

A dimly lit office interior with large windows overlooking a city skyline. The windows are divided into vertical panes. Outside, a cityscape is visible under a cloudy sky, with a prominent church spire in the center. Inside the office, desks with computer monitors and chairs are visible in the foreground and middle ground, though they are dark and out of focus.

Online and out-of-core learning an FTRL perspective

Kristian Holsheimer | 2018-03-27

Q What to do when your dataset doesn't fit in memory?

... especially if you don't have access to a large data cluster.

What?

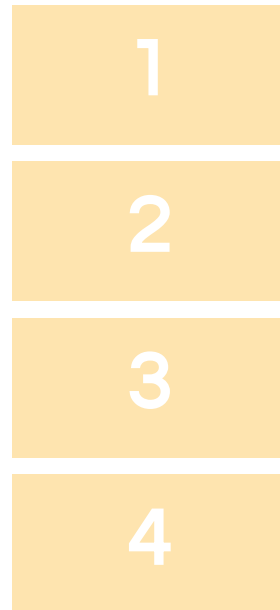
Q What to do when your dataset doesn't fit in memory?

A Train on smaller chunks and then combine models / parameters

batch



out-of-core



parallel



What?

Q What to do when your dataset doesn't fit in memory?

A Train on smaller chunks and then combine models / parameters

Parallel approach

- appears superior
- received a lot of attention
- needs a cluster of many machines (and infrastructure etc)
- feasible for large companies

Out-of-core approach

- appears inferior
- hasn't received the attention it deserves
- runs on a single machine
- fits small startups *and* large companies

this talk

Questions.

What is **out-of-core** learning?

What is **online** learning?

How do we apply it to ordinary **batch** learning?

Why would we expect this to work?

Can we do **better** than the naive approach?

Online-to-batch learning spotted **in the wild**?

→ *Vowpal Wabbit*

Program.

Out-of-core learning

Online learning

Online-to-batch conversion - [naive](#)

FTRL - Follow The (Regularized) Leader

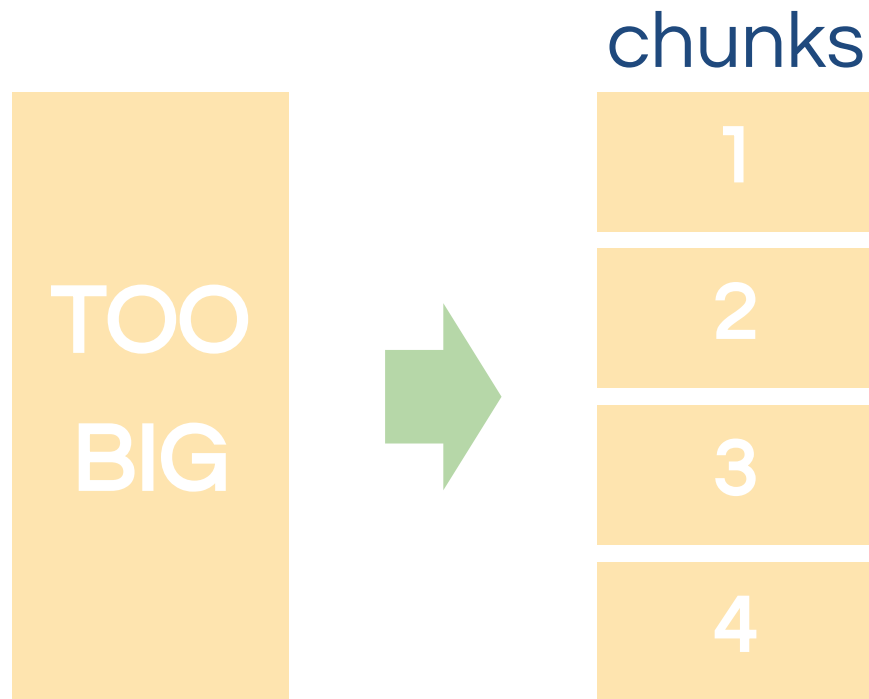
Online-to-batch conversion - [a closer look](#)

Business applications at Booking.com



Out-of-core learning

Out-of-core learning.



pseudo code

```
for x, y in chunks:  
    model.update(x, y)
```


Out-of-core learning.

