Sketch-based Change Detection: Methods, Evaluation, and Applications

ACM 2003

B Krishnamurthy (AT&T), Yin Zhang (University of Texas Austin), …

742 citations

Related Keywords: change detection, DataStream Computation, Sketch

---

تشخیص بی نظمی‌ها[1](تغییرات) در شبکه‌های پویای امروزی امری مهم می‌باشد. برخی از این بی نظمی‌ها قابل پیش بینی، عادی و تحمل پذیر می‌باشند، اما برخی دیگر نشانی از یک حمله می‌توانند باشند. دو روش شناسایی بی نظمی وجود دارد: مبتنی بر امضا[2] یا روش‌های آماری[3] که امکان شناسایی حملات جدید را نیز دارا می‌باشد و یک مدلی از رفتار نرمال شبکه به دست می‌آورد. الگوریتم‌های تشخیص بی نظمی، معمولاً ترافیک را به صورت مجموعه‌ای از جریان‌ها می‌بینند و سعی می‌کنند تغییرات را تشخیص دهند[4]. اما با افزایش حجم داده‌ها، نگهداری اطلاعات هر جریان[5] ناممکن است. اما ما در این مقاله از داده ساختاری جدید بر مبنای اسکچ[6]ها که اطلاعات را به صورت فشرده نگهداری می‌کنند، به نام k-array استفاده می‌کنیم. از میزان ثابت و ناچیزی از حافظه استفاده می‌کند و هزینه به روزرسانی و بازسازی هر رکورد نیز ثابت است. سپس یکسری مدلهای پیش‌بینی سری زمانی[7](ARIMA,HoltWinters,…) طبق این اطلاعات فشرده پیاده‌سازی می‌کنیم و جریان‌هایی که بیشترین خطا از مقدار پیش‌بینی شده را داشتند، شناسایی می‌کنیم. همچنین یک سری مکاشفه برای تنظیم پارامترهای خودکار مدل نیز ارایه می‌دهیم. سپس با استفاده از داده‌های جمع‌آوری شده از یک ISP، نتیجه می‌گیریم که روش ارایه شده دقت بالایی دارد و می‌تواند در دستگاه‌های با سخت افزار پایین پیاده‌سازی شود.

روش‌های قدیمی تشخیص تغییر بر پایه پیش بینی سری زمانی و آنالیز داده‌های پرت در نرخ ترافیک‌های حجیم امروزی با تعداد زیادی از سری‌های زمانی جواب نمی‌دهند چون برای هر جریان می‌خواهند اطلاعاتی نگهداری کنند. روشی که می‌خواهیم ارایه دهیم در زمینه data stream computation (اون دو ویژگی معروف یعنی بررسی هر تاپل تنها یک بار ویا به مقدار کم و استفاده محدود از حافظه را دارا می‌باشد) می‌باشد. اسکچ‌ها یکی از تکنیک های موجود در این زمینه می‌باشند که با استفاده بهینه از حافظه، از ویژگی های تضمین بازسازی با نرخ احتمال بالای قابل اثبات و خطی بودن برخوردار می‌باشند.

مدل‌های دیتااستریم: مدل‌های مختلفی برای توصیف دیتااستریم‌ها ارایه شده است: Time Series Model, Cache Register Model, Turnstile Model. که در اینجا از Turnstile Model استفاده می‌کنیم:

---

[1] Anomaly
[2] Signature-Based
[3] Statistical Approaches
[4] Change Detection
[5] Per-flow analysis
[6] Sketch
[7] Time-series forecasting

هدف این است که سیگنال‌های با تغییرات شاخص را شناسایی کنیم. مدل ارایه شده بسیار کلی است و می‌تواند در زمینه‌های مختلف به کاررود: مثلاً کلید می‌تواند همان مشخصات flow ID باشد و آپدیت هم اندازه ی جریان (به بایت) باشد. در این مقاله ما از تنها آدرس مقصد و جمع بایت‌ها در آزمایش‌هایمان استفاده کرده‌ایم.

