Australasian Transport Research Forum 2023 Proceedings
29 November – 1 December, Perth, Australia
Publication website: http://www.atrf.info

Warm-starting the rolling horizon-based traffic volume prediction using neural networks

Zay Maung Maung Aye, Saeed Asadi Bagloee, Majid Sarvi, Mobin Yazdani

Transport Engineering Group

Faculty of Engineering and Information Technology

The University of Melbourne

Victoria 3010, Australia

zay.aye@unimelb.edu.au

Abstract

Many real-world deployments of machine learning systems train the models in an online manner as new data is collected. These systems often need to manage a large number of models automatically and from the practical perspective, it is not realistic for the models to train from scratch with all the past data as it would make the memory, computation time, and resources unbounded. Therefore, warm starting the previously trained model to update on the recent training data is often used in practice. We have identified a serious problem related to warm-starting parameters in continual learning that could impact the downstream learning process. In this paper, we analyze practical ways to mitigate this issue that are suitable for an automated learning environment.

Keywords: traffic prediction, practical machine learning, warm-starting neural networks

1. Introduction

Machine learning algorithms often assume the availability of a large static dataset for training. In real-world applications, however, the data is collected in an ongoing manner and the concept shift of the dataset might occur over the time (Sarker, 2021). Hence, continual update of the machine learning model is a must. Nevertheless, the ceaseless arrival of new data can pose a serious problem in terms of computational effort as the model needs to be trained from scratch (Schwartz et al., 2020).

Warm starting or parameter reusing is an approach where the parameters of the trained model in the previous evaluation are used as the starting point for the next update cycle. That means the training is only based on feeding with the most recent data rather than training with the whole dataset from scratch. Intuitively, re-training the model from scratch seems to be wasteful of memory and time when it comes to dealing with computational complexities. Accordingly, warm-starting parameters may be a solution to tackle this issue.

The concept of warm starting has been successfully applied in transfer learning (Bengio, 2012), semi-supervised learning (Erhan et al., 2010), online learning (Käding et al., 2016), and fine-tuning the pre-trained models (Li et al., 2019). However, (Ash and Adams, 2020) examined the ineffectiveness of warm starting the parameters in the domain of deep neural networks. The

authors reported that supervised pre-training of deep neural network not only does not show a reasonable performance but also worsen the results compared to the cold started model. The issue arises from the fact that training the pre-trained model with similar data would potentially bias the learning and cause the model to fall in local optima. Hence, relying on the original parameters of the pre-trained model may not necessarily ensure the best starting point for warm starting a deep neural network. To this end, the authors proposed a trick called shrink and perturb to adjust the parameters of the pre-trained deep neural network to better function in warm-starting.

In this paper, we report a practically important problem associated with warm starting the parameters in online/continual learning. For most real-life machine learning systems, the learning processes are automated, and potentially many models are updated regularly as new data is received. It may be possible for a model to train poorly at an update cycle due to some circumstances such as extreme weather conditions, sensor faults in the data, missing data, or outliers. The automated learning process may keep warm-starting with the parameters from a bad model without realizing or mitigating the problem and therefore compounding the prediction errors. And even after the bad data is shifted out of the training data window after some time has passed, these badly warm-started models may fail to recover quickly even with the normal data. This makes it hard to debug the learning process as the current training data seem normal while the predictions are still bad.

Our contributions are as follows: firstly, we show that there are merits in warm starting online regression models with a rolling window of training data. This setting is different from the setting used in (Ash and Adams, 2020), as only a fixed amount of data is available at any given time and this setting is more suitable for practical applications as the resource utilization and cost should not grow with time. We then show that warm-starting can fail catastrophically in some cases, and it will be very hard to recover once this happens. Finally, we show how the shrink and perturb trick proposed by (Ash and Adams, 2020) can be used to mitigate these failed cases and recover the models.

2. Literature review

Different approaches to warm-start neural networks have been reported in literature. An evolutionary approach based on evolving deep architectures is proposed to warm start the deep neural networks by (Tirumala et al., 2016). Warm-starting deep neural networks using with decision tree is proposed by (Humbird et al., 2018). (Xu et al., 2015) warm starts the multi-loss deep neural network using the parameters from the model trained with single-loss function, in order to speed up the convergence. All these approaches warm start the main model to either improve accuracy or reduce the training time. This is different from our model as we are training in a continuous learning environment and the next model is warm started from the current model.

