Lifelong Learning with Dynamically Expandable Networks

Octave Mariotti & Rémy Portelas,
ICLR Reproducibility Challenge 2018

SORBONNE UNIVERSITÉS

Context

Lifelong learning[1] is a learning paradigm in which the model must learn tasks one after the other and tries to transfer knowledge from old tasks to new ones. This incremental deep learning setting raises challenging problems both in terms of scalability and efficiency.

A **Dynamically Expandable Network** (DEN) is a novel deep architecture meant to address these issues[2]. The idea is to dynamically expand the network to learn a compact overlapping knowledge structure while limiting semantic drift.

In this review we will give an **in-depth description** of the authors' DEN model. We will then show **our results** obtained by reproducing their experiment on performance using a variation of the MNIST dataset. Finally, we will **discuss** the limitations of this paper.

The DEN model

The DEN model combines several ideas from the literature in order to create a learning procedure able to automatically accommodate new tasks by adding neurons while retaining knowledge from the previous ones.

Algorithm 1: Base DEN algorithm

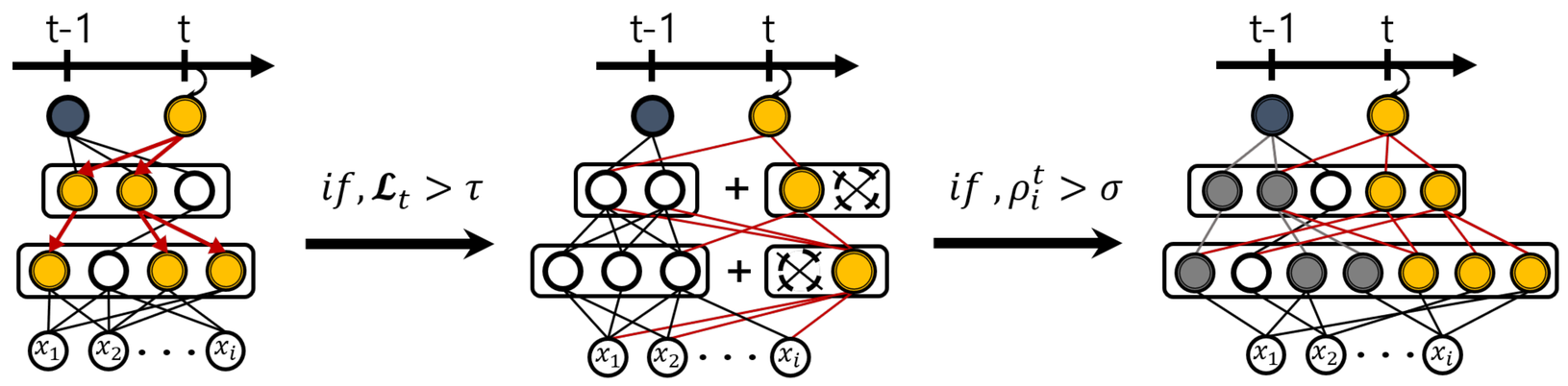
input : Dataset $D = (D_1, \ldots, D_T)$
output : Network parameters W_T

```
for t in 1,...,T do
    if t == 1 then
        train network with ℓ1 regularization;
    else
        W_sr = Selective_retraining(W_{t-1});
        if L > τ then
            W_ne = Network_expansion(W_sr);
        W_t = Duplication(W_ne);
```

Experimental Reproduction

Our experiments (box below on the right) show poor DEN performances along with unexpectedly high baselines, both averaged over five random splits.
Several discrepancies were observed between released code and the algorithm described in the article.

DEN Architecture



The 3 steps of a DEN: **Selective Retrain** (*left*), **Dynamic Network Expansion** (*middle*), **Duplication** (*right*).

Selective retraining

When a new tasks arrives, we want to learn it with the help of the knowledge from previous tasks. The **Selective retraining** algorithm is tasked with selecting neurons that are useful for this new task and training them.

Algorithm 2: Selective Retraining

input : Dataset D_t, previous parameters W_{t-1}
output : Network parameters W_{sr}

```
Retrain last layer only with ℓ1 regularization;
Perform breadth-first search in network to
    select useful neurons;
Train subnetwork with ℓ2 regularization;
```

Since the network is sparse, this algorithm only retrains neurons that are connected to the output neuron associated with the new task, preventing negative transfer.

Dynamic Network Expansion

If the loss is not low enough after **Selective Retraining**, then we must increase the network's size to improve our performances on the current task.

Algorithm 3: Dynamic Expansion

input : Dataset D_t,Loss $\mathcal{L}$
output : Expanded Network W_{ne}

```
if L > τ then
    Add k units h^N at all layers;
    Train network with ℓ1 and ℓ_gs;
forall layer l do
    Remove useless units in h_l^N;
```

The combined use of ℓ_1-norm and **group sparse regularization** ℓ_{gs} (a group being all incoming connections of a neuron) allows us to drop unnecessary neurons, effectively keeping the network both accurate and compact.

