A Convergence Proof

Theorem A.1. *Let $x_1,...,x_n$ be any token sequence generated by an arbitrary language distribution p with an alphabet of size d. Let $p'(x_1,\dots,x_n) = \mathbb{E}_\pi[p(\pi^{-1}(x_1),\dots,\pi^{-1}(x_n))]$. Then, for any $0<\epsilon,\delta<1/2$,*

$$\frac{1}{T}\sum_{t=1}^{T}\|p(x_t|x_1,...,x_{t-1})-p'(x_t|x_1,...,x_{t-1})\|_1 \leq \epsilon$$

with probability greater than $1-\delta$ when $T \geq \frac{d}{\epsilon^4}\operatorname{polylog}(d,\frac{1}{\epsilon},\frac{1}{\delta})$.

Proof. For any desired error $0<\epsilon<1/2$ and failure rate $0<\delta<1/2$, we will first prove the analogous statement for KL divergence instead of $\mathcal{L}_1$ distance, and then relate a bound on KL divergence back to $\mathcal{L}_1$ distance via Pinsker's inequality.

Throughout the rest of proof, we will work with a parameter $\epsilon' < O(\frac{\epsilon}{(\log(1/\delta))^{1/4}}) < \frac{1}{2}$, and will bound our KL divergence by ϵ'.

To prove the bound in terms of KL divergence, it will be useful to ensure to work with a "smoothed" version of p, which we denote by $\tilde{p}$, for which every token has some nonzero probability, σ/d, of appearing at each timestep, for a parameter $\sigma=\delta\epsilon'/T$:

$$\tilde{p}(x_T|x_1,...,x_{T-1}) = p(x_T|x_1,...,x_{T-1})(1-\sigma)+\frac{\sigma}{d}.$$

Similarly, let $\tilde{p}'(x_1,...,x_n)=\mathbb{E}_\pi[\tilde{p}^{-1}(\pi(x_1),...,\pi^{-1}(x_n))]$. We use $\tilde{\mathcal{P}}$ to denote the probabilities under this change. With probability at least $1-\sigma T \geq 1-\epsilon'\delta \geq 1-\frac{\delta}{2}$, the realized sequence $x_1,...,x_n$ drawn under p can be regarded as being drawn from $\tilde{p}$ (as these distributions can be coupled with this probability).

The key idea is then to show that $\tilde{p}'(y_{t+1}|y_{1:t})$, where $y_t = \pi^*(x_t)$ for some ground truth π^* unknown to p', is equivalent to using the multiplicative weights algorithm to predict y_{t+1} with the Hedge strategy, with the experts being each possible permutation of the tokens and the cost incurred by each expert being the negative log likelihood of the prediction. We denote $\tilde{\mathcal{P}}_{\pi'}(y_{1:n}) = \tilde{\mathcal{P}}(y_{1:n}|\pi=\pi') = \tilde{p}(\pi^{-1}(y_1),...,\pi^{-1}(y_n))$ and show this in Lemma A.2.

With this equivalence, we can then bound the difference between the prediction of p and p' as the regret of the multiplicative weights algorithm. Concretely, we show in Lemma A.3 that the regret of p' to any expert $\pi$is bounded as

$$\frac{1}{T}\sum_{t}^{T}\log\frac{\tilde{\mathcal{P}}_\pi(y_{t+1}|y_{1:t})}{\tilde{p}'(y_{t+1}|y_{1:t})} \leq 2\epsilon'^2$$

for $T \geq \left(4\log^2(\frac{d}{\sigma})\log(d!)\right)/\epsilon'^4$.

We can see $\tilde{p}$ as the particular expert/permutation $\tilde{\mathcal{P}}_I$. And we can further only consider the special case that π^* is also the identity permutation, then the same result holds over x_t and with $\tilde{\mathcal{P}}_\pi$ replaced by $\tilde{p}$, i.e.

$$\frac{1}{T}\sum_{t}^{T}\log\frac{\tilde{p}(x_{t+1}|x_{1:t})}{\tilde{p}'(x_{t+1}|x_{1:t})} \leq 2\epsilon'^2$$

Now we want to convert this bound on regret in terms of log likelihood to KL divergence, and eventually to $\mathcal{L}_1$ distance. To convert it to KL divergence regret, we construct a martingale:

$$Z_i = \sum_{t=1}^{i}\left(D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t})) - \log\frac{\tilde{p}(x_{t+1}|x_{1:t})}{\tilde{p}'(x_{t+1}|x_{1:t})}\right).$$

We verify that this is a martingale in Lemma A.4, with differences bounded by $2\log\frac{1}{\sigma}$, and bound the probability that Z_T exceeds $b=\log\frac{d}{\sigma}\sqrt{8T\log\frac{2}{\delta}}$ via Azuma's inequality Lemma A.6: with probability $1-\delta/2$, we have that $|Z_T| \leq b$.