Dataset

Suppose we have a fixed distribution:

$$\mathcal{D} = X \times Y$$

Training data consists of i.i.d. samples from \mathcal{D} :

$$(x, y) \sim \mathcal{D}$$

Therefore, training data set is an unordered sequence:

$$\mathcal{D}_{\text{train}} = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

Out-of-core learning.

Objective

Minimize expected loss:

$$L(\theta) = \mathbb{E}_{x,y \sim \mathcal{D}} \ell(x, y, \theta)$$

where ℓ is the **single-datapoint loss**, e.g. for linear regression:

$$\ell(x, y, \theta) = \frac{1}{2} (\theta \cdot x - y)^2$$

Out-of-core learning.

Gradient descent

$$\begin{aligned}\theta &\leftarrow \theta - \eta g(\theta) \\ g(\theta) &= \mathbb{E}_{x,y \sim \mathcal{D}} \nabla_{\theta} \ell(x, y, \theta)\end{aligned}$$

Ordinary (batch) gradient descent

$$g(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_{\theta} \ell(x_i, y_i, \theta)$$

Stochastic gradient descent

$$g(\theta) \approx \frac{1}{n} \sum_{i=1}^n \nabla_{\theta} \ell(x_i, y_i, \theta)$$

A dimly lit, modern interior space, possibly a lounge or study area. The room features a brick wall, a large potted plant, and a wooden table where two people are seated. The text "Online learning" is overlaid in white. The background includes a large window with a grid pattern, a wooden chair, and a small table with a lamp.

Online learning

Online learning set-up.

The data

- Data is **no longer i.i.d.**
- Sequence of observations is **ordered**

$$\mathcal{O}_t = (x_1, y_1), (x_2, y_2), \dots, (x_{t-1}, y_{t-1})$$

The objective

- Predict the next y_t given the next x_t as well as all observations \mathcal{O}_t so far:

$$\hat{y}_t = h(x_t, \theta_t)$$

Online learning.

Remember batch loss:

$$\text{Loss}(\theta) = \frac{1}{N} \sum_{n=1}^N \ell(x_n, y_n, \theta)$$

Instead, we **minimize Regret** (assume game *"not too unfair"*)

$$\text{Regret}(\vartheta, \theta_*) = \sum_{t=1}^T \left(\ell(x_t, y_t, \theta_t) - \ell(x_t, y_t, \theta_*) \right)$$

where

$$\vartheta = \{\theta_1, \dots, \theta_T\}$$

$$\theta_* = \arg \min_{\theta} \sum_{t=1}^T \ell(x_t, y_t, \theta)$$

Online learning.

Simplest approach: **Online Gradient Descent**

$$\begin{aligned}\theta_{t+1} &= \theta_t - \eta g_t \\ g_t &= \nabla_{\theta} \ell(x_t, y_t, \theta_t)\end{aligned}$$

where η can be simple learning rate schedule:

$$\eta = \frac{\alpha}{\sqrt{t}}$$

or adaptive per-coordinate learning rate (AdaGrad):

$$\eta_i = \frac{\alpha}{\sqrt{\sum_{s=1}^t g_{i,s}^2}}$$

Online learning.

When to use online learning?

- when consecutive observations are not i.i.d.
- when you care about errors you make early in the learning process
- underlying “state of the environment” changes over time

*e.g. binary classification for **spam detection***

Online-to-batch conversion - naive.

Simplest approach: **Take the last iterate**


- treat the i.i.d. dataset as an ordered sequence
- train as you would do for online learning
- take weights at round $t=T$ and treat them as optimal set of weights

However

- why would final weights at $t=T$ also be optimal for full dataset (at all previous $t < T$)?
- too much variance in weight updates?

Remember: **Online Gradient Descent**

$$\begin{aligned}\theta_{t+1} &= \theta_t - \eta g_t \\ g_t &= \nabla_{\theta} \ell(x_t, y_t, \theta_t)\end{aligned}$$

A photograph of the Golden Gate Bridge in San Francisco, viewed from a high angle looking down the length of the bridge towards the city. The bridge's iconic red-orange towers and suspension cables are prominent against a hazy blue sky and the bay water. The city skyline is visible in the distance.

Follow the (regularized) leader

Why do we expect online learning to work for batch learning?

$$\ell_s(\theta) = \ell(x_s, y_s, \theta)$$

Follow The (Regularized) Leader.

