Lifelong Optimization with Low Regret

A Proofs in Section 3

A.1 Proof of Theorem 3.1

Note that the expected loss of our algorithm equals $\sum_{k,s} \mathbb{E}_{g\sim\mathcal{G}_{k,s}}\left[\hat{\ell}_{k,s}(g)\right]$, and the regret of our algorithm can be divided into two parts:

$$\sum_{k,s}\left(\underset{g\sim\mathcal{G}_{k,s}}{\mathbb{E}}\left[\hat{\ell}_{k,s}(g)\right]-\hat{\ell}_{k,s}(g^*)\right) \tag{1}$$

$$+\sum_{k,s}\left(\hat{\ell}_{k,s}(g^*)-\ell_{k,s}(g^*,h_k^*)\right), \tag{2}$$

where h_k^* denotes the optimal predictor for g^* in task k. The sum in Eq.(1) is about learning g, which can be upper-bounded by $\mathtt{reg}_G(T)$. This is because we use the loss function $\hat{\ell}_{k,s}(\cdot)$ for $\mathtt{alg}_G$ to update the distribution $\mathcal{G}_{k,s}$. The sum in Eq.(2) is about learning h and is equal to

$$\sum_k\sum_s\left(\underset{h\sim\mathcal{H}_{k,s}^{(g^*)}}{\mathbb{E}}\left[\ell_{k,s}(g^*,h)\right]-\ell_{k,s}(g^*,h_k^*)\right)$$

which is at most

$$\sum_k \mathtt{reg}_H(T_k)$$

because for any task k, we use the loss function $\ell_{k,s}(g^*,\cdot)$ for $\mathtt{alg}_H^{(g^*)}$ to update the distribution $\mathcal{H}_{k,s}^{(g^*)}$. Theorem 3.1 then follows by combining these two bounds together.

A.2 Proof of Corollary 3.2

For the case with finite G and H and arbitrary loss functions, we can take the multiplicative update (MU) algorithm of Littlestone and Warmuth (1994); Freund and Schapire (1997) for the experts problem and use it for both $\mathtt{alg}_G$ and $\mathtt{alg}_H$, but with different learning rates.[1] From the regret bound of the MU algorithm, we have $\mathtt{reg}_G(T)\le\mathcal{O}(\sqrt{T\log G})$ and

$$\begin{aligned}\sum_{k=1}^K \mathtt{reg}_H(T_k) &\le \sum_{k=1}^K \mathcal{O}\left(\sqrt{T_k\log H}\right)\\ &\le \mathcal{O}\left(\sqrt{K\sum_{k=1}^K T_k\log H}\right)\\ &\le \mathcal{O}\left(\sqrt{KT\log H}\right)\end{aligned}$$

via Cauchy-Schwarz inequality. This proves the corollary.

[1]If we know the time horizon T and the tasks lengths $T_1,\ldots,T_k$ in advance, we can set the learning rates appropriately before hand. Otherwise, we can use the standard doubling tricks to set them adaptively.

A.3 Proof of Corollary 3.3

Since G is finite, we can again use the MU algorithm as $\mathtt{alg}_G$ and have the corresponding part of regret bounded by $\mathcal{O}\left(\sqrt{T\log G}\right)$. For learning predictors, under the assumption of the theorem, we can use the online gradient-descent (OGD) algorithm of Zinkevich (2003) as $\mathtt{alg}_H$. According to the regret bound of OGD, the part of our regret corresponding to learning predictors is at most

$$\sum_{k=1}^K \mathcal{O}\left(DR\sqrt{T_k}\right)\le\mathcal{O}\left(DR\sqrt{KT}\right).$$

Substituting these two bounds into Theorem 3.1, the corollary follows immediately.

A.4 Proof of Theorem 3.4

The lower bound is obtained by considering the following two special cases. First, for the case with $K=1$, the problem reduces to the traditional experts problem with GH experts, which is known to have an $\Omega(\sqrt{T\log(GH)})$ regret lower bound (see e.g. Section 3.7 in (Cesa-Bianchi and Lugosi, 2006)). Next, for the case with $G=1$, let us consider the scenario with each task lasting for T/K steps. Note that each task again reduces to the traditional experts problem with H experts, and each task can be considered separately as each is compared against a different predictor. Therefore, in this scenario, we can establish a regret lower bound of $K\cdot\Omega(\sqrt{(T/K)\log H})=\Omega(\sqrt{KT\log H})$.

