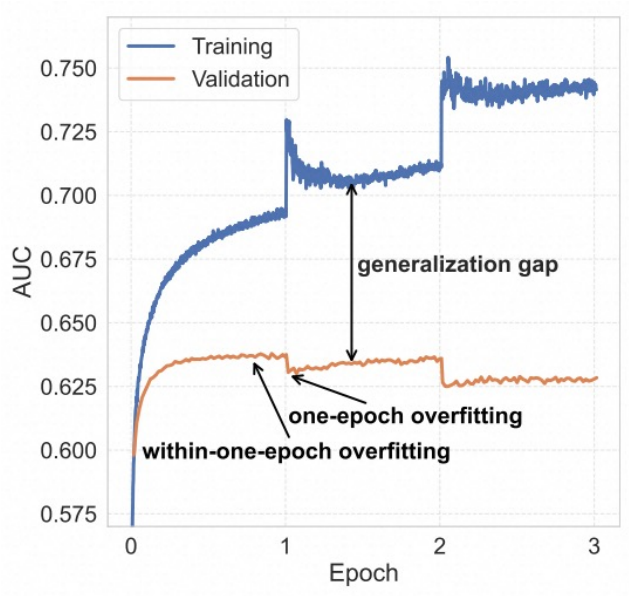
Adaptive Regularization for Large-Scale Sparse Feature Embedding Models

Mang Li, Wei Lyu

Institute of Intelligent Technology

Alibaba International Digital Commerce Group

One Epoch Phenomenon for Large Scale Embedding Model



- Over the past decade, deep learning has become prevalent in ASR domain, with most estimation models relying on large-scale sparse categorical features.
- These models often exhibit one-epoch overfitting, where performance drops sharply after the first training epoch.

Preliminary

- Neural Network Definition
  - Embedding of each feature $\boldsymbol{e}_i(t) = \boldsymbol{E}_i^\top \boldsymbol{x}_i(t)$
  - multi-layer perceptron $f(\cdot) = \boldsymbol{W}_L \sigma_{L-1}\left(\boldsymbol{W}_{L-1}\sigma_{L-2}\left(\ldots \sigma_1\left(\boldsymbol{W}_1 \cdot\right)\right)\right)$
  - prediction for each sample $y(t) = f\left(\left[\boldsymbol{E}_1^\top \boldsymbol{x}_1(t); \boldsymbol{E}_2^\top \boldsymbol{x}_2(t); \ldots; \boldsymbol{E}_S^\top \boldsymbol{x}_S(t)\right]\right)$
- Rademacher complexity Bound

$$\widehat{\mathcal{R}}_T(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\epsilon}}\left[\sup_{h\in\mathcal{H}} \frac{1}{T}\sum_{t=1}^{T} \epsilon_t\, h(\boldsymbol{z}_t)\right]$$

$$\widehat{\mathcal{R}}_T(\mathcal{H}_L) \le \frac{1}{T}\left(\prod_{l=1}^{L} M_F(l)\right)\left(\sqrt{\sum_{i=1}^{S} M_{E_i}^2}\right)\left(\sqrt{2\log(2)L}+1\right)\sqrt{\sum_{t=1}^{T}\sum_{i=1}^{S}\|\boldsymbol{x}_i(t)\|^2}$$

Optimization with Rademacher complexity constraint

- Problem formulation

  We cast the trade-off between training loss and the generalization error bound as a constrained optimization problem.

$$\min_{\tau_{ij}>0} \sum_{i=1}^{S}\sum_{j=1}^{N_i} m_{ij}\varphi(\tau_{ij}) \quad \text{s.t.} \quad \sum_{i=1}^{S}\sum_{j=1}^{N_i} \tau_{ij} \leq C$$

- Necessary Condition for the Optimal Regularization

Proposition 1 *A necessary condition for the optimal regularization multiplier λ_{ij}^* associated with the $\|\mathbf{e}_{ij}\|^2 \leq \tau_{ij}^*$ is given by $\lambda_{ij}^* = \mu_0/m_{ij}$, where μ_0 is the Lagrange multiplier corresponding to $\sum_{i=1}^{S}\sum_{j=1}^{N_i} \tau_{ij}^* \leq C$.*

Adaptive Regularization Method

- Last valid update step
  - estimate the update interval of each feature $I_{ij}^{k} = k - s_{ij}^{k-1} - 1$
  - adaptive regularization $\lambda_{ij}^{k} = \min\left(1, \alpha I_{ij}^{k}\right), i \in [S], j \in [N_i]$
- Optimization algorithm