Therefore, we have that with probability at least $1-\delta/2$

$$Z_T=\sum_{t=1}^{T}\left(D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t}))-\log\frac{\tilde{p}(x_{t+1}|x_{1:t})}{\tilde{p}'(x_{t+1}|x_{1:t})}\right)\leq b$$

$$\sum_{t=1}^{T}D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t}))\leq\sum_{t=1}^{T}\left(\log\frac{\tilde{p}(x_{t+1}|x_{1:t})}{\tilde{p}'(x_{t+1}|x_{1:t})}\right)+b$$

Putting this all together, since $\frac{1}{T}\sum_t^T\log\frac{\tilde{p}_i(x_{t+1}|x_{1:t})}{\tilde{p}'(x_{t+1}|x_{1:t})}\leq 2\epsilon'^2$ for $T\geq\left(4\log^2(\frac{d}{\sigma})\log(d!)\right)/\epsilon'^4$, we have the following:

$$\sum_{1}^{T}D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t}))\leq 2\epsilon'^2T+b.$$

We now convert our bound on KL divergence to a bound on $\mathcal{L}_1$ distance via Pinsker's inequality:

$$\|\tilde{p}(x_{t+1}|x_{1:t})-\tilde{p}'(x_{t+1}|x_{1:t})\|_1\leq\sqrt{\frac{1}{2}D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})||\tilde{p}'(x_{t+1}|x_{1:t}))}.$$

Further, at any given x_t, the difference between the redistributed probability distribution $\tilde{p}$ and a unmodified probability distribution p is at most σ, so

$$\|p(x_{t+1}|x_{1:t})-p'(x_{t+1}|x_{1:t})\|_1\leq\|\tilde{p}(x_{t+1}|x_{1:t})-\tilde{p}'(x_{t+1}|x_{1:t})\|_1+2\sigma.$$

We are interested in the average $\mathcal{L}_1$ across time steps:

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\|p(x_{t+1}|x_{1:t})-p'(x_{t+1}|x_{1:t})\|_1&\leq\frac{1}{T}\sum_{t=1}^{T}(\|\tilde{p}(x_{t+1}|x_{1:t})-\tilde{p}'(x_{t+1}|x_{1:t})\|_1+2\sigma)\\&\leq\frac{1}{T}\sum_{t=1}^{T}\sqrt{\frac{1}{2}D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t}))}+2\sigma\\&\leq\frac{1}{T}\sqrt{T\sum_{t=1}^{T}\frac{1}{2}D_{KL}(\tilde{p}(x_{t+1}|x_{1:t})\|\tilde{p}'(x_{t+1}|x_{1:t}))}+2\sigma,\end{aligned}$$

where in the last inequality we applied Cauchy–Schwarz. Hence for $T\geq\left(4\log^2\frac{d}{\sigma}\log(d!)\right)/\epsilon'^4$,

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\|p(x_{t+1}|x_1,...,x_t)-p'(x_{t+1}|x_1,...,x_t)\|_1&\leq\frac{1}{T}\sqrt{\frac{T}{2}(2\epsilon'^2T+b)}+2\sigma\\&\leq\sqrt{\epsilon'^2+\frac{b}{2T}}+2\sigma.\end{aligned}$$

Simplifying this for $b=\log\frac{d}{\sigma}\sqrt{8T\log\frac{2}{\delta}}$, $T\geq\left(4\log^2(\frac{d}{\sigma})\log(d!)\right)/\epsilon'^4$ and $\sigma=\epsilon'\delta/T$, we have

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\|p(x_{t+1}|x_1,...,x_t)-p'(x_{t+1}|x_1,...,x_t)\|_1&\leq\sqrt{\epsilon'^2+\frac{\sqrt{2\log\frac{2}{\delta}}}{\sqrt{\log(d!)}}\epsilon'^2}+\frac{2\epsilon'\delta}{T}\\&\leq\epsilon'(\frac{2\delta}{T}+\sqrt{1+\sqrt{\frac{2\log\frac{2}{\delta}}{\log(d!)}}})\\&\leq\epsilon'(1+\sqrt{1+\sqrt{2\log\frac{2}{\delta}}})\leq\epsilon'2\sqrt{2}(2\log\frac{2}{\delta})^{1/4}.\end{aligned}$$

