Can we extract user attributes from Twitter?

Chinese Restaurant Process and its variants

Masaya Sato
Kyushu University

KYUSHU UNIVERSITY

Introduction

Twitter is a social networking and microblogging service that enables the users to post messages of up to 140 characters, also known as *tweets*. Characteristics of Twitter are:

1. Amount of tweet data is accumulated;
2. Transition or variation of user attributes may be time-dependent;
3. Japanese users may want their accounts to be anonymous.

When we want to detect a variety of user attributes (age, gender, favorites, etc.) only from the text messages, we have to create a mathematical model that handles a collection of uncountably many incoming tweet data in order to extract such attributes efficiently.

Question: How do we create such a model?

Answer: Dirichlet process is useful.

Basics in NLP

A *Document-term matrix* is a real-valued matrix that represents the frequency of terms or words in a collection of documents. For example, if we have the following 3 documents

`doc 1` : I bought another MacBook Pro yesterday.
`doc 2` : He bought another MacBook Pro today.
`doc 3` : I buy a new MacBook Pro today.

then the associated document-term matrix is given by

$$\begin{array}{c} \\ \texttt{doc 1} \\ \texttt{doc 2} \\ \texttt{doc 3} \end{array} \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

with columns “I”, “he”, “buy”, “a”, “new”, “another”, “MacBook Pro”, “yesterday” and “today”.

Such a matrix or row vector with integer entries is called `tf`-*weighted* (`tf` for *term frequency*). However, `tf` may incorrectly emphasize the weight on a term because, for example, the indefinite article “a” or “an” is so common in English text documents.

So we define the *inverse document frequency* or `idf` by

$$\texttt{idf} = \log_2 \frac{N}{n_i} + 1,$$

where

- N: total # of documents;
- n_i: # of documents containing the term w_i.

Redefining the weight by `tf*idf` = `tf` $\times$ `idf`, we can measure how important w_i is.

When extracting user attributes efficiently, we may want to partition a collection of data into some groups in such a way that data in the same group are more similar to each other than to those in other groups. Such a task or algorithm is called *clustering* and each group is called a *cluster*.

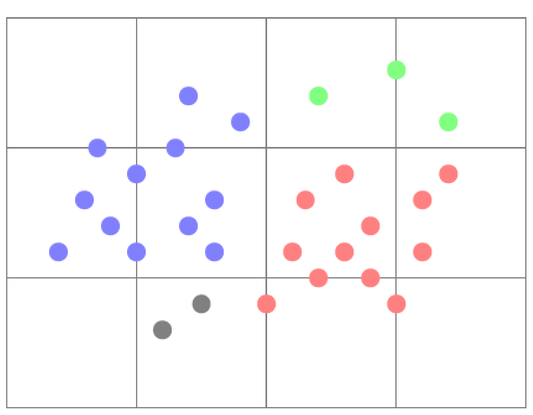


Figure 1: Clustering 2-dimensional data into 4 distinct clusters.

For instance, suppose that the above shows the height and weight of 30 extraterrestrials. Clustering in this context is to group the ETs into subcommunities or clusters based on weight and/or height. Then each cluster may be characterized by a gender and age group.

Modified ddCRP

Distance dependent Chinese Restaurant Process (ddCRP) originally implemented by Blei and Frazier is a variant of CRP, a representation of the *Dirichlet process* that generates infinitely many clusters.

The following are set up for ddCRP:

- K: total # of *tables* generated so far;
- n: total # of *customers* arrived so far;
- α: *concentration parameter* chosen empirically.

Customer x_1 sits at *table* c_1 i.e., $x_1 \in c_1$, where $K = 1$, $n = 1$ and $n_{c_1} = 1$. **for** $i = 2, \ldots,$ *customer* x_i sits at *table*

$$\begin{cases} c_k & \text{with probability } \propto f(d) \text{ for } k = 1, \ldots, K, \\ c_{K+1} & \text{with probability } \propto \alpha \text{ otherwise} \end{cases} \tag{1}$$

with $d = \texttt{distance function}$ and $f = \texttt{decay function}$
if c_{K+1} generated **then** set $K \leftarrow K + 1$
endif
set $x_i \in c_k$, $n_{c_k} \leftarrow n_{c_k} + 1$
endfor

Here `distance function` measures difference between the indices of incoming data and each table, and `decay function` is chosen to be an exponential or logistic decay function.

