Neural Networks 92 (2017) 17–28

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2017 Special Issue

Dual-memory neural networks for modeling cognitive activities of humans via wearable sensors

Sang-Woo Lee [a], Chung-Yeon Lee [a], Dong-Hyun Kwak [b], Jung-Woo Ha [c], Jeonghee Kim [c], Byoung-Tak Zhang [a,b,d,*]

[a] School of Computer Science and Engineering, Seoul National University, Seoul 08826, South Korea
[b] Interdisciplinary Program in Neuroscience, Seoul National University, Seoul 08826, South Korea
[c] NAVER LABS, NAVER Corp., Bundang 13561, South Korea
[d] Surromind Robotics, 1 Gwanak-ro Gwanak-gu, Seoul 08826, South Korea

ARTICLE INFO

Article history:
Available online 20 February 2017

Keywords:
Dual memory architecture
Complementary learning systems
Lifelog dataset
Online learning
Deep neural networks
Hypernetworks

ABSTRACT

Wearable devices, such as smart glasses and watches, allow for continuous recording of everyday life in a real world over an extended period of time or lifelong. This possibility helps better understand the cognitive behavior of humans in real life as well as build human-aware intelligent agents for practical purposes. However, modeling the human cognitive activity from wearable-sensor data stream is challenging because learning new information often results in loss of previously acquired information, causing a problem known as catastrophic forgetting. Here we propose a deep-learning neural network architecture that resolves the catastrophic forgetting problem. Based on the neurocognitive theory of the complementary learning systems of the neocortex and hippocampus, we introduce a dual memory architecture (DMA) that, on one hand, slowly acquires the structured knowledge representations and, on the other hand, rapidly learns the specifics of individual experiences. The DMA system learns continuously through incremental feature adaptation and weight transfer. We evaluate the performance on two real-life datasets, the CIFAR-10 image-stream dataset and the 46-day Lifelog dataset collected from Google Glass, showing that the proposed model outperforms other online learning methods.

1. Introduction

1.1. Wearable devices and the Lifelog dataset

Learning human behaviors in real-world settings is crucial for understanding human cognition in real life as well as for building human-aware intelligent systems such as personalized digital assistants. Recently, various types of wearable devices, including smart watches and Google Glass, have become popular. These devices can see and hear what the device user sees and hears; which differentiates them from classical agents such as personal computers or smartphones. To simulate the real-world environment, we collected a Lifelog dataset using Google Glass from three participants over 46 days. This dataset has two properties. First, high-level context is hidden in the raw data stream; e.g., an egocentric video recorded during a meeting includes various types of high-level contexts, although the data is only a stream of pixels and audio signals. Second, the data streams are often non-stationary; e.g., the life patterns of a student during the holiday and school term are different. We are interested in continually adapting the context-aware activity recognizer rapidly from human behavior gathered through wearable devices. Two algorithmic techniques are required to manage this task. First, a deep learning method is necessary to manage raw data efficiently. Second, an online learning algorithm is required to keep track of fast-changing life patterns of user behavior.

1.2. Online learning of deep neural networks and catastrophic forgetting

Online learning of deep neural networks (DNNs) that cover a lifetime is challenging, but learning over long time periods is fundamental to developing a real-time personalized recognizer. Assume that you trained a neural network using a first training, which was user data from the first week. Subsequently, a second

* Corresponding author at:School of Computer Science and Engineering, Seoul National University, Seoul 08826, South Korea.
E-mail address: btzhang@bi.snu.ac.kr (B.-T. Zhang).

http://dx.doi.org/10.1016/j.neunet.2017.02.008
0893-6080/

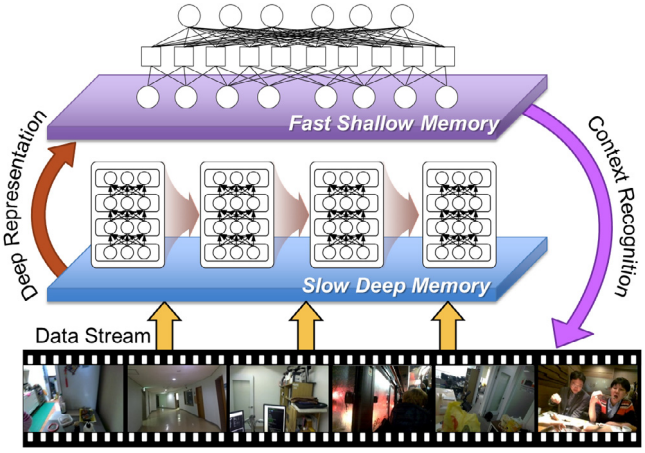

Fig. 1. Learning framework proposed in this work.

training dataset becomes available and is user data from the second week. You can train the neural network with the second training dataset; however, the information from the first training dataset may be lost, especially when the data stream is non-stationary. In short, when new data becomes available, the neural network often forgets old data; this phenomenon is known as catastrophic forgetting (Goodfellow, Mirza, Xiao, Courville, & Bengio, 2013).

Various approaches have been proposed for online learning of DNNs. Recently, online fine-tuning of convolutional neural networks (CNNs) using a simple online stochastic gradient descent (SGD) was successful at inferring a visual tracking task (Nam & Han, 2016). However, this ad hoc method does not guarantee the retention of old data. Several studies have adopted an incremental ensemble learning approach, whereby a weak learner is made to use the online dataset, and these multiple weak learners are combined to obtain a better predictive performance (Polikar, Upda, Upda, & Honavar, 2001). Unfortunately, in our experiment, a simple voting method with weak learners trained from a relatively small online dataset was unsuccessful; we presumed that a relatively small online dataset is insufficient for learning highly expressive representations of DNNs.

1.3. Complementary learning systems theory

To solve the problem of catastrophic forgetting, we apply the concept of complementary learning systems (CLS) theory; a framework based on the idea of a dual learning system structure in the brain (McClelland, McNaughton, & O'Reilly, 1995; O'Reilly, Bhattacharyya, Howard, & Ketz, 2014). According to the CLS theory, there are two critical areas in the brain that affect online learning: the neocortex and the hippocampus, with complementary functions. From the machine learning perspective, the neocortex is analogous to a deep neural module that can gradually learn to extract structure from real-world sensor streams (Guyonneau, VanRullen, & Thorpe, 2004). Further, corroborative evidence from cognitive neuroscience shows that the behavior of performance-optimized deep CNNs closely match the neural responses in the higher visual cortex of the brain in monkeys (Yamins et al., 2014). However, the key limitations introduced by the idea of online learning are that the neocortex does not rapidly learn a new concept in a single attempt nor does it process data at the instance-level to weigh specific events appropriately. In contrast, the hippocampus alleviates this problem by allowing rapid and individuated storage to memorize a new instances (Knierim & Neunuebel, 2016; Treves & Rolls, 1992). There is evidence that the hippocampus allows general statistics of the environment to be circumvented by weighting procedures such that statistically unusual but significant events may be afforded a privileged status (Bendor & Wilson, 2012; Carr, Jadhav, & Frank, 2011). However, a hippocampal system alone would be insufficient for continuous learning because of limitations on memory capacity and generalization ability.

1.4. Dual memory architecture

To address the issues of a neocortex-like or hippocampal-like system alone, we propose a dual memory architecture (DMA) (Fig. 1). The DMA trains two memory structures: one is an ensemble of DNNs, and the other consists of a shallow network that uses hidden representations of the DNNs as input. These two memory structures are designed to use different strategies. The ensemble of DNNs learns new information to adapt its representation to new data, whereas the shallow network aims to manage non-stationary distribution and unseen classes more rapidly.

Some additional techniques for online deep learning are incorporated in this study. First, the transfer learning method via weight transfer is applied to maximize the representation power of each neural module in online deep learning (Yosinski, Clune, Bengio, & Lipson, 2014). Second, the multiplicative Gaussian hypernetwork (mGHN) and its online learning method are developed. An mGHN concurrently adapts both structure and parameters to the data stream using an evolutionary method and a closed-form-based sequential update, which minimizes loss of past data. The mGHN possesses two good properties for online learning: it can learn from every new instance rapidly, even if the new instance is from a new class and it can handle incremental input features; a property that is essential when a new DNN is constructed and new input features appear in the fast memory.

1.5. Structure and contribution of the paper

The remainder of this paper is organized as follows. Section 2 reviews previous studies. Particularly, Section 2.4 provides an overview of various comparative models suggested for the online learning of DNNs. Section 3 introduces the general concept of DMA. Section 4 discusses the mGHN and its online learning method. Section 5 presents experimental results for the online learning of DNNs and analyzes the performance of the DMA. Section 5.1 explains the results for a non-stationary variant of the CIFAR-10 dataset, and Section 5.2 contains the description and the results for the Google Glass Lifelog dataset. Section 6 discusses the implication of the proposed architecture on other research fields, including CLS theory, Bayesian optimization, and lifelong learning.