Some practical issues related to warm-start have been reported in literature. (Senior et al., 2013) reported that increasing the learning rate is useful with random initialization but hurts warm-starting. Stochastic gradient decent with warm restarts (Loshchilov and Hutter, 2016) introduces the idea of resetting the learning rates decreasing it for a number of training epochs. They reported that restarting needs significantly fewer epochs to reach the same or better accuracy than the standard procedure of decaying learning rate. These issues are different from the issues we are discussing in this paper as we are focusing on extreme cases when warm-starting fails completely.

3. Experimental settings

The evaluations reported in this paper are conducted based on the following settings of regression model and data. All the evaluations are run 100 times to get a good estimate of the result unless a different number of runs is explicitly stated.

3.1. Data

The dataset contains the vehicle counts detected by a road camera. The measurements are aggregated to obtain the total vehicle counts for 15-minute intervals throughout the day, resulting in 96 counts for a road segment per day. This data can be considered as a time-series dataset.

3.2. Regression model

In this paper, the effect of warm started parameters for a neural network is evaluated in an online rolling window-based regression model. The simple regression task of predicting the next eight values using eight recent observed values is used to evaluate the effect of utilizing warm start in continual learning. The function is $f(x_{t-7}, x_{t-6}, \dots, x_t) \rightarrow x_{t+1}, \dots, x_{t+8}$, where x_t is the vehicle count at time t. To train this regression model in an online manner, the most recent 10 weekdays of data is used on a rolling basis daily. We designed a 2-layer neural network with 5 neurons in each layer with rectified linear activation function. The purpose of using a small neural network to prevent overfitting. For training the neural network, we tested L-BFGS and Adam optimizers. As L-BFGS achieves better accuracy with a good warm start for our experimental settings, the evaluation results will be reported using it unless explicitly stated as Adam optimizer. We split 80% of the data into the training set and 20% to validation set. For reporting the error, the root means square error is calculated on the validation set.

Random Initiation: randomly initialized parameters are generated using the method proposed by (Glorot and Bengio, 2010).

Root Mean Squared Error (RMSE): RMSE is the standard deviation of the prediction errors.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(A_i - P_i)^2}{N}}$$

where A is the observed value and P is the predicted value and N is the number of samples.

4. Is warm start better than training from random initialization for continuous learning?

In this section, we show that for our regression problem, warm starting with good parameters is helpful. In the evaluations, we trained the models 100 times (i.e., 100 random seeds and reshuffle data) and reported the validation errors. The intuition is that, if the parameters of the previously trained model are good then the next model is trained on warm started parameters is better than starting the fresh initialization with random parameters.

In Table 1, the results of warm started training the model on the second day based on the best model selected on the first day is compared against restarting the model training with randomly initialized parameters. For both optimizers, warm starting models provide better accuracy.

Table 1. Validation error comparison of warm-started vs randomly initialized parameters.

		Mean	**Stdv**	**Min**	**Max**
L-BFGS	Warm Start	41.69	0.13	41.48	42.06
	Random Init	54.32	18.13	42.81	107.91
Adam	Warm Start	43.68	0.17	43.30	44.07
	Random Init	50.43	15.79	43.84	205.77

Figure 1. Good warm started parameters (L-BFGS). Mean: 41.69, Stdv: 0.13, Min: 41.48, Max: 42.06

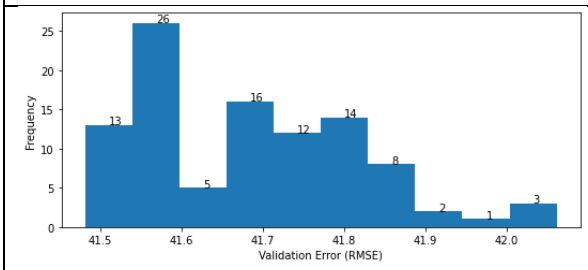


Figure 2. Good warm started parameters (Adam). Mean: 43.68, Stdv: 0.17, Min: 43.30, Max: 44.07

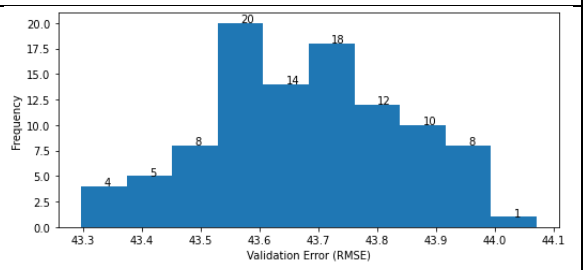


Figure 3. Distribution of validation errors based on different random initializations. Mean: 58.94, Stdv: 19.79, Min: 39.09, Max: 111.31 (L-BFGS optimizer).

