A study on catastrophic forgetting in deep LSTM networks

Monika Schak and Alexander Gepperth

University of Applied Sciences Fulda, 36037 Fulda, Germany

Abstract. We present a systematic study of catastrophic forgetting (CF), i.e., the abrupt loss of previously acquired knowledge, when retraining deep recurrent LSTM networks with new samples. CF has recently received renewed attention in the case of feed-forward DNNs, and this article is the first work that aims to rigorously establish whether deep LSTM networks are afflicted by CF as well, and to what degree. In order to test this fully, training is conducted using a wide variety of high-dimensional image-based sequence classification tasks derived from established visual classification benchmarks (MNIST, Devanagari, FashionMNIST and EMNIST). We find that the CF effect occurs universally, without exception, for deep LSTM-based sequence classifiers, regardless of the construction and provenance of sequences. This leads us to conclude that LSTMs, just like DNNs, are fully affected by CF, and that further research work needs to be conducted in order to determine how to avoid this effect (which is not a goal of this study).

Keywords: LSTM · Catastrophic Forgetting

1 Introduction

This article is in the context of deep recurrent neural networks (more specifically: deep Long Short-Term Memory (LSTM) networks [13]) applied to the classification of sequences. Sequence classification presents many challenges, such as their variable length and the fact that their elements are often presented one after the other (see [36] for a more in-depth review of this topic). Typical applications of sequence classifiers are hand gesture recognition [9], human activity recognition [33] and natural language processing [23]. Prominent recent methods for sequence classification are recurrent neural networks, and in particular LSTM networks and their deep "extensions", see [12] and references therein. These classification architectures are based on a similar concept as prior work on echo state networks or reservoir computing [14], where the dynamical state of a recurrent system (reservoir, LSTM layer) represents the current and previously presented sequence elements, and a linear read-out mechanism is added "on top" of that to infer the sequence class. LSTM networks are attractive for this purpose since they are trained by gradient descent, so the "reservoir" can be adapted to the sequences it should represent.

Especially in situations where sequence classifiers need to be retrained *in situ*, typically based on user interaction (e.g., learning a new hand gesture by demonstration), the question of incremental or continual learning becomes relevant: what happens to knowledge of previously trained sequences when a new sequence class is presented to a deep LSTM sequence classifier? In addition, sequence elements in typical applications, like hand gesture or human activity recognition, are typically images (or depth images) and thus quite high-dimensional and hard to classify in their own right. So a study on catastrophic forgetting in deep LSTM networks should be sure to address this case in particular.

1.1 Related work

The catastrophic forgetting effect Catastrophic forgetting (CF) in feed-forward neural networks was first observed in [1] and subsequently studied in, e.g., [7]. Recent studies in the context of DNNs are described below. Essentially, CF is observed when a neural network is first trained on a dataset D_1 and subsequently re-trained on a disjunct dataset D_2. Very counter-intuitively, the typical outcome of such an experimental scheme is that all that was learned from D_1 is forgotten virtually immediately, within one or two mini-batch steps. We consider exactly such a scenario in this article, a minor difference being that samples from D_1 and D_2 are image sequences, which is why LSTM classifiers are used.

Catastrophic forgetting in Deep Neural Networks (DNNs) The field of incremental learning is broad, e.g., [25] and [10]. Recent systematic comparisons between different DNN approaches to avoid CF are performed in, e.g., [29, 18] or [26]. Principal recent approaches to avoid CF include ensemble methods [28, 6], dual-memory systems [30, 17, 27, 8] and regularization approaches. Whereas [11] suggest Dropout for alleviating CF, the EWC method [20] proposes to add a term to the energy function that protects weights that are important for the previous sub-task(s). Importance is determined by approximating the Fisher information matrix of the DNN. A related approach is pursued by the Incremental Moment Matching technique (IMM) (see [22]), where weights from DNNs trained on current and past sub-tasks are "merged" using the Fisher information matrix. Other regularization-oriented approaches are proposed in [3, 32] and [19] which focus on enforcing sparsity of neural activities by lateral interactions within a layer.

Catastrophic forgetting in (deep) LSTM networks There is little to no previous work on measuring catastrophic forgetting in LSTM networks. There seems to be a tentative consensus that LSTM might subject to CF, but we found no scientific work documenting this systematically, for complex, high-dimensional sequence classification problems. A simpler recurrent sequence classification model is tested for CF in [5] with the result that this model (without modifications) exhibits strong CF. This article uses short image sequences derived from MNIST as a basis for its investigation, and individual sequence elements are further reduced in dimensionality by PCA. A modified form of LSTM

based on the Elastic Weight Consolidation method [20] is presented in [23], but CF behavior is not systematically analyzed as the objective of the article is the incremental training of conversational agents. A dual-memory approach to incremental LSTM is presented in [15] for the purpose of land cover prediction in high-dimensional images, again without documenting the CF effect systematically.

