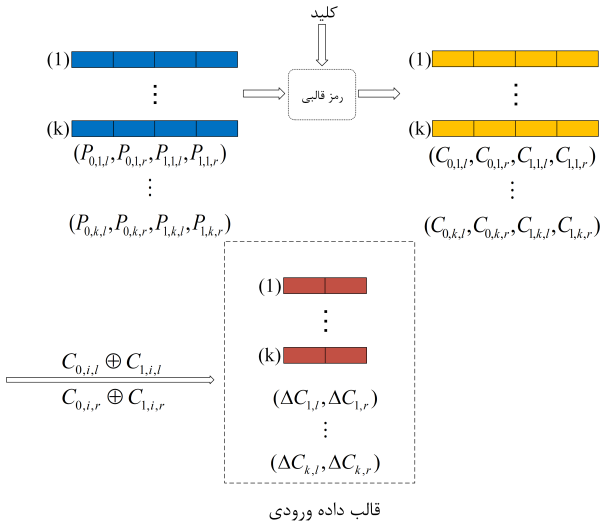


شکل ۲. تولید داده با زوج متن رمز چندگانه [۱۱]

نمونه‌های حاصل از تفاضل ورودی معین و برچسب ۰ به نمونه‌های حاصل از تفاضل ورودی تصادفی تعلق می‌گیرند.

در مقاله [۳] داده‌های تولید شده برای آموزش تمایزگر عصبی از دسته تک-زوج بودند، اما به طور ضمنی ایده استفاده از نمونه‌های حاوی چند زوج نیز در همین مقاله داده شده بود؛ تا این که در مقاله [۱۲] این ایده به صورت عملی استفاده شد و مورد بررسی قرار گرفت. پارامتر مهم تمایزگرهای چند-زوج اندازه k می‌باشد که معمولا از بین مقادیر $\{۲, ۴, ۸, ۱۶\}$ انتخاب می‌شود. با آزمایشات صورت‌گرفته در مقاله [۱۲] مشخص شد که تمایزگرهای چند-زوج دقت بالاتری نسبت به تمایزگرهای عصبی تک-زوج دارند و پس از آن در تحقیقات بعدی از این نوع تمایزگرها استفاده شد. اگر توزیع هدف بسیار متفاوت از توزیع یکنواخت باشد، با افزایش k، ویژگی‌های بیشتری در بین زوج‌های متن رمزشده وجود خواهد داشت. البته باید در نظر داشت که به ازای تعداد نمونه یکسان، استفاده مستقیم از چند زوج برای تولید نمونه، موجب k برابر شدن پیچیدگی داده می‌شود و بنابراین مقایسه مستقیم تمایزگرهای حاصل از این دو حالت متفاوت، منصفانه نیست. برای کنترل کردن این موضوع، طرحی برای استفاده مجدد از داده با استفاده از محاسبه تکرار یک زوج در گروه‌های مختلف و نیز شباهت گروه‌های مختلف ارائه شد [۱۲]. با تعیین یک کران بالا برای این دو معیار، گروه‌هایی که مقدار این دو معیار برای نمونه‌هایشان پایین‌تر از کران بالایی باشند، انتخاب خواهند شد.

قالب داده‌های تولید شده را از چند منظر دیگر می‌توان مورد بررسی قرار داد. همان‌طور که اشاره شد، تمایزگر پایه $ND_۱^{cp}$ ارائه شده در مقاله [۳]، زوج متن رمزشده $(C_{i,۰}, C_{i,۱})$ را به عنوان ورودی به تمایزگر عصبی می‌دهد. با استفاده از عملگر XOR و محاسبه تفاضل هر زوج متن رمزی $(C_{i,۰} \oplus C_{i,۱})$، می‌توان مقدار تفاضل خروجی را به صورت تنها یا



شکل ۳. تولید داده با تفاضل‌های خروجی چندگانه [۱۱]

همراه با زوج متن رمزی متناظر آن تفاضل، به عنوان نمونه ورودی مدل یادگیری در نظر گرفت. بر این اساس، در مقاله [۱۳]، ایده استفاده از k تفاضل زوج متن رمزی به عنوان یک نمونه در تولید مجموعه‌داده استفاده شد. در این حالت، تمایزگر را با نام ND_k^{cd} نشان می‌دهیم (نماد cd نشان‌دهنده تفاضل متن رمزی است) و نمونه jام مجموعه‌داده آموزشی آن با توجه به رابطه ۹ مشخص می‌شود (علامت $*$ نشان‌دهنده ضرب است).

$$((C_{(j-1)*k,0} \oplus C_{(j-1)*k,1},\ldots,C_{j*(k-1),0} \oplus C_{j*(k-1),1}),Y_j) \quad (۹)$$

با تبدیل k زوج متن رمزی به تفاضل‌های متناظر با آن‌ها در هر نمونه، شکل ۳ روش تولید داده را برای تمایزگر ND_k^{cd} نشان می‌دهد. در این‌جا نیز هر یک از متن‌های اصلی و رمزی با نمادگذاری زیروند l و r به دو واژه سمت چپ و سمت راست تبدیل شده و نمونه ورودی به مدل با محاسبه تفاضل بین واژه‌های سمت راست و سمت چپ متن رمزی $\{(\Delta C_{1,l},\Delta C_{1,r}),\ldots,(\Delta C_{k,l},\Delta C_{k,r})\}$ به دست خواهد آمد. با در نظر گرفتن $k=1$ نیز تمایزگر با ورودی یک تفاضل زوج متن رمزی به ازای هر نمونه را خواهیم داشت (ND_1^{cd}). به صورت مشابه اگر نمونه‌ها شامل k زوج متن رمزی باشند (تمایزگر ND_k^{cp})، نمونه jام را می‌توان طبق رابطه ۱۰ نشان داد.

$$((C_{(j-1)*k,0},C_{(j-1)*k,1},\ldots,C_{j*(k-1),0},C_{j*(k-1),1}),Y_j) \quad (۱۰)$$

در مقایسه تمایزگرهای عصبی با ورودی تفاضل و یا زوج متن رمزی (ND_k^{cp} و ND_k^{cd}) می‌توان گفت که هر دو عملکرد نسبتاً مشابهی با یکدیگر دارند؛ برای مثال، تفاضل منقطع [1] دوره‌های قبلی، تاثیر یکسانی بر هر دو دارد. با وجود این که باید زوج متن رمزی حاوی ویژگی و اطلاعات بیشتری نسبت به تفاضل بینشان باشد، اما تمایزگر با ورودی زوج متن رمزی ND_k^{cp}، تنها به مقدار کمی (حدود ۰٫۰۳) دقت بالاتری دارد. در نتیجه ویژگی‌های اضافی در زوج متن رمزی به سادگی برای مدل قابل تشخیص نیستند.

تاکنون انواع اصلی روش‌های تولید داده آموزشی برای مدل یادگیری و ساخت تمایزگر مبتنی بر آن ارائه شدند. در تحلیل رمز با استفاده از یادگیری ماشین، نرخ موفقیت و نیز پیچیدگی داده مورد نیاز برای حمله بازیابی کلید که در قسمت‌های بعدی به تفصیل به آن پرداخته خواهد شد، کاملاً وابسته به دقت تمایزگر عصبی هستند. یکی از راه‌های افزایش دقت تمایزگرها نیز فراهم کردن ویژگی‌های بیشتر و مناسب‌تر برای آن‌ها می‌باشد. علاوه بر راهکارهای اشاره شده در این قسمت برای افزایش ویژگی‌ها و به سبب آن افزایش دقت تمایزگر عصبی، برخی مطالعات تلاش نمودند تا اطلاعات و ویژگی‌هایی از دورهای ماقبل آخر الگوریتم رمز مورد بررسی را به نمونه‌های مجموعه‌داده ورودی اضافه نمایند.

به عنوان نمونه در مقاله [۱۱]، نویسندگان با فرض مارکوف بودن رمز [2] و مستقل و تصادفی بودن زیرکلیدها، با استفاده از زیرکلیدهای تصادفی، متن‌های رمزی را یک دور رمزگشایی کردند و یک خروجی غیرحقیقی برای دور یکی مانده به آخر از الگوریتم رمزهای Speck و Simon به دست آوردند. سپس با کنار هم قرار دادن متن‌های رمزی حقیقی و خروجی‌های غیرحقیقی حاصل از یک دور رمزگشایی با زیرکلید تصادفی، به تولید مجموعه‌داده آموزشی پرداختند. همچنین با XOR کردن زوج‌های حقیقی و زوج‌های غیرحقیقی و به دست آوردن تفاضل‌های هر کدام، با کنار هم قرار دادن این تفاضل‌ها یک مجموعه‌داده دیگر به دست آوردند. پس از آن نیز با به‌کارگیری ایده تمایزگرهای ND_k^{cd} و ND_k^{cp}، از k زوج برای تولید هر نمونه استفاده کردند. در نتیجه دقت بالاتری نسبت به تمایزگرهای قبلی برای انواع رمزهای خانواده‌های Speck و Simon به دست آوردند.

در مقاله [۱۴] نیز با به‌کارگیری ایده استفاده از ویژگی‌ها و اطلاعات دور ماقبل آخر رمز Speck، علاوه بر زوج متن رمزی خروجی، زوج سمت راست و تفاضل زوج سمت چپ ورودی دور آخر را به نمونه‌های مجموعه‌داده خود اضافه کردند. با توجه به ساختار رمز Speck، با داشتن زوج متن رمزی خروجی، می‌توان با XOR کردن سمت راست و سمت چپ هر متن رمزشده و چرخش آن به اندازه ۲ بیت به سمت راست، زوج ورودی سمت راست دور آخر (y_0,y_1) را بدون دخالت زیرکلید محاسبه کرد. اما برای محاسبه تفاضل ورودی سمت چپ دور آخر نمی‌توان بدون داشتن زیرکلید این کار را انجام داد.

با توجه به این موضوع و محاسبات انجام‌شده در مقاله [۱۴]، مشخص شد که برای محاسبه مقدار تفاضل سمت چپ زوج ورودی دور آخر، تنها بیت‌های اول و دوم کلید دخالت دارند و تاثیر گذارند. بنابراین با محاسبه $((C_{0,l} \boxminus y_0) \oplus (C_{1,l} \boxminus y_1)) \lll 7$ (که $(C_{0,l},C_{1,l})$ واژه‌های سمت چپ زوج متن رمزی هستند) می‌توان بیت‌هایی از مقدار تفاضل ورودی سمت چپ دور آخر را که از زیرکلید دور تاثیر نمی‌پذیرند، به دست آورد و به عنوان ویژگی برای دست‌یابی به دقت تمایز بالاتر، به همراه زوج ورودی سمت راست دور آخر، به نمونه‌های مجموعه‌داده اضافه

[1] Truncated differential

[2] Markov cipher assumption

کرد (نماد ⊟ نشان‌دهنده تفریق پیمانه‌ای است). این افزایش دقت حدود ۲٫۳۴ درصد با استفاده از معماری یکسان است. باید توجه داشت که مقدار محاسبه شده بدون دخالت زیرکلید، می‌تواند بخشی از بیت‌های مقدار ورودی حقیقی را بگیرد و تعداد بیت‌های گرفته‌شده در هر نمونه متفاوت است. مسلم است که دقت شبکه عصبی با جذب بیت‌های بیشتری از مقدار حقیقی بهبود خواهد یافت.

برای رمزهای Simon و Simeck نیز با توجه به ساختار آن‌ها، به طور کلی میتوان بدون دخالت کلید و به صورت قطعی زوج ورودی سمت چپ دور آخر و نیز تفاضل زوج ورودی سمت راست دور آخر را به دست آورد و همچنین به صورت جزیی، تفاضل ورودی سمت راست دور ماقبل آخر را محاسبه کرد و در قالب داده ورودی، آن‌ها را در نظر گرفت. با اضافه کردن این سه ویژگی به قالب داده ورودی در مقاله [۱۵]، بهترین تمایزگر عصبی–تفاضلی ۱۲ دوری برای رمز Simeck32/64 حاصل شده است.

در مقاله‌های [۱۶، ۱۷] مسئله تمایز بین متن‌های رمزی حاصل از یک تفاضل ورودی معین و متن‌های رمزی حاصل از یک تفاضل ورودی تصادفی، به مسئله تمایز بین متن‌های رمزی حاصل از t تفاضل ورودی (تفاضل متن اصلی) متفاوت و معین تبدیل شد و یک تمایزگر جدید با یک مسئله طبقه‌بندی دارای t کلاس ارائه گردید. برای تولید نمونه‌های آموزشی این تمایزگر، ابتدا به تعداد t تفاضل ورودی مناسب انتخاب می‌شود و برای هر کلاس، متن اصلی $P_{i,0}$ به صورت تصادفی انتخاب شده و زوج آن با XOR کردن $P_{i,0}$ با تفاضل tام ساخته می‌شود. برچسب هر نمونه با توجه به تفاضل ورودی آن یک مقدار از ۰ تا $t-1$ می‌باشد. پس از رمز کردن هر زوج متن اصلی و گرفتن متن رمزی، تفاضل هر زوج متن رمزی محاسبه شده و هر تفاضل به همراه برچسب متناظر خود تشکیل یک نمونه آموزشی می‌دهد. پس در این مقاله به نوعی از قالب ورودی تمایزگر ND_1^{cd} استفاده شده است. تفاوت دیگر این نوع تمایزگر در حداقل دقت مدل آن می‌باشد؛ به صورتی که حداقل دقت قابل قبول برای ایجاد تمایز با توجه به تعداد t کلاس، مقدار $\frac{1}{t}$ می‌باشد.

۲.۳ مرحله آموزش

در مرحله آموزش ابتدا مطابق مباحث مطرح‌شده در بخش تولید داده، داده‌های آموزش و اعتبارسنجی [1] تولید می‌شوند. تعداد آن‌ها را به ترتیب برابر با M و N در نظر می‌گیریم. در این صورت برای قالب‌های ND_k^{cp} و ND_k^{cd} تعداد $\frac{N}{2k}$ نمونه مثبت (با برچسب ۱) و $\frac{N}{2k}$ نمونه منفی (با برچسب ۰) برای مجموعه‌داده آموزشی خواهیم داشت. این مقادیر برای مجموعه‌داده اعتبارسنجی به ترتیب برابر با $\frac{M}{2k}$ و $\frac{M}{2k}$ می‌باشند. سپس شبکه عصبی مدنظر با مجموعه‌داده آموزشی و اعتبارسنجی به‌دست‌آمده تغذیه می‌شود و یادگیری را انجام می‌دهد. اگر دقت اعتبارسنجی مدل ND_k^{cp} و یا ND_k^{cd} بیشتر از مقدار ۰٫۵ باشد (برای تمایزگر [۱۶، ۱۷] همان‌طور که اشاره شد مقدار $\frac{1}{t}$)، یک تمایزگر عصبی r دوری به دست می‌آید، در غیر این صورت تمایزگر معتبر نیست.

[1] Validation dataset

در صورت حاصل نشدن تمایزگر عصبی معتبر، می‌توان مقدار α را تغییر داد و فرایند آموزش را دوباره تکرار کرد. با در نظر گرفتن یک تمایزگر عصبی r دوری معتبر، می‌توان مدل تمایزگر عصبی–تفاضلی را بر اساس رابطه ۱۱ توصیف نمود.

$$\begin{aligned}&\Pr(Y=1 \mid X_0, X_1, \ldots, X_{n-1})\\&= F(f(X_0), f(X_1), \ldots, f(X_{n-1}),\\&\quad \phi(f(X_0)), \phi(f(X_1)), \ldots, \phi(f(X_{m-1}))\\&X_i = (C_{i,0}, C_{i,1}), \quad i \in [0, n-1]\end{aligned} \tag{11}$$

این مدل با استفاده از احتمال $\Pr(Y=1 \mid X_0, X_1, \ldots, X_{n-1})$ عمل می‌کند که به وسیله تابع $F(\cdot)$ به عنوان تخمین‌زننده‌ی احتمال پسین جدید، براساس ویژگی‌های اصلی $f(X_i)$ و ویژگی‌های مشتق‌شده $\phi(f(X_i))$ از زوج‌های متن رمزی X_i، محاسبه می‌شود. هر X_i به صورت زوج $(C_{i,0}, C_{i,1})$ تعریف می‌شود. برای کسب جزییات و اطلاعات بیشتر در مورد روابط حاکم بر آموزش تمایزگر عصبی به مرجع [۱۸] مراجعه شود. در بخش ۴ نیز پس از بررسی ساختارهای مختلف تمایزگرهای عصبی، انواع روش‌های آموزش برای این تمایزگرها نیز معرفی و شرح داده خواهند شد.

۳.۳ انتخاب تفاضل ورودی

تحقیقات انجام‌شده نشان می‌دهند که تفاضل ورودی تاثیر به‌سزایی در دقت هر دو تمایزگر کلاسیک بر مبنای جدول توزیع تفاضل و تمایزگر عصبی دارد [۱۹]. آزمایش‌های ارائه شده در مقاله [۲۰] نشان داد که تفاضل ورودی بهترین مشخصه تفاضلی، حداقل برای رمز Speck، انتخاب خوبی برای تمایزگرهای عصبی نیست. برای مثال تفاضل ورودی بهترین مشخصه تفاضلی ۵ دوری برای رمز Speck، مقدار 0x2400/0020 است ولی با استفاده از این تفاضل نتیجه مطلوبی در تمایزگر عصبی حاصل نخواهد شد (مقدار تفاضل در مبنای ۱۶ و با نماد 0x نشان داده شده و نماد / جداکننده تفاضل کلمه‌های سمت چپ و سمت راست متن اصلی و یا متن رمزی است). همچنین با توجه به نتایج دیگر تحقیقات می‌توان گفت که این موضوع برای رمزهای دیگر نیز صادق خواهد بود؛ پس نیاز است که عوامل دخیل در انتخاب تفاضل ورودی مناسب بررسی و تحلیل شوند.

فرضیه مطرح شده در مقاله [۲۰] برای این اختلاف، مرتبط با تفاضل منقطع است. نتیجه این فرضیه بیان می‌کند که مدل یادگیری تمایزگر عصبی ۵ دوری برای رمز Speck، یادگیری خود را بر مبنای تفاضل‌های منقطع ۳ و ۴ دوری و بیت‌های دارای اریبی بالاتر در آن‌ها انجام می‌دهد.

در تمایزگر پایه مطرح‌شده در مقاله [۳]، مقدار تفاضل ورودی برای الگوریتم رمز قالبی Speck32/64 برابر با 0x0040/0000 در نظر گرفته شد. دلیل این انتخاب، پایین بودن وزن همینگ و نیز بهینه‌بودن آن از دیدگاه احتمال بوده است. پس در انتخاب تفاضل ورودی برای تحلیل رمز با یادگیری عمیق باید دو معیار پایین بودن وزن همینگ (که به احتمال زیاد منجر به پایین بودن وزن همینگ تفاضل خروجی خواهد شد) و نیز

جدول ۱. مروری بر تفاضل‌های ورودی استفاده‌شده

تفاضل ورودی مناسب	رمز
0x0040/0000	Speck32/64
0x0000/0040	Simon32/64
0x0000/0040	Simeck32/64

بالا بودن احتمال رخداد مشخصه‌های تفاضلی آن در نظر گرفته شوند.