Duplication

In lifelong learning, an important challenge is to prevent **catastrophic forgetting**, that is forgetting how to perform old tasks when learning new ones. The **duplication** algorithm selects neurons in the network that have changed from the last iteration and duplicates them in order to preserve performances on both the new and the old task.

Algorithm 4: Neuron Duplication

input : Dataset D_t, previous parameters W_{t-1}
output : Network parameters W_t

```
Retrain network with ℓ_d regularization;
forall neuron n do
    ρ_n = ||w_{n,t} − w_{n,t−1}||_2;
    if ρ_n > σ then
        Duplicate n
Retrain network with ℓ_d regularization;
```

The ℓ_d regularization is the **drift**, defined as $||W_t - W_{t-1}||_2$. It characterizes how much parameters have changed since the previous task.

Timestamped Inference

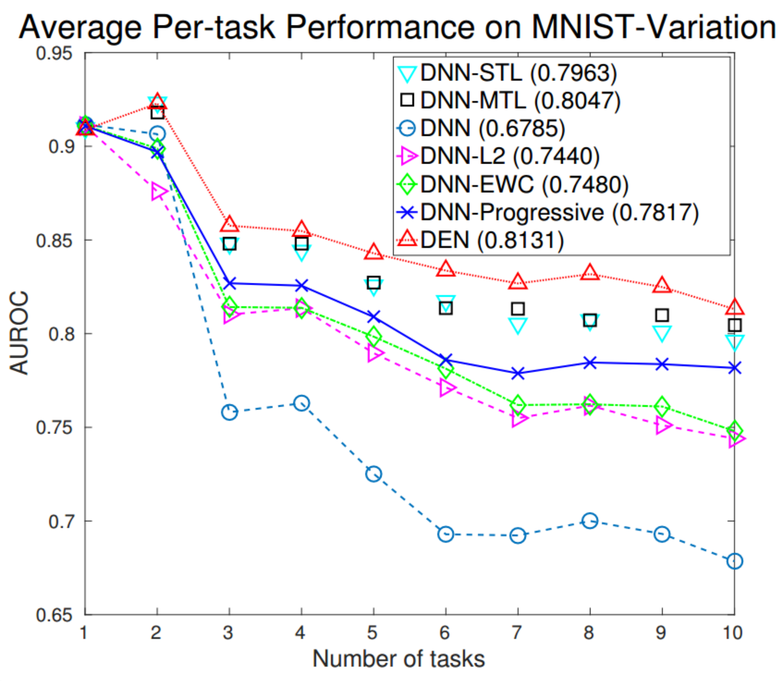
Because added neurons are unrelated to previous task, they could add noise when performing inference. To prevent this, we add a timestamp at creation and disregard neurons with higher timestamps during inference.

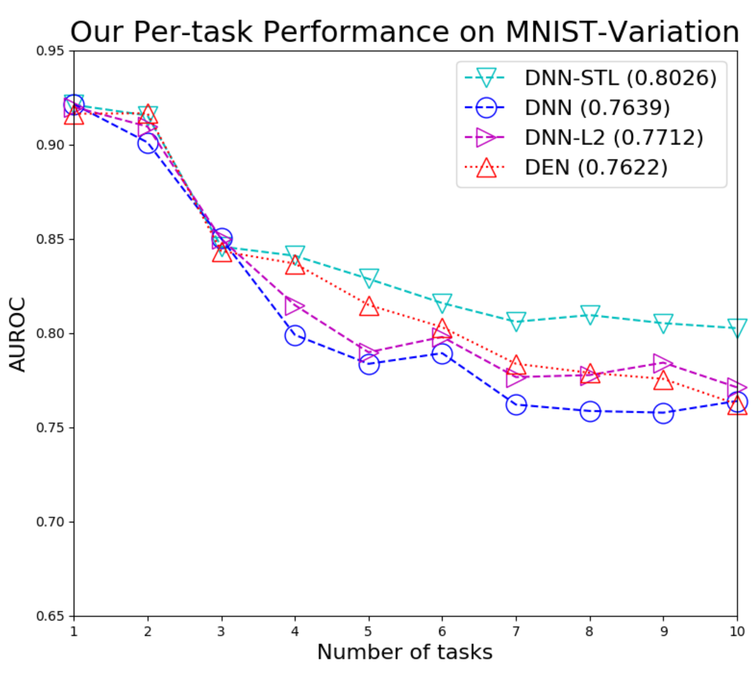
Discussion

This article proposes intuitive approaches on lifelong learning issues, but lacks detailed explanation of some keypoints.

- Hyperparameters values undisclosed
- Network sparsity is not guaranteed, contrary to what is claimed
- Questionable design of duplication (different from released code)
- MNIST dataset seems inadequate to test performances of lifelong learning models
- Dubious evaluation metric

Results





References

[1] Sebastian Thrun. A lifelong learning perspective for mobile robot control. In *In Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*. Elsevier, 1994.

[2] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung ju Hwang. Lifelong learning with dynamically expandable networks, 2017.