Overcoming Catastrophic Forgetting with Hard Attention to the Task

SUPPLEMENTARY MATERIALS

A. Data

The data sets used in our experiments are summarized in Table 1. The MNIST data set (LeCun et al., 1998) comprises 28×28 monochromatic images of handwritten digits. Fashion-MNIST (Xiao et al., 2017) comprises gray-scale images of the same size from Zalando's articles[1]. The German traffic sign data set (TrafficSigns; Stallkamp et al., 2011) contains traffic sign images. We used the version of the data set from the Udacity self-driving car github repository[2]. The NotMNIST data set (Bulatov, 2011) comprises glyphs extracted from publicly available fonts, making a similar data set to MNIST; we just need to resize the images[3]. The SVHN data set (Netzer et al., 2011) comprises digits cropped from house numbers in Google Street View images. The FaceScrub data set (Ng & Winkler, 2014) is widely used in face recognition tasks (Kemelmacher-Shlizerman et al., 2016). Because some of the images listed in the original data set were not hosted anymore on the corresponding Internet domains, we use a version of the data set stored on the MegaFace challenge website[4] (Kemelmacher-Shlizerman et al., 2016), from which we select the first 100 people with the most appearances[5]. The CIFAR10 and CIFAR100 data sets contain 32×32 color images (Krizhevsky, 2009).

To match the image input shape required in our experiments, some of the images in the corresponding data sets need to be resized (FaceScrub, TrafficSigns, and NotMNIST) or padded with zeros (MNIST and FashionMNIST). In addition, for the data sets comprising monochromatic images, we replicate the image across all RGB channels. Note that we do not perform any sort of data augmentation; we just adapt the inputs. We provide the necessary code to perform such adaptations in the links listed above.

Table 1. Data sets used in the study: name, reference, number of classes, and number of train and test instances.

Data set	Classes	Train	Test
CIFAR10 (Krizhevsky, 2009)	10	50,000	10,000
CIFAR100 (Krizhevsky, 2009)	100	50,000	10,000
FaceScrub (Ng & Winkler, 2014)	100	20,600	2,289
FashionMNIST (Xiao et al., 2017)	10	60,000	10,000
NotMNIST (Bulatov, 2011)	10	16,853	1,873
MNIST (LeCun et al., 1998)	10	60,000	10,000
SVHN (Netzer et al., 2011)	100	73,257	26,032
TrafficSigns (Stallkamp et al., 2011)	43	39,209	12,630

B. Raw Results

B.1. Task Mixture

We report all forgetting ratios $\rho^{\leq t}$ for $t = 1$ to 8 in Table 2. A total of 10 runs with 10 different seeds are performed and the averages and standard deviations are taken.

[1] https://github.com/zalandoresearch/fashion-mnist

[2] https://github.com/georgesung/traffic_sign_classification_german

[3] Code and processed data available on github: https://github.com/nkundiushuti/notmnist_convert

[4] http://megaface.cs.washington.edu/participate/challenge.html

[5] Code and processed data available on github: https://github.com/nkundiushuti/facescrub_subset

Table 2. Average forgetting ratio $\rho^{\leq t}$ for the considered approaches (10 runs, standard deviation into parenthesis).