This study is an extension of our previous work (Lee et al., 2016), which contributed the following to relevant literature: (1) Problem setup and review for the online learning of DNNs; (2) Proposal of the DMA and the mGHN; and (3) Empirical analysis of various online learning methods on real-life datasets. The main contributions of this study are: (1) Proof of the online parameter learning method of mGHNs; (2) Cognitive neuroscience perspective of DMA; (3) Making the Lifelog dataset publicly available; and (4) Implications of DMA on other machine learning fields. Moreover, additional empirical analysis and literature review are discussed in this study.

2. Related works

2.1. Deep learning and online learning

Deep learning algorithms, including CNNs and recurrent neural networks (RNNs), deliver state-of-the-art performance for various fields, including computer vision (He, Zhang, Ren, & Sun, 2015; Noh, Hong, & Han, 2015), speech recognition

(Graves, Mohamed, & Hinton, 2013; Sainath, Vinyals, Senior, & Sak, 2015), and natural language processing (Sundermeyer, Schlüter, & Ney, 2012; Sutskever, Vinyals, & Le, 2014), implying that representation learning with DNNs is essential for various context-aware problems. Further, this type of deep learning technique is also applicable for context-aware tasks in egocentric videos (Bettadapura, Essa, & Pantofaru, 2015; Doshi, Kira, & Wagner, 2015; Simonyan & Zisserman, 2014; Yu, Wang, Huang, Yang, & Xu, 2015). Online learning is a classical area of study in machine learning. Several studies have been conducted regarding online learning methods, especially for convex optimization problems, some of which have contributed to the development of the SGD method for learning large-scale neural networks (Duchi, Hazan, & Singer, 2011; Zinkevich, 2003).

2.2. Comparative models of dual memory architecture

Relatively few studies to date have been conducted on training deep networks online from data streams, due to the difficulty in optimizing the extremely large non-convex search space of a neural network (Bottou, 1998). We categorize these studies into three approaches. The first approach is *online fine-tuning*, which is online learning of an entire neural network based on SGD. In this setting, a deep network is continuously fine-tuned with new data as the data is accumulated. However, it is well-known that learning neural networks require many epochs of gradient descent over the entire dataset because the objective function space of neural networks is complex. Recently, in Nam and Han (2016), online fine-tuning of a CNN with simple online SGD was used in the inference phase of visual tracking and made state-of-the-art performance in the Visual Object Tracking Challenge 2015. However, this technique does not guarantee the retention of old data. The equation of this algorithm can be described as follows:

$$y = softmax(f(h^{\{1\}}(x))) \tag{1}$$

where f is a non-linear function of a deep neural network. This equation is the same in the case of batch learning, where *Batch* denotes the common algorithm that learns all the training data at once, with a single neural network.

The second approach is *last-layer fine-tuning*. In this method, only the last layer is fine-tuned, except the dataset of the first task. For the first task, the whole network is trained. According to recent work on transfer learning, the hidden activation of deep networks can be utilized as a satisfactory general representation for learning other related tasks. Training only the last-layer of a deep network often yields state-of-the-art performance on new tasks, especially when the dataset of a new task is small (Donahue et al., 2014; Zeiler & Fergus, 2014). This phenomenon makes online learning of only the last-layer of deep networks promising, because online learning of shallow networks is much easier than that of deep networks in general. Recently, online SVM with hidden representations of pre-trained deep CNNs using another large image dataset, ImageNet, performed well in visual tracking tasks (Hong, You, Kwak, & Han, 2015). Mathematically, the last-layer fine-tuning is expressed as follows:

$$y = \delta(w^T \phi(h^{\{1\}}(x))). \tag{2}$$

The third approach is incremental bagging. A considerable amount of research has sought to combine online learning and ensemble learning (Oza, 2005; Polikar et al., 2001). One of the simplest methods involves forming a neural network with some amount of online dataset and bagging in inference. Bagging is an inference technique that uses the average of the output probability of each network as the final output probability of the entire model. If deep memory is allowed to use more memory in our system, a competitive approach involves using multiple neural networks, especially when the data stream is non-stationary. In contrast to our approach, transfer learning techniques were not used in previous research. We refer to this method as *naïve incremental bagging*. The equation of incremental bagging can be described as follows:

$$y = \frac{1}{d}\sum_{i}^{d} softmax(f_d(h^{\{d\}}(x))). \tag{3}$$

2.3. Previous studies on online learning from data stream

Various studies have been conducted on online learning from data streams during an entire lifetime. Carlson et al. (2010) and Mitchell et al. (2015) proposed the Never-Ending Language Learner (NELL), which extracts a variety of information from the web and constructs a structured knowledge base. Chen, Shrivastava, and Gupta (2013) extended the NELL to develop the Never-Ending Image Learner (NEIL), which extracts both language and visual information from the web. Ha, Kim, and Zhang (2015) presented a model of deep concept hierarchy that enables progressive abstraction of concept knowledge at multiple levels in an online manner. They tested their algorithm on several hundred episodes of cartoon videos and showed that it can build vision-language concept hierarchies from non-stationary data streams. In our study, we attempt to extract abstracted concepts from the continuous sensing data of wearable devices or video streams.

2.4. Complementary learning systems and other machine learning algorithms

Recently, some machine learning algorithms related to CLS theory have been proposed. A recently published study reviewed the link between the core principles of the CLS theory and recent themes in machine learning (Kumaran, Hassabis, & McClelland, 2016); we follow this perspective.

The first example is model-free episodic control, which is a reinforcement learning method inspired by hippocampal episodic control applied to difficult sequential decision-making tasks (Blundell et al., 2016). In this method, a memorization strategy is utilized for learning a value function directly from experience without iterative estimation of the value function. This approach not only makes the learning significantly faster than comparable deep reinforcement learning algorithms, but also performs better on some of the more challenging domains.

The second example is external memory utilization for the neural network. Neural network researchers found that the capability of a neural network alone is limited in solving complex AI problems, including Q&A tasks. Thus, they have proposed coupling neural networks with external memory resources that correspond to hippocampal episodic memory. There are two methods exploiting external memory for neural networks. One method is the neural Turing machine (NTM) (Graves, Wayne, & Danihelka, 2014), where the framework consists of an external memory and an RNN controller that controls the reading from and writing to an external memory. The intermediate computations of the neural network are written to external memory, which can be read in when the results are required. Studies have shown that the NTM can learn the general concepts of a simple algorithm such as copying and sorting. The other method is the memory network (Weston, Chopra, & Bordes, 2014), where the external memory stores knowledge in a vectorized form. The memory network is mainly used for Q&A tasks and makes notable results (Sukhbaatar, Weston, & Fergus, 2015). In typical Q&A applications, appropriate vectorized knowledge (e.g., 'Tom left the basketball' and 'Tom

Algorithm 1 Online Learning of Dual Memory Architecture

```
if new instances come then
  if a new DNN is required then
    Initialize a new DNN with a previously constructed DNN.
    Train the new DNN with instances from storage.
    Update the input feature and corresponding structure of
    the shallow network.
  end if
  Update the parameters of the shallow network.
  Put the new instances into storage.
  Discard the oldest instances in storage.
end if
```

traveled to the garden') is retrieved by the similarity of a vectorized question (e.g., 'Where is the basketball') to find the right answer ('the garden'). However, these methods are criticized for their lower performance when compared with simpler models (Kim, Lee et al., 2016).

3. Dual memory architectures

The dual memory architecture (DMA) is a framework designed to learn continuously from data streams. The DMA framework is illustrated in Fig. 2, and the online learning process of the DMA is explained in Algorithm 1. The DMA consists of deep memory and fast memory. The structure of deep memory consists of several deep networks. Each of these networks is constructed when a specific amount of data from an unseen probability distribution is accumulated, thus creating a deep representation of data at a specific time. Examples of deep memory models include deep neural network classifier, CNNs, deep belief networks (DBNs), and recurrent neural networks (RNNs). The fast memory consists of a shallow network. The input to the shallow network is a set of hidden nodes from upper layers of the deep networks. Fast memory is updated immediately from each new instance. Examples of shallow networks include linear regressor, denoising autoencoder (Zhou, Sohn, & Lee, 2012), and support vector machine (SVM) (Liu, Zhang, Zhan, & Zhu, 2008) which can be learned online. The shallow network is in charge of making inference in the DMA; deep memory only generates the deep representation. The equation used for inference can be described as:

$$y = \delta(w^T \phi(h^{\{1\}}(x), h^{\{2\}}(x), \ldots, h^{\{k\}}(x))) \tag{4}$$

where x is the input (e.g., a vector of image pixels), y is the target, ϕ and w are kernel and corresponding weight respectively, h is values of the hidden layer of a deep network used for the input to the shallow network, δ is an activation function of the shallow network, and k is an index for the last deep network ordered by time.

Fast memory updates parameters of its shallow network immediately from new instances. If a new deep network is formed in the deep memory, the structure of the shallow network is changed to include the new representation. Fast memory is referred to as fast for two properties with respect to learning. First, a shallow network learns faster than a deep network in general. Second, a shallow network is better able to adapt to new data through online learning than a deep network. If the objective function of a shallow network is convex, a simple stochastic online learning method, such as online SGD, can be used to guarantee a lower bound to the objective function (Zinkevich, 2003). Therefore, an efficient online update is possible. Unfortunately, learning shallow networks in the DMA is more complex. During online learning, deep memory continuously forms new representations of a new deep network; thus, new input features appear in the shallow network. This task is a kind of online learning with an incremental feature set. In this case, it is not possible to obtain statistics of old data at new features; i.e., if a node in the shallow network is a function of $h^{\{k\}}$, statistics of the node cannot be obtained from the 1st–$k-1$th online dataset. In this paper, we explore online learning by shallow networks using an incremental feature set in the DMA.

