SoK: The Impact of Unlabelled Data in Cyberthreat Detection

Giovanni Apruzzese, Pavel Laskov, Aliya Tastemirova
Institute of Information Systems – University of Liechtenstein
{*name.surname*}*@uni.li*

***Abstract*—Machine learning (ML) has become an important paradigm for cyberthreat detection (CTD) in the recent years. A substantial research effort has been invested in the development of specialized algorithms for CTD tasks. From the operational perspective, however, the progress of ML-based CTD is hindered by the difficulty in obtaining the large sets of labelled data to train ML detectors. A potential solution to this problem are semisupervised learning (SsL) methods, which combine small labelled datasets with large amounts of unlabelled data.**

This paper is aimed at systematization of existing work on SsL for CTD and, in particular, on understanding the utility of unlabelled data in such systems. To this end, we analyze the cost of labelling in various CTD tasks and develop a formal cost model for SsL in this context. Building on this foundation, we formalize a set of requirements for evaluation of SsL methods, which elucidates the contribution of unlabelled data. We review the state-of-the-art and observe that no previous work meets such requirements. To address this problem, we propose a framework for assessing the benefits of unlabelled data in SsL. We showcase an application of this framework by performing the first benchmark evaluation that highlights the tradeoffs of 9 existing SsL methods on 9 public datasets. Our findings verify that, in some cases, unlabelled data provides a small, but statistically significant, performance gain. This paper highlights that SsL in CTD has a lot of room for improvement, which should stimulate future research in this field.

***Index Terms*—machine learning, semisupervised learning, cybersecurity, labelling, concept drift, threat detection**

1. Extended Results

This document extends the main paper [1] by providing the additional performance metrics of our experiments. In particular, for each dataset, we report the *F1-score*, *Precision* and *Recall* achieved by the developed ML models. Specifically:

$$Recall = \frac{tp}{tp+fn},$$

$$Precision = \frac{tp}{tp+fp},$$

$$F1\text{-}score = \frac{tp}{.5(fp+fn)+tp},$$

where *tp*, *fp*, *fn*, *tn* denote true positives, false positives, false negatives, true negatives. All these metrics consider 'positive' to be 'malicious' samples.

The results for Network Intrusion Detection are in Figs. 1–4. Those for Malware Detection are in Figs. 5–7. Finally, those for Phishing Website Detection are in Figs. 8–10.

Moreover, specifically for Network Intrusion Detection, we have computed the results on the "troubleshooted" version of the IDS17 dataset (described in [2]). These results, shown in Figs. 2, are obtained by setting (n, k)=$(25, 4)$, and by considering the following attacks: *DoS* (we grouped all DoS variants into a single family), *DDoS*, *Portscan*, *sshftp* — hence, N=4. We excluded the *webattacks* (which are included in the main paper) because the troubleshooted variant of this dataset presents an insufficient amount of samples for this specific attack to reproduce the same labelling budgets as in the main paper.

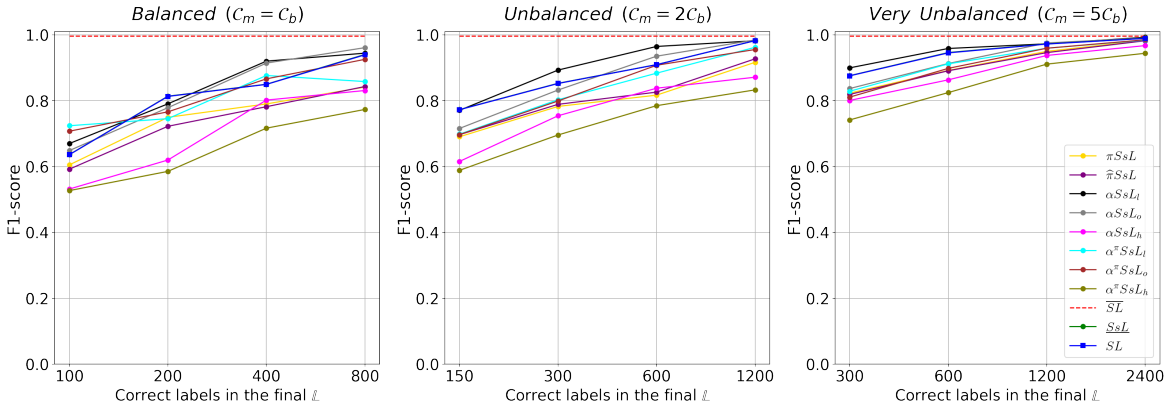
We also mention an intriguing finding. From an F1-score viewpoint, the best pseudo-labelling method is $\underline{SsL}$, which is statistically equivalent to SL due to a p-value of 0.17 — a result that also occurs for the 'original' version of IDS17. However, the best active-learning method is $\alpha^{\pi}SsL_o$, which is statistically superior than the baseline SL due to a p-value < 0.001 — which is in stark contrast to the best active-learning method on the 'original' version of IDS17, which was αSsL_l and which was statistically equivalent to the baseline SL.

Hence, using unlabelled data **can** provide an improvement on the fixed version of IDS17 (at least in our considered testbed).

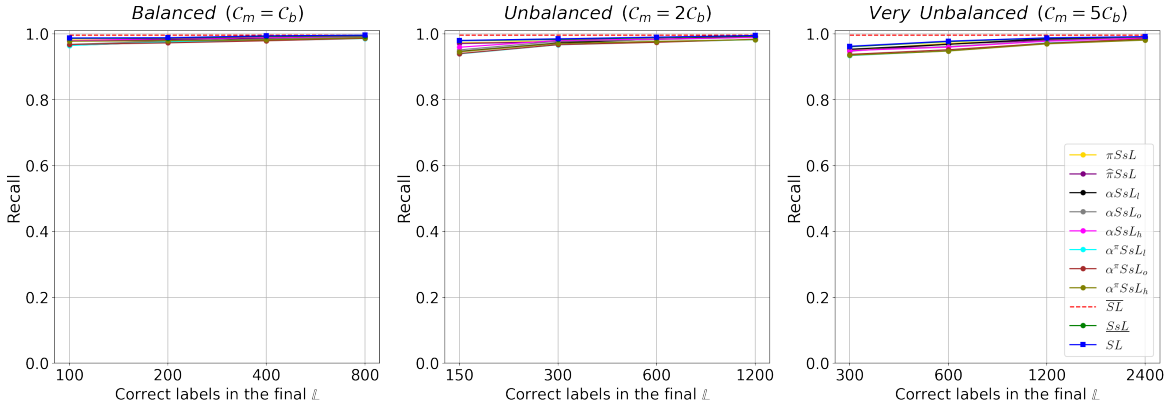
References

[1] G. Apruzzese, P. Laskov, and A. Tastemirova, "SoK: The Impact of Unlabelled Data in Cyberthreat Detection," in *Proc. 7th IEEE European Symposium on Security and Privacy*, 2022.

