AdaEnsemble: Learning Adaptively Sparse Structured Ensemble Network for Click-Through Rate Prediction

YaChen Yan, Liubo Li

August 2, 2023

Summary

Introduction

Background

- Learning feature interactions are important to the model performance of online ads ranking system.
- Extensive efforts have been introduced to learn diffrent type of feature interactions: DeepCrossing, DeepFM, PNN, xDeepFM, AutoInt, FiBiNET, xDeepInt, DCN V2.

Background Cont.

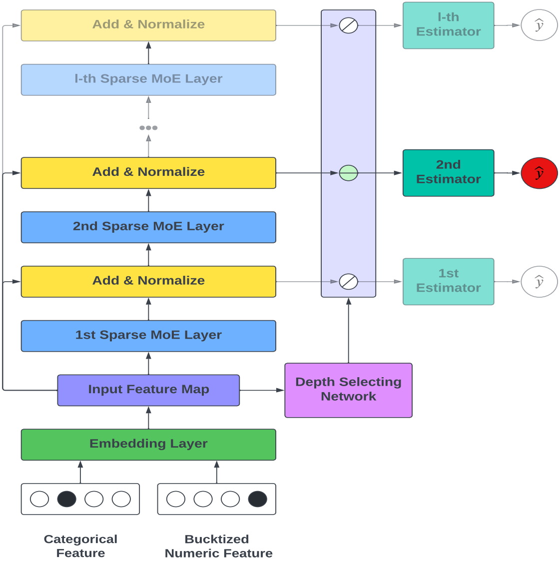
- The practical performance of those designs can vary by dataset.
- Different feature interaction learning methods may have different advantages and the interactions captured by them have non-overlapping information.

Motivation

- Ensemble
    - Ensemble of various interaction modules to generate heterogeneous feature interactions.
    - Each feature interaction learning approach can complement the non-overlapping knowledge.
- Conditional Computation (instance-aware)
    - Dynamically select feature interaction types.
    - Dynamically select optimal feature interaction depth.

Model Architecture

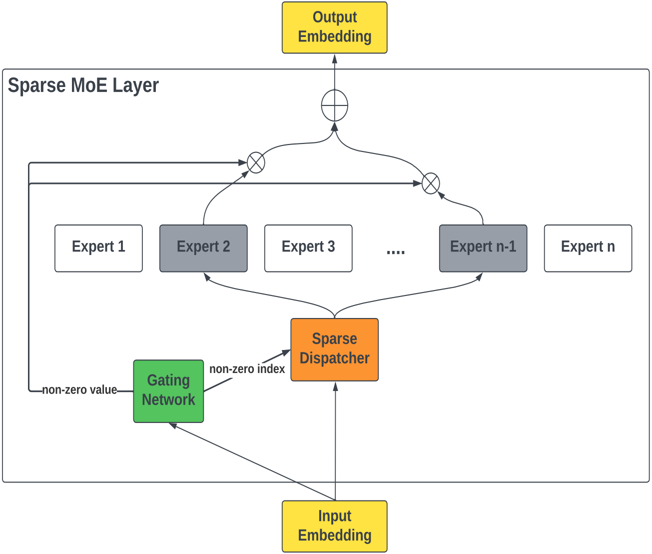
The Architecture of AdaEnsemble



AdaEnsemble Components

- Sparse Mixture-of-Experts Layer:
  - Feature Interaction Experts
  - Selecting the feature interaction experts
  - Dynamically (instance-aware) activate and ensemble them
- Depth Selecting Controller
  - Selecting the feature interaction depth
  - Recursively forward-propagate and compute deeper feature interactions
  - Compute final prediction when reaching selected depth (instance-aware)

The Architecture of Sparse Mixture-of-Experts Layer



SparseMoE Components

- Feature Interaction Experts
- Noisy Gating Network:
  - A neural network selecting the Top-K experts per instance.
  - Annealing Top-K Gating.
  - Load Distribution Regularization.
- Sparse Dispatcher
  - Dispatch input and sparsely activate corresponding experts.
  - Combine each expert's output.

Feature Interaction Experts

- Dense Layer
  - $X_l = \sigma(W_l \cdot X_{l-1})$
- Convolution Layer
  - $X_l = \text{Dense}(\text{Pooling}(\text{Conv1D}(\text{Reshape}(X_{l-1}))))$
- Multi-Head Self-Attention Layer
  - $X_l = \text{Dense}(\text{MultiHeadSelfAttention}(\text{Reshape}(X_{l-1}))$
- Polynomial Interaction Layer
  - $X_l = X_{l-1} \circ (W_l \cdot X_0)$
- Cross Layer
  - $X_l = X_0 \circ (W_l \cdot X_{l-1}) + b_l$