Table 1: Overview of each dataset's detailed properties. Image dimensions are given as width × height × channels. Concerning data imbalance, the largest percentual difference in sample count between any two classes is given for training and test data, a value of 0 indicating a perfectly balanced dataset.

Properties / **Dataset**	image size	number of elements		class balance (%)	
		train	test	train	test
Devanagari	32 ×32 ×1	18.000	2.000	0.3	2.7
EMNIST	28 ×28 ×1	345.035	57.918	2.0	2.0
FashionMNIST	28 ×28 ×1	60.000	10.000	0	0
MNIST	28 ×28 ×1	55.000	10.000	2.2	2.4

1.2 Goals and contributions of the article

We aim at determining unambiguously whether LSTM and deep LSTM-based sequence classifiers are prone to the catastrophic forgetting effect or not when retrained with samples from one or more additional sequence classes, especially for the case where sequence elements are high-dimensional images that require deep networks in order to be solved satisfactorily. We do not impose application constraints on memory consumption or execution time when performing incremental learning experiments as it is done in [26] since LSTM training is memory-consuming in any case. However, we ensure realism w.r.t. causality, meaning that the number and nature of additional classes are not known beforehand (i.e., in order to select a good topology for deep LSTM), which is in accordance with [26].

Regardless of the actual outcome (CF, no CF or CF in some cases), such an investigation is important because it provides solid justification for further work on avoiding the CF effect in deep LSTM classifiers, or else why CF can be ignored in applications of such architectures.

We wish to make it clear that this study does not propose methods to get rid of catastrophic forgetting (which has proven difficult for DNNs): for the time being, we just aim at clearly showing that this effect is an universally occurring one for LSTM networks.

Fig. 2: Visualization of one sample per class from the four visual classification benchmarks used in this article.

2 Methods

The experimental paradigm is based on the notion of an incremental sequence classification task (ISCT), which is, simply put, a sequence classification problem divided into two disjunct parts. The first part is used for the initial training of a deep LSTM network, whereas the second part is used for subsequent retraining. While training on the second part of the ISCT, accuracy on the union of test sets from both parts is monitored to detect catastrophic forgetting.

2.1 Visual benchmarks for constructing sequence classification tasks

We construct incremental sequence classification tasks based on images taken from the following visual classification benchmarks (see Table 1 for details about the benchmarks and Fig. 2 for a visual impression).
MNIST [21] is the common benchmark for computer vision systems and classification problems. It consists of gray scale images of handwritten digits (0-9).
EMNIST [4] is an extended version of MNIST with additional classes of handwritten letters. There are different variations of this dataset: we extract the ten best-represented classes from the *By_Class* variation containing 62 classes.
Devanagari [2] contains gray-scale images of Devanagari handwritten letters. From the 46 character classes (1.700 images per class) we extract ten random classes.
FashionMNIST [35] consists of images of clothes in ten classes and is structured like the MNIST dataset. We use this dataset for our investigations because it is a "more challenging classification task than the simple MNIST digits data [35]".

2.2 Incremental sequence classification tasks

Construction of sequence classes We construct a common pool of ten ($k = 0, \ldots, 9$) sequence classes characterized by vectors $\boldsymbol{s}^k \in \mathbb{R}^{N_k}$, whose length N_k is randomly varied between 5 and 15, and whose integer entries s_i^k are randomly chosen from the range $[0, 9]$. Each sample from a sequence class k thus has N_k elements (frames) $\boldsymbol{e}_i, i = 0, \ldots, N_k - 1$, each being a (flattened) image

Table 2: All ten sequence classes, with sequence class k being characterized by the vector $\boldsymbol{s}_k$ that defines the visual classes where individual sequence elements (frames) are chosen from, see text for details. Please note that a visual class (e.g., the digit class "1" from MNIST) can appear more than once in a given sequence class. Using these definitions, the ten sequence classes are generated for each of the four visual benchmarks.

seq. class	seq. def $\boldsymbol{s}^k$	seq.class	seq.def $\boldsymbol{s}^k$
0	44671365876	1	1373561961
2	35445909328241	3	9314487292918
4	3675082469	5	45406816421282
6	7534519793178	7	02890
8	69959	9	21024269755

Table 3: Incremental sequence classification tasks (ISCTs) used for measuring catastrophic forgetting. Shown are the sequence classes (see Table 2) used for initial training and retraining of deep LSTM models. Each ISCT is constructed for all the benchmarks: MNIST, FashionMNIST, EMNIST and Devanagari.