معماری:

روش ما شامل سه کامپوننت زیر است:

- Sketch module
- Forecasting Module
- Change Detection Module

Sketch module:

اسکچ‌ها از یک داده ساختار فشرده برای ذخیره v_a در هر دوره زمانی می‌کند. اسکچ‌ها داده ساختارهای احتمالی بر مبنای random projection می‌باشند. یک اسکچ خاص به نام k-array ارایه می‌دهیم که بر پایه count sketch می‌باشد. اما عملیات روی آن ساده تر و کاراتر می‌باشد. یک اسکچ S از HxK جدول رجیستری تشکیل شده‌است:

$Ts[i][j] : i \in [H], j \in [K]$

تعداد توابع هش

سایز هر اول هش

هر سطر یک تابع هش $h_i: [u] \to [k]$ می‌باشد.در واقع این داده ساختار، آرایه‌ای از جداول هش می‌باشد.
توابع هش بایستی 4-universal باشند تا دقت بازسازی تضمین شود. عملیاتی که داریم:

- Update : به روزرسانی اسکچ
- Estimate : بازسازی v_a برای کلید a
- estimateF2
- combine : ترکیب خطی چندین اسکچ

- Update(S, a, v): $\forall i \in [H], T_S[i][h_i(a)] += v$

- Estimate(S, a):

$$v_a^{estimated} = \text{median}_{i \in [H]} v_a^{h_i}$$

$$sum(S) = \sum_{j \in [K]} T_S[0][j]$$

$$v_a^{h_i} = \frac{T[i][h_i(a)] - \frac{sum(S)}{K}}{1 - \frac{1}{K}}$$

- EstimateF2(S):

$$F_2^{h_i} = \frac{K}{K-1} \sum_{j \in [K]} (T_S[i][j])^2 - \frac{1}{K-1} (sum(S))^2$$

$$F_2^{est} = \text{median}_{i \in [H]} F_2^{h_i}$$

- Combine($c_1, S_1, \dots, c_\ell, S_\ell$):

$$S = \sum_{k=1}^{\ell} c_k . S_k \quad , \quad T_S[i][j] = \sum_{k=1}^{\ell} c_k . T_{S_k}[i][j]$$

Forecasting module: از اسکچهای مشاهده شده در بازه های قبلی $S_o(t'<t)$ برای ساخت اسکچ پیشبینی $S_f(t)$ و سپس محاسبه خطای $S_e(t)$ استفاده می‌کند۶ مدل برای پیش بینی سری زمانی تک متغیره و تشخیص تغییر استفاده می‌کنیم.چهارتای اول مدلهای نرم کننده[8] ساده و بقیه از خانواده ARIMA می‌باشند.

Moving Average:

$$S_f(t) = \frac{\sum_i^w S_f(t-1)}{w} \quad (w>1)$$

وزنگ‌ها به همه نمونه‌های قبلی وزن یکسان می‌دهد

پارامترها ← w: تعداد نمونه‌های قبلی

مدلهای Arima : یک دسته از تکنیک‌های پیشبینی سری‌های زمانی می‌باشند. که می‌توانند رفتارهای متفاوتی را برای پیش‌بینی سری‌های زمانی تک متغیره و تشخیص تغییر مدل کنند.

ARIMA(p,d,q) یک مدل یعنی

p: # auto regressive parameters

q: # moving average parameters

d: # differencing passes

پارامترهای بالایی پس از چندبار تفاضل‌گیری سری زمانی اصلی

$$z_t - \sum_{i=1}^{q} MA_i \cdot z_{t-i} = C + e_t - \sum_{j=1}^{p} AR_j \cdot e_{t-i}$$

z_t : z بار d difference گرفتن سری زمانی اصلی

e_t : ارور پیش‌بینی در زمان t

MA_i, AR_j ضریب‌های ثابت‌اند. (p بزرگترین و q بزرگترین)

در بحث اسکچ‌ها دو نوع ARIMA را خواهیم داشت:

ARIMA0: $(p \le 2, d=0, q \le 2)$

ARIMA1: $(p \le 2, d=1, q \le 2)$

[8] Smoothing model

Change detection module: بعد از محاسبه $s_e(t) = s_0(t) - s_f(t)$. این ماژول بر اساس مقدار T_A تصمیم می‌گیرد.