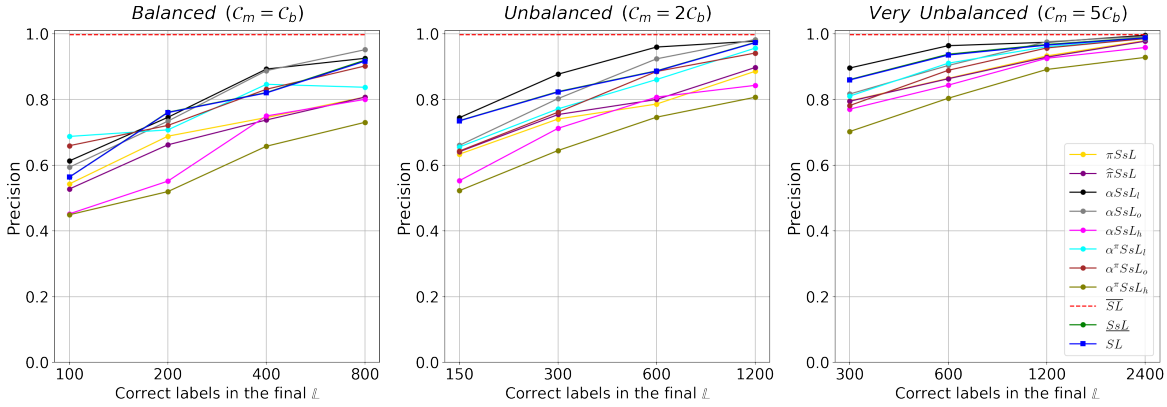
[2] G. Engelen, V. Rimmer, and W. Joosen, "Troubleshooting an intrusion detection dataset: the CICIDS2017 case study," in *Proc. IEEE Secur. Privacy Workshops*, 2021, pp. 7–12.



(a) *F1-score* on **IDS17**. For every plot, each 'point' represents the average *F1-score* of 225 models (and 5 times as much for all models using active learning).

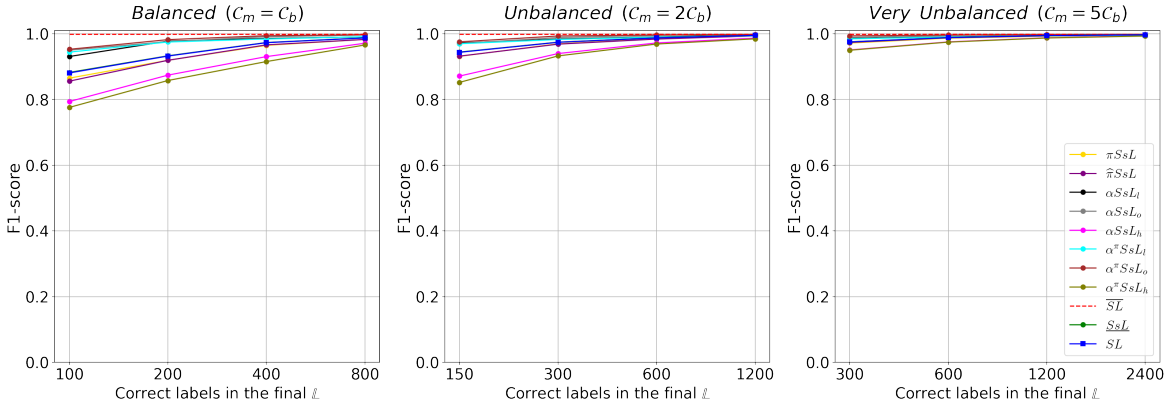


(b) *Recall* on **IDS17**. For every plot, each 'point' represents the average *Recall* of 225 models (and 5 times as much for all models using active learning).

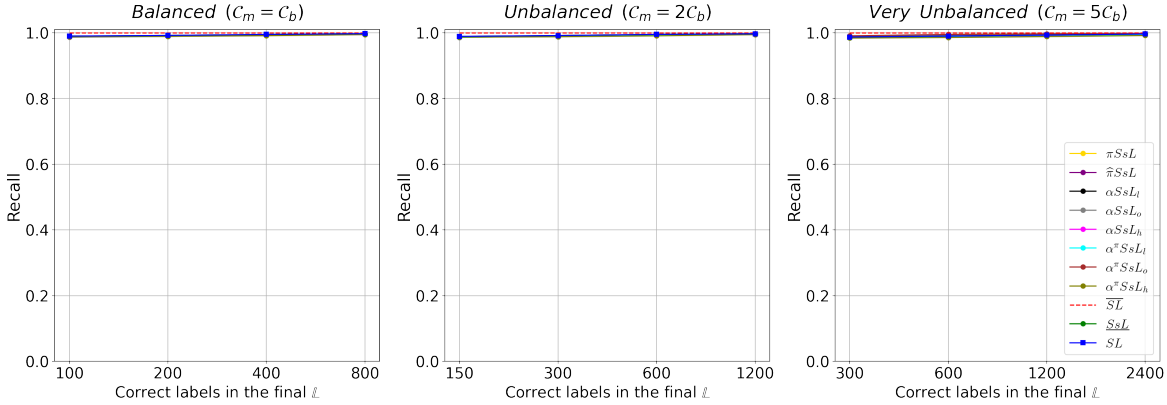


(c) *Precision* on **IDS17**. For every plot, each 'point' represents the average *Precision* of 225 models (and 5 times as much for all models using active learning).

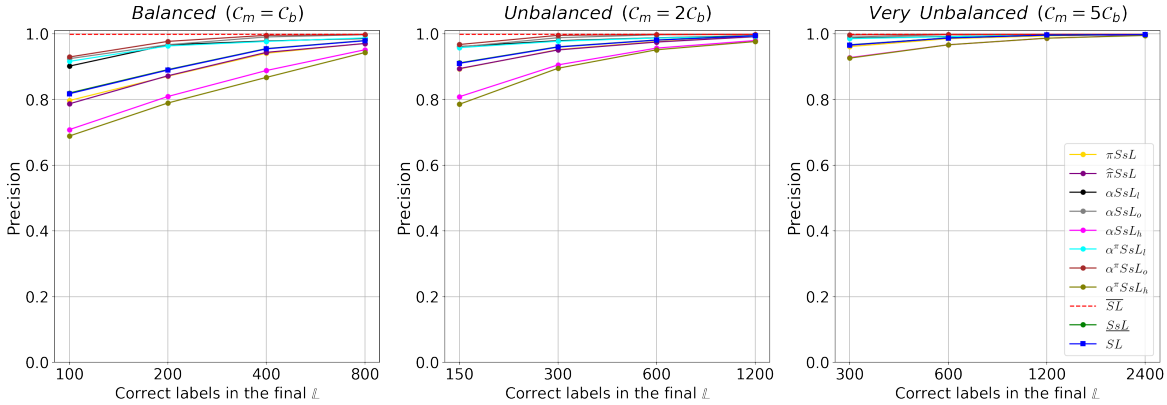
Figure 1: **Network Intrusion Detection**—IDS17. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on the 'fixed' **IDS17**. For every plot, each 'point' represents the average *F1-score* of 400 models (and 5 times as much for all models using active learning).