ISCT	initial	retrain
5-5a	0,4,5,6,9	1,2,3,7,8
5-5b	0,1,2,3,4	5,6,7,8,9
5-1a	0,4,5,6,9	8
5-1b	0,1,2,3,4	9

randomly taken from class s_i^k of one of the four visual benchmarks (see Sec. 2.1). An overview of the constructed sequence classes is given in Table 2, and Fig. 3 gives a visual impression of actual sequence samples. The chosen sequence construction strategy assumes the presence of ten visual classes in each benchmark: where more than ten classes are available, we keep the ten best-represented ones (EMNIST), or we keep ten random ones if all classes are equally well represented (Devanagari).

Construction of incremental sequence classification tasks From the pool of ten sequence classes, we construct four incremental sequence classification tasks (ISCT) for measuring catastrophic forgetting. Two of them (denoted 5-5a and 5-5b) use a subset of five sequence classes for initial training and five sequence classes for retraining, whereas the two others (denoted 5-1a and 5-1b) use five sequence classes for initial training and one sequence class for retraining. Each ISCT contains training and test sets that are constructed from an 80/20 partition of available sequence samples. Table 3 gives an overview of the ISCTs used for measuring catastrophic forgetting in this article.

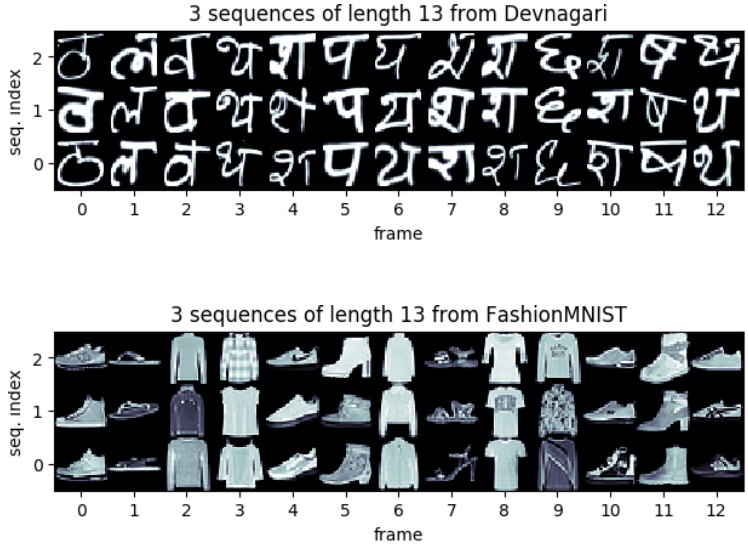

Fig. 3: Visualization of 3 samples (sequences), taken from an exemplary sequence class with 13 elements (frames), with frames coming from the classes 4-3-4-5-6-5-5-6-2-0-2-5-5 of the underlying benchmark, shown for Devanagari (top) and FashionMNIST (bottom).

2.3 Deep LSTM models

We use a standard deep LSTM architecture with linear softmax readout layer and cross-entropy loss function as outlined in [13]. Number and size of hidden layers, which are all set to have the same number of LSTM cells, will be varied in the experiments and are denoted (L, S). The LSTM model equations for computing activations $\boldsymbol{h}_t$ of a single LSTM layer read as follows:

$$\begin{aligned}
\boldsymbol{i}_t &= \sigma\left(W_{xi}\boldsymbol{x}_t + W_{hi}\boldsymbol{h}_{t-1} + W_{ci}\boldsymbol{c}_{t-1} + \boldsymbol{b}_i\right) \\
\boldsymbol{f}_t &= \sigma\left(W_{xf}\boldsymbol{x}_t + W_{hf}\boldsymbol{h}_{t-1} + W_{cf}\boldsymbol{c}_{t-1} + \boldsymbol{b}_f\right) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t\boldsymbol{c}_{t-1} + \boldsymbol{i}_t \tanh\left(W_{xc}\boldsymbol{x}_t + W_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c\right)\right) \\
\boldsymbol{o}_t &= \sigma\left(W_{xo}\boldsymbol{x}_t + W_{ho}\boldsymbol{h}_{t-1} + W_{co}\boldsymbol{c}_t + \boldsymbol{b}_o\right) \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \tanh(\boldsymbol{c}_t)
\end{aligned} \tag{1}$$