بر این اساس، می‌توان تفاضل‌هایی را در نظر گرفت که در تحقیقات قبلی متعلق به مشخصه تفاضلی دارای احتمال بالاگزارش شده‌اند. همچنین می‌توان تفاضل‌هایی را امتحان کرد که منجر به کمترین پراکنش (انتشار) در چند دور اول شوند [۱۹].

برای رمزهای شبه Simon نیز (در اینجا Simon و Simeck)، با توجه به مقالات مختلف، تفاضل ورودی بهینه تمایزگر، معمولاً برابر با مقدار 0x0000/0040 در نظر گرفته شده است و در ادامه حملات تمایزگر و بازیابی کلید با این تفاضل ورودی تمایزگر عصبی ارائه شده‌اند.

در مقاله [۲۱] تاثیر تفاضل ورودی تمایزگرهای عصبی با وزن همینگ کم‌تر از ۳ بر روی به دست آوردن تمایزگرهای ترکیبی (HD [1])، که در بخش حمله بازیابی کلید مورداستفاده قرار خواهند گرفت، بررسی شد و نشان داده شد که تفاضل به صورت $(0, e_i)$ (در این‌جا e_i بردارهای پایه استاندارد هستند) برای رمزهای شبه Simon انتخاب مناسبی می‌باشد؛ در نتیجه در مقدار تفاضل ورودی فقط یک بیت فعال خواهیم داشت. در جدول ۱ به صورت خلاصه تفاضل ورودی مناسب برای سه رمز مورد بررسی در این مقاله نشان داده شده است. مقادیر چرخش یافته یا همان تفاضل‌های ورودی معادل آن‌ها نیز می‌توانند انتخاب خوبی برای تفاضل ورودی باشند.

بخشی از مطالعه انجام‌شده در مقاله [۲۲] به ارائه یک الگوریتم تکاملی برای جستجوی خودکار تفاضل‌های ورودی خوب پرداخته است که مقیاس‌پذیر و قابل توصیف می‌باشد. در مقاله [۲۲] سعی بر آن است که روشی برای خودکارسازی فرایند ایجاد تمایزگر عصبی ارائه شود. دو مشکل اساسی در خودکارسازی این فرایند، انتخاب معماری و پارامترهای خود تمایزگر عصبی و دیگری تشخیص تفاضل ورودی مناسب می‌باشند.

در مقاله [۳] یک روش برای یافتن تفاضل ورودی بیان شد که با موفقیت برای رمز Speck32/64 این کار را انجام داد؛ اما تطبیق و به‌کارگیری آن روش در رمزهای دیگر کار آسانی نیست. در این روش، یک شبکه تک بلوکی (عمق ۱) با تفاضل ورودی تصادفی (اما ثابت) δ روی ۳ دور از رمز Speck با 2^{10} زوج متن رمزی آموزش داده می‌شود. سپس خروجی لایه ماقبل آخر این شبکه به عنوان ورودی برای آموزش یک طبقه‌بند رگرسیون ریج [2] بر روی تعداد کمی از نمونه‌ها برای تفاضل‌های جدید استفاده می‌شود. سپس یک الگوریتم حریص [3] برای پیشنهاد کاندیدهای جدید δ استفاده می‌شود. بنا به دلایل مطرح شده فوق، در مقاله [۲۲] یک روش خودکار برای پیدا کردن تفاضل ورودی مناسب شامل یک امتیاز اریبی برای رتبه‌بندی تفاضل‌های ورودی و یک بهینه‌ساز تکاملی که از این رتبه‌بندی استفاده می‌کند، ارائه شد.

این روش بیان می‌کند که تفاضل ورودی مناسب برای تمایزگرهای عصبی، تفاضلی است که امتیاز اریبی تعریف شده در رابطه ۱۲ را بیشینه کند. طبق این تعریف، امتیاز اریبی تقریبی با محاسبه t نمونه برای تفاضل ورودی α برابر است با مجموع اریبی‌های هر بیت jام در تفاضل خروجی.

$$b^t(\alpha) = \sum_{j=0}^{n-1} \left| 2 \cdot \frac{\sum_{i=0}^{t} \sum (E_K(X) \oplus E_K(X \oplus \alpha))_j}{2^{n+k}} - 1 \right| \quad (۱۲)$$

برای محاسبه سریع‌تر اریبی و جستجوی کاندیدهای بیشتر تفاضل ورودی، بهینه‌ساز تکاملی در مقاله [۲۲] ارائه شد. در این الگوریتم، ابتدا ۱۰۲۴ تفاضل ورودی تصادفی ایجاد می‌شوند و سپس با توجه به امتیاز تقریبی، ۳۲ تفاوت ورودی بهتر انتخاب می‌شوند. در هر دور، تفاضل‌های جدیدی از ترکیب تفاضل‌های فعلی ساخته می‌شوند و پس از ۵۰ دور، ۳۲ تفاضل ورودی با بالاترین امتیاز برگشت داده می‌شوند. البته این نکته قابل ذکر است که با انجام این روش تعداد زیادی تفاضل به عنوان خروجی به ما داده خواهد شد که ممکن است بسیاری از آن‌ها امتیازی نزدیک به هم داشته باشند و در این حالت انتخاب تفاضل بهینه دشوار خواهد بود. بر این اساس باید یک سطح آستانه در نظر گرفت و براساس آن فقط تعدادی از تفاضل‌ها را برگزید و باید به همان تعداد تمایزگر عصبی آموزش داده شود.

۴ یادگیری عمیق و شبکه عصبی

در این بخش، ابتدا مفاهیم یادگیری عمیق و شبکه‌های عصبی به صورت خلاصه مرور خواهند شد و سپس به بررسی معماری مدل پایه تمایزگرهای عصبی و پارامترهای آن خواهیم پرداخت. همچنین روش‌های مختلف آموزش مدل‌های تمایزگرهای عصبی در این بخش معرفی خواهند شد.

یک شبکه عصبی یک تابع $F : R_n \to S$ است که برای یک ورودی $x \in R_n$، یک بردار امتیاز $y \in R_\sigma$ تولید می‌کند، به‌طوری که جزء iام این بردار یک عدد حقیقی مثبت است که مقدار اطمینان ما نسبت به اینکه ورودی x از کلاس i می‌آید، است. برای اندازه‌گیری عملکرد شبکه، می‌توانیم یک تابع خطا تعریف کنیم که اندازه‌گیری میزان اختلاف پیش‌بینی‌های ما از داده‌های واقعی است. انتخاب رایج برای تابع خطا، فاصله اقلیدسی (رابطه ۱۳) است.

$$E(x) = \|C(x) - F(x)\|_2 \quad (۱۳)$$

برای اندازه‌گیری خطای شبکه روی کل مجموعه‌ی داده، یک تابع تلفات یا هزینه [4] تعریف می‌شود که به طور ساده طبق رابطه ۱۴، میانگین تابع خطا روی کل مجموعه‌ی داده است.

[1] Hybrid distinguisher [2] Ridge regression [3] Greedy

[4] Loss function

$$\mathcal{L} = \frac{1}{l}\sum_{i=1}^{l} E(x_i) \qquad (۱۴)$$

در اینجا، l تعداد اعضای مجموعه‌ی داده (اندازه مجموعه‌ی داده) است. همراه با تابع هزینه، معیار دیگری به نام دقت [۱] تعریف می‌شود که در این حالت به‌عنوان نسبتی تعریف می‌شود که داده‌های ما به درستی طبقه‌بندی شده‌اند، یعنی نسبت x_iهایی که برای آن‌ها رابطه زیر برقرار است:

$$y(x_i) = \text{argmax}\{F(x_i)\} \qquad (۱۵)$$

در رابطه ۱۵، argmax تابعی است که شاخص حداکثر مقدار را در یک دنباله یا مجموعه به ما می‌دهد و با استفاده از آن می‌توانیم برای یک مجموعه‌ی داده، مقدار بیشینه یک ویژگی خاص را پیدا کنیم. به عبارت دیگر، دقت شبکه معادل تعداد نمونه‌هایی از داده‌های ما است که به‌درستی به کلاس‌های مختلف تعلق می‌گیرند نسبت به کل تعداد نمونه‌هایی که درمجموعه‌ی داده هستند. هدف اصلی یک شبکه عصبی، بهینه‌سازی یا کمینه‌کردن تابع هزینه $\mathcal{L}$ بر روی هر مجموعه ورودی ممکن است و این کار با درنظرگرفتن تابع $\mathcal{L}$ به‌عنوان یک تابع از برخی از پارامترها به نام θ، که پارامترهای قابل‌آموزش شبکه هستند، انجام می‌شود. مرحله آموزش شبکه عصبی می‌تواند به‌عنوان یک مسئله بهینه‌سازی عددی دیده شود که در آن تابع $\mathcal{L}(\theta)$ با استفاده از روشی تکرارشونده به نام بهینه‌سازی شبکه، کمینه می‌شود. معروف‌ترین روش بهینه‌سازی، گرادیان نزولی است که در آن به‌صورت تکرارشونده پارامترهای θ را مطابق رابطه ۱۶ به‌روزرسانی می‌کند.

$$\theta^{(t+1)} = \theta^{(t)} - a\nabla\mathcal{L}(\theta^{(t)}) \qquad (۱۶)$$

در اینجا گرادیان با نماد ∇ نشان داده شده است و a به عنوان نرخ یادگیری شناخته می‌شود که باید به عنوان سرعت روند آموزش در نظر گرفته شود. مقادیر بزرگ‌تر a به طور کلی منجر به آموزش‌های سریع‌تر می‌شوند، درحالی‌که مقادیر کوچک‌تر به آموزش‌های کندتر اما دقیق‌تر منتهی می‌شوند. به دلیل این که شبکه‌های عصبی عمیق معمولاً از تعداد زیادی لایه تشکیل شده‌اند، گرادیان نزولی در عمل به طور کند انجام می‌شود و با افزایش ابعاد مجموعه‌ی داده، ممکن است دقت آن نیز ناپایدار شود. برای جلوگیری از این مشکلات، تکنیک‌های بسیاری معرفی شده‌اند که پارامترها را در یک زیرمجموعه کوچک از مجموعه‌ی داده اولیه به‌روزرسانی می‌کنند؛ سپس مجموعه‌ی داده با همان ترتیبی که بوده است به‌هم‌ریخته شده و فرایند بر روی زیرمجموعه‌های مختلف تکرار می‌شود. ابعاد زیرمجموعه‌ها به‌عنوان اندازه دسته [۲] شناخته می‌شوند، درحالی‌که فرایند یک بار عبور از کل مجموعه‌ی داده، به عنوان یک دوره [۳] شناخته می‌شود [۲۳].

شبکه‌های عصبی عمیق، قادر به استخراج ویژگی‌های غیرخطی دقیق از داده‌های آموزشی هستند، اما این ویژگی‌ها قابلیت اطمینان کافی را ندارند. در واقع، اضافه کردن مقدار کمی نویز [۴] (نوفه) در ورودی می تواند باعث انحراف این ویژگی‌ها و گیج شدن مدل شود. به عبارت دیگر، شبکه‌های عصبی عمیق، یک طبقه‌بند بسیار بی‌طرف (با اریبی پایین) هستند، اما دارای واریانس بالایی می‌باشند [۲۰].

طراحی یک شبکه عصبی شامل انتخاب تعداد لایه‌ها (عمق) و نوع لایه‌ها است. نوع هر لایه با نحوه اتصال نورون‌های لایه‌ها به یکدیگر تعیین می‌شود. در شبکه‌های عصبی، یک لایه می‌تواند متراکم [۵] باشد که در آن هر نورون به تمام نورون‌های لایه قبلی متصل است، یا پیچشی [۶] باشد که در آن هر نورون فقط به یک زیرمجموعه از نورون‌های لایه قبلی متصل است و این موضوع الهام گرفته شده از قشر بصری موجود در مغز حیوانات است. همچنین لایه‌های موجود در شبکه عصبی، نوع آن را تعیین می‌کنند؛ مانند شبکه عصبی کانولوشنی یا پیچشی (CNN [۷])، شبکه عصبی بازگشتی (RNN [۸]) و یا شبکه پرسپترون چند لایه متصل به صورت متراکم (MLP [۹]). معماری یک شبکه عصبی به ساختار کلی آن اشاره دارد که شامل مواردی مثل تعداد نورون‌های متصل، روش اتصال آن‌ها و تعداد لایه‌ها است [۲۲].

در آموزش شبکه‌های عصبی عمیق، یکی از مشکلات ممکن است این باشد که یک لایه از شبکه، دیگر یادگیری نکند و فرایند آموزش را متوقف کند. این موضوع باعث می‌شود که اطلاعات گرادیان به لایه‌های قبلی بازگشت پیدا نکند. برای رفع این مشکل، از اتصالات باقی‌مانده [۱۰] استفاده می‌شود. اغلب، شبکه‌های عصبی که از این نوع اتصالات استفاده می‌کنند، به عنوان شبکه باقی‌مانده [۱۱] شناخته می‌شوند [۲۴]. یکی از مهم‌ترین انتخابات طراحی، انتخاب تابع غیرخطی است که بعد از هر نورون به کار گرفته می‌شود. انتخاب‌های معمول شامل ReLU [۱۲]، سیگموید [۱۳] و tanh هستند. تابع فعال‌سازی سیگموید، مقداری بین ۰ و ۱ را خروجی می‌دهد که می‌توان آن را به عنوان امتیاز اطمینان تفسیر کرد.

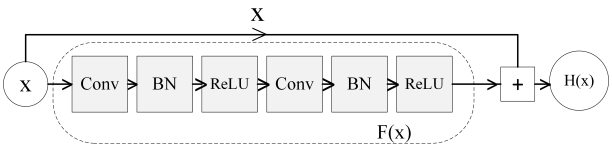
۱.۴ معماری عمومی تمایزگر عصبی

در بخش قبلی در ابتدا مفاهیم پایه یادگیری عمیق و معماری مدل‌های آن به صورت خلاصه بیان گردید. حال در این‌جا اولین معماری استفاده شده برای تحلیل رمز Speck توسط مقاله [۳] را معرفی و بررسی خواهیم کرد. به دلیل نتایج موفقیت‌آمیز و حاصل شدن دقت بالاتر نسبت به تمایزگرهای بر مبنای DDT، این معماری به صورت عمومی در اکثر پژوهش‌های بعدی نیز به همان شکل و یا با اضافه نمودن بلوک‌های دیگر برای بهبود آن استفاده شده است. بسیار مهم است که یک ND با عملکرد و دقت خوب قبل از انجام بازیابی کلید به دست آید.

در کنفرانس کریپتو ۲۰۱۹، آرون گور [۱۴] از شبکه عصبی باقی‌مانده که در بخش قبل معرفی شد، برای گرفتن اطلاعات و ویژگی‌های تفاضلی بین زوج‌های متن رمزی استفاده نمود و با این عمل اولین تمایزگر عصبی رمز

[1] Accuracy [2] Batch [3] Epoch [4] Noise

[5] Dense [6] Convolutional [7] Convolutional neural network [8] Recurrent neural network [9] Multi-layer perceptron [10] Residual [11] Residual network [12] Rectified linear unit [13] Sigmoid [14] Aron Ghor



شکل ۴. بلوک باقی‌مانده [۲۰]

Speck به دست آمد [۳]. مفهوم اصلی شبکه باقی‌مانده، معرفی اتصال میان‌بر[1] یا «اتصالات کوتاه» به شبکه عصبی پیچشی عادی است. به این معنی که خروجی داده از لایه قبلی به طور مستقیم بر روی ورودی لایه داده‌ای که بعد از آن قرار دارد، قرار می‌گیرد و یک یا چند لایه پیچشی را نادیده می‌گیرد. این شبکه از مجموعه‌ای از بلوک‌های باقیمانده تشکیل شده است. یک بلوک باقیمانده می‌تواند به شکل رابطه ۱۷ بیان شود.

$$H(x) = F(x) + x \qquad (۱۷)$$

در رابطه ۱۷، $H(x)$ نگاشت زیربنایی مورد نظر و x نگاشت مستقیم است؛ به این معنی که لایه‌های غیرخطی روی هم به صورت $F(x) := H(x) - x$ مطابقت دارند. شکل ۴ نمایانگر یک بلوک باقی‌مانده است.

لایه نرمال‌سازی دسته‌ای[2] در شکل ۴، با استفاده از روش‌هایی از نرمال‌سازی، داده‌های خروجی لایه پیچشی را به توزیع نرمال استاندارد با میانگین صفر و واریانس یک تبدیل می‌کند. این کار می‌تواند مشکل ناپدید شدن گرادیان را به طور مؤثری کاهش دهد و سرعت آموزش شبکه را افزایش دهد. تابع فعال‌سازی ReLU یک تابع فعال‌سازی اشباع‌شونده یک‌طرفه است که به صورت $f(x) = \max(۰, x)$ تعریف می‌شود؛ گرادیان‌های تابع فعال‌سازی ReLU در مقادیر مثبت ثابت می‌شوند و دیگر ناپدید نمی‌شوند. این بدان معناست که با استفاده از ReLU، از مشکل ناپدید شدن گرادیان به طور مؤثری می‌تواند جلوگیری شود.

برای یادگیری رابطه XOR در جایگاه یکسان متن رمزی، یک لایه پیچشی یک بعدی با اندازه هسته ۱ در ابتدای معماری شبکه گور استفاده می شود. به صورت کلی معماری عمومی گور مطابق شکل ۵ شامل چهار بخش می‌باشد:

- بلوک اول قالب ورودی شبکه را مشخص می‌کند.
- بلوک دوم لایه پیچشی یک بعدی با اندازه هسته ۱ و به دنبال آن لایه نرمال‌سازی دسته‌ای و تابع فعال‌سازی ReLU می‌باشد.
- بلوک سوم یک تا ده لایه دارد که هر لایه خود حاوی دو لایه پیچشی یک بعدی (بلوک‌های باقی‌مانده) با اندازه هسته ۳ می‌باشند و هر کدام دارای لایه نرمال‌سازی دسته‌ای و تابع فعال‌سازی ReLU هستند.
- بلوک چهارم و نهایی طبقه‌بند غیرخطی است که از سه لایه تماما متصل پرسپترون که با دو لایه نرمال‌سازی دسته‌ای و تابع فعال‌سازی ReLU جدا شده‌اند، تشکیل شده است. این بلوک در نهایت با یک تابع سیگموید برای مشخص کردن کلاس هر نمونه خاتمه می‌یابد.

