Continual Lifelong Learning

Jonas Wildberger
University of Oxford

Outline

Problem Description

- State-of-the art ANNs rival human performance in a variety of domain-specific tasks

Machine vs. Human learning

- State-of-the art ANNs rival human performance in a variety of domain-specific tasks
- Inspired by human learning, yet substantially different
  - Machine: Parameters used statically on new data
  - Human: On-the-fly update of memories and beliefs

Machine vs. Human learning

- State-of-the art ANNs rival human performance in a variety of domain-specific tasks
- Inspired by human learning, yet substantially different
  - Machine: Parameters used statically on new data
  - Human: On-the-fly update of memories and beliefs
- Necessity to retrain on entire dataset to avoid overfitting and catastrophic forgetting

Machine vs. Human learning

- State-of-the art ANNs rival human performance in a variety of domain-specific tasks
- Inspired by human learning, yet substantially different
  - Machine: Parameters used statically on new data
  - Human: On-the-fly update of memories and beliefs
- Necessity to retrain on entire dataset to avoid overfitting and catastrophic forgetting

How do we enable learning in an online fashion for ANNs?

- Performance on previous task catastrophically deteriorates when new task is learned

Catastrophic Forgetting I

- Performance on previous task catastrophically deteriorates when new task is learned
- Stability vs. Plasticity

Catastrophic Forgetting I

- Performance on previous task catastrophically deteriorates when new task is learned
- Stability vs. Plasticity
- Presumably caused by static weights

Catastrophic Forgetting I

- Performance on previous task catastrophically deteriorates when new task is learned
- Stability vs. Plasticity
- Presumably caused by static weights

Problem: Minimise total loss function without access to loss function of previous tasks

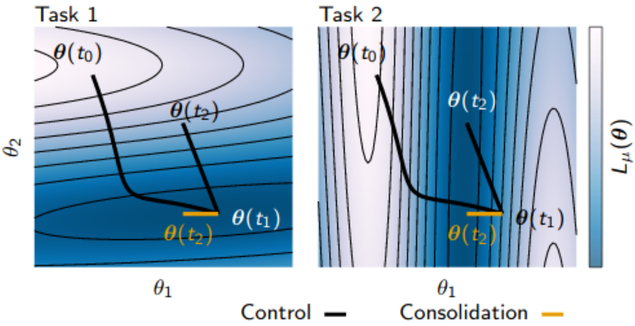


Figure: Illustration of Catastrophic Forgetting [ZPG17]

1. Architectural approach
   - Alter architecture of network, e.g. freezing weights, changing learning rate for layers shared with original task, feature augmentation

1. Architectural approach
   - Alter architecture of network, e.g. freezing weights, changing learning rate for layers shared with original task, feature augmentation
2. Functional approach
   - Add regularisation term that penalises changes in input-output function, i.e. predictions are similar across tasks
   - Computationally expensive: For every new data point, compute forward pass through old task's network

3. Structural approach
   - Penalties on the parameters s.t. they remain close to values for old task
   - Mimic complexity of biological synapses
   - Retain task relevant information and measure of importance

Elastic Weight Consolidation

Let $\mathcal{D}_A, \mathcal{D}_B$ be two independent tasks $\mathcal{D}_A \cup \mathcal{D}_B = \mathcal{D}$

$$\log p(\theta|\mathcal{D}) = \log p(\mathcal{D}|\theta) + \log p(\theta) - \log p(\mathcal{D}) \tag{1}$$

Independence of $\mathcal{D}_A, \mathcal{D}_B$ gives

$$\log p(\theta|\mathcal{D}) = \underbrace{\log p(\mathcal{D}_B|\theta)}_{=-\mathcal{L}_B} + \log p(\theta|\mathcal{D}_A) - \log p(\mathcal{D}_B) \tag{2}$$

All information (including which parameters are important) about task A absorbed in posterior $p(\theta|\mathcal{D}_A)$.

