Predicting Future Performance of Convolutional Neural Networks in Early Training Stages

Yunfeng Xin, Chengzhe Xu, Hangyi Zhao

{yunfeng.xin, czxu, hyz0815}@stanford.edu

Stanford Computer Science

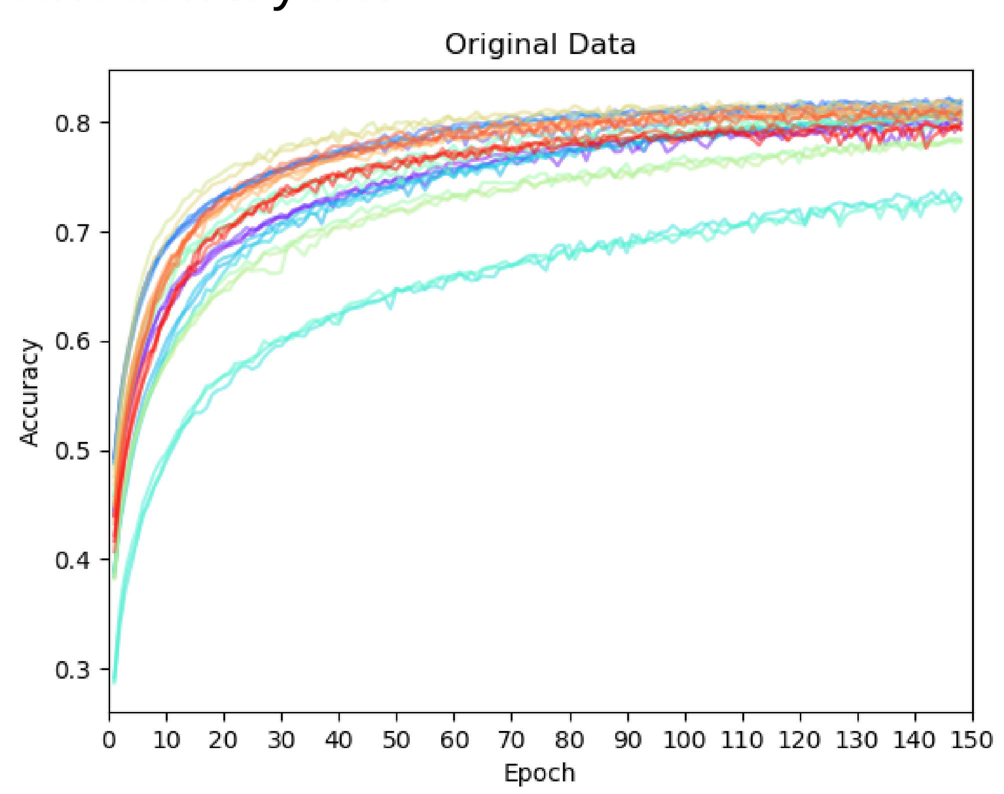
Motivation

The training process of Convolutional Neural Networks (CNN) can be lengthy. Being able to predict the performance of a certain set of hyperparameters on a specific CNN architecture can save both time and computation cost.

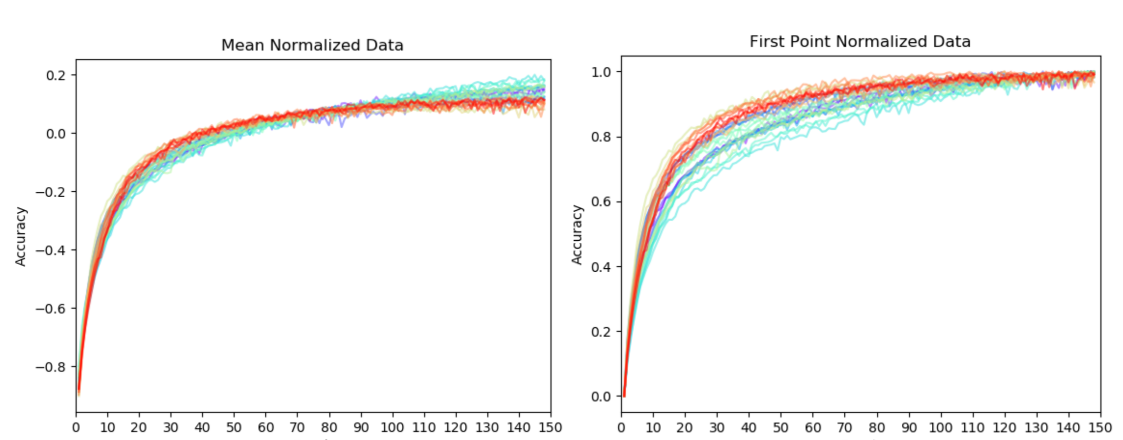
This project focuses on predicting the final accuracy of a CNN using only the validation accuracy of the initial 100 epochs. With our best model, we are able to achieve an average error of 0.38%.

