Overcoming Catastrophic Forgetting of Hard Attention Residual Networks

Author: Marius-Constantin Dinu (E10715010)
Professor: Hsing-Kuo Pao

10th January 2019

Motivation

- **Problem statement:** Neural networks forget previously learned tasks when optimizing towards new information (*catastrophic inference* [1] or *catastrophic forgetting*)
- To solve Artificial General Intelligence (AGI) we need to learn tasks in a sequential manner [2]
- Improving connectionist models of memory [3]
- Neural networks should be sensitive to, but not disrupted by, new information ('sensitivity-stability' dilemma [4] or the 'stability-plasticity' dilemma [5])

Motivation (cont'd)

Illustrating catastrophic forgetting when training on multiple tasks in a sequential manner:

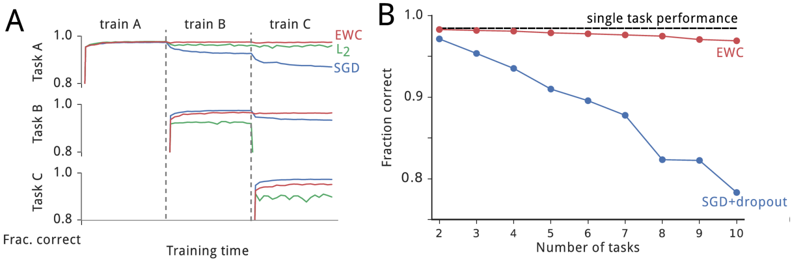


Figure: Elastic Weight Consolidation (EWC) paper [2]

Difficulties

Real-world settings to overcome [6]:

- a sequential learning of tasks, which may not be explicitly labelled
- tasks may switch unpredictably
- individual task may not recur for a long time period

Agents require a capacity for *continual learning*: learn consecutive task without forgetting previously trained tasks.

Current state of the art approaches handle this issue by applying multitask learning.

Multitask Learning Limitations

Given the task to train, an agent can only approach this problem by receiving the data as a **recording** of an **episodic memory system** and **replaying** it during training.

Limitations:

- The amount of memories to store and replay are proportional to the amount of tasks to solve!

General Approaches

- **Rehersal [7]:** storing information and reusing it to retrain the model
  - use of memory modules
  - encounter efficiency and capacity constraints
- **Pseudo-rehersal [7]:** transfer learning to maintain a certain accuracy on the source task
  - memory-free approach
  - recent approaches are generative networks
- **Reduce representational overlap [8]:**
  - can be applied at the input, intermediate and output levels [9] (e.g.: "structural regularization" [10])
  - challenges are to effectively **distribute capacity** of the network **across tasks while maintaining important weights** to reuse previous knowledge

Roadmap

1. Research on related work
2. Use the HAT paper as a baseline and reconstruct the results
3. Implement Residual Network with 18 layers to train on multiple tasks sequentially
4. Apply HAT extensions to ResNet18
5. Compare against baselines from the HAT paper based on the forgetting ratio

Hard Attention to the Task (HAT)

Idea:

- Task-based layer-wise attention mechanism to maintain previous tasks' information
- Learn almost-binary attention vectors through gated task embeddings

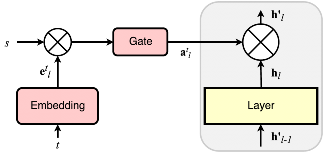


Figure: Forward pass

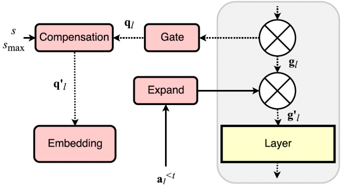


Figure: Backward pass

HAT - Architecture

Component-wise multiply:

$$\boldsymbol{h}_l' = \boldsymbol{a}_l^t \odot \boldsymbol{h}_l$$

- Instead of forming a probability distribution, $\boldsymbol{a}_l^t$ is a gated version of a task embedding $\boldsymbol{e}_l^t$ at layer l for the current task t
- $\boldsymbol{a}_l^t = \sigma(s\boldsymbol{e}_l^t)$, where $\sigma(x) \in [0, 1]$ and s is a positive scaling factor
- All layers $l = 1, \ldots, L-1$ operate equally except of the last layer L, where $\boldsymbol{a}_L^t$ is binary hard-coded

HAT - Architecture (cont'd)

- The attention mask dynamically creates or destroys pathways similar to PathNet, which can be preserved when learning new tasks
- Unlike PathNet, HAT does not rely on modules and does not require to pre-assign a module size
- It learns individual unit paths and automatically dimensions their total number to the task
- Does not require a second stage learning (e.g.: genetic algorithms) - it implicitly learns the paths with the rest of the network

HAT - Training

HAT conditions the weights' gradients to the cumulative attention vector after learning task t and obtaining $\boldsymbol{a}_l^t$ to compute

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

using component-wise maximum and the all-zero vector for $\boldsymbol{a}_l^0$.