Approach	$\rho^{\leq 1}$	$\rho^{\leq 2}$	$\rho^{\leq 3}$	$\rho^{\leq 4}$	$\rho^{\leq 5}$	$\rho^{\leq 6}$	$\rho^{\leq 7}$	$\rho^{\leq 8}$
LFL	-0.00 (0.01)	-0.73 (0.29)	-0.88 (0.18)	-0.89 (0.13)	-0.91 (0.11)	-0.90 (0.09)	-0.92 (0.08)	-0.92 (0.08)
LWF	-0.00 (0.01)	-0.14 (0.13)	-0.38 (0.17)	-0.63 (0.11)	-0.68 (0.08)	-0.70 (0.03)	-0.76 (0.06)	-0.80 (0.06)
SGD	-0.00 (0.00)	-0.20 (0.08)	-0.41 (0.09)	-0.49 (0.07)	-0.54 (0.07)	-0.57 (0.06)	-0.62 (0.06)	-0.66 (0.03)
IMM-Mode	-0.00 (0.01)	-0.11 (0.08)	-0.27 (0.12)	-0.37 (0.10)	-0.39 (0.07)	-0.45 (0.05)	-0.49 (0.06)	-0.49 (0.05)
SGD-F	-0.00 (0.00)	-0.20 (0.15)	-0.30 (0.15)	-0.38 (0.11)	-0.42 (0.09)	-0.44 (0.08)	-0.45 (0.07)	-0.44 (0.06)
IMM-Mean	-0.00 (0.00)	-0.12 (0.10)	-0.24 (0.11)	-0.32 (0.06)	-0.37 (0.06)	-0.40 (0.06)	-0.42 (0.07)	-0.42 (0.04)
EWC	-0.00 (0.00)	-0.08 (0.06)	-0.15 (0.11)	-0.18 (0.07)	-0.21 (0.07)	-0.23 (0.04)	-0.25 (0.05)	-0.25 (0.03)
PathNet	-0.02 (0.03)	-0.09 (0.16)	-0.11 (0.19)	-0.12 (0.21)	-0.14 (0.22)	-0.15 (0.23)	-0.17 (0.23)	-0.17 (0.23)
PNN	-0.10 (0.12)	-0.11 (0.10)	-0.13 (0.09)	-0.14 (0.04)	-0.13 (0.03)	-0.13 (0.02)	-0.12 (0.01)	-0.11 (0.01)
HAT	**-0.01 (0.02)**	**-0.02 (0.03)**	**-0.03 (0.03)**	**-0.03 (0.02)**	**-0.04 (0.02)**	**-0.05 (0.02)**	**-0.06 (0.02)**	**-0.06 (0.01)**

B.2. Layer Use

In Fig. 1 we show an example of layer capacity monitoring as the sequence of tasks evolves. As mentioned in the main paper, we can compute a percent of active weights for a given layer and task.

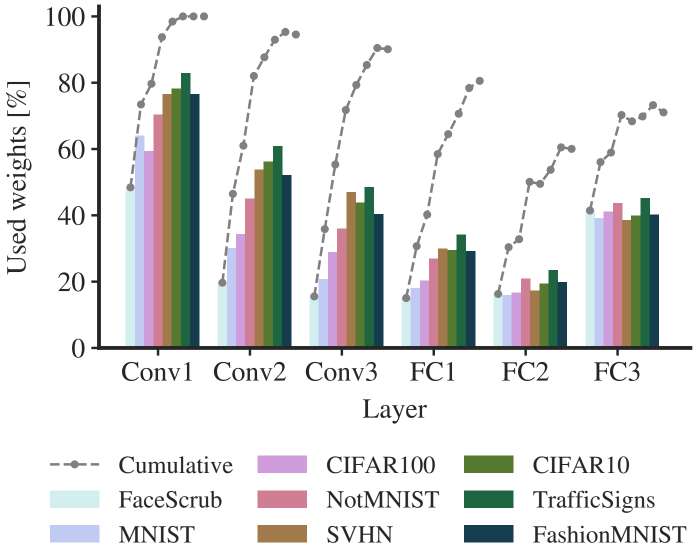

Figure 1. Layer-wise weight usage with sequential task learning, including (lines) and excluding (bars) the cumulative attention of past tasks. Task sequence corresponds to seed 0.

B.3. Network Compression

The final results of the network compression experiment reported in the main paper (after reaching convergence) are available in Table 3. We run HAT on isolated tasks with $c = 1.5$ and uniform embedding initialization $\mathcal{U}(0, 2)$.

B.4. Training Time

To have an idea of the training time for each of the considered approaches, we report some reference values in Table 4. We see that HAT is also quite competitive in this aspect.

C. Additional Results

C.1. Incremental CIFAR

As an additional experiment to complement our evaluation, we consider the incremental CIFAR setup, following a similar approach as Lopez-Paz & Ranzato (2017). We divide both CIFAR10 and CIFAR100 data sets into consecutive-class subsets

Table 3. Results for the compression experiment reported in the main paper: test accuracy A^1 with SGD, test accuracy A^1 after compressing with HAT, and percentage of network weights used after compression.