Table 1
Notations.

Symbol	Explanation
y	Class (i.e., location, sub-location, or activity)
ϕ	Kernel vector
ϕ_k	kth kernel
h	Hidden vector of DNNs
x	Input vector (i.e., input pixel)
k	Index of kernel set
d	Index of small dataset
n	Index of instance
$\hat{\mu}$, $\hat{\Sigma}$	Empirical sufficient the statistics
$\tilde{\mu}$, $\tilde{\Sigma}$	Approximated maximum likelihood solution of statistics
$\phi_{k\cdot(d,n)}$	(d, n)th instance of kth kernel
$\hat{\mu}_{k\cdot d}$, $\hat{\Sigma}_{k\cdot d}$	Empirical sufficient statistics of ϕ_k over dth small dataset

In learning deep memory, each deep neural network is trained with a corresponding online dataset by its objective function. Unlike the prevalent approach, we use the transfer learning technique proposed by Yosinski et al. (2014) to utilize the knowledge from an older deep network to form a new deep network. This transfer technique initializes the weights of a newly trained deep network W_k by the weights of the most recently trained deep network W_{k-1}. Once the training of the deep network from its own online dataset is complete, the weights of the network do not change even though new data arrives. This is aimed at minimizing changes of input in the shallow network in the fast memory. This original transfer method assumes that the two networks have the same structure. However, there are some extensions (Chen, Goodfellow, & Shlens, 2016; Wei, Wang, Rui, & Chen, 2016) that allow different widths and number of layers between some networks.

The proposed DMA is a combination of the ideas of the three previously proposed methods mentioned above. In DMA, a new deep network is formed when a dataset is accumulated, as in incremental bagging. However, the initial weights of new deep networks are drawn from the weights of the older deep networks, as in the online learning of neural networks. Moreover, a shallow network in the fast memory is concurrently trained with deep memory, which is similar to the last-layer fine-tuning approach.

4. Online learning of multiplicative-Gaussian hypernetworks

4.1. Multiplicative-Gaussian hypernetworks

In this section, we introduce a multiplicative Gaussian hypernetwork (mGHN) as an example of fast memory (Fig. 3). Table 1 summarizes the notation used in this section. mGHNs are shallow networks that use a multiplicative function as an explicit kernel in (5):

$$\begin{aligned} &\phi = [\phi^{(1)}, \ldots, \phi^{(p)}, \ldots, \phi^{(P)}]^T, \\ &\text{s.t., } \phi^{(p)}(h) = (h_{(p,1)} \times \cdots \times h_{(p,H_p)}), \end{aligned} \tag{5}$$

where P is a hyperparameter of the number of kernel functions, and $\times$ denotes scalar multiplication. h is the input feature of mGHNs, and also represents the hidden activation of DNNs. The set of variables of the pth kernel $\{h_{(p,1)}, \ldots, h_{(p,H_p)}\}$ is randomly chosen from h, where H_p is the order or the number of variables used in the pth kernel. The multiplicative form is used for two reasons, although an arbitrary form can be used. First, it is an easy, randomized method to put sparsity and non-linearity into

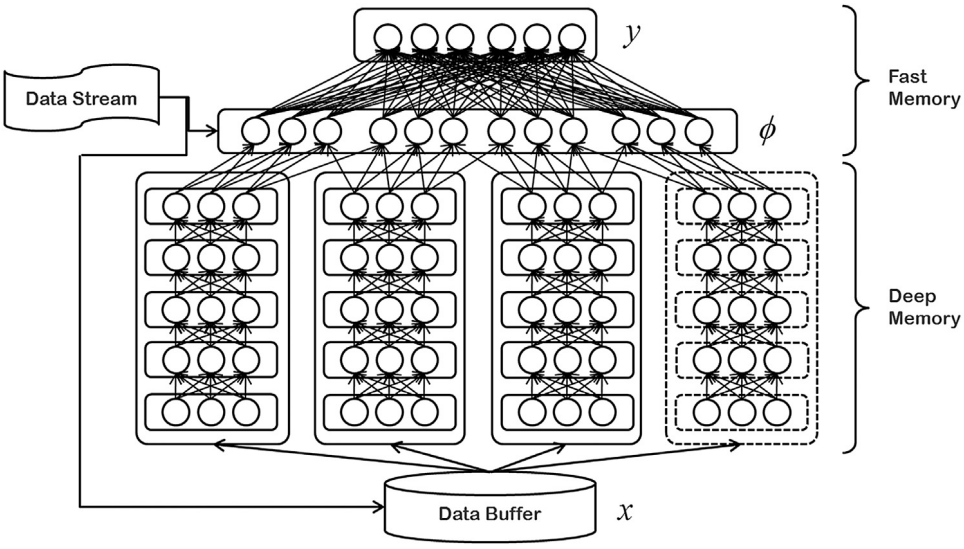

Fig. 2. A schematic diagram of the dual memory architecture (DMA). With continuously arriving data stream instances, fast memory updates its shallow network immediately. If a certain amount of data is accumulated, deep memory makes a new deep network with this new online dataset. Simultaneously, the shallow network changes its structure corresponding to the deep memory.

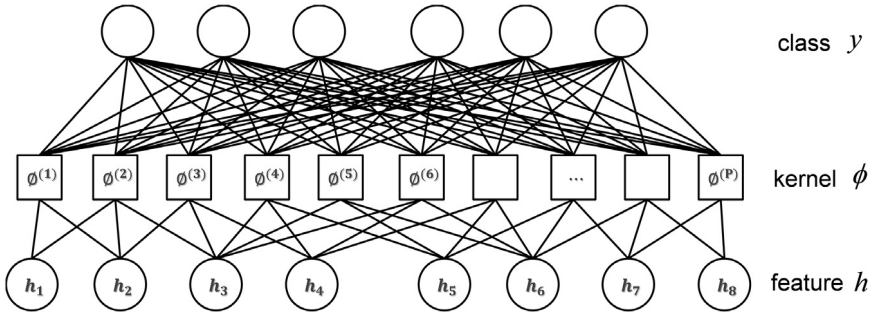

Fig. 3. A schematic diagram of the multiplicative-Gaussian hypernetworks.

the model, which is a point inspired by Zhang (2008) and Zhang, Ha, and Kang (2012). Second, the kernel can be controlled to be a function of a few neural networks. mGHNs assume the joint probability of target class y and ϕ is Gaussian as in (6):

$$p\left(\begin{bmatrix} y \\ \phi(h) \end{bmatrix}\right) = N\left(\begin{bmatrix} \mu_y \\ \mu_\phi \end{bmatrix}, \begin{bmatrix} \Sigma_{yy} & \Sigma_{y\phi} \\ \Sigma_{\phi y} & \Sigma_{\phi\phi} \end{bmatrix}\right), \tag{6}$$

where μ_y, μ_ϕ, Σ_{yy}, $\Sigma_{y\phi}$, $\Sigma_{\phi y}$, and $\Sigma_{\phi\phi}$ are the sufficient statistics of the Gaussian corresponding to y and ϕ. Target class y is represented by one-hot encoding. The discriminative distribution is derived by the generative distribution of y and ϕ, and predicted y is the real-valued score vector of the class in the inference.

$$E[p(y|h)] = \mu_y + \Sigma_{y\phi} \cdot \Sigma_{\phi\phi}^{-1} \cdot (\phi(h) - \mu_\phi). \tag{7}$$

The maximum likelihood solution of mGHNs can be updated immediately from any amount of new instances by an online update of the mean and covariance if the number of features does not increase. Note that the distribution over y and ϕ over the first and second datasets ($d = 1 : 2$) is as follows:

$$\begin{bmatrix} y \\ \phi_1 \end{bmatrix}|_{d=1:2} \sim N\left(\begin{bmatrix} \hat{\mu}_{y\cdot 1:2} \\ \hat{\mu}_{1\cdot 1:2} \end{bmatrix}, \begin{bmatrix} \Sigma_{yy\cdot 1:2} & \Sigma_{y1\cdot 1:2} \\ \Sigma_{1y\cdot 1:2} & \Sigma_{11\cdot 1:2} \end{bmatrix}\right) \tag{8}$$

where $\hat{\mu}_{\cdot 1:2}$, $\hat{\Sigma}_{\cdot 1:2}$ is the function of $\hat{\mu}_{\cdot 1}$, $\hat{\mu}_{\cdot 2}$, $\hat{\Sigma}_{\cdot 1}$, and $\hat{\Sigma}_{\cdot 2}$.