- Via this recursion, the attention for previous learned tasks is preserved, which are also conditioning future tasks

HAT - Training (cont'd)

To condition the training tasks $t+1$ the gradients $g_{l,i,j}$ are modified at layer l using the reverse of the minimum of the attention at the current and previous layer:

$$g'_{l,i,j} = \left[1 - \max\left(a_{l,i}^{\leq t}, a_{l-1,j}^{\leq t}\right)\right] g_{l,i,j}$$

where i and j correspond to the output (layer l) and input (layer $l-1$) units.

- This basically expands the vectors $\boldsymbol{a}_l^{\leq t}$ and $\boldsymbol{a}_{l-1}^{\leq t}$ to match the dimensions of the gradient tensor of the corresponding layer
- No attention is computed for the input layer ($l=1$)
- This also constraints the gradients to prevent large updates of weights for previous tasks that where important (similar to PackNet, but without heuristics selection and retraining in a post-training step)

HAT - Pseudo-Step Function

To get a fully differentiable mask a pseudo-step function is defined:

$$s = \frac{1}{s_{\max}} + \left(s_{\max} - \frac{1}{s_{\max}}\right) \frac{b-1}{B-1}$$

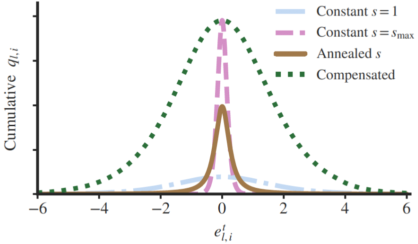
- $b = 1, \ldots, B$ is the batch index
- B is the total number of batches in an epoch
- $s_{\max} \geq 1$ hyperparameter controls the plasticity of the network's units
- If $s_{\max}$ is close to 1, the gating function operates like a sigmoid function, allowing the model to forget previous learned tasks
- If $s_{\max}$ is very large (e.g.: 100) it operates like a step function, with $a^t_{l,i} \rightarrow \{0, 1\}$, preventing changes in the backpropagation stage

HAT - Embeddings Gradient Compensation

Empirically results showed that embeddings $\boldsymbol{e}_l^t$ did not change much, due to annealing effects of s. This was corrected by defining:

$$q'_{l,i} = \frac{s_{\max}[\cosh(se^t_{l,i}) + 1]}{s[\cosh(e^t_{l,i}) + 1]} q_{l,i}$$

- For numerical stability $|se^t_{l,i}| \leq 50$ and $e^t_{l,i} \in [-6, 6]$



HAT - Promoting Low Capacity Usage

Promote sparsity on the set of attention vectors $A^t = \{\boldsymbol{a}_1^t, \ldots, \boldsymbol{a}_{L-1}^t$ by adding regularization to the loss function $\mathcal{L}$:

$$\mathcal{L}'(\boldsymbol{y}, \hat{\boldsymbol{y}}, A^t) = \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) + cR(A^t)$$

- c is the regularization constant and defines the capacity spend on each task
- $R(A^t) = \frac{\sum_{l=1}^{L-1} \sum_{i=1}^{N_l} a_{l,i}^t}{\sum_{l=1}^{L-1} N_l}$ is a normalized L1 regularization over the attention values $a_{l,i}^t$
- N_l is the number of units in layer l
- The regularization sparseness is similar to DEN, but without heuristics and applies in a single training phase

HAT - Limitations

- The network is not immune to catastrophic forgetting
- Network capacity defines the limit for the tasks that can be learned
- Embeddings have to be manually specified for inference
- Includes some "hackish" steps to enforce correct learning bahavior

