History Dependent Domain Adaptation

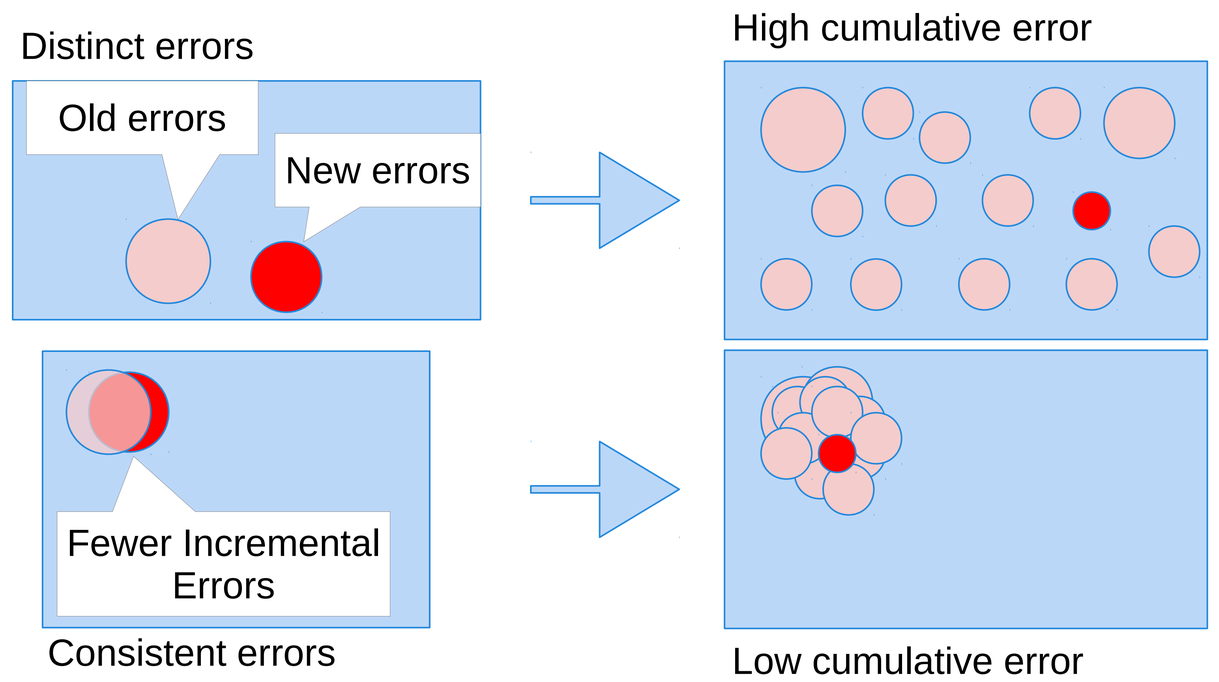
Allen Lavoie*, Matthew Eric Otey**, Nathan Ratliff**, D. Sculley**
*Rensselaer Polytechnic Institute
**Google Pittsburgh

Problem

How do we learn when our loss function depends on previous classifications, or on the correctness of previous classifications?

When humans manually correct misclassifications, there is a low cost associated with repeating errors; we can simply remember the human's label. However, human corrections take time, and reviewing every classification can be too expensive. Our aim is to learn while minimizing new errors, even if we don't know which classifications are errors.

In many large scale machine learning deployments, classification or regression is a service. These systems have an expectation of consistency and adaptability. Current machine learning research focuses on the latter: how well can we classify *now*? This work makes the trade-off explicit, and shows that major gains in consistency are possible without sacrificing adaptability.



Solutions

Averaging

$$w \leftarrow \alpha\, w_{t+1} + (1-\alpha)\, w$$

Average weights or model outputs (equivalent in the linear case). A linear combination of previous hypotheses gives us a simple baseline for comparison. Exponential averaging is extremely easy to implement.

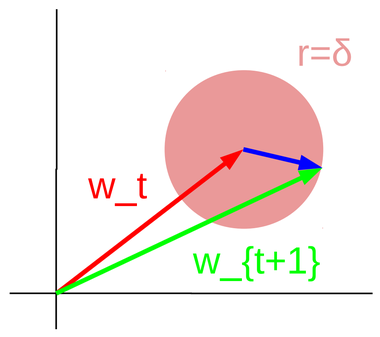
Warm start

Reduce divergence from previous hypotheses by using a small step size, or by taking fewer steps. In general, we might use an online learning algorithm.

Weight nearness constraint

$$\|w_{t+1} - w_t\| \leq \delta$$

Full optimization, with a hard constraint.



