Personalized Object Recognition for Augmenting Human Memory

Hosub Lee
University of California, Irvine
Irvine, CA, USA
hosubl@uci.edu

Steven Eliuk
Samsung Research America
Mountain View, CA, USA
s.eliuk@samsung.com

Cameron Upright
Samsung Research America
Mountain View, CA, USA
c.upright@samsung.com

Alfred Kobsa
University of California, Irvine
Irvine, CA, USA
kobsa@uci.edu

Ubicomp/ISWC'16 Adjunct, September 12-16, 2016, Heidelberg, Germany
 ISBN 978-1-4503-4462-3/16/09…$15.00
DOI: http://dx.doi.org/10.1145/2968219.2968568

Abstract
We propose a novel wearable system that enables users to create their own object recognition system with minimal effort and utilize it to augment their memory. A client running on Google Glass collects images of objects a user is interested in, and sends them to the server with a request for a machine learning task: training or classification. The server processes the request and returns the result to Google Glass. During training, the server not only aims to build machine learning models with user generated image data, but also to update the models whenever new data is added by the user. Preliminary experimental results show that our system DeepEye is able to train the custom machine learning models in an efficient manner and to classify an image into one of 10 different user-defined categories with 97% accuracy. We also describe challenges and opportunities for the proposed system as an external memory extension aid for end users.

Author Keywords
Personalization; Object Recognition; Google Glass; Memory Enhancement; Deep Learning; Convolutional Neural Network; Finetuning

ACM Classification Keywords
H.5.1. Information interfaces and presentation (e.g., HCI): Multimedia Information Systems; I.5.4. Pattern Recognition: Computer vision.

Introduction

With recent technological advances in wearable computing, users can more easily collect information about their surroundings. For instance, users can directly capture images of objects through a smart glass rather than a smartphone. Google Glass is a representative wearable that can translate this scenario into reality. Since the camera functionality of Google Glass is always ready to be activated instantaneously by the user's command, it is reasonable to assume that more image data representing users' personal interests could be collected. Consequently, there are more chances to build a user-tailored object recognition system by utilizing those collected images as training data.

Recently, deep learning has been making a breakthrough in diverse computer vision and pattern recognition problems [6, 10]. Deep learning is a machine learning technique that attempts to extract high-level concepts from data via a complex model composed of hierarchical processing units. The trained deep learning model then utilizes the extracted concepts in making predictions on new data. Deep convolutional neural networks (CNNs), a commonly used type of network, have been widely used in the computer vision community [14]. CNNs are biologically-inspired variants of artificial neural networks, which mimic how the human brain perceives and processes images. These networks are comprised of multiple layers of filters which hierarchically process segments of the input image. Pooling layers are often added to reduce dimensionality and add translational invariance. Finally, multiple fully connected layers may be used to combine the spatial features and produce a final classification. Specifically, outputs of convolutions in the lower layers are used to represent the primitive element that forms the image (e.g., edge). Then, these representations are integrated in the higher layers to express more abstract concepts (e.g., shape) of the image. With this architecture, we can train the whole network through the standard backpropagation algorithm by using the labeled images as training data. Recent studies proved that CNN-based image classifiers have reached near-human accuracy levels on diverse visual recognition tasks [1, 5, 11, 12].

However, most deep learning applications thus far have been developed for the general population, and not for the personal needs of individuals. Imagine a professor who gives a lecture to 300 students. This professor may want to wear Google Glass displaying the students' names during the lecture because it is difficult to memorize them all. Those with mild cognitive impairments could use a Google Glass system capable of recognizing their personal objects. This would help when they have memory problems. To realize all of these scenarios, each individual user needs to have a custom machine learning model trained on her/his own image data, and utilize the model for recognizing an input image.

