**Using XGBoost for Deep Learning**

Integrating XGboost with Convolutional Neural Networks

[[](https://freddynjagi.medium.com/?source=post_page-----8b2b23b50f8f--------------------------------)](https://freddynjagi.medium.com/?source=post_page-----8b2b23b50f8f--------------------------------)

[[Heartbeat](https://heartbeat.comet.ml/?source=post_page-----8b2b23b50f8f--------------------------------)](https://heartbeat.comet.ml/?source=post_page-----8b2b23b50f8f--------------------------------)

[Mwanikii](https://freddynjagi.medium.com/?source=post_page-----8b2b23b50f8f--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2Fa7870ab3194e&operation=register&redirect=https%3A%2F%2Fheartbeat.comet.ml%2Fusing-xgboost-for-deep-learning-8b2b23b50f8f&user=Mwanikii&userId=a7870ab3194e&source=post_page-a7870ab3194e----8b2b23b50f8f---------------------post_header-----------)

Published in

[Heartbeat](https://heartbeat.comet.ml/?source=post_page-----8b2b23b50f8f--------------------------------)

·

4 min read

·

Nov 15, 2023

53



Photo by [Alexander Grey](https://unsplash.com/es/@sharonmccutcheon?utm_source=medium&utm_medium=referral) on [Unsplash](https://unsplash.com/?utm_source=medium&utm_medium=referral" \t "_blank)

XGBoost is a powerful library that performs gradient boosting. It has an excellent reputation as a tool for predicting many kinds of problems in data science and machine learning. It is also used in regression and classification problems as it is highly intuitive and easily interpretable.

One good indicator of how powerful XGBoost is is the number of winning solutions that utilize this model on Kaggle. For many years, gradient-boosting models and deep-learning solutions have won the lion's share of Kaggle competitions.

Using XGBoost can quickly help you get from start to finish of a project with minimal initial feature engineering.

One robust use case for XGBoost is integrating it with neural networks to perform a given task. XGBoost is not limited to machine learning tasks, as its incredible power can be harnessed when harmonized with deep learning algorithms.

We can now focus on how to approach XGBoost from a deep learning standpoint and even leverage technical information on building the architecture we aim to leverage.

First of all, to implement this, we need to be familiar with the libraries that contain our model:

1. XGBoost
2. Tensorflow and Keras

All the above libraries can be installed by checking on their official sites. For clarity, Tensorflow and Pytorch can be used for building neural networks.

**ConvXGB Introduction**

ConvXGB is a model that incorporates both Convolutional Neural Networks and XGBoost. It was envisioned by Thongsuwan et al., who found inherent advantages to the architecture and highlighted a few shortcomings by trying to leverage each model individually. Still, there was an excellent way to look at this:

1. Using XGBoost is "still unclear for feature learning [1]."
2. Integrating Convolutional Neural Networks provides the added benefit of better feature learning, eliminating the shortcoming of XGBoost.

The process above is standard, as practitioners often combine models in different ways for better performance. ConvXGB explores this but with a few exciting caveats that the researchers included. The Convolutional Neural Network's architecture differs slightly from the usual architecture.

We can now explore ConvXGB's architecture and look keenly at this model's nitty-gritty details.

**Architecture**

This model's architecture is categorized into two sections: The feature learner and the class predictor. Both sections are further subdivided into the critical steps to implement this model successfully.

The feature learning section has an input section and a data preprocessing section. The idea behind this is that it's necessary to have data in the appropriate form to be ingested by the convolutional neural network. For data that is not in a tuple or Numpy array form, the data preprocessing layer is responsible for performing this conversion. If you aren't familiar with this step, check out my[articles](https://medium.com/cometheartbeat/understanding-memory-mapping-in-numpy-for-deep-learning-pt-1-2bfa8319f79b) discussing Numpy arrays and the constraints that you may experience dealing with them.

The convolutional neural network has some exciting features that are not typical of the average CNN. Thongsuwan et al. hypothesized that reducing parameters would reduce simplicity for a few reasons. This would happen by not including a fully connected layer and pooling layers because "it is not necessary to bring weights from the FC layers back to re-adjust weights in the previous layers [1]."

Now, the data can be fed into the predicting section of the model. This section consists of three parts: the reshaping layer, the class prediction layer, and the output layer.

**Benefits and Drawbacks**

This architecture has several benefits, as the paper's writers demonstrated, but they have shortcomings.

One of the most impressive benefits of this architecture is how well it performs against other similar algorithms. In many of the benchmarks in the paper, the ConvXGB architecture outperformed other similar models.

Despite the excellent performance, there were a few drawbacks. For instance, one has to get the number of convolution layers right, as coming up with too many may increase the architecture's computational cost. The authors admit that it might "outweigh any advantage, and we must balance carefully any increase in accuracy with the cost incurred [1]."

**Conclusion**

The ConvXGB is a powerful model that combines the power of XGBoost with deep learning. Its simplicity and ease of implementation allow for inference and strong performance if implemented correctly.

In our next article, we can try an implementation of the model.

**Sources:**

[1]Thongsuwan, Setthanun, Saichon Jaiyen, Anantachai Padcharoen, and Praveen Agarwal. "ConvXGB: A new deep learning model for classification problems based on CNN and XGBoost." *Nuclear Engineering and Technology* 53, no. 2 (2021): 522–531.

*Editor's Note:*[*Heartbeat*](https://heartbeat.comet.ml/)*is a contributor-driven online publication and community dedicated to providing premier educational resources for data science, machine learning, and deep learning practitioners. We're committed to supporting and inspiring developers and engineers from all walks of life.*

*Editorially independent, Heartbeat is sponsored and published by*[*Comet,*](http://comet.com/?utm_source=heartbeat&utm_medium=referral&utm_campaign=AMS_US_EN_AWA_heartbeat_EN)*an MLOps platform that enables data scientists & ML teams to track, compare, explain, & optimize their experiments. We pay our contributors, and we don't sell ads.*

*If you'd like to contribute, head on over to our*[*call for contributors*](https://heartbeat.comet.ml/call-for-contributors-fee7f5b80f3e)*. You can also sign up to receive our weekly newsletter (*[*Deep Learning Weekly*](https://www.deeplearningweekly.com/)*), check out the*[*Comet blog*](https://www.comet.com/site/blog/)*, join us on*[*Slack*](https://join.slack.com/t/cometml/shared_invite/zt-49v4zxxz-qHcTeyrMEzqZc5lQb9hgvw)*, and follow Comet on*[*Twitter*](https://twitter.com/Cometml)*and*[*LinkedIn*](https://www.linkedin.com/company/comet-ml/)*for resources, events, and much more that will help you build better ML models, faster.*