Prediction regularization

$$\sum_x \left(\langle w_{t+1}, x\rangle - \langle w_t, x\rangle\right)^2$$

$$\sum_x max\left(0, 1 - h_t(x)\, w_{t-1}^T x\right)$$

Add a regularization term which penalizes the model for differing from the previous model. The hinge loss term is equivalent to adding extra weighted examples to the data set.

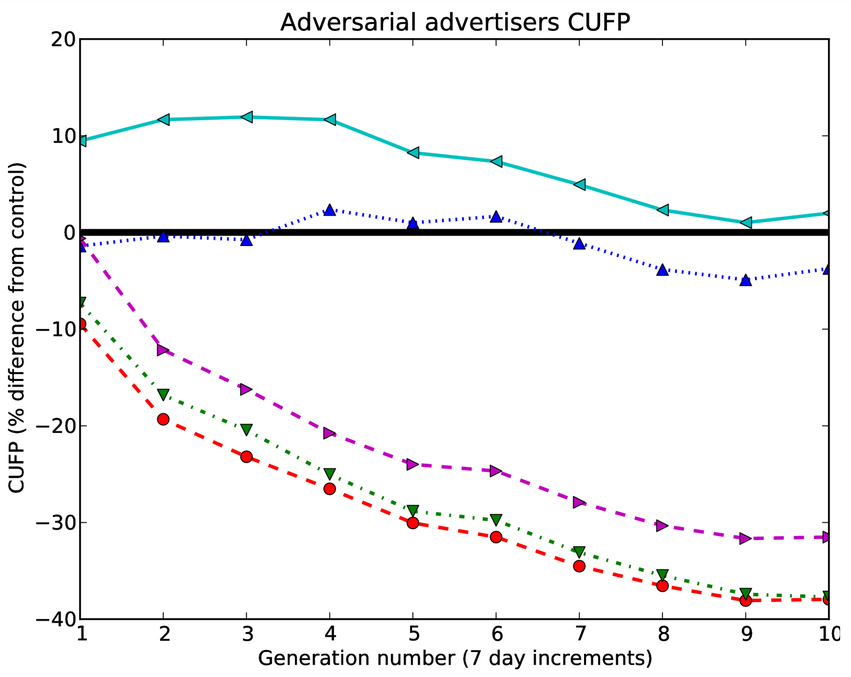
Evaluation

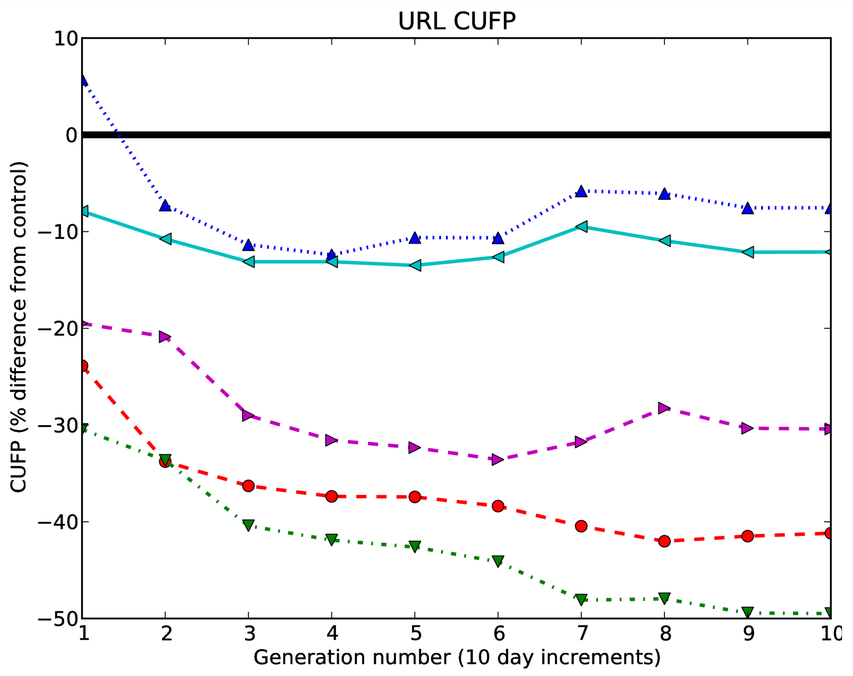
Metrics

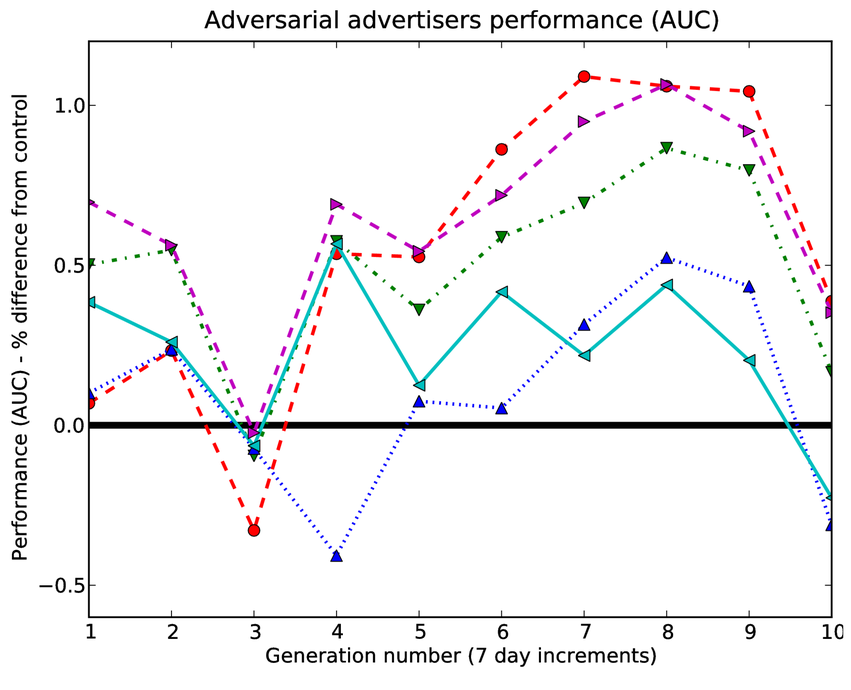
- Area under the ROC curve (AUC)
  - Instantaneous performance
  - We want to avoid decreasing this too much
- Cumulative Unique False Positives (CUFP)
  - Overall performance