We can bound this average L_1 error by ϵ if we set $\epsilon' = \frac{\epsilon}{2\sqrt{2}(2\log\frac{2}{\delta})^{1/4}} < \frac{1}{2}$, in which case our condition that $T \geq \left(4\log^2(\frac{dT}{\delta\epsilon'})\log(d!)\right)/\epsilon'^4$ becomes $T \geq \left(512\log\frac{2}{\delta}\log^2\frac{dT}{\delta\epsilon'}\log(d!)\right)/\epsilon^4$. The theorem now follows by simplifying this expression. Since $\log\frac{2}{\delta} \leq 2\log\frac{1}{\delta}$, and $\log(d!) \leq d\log(d)$, we can relax the condition on T as

$$T \geq \left(1024\log\frac{1}{\delta}\log^2(\frac{d}{\delta\epsilon'})\log^2(T)d\log(d)\right)/\epsilon^4 = \log^2(T)\frac{d}{\epsilon^4}\,polylog(d,\frac{1}{\epsilon},\frac{1}{\delta})$$

To remove the $\log^2 T$ from the right side, note that for any $W > 10$, if T > 10 $W\log^2 W$, then $T > Wlog^2 T$, yielding the further relaxed the condition on T as

$$T \geq \frac{d}{\epsilon^4}\,polylog(d,\frac{1}{\epsilon},\frac{1}{\delta}).$$

□

Lemma A.2. *Consider an arbitrary ground truth permutation π^*. For all time steps $t \in [1,n]$, let $y_t = \pi^*(x_t)$. Consider the online prediction game of predicting y_{t+1} at each time step given previous observation $y_{1:t}$ without knowing π^* but knowing $\tilde{p}$. Then, $\tilde{p}'(y_{t+1}|y_{1:t})$ is equivalent to the multiplicative weights algorithm's prediction of y_{t+1} with the Hedge strategy of Freund and Schapire [8], where it*

- *Considers $d!$ experts corresponding to guessing each permutation π' is the ground truth permutation.*
- *Maintains a weight $w_{\pi'}^{(t)}$ for each expert at time step t, and the weights are initially as $\tilde{\mathcal{P}}(\pi)$.*
- *Picks a distribution across experts $p_{\pi'}^{(t)} = \frac{w_{\pi'}^{(t)}}{\Phi^{(t)}}$ where $\Phi^{(t)} = \sum_j w_j^{(t)}$.*
- *Produces prediction of y_{t+1} as $\sum_{\pi'} p_{\pi'}^{(t)} \tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})$*
- *Receives a cost vector of $m_{\pi'}^{(t)} = -\frac{1}{\epsilon}\log\tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})$.*
- *Updates the weights $w_i^{(t+1)} = w_i^{(t)}\exp(-\epsilon m_i^{(t)})$ and repeat*

Proof. We can first see that $p_{\pi'}^{(t)} = \tilde{\mathcal{P}}(\pi'|y_{1:t})$ by induction:

Base case: $p_{\pi'}^{(0)} = \tilde{\mathcal{P}}(\pi)$ by assumption.

Inductive Case:

With the cost vector as $m_{\pi'}^{(t-1)} = -\frac{1}{\epsilon}\log\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})$, the update at step t is $w_{\pi'}^{(t)} = w_{\pi'}^{(t-1)}\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})$. Therefore, the probability over any particular expert π' is

$$\begin{aligned}
p_{\pi'}^{(t)} &= \frac{w_{\pi'}^{(t)}}{\Phi^{(t)}} \\
&= \frac{w_{\pi'}^{(t-1)}\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})}{\sum_j w_j^{(t-1)}\tilde{\mathcal{P}}_j(y_t|y_{1:t-1})} \\
&= \frac{p_{\pi'}^{(t-1)}\Phi^{(t-1)}\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})}{\sum_j p_j^{(t-1)}\Phi^{(t-1)}\tilde{\mathcal{P}}_j(y_t|y_{1:t-1})} \\
&= \frac{p_{\pi'}^{(t-1)}\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})}{\sum_j p_j^{(t-1)}\tilde{\mathcal{P}}_j(y_t|y_{1:t-1})}
\end{aligned}$$

This is equivalent to the update given by Bayes' rule when plugging in $p_{\pi'}^{(t)} = \tilde{\mathcal{P}}(\pi'|y_{1:t})$:

$$\tilde{\mathcal{P}}(\pi'|y_{1:t}) = \frac{\tilde{\mathcal{P}}(\pi'|y_{1:t-1})\tilde{\mathcal{P}}_{\pi'}(y_t|y_{1:t-1})}{\tilde{\mathcal{P}}(y_t|y_{1:t-1})}$$