In this paper, we propose a novel wearable system called DeepEye that enables users to create their own CNNs without any difficulties. To begin with, DeepEye employs Google Glass for collecting image data representing users' personal interests. A user can take

images of an object of interest and apply whatever label they want. DeepEye then transmits these labeled images to the GPU[1]-equipped server to train the CNN. In general, training deep learning models like a CNN from scratch uses a considerable amount of time on modern GPUs and requires very large volumes of training data. To make the training process more practical, we utilized a well-known neural network training approach named finetuning. Finetuning was designed to quickly train a new CNN with a relatively small amount of training data by utilizing the previously trained CNN as a starting point. DeepEye can also run as an image classifier to help users recall what they are looking at through Google Glass. If the user asks DeepEye to recognize an object from an image, DeepEye shows the classification result produced by the server. The server thereby utilizes the CNN trained specifically for the user. The proposed system showed about 97% accuracy in classifying an image taken by Google Glass into one of 10 user-defined categories. We discuss the potentials and the limitations of the proposed system in helping users with memory problems.

Related Work

There were several attempts to build object recognition systems on wearable computers to reinforce users' recollection powers. Steve Mann designed and prototyped a wearable personal imaging device that could recognize human faces in an image [8]. This wearable device was also equipped with a small head-mounted display to give textual information to users. The author stated that the system could act as a visual perception enhancer because it could provide users with real-time feedback on the image they were seeing. Even though it is considered as a pioneering work of a wearable object recognition system, the prototype was cumbersome to wear (e.g., a set of communication units were attached to the user's body).

This topic has not been actively studied after the early 2000s. This is probably because there were no commercial camera-equipped wearables available, leading to less opportunities for research in both academia and industry. However, the situation may change with the advent of Google Glass. For instance, Way *et al* designed a Google Glass application named ELEPHANT for retrieving meta information about the situation (e.g., location, time, date, etc) from an image [13]. They anticipated that ELEPHANT could help people with memory impairment because it can provide contextual information when they have difficulty remembering a specific object. This work is very relevant with ours, however, the authors has not yet prototyped its machine learning functionalities for retrieving information from images taken by Google Glass.

Personalized Object Recognition System

In this section, we discuss the design and implementation of our system in detail. First, we describe an overall system architecture including software/hardware specifications. Next, the functional details of the system are explained.

System Architecture

Our system is designed as a client-server model (Figure 1). As a client, Google Glass collects images when instructed by the user, and sends them to the server with a specific task type (training or classification). The server then carries out the requested task and returns the results to Google Glass. The server was designed to continuously train (or update) the CNN using finetuning

[1] Graphics Processing Unit

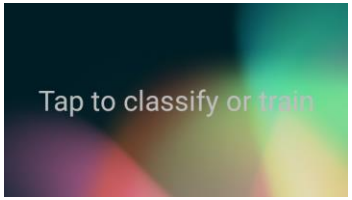

Figure 2: DeepEye initial screen

Figure 3: DeepEye training – labelling

Figure 4: DeepEye training – data collecting

whenever new image data is collected by Google Glass. When the server completes the training task, it replaces the preexisting CNN with the newly trained one that considers the most recent images. In summary, Google Glass acts as an image collector and interface which is visible to the end user. The server performs machine learning computations in the background, classifying images when needed and training new models when an object is added. We chose this architecture because Google Glass has limited computing power for training CNNs.

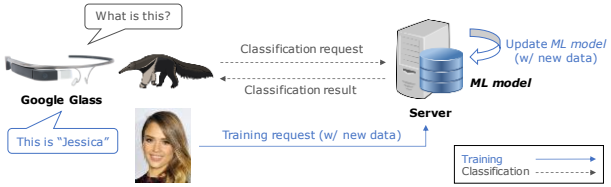

Figure 1: System architecture

CLIENT

We developed a Google Glass application named DeepEye. We wrote a function for DeepEye that takes a photo periodically upon the user's command. DeepEye sends these image data and messages to the server through Java socket communication over the Wi-Fi network. We used official Google libraries such as the Android 4.4.2 (API 19) SDK and the Glass Development Kit Preview in developing DeepEye.

