CS 6501 Natural Language Processing

Unsupervised Learning (I): Clustering Algorithms

Yangfeng Ji

October 29, 2019

Department of Computer Science
University of Virginia

UNIVERSITY of VIRGINIA | ENGINEERING

Overview

1. Introduction
2. K-means Algorithm
3. Brown Clustering Algorithm

Introduction

Supervised vs. Unsupervised Learning

- Supervised learning: learning with supervision

$$f : \mathcal{X} \rightarrow \mathcal{Y} \tag{1}$$

  Example: text classification, POS tagging, named entity recognition, syntactic parsing

Supervised vs. Unsupervised Learning

- Supervised learning: learning with supervision

$$f : \mathcal{X} \rightarrow \mathcal{Y} \tag{1}$$

  Example: text classification, POS tagging, named entity recognition, syntactic parsing
- Unsupervised learning: learning without supervision
  - Dimension reduction

$$f : \mathcal{X} \rightarrow \mathcal{Z} \in \mathbb{R}^{n} \tag{2}$$

    Example: latent semantic analysis (lecture 12)

Supervised vs. Unsupervised Learning

- Supervised learning: learning with supervision

$$f : \mathcal{X} \rightarrow \mathcal{Y} \tag{1}$$

  Example: text classification, POS tagging, named entity recognition, syntactic parsing
- Unsupervised learning: learning without supervision
  - Dimension reduction

$$f : \mathcal{X} \rightarrow \mathcal{Z} \in \mathbb{R}^n \tag{2}$$

    Example: latent semantic analysis (lecture 12)
  - Clustering

$$f : \mathcal{X} \rightarrow \mathcal{C} \in \mathbb{N} \tag{3}$$

    where $\mathcal{C}$ is a set of discrete values

Clustering

Clustering is the task of grouping a set of objects such that

- similar objects end up in the same group
- dissimilar objects are separated into different groups

Clustering

Clustering is the task of grouping a set of objects such that

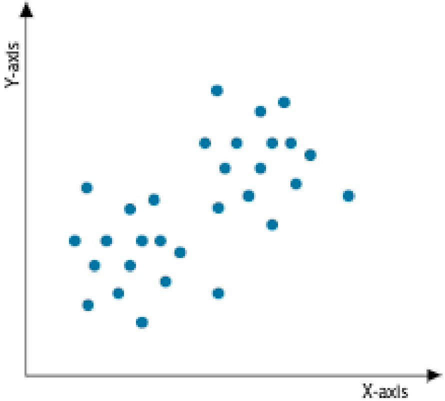
- similar objects end up in the same group
- dissimilar objects are separated into different groups

Common setup

- Input: a set of examples $\{x_i\}_{i=1}^n$
- Output: a partition of $\{x_i\}_{i=1}^n$, where each x_i has an associated cluster index $c_i \in \mathscr{C}$

Benefits

An important technique for exploratory data analysis: finding patterns from data



Clustering Algorithms

Two example clustering algorithms used in NLP:

- k-means clustering algorithm
- The Brown clustering algorithm
  - A special case of hierarchical clustering algorithms

K-means Algorithm

Objective Function

Given n examples $\{x_i\}_{i=1}^n$. For each i,

- $\boldsymbol{x}_i$ is the corresponding numeric representation of x_i
- z_i is the cluster index of x_i

Objective Function

Given n examples $\{x_i\}_{i=1}^{n}$. For each i,

- $\boldsymbol{x}_i$ is the corresponding numeric representation of x_i
- z_i is the cluster index of x_i

The objective function

$$\min_{\{\boldsymbol{v}_k\}} \sum_{k=1}^{K} \sum_{i=1}^{n} \delta(z_i, k) \cdot \|\boldsymbol{x}_i - \boldsymbol{v}_k\|_2^2 \tag{4}$$

where

- z_i is the cluster index of x_i
- $\delta(z_i, k) = 1$ if $z_i = k$; otherwise 0

Objective Function

Given n examples $\{x_i\}_{i=1}^n$. For each i,

- $\boldsymbol{x}_i$ is the corresponding numeric representation of x_i
- z_i is the cluster index of x_i