So we can conclude that $p_{\pi'}^{(t)} = \tilde{\mathcal{P}}(\pi'|y_{1:t})$, i.e. the process of updating the probability distribution across experts within the prediction game is equivalent to the process of the language model updating the probabilities $\tilde{\mathcal{P}}(\pi'|y_{1:t+1})$ across permutations π'. And this means that the algorithm's prediction $\sum_{\pi'} p_{\pi'}^{(t)} \tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t}) = \sum_{\pi'} \tilde{\mathcal{P}}(\pi'|y_{1:t}) \tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t}) = \tilde{\mathcal{P}}(y_{t+1}|y_{1:t}) = \tilde{p}'(y_{t+1}|y_{1:t})$ □

Lemma A.3. *When using the Hedge strategy for the multiplicative weights algorithm, the average difference between the weighted distribution across experts and any particular expert π is bounded as*

$$\frac{1}{T}\sum_{t}^{T} \log \frac{\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})}{\tilde{p}'(y_{t+1}|y_{1:t})} \leq 2\epsilon^2$$

for $\epsilon \leq 1$ and for $T \geq \left(4\log^2\left(\frac{d}{\sigma}\right)\log(d!)\right)/\epsilon^4$.

Proof. Consider an arbitrary expert π.

We first show that the cost vectors are bounded by $\rho = -\frac{1}{\epsilon}\log\frac{\sigma}{d}$: Recall we defined $m_{\pi}^{(t)} = -\frac{1}{\epsilon}\log\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})$. By the definition of our redistributed probability distribution, at time step $t \in [1,...,T]$,

$$\begin{aligned}
\frac{\sigma}{d} &\leq \tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t}) \leq 1 \\
\log\frac{\sigma}{d} &\leq \log\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t}) \leq 0 \\
0 \leq m_{\pi}^{(t)} &\leq -\frac{1}{\epsilon}\log\frac{\sigma}{d} \\
0 \leq m_{\pi}^{(t)} &\leq -\frac{1}{\epsilon}\log\frac{\sigma}{d}.
\end{aligned}$$

By corollary 16.3 in [1], if we have cost vectors $m^{(t)} \in [-\rho,\rho]^{d!}$, then for time $T \geq (4\rho^2\log(d!))/\epsilon^2$ where $\epsilon \leq 1$,

$$\frac{1}{T}\sum_{t}^{T} p^{(t)} \cdot m^{(t)} \leq \frac{1}{T}\sum_{t}^{T} m_{\pi}^{(t)} + 2\epsilon.$$

Note that we can simplify $T \geq \left(4\log^2\left(\frac{d}{\sigma}\right)\log(d!)\right)/\epsilon^4$.

We can now bound

$$\begin{aligned}
\frac{1}{T}\sum_{t}^{T}\left(p^{(t)} \cdot m^{(t)} - m_{\pi}^{(t)}\right) &\leq 2\epsilon \\
\frac{1}{T}\sum_{t}^{T}\left(\sum_{\pi'} p_{\pi'}^{(t)} m_{\pi'}^{(t)} - m_{\pi}^{(t)}\right) &\leq 2\epsilon \\
\frac{1}{T}\sum_{t}^{T}\left(\sum_{\pi'}\tilde{\mathcal{P}}(\pi'|y_{1:t})\left(-\frac{1}{\epsilon}\log\tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})\right) - \left(-\frac{1}{\epsilon}\log\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})\right)\right) &\leq 2\epsilon \\
\frac{1}{\epsilon T}\sum_{t}^{T}\sum_{\pi'}\left(\tilde{\mathcal{P}}(\pi'|y_{1:t})\left(\log\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t}) - \log\tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})\right)\right) &\leq 2\epsilon \\
\frac{1}{T}\sum_{t}^{T}\mathbb{E}_{\pi'}\log\frac{\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})}{\tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})} &\leq 2\epsilon^2
\end{aligned}$$

By Jensen's inequality, we also have that

$$\begin{aligned}
\frac{1}{T}\sum_{t}^{T}\log\frac{\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})}{\mathbb{E}_{\pi'}\tilde{\mathcal{P}}_{\pi'}(y_{t+1}|y_{1:t})} &\leq 2\epsilon^2 \\
\frac{1}{T}\sum_{t}^{T}\log\frac{\tilde{\mathcal{P}}_{\pi}(y_{t+1}|y_{1:t})}{\tilde{p}'(y_{t+1}|y_{1:t})} &\leq 2\epsilon^2
\end{aligned}$$

$\square$

Lemma A.4. *Let*

$$Z_i = \sum_{t=1}^{i} \left(D_{KL}(\tilde{\mathcal{P}}_I(x_{t+1}|x_{1:t}) \| \tilde{\mathcal{P}}(x_{t+1}|x_{1:t})) - \log \frac{\tilde{\mathcal{P}}_I(x_{t+1}|x_{1:t})}{\tilde{\mathcal{P}}(x_{t+1}|x_{1:t})} \right)$$

Z_i is a martingale.

