Mitigating Negative Transfer in Multi-Task Learning with Exponential Moving Average Loss Weighting Strategies

Anish Lakkapragada,[1] **Essam Sleiman,**[2] **Saimourya Surabhi,**[1] **Dennis P. Wall**[1]

[1] Department of Pediatrics (Systems Medicine), Stanford University, Stanford CA 94305
[2] Department of Computer Science, University of California, Davis CA 95616
anishlk@stanford.edu, esleiman@ucdavis.edu, mourya@stanford.edu, dpwall@stanford.edu

Abstract

Multi-Task Learning (MTL) is a growing subject of interest in deep learning, due to its ability to train models more efficiently on multiple tasks compared to using a group of conventional single-task models. However, MTL can be impractical as certain tasks can dominate training and hurt performance in others, thus making some tasks perform better in a single-task model compared to a multi-task one. Such problems are broadly classified as negative transfer, and many prior approaches in the literature have been made to mitigate these issues. One such current approach to alleviate negative transfer is to weight each of the losses so that they are on the same scale. Whereas current loss balancing approaches rely on either optimization or complex numerical analysis, none directly scale the losses based on their observed magnitudes. We propose multiple techniques for loss balancing based on scaling by the exponential moving average and benchmark them against current best-performing methods on three established datasets. On these datasets, they achieve comparable, if not higher, performance compared to current best-performing methods.

Introduction

A plethora of loss balancing methods have emerged to prevent certain tasks in MTL from dominating at the expense of other tasks' performances in training. In contrast to current loss balancing methods that employ complex optimization or numerical analysis, we propose a much simpler method, directly scaling each loss by its observed magnitude. To accomplish this, we simply scale task losses by their exponential moving averages (EMAs). We merge other loss balancing techniques centered around weighting tasks based on their training rates with ours. Our techniques are competitive and superior to several other best-performing techniques on the CelebA, AffWild2, and AffectNet datasets.

Methods

Current loss balancing methods generally either learn the loss coefficients for each task through parameter optimization or derive them from some measure of a training rate. GradNorm (Chen et al. 2018) employs both methods in determining the loss coefficients by optimizing them to keep the gradient magnitudes for all tasks more or less equivalent depending on their individual training rates. Dynamic Weight Average or DWA (Liu, Johns, and Davison 2019) calculates the descending speed of convergence of a task based on the ratio of that task's loss at the current compared to the past training iteration and assigns tasks with a higher ratio (and thus slower convergence) greater loss weightage. However, DWA doesn't account for the actual magnitudes of the individual task losses.

No method we have seen so far, to the best of our knowledge, considers simply scaling each of the task losses by their magnitudes. We accomplish this by dividing each of the task losses by their observed exponential moving average (EMA) so they are all on the same scale of one.

$$\tilde{L}_k(t) = \beta L_k(t) + (1-\beta)\tilde{L}_k(t-1)$$
$$\mathcal{L}_{MTL}(t) = \sum_{k=1}^{K} \lambda_k(t) L_k(t), \lambda_k(t) = \frac{1}{\tilde{L}_k(t)} \quad (1)$$

We show the formulation for our proposed method in Equation 1, where t is the training iteration and β is the weight term in calculating the task loss EMA $\tilde{L}_k$ – the reciprocal of which is the loss coefficient λ_k for task k. Chen et al. notes that the Uncertainty Weighting technique (Kendall, Gal, and Cipolla 2018), a successful loss balancing technique which learns λ_k to optimize the overall loss through gradient descent, often learns $\lambda_k \approx 1/L_k$. This validates our idea to directly scale task losses by their moving average. However, they also note Uncertainty Weighting (UW) can lead to volatile spikes in the λ_k, which leads to performance deterioration – in our case, β is a hyperparameter to help prevent such issues by tuning how fast to adapt the task loss EMA and thus the loss coefficients themselves.

We also merge our idea with DWA in our Rated Exponential Moving Average (REMA) method, where each of the losses are first scaled by their EMA and then shifted accordingly by their training rates r_k for task k to increase loss weightage on tasks with slower convergence. This is shown in Equation 2.

$$\mathcal{L}_{MTL}(t) = \sum_{k=1}^{K} r_k(t) \frac{L_k(t)}{\tilde{L}_k(t)}, r_k(t) = \frac{L_k(t-1)}{L_k(t-2)} \quad (2)$$

Method	Baseline	GradNorm	UW	DWA	EMA	DWEMA	REMA
Performance	0.772	0.771	0.768	0.772	0.782	0.788	**0.788**

Table 1: Overall performance on the 40 CelebA tasks for our experiments.

