Ứng dụng của học máy và thuật toán di truyền trong giải mã mật mã một bảng thế

Đỗ Minh Đức, Nguyễn Văn Nam, Phạm Xuân Trường

[a]*Đại học Bách khoa Hà Nội, Trường Công nghệ Thông tin và Truyền thông,*

Abstract

Với sự phát triển bùng nổ hiện nay của công nghệ thông tin và ứng dụng trong đời sống, đặc biệt là các hệ thống mạng truyền tin và các hệ thống thương mại điện tử, các vấn đề về an toàn và bảo mật trở nên có tầm quan trọng thời sự. Đề tài này nghiên cứu ứng dụng của học máy và thuật toán di truyền trong việc giải mã hệ mật mã một bảng thế. Chúng tôi xây dựng một hệ thống sử dụng thuật toán di truyền để tìm ra khóa giải mã tối ưu. Kết quả thử nghiệm cho thấy phương pháp của chúng tôi hiệu quả hơn các phương pháp truyền thống.

1. Giới thiệu

Hệ mật mã một bảng thế (Subtitution Cipher) là một phương pháp mã hóa trong đó các phần tử của bản rõ được thay thế bằng bản mã, theo một cách xác định, với sự trợ giúp của khóa. Một phần tử có thể là các chữ cái đơn lẻ, các cặp chữ cái, bộ ba chữ cái, hỗn hợp của những điều trên,... Người nhận giải mã văn bản bằng cách thực hiện quá trình thay thế nghịch đảo để trích xuất thông điệp gốc.

Tính bảo mật của hệ mật mã một bảng thế: Mặc dù số lượng hoán vị rất lớn ($26! \approx 2^{88.4}$), hệ mật mã này không thực sự mạnh và có thể được giải mã dễ dàng bằng cách phân tích tần suất xuất hiện của các chữ cái.

Trong bài viết này, chúng tôi muốn trình bày 1 hướng tiếp cận đơn giản để giải mã 1 bản rõ được mã hóa bằng mật mã một bảng thế, đó là áp dụng học máy và giải thuật di truyền.

2. Những nghiên cứu liên quan

Trong bài báo [2], vấn đề giải mã các mật mã thay thế được đại diện dưới dạng một bài toán đán nhãn xác suất. Mỗi chữ cái trong mã được gán xác suất để biểu diễn các chữ cái trong văn bản gốc. Các xác suất này được cập nhật song song cho tất cả các chữ cái trong mã, sử dụng các xác suất chung của các cặp chữ cái. Việc lặp lại quá trình cập nhật dẫn đến việc cải thiện ước lượng và cuối cùng giải mã mật mã.

Trong bài báo [3], một phương pháp hoàn toàn tự động để giải mã các mật mã thay thế được giới thiệu, dựa trên phương pháp thư giãn. Thuật toán thư giãn đã được giới thiệu gần đây trong xử lý hình ảnh [4, 6]. Đó là các thuật toán phân loại song song lặp, trong đó mỗi phần tử trong cấu trúc đồ thị cố gắng ước lượng xác suất thành viên lớp dựa trên xác suất của các hàng xóm. Quá trình được lặp lại cho đến khi đạt được một phân loại đáng chấp nhận. Một cách biểu diễn mới của phương pháp thư giãn [4, 5], dựa trên lý thuyết xác suất, mở ra cánh cửa cho các ứng dụng tổng quát hơn của phương pháp thư giãn. Việc sử dụng phương pháp thư giãn cho các lĩnh vực khác ngoài phân loại hình ảnh được thể hiện trong bài báo này.

