Bayesian Inference with Boosted Importance Nested Sampling

Garv Shah, Johannes Lange
Leinweber Center for Theoretical Physics, University of Michigan, Ann Arbor

Introduction

This project aims at improving the accuracy of artificial neural networks used as part of a larger algorithm for Bayesian posterior and evidence estimation. The Bayesian code uses neural networks to choose better samples from the parameter space. Different neural network architectures were implemented using the TensorFlow framework. Among others, we test how the predictive performance depends on the number of neurons and the activation function. The results are compared with the neural network architecture currently implemented in the codebase using the scikit-learn package.

Figure 1

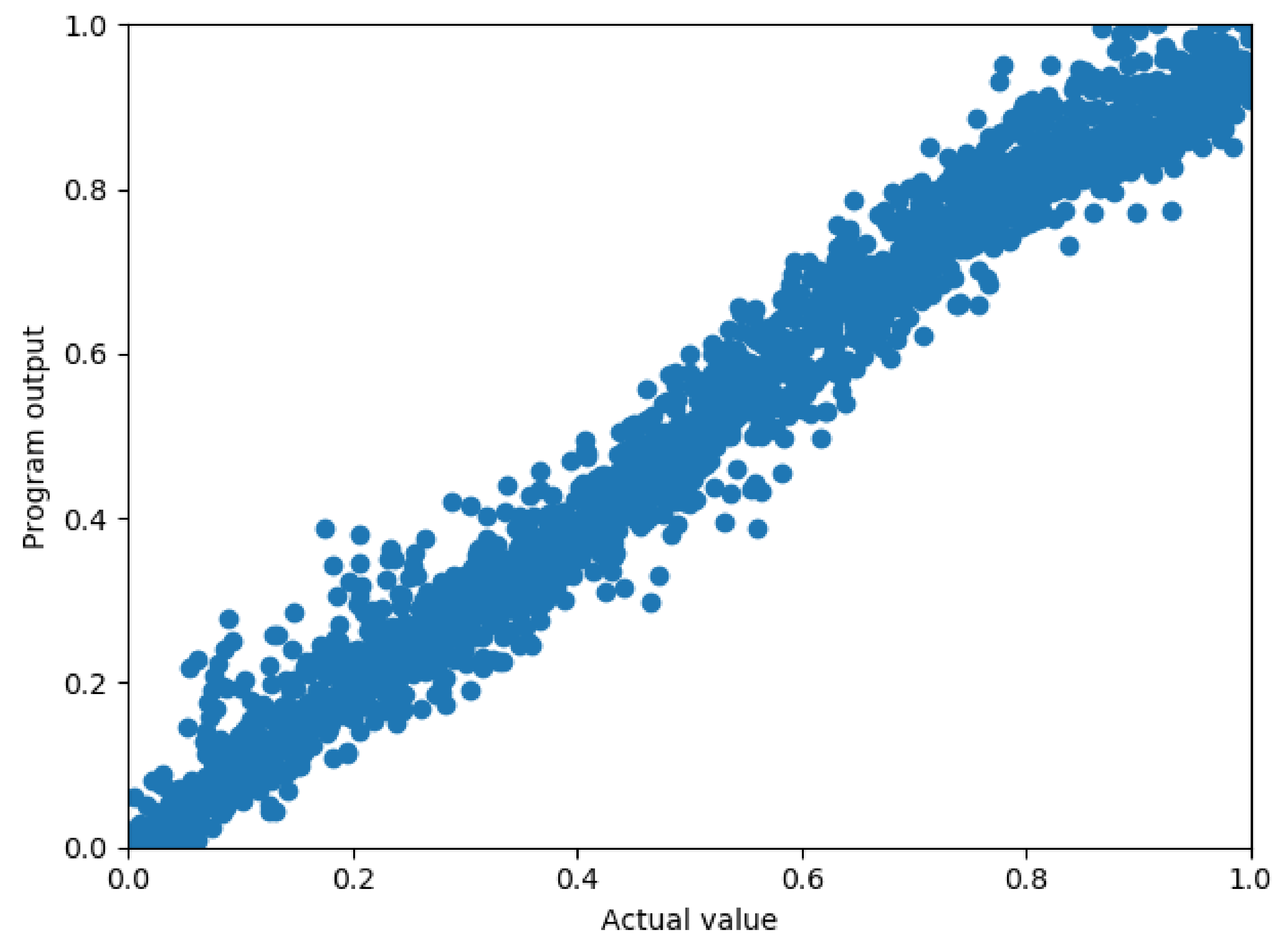
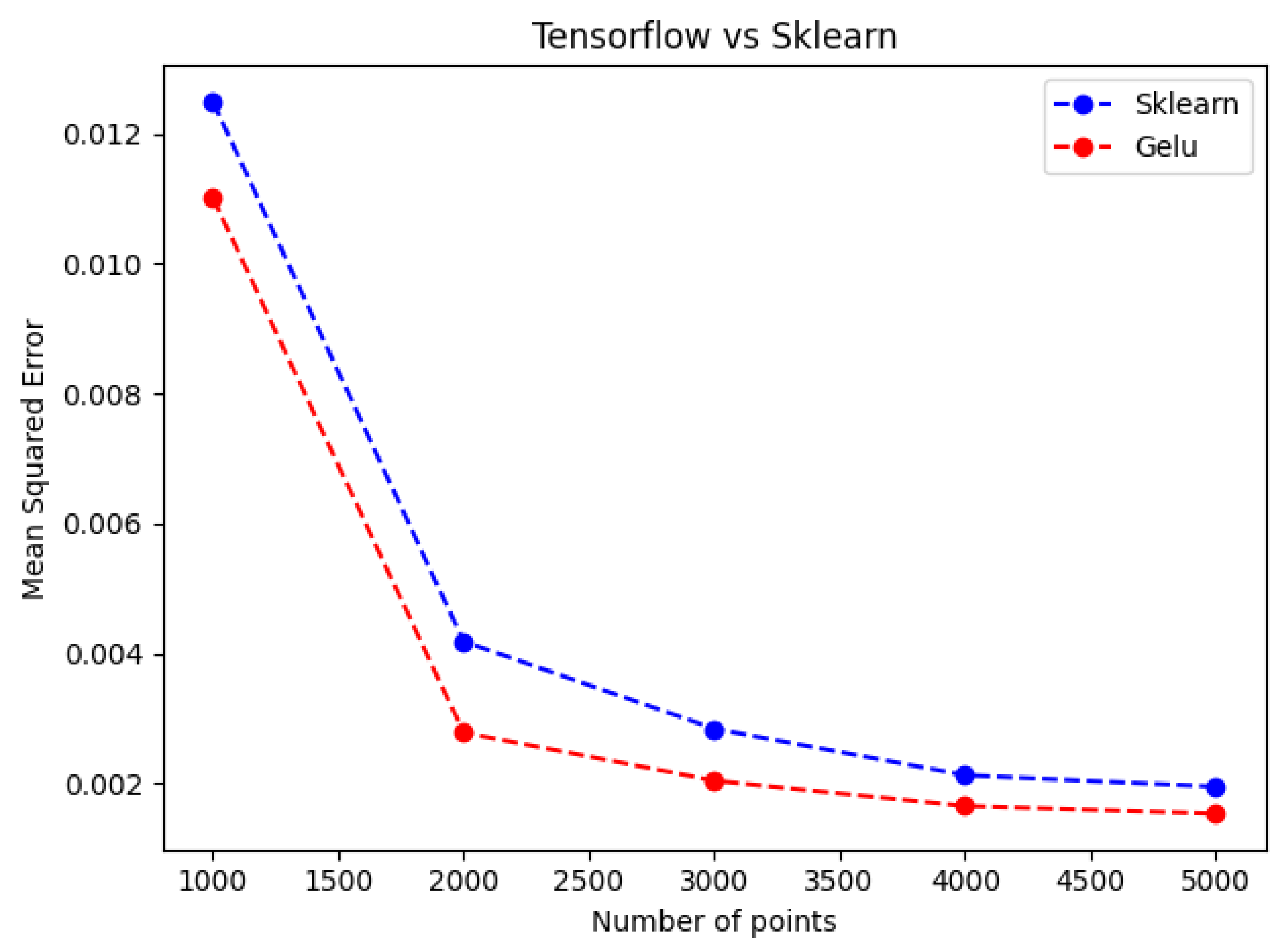


