Competitive Analysis

Lee, Chen-Chia Weng, Jeng-Chi

R10942172 B08901009

Abstract

In theoretical research, we often consider the worst-case performance of algorithms and since different measures of performance lead to different results, we seek to find a better way that comes up with a more realistic solution by introducing "competitive analysis". The concept of competition includes why we compete with others, how we compete and who we compete with, and these are all important factors to a great algorithm and the analysis of its performance. Furthermore, we demonstrate examples of competitive analysis in the field of information theory.

1 Introduction

In daily life, we often need to make decisions based on what we observe or experience from the past. For example, based on the conditions of the sky you see outside the window of your house, you might think whether to bring an umbrella with you. After a day, a straightforward way to determine whether it is the right decision is by observing the actual weather of the day, whether it rains or not.

However, for different problems, there exist different criteria for the greatness of the decisions, and what we are mostly concerned about is not only to find the optimal solution for the problem but also to ask how "good" a decision or a solution is? Both are of critical importance and are mutually dependent of each other.

The following chapters will be organized as follows:

In Chapter 2, we'll firstly take a look of a general choice of loss, analyze what we obtain form it and discuss its problem. To overcome its shortcoming, we introduce another approach, called competitive analysis. With a classical example, Buy/Rental Problem, we can catch a glimpse of the benefit of competitive analysis.

In Chapter 3, we'll discuss a problem that is analyzed in a competitive manner in the field of information theory called universal prediction. The general task of universal prediction can be further categorized into two different problem settings: the probabilistic setting and the deterministic setting. For different settings, their differences will be discussed and some interesting results and interpretations will be shown. For example, we'll see the benefits of defining the loss to be the self-information loss. Also, the redundancy-capacity theorem relates the redundancy, which is defined in class, to the capacity of a channel.

In Chapter 4, we'll dig more deeply into the deterministic setting of universal prediction and introduce an algorithm called Aggregating Algorithm, which is proposed by V.Vovk[1]. As in chapter 3, we will consider the self-information loss and discuss the connection of deterministic universal prediction and universal coding problem.

In the end, we gave two interesting examples of competitive analysis which use different approaches to apply competitive analysis.

2 Background

In theoretical research, we often need to solve problems of the following form:

$$\arg\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} f(x,y).$$

For example, in the task of the universal data compression, we may want to find a distribution $Q \in \mathcal{P}(S)$ which has the minimum expected code length. In other word, solving

$$\arg\min_{Q\in\mathcal{P}(S^n)} \max_{P\in\mathcal{P}(S^n)} \mathop{\mathbb{E}}_{S^n\sim P}[-\log Q(S^n)].$$

We can easily see that $\min_{Q\in\mathcal{P}(S^n)} \max_{P\in\mathcal{P}(S^n)} \mathbb{E}_{S^n\sim P}[-\log Q(S^n)] = n\log|S|$ for discrete case. So for different problem realization, we don't have the same performance benchmark. To overcome this issue, a trivial and meaningful performance measure is to compete with the best choice, which is to solve

$$\arg\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \left\{ f(x,y) - \min_{x'\in\mathcal{X}} f(x^{'},y) \right\}.$$

The method of performance analysis introduced above is called "competitive analysis". In the universal data compression case, $\min_{Q\in\mathcal{P}(S^n)} \mathbb{E}_{S^n\sim P}[-\log Q(S^n)] = H(P)$ and $\mathbb{E}_{S^n\sim P}[-\log Q(S^n)] - H(P) = D(P\|Q)$. Thus, applying competitive analysis on universal data compression task is as same as solving minimax expected redundancy which is taught in the lecture.

Competing with the best element in $\mathcal{X}$ sounds reasonable, but in some situations, this may be too strict and the outcome may be too pessimistic, so instead competing with all the elements in $\mathcal{X}$, we may want to take an alternative approach, which is to compete with the best element in a reasonable "subset" of $\mathcal{X}$:

$$\arg\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \left\{ f(x,y) - \min_{x'\in\mathcal{X}'} f(x^{'},y) \right\},$$

where $\mathcal{X}^{'} \subseteq \mathcal{X}$. This slightly different approach is often used in the field of online learning which is called the deterministic setting of universal prediction in the following articles.