Data set	Raw A^1	Compressed A^1	Size
CIFAR10	79.9%	80.8%	13.9%
CIFAR100	52.7%	49.1%	21.4%
FaceScrub	82.7%	82.3%	21.0%
FashionMNIST	92.4%	91.9%	2.3%
MNIST	99.5%	99.4%	1.2%
NotMNIST	90.9%	91.5%	5.7%
SVHN	94.2%	93.8%	3.1%
TrafficSigns	97.5%	98.1%	2.9%

Table 4. Wall-clock training time measured on a single NVIDIA Pascal Titan X GPU: total (after learning the 8 tasks), per epoch, and per batch (batches of 64). Batch processing time is measured for a forward pass (Batch-F), and for both a forward and a backward pass (Batch-FB).

Approach	Training time			
	Total [h]	Epoch [s]	Batch-F [ms]	Batch-FB [ms]
PNN	6.0	4.1	10.2	27.5
PathNet	4.5	3.6	10.6	23.9
EWC	3.9	3.1	7.9	19.7
Multitask	3.4	94.8	3.1	15.7
IMM-Mean	3.2	2.6	6.9	17.1
IMM-Mode	3.1	2.5	6.7	16.0
LWF	2.2	2.2	5.7	14.2
HAT	**2.2**	**1.6**	**4.0**	**11.7**
SGD	1.4	0.9	2.5	6.6
LFL	1.3	0.9	4.4	9.2
SGD-F	0.5	0.9	2.5	6.8

and use them as tasks, presented in random order according to the seed. We take groups of 2 classes for CIFAR10 and 20 classes for CIFAR100, yielding a total of 10 tasks. We decide to take groups of 2 and 20 classes in order to have a similar number of training instances per task. The rest of the procedure is as in the main paper. The most important results are summarized there. The complete numbers are depicted in Fig. 2 and reported in Table 5.

Table 5. Average forgetting ratio $\rho^{\leq t}$ for the incremental CIFAR task (10 runs, standard deviation into parenthesis).

Approach	$\rho^{\leq 1}$	$\rho^{\leq 2}$	$\rho^{\leq 3}$	$\rho^{\leq 4}$	$\rho^{\leq 5}$	$\rho^{\leq 6}$	$\rho^{\leq 7}$	$\rho^{\leq 8}$	$\rho^{\leq 9}$	$\rho^{\leq 10}$
LFL	-0.00 (0.01)	-0.53 (0.31)	-0.63 (0.25)	-0.67 (0.21)	-0.70 (0.20)	-0.74 (0.17)	-0.77 (0.15)	-0.79 (0.14)	-0.79 (0.14)	-0.78 (0.14)
LWF	-0.00 (0.02)	-0.10 (0.03)	-0.27 (0.05)	-0.42 (0.05)	-0.50 (0.06)	-0.53 (0.04)	-0.59 (0.06)	-0.64 (0.06)	-0.68 (0.05)	-0.70 (0.05)
SGD-F	-0.00 (0.01)	-0.25 (0.14)	-0.33 (0.16)	-0.35 (0.18)	-0.37 (0.16)	-0.40 (0.18)	-0.41 (0.18)	-0.41 (0.19)	-0.42 (0.19)	-0.43 (0.20)
PathNet	-0.15 (0.31)	-0.18 (0.20)	-0.21 (0.26)	-0.22 (0.28)	-0.24 (0.29)	-0.27 (0.29)	-0.28 (0.30)	-0.30 (0.30)	-0.32 (0.29)	-0.35 (0.28)
SGD	-0.00 (0.01)	-0.19 (0.09)	-0.27 (0.09)	-0.30 (0.04)	-0.30 (0.06)	-0.28 (0.04)	-0.31 (0.03)	-0.32 (0.04)	-0.30 (0.05)	-0.30 (0.04)
IMM-Mean	-0.00 (0.02)	-0.14 (0.08)	-0.21 (0.10)	-0.22 (0.10)	-0.25 (0.10)	-0.26 (0.08)	-0.27 (0.08)	-0.28 (0.08)	-0.29 (0.07)	-0.30 (0.07)
IMM-Mode	-0.00 (0.01)	-0.14 (0.10)	-0.21 (0.11)	-0.23 (0.06)	-0.25 (0.09)	-0.23 (0.07)	-0.26 (0.05)	-0.27 (0.04)	-0.25 (0.04)	-0.25 (0.04)
PNN	-0.26 (0.16)	-0.26 (0.08)	-0.25 (0.05)	-0.23 (0.04)	-0.22 (0.03)	-0.23 (0.03)	-0.22 (0.03)	-0.21 (0.02)	-0.21 (0.02)	-0.21 (0.02)
EWC	-0.00 (0.01)	-0.13 (0.09)	-0.15 (0.08)	-0.16 (0.07)	-0.17 (0.06)	-0.18 (0.06)	-0.19 (0.08)	-0.18 (0.07)	-0.18 (0.06)	-0.18 (0.06)
HAT	**-0.03 (0.04)**	**-0.05 (0.02)**	**-0.05 (0.02)**	**-0.06 (0.01)**	**-0.06 (0.01)**	**-0.07 (0.01)**	**-0.07 (0.01)**	**-0.08 (0.01)**	**-0.08 (0.01)**	**-0.09 (0.01)**