[1]Skip connection [2]Batch normalization

به صورت کلی در معماری شبکه گور، لایه ورودی، زوج متن رمزی ۶۴ بیتی رمز Speck32/64 را تغییر شکل داده و در یک ماتریس ۱۶ بیتی با ۴ کانال (ماتریس با ۴ سطر و ۱۶ ستون، هر سطر شامل یک واژه ۱۶ بیتی از کل چهار واژه زوج متن رمزی) بازسازی و جابجا می‌کند. از منظر رمزنگاری، تغییر شکل ورودی نمایانگر دانشی از ساختار واژه‌های ۱۶ بیتی (سمت چپ و راست ساختار رمز) است. با گرفتن زوج متن رمزی $(C_۰, C_۱)$ ابتدا زوج متن رمزی به ۴ واژه (واژه‌های سمت چپ و راست) تقسیم می‌شود که این مرحله را «لایه پیش‌پردازش» می‌نامیم.

در قسمت دوم، با اعمال یک لایه پیچشی تک بعدی

$$(\text{Conv1D}(k = ۱, f = ۳۲))$$

برای برش بیتی[3] از ۴ کانال استفاده می‌شود (لایه برش بیتی). عمل «برش» توسط اندازه هسته ۱ $(k = ۱)$ نمایان می‌شود. کانال خروجی برای هر فیلتر با اسکن کردن فیلتر مربوطه روی ورودی در یک بعد تولید می‌شود. پارامترهای قابل یادگیری شامل چهار وزن فیلتر و یک پارامتر اریبی برای هر یک از ۳۲ فیلتر $(f = ۳۲)$ هستند که در نتیجه تعداد کلی پارامترهای یادگیری برای این لایه Conv1D برابر با $۳۲ \times ۴ + ۳۲ = ۱۶۰$ می‌شود. آرایه خروجی از لایه پیچشی برش بیتی دارای ۱۶ بیت عرض و با عمق ۳۲ کانال است. در طول شبکه، پس از هر لایه پیچشی، لایه نرمال‌سازی دسته‌ای و تابع فعال‌سازی ReLU می‌باشد.

بخش سوم، بلوک‌های باقی‌مانده است. هر بلوک باقی‌مانده از دو لایه پیچشی $\text{Conv1D}(k = ۳, f = ۳۲)$ تشکیل شده است. در مقاله [۳]، تعداد بلوک‌های باقی‌مانده عمق شبکه تمایزگر را مشخص می‌کند (۱ تا ۱۰).

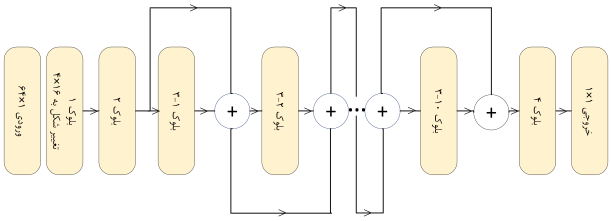
بخش چهارم شبکه، لایه تماما متصل با تابع فعال‌سازی ReLU و یک لایه خروجی با یک نورون با فعال‌سازی سیگموید است. در سرتاسر شبکه، هر لایه پیچشی و هر لایه تماما متصل توسط یک پارامتر $L۲ = ۱۰^{-۵}$ تنظیم می‌شوند. برای آموزش، از بهینه‌ساز آدام[4] همراه با نرخ یادگیری چرخشی استفاده می‌شود. خطا به عنوان میانگین مربعات خطا (MSE) بین پیش‌بینی و برچسب کلاس به همراه جریمه تنظیم $L۲$ محاسبه می‌شود.

برای تحلیل هر سه رمز کاهش دور یافته Speck، Simon و Simeck می‌توان از شبکه فوق به عنوان شبکه پایه استفاده نمود و تمایزگر عصبی را ایجاد کرد.

۲.۴ اساس انتخاب شبکه

در برخی مطالعات، مانند تحقیقات انجام شده در مقاله [۱۶]، نشان داده شده است که شبکه‌های عصبی پیچشی (CNN) برای پیدا کردن یک تمایزگر عصبی مناسب نیستند. دلیل این موضوع آن است که CNNها برای شناسایی الگوها در داده‌های ورودی طراحی شده‌اند که این امر در شناسایی تصاویر یا پردازش زبان طبیعی کمک‌کننده است؛ اما برای داده‌های ورودی رمزنگاری شده، جایی که بیت‌ها به هیچ وجه با یکدیگر

[3]Bit-slicing [4]Adam



شکل ۵. معماری شبکه تمایزگر عصبی با ۱۰ بلوک باقی‌مانده

مرتبط نیستند، کارایی ندارد. همچنین در این مقاله اشاره شده است که شبکه‌های LSTM [1] (یک نوع شبکه عصبی بازگشتی) در مقایسه با شبکه‌های پیچشی (CNN) برای تمایزگرهای عصبی عملکرد بهتری دارند، اما عملکرد آن‌ها از شبکه‌های چندلایه پرسپترون (MLP) که به‌خوبی تنظیم شده‌اند، بدتر است. از اصلی‌ترین معایب LSTM سرعت آموزش آن می‌باشد.

حال به بررسی دلیل انتخاب معماری عمومی تمایزگرهای عصبی (مطابق با شکل ۵) می‌پردازیم. شبکه‌های عمیق باقی‌مانده که ابتدا برای شناسایی تصویر معرفی شدند، از آن زمان در طیف گسترده‌ای از کاربردها، از جمله در بازی‌های تخته‌ای استراتژیک، به موفقیت‌های قابل توجهی دست یافته‌اند. این شبکه‌ها به دلیل عملکرد برجسته‌شان در مسائل طبقه‌بندی، به عنوان یک گزینه مطلوب برای طراحی تمایزگرها شناخته می‌شوند. دلیل این امر آن است که فرآیند ایجاد تمایزگر می‌تواند به مسئله‌ای از طبقه‌بندی بین دو یا چند گروه تقلیل یابد، که همان تشخیص تفاوت بین دو یا چند کلاس مختلف است. گور نشان داد که شبکه باقی‌مانده می‌تواند عدم تصادفی بودن توزیع زوج‌های خروجی را زمانی که زوج‌های ورودی الگوریتم رمز Speck دارای یک تفاضل خاص هستند، شناسایی کند.

در ساختار عمومی تمایزگر عصبی استفاده از لایه پیچشی اولیه با عرض ۱ به منظور تسهیل یادگیری توابع ساده برش بیتی مانند جمع بیتی و یا عملگر AND طراحی شده است. تعداد فیلترها در لایه پیچشی اولیه به منظور گسترش داده‌ها به فرمت مورد نیاز برای لایه باقی‌مانده در نظر گرفته شده است.

انتخاب کانال‌های ورودی با هدف آشنا ساختن شبکه با ساختار مبتنی بر واژه رمز است. استفاده از یک سر تصمیم‌گیری با اتصال متراکم، بیانگر این واقعیت است که برای تعداد دورهای بالا از الگوریتم رمز، انتظار نمی‌رود داده‌های ورودی تقارن فضایی قوی نشان دهند؛ بنابراین، هر تلاشی برای استخراج ویژگی‌های محلی از داده‌ها با استفاده از یک لایه ادغام فضایی متقارن به نوعی احتمالاً بی‌ثمر خواهد بود. همچنین اندازه لایه‌ها از طریق آزمایش تعیین می‌شوند.

عمق لایه باقی‌مانده نیز باید به گونه‌ای انتخاب شود تا اجازه دهد داده‌های ورودی در کل رشته ورودی طی لایه‌های پیچشی ادغام شوند. با این حال، حتی یک طراحی با تنها یک بلوک باقی‌مانده نیز نتایج قابل قبولی (که به وضوح از یک تمایزگر تفاضلی مبتنی بر DDT بهتر است) به دست می‌دهد.

از دیگر ساختارهای استفاده شده به همراه معماری شبکه باقی‌مانده برای بهبود کارایی و افزایش دقت تمایزگرهای عصبی می‌توان به SeNet [2] و ماژول تلقین [3] اشاره نمود. شبکه‌های SENet با اعمال مکانیزم نوآورانه Squeeze-and-Excitation به شبکه‌های عصبی پیچشی، عملکرد شبکه را از طریق تمرکز دقیق بر روی اهمیت کانال‌های ویژگی بهبود می‌بخشند. این رویکرد با اختصاص وزن‌های متفاوت به هر کانال، به شبکه اجازه می‌دهد تا ویژگی‌های مؤثرتری را برای تصمیم‌گیری شناسایی کند، در حالی که اثر ناچیزی بر بار محاسباتی دارد و به راحتی در معماری‌های موجود قابل ادغام است [۲۵]. این امر به SENetها امکان می‌دهد که در طیف وسیعی از کاربردها، از شناسایی تصویر گرفته تا تحلیل داده‌های پیچیده و تصادفی مانند داده‌های درگیر در تحلیل رمز، به بهبود قابل توجهی در دقت کمک کنند، بدون اینکه نیازی به افزایش قابل ملاحظه در منابع محاسباتی باشد.

همچنین ماژول تلقین، با قابلیت اجرای چندین فیلتر پیچشی به صورت موازی درون یک لایه، به شبکه‌های عصبی امکان می‌دهد تا ویژگی‌های داده‌ها را در مقیاس‌های مختلف به طور همزمان شناسایی و استخراج کنند. این امر باعث می‌شود شبکه بتواند از دیدگاه‌های متفاوتی به داده‌ها نگاه کند، که منجر به بهبود دقت در تشخیص الگوها می‌شود. همچنین، ساختار منحصر به فرد ماژول تلقین با کاهش نیاز به تنظیمات دستی برای اندازه فیلترها، فرایند طراحی شبکه را ساده‌تر می‌کند. علاوه بر این، با کاهش پارامترها و افزایش عمق و پهنای شبکه بدون افزایش بی‌رویه پیچیدگی محاسباتی، به بهینه‌سازی منابع کمک می‌کند. در نتیجه، ماژول‌های تلقین به عنوان یکی از موثرترین ابزارها در بهبود عملکرد شبکه‌های عصبی پیچشی، به ویژه در کاربردهای تجزیه و تحلیل داده‌های پیچیده مثل داده‌های رمزنگاری شده، شناخته می‌شوند.

به صورت کلی علی‌رغم ماهیت تصادفی داده‌های موجود در تحلیل رمز، با به کارگیری یک ساختار و شبکه طبقه‌بند مناسب مانند شبکه‌های عمیق باقی‌مانده، مشاهده می‌شود که یادگیری عمیق در تشخیص این داده‌های تصادفی به خوبی عمل می‌کند. اصلی‌ترین دلیل برای این موضوع آن است که شبکه‌های عصبی عمیق می‌توانند با اعمال فیلترهای متفاوت به داده‌ها و همچنین نگاه به آن‌ها از جهات گوناگون، با تشخیص وابستگی‌های پیچیده آماری موجود در داده‌های رمزشده اطلاعاتی فراتر از یک جدول توزیع تفاضل را یاد بگیرند و برای ایجاد تمایز از آن اطلاعات استفاده نمایند. همان‌طور که در مقاله [۲۰] اشاره شده است، این اطلاعات عموما شامل ویژگی‌های تفاضلی-خطی الگوریتم رمز مورد نظر و نیز ترکیبات خطی از زوج‌های متن رمزی و تفاضل‌های کلی یا جزیی آن‌هاست. همچنین تمایزگرهای عصبی تصمیمات خود را نه تنها بر اساس تفاضل زوج متن رمزی، بلکه بر اساس تفاضل حالت‌های میانی در دورهای نهایی نیز استوار می‌کنند. این مسئله در بخش ۶.۴ به صورت گسترده‌تر

[1] Long short-term memory

[2] Squeeze-and-Excitation network [3] Inception module

مورد بررسی قرار خواهد گرفت.

۳.۴ پارامترهای شبکه

زمینه تمایزگرهای عصبی هنوز در مراحل اولیه توسعه قرار دارد و هنوز مشخص نیست کدام معماری یادگیری عمیق بهتر عمل می‌کند. از شبکه گور (با تغییرات) در انواعی از کارهای همتا از جمله شبکه‌های MLP و CNN [۱۶] تا شبکه‌های بزرگ‌تر به مانند SeNet [۲۵] یا ترکیبی از ویژگی‌های دست‌ساخته با طبقه‌بندی‌های غیرعصبی [۲۰] استفاده شده است.

شبکه عصبی، مانند شبکه ای که در مقاله [۳] استفاده می‌شود، ساختار خاصی دارد که باید به طور پیش‌فرض تعیین شود. این ساختار توسط پارامترهای معمولاً به نام ابر پارامتر [1] تعریف می‌شود، مانند تعداد لایه‌ها یا نورون‌ها در هر لایه که می‌توانند تأثیر بزرگی روی توانایی شبکه در یادگیری داشته باشند. بنابراین، در یادگیری عمیق معمول است که با تلاش در ترکیب‌های مختلف، به دنبال پیدا کردن ابر پارامترهایی باشیم که منجر به بهترین نتایج می‌شوند. در مقاله [۱۹]، با جستجو بر روی فضای بزرگی از این پارامترها، میزان اهمیت و مقادیر مناسبی برای آن‌ها به دست آمد.

از میان تمامی پارامترها فقط برخی از پارامترها از جمله اندازه دسته، استفاده از لایه پیچشی دایروی، تعداد نورون‌ها در لایه‌های تماما متصل، تعداد فیلترها، اندازه فیلتر، نرخ یادگیری (استفاده از برنامه زمان‌بندی نرخ یادگیری چرخشی)، تعداد بلوک‌های باقی‌مانده و جریمه تنظیم $L2$ تأثیر مثبت قابل توجهی دارند. همچنین از میان این ابر پارامترها نیز ابر پارامترهای نرخ یادگیری، مقدار تنظیم $L2$ و مخصوصا اندازه فیلتر بیشترین اثر را دارند [۱۹]. بنابراین اگر کسی به دنبال استفاده از تمایزگر عصبی برای یک رمز جدید باشد، باید حداقل مقدار بهینه این ابر پارامترها را برای آن رمز جستجو کند. البته براساس مطالعات می‌توان گفت که ابر پارامترهای مهم‌تر مقادیر تقریباً مشابهی برای رمزهای قالبی دارند. در جدول ۲ مقادیر بهینه ابرپارامترهای مذکور برای هر سه رمز مورد بررسی آورده شده است.

در مورد رمز Simon، اندازه فیلتر ۷ باعث افزایش دقت چند درصدی نسبت به اندازه فیلتر ۳ می‌شود. یکی از توضیحات ممکن برای این موضوع این است که شبکه عصبی احتمالاً تبدیلات قطعی را در پایان فرایند رمزگذاری (تا اضافه کردن کلید) برعکس می‌کند. بنابراین، برعکس کردن چنین محاسباتی به صورت پیش‌فرض (یعنی قبل از ارائه نمونه‌ها به شبکه عصبی)، که به عنوان محاسبه برگشتی متن رمزی شناخته می‌شود، می‌تواند به عنوان یک پارامتر اضافی در فضای جستجو شناخته شود. این کار در رمزهای Simon و Simeck بهبود مشخصی را به همراه خواهد داشت. اما برای Speck، بهبود قابل توجهی مشاهده نمی‌شود که ممکن است به دلیل بهینه‌سازی شبکه عصبی برای انجام این محاسبات توسط خود شبکه باشد.

علاوه بر تنظیم خودکار ابر پارامترها، مقاله [۱۹] به دو بهینه‌سازی

[1] Hyper parameter

جدول ۲. مقادیر بهینه ابرپارامترهای مهم برای سه الگوریتم رمز Speck، Simon و Simeck

رمز	Speck	Simon	Simeck
بیشینه نرخ یادگیری	$۳/۵ \times ۱۰^{-۳}$	$۲/۷ \times ۱۰^{-۳}$	۲×۱۰^{-۳}
کمینه نرخ یادگیری	۲۲×۱۰^{-۵}	۲۰×۱۰^{-۵}	$۱۰^{-۴}$
تعداد نورون‌ها	۸۰	۶۴	۶۴
اندازه فیلتر	۳	۷	۳
تعداد فیلترها	۱۶	۳۲	۳۲
تنظیم $L2$	$۸۴/۹ \times ۱۰^{-۸}$	۲۲۰×۱۰^{-۸}	$۱۰^{-۵}$
لایه پیچشی دایروی	✗	✓	✗
محاسبه برگشتی	✓	✓	✓
اندازه دسته	۵×۱۰^{۳}	۵×۱۰^{۳}	۳×۱۰^{۴}
تعداد لایه باقی‌مانده	۱۰	۱۰	۵

دستی ممکن برای بهبود تمایزگر اشاره می‌کند. از یک طرف، رمزنگاران ممکن است تفاضل‌های ورودی بهتری پیدا کنند. از طرف دیگر، آن‌ها می‌توانند روش آموزشی دقیق‌تری انتخاب کنند که در قسمت بعدی به روش‌های آموزش پرداخته خواهد شد. همچنین طرح مطرح شده در مقاله [۲۲]، تعمیم‌پذیری بهتری نسبت به تنظیم خودکار ابر پارامترها در مقاله [۱۹] دارد که در بخش سوم روش خودکارسازی انتخاب تفاضل بهینه ورودی آن بیان شد و در ادامه روش آموزش این مقاله نیز مورد بررسی قرار خواهد گرفت.

۴.۴ روش‌های آموزش

برای آموزش شبکه‌های عصبی مطرح شده، در مقاله [۳] سه روش آموزش پیشنهاد و به کار گرفته شد. این سه روش عبارت‌اند از: روش آموزش مستقیم پایه‌ای، استفاده از الگوریتم میانگین‌گیری کلید [2] و آموزش مرحله‌ای.

اولین روش، طرح آموزشی پایه‌ای است که برای آموزش موفقیت‌آمیز تمایزگرهای کوتاه‌دور کافی است. دومین روش، طرح آموزشی بهبودیافته‌ای برای تمایزگرهای r–دوری است که خروجی الگوریتم میانگین‌گیری کلید را که با استفاده از یک تمایزگر $(r-۱)$–دوری به دست می‌آید، شبیه‌سازی می‌کند. با استفاده از روش دوم، بهترین تمایزگر غیرخطی ۷ دوری رمز Speck32/64 در مقاله [۳] به دست آمده است. بدین منظور، یک مجموعه آزمون یک میلیون عددی برای رمز Speck کاهش‌یافته به ۷ دور، طبق روشی که در ابتدای مقاله بیان شد، تولید می‌شود. سپس هر زوج متن‌رمزی در مجموعه آزمون با انجام جستجوی جامع کلید بر روی آخرین دور ارزیابی خواهد شد. در ادامه رمزگشایی‌های جزئی حاصل، با استفاده از یک تمایزگر عصبی ۶ دوری درجه‌بندی می‌شوند و نتایج برای زوج متن رمزی با تبدیل آن‌ها به یک امتیاز به طریق شباهت (حقیقی در مقابل تصادفی) و میانگین‌گیری، ترکیب می‌شوند. الگوریتم ۱ جزئیات روش

[2] Key averaging

مورد استفاده را ارائه می‌دهد.