In addition to using subtraction, we can also use division as a method of comparison. Let us give a concrete example of competitive analysis with division comparison: Buy/Rental problem. In Buy/Rental problem, a man is living in a house for an unknown number (d) of days. Every day, the man has to decide whether to rent or buy the house, rent the house for a day cost 1 dollar, buy the house cost B dollars.

If we simply take the approach of minimizing the worst-case actual total cost, which means to find an algorithm that achieves the following term.

$$\min_{\text{Alg}} \max_{d\in\mathbb{N}} Cost(\text{man}).$$

Clearly, the man will buy the house at the very first day and the cost will be B dollars, which sounds quite unrealistic and overly pessimistic. However, if we consider the ratio between the cost of the man and the cost of the person who knows d, i.e.

$$\frac{Cost(\text{man})}{Cost(\text{person who knows d})}$$

and further try to minimize it in the worst-case scenario, we get the following result.

$$\min_{\text{Alg}} \max_{d\in\mathbb{N}} \frac{Cost(\text{man})}{Cost(\text{person who knows d})} = \min_{\text{Alg}} \max_{d\in\mathbb{N}} \frac{Cost(\text{man})}{\min(d,B)}$$

$$= \frac{(B-1)+B}{B}$$
$$= 2 - \frac{1}{B}$$

which means the man will rent the house for the first $B-1$ days and buy the house at the B-th day, and the cost will not exceed two times of the best strategy. Therefore, from this example, we can clearly see the benefit of competitive analysis, where we set a benchmark to compare with and get a more reasonable result.

3 Universal Prediction

We will introduce universal prediction as a protocol to demonstrate competitive analysis, which will include the universal coding that have been taught in the lecture and the more general concept called online learning.

Protocol 1 Universal Prediction

For each round $t = 1, 2, \ldots,$

- Learner announces decision b_t based on the previous outcomes x^{t-1}
- Reality announces outcome $x_t \in \mathcal{X}$
- Learner suffers loss $\ell(b_t, x_t) \in \mathbb{R}^+$

Goal:
To minimize the loss (can be instantaneous loss, expected loss or other settings)

In general, the learner's decision b_t can be in different forms. For example, b_t can be $\hat{x_t}$, an estimate of x_t. For this part, we consider b_t as a conditional probability assignment of x_t given x^{t-1}.

Furthermore, in the task of universal prediction, there are different settings of the source P of Reality that is generating the outcomes x_t's $\in \mathcal{X}$. First, the source P can be unknown, which means there exists a source generating the outcomes following a probability distribution that is unknown to the learner. Another setting is that the source, in fact, is nonexistent. We will further discuss these two different settings respectively.

3.1 Probabilistic Setting

In a competitive manner, our goal is to keep the difference of the expected cumulative loss between a universal predictor $b_t^u(x^{t-1})$ and the optimal predictor that knows the underlying distribution P vanishing small for large n. The difference can be written in the following mathematical form.

$$\mathbb{E}\left\{\frac{1}{n}\sum_{t=1}^{n}\ell(b_t^u, X_t)\right\} - \frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\{\inf_b \mathbb{E}[\ell(b_t, X_t)|X^{t-1}]\}$$

Straightforwardly, if we define the loss as the self-information loss

$$\ell(b_t, X_t) = -\log(b(X)),$$

where $b(\cdot)$ is a probability function, and we further plug this term into the above expected cumulative loss of the optimal predictor,

$$\frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\{\inf_b \mathbb{E}[\ell(b_t, X_t)|X^{t-1}]\}$$

$$
\begin{aligned}
&= \frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\{\inf_{b}\mathbb{E}[-\log b(X_t|X^{t-1})]\} \\
&= \frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\{H(X_t|X^{t-1})\} \\
&= \frac{1}{n}H(X^n)
\end{aligned}
$$

we can see that, in this case, the goal of universal prediction is equivalent to that of universal coding, which is taught in class.