$$\hat{\mu}_{1\cdot 1} = \frac{1}{N_1}\sum_{n=1}^{N_1} \phi_{1\cdot(d=1,n)} \tag{9}$$

$$\hat{\mu}_{1\cdot 2} = \frac{1}{N_2}\sum_{n=1}^{N_2} \phi_{1\cdot(d=2,n)} \tag{10}$$

$$\begin{aligned} \hat{\mu}_{1\cdot 1:2} &= \frac{1}{N_1+N_2}\left[\sum_{n=1}^{N_1}\phi_{1\cdot(d=1,n)} + \sum_{n=1}^{N_2}\phi_{1\cdot(d=2,n)}\right] \\ &= \alpha\hat{\mu}_{1\cdot 1} + (1-\alpha)\hat{\mu}_{1\cdot 2} \end{aligned} \tag{11}$$

$$\alpha = \frac{N_1}{N_1+N_2} \tag{12}$$

$$\begin{aligned} \hat{\Sigma}_{11\cdot 1} &= \frac{1}{N_1}\sum_{n=1}^{N_1}(\phi_{1\cdot(d=1,n)} - \hat{\mu}_{1\cdot 1})(\phi_{1\cdot(d=1,n)} - \hat{\mu}_{1\cdot 1})^T \\ &= \frac{1}{N_1}\left[\sum_{n=1}^{N_1}\phi_{1\cdot(d=1,n)}\phi_{1\cdot(d=1,n)}^T\right] + \hat{\mu}_{1\cdot 1}\hat{\mu}_{1\cdot 1}^T \end{aligned} \tag{13}$$

$$\begin{aligned} \hat{\Sigma}_{11\cdot 1:2} &= \alpha\hat{\Sigma}_{11\cdot 1} + (1-\alpha)\hat{\Sigma}_{11\cdot 2} + \alpha(1-\alpha)(\hat{\mu}_{1\cdot 1} \\ &\quad - \hat{\mu}_{1\cdot 2})(\hat{\mu}_{1\cdot 1} - \hat{\mu}_{1\cdot 2})^T. \end{aligned} \tag{14}$$

In our model, the parameter of two datasets can be decomposed into the parameter of each dataset, which we refer to as *instance-decomposability*. This property allows our model to weigh specific events appropriately, which we refer to as *instance-scale reweighting*. For example, the model learns significantly r times from recent events as opposed to the penalized previous events by substituting α in (12) for following α':

$$\alpha' = \frac{N_1}{N_1 + r \cdot N_2}. \tag{15}$$

Algorithm 2 Structure Learning of mGHNs

repeat
  if New learned feature $h^{\{k\}}$ comes **then**
    Concatenate old and new feature (i.e., $h \leftarrow h \bigcup h^{\{k\}}$).
    Discard a set of kernel $\phi_{discard}$ in ϕ (i.e., $\hat{\phi} \leftarrow \phi - \phi_{discard}$).
    Make a set of new kernel $\phi_k(h)$ and concatenate into ϕ (i.e., $\phi \leftarrow \hat{\phi} \bigcup \phi_k$).
  end if
until forever

4.2. Evolutionary structure learning

If the kth deep neural network is formed in the deep memory, the mGHN in the fast memory receives a newly learned feature $h^{\{k\}}$, which consists of the hidden values of the new deep neural network. As the existing kernel vector is not a function of $h^{\{k\}}$, a new kernel vector ϕ_k should be formed. The structure of mGHNs is learned via an evolutional approach (Zhang et al., 2012), as illustrated in Algorithm 2.

The core operations in the algorithm consist of discarding less-important kernels and adding the new kernel. In our experiments, the set of $\phi_{discard}$ was picked by selecting the kernels with the lowest corresponding weights. From Eq. (7), ϕ is multiplied by $\Sigma_{y\phi}\Sigma_{\phi\phi}^{-1}$ to obtain $E[p(y|h)]$, such that weight $w^{(p)}$ corresponding to $\phi^{(p)}$ is the pth column of $\Sigma_{y\phi}\Sigma_{\phi\phi}^{-1}$ (i.e., $w^{(p)} = (\Sigma_{y\phi}\Sigma_{\phi\phi}^{-1})_{(p,:)}$.) The length of $w^{(p)}$ is the number of class categories, as the node of each kernel has a connection to each class node. We sort $\phi^{(p)}$ in descending order of $\max_j |w_j^{(p)}|$, where the values at the bottom of the $\max_j |w_j^{(p)}|$ list correspond to the $\phi_{discard}$ set. The size of $\phi_{discard}$ and ϕ_k are determined by $\alpha|\phi|$ and $\beta|\phi|$ respectively, where $|\phi|$ is the size of the existing kernel set, and α and β are predefined hyperparameters.

4.3. Online learning on incremental features

As the objective function of mGHNs follows the exponential of the quadratic form, second-order optimization can be applied for efficient online learning. For the online learning of mGHNs with incremental features, we derive a closed-form sequential update rule to maximize likelihood based on studies of regression with missing patterns (Little, 1992).

Suppose the first ($k = 1$) and the second ($k = 2$) kernel vectors ϕ_1 and ϕ_2 are constructed when the first ($d = 1$) and the second ($d = 2$) online datasets arrive. The sufficient statistics of ϕ_1 can be obtained for both the first and second datasets, whereas information of only the second dataset can be used for ϕ_2. Suppose $\hat{\mu}_{i\cdot d}$ and $\hat{\Sigma}_{ij\cdot d}$ are empirical estimators of the sufficient statistics of the ith kernel vector ϕ_i and jth kernel vector ϕ_j corresponding to the distribution of the dth dataset. If these sufficient statistics satisfy the following equations:

$$\phi|_{d=1} \sim N(\hat{\mu}_{1\cdot 1}, \hat{\Sigma}_{11\cdot 1}) \tag{16}$$

$$\begin{bmatrix}\phi_1\\ \phi_2\end{bmatrix}|_{d=2} \sim N\left(\begin{bmatrix}\hat{\mu}_{1\cdot 2}\\ \hat{\mu}_{2\cdot 2}\end{bmatrix}, \begin{bmatrix}\hat{\Sigma}_{11\cdot 2} & \hat{\Sigma}_{12\cdot 2}\\ \hat{\Sigma}_{21\cdot 2} & \hat{\Sigma}_{22\cdot 2}\end{bmatrix}\right) \tag{17}$$

$$\phi_1|_{d=1,2} \sim N(\hat{\mu}_{1\cdot 1:2}, \hat{\Sigma}_{11\cdot 1:2}) \tag{18}$$

the parametrization of maximum likelihood via the conditional Gaussian distribution can be done as follows:

$$\begin{aligned} p(\phi_2|\phi_1)|_{d=2} &= N(\hat{\mu}_{2|1\cdot 2}, \hat{\Sigma}_{22|1\cdot 2})\\ &= N(\hat{\mu}_{2\cdot 2} + \hat{\Sigma}_{21\cdot 2}\hat{\Sigma}_{11\cdot 2}^{-1}(\phi_1 - \hat{\mu}_{1\cdot 2}),\\ &\quad \times \hat{\Sigma}_{22\cdot 2} - \hat{\Sigma}_{21\cdot 2}\hat{\Sigma}_{11\cdot 2}^{-1}\hat{\Sigma}_{12\cdot 2}). \end{aligned} \tag{19}$$

Note the following properties of Gaussian:

$$E\begin{bmatrix}x\\ y\end{bmatrix} = \begin{bmatrix}\mu\\ A\mu + b\end{bmatrix} \tag{20}$$

$$Cov\begin{bmatrix}x\\ y\end{bmatrix} = \begin{bmatrix}\Lambda^{-1} & \Lambda^{-1}A^T\\ A\Lambda^{-1} & L^{-1} + A\Lambda^{-1}A^T\end{bmatrix} \tag{21}$$

where the probability of x and the conditional probability of y given x are as follows:

$$p(x) = N(x|\mu, \Lambda^{-1}) \tag{22}$$

$$p(y|x) = N(y|Ax + b, L^{-1}). \tag{23}$$

Thus, the maximum likelihood solution represents ϕ as (24).

$$\begin{aligned} &\begin{bmatrix}\phi_1\\ \phi_2\end{bmatrix}|_{d=1,2} \sim N\left(\begin{bmatrix}\hat{\mu}_{1\cdot 1:2}\\ \tilde{\mu}_2\end{bmatrix}, \begin{bmatrix}\hat{\Sigma}_{11\cdot 1:2} & \tilde{\Sigma}_{12}\\ \tilde{\Sigma}_{12}^T & \tilde{\Sigma}_{22}\end{bmatrix}\right),\\ &\tilde{\mu}_2 = \hat{\mu}_{2\cdot 2} + \hat{\Sigma}_{12\cdot 2}^T \cdot \hat{\Sigma}_{11\cdot 2}^{-1} \cdot (\hat{\mu}_{1\cdot 12} - \hat{\mu}_{1\cdot 2}),\\ &\tilde{\Sigma}_{12} = \hat{\Sigma}_{11\cdot 1:2} \cdot \hat{\Sigma}_{11\cdot 2}^{-1} \cdot \hat{\Sigma}_{12\cdot 2},\\ &\tilde{\Sigma}_{22} = \hat{\Sigma}_{22\cdot 2} - \hat{\Sigma}_{12\cdot 2}^T \cdot \hat{\Sigma}_{11\cdot 2}^{-1} \cdot (\hat{\Sigma}_{12\cdot 2} - \tilde{\Sigma}_{12}). \end{aligned} \tag{24}$$

(24) can also be updated immediately from a new instance by online update of the mean and covariance. Note that the proposed online learning algorithm estimates a generative distribution of ϕ, $p(\phi_1, \ldots, \phi_k)$. When training data having ϕ_k is relatively small, information of ϕ_k can be complemented by $p(\phi_k|\phi_{1:k-1})$, which helps create a more efficient prediction of y.

