
GrowNet Supplementary Material

1 Additional Related Work

A few works [4, 5] have also been proposed to directly combine Gradient Boosting with Convolutional Neural Nets (CNN). The authors of [5] propose to train gradient boosting machine with CNN as a base learner by introducing a custom multi-class softmax loss function for a specific scene classification task in the remote sensing domain. The work in [4], on the other hand, focuses on training each CNN sequentially on the mistakes of the previous networks, similar to Adaboost to perform on solely image classification task. Our method is different from [5, 4] as it is a unified framework to perform various machine learning tasks, such as classification, regression and even learning to rank. Moreover, unlike those two methods, we leveraged a corrective step to update the previously added predictor parameters and achieved a significant performance boost.

2 Additional Dataset Statistics

We evaluate our model on 5 datasets from 3 different tasks. A brief description of these datasets are presented in Table 1.

We used Higgs Bozon dataset¹ for classification. Higgs data is created using Monte Carlo simulations on high energy physics events. It is a binary event classification data with 28 attributes.

For the regression task, 2 datasets from the UCI machine learning repository are selected. The first one is Computed Tomography (CT) slice localization data² where the aim is to retrieve the location of CT slices on the axial axis. The data was constructed from a set of 53, 500 CT images that were taken from 74 different patients (43 male, 31 female).

The second regression dataset is YearPredictionMSD³ data, a subset of Million Song dataset, from the UCI repository. The goal is to predict the release year of a song from its audio features. The songs are mostly western, commercial tracks ranging from 1922 to 2011, with a peak in the year 2000s.

We choose Yahoo LTRC dataset⁴ [1] for the learning to rank task as it is a well-know benchmark dataset and also is used in the XGBoost paper. This dataset has 20K queries, each associated with approximately 22 documents. Train-test split from the original paper is preserved. The second benchmark ranking dataset we used is MSLR-WEB 10K⁵. The dataset contains 10K queries, each of which corresponds to a list of 100 – 200 documents.

3 Hyperparameters of GrowNet

Experiment Setup. All predictive functions added to the model are multilayer perceptrons with two hidden layers. More hidden layers degraded the performance as the model starts overfitting. We generally set the number of hidden layer units to roughly a third of, a half of or equal to the input feature dimension. 40 additive functions were employed in the experiments for all three tasks, and

¹<https://archive.ics.uci.edu/ml/datasets/HIGGS>

²<https://archive.ics.uci.edu/ml/datasets/Relative+location+of+CT+lices+on+axial+axis>

³<https://archive.ics.uci.edu/ml/datasets/YearPredictionMSD>

⁴<https://webscope.sandbox.yahoo.com/catalog.php?datatype=c>

⁵<http://research.microsoft.com/en-us/projects/mslr/>

Table 1: Datasets used in the experiments and their brief description. The second and third columns marked as N, M represent number of samples and feature dimension of the dataset, respectively.

Dataset	N	M	Task
Higgs Bozon	10M	28	Binary classification
Slice localization	53K	384	Regression
Year prediction	515K	90	Regression
Yahoo LTRC	473K	700	Learning to rank
MSLR-WEB 10K	1.2M	136	Learning to rank

number of weak predictors in test time is chosen by the validation results. From all the experiments, we observe that 30 weak learners are more than enough to get the best results before the model performance saturates. Early stopping or other heuristics can also be incorporated into the model to terminate the training before the model begin to overfit.

The boosting rate is initially set to 1 and automatically adjusted during corrective step. Depending on the dataset and the task at hand, it may be initially set to a lower number such as 0.1. In our experiments, we did not tune or alter the boost rate.

We trained each predictive function for just 1 epoch, and the entire model is also trained for 1 epoch during the corrective step using stochastic gradient descent with the Adam optimizer. The Adam optimizer is run with ℓ_2 regularization at a rate of 0.001. Epoch numbers are increased to 2 for the ranking task as we used larger batch sizes. Increasing the epoch number does not contribute to the performance, and higher numbers cause overfitting. We also performed $2D$ batch normalization for the hidden layers. The batch size for classification was set to 2048 and the learning rate was set to 0.005. ReLU was used as the activation function for the penultimate layer, whereas Leaky ReLU was used for the hidden layers. For the ranking task, we replaced ReLU with ReLU6.

The source code is uploaded in a separate file.

4 XGBoost and AdaNet Tuning

4.1 XGBoost Tuning

For XGBoost, we tuned the main parameters, including the number of trees, learning rate, maximum leaves and ℓ_2 regularization in the following range:

- Number of trees: {64, 128, 256, 512, 1000}
- Learning rate: {0.05, 0.1}
- Maximum number of leaves: {128, 256, 512}
- ℓ_2 regularization (lambda): {0, 0.2}

We did not tune XGBoost on Yahoo LTR (ranking task) and Higgs (classification task) datasets as we used the well tuned results reported in the original XGBoost paper [2] as is.