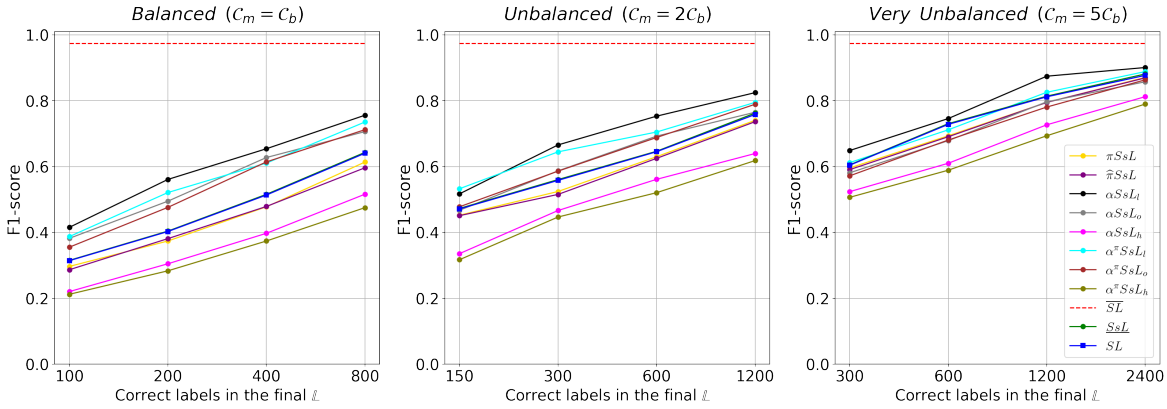


(b) *Recall* on the 'fixed' **IDS17**. For every plot, each 'point' represents the average *Recall* of 400 models (and 5 times as much for all models using active learning).

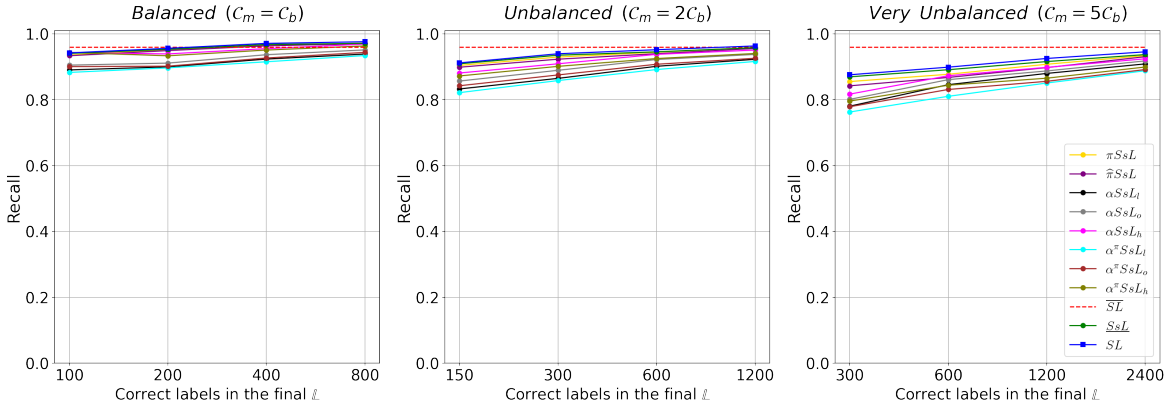


(c) *Precision* on the 'fixed' **IDS17**. For every plot, each 'point' represents the average *Precision* of 400 models (and 5 times as much for all models using active learning).

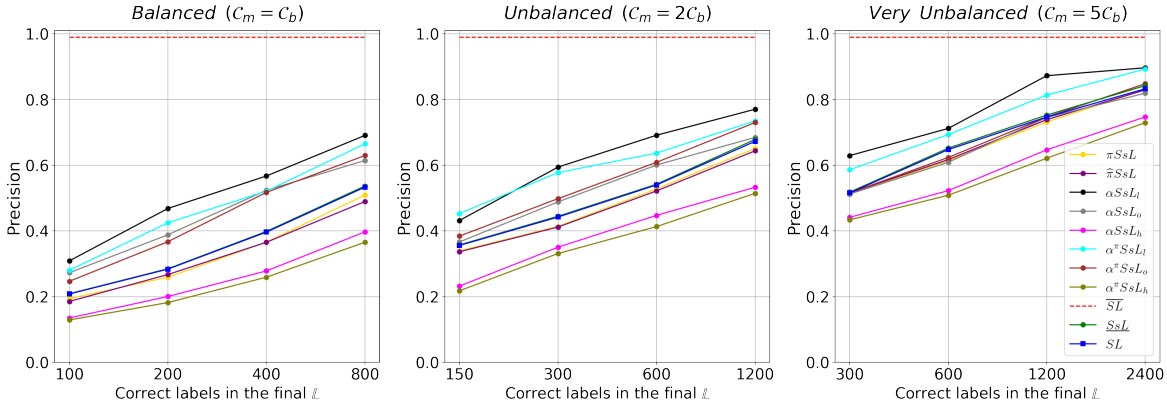
Figure 2: **Network Intrusion Detection**—`IDS17` (troubleshooted). We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **CTU13**. For every plot, each 'point' represents the average *F1-score* of 198 models (and 5 times as much for all models using active learning).

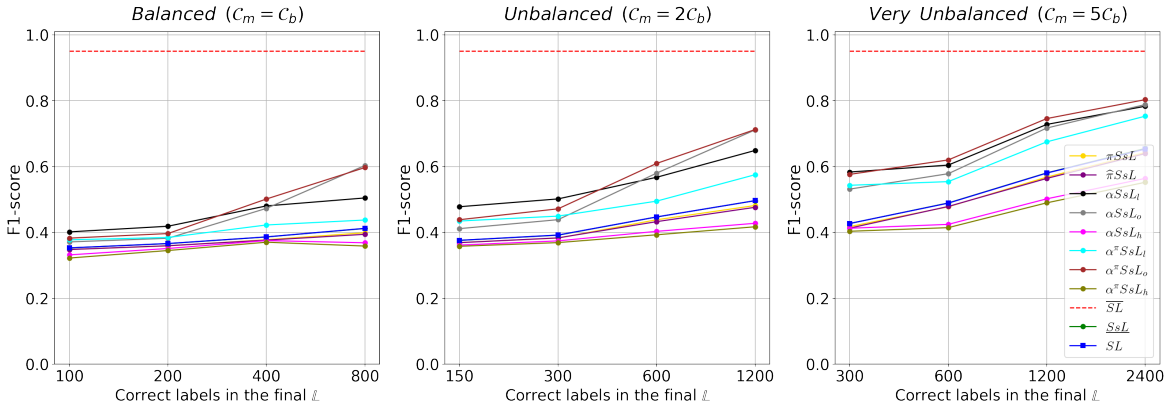


(b) *Recall* on **CTU13**. For every plot, each 'point' represents the average *Recall* of 198 models (and 5 times as much for all models using active learning).