Noisy Gating Network

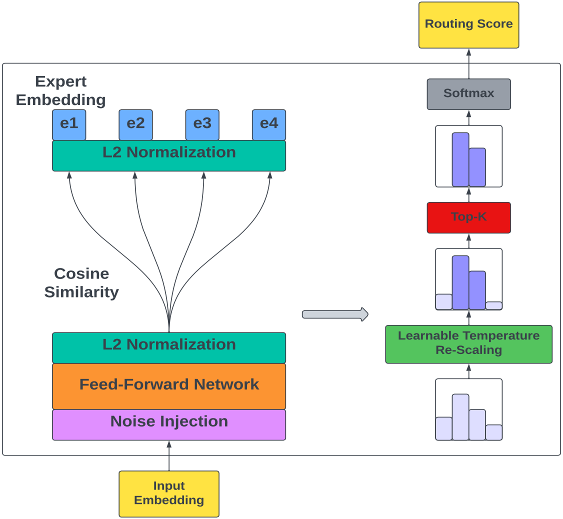


Figure: The Noisy Gating Network within Sparse Mixture-of-Experts Layer

Noisy Gating Network Cont.

- Annealing Top-K Gating:
  - Dense $\rightarrow$ Sparse.
  - All experts $\rightarrow$ Fewer experts.
  - Fully trained experts $\rightarrow$ Expert Routing.
- Load Distribution Regularization
  - Homogeneous Experts: $L_{\text{balance}} = \lambda \cdot N \cdot \sum_{j=1}^{N} f_j \cdot P_j$.
    - f_j is the fraction of examples dispatched to expert j
    - P_j is the average of the router probability allocated for expert j
  - Heterogeneous Experts: $L_{\text{distribution}} = \lambda \cdot \sum_{j=1}^{N} \frac{f_j \cdot P_j}{w_j}$.
    - $\sum_{j=1}^{N} w_j = 1$

Depth Selecting Controller

- Estimator Layer:
  - Different interaction depth has a corresponding estimator layer.
  - A dense layer computes the final prediction.
- Depth Selecting Network:
  - A neural network selects the feature interaction depth per instance.
  - Recursive Propagation
    - Compute deeper feature interactions (Enter).
    - Compute final predictions (Exit)

Recursive Propagation

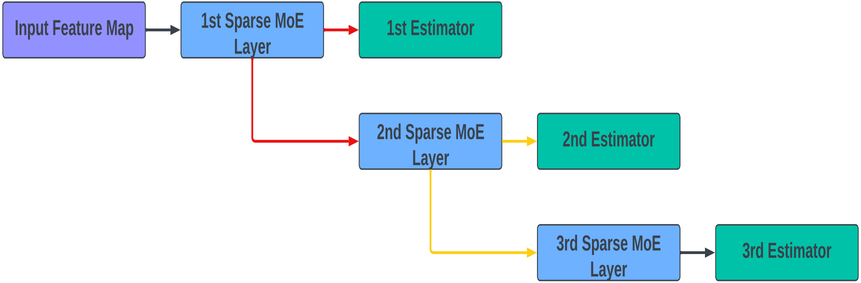


Figure: Visualization of Recursive Propagation

Loss Function

$$Loss = L_{\text{LogLoss}} + \lambda_1 L_{\text{distribution}}^{\text{expert}} + \lambda_2 L_{\text{distribution}}^{\text{depth}} \tag{1}$$

where λ_1 and λ_2 are the coefficients for weighting the load distribution regularization of experts and depth.

Bi-Level Optimization

- Bi-Level Optimization Algorithm:
  - Iteratively optimize the parameters W and α.
  - W: the expert layers and estimator layers.
  - α: the expert gating network and depth selecting network

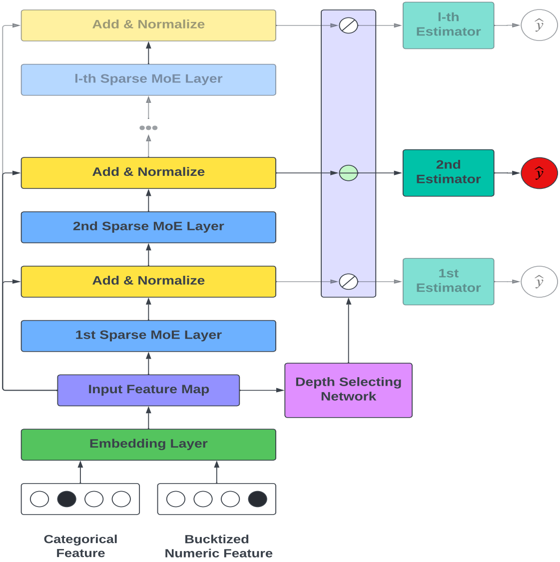
Algorithm 2 Bi-Level Optimization for AdaEnsemble