Finally, as the problem has these two special cases, we can conclude that it has a regret lower bound of $\max\{\Omega(\sqrt{T\log(GH)}),\Omega(\sqrt{KT\log H})\}\ \ge\ \Omega(\sqrt{T\log G}+\sqrt{KT\log H})$.

A.5 Proof of Theorem 3.5

As we now have an infinite number of representations, we can no longer run the MU algorithm to learn them. Instead, we will run the MU algorithm to learn predictors, but using the set $A = \mathcal{H}^K$ of all possible sequences of K predictors as its action set. The idea is that the objective function of the offline algorithm which we aim to compete to is

$$\min_g \min_{h_1,\dots,h_K} \sum_{k,s} \ell_{k,s}(g,h_k) = \min_{h_1,\dots,h_K} \min_g \sum_{k,s} \ell_{k,s}(g,h_k),$$

and while we considered the former in the case with a finite G, we can instead consider the latter now with a finite H. More precisely, for learning predictors, here we use a single copy of the MU algorithm on the action set $A = \mathcal{H}^K$. For learning representations, we use the OGD algorithm as expected, but for each $\vec{h} \in A$, we have a separate copy, denoted as $\mathtt{OGD}^{(\vec{h})}$. We will run the MU algorithm for learning predictors continuously across different tasks, instead of resetting it for each task.

Algorithm. Formally, at step s of task k, we first sample a vector $\vec{h} = (h_1,\dots,h_K)$ of predictors from A according to some distribution $\mathcal{H}_{k,s}$ of the MU algorithm, and then we obtain a representation $g_{k,s}$ by running the copy $\mathtt{OGD}^{(\vec{h})}$ on loss functions $\ell_{i,j}(\cdot,h_i)$'s, for all previous loss functions $\ell_{i,j}$'s. The joint action we play is $(g_{k,s},h_k)$, and the loss we suffer is $\ell_{k,s}(g_{k,s},h_k)$. After that, we update the distribution of the MU algorithm using the loss function

$$\hat{\ell}_{k,s}(\vec{h}) = \ell_{k,s}\left(\mathtt{OGD}^{(\vec{h})}_{k,s}, h_k\right),$$

where $\mathtt{OGD}^{(\vec{h})}_{k,s}$ denotes the representation the algorithm $\mathtt{OGD}^{(\vec{h})}$ outputs at step s of task k.

Regret. Note that the expected loss of our algorithm is $\sum_{k,s} \mathbb{E}_{\vec{h}\sim\mathcal{H}_{k,s}}\left[\hat{\ell}_{k,s}(\vec{h})\right]$, and the regret of our algorithm can be divided into two parts:

$$\sum_{k,s}\left(\mathop{\mathbb{E}}_{\vec{h}\sim\mathcal{H}_{k,s}}\left[\hat{\ell}_{k,s}(\vec{h})\right] - \hat{\ell}_{k,s}(\vec{h}^*)\right) \quad (3)$$

$$+\sum_{k,s}\left(\hat{\ell}_{k,s}(\vec{h}^*) - \ell_{k,s}(g^*,h_k^*)\right), \quad (4)$$

where h_k^* denotes the optimal predictor for g^* in task k and $\vec{h}^* = (h_1^*,\dots,h_K^*)$. The sum in Eq.(3) is about learning $\vec{h}$, which can be upper-bounded by $\mathcal{O}(\sqrt{T\log|A|}) = \mathcal{O}(\sqrt{TK\log H})$ according to the regret bound of the MU algorithm. The sum in Eq.(4) is about learning g and is equal to

$$\sum_{k,s}\left(\ell_{k,s}\left(\mathtt{OGD}^{(\vec{h}^*)}_{k,s}, h_k^*\right) - \ell_{k,s}(g^*,h_k^*)\right) \le \mathcal{O}(DR\sqrt{T}).$$

This is because we update the algorithm $\mathtt{OGD}^{(\vec{h}^*)}$ using the loss functions $\ell_{k,s}(\cdot,h_k^*)$'s, which are indeed convex functions of g as their second arguments are fixed. By combining these two bounds, Theorem 3.5 then follows.

