I was working on the contextual bandit problem and thought that we probably do not want to ignore the experts, so instead of talking about conditional probabilities over the actions I thought of learning a probability distribution over the experts. For this scenario what we would want to optimize becomes:

$$\mathbb{E}_{x,l}[\sum_{\pi\in\Pi} p_\pi l(\pi(x))]$$

Where the vector $l \in \mathbb{R}^K$ represents the loss of pulling each arm. I am sure someone has found this connection already but here is what I thought: I want to minimize my objective while keeping the distribution p_π as close as possible to a uniform distribution. This is a regularizer for the space of my probabilities. Therefore the optimization problem I want to solve is given by:

$$\min_p C\sum_{i=1}^{n}\sum_{\pi} p_\pi l_i(\pi(x_i)) + \sum_\pi p_\pi \log p_\pi \tag{1}$$

$$\text{s.t } \sum_\pi p_\pi = 1 \wedge p_\pi \geq 0, \tag{2}$$

where C is a tradeoff parameter.

By using the lagrange multipliers λ and μ_π we can easily solve this problem by solving the KKT equations:

$$C\sum_{i=1}^{n} l_i(\pi(x_i)) + \log(p_\pi) + 1 = \lambda + \mu_\pi$$

$$\mu_\pi p_\pi \ = 0.$$

By solving the first equation we see that $p_\pi \propto \exp\Big(-C\sum_{i=1}^n l_i(\pi(x_i)) + \lambda + \mu_\pi\Big)$. Furhtermore, from the second equation it follows that $\mu_\pi = 0$ and finally because p is a probability vector we must have:

$$p_\pi = \frac{e^{-C\sum_{i=1}^n l_i(\pi(x_i))}}{\sum_\pi e^{-C\sum_{i=1}^n l_i(\pi(x_i))}}.$$

And we have therefore recover the exponential weights algorithm! Again, I am sure someone has thought of this before, however I think that we only need to add to the optimization problem the use of $\widetilde{r}$ instead of r. What we have to justify now is the appeareance of the entropy term in the optimization problem. Intuitively, this should come from the usual learning bounds, we would need to bound the Rademacher complexity by a term depending on the entropy of p or more generally by an arbitrary f-divergence of p and the uniform distribution.