TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

———————— * ————————

Báo cáo thu hoạch: Project 1

Xây dựng mô hình cho bài toán phát hiện Crawler

Sinh viên thực hiện:
Cao Hải Nam - 20173275

Giáo viên hướng dẫn:
PGS.TS.
Trần Việt Trung

Hà Nội, Ngày 2 tháng 1 năm 2020

Mục lục

Danh sách hình vẽ

1 Giới thiệu về dữ liệu chuỗi thời gian

1.1 Khái niệm chuỗi thời gian

Chuỗi thời gian (time series) trong thống kê, xử lý tín hiệu, kinh tế lượng và toán tài chính là một chuỗi các điểm dữ liệu, được đo theo từng khoảng khắc thời gian liền nhau theo một tần suất thời gian thống nhất. Trong thực tế, có thể thấy nhiều ví dụ về chuỗi thời gian như: doanh số của công ty trong 20 năm gần đây, hoặc nhiệt độ ghi nhận tại một trạm quan trắc khí tượng, hoặc công suất điện năng tiêu thụ trong một nhà máy, đó là các ví dụ điển hình cho một chuỗi thời gian.

year	CPI	Lai_suat	GTSX_CN	PA
2007M1	111.0	6.5	49,212.0	12,000.0
2007M2	113.4	6.5	35,392.0	12,000.0
2007M3	113.1	6.5	45,154.0	12,000.0
2007M4	113.7	6.5	47,344.6	12,000.0
2007M5	114.5	6.5	47,953.4	12,000.0

Hình 1: Các chỉ số CPI, Lãi suất, ... là dữ liệu chuỗi thời gian

Về cơ bản có thể chia dữ liệu chuỗi thời gian thành hai dạng: rời rạc hoặc liên tục.

- Các dữ liệu rời rạc, chỉ các chuỗi dữ liệu có thời gian thu thập dữ liệu không liền mạch, chẳng hạn như dữ liệu đóng cửa sàn giao dịch chứng khoán. Các dữ liệu liện tục được thu thập theo khoảng thời gian liên tục, bằng nhau, chẳng hạn dữ liệu sử dụng băng thông của nhà cung cấp dịch vụ Internet.

- Trong trường hợp dữ liệu liên tục, t là thời gian thực và x(t) là các dữ liệu liên tục, ví dụ các dữ liệu từ các tín hiệu tương tự, để lựa chọn chuỗi x(t), ta phải lấy dữ liệu tại các điểm rời rạc hay nói cách khác là lấy mẫu, khi đó ta thu được chuỗi thời gian rời rạc x[t]. Để đảm bảo x[t] vẫn thể hiện được những được trưng tiêu biểu của x(t) thì việc lấy mẫu cần phải tuân theo lý thuyết lấy mẫu của Nyquist [1]

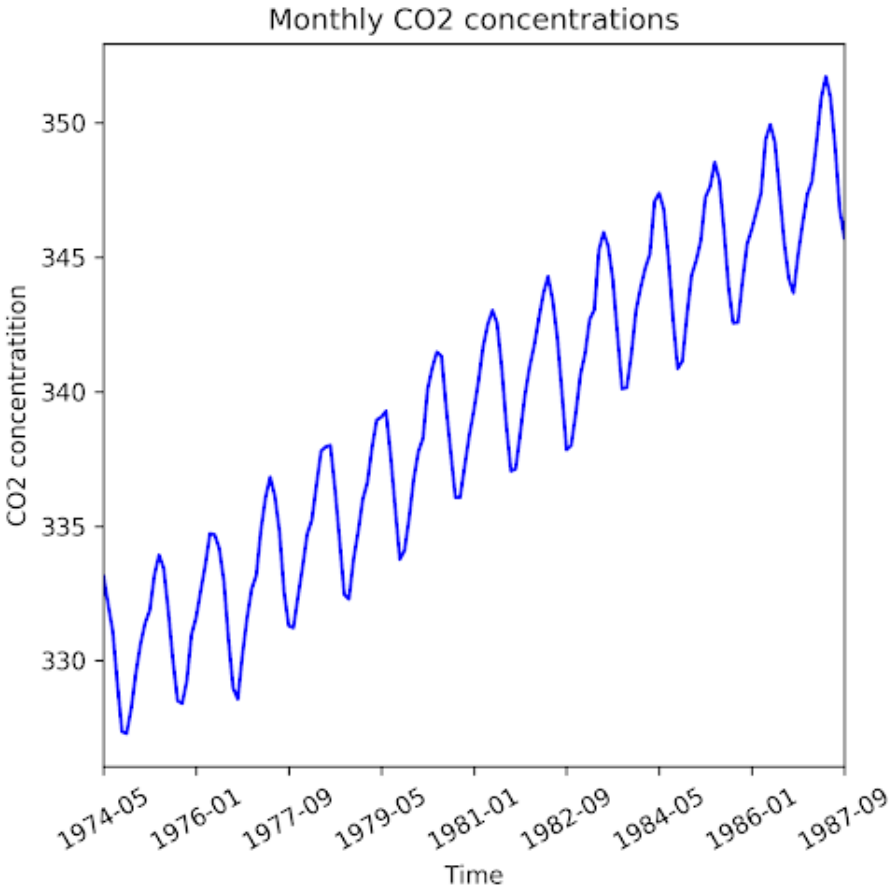
Có thể nói phần lớn dữ liệu phụ thuộc thời gian phản ánh các hoạt động của đời sống kinh tế - xã hội thường được đo tại các mốc thời gian cách đều nhau nên trong luận văn này chỉ quan tâm đến chuỗi thời gian rời rạc, ở đó các quan sát được đo trong các khoảng thời gian như nhau với phương pháp đo cố định.