الگوریتم ۱ (میانگین‌گیری کلید [۳]). استخراج یک تمایزگر تفاضلی برای یک رمز قالبی کاهش‌یافته به $r+1$ دور با تفاضل ورودی Δ از یک تمایزگر D متناظر r دوری. یک نمونه از توزیع حقیقی است اگر و فقط اگر خروجی الگوریتم برای آن بزرگ‌تر از ۰٫۵ باشد. (v: امتیاز تمایزگر به هر نمونه)

Require: زوج متن رمزی خروجی مشاهده‌شده $C_0, C_1 \in \{0,1\}^b$

$D_i \leftarrow [DecryptOneRound(C_i, k) \ for \ k \in Subkeys]$
$v_k \leftarrow D(D_0[k], D_1[k]) \ for \ all \ k \in Subkeys$
$v_k \leftarrow v_k/(1-v_k) \ for \ all \ k \in Subkeys$
$v \leftarrow Average([v_k, \ k \in Subkeys])$
$v \leftarrow v/(1+v)$
return v

هنگام آموزش یک تمایزگر عصبی، بالاترین دور قابل دست‌یابی ممکن است با استفاده از تکنیک‌های ساده قابل آموزش نباشد. سومین روش، یک روش آموزشی مرحله‌ای است که یک تمایزگر $(r-1)$-دوری از قبل آموزش‌دیده را در چند مرحله به یک تمایزگر r-دوری تبدیل می‌کند. با استفاده از روش سوم، طولانی‌ترین تمایزگر غیرخطی رمز Speck32/64 که یک تمایزگر ۸ دوری است، در مقاله [۳] به دست آمده است.

برای به دست آوردن یک تمایزگر ۸ دوری رمز Speck، گور نیاز به استفاده از یک طرح آموزش مرحله‌ای داشت؛ جایی که بهترین تمایزگر ۵ دوری مجدداً بر روی تفاضل ورودی 0x840a/8000 (که بیشترین احتمال ظاهرشدن را پس از سه دور دارد) آموزش داده می‌شود. این تمایزگر سپس برای ۸ دور و با پیچیدگی داده ۱۰۰ برابر بیشتر نسبت به سایر تمایزگرها، مجدداً آموزش داده می‌شود تا در نهایت به دقت اعتبارسنجی ۰٫۵۱۴ برسد.

مقاله‌های [۲۵] و [۱۹] نیز از روش‌های آموزش مرحله‌ای مشابهی برای تمایزگر ۱۰ دوری رمز Simon32/64 خود استفاده کرده‌اند. همچنین در مقاله [۱۵] تمایزگر ۱۲ دوری برای رمز Speck32/64 با استفاده از این روش حاصل شده است.

همان‌طور که قبلا بیان شد، مقاله [۲۲] با دیدگاه خودکار سازی فرایند یادگیری، روش خود را ارائه نمود. طرح‌های آموزش تفصیلی فوق، به سادگی قابل خودکارسازی نیستند؛ زیرا نیازمند مشاهده ویژگی‌های تفاضلی رمز مورد مطالعه هستند. به همین جهت در مقاله [۲۲] یک روش آموزش دیگر با استفاده از ایده آموزش مرحله‌ای، ارائه گردید.

در این روش یک طرح ساده برای آموزش یک تمایزگر عصبی برای دورهای R_s تا R_f پیشنهاد شد. یک شبکه R_s دوری برای R_{f+1} دور مجددا آموزش داده می‌شود تا یک تمایزگر R_f دوری به دست آید. در مورد رمز Speck32/64، ابتدا برای ۵ دور شبکه N5 آموزش داده می‌شود. سپس شبکه N5 را بر روی مجموعه‌داده ۶ دوری مجدداً آموزش داده تا N6 به‌دست آید، پس از آن N6 را بر روی ۷ دور آموزش داده تا N7 به دست آید و در نهایت N7 را بر روی ۸ دور آموزش داده تا N8 حاصل شود. همچنین شبکه نهایی به دست آمده در جهت افزایش دقت می‌تواند ۳ بار برای ۱ دوره روی مجموعه‌داده آموزشی جدید، مجدداً آموزش ببیند.

در آخرین تحقیقات انجام شده در این زمینه با استفاده از یک استراتژی تنظیم دقیق شبکه، یعنی لایه انجماد[1]، روشی برای محدود کردن فضای راه حل و آموزش شبکه تمایزگرهای عصبی ارائه شد [۲۶]. تمایزگرها به‌طورکلی از دو بخش تشکیل شده‌اند: لایه‌های پیچشی و لایه‌های کاملاً متصل. در زمینه هوش مصنوعی، تمام لایه‌های پیچشی به عنوان استخراج‌کننده ویژگی‌ها در نظر گرفته می‌شوند، در حالی که همه لایه‌های کاملاً متصل به عنوان یک طبقه‌بند در نظر گرفته می‌شوند. از این رو، استخراج‌کننده ویژگی را می‌توان مجدداً استفاده کرد اما طبقه‌بندها در دورهای مجاور نسبتاً مشابه هستند. بنابراین، برای آموزش یک تمایزگر ۸ دوری رمز Speck32/64، می‌توانیم به سادگی یک مدل ۷ دوری آموزش‌دیده را بارگذاری کنیم و تمام لایه‌های پیچشی آن را منجمد کنیم؛ به این معنی که فقط پارامترها در لایه‌های کاملا متصل می‌توانند به‌روزرسانی شوند. سپس می‌توانیم یک تمایزگر ۸ دوری با دقتی مشابه آن چه که در روش آموزش مرحله‌ای وجود دارد (و با پیچیدگی کمتر)، به دست آوریم و تمام ابر پارامترها در فرایند آموزش بدون تغییر باقی می‌مانند.

۵.۴ رویکردهای کلی بهبود تمایزگر عصبی و زمینه تحقیقات

برای بهینه‌سازی، بهبود و افزایش دقت تمایزگرهای عصبی با توجه به مباحث مطرح شده و تحقیقات انجام‌شده تا به اکنون، می‌توان ۴ ایده کلی زیر را مدنظر قرار داد:

- اولین مورد به‌کارگیری شبکه‌های عصبی متفاوت است. برای مثال در مقاله [۲۵] معماری SeNet برای آموزش تمایزگر عصبی به کار گرفته شد و در نتیجه تمایزگرهای عصبی ۷ تا ۱۱ دوری برای رمز Simon32/64 حاصل گردید. همچنین در مقاله [۲۷] با به‌کارگیری ایده استفاده از ماژول تلقین، شبکه جدیدی را برای آموزش تمایزگر معرفی نمود و با استفاده از آن تمایزگرهای عصبی ۵ تا ۸ دوری برای رمز Speck32/64 و ۷ تا ۱۲ دوری برای رمز Simon32/64 حاصل گردید.
- برای دومین ایده، می‌توان استفاده از قالب‌های ورودی متفاوت همان‌طور که در بخش ۱.۳ معرفی و ارائه گردید را در نظر گرفت.
- همچنین با انتخاب دقیق‌تر تفاضل‌های ورودی، امکان دستیابی به تمایزگرهای عصبی با عملکرد بهتر وجود دارد. به طور خاص، این امر به معنای انتخاب تفاضل‌های ورودی نیست که فقط تمایزگرهای عمیق‌تری ایجاد کنند؛ بلکه می‌توان با انتخاب یک تمایزگر عصبی کمی ضعیف‌تر و اما با کنترل دقیق‌تر رویدادهایی که تشخیص می‌دهد، به بهبودهای معناداری در اجرای یک حمله دست یافت. این رویکرد نشان می‌دهد که تعادل بین قدرت تمایزگر و قابلیت کنترل و شناسایی دقیق رویدادها می‌تواند در بهینه‌سازی عملکرد حمله نقش مهمی ایفا کند [۱۹].

[1] Freezing layer method

- برای چهارمین مورد، طراح می‌تواند سعی کند با اصلاح روش آموزش، تمایزگر عصبی را برای پوشش تعداد دور بیشتری کاربردی کند.

با توجه به اهمیت بالای معماری شبکه انتخاب شده به همراه پارامترهای آن برای تحلیل رمز، در مقاله [۲۸] مطالعه‌ای برای مقایسه اثربخشی و دقت دو شبکه ResNet و DenseNet روی رمز قالبی Speck32/64 صورت گرفت.

DenseNet با ارائه ساختاری متمایز شامل بلوک‌های متراکم و لایه‌های انتقال، به حل مشکل ناپدیدشدن گرادیان از طریق به حداکثر رساندن استفاده مجدد از ویژگی‌ها می‌پردازد. در تضاد با معماری‌های عصبی کلاسیک که لایه‌ها را به صورت خطی به یکدیگر متصل می‌کنند، DenseNet با ایجاد اتصالات متراکم بین لایه‌ها و بلوک‌ها، یک شبکه با قابلیت انتقال اطلاعات به شکل گسترده‌تر فراهم می‌کند. به جای استفاده از رویکرد جمعی مشابه ResNet برای ادغام خروجی‌های لایه‌های پیشین به لایه‌های بعدی، DenseNet هر خروجی گذشته را به عنوان ورودی به تمام لایه‌های بعدی می‌برد؛ نتیجتاً، هر لایه به طور مستقیم به تمام لایه‌های دیگر متصل می‌شود. این ساختار نه تنها به حفظ اطلاعات کمک می‌کند بلکه به افزایش کارایی شبکه در یادگیری ویژگی‌های پیچیده نیز منجر می‌شود.

برای این مقایسه داده ورودی به صورت (C_0, C_1) و سپس تبدیل آن به یک ماتریس ۱۶ × ۴ طبق توضیحات بخش‌های قبلی به هر دو مدل داده می‌شود. در نتیجه این مقایسه، مدل DenseNet به دقت اعتبارسنجی کمی بهتر از مدل ResNet برای دورهای ۵، ۶ و ۷ دست یافت. برای دور ۸، هر دو مدل نتوانستند نتیجه مناسبی ارائه دهند؛ زیرا مدل‌ها با روش آموزش پایه‌ای مستقیم نمی‌توانند یک الگوی دقیق را یاد بگیرند. در نتیجه انتخاب معماری در تحلیل رمز تفاضلی از اهمیت بالایی برخوردار است. در جدول ۳ مقایسه‌ای از عملکرد این دو مدل طبق مقاله [۲۸] آورده شده است.

همچنین برای دیگر معماری‌ها نیز می‌توان این مقایسه را انجام داد. شبکه دیگری که می‌تواند با ResNet مورد مقایسه قرار گیرد و حتی معماری آن با شبکه ResNet ترکیب شود (Se-ResNet)، شبکه SeNet است که برای دورهای بالای رمزهای مورد بررسی، تمایزگرهای با دقت بالاتر نسبت به ResNet ایجاد می‌کند [۲۵].

همچنین با توجه به مقاله [۲۲]، می‌توان اصلاحات تجربی صورت‌گرفته روی تمایزگر عمومی و پایه مطرح شده در مقاله [۳] را به چهار بعد تقسیم‌بندی نمود (k, t, T, E):

تعداد متن‌های رمزی در هر نمونه (k): یک راهکار موثر برای افزایش دقت تمایزگرهای عصبی، استفاده از گروه‌بندی چند زوج متن رمزی با برچسب‌های یکسان و تجمیع امتیازهای آن‌ها است. در این استراتژی، تمایزگر می‌تواند بر روی زوج‌های متن رمزی تکی آموزش دیده و سپس بر اساس مجموعه‌ای از زوج‌های با برچسب مشابه ارزیابی شود؛ رویکردی که در بازیابی کلید به کار رفته در مقاله [۳] مورد استفاده قرار گرفته است.

گاهی اوقات، این فرایند شامل الحاق چندین متن رمزی مختلف توسط خود شبکه عصبی است، که این امر به بهبود توانایی شبکه در تشخیص و تمایز بین کلاس‌ها یا برچسب‌های مختلف کمک می‌کند. این روش، با افزایش داده‌های مورد استفاده برای تصمیم‌گیری، به تقویت عملکرد کلی تمایزگر و افزایش دقت آن منجر می‌شود.

تعداد تفاضل‌های ورودی (t): این مورد در بخش‌های قبل با کار انجام شده در مقالات [۱۶، ۱۷] بررسی شد. همچنین مقاله [۲۹] مدلی به نام تمایزگر عصبی تفاضلی پلی‌تاپیک [1] را معرفی کرد. در این مدل از چندین تفاضل استفاده می‌شود که یک متن اصلی را در میان تفاضل‌ها ثابت نگه می دارد و دیگری را تغییر می‌دهد.

نوع مهندسی ویژگی (T): مهندسی ویژگی اغلب در یادگیری ماشین برای استخراج ویژگی‌های پیشرفته از مجموعه‌داده خام استفاده می‌شود. یک ایده ساده در این زمینه استفاده از تفاضل متن رمزی با XOR کردن زوج متن رمزی است که فرایند آموزش را به قیمت از دست دادن برخی اطلاعات ساده می‌کند. یکی از انواع پیشرفته مهندسی ویژگی نیز به عنوان مثال، رمزگشایی جزئی از متون رمزی است. مقاله [۲۱] با بهره‌برداری از اطلاعات استنباط‌شده از دو دور ماقبل آخر، تمایزگرهای بهتری را برای رمز Simon به نمایش گذاشت.

نوع آزمایش تمایز (E): محققان دریافته‌اند که تمایزگرهای عصبی مبتنی بر تفاضل در برخی از رمزها، عملکرد بهتری نسبت به تمایزگرهای مبتنی بر DDT دارند. اما این‌که این تمایزگرهای عصبی چه دانشی فراتر از تمایزگرهای مبتنی بر DDT یاد می‌گیرند، هنوز به صورت کامل مشخص نیست. تحقیقات قبلی نشان می‌دهد که این تمایزگرها از توزیع‌های تفاضل در دو دور آخر و خواص خطی–تفاضلی استفاده می‌کنند [۲۰]. در مقاله [۳]، «آزمایش تفاضل‌های واقعی» برای مشاهده این که چگونه شبکه‌های عصبی می‌توانند تفاضل‌های واقعی فراتر از DDT را تشخیص دهند، انجام شد. این آزمایش از زوج‌های متن‌رمزی تصادفی‌سازی‌شده با مقدار کورکننده R استفاده کرد تا اطلاعات فراتر از تفاضل را برای شبکه مبهم کند. در این آزمایش تمایزگر باید تشخیص دهد که آیا متن‌های رمزی با یک نقاب تصادفی DDT شده‌اند یا خیر. نتایج نشان داد که شبکه‌های عصبی می‌توانند تفاضل‌های واقعی را بدون آموزش صریح تشخیص دهند و زوج‌های متن رمزی دارای توزیع‌های غیریکنواخت در کلاس‌های هم ارزی تفاضل خود هستند. همچنین عملکرد موفقیت‌آمیز تمایزگرهای عصبی در این آزمایش بیانگر این موضوع است که آن‌ها اطلاعاتی فراتر از یک تفاضل XOR ساده را یاد می‌گیرند. اما در یک آزمایش دیگر با محدود کردن مقادیر کور کننده R در آزمایش تفاضل‌های واقعی به شکل $R = aa$ (a به عنوان هر کلمه ۱۶ بیتی)، تمایزگرها در تشخیص متن‌های رمزی تصادفی‌سازی شده شکست خوردند. این نشان می‌دهد که تمایزگرهای عصبی از فرانمای کلید استفاده نمی‌کنند؛ زیرا اضافه کردن مقدار کورکننده به این فرم معادل تغییر زیرکلید نهایی استفاده شده در رمزنگاری است. همچنین این موارد نشان می‌دهد که تمایزگرهای عصبی

[1] Polytopic

جدول ۳. مقایسه دقت آموزش و صحت‌سنجی برای مدل‌های ResNet و DenseNet روی الگوریتم رمز Speck32/64 [۲۸]

دورها	ResNet		DenseNet	
	صحت‌سنجی	آموزش	صحت‌سنجی	آموزش
۵	۰٫۶۷۷۹	۰٫۹۳۳۲	۰٫۷۰۰۵	۰٫۹۳۰۹
۶	۰٫۵۸۷۴	۰٫۷۹۵۲	۰٫۵۹۲۳	۰٫۷۹۱۹
۷	۰٫۵۲۶۷	۰٫۶۰۹۶	۰٫۵۳۱۳	۰٫۶۰۵۳
۸	۰٫۵۰۱۲	۰٫۴۹۹۶	۰٫۵۰۰۲	۰٫۴۹۹۸

قادر به ایجاد تمایزهای دقیق‌تری نسبت به کلاس‌های هم ارزی تفاضلی هستند.

در نهایت تحقیقات در موضوع تحلیل رمز به کمک یادگیری ماشین را می‌توان به دو منظر کلی تقسیم‌بندی کرد. در نگاه اول پژوهشگران به دنبال بهبود و اصلاح تمایزگرها و نتایج قبلی هستند و در نگاه دوم نیز محققان درصدد تفسیر و توضیح‌پذیری عملکرد درونی شبکه‌های عصبی و مدل‌های یادگیری عمیق می‌باشند. عوامل مؤثر بر دیدگاه بهبود و اصلاح تمایزگرها (که خود تاحدی موجب بالا رفتن سرعت و موفقیت بازیابی کلید خواهد شد) به طور کامل بررسی شد. از منظر تفسیر و توضیح‌پذیری مدل‌ها و نتایج نیز کارهای محدودی انجام شده که از مهم‌ترین آن‌ها می‌توان به مقالات [۲۰، ۲۶] اشاره نمود. به عنوان اولین و مهم‌ترین کار از منظر دوم تحقیقات، در ادامه و برای تکمیل این بخش به بررسی مقاله [۲۰] خواهیم پرداخت.

۶.۴ مروری بر اثر بنامیرا و همکاران و نتایج مربوطه

به منظور کشف عملکرد درونی تمایزگر عصبی ND_1^{cp} گور، یک کاوش جامع توسط بنامیرا [1] و همکاران در کنفرانس یوروکریپت ۲۰۲۱ [۲۰] انجام شد و یکپارچه‌سازی دیدگاه‌های تحلیل رمز و یادگیری ماشین صورت پذیرفت. از دیدگاه تحلیل رمز، آن‌ها دو کاوش متمایز را آغاز کردند. کاوش اول از تفاضل ورودی 0x2800/0010 بهترین مسیر تفاضلی ۵ دوری (در مقایسه با تفاضل 0x0000/0040 استفاده شده در مقاله [۳]) برای آموزش ND_1^{cp} استفاده کرد. با این حال، دقت به دست آمده با این ورودی جدید (یعنی ۷۵٫۸۵ درصد)، کمتر از دقت ۹۲٫۹ درصد به‌دست‌آمده با تفاضل 0x0000/0040 بود. کاوش دوم از تفاضل‌های زوج متن رمزی برای آموزش ND_1^{cd} استفاده کرد. دقت تمایزگرهای عصبی ۵، ۶ و ۷ دوری جدید به ترتیب برابر با ۹۰٫۶، ۷۵٫۴ و ۵۸٫۳ بودند که کمی بدتر از ND_1^{cp} است.

