Comparison of Multiple Interest Metrics in Open Set Recognition

António Morais*, Pedro Sá*

Universidade de Coimbra

**Both authors contributed equally to the writing and research in this study. Their names are listed in alphabetical order.*

December 19, 2019

Abstract

Recent surges in the quantity and variety of data lead to a constant search to find techniques that optimize the process of training machine learning classifiers. The cases where there are a lot of unlabelled data and the cost of annotating is not insignificant are particularly meaningful. Because of this, there has been progress in the area of active learning, which is related to minimizing the cost (computational and otherwise) of sample annotation. Additionally, the ever-growing spread of machine learning to real-life cases makes it important to steer away from static (closed-set) models and move into dynamic (open-set) models, which do not limit the classifier to a fixed number of classes. Since these are 2 heavily researched areas, it is pertinent to seek different querying metrics to optimize the performance of such classification algorithms. On this paper, we present a continuation of past research on the usage of *unknown interest* as described in *Macedo et al. (2011)* [1] to bring insight into the comparison of the performance of several querying metrics.

1 Introduction

Active learning aims to achieve higher accuracy with fewer labelled examples than passive learning by choosing from what data to learn from. It requires less training to have a similar or even better performance which becomes a huge advantage given that labelling samples can be expensive and time consuming. Fundamentally, one or more instances with a higher interest are chosen to be labelled by an oracle - an expert. The most widespread version of active learning is based on a fixed number of classes. This perspective is limited in the sense that in the real-world, there are often new/unknown types of samples. One solution to this limitation is the adoption of an open set recognition (OSR) approach, which is based on the assumption that the classifier may have an incomplete knowledge of the world, and as such allows the conception of new classes which can be submitted to an algorithm during training. This allows the classifier to adapt to previously unseen cases, which is a relevant characteristic in the mentioned circumstances. The OSR approach has been implemented by *Scheirer, Rocha, Sapkota Boult* (2013) [1] and has been explored by *Macedo et al.* (2011) [2]. The application of OSR to active learning has been implemented by *Liu Huang* (2019) in [3], with satisfactory results in cases where the training dataset is unlabeled. This technique has also been investigated in the field of Computer Vision by *Meyer Drummond* (2019) [4], applied to a high number of images. The method involve using several metrics, which we will refer to as interest metrics. New samples are queried to the oracle according to the value of a maximization function that accepts either an unknown interest, or an uncertainty interest. If this value is above a variable threshold, the sample is sent to the oracle for annotation. The problem with this approach is that the interest metrics are independent from one another; in other words, a sample can be picked for annotation if it has a sufficiently high unknown interest, regardless of the value of the uncertainty interest, which could possibly be very low. Our perspective is that other models should be explored, specifically models where both interest types have weight on whether the sample is picked for annotation or not. Additionally, we consider it is worth exploring other methods for measuring distance between samples, which we compare with the current method, Euclidean. Finally, we test other models for the uncertainty interest and compare the results with the entropy model. These changes could theoretically lead to better efficiency in the selection by choosing samples that provide higher information gain.

2 OSR and Active Learning

Traditionally classifier systems have a fixed set of classes, which is a characteristic of closed-set spaces. Real-world circumstances differ from these types of spaces in that they imply an incomplete knowledge of the world. What this means in practice is that the classifier algorithm has not only to correctly label samples belonging to known classes, but it also has to correctly identify samples that match classes that have not yet been observed. This is called open-set classification, and it is a feature that we implemented in our work.
Active learning, as explained by Settles in *An Analysis of Active Learning Strategies for Sequence Labeling Tasks (2008)* [5], is a case of semi-supervised machine learning that is built on the idea that the ability for a learning algorithm to select which samples to learn from can increase its overall performance. It is especially relevant in cases where labeling is expensive, challenging and slow, and where there are large quantities of data.
Our implementation of Active Learning utilizes pool-based sampling. First, the algorithm, called the active learner, is trained on a set of labeled samples, generating a model. This model is then used to provide information which in turn is used in the selection of unlabeled instances for labeling. Then they are sent to an oracle, which is an expert that provides the correct label to a given sample. There are systems where this is done by a human; in our case, since the original dataset is already fully labelled we merely take the label from that dataset.