A.6 Proof of Theorem 3.6

Our approach is to reduce the infinite setting to a finite one, and apply our result in the finite setting. Formally, we discretize $\mathcal{G}$ and $\mathcal{H}$ into finite sets $G' \subseteq \mathcal{G}$ and $\mathcal{H}' \subseteq \mathcal{H}$, respectively, so that for any $g \in \mathcal{G}$ and $h \in \mathcal{H}$, there exist $g' \in \mathcal{G}'$ and $h' \in \mathcal{H}'$, such that $\|g-g'\|_2 \le 1/(C_1T)$ and $\|h-h'\|_2 \le 1/(C_2T)$, and hence

$$\begin{aligned}
&|\ell_{k,s}(g,h) - \ell_{k,s}(g',h')| \\
&\quad\le\ |\ell_{k,s}(g,h) - \ell_{k,s}(g',h)| + |\ell_{k,s}(g',h) - \ell_{k,s}(g',h')| \\
&\quad\le\ 2/T. \qquad (5)
\end{aligned}$$

It is not hard to see that we can have $|\mathcal{G}'| \le T^{\mathcal{O}(n)}$ and $|\mathcal{H}'| \le T^{\mathcal{O}(d)}$, assuming that T is sufficiently large so that the constants R_1, R_2, C_1, C_2 are all at most T (just to make our regret bound cleaner).

Then by running our algorithm for the finite setting, having the set $\mathcal{G}'$ of representations and the set $\mathcal{H}'$ of predictors, we can achieve a regret bound of $\mathcal{O}\left(\sqrt{nT\log T} + \sqrt{dKT\log T}\right)$, from Theorem 3.2. This regret is actually measured against an offline algorithm with representation set $\mathcal{G}'$ and predictor set $\mathcal{H}'$. Nevertheless, the total loss achieved by such an offline algorithm and that using $\mathcal{G}$ and $\mathcal{H}$ differ by at most $T\cdot 2/T = 2$ according to Eq.(5). As a result, we have the theorem.

B Proof of Theorem 4.1

Our algorithm is summarized in Algorithm 1. To show why the algorithm works, there are two important keys. First, we have to show that the representation $\hat{g}$ we identify is the optimal g^* with high probability. This will be shown in Corollary B.2. Second, it is important to have a small $\hat{T}$, since the longer the exploration phase lasts, the higher the regret we will suffer. Afterwards, we can focus on $\hat{g}$ in the exploitation phase, and the problem reduces to an easier one and the regret after iteration $\hat{T}$ can be guaranteed by Lemma B.3.

The first key relies on the following lemma, which shows that the average empirical loss $\bar{L}_t(g)$ of any representation g in iteration t is likely to be close to the average mean loss $\bar{\mu}_t(g)$ defined as

$$\frac{1}{t-1}\sum_{i=1}^{k}\min_{h_i}\sum_{j\in I_i}\mu_i(g,h_i) = \frac{1}{t-1}\sum_{i=1}^{k}\sum_{j\in I_i}\mu_i(g).$$

Algorithm 1 FULL-INFORMATION STOCHASTIC ALGORITHM

```
Parameters: $G, H, T, K, T_k$ for $k \in [K]$.
Exploration phase: In each time iteration $t$, choose $(g_t, h_t)$ by our adversarial algorithm for the case of finite $G$ and finite $H$ with arbitrary loss. Go to the next phase when there is some $\hat{g}$ with
    $\bar{L}_t(\hat{g}) < \bar{L}_t(g) - 2\sigma_t, \quad \forall g \neq \hat{g},$
  where
    $\sigma_t = \sqrt{(c/t)\log(t^2 G H^K/\delta)},$    (6)
  for some large enough constant c (which can be determined in the proof of Lemma B.1).
Exploitation phase: For any task $k$ and any step $s$ in it, we always choose $g_{k,s} = \hat{g}$, and accompany with it the predictor
    $h_{k,s} = \arg\min_h \sum_{j<s} \ell_{k,j}(\hat{g}, h)$    (7)
  which is the best predictor empirically so far in the task.
```

Lemma B.1. *For the choice of σ_t in Eq.(6), we have* $\Pr\left[\forall t \forall g : \left|\bar{L}_t(g) - \bar{\mu}_t(g)\right| \leq \sigma_t\right] \geq 1 - \delta/2$.

We will prove the lemma in Subsection B.1. Note that as both $\bar{L}_t(g)$ and $\bar{\mu}_t(g)$ depend on choosing different predictors in different tasks, we need a lager confidence interval (larger σ_t), compared to the traditional case with only one task. With this lemma, we immediately have the following corollary, which establishes the first key for our algorithm.