3 Experiments

For our experiments we use the TensorFlow (v1.11) implementation of a Recurrent Neural Network with multiple LSTM cells under Python (v3.6). We always use the Adam optimizer included in TensorFlow for performing gradient descent. We distinguish two principal experimental objectives:

– **Consistency of deep LSTM models** In this step, we verify that our architecture is working correctly on the given classification problems by comparing them to the known performance of present-day DNNs on the visual benchmarks we use. For all visual benchmarks, we train deep LSTM models on ten sequence classes, each sequence class containing one element randomly chosen from a single, distinct image class in the benchmark. The classification of such one-element sequences amounts to classifying the images themselves, with recurrency being effectively switched off since the sequences have length one. If the deep LSTM architecture is chosen correctly, the classification accuracy should be comparable to the known accuracy of DNNs on a particular benchmark, thus establishing that our deep LSTMs are correctly used and parameterized.
– **Investigation of catastrophic forgetting** Here, we introduce incremental learning to our architecture: we train deep LSTM networks as described in Sec. 2.3 on the incremental sequence classification tasks (see Sec. 2.2) in two steps as outlined in Sec. 2: first on an initial set of sequence classes followed by retraining on a different set of sequence classes. During retraining, an evaluation of test accuracy is conducted on the union of test samples from both parts of the ISCT, with the aim of detecting catastrophic forgetting after the onset of retraining.

3.1 Consistency

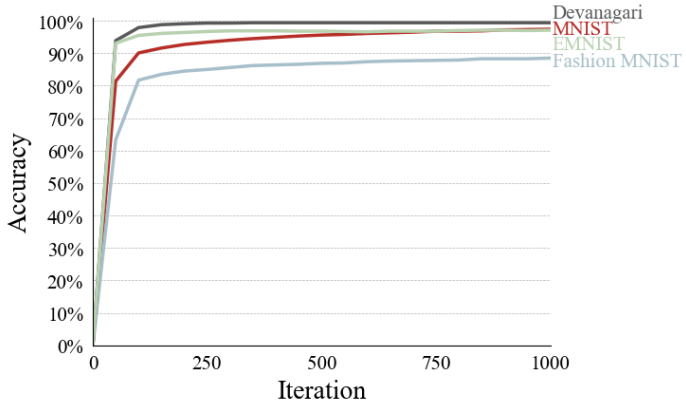

Fig. 4: Consistency test results

We vary the number of hidden layers and their size $(L, S) \in \{(1, 100), (1, 200), (1, 500), (3, 800), (5, 200)\}$ and use a fixed learning rate of $\epsilon = 0.0001$, a fixed batch size of $b = 1.000$ and a fixed number of iterations $I = 1.000$. To make sure the results are significant and consistent, we repeat every experiment five times

and calculate the average loss and accuracy. Table 4 shows the results for this preliminary experiment. As can be seen, our networks achieve accuracies that are generally comparable to those one would obtain when using simple DNN architectures, which makes it very plausible that our deep LSTMs are correctly parameterized and excludes gross errors in the deep LSTM setup. Fig. 4 shows the best result for each benchmark (in percent).

Table 4: Results of the consistency tests: Averaged accuracy over five experiments.

(L, S)	$(1, 100)$	$(1, 200)$	$(1, 500)$	$(3, 800)$	$(5, 200)$
MNIST	94.6	95.7	96.8	97.4	95.2
Fashion MNIST	86.7	87.7	88.5	88.4	85.8
Devanagari	87.9	93.3	97.1	99.4	98.4
EMNIST	87.1	88.8	91.0	97.1	95.2

3.2 Investigation of catastrophic forgetting

To test whether deep LSTM networks are prone to catastrophic forgetting when retraining an already trained one with new sequences, we perform initial training and retraining using the generated ISCTs (see Fig. 3 and Sec. 2.2). To exclude that results are due to a particular choice of topology, we vary the number and size of hidden layers $(L, S) \in \{(1, 100), (1, 200), (1, 500), (3, 800), (5, 200)\}$ and use a fixed learning rate $\epsilon = 0.0001$, a fixed batch size $b = 1.000$ and a fixed number of iterations for initial training and retraining $I_T = I_R = 1.000$. To make sure the results are significant, we repeat every experiment five times and calculate the average classification accuracy. To ensure the results are not influenced by our choice of sequence classes for initial training and retraining, we additionally average results over the 5-5a/5-5b and 5-1a/5-1b experiments. In total we run 400 different incremental learning experiments:

- 5 different topologies: $(L, S) \in \{(1, 100), (1, 200), (1, 500), (3, 800), (5, 200)\}$
- 4 different ISCTs (5-5a, 5-5b, 5-1a, 5-1b) from 4 visual benchmarks
- 10 repetitions for each ISCT

Table 5 shows the averaged accuracy for the 5-1 ISCTs for all tested architectures, at the end of initial training and during retraining ($I_T = 1000, t < I_R, t \in \{1, 1000\}$). The best achieved results for those experiments are shown in Fig. 5. As can be seen, the first part of the experiments where we perform initial training shows similar results to the ones achieved in our consistency tests (which is unsurprising). In the majority of cases, the results on five-element sequences are even better than those we obtain on single-element sequences in the consistency tests. As soon as we retrain the network with additional sequence classes, the

accuracy decreases drastically and almost instantaneously to 60-80% after one iteration and then to about 17% during the next few iterations, where it remains. This is indeed the result one would expect for a classifier that has learned about one sequence class and has totally forgotten about the other five it has learned before.

Similarly, Table 6 shows the averaged accuracy for the 5-5 ISCTs for all tested architectures, at the end of initial training and during retraining ($I_T = 1000, t < I_R, t \in \{1, 1000\}$). The best results for this part of our study are shown in Fig. 5. During the first couple iterations of retraining the network, the accuracy drops to between 20% and 50% (depending on the architecture), then increases to about 50-55% during the next 25 iterations and remains there until we stop the tests after 1.000 iterations. Again, this is the accuracy one would expect if half of the ten sequence classes has been well learned during retraining, but the other half has been completely forgotten.

4 Discussion and conclusions

Principal outcomes We find that sequence classifiers based on deep LSTM networks are heavily afflicted by catastrophic forgetting for complex, high-dimensional

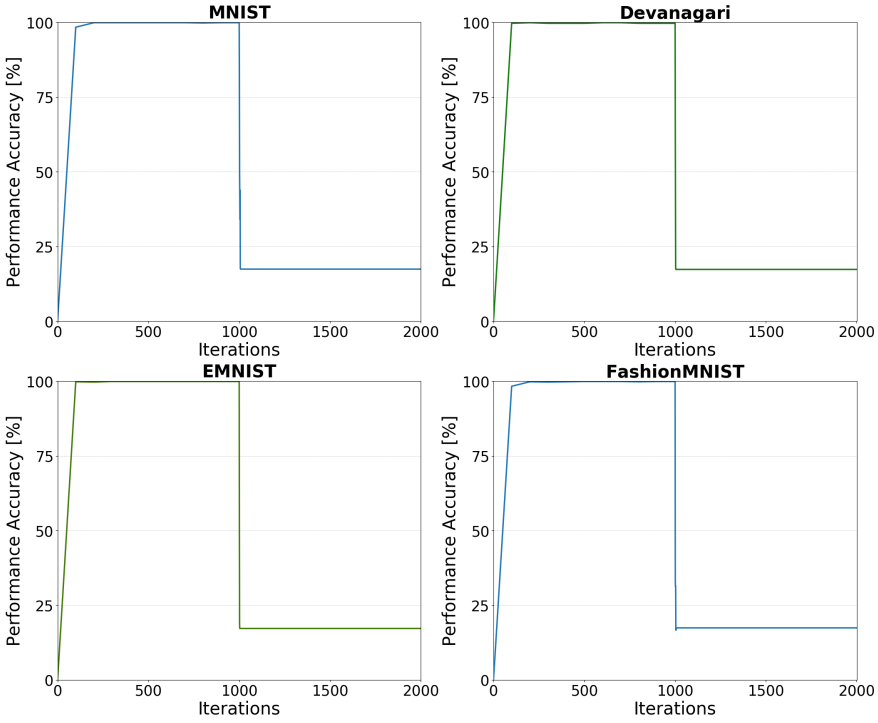

Fig. 5: Results of incremental learning for the 5-1 ISCTs.

incremental sequence classification tasks. Within only a few mini-batch iterations, almost all knowledge about previously trained data is lost and the network is able to perform an accurate classification only on the newly trained sequences. LSTM topology has no influence at all on this effect, to the extent we were able to test this, which is different from feed-forward DNNs where topology has a small influence without however in any way eliminating the problem [26]. It is relatively intuitive why this should be the case, since recurrent networks are conceivably more sensitive to even small changes in weights due to retraining, since each change is amplified by recurrent connections. Also in slight contrast to feed-forward DNNs, it does not make a difference whether a single or many classes are added during re-training, although the effect is slim at best even for DNNs, see [26].