Data

We collected validation accuracy curves from LeNet and Vgg-19 architecture with 11 different sets of hyperparameters and activation layers.



The data are normalized under two schemes:



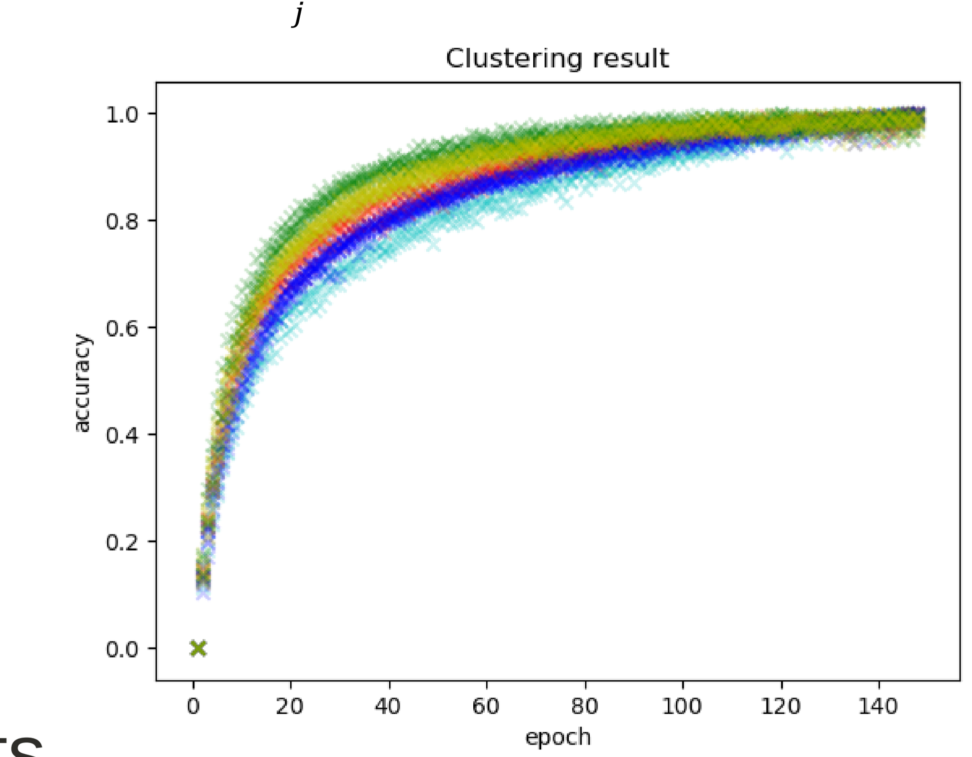
$$x' = \frac{x - mean(x)}{\max x - \min x} \qquad x' = \frac{x - x_1}{\max x - \min x}$$

The dataset consists of 34 curves and is split into 22/6/6 (train/val/test).