Corollary B.2. *If there is any $\hat{g}$ such that $\bar{L}_t(\hat{g}) < \bar{L}_t(g) - 2\sigma_t$ for every $g \neq \hat{g}$, with σ_t defined in Eq.(6), then we have $\hat{g} = g^*$ with probability at least $1 - \delta/2$.*

Proof. As we assume that every g has $\mu_k(g) \geq \mu_k(g^*)$ for any k and thus $\bar{\mu}_t(g) \geq \bar{\mu}_t(g^*)$ for any t, Lemma B.1 implies that with probability at least $1 - \delta/2$,

$$\bar{L}_t(g^*) \leq \bar{\mu}_t(g^*) + \sigma_t \leq \bar{\mu}_t(g) + \sigma_t \leq \bar{L}_t(g) + 2\sigma_t$$

for any t and g. Thus with this probability, we have $\hat{g} = g^*$ as no other g can dominate g^*, given that $\bar{L}_t(g) \geq \bar{L}_t(g^*) - 2\sigma_t$. □

The next key is to show that $\hat{T}$ is actually small. Similarly to previous works in the stochastic setting, our bound depends on some notion of gaps between arms. Recall from Eq. (2) in the main text that

$$\Delta = \min_k \min_{g \neq g^*} \left(\mu_k(g) - \mu_k(g^*)\right),$$

which is the smallest gap between the mean loss of g^* and those of others over tasks. This determines how hard it is to distinguish the optimal representation g^* from suboptimal ones. As $\bar{\mu}_t(g) - \bar{\mu}_t(g^*) \geq \Delta$ for any t, we know from Lemma B.1 that with probability at least $1 - \delta/2$,

$$\bar{L}_t(g) - \bar{L}_t(g^*) \geq \bar{\mu}_t(g) - \bar{\mu}_t(g^*) - 2\sigma_t \geq \Delta - 2\sigma_t$$

for any $g \neq g^*$ and any t. Note that we can have $\sigma_t < \Delta/4$ so that $\bar{L}_t(g) - \bar{L}_t(g^*) > 2\sigma_t$ for any $g \neq g^*$, whenever $t \geq \bar{T}$ for some

$$\bar{T} \leq \mathcal{O}((1/\Delta^2)\log(GH^K/(\Delta\delta))).$$

Therefore, with probability at least $1 - \delta/2$, g^* can dominate others when $t \geq \bar{T}$, which implies that $\hat{T} \leq \bar{T}$. Let us remark that as different tasks require different predictors, it now takes longer to find the optimal g^*, compared to the traditional case with only one task.

We are now ready to analyze the pseudo-regret of our algorithm. Recall that before iteration $\hat{T}$, we runs the adversarial algorithm of Theorem 3.2, which has regret (and hence pseudo-regret) at most

$$\begin{aligned} &\mathcal{O}\left(\sqrt{\hat{T}\log G} + \sqrt{K\hat{T}\log H}\right) \\ \leq\ &\mathcal{O}\left(\sqrt{\hat{T}\log(GH^K)}\right) \\ \leq\ &\mathcal{O}\left((1/\Delta)\log(GH^K/(\Delta\delta))\right) \end{aligned}$$

with probability at least $1 - \delta/2$. The pseudo-regret after iteration $\hat{T}$ can be bounded using standard analysis as the representation is fixed to g^*. Here we need another notion of gaps to capture how hard it is to learn the predictors for g^*. Recall from Eq. (3) in the main text that

$$\triangle_* = \min_k \min_{h \neq h_k^*} \left(\mu_k(g^*, h) - \mu_k(g^*)\right),$$

which is the smallest gap of mean losses from g^*'s suboptimal predictors over tasks. Then we have the following lemma, which we prove in Subsection B.2.

Lemma B.3. *With probability at least $1 - \delta$, the pseudo-regret after iteration $\hat{T}$ is at most*

$$\mathcal{O}((K/\triangle_*)\log(TH/\delta)).$$

Finally, by combining the regret bounds before and after $\hat{T}$, Theorem 4.1 follows.