Loss function

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2 \qquad (3)$$

λ: Compares importance of task A to task B, F_i: Diagonal of Fisher information matrix

Loss function

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2 \tag{3}$$

λ: Compares importance of task A to task B, F_i: Diagonal of Fisher information matrix

- Posterior $p(\theta|\mathcal{D}_A)$ approximated by F

Loss function

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2 \tag{3}$$

λ: Compares importance of task A to task B, F_i: Diagonal of Fisher information matrix

- Posterior $p(\theta|\mathcal{D}_A)$ approximated by F
- Iteratively applicable to more than 2 tasks

Continual Learning Through Synaptic Intelligence

Idea I

How does learning task k change the total loss? Let $\boldsymbol{g} = \partial_{\boldsymbol{\theta}}\mathcal{L}$:

$$\int_C \boldsymbol{g}(\boldsymbol{\theta}(t))d\boldsymbol{\theta} = \int_{t_0}^{t_1} \boldsymbol{g}(\boldsymbol{\theta}(t)) \cdot \boldsymbol{\theta}'(t)dt \tag{4}$$

$$= \sum_\mu \sum_k \int_{t^{\mu-1}}^{t^\mu} g_k(\theta(t))\theta'_k(t)dt \tag{5}$$

$$= -\sum_\mu \omega_k^\mu \tag{6}$$

ω_k^μ contribution of μth task and kth parameter to change in total loss

UNIVERSITY OF OXFORD

Idea II

- Update ω_k^μ online as running sum

$$\sum_t \partial_{\theta_k} \mathcal{L}(t) \cdot \partial_t \theta_k(t)$$

Idea II

- Update ω_k^μ online as running sum

$$\sum_t \partial_{\theta_k}\mathcal{L}(t)\cdot\partial_t\theta_k(t) \tag{7}$$

- Importance of θ_k determined by
  1. contribution of θ_k to drop in Loss ω_k^ν
  2. How much it changed $\theta_k(t^\nu) - \theta_k(t^{\nu-1}) = \Delta_k^\nu$

- Update ω_k^μ online as running sum

$$\sum_t \partial_{\theta_k} \mathcal{L}(t) \cdot \partial_t \theta_k(t)$$

- Importance of θ_k determined by
  1. contribution of θ_k to drop in Loss ω_k^ν
  2. How much it changed $\theta_k(t^\nu) - \theta_k(t^{\nu-1}) = \Delta_k^\nu$

  For current task μ:

$$\Omega_k^\mu = \sum_{\nu<\mu} \frac{\omega_k^\nu}{(\Delta_k^\nu)^2 + \xi} \tag{7}$$

Loss function

$$\tilde{\mathcal{L}}_\mu(\theta) = \mathcal{L}_\mu(\theta) + \underbrace{c \sum_k \Omega_k^\mu \left(\theta_k(t^{\mu-1}) - \theta_k\right)^2}_{\text{surrogate loss}} \tag{8}$$

c strength parameter trading off old vs. new memories

Loss function

$$\tilde{\mathcal{L}}_\mu(\theta) = \mathcal{L}_\mu(\theta) + \underbrace{c\sum_k \Omega_k^\mu \left(\theta_k(t^{\mu-1}) - \theta_k\right)^2}_{\text{surrogate loss}} \qquad (8)$$

c strength parameter trading off old vs. new memories

- Surrogate loss approximates summed loss functions of previous tasks
  - Same minimum as previous parameter configuration
  - Same ω_k^ν over Δ_k

$$\tilde{\mathcal{L}}_\mu(\theta) = \mathcal{L}_\mu(\theta) + \underbrace{c\sum_k \Omega_k^\mu \left(\theta_k(t^{\mu-1}) - \theta_k\right)^2}_{\text{surrogate loss}} \tag{8}$$

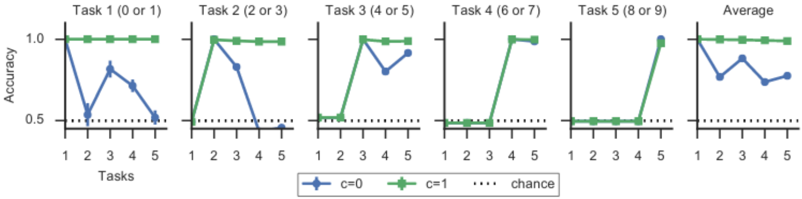
c strength parameter trading off old vs. new memories