Về mặt toán học, chuỗi thời gian được coi là một chuỗi các biến ngẫu nhiên $x_1, x_2, \ldots, x_i, \ldots$ với x_i tương ứng là biến ngẫu nhiên thể hiện giá trị quan sát tại thời điểm thứ i, và được kí hiệu là x_t với t là thời điểm quan sát. Dữ liệu của bất kỳ chuỗi thời gian nào đều có thể được coi là được tạo ra nhờ một quá trình ngẫu nhiên và một tập hợp dữ liệu cụ thể, tức là một mẫu, của quá trình ngẫu nhiên đó. Sự khác biệt giữa quá trình ngẫu nhiên và kết quả của nó giống như sự khác biệt giữa tổng thể và mẫu trong dữ liệu đối chiếu. Cũng như chúng ta sử dụng các dữ liệu

mẫu để suy ra các ước lượng về một tập hợp, thì trong lĩnh vực chuỗi thời gian, chúng ta dùng kết quả để suy ra các ước lượng về quá trình ngẫu nhiên đó.

Khi nghiên cứu về quá trình ngẫu nhiên, chúng ta cần xem xét các đặc trưng của nó. Đối với chuỗi thời gian, có 4 thành phần đặc trưng đó là:

1. *Xu hướng (T)*: phản ánh sự tiến triển dài hạn của hiện tượng quan sát. Xu hướng xuất hiện khi có một chiều hướng tăng hoặc giảm một cách lâu dài trong dữ liệu. Thành phần này không nhất thiết phải tuyến tính. Ví dụ như xu hướng tăng của nồng độ CO_2 theo dõi từ 1974 tới 1987 [3]:



Hình 2: Xu hướng tăng của nồng độ CO_2 theo dõi từ 1974 tới 1987

2. *Mùa (S)*: Biểu diễn sự tăng hoặc giảm mức độ của hiện tượng ở một số thời điểm (tháng, quý) nào đó và được lặp đi lặp lại. Ví dụ, nhu cầu về dép lê sẽ cao nhất trong những tháng mùa hè.

3. *Chu kỳ (C)*: Thể hiện biến động của hiện tượng được lặp lại với chu kì nhất định, thường kéo dài từ 2 đến 10 năm

4. *Dao động ngẫu nhiên (I)*: Xét đến sự dao động ngẫu nhiên xung quanh xu thế, điều này có thể làm ảnh hưởng đến chu kỳ và tính mùa của chuỗi quan sát.

Những thành phần này kết hợp với nhau trong chuỗi thời gian bằng nhiều cách thức khác nhau, chẳng hạn chuỗi thời gian X_t được mô tả là:

- $X_t = T * P * S * I$: gọi là mô hình tích
- $X_t = T + P + S + I$: gọi là mô hình tổng
- $X_t = T * P * S + I$: gọi là mô hình hỗn hợp

Phân tách chuỗi thời gian là một nhiệm vụ chính trong phân tích chuỗi thời gian. Nó là một hướng tiếp cận có ý nghĩa để mô hình một chuỗi thời gian thông qua các thành phần của nó. Việc phân tách chuỗi thời gian nhằm xác định các tác động chủ yếu dẫn tới xu hướng trong dữ liệu và dự đoán các giá trị trong tương lai, trong đó mục tiêu dự đoán được tập trung nhiều hơn. Kết quả của việc phân tách chuỗi thời gian đó là ước lượng được các thành phần đặc trưng trong chuỗi thời gian và cách thức kết hợp chúng với nhau trong chuỗi.

1.2 Các phương pháp phân tích chuỗi thời gian

Việc phân tích chuỗi thời gian dựa trên giả thuyết dữ liệu cần tiên đoán phụ thuộc vào các dữ liệu quá khứ, từ đây hình thành ý tưởng đó là xây dựng một hàm f để dự đoán giá trị xt dựa trên các giá trị quan sát trước đó:

$$x_t = f(x_{t-1}, x_{t-2}, ..., x_{t-p})$$

Việc xây dựng f có thể thông qua mô hình tuyến tính như AR, MA, ...hoặc mô hình phi tuyến như mô hình Markov ẩn, mô hình mạng neuron, ...

Mô hình tuyến tính

Khi sử dụng phương pháp thống kê, chúng ta cần đảm bảo dữ liệu chuỗi thời gian đạt được tính dừng. Như đã biết, về mặt toán học dữ liệu chuỗi thời gian được sinh ra từ một quá trình ngẫu nhiên, khi đó chuỗi thời gian đang xem xét được coi là một quá trình ngẫu nhiên dừng.

Trong toán học, một quá trình được gọi là dừng ngặt (strictly stationary) nếu phân phối xác suất của nó không độc lập theo thời gian. Nói cách khác, nếu chúng ta có 1 chuỗi thời gian được quan sát giữa 2 thời điểm t và T thì các kết hợp phân phối của những quan sát này phải giống nhau như các quan sát giữa 2 thời điểm t+k và T+k.

Tuy nhiên, đây chỉ là định nghĩa lý thuyết khó có thể áp dụng để kiểm tra tính dừng trong thực nghiệm. Trong thực nghiệm thời tính dừng của chuỗi được xét ở quan điểm nhẹ hơn, có thể dễ dàng thực hiện và kiểm tra, đó là khái niệm dừng hiệp phương sai (covariance stationary). Một chuỗi xt được gọi là covariance stationary phải thỏa mãn các điều kiện sau:

- Có một giá trị trung bình xác định, $E(x_t) = \mu < \infty$, nghĩa là giá trị kì vọng phải độc lập theo thời gian, không có xu hướng.
- Có một giá trị phương sai xác định, $Var(x_t) = \sigma_x^2 < \infty$
- Các giá trị hiệp phương sai chỉ phụ thuộc vào các quan sát khác nhau, k và độc lập theo thời gian: $Cov(x_t, x_{t-s}) = Cov(x_{t+k}, x_{t-s+k})$

Để kiểm tra tính dừng của một chuỗi, có nhiều phương pháp được sử dụng đó là: dựa vào biểu đồ tương quan [2], kiểm định nghiệm đơn vị [2], ...Khi chuỗi là không dừng, ta sẽ phải biến đổi chuỗi về dạng khác để có thể thu được một chuỗi dừng nhờ áp dụng các kĩ thuật của phương trình sai phân.