The objective function

$$\min_{\{\boldsymbol{v}_k\}} \sum_{k=1}^{K} \sum_{i=1}^{n} \delta(z_i, k) \cdot \|\boldsymbol{x}_i - \boldsymbol{v}_k\|_2^2 \tag{4}$$

where

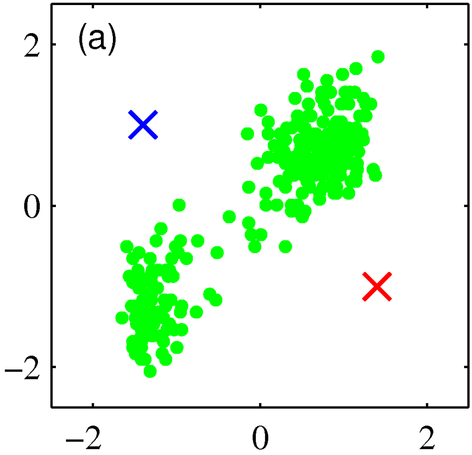
- z_i is the cluster index of x_i
- $\delta(z_i, k) = 1$ if $z_i = k$; otherwise 0
- $\boldsymbol{v}_k$ is the mean of the k-th cluster

$$\boldsymbol{v}_k = \frac{1}{\sum_{i=1}^n \delta(z_i, k)} \sum_{i=1}^{n} \delta(z_i, k)\boldsymbol{x}_i \tag{5}$$

which is dependent on $\{z_i\}$

K-means Algorithm: Initialization

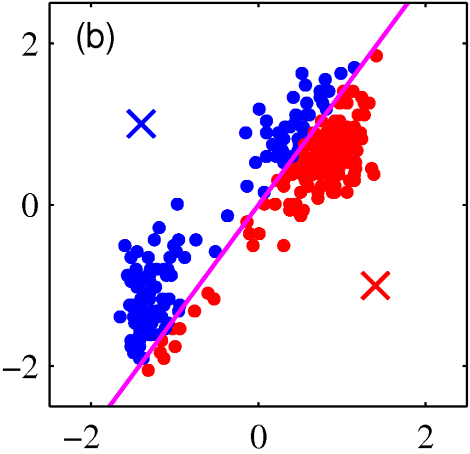
Initialization: Randomly initialize $\{v_k\}_{k=1}^{K}$



K-means Algorithm: Data partition

Step 1: For each example x_i, update its cluster index as

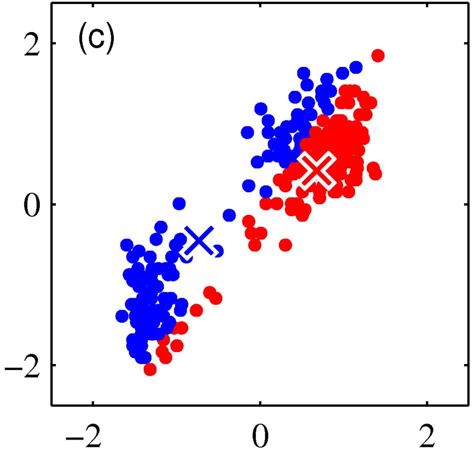
$$z_i \leftarrow \underset{k}{\operatorname{argmin}} \|\boldsymbol{x}_i - \boldsymbol{v}_k\|_2 \tag{6}$$



K-means Algorithm: Means update

Step 2: For each cluster k, update its mean as

$$v_k \leftarrow \frac{1}{\sum_{i=1}^{n} \delta(z_i, k)} \sum_{i=1}^{n} \delta(z_i, k) \boldsymbol{x}_i \tag{7}$$