B.1 Proof of Lemma B.1

Consider any iteration t, any representation g, and any sequence $\vec{h} = (h_1, \ldots, h_{t-1})$ of predictors such that those in the same task are the same. Let $X_t(g,\vec{h})$ denote the random variable $\frac{1}{t-1}\sum_{\tau=1}^{t-1}\ell_\tau(g,h_\tau)$ and $\nu_t(g,\vec{h})$ its mean. Then by Hoeffding bound, we have

$$\Pr\left[\left|X_t(g,\vec{h})-\nu_t(g,\vec{h})\right|>\sigma_t\right]\le e^{-\Omega(\sigma_t^2 t)}\le\frac{\delta}{4t^2GH^K},$$

for the choice of σ_t given in Eq.(6). As the number of such $\vec{h}$ is at most H^K and the number of g is G, a union bound shows that with probability at least $1-\sum_t\delta/(4t^2)\ge 1-\delta/2$, we have:

$$\left|X_t(g,\vec{h})-\nu_t(g,\vec{h})\right|\le\sigma_t \text{ for any } g,\ t, \text{ and such } \vec{h}.$$

As $\bar{L}_t(g)=\min_{\vec{h}}X_t(g,\vec{h})$ and $\bar{\mu}_t(g)=\min_{\vec{h}}\nu_t(g,\vec{h})$, we have with probability at least $1-\delta/2$ that for any g and t,

$$\begin{cases}\bar{L}_t(g)\le\min_{\vec{h}}\left(\nu_t(g,\vec{h})+\sigma_t\right)=\bar{\mu}_t(g)+\sigma_t,\\ \bar{L}_t(g)\ge\min_{\vec{h}}\left(\nu_t(g,\vec{h})-\sigma_t\right)=\bar{\mu}_t(g)-\sigma_t,\end{cases}$$

and hence $|\bar{L}_t(g)-\bar{\mu}_t(g)|\le\sigma_t$.

B.2 Proof of Lemma B.3

For any task k and its step $s\ge 2$, consider the random variable

$$Y_{k,s}(g,h)=\frac{1}{s-1}\sum_{r=1}^{s-1}\ell_{k,r}(g,h)$$

which has mean $\mu_k(g,h)$, for any g and h. By Hoeffding bound, we know that for any k, any $s\ge 2$, and any h,

$$\Pr\left[|Y_{k,s}(g^*,h)-\mu_k(g^*,h)|>\epsilon_s\right]\le e^{-\Omega(\epsilon_s^2 s)}\le\frac{\delta}{2TH},$$

for some $\epsilon_s\le\mathcal{O}(\sqrt{(1/s)\log(TH/\delta)})$. Then by a union bound, we know that with probability at least $1-\delta/2$, the following event happens

$$|Y_{k,s}(g^*,h)-\mu_k(g^*,h)|\le\epsilon_s$$

for any k, any $s\ge 2$, and any h. Suppose the event happens. Then there is some $\hat{S}\le\mathcal{O}((1/\vartriangle_*^2)\log(TH/\delta))$ such that $\epsilon_s<\vartriangle_*/2$ for any $s\ge\hat{S}$ and hence for any k, any $s\ge\hat{S}$, and any $h\ne h_k^*$,

$$\begin{aligned}&Y_{k,s}(g^*,h_k^*)-Y_{k,s}(g^*,h)\\ \le\ &\mu_k(g^*,h_k^*)-\mu_k(g^*,h)+2\epsilon_s\\ \le\ &-\vartriangle_*+2\epsilon_s\\ <\ &0.\end{aligned}$$

This implies that we will choose $h_k^*=\arg\min_h Y_{k,s}(g^*,h)$ as the predictor $h_{k,s}$, according to Eq.(7). By Corollary B.2, we have with probability at least $1-\delta/2$ that $\hat{g}=g^*$ after iteration $\hat{T}$. Therefore, we have with overall probability at least $1-\delta$, the pseudo-regret after iteration $\hat{T}$ is at most

$$\begin{aligned}&\sum_{k=1}^{K}\sum_{s=1}^{\hat{S}}\left(\mu_k(g^*,h_{k,s})-\mu_k(g^*,h_k^*)\right)\\ \le\ &\sum_{k=1}^{K}\sum_{s=1}^{\hat{S}}\left(Y_{k,s}(g^*,h_{k,s})-Y_{k,s}(g^*,h_k^*)+2\epsilon_s\right)\\ \le\ &\sum_{k=1}^{K}\sum_{s=1}^{\hat{S}}2\epsilon_s\end{aligned}$$

according to the choice of $h_{k,s}$ from Eq.(7). Using the upper bound of ϵ_s, the last sum above is at most

$$\sum_{k=1}^{K}\mathcal{O}\left(\sqrt{\hat{S}\log(TH/\delta)}\right)\le\mathcal{O}\left((K/\vartriangle_*)\log(TH/\delta)\right),$$

which proves the lemma.