4.2 AdaNet Tuning

We tuned 3 main parameters for AdaNet: the learning rate, the number of sub-networks and the complexity regularization parameter (λ) within the following ranges:

- Learning rate: {0.01, 0.001, 0.0001}
- AdaNet iterations (# subnetworks): {2, 3, 4}
- Complexity regularizer λ : {0.01, 0.001, 0.0001}

The model was first tuned with default mixture weights and $\lambda = 0$, as suggested in the authors' Github page⁶. From this experiment, we got the optimal learning rate and number of sub-networks.

⁶https://github.com/tensorflow/adanet/blob/master/adanet/examples/tutorials/adanet_objective.ipynb

	GrowNet	AdaNet	XGBoost
AUC	0.8401	0.8143	0.8304

Table 2: Classification results on Higgs-1M data. The scores are in AUC-ROC.

	GrowNet 1HL	GrowNet 2HL	GrowNet 3HL	GrowNet 4HL
AUC	0.8288	0.8401	0.8146	0.7801

Table 3: Results from hidden layer experiment.

Then using the learned parameters from the previous setting, AdaNet is again tuned to learn the mixture weights and regularization complexity parameter λ .

The model is trained for 30,000 epochs, and the number of neurons in layers is set to 512, following the results from AdaNet paper [3]. We also observed that the model with 512 neurons generally renders better performance.

4.3 Classification on Higgs-1M

Following the same data split on Higgs data from the XGBoost paper [2], we created Higgs-1M data. Table 2 reports the AUC scores on Higgs-1M data from GrowNet, AdaNet and XGBoost. GrowNet renders favorable results compared to XGBoost and 3% increase over AdaNet result.

4.4 Discussion

The main purpose of these experiments is not to display the absolute dominance of the GrowNet over XGBoost or AdaNet on all tasks, but to provide a proof-of-concept illustrating that our off-the-shelf Neural Network solution is competitive with existing state of the art approaches.

A more fine-tuned and extensive XGBoost results can be found in the [LightGBM Experiment Docs](#), where it is clearly shown that GrowNet still outperforms XGBoost and has on par results with LightGBM.

We did not compare the training time of GrowNet with XGBoost as (1) XGBoost on GPU does not render stable scores, no reproducible results, (2) it has a memory issue. Thus, comparing the training time of GrowNet on GPU and XGBoost on CPU heavily depends on the device capabilities and a fair comparison would be very challenging.

5 Additional Illustrations for Ablation Study

Analogous to Figure 2 from the main text, Figure 1 presents pairwise losses on Microsoft dataset from the ranking task.

Effect of hidden layers. Table 3 reports the results from the hidden-layer experiment. GrowNet final, employing weak learners with 2 hidden layers, got the best performance (AUC score is 0.8401). The model with a shallow network of 1 hidden layer as a weak learner obtains better performance (AUC of 0.8336) once the number of hidden units is increased from 16 to 32. The inverse effect on the model with weak learners of 3 or 4 hidden layers did not work as expected. That is, decreasing the number of neurons in the hidden layers for these predictive functions did not improve much the classification performance.

Details on DNN versus GrowNet

Both Deep Neural Network (DNN) models and Grownets are run on the same machine with NVIDIA Tesla V100 (16GB) GPU.

Unlike GrowNet, DNN performed better with SELU activation functions. We also applied batch normalization on the hidden layers of DNN. Each of DNN models run for 1000 epochs. The results are reported in Table 4. The best performing DNN model has 10 hidden layers, and each epoch took

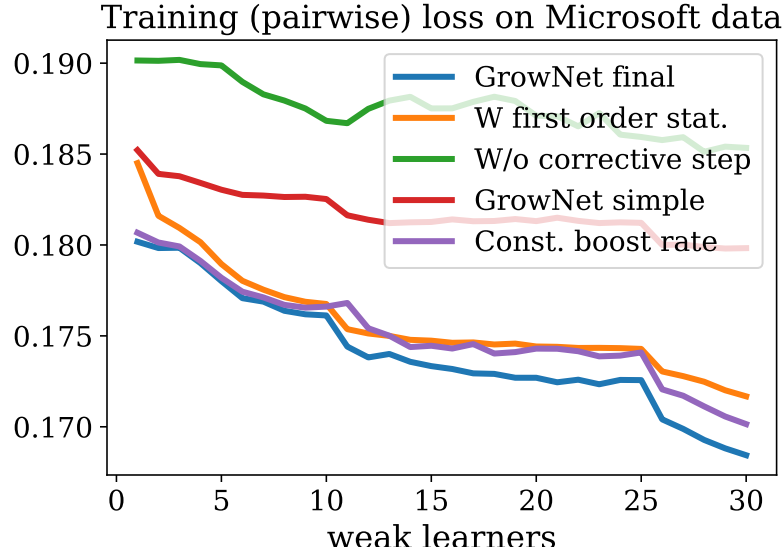


Figure 1: Training loss visualization for the learning to rank task on MSLR dataset. We used pairwise loss.

