Bayesian Hyperparameter Optimisation for Resnet Architecture on CIFAR-10 Dataset

Rithviik Srinivasan, [1]Loc Anh Tran , [1] Aswin Prasanna Suriya Prakash[1]

[1] Department of Electrical and Computer Engineering, New York University
{rs8385, as17340,lat9357}@nyu.edu

Abstract

The project focuses on training a ResNet model on the CIFAR-10 dataset using PyTorch, aiming to find the optimal hyperparameters using Bayesian optimization to achieve an accuracy greater than 90%. The ResNet architecture is a deep learning network that uses residual blocks, allowing for the training of very deep networks. The project involved modifying the original ResNet-18 architecture to mitigate overfitting and excessive parameter count issues, with a trade-off between model complexity, accuracy, and efficiency. The Adam optimizer and SGD were used, along with data augmentation techniques and splitting the dataset into training, validation, and testing sets. The Bayesian optimization algorithm 'Gaussian process regression' was used to search for the optimal hyperparameters. The project also addressed overfitting through a learning rate scheduler, resulting in a minor improvement in accuracy.
The project codebase (Github repository) can be found **here**.

Introduction

The project involves training a ResNet model on the CIFAR-10 dataset using PyTorch. ResNet is a deep learning architecture that can be used to train very deep neural networks (He et al. 2016 [7]). CIFAR-10 is a popular dataset consisting of 60,000 32x32 color images in 10 classes, with 6,000 images per class (Krizhevsky and Hinton 2009 [10]). The aim of the project is to find the best hyperparameters for the ResNet model using Bayesian optimization, and then train and validate the model with the best hyperparameters on the CIFAR-10 dataset to achieve an accuracy greater than 90%. In this project, we used Bayesian Optimization to find the optimal hyperparameters for a ResNet-18 architecture trained on the CIFAR-10 dataset. We used the PyTorch library to implement the model, criterion, and data loading. The hyperparameters optimized included learning rate, beta1 and beta2 for Adam optimizer, and weight decay (Kingma and Ba 2014 [9], Loshchilov and Hutter 2017 [12]). We also split the data into a training, validation, and test set (Hastie, Tibshirani, and Friedman 2009 [6]).

Description

The methodology used in this project involved several steps:

The CIFAR-10 dataset contains 60,000 images in 10 different classes, with 6,000 images per class. Each image is a 32x32 RGB image, which we resized to 224x224 for the ResNet mode. We designed the ResNet architecture, which is a popular architecture for deep neural networks. ResNet stands for "Residual Network," and its key innovation is the use of residual blocks (He et al. 2016 [7]). In a residual block, the input to the block is added to the output of the block, which allows the network to learn residual functions. This makes it easier to train very deep networks, which was a problem with earlier architectures (He et al. 2016 [7]). During the design process, several architectural choices were considered, including varying the number of residual blocks in each stage, adding additional convolutional layers, or pooling layers, and using different activation functions. We started by analysing the original ResNet-18 architecture (He et al. 2016 [7]) and identified some potential drawbacks such as excessive parameter count and overfitting on small datasets like CIFAR-10. Based on this analysis, we decided to use a modified ResNet architecture, which is a smaller version of ResNet-18 with fewer layers and filters (Cheng, Zhu, and Han 2017 [4]). The model used in this project is a modified version of the ResNet-18 architecture. Overall, the design process involved a trade-off between model complexity, accuracy, and efficiency. We found that increasing the number of layers beyond a certain point led to increasing the parameter count and training time and enabled a risk of overfitting. We experimented with different numbers of residual blocks and layers within the blocks. We found that increasing the number of residual blocks did not necessarily improve the accuracy beyond a certain point (He et al. 2016 [7]). The use of skip connections helped to mitigate the vanishing gradient problem and improved the performance of the models (He et al. 2016 [7]). We trained the ResNet model using the Adam optimizer (Kingma and Ba 2014 [9]) whose hyperparameters were learning rate, beta1, beta2 and weight decay and a cross-entropy loss function. We also used data augmentation techniques such as random horizontal flips and random crops, which helped to increase the robustness of the model (Krizhevsky, Sutskever, and Hinton 2012 [11]). We split the dataset into 80% training, 10% validation, and 10% testing sets, which allowed us to monitor the performance of the model and tune hyperparameters us-