$T_A = T \cdot [\text{ESTIMATEF2}(S_e(t))]^{1/2}$

برای هر کلید حال کافی است Estimate(S_e(t),a) را با T_A مقایسه کنیم.

این که به چه نحوی مقادیر کلید را ذخیره کنیم تا بتوانیم آنهارا به این ماژول بدهیم، یک مساله است. روشی که استفاده می‌کنیم (به جای نگهداری اطلاعات هر جریان) S_e(t) را در ابتدا محاسبه می‌کنیم و سپس تغییرات را در مرحله بعد حساب می‌کنیم. چون خود استریم ورودی تمام کلیدها را دارا می‌باشد نیازی به ذخیره همه آنها برای هر جریان نیست. اما نیاز به دسترسی مجدد به استریم دارد و لذا در موارد آفلاین کاربرد دارد.

تنظیم پارامترها:

پارامترهای Hو K اسکچ می‌تواند در کارایی و مطمین بودن ارور های پیش‌بینی تاثیرگذار باشد.یکی از توابع مکاشفه ای که می توانیم استفاده کنیم(برای پارامترهای مدل)، کمینه کردن $\sum_t F_2^{est}(S_e(t))$ (انرژی اضافی) می‌باشد.الگوریتمی که برای این کار استفاده می‌کند نیز جستجوی multi-pass grid می باشد(مثلاً برای مدل EMWA).

برای پارامترهای مدل: به مقاله مراجعه کن.

راه اندازی محیط آزمایشگاهی: دیتاست شامل داده‌های نت فلو که از روترهای موجود در یک ISP گرفته شده است.

نتایج آزمایش: پس از تعیین این‌که چه پارامترهایی برای هر مدل مناسب است (بر اساس grid search) ، باید درستی و دقت اسکچ‌ها را در مقایسه با per-flow ها نیز بررسی کنیم. برای K=32k نتیجه می‌شود که مثبت کاذب[9] و منفی کاذب[10] بسیار کاهش می‌یابد.

کارهای آینده:

[9] False positive
[10] False negative

- *شناسایی تغییرات به خط*[11]: برای شناسایی تغییرات به صورت بلادرنگ[12] یکی از الزامات اینست که پارامترهای مدل پیش‌بینی به کمک داده‌های قبلی به صورت دوره‌ای محاسبه شوند تا با تغییرات موجود در رفتار ترافیک سازگار باشند.
- *دوری از اثرات مرزی به دلیل سایز* ***interval*** *های ثابت*: راهکارهای ممکن شامل: اجرای چندین مدل به صورت همزمان با interval های مختلف و نقاط شروع متفاوت، یا تصادفی کردن سایز intervalمی‌باشد.
- *کاهش نرخ مثبت کاذب*: یکی از مشکلاتی که برای ترافیک‌هایی که عادی هستند رخ می‌دهد.(بی نظمی‌های عادی). با تنظیم کردن پارامترها با شرایط محیطی مساله می‌توان این را برطرف کرد. یک تکنیک اینست که تنها تغییرات اساسی را گزارش کند.
- *کامپایل با نمونه برداری*: به همراه اسکچ‌ها می‌تواند مقیاس پذیری را بالا ببرد.
- *راهنما برای انتخاب پارامترها*: با استفاده از تکنیکی که در مقاله ۵ انتخاب شده می توان محدوده شان را تعیین کرد.

[11] Online change detection
[12] Real-time