Learning from Delayed Outcomes via Proxies with Applications to Recommender Systems

Timothy Mann Swen Goval **András György** Ray Jiang
Huiyi Hu Balaji Lakshminarayanan Prav Srinivasan

DeepMind

ICML 2019
Poster #160

Motivating Example: Book Recommendation

- Task: Recommend books to customers.
- Success: Book is read in 90 days from purchase.
- Learning continuously (online) about new books is necessary.

  $\Rightarrow$ Waiting 90 days for the feedback is infeasible.

This is done by online learning algorithms in the literature (Weinberger and Ordentlich, 2002; Mesterharm, 2005; Joulani et al., 2013; Quanrud and Khashabi, 2015).

Motivating Example: Book Recommendation

- Task: Recommend books to customers.
- Success: Book is read in 90 days from purchase.
- Learning continuously (online) about new books is necessary.

  $\Rightarrow$ Waiting 90 days for the feedback is infeasible.

  This is done by online learning algorithms in the literature (Weinberger and Ordentlich, 2002; Mesterharm, 2005; Joulani et al., 2013; Quanrud and Khashabi, 2015).

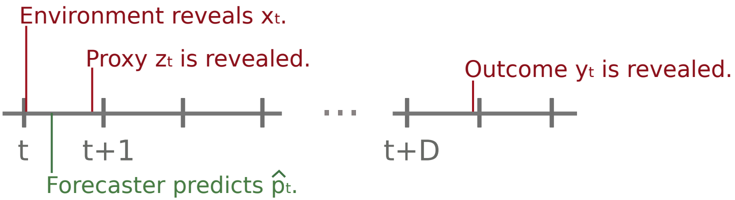
Our approach: Use less-delayed proxy information.

- E.g., if the customer starts reading the book on the first day.

Idealized Formal Model

Online learning problem:

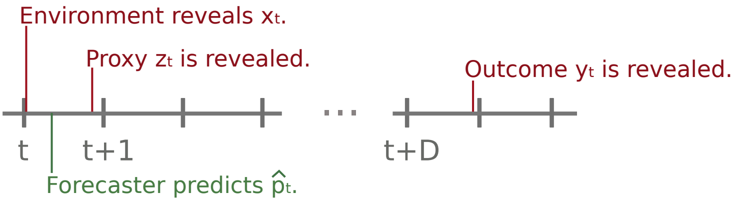
- Instance $x_t \in \mathcal{X}$ is revealed.
- Forecaster predicts an outcome distribution $\hat{p}_t$.
- Proxy $z_t \in \mathcal{Z}$ is revealed.
- Outcome $y_t \in \mathcal{Y}$ is revealed after delay D.



Idealized Formal Model

Online learning problem:

- Instance $x_t \in \mathcal{X}$ is revealed.
- Forecaster predicts an outcome distribution $\hat{p}_t$.
- Proxy $z_t \in \mathcal{Z}$ is revealed.
- Outcome $y_t \in \mathcal{Y}$ is revealed after delay D.



Goal: minimize regret (for the log-loss)

$$\mathbb{E}\left[\mathrm{Regret}_T\right] = \min_p \mathbb{E}\left[\sum_{t=1}^{T} \Big(\log p(y_t|x_t) - \log \hat{p}_t(y_t|x_t) \Big)\right].$$

Theoretical Analysis

Assumptions:

- $x_1, x_2, \ldots, x_T$ are selected by an oblivious adversary.
- Factored model:
  - $z_t \sim h(\cdot|x_t)$ independently;
  - $y_t \sim g(\cdot|z_t)$ independently.
- Optimal forecaster is factored: $p(y|x) = \sum_z g(y|z)h(z|x)$.

Theoretical Analysis

Assumptions:

- $x_1, x_2, \ldots, x_T$ are selected by an oblivious adversary.
- Factored model:
  - $z_t \sim h(\cdot|x_t)$ independently;
  - $y_t \sim g(\cdot|z_t)$ independently.
- Optimal forecaster is factored: $p(y|x) = \sum_z g(y|z)h(z|x)$.

Idea:

- Use factored forecasters $\hat{p}_t(y|x_t) = \sum_z g_t(y|z)h_t(z|x_t)$.
- Learn g_t and h_t separately.

Theoretical Analysis

Assumptions:

- $x_1, x_2, \ldots, x_T$ are selected by an oblivious adversary.
- Factored model:
  - $z_t \sim h(\cdot|x_t)$ independently;
  - $y_t \sim g(\cdot|z_t)$ independently.
- Optimal forecaster is factored: $p(y|x) = \sum_z g(y|z)h(z|x)$.

Idea:

- Use factored forecasters $\hat{p}_t(y|x_t) = \sum_z g_t(y|z)h_t(z|x_t)$.
- Learn g_t and h_t separately.

Theorem. Using the Laplace or Krichevsky-Trofimov estimators for g_t and h_t,

$$\mathbb{E}\left[\mathrm{Regret}_T\right] = O\big((D + |\mathcal{X}|)\log T\big).$$

Theoretical Analysis

Assumptions:

- $x_1, x_2, \ldots, x_T$ are selected by an oblivious adversary.
- Factored model:
  - $z_t \sim h(\cdot|x_t)$ independently;
  - $y_t \sim g(\cdot|z_t)$ independently.
- Optimal forecaster is factored: $p(y|x) = \sum_z g(y|z)h(z|x)$.

Idea:

- Use factored forecasters $\hat{p}_t(y|x_t) = \sum_z g_t(y|z)h_t(z|x_t)$.
- Learn g_t and h_t separately.

Theorem. Using the Laplace or Krichevsky-Trofimov estimators for g_t and h_t,

$$\mathbb{E}\left[\mathrm{Regret}_T\right] = O\big((D + |\mathcal{X}|)\log T\big).$$

Without proxy,

$$\mathbb{E}\left[\mathrm{Regret}_T\right] = \Omega(D|\mathcal{X}|\log T)$$

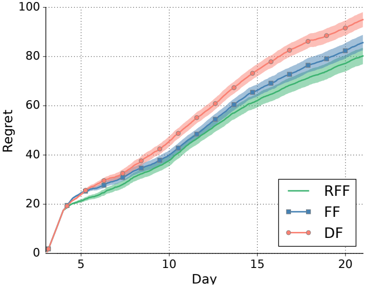
(Weinberger and Ordentlich, 2002).

Correcting for Modeling Errors

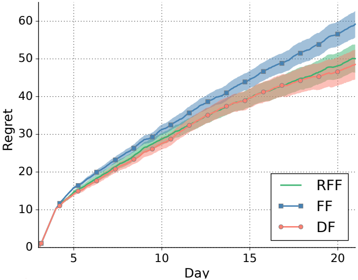
- Factorization assumption does not hold in practice.
- Practical solution: neural-network-based Factored Forecaster with Residual correction (RFF)

Correcting for Modeling Errors

- Factorization assumption does not hold in practice.
- Practical solution: neural-network-based Factored Forecaster with Residual correction (RFF)
- Real world data with delayed proxies



(a) Staggered schedule



(b) Uniform schedule

DF: Direct forecaster FF: Factored forecaster

Summary

- Less-delayed proxies in online learning with delayed feedback are useful.
- Contributions:
  - Theoretical analysis.
  - A factored neural network forecaster that works well in practice.
- Learn more about theory, algorithms, experiments:

Come to Poster #160!