%% Date: 12 March 2017

% just a simple analysis on NIPS papers

% for MATH project 1

% ===================================================================

%% Summarized the data

% the nips dataset was exported and save in table and array in .mat format

% so just simply load the data

% % load(' nipsET.mat');

% nips(find(isnan(nips.id)),:) = []; %clean the data, remove missing values

dataSize = height(nips);

keywords = cell(1,dataSize);

% clean the data

unwantedStr = {'.', ',', ' ', '\n', '?', '''', '<', '>','"', '\*', '~', ...

'1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '(', ')', ';', '+', ...

'-', '=', ' ', '!', '#', ' ', ' ', '%', '|', '&', '\', '[', ']', ...

'...', '^', '\tab', '$', '/', '{', '}' ':', '\_', '!!!', '!!', ' ',...

'`', '``', '@', '', '', ' ', '', '', '', ''};

% case 1: solve the abstract missing problem

mIndex = find(strcmp(nips.abstract, 'Abstract Missing'));

abstractM = nips.paper\_text(mIndex);

for j=1:length(mIndex)

startIndex = []; endIndex = [];

abstractCell = strsplit(char(abstractM(j)), unwantedStr);

abstToMatch = {'ABSTRACT', 'Abstract', 'abstract'};

k = 1;

while isempty(startIndex)

startIndex = strmatch(abstToMatch(k), abstractCell);

k = k + 1;

if k == 4

startIndex = length(strsplit(char(nips.title(mIndex(j))), ...

unwantedStr)) + 2;

break;

end

end

introToMatch = {'INTRODUCTION', 'Introduction', 'introduction'};

k = 1;

while isempty(endIndex)

endIndex = strmatch(introToMatch(k), abstractCell);

k = k + 1;

if k == 4

endIndex = startIndex +200;

break;

end

end

tb = strcat(nips.title(mIndex(j)), {' '}, ...

strjoin(abstractCell(startIndex+1:endIndex-1)));

% nips.abstract(mIndex(j)) = strjoin(abstractCell(startIndex+1:endIndex-1));

keywords(mIndex(j)) = {strsplit(lower(char(tb)), unwantedStr)'};

end

% case 2: when abstract is available: join title and abstract

nIndex = find(~strcmp(nips.abstract, 'Abstract Missing'));

ta = strcat(nips.title(nIndex), {' '}, nips.abstract(nIndex));

for i=1:length(nIndex)

keywords(:,nIndex(i)) = {strsplit(lower(char(ta(i))), unwantedStr)'};

end

% convert to categorical format

id = categorical(nips.id);

year = categorical(nips.year);

title = categorical(nips.title);

abstract = categorical(nips.abstract);

% get the unique keyword (feature) from all papers.

keyTable = table();

keyCell = cell(1,dataSize);

countCell = cell(1,dataSize);

for i = 1:length(keywords)

key = categorical(keywords{i});

key\_data = categories(key);

count\_data = countcats(key);

% keyI = strcat('k', num2str(i));

% assignin('base',keyI,key\_data);

% countI = strcat('c', num2str(i));

% assignin('base',countI,count\_data);

keyCell(i) = {key\_data};

countCell(i) = {count\_data};

end

tempKey = union(keyCell{1}, keyCell{2});

for i = 3:length(keyCell)

tempKey = union(tempKey, keyCell{i});

end

keyTable.key = tempKey;

% map the value to the key for each paper

cValue = [];

for i = 1:length(keyCell)

cTemp = double(ismember(keyTable.key, keyCell{i}));

cTemp(find(cTemp==1)) = countCell{i};

cValue = [cValue, cTemp];

end

% some statistic

% occurence of keyword across the papers from 1987 to 2016

[mm,nn] = size(cValue);

meanKeyword = mean(cValue, 2);

meanKeywordSort = sort(meanKeyword,'descend');

key = tempKey(find(ismember(meanKeyword, meanKeywordSort(1:30))));

value = meanKeyword(find(ismember(meanKeyword, meanKeywordSort(1:30))));

row1 = find(ismember(meanKeyword, meanKeywordSort(1)));

data\_row1 = cValue(row1,:);

row2 = find(ismember(meanKeyword, meanKeywordSort(2)));

data\_row2 = cValue(row2,:);

row3 = find(ismember(meanKeyword, meanKeywordSort(3)));

data\_row3 = cValue(row3,:);

y = countcats(year);

figure;

barh(1:1:30,value);

set(gca, 'YTickLabel',key)

figure;

markerS = {'b^', 'c\*', 'go', 'y<', 'k>', 'mv', 'r+', 'c\*', 'go', 'y<'};

scatter3(data\_row1(1:y(1)), data\_row2(1:y(1)), data\_row3(1:y(1)), 'r+')

hold on;

for i=2:10 %1987-1996

scatter3(data\_row1(y(i-1):y(i)), data\_row2(y(i-1):y(i)), data\_row3(y(i-1):y(i)), markerS{i})

end

figure;

scatter3(data\_row1(y(10):y(11)), data\_row2(y(10):y(11)), data\_row3(y(10):y(11)), 'r+')

hold on;