Algorithm 1 Adam with Adaptive Regularization (AdamAR)

given $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$, α, learning rate η
initialize time step $k \leftarrow 0$, parameter $\theta_p^{k=0}$, first moment $m_p^{k=0} \leftarrow 0$, second moment $v_p^{k=0} \leftarrow 0$, last update step state $s_p^{k=0} \leftarrow 0$
repeat
  $k \leftarrow k + 1$
  $g_p^k \leftarrow \nabla_{\theta_p} f\left(\theta_p^{k-1}\right)$ (Get gradients w.r.t. stochastic objective at timestep t)
  $m_p^k \leftarrow \beta_1 m_p^{k-1} + (1 - \beta_1)\, g_p^k$ (Update biased first moment estimate)
  $v_p^k \leftarrow \beta_2 v_p^{k-1} + (1 - \beta_2)\left(g_p^k\right)^2$ (Update biased second raw moment estimate)
  $\hat{m}_p^k \leftarrow m_p^k / \left(1 - \beta_1^k\right)$ (Compute bias-corrected first moment estimate)
  $\hat{v}_p^k \leftarrow v_p^k / \left(1 - \beta_2^k\right)$ (Compute bias-corrected second raw moment estimate)
  $\lambda_p^k \leftarrow \min\left(1, \left(k - s_p^{k-1} - 1\right)\alpha\right)$ (Compute adaptive regularization through equation 11)
  $s_p^k \leftarrow k$ if $||g_p^k|| > 0$ else s_p^{k-1} (Update the last update step when the gradient norm is greater than zero)
  $\theta_p^k \leftarrow \theta_p^{k-1} - \lambda_p^k \theta_p^{k-1} - \eta \cdot \hat{m}_p^k / \left(\sqrt{\hat{v}_p^k + \varepsilon}\right)$ (Update parameters)
until stopping criterion is met
return optimized parameters θ_p^t

Minimum Convergence

- Assumptions

Assumptions 1. The function f is differentiable and its gradient is Lipschitz continuous, i.e., there exists $L_0 > 0$ such that $\|\nabla f(\boldsymbol{\theta}^{k+1}) - \nabla f(\boldsymbol{\theta}^k)\| \leq L_0 \|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|, \forall k \geq 1$, and f is lower bounded at the optimal solution, i.e., $f^* > -\infty$.

Assumptions 2. $\boldsymbol{g}^k$ is an unbiased estimator of the full gradient, i.e., $\mathbb{E}\left[\boldsymbol{g}^k\right] = \nabla f(\boldsymbol{\theta}^k)$ with $M > 0$, and the algorithm accesses a bounded stochastic gradient, i.e., $\|\boldsymbol{g}^k\| \leq M$ a.s.

- Minimum Convergence Instruction

Proposition 3 *The adaptive regularization method preserves the minimum convergence bound of the Adam optimizer with stochastic conditions, which can be expressed as*

$$\min_{1 \leq k \leq K} \mathbb{E}\left[\left\|\nabla f\left(\boldsymbol{\theta}^k\right)\right\|^2\right] \leq \frac{C_1 + C_2 \sum_{k=1}^{K} \eta_k + C_3 \sum_{k=1}^{K} \eta_k^2}{\sum_{k=1}^{K} \eta_k} \tag{15}$$

where C_1, C_2 and C_3 are constants, η_k denotes the step size at iteration k out of K total iterations.