Chúng tôi sử dụng phân tích tần suất để huấn luyện một mô hình Transformer đa ngôn ngữ cho giải mã. Phương pháp của chúng tôi có khả năng giải mã 700 mật mã từ 14 ngôn ngữ khác nhau với tỷ lệ SER (tỷ lệ lỗi đúng) nhỏ hơn 1. Chúng tôi áp dụng phương pháp của mình vào mật mã Borg và đạt tỷ lệ SER là 5,47 bằng mô hình đa ngôn ngữ và 3,91 bằng mô hình đơn ngôn ngữ Latin. Ngoài ra, các thí nghiệm của chúng tôi cho thấy những mô hình này chống chịu tốt với các loại nhiễu khác nhau, và thậm chí có thể khôi phục từ nhiều loại nhiễu. Theo kiến thức của chúng tôi, đây là ứng dụng đầu tiên của mô hình nơ-ron chuỗi-qua-chuỗi cho việc giải mã.

3. Phát biểu bài toán

Ở đây, chúng ta coi như bài toán là "lý tưởng", nghĩa là kích cỡ của bản mã và bản rõ phải như nhau. Như vậy, việc giải mã sẽ là tìm một nghiệm lý tưởng, đó là một cách ánh xạ một- một từ tập C (bản mã) đến tập P (bản rõ).

Các thành phần của bản mã sẽ được gọi là kí hiệu (symbols) phân biệt với các thành phần của bản rõ, được gọi là kí hiệu (letters).

Khóa trong hệ mật mã một bảng thế sẽ là một bảng hoán vị, ánh xạ một- một từ tập chữ cái đến tập kí hiệu. Tập chữ cái là cố định, vì vậy nó sẽ không được tính tới trong khóa. Như vậy, khóa sẽ gồm tập các kí hiệu, gọi là bảng thay thế (substitution alphabet).

Để giải mã một bản mã, người có bản mã cần có khóa. Câu hỏi đặt ra là nếu không có khóa, người giữ bản mã có đoán được không?

Câu trả lời phụ thuộc vào số lượng khóa có thể có (hay còn gọi là độ lớn không gian khóa). Trong bài viết này, chúng ta sử dụng bảng chữ cái tiếng anh, như vậy tập chữ cái sẽ gồm 26 phần tử. Số khóa có thể là 26!.

Không gian khóa này là đủ lớn để chống lại phương pháp vét cạn không gian khóa. Tuy nhiên, có nhiều phương pháp phá giải tốt hơn việc vét cạn, phương pháp này quan sát mang tính thống kê, chẳng hạn sự xuất hiện không đồng đều của các chữ cái trong ngôn ngữ tự nhiên.

3.1. Bài toán

Bài toán đặt ra ở đây sẽ là: Cho một văn bản bất kì trong tiếng Anh gọi là bản rõ. Bằng hệ mật mã Một bảng thế, bản rõ sẽ được mã hóa thành bản mã. Coi như chúng ta chưa biết rõ khóa, chúng ta sẽ cố gắng giải mã bản mã để thu ngược lại được bản rõ, bằng việc sử dụng các quan sát mang tính thống kê về đặc tính của các chữ cái trong ngôn ngữ tự nhiên.

4. Phương pháp sử dụng

4.1. Mô hình Markov và Quy tắc chuỗi

4.1.1. Quy tắc chuỗi

Trong lý thuyết xác suất, Chain Rule (quy tắc chuỗi) cho phép tính toán bất kỳ phần tử nào trong phân phối chung của một tập hợp các biến ngẫu nhiên chỉ sử dụng xác suất điều kiện.Quy tắc này rất hữu ích trong việc nghiên cứu Bayes Networks - mô tả phân phối xác suất dưới dạng xác suất có điều kiện.

Công thức:

Với 2 sự kiện A và B, quy tắc chuỗi chỉ ra rằng:

$$\mathbb{P}(A \cap B) = \mathbb{P}(B \mid A)\mathbb{P}(A)$$

trong đó

$$\mathbb{P}(A \mid B)$$

là xác suất có điều kiện của sự kiện A khi sự kiện B xảy ra.