Quá trình phân tích, dự báo chuỗi thời gian x_t là để tìm ra các mô hình, quy luật ẩn trong nó, được thực hiện trên các mẫu quan sát, gồm có những bước sau:

- *Bước 1: Nhận dạng các thành phần ẩn tồn tại trong chuỗi thời gian*
  Phân tích, nghiên cứu hành vi cũng như dự báo biến động của chuỗi thời gian thì cần thiết phải ước lượng các thành phần đặc trưng trong chuỗi thời gian và cách thức kết hợp chúng với nhau trong chuỗi.

- *Bước 2: Làm trơn số liệu*
  Sau khi xác định các thành phần trên trong chuỗi thời gian, tiếp theo phải tiến hành làm trơn dữ liệu. Cụ thể hơn là loại trừ được thành phần xu thế và mùa vụ trong chuỗi thời gian. Chuỗi thu được sau cùng không còn chứa các thành phần đó (chuỗi được làm trơn), khiến cho việc phân tích, dự báo dễ dàng hơn.

- *Bước 3: Chọn lựa, ước lượng, đánh giá mô hình*
  Chọn lựa mô hình trong lớp mô hình phân tích, dự báo chuỗi thời gian, sao cho mô hình được lựa chọn là "tốt nhất", phải thoả mãn các tiêu chí kiểm định, đánh giá. Mô hình được lựa chọn cũng phải đơn giản và có thể hiểu được dễ dàng, sinh ra chuỗi "gần giống" với chuỗi quan sát thực.

- *Bước 4: Dự báo*
  Từ mô hình thực hiện dự báo giá trị tương lai cho chuỗi, phân tích sự phù hợp của giá trị dự báo cả về mặt thực nghiệm và lý thuyết. Xác định độ chênh giữa giá trị dự báo với giá trị quan sát thực và khoảng tin cậy của dự báo (giới hạn mà giá trị quan sát thực sẽ nằm trong đó).

- *Bước 5: Thử nghiệm kết quả*
  Ứng dụng mô hình để thực hiện các dự báo về các giá trị tương lai của hiện tượng nghiên cứu, trên cơ sở đó để lập kế hoạch, đề ra các quyết định trong sản xuất, kinh doanh hoặc đề ra chính sách. Đồng thời cung cấp thêm các giá trị quan sát mới vào dữ liệu chuỗi quan sát nhằm mục đích hiệu chỉnh lại mô hình để đưa ra dự báo tốt hơn.

Các mô hình thống kê thường được sử dụng đó là:

- Mô hình tự hồi quy (AR): Trong mô hình tự hồi quy, chuỗi thời gian x_t được mô tả bởi phương trình sau:

  $$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + \epsilon_t$$

  Phương trình này được gọi là phương trình biểu diễn của mô hình tự hồi quy bậc p (AR(p)). Trong đó:

  - $\phi_{i:1\to p}$ là các tham số của mô hình
  - ϵ_t là sai số ngẫu nhiên không tương quan hay nhiễu trắng

- Mô hình trung bình trượt (MA): Chuỗi thời gian x_t được gọi là quá trình trung bình trượt bậc q (MA(q)) nếu như mỗi quan sát x_t của quá trình MA(q) được viết dưới dạng sau:

$$x_t = \epsilon_t + \theta_t\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-1}$$

  Với ϵ_t là một quá trình nhiễu trắng với $E(\epsilon_t) = 0, Vary(\epsilon_t) = \sigma^2 = const$, $\theta_{i:1\to q}$ là các tham số của mô hình. Mô hình MA hoạt động mà không cần thông tin phản hồi.

- Mô hình tự hồi quy và trung bình trượt (ARMA): Các chuỗi thời gian đôi khi không thể mô hình hóa được bằng MA hay AR do chúng có đặc tính của cả hai quá trình này. Khi đó, để biểu diễn, người ta sử dụng mô hình ARMA, là pha trộn của cả hai mô hình MA và AR. Khi đó, quá trình ARMA(p, q) được mô tả như sau:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + \epsilon_t + \theta_t\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-1}$$

  Lúc này, việc dự báo có thể thực hiện được nhờ xác định p và q. Việc xác định này được thực hiện bởi người thực hiện dự báo thông qua kinh nghiệm. Trong đó, p được xác định dựa trên việc vẽ các hàm tự tương quan riêng phần (partial autocorrelation functions), đồng thời q được xác định thông qua các hàm tự tương quan (autocorrelation functions). Điều quan trọng là các mô hình này có thể giải thích được kết quả dự báo thông qua các công cụ trình diễn trên máy tính.

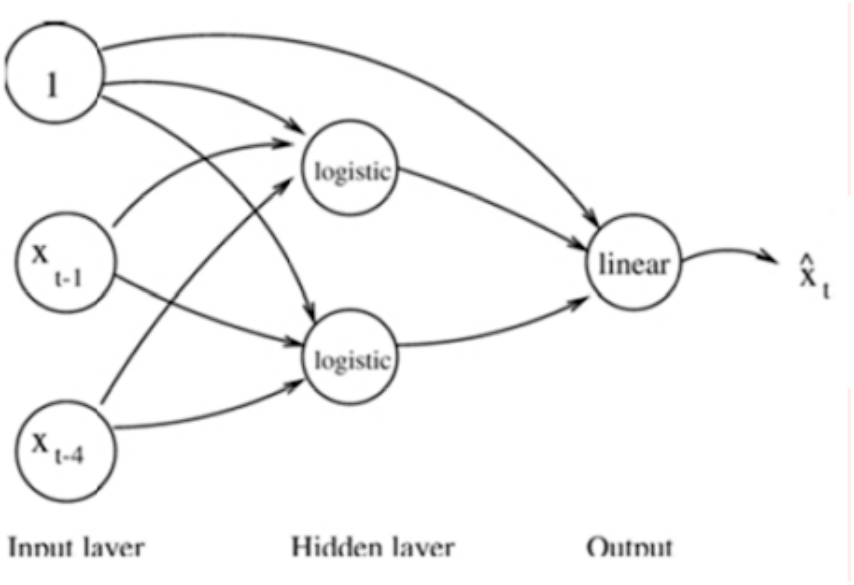
Mô hình phi tuyến

Để mô tả các quá trình phi tuyến tính, các mô hình này giả thiết dữ liệu chuỗi thời gian là phi tuyến tính. Điều này phù hợp với thực tế rằng các chuỗi thời gian không thể biết trước chúng có đặc tính là tuyến tính hay phi tuyến tính. Tuy nhiên, đặc điểm của mô hình này là sử dụng rất nhiều tham số xây dựng mô hình và do đó, rất khó giải thích quá trình xác định các tham số của mô hình. Vì đặc tính này, các mô hình phi tuyến tính được coi như quá trình hộp đen. Một số mô hình tiêu biểu sử dụng để dự báo dữ liệu chuỗi thời gian (theo [4]):

