Preventing Catastrophic Forgetting in an Online Learning Setting

Ayon Borthakur [1] **Matthew Einhorn** [1] **Nikhil Dhawan** [1]

Abstract

Catastrophic forgetting is an undesired phenomenon which prevents a neural network from being used for multiple tasks. We believe that online learning suffers from a similar issue as weights are updated based on most recent data. This can be particularly problematic for data-sets where the time-series data is cyclical in nature, e.g. weather data, and where the cyclic nature of the data is excluded as data feature. Using the NOAA weather data collected from 1981-2010, we first show as a baseline that simply training a multilayer perceptron(MLP) in an online fashion resulted in cyclic spikes in test error. Similar spikes were also observed when the model was re trained with random subsamples of previous data. Finally, we observed that an autoencoder with a novel weighted loss function for an additional regression layer for online learning outperformed the baseline MLP (t-test p value: 0.000027; Cohen's d: 0.81).

1. Introduction

In many real-world learning settings, agents must be able to adapt to different tasks while not necessarily having explicit knowledge of task changes (Legg & Hutter, 2007). However, when adapting to new tasks, agents must not "forget" how to solve previously encountered tasks. Neural networks unfortunately, have been shown to exhibit a phenomenon called "catastrophic forgetting" (McCloskey & Cohen, 1989) where neural networks trained sequentially on a set of different tasks are unable to perform well on previously encountered tasks. This occurs due to the way weights are updated in neural networks, namely when trained on task A, a neural network's weight is updated to minimize loss on task A. Then, when the neural network is trained on task B, the weights (some of which are critical for Task A) are updated to minimize loss for Task B (McCloskey & Cohen, 1989). Even modern neural networks trained using SGD and dropout experience catastrophic forgetting in multi-task settings (Goodfellow et al., 2013). While catastrophic forgetting has been observed in neural-networks used in multi-task settings, we assert that catastrophic forgetting is also a major problem in online-learning settings where data used to train a network that is cyclical in nature (i.e weather data). Online learning is commonly used on learning tasks where training batched methods is computationally infeasible or where the agent needs to adapt to new patterns over time (Ma et al., 2009). Typically, an online learning system receives data sequentially, makes a prediction on the data, and then incorporates the ground truth of the new data into its model for future classification. In this paper, we demonstrate the extent to which neural networks trained using online learning suffer from catastrophic forgetting and present methods to alleviate this phenomenon.

2. Related Work

Much of the work on catastrophic forgetting has been done in the multi-task setting. Rebuffi et al. proposed the use of separate memory modules while Robins proposed a memory free pseudo-rehearsal approach for avoiding catastrophic forgetting. Nguyen et al. implemented a generative model instead of a memory module. Velez & Clune proposed a localized, task specific formation of functional modules of nodes and connections for mitigating catastrophic forgetting.Serrà et al. recently proposed the learning of binary attention vectors concurrently with the task for constraining weight updates during learning of subsequent tasks. Kirkpatrick et al., Zenke et al. on the other hand slowed down learning of weights important for learning of previous tasks. Li et al. used Support Vector Machines (SVM) to identify the support data from the old data for learning subsequent data. Kemker et al. implemented a dual memory architecture for storing new and old memories where a third module decides the selection of appropriate memory module during inference. Prior work has shown some success in constraining the weights of machine learning algorithms during training. Ma et al. used a confidence-weighted (CW) algorithm when training an online algorithm to identify suspicious URLs. Rodrigues et al. observed that a neural network weather predictor performed better than a simple

[*]Equal contribution [1]Cornell University. Correspondence to: Ayon Borthakur <ab2535@cornell.edu>, Matthew Einhorn <me263@cornell.edu>, Nikhil Dhawan <nd353@cornell.edu>.

CS 6787: Advanced Machine Learning, Ithaca, NY, 2018.

linear regression model. To the best of our knowledge, none of these online learning architectures have been used for prediction of weather feature prediction.