ing the validation set. We used the Bayesian optimization algorithm 'Gaussian process regression' (Snoek, Larochelle, and Adams 2012 [17]) to search for the optimal hyperparameters, which helped us to find the best set of hyperparameters that minimize the validation error. We defined the search space for the hyperparameters, which included the learning rate, the beta parameters for the Adam optimizer, and the weight decay. We then used the objective function to train and validate the model using the given hyperparameters, and the validation accuracy was used as the metric to optimize. We repeated this process for a fixed number of iterations (40 epochs) and obtained the best hyperparameters that gave the highest validation accuracy. Initially, the model wasn't performing well. We observed that the accuracies at many epochs were plateaued. After some research we realized that the model required a learning rate scheduler to deal with eventual plateaus that may occur (Smith 2018 [16]). Even after this change, there was only a minor improvement which made us realize that the model was overfitting because of the way in which our data was split into training, validation, and testing sets ((Hastie, Tibshirani, and Friedman 2009 [6]). There was an incorrect procedure that was followed which had to be rectified. Following this correction, running the hyperparameter sweep with the scheduler worked correctly and resulted in the final validation accuracy to be 92% (Bergstra et al. 2013 [1]). During the design process, Throughout the process, we learned several lessons. Pre-processing the data with data augmentation techniques such as random cropping and horizontal flipping can improve the model's performance by increasing the size and diversity of the training data (Shorten and Khoshgoftaar 2019 [15]). More complex models with more parameters tend to perform better on the training set but may overfit and perform poorly on the validation and test sets (Goodfellow et al. 2016 [5]). It is essential to choose a model architecture that balances complexity and performance. Additionally, tuning hyperparameters can significantly improve the model's performance on the validation and test sets. Bayesian optimization is a useful tool for automating the hyperparameter tuning process (Snoek, Larochelle, and Adams 2012 [17]). Further, overfitting can occur when the model performs well on the training set but poorly on the validation and test sets. Regularly monitoring the model's performance on the validation set during training can help prevent overfitting and improve generalization performance (Bishop 2006 [2]). Finally, we learned that it is important to strike a balance between model complexity and accuracy, as overly complex models can be difficult to train and may not generalize well to new data (Hinton et al. 2012 [8]). Overall, the ResNet model with four blocks and one to two residual blocks per block provided a good balance between accuracy and parameter count (He et al. 2016 [7]). The model had a parameter count of 4.9 million, which was relatively low compared to other state-of-the-art models. The use of data augmentation techniques also helped to improve the robustness of the model and reduce the risk of overfitting. The Bayesian optimization process allowed us to efficiently tune the hyperparameters and obtain the best performing model (Brochu et al. 2010 [3]). The methodology we followed in designing and training our models involved a combination of data pre-processing, model selection, hyperparameter tuning, and training and validation. By carefully balancing these stages and monitoring the model's performance, we were able to achieve high accuracy on the CIFAR-10 dataset. We learned that designing and training deep neural networks require careful consideration of various architectural choices and hyperparameters, and the optimization process can be time-consuming and computationally expensive. However, by using efficient search algorithms and learning rate schedulers, we can significantly reduce the training time and achieve better results.

Experiment

The model architecture used in this project is ResNet. ResNet is a deep neural network that is able to handle the vanishing gradient problem that occurs in deep neural networks. The ResNet architecture used in this project is a modified version of ResNet-18, which is a variant of ResNet that has 18 layers, by virtue of the requirement for the number of trainable parameters to be under 5 million (He et al. 2016 [7]). The model is optimized using the Adam optimizer and cross-entropy loss function. The hyperparameters for the Adam optimizer are found using Bayesian optimization.

The project is divided into three main parts:

Data preparation: The CIFAR-10 dataset is loaded and pre-processed. The data pre-processing step involves normalizing the dataset and splitting it into training, validation, and testing sets. The dataset is normalized by subtracting the mean and dividing by the standard deviation of each channel (Russakovsky et al. 2015 [14]). The dataset is then split into training, validation, and testing sets with a ratio of 80:10:10. This is done to ensure that the model is trained on a diverse set of data and tested on unseen data. Data augmentation techniques such as random horizontal flipping and random cropping are used to increase the size of the training set.