2.1 Interest in Querying

In Macedo et al. [2] paper on exploration of an unknown environment it is used a multi agent approach in which, the explorer agent has the task of classifying the unknown objects, based on the available prototypes on already seen objects. It gives a probability distribution correlating the features of the new object with this prototypes using a weighed average of the Euclidean distance [6]. This gives us a good sense of similarity between the existing classes and the possibility of an unknown class and it's similarity with the new object is given by:

$$sim(O_{new}, O_{unknown}) = 1 - \max_i sim(O_{new}, O_i) \quad (1)$$

where O_i represents the i^{th} prototype. Given our different format, we decided to go a slightly different route by determining the $distance_to_unknown(O_{new})$as the difference between the maximum and the minimum euclidean distances. In this way, we can have a good grasp of the variance in the euclidean distances, where a larger difference means that the set of features is heavily correlated with one prototype, resulting in a greater distance to a new class. Higher the difference in the distances, lower will me the similarity to a new class.

$$\begin{aligned} &distance\_to\_unknown(O_{new}) = \\ &= \max_i distances\_to\_prototype(O_{new}, O_i) \\ &- \min_i distances\_to\_prototype(O_{new}, O_i) \end{aligned} \quad (2)$$

where again, O_i represents the i^{th} prototype. With the distance, we could now transform it in similarity metric, using as a rule of thumb, the following:

$$\begin{aligned} &sim(O_{new}, O_{unknown}) = \\ &= \frac{1}{1 + distance\_to\_unknown(O_{new})} \end{aligned} \quad (3)$$

and logically: $sim(O_{new}, O_i) = \frac{1}{1+distance\_to\_prototype(O_{new})}$

These correlation are transformed into a probability distribution with:

$$p(y_{new}, y_j | x_{new}) = \frac{sim(O_{new}, O_j)}{\sum_{i=1}^{n+1} sim(O_{new}, O_i)} \quad (4)$$

being x_{new} representative of the features of our new object, this gives us the probability of that some object to belong to an unknown class yet to be discovered. In order to make the decision, either the explorer agent calls another agent to classify the object given it can belong to a new class of objects or the agent classifies it himself at distance. This decision is based on two interest metrics, the first one being the interest for the unknown:

$$I_{unknown}(O_{new}) = tanh(2 * P(y_{new} = y_{unknown})) \quad (5)$$

and the second one being the interest on the uncertainty associated with labeling the object:

$$\begin{aligned} &I_{uncertainty}(O_{new}) = \\ &= -\sum_{j=1}^{n+1} P(y_{new} = y_j) * \log(P(y_{new} = y_j) \end{aligned} \quad (6)$$

given by the entropy formula which represents the chaos, the indecision of labeling the new instance. In this paper, the agent chooses the maximum interest as the overall interest and makes the decision to classify at distance or not if the value is above a certain threshold.

2.2 Combining Interest

As stated in Macedo's paper, future work should focus on combining these metrics rather than choosing the maximum value, which only utilizes one of them and disregards the other one. With that in mind, we propose the sum of these interests as well as a weighted sum given that while developing this experiment we noted that the interests had different scales. In order to fight that discrepancy we give an arbitrary value to the unknown interest based on the interest values outputted by program.

2.3 Methods

For implementing this project we used the Python programming language and the *libact* active learning library (`https://github.com/ntucllab/libact`). We adapted and used code from the *libact* library in the files that are mentioned:

2.3.1 *uncertainty_sampling.py*

We started by acknowledging the existence of another way to query the instances for the oracle to label and for that we updated the *_init_* function in order to accept the new query strategy: *interest*. Essentially, we altered the way the *make_query* functions in order to, when summoned, verifies if the query strategy asked by the user is *interest* and if true, as for the other metrics, it trains the model and gets the unlabeled part of the dataset but, in this case it updates the existing prototypes of the known classes (average values of the given features). With the prototypes defined, we now calculate the euclidean distance from all the unlabeled instances to all prototypes as well as the distance to an unknown class by implementing equation 2 and define the similarities with equation 3. We proceed by getting a probability distribution with equation 4 and with that we can now calculate the unknown interest, equation 5 and the interest in uncertainty equation 6. Keeping in mind our three different combinations of interest: *Max, Sum* and *Weighted*, the approach to pick the instance to be queried next is always the: pick the maximum value of interest given by the metric. In other words, for each unlabeled instance we will end up with two interest values and according to the metric chosen will we narrow it down to a single value, wither by picking the maximum, applying a weighted sum with an arbitrary value or simply adding both values. Having only one interest per instance, we will pick the instance with the maximum value out of the pool of instances yet to be labeled.

In future work, this can be optimized if we only updating the prototype from the class of the latest labeled instance as well as saving the previous distances and only computing the distance for the instances that weren't picked for the class that was labeled, as it is the only prototype that will go through changes.

2.3.2 *test_plot.py*

This file is the way the user interacts with the program by inputting the metric of the combination (*Max*, *Weight*, *Sum*), the dataset (*sat*, *iris*, *abalone*, *wine*) and the number of samples he wants to compute. We started by using the *plot.py* file in the *libact* examples which returned a plot displaying error by the number of queries made. The size of the test portion of the dataset is defaulted to 40% and the number of labeled instances to begin the training is 10. These values try to replicate to some extent, a real world scenario but can be easily changed to better suit the user intentions. We start by dividing the dataset in test and training sets and until all unlabeled instances are labeled, the system, using the combination metric for the interests inputed by the user, chooses what it considers to be the best instance to be labeled and an oracle, in this case, a function is called to label the entry with its true label. After this, the model is re-trained and predicts the labels from the test set. As stated later in this document, the **F1-Score** and **AUROC** (Area Under the Receiver Operating characteristics) is saved to later be plotted in relation to the number of queries.