- Mô hình Markov ẩn (Hidden Ma arkov Model) Mô hình Markov ẩn (HMM) cũng được sử dụng để dự báo dữ liệu chuỗi thời gian [5]]. Tuy vậy, mô hình này không thích hợp để giải quyết các vấn đề liên quan đến dữ liệu liên tục. Do vậy, các mô hình HMM đã được hiệu chỉnh để sử dụng trong giải quyết bài toán dự báo chuỗi thời gian. Theo đó, mô hình toán học của nó trở ở nên quá phức tạp để áp dụng thuật toán forward-backward xác định các tham số, độ phức tạp của giải thuật này là $O(N^2)$, nên rất khó mở rộng cho các tập dữ liệu kích thước lớn.

- Mô hình mạng neuron Việc sử dụng mạng nơron nhân tạo để dự báo chuỗi thời gian đã được nghiên cứu nhiều, do đặc điểm rất phù hợp với các dữ liệu phi tuyến tính. Cách xây dựng một mô hình mạng neuron để thực hiện dự báo chuỗi thời gian như sau: Các quan sát x_{t-s} được sử dụng làm đầu vào để dự báo giá trị x_t. Người ta sẽ xây dựng tập huấn luyện bằng phương pháp sau:

  1. Chuẩn hóa dữ liệu
  2. Xác định khoảng dự báo
  3. Chia tập dữ liệu ban đầu thành các tập: huấn luyện (training) ($>$ 50% số mẫu), kiểm tra (test) (10% -$>$ 30% số mẫu) và tập kiểm định (validation)



Hình 3: Cấu trúc cơ bản mạng neuron

4. Xây dựng tập dữ liệu với mẫu đầu tiên có đầu ra là x_s, các đầu vào là $x_{s-1}, x_{s-2}, \ldots, x_1$
5. Xây dựng mô hình mạng neuron áp dụng cho dự báo. Việc xác định cấu trúc mạng tối ưu cần quá trình thử - sửa sai
6. Huấn luyện mạng với các thông số khởi tạo trên các tập dữ liệu training, xác định lỗi với tập dữ liệu test để xác định khả năng tổng quát hóa
7. Sau khi huấn luyện, thực hiện kiểm định độ chính xác của mô hình với tập validation.

Có nhiều kiến trúc mạng neuron được sử dụng như: MLP (Multi-laye perceptron), mạng hồi quy - RNN (Recurrent neural networks), Mạng bộ nhớ dài-ngắn - LSTM (Long Short Term Memory networks), ... Trong bài toán mà báo cáo sẽ trình bày ở phần dưới, nhóm sử dụng một kiến trúc mạng được xây dựng dựa trên kiến trúc mạng LSTM để giải quyết bài toán đó là mạng GAN. Sau đây, báo cáo sẽ trình bày cơ bản về mạng GAN và LSTM. Hai kiến trúc mạng này sẽ được trình bày ở phần 3.

2 Bài toán phát hiện Crawler

2.1 Giới thiệu crawler

Trong thập kỷ qua, việc thu thập dữ liệu ngày càng trở nên quan trọng đối với nhiều lĩnh vực trong công việc và cuộc sống hàng ngày của chúng ta. Web crawler là một công cụ đặc biệt cho phép thu thập thông tin trên Internet một cách dễ dàng và tự động. Nó được dùng nhiều trong các ứng dụng về công cụ tìm kiếm, kiểm thử phần mềm và hệ thống phân tích dữ liệu. Web crawler được phân loại theo mục đích sử dụng bao gồm crawler dùng cho mục đích hợp pháp (như phân tích nội dung, đánh chỉ mục trong hệ thống tìm kiếm, ...) và crawler được sử dụng bởi tội phạm (như khai thác lỗ hổng, spam email, ...).

Web crawler cho phép thu thập dữ liệu một cách dễ dàng, tuy nhiên nó cũng dẫn đến nhiều vấn đề đã được nêu ra trong [9]. Những vấn đề đó bao gồm:

- suy giảm chất lượng dịch vụ: nếu không có ràng buộc thích hợp, các hành vi của crawler sẽ khiến server phải chịu thêm tải không cần thiết, từ đó gây nên suy giảm chất lượng dịch vụ cho người dùng thông thường. Theo báo cáo [8], khoảng 40%-60% lưu lượng truy cập Internet là xuất phát từ các chương trình web crawler.

- ảnh hưởng tới độ chính xác trong phân tích dữ liệu: giảm độ chính xác của việc phân tích dữ liệu người dùng truy cập website do hầu hết nhà phân tích đều coi rằng lượng truy cập website này đều đến từ các người dùng thông thường.

- lo lắng trong kinh doanh: dữ liệu trên nhiều website là những dữ liệu được sử dụng nhằm mục đích thương mại, do đó việc ngăn chặn hành vi thu thập dữ liệu của crawler được quân tâm hàng đầu.

