**Agenda, SSL Methods for data augmentation**

At the moment, we are basing our measurements on **three data sets**:

1. 20 newsgroups (16200 entries, 100 features, 4 classes). Worthile because it's small enough to work with RocSVM, unlike Reuters and KDD2001.

2. Reuters(8200 entries, 18933 features, 65 classes). The set of entries is already split into test/train, with similar distributions. 3713 postives (class 1), the rest relabeled to 0. The test and training set have the same positives/total ratio.

3. KDD2001 binding to thrombin - we have a proper training and test set. Few positives(42) in the training set. In this data set, the proportion of positives in the training set is 2.2%, whereas the proportion of positives in the test set is 23.66%, so this is a good set to examine how the disparity can affect the performance. This is an important point for discussion, as this disparity is quite common (proof!?) in biomedical data.

I **General paradigm**:

I A common framework for assesing the quality of data augmentation. The paradigm is the following:

1.1: Apply SVMs (SVM light/libSVM) on all data sets, treating them as typical (binary) classification problems and obtaining their performance metrics (precision, recall, f-score).

1.2: First, construct three reduced test sets( rTest = sample(testSet) ), as well as the remaining "unlabeled" set. First one is 10%, second 33%, third 80%. All of these should be saved and well documented, so that the results of all these experiments can be reproduced.

Train the SVM using the reduced test set; then, measure its perfomance on the test set.

Apply all three entity set expansion algorithms to the reduced test set, treating the Test / rTest as the Unlabeled set.

Having expanded the set of new positives from the unlabeled set, add them to the rTest, and retrain the SVM. Obtain the new set of performance metrics.

Together with the actual performance metrics of the entity expansion on these problems (managing expectations - need to see how precision of PU learning on rTest -> Test affects the actual, post data augmentation performance boost), we should have enough information for an informed discussion of applicability of these algorithms for data augmentation, as well as a potential mechanism to mechanically choose the best approach (of the three) for a given dataset.

Standard SVMs(like libSVM) are an adequate choice because they provide a uniform interface for multiple datasets, as well as a uniform framework for which we attempt data augmentation, and against which we can compare its performance.

**II List of further work**

1. Further work on processing all data sets. Constructing the reducedTrain(s), Train and Test sets and the respective labels for direct use/future reproduction/verification in Matlab.

2. Applying sEM, RocSVM and Spy-EM algorithm to reducedTrain to obtain augmented reduced sets; measuring the precision & recall of the entity set expansion these PU algorithms achieve.

3. Applying (and potentially mechanising?) application of Standard SVMs to our classification problems. Establishing a uniform method to assess their performance accross all data sets.

4. Using sEM to augment all three reducedTrain sets and observing how the new set improved/degraded the subsequent classification.

5. Determining a proper cutoff criteria for using Bayesian sets' rankings(scores) for data augmentation. Subsequently, examine its performance on our datasets.

5.1: Potentially use the topK, where K is the number of new ones identified by sEM. We might also observe the score trends and decrease/increase, depending on where the closest gap in these algorithms is.

6. RocSVM testing & optimization: operations must be vectorised so that we can apply RocSVM on large data sets. Subsequently, examine its performance on our datasets in the same way as sEM.

7. Create an automatic method that will, given test/train: create three redux sets, run SVM, run all data augmentation methods on reduxes, run SVMs again, and output all measurement results neatly summarized, so that we can asses the performance of data augmentation AND entity set expansion (N.B: Potentially distinct, and intersting to consider separately, as we can actually always measure the second - if it implies something about the first, it can be used to conclude which algorithm is best to use in a new situation.

8.W.R.T. Negatives can be harmful to binary classification. We can assess this by having bicategorical training drawn from i.e. Reuters(i.e. use mostly 2. class for negatives), then use testing data where negatives are drawn mostly from the class 3, and see how SVMs do. Once we've done this, we can compare how this degrades the quality w.r.t. having negatives from the same distribution as training. Once we have that comparison, we use SEM/RocSVM to join training positives(read up more in the paper) with test, and just extract them, then see if it does better.

Other considerations: cross validation; movieLens dataset; Once we look at performance of SVMs with augmented data, compare it to performance with the whole actual train data set.

IMPORTANT: Utilization of the negatives - expansion of negatives (in addition to positives - any sense, given usuall applications??) for SVM classification.

Stochastic gradient descent as a faster alternative to SVMs. http://scikit-learn.org/stable/modules/sgd.html

When applying libSVM in Matlab, overfitting manifests itself as skewed precision/recall tradeoff. This doesn't seem to be an issue with Reuters, just with KDD => this is proof of impact of skewed distributions on quality of classification (cite relevant paper).

Validity of just trying to improve binary classification? Reducing multi-label problems to binary ones (standard PU setting).

My SpyEM and RocSVM do not return a classifier - and they don't need to (they could easily), as we are only using them for entity set expansion.

However, metrics used so far might be inadequate - we should be measuring the precision and recall on the new/hidden positives, not including the known positives.

Whether or not this is true/relevant, the impact that data augmentation achieved doesn't depend on this metric! So, when we get new positives, we may get away just with checking whether they are in P - as we're dealing with reduxes, this should be fine. We can just check how many of these are actual positives and output this - this will be the best (in applicable terms) measurement comparison of these three data augmentation methods performance in practice!

What are we doing to make this an open source, applicable library??

Data augmentation on multilabel vs bi-label settings?? In multilabel, we can just augment every class. Also, use and distinguish all classes to help.

Implicit relevance feedback - Iterative Bayesian Sets. Blind relevance feedback. Relevance feedback!

Take a frequentist approach to determining the approximate number to take out via IterativeBayesian - based on the training data, expect that we have a similar ratio of P:N in the unlabeled set - which is usually the case, as we're choosing random elements to classify manually, and expand from the body not searched yet