From the above, we see that using the self-information loss simplifies the original problem, universal prediction, that if of our interest. In fact, self-information loss does give us other benefits.

First, $b_t(x_t|x^{t-1})$ is a conditional probability distribution, and combining every $b_t(x_t|x^{t-1})(1 \le t \le n)$ together leads to a probability assignment $Q(x^n)$ of the entire sequence x^n.

$$
Q(x^n) = \prod_{t=1}^{n} b_t(x_t|x^{t-1})
$$

On the other hand, if we have $Q(x^n)$, we can also come up with a conditional probability distribution $b_t(x_t|x^{t-1})$.

$$
b_t(x_t|x^{t-1}) = \frac{Q(x^t)}{Q(x^{t-1})}
$$

Utilizing the property of self-information loss along with this duality of $b_t(x_t|x^{t-1})(1 \le t \le n)$ and $Q(x^n)$,

$$
\mathbb{E}\left\{\frac{1}{n}\sum_{t=1}^{n} -\log(b(X_t|X^{t-1}))\right\} = \mathbb{E}\left\{\frac{1}{n}(-\log(Q(X^t)))\right\}
$$

we see that for the problem of universal prediction, finding $b_t(x_t|x^{t-1})$ is equivalent to finding Q for the entire sequence if we define loss as the self-information loss.

In class, we've analyze the task of universal source coding, which is a special case of universal prediction if we define the loss as the self-information loss as mentioned above, by considering the *minimax redundancy*. In that discussion, we also assume $X^n \sim P_\theta$, where $\theta \in \Theta$, and denote R_n^+ as the normalized minimax redundancy.

$$
\begin{aligned}
R_n^+ &= \inf_{Q\in\mathcal{P}(\mathcal{X}^n)}\sup_{\theta\in\Theta}\frac{1}{n}\{\mathop{\mathbb{E}}_{P_\theta}[\log\frac{1}{Q(X^n)}] - H_n(P_\theta)\} \\
&= \inf_{Q\in\mathcal{P}(\mathcal{X}^n)}\sup_{\theta\in\Theta}\frac{1}{n}\{D_n(P_\theta\|Q)\}
\end{aligned}
$$

This can be seen as Learner is "competing" with a person that knows the underlying distribution P_θ, and we set his loss as a benchmark to compare with. Here, we're interested in another approach, *maximin approach*.

For *maximin approach*, we've swapped the order of decision, where Learner assigns the probability Q after Reality announces the $w(\cdot)$, the probability density function over Θ. Therefore, the performance of Q will be judged with normalized weighted average redundancy,

$$
R_n(Q, w) = \frac{1}{n}\int_\Theta \mathrm{d}w(\theta)D_n(P_\theta\|Q).
$$

Then, consider the minimax problem

$$\sup_w \inf_Q R_n(Q, w),$$

we can see that

$$\arg\min_Q R_n(Q, w) = Q_w(\cdot) = \int_\Theta \mathrm{d}w(\theta) P_\theta(\cdot).$$

If we further the plug in $Q_w(\cdot)$ to the term $\sup_w \inf_Q R_n(Q, w)$,

$$\begin{aligned}
&\sup_w \inf_Q R_n(Q, w) \\
&= \sup_w \frac{1}{n} \int_\Theta \mathrm{d}w(\theta) \sum_{x^n} P_\theta(x^n) \log \frac{P_\theta(x^n)}{\int_\Theta P_\theta(x^n)\mathrm{d}w(\theta)} \\
&= \sup_w \frac{1}{n} \int_\Theta \sum_{x^n} \mathrm{d}w(\theta) P_\theta(x^n) \log \frac{w(\theta)P_\theta(x^n)}{w(\theta)\int_\Theta P_\theta(x^n)\mathrm{d}w(\theta)} \\
&= \sup_w \frac{1}{n} I_w(\Theta; X^n) \triangleq R_n^-
\end{aligned}$$

we end up taking the supremum of a mutual information. Therefore, interestingly, we can interpret R_n^- as the capacity of a "channel" from Θ to X^n, denoted by C_n. Furthermore, it was proven by Davisson (1973)[2] that $R_n^+ = R_n^- = C_n$ and $w^* = \arg\max_w \frac{1}{n} I_w(\Theta; X^n)$ is both minimax and maximin optimal. This is called the the "Redundancy-Capacity Theorem".

