Aggregating Algorithm for Prediction of Packs

Yuri Kalnishkan

Computer Learning Reseach Centre and
Department of Computer Science
Royal Holloway, University of London

September 2019

Online Learning Framework

- the outcomes $\omega_1, \omega_2, \ldots$ occur one after another
  — outcomes come from an outcome space Ω
- before seeing the outcome ω_t we output a prediction $\gamma_t \in \Gamma$
  — predictions can be drawn from a prediction space Γ
- discrepancies between predictions and outcomes leads to loss given by a loss function $\lambda : \Gamma \times \Omega \to [0, +\infty]$
  — we want the cumulative loss

  $$\mathrm{Loss}_T = \sum_{t=1}^{T} \lambda(\gamma_t, \omega_t)$$

  to be small
- the triple $\langle \Omega, \Gamma, \lambda \rangle$ is called a game

Experts

- there are N experts $E_1, E_2, \ldots, E_N$ predicting the same sequence

  (1) FOR $t = 1, 2, \ldots$
  (2) the experts output predictions $\gamma_t^n \in \Gamma$, $n = 1, \ldots, N$
  (3) the learner produces $\gamma_t \in \Gamma$
  (4) the nature outputs $\omega_t \in \Omega$
  (5) the learner suffers loss $\lambda(\gamma_t, \omega_t)$
  (6) the experts suffer losses $\lambda(\gamma_t^n, \omega_t)$, $n = 1, 2, \ldots, N$
  (7) END FOR
- we want to construct a merging algorithm ensuring that our loss is the same or little worse than those of the best expert
  — i.e., we want guarantees of the type
  $\mathrm{Loss}_T(\text{Learner}) \lesssim \mathrm{Loss}_T(E_n)$ for all n and T

Aggregating Algorithm

parameters: η and initial distribution $q_1, q_2, \ldots, q_N$

(1) initialise weights $w_0^n = q_n$, $n = 1, 2, \ldots, N$
(2) FOR $t = 1, 2, \ldots$
(3) read the experts' predictions γ_t^n, $n = 1, 2, \ldots, N$
(4) normalise the weights $p_{t-1}^n = w_{t-1}^n / \sum_{n=1}^N w_{t-1}^n$
(5) solve the system ($\omega \in \Omega$):
$\lambda(\gamma, \omega) \le -\frac{C_\eta}{\eta} \ln \sum_{n=1}^N p_{t-1}^n e^{-\eta\lambda(\gamma_t^n, \omega)}$
w.r.t. γ and output a solution γ_t
(6) observe the outcome ω_t
(7) update the experts' weights $w_t^n = w_{t-1}^n e^{-\eta\lambda(\gamma_t^n, \omega_t)}$, $n = 1, 2, \ldots, N$
(8) END FOR

[Vovk, 1991]

Mixability Constant

- for every $\eta > 0$, the mixability constant C_η is the minimal C such that for all arrays of predictions $\gamma^1, \ldots, \gamma^N$ and weights $p^1, \ldots, p^N$ there is γ, such that for all ω

$$\lambda(\gamma, \omega) \le -C\frac{1}{\eta} \ln \sum_{n=1}^{N} p^n e^{-\eta\lambda(\gamma^n, \omega)}$$

— we can solve the system of inequalities "relaxed" by C_η

- we get the guarantee [Vovk, 1991]

$$\mathrm{Loss}_T(\mathrm{Learner}) \le C_\eta \mathrm{Loss}_T(E_n) + \frac{C_\eta}{\eta} \ln N$$

— for every expert E_n, all experts' predictions, and all moments T

Optimality of the AA

- if any algorithm is capable of achieving

$$\mathrm{Loss}_T(\mathrm{Learner}) \le A \,\mathrm{Loss}_T(E_n) + B \ln N$$

for every expert E_n, all experts' predictions, and all moments T
– then AA can do the same or better for some η:

$$C_\eta \le A$$
$$\frac{C_\eta}{\eta} \le B$$

for some η

[Vovk, 1998]

Mixability

- we have $C_\eta \ge 1$; if $C_\eta = 1$, the game is called η-mixable
- example: square-loss game $\Omega = \Gamma = [A, B]$ is mixable for $\eta \le 2/(B - A)^2$
- the minimal η such that the game is η-mixable is the obvious choice

Packs

- suppose that on step t several outcomes $\omega_{t,1}, \omega_{t,1}, \ldots \omega_{t,K_t}$ happen
— and we need to make K_t predictions $\gamma_{t,1}, \gamma_{t,1}, \ldots \gamma_{t,K_t}$
— we do not predict outcomes one by one, but predict a pack of them
- K_t can stay the same or vary from step to step
- the plain loss is

$$\mathrm{Loss}_T = \sum_{t=1}^{T} \sum_{k=1}^{K_t} \lambda(\gamma_{t,k}, \omega_{t,k})$$