3. Methods

We used the "U.S. Climatic Normals" dataset (Arguez et al., 2010) which provides 29 features from hourly weather readings, such as temperatures, wind, cloud cover, dew points, etc. from many weather stations across the USA for years 1981 - 2010. For this work, we used the average of the 30 years of data, which is equivalent to data from a single year. We chose to use a weather data set due to the inherent cyclic nature of weather patterns which would present a challenge to a standard online MLP which may forget how to predict weather in the summer when learning to predict weather in the winter. To investigate whether or not we could overcome catastrophic forgetting, we trained two baselines, a weight-constrained MLP, and an auto-encoder regression model to predict the mean temperature, 90th percentile temperature, and 10th percentile temperature at a given weather station from the other 26 features.

3.1. Baseline MLP

As a baseline, we first evaluated the performance of two standard multi-layer perceptions (MLPs) trained on the NOAA weather data which learned in a offline and online fashion respectively. We used a 2-hidden layer, 512 units, ReLU connected MLP with batch-normalization similar to the one used in binary connect (Serrà et al., 2018) due to the somewhat shared similarity in complexity of our problem and the MNIST problem solved in binary connect. Initially, we trained our baseline using the ADAM optimizer, however to maintain consistency with our other experiments that are incompatible, we switched to SGD with momentum (0.9) for all experiments. We also further evaluate the online baseline MLP using a re-sampling technique which periodically retrains the network on a subset of previously encountered data.

3.2. Weight Constrained Learning MLP

Taking inspiration from the CW algorithm used by Ma et al., we evaluated the performance of an MLP with the same structure as the baseline and optimized using SGD with momentum, however, we use a modified loss function to limit changes in the weights of the MLP. Similar to (Dredze et al., 2008), we theorized that by adding a penalty for changes to the parameters, weights relevant to previously seen examples will not be modified or forgotten. The loss functions for this technique, which we call weight-constrained learning (WCL), take two forms and are shown below in Eq. 1 and 2.

$$\mathcal{L}^*(\vec{w}_t) = \mathcal{L}(\vec{w}_t) + \frac{\lambda}{2}(\vec{w}_t - \vec{w}_{t-1})^2 \tag{1}$$

$$\mathcal{L}^*_{KL}(\vec{w}_t) = \mathcal{L}(\vec{w}_t) + \lambda \cdot KL(\vec{w}_t || \vec{w}_{t-1}) \tag{2}$$

where $\vec{w}_t$ is the weights of the MLP at time t, $\mathcal{L}(\vec{w}_t)$ is a standard mean-squared error (MSE) loss, and λ is the relative loss weighing.

3.3. Autoencoder

The principle behind a standard autoencoder (AE) is to be able to reconstruct input data in an unsupervised manner. Hence, we expected that an autoencoder would be more resilient to the learning of new data. In particular, if previous data is periodically resampled, we hypothesized that an AE would require less samples as it it may learn the sample space better than a standard MLP.

To test our hypothesis, we added a decoder structure to our baseline MLP converting it to a autoencoder. The final hidden layer of the encoder was connected to the regression layer that predicts the temperature. The overall network then contained two hidden layers for the encoder and decoder, each, as well as a regression layer and a final reconstruction layer. The loss function of the standard autoencoder was modified in order to the mean squared error for both the input reconstruction and wheather prediction regression (Eq.3).

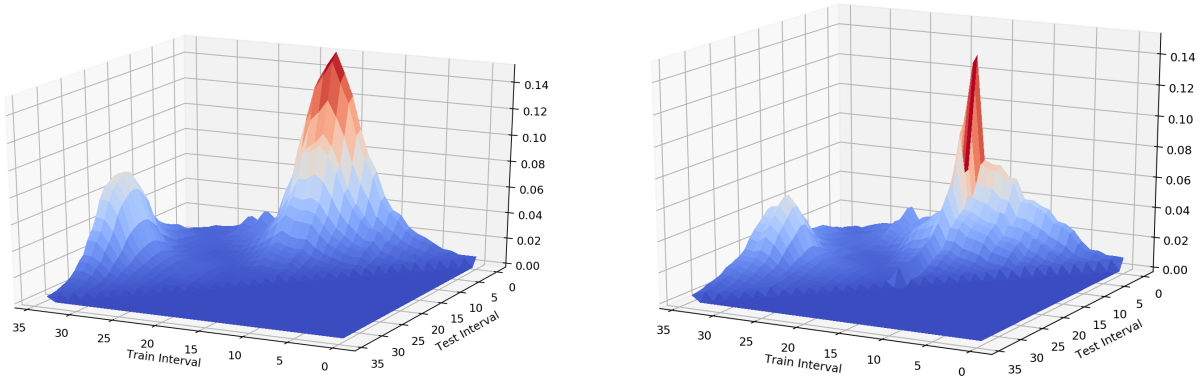
$$\mathcal{L}^*(\vec{w}_t) = \mathcal{MSE}_{\mathcal{R}}(\vec{w}_t) + \lambda \cdot \mathcal{MSE}_{\mathcal{P}}(\vec{w}_t) \tag{3}$$