Với tập sự kiện $A_1, \ldots, A_n$ không giao nhau và khác không, quy tắc chuỗi có dạng:

$$\begin{aligned}
&\mathbb{P}\left(A_1 \cap A_2 \cap \ldots \cap A_n\right) \\
&\quad = \mathbb{P}\left(A_n \mid A_1 \cap \ldots \cap A_{n-1}\right) \mathbb{P}\left(A_1 \cap \ldots \cap A_{n-1}\right) \\
&\quad = \prod_{k=1}^{n} \mathbb{P}(A_k \mid A_1 \cap \cdots \cap A_{k-1}) \\
&\quad = \prod_{k=1}^{n} \mathbb{P}\left(A_k \,\middle|\, \bigcap_{j=1}^{k-1} A_j\right).
\end{aligned}$$

Sử dụng công thức trên, ta xây dựng 1 hàm tính toán xác suất xuất hiện của 1 từ. Giả sử p(word) là xác suất của từ "word". Ta sẽ phân rã từ này ra theo công thức trên như sau:

$$\begin{aligned}
&\mathbb{P}(\text{"word"}) \\
&= \mathbb{P}(\text{"d"} \mid \text{"wor"}) \cdot \mathbb{P}(\text{"r"} \mid \text{"wo"}) \cdot \mathbb{P}(\text{"o"} \mid \text{"w"}) \cdot \mathbb{P}(\text{"w"}) \\
&= \mathbb{P}(\text{"d"} \mid \text{"r"}) \cdot \mathbb{P}(\text{"r"} \mid \text{"o"}) \cdot \mathbb{P}(\text{"o"} \mid \text{"w"}) \cdot \mathbb{P}(\text{"w"})
\end{aligned}$$

(Do trạng thái n chỉ phụ thuộc vào trạng thái n-1)

4.1.2. Mô hình Markov

Trong lý thuyết xác suất, mô hình Markov là một mô hình ngẫu nhiên mô tả 1 tập các sự kiện có thể 1 cách tuần tự khi mà xác xuất xảy ra của trạng thái tiếp theo chỉ phụ thuộc vào trạng thái trước đó. (nghĩa là nó thoả mãn Markov Property).

Giả thiết Markov (Markov Assumption): giả thiết Markov cho rằng xác suất của một trạng thái tại thời điểm hiện tại chỉ phụ thuộc vào trạng thái trước đó. Markov Assumption cung cấp nền tảng cho việc mô hình hoá việc ra quyết định trong các tình huống mà kết quả là một phần ngẫu nhiên và 1 phần dưới sự điều khiển của người ra quyết định. Ở đây, chúng ta sử dụng Markov Assumption để tính toán xác suất xuất hiện của 1 từ.

4.2. Thuật toán di truyền

4.2.1. Giới thiệu chung

Thuật toán mô phỏng quá trình tiến hoá tự thích nghi của các quần thể sinh học dựa trên học thuyết Darwin để tìm lời giải các bài toán tối ưu. Quá trình phát triển của mỗi quần thể tuân theo quy luật chọn lọc tự nhiên, tiến hoá qua các thế hệ kế tiếp nhau.

Các hậu duệ được sinh ra từ thế hệ trước thông qua quá trình sinh sản (di truyền và biến dị) cạnh tranh tự nhiên, cá thể nào thích nghi cao hơn với môi trường thì sẽ có khả năng lớn hơn trong tồn tại và sản sinh con cháu (cá thể nào "khỏe mạnh" sẽ sống sót và phát triển, các cá thể "yếu" sẽ bị loại bỏ).

4.2.2. Các bước tiến hành

Bước 1. Chọn cấu trúc nhiễm sắc thể biểu diễn lời giải của bài toán.

Bước 2. Khởi tạo tập lời giải ban đầu. Việc khởi tạo là ngẫu nhiên hay có thể áp dụng một vài thuật toán heuristic.