1 Overview
- Motivation

2 Related Work
- Hard Attention to the Task (HAT)

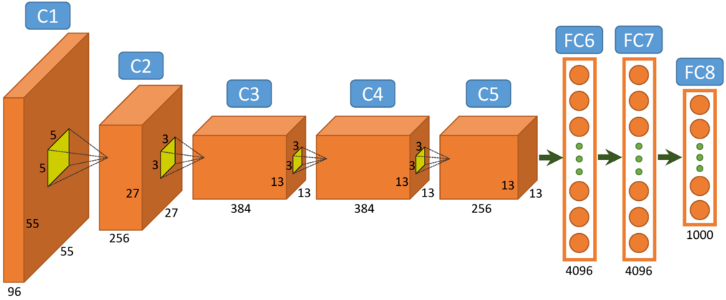
3 Concept
- AlexNet
- ResNet

4 Experiments
- Dataset
- Training

5 Conclusions

6 Appendix
- Bibliography

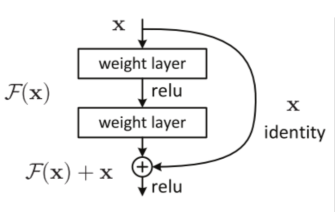
AlexNet

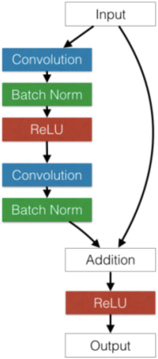

C1
C2
C3
C4
C5
FC6
FC7
FC8
5
5
55
55
96
3
3
27
27
256
3
3
13
13
384
384
256
4096
4096
1000

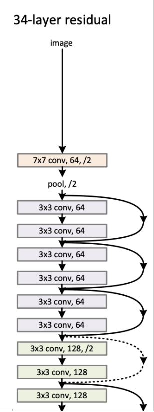
AlexNet with HAT

```
# definition
self.c1 = torch.nn.Conv2d(ncha, 64, kernel_size=size // 8)
self.ec1 = torch.nn.Embedding(len(self.taskcla), 64)
self.drop1 = torch.nn.Dropout(0.2)
self.maxpool = torch.nn.MaxPool2d(2)
self.relu = torch.nn.ReLU()
self.gate = torch.nn.Sigmoid()
...
# forward pass
gc1 = self.gate(s * self.ec1(t))
h = self.maxpool(self.drop1(self.relu(self.c1(x))))
h = h * gc1.view(1, -1, 1, 1).expand_as(h)
```

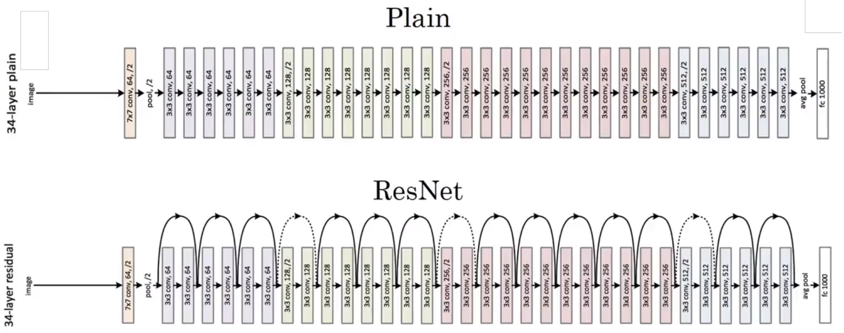
ResNet


x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu


Input
Convolution
Batch Norm
ReLU
Convolution
Batch Norm
Addition
ReLU
Output


34-layer residual
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 128

ResNet (cont'd)



Difficulties

- Handle Residual Skip Connections to ensure gradient flow
- Handle convolutional layer weights for programming cumulative attention
- Greater depth requires stabilizing the activations
- Apply batch normalization to ensure unit variance and zero mean for activations
- Found out that we require task dependent batch normalization

Training Set

- 2 different image data sets
- Adapt input to a size of 32x32x3 pixels by resizing, zero padding, or replicating values
- Number of classes goes from 10 to 100, training set sizes from 16,853 to 73,257 and test set sizes from 1,873 to 26,032
- Each task randomly splits 15 % of size for validation purpose
- Dataset: Randomly break tasks from CIFAR10, CIFAR100

Experiments

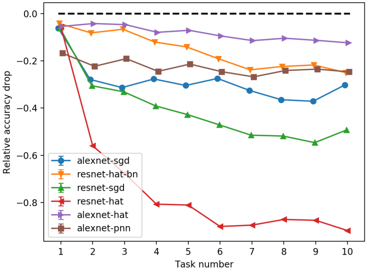
- **AlexNet-SGD:** AlexNet - Standard SGD and Dropout [14]
- **AlexNet-PNN:** AlexNet - Progressive Neural Networks [15]
- **AlexNet-HAT:** AlexNet - Hard Attention to the Task
- **ResNet18-Joint** Residual Network 18 Layers - SGD and Dropout with all Datasets
- **ResNet18-SGD:** ResNet18 - Standard SGD
- **ResNet18-HAT:** ResNet18 - Hard Attention to the Task
- **ResNet18-HAT-BN:** ResNet18 with HAT and Batch Normalization