where MSE_R is the mean squared error of reconstruction, MSE_P is the mean squared error of prediction, and λ is the relative loss weighing.

4. Experiments and Results

We trained our models on the NOAA weather data (Arguez et al., 2010). Unless otherwise specified, we normalized the inputs to our model on a per-feature basis by subtracting the mean and dividing by the standard deviation of the dataset. We split our dataset into train/val/test using a 80/10/10 split, resulting in approximately 2.15 million/270 thousand/270 thousand samples for each dataset, respectively. We removed the time and date features of the sample from all the samples to prevent the model from using the time to help predict because we wanted to test online models.

It is also important to note that all MSE values reported with the exception of those in Figure 1 are the *maximum MSE* over all intervals. We chose to report MSE this way since we are merely concerned with reducing the error on



(a) No Resampling (b) With Resampling

Figure 1. A baseline MLP trained in an online manner without (a) and with (b) re-sampling. The "train interval" axis represents the latest interval of data that the online MLP trained on. The "test interval" axis shows for a given "train interval" the test error across all intervals. E.g. the "test interval" slice when "train interval" is 15, shows the MSE for each interval in the test set after the model was trained on interval 15 of the train set. E.g. for train interval 15, we see a large loss for test interval 0 because interval 15 is in June and 0 is in January so the model seems to "forget" how to classify January after training on June.

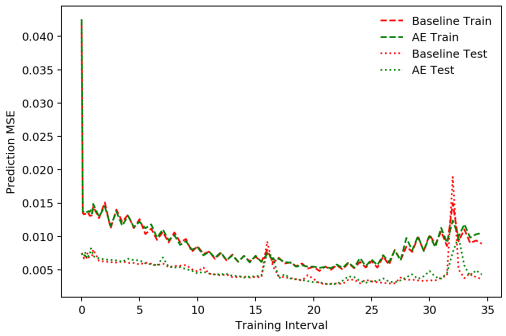


Figure 2. The training and testing loss for online baseline and autoencoder model

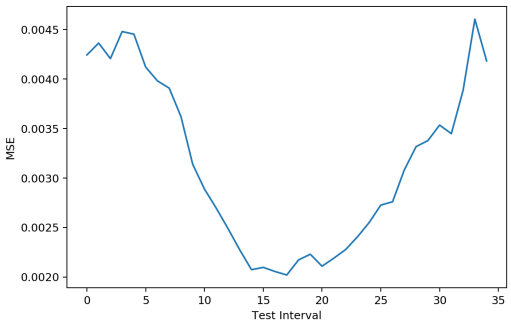


Figure 3. The training MSE validation set loss for the offline baseline MLP which was trained over the entire NOAA dataset. The test interval are the samples in the validation set that corresponds to that interval of time. E.g. interval 15 is around June because each interval is 10 days.

intervals that the model has "forgotten" how to handle. Thus, a reduction in the maximum MSE over all intervals will indicate an improvement in the model's "memory".

For the offline experiment, we trained on the full training set for 10 epochs. For all online experiments, we pre-trained the model by initially training it for 10 epochs on the first 30 days of the year. During the online phase, we trained 2 epochs on intervals of 10 days until the dataset was exhausted when we reached the end of December.