The alternative of this generative approach is a discriminative approach. For example, in Liu et al. (2008), LS-SVM is directly optimized to get the maximum likelihood solution over $p(y|\phi_{1:k})$. However, equivalent solutions from the discriminative method can also be produced by the method of filling in the missing values with 0 (e.g., assume $\phi_2|_{d=1}$ as 0), which is not what we desire intuitively.

$$\begin{aligned} &\hat{\mu}_{y|x}, \hat{\Sigma}_{y|x} = \operatorname{argmax} p(y|\phi_{1:k}),\\ &p(y|\phi_{1:k}) \sim N(\hat{\mu}_{y|\phi}, \hat{\Sigma}_{y|\phi}). \end{aligned} \tag{25}$$

Moreover, (24) can be extended to sequential updates, when there is more than one increment of the kernel set (i.e., $\phi_3, \ldots, \phi_k$), because the equation of maximum likelihood of whole kernel observation ϕ_{obs} can be decomposed as follows:

$$p(\phi_{obs}) = \prod_{i=1}^{k}\prod_{d=1}^{k}\prod_{n=1}^{N_d} p(\phi_{i\cdot(d,n)}|\phi_{1:(i-1)\cdot(d,n)}). \tag{26}$$

5. Experiments

5.1. Non-stationary image data stream

We investigate the strengths and weaknesses of the proposed DMA in an extreme non-stationary environment using a well-known benchmark dataset. The proposed algorithm was tested on the CIFAR-10 image dataset consisting of 50,000 training images and 10,000 test images from 10 different object classes. The performance of the algorithm was evaluated using a 10-split experiment where the model is learned in a sequential manner from 10 online datasets. In this experiment, each online dataset consists of images of 3–5 image classes to make 10 online datasets for an extremely changing environment. Fig. 4 shows the distribution of the data stream. In particular, the first online dataset contains 40% instances of classes 1 and 2 respectively, and 20% instances of class 3. Meanwhile, the second online dataset contains 40% instances of class 1, and 20% instances of classes 2, 3, and 4 respectively. We used the Network in Network (NIN) model

Table 2
Properties of DMA and comparative models.

	Many deep networks	Online learning	Dual memory structure
Online fine-tuning		✓	
Last-layer fine-tuning		✓	
Naïve incremental bagging	✓	✓	
DMA (our proposal)	✓	✓	✓
Incremental bagging w/transfer	✓	✓	
DMA w/last-layer retraining	✓		✓
Batch			

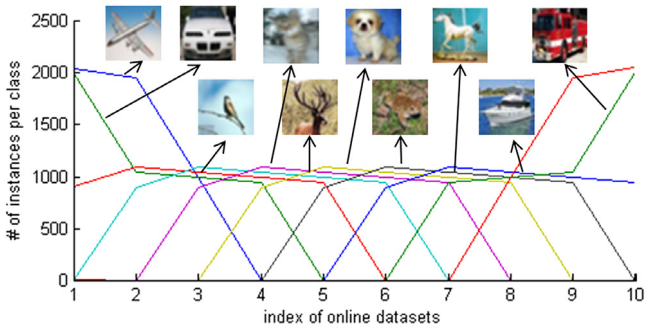

Fig. 4. Distribution of non-stationary data stream of CIFAR-10 in the experiment.

(Lin, Chen, & Yan, 2014), a kind of deep CNN, implemented using the MatConvNet toolbox (Vedaldi & Lenc, 2015). In the NIN, the conventional convolution linear filters are replaced by multi-layer DNNs. This idea is utilized by various CNNs, including inception networks (Szegedy et al., 2015), the winner of ImageNet Challenge 2014. In our learning phase, the learning rate is set to 0.25 and then is reduced by a constant factor of 5 at some predefined steps. The rate of weight decay is 5×10^{-4}, and the rate of momentum is 0.9. The dropping learning rate strategy, weight decay, and rate of momentum values are commonly used in deep learning (Bottou, 2012) and also are the default settings of CIFAR-10 learning in MatConvNet toolbox.

We evaluate the performance of DMA with comparative models. The properties of DMA and comparative methods are listed in Table 2. Specifically, we propose two comparable learning methods to clarify the concept of DMA. The first one is *incremental bagging with transfer*. Unlike the naïve incremental bagging, this method transfers the weights of the older deep networks to the new deep network, as in DMA. The other is *DMA with last-layer retraining* where a shallow network is retrained in a batch manner. Although this algorithm is not part of online learning, it is practical because batch learning of shallow networks is much faster than that of deep networks in general.

Fig. 5 illustrates 10-split experimental results on non-stationary data. The models with only one CNN, which are online fine-tuning and last-layer fine-tuning in this experiment, show inferior results compared to other algorithms. These models did not adapt to extreme non-stationary data streams in the experiment on CIFAR-10. In the last-layer fine-tuning, a CNN trained by the first online dataset was used. The model has deep representation for discriminating only three image classes. Hence, the performance does not increase significantly. In the case of online fine-tuning, the model loses the previously seen information, which reduces the performance of test accuracy as time progresses. Meanwhile, incremental bagging increases its performance continuously with the non-stationary data stream. Incremental bagging that uses many networks outperforms online fine-tuning that uses only one deep network. On the other hand, the proposed DMA outperforms incremental bagging consistently. This result shows learning a shallow network and deep networks concurrently is advantageous compared to naïvely averaging the softmax output probability of

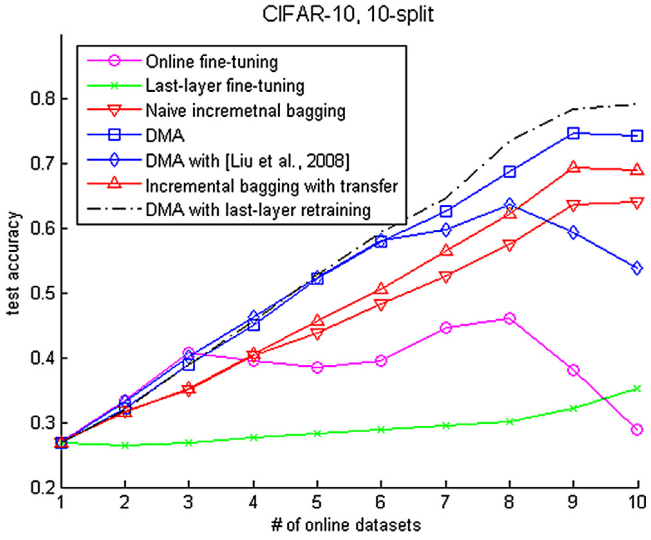

Fig. 5. Test accuracies of learning algorithms on non-stationary CIFAR-10 data stream.

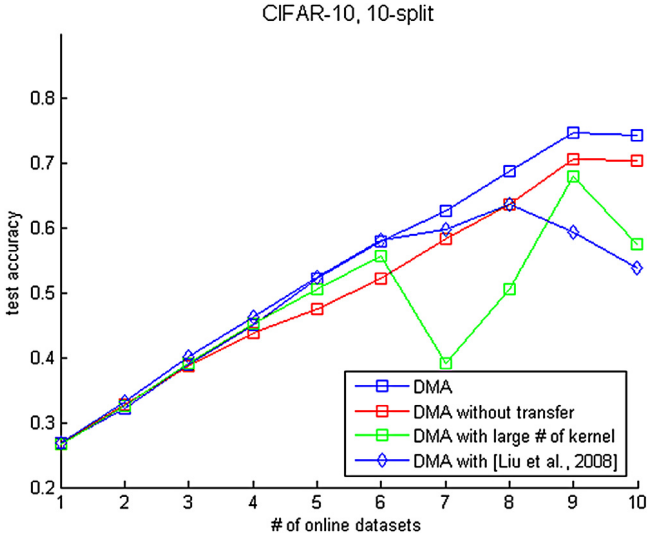

Fig. 6. Test accuracies of DMA on CIFAR-10 data stream under various settings.

each CNN. When the last layer is retrained for the whole dataset using the batch learning scheme, the performance improves. Nevertheless, the proposed online learning scheme of DMA works reasonably, and the performance gap is not large.

Alternative configurations of the DMA were also explored in our study to discover the characteristics of DMA (Fig. 6). First, we analyze the effect of weight transfer in deep memory. The DMA with weight transfer performs better than the DMA without weight transfer (+3.80%), as shown in Fig. 6. The performance gap is much larger between incremental bagging with weight transfer and the model without weight transfer (+4.89%), as shown in Fig. 5.

Table 3
Statistics of the Lifelog dataset of each subject.