BOLD

- we can run several instances of a regular merging algorithm at the same time
— when we get experts' predictions $\gamma^1_{t,k},\dots,\gamma^N_{t,k}$, we feed them to an available instance of the algorithm
— it gives us a prediction $\gamma_{t,k}$ and gets blocked until the outcome $\omega_{t,k}$ is known; then we give it to the algorithm and it becomes available to merge more experts' predictions
— if no instances are available, we start a new one
- loss bound with AA as the base algorithm:

$$\mathrm{Loss}_T(\text{Learner}) \le C_\eta \, \mathrm{Loss}_T(E_n) + \max_{t=1,\dots,T} K_t \frac{C_\eta}{\eta} \ln N$$

[Joulani et al, 2013]

Question

- can we manage with one instance of AA?
- BOLD depends on the order of outcomes within a pack
— can we have an order-independent algorithm?
- BOLD has $K\cdot N$ weights for N experts
— can we manage with N?

Mixability for Packs

- for a game $\mathfrak{G}=\langle\Omega,\Gamma,\lambda\rangle$ consider the game $\mathfrak{G}^K=\langle\Omega^K,\Gamma^K,\lambda^{(K)}\rangle$
— where

$$\lambda^{(K)}((\gamma_1,\dots,\gamma_K),(\omega_1,\dots,\omega_K)) = \sum_{k=1}^{K}\lambda(\gamma_k,\omega_k)$$

- Theorem $C^{(K)}_{\eta/K} = C_\eta$
— for the equality we need to assume convexity of λ in γ; but we always have $C^{(K)}_{\eta/K} \le C_\eta$

Aggregating Algorithm for Packs

(1) initialise weights $w^n_0 = q_n$, $n=1,2,\dots,N$
(2) FOR $t=1,2,\dots$
(3) normalise the weights $p^n_{t-1} = w^n_{t-1}/\sum_{n=1}^N w^n_{t-1}$
(4) FOR $k=1,2,\dots,K_t$
(5) read the experts' predictions $\gamma^n_{t,k}$, $n=1,2,\dots,N$
(6) solve the system ($\omega\in\Omega$):
$\lambda(\gamma,\omega) \le -\frac{C_\eta}{\eta}\ln\sum_{n=1}^N p^n_{t-1} e^{-\eta\lambda(\gamma^n_{t,k},\omega)}$
w.r.t. γ and output a solution $\gamma_{t,k}$
(7) END FOR
(8) observe the outcomes $\omega_{t,1},\dots,\gamma_{t,K_t}$
(9) update the experts' weights $w^n_t = w^n_{t-1} e^{-\eta\sum_{t=1}^{K_t}\lambda(\gamma^n_{t,k},\omega_{t,K})/\overline{K_t}}$, $n=1,2,\dots,N$
(10) END FOR

Guarantees

- what is $\overline{K_t}$ on step t?
  — there are two options
- AAP-incremental: take $\overline{K_t} = \max_{s=1,2,\ldots,t} K_s$; then

$$\text{Loss}_T(\text{Learner}) \le C_\eta \, \text{Loss}_T(E_n) + \max_{t=1,\ldots,T} K_t \frac{C_\eta}{\eta} \ln N$$

  as in BOLD
- AAP-current: take $\overline{K_t} = K_t$: then

$$\text{Loss}_T^{\text{average}}(\text{Learner}) \le C_\eta \, \text{Loss}_T^{\text{average}}(E_n) + \frac{C_\eta}{\eta} \ln N,$$

  where

$$\text{Loss}_T^{\text{average}} = \sum_{t=1}^{T} \frac{\sum_{k=1}^{K_t} \lambda(\gamma_{t,k}, \omega_{t,k})}{K_t}$$