In our setting, clustering is achieved by first considering time stamps and similarities among incoming data. Let x_i be data input at time τ_i. Then

$$d(x_i) = d(x_i; x_j) := |\tau_i - \tau_j| \quad (j \le i)$$

with x_j the first data that generates a new table. For every x_i in c_k

$$sim^{c_k}_{\texttt{tf*idf}}(x_i) := \begin{cases} \max\{\cos(x_j, x_i)\} & \text{if } j \le i, \\ \infty & \text{otherwise,} \end{cases}$$

where $\cos(\cdot, \cdot)$ denotes the cosine between x_j and x_i. Then replace (1) with (2):

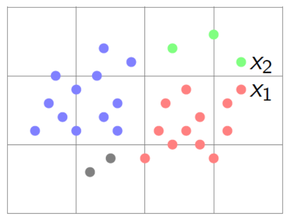
$$\begin{cases} c_k & \text{with probability } \propto sim \cdot f(d) \text{ for } k = 1, \ldots, K, \\ c_{K+1} & \text{with probability } \propto \alpha \text{ otherwise.} \end{cases} \tag{2}$$

Semisupervised Learning

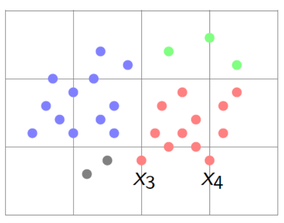
Since

1. Chinese Restaurant Process is *unsupervised learning*;
2. `tf*idf` may or may not be the desired weight,

we want to extend the model using *semisupervised learning* or *metric learning* so that we can “customize” our clustering. I.e., we will consider the *must-link* set $\mathcal{M}$ and *cannot-link* set $\mathcal{C}$ as our *training data* and determine another weight.



x_1 and x_2 look similar
$\Downarrow$
must be in the same cluster
$\Downarrow$
$(x_1, x_2) \in \mathcal{M}$



x_3 and x_3 look dissimilar
$\Downarrow$
cannot be in the same cluster
$\Downarrow$
$(x_3, x_4) \in \mathcal{C}$

Then we solve the following optimization problem for A diagonal:

Let f be logistic decay function and `tf`-weighted data x with time stamp τ. Then

$$\texttt{maximize} \quad \sum_{(x_i, x_j) \in \mathcal{M}} \cos_A(x_i, x_j) \cdot f(d(x_i; x_j))$$

$$\texttt{subject to}$$

$$\sum_{(x_m, x_n) \in \mathcal{C}} \cos_A(x_m, x_n) \cdot f(d(x_m; x_n)) \le \#\mathcal{C} \cdot \alpha$$

and

$$A \succeq 0,$$

where $\cos_A(x_i, x_j)$ is the cosine between A-weighted data x_i and x_j.

This optimization problem is called *convex* and can be solved by the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, the most popular quasi-Newton method that does not require the Hessian.

Data Set & Accuracy

Data set: 405 tweets with time stamps by single user, 1 tweet counted as 1 document.

Document-term matrix $\cdots$ 405 $\times$ 1142 matrix consisting of only nouns, emoticons excluded.

Define the accuracy of clustering as follows: for randomly chosen 100 pairs of data (x_i, x_j) with $i < j$

$$\texttt{accu pts} = \sum_{i<j} \chi(\chi((x_i, x_j) \in c \times c) = \chi((x_i, x_j) \in \hat{c} \times \hat{c})),$$

where

- χ is a characteristic function with values $\chi(\texttt{true}) = 1$ and $\chi(\texttt{false}) = 0$;
- c = cluster generated by ddCRP;
- $\hat{c}$ = *desired* cluster.