4.1. Baseline MLP

As mentioned in section 3.1, we trained two baseline MLPs with 2-hidden layers to minimize MSE loss. The first MLP was trained in a standard offline manner using the full training data-set and the second was trained in an online manner where data was provided to the model sequentially as described above. As illustrated in Figure 3, the offline MLP resulted in poor test accuracy for intervals between $0-10$ and $25-35$ indicating that the model was only able to accurately classify over a limited region of the problem space. The poor performance of the offline MLP is most likely a result of removing information about time-of-year from the feature set.

The second MLP was trained using an online manner. Figure 2 shows that the online baseline did indeed converge. It is important to note that the sharp decrease at the 0th interval is a result of the warm-up period which actually spans 10 epochs. Figure 1 shows the online MLP performance on the test set throughout the training period with and without re-sampling. In particular, Figure 1a shows that around ap-

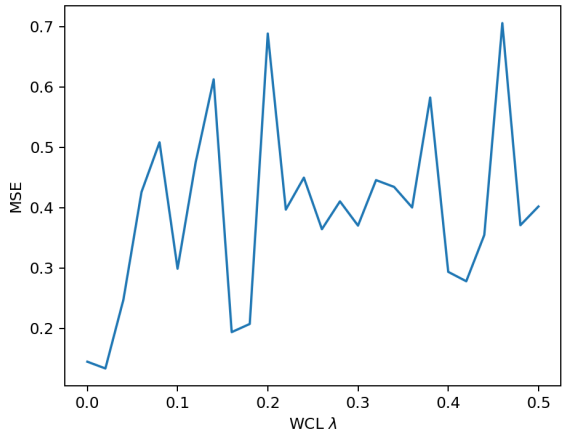


Figure 4. Grid search showing the MSE validation set prediction loss for various values of λ for the weight constrained MLP trained in an online fashion.

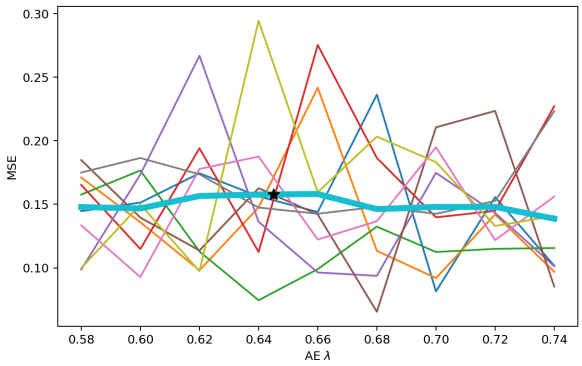


Figure 5. Grid search showing the MSE validation set prediction loss for various values of λ targeted around 0.645. Thick blue line is the mean of the 9 searches. Compared to figure 6 we don't see a minima at 0.645

proximately the 15th interval, the model exhibits a dramatic increase in test error on the interval 0-10, indicating the model "forgot" how to predict temperature for the previous intervals. Similarly, by the final training interval, the model exhibits an increase in test error on the interval $10 - 30$ but has low test error for the intervals $0 - 10$ and $30 - 35$. The low test error on the intervals $0 - 10$ and $30 - 35$ further support the claim that the model is exhibiting catastrophic forgetting as these intervals occur during the same season. As shown in Figure 1b, even with re-sampling, while the width and to a limited extent, the height of the error peaks is reduced, the error surface is generally similar to that of Figure 1a, indicating that re-sampling may not be the best approach for preventing catastrophic forgetting.

4.2. Weight Constrained Learning

Our initial attempt at overcoming catastrophic forgetting in MLPs for the NOAA dataset involved constraining the weights of the MLP as described in section 3.2. We first conducted a grid-search across λ for values between 0.0 and 0.5. The initial grid search, which showed no reduction in testing loss for non-zero λ values was done using an MLP that did not employ any normalization of the inputs and thus we have omitted those results. We further evaluated our weight constrained model by penalizing the KL divergence between the updated and previous weight distributions. (See Equation 2) However, after normalizing the inputs, as shown in Figure 4, testing loss never fell below the loss from an MLP with $\lambda = 0$, which is effectively the baseline MLP. While we did measure the wall-clock time for the weight constrained learning algorithm (see Table 2), we abandoned investigating it further as there was no indication of improved loss with a non-zero λ even when normalizing inputs.