2.4 Experiments

We ran the file on the terminal in Mac OS X using the following command:
test_plot.py interest_metric dataset_name n_samples
where *interest_metric* represents the combination of both interests: *Max*, *Weight*, *Sum*; *dataset_name* can be one of the names of the datasets used (referenced later in this document); *n_samples* as stated before, represents the number of times the combination is repeated, later averaging the result and consequently plotting it. In the code we used *matplotlib.pyplot.plot* function which is a graphical tool from the *libact* AL library. With this, we display 2 different metrics, AUROC [7][8] and F1-score, from the different types of query strategy, represented by different colors. These metrics are used to provide graphical information regarding the performance of all query strategies. Additionally, we saved the AUROC and the F1_Scores to a separate text file to perform a more detailed analysis of the results as well as comparing the two different metrics so we could establish a value for the weighted sum metric [1]. We utilize 4 different databases:

- **iris**: morphological variation of iris flowers of three related species (150 entries, 3 classes, 4 features) [9];
- **abalone**: physical measurements of several abalone specimens in order to predict the age (4177 entries, 3 classes, 8 features) [10];
- **wine**: physiochemical and sensory evaluations of white wine to classify it in terms of quality (4898 entries, 7 classes, 11 features) [11];
- **sat**: multi-spectral values of pixels in 3x3 neighbourhoods in a satellite image, and the classification associated with the central pixel in each neighbourhood (6435 entries, 7 classes, 36 features) [12];

We aimed to get a vast variety of entries, features and classes to get a better understanding of our metrics im-

[1]several values were experimented trying to balance out the difference between these values

Dataset	Query Strategy	F1-Score	AUROC
iris	Max	1	2
iris	Sum	3	4
iris	Weight5	5	6
abalone	Max	7	8
abalone	Sum	9	10
abalone	Weight15	11	12
wine	Max	13	14
wine	Sum	15	16
wine	Weight15	17	18
sat	Max	19	20
sat	Sum	21	22
sat	Weight50	23	24

Table 1: Reference to figures in the Appendix section

pact in active learning. Adding to that, we utilize the already implemented *libact* query strategies, random and uncertainty sampling to have a way to compare our results and adjust our idea and implementation to achieve the best result possible. We were able to perform tests combining all interest metrics for each dataset but due to time constrains and the heavy computing required to run the experiment several times, recognizing the alterations we made to the fundamental way *libact* works aggravating the processing power needed, the number of experiments did not meet our expectations but some conclusions can be made as it follows.

2.5 Results and Discussion

In this section we present our results for all datasets that were described in the previous part and are presented in the Appendix in the latter part of this paper (Tab. 1). We can extract some information about the behaviour of the different query strategies. We use *max*,*sum* and *weighted* query strategies as defined before. The value for the weight is dependant on the value of *unknown interest* and of *uncertainty interest*; the goal is for the *unknown interest* to have similar values to the *uncertainty interest* so that they both have the same weight; for example, in the case of the **iris** dataset, the average value of *unknown interest* is 5 times lower than the average value of *uncertainty interest.* Because of this, the ratio for the *weighted* query strategy is 5. The weighted ratios are different because the value of *unknown interest* differs according to the number of classes and features. As a rule of thumb, the increase in the number of classes results in a lower value of *unknown interest.*

iris: comparing *max* (fig. 1, 2) and *sum* (fig. 3, 4) approaches is clear the results are similar and better than the *uncertainty* method, this is due to the minimal unknown interest values making the two different metrics very much alike; *weight* (fig. 5, 6)is used with a weight value 5 for the unknown interest in in the pursue of equivalent results in both interests;

abalone: considering the *weight* (fig. 11, 12) with weight value at 15, we can see that the beginning portion of the results presents a better performance, but overall, it has a similar behaviour to the *random* method; the *sum* (fig. 9, 10) method has a very similar comportment to the *entropy* results and the *max* (fig. 7, 8) outputs, in general, a better result than the other two query strategies; note that the average F1-Score and AUROC are worse than in previous datasets and this can be due to it being not so balanced;

wine: compared with the previous datasets, both performance metrics showed poorer results in general, with values being roughly halved. As for the query strategies, they demonstrate different values depending on the methods utilized. In *max* (fig. 13, 14), the performance is superior or equal to *random* sampling, and always higher than the *uncertainty.* In *sum* (fig. 15, 16), the performance is not conclusively better than any of the other query strategies. Finally in *weight* (fig. 17, 18), with a ratio of 15, there is considerably higher performance with a lower number of queries, especially compared to *random*; however for a higher number of queries, they converge.

sat: lastly, the *sat* dataset presents the worst performance metrics results. This may be caused by the huge discrepancy between the interest values that make the *max* (fig. 19, 20) pick the uncertainty value as the interest value and as stated before, this is calculated with the probability values computed with the euclidean distance which explained the notable difference from the *entropy* line. The adoption of a *weight* (fig. 23, 24) 50 query strategy increases the performance slightly, although in general it is still worse.

