Reviewer #2: Thank you for the thorough review. Indeed, better theoretical guidance enables XNAS to naturally select several hyper parameters, achieving best performance and accelerate differential architecture search.

Experiments: since the compared methods use different sets of training tricks, we adopted the exact ones reported in SHARPDARTS, which is the newest NAS compared. The additional tricks in SHARPDARTS vs DARTS are: 1.AutoAugment 2.power cosine annealing learning rate, instead of cosine annealing.

Since DARTS' code is published, a fair comparison is attainable:

CIFAR-10 error	DARTS repo	XNAS repo	FLOPS(M)	inference time (ms, 1080ti)
DARTS-V2 (36 channels)	2.76%	$\mathbf{2.42}\%$	550	4.2
XNAS-small (36 channels)	$\mathbf{2.16}\%$	1.81%	529	3.6

In Bold are the results of new experiments following the reviewer's request. It can be seen that while additional tricks do help, XNAS cells exhibit significantly better performance even without those. For clarity, we will add 'XNAS-plain' results in the camera-ready version. Plain-mode is now supported by XNAS repo, which will be published upon acceptance to enable further comparisons. FLOPs and inference time comparisons were out of this paper's scope, and were not reported by the majority of compared methods. We will report our results in the revised experiments section.

EG/Wipeout ablation: Please find EG/Wipeout ablation below.

Paper clarity: the revised version now includes clearer explanations and modified figures as remarked by the reviewer.

Specifically, $p_t \doteq p_t^{(j,k)}$ is the prediction of a forecaster, as "superscript indices are ignored for brevity" (line 90).

For citation we simply used the command *cite* with NeurIPS style.

Fig.1 was replaced with an illustration of multiplicative vs additive updates for differential architecture search.

Reviewer #3: Thank you for your review. Apparently our contribution and novelty was not stated clearly.

main difference compared to DARTS method is that it has a wipe-out method - While DARTS picks a generic optimizer for architecture optimization, we derive an optimizer tailor-made for NAS based on EG. We suggest regret-minimization as a novel criterion for NAS to derive XNAS, and prove it achieves optimality under this criterion. We explore the algorithm's appealing properties, like robustness to initialization and the usage of multiple theoretically-derived learning rates, and demonstrate these in different setups. XNAS is lighter and x3.3 faster than DARTS, thanks to simple multiplicative updates and less hyper-parameters involved. It outperforms DARTS in $7/7$ datasets, including -42% error in CIFAR-10. The wipeout is merely a part of this synergy. Please find EG/Wipeout ablation below.

wipeout ... has the risk of eliminating useful "expert" - the wipeout's merit is the elimination of unuseful experts only. Lemma 1 states that clearly: "In XNAS, The optimal expert in hindsight cannot be wiped-out".

In the light of the above, we ask the reviewer to reconsider his review for this work.

Reviewer #4: Thank you for the valuable remarks which help us make the revised version much clearer, and also for acknowledging the originality and the need for EG in the context of NAS.

Regret analysis assumptions: *provided under conditions that are not obvious to hold* - that is correct. In the analysis, we assume gradient-boundness and convexity, as the former is enforced by applying gradient-clipping, the latter does not hold in general for DNNs. However, the convexity of the loss is a very common assumption for regret minimization and online learning in general. It is assumed when deriving similar theoretical guarantees for *all* previous NAS optimizers, e.g. gradient descent (ASAP) and ADAM (DARTS). In addition, we put to test the optimizer in several toy setups and simulations, deterministic and statistical (sections 8.1, 8.2, 8.3), investigate its properties and compare with common optimizers. Finally, extensive experimentation in NAS provides an empirical support for the algorithm derived under that assumption. Based on that, we believe that XNAS analysis has a high significance as well.

EG loss function: *Can the authors provide the form of this loss function?* - The loss function is the parent network's cross-entropy loss over the validation set. This loss is not defined explicitly for a predictor in an intermediate node. However, its back-propagated gradient is: $\nabla_{p_t} \ell_{\text{val}}(p_t)$. Therefore, unlike loss-based PEA methods like HEDGE, we derived a gradient-based NAS optimizer, via auxiliary-losses (Eq. 22). We will make this observation clearer.

Reward bounds, gradient clipping and optimal learning rates - the bounds are over the reward values (Alg.1, line 8), $|R_{t,i}| \leq \mathcal{L}$, as required by our theory and fulfilled by a simple gradient-clipping. We will make that clearer in the paper. The optimal learning rate derived from theorem 1 depends on that bound (line 163) and is used in our experiments (values-line 229). The successful usage of the theoretical learning rate in practice is a key property of XNAS.

EG/Wipeout ablation: The revised version will contain a section for the analysis of EG without wipeout. We include here only the summary and omit the full details and graphs due to the limited space.

Runtime: *EG*: 0.35 GPU-days. Faster than ADAM which is used in DARTS(1nd) and runs for 0.4 GPU-days with a significantly inferior performance (table 2). This is mostly due to fewer calculations and no momentum updates (section 4.2.2). *Wipeout contribution*: -0.05 GPU-days. Wipeout takes effect around the last 30% of the run, and reduces this part's runtime by half. If one desires a further acceleration (e.g. a large dataset or many operations), while risking with the elimination of useful experts - the wipeout factor discussed in section 4.2.1 controls this trade-off.

Performance: EG is responsible for most of the improvement compared to previous methods. Based on a few runs, the mean error for CIFAR-10 is around 1.75% (50 channels). The wipeout contributes to the overall mean performance, and also reduces its standard deviation caused by hard selections of operations ('relaxation bias', line 128).