  - Number of examples misclassified at least once

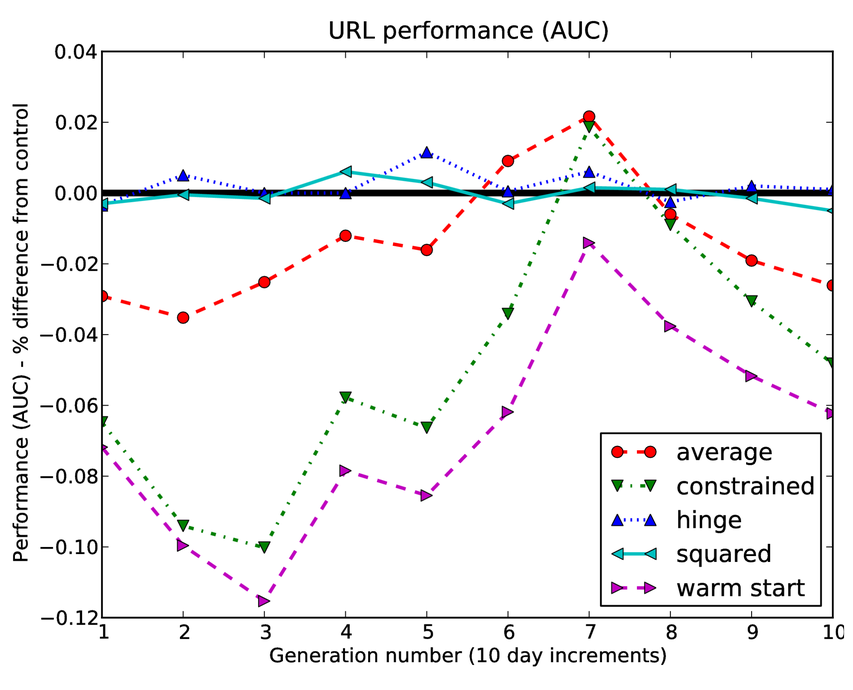
Data

- Adversarial advertisements (Sculley 2011)
  - Adversarial (positive) or non-adversarial (negative)
  - Sparse, high-dimensional
- Malicious URL Identification (Ma 2009)
  - Malicious (positive) or non-malicious (negative)
  - Qualitatively similar, public









Results

We see up to a 50% reduction in CUFP, with only a very minor reduction in AUC (0.04%)! Warm start, the weight nearness constraint, and averaging all performed quite well.

Relatively simple methods can drastically improve consistency. Can we do better? Why do some methods work better than others? Can we make use of unlabeled data?