Notes: iris data is perfectly balanced contrary to the others; the *random* query strategies is the best strategy in general displayed the best results.

3 Conclusions

In conclusion, attributing a value to the possibility of having an unknown class never seen before adds a certain complexity to the model which we tried to handle in the form of combinations, *max* value, *sum* and *weighted sum*, to reach an overall metric that accounted for both interests at different datasets. The results prove that it is very difficult to achieve a perfect combination that satisfies all problems but sheds light on future work on these interests metrics.

Acknowledgements

We would like to thank our professor Macedo for the guidance and useful suggestions along the development of this paper.

References

[1] Scheirer, W., Rocha, A., Sapkota, A., Boult, T. (2013, July) *Toward Open Set Recognition* in IEEE Transactions on Software Engineering

[2] Macedo, L., Tavares, M., Gaspar, P., Cardoso, A. (2011,October) *Uncertainty and Novelty-based Selective Attention in the Collaborative Exploration of Unknown Environments* in 5th Portuguese Conference on Artificial Intelligence (EPIA 2011)

[3] Liu, Z., Huang, S. (2019, July) *Active Sampling for Open-Set Classification without Initial Annotation* in Vol 33 No 01: AAAI-19, IAAI-19, EAAI-20

[4] Meyer, B., Drummond, T. (2019,February) *The Importance of Metric Learning for Robotic Vision: Open Set Recognition and Active Learning* in 2019 IEEE International Conference on Robotics and Automation (ICRA)

[5] Burr Settles, Mark Craven (October 25 - 27, 2008) An Analysis of Active Learning Strategies for Sequence Labeling Tasks. Retrieved from https://dl.acm.org/citation.cfm?id=1613855

[6] Steve Borgatti, Boston College (n.d.) Distance and Correlation. Retrieved from http://www.analytictech.com/mb876/handouts/distance_and_correlation.htm

[7] J A Hanley, B J McNeil (1982, April 1) The meaning and use of the area under a receiver operating characteristic (ROC) curve. Retrieved from https://pubs.rsna.org/doi/abs/10.1148/radiology.143.1.7063747

[8] Global Score: The Area under the Learning Curve. (n.d) Retrieved from http://www.causality.inf.ethz.ch/activelearning.php?page=evaluationcont

[9] Fisher (1936) Iris Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Iris

[10] Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn and Wes B Ford (1994) Abalone Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Abalone

[11] Paulo Cortez (2009) Wine Quality Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Wine+Quality

[12] Ashwin Srinivasan (1993) Statlog (Landsat Satellite) Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)