Proof. Consider

$$\begin{aligned}
\mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}[Z_i] &= \mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}\left[\sum_{t=1}^{i} \left(D_{KL}(\tilde{\mathcal{P}}_I(x_{t+1}|x_{1:t}) \| \tilde{\mathcal{P}}(x_{t+1}|x_{1:t})) - \log \frac{\tilde{\mathcal{P}}_I(x_{t+1}|x_{1:t})}{\tilde{\mathcal{P}}(x_{t+1}|x_{1:t})} \right)\right] \\
&= \mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}\left[D_{KL}(\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i}) \| \tilde{\mathcal{P}}(x_{i+1}|x_{1:i})) - \log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})} + Z_{i-1} \right]
\end{aligned}$$

Observe that Z_{i-1} has no dependence on x_{i+1}.

$$\begin{aligned}
\mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}[Z_i] &= \mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}\left[\mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I} \log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})}\right] - \mathbb{E}_{x_{i+1}\sim\tilde{\mathcal{P}}_I}\left[\log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})}\right] + Z_{i-1} \\
&= Z_{i-1}
\end{aligned}$$

Therefore, Z_i is a martingale. $\square$

Lemma A.5. $|Z_i - Z_{i-1}| \le c_i$ *where* $c_i = 2|\log \frac{d}{\sigma}|$

Proof. We have

$$|Z_i - Z_{i-1}| = \left| D_{KL}(\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i}) \| \tilde{\mathcal{P}}(x_{i+1}|x_{1:i})) - \log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})} \right|$$

In our redistributed probability distribution $\tilde{\mathcal{P}}$, we have $\frac{\sigma}{d} \le \tilde{\mathcal{P}}_\pi(x_i|x_{1:i-1}) \le 1$ for any π at any time i. Therefore,

$$\log \frac{\sigma}{d} \le \log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})} \le \log \frac{d}{\sigma}.$$

Also, we can find an upper bound for the KL divergence by maximizing $\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})$ to 1 and minimizing $\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})$ to $\frac{\sigma}{d}$ so that

$$\begin{aligned}
D_{KL}(\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i}) \| \tilde{\mathcal{P}}(x_{i+1}|x_{1:i})) &= \sum_{x_{i+1}} \tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i}) \log \frac{\tilde{\mathcal{P}}_I(x_{i+1}|x_{1:i})}{\tilde{\mathcal{P}}(x_{i+1}|x_{1:i})} \\
&\le \log \frac{d}{\sigma}
\end{aligned}$$

We can maximize $|Z_i - Z_{i-1}|$ by maximizing the first term and minimizing the second term, or vice versa. In the first case, $|Z_i - Z_{i-1}| \le |\log \frac{d}{\sigma} - \log \frac{\sigma}{d}| = 2|\log \frac{d}{\sigma}|$. In the other case, $|Z_i - Z_{i-1}| \le |0 - \log \frac{d}{\sigma}| = |\log \frac{d}{\sigma}|$.

Therefore, $|Z_i - Z_{i-1}| \le c_i$ where $c_i = 2|\log \frac{d}{\sigma}|$. $\square$

Lemma A.6. *By Azuma's inequality, with probability* $1-\delta$*, we have that* $\|Z_T\| \le b$ *where* $b = 2\log \frac{d}{\sigma} \sqrt{-8T \log \frac{1}{\delta}}$

Proof. By Azuma's inequality, for all positive reals b,

$$\begin{aligned}
P(Z_T - Z_1 \geq b) &\leq \exp\left(\frac{-b^2}{2\sum_{k=2}^{T} c_k^2}\right) \\
P(Z_T - Z_1 \leq b) &\geq 1 - \exp\left(\frac{-b^2}{2\sum_{k=2}^{T} c_k^2}\right) \\
&\geq 1 - \exp\left(\frac{-b^2}{8\sum_{k=2}^{T} \log^2 \frac{d}{\sigma}}\right)
\end{aligned}$$

We can rewrite in terms of $\delta = \exp\left(\frac{-b^2}{8\sum_{k=2}^{T} \log^2 \frac{d}{\sigma}}\right)$ so

$$\begin{aligned}
b &= \sqrt{-\left(8\sum_{k=2}^{T} \log^2 \frac{d}{\sigma}\right) \log \delta} \\
&\leq \log \frac{d}{\sigma} \sqrt{-8T \log \frac{1}{\delta}}
\end{aligned}$$

Therefore,

$$P(Z_T - Z_1 \leq b) \geq 1 - \delta$$

□

B Model Architecture Details

In addition, we add a learnable scaling and bias parameter to the result of the embedding layer, so that the model can still learn to scale it as needed.