سپس، چهار آزمایش (آزمایش‌های A، B، C و D) برای تجزیه و تحلیل زوج‌های متن رمزی انجام شد. آزمایش A با هدف ارزیابی این‌که آیا ND_1^{cp} ۵ دوری احتمال بیشتری دارد که تفاضل متن رمزی ۵ دوری را با احتمال بالاتر مانند تمایزگر کلاسیک تشخیص دهد یا خیر، انجام شد. با این حال، برخلاف انتظار، نتایج خلاف این فرضیه را نشان داد. پس از آزمایش A، آزمایش B به بررسی ظرفیت ND_1^{cp} ۵ دوری برای شناسایی مؤثر و دقیق زوج‌های متن رمزی ۵ دوری که با احتمال قوی در تفاضل‌های ۳ و ۴ دوری آن‌ها مشخص می‌شوند، پرداخت. نتایج آزمایش‌ها نشان داد که ND_1^{cp} می‌تواند این زوج‌ها را با دقت تقریباً ۱۰۰ درصد به خوبی تشخیص دهد.

علاوه بر این، آن‌ها تحقیقی در آزمایش C در مورد اریبی نشان داده شده توسط بیت‌های تفاضل منقطع ۳ و ۴ دوری ($TD_{3/4}$) انجام دادند و مشاهده کردند که ND_1^{cp} ۵ دوری به شدت به $TD_{3/4}$ متکی است. برای تأیید قابلیت ND_1^{cp} r دوری در شناسایی تفاضل‌های منقطع، یک تمایزگر عصبی ۲ دوری توسط بنامیرا و همکاران آموزش داده شد. با استفاده از TD_3 در آزمایش D، دقت این تمایزگر عصبی ۲ دوری به ۹۶٫۵۷ درصد رسید که نشان دهنده توانایی آن در شناسایی تفاضل‌های منقطع است.

از منظر یادگیری ماشین، آن‌ها قصد داشتند امکان‌سنجی جایگزینی ND_1^{cp} r دوری را با یک استراتژی ترکیبی الهام‌گرفته از تحلیل رمز تفاضلی و یادگیری ماشین بررسی کنند. برای دست‌یابی به این هدف، آن‌ها به سه جزء اساسی شبکه عصبی پرداختند: لایه پیچشی، بلوک‌های باقیمانده ۱۰ لایه‌ای و بلوک MLP. لایه پیچشی ابتدایی برای تبدیل ورودی از قالب $(C_{0l}, C_{0r}, C_{1l}, C_{1r})$ (که l و r به ترتیب نشان‌دهنده سمت چپ و راست متن هستند) به قالب زیر و در کنار ادغام خطی این عناصر، عمل می‌کند:

$$(\Delta L, \Delta V, V_0, V_1) = (C_{0l} \oplus C_{1l}, C_{0l} \oplus C_{0r} \oplus C_{1l} \oplus C_{1r}, C_{0l} \oplus C_{0r}, C_{1l} \oplus C_{1r})$$

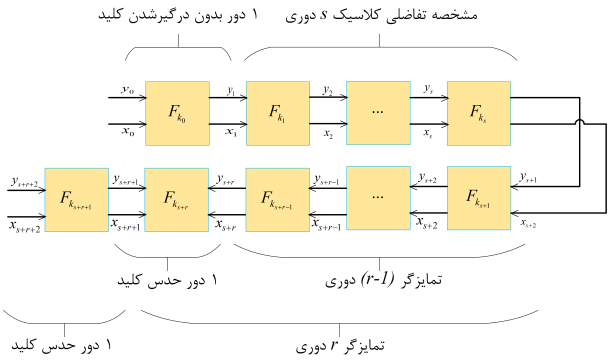
برای بلوک‌های باقیمانده ۱۰ لایه‌ای که در وسط قرار گرفته‌اند، از جدول توزیع خروجی نقاب‌دار (M-ODT [2]) برای جایگزینی آن استفاده کردند. بلوک نهایی MLP با یک طبقه‌بند غیر عصبی به نام ماشین تقویت‌کننده گرادیان سبک (LGBM [3]) جایگزین شد. در نهایت، آن‌ها با موفقیت به یک مدل غیرعصبی دست یافتند که دقت آن تنها ۰٫۶ درصد کمتر از ND_1^{cp} ۵ دوری بود.

علاوه بر کنکاش در عملکرد داخلی ND_1^{cp}، بنامیرا و همکاران روشی را برای بهبود دقت تمایزگر عصبی با استفاده از ورودی‌های متن رمزی به صورت دسته‌ای پیشنهاد کردند که همان‌طور که در بخش‌های ابتدایی بیان شد، این موضوع بعدا در مقاله [۱۲] برای معرفی تمایزگر ND_k^{cp} به کار گرفته شد.

همچنین طبق مبحث مطرح شده در بخش تولید داده مبنی بر مشابهت تمایزگرهای ND_1^{cp} و ND_1^{cd}، نتایج حاصل شده از آزمایشات بنامیرا و همکاران برای ND_1^{cp} را می‌توان به ND_1^{cd} نیز تعمیم داد. این تعمیم با انجام دادن آزمایشات مشابه روی تمایزگر ND_1^{cd} حاصل خواهد شد. از دیدگاه یادگیری عمیق نیز تفاضل متن رمزی (cd) یک ویژگی

[1] Benamira

[2] Masked-output difference table [3] Light gradient boosting machine



شکل ۶. حمله بازیابی کلید $(۱ + s + r + ۱)$ دوری [۲۶]

از تمایزگر زوج متن رمزی (cp) است و در نتیجه، وابستگی‌های cd شامل وابستگی‌های cp می‌شود. از طرفی دقت تمایزگر cp بیشتر است و مشخص است که چیزی بیشتر از تفاضل ورودی یاد می‌گیرد. بنابراین هر چیزی که تمایزگر cd یاد می گیرد، تمایزگر cp نیز یاد خواهد گرفت.

۵ تشریح حمله بازیابی کلید

در مقاله [۳] آرون گور روشی را برای تحلیل رمز عصبی–تفاضلی ابداع کرد که می‌تواند زیرکلیدهای دو دور آخر رمز Speck32/64 را بازیابی کند. در این روش با استفاده از یک زیرکلید حدس زده شده، پیام رمزنگاری شده رمزگشایی می‌شود و سپس از یک تمایزگر عصبی برای محاسبه فاصله بین زیرکلید حدس زده شده و کلید اصلی استفاده می‌شود. فرایند کلی یک حمله بازیابی کلید مبتنی بر یک تمایزگر عصبی–تفاضلی در شکل ۶ نشان داده شده است که در آن ND تمایزگر عصبی–تفاضلی آموزش دیده و F_k تابع دور از رمز به همراه کلید k هستند. همچنین در شکل ۶ هر زوج متن نمونه به صورت کلی با (x_i, y_i) نمایش داده شده که زیروند آن‌ها در هر قسمت مشخص‌کننده دور می‌باشد. حمله بازیابی کلید $(۱+s+r+۱)$ دوری از یک تمایزگر عصبی r دوری اصلی و $(r-۱)$ دوری کمکی استفاده می‌کند که با استفاده از زوج‌های ورودی با تفاضل ΔP آموزش دیده‌اند. یک مشخصه تفاضلی کلاسیک کوتاه s دوری $(\Delta S \to \Delta P)$ با احتمال ۲^{-p} قبل از تمایزگر عصبی قرار می‌گیرد تا تعداد دورهای حمله بازیابی کلید افزایش یابد. برای اطمینان از وجود زوج‌های داده‌ای که تفاضل ΔP را پس از s دور رمز کردن برآورده می‌کنند، طبق احتمال انتشار تفاضل، به حدود $c \cdot ۲^p$ (که با نماد n_{cts} نشان داده می‌شود) زوج داده با تفاضل ΔS نیاز است، که در آن c یک ثابت کوچک است.

از بیت‌های خنثی (NB[1]) تفاضل کلاسیک s دوری برای گسترش هر زوج داده به یک ساختار متشکل از n_b زوج داده استفاده می‌شود. سپس تمامی n_{cts} ساختار متشکل از زوج‌های داده با استفاده از 0 به عنوان زیرکلید برای دریافت ساختارهای متن اصلی رمزگشایی می‌شوند، زیرا عملیات غیرخطی قبل از اضافه کردن کلیدها برای رمز Speck رخ می‌دهد و این کار را برای رمزهای با ساختار مشابه مانند Simon و Simeck نیز می‌توان انجام داد (کلید سفیدسازی وجود ندارد و اولین زیرکلید پس از اولین عملیات غیرخطی XOR می‌شود که دور اول را در حمله تفاضلی آزاد می‌کند). همه ساختارهای متن اصلی رمزنگاری می‌شوند تا ساختارهای رمزنگاری‌شده مربوطه به دست آیند. هر ساختار رمزنگاری‌شده برای انتخاب کاندیدای زیرکلید توسط تمایزگر عصبی اصلی r دوری بر اساس نوعی از بهینه سازی بیزی استفاده می‌شود. الگوریتم ۲ بیان‌گر جستجوی کلید بیزی است.

الگوریتم ۲ (جستجوی کلید بیزی [۳]) به طورکارآمد لیستی از کاندیدهای کلید محتمل را با توجه به ساختار متن رمزی که تفاضل اولیه حمله را برآورده می کند، می‌یابد.

Require: ساختار متن رمزی $C = C_0, \ldots, C_{m-1}$, تمایزگر عصبی ND, تعداد کاندیدای تولیدشده n, تعداد تکرار l

انتخاب به صورت تصادفی و $S = \{k_0, k_1, \ldots, k_{n-1}\} \leftarrow$ بدون جایگزینی از مجموعه تمامی کاندیدای زیرکلید
$L \leftarrow \{\}$
for $j \in \{0, 1, \ldots, l-1\}$ **do**
  $P_{i,k} \leftarrow Decrypt(C_i, k)\ for\ all\ i \in \{0, 1, \ldots, m-1\},\ k \in S$
  $v_{i,k} \leftarrow ND(P_i, k)\ for\ all\ i, k$
  $w_{i,k} \leftarrow \log_2(v_{i,k}/(1 - v_{i,k}))\ for\ all\ i \in \{0, \ldots, m-1\},\ k \in S$
  $w_k \leftarrow \sum_{i=1}^{n} v_{i,k}\ for\ all\ k \in S$
  $L \leftarrow L \| [(k, w_k)\ for\ k \in S]$ (‖:الحاق)
  $m_k \leftarrow \sum_{i=0}^{n-1} v_{i,k}/n\ for\ k \in \{k_0, k_1, \ldots, k_{n-1}\}$
  $\lambda_k \leftarrow \sum_{i=0}^{n-1} (m_{k_i} - \mu_{k_i \oplus k})^2 / \sigma^2_{k_i \oplus k}\ for\ k \in \{0, 1, \ldots, 2^{16} - 1\}$
  $S \leftarrow argsort_k(\lambda)[0 : n-1]$
end for
return L

صرف مقدار یکسانی از محاسبات روی هر ساختار متن رمزی به دست آمده، ناکارآمد است. فرض کنید مجموعه ای از t ساختار متن رمزی داشته باشیم. ابتدا الگوریتم بیزی بر روی هر ساختار امتحان می‌شود. به دلیل احتمالی بودن الگوریتم بیزی اگر هیچ راه حلی پیدا نشود، برای تعداد تکرارهای از پیش تعیین شده‌ای این روند ادامه پیدا می‌کند تا جفت زیرکلید با بالاترین امتیاز برای دو دور آخر را برگردانده شود. در طول این تکرارهای اضافی، باید فعالانه تصمیم گرفته شود که بودجه و منبع محاسباتی برای کدام یک از ساختارهای متن رمزی استفاده شود. این موضوع به عنوان یک مسئله انتخاب چندگانه در نظر گرفته شده و انتخاب و استفاده از ساختارهای متن رمزی با استفاده از یک تکنیک استاندارد کاوش–استخراج که نشات گرفته از یادگیری تقویتی می‌باشد، یعنی UCB[2]، صورت می‌گیرد. در این تکنیک به هر ساختار رمزنگاری‌شده

[1] Neutral bit

[2] Upper confidence bound

بر اساس امتیاز زیرکلیدهای توصیه‌شده و تعداد دفعات بازدید، اولویتی اختصاص داده می‌شود. امتیاز اولویت هر ساختار متن رمزشده براساس رابطه ۱۸ مشخص می‌شود. در رابطه ۱۸ $\omega^i_{\max}$ بیانگر بالاترین امتیاز تمایزگر، n_i تعداد تکرارهای قبلی روی ساختار متن رمزشده i، j تعداد تکرارهای فعلی و n_c تعداد ساختارهای متن رمزی در دسترس هستند [۳].

$$s_i = \omega^i_{\max} + \sqrt{n_c} \cdot \sqrt{\log_2{}^{(j)}/n_i} \qquad (۱۸)$$

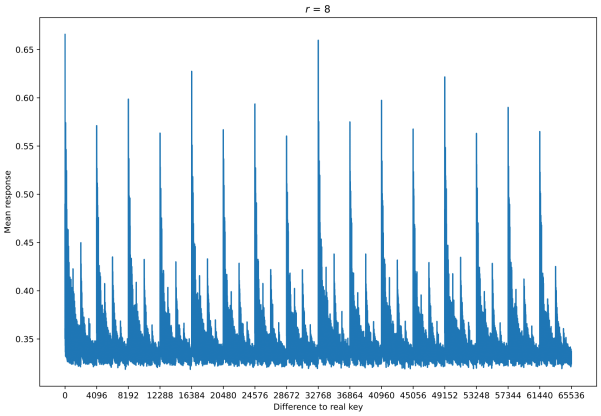
با در نظر گرفتن $v_{i,k}$ به عنوان امتیاز یا پاسخ تمایزگر عصبی به کلید k برای متن iام، بدون انجام رمزگشایی آزمایشی جامع، سیاست جستجوی کلید به پاسخ $v_{i,k}$ تمایزگر عصبی هنگام رمزگشایی با کلید اشتباه بستگی دارد. نمایه پاسخ کلید اشتباه [۱] برای توصیه مقادیر کاندیدای جدید از مقادیر کاندیدای قبلی در حالی که فاصله اقلیدسی وزنی را در الگوریتم جستجوی کلید بیزی به حداقل می‌رساند، به کار گرفته می‌شود [۲۶].

در مقاله [۳]، آرون گور با استفاده از تمایزگر ۷ دوری و اضافه کردن یک دور به ابتدا و انتهای آن، حمله بازیابی کلید ۹ دوری را با نرخ موفقیت ۳۵۸ درصد اعمال کرد. سپس با به‌کارگیری سیاست جستجوی کلید بیزی و حمله ۹ دوری به‌دست‌آمده، حمله ۱۱ دوری را با اضافه کردن مشخصه تفاضلی کلاسیک به تمایزگر عصبی و پیچیدگی زمانی $۲^{۳۸}$ به دست آورد که نسبت به حملات قبلی (با پیچیدگی $۲^{۴۶}$) بهبود حاصل شد.

۱.۵ نمایه پاسخ با کلید اشتباه

از آنجایی که زمان امتیازدهی تمامی فضای کلید بسیار زیاد است، گور کلیدهای کاندید با نمره بالا را با روشی تکرارشونده بر اساس بهینه سازی بیزی جستجو کرد. سیاست جستجوی کلید مبتنی بر بهینه‌سازی بیزی تعداد رمزگشایی‌های آزمایشی را به شدت کاهش می‌دهد. ایده اصلی این سیاست، فرضیه تصادفی‌سازی کلید اشتباه [۲] است. این فرضیه زمانی که فقط یک دور رمزگشایی آزمایشی انجام می‌شود، به خصوص در یک رمز سبک‌وزن، صادق نیست. پاسخ مورد انتظار ND در رمزگشایی با کلید اشتباه به تفاوت بیتی بین کلیدهای آزمایشی و واقعی بستگی دارد. این نمایه پاسخ با کلید اشتباه را می‌توان در یک پیش محاسبه ثبت کرد [۱۵].

به منظور تولید کلیدهای کاندید جدید در هر تکرار، گور از توزیع ناهموار سیگنال خروجی تمایزگر عصبی به ازای رمزگشایی با کلید اشتباه استفاده کرد. سپس پاسخ به کلید اشتباه را می توان به عنوان یک متغیر تصادفی مرتبط با اختلاف کلید واقعی و زیرکلید حدس زده شده در نظر گرفت که از توزیع گوسی با میانگین μ و انحراف استاندارد σ پیروی می‌کند. با استفاده از تفاوت در توزیع کلیدهای اشتباه مختلف، می توان کلید حدس زده شده را تولید کرد. اگر امتیاز یک کلید کاندید از یک سطح آستانه انتخابی بیشتر شود، تکرار به پایان می‌رسد [۱۳].



شکل ۷. نمایه پاسخ کلید اشتباه برای ۸ دور از رمز Speck32/64 [۲۶]

یک نمونه از نمایه پاسخ کلید اشتباه بعد از حمله کلید مرتبط به رمز Speck32/64 ۸ دوری توسط مقاله [۲۶] را در شکل ۷ می‌توان مشاهده کرد. شکل ۷ پاسخ میانگین برای فاصله‌های همینگ مختلف بین کلیدهای آزمایشی و واقعی را نشان می‌دهد. در نمودار این شکل، محور افقی بیانگر اختلاف یا تفاضل بین کلید واقعی و حدس زده شده و محور عمودی نشان‌دهنده پاسخ میانگین تمایزگر است. با توجه به شکل ۷ هنگامی که اختلاف در کلیدها کم است، به خصوص اگر اختلاف مربوط به مقادیر $\{۱۶۳۸۴, ۳۲۷۶۸, ۴۹۱۵۲\}$ باشد، پاسخ و امتیازهای بالایی به وجود می‌آیند. این نشان می‌دهد که خطاها در بیت‌های ۱۴ و ۱۵ زیرکلید تأثیر حداقل بر امتیازات دارند که امکان کاهش فضای حدس کلید را فراهم می‌کند.