	Instances (s/day)		Number of class		
	Training	Test	Location	Sub-location	Activity
A	105 201 (13)	17 055 (5)	18	31	39
B	242 845 (10)	91 316 (4)	18	28	30
C	144 162 (10)	61 029 (4)	10	24	65

Table 4
Top-5 classes in each label of the Lifelog dataset.

Location	Sub-location	Activity
Office (196 839)	Office-room (182 884)	Working (204 131)
University (147 045)	Classroom (101 844)	Commuting (102 034)
Outside (130 754)	Home-room (86 588)	Studying (90 330)
Home (97 180)	Subway (35 204)	Eating (60 725)
Restaurant (22 190)	Bus (34 120)	Watching (35 387)

Part of the entire data is not sufficient for learning discriminative representations in the weak learner for the whole class. In the experiment, weight transfer alleviates this problem for both the DMA and incremental bagging.

Second, we substitute our generative assumption over class y and kernel ϕ to the discriminative counterpart suggested by Liu et al. (2008). The performance of the two approaches does not differ at the early phase of the learning. However, the performance of the discriminative approach deteriorates as extreme non-stationary data is encountered continuously. This result supports our argument that a generative approach is one of the key points of successful online learning of mGHNs.

Lastly, we discuss the effect of the number of kernel features in the mGHN. The generative approach of mGHN needs more parameters compared to the discriminative approach. For example, the number of parameters of the covariance matrix in our generative approach is directly proportional to square of the number of kernel features, whereas the number of parameters of the weight matrix in the discriminative counterpart is directly proportional to the number of kernel features. The performance of the DMA severely decreases if the kernel size increase from 1000 to 4900 variables at the end of training as shown in *DMA with large # of kernel* in Fig. 6.

5.2. Lifelog dataset

We collected a Google Glass Lifelog dataset recorded over 46 days from three participants. The 660,000 s of the egocentric video stream data reflects the behaviors of the participants; including indoor activities, such as 'working in an office' or 'eating in a restaurant', and outdoor activities, such as 'walking on the road' or 'waiting for the arrival of the subway car'. The subjects were asked to notate what they were doing and where they were in real-time by using a life-logging application installed on their mobile phones. The notated data was then used as labels for the classification task in our experiments. In this study, the classification task of location, sub-location, and daily activity are considered. For evaluation, the dataset of each subject is separated into the training set and test set in order of time. A frame image of each second is used and classified as one instance. Table 3 summarizes the dataset statistics and Table 4 presents the distribution of the five major classes in each class type. We allowed the AlexNet features and the labels of class types of the Lifelog dataset to be publicly available (*version 1*).[1]

Two kinds of neural networks were used to extract the representations in this experiment. One is AlexNet, a prototype network trained by ImageNet (Krizhevsky, Sutskever, & Hinton, 2012). The other is referred to as LifeNet, a network trained with our Lifelog dataset. The structure of LifeNet is similar to AlexNet, but the number of nodes in LifeNet is half that of AlexNet. Both AlexNet and LifeNet were implemented using the MatConvNet toolbox. We chose a 1000-dimensional softmax output vector of AlexNet for representation of online deep learning algorithms, as we assume the probability of an object's appearance in each scene to be highly related to the daily activity represented by each scene.

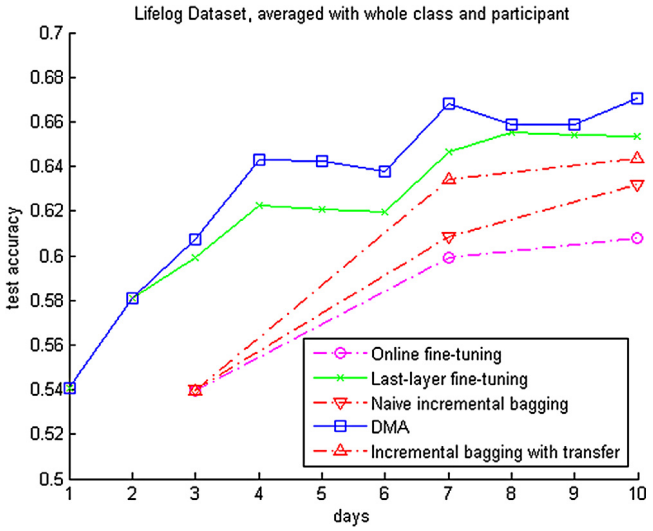

Fig. 7. Averaged test accuracies of various learning algorithms on the Lifelog dataset. The location, sub-location, and activity are classified separately for each of the three subjects.

The performances on the Lifelog dataset were evaluated in a 10-split experiment. Each online dataset corresponds to each day for the subjects B and C, respectively. However, for subject A, the 13 days of training data was changed into 10 online datasets by merging 3 of the days into its next days. Each online dataset is referred to as a day. LifeNets made from 3 groups of online Lifelog datasets, with sets of consecutive 3, 4 and 3 days for each group. In the entire learning of LifeNet, the learning rate is set to 0.0025, the rate of weight decay is 5×10^{-4}, and the rate of momentum is 0.9. In the experiment, LifeNet is used for online fine-tuning and incremental bagging, AlexNet for last-layer fine-tuning and both the LifeNet and AlexNet are used for DMA.

Fig. 7 shows the experimental results from the Lifelog dataset. The experiments consist of three subjects whose tasks are classified into three categories. A total of nine experiments are performed and their averaged test accuracies from a range of learning algorithms are plotted. Unlike the previous result on CIFAR-10, last-layer fine-tuning that uses one AlexNet outperforms other online deep learning algorithms that use many LifeNets. However, these learning algorithms perform worse than DMA that uses numerous LifeNets and one AlexNet. This implies that usage of pre-trained deep networks by a large corpus dataset is effective on the Lifelog dataset. From the perspective of personalization, a representation obtained by existing users or other large datasets can be used together with a representation obtained by a new user. However, DMA that uses both AlexNet and LifeNet works better than last-layer fine-tuning, which implies again that using multiple networks is necessary for online deep learning.

Figs. 8 and 9 show the accuracies by each subject and class type respectively. In some experiments, at times the performance of algorithms decreases with the incoming stream of data. While learning a non-stationary data stream is a natural phenomenon, it would occur in situations where the test data is more similar to the training data encountered earlier than later during the learning process. Although, such fluctuations can occur, on average,

[1] bi.snu.ac.kr/datasets/lifeome/.

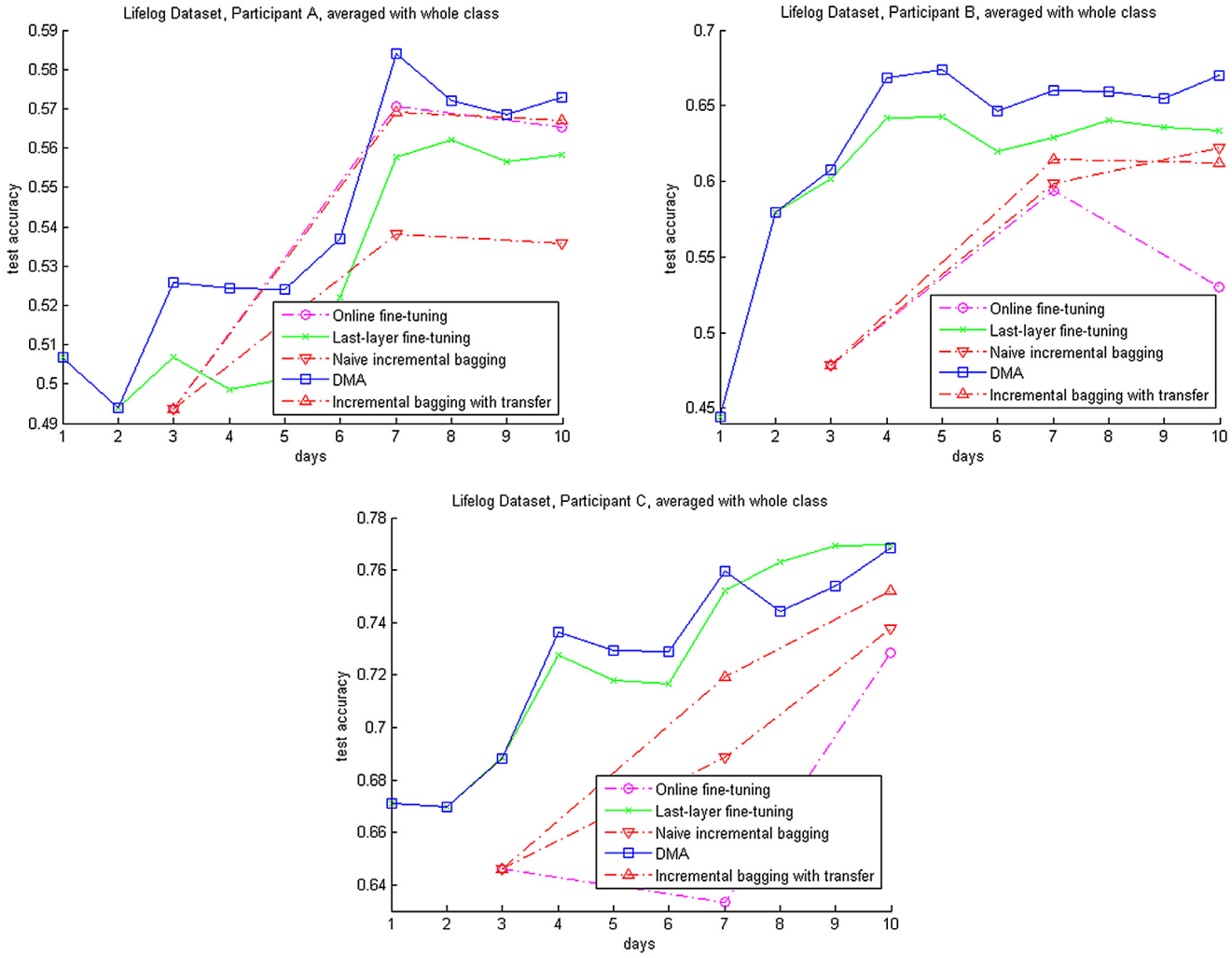

Fig. 8. Averaged test accuracies of various learning algorithms on the Lifelog dataset. The result of each class type is evaluated separately for each participant.

however, the accuracies of algorithms increase steadily with the incoming stream of data.