3.2 Deterministic Setting

In the deterministic setting of the universal prediction or say "Online Learning", the outcome sequence x_t need not be generated by a probability distribution, it is even possible for the Reality to select the corresponding output with respect to the output of the Learner. An issue arises if we trivially compete with the best decision in the whole decision space, that is, for any output Reality announces, there always exists a best choice in the decision space. For example, if we want to predict whether tomorrow will rain or not, once we announce a decision, the Reality can output the opposite output, which means we will lose in every round, but there always exists a sequence of predictions that is the same as the output Reality gives. Thus, we have to introduce some restrictions on the competitors to give a more reasonable performance measure.

The first case is to compete with the best constant decision, which is often used in Online Convex Optimization. To analyze the performance of the Learner, we define a quantity called "Regret",

$$R_T = \sum_{t=1}^T \ell(b_t, x_t) - \min_b \sum_{t=1}^T \ell(b, x_t),$$

the goal of the learner is to minimize the regret as small as possible. We say the algorithm is "no regret" if $R_T = o(T)$.

The second case is to compete with the best "expert", which is called "Learning with Expert Advice (LWEA)". In LWEA, we no longer compete with the constant decision but the constant expert that can be any other algorithm that need not output the same decision anymore. The protocol of LWEA is as follow,

Protocol 2 Learning with Expert Advice (LWEA)

For each round $t = 1, 2, \ldots,$

- Each expert $\theta \in \Theta$ announces decision $\xi_t(\theta) \in \mathcal{B}$
- Learner announces decision $b_t \in \mathcal{B}$ based on the previous outcomes x^{t-1} and each $\xi_t(\theta)$.
- Reality announces outcome $x_t \in \mathcal{X}$
- Learner suffers loss $\ell(b_t, x_t) \in \mathbb{R}^+$

The definition of the regret in LWEA is as follow,

$$R_T = \sum_{t=1}^{T} \ell(b_t, x_t) - \min_{\theta} \sum_{t=1}^{T} \ell(\xi_t(\theta), x_t).$$

There are several algorithms for solving LWEA, such as

- Follow the Leader (FTL)
- Weighted Majority
- Multiplicative Weight Update
- Hedge Algorithm
- Aggregating Algorithm (AA)

Proposing a new analysis method is not enough. We also need to provide an algorithm with good performance. In the next chapter we are going to introduce the Aggregating Algorithm which is developed by V.Vovk.[1]

4 Aggregating Algorithm

The Aggregating Algorithm (AA) shares the same spirit with the Mixture Algorithm that is taught in the lecture, they both need to assign a weight or say probability on the parameter space, and construct the output by averaging. The main difference between them is that AA has an extra step that maps the averaging to the exact output. Before introducing AA we have to introduce a pseudo-algorithm called "Aggregating Pseudo-Algorithm (APA)" that does not have the extra step mentioned above and can help us analyze the regret bound of AA.

4.1 APA and AA

The main spirit of AA and APA is to update the prior probability of the experts by the exponential weight update method.

Algorithm 3 Aggregating Pseudo-Algorithm (APA)

Learner choose a learning rate $\eta > 0$, let $\beta = \mathrm{e}^{-\eta}$, and a Prior distribution P_0 on Θ.
Each round $t = 1, 2, \ldots,$

- Learner updates the expert's weights by

$$P_t(\mathrm{d}\theta) = \beta^{\ell(\xi_t(\theta), x_t)} P_{t-1}(\mathrm{d}\theta)$$

- Learner announces a generalized prediction

$$g_t(\cdot) = \log_\beta \int_\Theta \beta^{\ell(\cdot, \xi_t(\theta))} P^*_{t-1}(\mathrm{d}\theta),$$

where $P^*_{t-1}(\mathrm{d}\theta) = \dfrac{P_{t-1}(\mathrm{d}\theta)}{P_{t-1}(\mathrm{d}\Theta)}$.