The idea

- Adjust course “now”, but in a way that would **also minimize prior loss**
- Update based on **total loss so far**

$$\theta_{t+1} = \arg \min_{\theta} \sum_{s=1}^t \ell_s(\theta)$$

- Add regularization:

$$\theta_{t+1} = \arg \min_{\theta} \sum_{s=1}^t \ell_s(\theta) + \frac{1}{2\eta} \|\theta\|^2$$

Follow The Regularized Leader.

e.g. for least squares:

$$g_t = x_t (\theta_t \cdot x_t - y_t)$$

Useful simplification: linearization

Recall power-series expansion (Taylor):

$$f(a) = f(b) + (a - b) f'(b) + \frac{1}{2} (a - b)^2 f''(b) + \dots$$

Use subgradient formulation (i.e. “linear upper bound”)

$$\begin{aligned} \text{Regret}_T(\vartheta, \theta_*) &= \sum_{t=1}^T \left(\ell_t(\theta_t) - \ell_t(\theta_*) + \frac{\|\theta_t\|^2 - \|\theta_*\|^2}{2\eta} \right) \\ &\leq \sum_{t=1}^T \left((\theta_t - \theta_*) \cdot g_t + \frac{\|\theta_t\|^2 - \|\theta_*\|^2}{2\eta} \right) \end{aligned}$$

Follow The Regularized Leader.

Online Gradient Descent from FTRL

$$\begin{aligned}\theta_{t+1} &= \arg \min_{\theta} \text{Regret}_t(\theta, \theta_*) \\ &\approx \arg \min_{\theta} \sum_{s=1}^t \theta \cdot g_s + \frac{1}{2\eta} \|\theta\|^2 + \text{const} \\ &= -\eta \sum_{s=1}^t g_s \\ &= \theta_t - \eta g_t\end{aligned}$$

Follow The Regularized Leader.

In short,

- FTRL attempts to minimize Regret *explicitly*, albeit greedy
- FTRL is a general framework that encompasses many algorithms (incl. OGD)
- FTRL picks weights at each round to get closer to the optimum θ_*
(thus good candidate for batch optimization)

If there's time, will explain the well-known *Adaptive FTRL-Proximal* algorithm



Online-to-batch conversion

How to use **online learning** as a tool for ordinary **batch learning**?

Online-to-batch conversion.

arxiv.org/abs/1109.5647

arxiv.org/abs/1212.1824

Some choices:

Standard averaging:

$$\theta = \frac{1}{T} \sum_{t=1}^T \theta_t$$

$$\text{Loss}(\theta) - \text{Loss}(\theta_*) = \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$$

Take the last $k = \alpha T$ iterates:

$$\theta = \frac{1}{k} \sum_{t=1}^k \theta_{t+T-k}$$

$$\text{Loss}(\theta) - \text{Loss}(\theta_*) = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$$

Or simply the last iterate:

$$\theta = \theta_T$$

$$\text{Loss}(\theta) - \text{Loss}(\theta_*) = \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$$

Online-to-batch conversion.

Take-home message:

Taking the last iterate $\theta = \theta_T$ is sub-optimal ... but only **marginally** so.

Ask yourself:

Is it worth the trouble to do a proper online-to-batch conversion?

... or will I train my model slightly longer?

A photograph of the Golden Gate Bridge in San Francisco, taken at dusk. The bridge's iconic red-orange towers and suspension cables are silhouetted against a deep blue twilight sky. The bridge spans a wide expanse of the San Francisco Bay, with the city lights visible in the distance. The water is a dark blue, and the overall scene is serene and majestic.

Neat examples

Paid search results on Google.

