Project 4 - Example Main Script

Team 11

04/14/2017

# Summary:

#### In this project, we implemented, evaluated and compared the algorithms in paper 1 : Information Processing and Management (Kang 2009), and paper 5: Author Disambiguation using Error-driven Machine Learning with a Ranking Loss Function(Culotta 2007) for Entity Resolution. We created an author disambiguation system that divides the same-name author occurrences in citation data into different clusters, each of which are expected to correspond to a real individual. We used hierarchical clustering for both papers. In addition, we implemented Cluster Scoring Function, Error-driven Online Training, and Ranking MIRA. After comparing these two methods, we find out that for large dataset, paper 5 performed better than Paper1 did. For instance, algorithm in paper 5 offered much better results than paper 1 when tackling datasets of which author names are 'JLee', 'JSmith', 'SLee' and 'YChen' with the number of observations 1419, 927, 1464 and 1265 respectively.

## Step 0: Load the packages, specify directories

if (!require("pacman")) install.packages("pacman")

## Loading required package: pacman

pacman::p\_load(text2vec, plyr,qlcMatrix, kernlab, knitr)  
setwd("~/Spr2017-proj4-team-11/doc")  
library(ggplot2)

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:kernlab':  
##   
## alpha

## Step 1: Load and process the data

For each record in the dataset, there are some information we want to extract and store them in a regular form: canonical author id, coauthors, paper title, publication venue title.

After generated a list of 14 elements using Professor Zheng's code, we reorganizes it into a list of 14 dataframes for easier access and processing.

source("../lib/dataclean.R")

## Step 2: Feature design

### Paper 1 : Following the section 5.2, that each name occurrence is represented by a set of his/her coauthor names. We count the number of matched coauthors between two authors.

### Paper 5: We want to use coauthors, paper titles and journey titles to design features for citations.

#### • TF-IDF and cosine similarity: Term Frequency/Inverse Document Frequency weighting

#### • Same-Coauthor occurrences

#### • Edit distance: Compute the approximate string distance between character vectors. The distance is a generalized Levenshtein (edit) distance, giving the minimal possibly weighted number of insertions, deletions and substitutions needed to transform one string into another

#### • Bigram and Trigram: Count the Frequency of Pairs/Triple characters

#### • Journey Title Similarity

#### •

#### •

# source("../lib/coauthormatrix.R")  
load("../data/sim\_matrix.RData")

## Step 3: Clustering

We used a hierarchical clustering method for both paper 1 and paper 5. The algorithm also follows section 5.2 in paper 1.

We set the number of overlapping coauthors to 1.

source("../lib/singlelink.R")  
start.time <- Sys.time()  
cluster\_temp.list <- NULL  
cluster\_temp.list <- llply(simmatrix.list,singlecluster,theta=1)  
# load("../doc/cluster\_temp\_1.RData")  
cluster.combined <- NULL  
cluster.combined <- llply(cluster\_temp.list,comninecluster)  
cluster.notcombined <- NULL  
cluster.notcombined <- llply(cluster\_temp.list,splitcluster)  
end.time <- Sys.time()  
time\_scluter <- end.time - start.time  
time\_scluter

## Time difference of 13.5362 mins

We also considered two scenarios : all the single-element cluster are combined; or we don't combine them. Here, I only showed the cluster result of a subset of the data, "AGupta.txt" to further illustrate the difference of these two scenarios.

# combind cluster table for AGupta  
table(cluster.combined[[1]])

##   
## 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17   
## 62 114 58 24 24 24 23 22 19 16 15 14 13 12 11 9 7 7   
## 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35   
## 7 6 5 5 4 4 4 4 3 3 3 3 3 3 3 3 2 2   
## 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53   
## 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

# do not combine single-element cluster   
table(cluster.notcombined[[1]])

##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18   
## 114 58 24 24 24 23 22 19 16 15 14 13 12 11 9 7 7 7   
## 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36   
## 6 5 5 4 4 4 4 3 3 3 3 3 3 3 3 2 2 2   
## 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54   
## 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1   
## 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72   
## 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90   
## 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108   
## 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1   
## 109 110 111 112 113 114 115   
## 1 1 1 1 1 1 1

### Procedure:

#### • An initial guess, , assigns weights to each feature. We used odds ratio as our initial guess.

#### • Error-driven Trainging: We use hierarchical clustering. After first iteration, we have a partition T hat. We calculate a score S and a true socre(accuracy) for our partition. T star is a partition with higher true score in the neigborhood of T hat, and the existence of T star means that T hat is not the best partition and therefore we need to update .

#### • Ranking MIRA: We update with three constratins. The optimization is solved using Python through the quadratic program.