We can find out that in APA, the learner does not output a "prediction" but a function with real value range which is called "generalized prediction". The definition of generalized prediction is as follows.

Definition 4 (Generalized Prediction). A function g is a generalized prediction if it maps $\mathcal{X}$ to $[0, \infty]$.

Why do we call g_t a "generalized prediction"? The generalized prediction only cares about the loss of the prediction and does not actually give the actual prediction. For example, denote the permitted prediction $b \in \mathcal{B}$ as a generalized prediction, we will get $\ell(\cdot, b)$.

Lemma 5. For any learning rate $\eta > 0$ and prior P_0, and $T = 1, 2, \ldots,$

$$L_T(\mathrm{APA}(\eta, P_0)) = \log_\beta \int_\Theta \beta^{L_T(\theta)} P_0(\mathrm{d}\theta).$$

The proof is straightforward by the definition of g_t and we omit it here. We are ready to introduce AA, and recall that the only difference between APA and AA is that AA outputs a permitted prediction $b \in \mathcal{B}$ instead of a generalized prediction.

Algorithm 6 Aggregating Algorithm (AA)

Learner chooses a learning rate $\eta > 0$, let $\beta = \mathrm{e}^{-\eta}$, and a Prior distribution P_0 on Θ.
Each round $t = 1, 2, \ldots,$

- Learner updates the expert's weights by

$$P_t(\mathrm{d}\theta) = \beta^{\ell(\xi_t(\theta), x_t)} P_{t-1}(\mathrm{d}\theta)$$

- Learner announce a prediction $\Sigma(g_t)$, where, $g_t(\cdot) = \log_\beta \int_\Theta \beta^{\ell(\cdot, \xi_t(\theta))} P^*_{t-1}(\mathrm{d}\theta)$ and $P^*_{t-1}(\mathrm{d}\theta) = \dfrac{P_{t-1}(\mathrm{d}\theta)}{P_{t-1}(\mathrm{d}\Theta)}$.

The function Σ is called the substitution function, which maps the generalized prediction $g : \mathcal{X} \to [0, \infty]$ to a permitted prediction $\Sigma(g) \in \mathcal{B}$. The substitution function should be chosen carefully so that we can share the same cumulative loss bound in APA. First, we observe that all possible generalized predictions that APA announces is in the form $\log_\beta \int_\Theta \beta^{\ell(x,b)} Q(\mathrm{d}b)$, where Q is a probability distribution on $\mathcal{B}$. We

denote the set of distribution with this form as $\mathcal{P}(\ell, \eta)$.Then, we say the substitution function is perfect (w.r.t. ℓ, η) if

$$\forall g \in \mathcal{P}(\ell, \eta),\ \ell(x, \Sigma(g)) \leq g(x).$$

If such substitution function exists, we call our game $(\mathcal{X}, \mathcal{B}, \ell)$ a $\eta -$ mixable (perfect mixable) game.

Lemma 7. For an η-mixable game,

$$L_T(\mathrm{AA}(\eta, P_0)) \leq \log_\beta \int_\Theta \beta^{L_T(\theta)} P_0(\mathrm{d}\theta)$$

This is a direct result of lemma 6. Once we have lemma 8, we can get a well-known result of AA for a finite number of experts.

Proposition 8. For η-mixable game with n experts,

$$L_T(\mathrm{AA}(\eta, P_0)) - \min_{\theta \in \Theta} L_T(\theta) \leq \frac{\ln n}{\eta}$$

Now let us give a concrete example of applying AA to a specific LWEA problem.