C Proofs in Section 5

C.1 Proof of Lemma 5.1

Let $\mathcal{Q}_{k,s}$ denote the distribution over the GH^K experts played by the EXP3 algorithm at step s of task k, and let $\bar{\mathcal{Q}}_{k,s}$ denote the corresponding distribution over the actions, with

$$\bar{\mathcal{Q}}_{k,s}(g,h)=\sum_{\vec{h}:h_k=h}\mathcal{Q}_{k,s}(g,\vec{h}),$$

where we use h_k to denote the k-th component of the vector $\vec{h}$.

We prove the lemma by induction. Initially, EXP3 plays any expert $(q,\vec{h})$ with probability $1/(GH^K)$, so that we have

$$\bar{\mathcal{Q}}_{1,1}(g,h)=\sum_{\vec{h}:h_1=h}\mathcal{Q}_{1,1}(g,\vec{h})=\frac{1}{GH}=\mathcal{P}_{1,1}(g,h).$$

Next, assume that at step s of task k, we have $\bar{\mathcal{Q}}_{k,s}(g,h)=\mathcal{P}_{k,s}(g,h)$ for every (g,h). Our goal is to show that the two distributions still match at the next step. For this, we consider the following two cases.

First, when task k does not ends at step s, we have

$$\begin{aligned}\bar{\mathcal{Q}}_{k,s+1}(g,h)&=\sum_{\vec{h}:h_k=h}\mathcal{Q}_{k,s+1}(g,\vec{h})\\ &=\sum_{\vec{h}:h_k=h}\mathcal{Q}_{k,s}(g,\vec{h})\frac{e^{-\eta\bar{\ell}_{k,s}(g,h)}}{Z_{k,s}},\end{aligned}$$

for some normalization factor $Z_{k,s}$, according to the update rule of EXP3 based on the loss estimator $\bar{\ell}_{k,s}$. By definition, the last line above equals

$$\begin{aligned}\bar{\mathcal{Q}}_{k,s}(g,h)\frac{e^{-\eta\bar{\ell}_{k,s}(g,h)}}{Z_{k,s}} &= \mathcal{P}_{k,s}(g,h)\frac{e^{-\eta\bar{\ell}_{k,s}(g,h)}}{Z_{k,s}} \\ &= \mathcal{P}_{k,s+1}(g,h).\end{aligned}$$

where the first equality follows from the inductive hypothesis.

In the second case when task k ends at step s, we have

$$\begin{aligned}\bar{\mathcal{Q}}_{k+1,1}(g,h) &= \sum_{\vec{h}:h_{k+1}=h}\mathcal{Q}_{k+1,1}(g,\vec{h}) \\ &= \sum_{\vec{h}:h_{k+1}=h}\mathcal{Q}_{k,s}(g,\vec{h})\frac{e^{-\eta\bar{\ell}_{k,s}(g,h_k)}}{Z_{k,s}},\end{aligned}$$

according to the update rule of EXP3. Note that for any $\vec{h}$ and $\vec{h'}$ with $h_i = h'_i$ for $i \le k$, we have $\mathcal{Q}_{k,s}(g,\vec{h}) = \mathcal{Q}_{k,s}(g,\vec{h'})$, which implies that the last line above equals

$$\begin{aligned}&\sum_{\vec{h}}\frac{1}{H}\mathcal{Q}_{k,s}(g,\vec{h})\frac{e^{-\eta\bar{\ell}_{k,s}(g,h_k)}}{Z_{k,s}} \\ &\quad= \sum_{h_k}\frac{1}{H}\bar{\mathcal{Q}}_{k,s}(g,h_k)\frac{e^{-\eta\bar{\ell}_{k,s}(g,h_k)}}{Z_{k,s}} \\ &\quad= \sum_{h_k}\frac{1}{H}\mathcal{P}_{k,s}(g,h_k)\frac{e^{-\eta\bar{\ell}_{k,s}(g,h_k)}}{Z_{k,s}}\end{aligned}$$

by inductive hypothesis. The last line above equals

$$\sum_{h'}\mathcal{P}_{k,s}(g,h')\frac{e^{-\eta\bar{\ell}_{k,s}(g,h')}}{\bar{Z}_{k,s}} = \mathcal{P}_{k+1,1}(g,h),$$

with the normalization factor $\bar{Z}_{k,s} = HZ_{k,s}$.