SERVER

The main purpose of the server is to quickly train deep learning models with reasonable prediction accuracy. In order to achieve this, we built a Java server on a Linux workstation equipped with a modern GPU (NVIDIA GeForce GTX 970). We then deployed the state-of-the-art open source deep learning framework named Caffe [3] on the server. If the server receives a request for the specific task from DeepEye, it then executes a corresponding Caffe command (e.g., train a CNN or classify an image with a CNN) through its python interface, and returns the result.

Workflow

As discussed earlier, DeepEye has two main tasks: training and classification. Here, we describe each task step by step. When DeepEye is started, a user is asked to choose between two tasks via the Google Glass touch pad (Figure 2).

TRAINING

For the training task, the user enters the name of the target object (i.e., its label) through Google Voice Input (Figure 3). The user can try again if the result of the speech recognition was incorrect. When the user confirms the label, DeepEye begins to take a photo of the object every five seconds, and transmits it with a message representing the current task (`_train`) to the server. This process is repeated as long as DeepEye receives an ACK message from the server and the user has not explicitly terminated the training task (Figure 4). The server will use the transferred image data for training a deep learning model via finetuning.

As discussed, training deep learning models from scratch is very expensive and time-consuming. For example, training a CNN on the ImageNet dataset which contains 1.2 million images with 1,000 categories can take several weeks on a single GPU or hours/days in a distributed setting [5]. For these reasons, it is more common to train a new model by recycling the fully trained model on a larger dataset if we need to

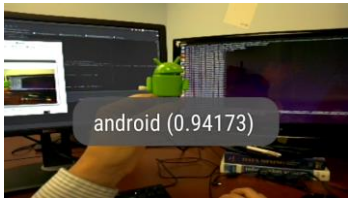

Figure 5: DeepEye classification

repurpose an existing model for different tasks [4]. For instance, we can exploit the pre-trained CNN's well-learned parameters representing generic visual features like edges at the beginning of the training. Then, we can focus on updating parameters which were designed to extract more object-specific (high-level) features from our image data. This approach is known as finetuning, one kind of transfer learning algorithm. Finetuning is widely used to avoid expensive training efforts in diverse machine learning tasks [9].

By using finetuning as a main building block, our server program works as follows. To train a new CNN model for a new task, it iteratively retrains the pre-trained CNN on a newly created dataset. Let's assume there exists a CNN trained to classify an image into three user-defined categories A, B, and C (CNN_ABC). If a user adds a new category D with the corresponding image data, the server constructs a new model (CNN_ABCD) on new training data while using the old model (CNN_ABC) as a starting point. More specifically, the server defines a new CNN by adopting an under-lying network architecture of the pre-trained CNN, but changes its classification layer to have a correct number of outputs based on the given task (e.g., 4 output nodes for CNN_ABCD). Next, the server initializes parameters (weights) of the new CNN with that of the pre-trained CNN, then progressively updates the weights of the new CNN through the back-propagation algorithm, on a new dataset. This process can be continued whenever new types of training data became available.

The training process begins if there are at least two user-defined categories with a sufficient amount of training data. Through repeated experiments, we determined that 100 images per class represent a sufficient threshold for the training data. The process also checks whether there are any ongoing CNN training processes on the system. If training is already in progress, it will not try to train a new model until the ongoing process has ended. Next, if this is the first finetuning attempt, it trains a new model by using the pre-trained CNN named CaffeNet. We utilized CaffeNet as a base model because it is a publicly available pre-trained CNN that has a reasonable performance on 1,000 class object recognition task (ImageNet challenge [5]). Otherwise, it trains a new model by finetuning the pre-trained CNN on the previous finetuning stage. When a single finetuning process has finished, the previously trained CNN is replaced with the newly trained CNN.