The screenshot shows a Google search interface with the query "hotels in tel aviv". The search bar includes the Google logo, the text "hotels in tel aviv", and icons for voice search and image search. Below the search bar are navigation tabs: "All" (selected), "Maps", "Images", "News", "Shopping", "More", "Settings", and "Tools". The results section indicates "About 320,000 results (0.68 seconds)". Two paid advertisements are displayed:

500 Hotels in Tel Aviv, Israel | Search and Book Now | booking.com
[Ad] www.booking.com/Tel-Aviv/Hotels ▼
★★★★★ Rating for booking.com: 4.6 - 259 reviews
Book your **Hotel in Tel Aviv** online. No reservation costs. Great rates.
Read Real Guest Reviews · No Booking Fees · 24/7 Customer Service · Get Instant Confirmation
Types: Hotels, Apartments, Villas, Hostels, Resorts, B&Bs
[Book for Tonight](#) · [Book Now](#) · [Book for Tomorrow](#) · [Secure Booking](#) · [Rent out your property](#)

Hotels In Tel Aviv | Top 10 Hotels (Tel Aviv) | TripAdvisor.co.uk
[Ad] www.tripadvisor.co.uk/ ▼
Compare Prices & Save Money with TripAdvisor (World's Largest Travel Website).
Candid traveller photos · Easy price comparison · Millions of hotel reviews
Brands: IHG, Wyndham, Carlson
[Hotels](#) - from £79.00/night - [Compare Prices](#) · [More](#) ▼

Paid search results on Google.

Context

- Compute bids for Google AdWords
- We have a **very large number** of distinct keywords
- Runs daily

Old setup

- ML framework: [pyspark.ml](#)
- Training time: **90 minutes**
- Prediction time: **2 hours**

Current setup

- ML framework: [Vowpal Wabbit](#)
- Training time: **10 minutes** (single instance)
- Prediction time: **2 minutes** (parallelized on Hadoop)

Spam detection.

Context

- Customer Care
- Detect spam specific to Booking.com (e.g. auto-replies, receipt notification, etc.)
- Want to maximize precision/recall at fixed human capacity

Approach

- bag-of-words
- ML framework: [Vowpal Wabbit](#)
- feature engineering done by Vowpal Wabbit (n-grams, skip-grams, "tf-idf", etc.)

Missing children.

Context

30% of the searches done by 'Family with children' guests
do not specify number of children!

Hypothesis

They forgot their children



Missing children.

Actions taken

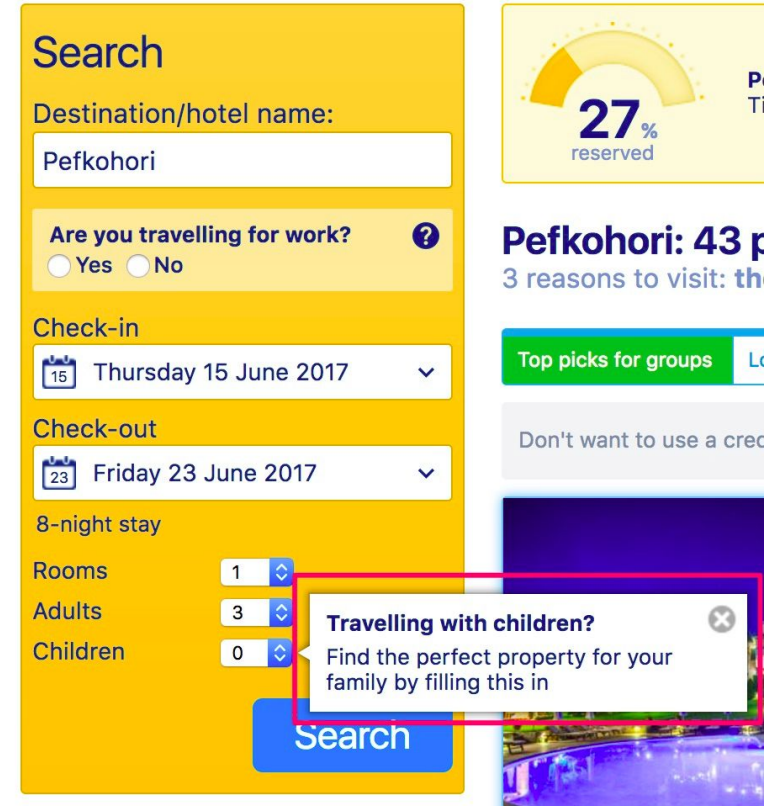


Booking.com

Amsterdam

May 13 - May 14 (1 night) 2 adults

Traveling With Kids?
Add your kids to find the perfect property!



