Bayesian Additive Neural Network via Neural Tangent Kernel

##Architectural advantages of using Computation Skeleton

Building the Bayesian Additive Neural Network, we adapted the architecture called computation skeleton by constructing BNNs block by block to help us efficiently approximate the posterior of DGP and incorporate structural assumptions on BNNs. Besides, the computation skeleton can also provide additional advantages when relating kernel methods and neural networks in an over-parameterized regime. We explore this advantage by making a connection with hierarchical learning that is inherent in our structure.

We first review [Allen-Zhu et al.](https://arxiv.org/pdf/1905.10337.pdf), where they propose a theory that proves a multi-layer residual network can implicitly perform hierarchical learning using different layers which could reduce the sample complexity without any distributional assumption and compares it to 'one-shot' learning algorithms such as kernel methods. In the paper, they point out that to deal with the difficulty of modeling real-world distribution, kernel methods are popularly used; and a recently developed technique called Neural tangent Kernel is well used to relate over-parameterized neural networks and kernel method. However, they argue in practical tasks, neural networks yield much better generalization error compared to kernel methods. Therefore, they try to prove that neural networks can also learn concept classes in a distribution-free setting. They argue that in the simultaneous training process of residual network, the lower-level layers automatically learn an 'approximation' of the lower complex base signal in the target function. It then forwards these features via *residual link* to the higher-level layer to further learn the more complicated composite signal which reduces the sample complexity and avoids overfitting with a different inductive bias.

This theory of hierarchical learning parallels nicely with our idea of using the computation skeleton for BNNs. Our computation skeleton has a feed-forward computation multi-layer graph structure, in which the first hidden layer serves to learn simple features and the second hidden layer learns complex features with help of what they learned prior.

Additionally, our main architecture, Bayesian Additive Neural Network, has an additive structure where a large neural network is composed by summing several sub neural networks with different widths on the second hidden layer of respective sub neural networks. To be specific, we set all the first hidden layer of sub neural networks with the same number of neurons

which we defined as a Random feature block. This will help us construct random feature approximation for kernels to leverage the expressive power of DGP but also learn 'shared' simple features or lower complex base signals. Then, we set the second hidden layer of sub neural networks with different numbers of neurons each one wider than the others in which will help us not only with defining the bayesian approximation but also learn different levels of the complex features. Therefore, our additive structure can imitate the inherent hierarchical learning using the residual link on ResNet by summing and feed forwarding these simultaneously trained sub neural networks with different training dynamics.

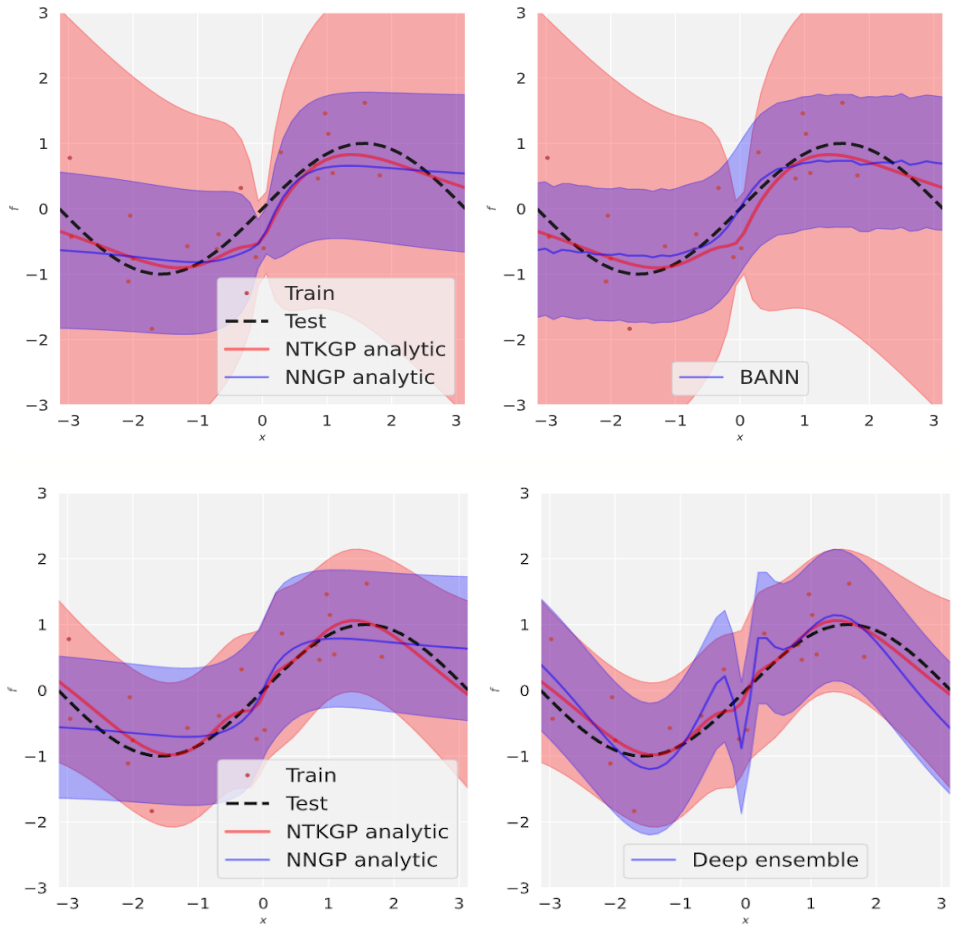
Posterior predictive via NTK with Bayesian Additive Neural network