Appendix

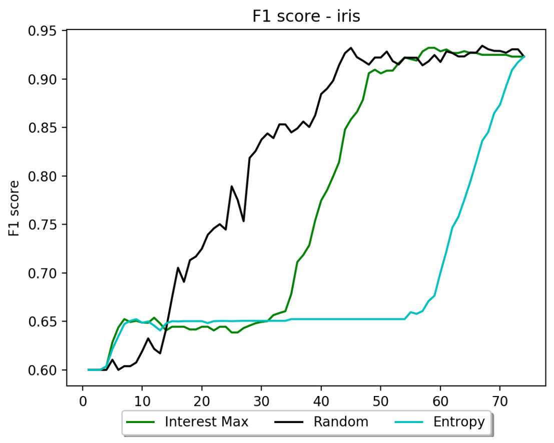

Figure 1: F1-Score with Max interest combination for dataset *iris*

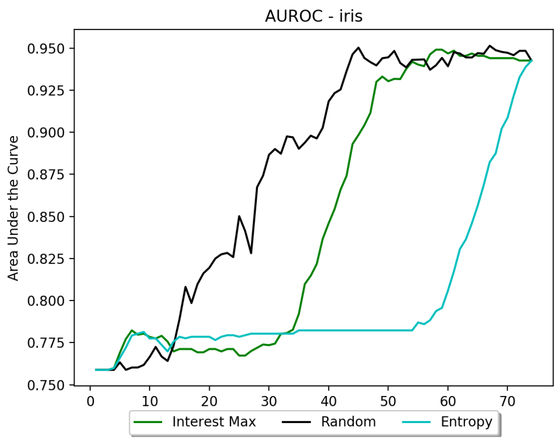

Figure 2: AUROC with Max interest combination for dataset *iris*

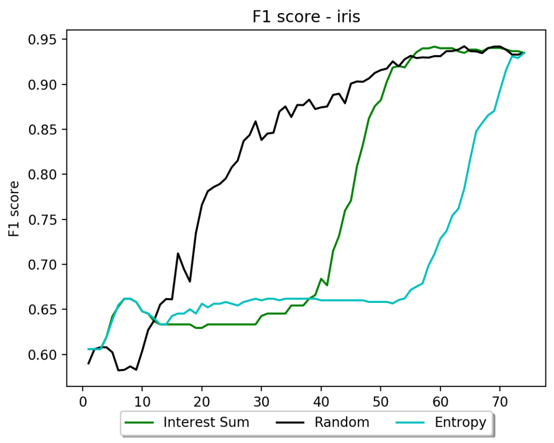

Figure 3: F1-Score with Sum interest combination for dataset *iris*

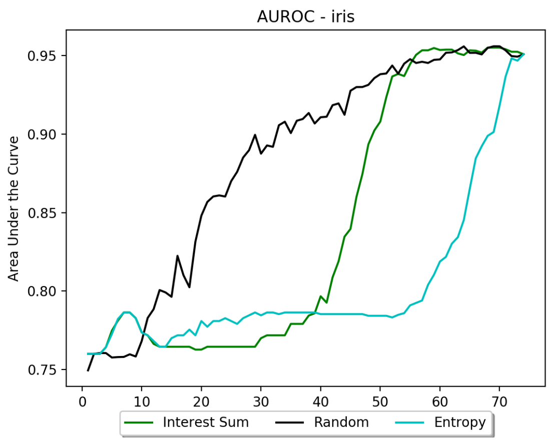

Figure 4: AUROC with Sum interest combination for dataset *iris*

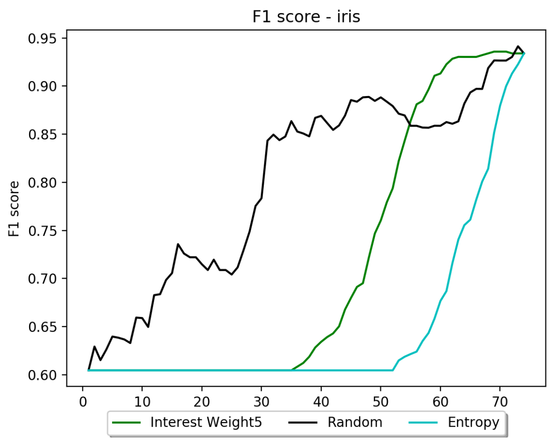

Figure 5: F1-Score with Weight interest combination (value=5) for dataset *iris*

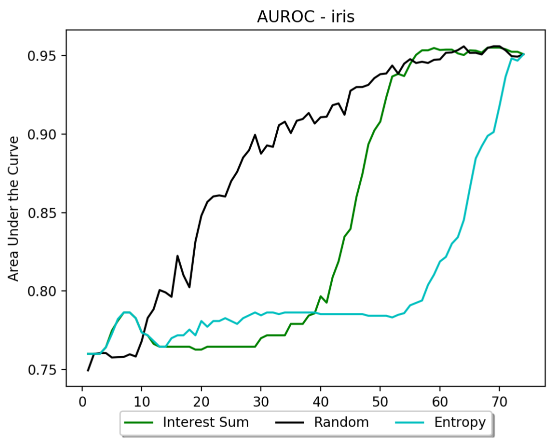

Figure 6: AUROC with Weight interest combination (value=5) interest combination for dataset *iris*