Bước 3. Chọn hàm đánh giá mức độ tốt của lời giải (để đánh giá các cá thể trong quần thể lời giải theo độ thích nghi của chúng).

Bước 4. Thiết kế các toán tử di truyền. Các toán tử này được thiết kế nên dựa trên nhiễm sắc thể, đặc điểm bài toán và các ràng buộc của nó.

Bước 5: Xác định các tham số cho bài toán. Các tham số này có thể không thay đổi trong quá trình tiến hoá hoặc có thể tự điều chỉnh tham số theo thời gian.

4.2.3. Lược đồ thuật toán

Algorithm 1 Giải thuật di truyền

```
1: procedure DiTruyen
2:     Khởi tạo quần thể ban đầu
3:     while không đạt điều kiện dừng do
4:         Đánh giá và lựa chọn cá thể tốt nhất
5:         Lai ghép cá thể để tạo ra thế hệ mới
6:         Đột biến các cá thể trong thế hệ mới
7:         Thay thế thế hệ cũ bằng thế hệ mới
8:     end while
9: end procedure
```

5. Xây dựng thuật toán

5.1. Sử dụng N- Gram để xây dựng mô hình Markov

N-Gram là một chuỗi liên kết của N items từ một mẫu văn bản hoặc bài phát biểu. N-Gram có thể bao gồm các khối từ lớn hoặc các tập hợp âm tiết nhỏ. N-Gram được sử dụng làm cơ sở cho các mô hình N-Gram hoạt động, là công cụ trong quá trình xử lý ngôn ngữ tự nhiên như một cách dự đoán văn bản hoặc bài phát biểu sắp tới.

Ở đây chúng ta sẽ dùng Bigram và Unigram để xây dựng mô hình Markov , ta chỉ cần tính xác suất của

$$\mathbb{P}(\text{"d"} \mid \text{"r"})$$

và

$$\mathbb{P}(\text{"w"})$$

Algorithm 2 Hàm khởi tạo ác ma trận N-Gram

for *i* 0 25 **do**

  for *j* 0 25 **do** bigram_Matrix[*i*][*j*] = 1

  for *i* 0 25 **do** unigram_Matrix[*i*] = 1

Ở đây ta định nghĩa thêm 1 số hàm để xử lý, cập nhật các ma trận N-gram :

Hàm update_Bigram_Matrix và update_Unigram_Matrix thao tác trên từng cặp chữ cái / 1 chữ cái.

Algorithm 3 Hàm cập nhật các ma trận

```
1: function UPDATE_BIGRAM_MATRIX(a, b)
2:     bigram_Matrix[get_Index(a)][get_Index(b)] += 1
3: end function
4: function UPDATE_UNIGRAM_MATRIX(ch)
5:     unigram_Matrix[get_Index(ch)] += 1
6: end function=0
```

Hàm update_Prob_Matrixs cập nhật xác suất của 1 từ bằng cách tách nó ra thành Bigram và Unigram.

Algorithm 4 Hàm cập nhật Xác suất

```
1: function UPDATE_PROB_MATRIXS(words)
2:     words ← split(words)
3:     for each word in words do
4:         UPDATE_UNIGRAM_MATRIX(word[0])
5:         for i from 0 to length(word) − 2 do
6:             UPDATE_BIGRAM_MATRIX(word[i], word[i + 1])   ▷ update 2 adjoining characters
7:         end for
8:     end for
9: end function
```

5.2. Xây dựng Class Decoder bằng thuật toán di truyền

dna_Set: cấu trúc NST biểu diễn lời giải bài toán. Ở đây số NST được khởi tạo = 20.

init (): khởi tạo tập lời giải ban đầu.

population_Score(): hàm đánh giá độ tốt của lời giải.

