COMS E6998: Algorithms in LLMs (Fall'26) Oct 19 & 21, 2026

Lectures 12 & 13: Algorithmic Foundations of Mixture of Experts

Instructors: *Alex Andoni, Edo Liberty* Lecturer: *Edo Liberty*

Lecture Overview & Pedagogical Roadmap

Modern frontier Large Language Models decouple **knowledge capacity (total parameter count)** from **active compute per token (FLOPs)** using **Sparse Mixture of Experts (MoE)** layers [7, 10, 5]. Instead of passing every token through a single monolithic Feed-Forward Network (FFN), a lightweight **router** dynamically dispatches each token $x_t \in \mathbb{R}^d$ to a small subset of $k \ll K$ specialized expert sub-networks.

However, training and routing in Sparse MoEs introduce fundamental algorithmic challenges: **partial (bandit) feedback** (we never observe how unchosen experts would have performed on x_t), **routing collapse** (rich-get-richer training dynamics), **non-differentiability** of discrete top-k choices, and **hardware load imbalance**. Across these two 90-minute sessions, we build a rigorous mathematical foundation for expert routing from first principles:

1. **Session 1 — MoE Foundations, Regret & Multiplicative Weights (Sections 1–4):** After motivating MoE architectures (**Mixtral 8x7B** [8], **DeepSeek-V3** [3]) and their core training challenges (**Sections 1–2**), we formalize sequential routing as **Online Learning with Expert Advice** (**Section 3**) and prove the $\mathcal{O}(\sqrt{T \ln K})$ regret bound of **Multiplicative Weights Update (MWU)** [9, 6, 1] in the full-information setting (**Section 4**).

2. **Session 2 — Connection to Learning, Bandits & Contextual Routing (Sections 5–8):** Starting from the $\sqrt{2T \ln K}$ MWU bound, we first show how replacing experts with d-bit binary classifiers ($K \leq 2^d$) yields $R_T = \mathcal{O}(\sqrt{Td})$—recovering the classical $T = \Omega(d/\varepsilon^2)$ **PAC learning sample complexity** completely without i.i.d. randomness assumptions (**Section 5**). We then remove the full-information assumption to study **Multi-Armed Bandits**, proving the $\mathcal{O}(\sqrt{TK \ln K})$ regret bound of **Exp3** [2] via inverse-propensity importance weighting (**Section 6**). Finally, because routing every token to a single fixed expert i^* would be routing collapse, we formulate **Contextual Bandits** (**Section 7**) and analyze the `EXP4` contextual routing algorithm [2], concluding with a sample complexity bound $T = \Omega(Kd/\varepsilon^2)$ for learning d-bit multi-label classifiers and routers under bandit feedback (**Section 8**).

Session 1: MoE Architecture, Routing Challenges, Online Regret & Multiplicative Weights

1 Mixture of Experts in Modern Neural Networks & Its Benefits

1.1 Dense Feed-Forward Networks vs. Sparse Mixture of Experts

In a standard Transformer [11], each token in a sequence is represented by an embedding vector $x \in \mathbb{R}^d$. Every layer processes x through two sub-layers:

1. **Attention:** Mixes information across different tokens in the context window.
2. **Feed-Forward Network (FFN / MLP):** Applies a massive non-linear transformation to each token vector x independently to retrieve stored factual and linguistic knowledge. Roughly 2/3 **to** 3/4 **of all parameters** in an LLM reside in these FFN layers.

In a traditional **dense** LLM, a single giant FFN $f(x) = W_2\sigma(W_1x)$ processes every token x. Consequently, **every token activates** 100% **of the model's parameters**: doubling the model's knowledge capacity (parameter count) automatically doubles the arithmetic compute (FLOPs) required per token.

Sparse Mixture of Experts (MoE) [7, 10, 5] breaks this coupling between **total parameter count** and **compute per token**. An MoE layer replaces the single giant FFN with:

- K **Experts:** Independent smaller feed-forward networks $E_1, E_2, \ldots, E_K : \mathbb{R}^d \to \mathbb{R}^d$.
- **A Router (Gating Network):** A lightweight linear layer with prototype vectors $w_1, \ldots, w_K \in \mathbb{R}^d$ that computes affinity scores $s_i(x) = \langle w_i, x \rangle$ for each expert $i \in [K] \triangleq \{1, \ldots, K\}$.

$$\texttt{Token}\ x \in \mathbb{R}^d \implies \Big[\textbf{Router: } s_i(x) = \langle w_i, x\rangle\Big] \implies \text{Activate ONLY } S(x) = \text{Top-}k(s(x)) \subset [K]$$
$$\text{Active Experts } \{E_i(x)\}_{i\in S(x)} \ \ (\text{other } K-k \text{ experts asleep: 0 FLOPs}) \implies y = \sum_{i\in S(x)} g_i(x) E_i(x)$$

For each incoming token $x \in \mathbb{R}^d$, the router selects only the k **highest-scoring experts** $S(x) = \text{Top-}k(s(x)) \subset [K]$ (where $k \ll K$, typically $k = 1$ or 2 out of $K = 8$ to 256) and computes a sparse weighted combination:

$$y = \sum_{i\in S(x)} g_i(x)\, E_i(x), \qquad \text{where} \qquad g_i(x) = \frac{\exp(s_i(x))}{\sum_{j\in S(x)} \exp(s_j(x))}. \tag{1}$$

Because the remaining $K - k$ experts are never evaluated on x, the compute cost per token scales only with k rather than K.