Clustering Algorithm

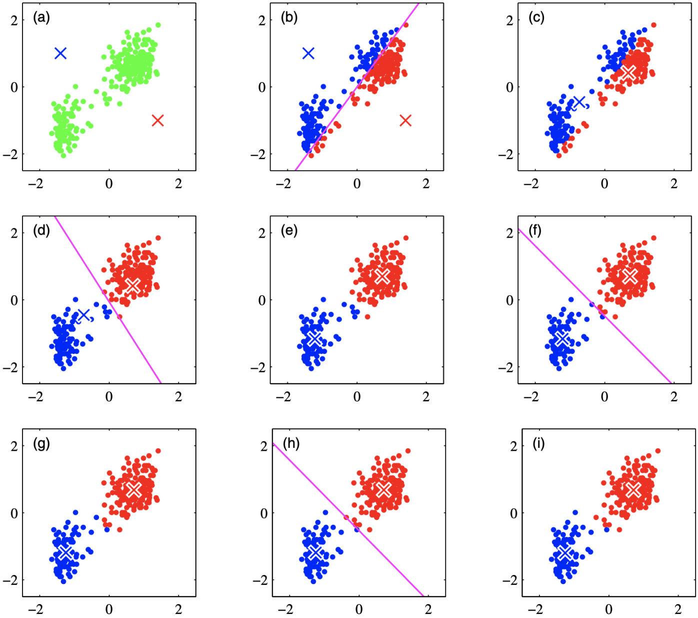
- Randomly initialize $\{\boldsymbol{v}_k\}_{k=1}^{K}$
- Repeat the following two steps, until converged
  1. For each example, update its cluster index as

$$z_i \leftarrow \operatorname*{argmin}_k \|\boldsymbol{x}_i - \boldsymbol{v}_k\|_2 \tag{8}$$

  2. For each cluster, update its mean as

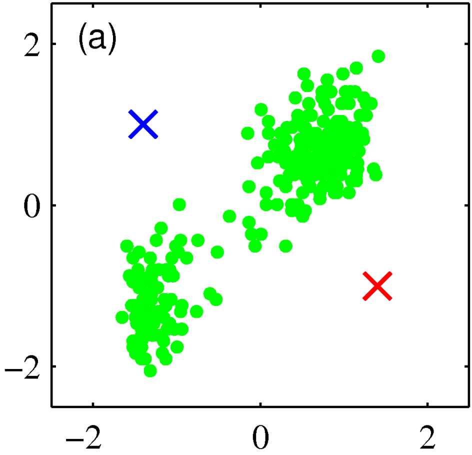
$$\boldsymbol{v}_k \leftarrow \frac{1}{\sum_{i=1}^{n} \delta(z_i, k)} \sum_{i=1}^{n} \delta(z_i, k)\boldsymbol{x}_i \tag{9}$$

Example: Clustering



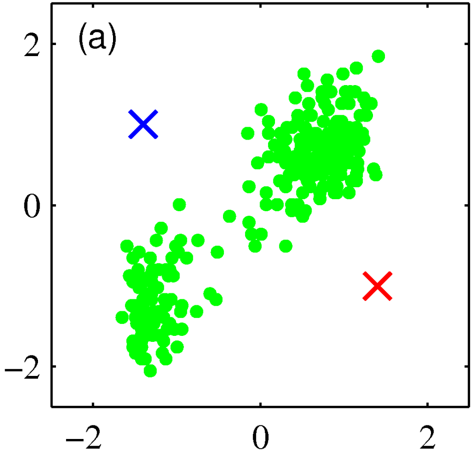
Issues of K-means Clustering

Issue 1: how to choose the value of K (number of clusters)?



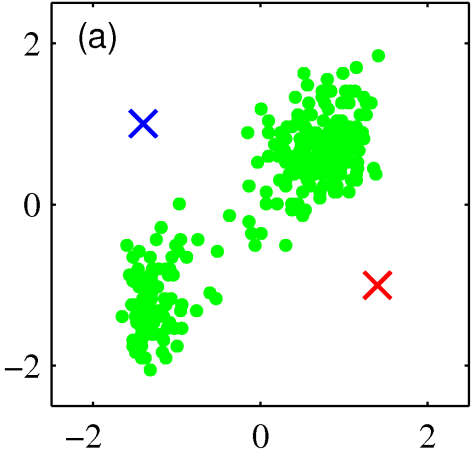
Issues of K-means Clustering

Issue 2: sensitivity of initialization



Issues of K-means Clustering

Issue 2: sensitivity of initialization



K-means can only find local optima

Brown Clustering Algorithm

Problem Setup

Class-Based *n*-gram Models of Natural Language

Peter F. Brown*
Peter V. deSouza*
Robert L. Mercer*
IBM T. J. Watson Research Center