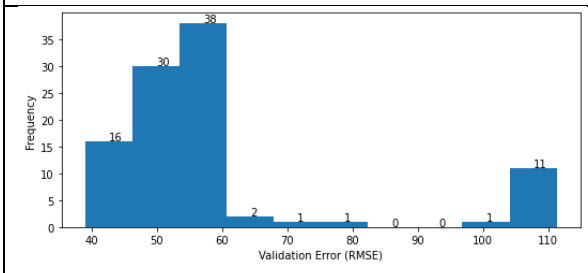
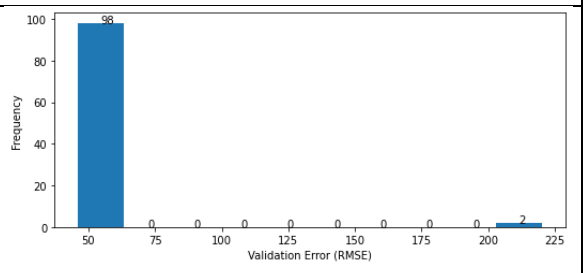


Figure 4. Distribution of validation errors based on different random initializations. Mean: 56.97, Stdv: 23.43, Min: 45.97, Max: 220.39 (ADAM optimizer)



In Figure 1 and Figure 2 for both L-BFGS and Adam optimizers, when training with good warm-started parameters the variation of validation errors between different runs is very small and centered around a small error. For comparison in Figure 3 and Figure 4 when training with randomly initialized parameters, the variation of errors can be very large and there is a higher chance that a suboptimal or bad model is selected.

These results suggest that there are incentives for using warm started parameters in continual learning over randomly initialized parameters for achieving better accuracy. It has been shown

in various literature and practical applications that warm starting converges faster than training from scratch, and therefore, we are not focusing on the running time in this paper.

5. Warm start cascading failure problem

In the previous section, we reported that a warm start is better than random initialization when the past model is trained well. However, as the error distribution of the first model train with randomly initialized parameters can be very large, a bad model can potentially be selected during an evaluation/training stage. In this section, we analyze the impact of selecting such cases for later evaluations in continual learning.

Figure 3 and Figure 4 show the distributions of validation errors (RMSE) resulting from 100 random initialization on the first day. As we can see that although the training of randomly initialized models converges to low error most of the time, there are some extreme cases where the models are getting stuck at a bad model with a very high validation error. If one of these bad models is obtained while training in an online learning manner, which in this case is a daily update, it will be hard for the model to recover in a short time with subsequent updates.

Another issue is that although these bad models are obtained rarely when the data quality is good, this distribution of model performance could change if the data quality is poor. And in many practical real-world applications (e.g., road traffic data), the quality of data can be varied because of weather conditions or sensor faults and the quality of the data can be hard to guarantee because the data could be streaming from different data providers. An automated machine learning system may be training many models, these cases might be happening in the system, without being reported or mitigated.

In comparison with Figure 3 when the data is normal, Figure 5 shows that the error distribution changes drastically when 5% of the data is missing and Figure 6 shows the changes in error distribution when 1% of data become outliers. For the evaluation of missing data, if one of the values is missing either in the inputs or the labels, these cases are excluded from the training data. For the evaluation of outliers, 0.05% of the data is increased by 5 times, and another 0.05% is decreased by 5 times, to add outliers randomly in the training data.

Figure 5. Changes in error distribution due to data issues (Missing Data) (L-BFGS: 5% missing value, resulting in only 42.2% of original features usable).