Significance of results We find this to be a very important result about LSTM sequence classifiers: catastrophic forgetting is a universally occurring effect. So it is not possible to add new knowledge to a trained LSTM classifier in a naive way without losing all previously acquired knowledge. While forgetting in such a scenario is not unreasonable to expect simply due to limited network resources, it should be gradual so that re-training can be stopped whenever the onset of forgetting is detected. This "graceful decay" behavior is however not

Table 5: Results of incremental learning for the 5-1 ISCTs: Averaged test accuracy (in percent) over ten experiments. During initial training, test accuracy is measured on the first part of each ISCT, during retraining on the six sequence classes included in either training or retraining.

(L, S)	$(1, 100)$	$(1, 200)$	$(1, 500)$	$(3, 800)$	$(5, 200)$
MNIST					
$I_T = 1.000$	99.9	99.9	99.9	99.9	99.9
$I_R = 1$	83.2	83.1	81.9	67.2	68.8
$I_R = 1.000$	16.6	16.6	16.4	16.7	16.8
Fashion MNIST					
$I_T = 1.000$	99.9	99.9	99.9	99.9	99.9
$I_R = 1$	83.2	82.8	74.3	64.1	61.9
$I_R = 1.000$	16.6	16.6	16.7	18.4	16.8
Devanagari					
$I_T = 1.000$	69.7	71.3	77.7	98.9	97.1
$I_R = 1$	57.9	58.9	64.9	59.4	57.4
$I_R = 1.000$	17.4	17.1	16.7	16.0	16.6
EMNIST					
$I_T = 1.000$	93.0	93.4	95.0	99.9	99.9
$I_R = 1$	77.5	77.6	78.9	56.6	68.4
$I_R = 1.000$	17.6	16.7	16.5	16.8	16.8

Table 6: Results of incremental learning for the 5-5 ISCTs: averaged test accuracy (in percent) over ten experiments. During initial training, test accuracy is measured on the first part of each ISCT, during retraining on all ten sequence classes.

(L, S)	$(1, 100)$	$(1, 200)$	$(1, 500)$	$(3, 800)$	$(5, 200)$
MNIST					
$I_T = 1.000$	99.9	99.9	99.9	99.9	99.9
$I_R = 1$	50.1	49.8	48.3	32.1	20.2
$I_R = 1.000$	55.3	55.0	55.1	54.8	54.9
Fashion MNIST					
$I_T = 1.000$	99.9	99.9	99.9	99.9	99.9
$I_R = 1$	49.9	49.2	44.6	32.2	19.9
$I_R = 1.000$	55.0	55.1	54.9	55.3	55.0
Devanagari					
$I_T = 1.000$	70.1	71.0	78.3	98.8	96.7
$I_R = 1$	34.9	36.3	38.6	39.1	25.7
$I_R = 1.000$	46.1	46.2	49.2	55.5	53.9
EMNIST					
$I_T = 1.000$	92.9	93.4	95.4	99.9	99.9
$I_R = 1$	46.5	46.8	47.6	37.7	24.1
$I_R = 1.000$	54.4	54.1	54.1	55.0	54.9

what is observed in our experiments, and once forgetting is detected, it is already too late to stop re-training.

Justification of using LSTMs We employ deep LSTM classifiers in this article because the problems treated here are inherently high-dimensional and, above all, sequential. The most important property of sequences, for the purposes of this article, is that the different sequence classes need not be the same length, i.e., samples from different sequence classes may well contain a different number of frames. This, together with the high-dimensional nature of the images, effectively excludes strategies that concatenate all images in a given sequence and present the result to a feed-forward DNN. First of all, memory usage would be excessive. More importantly, a sequence could only be classified once its end was reached: but to determine that, its class would have to be known. Of course, a fixed upper limit on sequence length could be imposed, but this would incur even higher memory requirements. For all these reasons, we believe that the use of deep LSTMs is the only feasible choice for the problems presented here, which are typical representatives for, e.g., video classification tasks.