- Surrogate loss approximates summed loss functions of previous tasks
  - Same minimum as previous parameter configuration
  - Same ω_k^ν over Δ_k
- Derivation only valid for two task; but empirically works for more

Comparison and Outlook

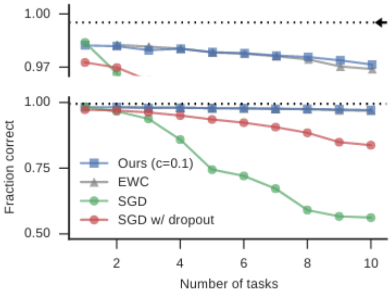
Split MNIST

- Divide MNIST into 5 subsets of consecutive digits; learn to distinguish between two consecutive digits



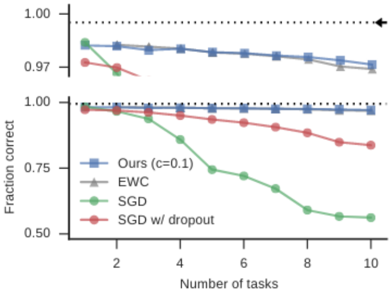
Permuted MNIST

- Randomly permute all MNIST pixels for a task
- Performance measured by correctness across all tasks



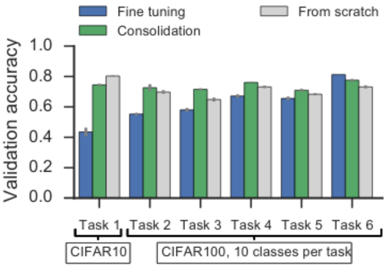
Permuted MNIST

- Randomly permute all MNIST pixels for a task
- Performance measured by correctness across all tasks
- Correlations of ω_k^μ decrease across different tasks μ: "Using different weights to learn new tasks"



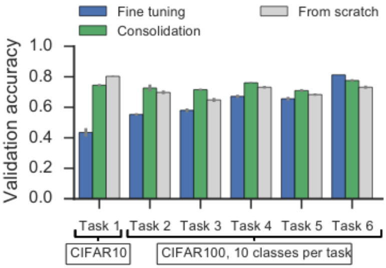
Split CIFAR-10/CIFAR-100

- First image classification on CIFAR-10, then 5 additional tasks corresponding to 10 consecutive classes from CIFAR-100



Split CIFAR-10/CIFAR-100

- First image classification on CIFAR-10, then 5 additional tasks corresponding to 10 consecutive classes from CIFAR-100
- Better validation accuracy than networks trained on single task only: Less prone to overfitting



- Penalising changes to most important synapses can alleviate catastrophic forgetting

Discussion

- Penalising changes to most important synapses can alleviate catastrophic forgetting
- EWC:
  - Offline, point estimate of posterior probability
  - Computing diagonal of Fisher has linear complexity in number of data points

- Penalising changes to most important synapses can alleviate catastrophic forgetting
- EWC:
  - Offline, point estimate of posterior probability
  - Computing diagonal of Fisher has linear complexity in number of data points
- Synaptic Intelligence
  - Online estimate over entire learning trajectory
  - Doesn't scale naturally to multiple tasks

- To what extent does this help to decrease the number of examples needed for learning new tasks?

Outlook and Questions

- To what extent does this help to decrease the number of examples needed for learning new tasks?
- Fixing the hyperparameters λ, c is quite expensive. Can we adaptively learn or predict them based on a priori knowledge about new task?

- To what extent does this help to decrease the number of examples needed for learning new tasks?
- Fixing the hyperparameters λ, c is quite expensive. Can we adaptively learn or predict them based on a priori knowledge about new task?
- Can we cluster important weights and use other sparsity regularisation strategies such as group Lasso?

In addition to adding depth to our networks, we may need to add intelligence to our synapses.

References

Friedemann Zenke, Ben Poole, and Surya Ganguli.
Continual learning through synaptic intelligence, 2017.