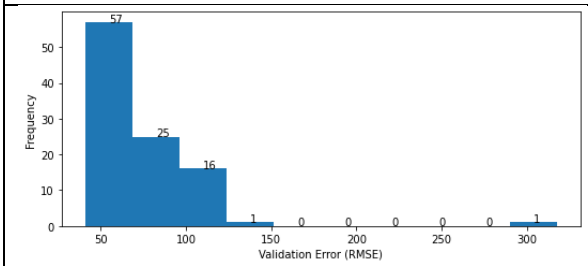
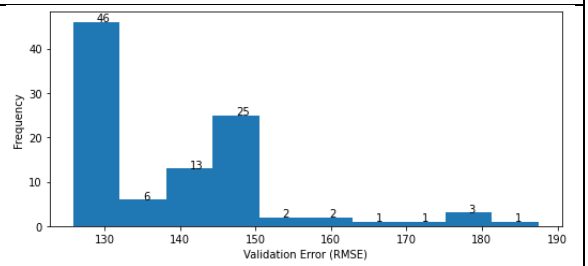


Figure 6. Changes in error distribution due to data issues (Outliers) (L-BFGS: 1% outliers added).



5.1. Can this problem be solved automatically as new data is received and old data is discarded?

In the previous sections, we showed that with randomly initialized parameters, some of the models end up with very high validation errors by chance. These cases will be considered as bad initial model to warm start (as we will be starting from a highly inaccurate model). If the subsequent model is updated using the parameters from these bad models, they may fail to improve accuracy.

In this section, we evaluated the effect of bad warm-started models on later update cycles in a continuous learning process. Figure 7 compares the effect of good and bad warm started parameters on model updates in the later training cycles. In this experiment, only the parameters of the initial model (the starting parameters on the first day) are different, and the training procedure and data are the same. The training data window size is 10 weekdays and as the data for the new day is added to the rolling window, the oldest day's data is removed. We can see that it can take a long time to recover from a badly trained model by just relying on the new training data. This would mean that the model will make bad predictions for a long time and will impact the overall application significantly.

Figure 7. Comparison of warm-started continuous learning from a good and bad initial model (L-BFGS)

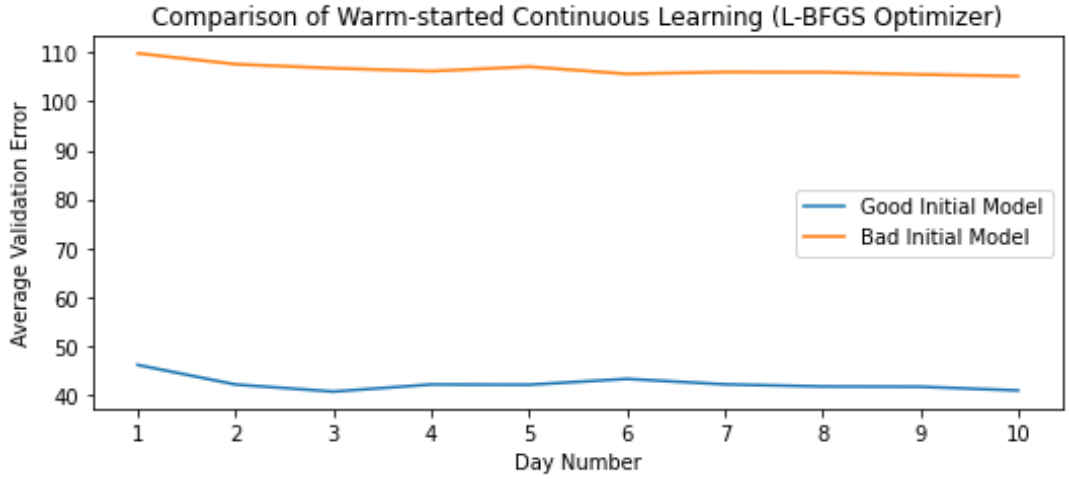
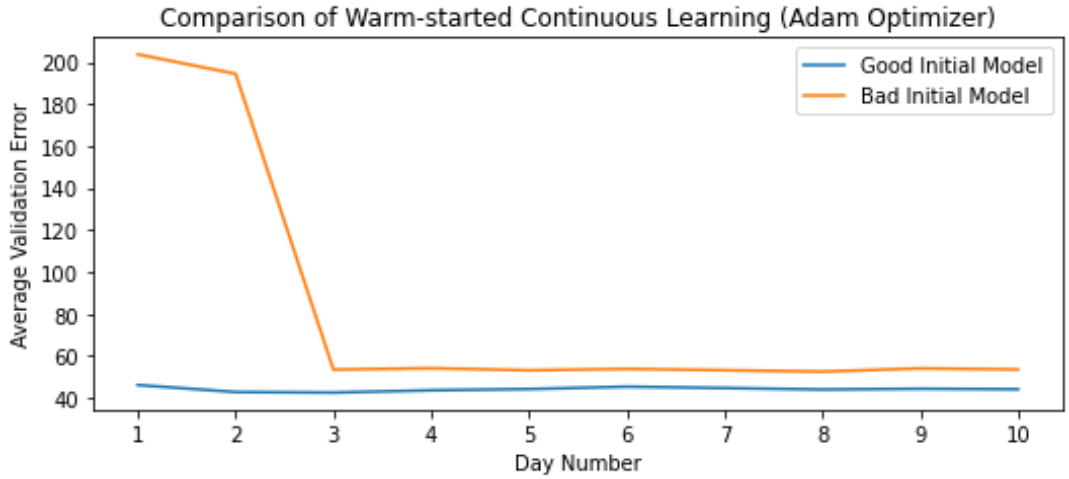