C Convergence on other datasets

Figure 7 shows the perplexity of lexinvariant LMs across the three different datasets. Note that Github converges significantly faster than standard Engish text like Wiki-40B, since code is more structured and easier to decipher the token permutation.

D Code Deciphering Full Examples

Java:

```
binary_search()z
  if (high >= low)z
    mid = (high + low) / 2;
    if (arr[mid] == x)
      return mid;
    if (arr[mid] > x)z
      high = mid - 1;
      return binary_search();
    } elsez
      low = mid + 1;
      return binary_search();
    }
  } elsez
    return -1;
  }
}
void func2()z
```

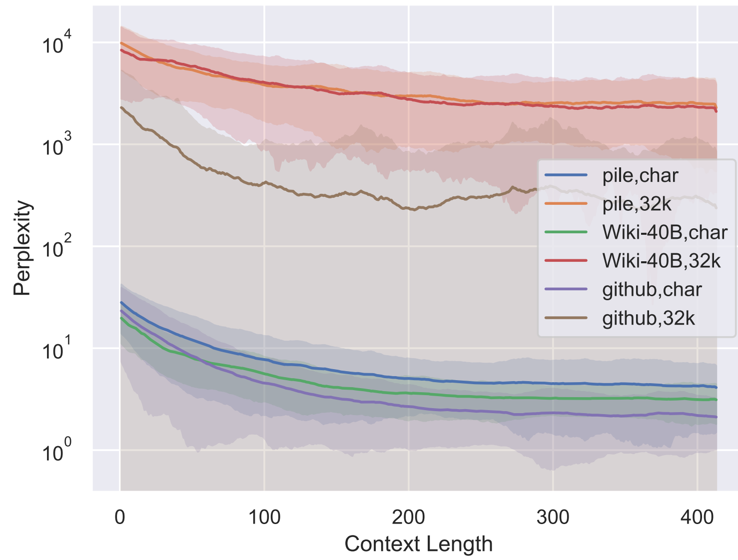

Figure 7: Smoothed Token Perplexity over the Pile, Wiki-40B and Github, with character-level and T5 default vocab

Python:

```
binary_search()z
  if (high >= low)z
    mid = (high + low) // 2
    if (arr[mid] == x)z
      return mid
    if (arr[mid] > x)z
      high = mid − 1
      return binary_search()
    elsez
      low = mid + 1
      return binary_search()
  elsez
      return −1
def func2()z
```

E Semantic Deciphering Full Example

’crash!’ ’aaah!’ i looked up from my cup of coffee. ’crash!’ - that was the cafe window. and ’aaah!’ - that was kate. people in the cafe shouted. kate and i ran to the window. there was no one there. then i turned to kate and put my arm around her. ’are you all right?’ i asked. ’yes,’ she said. ’i think so.’ ’what is it?’ some one shouted and a short red-faced man ran into the room. the man took my arm. ’matt! what are you doing to kate?’ he asked. ’nothing, papa,’ kate replied. ’it wasn’t him. it was from out in the street.’ the red-faced man looked at the window and then at me. he turned to his daughter. ’are you ok, kate?’ he asked. kate gave him a little smile. ’yes, i think i am, papa,’ she said. then her father spoke to me. ’sorry, matt. i heard kate and i thought...’ ’that’s ok, paolo,’ i answered. it was ok. you see, this is soho, in the centre of london. in the day it’s famous for music and films. at night people come and eat and drink

```
in the restaurants.  expensive restaurants and cheap restaurants; italian
restaurants and chinese restaurants.  and day and night there are internet
cafes like the web cafe.  in soho you can buy any thing and any one.  there
are lots of nice people in soho.  but there are also lots of people who are
not very nice.  i know because i live and work here.  i often take a drink to
a shop or cafe.  i'm not rich and famous.  and i don't know a lot.  but i do
know soho.  what one here is a drink - restaurants - music - coffee - father
the one here that drink is
```

Example prediction of the lexinvariant with 32k vocabulary train on the Pile:

```
 - coffee.  and i
```

F Synthetic Reasoning Task

Table 2 shows a variant of the synthetic reasoning task results in Subsection 1, where the symbols are instead sampled proportion to the token frequencies. Although the improvement still generally holds, the standard LM with character-based vocabulary becomes significantly better. We believe that this is because the model can get a significant advantage by guessing among the most common letter.