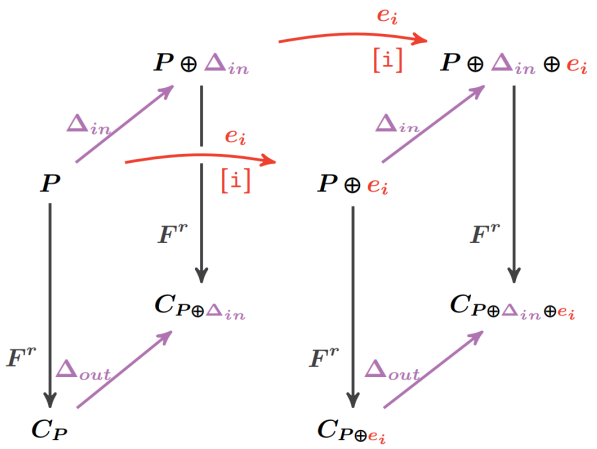
۲.۵ تمایزگرهای با پاسخ ترکیب‌شده

به عنوان یک تکنیک جدید و متفاوت از تمایزگرهای قبلی، استفاده از تمایزگرهای عصبی (ND) محدودیت‌هایی دارد. وقتی دقت آن به صورت حاشیه‌ای بیشتر از ۰٫۵ می‌شود، استفاده از آن در حملات بازیابی کلید سخت است. بنابراین، گور از پاسخ ترکیبی (مطابق با معادله ۱۹) تمایزگر عصبی بر روی تعداد زیادی نمونه از توزیع مشابه به عنوان یک تمایزدهنده استفاده می‌کند (که به عنوان تمایزگر پاسخ ترکیبی، CRD [۳] نامیده می‌شود).

$$w_{CRD} = \sum_{i=۱}^{n_b-۱} \log_2 \left(\frac{v_i}{۱ - v_i} \right) \qquad (۱۹)$$

در رابطه ۱۹، w_{CRD} و v_i به ترتیب بیانگر پاسخ ترکیب‌شده تمایزگر و پاسخ تمایزگر به نمونه i هستند. برای توضیحات بیشتر به مقاله [۲۵] رجوع شود.

[1] Wrong key response profile
[2] Wrong key randomization hypothesis
[3] Combined response distinguisher



شکل ۸. تعریف بیت خنثی [۲۵]

۳.۵ بیت‌های خنثی و بیت‌های خنثی تعمیم‌یافته

به طور کلی با توجه به ساختار فرایند بازیابی کلید که از دو بخش اصلی تمایزگر عصبی و مشخصه تفاضلی کوتاه s دوری تشکیل می‌شود، بهبود و بالا بردن موفقیت بازیابی کلید به گسترش و بهبود این دو بخش وابسته است. برای توسعه مشخصه تفاضلی کلاسیک ابتدایی می‌توان بیت‌های خنثی آن مشخصه را به کارگرفت تا از هر زوج داده، ساختاری از زوج‌های داده به دست آورد.

به دلیل وجود مشخصه تفاضلی کلاسیک در تمایزگرهای ترکیبی، تجمع تعداد کافی نمونه از یک توزیع مشابه برای تغذیه به تمایزگر عصبی ساده نیست و این مورد بر عملکرد CRD تأثیر می‌گذارد. برای رفع این مشکل، گور از بیت‌های خنثی برای تولید تعداد کافی نمونه از یک توزیع مشابه استفاده می‌کند. در مقاله [۲۵]، بائو[1] و همکاران، توضیحات دقیق‌تری درباره بیت‌های خنثی ارائه داده‌اند. در ادامه، تعاریف انواع مختلفی از بیت‌های خنثی و سایر عناصر مرتبط به طور خلاصه بیان می‌شوند.

تعریف ۱. فرض کنید تفاضلی به نام $\delta : \Delta_{in} \to \delta_{out}$ برای r دور وجود داشته باشد. یک زوج متن اصلی $(P, P \oplus \Delta_{in})$ و زوج متن‌های رمزشده متناظر (C_0, C_1) که r دور رمزنگاری شده‌اند را در نظر بگیرید. اگر $C_0 \oplus C_1 = \Delta_{out}$ باشد، زوج $(P, P \oplus \Delta_{in})$ به عنوان زوج منطبق[2] برای δ نامیده می‌شود.

تعریف ۲. فرض کنید e_i تفاضلی با تنها یک بیت فعال که i جایگاه بیت غیر صفر آن است، باشد. در این صورت بیت i یک بیت خنثی از δ است، اگر برای هر زوج منطبق $(P, P \oplus \Delta_{in})$، زوج $(P \oplus e_i, P \oplus \Delta_{in} \oplus e_i)$ نیز یک زوج منطبق باشد. شکل ۸ نشان‌دهنده بیت خنثی می‌باشد.

زوج منطبق و بیت‌های خنثی آن را می‌توان با جستجوی جامع پیدا کرد. به طور معمول، هر چه احتمال یک مشخصه تفاضلی کمتر باشد، بیت‌های خنثی کمتری خواهد داشت. با توجه به اینکه بیت‌های خنثی برای اکثر تفاضل‌ها کمیاب هستند، بیت‌های خنثی تعمیم‌یافته در مقاله [۲۵] نشان داده شده‌اند.

[1] Bao [2] Conforming pair

جدول ۴. متغیرهای حمله بازیابی دو زیرکلید آخر

تعریف	متغیر
تعداد مقادیر ممکن برای چند بیت حدس زده شده k_0	n_{kg}
تعداد ساختارهای متن رمزی، تعداد زوج متن رمزشده در هر ساختار متن رمزی $(2^{|NB|+1})$	n_{cts}, n_b
تعداد کل تکرارها روی ساختارهای متن رمزی	n_{it}
مقادیر قطع (Cutoff) با توجه به امتیازهای دو زیرکلید آخر توصیه‌شده	c_1 و c_2
تعداد تکرارها و تعداد نامزدهای کلید در n_{byit1} و n_{cand1}، n_{byit2} و n_{cand2} هر تکرار در رویه‌های جستجوی کلید بیزی	

تعریف ۳. فرض کنید e_i تفاضلی با تنها یک بیت فعال که i جایگاه بیت غیر صفر آن است، باشد. بیت iیک بیت خنثی احتمالی[3] از δ است، در صورتی که اگر $(P, P \oplus \Delta_{in})$ یک زوج منطبق باشد، رویدادی که $(P \oplus e_i, P \oplus \Delta_{in} \oplus e_i)$ یک زوج منطبق باشد قطعی نیست و دارای احتمال است.

تعریف ۴. فرض کنید $I_s = \{i_1, i_2, \ldots, i_s\}$ یک مجموعه از اندیس بیت‌ها باشد. I_s یک مجموعه بیت همزمان خنثی[4] از δ است اگر برای هر زوج منطبق $(P, P \oplus \Delta_{in})$، زوج

$$(P \oplus (\oplus_{i \in I_s} e_i), P \oplus \Delta_{in} \oplus (\oplus_{i \in I_s} e_i))$$

نیز یک زوج منطبق باشد؛ در صورتی که برای هر زیرمجموعه از I_s، منطبق بودن زوج حاصل همیشه برقرار نیست.

جستجوی جامع برای همه مجموعه بیت‌های همزمان خنثی برای اکثر تفاضل‌ها دشوار است. به طور کلی، اندازه کاندیدای مجموعه بیت همزمان خنثی کوچک است.

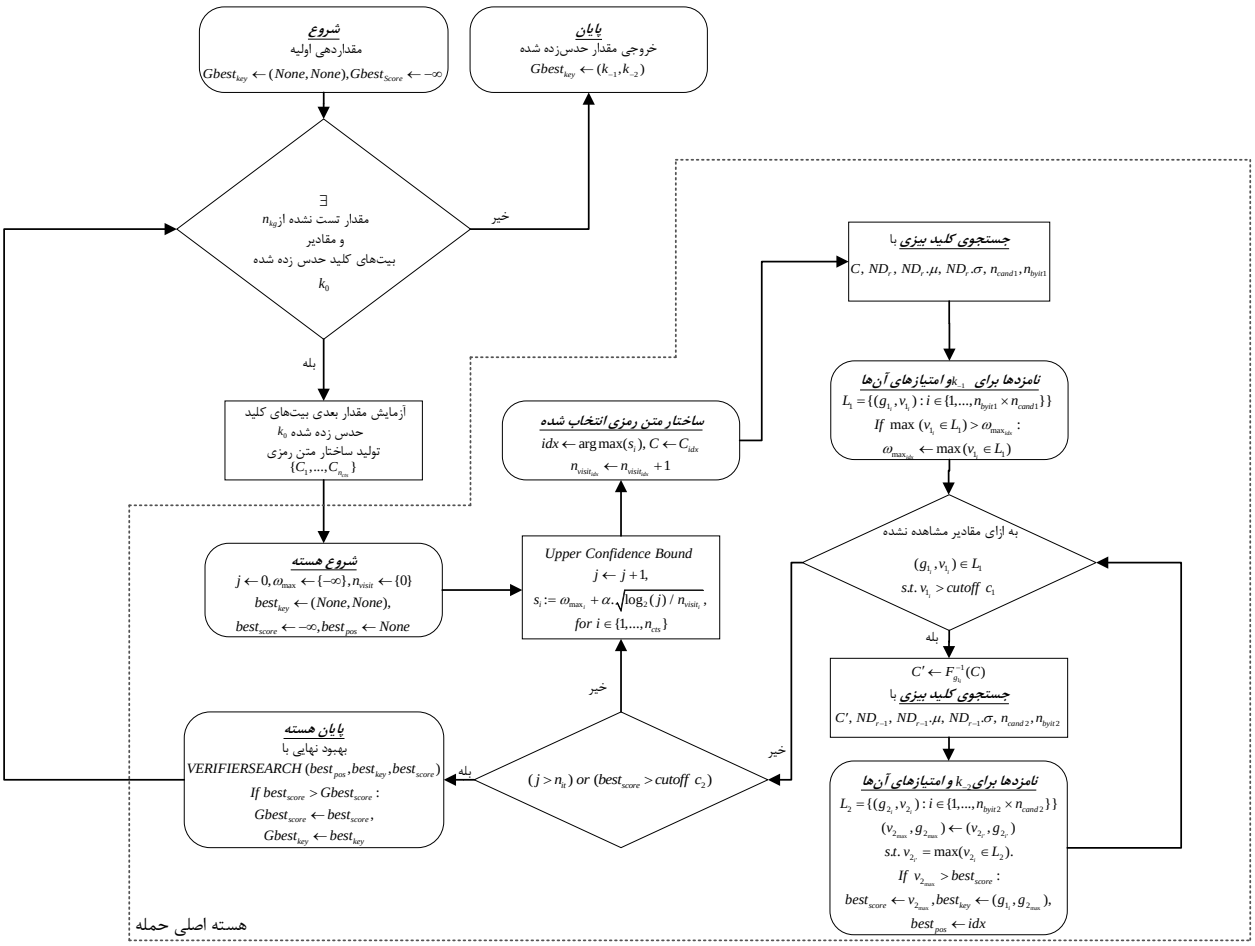
با توجه به مطالب بیان‌شده برای حمله بازیابی کلید، چارچوب کلی حملات بازیابی کلید در شکل ۹ نشان داده شده است. همچنین متغیرهای به کار برده شده برای بازیابی دو زیرکلید آخر در جدول ۴ تعریف شده‌اند.

۶ مقایسه تحلیل تفاضلی رمزهای قالبی

در این قسمت با توجه به مطالب بیان‌شده در بخش‌های قبلی شامل شکل ورودی، تفاضل ورودی، روش‌های تولید مجموعه‌داده و آموزش، معماری شبکه و چارچوب بازیابی کلید، بهترین نتایج حملات تمایزگر عصبی–تفاضلی و بازیابی کلید به دست آمده به هر یک از رمزهای قالبی Speck32/64، Simon32/64 و Simeck32/64 را که در مطالعات مرتبط گزارش شده‌اند، با یکدیگر بررسی و مقایسه می‌نماییم. جداول ۵ و ۶ به ترتیب نشان‌دهنده مقایسه تمایزگرها و مقایسه حملات بازیابی کلید هستند.

در جدول ۵ با توجه به داده‌های به دست از آمده از مقالات مختلف،

[3] Probabilistic neutral bit [4] Simultaneous-neutral bit-set



شکل ۹. چارچوب حملات بازیابی کلید [۲۵]

بهترین نتایج تمایزگرهای کنونی روی سه رمز قالبی مورد بررسی برای تعداد دورهای مختلف از کمترین تا بیشترین دقت ذکر شده‌اند. بر این اساس در جدول ۶ نیز تمایزگرهایی که منجر به حملات بازیابی کلید شده‌اند، آورده شده است. در ادامه به تحلیل جدول ۵ براساس پارامترهای موجود در هر ستون می‌پردازیم.

با توجه به معماری‌های متفاوت شبکه، همان‌طور که مشاهده می‌شود ساختار باقی‌مانده یا همان ResNet، به دلیل عملکرد مناسب در مسئله طبقه‌بندی و موفقیت در استخراج ویژگی‌های مناسب برای تحلیل رمزهای قالبی و ایجاد تمایزگر، تقریبا در تمامی تمایزگرهای به دست آمده مورد استفاده قرار گرفته و بهترین نتایج را به همراه داشته است. همچنین با ترکیب شبکه باقی‌مانده با دیگر ساختارهای موجود مانند ماژول تلقین و SeNet، در ازای افزایش پیچیدگی مدل و افزایش زمان آموزش، با توجه به تفاوت در نحوه استخراج ویژگی هر یک از ساختارهای مذکور، ویژگی‌های بیشتری برای ایجاد تمایزگر توسط شبکه استخراج شده و در نتیجه دقت بالاتر و افزایش تعداد دورهای تمایزگر از این طریق حاصل می‌شود. بر این اساس انتخاب معماری مناسب برای مدل تمایزگر و ترکیب هوشمندانه چندین شبکه درکنار یکدیگر، پارامتری مهم برای افزایش دقت و تعداد دورهای تمایزگر عصبی است.

البته باید به این نکته مهم توجه داشت که افزایش پیچیدگی شبکه‌های عصبی عمیق لزوماً منجر به بهبود دقت نمی‌شود. در حالی که مدل‌های عمیق‌تر و پیچیده‌تر می‌توانند قابلیت‌های بیشتری برای یادگیری ویژگی‌های پیچیده از داده‌ها داشته باشند، مواردی مانند بیش‌برازش [1] روی داده‌های آموزشی، نیاز به داده‌های آموزشی، زمان آموزش و منابع محاسباتی بیشتر و نیز مشکلات بهینه سازی مانند همگرایی و گرفتار شدن در کمینه‌های محلی باید در نظر گرفته شوند.

تنوع در روش‌های آموزش موضوع مهم دیگری است که باید مورد توجه قرار گیرد. از میان روش‌های آموزش موجود با توجه به بخش ۳.۴ و نتایج حاصل شده برای تمایزگرها در جدول ۵، می‌توان دریافت که روش‌های آموزش مرحله‌ای در ازای پیچیدگی زمانی و داده‌ی بیشتر، دقت و تعداد دور تمایزگر بیشتری را به همراه خواهند داشت. برخلاف روش آموزش مرحله‌ای، روش‌های میانگین‌گیری کلید و پایه‌ای مستقیم دقت پایین‌تری را در کنار پیچیدگی کمتر خواهند داشت. همچنین به کارگیری

[1] Overfitting

روش آموزش لایه انجماد در مقاله [۲۶] در کنار مواردی مانند ترکیب دو معماری ResNet و ماژول تلقین، تنها تمایزگر ۱۰ دوری تفاضلی کلید مرتبط بر پایه یادگیری عمیق را به همراه داشته است.

دیگر پارامتر مهم که تاثیر به‌سزایی در افزایش دقت و تعداد دور تمایزگرهای عصبی به همراه دارد، تعداد متن رمزی در هر نمونه مجموعه داده (k) می‌باشد. مشاهده می‌شود که با افزایش این پارامتر که مقدار آن معمولا توانی از عدد ۲ در نظر گرفته می‌شود، شبکه عصبی می‌تواند روابط و وابستگی‌های بیشتری بین متن‌های رمزی را یاد گرفته و در جهت بهبود مدل عمل کند. به عنوان نمونه برای الگوریتم Simon، با افزایش مقدار k از ۳۲ به ۶۴ در کنار تغییر روش آموزش از پایه‌ای مستقیم به ترکیب امتیاز، دقت تمایزگر ۱۱ دوری از ۶۰٫۸۱ به ۶۷٫۵ درصد افزایش داشته است.

نوع ویژگی‌های ورودی به همراه استفاده از محاسبات بازگشتی در پردازش متن رمزی از اهمیت بالایی برخوردارند، که در تولید ویژگی‌ها و ساخت مجموعه داده‌های آموزشی نقش کلیدی دارند. این موضوع به دقت در بخش ۱.۳ مورد بررسی قرار گرفته است. بر اساس داده‌های ارائه شده در جدول ۵، استفاده از ترکیب متن رمزی و تفاضل بیت‌های آن، در کنار به کارگیری محاسبات بازگشتی، موجب تامین ویژگی‌های افزودنی برای شبکه می‌شود که این امر به نوبه خود به افزایش دقت مدل تمایزگر عصبی منجر می‌شود.محاسبات بازگشتی به شبکه‌های عصبی اجازه می‌دهند تا از اطلاعات مربوط به دورهای پیشین استفاده کنند. بر اساس داده‌های جدول ۵، مشخص است که با شرایط یکسان برای تمایزگر ۸ دوری الگوریتم رمز Speck، افزودن ویژگی‌های حاصل از محاسبات بازگشتی به متن رمزی، می‌تواند دقت تمایزگر را از ۵۶٫۴۹ درصد به ۶۵٫۰۲ درصد بهبود ببخشد، که نشان‌دهنده‌ی ارتقاء قابل توجهی در عملکرد است. این تغییر دقت می‌تواند نتیجه‌ی بهره‌برداری مؤثر از وابستگی‌های موجود در داده‌های رمزنگاری شده باشد، که امکان فهم بهتر الگوهای موجود در متن رمزی را به تمایزگر می‌دهد.

همچنین در تحلیل عملکرد تمایزگرهای عصبی در سناریوی حملات کلید مرتبط، فرض می‌شود که مهاجم قادر به رمزنگاری با کلیدهایی است که به شکلی معین به هم مرتبط هستند. این قابلیت به تمایزگر اجازه می‌دهد تا اطلاعات بیشتری از وابستگی‌های بین کلیدهای مختلف استخراج کند و به این ترتیب، دقت تشخیص خود را افزایش دهد. در نتیجه، تمایزگرهای عصبی–تفاضلی که بر این اساس آموزش دیده‌اند، موفق به کسب دقت‌های بالاتر و انجام تعداد دورهای بیشتری در تحلیل الگوریتم‌های رمزنگاری مانند Speck، Simon و Simeck شده‌اند که بدون استراتژی کلید مرتبط قابل دستیابی نیستند.