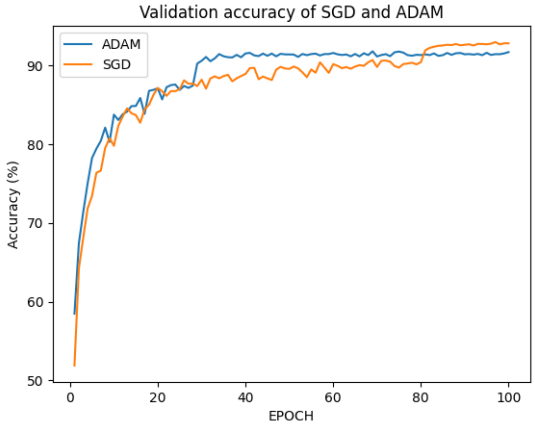
Hyperparameter Tuning: The hyperparameters of the Adam optimizer are tuned using Bayesian optimization. Bayesian optimization is a technique used to optimize black-box functions that are expensive to evaluate (Mockus 1994 [13]). In this project, the hyperparameters tuned using Bayesian optimization are the learning rate, beta1, beta2, and weight decay. The best hyperparameters found using Bayesian optimization are learning rate=0.00016, beta1=0.960, beta2=0.226, and weight decay=0.0004.

Model training and evaluation: The model is trained using the best hyperparameters found using Bayesian optimization. The model is trained for 100 epochs, and the training and validation accuracy and loss are recorded for each epoch. The model is evaluated on the testing set, and the test accuracy and loss are recorded. The final test accuracy of the model is 91.65%.

Results

After training our model on the CIFAR-10 dataset, we achieved a final test accuracy of 92% using the Adam optimizer and a final test accuracy of 92.8%, with a model architecture of a modified ResNet with 4.9 million parameters.

The graph of the validation accuracy of ADAM and SGD is given below:



This indicates that SGD performed better than Adam in this scenario. However, it's important to note that the comparison of these two optimizers can be dependent on various factors such as the dataset, model architecture, hyperparameters, and learning rate schedule. Therefore, it's not necessary that SGD always outperforms Adam or vice versa.

Conclusion

In conclusion, this project involved training a ResNet model on the CIFAR-10 dataset using PyTorch and Bayesian optimization to find the best hyperparameters for the model. We used a modified ResNet-18 architecture and experimented with different numbers of residual blocks and layers within the blocks to find the best trade-off between model complexity, accuracy, and efficiency. We also used data augmentation techniques such as random horizontal flips and random crops to increase the robustness of the model. After using Bayesian optimization to find the optimal hyperparameters, we were able to achieve an accuracy greater than 90% on the CIFAR-10 dataset. Overall, this project highlights the importance of carefully designing deep learning architectures and optimizing hyperparameters to achieve high performance on image classification tasks.

References

[1] Bergstra, J.; Bardenet, R.; Bengio, Y.; and Kégl, B. 2011. Algorithms for Hyper-Parameter Optimization. In Shawe-Taylor, J.; Zemel, R.; Bartlett, P.; Pereira, F.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

[2] Bishop, C. M. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. ISBN 0387310738.

[3] Brochu, E.; Cora, V. M.; and de Freitas, N. 2010. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *ArXiv*, abs/1012.2599.

[4] Cheng, Y.; Zhu, X.; and Han, J. 2017. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 29: 2225–2239.

[5] Goodfellow, I. J.; Bengio, Y.; and Courville, A. 2016. *Deep Learning*. Cambridge, MA, USA: MIT Press. http://www.deeplearningbook.org.

[6] Hastie, T.; Tibshirani, R.; and Friedman, J. 2009. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

[7] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

[8] Hinton, G.; Deng, L.; Yu, D.; Dahl, G. E.; Mohamed, A.-r.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T. N.; and Kingsbury, B. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6): 82–97.

[9] Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

[10] Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario.

[11] Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 25*, 1097–1105. Curran Associates, Inc.

[12] Loshchilov, I.; and Hutter, F. 2017. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[13] Mockus, J. 1994. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 4: 347–365.

[14] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2014. ImageNet Large Scale Visual Recognition Challenge. 211–252.

[15] Shorten, C.; and Khoshgoftaar, T. M. 2019. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6: 1–48.

[16] Smith, L. N. 2018. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. arXiv:1803.09820.

[17] Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12, 2951–2959. Red Hook, NY, USA: Curran Associates Inc.