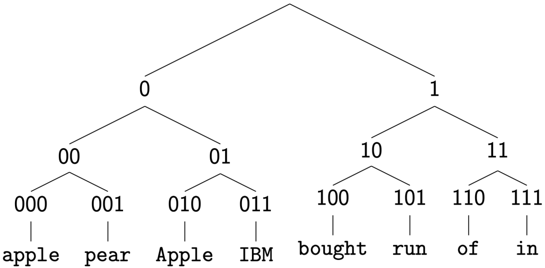
Vincent J. Della Pietra*
Jenifer C. Lai*

- Input: a large corpus of words
- Output:
  - a hierarchical word clusters
  - also, a partition of words into clusters

[Brown et al., 1992]

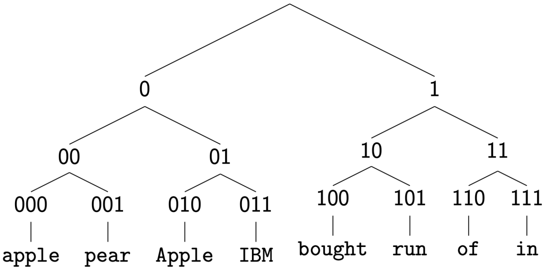
Example: Hierarchical clusters

A hierarchical word clusters



Example: Hierarchical clusters

A hierarchical word clusters



A partition of words into clusters by getting the cluster index from the prefix of a code

00	apple, pear
01	Apple, IBM
10	bought, run
11	of, it

The Formulation

- Intuition: similar words appear in similar contexts

The Formulation

- Intuition: similar words appear in similar contexts
- $\mathscr{V}$ is the set of all words seen in the corpus
- $\mathscr{C} : \mathscr{V} \to \{1, 2, \ldots, K\}$ is a partition of the vocabulary into K classes

The Formulation

- Intuition: similar words appear in similar contexts
- $\mathscr{V}$ is the set of all words seen in the corpus
- $\mathscr{C} : \mathscr{V} \rightarrow \{1, 2, \ldots, K\}$ is a partition of the vocabulary into K classes
- Given a text consisting of word sequence $\{w_1, \ldots, w_n\}$, the probabilistic model:

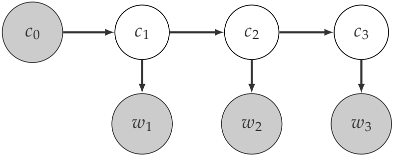
$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} \{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \tag{10}$$

where $c_i \in \mathscr{C}$ is the cluster index of word w_i, c_0 is the special starting state

The Formulation (II)

Represent it as a HMM:

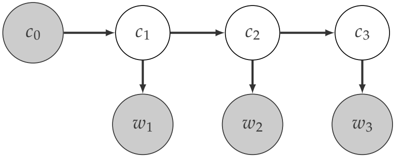
$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} \{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \tag{11}$$



The Formulation (II)

Represent it as a HMM:

$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} \{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \tag{11}$$

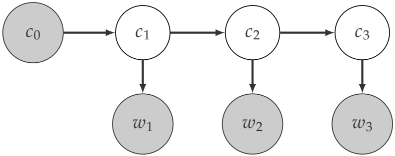


- $p(c_i \mid c_{i-1})$: transition probability
- $p(w_i \mid c_i)$: emission probability

The Formulation (II)

Represent it as a HMM:

$$p(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} \{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \tag{11}$$



- $p(c_i \mid c_{i-1})$: transition probability
- $p(w_i \mid c_i)$: emission probability
- c_i is unobserved in training examples
  - different from the (supervised) POS tagging in lecture 6