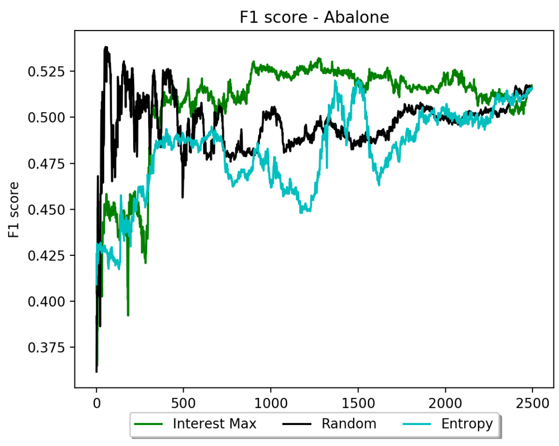

Figure 7: F1-Score with Max interest combination for dataset *abalone*

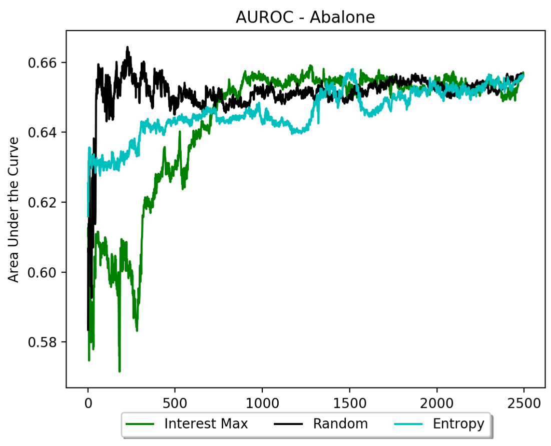

Figure 8: AUROC with Max interest combination for dataset *abalone*

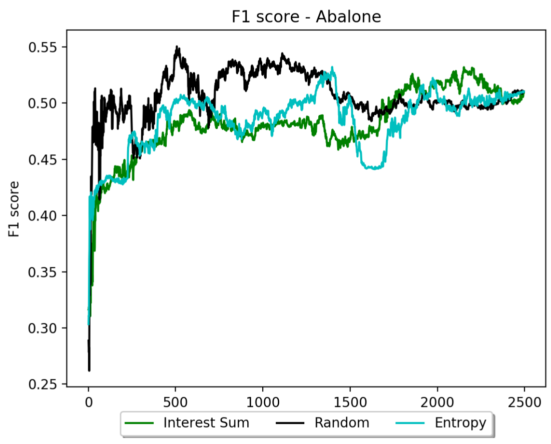

Figure 9: F1-Score with Sum interest combination for dataset *abalone*

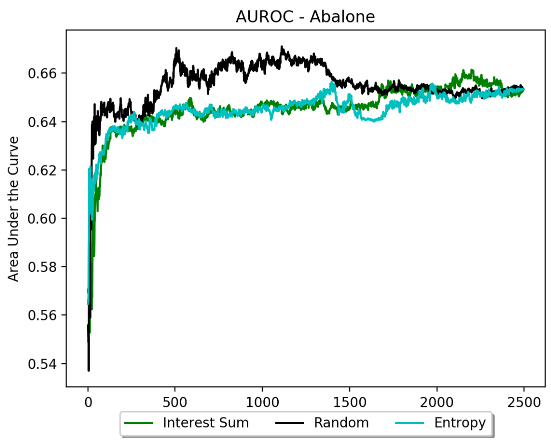

Figure 10: AUROC with Sum interest combination for dataset *abalone*

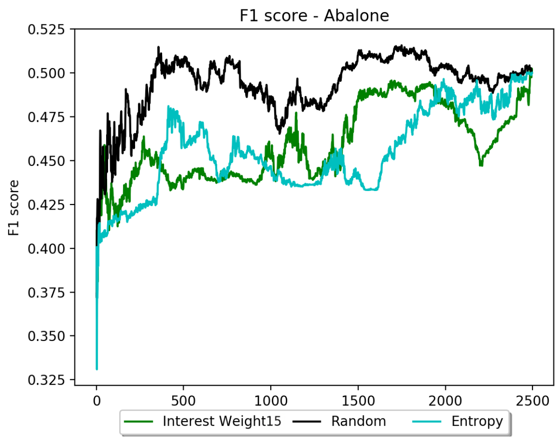

Figure 11: F1-Score with Weight interest combination (value=5) for dataset *abalone*

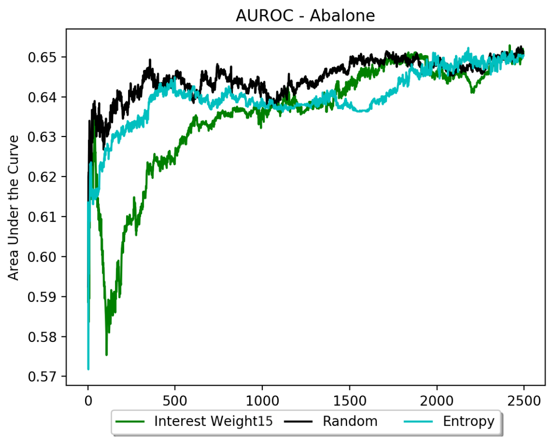

Figure 12: AUROC with Weight interest combination (value=5) for dataset *Abalone*