Model Architecture	Total Parameters (Memory Capacity)	Active Parameters / Token (Compute FLOPs)	Sparsity Ratio (k/K)
Llama-3 70B [4] *(Dense)*	70B	70B (100%)	1/1
Mixtral 8x7B [8] *(Sparse MoE)*	47B	**13B** ($\sim 28\%$)	$k = 2$ of $K = 8$
DeepSeek-V3 [3] *(Fine-Grained MoE)*	671B	**37B** ($\sim 5.5\%$)	$k = 8$ of $K = 256$

Table 1: Comparison of total parameter capacity versus active parameters per token across dense and Sparse MoE frontier models.

1.2 Benefits of MoE in Frontier LLMs

1. **Massive Capacity at Small-Model Compute Cost:** An MoE model can store 10× more knowledge in its parameters while running each token through only a small fraction (k/K) of the weights.
2. $4\times - 7\times$ **Faster Training under Fixed Compute:** For any fixed training FLOP budget, Sparse MoE models reach lower perplexity than dense models [5] because extra experts provide dedicated capacity for long-tail facts and specialized domains.
3. **Emergent Specialization:** Without human supervision, experts naturally partition the token space (e.g., dedicating distinct experts to Python indentation, LaTeX math, multilingual morphology, or punctuation).

2 Main Difficulties in Training and Routing

Introducing a discrete router that activates only $k \ll K$ experts creates four fundamental algorithmic challenges:

1. **Partial / Bandit Feedback (Counterfactual Blindness):** To save compute, we only evaluate the chosen experts $i \in S(x_t)$ on token x_t. We **never observe how well the other $K-k$ unchosen experts would have performed** on x_t. How can the router learn the optimal assignment when it only observes the loss of the experts it actually picks?
2. **Routing Collapse ("The Rich-Get-Richer Trap"):** During training, the router and the experts are optimized jointly. If Expert i happens by random initialization to perform slightly better early in training, the router increases its probability of picking Expert i. Because Expert i receives more tokens, it receives more gradient updates and improves faster, causing the router to send it *even more* tokens. Left unchecked, **a single expert (or k experts) captures** 100% **of all tokens**, while the remaining $K - k$ experts receive zero traffic, zero gradients, and permanently die [10, 5].
3. **Non-Differentiability of Discrete Selection:** The operator $S(x) = \text{Top-}k(s(x))$ is a discrete step function: infinitesimal changes to the router weights w_i do not change the discrete set of selected experts, giving zero gradient almost everywhere with respect to the discrete choice itself.
4. **Hardware Load Imbalance (Parallel GPU Bottlenecks):** In distributed systems, different experts reside on different GPUs. If a batch of n tokens routes 90% of its tokens to Expert 1 and

almost none to Experts $2, \ldots, K$, GPU 1 overflows its memory buffer and bottlenecks execution while $K-1$ GPUs sit idle.

3 Starting Simple: Online Learning & Defining Online Regret

To build a rigorous theory of how a router learns to select experts sequentially—and why randomization is required to avoid collapse and handle partial feedback—we begin with the classical theoretical computer science framework of **Online Learning with Expert Advice** [9, 6, 1].

3.1 The Online Expert Protocol

Consider a sequential game played over T rounds ($t = 1, 2, \ldots, T$) with K available experts (actions) indexed by $[K] = \{1, \ldots, K\}$. At each round t:

1. **Router Decision:** The learner (router) chooses a probability distribution $p_t = (p_{t,1}, \ldots, p_{t,K}) \in \Delta^K$ over the K experts (where $\sum_{i=1}^K p_{t,i} = 1$ and $p_{t,i} \geq 0$), and samples an expert $I_t \sim p_t$.
2. **Loss Realization:** Nature (or an adversary) determines a loss vector $\ell_t = (\ell_{t,1}, \ldots, \ell_{t,K}) \in [0,1]^K$, where $\ell_{t,i}$ measures how poorly expert i performs on round t.
3. **Router Loss:** The learner incurs the loss ℓ_{t,I_t} of the chosen expert, whose expected value over the learner's randomization is:

$$\mathbb{E}_{I_t \sim p_t}\left[\ell_{t,I_t}\right] = \sum_{i=1}^{K} p_{t,i}\ell_{t,i} = \langle p_t, \ell_t \rangle. \tag{2}$$

3.2 Formal Definition of Online Regret

Because the sequence of tokens and losses $\ell_1, \ldots, \ell_T$ can be arbitrary, no online algorithm can guarantee small absolute loss if every expert performs poorly. Instead, we measure performance *relativized* to the best fixed expert in hindsight.

Definition 3.1 (External Regret)**.** *Given a sequence of loss vectors $\ell_1, \ldots, \ell_T \in [0,1]^K$, the **cumulative expected regret** R_T of an online routing algorithm choosing distributions $p_1, \ldots, p_T \in \Delta^K$ is the excess loss compared to the single best fixed expert $i^* \in [K]$ in hindsight:*

$$R_T \triangleq \sum_{t=1}^{T} \langle p_t, \ell_t \rangle - \min_{i^* \in [K]} \sum_{t=1}^{T} \ell_{t,i^*}. \tag{3}$$

*An algorithm is called **no-regret** (or* Hannan consistent*) if its average regret per round vanishes as $T \to \infty$:*

$$\lim_{T\to\infty} \frac{R_T}{T} = 0 \qquad \Longleftrightarrow \qquad R_T = o(T). \tag{4}$$

Why Deterministic Routing Fails ($\Omega(T)$ Regret). Suppose a router deterministically picks a single expert $I_t \in [K]$ at each round (e.g., always picking the expert with the smallest cumulative loss so far). For $K = 2$ experts, an adversary that alternates losses $\ell_1 = (1, 0)$, $\ell_2 = (0, 1)$, $\ell_3 = (1, 0), \ldots$ forces the deterministic router to incur loss 1 on every single round ($\sum_{t=1}^T \ell_{t,I_t} = T$), whereas the best expert in

hindsight has loss $T/2$, yielding linear regret $R_T = T/2 = \Omega(T)$. Thus, **randomized routing ($p_t \in \Delta^K$) is strictly necessary** to achieve sublinear regret.