Models

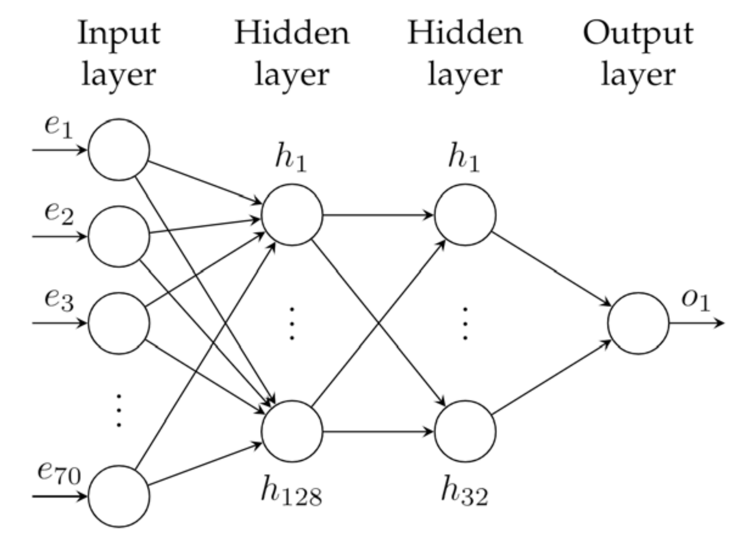
(K-)Nearest Neighbor:
- New input resembles known shapes in the training set.
- Loss for curve fitting: $\mathcal{L} = \sum_j |y_j - \hat{y}_j|$

Clustering:
- Time-series probabilistic distribution assumption.
- Loss: MSE = $\frac{1}{n}\sum_j (y_j - \hat{y}_j)^2$

