BANNER - Problem Statement

# Overview

The BioNLP (Biomedical Natural Language Processing) community, represented by the academic group from Scripps, is testing the limits of crowd-sourcing for generating annotated corpora within the biomedical domain and for doing information extraction directly. To accomplish these tasks effectively, algorithms are needed that can learn to accurately merge data collected from multiple annotators of varying quality and integrate this data into predictive models.

There is an already developed open-source supervised learning system called BANNER that achieves a good level of prediction power after being trained on subset of abstracts, manually annotated by experts. However, the training capabilities of the current algorithm are restricted to a very small dataset, limited by expensive expert's time. There is an idea that this limitation can be overcome if we teach BANNER algorithm how to further improve its accuracy by training on [MTurk](http://www.mturk.com/)-annotated abstracts, potentially available in the data in much larger quantities.

In this contest, the goal is to improve the BANNER accuracy by teaching it on MTurk-annotated abstracts. Meanwhile, some different and interesting competition styles will be explored by Harvard Business School.

# Available Software

The BANNER Java source code can be downloaded from <https://svn.code.sf.net/p/banner/code/trunk/> for usage and modifications. The BANNER\_BioC.sh script in the scripts folder can be used to execute BANNER on BioC formatted training and testing files.

The software trains a model by taking a number of annotated abstracts as input. The trained model can then be used to annotate abstracts that have not been annotated before.

Example code can be download [[here](https://bitbucket.org/sulab/crowd_words)] that performs the following:

* Reads in a MTurk annotation file
* Applies a simple voting procedure to make reasonably good approximations of anon-redundant training corpus and stores them as new BioC files
* The BioC files can be used directly in order to train BANNER.

# Available Data

The training data can be downloaded [here] and the testing data can be downloaded [here].

Some of the abstracts are labelled by experts, while some are labelled by MTurk workers (15 annotators per abstract). You will be given the following training data:

* XXX abstracts annotated by both experts and MTurk.
* YYY abstracts annotated only by experts and ZZZ annotated only with MTurk.

You will be given AAA abstracts that needs to be annotated. Only a sub set of 200 will be used for scoring. The remaining abstracts will be used to prevent contestants from manually annotating the abstracts. The example case will only score 1 abstract, for provisional scoring 49 abstracts will be used and the remaining 150 abstracts will be used for system tests.

In order to be eligible for a prize, you will be requested to submit all the source code and data that you used to generate your results. We must be able to re-create your submission results from the provided source.

# Different and interesting competition styles

Competitors are going to be randomly split into several competition rooms. There are three types of rooms: ***marathon***, ***race*** and ***minimum-quality marathon***.

* The ***marathon*** type is the traditional competition but in smaller groups.
* The ***race*** maintains the same structure of a traditional MM (including, preliminary scoring, leaderboard, testing data, etc). The only difference is that the final score is also computed at each submission (but not shown to coders). So that as soon as ***N*** coders achieve a final score equal or higher than a given level ***Q***, the competition is over and the system communicates this event to all participants. The first ***N*** coders to achieve ***Q*** will be awarded ***N*** different prizes.
* In a ***minimum-quality marathon***, the same rules of MM hold but coders are awarded a prize if and only if they achieved a final score equal or higher than ***Q***.

Based on the current performance of BANNER, we set ***Q*** = 0.9 and ***N*** = 3 [TODO].

# Implementation

In this contest, the training and testing data will remain the same and you have to generate your results offline.

Your annotate method should return the contents of the generated annotated abstracts. BANNER will create an annotated file when the testing data have been processed. You need to include the contents of this file in your submission. Each element in the return array should contain one line from your results file.

# Scoring

Your score will be calculated by matching your annotations with our expert annotations. <http://en.wikipedia.org/wiki/F1_score> over the set of abstracts will be used and multiplied by 1000000.

You can see these scores for example test cases when you make example test submissions. If your solution fails to produce a proper return value, your score for this test case will be 0.

## Definition

Class: BannerClassifier  
Method: annotate  
Parameters: void  
Returns: vector<string>  
Method signature: vector<string> annotate(vector<string> trainingData, vector<string> testingData) (be sure your method is public)

# Additional Information

## Special Rules

* In order to receive the prize money, you will need to fully document your code and explain your algorithm. If any parameters were obtained from the training data set, you will also need to provide the program used to generate these parameters. There is no restriction on the programming language used to generate these training parameters. Note that all this documentation should not be submitted anywhere during the coding phase. Instead, if you win a prize, a TopCoder representative will contact you directly in order to collect this data.
* You may not use any external (outside of this competition) source of data to train your solution.
* You are not limit to use BANNER, you can use any other publicly available tool or write your own software to produce the output.
* You are not allowed to manually annotate the testing data.

## Notes

* The match forum is located [here]. Please check it regularly because some important clarifications and/or updates may be posted there.
* You can train your solution offline based on the given files and you can hardcode data into your solution--just remember that you can't use data from other sources than this contest.
* Time limit is 5 minutes per test set and memory limit is 1024MB.
* There is no explicit code size limit. The implicit source code size limit is around 1 MB (it is not advisable to submit codes of size close to that or larger).
* The compilation time limit is 60 seconds. You can find information about compilers that we use and compilation options [here].