4 Multiplicative Weights Update (MWU)

We first solve the **full-information setting**: suppose at the end of round t, we evaluate all K experts and observe the entire loss vector $\ell_t = (\ell_{t,1}, \ldots, \ell_{t,K}) \in [0,1]^K$.

The canonical algorithm is **Multiplicative Weights Update (MWU)** (also known as **Hedge** [9, 6, 1]), which maintains a positive weight $w_{t,i}$ for each expert and decays each expert's weight exponentially in its observed loss.

Algorithm `MWU / HEDGE` (Full-Information Online Expert Routing)

Input: Number of experts K, time horizon T, learning rate $\eta > 0$.
Initialize: $w_{1,i} \leftarrow 1$ for all $i \in \{1, \ldots, K\}$.

1. **For** $t = 1, 2, \ldots, T$:
   (a) Compute potential (normalizer): $\Phi_t \leftarrow \sum_{j=1}^K w_{t,j}$.
   (b) Set routing distribution: $p_{t,i} \leftarrow w_{t,i}/\Phi_t$ for all $i \in [K]$.
   (c) Sample expert $I_t \sim p_t$ and observe full loss vector $\ell_t \in [0,1]^K$.
   (d) Multiplicative weight update:

$$w_{t+1,i} \leftarrow w_{t,i} \exp(-\eta \ell_{t,i}) \qquad \forall\, i \in [K].$$

Complexity: $\mathcal{O}(K)$ time and $\mathcal{O}(K)$ memory per round.

Theorem 4.1 (Regret Bound of Multiplicative Weights Update [9, 6, 1])**.** *For any sequence of loss vectors $\ell_1, \ldots, \ell_T \in [0,1]^K$ and any learning rate $\eta > 0$, the Multiplicative Weights Update algorithm satisfies:*

$$\sum_{t=1}^T \langle p_t, \ell_t \rangle - \min_{i^* \in [K]} \sum_{t=1}^T \ell_{t,i^*} \le \frac{\ln K}{\eta} + \frac{\eta}{2} \sum_{t=1}^T \sum_{i=1}^K p_{t,i} \ell_{t,i}^2 \le \frac{\ln K}{\eta} + \frac{\eta}{2} T. \tag{5}$$

Choosing the optimal learning rate $\eta = \sqrt{\frac{2 \ln K}{T}}$ yields sublinear regret:

$$R_T \le \sqrt{2T \ln K}. \tag{6}$$

Proof. We track the evolution of the log-potential $\ln \Phi_t = \ln \left(\sum_{i=1}^K w_{t,i} \right)$ by bounding $\ln(\Phi_{T+1}/\Phi_1)$ from above and below.

Step 1 (One-Step Potential Upper Bound). By definition of the update rule $w_{t+1,i} = w_{t,i} e^{-\eta \ell_{t,i}}$ and $p_{t,i} = w_{t,i}/\Phi_t$, the ratio of consecutive potentials is:

$$\frac{\Phi_{t+1}}{\Phi_t} = \sum_{i=1}^K \frac{w_{t+1,i}}{\Phi_t} = \sum_{i=1}^K \frac{w_{t,i}}{\Phi_t} e^{-\eta \ell_{t,i}} = \sum_{i=1}^K p_{t,i} e^{-\eta \ell_{t,i}}.$$

Since $\eta\ell_{t,i} \geq 0$, we apply the elementary Taylor inequality $e^{-x} \leq 1 - x + \frac{x^2}{2}$ (valid for all $x \geq 0$) to each term:

$$\frac{\Phi_{t+1}}{\Phi_t} \leq \sum_{i=1}^{K} p_{t,i} \left(1 - \eta\ell_{t,i} + \frac{\eta^2}{2}\ell_{t,i}^2\right) = 1 - \eta\langle p_t, \ell_t\rangle + \frac{\eta^2}{2}\sum_{i=1}^{K} p_{t,i}\ell_{t,i}^2. \tag{7}$$

Taking the natural logarithm of both sides and invoking $\ln(1+u) \leq u$ (valid for all $u > -1$):

$$\ln\frac{\Phi_{t+1}}{\Phi_t} \leq -\eta\langle p_t, \ell_t\rangle + \frac{\eta^2}{2}\sum_{i=1}^{K} p_{t,i}\ell_{t,i}^2.$$

Summing this inequality over $t = 1, \ldots, T$ telescopes the left-hand side:

$$\ln\frac{\Phi_{T+1}}{\Phi_1} = \sum_{t=1}^{T}\ln\frac{\Phi_{t+1}}{\Phi_t} \leq -\eta\sum_{t=1}^{T}\langle p_t, \ell_t\rangle + \frac{\eta^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{K} p_{t,i}\ell_{t,i}^2. \tag{8}$$

Step 2 (Potential Lower Bound via the Best Expert i^*). At $t = 1$, $\Phi_1 = \sum_{i=1}^{K} 1 = K$. At $t = T+1$, since all weights are positive, the sum Φ_{T+1} is at least the weight of the single best expert $i^* \in [K]$:

$$\Phi_{T+1} = \sum_{i=1}^{K} w_{T+1,i} \geq w_{T+1,i^*} = \prod_{t=1}^{T} e^{-\eta\ell_{t,i^*}} = \exp\left(-\eta\sum_{t=1}^{T}\ell_{t,i^*}\right).$$