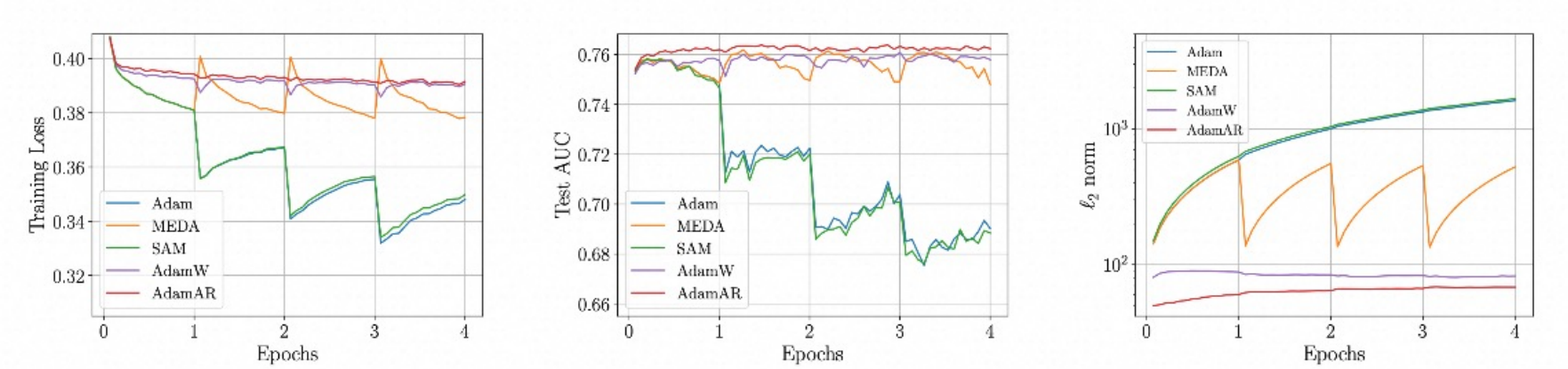
Experiments and Results

we compare test AUC across different datasets and MLP backbones to demonstrate the generalization capability of our method. Each experiment is repeated three times with different seed

Dataset	Method	DNN				WDL				xDeepFM				WuKong			
		E1	E2	E3	E4	E1	E2	E3	E4	E1	E2	E3	E4	E1	E2	E3	E4
iPinYou	Adam	0.7515	0.7304	0.7061	0.7014	0.7619	0.7320	0.7028	0.6987	0.7590	0.7391	0.6969	0.6844	0.7611	0.7442	0.7082	0.6915
	MEDA	0.7515	0.7644	0.7684	0.7717	0.7619	0.7589	0.7565	0.7551	0.7590	0.7584	0.7575	0.7597	0.7611	0.7663	0.7662	0.7706
	SAM	0.7510	0.7593	0.7445	0.7256	0.7610	0.7581	0.7404	0.7248	0.7565	0.7517	0.7413	0.7274	0.7033	0.7485	0.7465	0.7306
	AdamW	0.7475	0.7592	0.7623	0.7568	0.7551	0.7656	0.7646	0.7634	0.7607	0.7660	0.7651	0.7574	0.7579	0.7634	0.7605	0.7511
	AdamAR	**0.7566**	**0.7692**	**0.7688**	**0.7724**	**0.7655**	**0.7729**	**0.7670**	**0.7668**	**0.7725**	**0.7733**	**0.7711**	**0.7678**	**0.7653**	**0.7736**	**0.7748**	**0.7736**
Amazon	Adam	0.8482	0.8548	0.8335	0.8180	0.8474	0.8510	0.8261	0.8156	0.8460	0.8535	0.8287	0.8163	0.8580	0.8600	0.8348	0.8232
	MEDA	0.8482	0.8544	0.8556	0.8573	0.8474	0.8506	0.8566	0.8566	0.8460	0.8519	0.8551	0.8562	0.8580	0.8611	0.8621	0.8626
	SAM	0.8507	0.8587	0.8417	0.8249	**0.8516**	0.8567	0.8396	0.8213	**0.8505**	0.8587	0.8390	0.8232	**0.8588**	0.8639	0.8404	0.8225
	AdamW	0.8476	0.8571	0.8426	0.8276	0.8461	0.8533	0.8380	0.8223	0.8446	0.8557	0.8381	0.8240	0.8564	0.8632	0.8478	0.8349
	AdamAR	**0.8507**	**0.8683**	**0.8708**	**0.8686**	0.8496	**0.8654**	**0.8689**	**0.8659**	0.8483	**0.8676**	**0.8687**	**0.8675**	0.8582	**0.8696**	**0.8693**	**0.8664**
Avazu	Adam	0.7461	0.7205	0.7014	0.6883	0.7483	0.7221	0.6982	0.6886	0.7488	0.7217	0.7019	0.6869	0.7514	0.7360	0.7141	0.7079
	MEDA	0.7461	0.7498	0.7489	0.7485	0.7483	0.7488	0.7480	0.7494	0.7488	0.7489	0.7505	0.7506	0.7514	0.7548	0.7553	0.7571
	SAM	0.7451	0.7194	0.7013	0.6899	0.7477	0.7205	0.6993	0.6902	0.7484	0.7190	0.7010	0.6902	0.7513	0.7333	0.7155	0.7156
	AdamW	0.7572	0.7582	0.7582	0.7583	0.7585	0.7581	0.7563	0.7570	0.7583	0.7582	0.7589	0.7593	0.7542	0.7547	0.7558	0.7564
	AdamAR	**0.7617**	**0.7631**	**0.7629**	**0.7629**	**0.7629**	**0.7629**	**0.7627**	**0.7626**	**0.7628**	**0.7633**	**0.7638**	**0.7636**	**0.7624**	**0.7612**	**0.7623**	**0.7624**
LZD	Adam	0.7118	0.6613	0.6252	0.6065	0.7155	0.6726	0.6308	0.6079	0.7164	0.6787	0.6321	0.6050	0.7101	0.6645	0.6102	0.6044
	MEDA	0.7118	0.7105	0.7081	0.7176	0.7155	0.7162	0.7177	0.7162	0.7164	0.7170	0.7152	0.7139	0.7101	0.7170	0.7129	0.7128
	SAM	0.7130	0.6696	0.6341	0.6155	0.7161	0.6795	0.6360	0.6111	0.7166	0.6750	0.6344	0.6134	0.7146	0.6713	0.6443	0.6212
	AdamW	0.7132	0.7135	0.7140	0.7139	0.7143	0.7138	0.7142	0.7131	0.7142	0.7151	0.7139	0.7139	0.7115	0.7135	0.7152	0.7142
	AdamAR	**0.7229**	**0.7235**	**0.7241**	**0.7234**	**0.7233**	**0.7240**	**0.7246**	**0.7240**	**0.7244**	**0.7256**	**0.7242**	**0.7238**	**0.7227**	**0.7215**	**0.7208**	**0.7202**