Dataset	Vocab	LookUp Acc		Permutation Acc	
		Standard	LI	Standard	LI
Pile	char	72.80	90.95	40.63	60.47
	32k	61.20	90.95	40.55	54.55
Wiki-40B	char	75.55	63.45	42.71	59.86
	32k	41.05	57.95	26.81	51.86
Github	char	66.00	86.75	36.62	70.77
	32k	59.25	78.45	37.46	65.04

Table 2: Synthetic Reasoning Tasks (adjusted for token frequencies)

G Language Models Regularized with Lexinvariance and BIG-bench Results

As described in the main paper, we implement a lexinvariance regularized Model in a way similar to embedding dropout. Note that one problem in implementing it naively by using random Gaussian embeddings and learned embedding in a mixture is that the two would become quickly distinguishable from each other during training since learned embeddings often have larger norms, allowing the model simply ignore the randomized tokens. So instead of using random Gaussian embedding matrices in place of a learned embedding matrix, we explored another approach for training a lexinvariant regularized LM: training a standard LM with learnable embedding matrix over sequences partially applied with a random token permutation $B_p(x_1,\pi),...,B_p(x_1,\pi)$, where $B_p(x_i,\pi) = \pi(x_i)$ with probability p and $B_p(x_i,\pi)=x_i$ with probability $1-p$. Since each token can be remapped to any other token with equal chance, the produced model should ideally also be lexinvariant when $p=1$, though with no strict guarantees. In practice, we found the models trained this way behave very similarly to models with random Gaussian embedding.

We evaluate our model over BIG-bench tasks where the language model performance scales well, and we prioritize evaluating generative tasks over multiple-choice tasks. Tasks we evaluated on:

gre reading comprehension.mul, linguistics puzzles.gen, linguistics puzzles.gen, rhyming.gen, tellmewhy.gen, simple arithmetic multiple targets json.gen, simple arithmetic json subtasks.gen, disfl qa.gen, arithmetic.gen, bridging anaphora resolution barqa.gen, matrixshapes.gen, sufficient information.gen, logical args.mul, novel concepts.mul, code line description.mul, unnatural in context learning.gen, unit interpretation.mul, english proverbs.mul, general knowledge.mul, geometric shapes.gen, human organs senses.mul, contextual parametric knowledge conflicts.gen, crass ai.mul, auto categorization.gen, penguins in a table.gen, hindu knowledge.mul, english russian proverbs.mul, modified arithmetic.gen, cryobiology spanish.mul, evaluating information essentiality.mul, intent recognition.mul, understanding fables.mul, figure of speech detection.mul, empirical judgments.mul, simple ethical questions.mul, swahili english proverbs.mul, language identification.mul, phrase relatedness.mul, nonsense words grammar.mul, undo permutation.mul, object counting.gen, identify odd metaphor.mul, elementary math qa.mul, social iqa.mul, parsinlu qa.mul, metaphor understanding.mul, timedial.mul, causal judgment.mul, list functions.gen, implicatures.mul, date understanding.mul, codenames.gen, fact checker.mul, physics.mul, abstract narrative understanding.mul, emojis emotion prediction.mul, metaphor boolean.mul, strategyqa.gen, ascii word recognition.gen, auto debugging.gen, cause and effect.mul, conlang translation.gen, cryptonite.gen, cs algorithms.mul, dyck languages.mul, gender inclusive sentences german.gen, hindi question answering.gen, international phonetic alphabet transliterate.gen, irony identification.mul, logical fallacy detection.mul, movie dialog same or different.mul, operators.gen, paragraph segmentation.gen, parsinlu reading comprehension.gen, repeat copy logic.gen, rephrase.gen, simple arithmetic json.gen, simple arithmetic multiple targets json.gen, sports understanding.mul, word unscrambling.gen, hyperbaton.mul, linguistic mappings.gen, anachronisms.mul, indic cause and effect.mul, question selection.mul, hinglish toxicity.mul, snarks.mul, vitaminc fact verification.mul, international phonetic alphabet nli.mul, logic grid puzzle.mul, natural instructions.gen, entailed polarity.mul, list functions.gen, conceptual combinations.mul, goal step wikihow.mul, logical deduction.mul, conlang translation.gen, strange stories.mul, odd one out.mul, mult data wrangling.gen, temporal sequences.mul, analytic entailment.mul, disambiguation qa.mul, sentence ambiguity.mul, swedish to german proverbs.mul, logical sequence.mul, chess state tracking.gen, reasoning about colored objects.mul, implicit relations.mul, riddle sense.mul, physical intuition.mul, simple arithmetic json multiple choice.mul, geometric shapes.gen, gem.gen, simp turing concept.gen, common morpheme.mul, qa wikidata.gen, international phonetic alphabet transliterate.gen, similarities abstraction.gen, rephrase.gen, emoji movie.gen, qa wikidata.gen, word sorting.gen, emoji movie.gen, qa wikidata.gen, periodic elements.gen, hindi question answering.gen

Bellow, we plot the net percentage of tasks improved/deproved in each of the BIG-bench categories, out of the tasks that are changed by at least a threshold amount.

