ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

Temporal Transfer Learning for Drift Adaptation

Won, Daegun and Jansen, Peter J. and Carbonell, Jaime G.

Carnegie Mellon University - Language Technologies Institute
Pittsburgh, PA, USA

Abstract. Whereas detecting and adapting to concept drift has been well studied, predicting temporal drift of decision boundaries has received much less attention. This paper proposes a method for drift prediction, drift projection, and active-learning for adjusting the projected decision boundary so as to regain accuracy with minimal additional labeled samples. The method works with different underlying learning algorithms. Results on several data sets with translational and rotational drift and corresponding boundary projection show regained accuracy with significantly fewer labeled samples, even in the presence of noisy drift.

1 Introduction: (Semi-) Predictable Concept Drift

Temporal concept drift abounds. Consider, for instance, the concept of "a powerful laptop" ten years ago compared to the present day. Processor performance, number of cores, etc., that would have qualified as powerful ten years ago are considered quaint or under-powered today. For smart phones the evolution is even more striking and rapid. Clothing considered fashionable years ago may be considered passé today. A company that was financially sound a few years ago might now be heading for Chapter 11 due to e-commerce competition. While the above examples entail massive concept drift, smaller and subtler drift is also common, and all drift requires updating models with additional labeled training data in order to maintain accuracy.

Concept drift has been widely studied in the literature (e.g., [1, 2, 3, 4, 5, 6]). Transfer learning [7] has been considered for drift tracking and adaptation [2], if in adversarial setting. To our knowledge, however, existing drift recovery methods are not based on predicting properties of the evolving decision boundaries. Instead, some make use of assumptions such as *bounded drift* (see below) to regain lost classification accuracy faster through additional sampling of newly labeled data. While useful and effective, this fails to exploit the presence of predictable or semi-predictable drift.

More formally, consider a supervised learning setting where the set of classifiers F map input attributes $\{x_1, x_2, \dots, x_n\}$ into the correct output class $y_j \in Y$. We can write this as: $F : \vec{X_i} \rightarrow Y$. A trained classifier includes model parameters $\vec{\theta} = \{\theta_1, \theta_2, \dots, \theta_k\}$. These parameters are, for instance, the coefficients in logistic regression, or the weights on the links between layers of a neural network. Hence we slightly expand the above to: $F_\theta : \vec{X_i} \rightarrow Y$,

For temporal drift, a classifier trained in earlier epochs may re-estimate different values for its parameters at each time step to regain classification accuracy. The model parameters are therefore time-dependent: $F_{(\theta(t))} : \vec{X_i} \rightarrow Y$.

If we can assume that the drift is bounded – that is, the change in the parameter values is small in adjacent time steps – we may bound the total change in model parameters, for any norm p, by: $\frac{1}{k}\sum_{i=1}^{k}\|\theta_i(t)-\theta_i(t+1)\|_p < \delta_{max\text{-}drift}$. Alternatively, we can incorporate the above expression as a regularization term in the loss function when re-estimating the model parameters at time $t+1$.

When dealing with *non-predictable concept drift*, the best we can do is to use the previous model parameters at time t, as the best initial estimate for time $t+1$. In other words, we are simply projecting $\theta(t) \rightarrow \hat{\theta}(t+1)$ assuming the identity projection function. However, if the drift is at least *partially predictable*, we may do better, assuming a projection function $\pi(\theta(t)) \rightarrow \hat{\theta}(t+1)$, which gives us a better estimate of $\hat{\theta}(t+1)$ prior to active learning.

Note that if $\pi(\theta(t))$ is a better approximation for $\hat{\theta}(t+1)$ than $\theta(t)$, then the empirical loss of the classifier based on projecting the drift is smaller than the loss of simply starting with the previous model parameters, i.e.:

$$\sum_{i=1}^{n}\|y_{i,t+1} - f_{\pi(\theta_t)}(\overrightarrow{x_i})\|_p < \sum_{i=1}^{n}\|y_{i,t+1} - f_{\theta_t}(\overrightarrow{x_i})\|_p$$

The above means that fewer additional labeled training instances should be required at each time interval to restore accuracy after drift by projecting the direction and magnitude of said drift. As we see below in our experimental results, this expectation is indeed borne out.