Figure 8. Comparison of warm-started continuous learning from a good and bad initial model (Adam)



However, in some cases, a bad model may be able to recover after some days without intervening as in Figure 8. However, there is no guarantee on whether they will recover, or the number of days required to recover the model performance. A bad model may still fail to recover even after training with new data. Therefore, we should not rely just on more training using new data in such cases and should explore ways to mitigate this problem.

6. Mitigating the bad warm start problem

As it can take very long for the model to recover from a bad warm start if no intervention is provided, we will need to consider several options when a bad model is learned. One approach is to introduce some type of perturbation so that the next learning cycle can be escaped from a saddle point or a bad local minimum of the current model. Another approach is to restart the learning process using a random initiation.

6.1. Shrink and perturb trick revisited

(Ash and Adams, 2020) proposed the shrink and perturb trick to improve the training with a warm start on neural networks. Their focus is for making the accuracy of warm-started training model closer to a model trained with randomly initialized parameters from scratch, while keeping the training time faster.

The idea is to reduce the weights toward Zero by a ratio λ (i.e., shrinking step) and to add some random perturbation noise p (i.e., perturbation step). The formulation of the shrink and perturb trick is as follow: $\theta_i^t \leftarrow \lambda\theta_i^{t-1} + p_i^t$, where $p_i^t \sim N(0, \sigma^2)$ and $0 < \lambda < 1$, where θ_i^t is the weight of a neural network after evaluation t and λ is the shrink ratio and p^t is the perturbation noise.

They evaluated this trick on classification problems and found that shrinking the weight is important to reduce the influence of past training data and make the model relearn again after past training data and new training data are combined.

The purpose of shrinking the weights toward zero is to reduce the influence of the past model and according to this assumption if the previous model is badly trained, the shrink ratio should be larger. In machine learning, perturbation is often used as a solution to mitigate getting stuck at a saddle point (Du et al., 2017, Elidan et al., 2002, Neelakantan et al., 2015). In our evaluations with continuous learning of regression model, we found new insights in utilizing shrink and perturb trick.

6.2. Recovering a bad model with shrink and perturb trick

Figure 7 demonstrates that it is very difficult to recover a bad model in warm-started continuous learning without applying a mitigation procedure. In some cases, a bad model may be able to recover without intervention as new training data is added to the training data. We can see it happening with Adam optimizer in Figure 8 after updating the model daily for 3 days. However, as there is no guarantee on whether the model can recover or the time a take to recover, an intervention is needed when a bad model is trained, to mitigate this problem in an automated manner.

So, in this section, we examine the mitigation options: 1. Shrink and perturb-based approach 2. Using random initiation and 3. Warm start only with no mitigation. The results of applying

different settings of shrink (λ) and perturb (σ) to a bad warm-started parameter are reported in Table 2 and Table 3. The accuracy is very sensitive to their setting with high error variances, making it hard to do efficient parameter search. However overall patterns we found for recovering from the bad initial parameters are shrinking them is generally helpful and small perturbations can improve the models while large perturbation tends to produce bad results. Just continuing with bad warm-started parameters without any intervening can make the model get stuck for both L-BFGS and Adam optimizers. For recovering a bad model, randomly initialized parameters are almost as good as the best shrink and perturb setting.