Vấn đề phát hiện crawler được phát biểu như sau [9]: Cho R là tập hợp các bản ghi http request tới một máy chủ web, cần phân loại mỗi http request $r \in R$ tới máy chủ được gửi từ người dùng hay được gửi từ chương trình tự động web crawler. Mỗi bản ghi http request tới máy chủ có định dạng được xây dựng theo kịch bản cài đặt máy chủ của quản trị viên, mức độ chi tiết của bản ghi phụ thuộc vào cấu hình hệ thống log ở máy chủ. Về cơ bản, mỗi bản ghi sẽ gồm:

- địa chỉ gửi,

- URL của tài nguyên yêu cầu,

- mã http response (ví dụ 404 nếu tài nguyên không tìm thấy ở máy chủ) và

- trường user-agent.

Các hướng giải quyết đã đưa ra cho bài toán phát hiện crawler thực hiện phân loại http request theo từng phiên (session) thay vì từng http request riêng. Một session $S \subseteq R$ là tập các bản ghi http request được gửi từ một người dùng trong một phiên truy cập website. Trong trường hợp này, nhãn của session S sẽ tương ứng với nhãn của tất cả các http request trong S và trong R được xây dựng lên từ tập các session $S_1, S_2, S_3, \ldots, S_n$, tức là $S_1 \cap S_2 \cap S_3 ... \cap S_n = \mathrm{R}$.

Yêu cầu đặt ra cho lời giải bài toán đó là việc phát hiện phải được thực hiện theo thời gian thực, tức là crawler cần phải được phát hiện trong khi nó đang thực hiện thu thập dữ liệu từ website. Vấn đề phát hiện crawler theo thời gian thực có thể được xem là sự kết hợp giữa vấn đề phòng chống xâm nhập và phát hiện xâm nhập. Phòng chống thâm nhập nghĩa là ngăn chặn các truy cập không được cấp phép tới hệ thống, còn phát hiện xâm nhập là phát hiện các truy cập trái phép đang hoạt động trong hệ thống. Lợi ích của việc phát crawler thời gian thực đó là cho phép áp đặt các hạn chế ngay lập tức đối với crawler đang truy cập tài nguyên trái phép. Các hạn chế đó có thể là ngăn chặn crawler truy cập hay giảm độ ưu tiên khi phục vụ yêu cầu từ crawler so với yêu cầu từ người dùng khi server chịu tải quá mức nào đó, ...

2.2 Phát hiện crawler theo session

Session là tập hợp gồm các truy vấn do một đối tượng có địa chỉ IP xác định thực hiện, thể hiện bằng việc đối tượng đó truy cập vào các đường link, hình ảnh có trong một trang web. Bài toán của chúng ta là làm thế nào để từ những session đó, chúng ta có thể phát hiện được session nào là do người thật thực hiện, session nào là do bot thực hiện để chúng ta có hình thức ngăn chặn.

Để phân biệt được session nào là do bot thực hiện, session nào là do người thật thực hiện, ta có thể dựa vào 2 thông tin chính như sau:

- Thứ nhất, crawler đươc lập trình để có thể thu thập được thông tin càng nhiều, càng nhanh càng tốt. Vì vậy, trong quá trình crawl một trang web cụ thể, crawler sẽ truy cập vào tất cả các đường dẫn mà nó gặp. Điều này sẽ dẫn đến sự bất thường khi nhìn vào thời gian δ_t, nghĩa là những truy vấn ở gần nhau thì khoảng thời gian truy cập vào những truy vấn này sẽ rất gần nhau, trong khi đó những truy vấn nằm ở vị trí xa hơn thì thời gian truy cập sẽ lâu hơn.

- Thứ hai, cũng chính vì mục đích ban đầu là thu thập được càng nhiều thông tin càng tốt mà đôi khi crawler thường sẽ không quan tâm đến nội dung của bài viết mà nó sẽ truy cập. Chính vì vậy, nội dung những bài viết mà một crawler truy cập đôi khi sẽ không liên quan đến nhau, và đây là là một dấu hiêu quan trọng để ta nhận biết được những truy vấn được thực hiện từ địa chỉ IP xác định này có phải là bot hay không, bởi vì thông thường, con người thường sẽ có xu hướng đọc và tìm hiểu những thông tin có liên quan đến nhau.

Ví dụ minh họa cho trường hợp này, số lần truy cập vào đối tượng ảnh của một người bình thường thường chiếm 10% trong tổng số các đối tượng, đường dẫn mà họ truy cập, trong khi đó chỉ 1% của tập session được truy cập bởi crawler có chứa yêu cầu truy cập ảnh mà thôi.

Ngoài ra, ta hoàn toàn có thể có một vài căn cứ khác, để đưa ra quyết định nhưng truy vấn trong 1 tập session có phải là do bot thực hiện hay không, đó là căn cứ vào vị trí của các đường link trong một trang web cụ thể. Điều này có thể biết được khi ta đọc mã nguồn html của trang web. Nếu ta phát hiện thấy các đối tượng truy cập vào các đường link một cách lộn xộn, không theo thứ tự trái phải trên xuống chẳng hạn, thì đó cũng là một cơ sở để ta quyết định xem đó là bot hay là người.

2.3 Mô tả bài toán

3 Mô hình đề xuất

Bài báo cáo này xây dựng mô hình phát hiện crawler dựa trên kiến trúc của mạng GAN [6] và LSTM [5]. Trước hết, báo cáo xin trình bày cơ bản về mạng GAN và mạng LSTM.

3.1 Kiến trúc mạng sử dụng

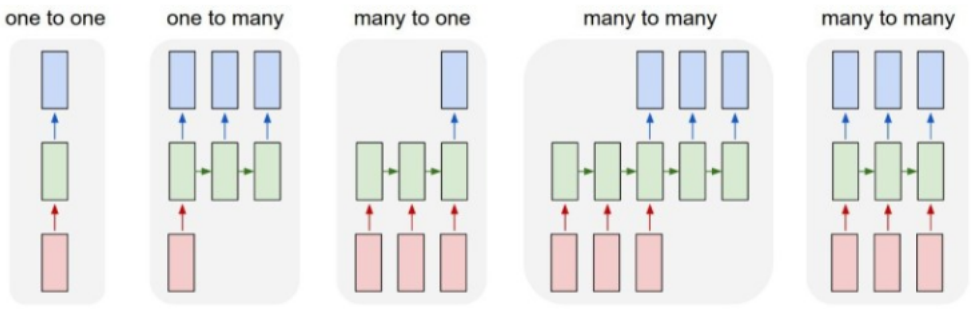
Mạng LSTM

Đối với dữ liệu dạng chuỗi tuần tự, có một mô hình cơ bản phù hợp với loại dữ liệu này đó là mạng neuron hồi quy RNN (Recurrent Neuron Network), và LSTM là một biến thể của RNN cho phép khắc phục được một số nhược điểm của RNN. Phân loại các bài toán sử dụng mạng RNN như sau:

- *One to one:* mẫu bài toán cho Neural Network (NN) và Convolutional Neural Network (CNN), 1 input và 1 output, ví dụ với CNN input là ảnh và output là ảnh được segment.