For simplicity, we assume that the resulting classification boundary can be approximated by a linear hyperplane subject to drift in its parameters, as illustrated in Fig. 1. Our system (dubbed ADAPT[1]), designed to track and project such boundaries, is described in section 2. Experiments on multiple data sets with synthetic drift are described in section 3.

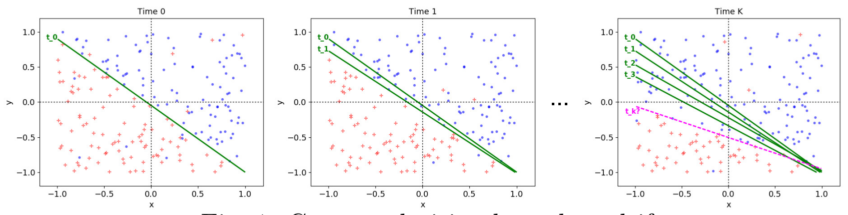

Fig. 1: Concept decision-boundary drift.

2 Temporal Transfer Learning for drift prediction

Our ADAPT system consists of three major components, depicted in Figure 2. The *drift simulator* relabels instances at each time point, according to a programmable drift function that reflects the changes on the decision boundary, along with parameterized noise insertion. For data with true temporal drift, this

[1]ADAPT: Automated Drift-Aware Projection and Transfer

simulator would be unnecessary. The *drift predictor* contains a model trained jointly over all the observed labeled instances, including those from past time steps, to model the drift. It essentially extrapolates near-future drift based on recent drift history. The *target classifier* learns the expected decision boundary first at time 0 (from an initial training set), and then incorporates the drift prediction and incrementally adjusts the decision boundary via active learning.

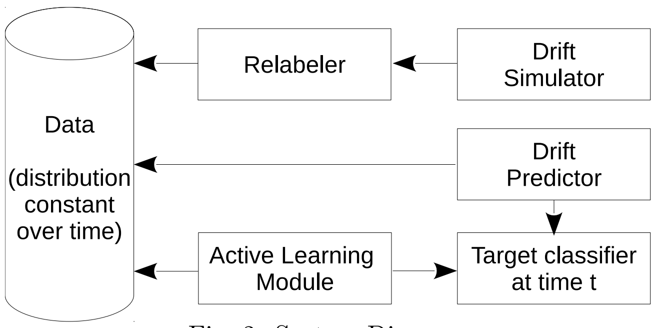

Fig. 2: System Diagram

The drift predictor, which is our primary unique contribution, uses a joint model optimizing the following objective function at each time t:

$$\begin{aligned} L_{dp}(D^0,\ldots,D^t|\overrightarrow{w_0},c_0,\delta\overrightarrow{w},\delta c) = &\sum_{i=0..t} L_{LR}(D^i|\overrightarrow{w_0}+i\cdot\delta\overrightarrow{w},c+i\cdot\delta c) \\ &+ \lambda_1\|\overrightarrow{w_0}\|^2+\lambda_1\|\overrightarrow{w_0}+t\cdot\delta\overrightarrow{w}\|^2+\lambda_2\|\delta\overrightarrow{w}\|^2 \end{aligned} \quad (1)$$

where D^i is the training set used at time i and $L_{LR}(D|\overrightarrow{w},c)$ is the loss function for logistic regression with parameters $\overrightarrow{w},c$. This function lets us find the best set of base parameters and their increments by joint optimization. For the *target classifier* at each time point, we use regularized logistic regression. Rather than just projecting model parameters for time $t+1$, we use uncertainty-based active learning [8, 9]. As specified in Algorithm 1, whenever a new active learning phase starts, the *drift prediction* $\theta_{h'} = (\overrightarrow{w}', c')$ is added as a prior in the regularization term of the following objective function to transfer the knowledge:

$$L_{TransferLR}(D|\overrightarrow{w},c;\overrightarrow{w}',c') = L_{LR}(D|\overrightarrow{w},c)+\lambda\|\overrightarrow{w}-\overrightarrow{w}'\|^2.$$

3 Experiments

Data: We used five UCI binary prediction datasets[2] to evaluate our algorithm. As these datasets do not have time stamp information, we used our drift simulator component, for which drift and noise parameters can be specified, to augment the (real) data with simulated drift.