Measuring the Quality of $\mathscr{C}$

How to measure the quality of a partition $\mathscr{C}$?

$$\begin{aligned}
\text{Quality}(\mathscr{C}) &= \sum_{i=1}^{n} \log\{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \\
&= \sum_{c=1}^{k} \sum_{c'=1}^{k} p(c, c') \log \frac{p(c, c')}{p(c)p(c')} + \text{Constant} \qquad (12)
\end{aligned}$$

[Collins, 2017]

Measuring the Quality of $\mathscr{C}$

How to measure the quality of a partition $\mathscr{C}$?

$$\begin{aligned} \text{Quality}(\mathscr{C}) &= \sum_{i=1}^{n} \log\{p(w_i \mid c_i) \cdot p(c_i \mid c_{i-1})\} \\ &= \sum_{c=1}^{k} \sum_{c'=1}^{k} p(c, c') \log \frac{p(c, c')}{p(c)p(c')} + \text{Constant} \end{aligned} \tag{12}$$

where

$$p(c, c') = \frac{\#(c, c')}{\sum_{c,c'} \#(c, c')} \quad p(c) = \frac{\#(c)}{\sum_{c} \#(c)} \tag{13}$$

and $\#(c, c')$ is the number of times that c' is following c.

[Collins, 2017]

Optimization

- Initialization: $|\mathcal{V}|$ clusters — each word get its own cluster
- Output: K final clusters

Optimization

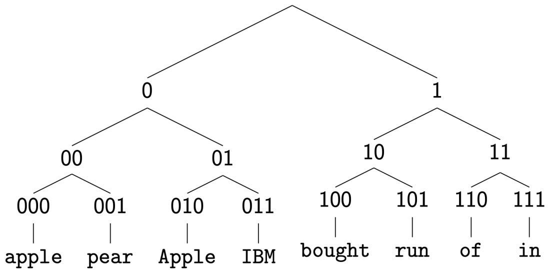
- Initialization: $|\mathcal{V}|$ clusters — each word get its own cluster
- Output: K final clusters
- For $t = 1 : (|\mathcal{V}| - K)$ steps
  1. Pick any two clusters c_i and c_j, merge them into a single cluster, and compute
  $$\text{Quality}(\mathscr{C}) \tag{14}$$
  of the newly formed clusters

Optimization

- Initialization: $|\mathscr{V}|$ clusters — each word get its own cluster
- Output: K final clusters
- For $t = 1 : (|\mathscr{V}| - K)$ steps
  1. Pick any two clusters c_i and c_j, merge them into a single cluster, and compute
  $$\text{Quality}(\mathscr{C}) \tag{14}$$
  of the newly formed clusters
  2. Greedily pick merges such that Quality($\mathscr{C}$) is maximized after the merge step

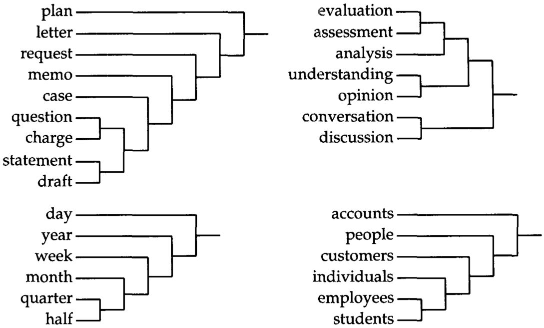
Please refer to [Liang, 2005] for more efficient algorithms

The Toy Example



- A hierarchical word clusters
- A partition of words into clusters

Examples



[Brown et al., 1992]

Applications

Example applications of the Brown clusters

- Language modeling
- Part-of-Speech tagging
- Named entity recognition
- Dependency parsing
- ...