C.2. Permuted MNIST

A common experiment is the one proposed by Srivastava et al. (2013), and later employed to evaluate catastrophic forgetting by Goodfellow et al. (2014). It consists of taking random permutations of the pixels in the MNIST data set as tasks. Typically, the average accuracy after sequentially training on 10 MNIST permutations is reported. To match the different number of parameters used in the literature, we consider a small, medium, and a large network based on a two-layer fully-connected

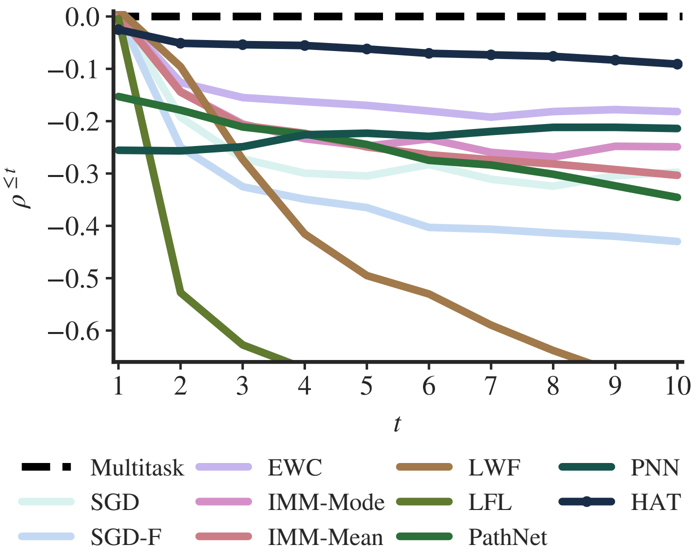

Figure 2. Average forgetting ratio $\rho^{\leq t}$ for the incremental CIFAR task (average after 10 runs).

architecture as Zenke et al. (2017), with 100, 500, and 2000 hidden units, respectively. For the large network we set dropout probabilities as Kirkpatrick et al. (2017). We use $s_{\max} = 200$ and $c = 0.5$ for the small network, and $s_{\max} = 400$ and $c = 0.5$ for the medium and large networks. The results are available in Table 6.

C.3. Split MNIST

Another popular experiment is to split the MNIST data set into tasks and report the average accuracy after learning them one after the other. We follow Lee et al. (2017) by splitting the data set using labels 0–4 and 5–9 as tasks and running the experiment 10 times. We also match the base network architecture to the one used by Lee et al. (2017). We train HAT for 50 epochs with $c = 0.1$. Results are reported in Table 7. In preliminary experiments we observed that dropout could increase accuracy by some percentage. However, to keep the same configuration as in the cited reference, we finally did not use it.

D. Variations to the Proposed Approach

In this section, we want to mention a number of alternatives we experimented with during the development of HAT. The purpose of the section is not the report a formal set of results, but to inform the reader about potential different choices when implementing HAT, or variations of it, and to give an intuition on the outcome of some of such choices.

D.1. Embedding Learning

When we realized that the embedding weights $\mathbf{e}_l^t$ were not changing much and that their gradients were small compared to the rest of the network due to the introduced annealing of s, we initially tackled the issue by using a different learning rate for the embeddings. With that, we empirically found that factors of 10–50 times the original learning rate were leading to performances that were almost as good as the final ones reported in the main paper. However, the use of a different learning rate introduced an additional parameter that we could not conceptually relate to catastrophic forgetting and that could have been tricky to tune for a generic setting.