We now take our model to an over-parameterized regime by adapting NTK parameterization and see how our learning process performs posterior interpretation in the infinite width limit. In the prior experiments, to approximate the posterior, we defined our kernels with help from our computation skeleton's Random feature Block and approximated the posterior via doubly stochastic variation inferences with uncertainty estimates. For this experiment, we first use NTK parameterization to provide a posterior predictive from our model when taken to the infinite width limit. To be more specific, unlike our prior experiment, we will define our kernel by capturing randomly initialized parameters in a wide neural network during training and taking first-order Taylor expansion regarding parameters at initialization. While the Neural Tangent Kernel will only depend on certain architecture choices, not width since it will let hidden layer widths of our model go to infinity, this method will allow us to achieve analytically available NTKGP posterior predictive and NNGP posterior predictive.
Next, we provide another posterior interpretation in the infinite width limit using standard parameterization by training all layers of a finite width neural network. While standard parameterization is not so much different from NTK parameterization on the perspective that both parameterizations produce fixed kernels corresponding to an infinite neural network, NTK parameterization fails to take the relative widths in a finite network into account which is an important aspect to capture for the training dynamics of our model architecture. Standard parameterization proposed by [Sohl-Dickstein et al.](https://arxiv.org/pdf/2001.07301.pdf) allows us to take differences in relative layer widths into account that

will have an effect on training dynamics of weight and bias parameter and learning rate scale. Therefore, by defining the kernel using standard parameterization, it will let us train all layers of our model and capture important aspects of our architecture including the inherent ability of hierarchical learning process, and also use L2 norm as a regularizer as we did when approximating Bayesian posterior with finite width BANN.
In the experiments, we compare our results with Bayesian Deep Ensembles via NTK proposed by [He et al.](https://arxiv.org/abs/2007.05864) which takes simple modification to train the ensembles of neural networks and obtain posterior predictive in the infinite width limit.

##Experiments

In our first experiment, we use noisy toy data to perform the regression task and try to achieve posterior predictive for the target function. The most compelling part of the result from this experiment is that posterior predictive from BANN emulates the posterior predictive performed with the NNGP kernel. This is plausible since similar learning processes to computation skeleton are used to produce posterior predictive with NNGP kernel. [Lee et al.](https://arxiv.org/abs/1902.06720) showed that to obtain a posterior interpretation to an infinitely wide network, one way is to randomly initialize the network but only train the parameters in the final layer. Then, the sample from the posterior with NNGP kernel as prior corresponds to the infinite neural network since NNGP(Neural Network Gaussian Process) kernel can be seen as a contribution to the NTK from the parameters in the final hidden layer. This process can also be seen as a "sample-then-optimize" procedure from [Matthews et al.](http://approximateinference.org/2017/accepted/MatthewsEtAl2017.pdf) except it limits the earlier layers in the model exclusively to be random feature extractors by only training the final layer. This procedure aligns with our structural design using computation skeleton, as we locate the Random feature block first to extract random features, following with Function block to define Bayesian approximation in our respective sub neural networks.
In this experiment, to precisely show the architectural effect when obtaining posterior predictive in infinite width limit we ensembled our BANN model with the same ensemble size as standard deep ensembles. Plots on the left of [Figure 1] show posterior using NTK parameterization and posterior using standard parameterization on the right.



[Figure 1](BANN vs Deep ensembles on toy data)

Next, we perform the classification task with MNIST data for large-scale experiments. Since our prior experiments performed large-scale regression experiments using BANN, for this experiment we show our model can also perform well on the classification task. For this experiment, we will not use the ensemble method and change our loss method and optimizer method to Adam from SGD. We will also use Relu activation instead of Erf activation since NTK GP posterior will not be analytically available for large-scale experiments. We can see from the result that our model learns well for the classification task, however, in order to get posterior interpretation in infinite width limit, we needed to treat the label smooth targets as regression targets.

##Discussions

There are few aspects of the experiment settings that we can discuss, for example, by using the standard parameterization we can take different

training dynamics set by variation of width in the finite BANN. How do these different training dynamics and learning rate scales contribute to our inherent ability of hierarchical learning in the computation skeleton?
Another concern is, in the first experiment using noisy toy data we ensembled our BANN to achieve Bayesian posterior predictive. This was to compare and show our architectural advantage, however, without ensemble BANN posterior in infinite width limit does not show perfectly smooth line. This could be due to the narrow width on our subNN compare to the standard deep ensemble([Arora et al.](https://arxiv.org/abs/1904.11955) says there is a certain threshold for the width in the finite NN) but it could be other reasons, such as lack of extra variance.
Lastly, how can we elaborate on how the dropout layer in our subNN contributed to obtaining posterior predictive in infinite width limit?