Dividing by $\Phi_1 = K$ and taking logarithms gives:

$$\ln\frac{\Phi_{T+1}}{\Phi_1} \geq -\eta\sum_{t=1}^{T}\ell_{t,i^*} - \ln K. \tag{9}$$

Step 3 (Chaining and Optimizing η). Combining (8) and (9) yields:

$$-\eta\sum_{t=1}^{T}\ell_{t,i^*} - \ln K \leq -\eta\sum_{t=1}^{T}\langle p_t, \ell_t\rangle + \frac{\eta^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{K} p_{t,i}\ell_{t,i}^2.$$

Rearranging and dividing by $\eta > 0$ gives the first inequality in (5). Since $\ell_{t,i} \in [0,1]$, we have $\sum_{i=1}^{K} p_{t,i}\ell_{t,i}^2 \leq \sum_{i=1}^{K} p_{t,i} = 1$, which bounds the second term by $\frac{\eta}{2}T$. Setting $\eta = \sqrt{\frac{2\ln K}{T}}$ equalizes the two terms $\frac{\ln K}{\eta} = \frac{\eta}{2}T = \sqrt{\frac{1}{2}T\ln K}$, yielding $R_T \leq \sqrt{2T\ln K}$. □

Remark 4.2 (Logarithmic Scaling in K). Notice that the regret bound $\sqrt{2T\ln K}$ grows only **logarithmically** with the number of experts K: we can combine $K = 1{,}000{,}000$ experts while paying a regret overhead of only $\sqrt{\ln(10^6)} \approx 3.7$.

Session 2: PAC Learning via MWU, Multi-Armed Bandits (Exp3) & Contextual Bandits

5 MWU Recap & Connection to Learning

5.1 Reminder: The Main Result of Multiplicative Weights Update

At the conclusion of Session 1, we showed that in the full-information online expert game (K experts, T rounds, losses $\ell_t \in [0,1]^K$), maintaining weights $w_{t+1,i} = w_{t,i}e^{-\eta\ell_{t,i}}$ and routing according to $p_{t,i} = w_{t,i}/\sum_{j=1}^K w_{t,j}$ guarantees (Theorem 4.1):

$$R_T = \sum_{t=1}^T \langle p_t, \ell_t\rangle - \min_{i^*\in[K]} \sum_{t=1}^T \ell_{t,i^*} \le \frac{\ln K}{\eta} + \frac{\eta}{2}\sum_{t=1}^T\sum_{i=1}^K p_{t,i}\ell_{t,i}^2 \le \sqrt{2T\ln K} \qquad \left(\text{for } \eta = \sqrt{\frac{2\ln K}{T}}\right). \quad (10)$$

Crucially, this bound makes **zero distributional or stochastic assumptions** on how the losses $\ell_1, \ldots, \ell_T$ are generated, and depends only **logarithmically** ($\sqrt{\ln K}$) on the total number of experts K.

5.2 Connection to Learning: Reconstructing PAC Bounds Without Randomness

Because the regret bound (10) grows only as $\sqrt{\ln K}$, we can use MWU to prove that—at least theoretically, ignoring computational runtime—online weight updates **learn** any finite model class with the exact same sample complexity as classical statistical learning theory.

Setup (Replacing Experts with d-Bit Classification Models).

1. **Models as Experts:** Replace each "expert" $i \in [K]$ with a candidate machine learning model (classifier) $h_i : \mathcal{X} \to \{0,1\}$ for a binary classification task.
2. **Binary 0-1 Loss:** At each round $t = 1, \ldots, T$, an example $(x_t, y_t) \in \mathcal{X} \times \{0,1\}$ arrives. Each model h_i outputs a prediction $h_i(x_t) \in \{0,1\}$ and suffers the 0-1 classification loss:
$$\ell_{t,i} = \mathbb{I}\big[h_i(x_t) \neq y_t\big] = \begin{cases} 0 & \text{if } h_i(x_t) = y_t \ \text{ (correct)}, \\ 1 & \text{if } h_i(x_t) \neq y_t \ \text{ (incorrect)}. \end{cases} \quad (11)$$
3. **Model Size in Bits ($K \le 2^d$):** Suppose the model size is d **bits**—that is, every model h_i in our hypothesis class $\mathcal{H}$ (for example, a neural network or circuit whose weights are represented in d total bits) is uniquely specified by a d-bit binary string. Then the total number of distinct models (experts) is at most:
$$K = |\mathcal{H}| \le 2^d \qquad \Longrightarrow \qquad \ln K = \mathcal{O}(d). \quad (12)$$

Corollary 5.1 (Sample Complexity of Learning d-Bit Models via MWU)**.** *Consider running Multiplicative Weights Update over the class $\mathcal{H}$ of all d-bit binary classifiers ($K \le 2^d$) on any sequence of T labeled*

examples $(x_1, y_1), \ldots, (x_T, y_T) \in \mathcal{X} \times \{0, 1\}$. *The cumulative regret satisfies:*

$$R_T = \sum_{t=1}^{T} \langle p_t, \ell_t \rangle - \min_{h^* \in \mathcal{H}} \sum_{t=1}^{T} \ell_{t,h^*} = \mathcal{O}\big(\sqrt{Td}\big). \tag{13}$$