Combining these two cases, we have the lemma by induction.

C.2 Proof of Theorem 5.2

Our algorithm is summarized in Algorithm 2.

Algorithm 2 Adversarial Bandit Algorithm

```
parameters: $\eta \in (0,1)$
Initialization: $\mathcal{P}_{1,1}(g,h) = \frac{1}{GH}$ for $(g,h) \in \mathcal{G}\times\mathcal{H}$.
for task $k = 1,2,\cdots$ do
  for step $s = 1,2,\cdots,T_k$ do
    Sample $(g_{k,s},h_{k,s})$ from the distribution $\mathcal{P}_{k,s}$.
    Receive the loss $\ell_{k,s}(g_{k,s},h_{k,s})$.
    Construct the estimator: for $(g,h) \in \mathcal{G}\times\mathcal{H}$,
        $\bar{\ell}_{k,s}(g,h) = \frac{\ell_{k,s}(g,h)}{\mathcal{P}_{k,s}(g,h)}\mathbb{1}_{g=g_{k,s},h=h_{k,s}}$.
    Update the distribution: for $(g,h) \in \mathcal{G}\times\mathcal{H}$, let
        $\mathcal{P}_{k,s+1}(g,h) = \mathcal{P}_{k,s}(g,h)\cdot\frac{e^{-\eta\bar{\ell}_{k,s}(g,h)}}{Z_{k,s}}$
    if $s < T_k$; otherwise, let
        $\mathcal{P}_{k+1,1}(g,h) = \sum_{h'}\mathcal{P}_{k,s}(g,h')\cdot\frac{e^{-\eta\bar{\ell}_{k,s}(g,h')}}{\bar{Z}_{k,s}}$,
    where $Z_{k,s}$ and $\bar{Z}_{k,s}$ are normalization factors.
  end for
end for
```

To analyze its regret, first note that the expectation of $\bar{\ell}_{k,s}(\cdot)$ conditioned on all previous randomness equals exactly $\ell_{k,s}(\cdot)$, for any k and s. Moreover, Lemma 5.1 shows that our algorithm plays the same distributions of actions as the EXP3 algorithm based on the loss estimators $\bar{\ell}_{k,s}(\cdot)$'s. Therefore, we can follow the standard analysis of the EXP3 algorithm (see e.g. Theorem 2.22 and Theorem 4.1 of Shalev-Shwartz (2011)) and upper-bound the total regret of our algorithm by

$$\frac{\ln(GH^K)}{\eta} + \eta\sum_{k,s}\mathbb{E}\left[\sum_{g,\vec{h}}\mathcal{Q}_{k,s}(g,\vec{h})\left(\bar{\ell}_{k,s}(g,h_k)\right)^2\right],$$

where the expectation is over the randomness of $g_{k,s}$'s and $h_{k,s}$'s which determines $\bar{\ell}_{k,s}$'s. For any k and s, the expectation above according to Lemma 5.1 equals

$$\begin{aligned}&\mathbb{E}\left[\sum_{g,h}\mathcal{P}_{k,s}(g,h)\left(\bar{\ell}_{k,s}(g,h)\right)^2\right] \\ &\quad= \mathbb{E}\left[\sum_{g,h}\mathcal{P}_{k,s}(g,h)\left(\frac{\ell_{k,s}(g,h)}{\mathcal{P}_{k,s}(g,h)}\mathbb{1}_{g=g_{k,s},h=h_{k,s}}\right)^2\right],\end{aligned}$$

by the definition of $\bar{\ell}_{k,s}$. As we assume that $\ell_{k,s}(g,h) \in [0,1]$, the expectation above is at most

$$\begin{aligned}\mathbb{E}\left[\frac{1}{\mathcal{P}_{k,s}(g_{k,s},h_{k,s})}\right] &= \sum_{g,h}\mathcal{P}_{k,s}(g,h)\cdot\frac{1}{\mathcal{P}_{k,s}(g,h)} \\ &= GH.\end{aligned}$$