Hàm này sử dụng hàm get_Sentence_Prob() (sẽ được xây dựng ở phần Machine Learning) để đánh giá xác suất xuất hiện của 1 sentence.

remove_Weak_Entity(): loại bỏ cá thể "yếu" dựa trên hàm đánh giá xây dựng ở trên.

next_Gen(): thiết kế toán tử di truyền (toán tử đột biến).

Mỗi cá thể được lai tạo với chính nó và các con sinh ra được đánh giá để giữ lại cá thể tốt nhất.

decipher(): hàm giải mã.

Hàm này nhận tham số là encode_Text (bản mã).

Hàm sẽ tính toán độ tốt của bản mã, sau đó loại bỏ các cá thể yếu (chỉ giữ lại 5 cá thể tốt nhất).

Tham số thứ 2 được truyền vào là num_Iters, nghĩa là số kí tự được dùng để train, từ đó dùng để decode.

Giá trị trả về của hàm này là cá thể tốt nhất, nghĩa là bản rõ có giá trị hàm đánh giá cao nhất.

5.3. Xây dựng thuật toán học máy

Ở đây chúng ta xây dựng 3 hàm, theo thứ tự là:

get_Word_Prob(word): lấy xác suất của từ truyền vào sẽ xuất hiện trong bản rõ.

Algorithm 5 Hàm tính xác suất từ

```
1: function GET_WORD_PROB(word)
2:     logProb ← 0
3:     logProb ← logProb + log(unigram_Matrix[get_Index(word[0])])
4:     for i from 0 to length(word) − 2 do
5:         logProb ← logProb + log(bigram_Matrix[get_Index(word[i]), get_Index(word[i+1])])
6:     end for
7:     return logProb
8: end function
```

get_Sentence_Prob(words): lấy xác suất của câu văn truyền vào sẽ xuất hiện trong bản rõ. Ở đây ta hiểu câu văn là 1 tập các từ liên tiếp. Do đó ta sẽ tính xác suất của câu văn bằng cách tính tổng xác suất của các từ trong câu văn đó. (sử dụng hàm get_Word_Prob).

Algorithm 6 Hàm tính xác suất câu

```
1: function GET_SENTENCE_PROB(words)
2:     if type(words) == str then
3:         words ← words.split()
4:     end if
5:     logProb ← 0
6:     for each word in words do
7:         logProb ← logProb + get_Word_Prob(word)
8:     end for
9:     return logProb
10: end function
```

create_Prob_Matrix(path): khởi tạo ma trận xác suất để huấn luyện. Hàm này nhận tham số là đường dẫn đến dữ liệu huấn luyện.

Algorithm 7 Hàm khởi tạo ma trận

```
function HÀM KHỞI TẠO MA TRẬN(path)
    special_Regex ← compile('[a − zA − Z]')
    global unigram_Matrix, bigram_Matrix

    with open(path,' r', encoding =' utf − 8') as train :
    for each line in train: do
        line ← line.strip()
        if line then
            line ← special_Regex.sub("", line)
            words ← line.lower()
            update_Prob_Matrixs(words)
        end if
    end for

    unigram_Matrix /= unigram_Matrix.sum()
    bigram_Matrix /= bigram_Matrix.sum(axis = 1, keepdims=True)
end function
```

6. Kết quả thu được

Ở đây tôi sẽ lấy 1 bản rõ bất kì, sau đó áp dụng hàm encrypt() để mã hoá nó. Giả định rằng chúng ta không biết map_Table, thuật toán sẽ tính toán xác suất của từng chữ, rồi đến từng từ và cuối cùng là đến 1 câu văn hoàn chỉnh.