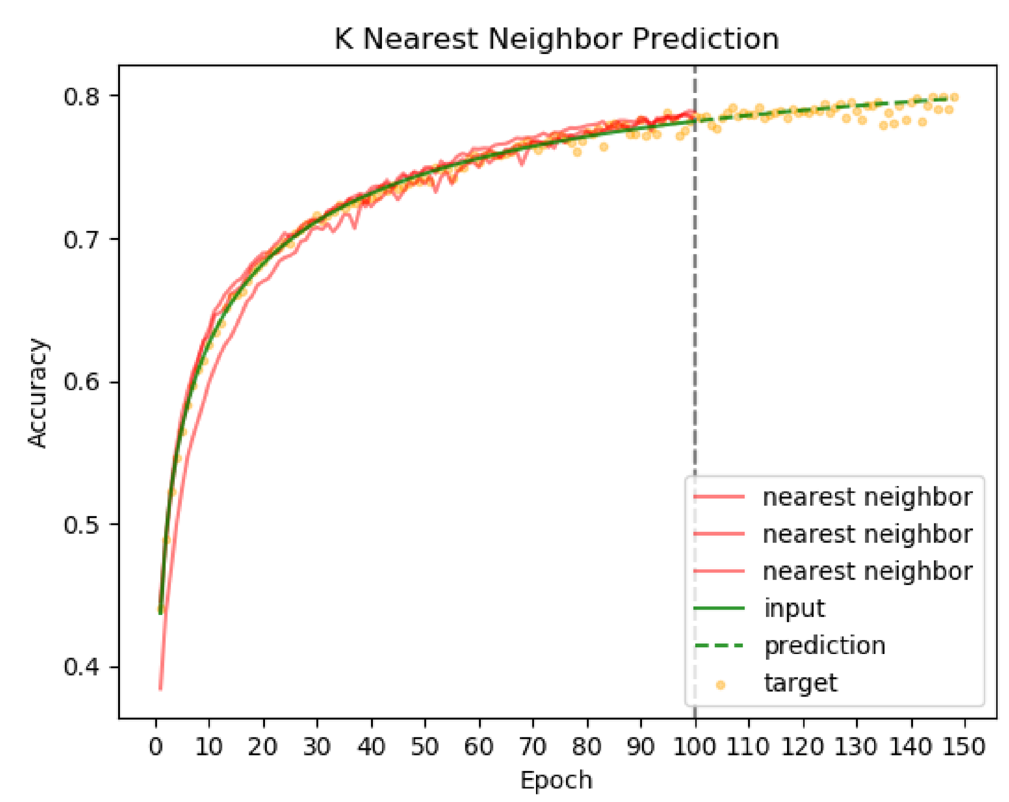


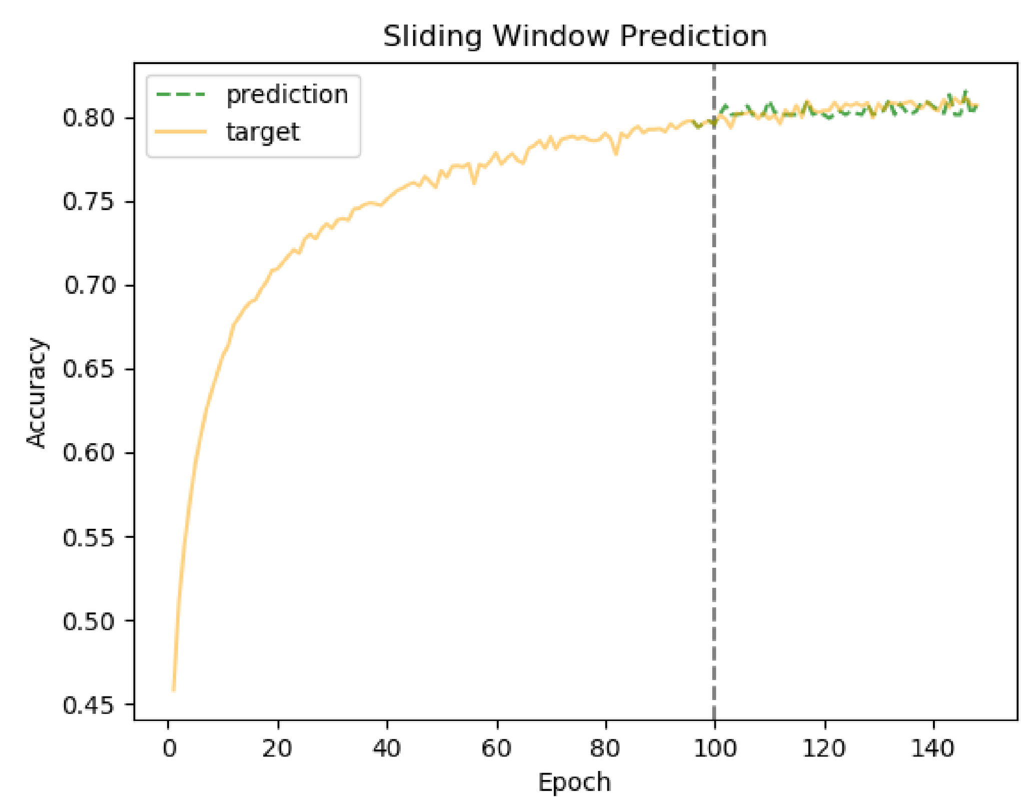
Markov Chain Monte Carlo:
- Time-series probabilistic distribution and first order Markov assumption $y_{\text{epoch}} \sim P_{epoch}(y_{epoch} | y_{epoch-1})$.
- Gaussian prior distribution and Gaussian white noises.
- Metropolis-Hastings sampling.

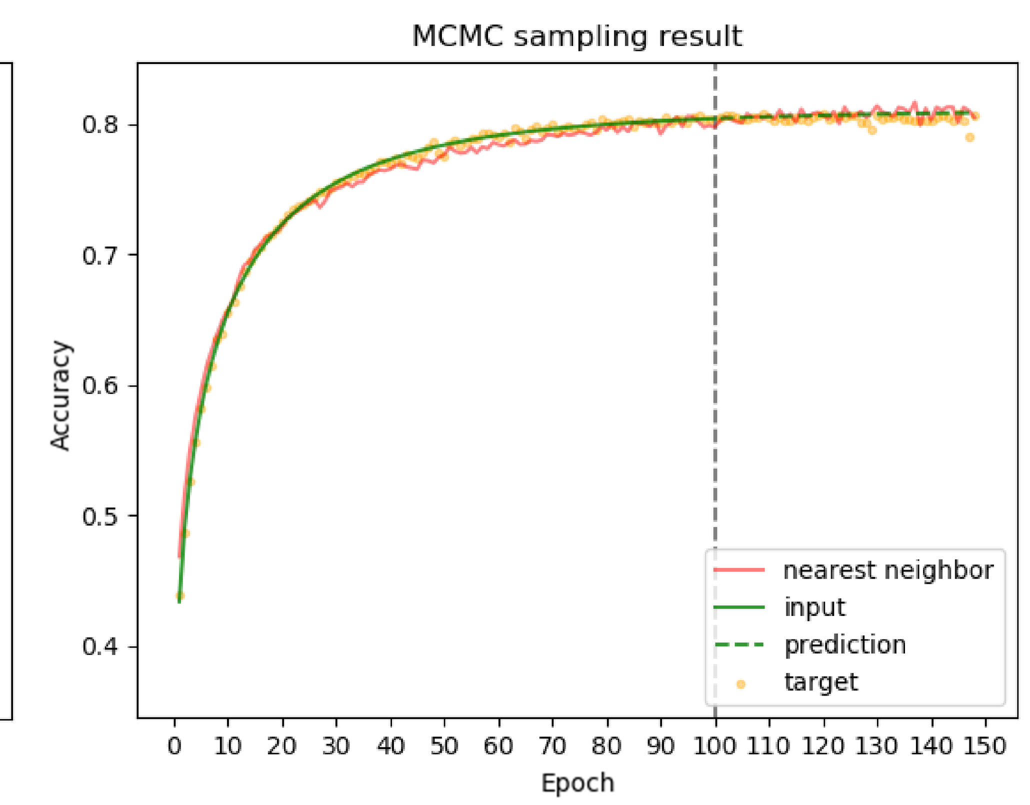
Sliding Window Prediction:
- Capture the pattern in the change rate of the curve.
- Sliding window approximation of the curve.
- Three-layer neural network for capturing pattern with early stopping and dropout.