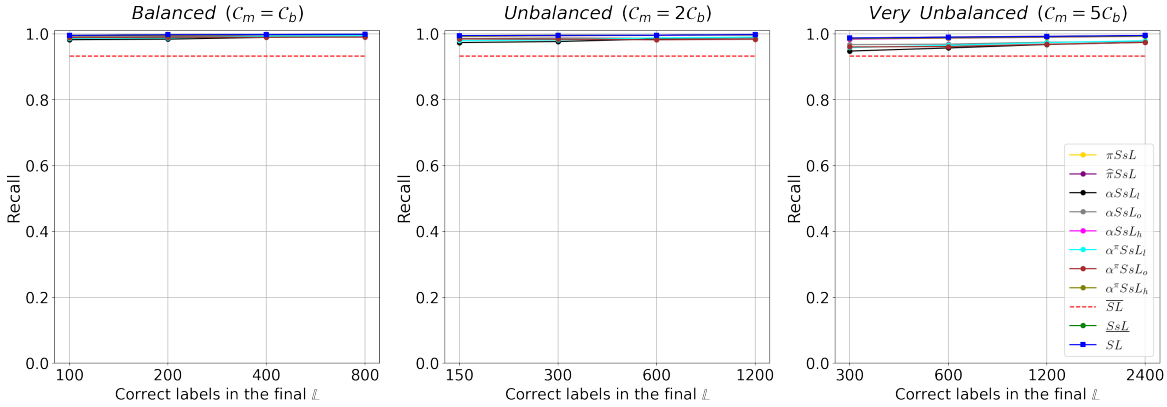


(c) *Precision* on **CTU13**. For every plot, each 'point' represents the average *Precision* of 198 models (and 5 times as much for all models using active learning).

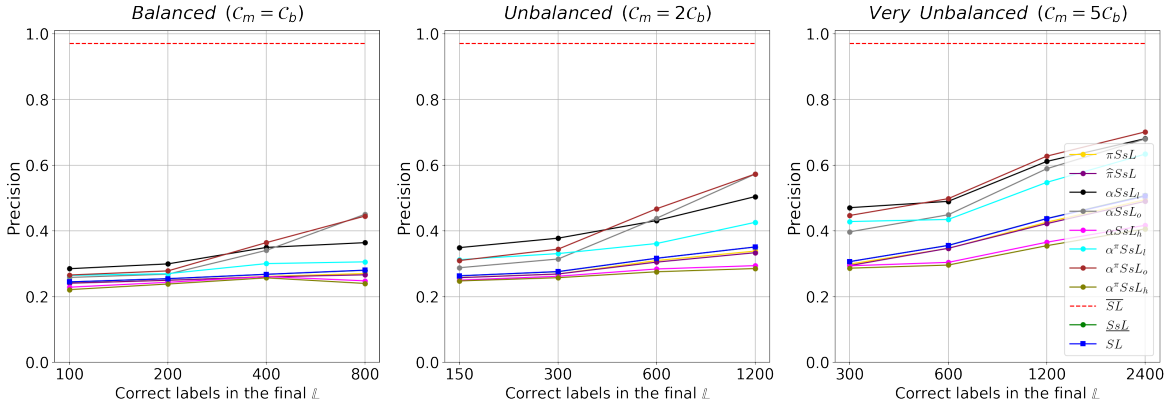
Figure 3: **Network Intrusion Detection**—CTU13. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **UNB15**. For every plot, each 'point' represents the average *F1-score* of 368 models (and 5 times as much for all models using active learning).

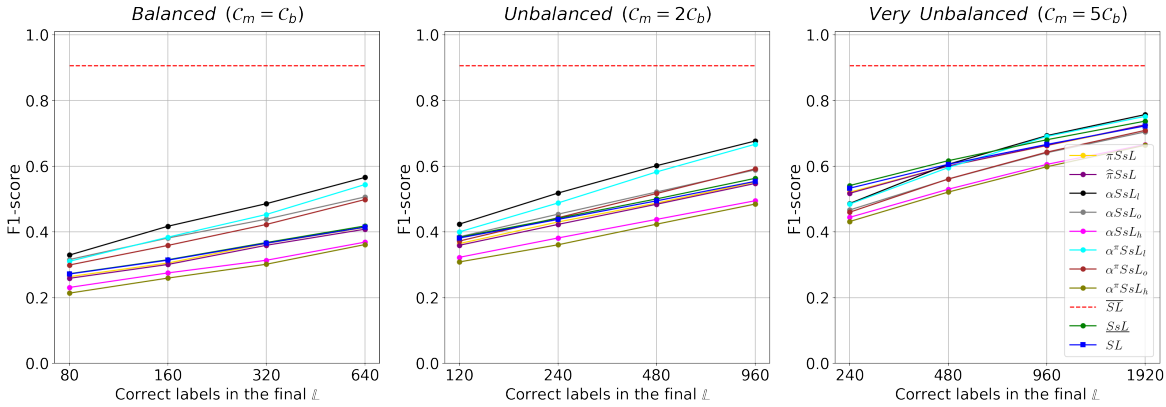


(b) *Recall* on **UNB15**. For every plot, each 'point' represents the average *Recall* of 368 models (and 5 times as much for all models using active learning).

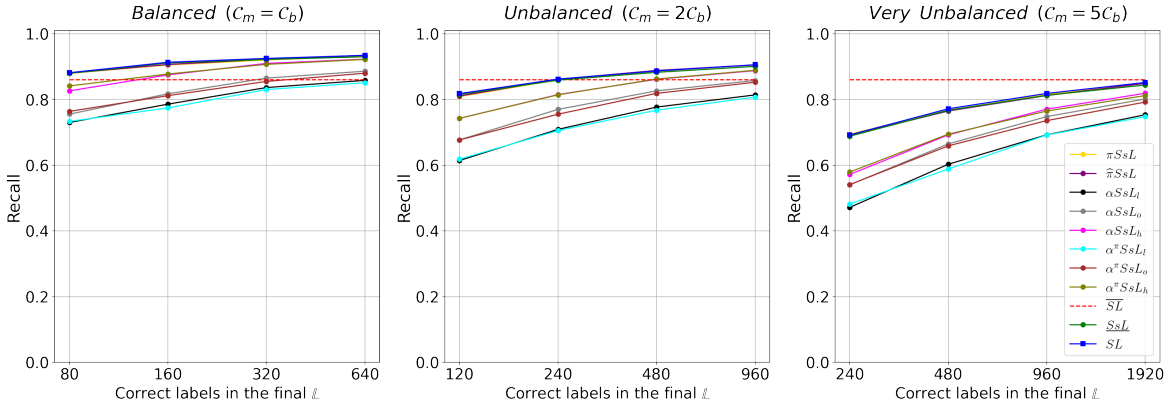


(c) *Precision* on **UNB15**. For every plot, each 'point' represents the average *Precision* of 368 models (and 5 times as much for all models using active learning).

