COMP-SCI 5590 / MATH 5555

Final Project Paper

12/17/2023

Team Members: Michael Dang, Karthik Chellamuthu, Adriana Martinez Cappello

For the In-class Kaggle competition, we decided to use ResNet-50, which is known for its effectiveness in addressing challenges related to the training of deep networks. Its innovative design enables better convergence during training, improved generalization performance, and state-of-the-art results on tasks like image classification, object detection, and segmentation.

Nesterov Accelerated Adaptive Moment Estimation, and it is an optimization algorithm that combines the advantages of Nesterov accelerated gradient (NAG) and Adam optimization, where Adam can be viewed as a combination of RMSprop with momentum. NAG involves looking ahead in the direction of the momentum, and Adam optimizes the learning rates for each parameter individually. Nadam is known for its efficiency in training deep neural networks, particularly in tasks involving large datasets.

When using the Nadam optimizer, practitioners often experiment with hyperparameters such as the learning rate, beta parameters, and epsilon. The learning rate is a crucial parameter, and its selection can significantly impact the convergence and performance of the model. It is common practice to perform a grid search or use techniques like random search to find suitable hyperparameter values.

In conclusion, Nadam has shown good performance in various deep-learning tasks, especially when dealing with large datasets and complex models. The combination of Nesterov momentum and adaptive learning rates in Nadam helps in achieving faster convergence and better generalization. However, the effectiveness of an optimizer may also depend on the specific characteristics of the dataset and the architecture of the neural network. Experimentation with hyperparameters remains a key aspect of achieving optimal results.

Some changes appear to Nadam vs the original SGD provided:

- Adam can be viewed as a combination of RMSprop with momentum. RMSprop contributes to the exponentially decaying average of past squared gradients $v_t$, while momentum accounts for the exponentially decaying average of past gradients $m_t$. Hence, we add in the exponential moving average of the squared gradients and gradients.
- Add the bias correction terms of the momentum vector of the previous time step $\frac{\beta_1 m_{t-1}}{1-\beta_1^t}$.
- Add the Adam step $\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\widehat{v}_t}+\epsilon}(\beta_1 \widehat{m}_{t-1} + \frac{(1-\beta_1)g_t}{1-\beta_1^t})$.
- Replace the momentum != 0 to the end part with 3 steps above.

Why Nadam vs SGD?

Nadam is often preferred over SGD when dealing with deep neural networks, especially in scenarios with large datasets and complex architectures. The adaptive learning rates and Nesterov momentum in Nadam contribute to faster convergence and robustness to various challenges encountered during training. However, it's essential to experiment and fine-tune hyperparameters to find the optimal optimizer for your specific task and model.

Nadam vs SGD vs Adam

SGD (Stochastic Gradient Descent):

- Pros:
  - Simplicity: SGD is straightforward and computationally less intensive compared to more complex optimizers.
  - Memory Efficiency: It requires less memory compared to optimizers like Adam and Nadam.
- Cons:
  - Fixed Learning Rate: SGD uses a fixed learning rate for all parameters, making it less adaptive to different features and landscapes in the loss function.
  - Slow Convergence: Vanilla SGD can converge slowly, especially in the presence of sparse or noisy gradients.

Adam (Adaptive Moment Estimation):

- Pros:
    - Adaptive Learning Rates: Adam adapts the learning rates for each parameter individually, which can be beneficial in dealing with features that have different scales.
    - Momentum: Adam includes momentum to help accelerate convergence, making it more resilient to noisy gradients.
- Cons:
    - Memory Usage: Adam maintains additional moving averages, which can increase memory requirements.
    - Sensitivity to Hyperparameters: Adam's performance can be sensitive to the choice of hyperparameters, and finding the right combination is crucial.

Nadam (Nesterov Adam):

- Pros:
    - Adaptive Learning Rates: Similar to Adam, Nadam incorporates adaptive learning rates.
    - Nesterov Momentum: Nadam combines Nesterov momentum with adaptive learning rates, potentially providing faster convergence and better handling of oscillations.
- Cons:
    - Computational Cost: Nadam can be computationally more expensive than simpler optimizers like SGD.
    - Hyperparameter Sensitivity: Like Adam, Nadam's performance may be sensitive to hyperparameter choices.

Considerations for Choosing:

- For simplicity and computational efficiency, SGD might be suitable for smaller datasets and simpler models.
- Adam is often a good default choice due to its adaptive learning rates and momentum, making it robust across a variety of scenarios.

- Nadam can be advantageous when faster convergence is desired, and the dataset is large or the model architecture is complex.

In practice, it's common to experiment with multiple optimizers and hyperparameter settings to determine the best combination for a specific task, as we did for the in-class Kaggle competition. In our case, Nadam provided the best scores when implementing it to this particular project.