Table 2. Recovering a bad model for L-BFGS Optimizer (λ is the shrinking ratio and σ is the standard deviation of the perturbation noise)

	Mean	**Stdv**	**Min**	**Max**
λ=0.01,σ=0.01	87.70	27.51	50.20	107.89
λ=0.01,σ=0.1	52.16	15.53	42.41	107.89
λ=0.01,σ=0.5	**52.23**	**14.95**	**42.12**	**107.89**
λ=0.01,σ=1	68.61	57.55	43.49	588.10
λ=0.1,σ=0.01	106.55	8.15	50.21	107.89
λ=0.1,σ=0.1	84.87	27.86	42.97	107.89
λ=0.1,σ=0.5	61.93	30.20	41.94	258.45
λ=0.1,σ=1	62.79	23.64	43.41	107.89
λ=0.5,σ=0.01	107.89	0	107.89	107.89
λ=0.5,σ=0.1	106.16	6.72	50.21	107.89
λ=0.5,σ=0.5	87.50	26.79	43.42	107.89
λ=0.5,σ=1	82.34	27.81	46.09	108.54
λ=1,σ=0.01	107.89	0	107.89	107.89
λ=1,σ=0.1	107.42	2.31	94.53	108.29
λ=1,σ=0.5	98.77	18.66	50.03	107.92
λ=1,σ=1	97.79	19.87	41.87	107.89
λ=1,σ=0 (warm start only)	107.89	0	107.89	107.89
Randomly initialized	**53.90**	**18.17**	**42.81**	**107.89**

Table 3. Recovering a bad model for Adam Optimizer (λ is the shrinking ratio and σ is the standard deviation of the perturbation noise)

	Mean	**Stdv**	**Min**	**Max**
λ=0.01,σ=0.01	52.58	10.21	45.70	123.50
λ=0.01,σ=0.1	**48.57**	**2.04**	**44.61**	**52.52**
λ=0.01,σ=0.5	107.45	72.73	42.81	344.47
λ=0.01,σ=1	529.59	578.38	45.08	3302.39
λ=0.1,σ=0.01	70.36	50.39	45.30	200.67
λ=0.1,σ=0.1	51.94	21.35	44.32	200.67
λ=0.1,σ=0.5	108.48	70.08	43.32	278.57
λ=0.1,σ=1	563.79	589.36	45.23	2564.34
λ=0.5,σ=0.01	95.50	68.86	47.73	200.67
λ=0.5,σ=0.1	79.25	60.74	44.73	200.67
λ=0.5,σ=0.5	127.18	76.88	43.71	350.29
λ=0.5,σ=1	577.03	580.08	48.22	3093.70
λ=1,σ=0.01	110.46	73.67	47.54	200.67
λ=1,σ=0.1	83.00	62.52	43.62	200.68
λ=1,σ=0.5	164.39	70.86	47.00	401.31
λ=1,σ=1	643.63	591.51	48.92	3556.78
λ=1,σ=0 (warm start only)	200.67	0	200.67	200.67
Randomly initialized	52.15	22.05	43.35	205.82

Figure 9 and Figure 10 illustrate the effect of recovering bad initial parameters on the later continual learning process. For these figures, the starting point is that a bad model is obtained just before day one, and on day 1 some intervention is done to improve the model for later days. For the shrink and perturb approach λ=0.1 and σ=0.1 are used as a good option on the first day to recover the model from getting stuck and once an intervention is completed, a warm start is used without shrink and perturb for the future updates. And for random initializations, there are two options: 1) randomly initialized parameters on day 1 as an intervention from the previous bad model and then warm start once the model is improving, and 2) always used randomly initialized parameters without warm start.

Random initialization is a better option for L-BFGS and shrinks and perturbs achieve slightly better accuracy for Adam optimizer, although the performance difference between them is relatively small. Both approaches have stable accuracy on the later days and the models improve slightly as the number of training days increases.

Figure 9. Effect of perturbing the weights on the continual learning process (L-BFGS)

