Hw7

- Task: the text classification task

- Transformation has the format “feat class1 class2”, which means:

```
if (feat is present) && (CurLabel == class1)
    then set CurLabel=class2
```

which is equivalent as

```
if (feat is present)
   then change CurLabel from class1 to class2
```

Q1: TBL trainer

- TBL_train.sh train_data model_file min_gain

  if net_gain < min_gain

  then stop iteration

- The format of model_file:

  init_class_name

  featName from_classname to_classname net_gain

  ….

- Ex of model_file:

  guns

  talk guns mideast 89

Q2: TBL decoder

- TBL_classify.sh   test_data   model_file sys_output > acc

- The format of sys_output:

  instanceName   trueLabel SysLabel trans1 trans2 ….

  Each transformation has the format:

  featName from_class   to_class

- Ex of sys_output:

  file1 guns mideast   we guns misc   talk misc mideast

Efficiency issue for training

- Method 1:
  - Read in all the training data to get a list of features and labels: map feat to feat-idx and label to label-idx for speedup.
  - Generate all the transformations with the form

    (feat, from-label, to-label)
  - Repeat
    - For each transformation, go through the data once to calculate its net-gain
    - Choose the best transformation with the highest gain
    - If the highest gain is less than min-gain

      then last;
    - apply the best transformation to update the last column

➔ For each iteration, go through the training data T times, where T is the number of transformations.

Calculate net gain of a transformation

- Suppose the transformation is (feat, from-label, to-label)

- net-gain = 0;

```
• For each training instance x {
    ▪ Let x  be  ({f_i}, gold-label, cur-label)   // {f_i} is the set of feats present in x
    ▪ If (feat does not belong to {f_i}) or (from-label != cur-label)
        then next;    //  no change to net-gain
    ▪ If to-label == gold-label
        then net-gain ++
       else {
           if cur-label == gold-label
             then net-gain --;
            else  nothing to do;  // no change to net-gain
       }
  }
```

Efficiency issue for training

- Method 2:
  - Read in all the training data to get a list of features and labels: map feat to feat-idx and label to label-idx for speedup.

  - Repeat
    - Go through the data once to generate transformations and calculate the net gains for all the transformations
    - Choose the best transformation with the highest gain
    - If the highest gain is less than min-gain then last;
    - apply the best transformation to update the last column

➔ For each iteration, go through the training data once.

Calculate net gains of all transformations

- net-gains = []; // net-gains is an array storing the net gains, and each element has value 0.

- Let C be the set of all the class labels

- For each training instance x {
  - Let x be ({f_i}, gold-label, cur-label) // {f_i} is the set of feats present in x
  - For each feature feat in {f_i} {

```
for every label to-label in C  that is different from cur-label {
    if to-label == gold-label
        then net-gains[idx-of(feat, cur-label, to-label)] ++;
        else if (cur-label == gold-label)
            then net-gains[idx-of(feat, cur-label, to-label)] --;
}
```

An example

Net-gains[(feat, cur-label, to-label)] =0

- x1 c1 f1 f20   (current-label=c2)

net-gains[(f1, c2,c1)] ++
net-gains[(f20, c2, c1)]++
%% net-gains[(f1, c2, c3)] and net-gain[(f20, c2, c3)] remain unchanged

- x10 c2 f3 f5   (current-label=c2)

net-gains[(f3, c2, c1)] --
net-gains[(f3, c2, c3)] –
net-gains[(f5, c2, c1)] --
net-gains[(f5, c2, c3)] –