Datasets used for this study The datasets, that is, the incremental sequence classification tasks (ISCTs) used in this study consist of image sequences

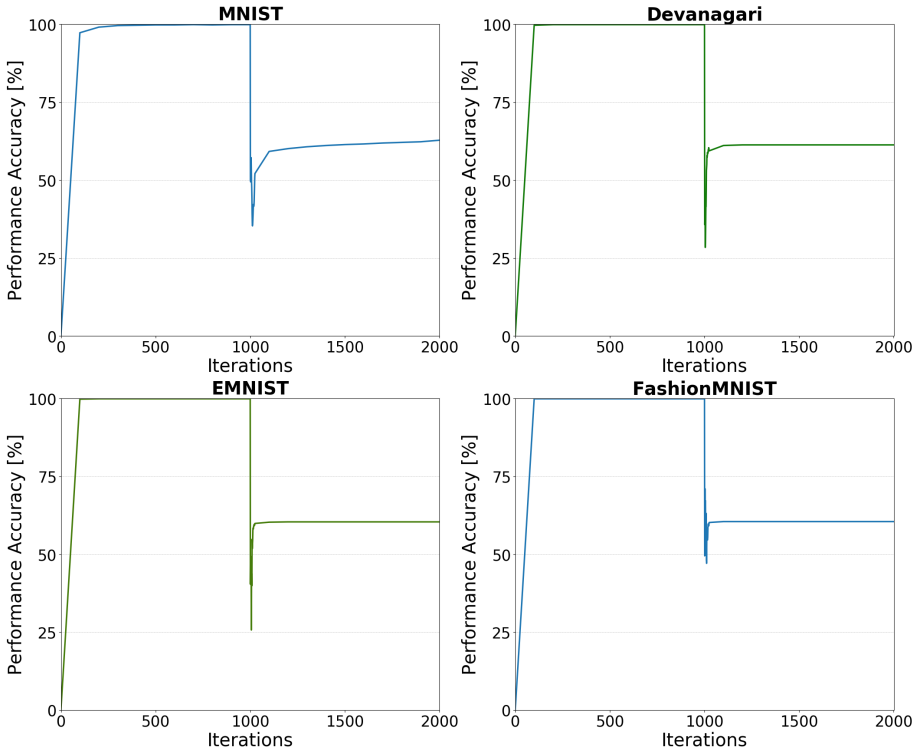

Fig. 6: Results of incremental learning for the 5-5 ISCTs.

and are thus related to videos. The reason for not using datasets containing real videos is that we wished to treat problems which, when not treating sequential learning problems, can be solved to a high degree of precision so that the forgetting effect is pronounced enough to be observed. So, while it might be argued that we used artificial data that are really too simple to give meaningful results, we point out that if CF occurs even for relatively simple problems like these, it is sure to occur for more complex problems as well (as it is the case for DNNs, see, e.g., [26].

Context and next steps This study deliberately does not propose a solution to the problem because we believe the existence of the problem needs to be rigorously established first. It is conceivable that EWC or IMM-like mechanisms [20, 22] may alleviate catastrophic forgetting for deep LSTM networks, and approaches based on generative replay [30, 34, 16] presumably generalize to sequence classification although the generation of sequences as opposed to single images may prove challenging. Approaches based on the so-called "distillation loss" regularization [31, 24] will be looked into as well, mostly because they should be pretty straightforward to implement for LSTM networks. We

hope, by presenting these results, to encourage researchers to investigate continual training methods not only for DNNs, but for LSTM sequence classifiers as well.