Consequently, to guarantee that the learner's total regret is at most εT *(*$R_T \leq \varepsilon T$*, so its average per-example classification error is within* ε *of the best* d*-bit model* $h^* \in \mathcal{H}$*), it suffices that the number of examples* T *satisfies:*

$$T = \Omega\left(\frac{d}{\varepsilon^2}\right). \tag{14}$$

Proof. Substituting $K \leq 2^d$ ($\ln K = \mathcal{O}(d)$) directly into the MWU regret bound $R_T = \mathcal{O}(\sqrt{T \ln K})$ of Theorem 4.1 gives:

$$R_T = \mathcal{O}\big(\sqrt{T \ln(2^d)}\big) = \mathcal{O}\big(\sqrt{Td}\big).$$

Requiring that the cumulative regret satisfy $R_T \leq \varepsilon T$ (or equivalently, that the average excess error $\frac{R_T}{T}$ over the T examples be at most ε) holds whenever:

$$\mathcal{O}\big(\sqrt{Td}\big) \leq \varepsilon T \qquad \Longleftrightarrow \qquad T = \Omega\left(\frac{d}{\varepsilon^2}\right). \qquad \square$$

Remark 5.2 (PAC Learning Without Randomness in the Data). In classical **PAC (Probably Approximately Correct) learning**, proving that $T = \mathcal{O}(d/\varepsilon^2)$ examples suffice to learn a hypothesis class of size $|\mathcal{H}| = 2^d$ requires assuming that examples (x_t, y_t) are drawn **i.i.d.** from a fixed unknown distribution $\mathcal{D}$, and applying Hoeffding's inequality plus a union bound over all 2^d hypotheses. Here, MWU reconstructs the exact same $T = \mathcal{O}(d/\varepsilon^2)$ sample complexity bound **without any distributional or i.i.d. assumption on the data**, using a purely deterministic potential-function derivation! (Of course, maintaining $K = 2^d$ weights explicitly takes exponential time $\Theta(2^d)$, which is why practical learning replaces explicit enumeration over 2^d models with gradient-based optimization or structured routing.)

6 Multi-Armed Bandits (MAB) & Short Proof of Exp3

6.1 Why Sparse Routing Is a Multi-Armed Bandit Problem

In Multiplicative Weights Update, updating $w_{t+1,i} = w_{t,i} e^{-\eta \ell_{t,i}}$ requires knowing $\ell_{t,i}$ for **all** K **experts**. In an LLM, computing $\ell_{t,i}$ for all K experts requires running a forward pass through all K expert networks—completely destroying the computational savings of Sparse MoE!

In a **top-1 Sparse MoE**, at each step t we sample and run **only a single expert** $I_t \sim p_t$, and therefore observe **only the single scalar loss** $\ell_{t,I_t} \in [0, 1]$. The losses $\ell_{t,j}$ for all $j \neq I_t$ remain completely unobserved. This partial-feedback setting is the **Multi-Armed Bandit (MAB)** problem [2].

Why Naive Zero-Imputation Fails. Suppose we feed MWU a naive loss vector that sets $\tilde{\ell}_{t,I_t} = \ell_{t,I_t}$ for the chosen expert and $\tilde{\ell}_{t,j} = 0$ for all unchosen experts $j \neq I_t$. Because $\mathbb{E}[\tilde{\ell}_{t,i}] = p_{t,i} \ell_{t,i}$, an expert with high probability $p_{t,i}$ suffers a much larger expected loss update than a rarely chosen expert—creating a severe bias that punishes frequently used experts!

6.2 Importance-Weighted Loss Estimation & The Exp3 Algorithm

To run MWU under bandit feedback, Auer, Cesa-Bianchi, Freund, and Schapire [2] introduced the **inverse-propensity (importance-weighted) loss estimator**, which divides the observed loss by the probability $p_{t,i}$ of having selected expert i:

$$\hat{\ell}_{t,i} \triangleq \frac{\ell_{t,i}}{p_{t,i}} \mathbb{I}[I_t = i] = \begin{cases} \dfrac{\ell_{t,i}}{p_{t,i}} & \text{if } I_t = i, \\ 0 & \text{if } I_t \neq i. \end{cases} \tag{15}$$

Lemma 6.1 (Unbiasedness and Second Moment of $\hat{\ell}_t$)**.** *Conditioned on the router's distribution $p_t \in \Delta^K$ at round t, the estimator $\hat{\ell}_{t,i}$ satisfies for every $i \in [K]$:*

1. ***Unbiasedness:*** $\mathbb{E}_{I_t \sim p_t}\left[\hat{\ell}_{t,i}\right] = \ell_{t,i}$.
2. ***Weighted Second Moment:*** $\mathbb{E}_{I_t \sim p_t}\left[\sum_{i=1}^K p_{t,i} \hat{\ell}_{t,i}^2\right] = \sum_{i=1}^K \ell_{t,i}^2 \leq K$.

Proof. Since $\mathbb{I}[I_t = i]$ takes value 1 with probability $p_{t,i}$ and 0 with probability $1 - p_{t,i}$:

1. $\mathbb{E}_{I_t \sim p_t}[\hat{\ell}_{t,i}] = p_{t,i} \cdot \left(\frac{\ell_{t,i}}{p_{t,i}}\right) + (1 - p_{t,i}) \cdot 0 = \ell_{t,i}$.
2. Similarly, $\mathbb{E}_{I_t \sim p_t}[\hat{\ell}_{t,i}^2] = p_{t,i} \cdot \left(\frac{\ell_{t,i}}{p_{t,i}}\right)^2 = \frac{\ell_{t,i}^2}{p_{t,i}}$. Multiplying by $p_{t,i}$ and summing over $i \in [K]$ cancels $p_{t,i}$ in the denominator:
$$\mathbb{E}_{I_t \sim p_t}\left[\sum_{i=1}^K p_{t,i} \hat{\ell}_{t,i}^2\right] = \sum_{i=1}^K p_{t,i} \cdot \frac{\ell_{t,i}^2}{p_{t,i}} = \sum_{i=1}^K \ell_{t,i}^2 \leq K,$$
since each $\ell_{t,i} \in [0, 1]$. □

The **Exp3** (*Exponential-weight algorithm for Exploration and Exploitation* [2]) algorithm simply runs **MWU** on the unbiased synthetic loss vectors $\hat{\ell}_t$:

Algorithm `EXP3` **(Multi-Armed Bandit Routing [2])**

Input: Number of experts K, time horizon T, learning rate $\eta > 0$.
Initialize: $w_{1,i} \leftarrow 1$ for all $i \in \{1, \ldots, K\}$.