4.3. Autoencoder

To test our hypothesis that an autoencoder with a regression layer will improve the online performance, we first did an an extensive grid search for the λ loss weighing parameter to find potential values that improve the loss over the baseline (Figure 6). After a single search on the validation set over $[0., 1.5]$ we searched the range $[0., 0.68]$ (Figure 6) and identified the best broad minima for MSE when λ is 0.645. We used this value of λ for all our subsequent experiments.

Unlike for the WCL, the grid search for the AE showed an improvement over the the baseline (when λ is zero). To verify that 0.645 is indeed a good minima, we performed a targeted search on the validation set around this value (Figure 5). After 9 repeated grid searches in this range we did not find a minima at this value (Figure 5). However, we still used 0.645 for the subsequent experiments.

We hypothesized that an AE would perform better than the baseline MLP for the online setting, requiring less samples when re-sampling previous samples. We tested by randomly re-sampling during training with $0, 1, 3, 5, 50$, or 100 samples from each previous interval (Figure 7). We found that increasing the number of samples that are re-sampled improved the MSE loss for all algorithms. Additionally, the autoencoder performed better than the online baseline MLP when re-sampled 3 samples and no re-sampling.

To better estimate the performance for the AE, we repeated the experiment on the untouched test set using λ 0.645 58 times comparing the baseline to the AE when re-sampled with 0, 5, and 100 samples (Figure 8, Table 1).

Using Welch's t-test, we found a significant improvement of 0.04 MSE for the AE when it was not re-sampled over the baseline (T-test p-value is 0.000027). Cohen's d is 0.81, indicating a large effect of the AE treatment. For the other

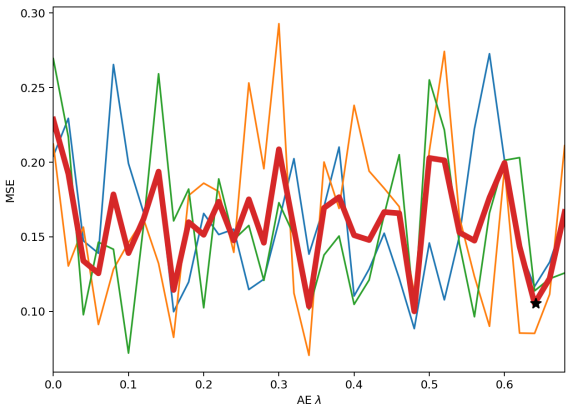


Figure 6. Grid search showing the MSE validation set prediction loss for various values of λ. 0.645 seems to the broadest minima. Thick red line is the mean of the 3 searches.

Sampling	Baseline	AE	d	p Value	N
0	0.19	0.15	0.81	2.70e-05	58
5	0.10	0.10	0.03	0.87	58
100	0.05	0.04	0.19	0.31	58

Table 1. The autoencoder vs baseline MLP MSE loss on the untouched test set. We also show the Cohen's d value, the Welch t-test p value, and the sample size. Sampling indicates the number of samples per interval with which we re-sampled.

re-sampling we found no significant difference.

To verify that the improvement for the AE is not due to the increase in network size (the decoder adds two more hidden layers), we also trained the online baseline MLP with 4 instead of 2 layers. We found a higher MSE for the 4-layer baseline MLP compared to the 2-layer baseline MLP (Figure 8).

4.4. Wall Clock Time

To compare the wall clock time required for each model type, we measured the time it took to fully train each model (Table 2). We observed that the offline baseline trained the slowest, however, that trained on all samples for 10 epochs, compared to the online algorithms that only trained for two epochs for the online samples. Among the online algorithms, the AE took longer to train than the baseline, compared to the WCL which was intermediate. Re-sampling slowed down training only for the AE.