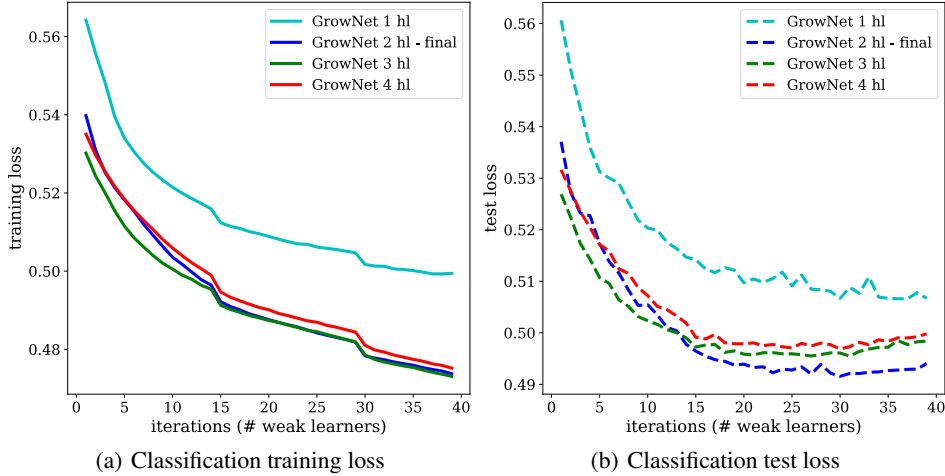


Figure 2: Effect of hidden layers on model training and classification performance (AUC).

approximately 12 seconds. The model reaches its best performance after epoch 900. GrowNet shows a clear advantage on both classification performance and training time.

Both methods, DNN and GrowNet are not fully optimized, thus their training time can slightly be improved. Figure 3 displays the training time of GrowNet while adding new weak learners. DNN with 30 hidden layers are implemented with Dropout(0.3), as without Dropout the model started to overfit immediately after a few epochs. That also explains very close training times of DNN with 20 and 30 layers.

Models	DNN-5	DNN-10	DNN-20	DNN-30	GrowNet
Training time (sec)	10.2	11.6	15.2	15.0	50.1
AUC	0.8288	0.8342	0.8338	0.8301	0.8401

Table 4: Training time and performance comparison between DNN and GrowNet on Higg-1M data. Training time for DNNs are average seconds per epoch and for GrowNet average seconds per stages.

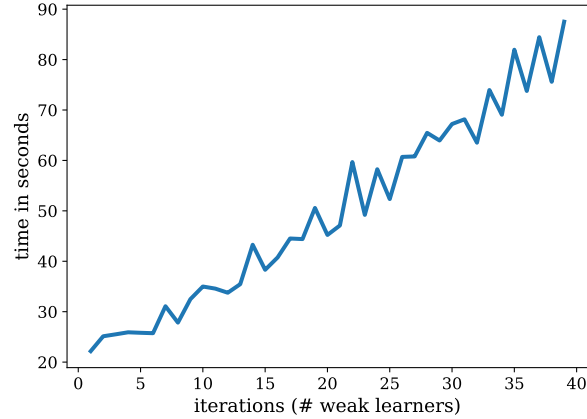


Figure 3: Training time over iterations. As observed, training time is linearly correlated with number of weak learners.

The main point we want to convey in this experiment is showing that GrowNet is (1) much less cumbersome in hyperparameter tuning, (2) faster in training, and (3) better in overall performance than traditional DNNs. We agree that the gain is marginal on classification task, yet the data itself is challenging and 1% improvement is significant in some scenarios. Furthermore, to show the model robustness in hyperparameter selection, we kept the model simple by fixing the hidden unit size to half of the feature dimension (as observed the best results from GrowNet on Higgs data is with 128 hidden units).

References

- [1] Chapelle, O. and Chang, Y. Yahoo! learning to rank challenge overview. *Journal of Machine Learning Research - W & CP*, 14:1–24, 2011. [1](#)
- [2] Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. ACM, 2016. [2](#), [3](#)
- [3] Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., and Yang, S. AdaNet: Adaptive structural learning of artificial neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, 2017. [3](#)
- [4] Moghimi, M., Belongie, S. J., Saberian, M. J., Yang, J., Vasconcelos, N., and Li, L.-J. Boosted convolutional neural networks. In *BMVC*, 2016. [1](#)
- [5] Zhang, F., Du, B., and Zhang, L. Scene classification via a gradient boosting random convolutional network framework. *IEEE Transactions on Geoscience and Remote Sensing*, 54, 2016. [1](#)