1. **For** $t = 1, 2, \ldots, T$:
   (a) Set routing probabilities: $p_{t,i} \leftarrow w_{t,i} / \sum_{j=1}^K w_{t,j}$.
   (b) Sample a **single** expert $I_t \sim p_t$, run Expert I_t, and observe **only** ℓ_{t,I_t}.
   (c) Construct unbiased loss estimate:
$$\hat{\ell}_{t,i} \leftarrow \frac{\ell_{t,i}}{p_{t,i}} \mathbb{I}[I_t = i] \qquad \forall\, i \in [K].$$
   (d) Multiplicative weight update (only the chosen expert's weight changes!):
$$w_{t+1,i} \leftarrow w_{t,i} \exp(-\eta \hat{\ell}_{t,i}) \qquad \forall\, i \in [K].$$

Theorem 6.2 (Exp3 Regret Bound — Auer, Cesa-Bianchi, Freund, & Schapire, 2002 [2])**.** *For any sequence of loss vectors $\ell_1, \ldots, \ell_T \in [0,1]^K$, running* ***Exp3*** *with learning rate $\eta = \sqrt{\frac{2\ln K}{TK}}$ achieves expected regret:*

$$\mathbb{E}[R_T] = \mathbb{E}\left[\sum_{t=1}^{T} \ell_{t,I_t}\right] - \min_{i^* \in [K]} \sum_{t=1}^{T} \ell_{t,i^*} \leq \sqrt{2TK \ln K}. \tag{16}$$

Proof. Because `Exp3` is identically `MWU` applied to the non-negative loss vectors $\hat{\ell}_1, \ldots, \hat{\ell}_T \in \mathbb{R}^K_{\geq 0}$, the deterministic inequality (5) of Theorem 4.1 holds pathwise for every realization of $I_1, \ldots, I_T$:

$$\sum_{t=1}^{T} \langle p_t, \hat{\ell}_t \rangle - \sum_{t=1}^{T} \hat{\ell}_{t,i^*} \leq \frac{\ln K}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} p_{t,i} \hat{\ell}_{t,i}^2. \tag{17}$$

Taking the expectation $\mathbb{E}[\cdot]$ over the random routing choices $I_1, \ldots, I_T$ and applying the law of iterated expectations at each round t:

- By Lemma 6.1(1), $\mathbb{E}_{I_t}[\langle p_t, \hat{\ell}_t \rangle] = \langle p_t, \mathbb{E}_{I_t}[\hat{\ell}_t] \rangle = \langle p_t, \ell_t \rangle = \mathbb{E}_{I_t}[\ell_{t,I_t}]$, and $\mathbb{E}_{I_t}[\hat{\ell}_{t,i^*}] = \ell_{t,i^*}$.
- By Lemma 6.1(2), $\mathbb{E}_{I_t}\left[\sum_{i=1}^{K} p_{t,i} \hat{\ell}_{t,i}^2\right] = \sum_{i=1}^{K} \ell_{t,i}^2 \leq K$.

Substituting these expectations into (17) yields:

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_{t,I_t}\right] - \sum_{t=1}^{T} \ell_{t,i^*} \leq \frac{\ln K}{\eta} + \frac{\eta}{2} TK. \tag{18}$$

Setting $\eta = \sqrt{\frac{2\ln K}{TK}}$ balances the two terms and gives $\mathbb{E}[R_T] \leq \sqrt{2TK \ln K}$. □

Remark 6.3 (The Statistical Price of Sparsity: $\ln K$ vs. $K \ln K$)*.* Comparing Theorem 4.1 ($\sqrt{2T \ln K}$) with Theorem 6.2 ($\sqrt{2TK \ln K}$) exposes the exact mathematical trade-off of sparse MoE routing: evaluating only 1 out of K experts reduces compute per round by a factor of K, but dividing by $p_{t,i}$ in (15) inflates the second moment from 1 to K, requiring $K\times$ more rounds to achieve the same average regret.

7 Defining Contextual Bandits

7.1 Why Standard Bandits Are Not Enough for MoE

In the Multi-Armed Bandit framework (`Exp3`), the benchmark in (16) is the **single best fixed expert $i^* \in [K]$ across all T rounds**. If an MoE layer converged to routing every token to the same fixed expert i^*, all other $K-1$ experts would go unused—which is literally **routing collapse**!

In an LLM, tokens are not identical: at each round t, the router first observes a **context**—the current token's embedding vector $x_t \in \mathcal{X} \subset \mathbb{R}^d$. The best expert for a Python code token x_t is different from the best expert for a French poetry token $x_{t'}$. Therefore, the router should compete not against a single fixed expert i^*, but against the best **context-dependent routing function (policy)** $\pi^* : \mathcal{X} \to [K]$.