We also studied the use of an adaptive optimizer such as Adagrad (Duchi et al., 2011) or Adam (Kingma & Ba, 2015) for the embedding weights. The idea was that an adaptive optimizer would be able to automatically introduce an appropriate scaling factor. We found that this option was effectively learning suitable values for $\mathbf{e}_l^t$. However, its performance was worse than the constant-factor SGD boost explained in the previous paragraph. Noticeably, introducing an adaptive optimizer also introduces a number of new hyperparameters: type of optimizer, another learning rate, possible weight decays, etc.

Table 6. Accuracy on the permuted MNIST task (Srivastava et al., 2013), taking the average after training 10 tasks. The only exception is the generative replay approach, whose performance was assessed after 5 tasks. Superscripts indicate results reported by (1) Nguyen et al. (2017) and (2) He & Jaeger (2018). An asterisk after parameter count indicates that the approach presents some additional structure not included in such parameter count (for instance, some memory module or an additional generative network).

APPROACH	PARAMETERS	$A^{\leq 10}$
GEM (LOPEZ-PAZ & RANZATO, 2017)	0.1 M*	82.8%
SI (ZENKE ET AL., 2017)[1]	0.1 M	86.0%
EWC (KIRKPATRICK ET AL., 2017)[2]	0.1 M	88.2%
MBPA + EWC – 1000 EX. (SPRECHMANN ET AL., 2018)	UNKNOWN*	89.7%
VCL (NGUYEN ET AL., 2017)	0.1 M*	90.0%
HAT – SMALL	**0.1 M**	**91.6%**
GENERATIVE REPLAY (SHIN ET AL., 2017)	UNKNOWN*	94.9%
CAB (HE & JAEGER, 2018)	0.7 M	95.2%
EWC (KIRKPATRICK ET AL., 2017)	5.8 M	96.9%
SI (ZENKE ET AL., 2017)	5.8 M	97.1%
HAT – MEDIUM	**0.7 M**	**97.4%**
HAT – LARGE	**5.8 M**	**98.6%**

Table 7. Average accuracy on the split MNIST task, following the setup of Lee et al. (2017) using 10 runs (standard deviation into parenthesis). Superscript (1) indicates results reported by Lee et al. (2017).

APPROACH	PARAMETERS	$A^{\leq 2}$
SGD (GOODFELLOW ET AL., 2014)[1]	1.9 M	71.3% (1.5)
L2-TRANSFER (EVGENIOU & PONTIL, 2004)[1]	1.9 M	85.8% (0.5)
IMM-MEAN (LEE ET AL., 2017)	1.9 M	94.0% (0.2)
IMM-MODE (LEE ET AL., 2017)	1.9 M	94.1% (0.3)
CAB (HE & JAEGER, 2018)	1.9 M	94.9% (0.3)
HAT	**1.9 M**	**99.0% (0.0)**

D.2. Annealing

In our effort to further reduce the number of hyperparameters, we experimented for quite some time with the annealing

$$s = \tan\left(\frac{\pi}{4}\left(1 + \frac{b-1}{B-1}\right)\right)$$

or using variants of

$$s = \alpha + \beta \tan\left(\frac{\pi}{2}\frac{b-1}{B-1}\right).$$

The rationale for the first expression is that one starts with a sigmoid $\sigma(sx)$ that is equivalent to a straight line of 45 degrees for $b = 1$ and $x \approx 0$. Then, with b increasing, it linearly increases the angle towards 90 degrees at $x = 0$. The second expression is a parametric evolution of the first one.

These annealing schedules have the (sometimes desirable) feature that the maximum s is infinite, yielding a true step function in inference time. Therefore, we obtain truly binary attention vectors $\mathbf{a}_l^t$ and no forgetting. In addition, if we use the first expression, we are able to remove the $s_{\max}$ hyperparameter. Nonetheless, we found the first expression to perform worse than the solution proposed in the main paper. The introduction of the second expression with $\alpha = 1$ and $\beta < 1$ improved the situation, but results were still not as good as the ones in the main paper and the tuning of β was a bit tricky.

To conclude this subsection, note that if $s_{\max}$ is large, for instance $s_{\max} > 100$, one can use

$$s = s_{\max}\frac{b-1}{B-1},$$

which is a much simpler annealing formula that closely approximates the one in the main paper. However, one needs then to be careful with the denominator of the embedding gradient compensation when $s = 0$.

