gen_bigrams.prl

```
#!/usr/local/bin/perl

#########################################################
## This perl script is used to generate all the bigrams
## within the .stemmed files and .tokenized files (except
## the files for interactive queries which will be generated
## dynamically each time a use make inputs.
##
## @INPUT                       @OUTPUT
## cacm.stemmed         >       cacm.stemmed.bigrams
## cacm.stemmed.hist    >       cacm.stemmed.hist.bigrams
## cacm.tokenized       >       cacm.tokenized.bigrams
## cacm.tokenized.hist  >       cacm.tokenized.hist.bigrams
## query.stemmed        >       query.stemmed.bigrams
## query.stemmed.hist   >       query.stemmed.hist.bigrams
## query.tokenized      >       query.tokenized.bigrams
## query.tokenized.hist >       query.tokenized.hist.bigrams
##
#########################################################

use Carp;
use FileHandle;

my $DIR = "/home/pan/test/hw2";

# Initialize the stopword list for both stemmed and tokenlized
my $stoplist   = "$DIR/common_words";
my $stoplist_stemmed   = "$DIR/common_words\.stemmed";
my %stoplist_hash = ();
my %stoplist_stemmed_hash = ();

# Initialize the tokenlized stopword list
my $stoplist_fh   = new FileHandle $stoplist  , "r"
        or croak "Failed [stoplist_fh] $stoplist";

while (defined( $line = <$stoplist_fh> )) {
        chomp $line;
        $stoplist_hash{ $line } = 1;
}

# Initialize the stemmed stopword list
my $stoplist_stemmed_fh   = new FileHandle $stoplist_stemmed  , "r"
        or croak "Failed [stoplist_fh] $stoplist_stemmed";

while (defined( $line = <$stoplist_stemmed_fh> )) {
        chomp $line;
        $stoplist_stemmed_hash{ $line } = 1;
}


&gen_bigrams("cacm");
&gen_bigrams("query");

exit(0);


#########################################################
## GEN_BIGRAMS
##
## This function will generate the bigrams term sets
#########################################################
sub gen_bigrams {

        my $file_name = shift;

        # array to store all the bigrams as value
        my @bigrams = ();
        my $last_word = undef;  # used to store the last word so we could have pairs

        # my $file_name      = "interactive";
        my $raw_file       = "$file_name\.raw";
        my $stemmed_file   = "$file_name\.stemmed";
        my $tokened_file   = "$file_name\.tokenized";
        my $stem_hist_file = "$file_name\.stemmed\.hist";
        my $tokn_hist_file = "$file_name\.tokenized\.hist";
        my $stem_hist_com = "cat $stemmed_file\.bigrams | perl $DIR/make_hist.prl > $stem_hist_file\.bigrams";
        my $tokn_hist_com = "cat $tokened_file\.bigrams | perl $DIR/make_hist.prl > $tokn_hist_file\.bigrams";

        my $tokened_file_fh = new FileHandle $tokened_file, "r"
                or croak "Failed $tokened_file";

        my $stemmed_file_fh = new FileHandle $stemmed_file, "r"
                or croak "Failed $stemmed_file";


        #########
        # make the stemmed bigrams
        #########
        open(RAWFILE, ">$stemmed_file\.bigrams");
```

```
90
91          while (defined( $word = <$stemmed_file_fh> )) {
92                  chomp $word;
93
94                  # if we encounter a normal word, we check if it eligible for bigram
95                  if ($last_word =~ /^[a-zA-Z]/ and $word =~ /^[a-zA-Z]/) {   # start of query tokens
96                    # if both words are not in the stoplist, we added to the bigrams array
97                    if (! exists $stoplist_stemmed_hash{ $last_word } and ! exists $stoplist_stemmed_hash{ $word }) {
98                          push (@bigrams, "$last_word+$word");
99                    }
100                 }
101
102                 # if next doc/query, pop all the elements in the array into file
103                 if ($word =~ /^\.[A-Z]/ and $#bigrams != -1) {
104                         # after we scanned each doc/qry, all the bigrams are in the array now
105                         # we append all the bigrams to the end of doc/qry
106                         foreach $bigram (@bigrams) {
107                                 print RAWFILE "$bigram\n";
108                         }
109                         # clean up the array
110                         @bigrams = ();
111                 }
112
113                 # we shift last_word by 1 and write the current word into the output file
114                 $last_word = $word;
115                 print RAWFILE "$word\n";
116         }
117
118         # do it once again to make sure we have everything write into the file
119         if ($#bigrams != -1) {
120                 foreach $bigram (@bigrams) {
121                         print RAWFILE "$bigram\n";
122                 }
123         }
124
125         # close the bigram file after we created it
126         close(RAWFILE);
127
128         # generate hist file
129         system ("$stem_hist_com") and die "Failed $DIR/make_hist.prl: $!\n";
130
131         # clean up the array
132         @bigrams = ();
133
134         #########
135         # make the tokenlized bigrams
136         #########
137         open(RAWFILE, ">$tokened_file\.bigrams");
138
139         while (defined( $word = <$tokened_file_fh> )) {
140                 chomp $word;
141
142                 # if we encounter a normal word, we check if it eligible for bigram
143                 if ($last_word =~ /^[a-zA-Z]/ and $word =~ /^[a-zA-Z]/) {   # start of query tokens
144                   # if both words are not in the stoplist, we added to the bigrams array
145                   if (! exists $stoplist_hash{ $last_word } and ! exists $stoplist_hash{ $word }) {
146                         push (@bigrams, "$last_word+$word");
147                   }
148                 }
149
150                 # if next doc/query, pop all the elements in the array into file
151                 if ($word =~ /^\.[A-Z]/ and $#bigrams != -1) {
152                         # after we scanned each doc/qry, all the bigrams are in the array now
153                         # we append all the bigrams to the end of doc/qry
154                         foreach $bigram (@bigrams) {
155                                 print RAWFILE "$bigram\n";
156                         }
157                         # clean up the array
158                         @bigrams = ();
159                 }
160
161                 # we shift last_word by 1 and write the current word into the output file
162                 $last_word = $word;
163                 print RAWFILE "$word\n";
164         }
165
166         # do it once again to make sure we have everything write into the file
167         if ($#bigrams != -1) {
168                 foreach $bigram (@bigrams) {
169                         print RAWFILE "$bigram\n";
170                 }
171         }
172
173         # close the bigram file after we created it
174         close(RAWFILE);
175
176         # generate hist file
177         system ("$tokn_hist_com") and die "Failed $DIR/make_hist.prl: $!\n";
178
179         # clean up the array
180         @bigrams = ();
```

```
181     }
```