Training

- Architectures: AlexNet, ResNet18
- All layers are randomly initialized with Xavier uniform initialization except the embedding layers (Gaussian distribution $\mathcal{N}(0,1)$)
- All baseline approaches where adapted to match the number of parameters to 7.1 M for AlexNet and 11.2 M for ResNet18
- Training with plain SGD with a learning rate of 0.05 and decay it by a factor of 3 if no improvements over 5 epochs
- Stopping criteria: Either 200 epochs or learning rate below 10^{-4}

Task Comparison

Forgetting ratio: $p^{\tau \leq t} = \frac{A^{\tau \leq t} - A_R^{\tau}}{A_J^{\tau \leq t} - A_R^{\tau}} - 1$, whereas $A^{\tau \leq t}$ is the accuracy measured on task τ after sequentially learning task t, A_R^{τ} is the accuracy of a random, frequency-based classifier solely trained on task τ, and $A_J^{\tau \leq t}$ is the accuracy measured on task τ after jointly learning all t tasks in a multitask fashion.



Conclusions

- We require task dependent batch normalization
- The deeper the network becomes the harder it gets to train pseudo-binary masks to match minimal overlapping neurons per layer
- Handling parallel branches increases complexity on finding a unique mask instance

- McCloskey, M. & Cohen, N. (1989) Catastrophic interference in connectionist networks: The sequential learning problem. In G. H. Bower (ed.) The Psychology of Learning and Motivation,24, 109-164
- James Kirkpatricka, Razvan Pascanua, Neil Rabinowitza, Joel Venessa, Guillaume Desjardinsa, Andrei A. Rusua, Kieran Milana, John Quana, Tiago Ramalhoa, Agnieszka Grabska-Barwinska, Demis Hassabisa, Claudia Clopathb, Dharshan Kumarana, Raia Hadsella (2017) Overcoming catastrophic forgetting in neural networks, arXiv:1612.00796
- Ratcliff, R. (1990) Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. Psychological Review,97, 285-308
- Hebb, D.O. (1949). Organization of Behaviour. New York: Wiley

- Caroebterm G., & Grossberg, S. (1987) ART 2: Self-organization of stable category recognition codes for analog input patterns. Applied Optics, 26, 4919-4930
- Shane Legg and Marcus Hutter (2007) Universal intelligence: A definition of machine intelligence. Minds and Machines, 17(4):391-444
- Robins, Anthony (1995) Catastrophic Forgetting, Rehearsal, and Pseudorehearsal. Connection Science, 7:123?146
- French, Robert M. (1991) Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In Proc. of the Annual Conf. of the Cognitive Science Society (CogSci), pp. 173?178, 1991
- He, X. and Jaeger, H. (2017) Overcoming catastrophic interference by conceptors. ArXiv, 1707.04853

- Zenke, F., Poole, B., and Ganguli, S. (2017) Improved multitask learning through synaptic intelligence. In Proc. of the Int. Conf. on Machine Learning (ICML), pp. 3987?3995
- Guang Yang, Feng Pan, and Wen-Biao Gan (2009) Stably maintained dendritic spines are associated with lifelong memories. Nature, 462(7275):920-924
- Marcus K Benna and Stefano Fusi (2016) Computational principles of synaptic memory consolidation. Nature neuroscience
- Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, Raia Hadsell (2016) Progressive Neural Networks. arXiv:1606.04671

Goodfellow, I., Mizra, M., Da, X., Courville, A., and Bengio, Y. (2014) An empirical investigation of catastrophic forgetting in gradient-based neural networks. In Proc. of the Int. Conf. on Learning Representations (ICLR). arXiv:1312.6211v3

Lee, S.-W., Kim, J.-H., Jun, J., Ha, J.-W., and Zhang, B.-T. (2017) 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. arXiv:1703.08475v3

Li, Z. and Hoiem, D. (2017) Learning without forgetting. IEEE Trans. on Pattern Analysis and Machine Intelligence, PP (99):1?1.. arXiv:1606.09282v3

- Jung, H., Ju, J., Jung, M., and Kim, J. (2016) Less-forgetting learning in deep neural networks. arXiv:1607.00122v1
- Sergey Ioffe, Christian Szegedy - *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, 2015*, arXiv:1502.03167
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun - *Deep Residual Learning for Image Recognition, 2015*, arXiv:1512.03385
- Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton - *ImageNet Classification with Deep Convolutional Neural Networks, 2012*, Advances in Neural Information Processing Systems 25, page 1097-1105, Curran Associates, Inc.