[2]Available from https://archive.ics.uci.edu/ml/datasets.html.

```
input: An unlabeled dataset $X = \{x_i\}$
        // Requires drift simulator or data with drift
        Drift Simulator: $\pi_s : \mathbb{N} \to \mathbb{H}$, Drift Predictor: $\pi_p : \mathbb{N} \to \mathbb{H}$
        Required number of periods to observe drift: $t_d$
        Target classifier: $C$, Labeling budget: $B$
for $t \leftarrow 0$ to $T$ do
    $h^t \leftarrow \pi_s(t)$ // Acquire / Generate drift
    $Y^t \leftarrow \{y_i^t \leftarrow h^t(x_i) \quad \forall x_i \in X\}$ // Relabel w.r.t the new boundary
    Initialize $D_l^t$ with a few randomly chosen instances from $D^t \leftarrow \{(x_i, y_i^t)\}$
    $D_u^t \leftarrow D^t \setminus D_l^t$
    // Predict new concept boundary
    if $t > t_d$ then $h' \leftarrow \pi_p(t)$ else $h' \leftarrow$ None
    for $i \leftarrow 0$ to $B$ do
        Train $C$ with $D_l^t, \theta_{h'}$
        Sample $(x, y) \in D_u^t$ // Sample via active learning
        $D_u^t \leftarrow D_u^t \setminus \{(x, y)\}$
        $D_l^t \leftarrow D_l^t \cup \{(x, y)\}$
    end
    $D_{all} \leftarrow D_{all} \cup \{D_l^t\}$
    Update $\pi_p$ with $D_{all}$ // Update drift predictor
end
```

Algorithm 1: Overall system flow

Assumptions and Experimental setup: We assume the distribution of the feature vectors is stationary and that the labels, i.e. the conditional probabilities, are changing. The predicted parameters of a Logistic Regression classifier are used to transfer knowledge from prior epochs when available. We set the final target decision boundary by adding relative Gaussian noise sampled at $\mathcal{N}(0.25 \cdot \overrightarrow{\mathbf{1}}, 0.25 \cdot \mathbb{I})$ to the decision boundary of the original data set. The simulated drift is linear, both rotational and translational, with Gaussian noise.

At each time t, we use active learning with uncertainty sampling (with budgets according to dataset as listed in Table 1). As a few drift measurements are needed to establish the linear-transformation extrapolation, we start projecting future drift only after a warm-up period of t_d epochs. A setting of $t_d = 4$ appears satisfactory (optimizing this parameter is left for future work). To simulate the presence of noise, the drift simulator (1) adds a relative 5% Gaussian noise to the parameters of the new boundary, and (2) flips the labels on the samples with a $0.1 \times P(\sim Y|X)$ probability.

The results are compared with an active-learning baseline that transfers the knowledge from the previous time step with an identity projection. To quantify the performance differences, we used the following metrics:

- *Area under the Learning Curve (ALC)*: This estimates how the projected drift priors affect the performance of the system throughout its lifetime.
- *Recovery speed*: Lx = Number of Labeled instances needed to achieve a (pre-specified) x% of the highest accuracy during the previous time period. This metric is directly relevant, but not always applicable.

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

Results: Figure 3 shows the learning curve for ADAPT and the baseline, for the KR-vs-KP (Chess) dataset, in the case of added noise (the figure without noise is nearly identical). As soon as drift predictions are included (from $t = 5$ onward), ADAPT outperforms the baseline in both cases. Table 1 summarizes the performance comparison under simulated noise on all the datasets we used. For the `waveform` dataset, where we simplified the task to a binary classification using the first two classes only, our measure $L95$ did not yield a significant difference, hence we also include $L97$. On average, our method needs 20% to 60% fewer labels to recover accuracy.

	Budget	ADAPTd (LR)		Baseline (LR)	
		ALC	$L95$	ALC	$L95$
`Ionosphere`	80	**497.81**	**8.57**	492.90	14.71
`KR-vs-KP`	100	**633.12**	**23.14**	621.23	40.00
`Magic04`	100	**654.09**	**3.43**	649.04	7.14
`Sonar`	60	**365.89**	**8.86**	358.34	11.29
`Waveform`	100	**557.27**	3.17 (L97 **4.33**)	553.74	**2.5** (L97 11.33)

Table 1: Area under the Learning Curve(ALC) and Recovery Speed($L95$) comparison between ADAPT and the baseline after time 3 (Logistic Regression)