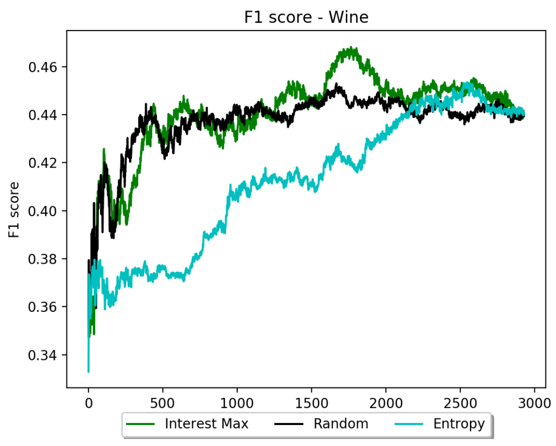

Figure 13: F1-Score with Max interest combination for dataset *wine*

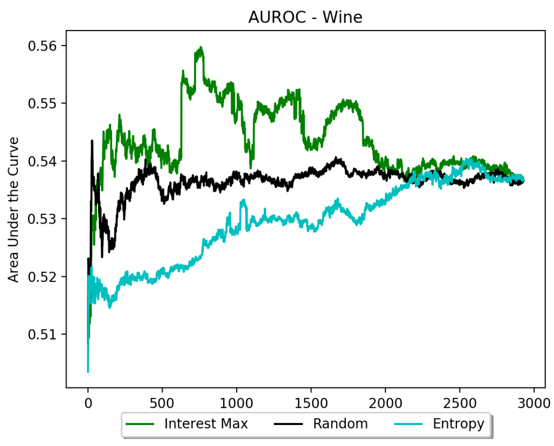

Figure 14: AUROC with Max interest combination for dataset *wine*

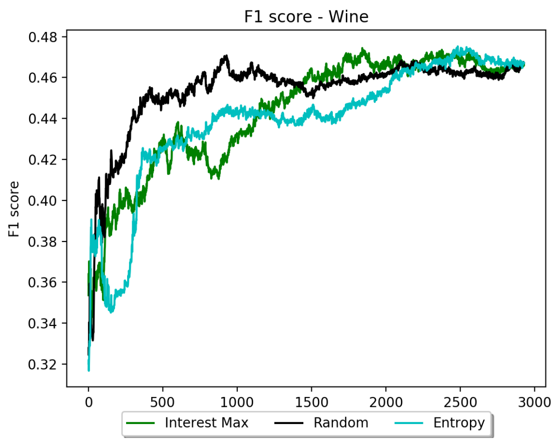

Figure 15: F1-Score with Sum interest combination for dataset *wine*

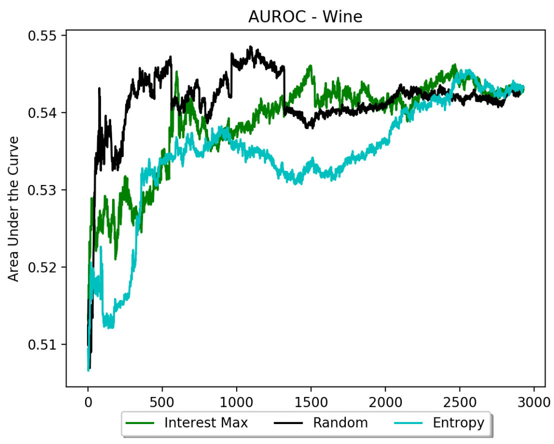

Figure 16: AUROC with Sum interest combination for dataset *Wine*

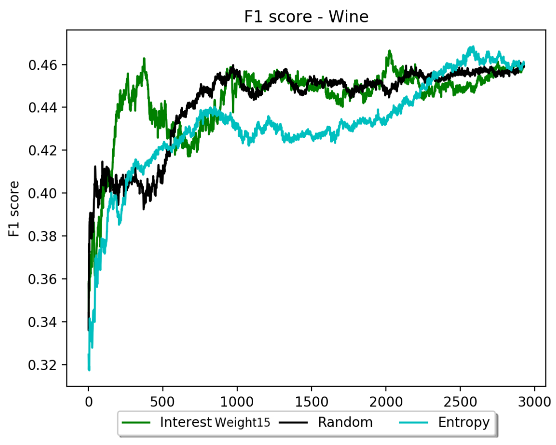

Figure 17: F1-Score with Weight interest combination (value=15) for dataset *Wine*

