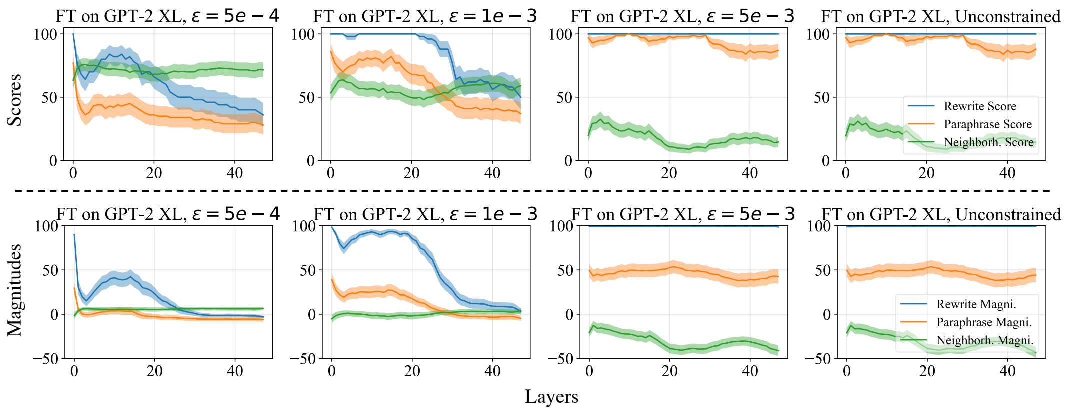

Figure 17: **GPT-2 XL hyperparameter sweeps across layer and L_∞ constraint values for fine-tuning-based methods**. Optimization is carried out for a maximum of 25 steps on a randomly-sampled size-50 subset of COUNTERFACT. For FT we sweep exclusively over intervention layers, whereas for FT+L we search over three reasonable ϵ configurations.

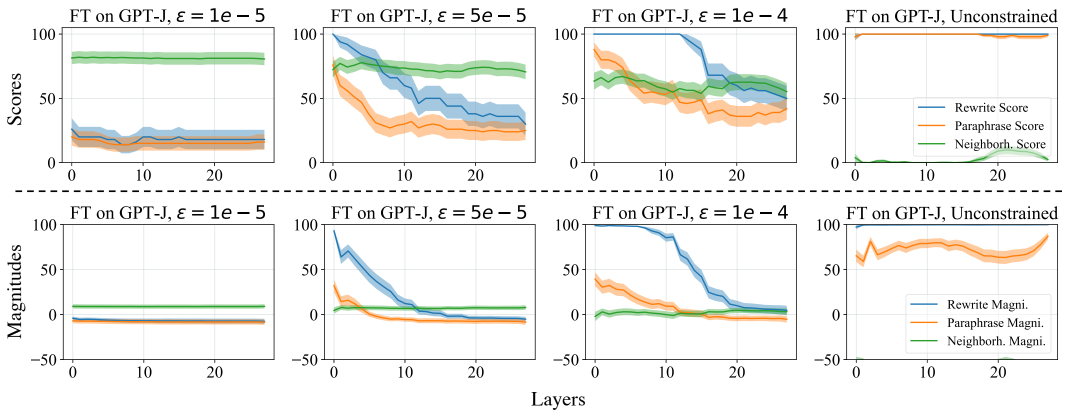

Figure 18: **GPT-J hyperparameter sweeps**. The experimental setup is identical to that of GPT-2 XL.

E Method Implementation Details

E.1 [GPT-2 XL, GPT-J] Fine-Tuning (FT), Constrained Fine-Tuning (FT+L)

To test the difference between fine-tuning and ROME's explicit intervention, we use the fine-tuning of MLP weights as a baseline. Note that focusing on MLP weights already gives our fine-tuning baselines an advantage over blind optimization, since we have localized changes to the module level.

For basic Fine-Tuning (FT), we use Adam Kingma & Ba (2015) with early stopping to minimize $-\log \mathbb{P}_{G'}[o^* \mid p]$, changing only mlp_{proj} weights at one layer. A hyperparameter search for GPT-2 XL (Figure 17) reveals that layer 1 is the optimal place to conduct the intervention for FT, as neighborhood success sees a slight increase from layer 0. Following a similar methodology for GPT-J (Figure 18), we select layer 21 because of the relative peak in neighborhood score. For both models, we use a learning rate of 5×10^{-4} and early stop at a 0.03 loss.

For *constrained* fine-tuning (FT+L), we draw from Zhu et al. (2020) by adding an L_∞ norm constraint: $\|\theta_G - \theta_{G'}\|_\infty \leq \epsilon$. This is achieved in practice by clamping weights θ'_G to the $\theta_G \pm \epsilon$ range at each gradient step. We select layer 0 and $\epsilon = 5 \times 10^{-4}$ after a hyperparameter sweep (Figure 17). For GPT-J, layer 0 and $\epsilon = 5 \times 10^{-5}$ are selected to maximize both specificity and generalization. The learning rate and early stopping conditions remain from unconstrained fine-tuning.