Results





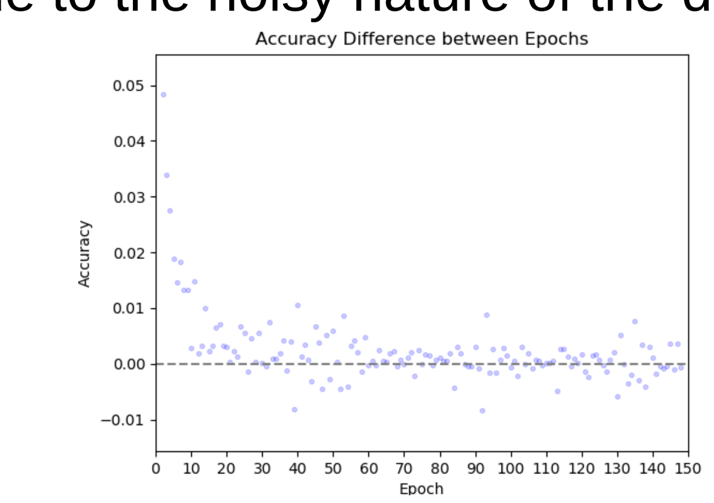


Algorithm	Min Val. Error	Max Val. Error	Avg. Val. Error	Avg. Test Error
Direct Curve Fitting	0.1761%	0.7523%	0.6331%	0.5264%
(K-)Nearest Neighbor	0.2250%	0.6753%	0.4276%	0.3824%
Clustering-6	0.1176%	1.8678%	0.8139%	0.7335%
Markov Chain Monte Carlo	0.1466%	0.7043%	0.3853%	0.3825%
Sliding Window Prediction	0.3218%	0.8258%	0.5953%	0.5967%

Analysis

Normalization:
- Helps generalize the curve shape.
- Both schemes achieve approximately the same level of performance.

Model Performance:
- Direct curve fitting can often fail to predict the decay rate of the curve in later epochs using the prior epochs.
- MCMC achieves the best performance as it takes time-series analysis into account, but can be unstable due to sampling.
- Nearest Neighbor achieves good performance by using generative methods to account for uncertainty. It takes significantly less computation than MCMC (2 secs vs. 45 mins).
- Sliding window prediction does not capture the increase rate as expected due to the noisy nature of the data.



Future Work

- Generate data using more advanced architectures (ResNet/MobileNetV2).
- Use 1-D convolution / averaging to capture the trend of the curve.

References

[1] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," J. Mach. Learn. Res., vol. 13, pp. 281–305, Feb. 2012.[Online].
[2] K. Eggensperger, "Towards an empirical foundation for assessing bayesian optimization of hyperparameters," 2013.
[3] P. Kolachina, N. Cancedda, M. Dymetman, and S. Venkatapathy, "Prediction of learning curves in machine translation," in Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 22–30. [Online]. [4] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in Proceedings of the 24th International Conference on Artificial Intelligence, ser.IJCAI'15. AAAI Press, 2015, pp. 3460–3468. [Online].
[5] A. Klein, S. Falkner, J. T. Springenberg, and F. Hutter, "Learning curve prediction with Bayesian neural networks," in ICLR , 2017.