7.2 The Contextual Bandit Protocol & Contextual Regret

At each round $t = 1, 2, \ldots, T$:

1. **Context Arrival:** Nature reveals a context vector (token embedding) $x_t \in \mathcal{X}$, while the loss vector $\ell_t = (\ell_{t,1}, \ldots, \ell_{t,K}) \in [0,1]^K$ for this token remains hidden.
2. **Contextual Routing:** Based on x_t and past history, the router selects a distribution $p_t \in \Delta^K$, samples a single expert $I_t \sim p_t$, and executes $E_{I_t}(x_t)$.
3. **Bandit Feedback:** The router observes **only** the scalar loss $\ell_{t,I_t} \in [0,1]$ of the chosen expert I_t.

Definition 7.1 (Policy Class and Contextual Regret)**.** *Let $\Pi = \{\pi : \mathcal{X} \to [K]\}$ be a hypothesis class of **routing policies**, where each policy $\pi \in \Pi$ maps every context $x \in \mathcal{X}$ to a recommended expert $\pi(x) \in [K]$. The **contextual regret** of a routing algorithm with respect to the policy class Π over T rounds is:*

$$R_T(\Pi) \triangleq \mathbb{E}\left[\sum_{t=1}^{T} \ell_{t,I_t}\right] - \min_{\pi^* \in \Pi} \sum_{t=1}^{T} \ell_{t,\pi^*(x_t)}. \tag{19}$$

Notice that standard Multi-Armed Bandits is the special case where Π consists of the K *constant* policies $\{\pi_1, \ldots, \pi_K\}$ with $\pi_i(x) \equiv i$ for all $x \in \mathcal{X}$.

8 A Simple Contextual Bandits Algorithm (EXP4)

8.1 The EXP4 Algorithm and Regret Bound

Given a finite collection of $N = |\Pi|$ candidate routing policies $\Pi = \{\pi_1, \ldots, \pi_N\}$, how can we learn to route nearly as well as the best policy $\pi^* \in \Pi$ under bandit feedback?

The EXP4 algorithm (*Exponential-weight algorithm for Exploration and Exploitation using Experts*, Auer et al., 2002 [2]) solves this directly by maintaining Multiplicative Weights over all policies $\pi \in \Pi$ and evaluating every policy simultaneously using the importance-weighted arm loss $\hat{\ell}_{t,i}$ from Exp3:

Algorithm EXP4 (Contextual Bandits over a Policy Class Π [2])

Input: Policy class Π, number of experts K, horizon T, learning rate $\eta > 0$.
Initialize: Policy weights $w_{1,\pi} \leftarrow 1$ for all $\pi \in \Pi$.

1. **For** $t = 1, 2, \ldots, T$:

   (a) Observe context $x_t \in \mathcal{X}$.

   (b) Normalize policy distribution: $q_{t,\pi} \leftarrow w_{t,\pi} / \sum_{\pi' \in \Pi} w_{t,\pi'}$.

   (c) Aggregate policy recommendations into an expert routing distribution $p_t \in \Delta^K$:

$$p_{t,i} \leftarrow \sum_{\pi \in \Pi \,:\, \pi(x_t) = i} q_{t,\pi} \qquad \forall\, i \in [K].$$

   (d) Sample expert $I_t \sim p_t$, run Expert $I_t(x_t)$, and observe scalar loss ℓ_{t,I_t}.

   (e) Construct unbiased expert loss vector:

$$\hat{\ell}_{t,i} \leftarrow \frac{\ell_{t,i}}{p_{t,i}} \mathbb{I}[I_t = i] \qquad \forall\, i \in [K].$$

   (f) Update every policy $\pi \in \Pi$ using the estimated loss of its recommended expert $\pi(x_t)$:

$$w_{t+1,\pi} \leftarrow w_{t,\pi} \exp\big(-\eta \hat{\ell}_{t,\pi(x_t)}\big) \qquad \forall\, \pi \in \Pi.$$

Theorem 8.1 (EXP4 Contextual Regret Bound — Auer et al., 2002 [2])**.** *For any finite policy class Π and any sequence of contexts and losses $(x_t, \ell_t)_{t=1}^T$, running EXP4 with learning rate $\eta = \sqrt{\frac{2 \ln |\Pi|}{TK}}$ achieves contextual regret:*

$$R_T(\Pi) \le \sqrt{2TK \ln |\Pi|}. \tag{20}$$

Proof. EXP4 runs MWU over the $|\Pi|$ policies using the synthetic policy loss $\hat{y}_{t,\pi} \triangleq \hat{\ell}_{t,\pi(x_t)}$. By the MWU inequality (5) with $|\Pi|$ in place of K:

$$\sum_{t=1}^{T} \sum_{\pi \in \Pi} q_{t,\pi} \hat{y}_{t,\pi} - \sum_{t=1}^{T} \hat{y}_{t,\pi^*} \le \frac{\ln |\Pi|}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{\pi \in \Pi} q_{t,\pi} \hat{y}_{t,\pi}^2. \tag{21}$$

Because $p_{t,i} = \sum_{\pi : \pi(x_t) = i} q_{t,\pi}$, grouping policies by the expert $i = \pi(x_t)$ they recommend collapses the sums over Π into sums over $[K]$:

$$\sum_{\pi \in \Pi} q_{t,\pi} \hat{y}_{t,\pi} = \sum_{i=1}^{K} p_{t,i} \hat{\ell}_{t,i} \qquad \text{and} \qquad \sum_{\pi \in \Pi} q_{t,\pi} \hat{y}_{t,\pi}^2 = \sum_{i=1}^{K} p_{t,i} \hat{\ell}_{t,i}^2.$$