Language Modeling

With a pre-defined Brown clusters

$$p(w_i \mid w_{i-1}) = \underbrace{p(c_{i-1} \mid w_{i-1})}_{=1} \cdot p(c_i \mid c_{i-1}) \cdot p(w_i \mid c_i) \tag{15}$$

With a pre-defined Brown clusters

$$p(w_i \mid w_{i-1}) = \underbrace{p(c_{i-1} \mid w_{i-1})}_{=1} \cdot p(c_i \mid c_{i-1}) \cdot p(w_i \mid c_i) \tag{15}$$

- From the original work of the Brown clustering [Brown et al., 1992]
- Less parameters

Language Modeling

With a pre-defined Brown clusters

$$p(w_i \mid w_{i-1}) = \underbrace{p(c_{i-1} \mid w_{i-1})}_{=1} \cdot p(c_i \mid c_{i-1}) \cdot p(w_i \mid c_i) \tag{15}$$

- From the original work of the Brown clustering [Brown et al., 1992]
- Less parameters
- Easier prediction in each step
- Same idea used in neural LM [Baltescu and Blunsom, 2014]

Part-of-Speech Tagging

	Binary path	Top words (by frequency)
A1	111010100010	lmao lmfao lmaoo lmaooo hahahahaha lool ctfu rofl loool lmfaoo lmfaooo lmaoooo lmbo **lololol**
A2	111010100011	haha hahaha hehe hahahaha hahah aha hehehe ahaha hah hahahah kk hahaa ahah
A3	111010100100	yes yep yup nope yess yesss yessss ofcourse yeap likewise yepp yesh yw yuup yus
A4	111010100101	yeah yea nah naw yeahh nooo yeh noo noooo yeaa **ikr** nvm yeahhh nahh nooooo
A5	11101011011100	**smh** jk #fail #random #fact smfh #smh #winning #realtalk smdh #dead #justsaying
B	011101011	**u** yu yuh yhu uu yuu yew y0u yuhh youh yhuu iget yoy yooh yuo ü yue juu Ʊ dya youz yyou
C	11100101111001	w fo fa fr fro ov fer **fir** whit abou aft serie fore fah fuh w/her w/that fron isn agains
D	111101011000	facebook **fb** itunes myspace skype ebay tumblr bbm flickr aim msn netflix pandora
E1	0011001	tryna gon finna bouta trynna boutta gne fina gonn tryina fenna qone trynaa qon
E2	0011000	gonna gunna gona gna guna gnna ganna qonna gonnna gana qunna gonne goona
F	0110110111	soo sooo soooo sooooo soooooo sooooooo soooooooo sooooooooo soooooooooo
G1	11101011001010	;) :p :-) xd ;-) ;d (; :3 ;p =p :-p =)) ;] xdd #gno xddd >:) ;-p >:d 8-) ;-d
G2	11101011001011	:) (: =) :)) :] ☺ :') =] ^_^ :))) ^.^ [: ;)) 😊 ((: ^__^ (= ^-^ :))))
G3	1110101100111	:(:/ -_- -.- :-(:'(d: :| :s -__- =(=/ >.< -___- :-/ </3 :\ -____- ;(/: :((>_< =[:[#fml
G4	111010110001	<3 ♥ xoxo <33 xo <333 ❤ ♡ #love s2 <URL-twitition.com> #neversaynever <3333

[Owoputi et al., 2013]

Named Entity Recognition

Nike	1011011100100101011100
Maytag	10110111001001010111010
Generali	10110111001001010111011
Gap	1011011100100101011110
Harley-Davidson	10110111001001010111110
Enfield	101101110010010101111110
genus	101101110010010101111111
Microsoft	10110111001001011000
Ventritex	101101110010010110010
Tractebel	1011011100100101100110
Synopsys	1011011100100101100111
WordPerfect	1011011100100101101000

John	101110010000000000
Consuelo	101110010000000001
Jeffrey	101110010000000010
Kenneth	10111001000000001100
Phillip	101110010000000011010
WILLIAM	101110010000000011011
Timothy	10111001000000001110
Terrence	101110010000000011110
Jerald	101110010000000011111
Harold	101110010000000100
Frederic	101110010000000101
Wendell	10111001000000011

[Miller et al., 2004]