4.2 Log-Loss Game

Let the decision space be $\mathcal{B} = \triangle(\mathcal{X})$ and the loss function be $\ell(x, b) = -\ln b(x)$, which is called self-information loss in the previous chapter. The log-loss game is an online version of the universal coding problem.

Protocol 9 Log-Loss game

Assume $|\mathcal{X}| < \infty$ and $|\Theta| < \infty$.
For each round $t = 1, 2, \ldots,$

- Each expert $\theta \in \Theta$ announces decision $\xi_t^\theta \in \triangle(\mathcal{X})$.
- Learner announces decision $b_t \in \triangle(\mathcal{X})$ based on the previous outcomes x^{t-1} and each ξ_t^θ.
- Reality announces outcome $x_t \in \mathcal{X}$.
- Learner suffers loss $-\ln b_t(x_t) \in \mathbb{R}^+$.

To apply AA to the log-loss game, we have to specify the substitution function. To do so, let us first analyze the generalized prediction that occurs in AA for the log-loss game. Let P_0 be the prior and set $\eta = 1$. By the discussion in chapter 3, we can associate θ to a distribution Q^θ on $\mathcal{X}^\infty$, where for all t, $\xi_t(\theta) = \xi_t^\theta$ is the conditional distribution $Q^\theta_{x^{t-1}}$. Then the weight update step in AA becomes

$$P_t(\theta) = \beta^{\ell(x_t, \xi_t(\theta))} P_{t-1}(\theta) = \xi_t^\theta(x_t) P_{t-1}(\theta) = Q_\theta[x_1, \ldots, x_t] P_0(\theta),$$

where $[x_1, \ldots, x_t]$ stands for the set of all sequences in $\mathcal{X}^\infty$ that begin with $x_1, \ldots, x_t$.
The normalized weights become

$$P_t^*(\mathrm{d}\theta) = \frac{P_t(\mathrm{d}\theta)}{P_t(\Theta)} = \frac{\xi_t^\theta(x_t) P_t(\theta)}{\sum_\theta \xi_t^\theta(x_t) P_t(\theta)} = \frac{Q_\theta[x_1, \ldots, x_t] P_0(\theta)}{\sum_\theta Q_\theta[x_1, \ldots, x_t] P_0(\theta)}.$$

The generalized prediction becomes

$$\begin{aligned} g_t(x) &= \log_\beta \int_\Theta \beta^{\ell(x,\xi_t(\theta))} P^*_{t-1}(\mathrm{d}\theta) \\ &= \log_{\frac{1}{\mathrm{e}}} \int_\Theta \mathrm{e}^{\ln \xi^\theta_t(x)} P^*_{t-1}(\mathrm{d}\theta) \\ &= -\ln \int_\Theta \xi^\theta_t(x) P^*_{t-1}(\mathrm{d}\theta) \\ &= -\ln \sum_\theta \frac{Q_\theta[x_1,\ldots,x_{t-1},x]P_0(\theta)}{\sum_\theta Q_\theta[x_1,\ldots,x_{t-1}]P_0(\theta)} \end{aligned}$$

Where the term in $-\ln$ is the predictive distribution of Bayesian mixture $\int_\Theta Q_\theta P_0(\mathrm{d}\theta)$. Thus, the substitution function can easily output the predictive distribution of x_t according to the Bayesian mixture.

5 More examples for Competitive Analysis

Besides the discussion above, we give two more interesting competitive analysis examples. The first is the "Composite Hypothesis Test", which applies the competitive analysis by division apart from subtraction that we used in previous sections. The second is "Competitive distribution estimation", which competes with the so-called "nature estimator" that makes sense in the problem setting.

5.1 Composite Hypothesis Test.

In hypothesis testing, each parameter may take value in a different set, which we called the "composite hypothesis test", Feder, Meir and Merhav, Neri proposed a competitive analysis approach for the composite hypothesis test [3]. Here we give a mathematical representation of the composite hypothesis test. Let $y = (y_1, y_2, \ldots, y_n) \in \mathcal{Y}^n$ be the n dimensional vector that denote the observed data, and let $H_0, H_1, \ldots, H_{M-1}$ denote M hypothesis associate with M parameter $\theta = (\theta_0, \theta_1, \ldots, \theta_{M-1})$ which take value in $\Lambda_0, \Lambda_1, \ldots, \Lambda_{M-1}$ respectively.