Taking expectations via Lemma 6.1 shows that $\mathbb{E}\big[\sum_{\pi \in \Pi} q_{t,\pi} \hat{y}_{t,\pi}^2\big] = \sum_{i=1}^{K} \ell_{t,i}^2 \le K$ (growing only with the number of experts K, **not** $|\Pi|$!). Substituting $\eta = \sqrt{\frac{2 \ln |\Pi|}{TK}}$ into (21) yields $R_T(\Pi) \le \sqrt{2TK \ln |\Pi|}$. □

8.2 Concluding Learning Result: Multi-Label Classification & Routing via EXP4

Just as MWU yielded a PAC learning guarantee for binary classification in Section 5, EXP4 immediately yields a general learning result for K**-label classification** (and K**-expert MoE routing**) under **bandit feedback**:

1. Replace each policy $\pi \in \Pi$ with a candidate K**-label classifier** (or router) $\pi : \mathcal{X} \to [K]$ that maps an input $x_t \in \mathcal{X}$ to one of K possible labels (or experts).
2. When the learner predicts label $I_t \in [K]$ on x_t, it observes **only** whether its chosen label I_t was correct or incorrect ($\ell_{t,I_t} \in \{0, 1\}$, or the loss of the chosen expert $E_{I_t}(x_t)$)—without observing the losses of the other $K - 1$ labels/experts.
3. Assume as before that each model $\pi \in \Pi$ is specified using d **bits**, so the total number of distinct models is at most:
$$|\Pi| \leq 2^d \qquad \Longrightarrow \qquad \ln |\Pi| = \mathcal{O}(d). \tag{22}$$

Corollary 8.2 (Sample Complexity of Learning d-Bit Multi-Label Classifiers via EXP4)**.** *Consider running EXP4 over the class Π of all d-bit multi-label classifiers $\pi : \mathcal{X} \to [K]$ ($|\Pi| \leq 2^d$) over T rounds under bandit feedback. The contextual regret satisfies:*

$$R_T(\Pi) = \mathbb{E}\left[\sum_{t=1}^{T} \ell_{t,I_t}\right] - \min_{\pi^* \in \Pi} \sum_{t=1}^{T} \ell_{t,\pi^*(x_t)} = \mathcal{O}\big(\sqrt{TKd}\big). \tag{23}$$

Consequently, to guarantee $R_T(\Pi) \leq \varepsilon T$ (so the average per-example error is within ε of the best d-bit multi-label classifier $\pi^* \in \Pi$), it suffices that the number of examples T satisfies:

$$T = \Omega\left(\frac{Kd}{\varepsilon^2}\right). \tag{24}$$

Proof. Substituting $|\Pi| \leq 2^d$ ($\ln |\Pi| = \mathcal{O}(d)$) into the EXP4 regret bound $R_T(\Pi) \leq \sqrt{2TK \ln |\Pi|}$ of Theorem 8.1 gives:

$$R_T(\Pi) = \mathcal{O}\big(\sqrt{TK \ln(2^d)}\big) = \mathcal{O}\big(\sqrt{TKd}\big).$$

Requiring that the cumulative regret satisfy $R_T(\Pi) \leq \varepsilon T$ holds whenever:

$$\mathcal{O}\big(\sqrt{TKd}\big) \leq \varepsilon T \qquad \Longleftrightarrow \qquad T = \Omega\left(\frac{Kd}{\varepsilon^2}\right). \qquad \square$$

Remark 8.3 (The Bandit Overhead in Multi-Label Learning & MoE Routing). Comparing Corollary 5.1 ($T = \Omega(d/\varepsilon^2)$ under full information) with Corollary 8.2 ($T = \Omega(Kd/\varepsilon^2)$ under bandit feedback) crystallizes our two-session arc: EXP4 learns the best d-bit routing policy or K-label classifier without any distributional assumptions on the data, paying only a linear factor of K in sample complexity for evaluating 1 out of K experts per token.

Summary Table: From Full-Information Experts to Contextual Bandit Routing

Setting	Feedback / Round	Benchmark	Regret (R_T)	Sample Complexity ($R_T \le \varepsilon T$)
Full Information (`MWU / Hedge`)	All K losses $\ell_t \in [0,1]^K$	Best expert $i^* \in [K]$	$\sqrt{2T \ln K}$	$T = \mathcal{O}\left(\frac{\ln K}{\varepsilon^2}\right)$
Learning d-Bit Binary Classifiers (`MWU`)	0-1 losses of $K = 2^d$ models	Best d-bit model $h^* \in \mathcal{H}$	$\mathcal{O}(\sqrt{Td})$	$T = \mathcal{O}\left(\frac{d}{\varepsilon^2}\right)$ *(PAC)*
Multi-Armed Bandits (`Exp3`)	Single chosen loss ℓ_{t,I_t}	Best expert $i^* \in [K]$	$\sqrt{2TK \ln K}$	$T = \mathcal{O}\left(\frac{K \ln K}{\varepsilon^2}\right)$
Contextual Bandits (`EXP4`)	Context x_t + single loss ℓ_{t,I_t}	Best policy $\pi^* \in \Pi$	$\sqrt{2TK \ln \lvert\Pi\rvert}$	$T = \mathcal{O}\left(\frac{K \ln \lvert\Pi\rvert}{\varepsilon^2}\right)$
Learning d-Bit Multi-Label Classifiers (`EXP4`)	Context x_t + single loss ℓ_{t,I_t}	Best d-bit router $\pi^* \in \Pi$ ($\lvert\Pi\rvert \le 2^d$)	$\mathcal{O}(\sqrt{TKd})$	$T = \mathcal{O}\left(\frac{Kd}{\varepsilon^2}\right)$

References

[1] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[2] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multi-armed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[3] DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[4] Abhimanyu Dubey et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[5] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[6] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[7] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[8] Albert Q. Jiang et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[9] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[10] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.