D.3. Gate

We also studied the use of alternatives to the sigmoid gate. Apart from the rescaled $\tanh$, an interesting alternative we thought of was a clamped version of the linear function,

$$\mathbf{a}_l^t = \max\left(0, \min\left(1, \frac{s\mathbf{e}_l^t}{r} + \frac{1}{2}\right)\right),$$

where r defines the 'valid' range for the input of the gate. This gate yields a much simpler formulation for the gradient compensation described in the main paper. However, it implies that we need to set r, which could be considered a further hyperparameter. It also implies that embedding values that are far away from 0, the step transition point, receive a proportionally similar gradient to the ones that are close to it. That is, values of $\mathbf{e}_l^t$ that yield $\mathbf{a}_l^t$ that are very close to 0 or 1 (in the constant region of the pseudo-step function) are treated equal to the ones that are still undecided (in the transition region of the pseudo-step function). We did not test this alternative gate quantitatively.

D.4. Cumulative Attention

In the most preliminary stages we used

$$\mathbf{a}_l^{\leq t} = 1 - \left[\left(1 - \mathbf{a}_l^t\right) \odot \left(1 - \mathbf{a}_l^{\leq t-1}\right)\right]$$

for accumulating attention across tasks, but it was soon dismissed for the final $\max$-based formula. The previous equation could be interesting for online learning scenarios with limited model capacity, together with

$$\mathbf{a}_l^{\leq t} = \max\left(\mathbf{a}_l^t, \kappa\, \mathbf{a}_l^{\leq t-1}\right),$$

where κ is a constant slightly lower than 1 (for instance $\kappa = 0.9$ or $\kappa = 0.99$).

D.5. Embedding Initialization

We ran a set of experiments using uniform initialization $\mathcal{U}(0, k_1)$ for the embeddings $\mathbf{e}_l^t$ instead of Gaussian $\mathcal{N}(0, 1)$. We also experimented with $\mathcal{N}(k_2, 1)$. The idea behind these alternative initializations was that, for sufficiently large $s_{\max}$, all or almost all $\mathbf{a}_l^t$ start with a value of 1, which has the effect of distributing the attention over all units for more time at the beginning of training. Using values of $k_1 \in [1, 6]$ and $k_2 \in [0.5, 2]$ yielded competitive results, yet worse than the ones using $\mathcal{N}(0, 1)$. Our intuition is that a uniform initialization like $\mathcal{U}(0, 2)$ is better for a purely compressive approach, as used in the last experiment of the main paper.

D.6. Attention Regularization

We initially experimented with a normalized L1 regularization

$$R\left(\mathsf{A}^t\right) = \frac{\sum_{l=1}^{L-1} \sum_{i=1}^{N_l} a_{l,i}^t}{\sum_{l=1}^{L-1} N_l}.$$

Results were a small percentage lower than the ones with the attention-weighted regularization of the main paper. We also exchanged the previous L1 regularization with the L2-based regularization

$$R\left(\mathsf{A}^t\right) = \frac{\sum_{l=1}^{L-1} \sum_{i=1}^{N_l} (a_{l,i}^t)^2}{\sum_{l=1}^{L-1} N_l}.$$

With that, we observed similar accuracies as the L1 regularization, but under different values for the hyperparameter c.

D.7. Hard Attention to the Input

As mentioned in the main paper, no attention mask is used for the input (that is, there is no $\mathbf{a}_0^t$). We find this is a good strategy for a general image classification problem and for first-layer convolutional filters in particular. However, if the input consists of independent, isolated features, one may think of putting hard attention to the input as a kind of supervised feature selection process. We performed a number of experiments using only fully-connected layers and the MNIST data as above, and introduced additional hard attention vectors $\mathbf{a}_0^t$ that directly multiplied the input of the network. The results suggested that it could potentially be a viable option for feature selection and data compression (Fig. 3).

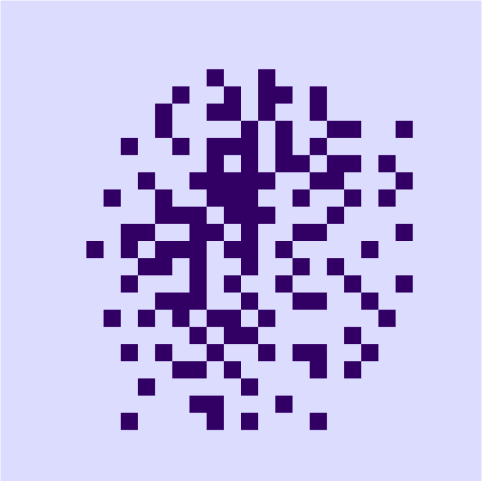

Figure 3. Example of an input mask for MNIST data after training to convergence.