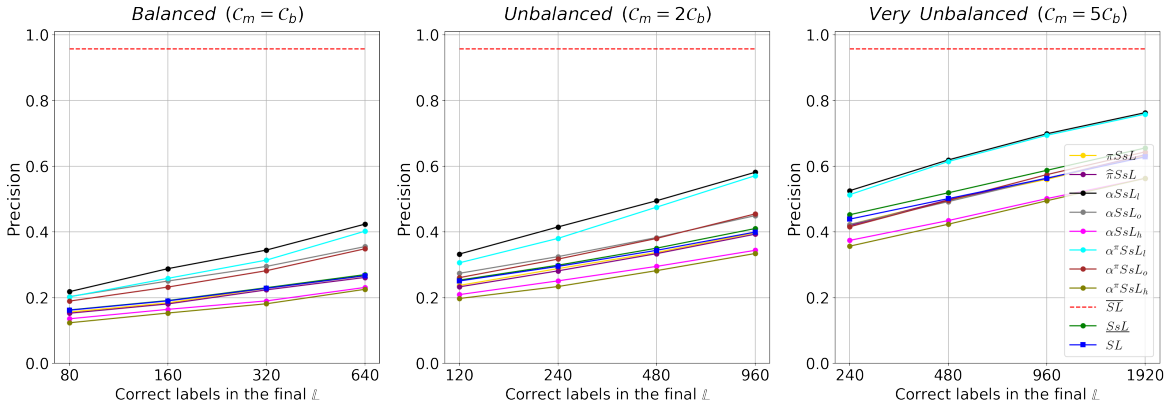
Figure 4: **Network Intrusion Detection**—UNB15. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **DREBIN**. For every plot, each 'point' represents the average *F1-score* of 100 models (and 5 times as much for all models using active learning).

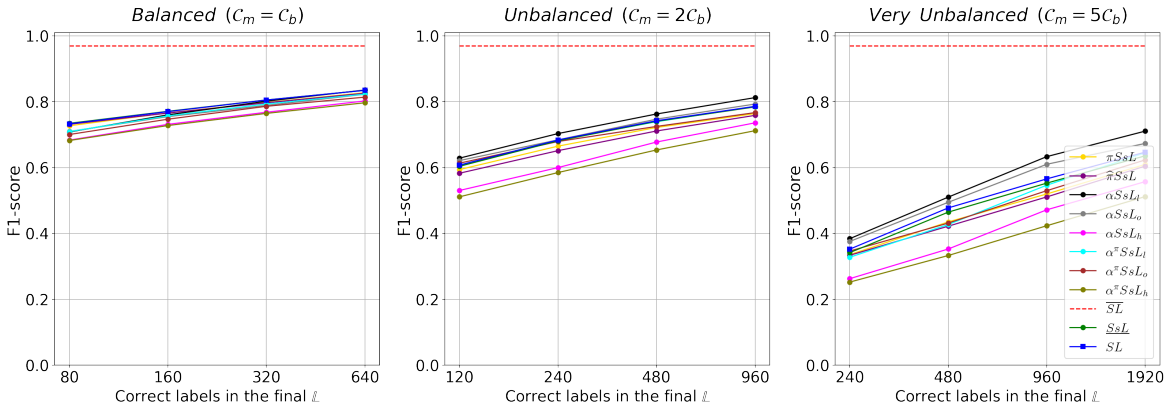


(b) *Recall* on **DREBIN**. For every plot, each 'point' represents the average *Recall* of 100 models (and 5 times as much for all models using active learning).

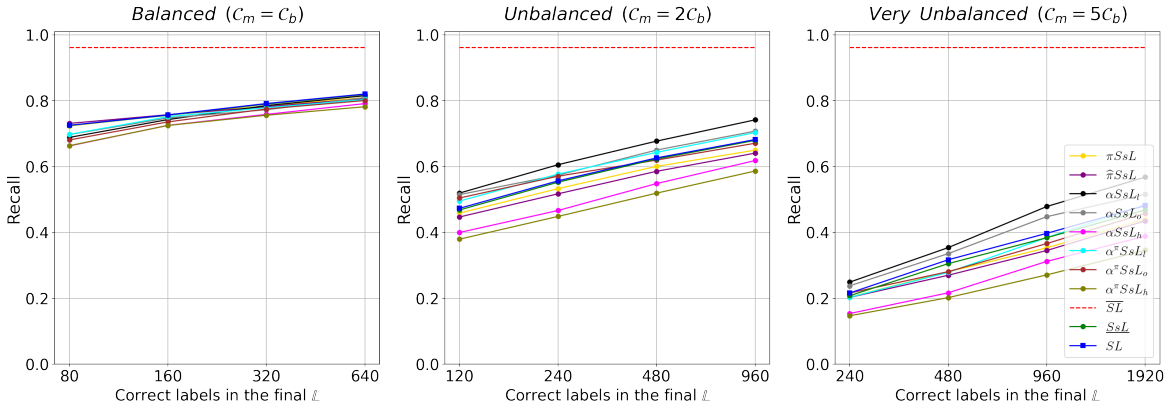


(c) *Precision* on **DREBIN**. For every plot, each 'point' represents the average *Precision* of 100 models (and 5 times as much for all models using active learning).

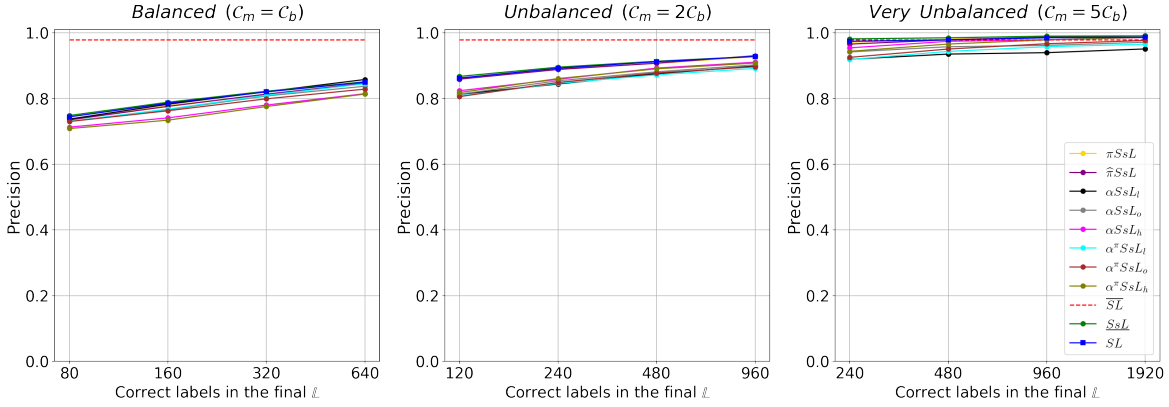
Figure 5: **Malware Detection**—DREBIN. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **Ember**. For every plot, each 'point' represents the average *F1-score* of 100 models (and 5 times as much for all models using active learning).