- *One to many:* bài toán có 1 input nhưng nhiều output, ví dụ: bài toán caption cho ảnh, input là 1 ảnh nhưng output là nhiều chữ mô tả cho ảnh đấy, dưới dạng một câu.

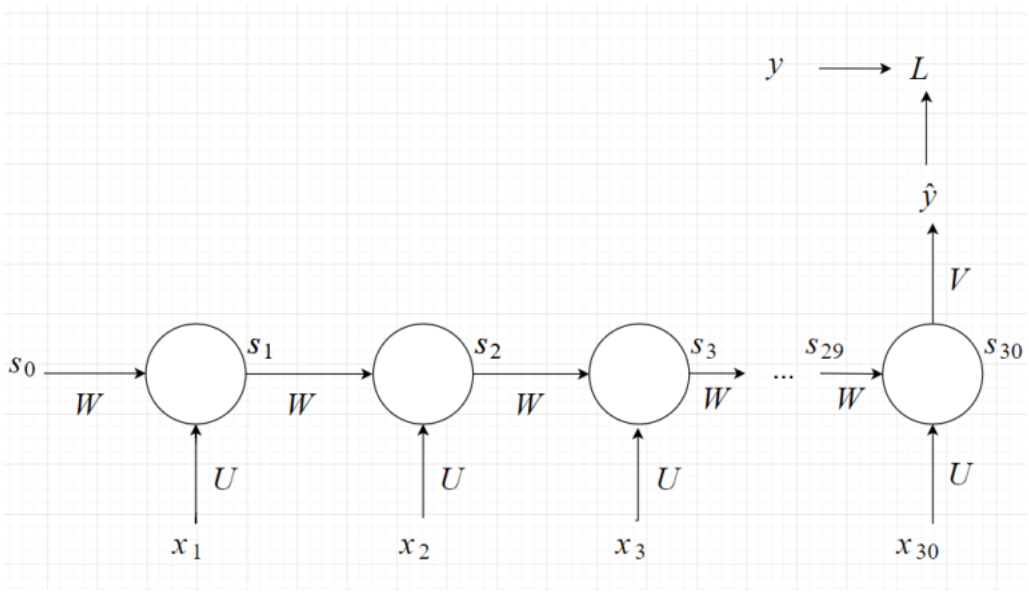
- *Many to one:* bài toán có nhiều input nhưng chỉ có 1 output, ví dụ bài toán phân loại hành động trong video, input là nhiều ảnh (frame) tách ra từ video, output là hành động trong video.



Hình 4: Phân loại bài toán sử dụng mạng RNN

- *Many to many:* bài toán có nhiều input và nhiều output, ví dụ bài toán dịch từ tiếng anh sang tiếng việt, input là 1 câu gồm nhiều chữ: "I love Vietnam" và output cũng là 1 câu gồm nhiều chữ "Tôi yêu Việt Nam".

Kiến trúc mạng RRN: Giả sử ta có đầu vào là chuỗi $(x_1, x_2, \ldots, x_{30})$, U, W, V là các tham số



Hình 5: Kiến trúc mạng RNN

của mô hình; $\widehat{y}$ là đầu ra của mô hình; L là hàm mất mát. Chúng ta cần phải tối ưu L theo các tham số U, W, V bằng phương pháp gradient descent thông qua việc tính $\frac{\partial L}{\partial U}, \frac{\partial L}{\partial V}, \frac{\partial L}{\partial W}$. Chúng ta sẽ nhìn thấy vấn đề của mạng RNN thông qua việc tính đạo hàm. Ở đây, ví dụ với việc tính đạo hàm $\frac{\partial L}{\partial W}$:

$$\frac{\partial L}{\partial W} = \sum_{i=0}^{30} \frac{\partial L}{\partial \widehat{y}} * \frac{\partial \widehat{y}}{\partial s_{30}} * \frac{\partial s_{30}}{\partial s_i} * \frac{\partial s_i'}{\partial W}$$

Trong đó: $\frac{\partial s_i'}{\partial W}$ là đạo hàm của s_i với W khi coi s_{i-1} là không đổi đối với W,

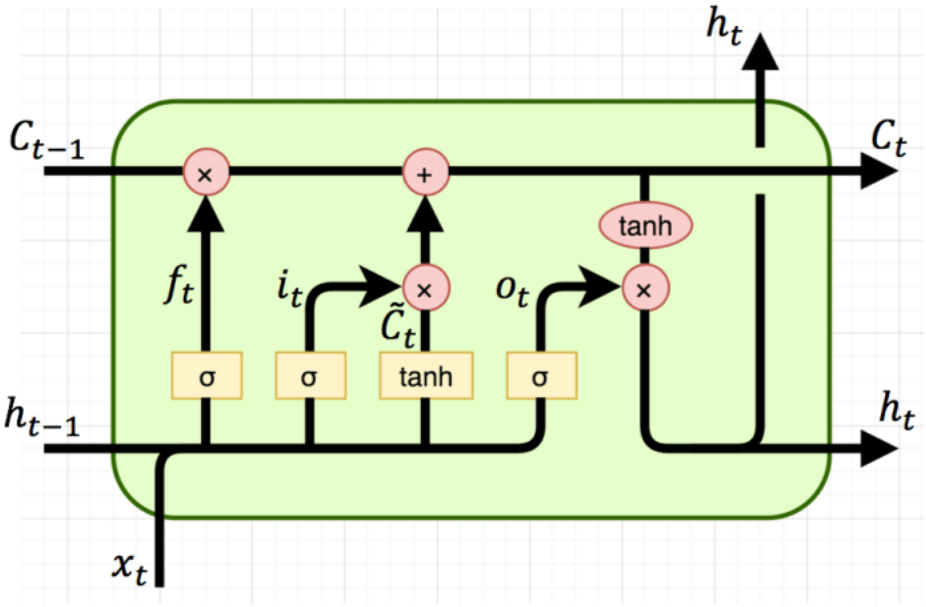
$$\frac{\partial s_{30}}{\partial s_i} = \prod_{j=i}^{29} \frac{\delta s_{j+1}}{\delta s_j}$$