Search

Destination/hotel name:
Pefkohori

Are you travelling for work?
☐ Yes ☐ No

Check-in
Thursday 15 June 2017

Check-out
Friday 23 June 2017

8-night stay

Rooms 1
Adults 3
Children 0

Travelling with children?
Find the perfect property for your family by filling this in

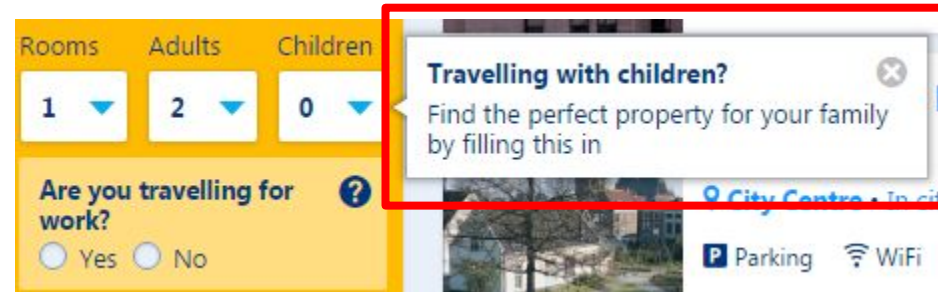
Search

27% reserved

Pefkohori: 43 p
3 reasons to visit: th

Top picks for groups

Don't want to use a cred



Rooms 1
Adults 2
Children 0

Are you travelling for work?
☐ Yes ☐ No

Travelling with children?
Find the perfect property for your family by filling this in

City Centre • In city

Parking WiFi

Missing children.

Target

At the Stay Review Form users tell us if they are a Family, a Group, Solo or a Couple

Build a Machine Learning Model that guesses the Traveller Type

Use information like Location, Destination, Filter Usage, etc.

Application

Apply the treatment only when the model says the user is most likely a Family.

Some details

Multiclass classification problem

Used 1 year data (200M examples / 200k features)

Missing children.

Outcome

A/B experiment a big success



A background image of the Golden Gate Bridge in San Francisco, viewed from a high angle looking down the length of the bridge towards the city. The bridge's iconic red-orange towers and suspension cables are prominent against a hazy blue sky and the bay water. The text is overlaid on the lower-left portion of the image.

Bonus material

Adaptive FTRL-proximal - “the FTRL optimizer”

Follow The Regularized Leader.

Adaptive FTRL-proximal

ordinary L1 regularization

$$\theta_{t+1} = \arg \min_{\theta} \sum_{s=1}^t \theta \cdot g_s + \lambda_1 \|\theta\|_1 + \frac{\sigma_s}{2} \|\theta - \theta_s\|_2^2$$

adaptive

proximal

Follow The Regularized Leader.

$$z_t = \sum_{s=1}^t g_s - \sigma_s \theta_s$$

$$\eta_t = \frac{1}{\sum_{s=1}^t \sigma_s}$$

Adaptive FTRL-proximal

$$\begin{aligned}\theta_{t+1} &= \arg \min_{\theta} \sum_{s=1}^t \theta \cdot g_s + \lambda_1 \|\theta\|_1 + \frac{\sigma_s}{2} \|\theta - \theta_s\|_2^2 \\ &= \arg \min_{\theta} \theta \cdot z_s + \lambda_1 \|\theta\|_1 + \frac{1}{2\eta_t} \|\theta\|_2^2 + \text{const} \\ &= \begin{cases} -\eta_t (z_t - \text{sign}(z_t) \lambda_1) & \text{for } |z_t| > \lambda_1 \\ 0 & \text{otherwise} \end{cases}\end{aligned}$$

Follow The Regularized Leader.

Adaptive learning rate schedule

- Adagrad:

$$\eta_{t,i} = \frac{\alpha}{\sqrt{\sum_{s=1}^t g_{s,i}^2}}$$

- FTRL-proximal:

$$\eta_{t,i} = \frac{\alpha}{\sqrt{\sum_{s=1}^t g_{s,i}^2 + \beta + \alpha \lambda_2}}$$

A modern office interior with a ping pong table and a foosball table. The room has large windows, a wooden ceiling, and a wall with green circular decorations. People are sitting and standing in the background.

The end. Thanks!

workingatbooking.com



Secure | <https://booking.ai>



Booking.datascience

The Data Science, Machine Learning and Analytics behind Booking.com

[ARCHIVE](#)

[DATA SCIENCE CAREERS](#)



[Follow](#)



How Booking.com increases the power