for i=12:20 %1997-2006

scatter3(data\_row1(y(i-1):y(i)), data\_row2(y(i-1):y(i)), data\_row3(y(i-1):y(i)), markerS{i-10})

end

figure;

scatter3(data\_row1(y(20):y(21)), data\_row2(y(20):y(21)), data\_row3(y(20):y(21)), 'r+')

hold on;

for i=22:30 %2007-2016

scatter3(data\_row1(y(i-1):y(i)), data\_row2(y(i-1):y(i)), data\_row3(y(i-1):y(i)), markerS{i-20})

end

%% PCA -

% 1. normalized the data

cMean = mean(cValue);

cStd = std(cValue);

cNormalized =(cValue-repmat(cMean, [mm, 1]))./repmat(cStd, [mm, 1]);

cNormalized = cNormalized';

% 2. eigenfunction and covariance

[V, D] = eig(cov(cNormalized));

D = diag(D);

Dcum = cumsum(sort(D, 'descend')/sum(D));

plot(Dcum);

minD = find(Dcum>=0.95); %min dimension to achieve at least 95% of the variance

% 3. extract the principal components

% PC = cNormalized\*V; %========================

% this is to extract all PC(s)

% but for our case, let's just consider 2D first

sizeV = size(V);

V = flipud(V);

PC1 = cNormalized\*V(:, 1);

PC2 = cNormalized\*V(:, 2);

PC3 = cNormalized\*V(:, 3);

plot(PC1, PC2, 'rx');

plot3(PC1, PC2, PC3, 'rx');

% if we would like to achieve the 95% found above

PC = cNormalized\* V(:, 1:minD(1));

PCT = PC \* V(:, 1:minD(1))';

% 4. get back the compressed data

czValue = ((PCT) .\* repmat(cStd,[mm 1])) + repmat(cMean,[mm 1]);

meanKeyword2 = mean(czValue, 2);

meanKeywordSort2 = sort(meanKeyword2,'descend');

key2 = tempKey(find(ismember(meanKeyword2, meanKeywordSort2(1:30))));

value2 = meanKeyword2(find(ismember(meanKeyword2, meanKeywordSort2(1:30))));

row12 = find(ismember(meanKeyword2, meanKeywordSort2(1)));

data\_row12 = czValue(row12,:);

row22 = find(ismember(meanKeyword2, meanKeywordSort2(2)));

data\_row22 = czValue(row22,:);

row32 = find(ismember(meanKeyword2, meanKeywordSort2(3)));

data\_row32 = czValue(row32,:);

y = countcats(year);

figure;

barh(1:1:30,value2);

set(gca, 'YTickLabel',key2)

%% ++Selection + denoise

% find those very common keywords

commonWordIndex = find(mode(cValue, 2)> 0);

tempKeyFilter = tempKey;

tempKeyFilter(commonWordIndex) = [];

cValueFilter = cValue;

cValueFilter(commonWordIndex,:)=[];

% 1. normalized the data

[mmf,nnf] = size(cValueFilter);

cfMean = mean(cValueFilter);

cfStd = std(cValueFilter);

cfNormalized =(cValueFilter-repmat(cfMean, [mmf, 1]))./repmat(cfStd, [mmf, 1]);

% 2. eigenfunction and covariance

[Vf, Df] = eig(cov(cfNormalized));

Df = diag(Df);

Dcumf = cumsum(sort(Df, 'descend')/sum(Df));

plot(Dcumf);

minDf = find(Dcumf>=0.95); %min dimension to achieve at least 95% of the variance

% 3. extract the principal components

% PC = cNormalized\*V; %========================

% this is to extract all PC(s)

% but for our case, let's just consider 2D first

Vf = flipud(Vf);

PCf1 = cfNormalized\*Vf(:, 1);

PCf2 = cfNormalized\*Vf(:, 2);

PCf3 = cfNormalized\*Vf(:, 3);

plot(PCf1, PCf2, 'rx');

plot3(PCf1, PCf2, PCf3, 'rx');

% Denoise

Vdn = Vf;

Vdn(:, minDf(100:end)) = 0;

Vdn = Vdn \* Vdn';

PCfdn = cfNormalized\* Vdn;

% 4. retrieve back the data after deonise process

czfdnValue = ((PCfdn) .\* repmat(cfStd,[mmf 1])) + repmat(cfMean,[mmf 1]);

meanKeyword2 = mean(czfdnValue, 2);

meanKeywordSort2 = sort(meanKeyword2,'descend');

key2 = tempKeyFilter(find(ismember(meanKeyword2, meanKeywordSort2(1:100))));

value2 = meanKeyword2(find(ismember(meanKeyword2, meanKeywordSort2(1:30))));

row12 = find(ismember(meanKeyword2, meanKeywordSort2(1)));

data\_row12 = czValue(row12,:);

row22 = find(ismember(meanKeyword2, meanKeywordSort2(2)));

data\_row22 = czValue(row22,:);

row32 = find(ismember(meanKeyword2, meanKeywordSort2(3)));

data\_row32 = czValue(row32,:);

y = countcats(year);

figure;

barh(1:1:30,value2);

set(gca, 'YTickLabel',key2)