Giả sử activation là tanh function $s_t = tanh(U * x_t + W * s_{t-1})$

$$\frac{\delta s_t}{\delta s_{t-1}} = (1 - s_t^2) * W \Rightarrow \frac{\delta s_{30}}{\delta s_i} = W^{30-i} * \prod_{j=i}^{29}(1 - s_j^2)$$

Do $s_j < 1, W < 1$, nên những state ở xa thì $\frac{\delta s_{30}}{\delta s_i} \approx 0$, đây là hiện tượng vanishing gradient.

Ta có thể thấy là các state càng xa ở trước đó thì càng bị vanishing gradient và các hệ số không được update với các frame ở xa. Hay nói cách khác là RNN không học được từ các thông tin ở trước đó xa do vanishing gradient. Do đó dẫn tới sự ra đời của LSTM. Mỗi cell trong mạng LSTM có cấu trúc như sau: Ở state thứ t của mô hình LSTM:



Hình 6: Kiến trúc một cell trong LSTM

- Output: C_t, h_t, ta gọi C là cell state, h là hiden state
- Input: C_{t-1}, h_{t-1}, x_t. Trong đó, x_t là input ở state thứ t của model. C_{t-1}, h_{t-1} là output của layer trước.

Từ biểu đồ, phép nhân ở đây là element wise multiplication, phép cộng là phép cộng ma trận. F_t, i_t, o_t tương ứng với forget gate, input gate và output gate:

- Forget gate:

$$f_t = \sigma(U_f * x_t + W_f * h_{t-1} + b_f)$$

- Input gate:

$$i_t = \sigma(U_i * x_t + W_i * h_{t-1} + b_i)$$

- Output gate:

$$o_t = \sigma(U_0 * x_t + W_0 * h_{t-1} + b_0)$$

Forget gate quyết định xem cần lấy bao nhiêu từ cell state trước và input gate sẽ quyết định lấy bao nhiêu từ input của state và hidden layer của layer trước. Output gate quyết định xem cần lấy bao nhiêu từ cell state để trở thành output của hidden state. Ngoài ra h(t)cũng được dùng để tính ra output y(t) cho state t.

Mạng GAN

GAN là viết tắt "Generative Adversary Networks", hướng tới việc sinh ra dữ liệu mới sau quá trình học. Một mạng GAN thì thường gồm có 2 phần chính, đó là Discriminative model, viết tắt là D, phần thứ 2 là Generative model, viết tắt là G.

- Discriminative model Trước đây, hầu hết các mạng đều thuộc dạng Discriminative model, tức trải qua quá trình huấn luyện, model đạt được khả năng định vị được vị trí của một điểm dữ liệu trong phân bố dữ liệu (điển hình là bài toán phân loại). Ví dụ: một đứa trẻ được cho xem 1000 ảnh có mèo và 1000 ảnh không mèo (ảnh kèm nhãn có-không). Trải qua quá trình quan sát và học hỏi, đứa trẻ đó sẽ có khả năng phân biệt những ảnh mới xem ảnh nào có, ảnh nào không có mèo. Đó là điển hình cho một "discriminative model". Các bài toán classify, regression, image semantic segmentation, object detection . . . bản chất đều liên quan tới "discriminative model".

- Generative model Cũng với ví dụ trên, trải qua sau quá trình học liệu đứa bé có thể tự hình dung ra hình ảnh một con mèo mới nào đó. Việc sinh ra ảnh mới đó là việc của "generative model".

Thuật toán của GAN được kết hợp bởi 2 model, cụ thể như sau:

1. Bước 1: Từ một nhiễu z bất kì, G sinh ra fake-data G(z) có nội dung giống như dữ liệu thật (dữ liệu là x). Tại lần sinh đầu tiên, G(z) hoàn toàn là dữ liệu nhiễu, không có bất kì nội dung gì đặc biệt.

2. Bước 2: x và G(z) cùng được đưa vào D kèm nhãn đúng sai. Train D để học khả năng phân biệt dữ liệu thật, dữ liệu giả.

3. Bước 3: Đưa G(z) vào D, dựa vào feedback của D trả về, G sẽ cải thiện khả năng fake của mình.

4. Bước 4: Quá trình trên sẽ lặp đi lặp lại như vậy, D dần cải thiện khả năng phân biệt, G dần cải thiện khả năng fake. Đến khi nào D không thể phân biệt được dữ liệu nào là dữ liệu do G tạo ra, dữ liệu nào là x, khi đó quá trình dừng lại.

Để train được D, input gồm cả G(z) và x kèm nhãn. Như vậy, mục tiêu của D là maximinze:

$$\max_D V(D) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1 - D(G(z)))]$$

Thành phần thứ nhất là giúp nhận diện dữ liệu thật tốt hơn, thành phần thứ hai giúp nhận diện dữ liệu tạo ra từ G tốt hơn.