Let $\Omega = \{\omega(0|y), \omega(1|y), \ldots, \omega(M-1|y), y \in \mathcal{Y}^n\}$ denotes a test, where each $\omega(i|y)$ is a conditional probability of hypothesis. If we directly analyze the probability of error and seeks the test that minimizes the probability of error, we will get the maximum likelihood test which is a deterministic test $\Omega^*(\theta)$. Let $P^*_e(\theta) = P_e(\Omega^*(\theta)|\theta)$ and define $K_n(\Omega, \theta) = \frac{P_e(\Omega|\theta)}{P^*_e(\theta)}$, we apply competitive analysis and define

$$K_n = \inf_\Omega \sup_\theta K_n(\Omega, \theta).$$

For the competitive minimax criterion, we may get a randomized decision rule as an optimum solution.

5.2 Competitive distribution estimation

Competitive distribution estimation, proposed by A. Orlitsky and A. H. Suresh[4], discusses the problem of estimating distributions over large alphabets from observed data in a competitive manner. One kind of competitor that they choose is a group of so-called "natural estimators" that have the exact knowledge of the underlying distribution. Natural estimators are the estimators that are restricted to assign the same probability to the elements that appear with the same amount of times in the data sequence. This is a reasonable comparison as we expect all data-driven estimators to be natural.

In mathematical form, they seek to solve the following minimax problem.

$$\min_{q} \max_{p \in \triangle_k} \left\{ r_n(q,p) - \min_{q' \in \mathcal{Q}^{\text{nat}}} r_n(q',p) \right\},$$

where $r_n(q,p)$ is the regret for a specific (q,p) pair, $\triangle_k$ is the probability simplex in dimensions of the alphabet size k and set $\mathcal{Q}^{\text{nat}}$ includes all natural estimators.

6 Conclusion

As we have seen, competitive analysis solves the problem that we don't have the same performance benchmark by competing with other outputs, and also give a more realistic output in some problem such as Buy/Rental problem. Consider self-information loss, or say log loss, in the universal prediction problem, it can be reduced to the universal coding problem. For probability setting, with competitive analysis and considering both minimax and maximin, we can observe the "Redundancy-Capacity Theorem", which connects the redundancy and the capacity between a parameter space Θ and a signal space $\mathcal{X}$. For the deterministic setting, without competitive analysis, the problem will become unsolvable because the reality can be strategic and there always exists a competitor that luckily guesses the right answer each round. To overcome this issue, applying competitive analysis, where we only compete with a much smaller set of competitors, gives us a more meaningful problem setup and an acceptable result. As in the probability setting, considering self-information loss, we can connect the universal prediction problem to the online version universal coding problem. Furthermore, by applying the Aggregating Algorithm, we can observe a logarithmic rate of regret, and the output of AA shares the same spirit with the Mixture algorithm in this specific problem.

Division of Work

Weng, Jeng-Chi: 50%, section 3 (Probability setting of universal prediction)

Lee, Chen-Chia: 50%, section 4 (Deterministic setting of universal prediction)

Other sections are finished by both of us.

References

[1] V. Vovk, "Competitive on-line statistics," *International Statistical Review*, vol. 69, no. 2, pp. 213–248, 2001.

[2] L. Davisson, "Universal noiseless coding," *IEEE Transactions on Information Theory*, vol. 19, no. 6, pp. 783–795, 1973.

[3] M. Feder and N. Merhav, "Universal composite hypothesis testing: A competitive minimax approach," *IEEE Transactions on information theory*, vol. 48, no. 6, pp. 1504–1517, 2002.

[4] A. Orlitsky and A. T. Suresh, "Competitive distribution estimation: Why is good-turing good," *Advances in Neural Information Processing Systems*, vol. 28, 2015.