به عنوان مثال، بر اساس داده‌های ارائه شده در جدول ۵، بیشترین تعداد دورهای تشخیص توسط تمایزگرهای عصبی–تفاضلی کلید مرتبط برای الگوریتم‌های رمز Speck، Simon و Simeck به دست آمده است که این نشان دهنده تأثیرگذاری استراتژی کلید مرتبط در بهبود عملکرد این تمایزگرها است. بالاترین تعداد دور تشخیص برای هر یک از این الگوریتم‌ها به ترتیب ۱۰، ۱۳ و ۱۵ دور است که با دقت‌های ۵۶٫۴۳، ۵۴٫۳۷ و ۵۴٫۶۷ درصد همراه بوده‌اند.

این داده‌ها نشان می‌دهند که حملات کلید مرتبط می‌توانند به عنوان یک ابزار قدرتمند برای بهبود توانایی تمایزگرهای عصبی در تشخیص و تفسیر الگوریتم‌های رمزنگاری استفاده شوند. همچنین، این نتایج اهمیت داشتن داده‌های کافی و مرتبط برای آموزش مدل‌های یادگیری عمیق را تأیید می‌کنند، به ویژه در زمینه‌هایی که داده‌ها دارای وابستگی‌های پیچیده‌ای هستند.

حال به سراغ بررسی داده‌های موجود در جدول ۶ می‌رویم. همانطور که در بخش ۵ توضیح داده شد، برای انجام یک حمله بازیابی کلید مبتنی بر یادگیری عمیق، نیازمند یک مشخصه کلاسیک s–دوری ابتدایی با احتمال مناسب که دارای تعداد کافی بیت‌های خنثی باشد و همچنین یک تمایزگر عصبی r–دوری با دقت بالا هستیم. اگر احتمال مشخصه s–دوری ابتدایی پایین باشد، حجم داده‌های لازم برای اجرای حمله بازیابی کلید به شدت افزایش می‌یابد. بنابراین، با دستیابی به یک تمایزگر عصبی با دقت بالا و افزودن مشخصه s–دوری به ابتدای آن و همچنین جستجو برای بیت‌های خنثی مرتبط با این مشخصه، حمله بازیابی کلید مطابق توضیحات بخش ۵ انجام خواهد شد.

با توجه به پارامترهای موجود در جدول ۵، پیچیدگی داده به صورت تئوری مطابق رابطه ۲ $n_{cts} \times n_b \times k \times 2$ محاسبه می‌شود. البته در هنگام آزمایش واقعی، زمانی که دقت تمایزگر عصبی–تفاضلی بالا باشد، تمامی داده مورد استفاده قرار نمی‌گیرد و در نتیجه پیچیدگی داده واقعی کمتر از مقدار تئوری آن خواهد بود.

همچنین محاسبه پیچیدگی زمانی نیز به صورت کلی طبق رابطه $dr \times rt$ صورت می‌پذیرد. در این رابطه dr بیانگر نرخ رمزگشایی برای الگوریتم رمز مورد نظر برحسب دور بر ثانیه است که به سخت‌افزار و محیط آزمایش وابسته است. rt نیز به معنای میانگین زمان اجرا است که باید در طیف وسیعی از آزمایش‌ها مشاهده شود. در نهایت نرخ موفقیت (sr) نسبت تعداد زیرکلیدهایی که با موفقیت بازیابی شده‌اند به کل تعداد آزمایش‌های انجام شده را اندازه‌گیری می‌کند. در مراجع [۱۵، ۲۶] پیچیدگی زمانی براساس رابطه $dr \times rt \times ne$ محاسبه شده و موفقیت یک آزمایش زمانی تعیین می‌شود که نرخ موفقیت کلی به ۹۹ درصد برسد. این مورد معادل با تعداد آزمایشی (ne) است که معادله $1-(1-sr)^{ne} = 0.99$ را تحقق می‌بخشد.

همان‌طور که در جدول ۶ مشاهده می‌شود، حملات بازیابی کلید ۱۳ و ۱۴ دوری برای الگوریتم رمز Speck بر پایه تمایزگرهای عصبی–تفاضلی کلید مرتبط مقاله [۲۶] صورت گرفته‌اند که با این استراتژی موجب کاهش فضای کلید، پیچیدگی زمانی و داده مورد نیاز نیز شده‌اند. در مقاله [۳۱] با وجود تمایزگرهای عصبی–تفاضلی کلید مرتبط برای دو الگوریتم رمز Simon و Simeck، هیچ گونه حمله بازیابی کلید انجام نشده است. بر این اساس برای این دو الگوریتم، بهترین نتایج مربوط به حملات ۱۷

جدول ۵. مقایسه تمایزگرهای عصبی

مرجع	دقت %	تعداد دور	استفاده از محاسبات بازگشتی متن رمزشده	نوع ویژگی‌های ورودی	k	روش آموزش	معماری شبکه	رمز
[۲۰]	۷۷٫۹	۶	خیر	متن رمزی و تفاضل	۱	پایه‌ای مستقیم	ResNet + LGBM	Speck32/64
[۳]	۵۹٫۱	۷	خیر	متن رمزی	۱	میانگین‌گیری کلید	ResNet	
[۱۲]	۶۶٫۹۴	۷	خیر	متن رمزی	۱۶	پایه‌ای مستقیم	ResNet	
[۳۰]	۶۹٫۳۹	۷	خیر	تفاضل بیت‌های جزیی متن رمزی	۴	پایه‌ای مستقیم	Separable Convolution	
[۱۴]	۹۷٫۱۳	۷	بله	متن رمزی و تفاضل	۳۲	پایه‌ای مستقیم	ResNet + Inception	
[۱۹]	۹۹٫۱	۷	خیر	متن رمزی	۶۴	ترکیب امتیاز	ResNet	
[۳]	۵۱٫۴	۸	خیر	متن رمزی	۱	مرحله‌ای	ResNet	
[۲۲]	۵۱٫۴	۸	خیر	متن رمزی	۲	مرحله‌ای ساده	DbitNet	
[۱۳]	۵۶٫۴۹	۸	خیر	تفاضل	۳۲	پایه‌ای مستقیم	ResNet	
[۱۱]	۶۵٫۰۲	۸	بله	تفاضل	۳۲	پایه‌ای مستقیم	ResNet	
[۲۷]	۵۰٫۵	۹	خیر	متن رمزی	۱۶	مرحله‌ای	ResNet + Inception	
[۲۶]	۵۶٫۴۳	۱۰	بله	متن رمزی (کلید مرتبط)	m	لایه انجماد	ResNet + Inception	
[۲۰]	۸۳٫۴	۸	خیر	متن رمزی و تفاضل	۱	پایه‌ای مستقیم	ResNet + LGBM	Simon32/64
[۱۳]	۶۱٫۰۹	۱۰	خیر	تفاضل	۳۲	پایه‌ای مستقیم	ResNet	
[۲۵]	۵۱٫۷۴	۱۱	خیر	متن رمزی	۱	مرحله‌ای	SeNet	
[۲۲]	۵۱٫۸	۱۱	خیر	متن رمزی	۲	مرحله‌ای ساده	DbitNet	
[۱۱]	۶۰٫۸۱	۱۱	بله	تفاضل	۳۲	پایه‌ای مستقیم	ResNet	
[۱۹]	۶۷٫۵	۱۱	خیر	متن رمزی	۶۴	ترکیب امتیاز	ResNet	
[۲۷]	۵۲٫۲۵	۱۲	خیر	متن رمزی	۱۶	مرحله‌ای	ResNet + Inception	
[۳۱]	۵۴٫۳۷	۱۳	بله	متن رمزی و تفاضل (کلید مرتبط)	۸	مرحله‌ای	Se-ResNet	
[۳۲]	۵۴٫۳۸	۱۰	خیر	متن رمزی	۱	پایه‌ای مستقیم	ResNet	Simeck32/64
[۳۳]	۵۰٫۲۲	۱۱	خیر	متن رمزی	۱ ($۴ = t$)	پایه‌ای مستقیم	ResNet	
[۱۵]	۵۱٫۶۱	۱۲	بله	متن رمزی و تفاضل	۸	مرحله‌ای	ResNet + Inception	
[۳۱]	۵۴٫۶۷	۱۵	بله	متن رمزی و تفاضل (کلید مرتبط)	۸	مرحله‌ای	Se-ResNet	

جدول ۶. مقایسه حملات بازیابی کلید

مرجع	فضای کلید	نرخ موفقیت %	داده	زمان	پیکربندی	تمایزگر	دور	رمز
[۳]	2^{64}	۵۲	$2^{14.5}$	2^{38}	۱ + ۲ + ۷ + ۱	تمایزگر عصبی	۱۱	Speck32/64
[۱۲]	2^{64}	۵۴٫۴	$2^{14.5}$	2^{36}	۱ + ۲ + ۷ + ۱	تمایزگر عصبی	۱۱	
[۳]	2^{64}	۴۰	$2^{22.87}$	$2^{46.57}$	۱ + ۲ + ۸ + ۱	تمایزگر عصبی	۱۲	
[۲۵]	2^{63}	۸۲	2^{29}	$2^{50.17}$	۱ + ۳ + ۸ + ۱	تمایزگر عصبی	۱۳	
[۲۷]	2^{63}	۴۹	2^{31}	$2^{52.68}$	۱ + ۳ + ۸ + ۱	تمایزگر عصبی	۱۳	
[۲۶]	2^{46}	۴۳٫۳۳	2^{10}	$2^{31.79}$	۱ + ۲ + ۹ + ۱	تمایزگر عصبی کلید مرتبط	۱۳	
[۲۶]	2^{41}	۷۱٫۴۳	2^{15}	$2^{35.78}$	۱ + ۳ + ۹ + ۱	تمایزگر عصبی کلید مرتبط	۱۴	
[۱۳]	2^{64}	۹۳	$2^{12.5}$	$2^{16.4}$	۱ + ۲ + ۹ + ۱	تمایزگر عصبی	۱۳	Simon32/64
[۲۵]	2^{64}	۴۹	2^{21}	$2^{46.98}$	۱ + ۳ + ۱۱ + ۱	تمایزگر عصبی	۱۶	
[۲۷]	2^{64}	۸۰	2^{22}	$2^{42.79}$	۱ + ۳ + ۱۱ + ۱	تمایزگر عصبی	۱۶	
[۲۷]	2^{64}	۹	2^{28}	$2^{54.01}$	۱ + ۴ + ۱۱ + ۱	تمایزگر عصبی	۱۷	
[۳۴]	2^{64}	۹۸	$2^{17.7}$	$2^{32.8}$	۱ + ۲ + ۹ + ۱	تمایزگر عصبی	۱۳	Simeck32/64
[۳۲]	2^{64}	۸۸	2^{23}	$2^{32.99}$	۱ + ۳ + ۹ + ۱	تمایزگر عصبی	۱۴	
[۱۵]	2^{64}	۹۹٫۱۷	2^{22}	$2^{35.309}$	۱ + ۳ + ۱۰ + ۱	تمایزگر عصبی	۱۵	
[۱۵]	2^{64}	۱۰۰	2^{24}	$2^{38.189}$	۱ + ۳ + ۱۱ + ۱	تمایزگر عصبی	۱۶	
[۱۵]	2^{64}	۳۰	2^{26}	$2^{45.037}$	۱ + ۳ + ۱۲ + ۱	تمایزگر عصبی	۱۷	

دوری گزارش شده‌اند.

با مقایسه حملات ۱۶ و ۱۷ دوری الگوریتم رمز Simon بر پایه تمایزگر عصبی ۱۱ دوری، مشاهده می‌شود که با افزایش یک دوری مشخصه کلاسیک از ۳ به ۴ دور، به علت کاهش احتمال و تعداد بیت‌های خنثی برای آن، با وجود افزایش پیچیدگی داده و زمانی، نرخ موفقیت برای حمله ۱۷ دوری به ۹ درصد کاهش یافته است.

همچنین با توجه به جدول ۶ و حملات ۱۶ و ۱۷ دوری الگوریتم رمز سایمک، با وجود استفاده از مشخصه کلاسیک ۴ دوری برای هر دو حمله، به علت کمتر بودن دقت تمایزگر عصبی ۱۲ دوری در مقایسه با تمایزگر عصبی ۱۱ دوری، در کنار افزایش پیچیدگی داده و زمانی، نرخ موفقیت حمله ۱۷ دوری نسبت به ۱۶ دور به مقدار ۷۰ درصد کاهش یافته است.

در نهایت با نگاه کلی به دو جدول ۵ و ۶، افزون بر عملکرد بهتر حملات با کمک یادگیری عمیق نسبت به حملات کلاسیک از دیدگاه دقت تمایزگرها، پیچیدگی زمانی و نرخ موفقیت، با توجه به این جداول دریافت می‌شود که با به‌کارگیری شکل داده ورودی مناسب به شبکه با اطلاعات اضافی بیشتر و بهتر، تفاضل ورودی بهینه برای تمایزگرهای عصبی، معماری شبکه مناسب برای آموزش و استخراج ویژگی‌های اضافه‌تر، روش‌های تولید داده و روش‌های آموزش خلاقانه و بهینه، خودکارسازی فرایند تحلیل و همچنین ترکیب با مفاهیم تحلیل رمز کلاسیک و تمایزگرهای مبتنی بر جدول توزیع تفاضل، می‌توان به تعداد دورهای بالاتر، دقت و نرخ موفقیت بیشتر در کنار پیچیدگی داده و زمانی کمتر دست‌یافت.

۷ گسترش و تعمیم به دیگر حملات کلاسیک

پس از مقاله گور در سال ۲۰۱۹ [۳]، تحلیل تفاضلی مورد توجه بسیاری از پژوهشگران در زمینه تحلیل رمز با استفاده از یادگیری عمیق قرار گرفت و نتایج قابل توجهی از تحقیقات آن‌ها به دست آمد که در بخش‌های قبل مورد بررسی و مقایسه قرار گرفتند. اما از سوی دیگر برخی مطالعات تمرکز خود را علاوه بر اعمال تمایزگرهای عصبی–تفاضلی به الگوریتم‌های رمز مختلف، بر روی به‌کارگیری حملات متفاوت تحلیل رمز کلاسیک و تعمیم به آن حملات قرار داده‌اند.

از حملات دیگری که با کمک یادگیری عمیق روی رمزهای قالبی اعمال شده‌اند می‌توان به حمله خطی [۳۵، ۳۶]، حمله XOR چرخشی (RX[1]) [۳۷]، حمله انتگرالی [۳۸] و حملات تفاضلی کلید مرتبط [۲۶، ۳۱] اشاره نمود. این دسته از مطالعات نیز در جایگاه گسترش از حمله تفاضلی به دیگر حملات کلاسیک موفقیت‌آمیز بوده‌اند و این نوید را می‌دهند که می‌توان کار گور را که با استفاده از حمله تفاضلی صورت گرفته است، به دیگر حملات برجسته تحلیل رمز کلاسیک گسترش داد. اما تا به اکنون علاوه بر محدود بودن کارهای انجام شده و تعداد اندک آن‌ها، نتایج به‌دست‌آمده

[1] Rotational XOR

برخلاف تحلیل تفاضلی، اغلب بدتر از نتایج تحلیل کلاسیک هستند و همچنین روی تعداد بسیار کمی از الگوریتم‌های رمز اعمال شده‌اند. به عنوان یک تحلیل متن اصلی-معلوم قدرتمند، از تحلیل خطی به طور گسترده در رمزهای قالبی مختلف استفاده شده است؛ بنابراین مطالعه بر روی تحلیل رمز خطی به کمک یادگیری ماشین، معنادار و مناسب است. در سال ۲۰۲۰، هو [1] و همکاران [۳۵]، روشی را برای ترکیب تحلیل رمز خطی با شبکه‌های عصبی پیشنهاد دادند.

کار آن‌ها داده‌های آموزشی را با انتخاب بیت‌های جزئی از زوج‌های متن اصلی و متن رمزشده می‌سازد و سپس لایه‌گذاری اضافی انجام می‌شود تا اندازه نمونه آموزشی به طول ثابت برسد. بر این اساس، هو و همکاران، حملات بازیابی کلید یک بیتی و چند بیتی را بر روی الگوریتم رمز DES [۳۹] کاهش دور یافته انجام دادند. با این حال، برخلاف کارهای قبلی که مزیت یادگیری ماشین را نسبت به تحلیل رمز تفاضلی کلاسیک نشان می‌دهند، این مقاله پیچیدگی را کاهش نمی‌دهد و یا نرخ موفقیت را افزایش نمی‌دهد. حتی حملات بازیابی کلید بدتری را نسبت به تحلیل رمز کلاسیک انجام می‌دهد. این نشان می‌دهد که کار هو و همکاران از قابلیت شبکه‌های عصبی برای تشخیص ویژگی‌های غیرتصادفی تمایزگرهای خطی به طور کامل استفاده نمی‌کند.

به دنبال مقاله [۳۵]، ژو [2] و همکاران [۳۶]، تمایزگرهای عصبی-خطی جدیدی را طراحی کرده و حملات بازیابی کلید را برای الگوریتم رمز DES کاهش دور یافته یافته انجام دادند که نتایج مقاله [۳۵] را بهبود می‌بخشد. با این حال، نتایج کار ژو و همکاران به خط پایه تحلیل خطی کلاسیک هم نمی‌رسد. علاوه بر این، با توجه به این‌که برچسب هر نمونه آموزشی برابر با مقدار تقریب خطی در مقاله [۳۶] است، تمایزگرهای عصبی-خطی ژو و همکاران نمی‌توانند در تشخیص داده‌های تصادفی از داده‌های حقیقی به خوبی عمل کنند. این معایب در روش ژو و همکاران، کاربرد تمایزگرشان را محدود می‌کند.

علاوه بر این، ژو و همکاران در مقاله [۳۶] ادعا می کنند که نتایج بهتری را در تحلیل ۵ دور از الگوریتم رمز DES نسبت به روش کلاسیک به دست می‌آورند. آن‌ها فکر می‌کنند نرخ موفقیت با استفاده از ۴۶۱۷ متن اصلی در روش تحلیل خطی کلاسیک تنها ۹۰ درصد است و میزان موفقیت آن‌ها که با استفاده از ۲۰۴۸ متن اصلی بر اساس تمایزگر عصبی-خطی و الگوریتم بازیابی کلید یک بیتی ۹۱٫۱۵ درصد است، بهتر از تحلیل کلاسیک است. اما در این‌جا نشان می‌دهیم که میزان موفقیت (۹۰ درصد با استفاده از ۴۶۱۷ متن اصلی) روش کلاسیک محاسبه و ارائه‌شده توسط ژو و همکاران طبق الگوریتم ۱ ماتسویی [3] غلط است.