5. Discussion

Our baseline results demonstrate that a standard online MLP does indeed exhibit catastrophic forgetting. We initially tried to solve this issue by constraining the weights such

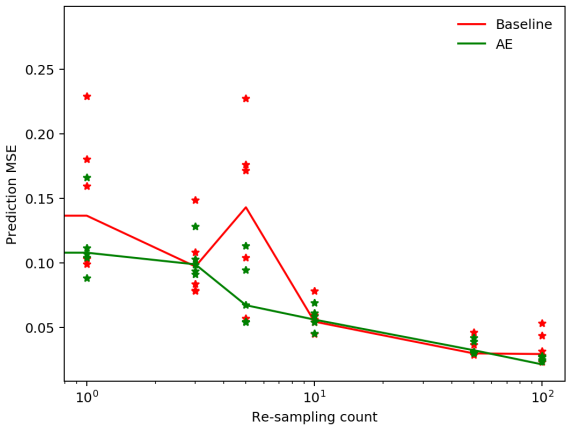


Figure 7. The MSE prediction loss of the baseline and the AE with λ 0.645 for different re-sampling values. The lines are the mean values.

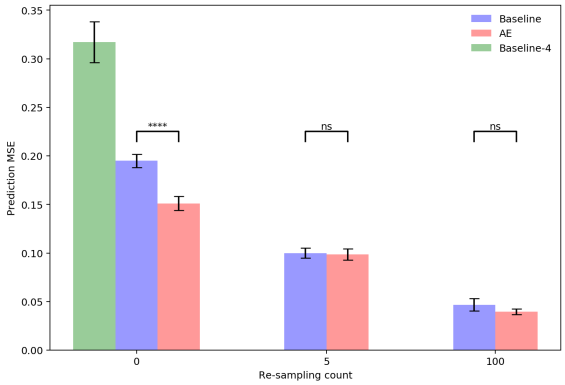


Figure 8. The autoencoder with λ 0.645 vs baseline MLP MSE loss on the untouched test set. We compare the MLP vs baseline for the re-sampling conditions 0, 5, and 100. Re-sampling is the number of random samples from previous intervals we re-use. The significance is from a Welch t-test. We also show the MSE loss for the MLP baseline with a larger model (4 hidden layers) compared to the other bars that were trained on a baseline with 2 hidden layers.

Experiment	Duration (minutes)
Offline Baseline	11.47
Online Baseline	7.07
Online Baseline Resampled	6.96
WCL	7.38
AE	8.08
AE Resampled	8.45

Table 2. The wall clock time for the various models and training methods. This measured the total time it took tor fully train the model. The resampled condition resampled 5 samples from previous days.

that significant changes to the weights would be penalized. We hypothesized that by constraining the weights of the model, weights important for regressing each interval would be preserved.

We demonstrated however, that this technique performs poorly when compared to the baseline MLP. Our actual goal was to prevent the model from forgetting previous examples, this suggests that constraining weights is a poor proxy for preventing forgetting of previous examples.

Next we hypothesized that an autoencoder would better learn the entire space such that when encountering new examples it would be less likely to forget. In particular, we believed that it would need less re-sampling to prevent forgetting. Our results do not show a significant re-sampling improvement difference between the AE and baseline. However, we did find that with zero re-sampling the AE significantly out performed the baseline on the untouched test set. To show that it is not simply due to the model size increase in the AE, we also trained a larger baseline model and we found it did not improve performance.

We also did a grid search to find the best value of the reconstruction vs prediction loss. After 9 repeated grid searches, we were unable to find a specific minima. However, the λ we selected earlier based on an initial grid search resulted in the improvement shown above. In our final results we also saw a large distribution of MSE for both the AE and baseline in the individual experimental runs. This suggests that there is some source of randomness that prevented us finding a consistent minima for λ across multiple repeats. We think it could be due to the model size being too small.

6. Future Work

Our motivation for selecting an autoencoder to reduce forgetting was because we believed an autoencoder would better learn the space due to the loss imposed by the reconstruction. However, ideally we would like a loss function that penalizes forgetting directly. We would like to investigate using a network that detects when forgetting occurs, and provides a corresponding loss. We would like to train a model that recognizes when forgetting occurs and use that as an additional loss directly.

Our dataset, while large, has only two major seasons. We would like to investigate using a dataset which changes frequently to see whether an autoencoder still provides benefits for longer term online training.

