Table 1: Starting from one auxiliary task (Exemplar-MT), we keep increasing the number of auxiliary tasks from one to four by adding one auxiliary task at a time in the order of {Rotation, VAT, EntMin}.

K	1 (+Exemplar-MT)	2 (+Rotation)	3 (+VAT)	4(+EM)
CIFAR-10	17.02	13.68	12.47	**11.80**
SVHN	9.65	5.89	5.00	**4.70**

Table 2: Effect of ARML on harmful tasks.

Exemplar-MT	+ Rotation (harmful)
17.02	**16.95** (w/ ARML) 83.76 (w/o ARML)

We would like to thank all the reviewers for writing the insightful comments, especially during this difficult time.

R1.1 - **L110 "$p^*(\theta)$ is 'covered' by $p(\mathcal{T}_m|\theta)$" is not clearly explained**: Thanks! By 'covered', we mean $p(\mathcal{T}_m|\theta)$ has high density both in the support set S of $p^*(\theta)$, and in some regions outside S. We will clarify this in the final version.

R1.2 - **Our contribution and difference from previous works**: Thanks. We proposed ARML for auxiliary task reweighting so that the least data is required to find the true parameter. Although ARML and some previous algorithms (*e.g.* CosineSim, OL_AUX) both update task weights based on the similarity between main/auxiliary task gradient, the previous algorithms mainly focus on the **training loss**. For example, CosineSim discards the harmful tasks so that the training loss of the main task will not increase in one step. OL_AUX updates the task weights so that the training loss decreases the fastest. However, lowering training loss may cause overfitting, especially when training data is scarce. In contrast, ARML is guaranteed to find a good prior so that the least data is required to find the parameter which generalizes the best (*i.e.* having the lowest **test error**). The superiority of ARML is verified in the experiments.

R1.3 - **Ablation on number of auxiliary tasks**: Thanks. We ablate on the number of auxiliary tasks in SSL, and observe the results in Table 1. The error rates decreases when each new task is added.

R1.4 - **Line 239-242 the text here is confusing**: Thanks. For domain generalization baseline, the model is trained only with source domain (auxiliary) data. In 'Baseline + ARML', for fair comparison, we stick to the same training process, *i.e.*, updating θ using only auxiliary loss (NOT using the main task loss), and the main task data is used to adjust the task weights $\boldsymbol{\alpha}$ which brings the result improvement. We will add more elaboration on this in the final version.

R1.5 - **Non-classification experiments**: Thanks for the suggestion! We will try other tasks, *e.g.*, reinforcement learning.

R2.1 - **More explanation in the Algorithm section**: Thanks! (1) We use mini-batch sampling for main/auxiliary task data. In each mini-batch, we sample the same amount of two kinds of data. (2) Yes, during early epochs the sampling process is still warming up, and after that the estimation is more accurate. We will add more details in the section.

R2.2 - **Discussion on the efficiency of the proposed method**: Thanks. The only overhead of ARML is the update of $\boldsymbol{\alpha}$. This has little extra cost because the main/auxiliary task gradients are already calculated when updating θ, and we only have to calculate the gradient w.r.t. $\boldsymbol{\alpha}$. We observed a $< 10\%$ extra cost for PyTorch, and nearly no extra cost for TensorFlow (because of the different autograd strategies of the two frameworks). The efficiency is similar to previous algorithms (*e.g.* OL_AUX), and much higher than GridSearch. We will update this in the final version.

R3.1 - **More experiments on multi-label classification**: Thanks! We will add more experiments including 1) analyzing if the learned face attribute relationship aligns with human's intuition, 2) varying the number of auxiliary attributes.

R3.2 - **Can ARML prevent harmful tasks?**: Thanks! Ideally, ARML can discard a harmful task by lowering its weight to 0. For verification, we turn the Rotation loss in S4L into its negative, making it a harmful task (which means when training we are actually increasing the loss). Meanwhile we keep the Exemplar-MT loss unchanged. We find that ARML lowers the weight of Rotation to 0, getting the same error rate as when only Exemplar-MT is applied (Table 2).

R3.3 - **Why does VAT + EntMin works better than S4L + ARML? Have you tried MOAM?**: Since ARML already finds the optimal task weights, the performance is mainly limited by the auxiliary tasks themselves. For example, if we further add VAT & EntMin as auxiliary tasks (which makes it MOAM), then ARML surpasses SOTA (R1.3 & Table 1).

R4.1 - **Justification behind switching sampling probability (Theorem 1)**: Thanks. We think there may be a small misunderstanding, because Theorem 1 does not mean that we are minimizing a lower bound of the true objective. Our true objective is to find the optimal task weights $\boldsymbol{\alpha}^\star$ so that $D_{\mathrm{KL}}(p^* \parallel p_{\boldsymbol{\alpha}^\star})$ is the smallest. Theorem 1 states that, if we choose the weights $\hat{\boldsymbol{\alpha}}$ (from the optimization of the surrogate objective), then the corresponding true objective $D_{\mathrm{KL}}(p^* \parallel p_{\hat{\boldsymbol{\alpha}}})$ is also very small, *i.e.*, upper-bounded near the optimal value $D_{\mathrm{KL}}(p^* \parallel p_{\boldsymbol{\alpha}^\star})$.

R4.2 - **The concept of "true prior"**: This is a good point! The "true prior" is an objective prior reflecting the randomness of the environment. For example, for image classification problems with pictures taken in different places, the model parameter should stay similar to extract the same features, while bearing some slight changes (*e.g.* different image statistics in BN). The "true prior" can also be regarded as a Dirac delta if not considering such randomness.

R4.3 - **Details**: Thank you! We will correct the typos in the final version. 4) WRN-28-2 is used for SSL experiments, and ResNet18 is used for others. 5) We take three runs for each result. 6) Thanks. We will add the illustrations.