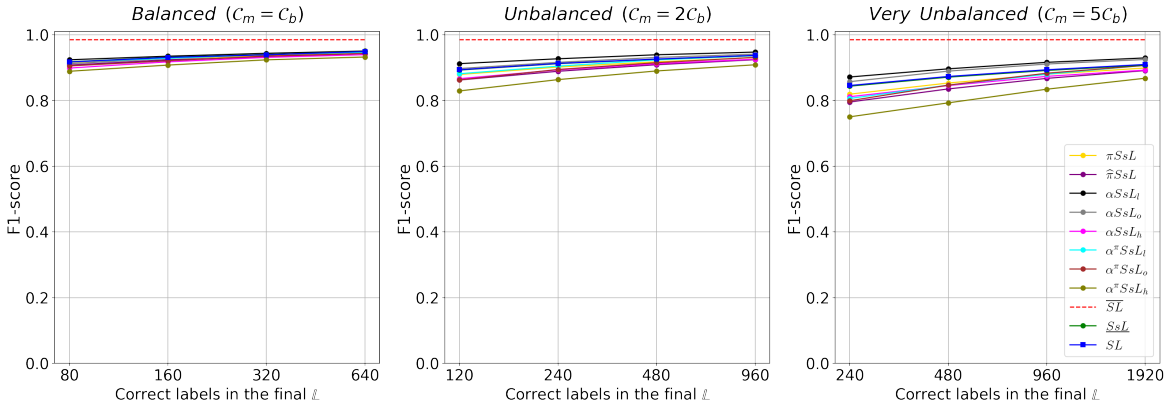


(b) *Recall* on **Ember**. For every plot, each 'point' represents the average *Recall* of 100 models (and 5 times as much for all models using active learning).

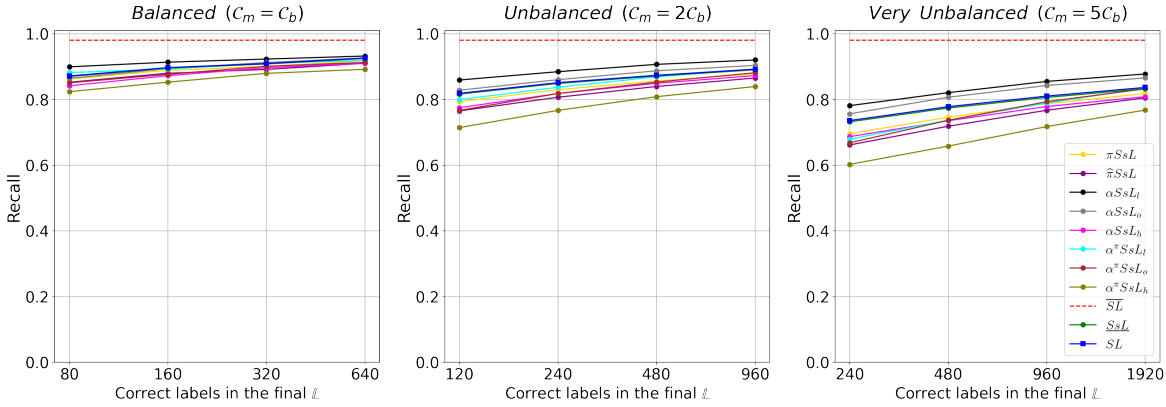


(c) *Precision* on **Ember**. For every plot, each 'point' represents the average *Precision* of 100 models (and 5 times as much for all models using active learning).

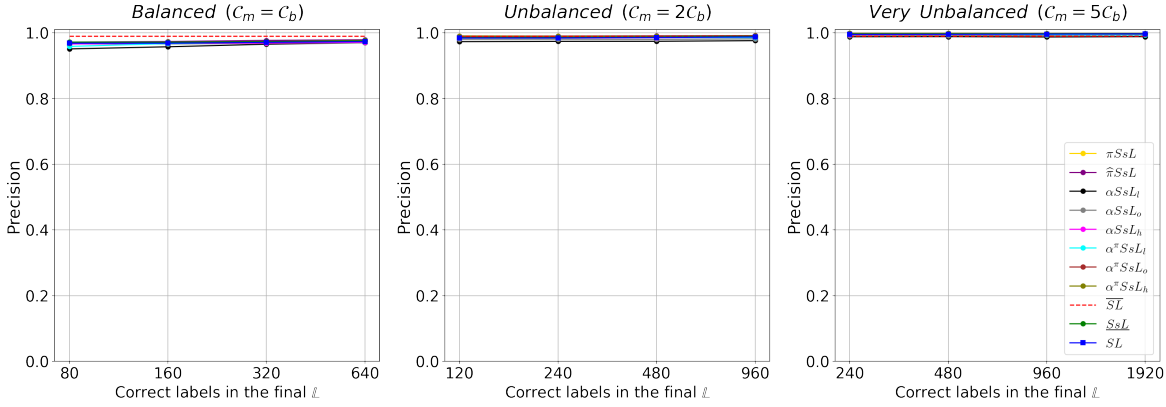
Figure 6: **Malware Detection**—`Ember`. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **AndMal20**. For every plot, each 'point' represents the average *F1-score* of 100 models (and 5 times as much for all models using active learning).

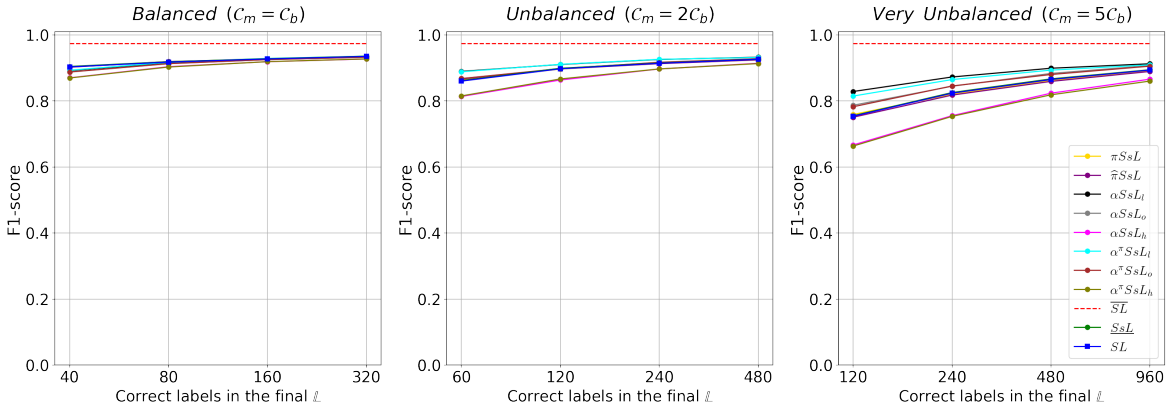


(b) *Recall* on **AndMal20**. For every plot, each 'point' represents the average *Recall* of 100 models (and 5 times as much for all models using active learning).