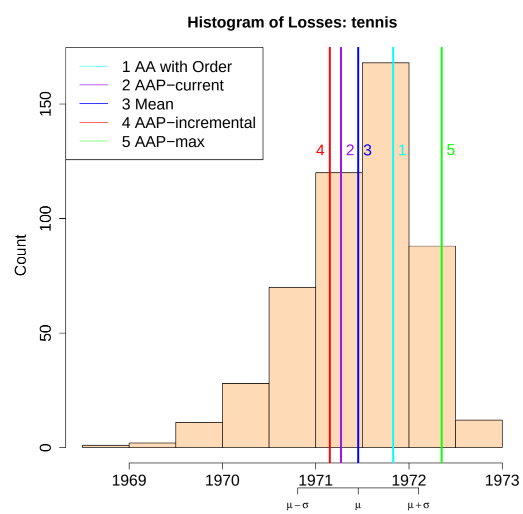
Experiments

- bookmakers' data (after [Vovk and Zhdanov, 2009]
  — tennis: artificial packs
  — football: true packs
- house sales data
  — we want to work out the house sale price from the house description
  — the experts are regressions and trees trained on a month of data from the first year: January, February, . . . , December
  — a pack is made of all houses sold in a month
  — Ames dataset: 2930 transactions
  — London area house sales: 1.38 million transactions

BOLD vs AAP

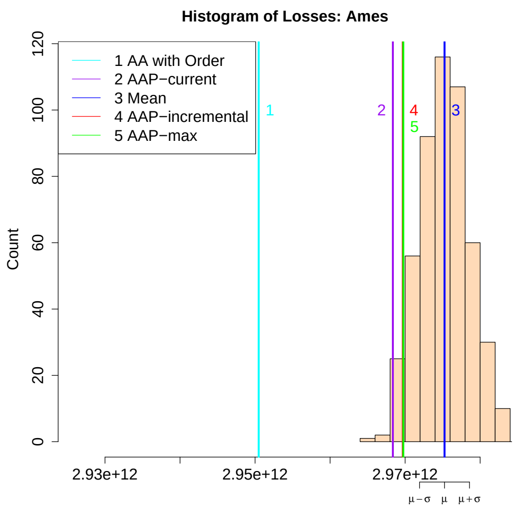
- BOLD depends on the order within a pack and AAP does not
  — let us shuffle the data within a pack and see what happens

Tennis



- the histogram shows the losses of BOLD under shuffling within packs
- AAP-incremental and AAP-current beat the average

Regression on Ames House Prices



- AAP-incremental and AAP-current still beat the average
- but BOLD with the original order beats them all by far
- the order conveys useful information (location?)

Incremental vs Current

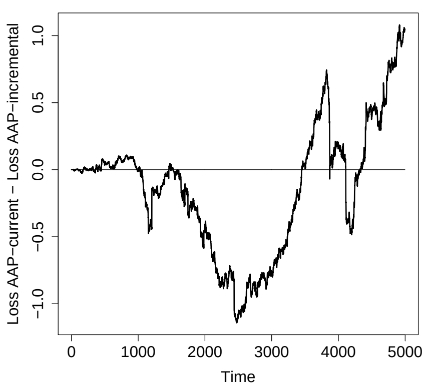
- AAP-incremental achieves

$$\text{Loss}_T(\text{Learner}) \le C_\eta \, \text{Loss}_T(E_n) + \max_{t=1,\ldots,T} K_t \frac{C_\eta}{\eta} \ln N$$

— whenever K_t is uses a suboptimal learning rate
- AAP-current uses the optimal learning rate
— but for the plain cumulative loss we only get

$$\text{Loss}_T(\text{Learner}) \le \frac{\max_{t=1,\ldots,T} K_t}{\min_{t=1,\ldots,T} K_t} C_\eta \, \text{Loss}_T(E_n) + \max_{t=1,\ldots,T} K_t \frac{C_\eta}{\eta} \ln N$$

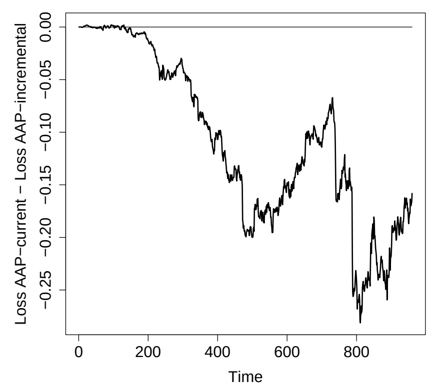
Tennis Small Packs



- the difference $\text{Loss}_T(\text{AAP-current}) - \text{Loss}_T(\text{AAP-incremental})$ is plotted vs T
- artificial packs of size 1 to 12 are used

$$\frac{\max_{t=1,\ldots,T} K_t}{\min_{t=1,\ldots,T} K_t} = 12$$

Tennis Large Packs



- the difference $\text{Loss}_T(\text{AAP-current}) - \text{Loss}_T(\text{AAP-incremental})$ is plotted vs T
- artificial packs of size 5 to 16 are used

$$\frac{\max_{t=1,\ldots,T} K_t}{\min_{t=1,\ldots,T} K_t} = 3.2$$