2022 年 9 月 29 日

2022 年度 I 類 (情報系) コンピュータサイエンスプログラム 卒業研究中間発表会

動的に要素が変化する多重集合に対する Min-hash の空間計算量の削減

I 類 (情報系) 学籍番号：1920031 古賀研究室 山川 竜太郎

1 はじめに

近年、IoT や SNS の発展に伴いストリームデータが取り扱われる機会が増え、ストリームデータを対象とする類似検索の重要性が増している。類似検索ではストリームデータを要素が動的に変わる集合とみなし、集合間類似検索により、類似ストリームデータを探す。集合間類似度としては Jaccard 係数が良く用いられる。しかし、集合が変わるたびに Jaccard 係数を計算しなおすのはオーバーヘッドが大きい。そのため、各集合に対するコンパクトなスケッチを Min-Hash というハッシュ関数により生成し、スケッチ間で Jaccard 係数を近似計算する手法が提案されている。本研究では、「データストリームを対象とした動的多重集合に対する Min-hash の高速計算アルゴリズム」の研究を引継いでいる。[1]

1.1 Min-hash

集合に対する確率的なハッシュ関数である。Jaccard 係数の算出を高速化するために、Min-hash は計算されたハッシュ値が一致する確率は Jaccard 係数と一致するという性質を持ち、それを利用する。(1)

$$Pr[h(A) = h(B)] = \frac{A \cap B}{A \cup B} \tag{1}$$

集合に対する類似検索をハッシュテーブルで実現可能である。

式 (1) で h はハッシュ関数であり、A,B は集合である。Min-hash によるハッシュ値の計算方法は、$\Phi = \{x_1, x_2, \cdots x_{|\Phi|}\}$ をアルファベット集合としたとき、ハッシュ関数はΦの各アルファベットに対して $\{1, 2, \cdots |\Phi|\}$ の中から被らないようにランダムな値を割り当てることで決定される。1 つのある集合の中のアルファベットを見て、その中の要素に対応する割り当て値の中から最小値を選ぶ。最小値が Min-hash によるハッシュ値となる。

1.2 スライディングウインドウモデル

スライディングウインドウモデルは、データストリームの直近 W 個の要素をスライディングウインドウと定義し、時刻が進むとウインドウに到着データを追加し、ウインドウ内の最古のデータを破棄する手法である。

	a	b	c	d
1個目	3	4	7	6
2個目	5	1	8	2

図 1: 多重集合のハッシュ値の割り当て表

1.3 多重集合

一般にオブジェクトの集まりを集合と呼ぶ．例えば a, b, c はアルファベットを要素とする集合である。一般的には要素群を表すアルファベットに対して、その要素を 0 個以上含むものが集合となる。通常、集合は同じ種類の要素を 1 つしか含まない。多重集合とは、同一要素の重複を許す集合である。例は、a,b,b,c,c,d,d である。

1.4 従来研究 SWMH (Sliding Window Min-hash)

ストリームデータをスライディングウィンドゥモデルで多重集合を扱える手法が従来研究としてある。それが SWMH(Sliding Window Min-hash) であり、ストリームデータに対するハッシュ値算出アルゴリズムである。SWMH は、時間経過によって要素が動的に変化する。

2 多重集合に対する Min-hash

多重集合に対して Min-hash のハッシュ値を計算する手法として、多重集合内の複数個のラベルに異なる値を割り当てている。その時に将来に割り当て値が最小になる可能性が絶対にない割り当て要素は、直前の割り当て値と同じなるように割り当て表を編集する。割り当て表を図 (1) に示す。将来最小値になりうる要素のみを Minlist というリストで管理する。

2.1 ハッシュ値の算出

多重集合に対するハッシュ値算出アルゴリズムは以下のとおりである。

1 各要素に割り当て表の数値を割り当てる

2 最小値をハッシュ値とする

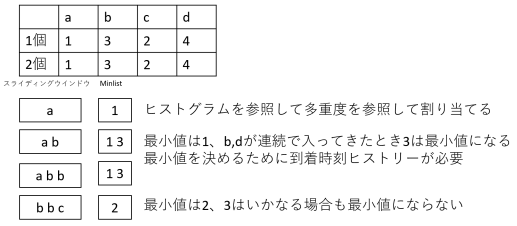

図 2: Minlist の更新アルゴリズム

3 Minlist を使用した高速化

Minlist は最小値となる候補を持っているデータ構造である。新しくデータがスライディングウインドウに入ってきたとき以下の方法で Minlist を行進する。Minlist の更新アルゴリズムを図 (2) に示す

1 ヒストグラムを参照して多重度を参照して割り当てる

2 到着時刻ヒストリを参照して到着時刻によって Minlist を更新する

3.1 要素のヒストグラム

スライディングウインドウ内に要素の数をカウントする。多重集合である場合、同一要素が含まれているかわからないと Min-hash のハッシュ値を計算できない。

3.2 到着時刻のヒストリ

Minlist を作成するために必要があるため、要素がいつの時刻で到着したかリストとして所持する。リストは到着時刻が昇順で保持される。新たにウィンドウに到着した要素の到着時刻がエンキューされ、ウインドウから出ていく要素の到着時刻がデキューされる

4 従来研究の課題

SWMH(Sliding-Window min-Hash) は、データを保持するために空間計算量が大きい。そのため「要素のヒストグラム」「Min-Hash の割り当て表」「到着時刻のヒストリ」のデータ構造をそれぞれ小さくする。「要素のヒストグラム」は、リストのサイズが要素の種類数になる。「Min-Hash の割り当て表」は要素の種類数×多重度の上限値のサイズを持つ。「到着時刻のヒストリ」は、スライドウインドウ内の全要素の到着時刻を記録している。

5 改善案

「要素のヒストグラム」「Min-Hash の割り当て表」「到着時刻のヒストリ」のそれぞれに改善する方針を決めた。

1 要素のヒストグラム: Count-Min Sketch による近似ヒストグラムを使う [2]

2 Min-Hash の割り当て表: Active Index を使って無駄な多重度を削減

3 到着時刻ヒストリ: リストを固定長にして、使用確立が低いものはあらかじめ持たない

5.1 Active Index

Active Index を用いて割り当て表に数値を割り当てるとき、多重度によって割り当て値が切り替わる境界値のインデックスと割り当て値を持つ。例は、多重度 1 のときに割り当て値が 10、多重度 2 以降の割り当て値が 9 の場合は、1,10,2,9 のように割り当て表を持つ。

6 今後の予定

ヒストグラムと割り当て表の削減を行う。現在は、Active Index と Count-Min Sketch を SWMH モデルに適用できるかどうか、検討している段階である。参考文献を探して、SWMH の空間計算量を削減することを目標としている。まずは SWMH の中に組み込めるか否かを疑似コードを作成して検証する。到着時刻ヒストリについてはすでに実装が完了している。

参考文献

[1] 三原寛寿, 古賀久志, “データストリームを対象とした動的多重集合に対する Min-hash の高速計算アルゴリズム,“ 電気通信大学情報理工学研究, 2022

[2] Florian Hartmann, “Count-Min Sketch,“ Google Research, 2019