Introduction

This project uses Self-Organizing Map and Restricted Boltzmann Machine to visualize temporal checkout trends of each 2nd-level Dewey class in the Seattle Library Dataset.

Explanation

The query extracts the number of checkouts for each Dewey category from 2010 to 2014 monthly. There are 100 Dewey classes and 60 months in total. Each category is represented by its number of checkouts for each month, which form a 60-dimensional vector. We have 100 such vectors, one for each category.

The visualization uses two methods: SOM and RBM+PCA to visualization the temporal trends among these categories. The main goal is to compare the two methods, and see if we can find something meaningful from the visualizations.

The SOM works by taking each Dewey category's trend as an training vector, and then run the SOM for each category to find the winning neuron where the category is drawn at.

The RBM+PCA takes a different approach, which it builds a model by concatenating a RBM and a PCA. Then uses the trends for the 100 categories as training vectors, and finally run the trained model to find the coordinates for each category.

The third view at the bottom is just the trends themselves.

A simple hover-and-highlight interaction is implemented.

Query

Extract temporal features:

```
SELECT
  COUNT(*) AS count,
  TIMESTAMPDIFF(DAY, "2010-01-01 00:00:00", checkOut) AS day,
  deweyClass AS deweyClass
FROM
  transactions,
  deweyClass
WHERE
  transactions.bibNumber = deweyClass.bibNumber
  AND YEAR(checkOut) >= 2010 AND YEAR(checkOut) <= 2014
  AND deweyClass != ""
GROUP BY
  day, deweyClass
```

Count number of bibs in each Dewey class:

```
SELECT
  COUNT(*) AS count,
  FLOOR(deweyClass / 10) AS deweyClass2
FROM deweyClass
WHERE deweyClass != ""
GROUP BY deweyClass2
```

The queries here are just to extract data for further processing with Python scripts. See the next section for more detail.

How to Run

The python scripts processes the query results into Processing's PDE files. Run them in the following order:

1. `matrix.py` : Collect data from the query result, and write a matrix of temporal features to `matrix.npy` .
2. `matrix_content.py` : Process the temporal trends into `deweyTimeline.pde` .
3. `matrix_som.py` : Train and run the SOM, generate `deweySOM.pde` .
4. `matrix_rbm.py` : Train and run the RBM, generate `deweyRBM.pde` .

Each time you run `matrix_som.py` and `matrix_rbm.py` , you'll get a different visualization.