Results

1. Original ddCRP by Blei and Frazier (# of clusters = 260)

Words characterizing largest 3 clusters:

c_1: “娘”, “熱”

c_2: “東方神起”, “インコ”

c_3: “義母

`accu pts = 48`

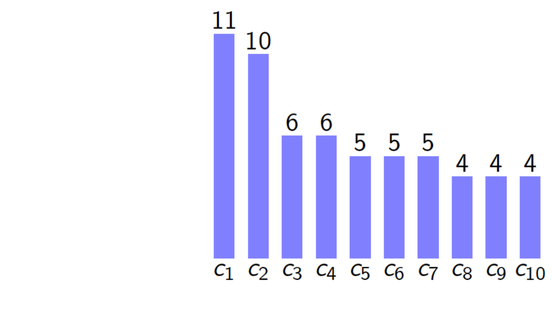


Figure 2: Original ddCRP

2. Modified ddCRP (# of clusters = 101)

Words characterizing largest 3 clusters:

c'_1: “娘”, “熱”

c'_2: “何”

c'_3: “今日”, “娘”

`accu pts = 71`

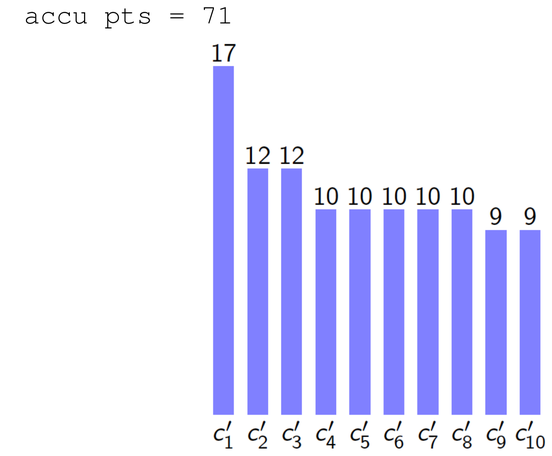


Figure 3: Modified ddCRP

3. Semisupervised ddCRP (# of clusters = 107)

Words characterizing largest 3 clusters:

c''_1: “インコ”

c''_2: “義母”, “熱”

c''_3: “娘”, “熱”

`accu pts = 68`

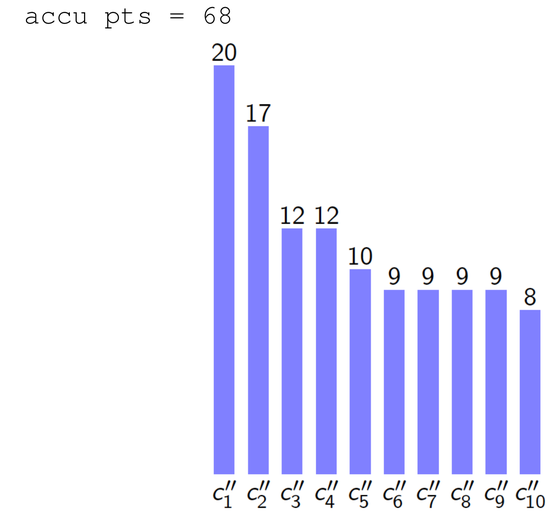


Figure 4: Semisupervised ddCRP

Conclusion

1. Modified ddCRP improved original ddCRP.
2. Semisupervised version of ddCRP may extend modified ddCRP but optimization problem needs to be considered carefully.
3. Or need more training data for metric learning?
4. What about large amount of tweet data of multiple users?

References

1. Basu et al. *A Probabilistic Framework for Semi- Supervised Clustering*, Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004.
2. Blei et al. *Distance Dependent Chinese Restaurant Process*, Journal of Machine Learning Research, 12:2461-2488, 2011.
3. Xing et al. *Metric Learning, with Application to Clustering with Side-information*, Advances in Neural Information Processing Systems 15, 2002.