E. A Note on Binary Masks

After writing a first version of the paper, we realized that the idea of a binary mask that affects a given unit could be potentially traced back to the "inhibitory synapses" of McCulloch & Pitts (1943). This idea of inhibitory synapses is quite unconventional and rarely seen today (Wang & Raj, 2017) and, to the best of our knowledge, no specific way for learning such inputs nor a specific function for them have been proposed. Weight-based binary masks are implicitly or explicitly used by many catastrophic forgetting approaches, at least by Rusu et al. (2016); Fernando et al. (2017); Mallya & Lazebnik (2017); Nguyen et al. (2017); Yoon et al. (2018). HAT is a bit different, as it learns unit-based attention masks with possible (but not necessarily) binary values.

References

Bulatov, Y. NotMNIST dataset. Technical report, 2011. URL http://yaroslavvb.blogspot.it/2011/09/notmnist-dataset.html.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Evgeniou, T. and Pontil, M. Regularized multi-task learning. In *Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pp. 109–117, 2004.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. PathNet: evolution channels gradient descent in super neural networks. *ArXiv*, 1701.08734, 2017.

Goodfellow, I., Mizra, M., Da, X., Courville, A., and Bengio, Y. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2014.

He, X. and Jaeger, H. Overcoming catastrophic interference using conceptor-aided backpropagation. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2018.

Kemelmacher-Shlizerman, I., Seitz, S. M., Miller, D., and Brossard, E. The megaface benchmark: 1 million faces for recognition at scale. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 4873–4882, 2016.

Kingma, D. P. and Ba, J. L. Adam: a method for stochastic optimization. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2015.

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., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proc. of the National Academy of Sciences of the USA*, 114(13):3521–3526, 2017.

Krizhevsky, A. *Learning multiple layers of features from tiny images*. Msc thesis, University of Toronto, Toronto, Canada, 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, S.-W., Kim, J.-H., Jun, J., Ha, J.-W., and Zhang, B.-T. Overcoming catastrophic forgetting by incremental moment matching. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 30, pp. 4655–4665. Curran Associates Inc., 2017.

Lopez-Paz, D. and Ranzato, M. A. Gradient episodic memory for continuum learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 30, pp. 6449–6458. Curran Associates Inc., 2017.

Mallya, A. and Lazebnik, S. PackNet: adding multiple tasks to a single network by iterative pruning. *ArXiv*, 1711.05769, 2017.

McCulloch, W. S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning (NIPS-DeepLearning)*, 2011.

Ng, H.-W. and Winkler, S. A data-driven approach to cleaning large face datasets. In *Proc. of the IEEE Int. Conf. on Image Processing (ICIP)*, pp. 343–347, 2014.

Nguyen, C., Li, Y., Bui, T. D., and Turner, R. E. Variational continual learning. *ArXiv*, 1710.10628, 2017.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *ArXiv*, 1606.04671, 2016.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 30, pp. 2993–3002. Curran Associates Inc., 2017.

Sprechmann, P., Jayakumar, S., Rae, J., Pritzel, A., Puigdomènech, A., Uria, B., Vinyals, O., Hassabis, D., Pascanu, R., and Blundell, C. Memory-based parameter adaptation. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2018.

Srivastava, R. K., Masci, J., Kazerounian, S., Gomez, F., and Schmidhuber, J. Compete to compute. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 2310–2318. Curran Associates Inc., 2013.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. The German traffic sign recognition benchmark: a multi-class classification competition. In *Proc. of the Int. Joint Conf. on Neural Networks (IJCNN)*, pp. 1453–1460, 2011.

Wang, H. and Raj, B. On the origin of deep learning. *ArXiv*, 1702.07800, 2017.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *ArXiv*, 1708.07747, 2017.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. Lifelong learning with dynamically expandable networks. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2018.

Zenke, F., Poole, B., and Ganguli, S. Improved multitask learning through synaptic intelligence. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pp. 3987–3995, 2017.