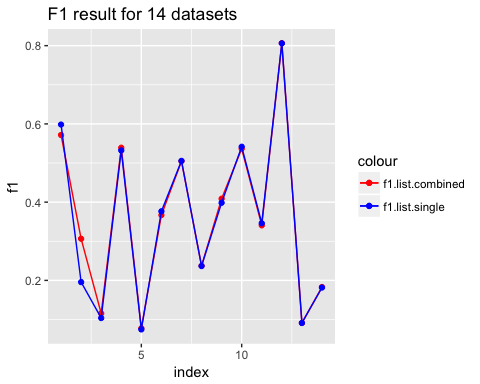
#### • After iterations and updates, we have our final

## Step 4: Evaluation

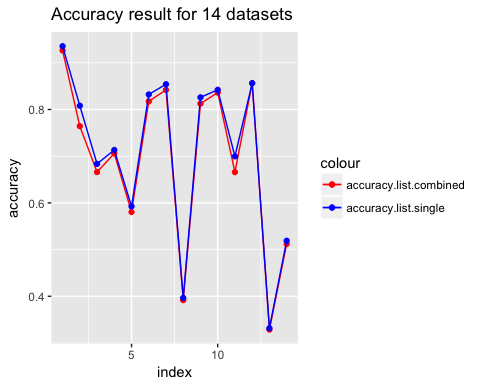
To evaluate the performance of the method, it is required to calculate the degree of agreement between a set of system-output partitions and a set of true partitions. In general, the agreement between two partitioins is measured for a pair of entities within partitions. The basic unit for which pair-wise agreement is assessed is a pair of entities (authors in our case) which belongs to one of the four cells in the following table (Kang et at.(2009)):

Let be the set of machine-generated clusters, and the set of gold standard clusters. Then. in the table, for example, is the number of pairs of entities that are assigned to the same cluster in each of and . Hence, and are interpreted as agreements, and and disagreements. When the table is considered as a confusion matrix for a two-class prediction problem, the standard "Precision", "Recall","F1", and "Accuracy" are defined as follows.

source("../lib/evaluation\_measures.R")  
  
#### paper 1   
matching\_matrix\_single <- NULL  
matching\_matrix\_combined <- NULL  
for (i in 1:14){  
 matching\_matrix\_single[[i]] <- matching\_matrix(data[[i]],cluster.notcombined[[i]])  
 matching\_matrix\_combined[[i]] <- matching\_matrix(data[[i]],cluster.combined[[i]])  
}  
  
f1.list.single <- NULL  
accuracy.list.single <- NULL  
f1.list.combined <- NULL  
accuracy.list.combined <- NULL  
clustering\_errors\_single <- NULL  
clustering\_errors\_combined <- NULL  
for (i in 1:14){  
 f1.list.single[i] <- performance\_statistics(matching\_matrix\_single[[i]])$f1  
 f1.list.combined[i] <- performance\_statistics(matching\_matrix\_combined[[i]])$f1  
 accuracy.list.single[i] <- performance\_statistics(matching\_matrix\_single[[i]])$accuracy  
 accuracy.list.combined[i] <- performance\_statistics(matching\_matrix\_combined[[i]])$accuracy  
}  
  
  
f1.combined <- as.data.frame(f1.list.combined)  
f1.combined$index <-c(1:14)  
f1.single <- as.data.frame(f1.list.single)  
f1.single$index <- c(1:14)  
acc.combined <- as.data.frame(accuracy.list.combined)  
acc.combined$index <-c(1:14)  
acc.single <- as.data.frame(accuracy.list.single)  
acc.single$index <- c(1:14)  
  
# f1 result  
ggplot()+  
 geom\_point(mapping=aes(x=index,y=f1.list.combined,colour="f1.list.combined"), data=f1.combined)+   
 geom\_line(mapping=aes(x=index,y=f1.list.combined,colour="f1.list.combined"), data=f1.combined)+  
 geom\_point(mapping=aes(x=index,y=f1.list.single,colour="f1.list.single"),  
 data=f1.single)+  
 geom\_line(mapping=aes(x=index,y=f1.list.single, colour="f1.list.single"),  
 data=f1.single)+  
 labs(title="F1 result for 14 datasets",  
 x="index",  
 y="f1")+  
 scale\_color\_manual(values=c(f1.list.combined="red",f1.list.single="blue"))



# Accuracy result   
ggplot()+  
 geom\_point(mapping=aes(x=index,y=accuracy.list.combined,colour="accuracy.list.combined"), data=acc.combined)+   
 geom\_line(mapping=aes(x=index,y=accuracy.list.combined,colour="accuracy.list.combined"), data=acc.combined)+  
 geom\_point(mapping=aes(x=index,y=accuracy.list.single,colour="accuracy.list.single"),  
 data=acc.single)+  
 geom\_line(mapping=aes(x=index,y=accuracy.list.single, colour="accuracy.list.single"),  
 data=acc.single)+  
 labs(title="Accuracy result for 14 datasets",  
 x="index",  
 y="accuracy")+  
 scale\_color\_manual(values=c(accuracy.list.combined="red",accuracy.list.single="blue"))



#### Performance on paper 5,8 and 13 is really bad. So, we use these papers as test sets for algorithm in paper 5. And here are the f1 and accuracy results for these papers.

# Conclusion

#### Compared two methods, method in paper 1 doesn't do well in large dataset. On the other hand, method in paper 5 solves the issue and increases the accuracy.