6. Discussion

Through the previous section, the online learnability of the DMA over a non-stationary stream data was evaluated and discussed. Moreover, the idea presented in our study can be utilized in other research fields. This section discusses the implications of the DMA to the CLS theory, Bayesian optimization, and online multi-task learning.

6.1. Complementary learning systems theory and dual memory architecture

We have previously mentioned that the DMA is inspired by the CLS theory. This study not only first raises the catastrophic forgetting problem, but also suggests the dual structure to solve the problem; deep memory corresponds to the neocortex, whereas fast memory corresponds to the hippocampus. Further, the following questions on missing links in the neurocognitive modeling of CLS are in our interest: First, how can the neocortex as the connectionist parametric model and the hippocampus as the instance-based non-parametric model work together? Second, what is the mechanism of neurocognitive data regeneration? Is it a database-style memorization or does a more efficient algorithm exist for preserving the information of data patterns?

Two properties of the DMA can be used to answer these questions. First, the parameter μ and Σ of an mGHN is decomposable at each instance even though DNN and mGHN infer as one neural network. This instance-decomposability allows one-shot learning (Fei-Fei, Fergus, & Perona, 2006), learning information about categories, from one, or only a few training instances, and instance-scale reweighting, learning with over-weight and under-weight instances by their importance.

Second, mGHNs can recover old information using an analytic method over the maximum likelihood solution from a pattern-completion perspective. This property accords with the interpretation of the CLS theory that there are areas for pattern classification (dentate gyrus) and pattern generation (CA3) in the hippocampus.

6.2. Bayesian optimization

Bayesian optimization is a global black-box optimization technique that can be effectively applied on functions whose evaluation cost is expensive and distribution is unknown. To determine the next search point, Bayesian optimization uses a surrogate estimator of the real search space, which gives an estimated score and its predictive uncertainty for each point. The most typical algorithm for Bayesian optimization is the Gaussian process (GPs) that models non-linear search space when the

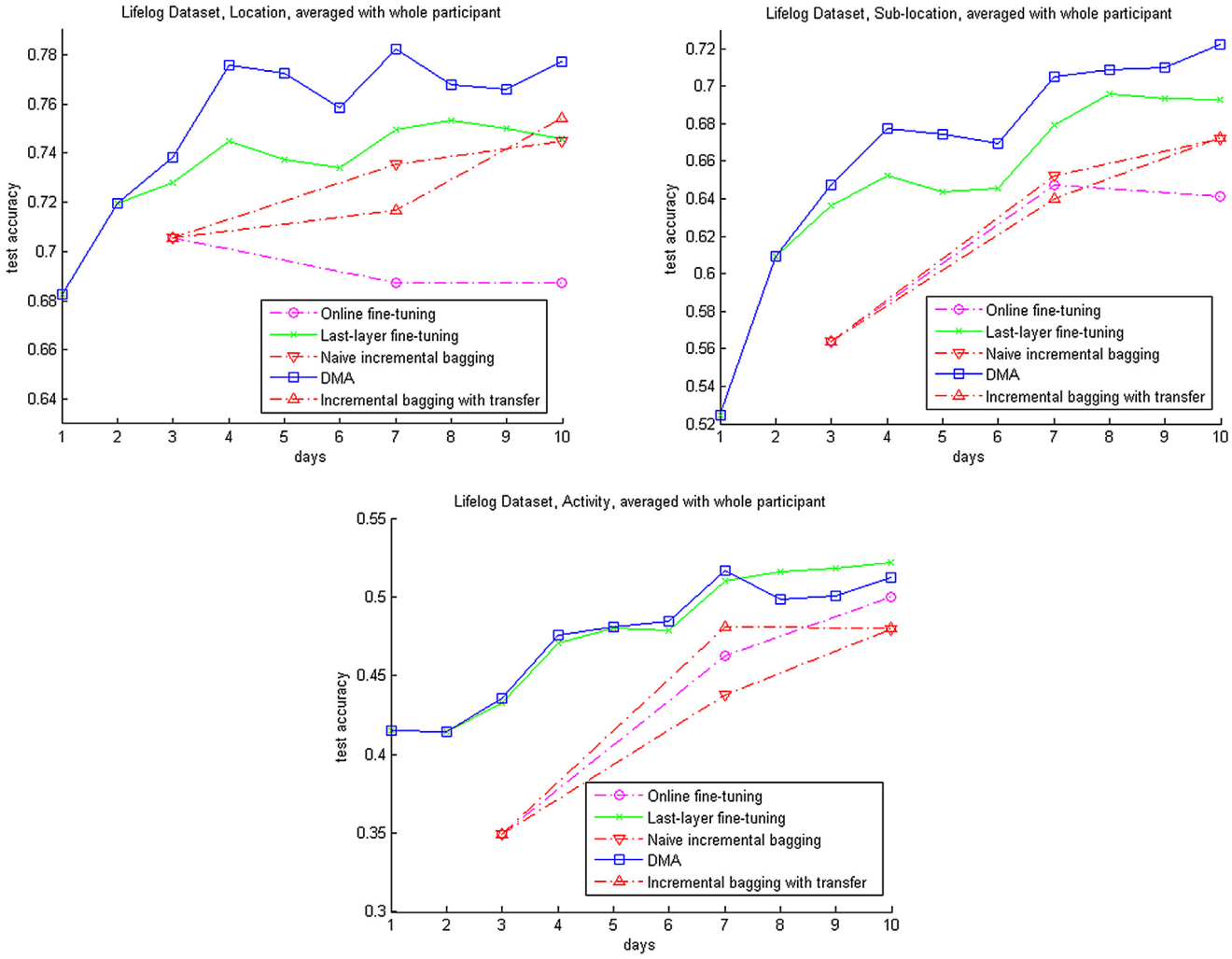

Fig. 9. Averaged test accuracies of various learning algorithms on the Lifelog dataset. The result of each participant is evaluated separately for each of the class type.

speed of inference is high. However, the inference cost of GPs become expensive when the size of the training data is large, since GPs scale cubically with the number of observations. Snoek et al. (2015) propose DNN-based modeling which alleviates this problem, because the DNN is adaptable for large-scale data. In their model, a Bayesian linear regression model is added to the last layer of the DNN. This existing method of DNN does not fit stream data for the development of deep representations in an online manner. Our method can be applied to this kind of setting. The DMA updates not only the mGHN, which can work as the regression model in the last layer of DNN, but also the deep representation. To use mGHNs as Bayesian models, the prior can be set so as to satisfy the statistic of Gaussian parameters μ, Σ over class y and kernel ϕ. An example structure of the Bayesian version of DMA is shown in Kim et al.'s work (Kim, Jang, Han, & Choi, 2016), where effective deep representations of a CNN pre-trained by ImageNet are selected with the Bayesian least-square support vector machine (LS-SVM) model to classify into a new task.

6.3. Online multi-task learning

Online multi-task learning refers to the learning of multiple consecutive tasks with never-ending exploration and continuous discovery of knowledge from data streams. This task has also been referred to as continual learning or lifelong learning. This learning is crucial for the creation of intelligent and flexible general-purpose machines such as personalized digital assistants and autonomous humanoid robots (Thrun & O'Sullivan, 1996). Recently, some studies have proposed methods that maintain a sparsely shared base of the shallow network for all task models (Ruvolo & Eaton, 2013). However, these approaches cannot learn deep shared representations over all tasks, which degrades the performance of these models. Online deep learning has useful properties from the perspective of lifelong learning because deep neural networks show high performance in transfer and multi-task learning (Heigold et al., 2013; Yosinski et al., 2014). In the DMA framework, online multi-task learning can be easily achieved by allowing one mGHN for each task in the fast memory and shared deep memory for all tasks. For the Lifelog dataset, this approach can be applied to the shared deep representation for recognizing location, sub-location, and activity.