Bản rõ:

"'The Foundation is committed to complying with the laws regulating charities and charitable donations in all 50 states of the United States. Compliance requirements are not uniform and it takes a considerable effort, much paperwork and many fees to meet and keep up with these requirements. We do not solicit donations in locations where we have not received written confirmation of compliance. To SEND DONATIONS or determine the status of compliance for any particular state visit"'

Sau khi chạy thuật toán, chúng ta có bản mã như sau:

""ctgzyer droqgdrpryjf zdr ymj qygsmdp zyk gj jzbrf z emyfgkrdzitr rssmdj pqeh czcrdvmdb zyk pzyu srrf jm prrj zyk brrc qc vgjh jhrfr droqgdrpryjf vr km ymj fmtgegj kmyzjgmyf gy tmezjgmyf vhrdr vr hznr ymj drergnrk vdgjjry emysgdpzjgmy ms empctgzyer jm fryk kmyzjgmyf md krjrdpgyr jhr fjzjqf ms empctgzyer smd zyu czdjgeqtzd fjzjr ngfgj"'

Kết quả của hàm đánh giá:

```
Progress: 0/1000, Best score: -1698.1312687205655
Progress: 100/1000, Best score: -1168.0119617680657
Progress: 200/1000, Best score: -1085.1026083099484
Progress: 300/1000, Best score: -993.70456440888795
Progress: 400/1000, Best score: -987.7619538426014
Progress: 500/1000, Best score: -987.7619538426014
Progress: 600/1000, Best score: -987.7619538426014
Progress: 700/1000, Best score: -987.7619538426014
Progress: 800/1000, Best score: -987.7619538426014
Progress: 900/1000, Best score: -987.7619538426014
```

Hình 1: Kết quả khi chạy hàm đánh giá

Qua hình chúng ta có thể thấy, progress càng tăng thì điểm số của hàm đánh giá càng được cải thiện (cho đến 1 mức nào đó thì nó sẽ hội tụ).

Kết quả giải mã được lưu trong file .txt được generate trong quá trình chạy, có thể thấy thuật toán đã giải mã ra kết quả giống như bản rõ ban đầu.

Thuật toán trên đã được thử nghiệm và cải tiến nhiều lần, nên hầu hết đều đưa ra kết quả đúng với thời gian ngắn.

7. Kết luận

Trong bài báo cáo này, chúng tôi đã xem xét và nghiên cứu việc giải hệ mật mã một bảng thế. Chúng tôi đã đề xuất sử dụng học máy và giải thuật di truyền nhằm giải quyết bài toán trên. Kết quả nghiên cứu cho thấy phương pháp đưa ra được kết quả đúng với thời gian ngắn.

Tuy nhiên, còn một số hạn chế và thách thức cần được xem xét. Ví dụ, thách thức có thể gặp phải trong việc áp dụng phương pháp với bảng chữ cái ngoài Latinh. Đồng thời, việc mở rộng có thể được thực hiện để nâng cao hiệu quả và khả năng áp dụng của thuật toán trong tương lai.

Tổng kết lại, bài báo cáo này đã đưa ra một một cách tiếp cận trong việc giải hệ mật mã một bảng thế, đồng thời đánh giá hiệu quả và công trình đóng góp của nó. Kết quả nghiên cứu này góp phần nâng cao hiểu biết và thúc đẩy sự phát triển trong lĩnh vực này. Hy vọng rằng bài báo cáo của chúng tôi sẽ truyền đạt được những thông tin quan trọng và mang lại giá trị cho cộng đồng nghiên cứu và các nhà quản lý trong an toàn thông tin.

Tài liệu

[1] Servos, William, Using a genetic algorithm to break Alberti Cipher. Journal of Computing Sciences in Colleges, 2004.

[2] Shmuel Peleg and Azriel Rosenfeld. 1979. Breaking substitution ciphers using a relaxation algorithm. Commun. ACM 22, 11 (Nov. 1979), 598–605. https://doi.org/10.1145/359168.359174

[3] Patel, Falguni Farik, Mohammed. (2016). A New Substitution Cipher -Random-X. International Journal of Scientific Technology Research. 5. 125-128.

[4] Aldarrab, Nada May, Jonathan. (2020). Can Sequence-to-Sequence Models Crack Substitution Ciphers?.