در مقاله [۲]، ماتسویی یک لم برای محاسبه نرخ موفقیت الگوریتم ۱ خود ارائه داده است و طبق آن، نرخ موفقیت الگوریتم ۱ ماتسویی با استفاده از رابطه ۲۰ (p نشان‌دهنده احتمال رابطه تقریب خطی) حدود ۹۲٫۱ درصد است. با توجه به این که احتمال عبارت ۵ دوری مورد استفاده در کار ژو و همکاران $p = ۰٫۵ + ۱٫۲۲ \times ۲^{-۶}$ است، میزان موفقیت الگوریتم ۱ ماتسویی، با توجه به رابطه ۲۰ تنها با استفاده از حدود $۰٫۵ \times |۰٫۵ + ۱٫۲۲ \times ۲^{-۶} - ۰٫۵|^{-۲} \approx$ ۱۳۷۶ متن اصلی بیش از ۹۰ درصد است (بسیار کمتر از ۴۶۱۷ متن اصلی). بنابراین مقایسه در کار ژو و همکاران بر روی نرخ موفقیت اشتباه محاسبه شده الگوریتم ۱ ماتسویی استوار است؛ یعنی نتیجه‌گیری آن‌ها غیرقابل اعتماد است و نتایجی بهتر از تحلیل خطی کلاسیک به دست نیاورده‌اند.

$$\text{تعداد متن اصلی مورد نیاز} = ۰٫۵ \times |p - ۰٫۵|^{-۲} \quad (۲۰)$$

علاوه بر حمله خطی، در مقاله [۳۷] بررسی جامعی از تحلیل RX به کمک یادگیری عمیق روی رمزهای AND-RX مانند Simon و Simeck انجام شده است. در این مقاله با استفاده از مدل ResNet و روش جستجوی خودکار که در بخش‌های قبلی توضیح داده شد، ابتدا مقادیر بهینه برای مقدار چرخش و همچنین تفاضل متن اصلی و کلید برای دو رمز قالبی Simon و Simeck به دست آمده و سپس مجموعه‌داده آموزشی با استفاده از همان مقادیر چرخش و تفاضل تولید شدند. در نهایت تمایزگر ۱۵ دوری برای رمز Simeck32/64 با دقت ۵۱٫۳۴ درصد و همچنین تمایزگر ۱۱ دوری برای Simon32/64 با دقت ۵۴٫۴۵ درصد و با استفاده از رویکرد تحلیل یای انحصاری چرخشی حاصل شدند.

نتایج به‌دست‌آمده برای رمز Simeck تقریباً برابر با تمایزگرهای کلید مرتبط کلاسیک مقالات گذشته است و بیشترین تعداد دور بین تمایزگرهای مبتنی بر یادگیری عمیق را دارد، اما تمایزگرهای گزارش‌شده برای رمز Simon در این مقاله عملکرد بدتری نسبت به تمایزگرهای مبتنی بر یادگیری عمیق قبلی دارند. یک دلیل ممکن می‌تواند ساختارهای مختلف لایه انتشار در توابع دور رمزهای Simon و Simeck باشد. انتخاب‌های مختلف پارامترهای چرخش بیتی در این توابع می‌تواند منجر به مقاومت متفاوت در برابر تحلیل رمز RX شود [۳۷].

در مقاله [۳۸]، از قابلیت‌های یادگیری عمیق برای توسعه یک طرح تمایزگر انتگرالی استفاده شد و نویسندگان مقاله آن را در چندین رمز قالبی از جمله Speck و Simeck اعمال کردند. نتایج این مطالعه نشان می‌دهد که طرح تمایزگر انتگرالی مبتنی بر شبکه عصبی، از تحلیل رمز انتگرالی کلاسیک (ویژگی تقسیم مبتنی بر بیت) بهتر عمل می‌کند و برای اکثر رمزهای قالبی، تعداد دورهای تمایزگر را برای حداقل ۱ دور اضافی افزایش می‌دهد.

با توجه به مقالات نام‌برده شده فوق، می‌توان توانایی و قابلیت یادگیری عمیق در به‌کارگیری آن برای تحلیل رمزهای گوناگون با استفاده از رویکرد حملات مختلف کلاسیک را متوجه شد. نتایج به دست آمده مذکور نشان می‌دهند که می‌توان تحلیل‌های متفاوتی را بر روی الگوریتم‌های رمز به کمک یادگیری عمیق انجام داد که حتی دقت و کارایی بالاتری نسبت به تحلیل کلاسیک داشته باشند.

[1] Hou [2] Zhou [3] Matsui

۸ نتیجه‌گیری و پژوهش‌های آینده

در این پژوهش با هدف بررسی و تشریح فرایند تحلیل تفاضلی رمزهای قالبی به کمک یادگیری عمیق به عنوان یک زمینه تحقیقاتی بدیع در علم رمزنگاری، ابتدا به تفصیل روش تحلیل و به‌کارگیری تمایزگر عصبی و حمله بازیابی کلید به کمک آن شرح داده شد و نیز عوامل مؤثر بر آن‌ها شامل انتخاب تفاضل ورودی بهینه، روش‌های تولید داده و قالب‌های مختلف مجموعه‌داده، معماری و پارامترهای مهم شبکه عصبی واکاوی و مقایسه شدند. همچنین به بررسی رفتار درونی تمایزگرهای عصبی پرداخته شد و دلایلی برای عملکرد تقریبا مشابه تمایزگرهای $ND_{۱}^{cp}$ و $ND_{۱}^{cd}$ ارائه شد.

در نهایت نیز با مقایسه تحلیل‌های صورت‌گرفته روی سه رمز قالبی Speck32/64، Simon32/64 و Simeck32/64 دیدیم که علاوه بر عملکرد بهتر این نوع از تحلیل از نظر سرعت، دقت و پیچیدگی نسبت به تحلیل رمز کلاسیک، می‌توان از یادگیری عمیق برای تحلیل رمزهای قالبی متفاوت و با اندازه ورودی بزرگ‌تر استفاده نمود و همچنین برخی دیگر از حملات کلاسیک را به آن‌ها با موفقیت اعمال کرد.

ترکیب روش‌های تحلیل کلاسیک با سرعت ماشین برای ارزیابی کارآمد و هوشمندانه امنیت اجزای الگوریتم رمزنگاری، یکی از نکات و روندهای حیاتی تحقیقات کنونی است. توسعه هوش مصنوعی فرصت‌های جدیدی را برای تحلیل رمز فراهم می‌کند. استفاده از شبکه‌های عصبی عمیق می‌تواند به خودکارسازی و نیمه خودکارسازی فرایند تحلیل رمز منجر شود و نیز به عنوان یک ابزار جدید در کنار تحلیل رمز کلاسیک قرار گرفته و مورد استفاده محققین این حوزه قرار گیرد. در فضای آکادمیک تحلیل رمز، این امر به بهبود دقت، سرعت و پیچیدگی تحلیل رمز کلاسیک منجر می‌شود. در زمان طراحی الگوریتم رمز جدید نیز مدل‌های یادگیری عمیق می‌توانند به کمک طراح برای تحلیل رمز بیایند و در جهت پیش‌بینی نقاط ضعف احتمالی الگوریتم به کار برده شوند.

همچنین با گسترش استفاده از رمزهای سبک‌وزن در دستگاه‌های با توان محدود، توجه به امنیت آن حائز اهمیت است و یادگیری عمیق می‌تواند برای تحلیل نقاط قوت و ضعف رمزهای قالبی سبک‌وزن و شناسایی آسیب‌پذیری‌هایی که توسط حمله‌کنندگان می‌توانند بهره‌برداری شوند، استفاده شود و در جهت امنیت اطلاعات تأثیرگذار باشد. به‌طورکلی یادگیری عمیق می‌تواند برای تشخیص الگوها، وابستگی‌های آماری الگوریتم رمز و روندهایی که ممکن است نشان‌دهنده وجود تهدید باشند و از دید تحلیل کلاسیک پنهان هستند، به کار گرفته شود.

به عنوان پژوهش‌های آینده نیز می‌توان از یک دیدگاه به بررسی مدل‌های موجود و تفسیر نحوه یادگیری و عملکرد درونی آن‌ها با توجه به ذات جعبه–سیاه شبکه‌های عصبی عمیق پرداخت. از دیدگاه دیگر نیز با توجه به مقایسه‌های انجام شده در این تحقیق روی مولفه‌های دخیل و تأثیرگذار تحلیل رمز با یادگیری عمیق، می‌توان به بهینه‌سازی و همچنین خودکارسازی فرایند محاسبه آن‌ها پرداخت و پس از آن برای ارزیابی عملکرد، آن‌ها را برای تحلیل رمزهای متفاوت به کار گرفت. همچنین به عنوان یک موضوع کمتر دیده‌شده، می‌توان امکان اعمال حملات متفاوت مانند بومرنگ، تفاضلی ناممکن، حملات جبری و نیز حملات خطی و تفاضلی مراتب بالا را با استفاده از یادگیری عمیق روی الگوریتم‌های رمز گوناگون بررسی کرد.

قدردانی

نصور باقری در قالب پژوهانه شماره ۴۹۶۸ توسط دانشگاه تربیت دبیر شهید رجایی حمایت شده است. این پژوهش توسط بنیاد ملی علم ایران (INSF) و در قالب قرارداد شماره ۴۰۲۶۸۰۶ حمایت شده است.

مراجع

[1] Biham, Eli and Shamir, Adi. Differential cryptanalysis of des-like cryptosystems. *Journal of CRYPTOLOGY*, 4:3–72, 1991.

[2] Matsui, Mitsuru. Linear cryptanalysis method for des cipher. in *Workshop on the Theory and Application of of Cryptographic Techniques*, pp. 386–397. Springer, 1993.

[3] Gohr, Aron. Improving attacks on round-reduced speck32/64 using deep learning. in *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*, pp. 150–179. Springer, 2019.

[4] Soos, Mate, Nohl, Karsten, and Castelluccia, Claude. Extending sat solvers to cryptographic problems. in *International Conference on Theory and Applications of Satisfiability Testing*, pp. 244–257. Springer, 2009.

[5] Mouha, Nicky, Wang, Qingju, Gu, Dawu, and Preneel, Bart. Differential and linear cryptanalysis using mixed-integer linear programming. in *Information Security and Cryptology: 7th International Conference, Inscrypt 2011, Beijing, China, November 30–December 3, 2011. Revised Selected Papers 7*, pp. 57–76. Springer, 2012.

[6] Ray, Beaulieu, Douglas, Shors, Jason, Smith, Stefan, Treatman-Clark, Bryan, Weeks, and Louis, Wingers. The simon and speck families of lightweight block ciphers. *Technical report, Cryptology ePrint Archive, Report./404*, 2013.

[7] Yang, Gangqiang, Zhu, Bo, Suder, Valentin, Aagaard, Mark D, and Gong, Guang. The simeck family of

[18] Zhang, Liu, Wang, Zilong, et al. Improving differential-neural cryptanalysis. *Cryptology ePrint Archive*, 2022.

[19] Gohr, Aron, Leander, Gregor, and Neumann, Patrick. An assessment of differential-neural distinguishers. *Cryptology ePrint Archive*, 2022.

[20] Benamira, Adrien, Gerault, David, Peyrin, Thomas, and Tan, Quan Quan. A deeper look at machine learning-based cryptanalysis. in *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I 40*, pp. 805–835. Springer, 2021.

[21] Lu, Jinyu, Liu, Guoqiang, Liu, Yunwen, Sun, Bing, Li, Chao, and Liu, Li. Improved neural distinguishers with (related-key) differentials: Applications in simon and simeck. *IACR Cryptol. ePrint Arch.*, 2022:30, 2022.

[22] Bellini, Emanuele, Gerault, David, Hambitzer, Anna, and Rossi, Matteo. A cipher-agnostic neural training pipeline with automated finding of good input differences. *Cryptology ePrint Archive*, 2022.

[23] Bellini, Emanuele and Rossi, Matteo. Performance comparison between deep learning-based and conventional cryptographic distinguishers. in *Intelligent Computing: Proceedings of the 2021 Computing Conference, Volume 3*, pp. 681–701. Springer, 2021.

[24] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[25] Bao, Zhenzhen, Guo, Jian, Liu, Meicheng, Ma, Li, and Tu, Yi. Enhancing differential-neural cryptanalysis. in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 318–347. Springer, 2022.

[26] Bao, Zhenzhen, Lu, Jinyu, Yao, Yiran, and Zhang, Liu. More insight on deep learning-aided cryptanalysis. in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 436–467. Springer, 2023.

[27] Zhang, Liu, Wang, Zilong, et al. Improving differential-neural cryptanalysis. *Cryptology ePrint Archive*, 2022.

[28] Sajwan, Ayan and Mishra, Girish. Comparative analysis of resnet and densenet for differential cryptanalysis of lightweight block ciphers. in *International workshop on cryptographic hardware and embedded systems*, pp. 307–329. Springer, 2015.

[8] Blondeau, Céline and Gérard, Benoît. Multiple differential cryptanalysis: theory and practice. in *Fast Software Encryption: 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers 18*, pp. 35–54. Springer, 2011.

[9] Picek, Stjepan, Heuser, Annelie, Jovic, Alan, Ludwig, Simone A, Guilley, Sylvain, Jakobovic, Domagoj, and Mentens, Nele. Side-channel analysis and machine learning: A practical perspective. in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 4095–4102. IEEE, 2017.

[10] Benadjila, Ryad, Prouff, Emmanuel, Strullu, Rémi, Cagli, Eleonora, and Dumas, Cécile. Deep learning for side-channel analysis and introduction to ascad database. *Journal of Cryptographic Engineering*, 10(2):163–188, 2020.

[11] Liu, JiaShuo, Ren, JiongJiong, Chen, ShaoZhen, and Li, ManMan. Improved neural distinguishers with multi-round and multi-splicing construction. *Journal of Information Security and Applications*, 74:103461, 2023.

[12] Chen, Yi and Yu, Hongbo. A new neural distinguisher model considering derived features from multiple ciphertext pairs. *IACR Cryptol. ePrint Arch.*, 2021:310, 2021.

[13] Hou, Zezhou, Ren, Jiongjiong, and Chen, Shaozhen. Improve neural distinguisher for cryptanalysis. *Cryptology ePrint Archive*, 2021.

[14] Yue, Xiaoteng and Wu, Wanqing. Improved neural differential distinguisher model for lightweight cipher speck. *Applied Sciences*, 13(12):6994, 2023.

[15] Zhang, Liu, Lu, Jinyu, Wang, Zilong, and Li, Chao. Improved differential-neural cryptanalysis for round-reduced simeck32/64. *Frontiers of Computer Science*, 17(6):176817, 2023.

[16] Baksi, Anubhab and Baksi, Anubhab. Machine learning-assisted differential distinguishers for lightweight ciphers. *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*, pp. 141–162, 2022.

[17] Baksi, Anubhab, Breier, Jakub, Dasu, Vishnu Asutosh, Hou, Xiaolu, Kim, Hyunji, and Seo, Hwajeong. New results on machine learning-based distinguishers. *IEEE Access*, 2023.

speck 32/64 lightweight block cipher. *Cryptology ePrint Archive*, 2023.

[29] Su, Heng-Chuan, Zhu, Xuan-Yong, and Ming, Duan. Polytopic attack on round-reduced simon32/64 using deep learning. in *Information Security and Cryptology: 16th International Conference, Inscrypt 2020, Guangzhou, China, December 11–14, 2020, Revised Selected Papers*, pp. 3–20. Springer, 2021.

[30] Liu, JiaShuo, Ren, JiongJiong, and Chen, ShaoZhen. A deep learning aided differential distinguisher improvement framework with more lightweight and universality. *Cybersecurity*, 6(1):47, 2023.

[31] Lu, Jinyu, Liu, Guoqiang, Sun, Bing, Li, Chao, and Liu, Li. Improved (related-key) differential-based neural distinguishers for simon and simeck block ciphers. *The Computer Journal*, 67(2):537–547, 2024.

[32] Lyu, Lijun, Tu, Yi, and Zhang, Yingjie. Deep learning assisted key recovery attack for round-reduced simeck32/64. in *International Conference on Information Security*, pp. 443–463. Springer, 2022.

[33] Wang, Huijiao, Tian, Jiapeng, Zhang, Xin, Wei, Yongzhuang, Jiang, Hua, et al. Multiple differential distinguisher of simeck32/64 based on deep learning. *Security and Communication Networks*, 2022, 2022.

[34] Chao-Hui, FU, Ming, DUAN, Qiang, WEI, Qian-Qiong, WU, Rui, ZHOU, and Heng-Chuan, SU. Polytopic differential attack based on deep learning and its application. *Journal of Cryptologic Research*, 8(4):591–600, 2021.

[35] Hou, Botao, Li, Yongqiang, Zhao, Haoyue, and Wu, Bin. Linear attack on round-reduced des using deep learning. in *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part II 25*, pp. 131–145. Springer, 2020.

[36] Zhou, Rui, Duan, Ming, Wang, Qi, Wu, Qianqiong, Guo, Sheng, Guo, Lulu, and Gong, Zheng. Neural-linear attack based on distribution data and its application on des. *Cryptology ePrint Archive*, 2023.

[37] Palmieri, Paolo. Deep learning-based rotational-xor distinguishers for and-rx block ciphers: Evaluations on simeck and simon.

[38] Zahednejad, Behnam and Lyu, Lijun. An improved integral distinguisher scheme based on neural networks. *International Journal of Intelligent Systems*, 37(10):7584–7613, 2022.

[39] Standard, Data Encryption et al. Data encryption standard. *Federal Information Processing Standards Publication*, 112:3, 1999.

MonadiAfta-Bianual Scientific Journal for Cyberspace Security
2024, Volume 21, Number 1 (pp. 66–91)
https://monadi.isc.org.ir

A comprehensive exploration of deep learning approaches in differential cryptanalysis of lightweight block ciphers

Iman Mirzaali Mazandarani [1], Nasour Bagheri [1,*] and Sadegh Sadeghi [2]

[1] *Department of TeleCommunications, Shahid Rajaee Teacher Training University, Tehran, Iran*
[2] *Department of Mathematics, Institute for Advanced Studies in Basic Sciences, Zanjan, Iran*

ARTICLE INFO.

Article history:
Received: December 15, 2023
Accepted: February 17, 2024
Published Online: March 26, 2024

Keywords:
Block cipher
Cryptanalysis
Neural distinguisher
Key recovery
Deep learning
Neural network

Type: Review paper

ABSTRACT

With the increasing and widespread application of deep learning and neural networks across various scientific domains and the notable successes achieved, deep neural networks were employed for differential cryptanalysis in 2019. This marked the initiation of growing interest in this research domain. While most existing works primarily focus on enhancing and deploying neural distinguishers, limited studies have delved into the intrinsic principles and learned characteristics of these neural distinguishers. In this study, our focus will be on analyzing block ciphers such as Speck, Simon, and Simeck using deep learning. We will explore and compare the factors and components that contribute to better performance. Additionally, by detailing attacks and comparing results, we aim to address the question of whether neural networks and deep learning can effectively serve as tools for block cipher cryptanalysis or not.

* Corresponding author
Email addresses: iman.mirzaali@sru.ac.ir (Iman Mirzaali Mazandarani), Nbagheri@sru.ac.ir (Nasour Bagheri), s.sadeghi@iasbs.ac.ir (Sadegh Sadeghi)