We compare this to directly scaling the DWA coefficients by $\tilde{L}_k$, instead of just r_k. We refer to this method as Dynamic Weighted Exponential Moving Average (DWEMA). Unlike DWA, both DWEMA and REMA actively prevent task domination by first scaling losses by their magnitudes before applying training rate adjustments.

Experiments

We test our models on three datasets, CelebA (Liu et al. 2015), AffWild2 (Kollias 2022), and AffectNet (Mollahosseini, Hasani, and Mahoor 2017). All contain cropped and aligned facial images; we define the tasks for each of the datasets as all the attributes labeled for each image (except for the binary labels for each of the twelve action units annotated in AffWild2, which are treated as one task). We detail our metrics of overall performance for each MTL model on all our datasets in our supplementary material.

Our AffWild2 and AffectNet MTL models were trained using standard hard parameter sharing on a ResNet50 backbone. On the CelebA dataset, due to computational constraints, we use a ResNet18 backbone feeding to a dense layer of 40 neurons to predict on the 40 tasks. In order to frame our CelebA experiments as MTL, we backpropagate on each task individually, based on its individual loss coefficient.

We train all our models with the same initialization of parameters. We set β to 0.2 on CelebA runs and 0.1 for AffWild2 and AffectNet runs. The performances for CelebA are detailed in Table 1 and the individual task and overall performances for AffWild2 and AffectNet are described in Table 2.

We observe the merit of scaling losses by their EMA through the higher performance of the EMA and variants, REMA and DWEMA, approaches compared to other past methods. These two variants based on training rates have a bigger impact on performance in the CelebA dataset compared to the others. We conjecture that when training rates r_k are more varied, weighting certain tasks by their training rate would be more beneficial for performance through a comparison on the training rates for each dataset.

Conclusion

We propose three EMA-based techniques for mitigating negative transfer in MTL models. We achieve superior performance on these techniques on the CelebA, AffectNet, and AffWild2 dataset compared to several best-performing methods. Overall, we reason that our method's higher performance is due to the explicit scaling of the losses being more defined than in past approaches. We foresee our merge of multi-task learning on computer vision tasks with expression-cented AI efforts to be more relevant in the future as MTL provides more efficient computing on edge devices

Method	Dataset	AU	Emotion	VA	Overall
Single Task	**AffWild2**	0.574	0.057	0.068	0.699
	AffectNet	–	0.426	0.419	0.843
Baseline	**AffWild2**	0.579	**0.082**	0.083	0.744
	AffectNet	–	0.425	0.428	0.853
GradNorm	**AffWild2**	0.579	0.080	0.072	0.730
	AffectNet	–	0.401	0.408	0.809
UW	**AffWild2**	0.578	0.081	0.087	0.746
	AffectNet	–	0.393	0.410	0.803
DWA	**AffWild2**	0.580	0.079	0.102	0.760
	AffectNet	–	0.407	0.406	0.813
REMA	**AffWild2**	0.586	0.080	0.100	0.764
	AffectNet	–	0.421	0.449	0.869
DWEMA	**AffWild2**	0.588	0.081	0.106	0.775
	AffectNet	–	0.425	0.443	0.868
EMA	**Affwild2**	**0.590**	0.080	**0.133**	**0.800**
	AffectNet	–	**0.427**	**0.471**	**0.898**

Table 2: Analysis of each of the performances on the Action Units (AUs, only for AffWild2), Emotion, and Valence & Arousal (VA) tasks and overall performance for our AffWild2 and AffectNet experiments.

where such applications are likely to be deployed. We explain in greater depth our implementation, hyperparameter choices, and reasoning in our supplementary material.

References

Chen, Z.; Badrinarayanan, V.; Lee, C.-Y.; and Rabinovich, A. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, 794–803. PMLR.

Kendall, A.; Gal, Y.; and Cipolla, R. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Kollias, D. 2022. Abaw: Learning from synthetic data & multi-task learning challenges. *arXiv preprint arXiv:2207.01138*.

Liu, S.; Johns, E.; and Davison, A. J. 2019. End-to-End Multi-task Learning with Attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1871–1880.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Mollahosseini, A.; Hasani, B.; and Mahoor, M. H. 2017. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1): 18–31.