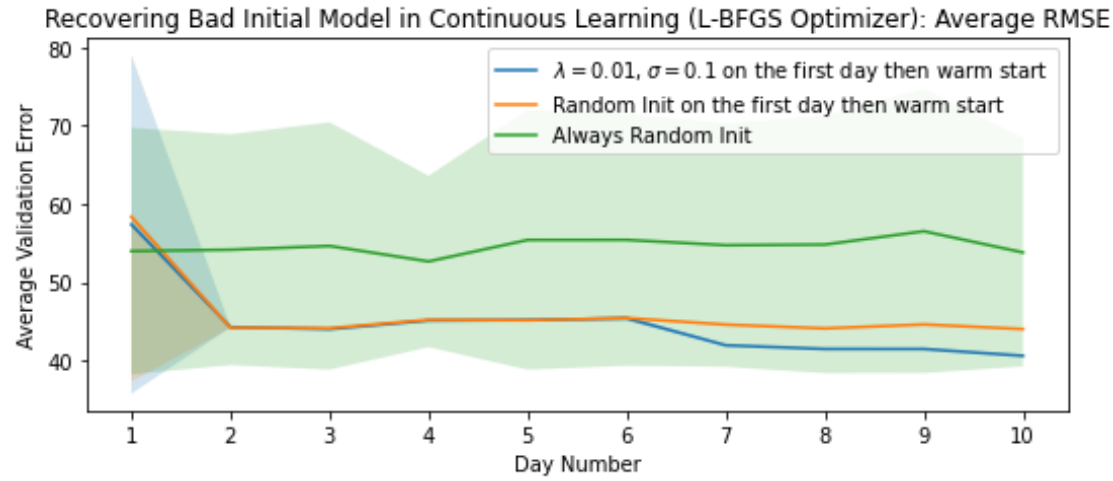
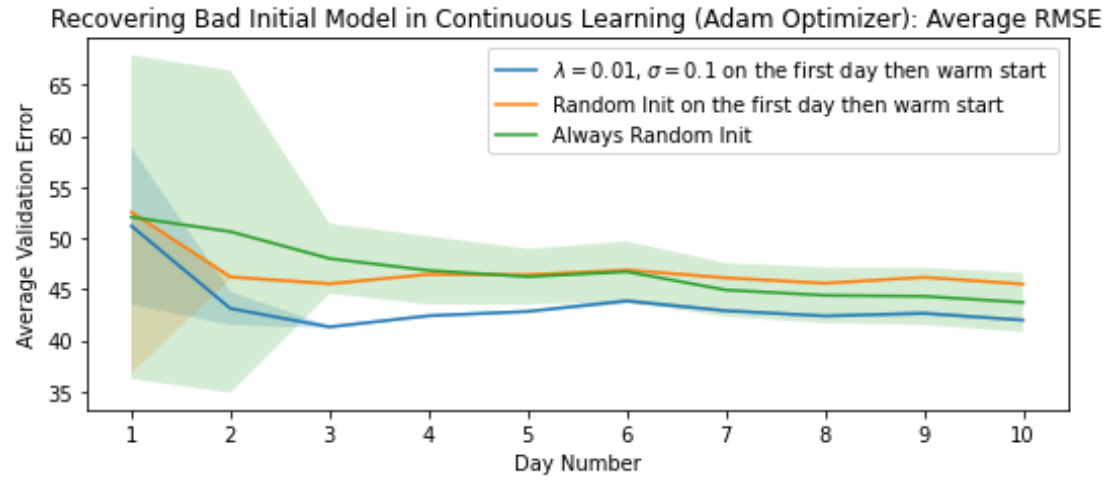


Figure 10. Effect of perturbing the weights on the continual learning process (Adam)



6.3. What happens if the shrink and perturb trick is applied to a good initial model?

We also evaluated the effect of shrinking and perturbing the good warm-started weights. In contrary to the findings of (Ash and Adams, 2020), applying shrink and perturb tricks to good

warm started parameters will hurt the model when training rolling window-based online regression models. In Table 4, smaller λ or larger σ reduces the model accuracy. This is because the past parameters are already well trained and small λ reduces the importance of these parameters and larger σ adds more perturbation on these parameters.

Table 4. Evaluation of applying shrink and perturb to good warm start parameters.

	Mean	**Stdv**	**Min**	**Max**
λ=0.01,σ=0.01	45.49	2.73	42.61	60.91
λ=0.01,σ=0.1	49.31	13.76	41.56	107.89
λ=0.01,σ=0.5	55.09	19.82	41.74	108.02
λ=0.01,σ=1	62.66	24.39	42.35	107.89
λ=0.1,σ=0.01	44.73	1.69	41.88	53.05
λ=0.1,σ=0.1	45.04	2.20	41.80	51.33
λ=0.1,σ=0.5	50.40	13.49	41.83	107.89
λ=0.1,σ=1	159.74	961.15	43.46	9719.60
λ=0.5,σ=0.01	46.41	1.36	41.64	46.97
λ=0.5,σ=0.1	44.77	3.40	41.27	60.91
λ=0.5,σ=0.5	49.11	12.15	41.70	107.89
λ=0.5,σ=1	57.90	20.97	42.26	107.89
λ=1,σ=0.01	41.82	0.33	41.45	43.59
λ=1,σ=0.1	42.90	1.52	41.54	50.84
λ=1,σ=0.5	47.17	7.0	41.80	107.89
λ=1,σ=1	60.08	39.20	41.84	399.16
λ=1,σ=0 (warm start only)	**41.69**	0.13	41.48	42.06
Randomly initialized	53.90	18.17	42.81	107.89