7. Conclusion

We presented a dual-memory architecture (DMA) for deep neural networks that learns from continuous user behavior in everyday life using wearable devices. Our experimental results showed that the proposed method overcomes catastrophic forgetting in the learning of real non-stationary data. This property was utilized for implementing an advanced personalized context recognition system.

This success is an example of solving engineering problems using inspiration drawn from theories in cognitive neuroscience,

in this case the CLS theory. Our DMA implementation was used to discuss how a neural network model and an instance-based method work together from the CLS theory perspective. We also discussed two variants of the DMA that can potentially contribute to other machine learning fields. One model is the novel Bayesian optimization method having deep representation and online learnability. The other model is a novel online multi-task learner with evolving deep representation.

Acknowledgments

Sang-Woo Lee would like to thank Min-Oh Heo, Christina Beak, Patrick Emaase, and Heidi Tessmer for helpful comments and editing. This work was supported by the Naver Corp. and partly by the Korean government (IITP-R0126-16-1072-SW.StarLab, KEIT-10060086-HRI.MESSI, KEIT-10044009-RISF, ADD-UD130070ID-BMRR).

References

Bendor, D., & Wilson, M. A. (2012). Biasing the content of hippocampal replay during sleep. *Nature Neuroscience*, *15*, 1439–1444.
Bettadapura, V., Essa, I., & Pantofaru, C. (2015). Egocentric field-of-view localization using first-person point-of-view devices. In *IEEE winter conference on applications of computer vision* (pp. 626–633).
Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J.Z., Rae, J., Wierstra, D., & Hassabis, D. (2016). Model-free episodic control. arXiv preprint arXiv:1606.04460.
Bottou, L. (1998). Online learning and stochastic approximations. In *On-line learning in neural networks* (pp. 9–42). Cambridge University Press, (Chapter 2).
Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade* (pp. 421–436). Springer.
Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., & Mitchell, T.M. (2010). Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI conference on artificial intelligence* (pp. 1306–1313).
Carr, M. F., Jadhav, S. P., & Frank, L. M. (2011). Hippocampal replay in the awake state: a potential substrate for memory consolidation and retrieval. *Nature Neuroscience*, *14*, 147–153.
Chen, T., Goodfellow, I., & Shlens, J. (2016). Net2net: Accelerating learning via knowledge transfer. In *International conference on learning representations*.
Chen, X., Shrivastava, A., & Gupta, A. (2013). Neil: Extracting visual knowledge from web data. In *Proceedings of the IEEE international conference on computer vision* (pp. 1409–1416).
Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31th international conference on machine learning* (pp. 647–655).
Doshi, J., Kira, Z., & Wagner, A. (2015). From deep learning to episodic memories: Creating categories of visual experiences. In *Proceedings of the third annual conference on advances in cognitive systems ACS*.
Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*, 2121–2159.
Fei-Fei, L., Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*, 594–611.
Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., & Bengio, Y. (2013). An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211.
Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *IEEE international conference on acoustics, speech and signal processing* (pp. 6645–6649).
Graves, A., Wayne, G., & Danihelka, I. (2014). Neural turing machines. arXiv preprint arXiv:1410.5401.
Guyonneau, R., VanRullen, R., & Thorpe, S. J. (2004). Temporal codes and sparse representations: a key to understanding rapid processing in the visual system. *Journal of Physiology-Paris*, *98*, 487–497.
Ha, J.-W., Kim, K.-M., & Zhang, B.-T. (2015). Automated construction of visual-linguistic knowledge via concept learning from cartoon videos. In *Proceedings of the 29th AAAI conference on artificial intelligence* (pp. 522–528).
He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385.
Heigold, G., Vanhoucke, V., Senior, A., Nguyen, P., Ranzato, M., Devin, M., & Dean, J. (2013). Multilingual acoustic models using distributed deep neural networks. In *IEEE international conference on acoustics, speech and signal processing* (pp. 8619–8623).
Hong, S., You, T., Kwak, S., & Han, B. (2015). Online tracking by learning discriminative saliency map with convolutional neural network. In *Proceedings of the 32th international conference on machine learning* (pp. 597–606).
Kim, Y.-D., Jang, T., Han, B., & Choi, S. (2016). Learning to select pre-trained deep representations with Bayesian evidence framework. In *Proceedings of the IEEE international conference on computer vision* (pp. 5318–5326).
Kim, J.-H., Lee, S.-W., Kwak, D.-H., Heo, M.-O., Kim, J., Ha, J.-W., & Zhang, B.-T. (2016). Multimodal residual learning for visual qa. arXiv preprint arXiv:1606.01455.
Knierim, J. J., & Neunuebel, J. P. (2016). Tracking the flow of hippocampal computation: Pattern separation, pattern completion, and attractor dynamics. *Neurobiology of Learning and Memory*, *129*, 38–49.
Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).
Kumaran, D., Hassabis, D., & McClelland, J. L. (2016). What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in Cognitive Sciences*, *20*, 512–534.
Lee, S.-W., Lee, C.-Y., Kwak, D.H., Kim, J., Kim, J., & Zhang, B.-T. (2016). Dual-memory deep learning architectures for lifelong learning of everyday human behaviors. In *Proceedings of the international joint conference on artificial intelligence* (pp. 1669–1675).
Lin, M., Chen, Q., & Yan, S. (2014). Network in network. In *International conference on learning representations*.
Little, R. J. (1992). Regression with missing x's: a review. *Journal of the American Statistical Association*, *87*, 1227–1237.
Liu, X., Zhang, G., Zhan, Y., & Zhu, E. (2008). An incremental feature learning algorithm based on least square support vector machine. In *Proceedings of the 2nd annual international workshop on frontiers in algorithmics* (pp. 330–338).
McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, *102*, 419.
Mitchell, T., Cohen, W., Hruschka, E., Talukdar, P., Betteridge, J., Carlson, A., Dalvi, B., Gardner, M., Kisiel, B., & Krishnamurthy, J. et al. (2015). Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI conference on artificial intelligence* (pp. 2302–2310).
Nam, H., & Han, B. (2016). Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE international conference on computer vision* (pp. 4293–4302).
Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1520–1528).
O'Reilly, R. C., Bhattacharyya, R., Howard, M. D., & Ketz, N. (2014). Complementary learning systems. *Cognitive Science*, *38*, 1229–1248.
Oza, N.C. (2005). Online bagging and boosting. In *IEEE international conference on systems, man and cybernetics* (pp. 2340–2345).
Polikar, R., Upda, L., Upda, S. S., & Honavar, V. (2001). Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, *31*, 497–508.
Ruvolo, P.L., & Eaton, E. (2013). Ella: An efficient lifelong learning algorithm. In *Proceedings of the 30th international conference on machine learning* (pp. 507–515).
Sainath, T.N., Vinyals, O., Senior, A., & Sak, H. (2015). Convolutional, long short-term memory, fully connected deep neural networks. In *IEEE international conference on acoustics, speech and signal processing* (pp. 4580–4584).
Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems* (pp. 568–576).
Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., & Adams, R. (2015). Scalable Bayesian optimization using deep neural networks. In *Proceedings of the 32nd international conference on machine learning* (pp. 2171–2180).
Sukhbaatar, S., Weston, J., Fergus, R., et al. (2015). End-to-end memory networks. In *Advances in neural information processing systems* (pp. 2440–2448).
Sundermeyer, M., Schlüter, R., & Ney, H. (2012). Lstm neural networks for language modeling. In *Interspeech* (pp. 194–197).
Sutskever, I., Vinyals, O., & Le, Q.V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp. 3104–3112).
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).
Thrun, S., & O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The tc algorithm. In *Proceedings of the 13th international conference on machine learning* (pp. 489–497).
Treves, A., & Rolls, E. T. (1992). Computational constraints suggest the need for two distinct input systems to the hippocampal ca3 network. *Hippocampus*, *2*, 189–199.
Vedaldi, A., & Lenc, K. (2015). Matconvnet – convolutional neural networks for matlab. In *Proceedings of the ACM international conference on multimedia* (pp. 689–692).
Wei, T., Wang, C., Rui, R., & Chen, C.W. (2016). Network morphism. In *Proceedings of the 33th international conference on machine learning*.
Weston, J., Chopra, S., & Bordes, A. (2014). Memory networks. In *International conference on learning representations*.
Yamins, D. L., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., & DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, *111*, 8619–8624.
Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems* (pp. 3320–3328).
Yu, H., Wang, J., Huang, Z., Yang, Y., & Xu, W. (2015). Video paragraph captioning using hierarchical recurrent neural networks. arXiv preprint arXiv:1510.07712.
Zeiler, M.D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Proceedings of European conference on computer vision* (pp. 818–833).

Zhang, B.-T. (2008). Hypernetworks: A molecular evolutionary architecture for cognitive learning and memory. *IEEE Computational Intelligence Magazine*, *3*, 49–63.

Zhang, B.-T., Ha, J.-W., & Kang, M. (2012). Sparse population code models of word learning in concept drift. In *Proceedings of the 34th annual conference of cogitive science society* (pp. 1221–1226).

Zhou, G., Sohn, K., & Lee, H. (2012). Online incremental feature learning with denoising autoencoders. In *International conference on artificial intelligence and statistics* (pp. 1453–1461).

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning* (pp. 928–936).