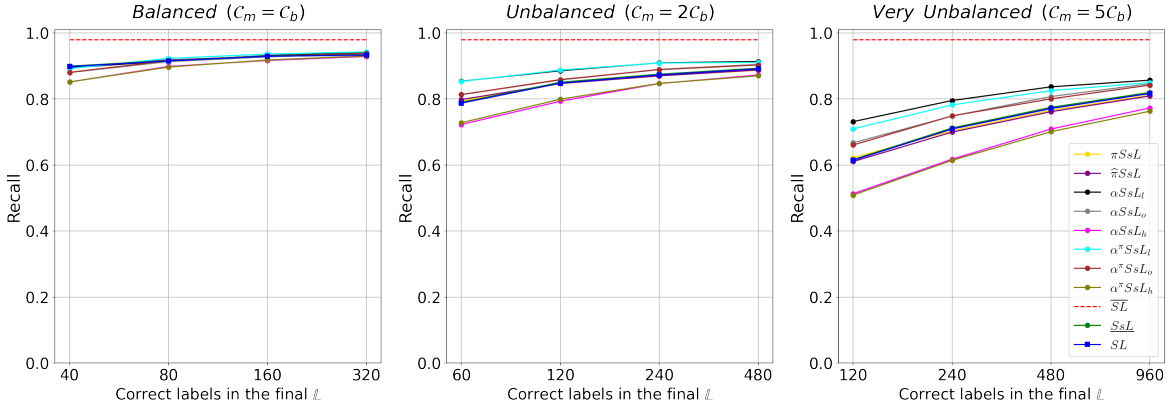


(c) *Precision* on **AndMal20**. For every plot, each 'point' represents the average *Precision* of 100 models (and 5 times as much for all models using active learning).

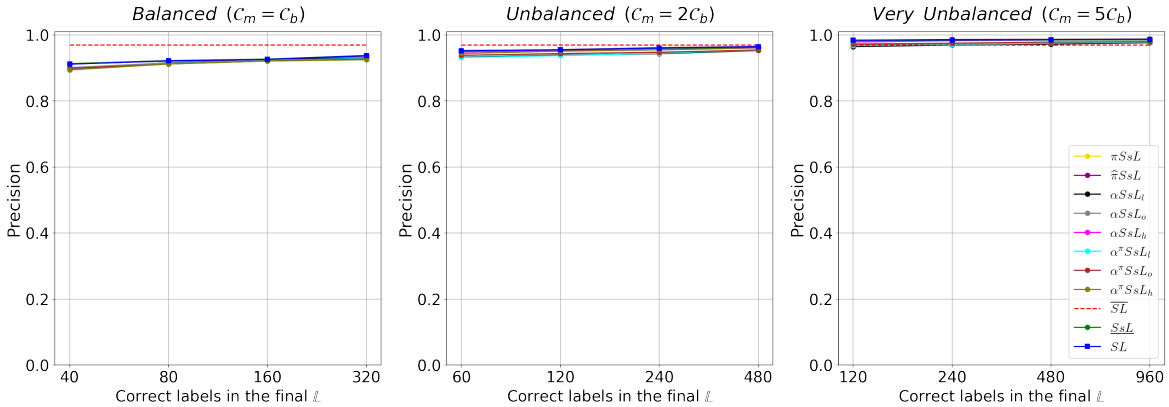
Figure 7: **Malware Detection**—AndMal20. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) F1-score on **UCI**. For every plot, each 'point' represents the average F1-score of 100 models (and 5 times as much for all models using active learning).

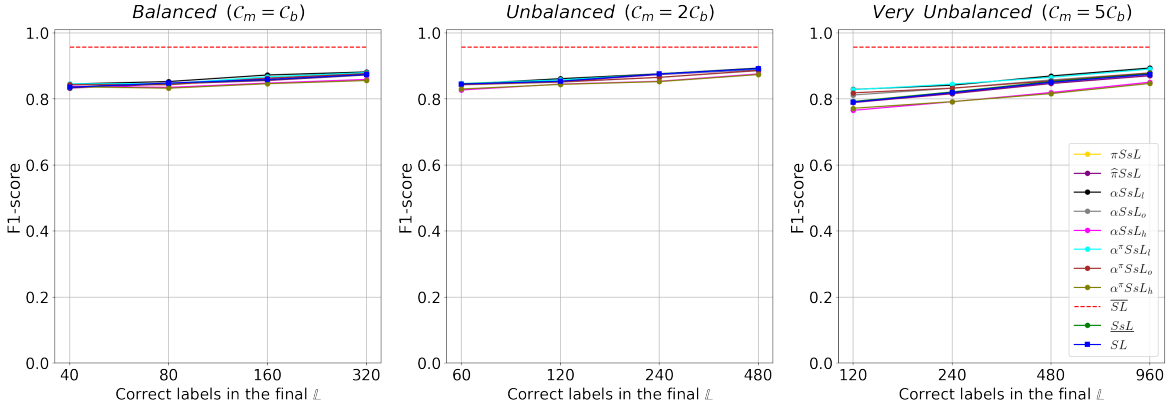


(b) Recall on **UCI**. For every plot, each 'point' represents the average Recall of 100 models (and 5 times as much for all models using active learning).

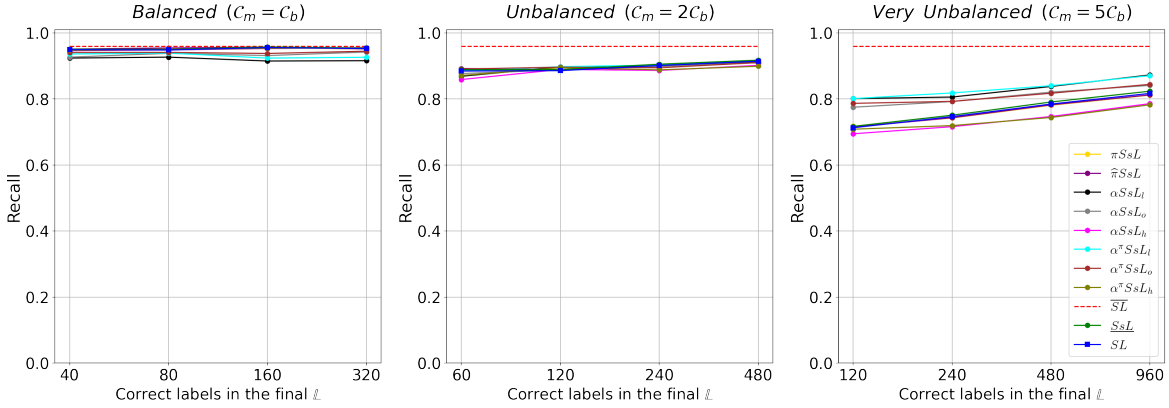


(c) Precision on **UCI**. For every plot, each 'point' represents the average precision of 100 models (and 5 times as much for all models using active learning).