Dataset	Method	DNN				WDL				xDeepFM				WuKong			
		E1	E2	E3	E4	E1	E2	E3	E4	E1	E2	E3	E4	E1	E2	E3	E4
iPinYou	Adagrad	0.7593	0.6507	0.6231	0.6095	0.7646	0.6653	0.6321	0.6241	0.7674	0.6843	0.6505	0.6527	0.7661	0.6900	0.6497	0.6320
	MEDA	0.7593	0.7686	0.7710	0.7729	0.7646	0.7681	0.7685	0.7715	0.7674	0.7722	0.7728	0.7740	0.7661	0.7705	0.7729	0.7695
	SAM	0.7576	0.7661	0.7487	0.7303	0.7624	0.7676	0.7518	0.7369	0.7637	0.7661	0.7499	0.7363	0.7409	0.7678	0.7525	0.7427
	AdagradW	0.7558	0.7651	0.7667	0.7595	0.7593	0.7675	0.7635	0.7593	0.7665	0.7673	0.7666	0.7652	0.7578	0.7661	0.7649	0.7600
	AdagradAR	**0.7681**	**0.7754**	**0.7760**	**0.7744**	**0.7731**	**0.7772**	**0.7748**	**0.7720**	**0.7760**	**0.7768**	**0.7762**	**0.7745**	**0.7718**	**0.7776**	**0.7782**	**0.7774**
Amazon	Adagrad	0.8438	0.8402	0.8141	0.8042	0.8406	0.8345	0.8085	0.7982	0.8405	0.8374	0.8103	0.7996	0.8491	0.8472	0.8156	0.8046
	MEDA	0.8438	0.8481	0.8495	0.8505	0.8406	0.8470	0.8497	0.8510	0.8405	0.8463	0.8476	0.8500	0.8491	0.8569	0.8587	0.8609
	SAM	**0.8500**	0.8527	0.8361	0.8193	**0.8490**	0.7995	0.8325	0.8155	**0.8483**	0.8521	0.8325	0.8170	**0.8556**	0.8567	0.8356	0.8174
	AdagradW	0.8428	0.8578	0.8563	0.8535	0.8424	0.8553	0.8531	0.8498	0.8410	0.8565	0.8522	0.8508	0.8513	0.8630	0.8617	0.8606
	AdagradAR	0.8479	**0.8659**	**0.8711**	**0.8712**	0.8453	**0.8631**	**0.8687**	**0.8707**	0.8444	**0.8641**	**0.8681**	**0.8703**	0.8538	**0.8690**	**0.8708**	**0.8700**
Avazu	Adagrad	0.7541	0.7323	0.7164	0.7074	0.7543	0.7305	0.7158	0.7073	0.7550	0.7319	0.7160	0.7069	0.7548	0.7311	0.7160	0.7041
	MEDA	0.7541	0.7549	0.7544	0.7545	0.7543	0.7547	0.7540	0.7551	0.7550	0.7538	0.7550	0.7553	0.7548	0.7551	0.7556	0.7564
	SAM	0.7553	0.7333	0.7199	0.7094	0.7557	0.7328	0.7178	0.7079	0.7564	0.7332	0.7198	0.7094	0.7558	0.7353	0.7211	0.7110
	AdagradW	0.7578	0.7580	0.7575	0.7566	0.7584	0.7581	0.7567	0.7585	0.7583	0.7581	0.7580	0.7585	0.7524	0.7555	0.7555	0.7563
	AdagradAR	**0.7628**	**0.7629**	**0.7630**	**0.7624**	**0.7631**	**0.7634**	**0.7623**	**0.7626**	**0.7636**	**0.7633**	**0.7635**	**0.7633**	**0.7628**	**0.7626**	**0.7627**	**0.7620**
LZD	Adagrad	0.7226	0.6658	0.6433	0.6376	0.7244	0.6695	0.6389	0.6386	0.7222	0.6675	0.6400	0.6382	0.7247	0.6726	0.6412	0.6339
	MEDA	0.7226	0.7183	0.7150	0.7236	0.7244	0.7241	0.7239	0.7219	0.7222	0.7237	0.7253	0.7256	0.7247	0.7237	0.7234	0.7239
	SAM	0.7213	0.6761	0.6446	0.6222	0.7229	0.6839	0.6481	0.6266	0.7229	0.6854	0.6480	0.6268	0.7248	0.6802	0.6479	0.6299
	AdagradW	0.7145	0.7132	0.7135	0.7154	0.7138	0.7149	0.7137	0.7145	0.7155	0.7135	0.7117	0.7150	0.7127	0.7111	0.7147	0.7136
	AdagradAR	**0.7253**	**0.7247**	**0.7234**	**0.7241**	**0.7269**	**0.7258**	**0.7251**	**0.7257**	**0.7273**	**0.7268**	**0.7264**	**0.7263**	**0.7269**	**0.7260**	**0.7252**	**0.7239**