7. Discussion

Warm starting the models in online learning is useful for accuracy and efficient training. This approach has been used in many practical machine learning systems, however, the practical issues related to it are rarely reported in the literature. We have reported one major problem with warm starting the models in automated online learning and examining its behavior for the continual model updates.

We explore literature and examine a recent approach called the shrink and perturb tricks to mitigate the problem. However, this approach is very sensitive to the parameters, we also explore another option of restarting the model with randomly initialized parameters when a model is getting stuck at bad results and fails to improve in the later learning steps. As the later approach is not sensitive to parameters settings, this approach is more suitable for practical automated machine learning applications. An optimal result might be achieved by combining both warm-starting and random restart. For example, as long as the model accuracy is high keep using warm-starting and once the errors are significantly high, use randomly initialized parameters to restart the bad models.

Future research directions of interest are a better theoretical understanding of warm-starting in continual learning and finding a way to proactively prevent the related problems in the first place.

8. References

ASH, J. & ADAMS, R. P. 2020. On warm-starting neural netswork training. *Advances in Neural Information Processing Systems,* 33**,** 3884-3894.

BENGIO, Y. Deep learning of representations for unsupervised and transfer learning. Proceedings of ICML workshop on unsupervised and transfer learning, 2012. JMLR Workshop and Conference Proceedings, 17-36.

DU, S. S., JIN, C., LEE, J. D., JORDAN, M. I., SINGH, A. & POCZOS, B. 2017. Gradient descent can take exponential time to escape saddle points. *Advances in neural information processing systems,* 30.

ELIDAN, G., NINIO, M., FRIEDMAN, N. & SCHUURMANS, D. 2002. Data perturbation for escaping local maxima in learning. *AAAI/IAAI,* 132**,** 139.

ERHAN, D., COURVILLE, A., BENGIO, Y. & VINCENT, P. Why does unsupervised pre-training help deep learning? Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010. JMLR Workshop and Conference Proceedings, 201-208.

GLOROT, X. & BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010. JMLR Workshop and Conference Proceedings, 249-256.

HUMBIRD, K. D., PETERSON, J. L., MCCLARREN, R. G. & SYSTEMS, L. 2018. Deep neural network initialization with decision trees. *IEEE transactions on neural networks,* 30**,** 1286-1295.

KÄDING, C., RODNER, E., FREYTAG, A. & DENZLER, J. Fine-tuning deep neural networks in continuous learning scenarios. Asian Conference on Computer Vision, 2016. Springer, 588-605.

LI, H., SINGH, B., NAJIBI, M., WU, Z. & DAVIS, L. S. 2019. An analysis of pre-training on object detection. *arXiv preprint arXiv:1904.05871*.

LOSHCHILOV, I. & HUTTER, F. J. A. P. A. 2016. SGDR: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.

NEELAKANTAN, A., VILNIS, L., LE, Q. V., SUTSKEVER, I., KAISER, L., KURACH, K. & MARTENS, J. 2015. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*.

SARKER, I. H. 2021. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science,* 2**,** 1-21.

SCHWARTZ, R., DODGE, J., SMITH, N. A. & ETZIONI, O. 2020. Green AI. *Communications of the ACM,* 63**,** 54-63.

SENIOR, A., HEIGOLD, G., RANZATO, M. A. & YANG, K. An empirical study of learning rates in deep neural networks for speech recognition. 2013 IEEE international conference on acoustics, speech and signal processing, 2013. IEEE, 6724-6728.

TIRUMALA, S. S., ALI, S. & RAMESH, C. P. Evolving deep neural networks: A new prospect. 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016. IEEE, 69-74.

XU, C., LU, C., LIANG, X., GAO, J., ZHENG, W., WANG, T., YAN, S. & TECHNOLOGY, S. F. V. 2015. Multi-loss regularized deep neural network. *IEEE Transactions on Circuits,* 26**,** 2273-2283.