GSLetterNeo vol.133

2019年8月

データの特徴を軸にして可視化するプログラミング

松原 伸人 matubara@sra.co.jp

はじめに

「データの特徴を2軸にしてインタラクティブにデータセットの分布を見る」(GSLetterNeo Vol.131)では、データセットをデータの特徴に基づいて2軸の散布図のようにプロットして見る方法を紹介しました。

Vol.131で紹介した試作している画像の分布を見るツール[図1]での可視化方法を題材に、データの可視化と指定を行うプログラミングを紹介します。

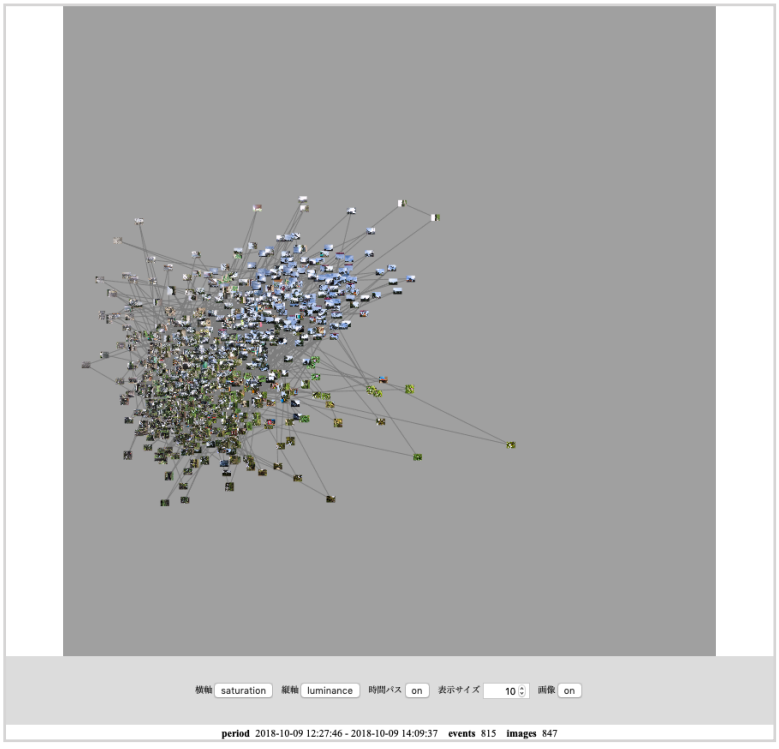

図1 画像の分布を見るツールの画面

本内容は、科学技術振興機構(JST)戦略的創造研究推進事業(CREST)「データ粒子化による高速高精度な次世代マイニング技術の創出」プロジェクト(研究代表者:宇野毅明 教授 (国立情報学研究所))で行なっている研究開発の一部です。今回使用している画像は、筆者自身と、共同研究者の山本恭裕 特任教授 (公立はこだて未来大学)、 および先端技術研究所所長でもある中小路久美代 教授 (公立はこだて未来大学)の協力により得られた研究データの一部です。両者の同意のもと掲載させていただいています。

データの特徴を軸にして可視化するプログラミングでは、大まかに次のことを行います。

- 特徴量の範囲、最小値と最大値を決める
- 特徴量を[0,1]の数値に正規化する
- 表示エリア上のピクセル単位の座標を求める

特徴量の範囲、最小値と最大値を決める

ピクセルをもとにした RGB の場合、ピクセルの最小値0と最大値255を用います。HTML の Canvas を用いる場合、画像のピクセル配列を getImageData() 関数で得られます。

CanvasRenderingContext2D.getImageData() - Web API | MDN

HSL の場合、算出方法により表現方法が異なりますが、例えば Hue は0度から360度の角度で表し、Saturation と Brightness は0%から100%の百分率で表すことが多いです。

パーセント - Wikipedia

HSV色空間 - Wikipedia

作成日など時間情報の場合、データセットの期間が分かっていればそれを用います。期間が分からない場合、データセットの全データから作成日の最初の日と最後の日を最小値と最大値に用います。JavaScript の場合、日付を表す Date オブジェクトの関数 getTime() を用いて、1970年1月1日からの経過時間としてミリ秒で表した数値が得られます。

Date.prototype.getTime() - JavaScript | MDN

緯度や経度の場合、データセットの対象エリアが分かっていればそれを用います。対象エリアが分からない場合、緯度と経度の最小値と最大値を求めて用います。試作ツールでは縦軸と横軸に指定する特徴の種類をあらかじめプログラム内に書いてあります。縦軸と横軸で指定できる特徴の種類は同じで、画像のピクセルの RGB の赤色 (red) 成分、青色 (blue) 成分、緑色 (green) 成分、HSLの色相 (hue) 成分、彩度 (saturation) 成分、輝度 (luminance) 成分、グレースケール換算値 (grayscale)、撮影時刻 (time)、撮影緯度 (latitude) および撮影経度 (longitude) の10種類です。図1は横軸に彩度、縦軸に輝度を指定した様子です。

特徴量を[0,1]の数値に正規化する

各特徴量 v を、求めた最小値 min と最大値 max で、最小値 0 最大値 1 の数値 d に正規化します。

```
d = (v - min) / (max - min)
```