Named Entity Recognition

Nike	1011011100100101011100
Maytag	10110111001001010111010
Generali	10110111001001010111011
Gap	1011011100100101011110
Harley-Davidson	10110111001001010111110
Enfield	101101110010010101111110
genus	101101110010010101111111
Microsoft	10110111001001011000
Ventritex	101101110010010110010
Tractebel	1011011100100101100110
Synopsys	1011011100100101100111
WordPerfect	1011011100100101101000

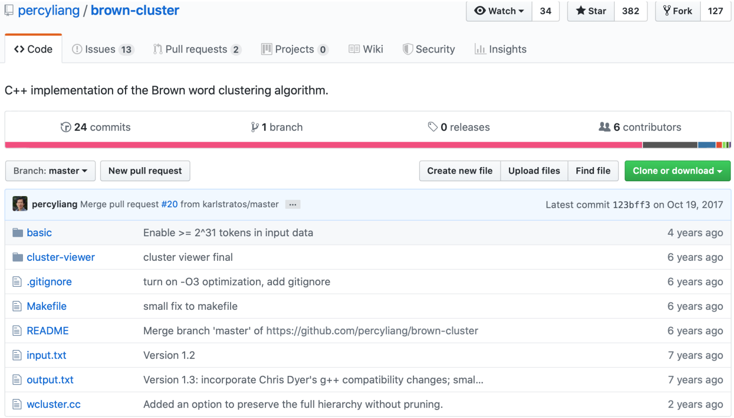
John	1011100100000000000
Consuelo	1011100100000000001
Jeffrey	1011100100000000010
Kenneth	101110010000000001100
Phillip	1011100100000000011010
WILLIAM	1011100100000000011011
Timothy	101110010000000001110
Terrence	1011100100000000011110
Jerald	1011100100000000011111
Harold	1011100100000000100
Frederic	1011100100000000101
Wendell	101110010000000011

Feature template with $n = 8, 12, 16, 20$

POS-TAG + FIRST-n-PREFIX-CODE

[Miller et al., 2004]

Implementation



Implementation (II)

```
➜  brown-cluster git:(master) ✗ ./wcluster
usage: ./wcluster
  chk                 : Check data structures are valid (expensive). [false]
  stats               : Just print out stats. [false]
  paths2map           : Take the paths file and generate a map file. [false]
  no_prune            : Do not prune the hierarchy (show all N leaf clusters) [false]
  ncollocs      <int> : Collocations with most mutual information (output). [500]
  c             <int> : Number of clusters. [1000]
  plen          <int> : Maximum length of a phrase to consider. [1]
  min-occur     <int> : Keep phrases that occur at least this many times. [1]
  rand          <int> : Number to call srand with. [1305448748]
  threads       <int> : Number of threads to use in the worker pool. [1]
  max-ind-level <int> : Maximum indent level for logging [3]
  ms-per-line   <int> : Print a line out every this many milliseconds [0]
  output_dir    <str> : Output everything to this directory. []
  text          <str> : Text file with corpora (input). []
  restrict      <str> : Only consider words that appear in this text (input). []
  paths         <str> : File containing root-to-node paths in the clustering tree (input/output). []
  map           <str> : File containing lots of good information about each phrase, more general than paths (output) []
  collocs       <str> : Collocations with most mutual information (output). []
  featvec       <str> : Feature vectors (output). []
  comment       <str> : Description of this run. []
  log           <str> : File to write log to ("" for stdout) []
```

Reference

Baltescu, P. and Blunsom, P. (2014).
Pragmatic neural language modelling in machine translation.
arXiv preprint arXiv:1412.7119.

Brown, P. F., Desouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992).
Class-based n-gram models of natural language.
Computational linguistics, 18(4):467–479.

Collins, M. (2017).
Natural language processing: Lecture notes.

Liang, P. (2005).
Semi-supervised learning for natural language.
PhD thesis, Massachusetts Institute of Technology.

Miller, S., Guinness, J., and Zamanian, A. (2004).
Name tagging with word clusters and discriminative training.
In *NAACL*.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013).
Improved part-of-speech tagging for online conversational text with word clusters.
In *NAACL*.