Therefore, the regret of our algorithm can be upper-bounded by

$$\frac{\ln(GH^K)}{\eta} + \mathcal{O}\left(\eta TGH\right) \le \mathcal{O}\left(\sqrt{TGH\ln(GH^K)}\right)$$

with the choice of

$$\eta = \sqrt{\frac{\ln(GH^K)}{TGH}}.$$

C.3 Proof of Theorem 5.3

The proof is similar to that for Theorem 3.4 in the full-information setting, by considering two special cases. First, for the case with $K = 1$, the problem reduces to the traditional experts problem with GH experts, which is known to have an $\Omega(\sqrt{TGH})$ regret lower bound in the bandit setting (Auer et al., 2002). Next, for the case with $G = 1$, let us consider the scenario with each task lasting for T/K steps. Note that each task again reduces to the traditional experts problem with H experts, and each task can be considered separately as each is compared against a different predictor. Therefore, in this scenario, we can establish a regret lower bound of $K \cdot \Omega(\sqrt{(T/K)H}) = \Omega(\sqrt{KTH})$.

As the problem has these two special cases, we can conclude that it has a regret lower bound of $\max\{\Omega(\sqrt{TGH}), \Omega(\sqrt{KTH})\} = \Omega(\sqrt{TGH} + \sqrt{KTH})$.

C.4 Proof of Theorem 5.4

Algorithm 3 Stochastic Bandit Algorithm

```
**Parameters:** $G, H, T, K, T_k$ for $k \in [K]$, and $\Delta$.
**Exploration phase:** In each iteration $t$, choose $(g_t, h_t)$ according to Algorithm 2. Go to the next phase when
    $t \geq \tilde{T} = c\left(\frac{GH\ln(GH^K)}{\Delta^2}\right)$
for a large enough constant $c$.
**Exploitation phase:** Let $\hat{g}$ be the representation that was played more than $\tilde{T}/2$ iterations in the previous phase. Then for each remaining task, we always choose $\hat{g}$, and we rerun the UCB algorithm to select the accompanying predictors.
```

Our algorithm is summarized in Algorithm 3.

In the first phase, since we choose actions based on our adversarial bandit algorithm, the regret (and hence pseudo-regret) in this phase by Theorem 5.2 is at most $\mathcal{O}(\sqrt{tGH\ln(GH^K)})$. This implies by Markov's inequality that with high probability[2], we have the nice event that the pseudo-regret is at most $c_0\sqrt{tGH\ln(GH^K)}$, for some constant c_0. Let us assume in the following that we indeed have the nice event. Now observe that whenever we fail to play the optimal representation g^* in an iteration, we suffer at least Δ in the pseudo-regret. Therefore, during the first t iterations, the number of iterations that we fail to play g^* is at most

$$\frac{1}{\Delta} \cdot c_0\sqrt{tGH\ln(GH^K)} \leq \frac{t}{2}, \tag{8}$$

[2]with probability $1 - \alpha$, where α can be any constant

whenever

$$t \geq \tilde{T} = c\left(\frac{GH\ln(GH^K)}{\Delta^2}\right)$$

for the constant $c = c_0^2$. That is, during the first $\tilde{T}$ iterations, we choose the optimal representation g^* for the majority of iterations, which implies that we have $\hat{g} = g^*$. The pseudo-regret in the first phase is then bounded by

$$\begin{aligned} c_0\sqrt{\tilde{T}GH\ln(GH^K)} &\leq \frac{\Delta\tilde{T}}{2} \\ &\leq \mathcal{O}\left(\frac{GH\ln(GH^K)}{\Delta}\right), \end{aligned}$$

where the first inequality is because $\tilde{T}$ satisfies the inequality in Eq.(8).

In the second phase, the representation has been fixed to $\hat{g} = g^*$. Therefore, we can simply apply the regret bound of the UCB algorithm to upper-bound the pseudo-regret in this phase by

$$\mathcal{O}\left(\sum_{k=1}^{K}\sum_{h \neq h_k^*}\frac{\ln T}{\mu_k(g^*, h) - \mu_k(g^*)}\right) \leq \mathcal{O}\left(\frac{KH\ln T}{\Delta_*}\right).$$

Then Theorem 5.4 follows by combining the two bounds for the two phases together.

References

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (2002). The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77.

Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

Littlestone, N. and Warmuth, M. K. (1994). The weighted majority algorithm. *Information and computation*, 108(2):212–261.

Shalev-Shwartz, S. (2011). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936.