We thank all reviewers for the thoughtful comments and constructive suggestions to improve our paper. In general, all reviewers find our general message: "*the model learned in each task is itself part of the inductive bias*" convincing. The core idea of incorporating model complexity into task embedding is "*well motivated and interesting*" (R4), "*novel and interesting, and generally applicable to many meta-learning models*" (R5), and "*is interesting and well-illustrated*" (R6). The only reservation shared by all reviewers is that experiments are not sufficient to support this claim.

Here, we address the major concern raised by all three reviewers — **generalization to more baselines**. We conducted additional experiments on two competitive baselines with large backbone feature extractors. To summarize, MATE brings consistent improvements by exploiting model information in task representations, which confirms our original finding. We plan to try more baselines and report in the final version. We also provide details about **meta-testing protocol** (R4), discuss **the gain brought by MATE** (R4, R5) and **the choice of FiLM layer conditioning** (R5).

Model	Backbone	5-way 1-shot	5-way 5-shot
MetaOptNet [20]	ResNet-12	$72.00 \pm 0.70\%$	$84.20 \pm 0.50\%$
MetaOptNet + MATE	ResNet-12	$72.30 \pm 0.70\%$	$85.20 \pm 0.40\%$
ProtoNets [43]	ResNet-12	$71.35 \pm 0.73\%$	$84.07 \pm 0.51\%$
ProtoNets + MATE	ResNet-12	$71.49 \pm 0.70\%$	$84.71 \pm 0.50\%$
R2D2 [6]	ResNet-12	$72.51 \pm 0.72\%$	$84.60 \pm 0.50\%$
R2D2 + MATE	ResNet-12	$72.59 \pm 0.70\%$	$85.04 \pm 0.50\%$

$\triangleright$ **Applying MATE to more baselines (R4, R5, R6).** Per all your suggestions, we conducted experiments on two more baselines. Due to the limited rebuttal time window and the well-known difficulty in finding suitable, reproducible implementation of SOTA meta learning works, we turn to two baselines that have been compared in this paper, namely, Prototypical Networks [43] and R2D2 [6], but use larger convolutional backbones. We limit the experiments on CIFAR-FS, and will include miniImageNet results on the new baselines. The results are shown in the above table. Although ProtoNets and R2D2 are somehow old, we would still like to justify that comparing on these two are meaningful and can help to corroborate the generality of MATE framework. It is known that the original ProtoNets and R2D2 have much lower performance than more recent works, e.g. they are 12.2% and 4.8% lower in 5-way 5-shot accuracy on CIFAR-FS compared to MetaOptNet [20], respectively. However, once we try replace the backbone feature extractor with the same ResNet-12 used in MetaOptNet, ProtoNets and R2D2 both show competative results, and especially R2D2 already performs better than MetaOptNet just by ensuring a fair backbone. Then, MATE can still consistently provide improvements to both (enhanced) baselines: 1) applying MATE to ProtoNets+ResNet12 yields $+0.64\%$ 5-shot accuracy and slightly better 1-shot accuracy ($+0.14\%$); 2) applying MATE to R2D2+ResNet12 yields $+0.44\%$ 5-shot accuracy improvement and similar 1-shot accuracy ($+0.08\%$). These results are hence consistent with our original finding that MATE brings more benefits to 5-shot accuracy than to 1-shot, which is reasonable because we can obtain more accurate information about data distribution on the task with more data and thus task representation of higher quality.

$\triangleright$ **Protocal used for meta-testing (R4).** During the meta-testing stage, we sample 1,000 episodes (Section 3.2) from the meta-testing set following either 5-way 1-shot or 5-way 5-shot settings. The query set in each meta-testing episode contains 15 query images over which we calculate the meta-testing accuracy. We then report the average accuracy and standard devation of the accuracies over the 1,000 meta-testing episodes. Due to large amount of tesing episodes used, the standard deviation of the accuracies is sometimes very close. We confirm that the numbers reported in the tables in this paper are all correct. We would like to thank R4 for pointing out the ambiguity of the testing protocol. We will clarify this and add more details of the experiments in the final version.

$\triangleright$ **Limited performance gain (R4, R5).** Firstly, we thank R4 for the appreciation of the improvement brought by MATE, which we think can be further strengthened by the additional experiments we just conducted. Secondly, we humbly clarify that we calculate the accuracy standard deviation over 1,000 meta-testing tasks instead of the confidence interval. Hence, the accuracy improvement over 0.5% can show consistent improvement over a large sample of tasks. We'd also like to emphasize that incorporating model information into task embedding does help with and improve the performance, which is supported by the comparison of FiLM+KME and FiLM+SVM in Table 3 (2nd and 3rd rows).

$\triangleright$ **Conditioning FiLM layers on ω (R5)?** If we understand correctly, R5 suggests to condition FiLM layers on the optimal parameters learned by SVM (Section 3.1), instead of the model-aware task features proposed in this paper. We think this question can be answered well by humbly reminding R5 of the connection of our proposed method with kernel mean embedding (KME) [28], as we described before Section 3.1. In Eq. (1), if we ignore the model information by taking $f_M(x) \equiv 1$, Eq. (1) reduces to KME. Further, if ϕ corresponds to the canonical feature map of the characteristic kernel, the map defined by Eq. (1) is injective, i.e., the representation $\Phi(T)$ captures all information about the task T [10, Lemma 2]. Therefore, condtioning FiLM layers on the model-aware task feature defined in Eq. (1), which is very likely to contain most of information on the task, could possibly make the FiLM easier to train and, more importantly, more interpretable. We plan to conduct comparision experiments and report related results in final version.