Input: training examples with corresponding labels, step size t

Output: well-learned parameters $\mathbf{W}^*$ and α^*

while not converged **do**
  Sample a mini-batch of validation data
  Updating α by descending $\nabla_\alpha \mathcal{L}_{val}(\mathbf{W} - \xi \nabla_{\mathbf{W}} \mathcal{L}_{train}(\mathbf{W}, \alpha), \alpha)$
  ($\xi = 0$ for first-order approximation)
  for $i \leftarrow 1, t$ **do**
    Sample a mini-batch of training data
    Update $\mathbf{W}$ by descending $\nabla_{\mathbf{W}} \mathcal{L}_{train}(\mathbf{W}, \alpha)$
  end for
end while

Recap



Add & Normalize
l-th Sparse MoE Layer
Add & Normalize
2nd Sparse MoE Layer
Add & Normalize
1st Sparse MoE Layer
Input Feature Map
Embedding Layer
Categorical Feature
Bucktized Numeric Feature
Depth Selecting Network
l-th Estimator
2nd Estimator
1st Estimator
$\hat{y}$
$\hat{y}$
$\hat{y}$

Experiment

Model Performance Comparison

	Criteo		Avazu		iPinYou	
Model	AUC	LogLoss	AUC	LogLoss	AUC	LogLoss
LR	0.7924	0.4577	0.7533	0.3952	0.7692	0.005605
FM	0.8030	0.4487	0.7652	0.3889	0.7737	0.005576
DNN	0.8051	0.4461	0.7627	0.3895	0.7732	0.005749
Wide&Deep	0.8062	0.4451	0.7637	0.3889	0.7763	0.005589
DeepFM	0.8069	0.4445	0.7665	0.3879	0.7749	0.005609
DeepCrossing	0.8068	0.4456	0.7628	0.3891	0.7706	0.005657
DCN	0.8056	0.4457	0.7661	0.3880	0.7758	0.005682
PNN	0.8083	0.4433	0.7663	0.3882	0.7783	0.005584
xDeepFM	0.8077	0.4439	0.7668	0.3878	0.7772	0.005664
AutoInt	0.8053	0.4462	0.7650	0.3883	0.7732	0.005758
FiBiNET	0.8082	0.4439	0.7652	0.3886	0.7756	0.005679
xDeepInt	0.8111	0.4408	0.7672	0.3876	0.7790	0.005567
DCN V2	0.8086	0.4433	0.7662	0.3882	0.7765	0.005593
AdaEnsemble	**0.8132**	**0.4394**	**0.7687**	**0.3865**	**0.7807**	**0.005550**

Feature Interaction Expert Selection Analysis

Table: Performance Comparison of SparseMoE and DenseMoE on Criteo Dataset.

	AUC	LogLoss	FLOPs
SparseMoE(k=1)	0.8096	0.4423	2.26M
SparseMoE(k=2)	0.8121	0.4400	4.14M
SparseMoE(k=3)	0.8132	0.4394	6.02M
SparseMoE(k=4)	0.8133	0.4393	7.09M
DenseMoE	0.8133	0.4392	9.78M
Ensemble	0.8120	0.4401	12.15M
Dense Expert Only	0.8050	0.4463	3.71M
Cross Expert Only	0.8086	0.4433	3.36M
Polynomial Expert Only	0.8111	0.4408	3.32M
CNN Expert Only	0.8022	0.4501	1.11M
MHSA Expert Only	0.8051	0.4465	2.17M

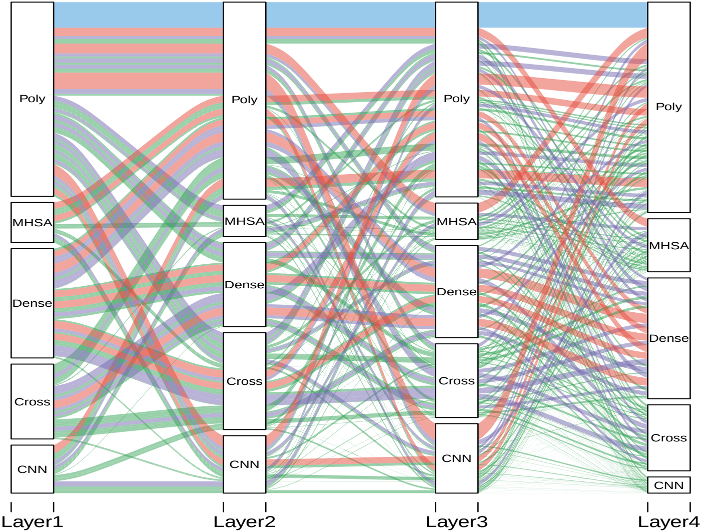


Figure: The Alluvial diagram for illustrating the dependency of each SparseMoE

Conclusions

Conclusions

- We propose a AdaEnsemble leveraging a Sparse-Gated Mixture-of-Experts (SparseMoE) layer and a Depth Selecting Controller, which increased model capacity without raising online inference cost.
- With the conditional computation mechanism applied, the model selects feature interaction experts and optimal depth for each example simultaneously.

Thank You!