7. Conclusion

In this paper, we presented evidence of catastrophic forgetting in MLPs trained in an online fashion for weather forecasting. We subsequently introduced a novel approach to overcome catastrophic forgetting in an online setting using an autoencoder. The effectiveness of our approach was validated on the NOAA U.S. Climatic Normals dataset - an ideal dataset to evaluate online catastrophic forgetting due to its temporal and cyclical nature. Our autoencoder showed an improvement in test error by 0.04 with an effect size of 0.81 and a p-value of $2.70 * 10^{-5}$.

References

Arguez, Anthony, Durre, Imke, Applequist, Scott, Squires, Mike, Vose, Russell, Yin, Xungang, and Bilotta, Rocky. U.s. daily climate normals (1981-2010), 2010. URL https://data.nodc.noaa.gov/cgi-bin/iso?id=gov.noaa.ncdc:C00823.

Dredze, Mark, Crammer, Koby, and Pereira, Fernando. Confidence-weighted linear classification. In *International Conference on Machine Learning (ICML)*, 2008.

Goodfellow, Ian J., Mirza, Mehdi, Da, Xia, Courville, Aaron C., and Bengio, Yoshua. An empirical investigation of catastrophic forgeting in gradient-based neural networks. *CoRR*, abs/1312.6211, 2013.

Kemker, Ronald, Abitino, Angelina, McClure, Marc, and Kanan, Christopher. Measuring catastrophic forgetting in neural networks. *CoRR*, abs/1708.02072, 2017. URL http://arxiv.org/abs/1708.02072.

Kirkpatrick, James, Pascanu, Razvan, Rabinowitz, Neil C., Veness, Joel, Desjardins, Guillaume, Rusu, Andrei A., Milan, Kieran, Quan, John, Ramalho, Tiago, Grabska-Barwinska, Agnieszka, Hassabis, Demis, Clopath, Claudia, Kumaran, Dharshan, and Hadsell, Raia. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016. URL http://arxiv.org/abs/1612.00796.

Legg, Shane and Hutter, Marcus. A collection of definitions of intelligence. *CoRR*, abs/0706.3639, 2007. URL http://arxiv.org/abs/0706.3639.

Li, Yu, Li, Zhongxiao, Ding, Lizhong, Yang, Peng, Hu, Yuhui, Chen, Wei, and Gao, Xin. Supportnet: solving catastrophic forgetting in class incremental learning with support data. *CoRR*, abs/1806.02942, 2018.

Ma, Justin, Saul, Lawrence K., Savage, Stefan, and Voelker, Geoffrey M. Identifying suspicious urls: An application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 681–688, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553462. URL http://doi.acm.org/10.1145/1553374.1553462.

McCloskey, Michael and Cohen, Neal J. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165, 1989. ISSN 0079-7421. doi: 10.1016/S0079-7421(08)60536-8.

Nguyen, Cuong V., Li, Yingzhen, Bui, Thang D., and Turner, Richard E. Variational continual learning. *CoRR*, abs/1710.10628, 2017.

Rebuffi, Sylvestre-Alvise, Kolesnikov, Alexander, and Lampert, Christoph H. icarl: Incremental classifier and representation learning. *CoRR*, abs/1611.07725, 2016. URL http://arxiv.org/abs/1611.07725.

Robins, A. Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pp. 65–68, Nov 1993. doi: 10.1109/ANNES.1993.323080.

Rodrigues, Eduardo R., Oliveira, Igor, Cunha, Renato L. F., and Netto, Marco Aurélio Stelmar. Deepdownscale: a deep learning strategy for high-resolution weather forecast. *CoRR*, abs/1808.05264, 2018.

Serrà, Joan, Surís, Dídac, Miron, Marius, and Karatzoglou, Alexandros. Overcoming catastrophic forgetting with hard attention to the task. *CoRR*, abs/1801.01423, 2018. URL http://arxiv.org/abs/1801.01423.

Velez, Roby and Clune, Jeff. Diffusion-based neuromodulation can eliminate catastrophic forgetting in simple neural networks. *CoRR*, abs/1705.07241, 2017. URL http://arxiv.org/abs/1705.07241.

Zenke, Friedemann, Poole, Ben, and Ganguli, Surya. Improved multitask learning through synaptic intelligence. *CoRR*, abs/1703.04200, 2017. URL http://arxiv.org/abs/1703.04200.