Learning Curve and Generalization Results

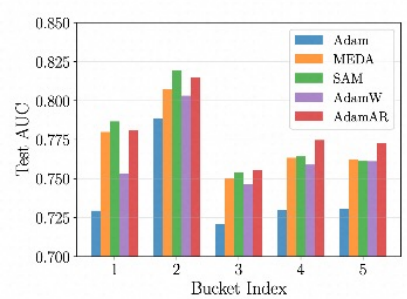


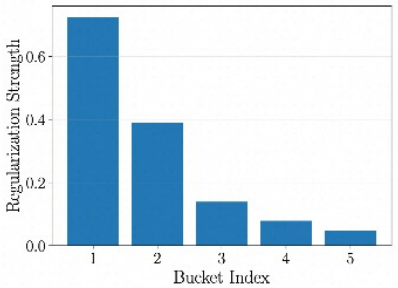
- For the native optimizer, training loss decreases rapidly after the first epoch. However, test AUC drops sharply with additional epochs, indicating clear overfitting.

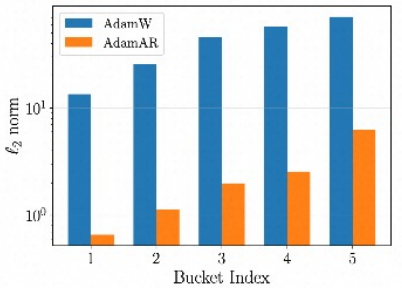
- MEDA and weight decay alleviate this problem, yielding more stable test AUC throughout training. Our method achieves the best overall performance.

Ablation Study

the bucket norms can be controlled via adaptive regularization strength while preserving the AUC gains across all buckets







Experiment Setting	E1	E2	E3	E4
AdamW	0.7486	0.7595	0.7628	0.7500
AdamAR-Bucket 1 & AdamW-Bucket 2-5	0.7457	0.7607	0.7646	0.7520
AdamAR-Bucket 1-2 & AdamW-Bucket 3-5	0.7496	0.7622	0.7655	0.7556
AdamAR-Bucket 1-3 & AdamW-Bucket 4-5	0.7510	0.7646	0.7657	0.7614
AdamAR-Bucket 1-4 & AdamW-Bucket 5	0.7470	0.7656	0.7660	0.7635
AdamAR	0.7549	0.7728	0.7730	0.7725

compared with using a constant weight decay, decreasing the weight decay for high-frequency features and increasing it for low-frequency features can further improve performance while alleviating the one-epoch issue

Conclusion

- We propose an adaptive regularization method to address the one-epoch problem in estimation models for the ASR domain.
- We provide a theoretical explanation for the one-epoch phenomenon and illustrate how the proposed method takes effect through analytical derivations and experiments.
- Experimental results demonstrate that our approach effectively mitigates the one-epoch issue and improves estimation performance.

code: https://github.com/alibaba-aidc/adaptive-regularization.git