表示エリア上の座標を求める

表示エリアの左下端を横軸と縦軸の原点になるように描画する場合、特徴量 v の横軸上の座標 x は、正規化した数値 d に表示エリアの横幅 width を掛けた値になります。同様に、特徴量 v の縦軸上の座標 y は、正規化した数値 d に表示エリアの縦幅 height を掛けた値になります。HTML の Canvas を描画に用いる場合、Canvas の描画エリアの原点は左上端のため、正規化した数値を逆にして縦幅 height を掛けます。特徴の描画エリアの横方向の位置x、縦方向の位置yとする座標 pv は次のように書けます。

```
let pv = {
    x: d * width
    y: (1 - d) * height
}
```

求めた座標 pv を用いて特徴や画像などのオブジェクトを HTML の Canvas に散布図のように点描するには moveTo 関数と arc 関数や fillRect 関数を用います。

CanvasRenderingContext2D.moveTo() - Web API | MDN

CanvasRenderingContext2D.arc() - Web API | MDN

CanvasRenderingContext2D.fillRect() - Web API | MDN

冒頭の図1のように画像を描画するには drawImage 関数を用います。

CanvasRenderingContext2D.drawImage() - Web APIs | MDN

データを指定するプログラミング

マウスのクリックやタブレットのタッチで画面上に描画したデータを指定する方法を紹介します。クリックやタッチ入力イベントを受け取り、イベントから表示エリアの横方向と縦方向の座標 pi を取得します。座標 pi と上記で計算した特徴の座標 pv から該当するデータを得ます。入力イベントは、HTML では JavaScript を用いて HTML エレメント の関数 addEventListener を使って取得できます。HTML の Canvas 上でマウスのクリックイベントを取得して、入力座標 pi を取得するコードは次のように書けます。

```
let aCanvas = document.getElementById 関数などで取得した対象 Canvas オブジェクトが入ります
aCanvas.addEventListener("click", (event) => {
    let pi = {
        x: event.offsetX,
        y: event.offsetY
    }
})
```

MouseEvent - Web API | MDN

タッチイベントでは、画面に触れた指の本数分の座標が Touch オブジェクトの配列 touches プロパティ

から得られます。指一本で触れることを想定すると、touches の最初の Touch オブジェクトの座標の横方向の位置 clientX プロパティと縦方向の位置 clientY プロパティを参照します。タブレットでのタッチイベントから入力座標 pi を取得するコードは次のように書けます。

```
aCanvas.addEventListener("touchstart", (event) => {
    let touch = event.touches[0]
    let pi = {
        x: touch.clientX,
        y: touch.clientY
    }
})
```

Touch events - Web API | MDN

入力座標 pi が、データを表す形状の中にあるデータを得ます。データの形状には、形状の輪郭を直線でつないで表すポリラインや、形状の代わりに形状の外接矩形や外接円を用いられます。前述した描画方法 arc 関数 で散布図のように丸を描画した場合はデータの形状が円形です。円形の中に入力座標があるかは、入力座標 pi と円の中心座標 pv の距離が、円の半径 radius より短いと円の中にあると分かります。pi と pv の距離はユークリッド距離で、距離の計算と円の中にあるか判定するプログラムはそれぞれ次の distancePoints 関数と containsPoint 関数のように書けます。

```
function distancePoints (pv, pi) {
    return Math.sqrt(Math.pow(pi.x - pv.x, 2) + Math.pow(pi.y - pv.y, 2))
}
function containsPoint (pv, radius, pi) {
    return distancePoints(pv, pi) <= radius
}
```

ユークリッド距離 - Wikipedia

2号に渡りデータの特徴を2軸にしてインタラクティブにデータセットの分布を見る試作ツールと、これに用いているデータの描画と指定のプログラミングについて紹介しました。これらのプログラミングは、基本的な方法の一例ですが同様の方法で色々なデータとの対話的なアプリケーションの実装ができると思います。

images plot

images plot は、vol131 と本記事で紹介した画像群の特徴を軸にして分布を見るツールのプロトタイプです。先端技術研究所の次の URL にアクセスして試用できます。

vial-images-plot

https://www3.sra.co.jp/ktl/deep/vial-images-plot.html

Webブラウザでリンクを開くと「Select data folder」と書かれた画面がでます。

画像ファイル群と、Vol.131 の図2データセットファイルのような内容のファイル拡張子が .md のプレーンテキストファイルが入っているフォルダを選びます。

画像ファイル群の読み込みがはじまります。読み込みが終わると、画像群を縦軸と横軸を grayscale で表示します。

画像データから特徴量を計算するコードや、本記事で書いた正規化のコードは vial-images-plot.html の calculateEventImageFeature 関数に書いてあります。

上記 URL を開いたあと、ブラウザの Web インスペクタを開くと、vial-images-plot.html や ImagesPlot.js など vial-images-plot を実装しているプログラムを見れます。

表示エリア上の座標を求めるコードは、ImagesPlot.js の dataFeatureToX および dataFeatureToY に書いてあります。

データを指定するプログラミングについては、ImagesPlot.js の ImagesPlotController に書いてあります。このコントローラは、画面上の画像の形を円形とみなし、クリックやタップした座標にある画像データを見つけてコンソールに出力します。

画像データには、データセットファイルに書いてあるファイルパスや年月日や経緯度と、特徴量の計算結果が書かれています。

コンソール上で、imagesplot.plotData() と入力すると全画像の特徴量の計算結果を見れます。

GSLetterNeo vol.133
発行日 2019年8月20日
発行者 株式会社 ＳＲＡ 先端技術研究所
編集者 土屋 正人
バックナンバー https://www.sra.co.jp/gsletter/

お問い合わせ
gsneo@sra.co.jp
〒171-8513 東京都豊島区南池袋2-32-8
株式会社SRA
夢を、Yawaraka Innovation