Figure 1 represents a scatterplot of the predicted vs actual likelihood score for a test problem. The problem is question is loggamma[1], with 10 dimensions. We can see that the line of best fit is nearly an x = y line. That means the TensorFlow model was able to predict the likelihood scores for a randomly generated live set with good accuracy. This model was trained with 4 layers, the 'gelu'[2] activation function, and 'adam' as the optimizer.

Results

Figure 2

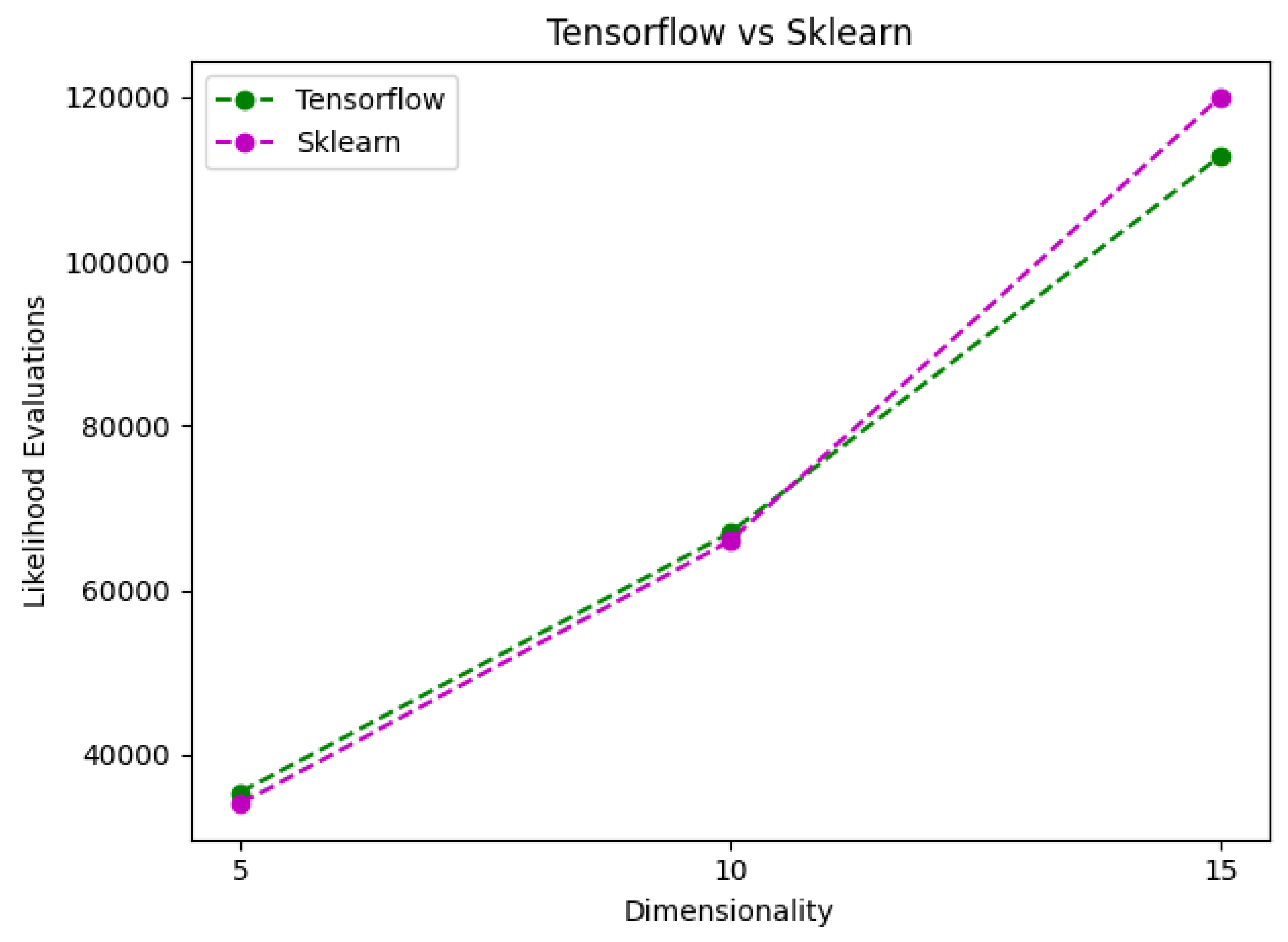


In the study, gelu, tanh, and swish activation functions within TensorFlow were tested on the loggamma-10 problem, all with the adam optimizer. Out of the three, gelu proved the most promising. The TensorFlow model (with gelu) and the Sklearn model were run 100 times each for 1000, 2000, 3000, 4000, and 5000 points. The results have been plotted on Figure 2. Both models perform very similarly, with the TensorFlow model slightly outperforming the Sklearn model. To achieve these results, the TensorFlow model required 120 epochs to be run. Despite utilizing GPU optimization, the runtime of the TensorFlow model was considerably longer than that of the Sklearn model.

From these results, we expect the TensorFlow model to perform slightly better in the nautilus[3] implementation. The results for the nautilus implementation have been discussed in detail in the conclusion.

Conclusion

Figure 3



From Figure 3, we can see that the Sklearn model performs slightly better for a lower number of dimensions, but the TensorFlow model performs noticeably better at higher dimensions, for the loggamma problem. For 5 dimensions, the Sklearn model makes 34,000 likelihood evaluations and the TensorFlow model makes 35,400. For 15 dimensions, the Sklearn model makes 120,000 likelihood evaluations, whereas the TensorFlow model makes 112,800. This is significantly better.

When considering the use of the TensorFlow model, it's essential to take into account that this model requires a much longer runtime due to the high number of epochs required for optimal performance. While the TensorFlow model may yield better results for certain tasks, it might not be the most efficient choice for lower-dimensional tasks or when time is a critical factor. Therefore, it may be effective to use the TensorFlow model only for a higher number of dimensions when the results are significantly better.

References:
1. Buchner, J. (2014). *A statistical test for Nested Sampling algorithms,* section 8.2. Springer Science+Business Media New York 2014
2. (n.d.). *Tf.Keras.Activations.Gelu.* TensorFlow. Retrieved April 7, 2023, from https://www.tensorflow.org/api_docs/python/tf/keras/activations/gelu#reference_1
3. Lange, J. U. (n.d.). *Nautilus.* Nautilus. Retrieved April 9, 2023, from https://nautilus-sampler.readthedocs.io/en/stable/
4. Lange, J. U. (2023). *Boosting Importance Nested Sampling for Bayesian Posterior and Evidence Estimation with Neural Networks* (Publication Pending)