H Compute

We use one TPU v3-8 for all our pretraining runs. It takes approximately 23 hours for each pretraining run.

I Broader Impacts

Our work primarily provides a scientific exploration and understanding of the properties of lexinvariant language models. More broadly, these properties could potentially help improve the robustness, generalizability, and reasoning ability of LMs in the future works. In general we don't foresee more specific negative societal impacts from this work other than general misuse of language models.

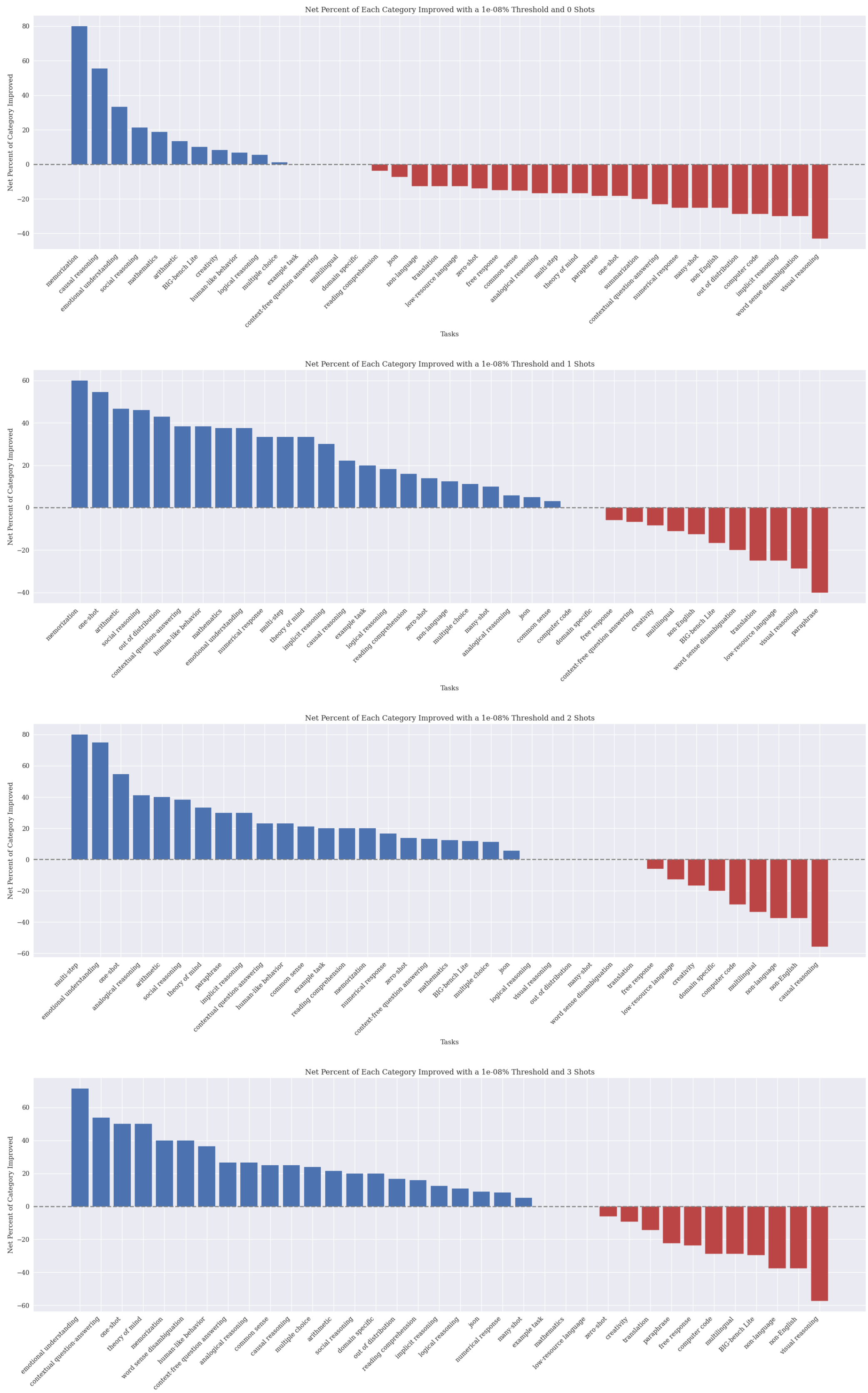

Figure 8: BIG-bench results with 0,1,2 and 3 shots.