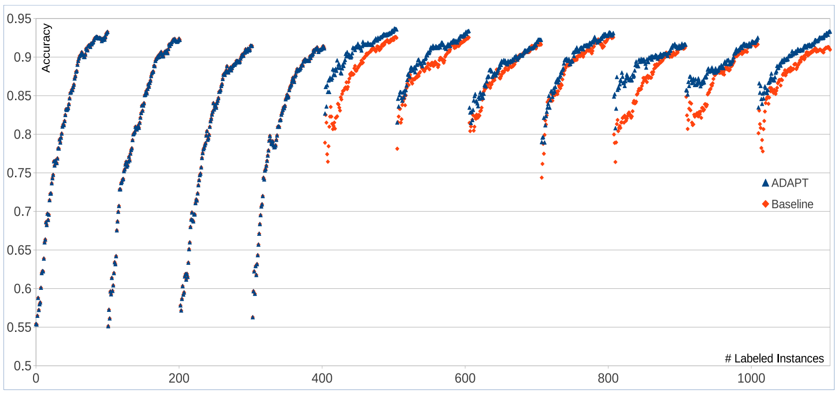

Fig. 3: Accuracy for kr-vs-kp dataset with noise in both drift and labels

Validation on an alternative classifier: Instead of a logistic regression learner, we also implemented and tested an SVM learner within the ADAPT framework on the same data. While SVM appears to take less advantage of our temporal transfer learning, in most cases ADAPT clearly outperforms the (SVM) baseline, as illustrated in Figure 4.

4 Conclusions and Future Work

We have proposed a method for drift prediction and tracking and shown that our implementation ('ADAPT') requires significantly fewer samples to regain accu-

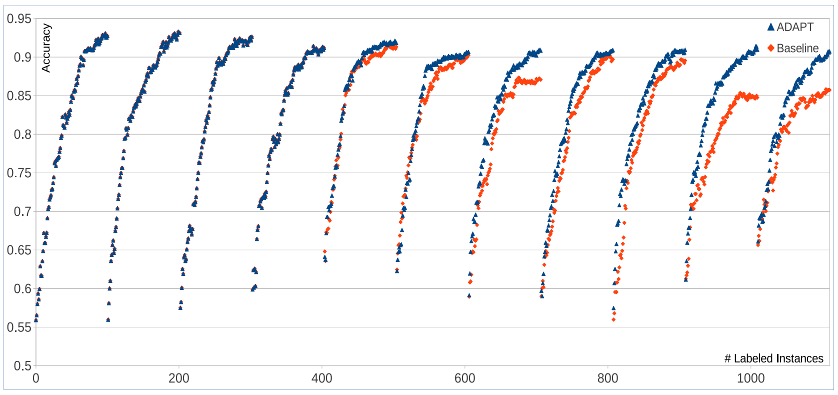

Fig. 4: Accuracy for kr-vs-kp dataset (SVM)

racy than an adaptive baseline in the case of linear translational and rotational drift, both in the absence and the presence of noise, on several UCI datasets, for two different underlying machine learning methods (logistic regression and SVM). We plan to extend ADAPT to handle non-linear (quadratic, cyclic) drift, as well as validate the techniques on large real-life datasets (e.g.[4]).

References

[1] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46(4):44, 2014.

[2] Michal Dereziński and Badri Narayan Bhaskar. Anticipating concept drift in online learning. In *Advances in Neural Information Processing Systems*, 2015.

[3] John Case, Sanjay Jain, Susanne Kaufmann, Arun Sharma, and Frank Stephan. Predictive learning models for concept drift. *Theoretical Computer Science*, 268(2):323–349, 2001.

[4] Indrė Žliobaitė, Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. MOA concept drift active learning strategies for streaming data. In *JMLR: Workshop and Conference Proceedings 17*, 2nd Workshop on Applications of Pattern Analysis, pages 48–55, 2011.

[5] Indrė Žliobaitė, Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. Active learning with drifting streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):27–39, 2014.

[6] Yu Sun, Ke Tang, Zexuan Zhu, and Xin Yao. Concept drift adaptation by exploiting historical knowledge. *CoRR*, abs/1702.03500, 2017.

[7] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, October 2010.

[8] Y. Baram, R. El-Yaniv, and K. Luz. Online choice of active learning algorithms. In *Proceedings of ICML '03*, pages 19–26. ICML, 2003.

[9] Pinar Donmez, Jaime G Carbonell, and Paul N Bennett. Dual strategy active learning. In *Machine Learning: ECML 2007*, pages 116–127. Springer, 2007.