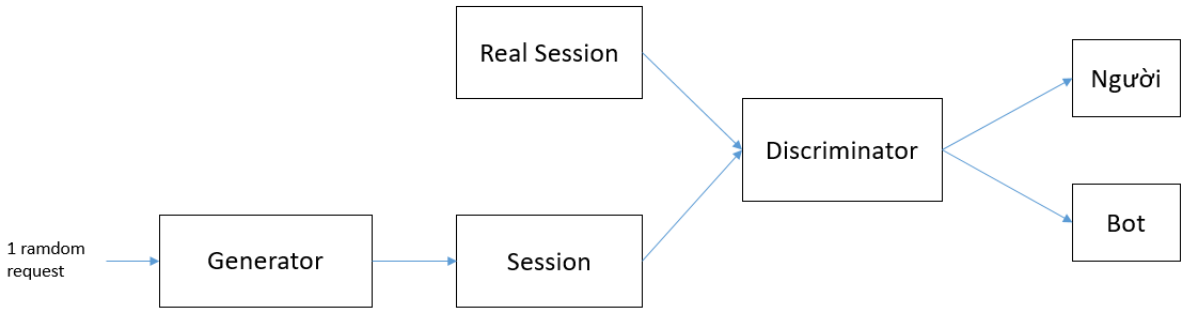
Để train được G, ta dựa vào D(G(z)). Bước này nhằm mục đích update các weight của G sao cho G(z) có thể đánh lừa được D, khiến D đoán nhầm nhãn của G(z) là y = 1. G cố gắng minimize:

$$\min_G V(G) = E_{z\sim p_z(z)}[log(1 - D(G(z)))]$$

Tổng hợp lại ta có D, G là kết quả của quá trình:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1-D(G(z)))]$$

3.2 Áp dụng vào bài toán



Hình 7: Mô hình đề xuất

Mục đích của chúng ta là tạo ra một hệ thống vừa có thể phát hiện được đâu là bot, đâu là người để có biện pháp xử lý thích hợp, lại vừa có thể tạo ra được 1 crawler có khả năng vượt qua được nhữn hệ thống phát hiện bot hiện nay đang có. Vì vậy, mô hình mạng GAN được sử dụng để đáp ứng yêu cầu này.

Bài toán xử lý thông tin dạng chuỗi, yêu cầu độ chính xác cao, phát hiện theo thời gian thực, đảm bảo có thể lưu trữ được các state ở xa để phục vụ cho việc dự đoán, chính vì vậy LSTM(Long Short Term Memory) được sử dụng ở cả 2 model G và D.

Mô hình sinh G: Sinh ra 1 session từ một url ngẫu nhiên bất kỳ, được chọn từ tập không gian url đã biết.

Mô hình kiểm tra D: từ tập dữ liệu được gán nhãn, tiến hành phân loại các session, nếu session đó được thực hiện bởi bot, thì gán nhãn là 0, nếu là do người thực hiện thì được gán nhãn là 1

GAN: đầu vào của là đầu vào của G, đầu ra của GAN là đầu ra của D, mục đích là tạo ra một G đủ mạnh để kiếm cho việc phân loại của D gặp nhiều khó khăn nhất có thể

3.3 Một số vấn đề tồn tại và hướng giải quyết đề xuất

Các vấn đề còn tồn tại trong khi thử nghiệm mô hình:

- Chưa thống nhất được phương pháp vector hóa các url để đưa vào mô hình một cách hiệu quả.
- Việc sinh ra một session có chất lượng từ một url cho trước còn chưa rõ ràng.
- Đầu ra của G có dạng chưa ăn khớp với đầu vào của D, chính vì vậy mà chưa thực thi được mô hình.

Một số hướng giải quyết đề xuất:

- Thứ nhất, đó là sửa lại G sao cho đầu ra của nó tương thích với đầu vào của D.
- Thứ hai, nếu việc làm trên không khả thi, phải định nghĩa lại mô hình GAN.

4 Kết luận

Báo cáo đã trình bày các khái niệm liên quan tới dữ liệu chuỗi thời gian bao gồm khái niệm, các đặc trưng và các phương pháp thống kê và học máy áp dụng trong phân tích chuỗi thời gian. Đồng thời cũng áp dụng phương pháp học máy vào xây dựng mô hình phát hiện crawler, với dữ liệu chuỗi thời gian ở đây đó chính là thông tin về các request mà crawler gửi tới máy chủ. Tuy nhiên, mô hình mới dừng lại ở mức đề xuất, trong quá trình thử nghiệm có xuất hiện nhiều vấn đề đã có hướng giải quyết đề xuất nhưng chưa được triển khai. Hy vọng, mô hình bài toán sẽ được hoàn chỉnh trong thời gian tới.

Tài liệu tham khảo

[1] Định lý Nyquist-Shannon,
`https://en.wikipedia.org/wiki/Nyquist-Shannon_sampling_theorem`

[2] Quá trình ngẫu nhiên,
`https://en.wikipedia.org/wiki/Stochastic_process`

[3] Thống kê nồng độ CO_2,
`https://www.co2.earth/monthly-co2`

[4] G.E.P.Box, G.M.Jenkins and G.C.Reinsel. *"Time Series Analysis: Forecasting and Control"*, San Francisco: Holden-Day, 1994

[5] Mô hình mạng LSTM,
`https://colah.github.io/posts/2015-08-Understanding-LSTMs`

[6] Mô hình mạng GAN,
`https://pathmind.com/wiki/generative-adversarial-network-gan`

[7] Alexander Menshchikov, Antonina Komarova, Yurij Gatchin, Anatoly Korobeynikov, Nina Tishukova. *"A Study of Different Web-Crawler Behaviour"*, In PROCEEDING OF THE 20TH CONFERENCE OF FRUCT ASSOCIATION

[8] Lu, P. *Bring you into the world of crawler and anti-crawler*, Softw. Integr. Circ.12, 12–13 (2016)

[9] Derek Doran, Swapna S. Gokhale. *"Web robot detection techniques: overview and limitations"*, Data Min Knowl Disc (2011) 22:183–210

[10] Weiping Zhu, Hang Gao, Zongjian He, Jiangbo Qin, and Bo Han. *"A Hybrid Approach for Recognizing Web Crawlers"*, Springer Nature Switzerland AG 2019, E. S. Biagioni et al. (Eds.): WASA 2019, LNCS 11604, pp. 507–519, 2019.