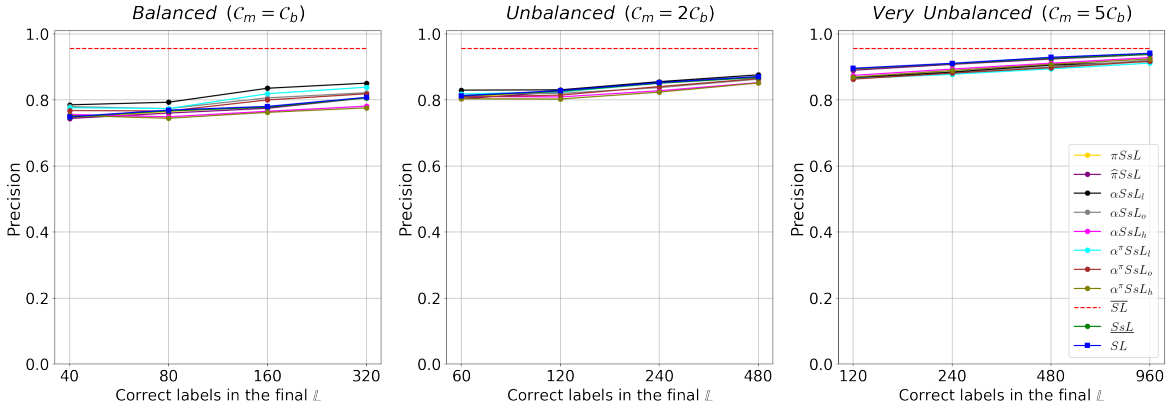
Figure 8: **Phishing Website Detection**—UCI. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on **Mendeley**. For every plot, each 'point' represents the average *F1-score* of 100 models (and 5 times as much for all models using active learning).

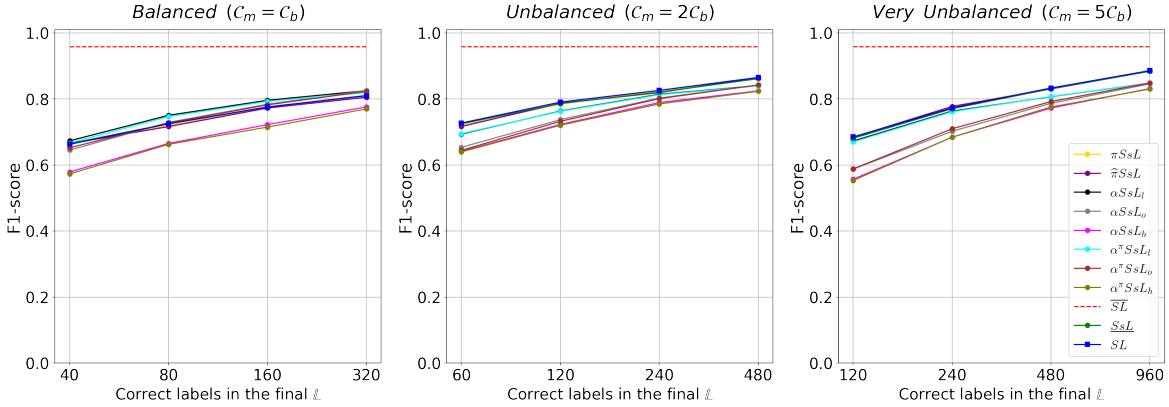


(b) *Recall* on **Mendeley**. For every plot, each 'point' represents the average *Recall* of 100 models (and 5 times as much for all models using active learning).

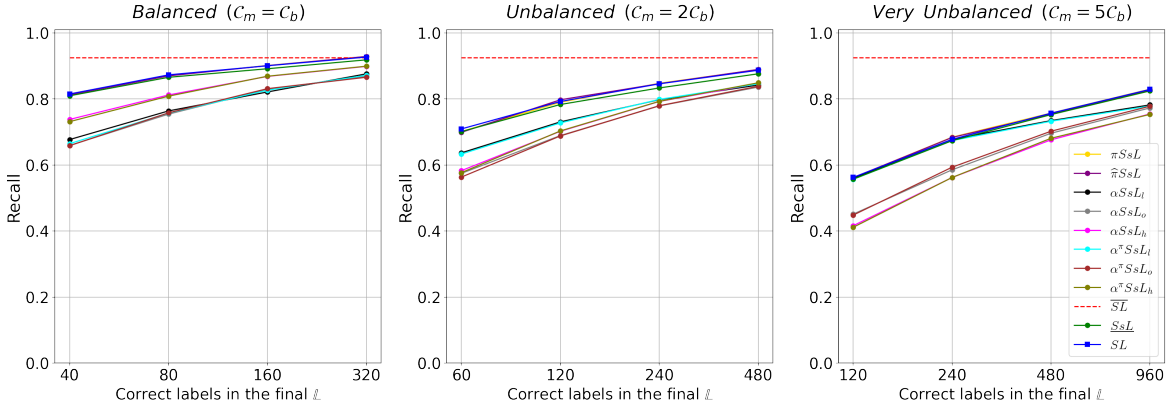


(c) *Precision* on **Mendeley**. For every plot, each 'point' represents the average *Precision* of 100 models (and 5 times as much for all models using active learning).

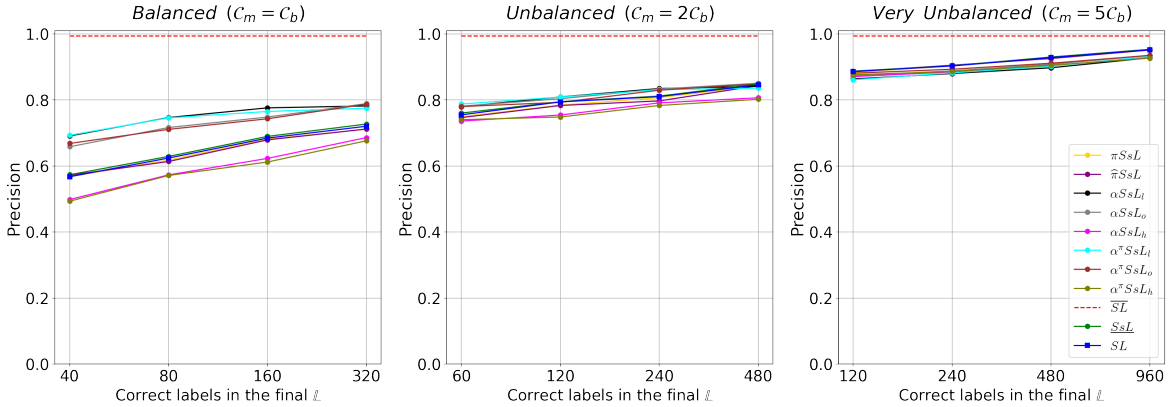
Figure 9: **Phishing Website Detection**—`Mendeley`. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.



(a) *F1-score* on δ**Phish**. For every plot, each 'point' represents the average *F1-score* of 100 models (and 5 times as much for all models using active learning).



(b) *Recall* on δ**Phish**. For every plot, each 'point' represents the average *Recall* of 100 models (and 5 times as much for all models using active learning).



(c) *Precision* on δ**Phish**. For every plot, each 'point' represents the average *Precision* of 100 models (and 5 times as much for all models using active learning).

Figure 10: **Phishing Website Detection**—δPhish. We report the performance for the three cost (or balancing) scenarios. Each scenario is shown in a plot, where the y-axis denotes a performance metric and the x-axis the (increasing) labelling budget. Each method is denoted with a line on each plot.