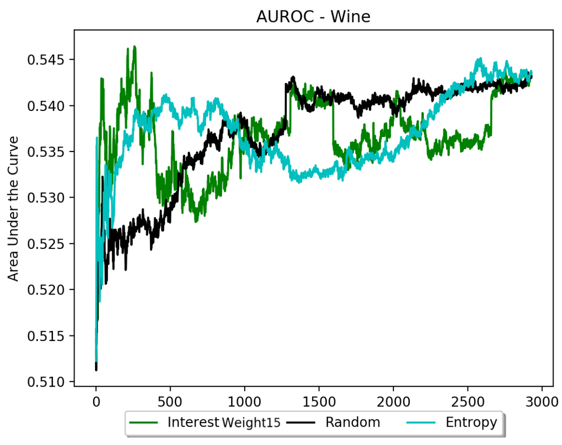

Figure 18: AUROC with Weight interest combination (value=15) for dataset *Wine*

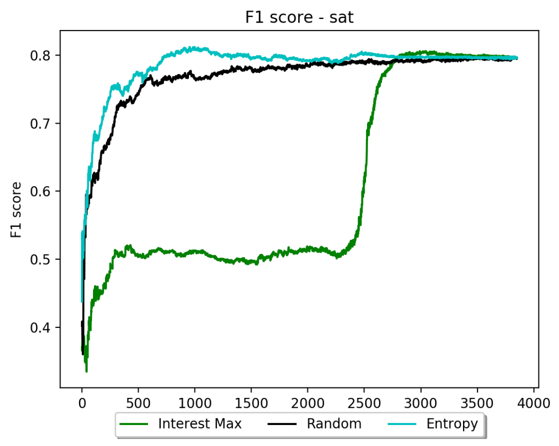

Figure 19: F1-Score with Max interest combination for dataset *Sat*

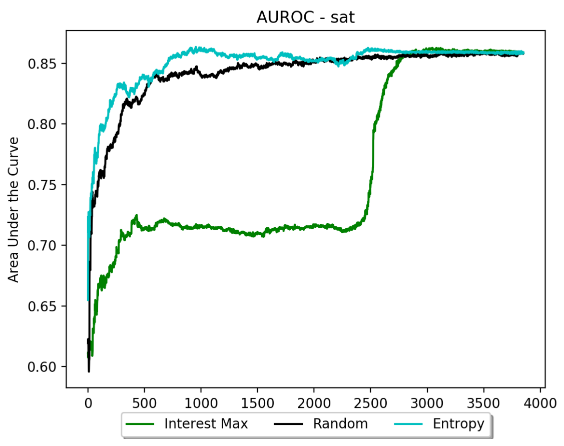

Figure 20: AUROC with Max interest combination for dataset *sat*

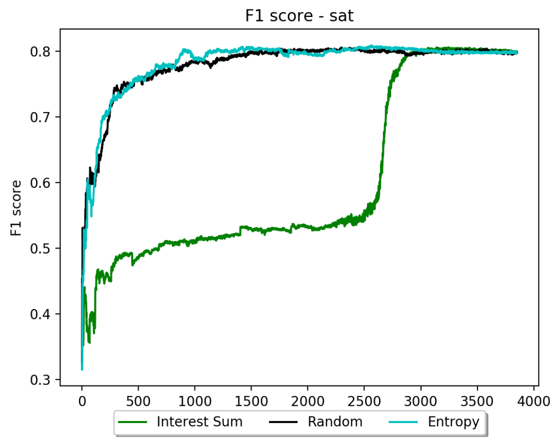

Figure 21: F1-Score with Sum interest combination for dataset *sat*

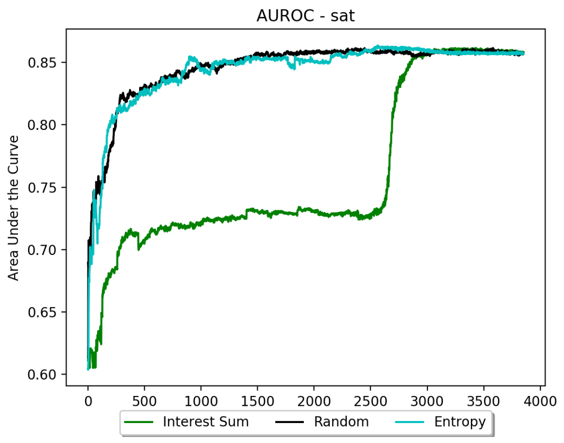

Figure 22: AUROC with Sum interest combination for dataset *Sat*

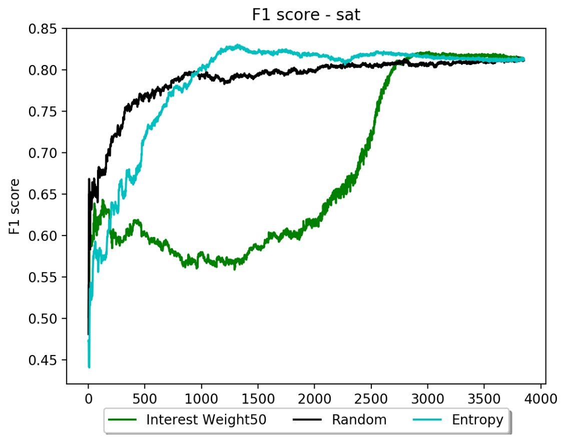

Figure 23: F1-Score with Weight interest combination (value=50) for dataset *Sat*

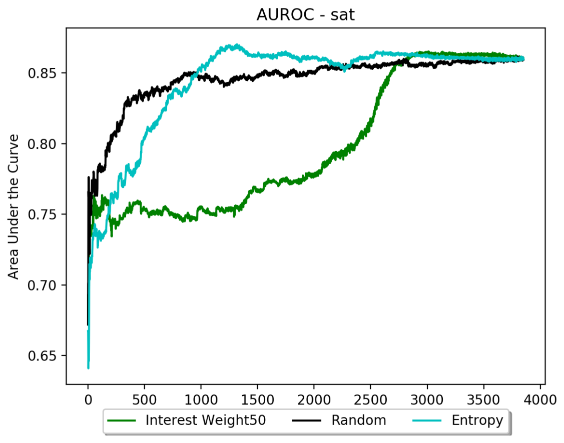

Figure 24: AUROC with Weight interest combination (value=50) for dataset *sat*