References

1. Catastrophic interference in connectionist networks: the sequential learning problem
2. Acharya, S.: Deep Learning Based Large Scale Handwritten Devanagari Character Recognition (2015)
3. Aljundi, R., Rohrbach, M., Tuytelaars, T.: Selfless Sequential Learning (2018)
4. Cohen, G., Afshar, S., Tapson, J., Van Schaik, A.: EMNIST: Extending MNIST to handwritten letters. Proceedings of the International Joint Conference on Neural Networks **2017-May**, 2921–2926 (2017). https://doi.org/10.1109/IJCNN.2017.7966217
5. Coop, R., Arel, I.: Mitigation of catastrophic forgetting in recurrent neural networks using a fixed expansion layer. In: The 2013 International Joint Conference on Neural Networks (IJCNN). pp. 1–7 (Aug 2013). https://doi.org/10.1109/IJCNN.2013.6707047
6. Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A., Wierstra, D.: PathNet: Evolution Channels Gradient Descent in Super Neural Networks (2017)
7. French, R.: Catastrophic forgetting in connectionist networks. Trends in Cognitive Sciences **3**(4), 128–135 (1999). https://doi.org/10.1016/S1364-6613(99)01294-2
8. Gepperth, A., Karaoguz, C.: A bio-inspired incremental learning architecture for applied perceptual problems. Cognitive Computation (2015), accepted
9. Gepperth, A., Sarkar, A., Kopinski, T., Handmann, T.: Dynamic hand gesture recognition for mobile systems using deep LSTM. In: International Conference on Intelligent Human Computer Interaction (2017)
10. Gepperth, A., Hammer, B.: Incremental learning algorithms and applications. European Symposium on Artificial Neural Networks ({ESANN}) (April), 357–368 (2016)
11. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks (2013). https://doi.org/10.1088/1751-8113/44/8/085201
12. Graves, A.: Supervised sequence labelling. In: Supervised sequence labelling with recurrent neural networks, pp. 5–13. Springer (2012)
13. Graves, A., Jaitly, N.: Towards end-to-end speech recognition with recurrent neural networks. In: International Conference on Machine Learning. pp. 1764–1772 (2014)
14. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In: Advances in neural information processing systems. pp. 609–616 (2003)
15. Jia, X., Khandelwal, A., Nayak, G., Gerber, J., Carlson, K., West, P., Kumar, V.: Incremental dual-memory LSTM in land cover prediction. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 867–876. KDD '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3097983.3098112, http://doi.acm.org/10.1145/3097983.3098112
16. Kamra, N., Gupta, U., Liu, Y.: Deep generative dual memory network for continual learning. arXiv preprint arXiv:1710.10368 (2017)

17. Kemker, R., Kanan, C.: FearNet: Brain-Inspired Model for Incremental Learning pp. 1–16 (2017)
18. Kemker, R., McClure, M., Abitino, A., Hayes, T., Kanan, C.: Measuring Catastrophic Forgetting in Neural Networks (2017). https://doi.org/10.1073/pnas.1611835114
19. Kim, H.E., Kim, S., Lee, J.: Keep and Learn: Continual Learning by Constraining the Latent Space for Knowledge Preservation in Neural Networks (2018)
20. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R.: Overcoming catastrophic forgetting in neural networks (2016). https://doi.org/10.1073/pnas.1611835114, http://arxiv.org/abs/1612.00796
21. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Apllied to Document Recognition (1998)
22. Lee, S.W., Kim, J.H., Jun, J., Ha, J.W., Zhang, B.T.: Overcoming Catastrophic Forgetting by Incremental Moment Matching (Nips), 1–16 (2017)
23. Lee, S.: Toward continual learning for conversational agents. CoRR **abs/1712.09943** (2017), http://arxiv.org/abs/1712.09943
24. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2018)
25. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual Lifelong Learning with Neural Networks: A Review pp. 1–29 (2018)
26. Pfülb, B., Gepperth, A.: A comprehensive, application-oriented study of catastrophic forgetting in dnns. In: International Conference on Learning Representations (ICLR) (2019), accepted
27. Rebuffi, S.a., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL : Incremental Classifier and Representation Learning pp. 2001–2010 (2017)
28. Ren, B., Wang, H., Li, J., Gao, H.: Life-long learning based on dynamic combination model. Applied Soft Computing Journal **56**, 398–404 (2017). https://doi.org/10.1016/j.asoc.2017.03.005
29. Serrà, J., Surís, D., Miron, M., Karatzoglou, A.: Overcoming catastrophic forgetting with hard attention to the task. arXiv preprint arXiv:1801.01423 (2018)
30. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual Learning with Deep Generative Replay (Nips) (2017)
31. Shmelkov, K., Schmid, C., Alahari, K.: Incremental learning of object detectors without catastrophic forgetting. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3400–3409 (2017)
32. Srivastava, R.K., Masci, J., Kazerounian, S., Gomez, F., Schmidhuber, J.: Compete to Compute. Nips pp. 2310–2318 (2013)
33. Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L.: Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters (2018). https://doi.org/https://doi.org/10.1016/j.patrec.2018.02.010, http://www.sciencedirect.com/science/article/pii/S016786551830045X
34. Wu, C., Herranz, L., Liu, X., Wang, Y., van de Weijer, J., Raducanu, B.: Memory replay gans: learning to generate images from new categories without forgetting. arXiv preprint arXiv:1809.02058 (2018)
35. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms pp. 1–6 (2017)
36. Xing, Z., Pei, J., Keogh, E.: A brief survey on sequence classification. ACM Sigkdd Explorations Newsletter **12**(1), 40–48 (2010)