Support Vector Machine (SVM) Algorithm Implementation Report:

Rahul Jha | rahuljha@umd.edu | University of Maryland College Park

Accomplishments Description

Implementing and improving the Structured Support Vector Machine (SVM) loss function for multi-class classification was the aim of this study. The main accomplishments include Vectorized computation of SVM loss, Gradient computation utilizing vectorization and Optimization via Stochastic Gradient Descent (SGD).

Key tasks:

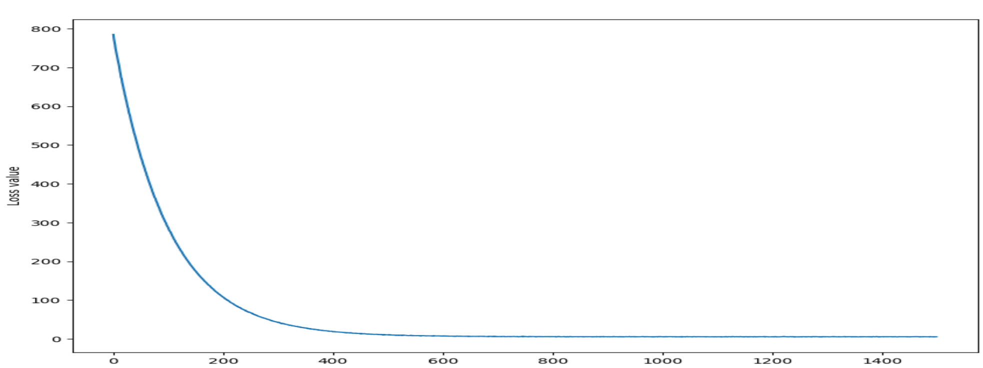
- **SVM Loss and Gradient:** A fully vectorized SVM loss function and its matching gradient have been built successfully. Training time was shortened by the vectorized implementation's notable increase in computing performance. The gradient and naive loss were computed in 0.117929 seconds. Computing vectorized loss and gradient took 0.009809 seconds.
- **Hyperparameter Tuning:** Carried out a thorough grid search to determine the ideal regularization strength and learning rate. A regularization strength of 2.5e4 and a learning rate of 5e-5 were determined to be the optimal hyperparameters.
- **Model Training:** Stochastic gradient descent (SGD) was used to train the SVM model until convergence. The lowering loss function over iterations shows that the model effectively converged.

Missed Points:

- Even though the SVM model performed reasonably well, accuracy may be increased by investigating feature engineering and kernel techniques further.
- Decision boundaries can be greatly impacted by outliers in the data, which SVM models may be particularly sensitive to. Subpar performance can result from outliers' disproportionate influence on the model's learning process.

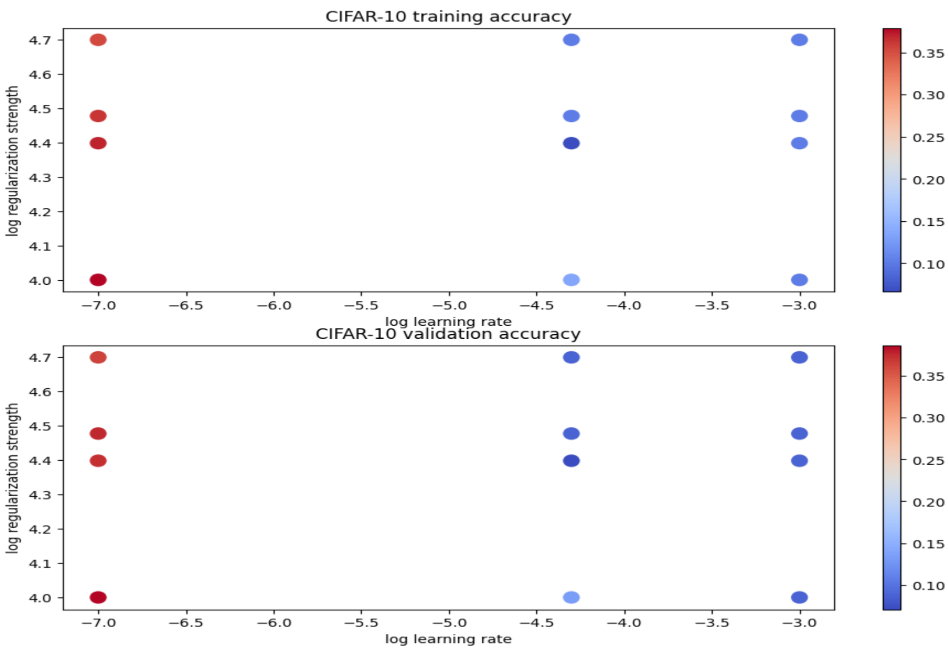
Explanation of Implementation Decisions

- **Vectorization:** To increase computing efficiency and prevent slow Python loops, vectorized operations were utilized. Making this decision was essential for managing big datasets and maximizing training duration.
- **Hyperparameter Tuning:** Grid search was chosen as a simple method to experiment with various hyperparameter combinations. To determine the ideal configuration, grid search made it possible to evaluate different learning rates and regularization strengths methodically.
- **Stochastic Gradient Descent (SGD):** SGD was selected because of its effectiveness and ease of use, particularly with big datasets. Using gradients computed on mini-batches of data, SGD iteratively adjusts the model weights, which makes it appropriate for large-scale learning applications.



Critical Thinking and Analysis

- Hyperparameter Influence: The model's performance was greatly affected by the regularization strength and learning rate selections.
- Achieving strong generalization and preventing overfitting or underfitting required a well-tuned hyperparameter configuration.
- Interpretation of SVM Weights: The features that the model deems significant for classification are revealed by the learned SVM weights.
- But understanding complex high-dimensional weights can be difficult, particularly when dealing with big feature spaces.



The Reason Behind These Outcomes:

1. **Data Quality:** The model's performance can be impacted by the training data's quality, which includes noise and feature relevance.
2. **Selecting the Wrong Hyperparameters:** Inadequate performance can result from using the wrong hyperparameters.
3. **Model Complexity:** For extremely non-linear situations in particular, the SVM's ability to capture complicated relationships in the data may be restricted.

Next Actions:

1. **Kernel Techniques:** Examine how to manage non-linear relationships in the data by utilizing kernel functions.
2. **Group Techniques:** To improve accuracy and robustness, take into account combining multiple SVM models (e.g., using bagging or boosting).
3. **Visualization:** To have a better understanding of the learnt SVM weights and how they relate to the data, investigate more sophisticated visualization tools.

Supporting References

CS231n: Convolutional Neural Networks for Visual Recognition, Understanding Support Vector Machines, Gradient Descent Optimization Algorithms, Support Vector Machines for Pattern Classification, Scikit-learn Documentation