CLASSIFICATION

Classification is relatively simple. When a user chooses the classification task, they take a picture of the object by clicking the Google Glass touch pad. Similar to the training task, DeepEye sends the image to the server, but with a different message (`_classify`). Next, the server uses the latest trained CNN to execute the Caffe classification command on the image. If no error occurs, the server sends the classification result (with probability) back to DeepEye. If DeepEye receives the result from the server, it displays them to the user through Google Glass's heads-up display (Figure 5).

Experiment: 10 Class Object Recognition

Overview

In this preliminary experiment, we aimed to evaluate the prediction power of finetuned CNNs in a real world scenario. To this end, we trained a CNN so that it can recognize 10 different types of objects from images

Figure 6: Sample training image

Figure 7: Sample validation image

taken by Google Glass. The ultimate aim of such a system would be to help people with memory problems to recognize their personal belongings.

Training and Validation Data

To begin with, we chose 10 personal objects (small toy, badge, baseball cap, key, glass, pouch, food container, lotion, watch, wallet) of a member of our research team, and collected images using the proposed system consisting of DeepEye and the server. We collected the exact same amount of training data for each class, namely 100 images. We also augmented training data by creating additional image transformations using ImageMagick's `convert` tool. Specifically, we created four variations of each original image that were rotated by 90, 180, and 270 degrees, and were mirrored. We included this step to alleviate potential overfitting problems as much as possible by providing more training data without extra labelling cost (data augmentation [2, 5]).

We also collected 30 additional images for each class as validation data. To differentiate these from original training data, we deliberately changed the photographing conditions such as lighting, angle and background (see Figure 6, 7). Both training and validation images were taken by a single participant in a standard office setting. Even though Google Glass is equipped with a 5MP camera capable of taking 2,560 by 1,888 resolution JPG images with a file size of about 2 megabytes, we collected reduced-size versions of the images (1296 by 972 pixels) to avoid any network delays between DeepEye and the server.

Finetuned Model (# of classes)	Base Model (# of classes)	Accuracy (loss)
CNN (3)	CaffeNet (1000)	0.99 (0.0212)
CNN (4)	CNN (3)	0.99 (0.1819)
CNN (5)	CNN (4)	0.99 (0.0555)
CNN (6)	CNN (5)	0.99 (0.0462)
CNN (7)	CNN (6)	0.99 (0.0454)
CNN (8)	CNN (7)	0.96 (0.2696)
CNN (9)	CNN (8)	0.94 (0.2319)
CNN (10)	CNN (9)	0.97 (0.116)

Table 1: 10 class object classification – validation accuracy

Result

Table 1 summarizes the measured prediction power of all finetuned CNNs on the validation data set. For up to 7 different objects, the trained CNNs showed a near perfect performance in recognizing objects without any serious overfitting concerns. However, the validation accuracy was slightly diminished as the number of object categories increases from 8 to 9. It may be improved if we collect additional training images of the corresponding objects, and train a new CNN at the next finetuning stage. The final trained CNN's validation accuracy was 97% with a loss of 0.116, and took approximately 7 minutes to train this model on our GPU environment.

Discussion and Future Work

Google Glass

We showed the feasibility of the proposed object recognition system via Google Glass. Yet, it still has some issues that need to be overcome before practical use. Google Glass emits a lot of heat when it continuously utilizes the camera function. According to [7], a single camera shot heats Google Glass 28°C

above the surrounding temperature. Because Google Glass is in direct contact with the skin, the heated surface may lead to discomfort and potentially even health problems for users. Therefore, users may have trouble collecting multiple images at once via Google Glass. We expect Google to fix these issues in the next generation of Google Glass.

Scalability and Applicability
At this moment, there exists no large-scale image dataset collected from wearable computers such as Google Glass. Therefore, we generated the custom dataset using DeepEye in our experiment, and utilized it for testing the proposed system. However, we will need to evaluate it with larger image datasets. Specifically, it is necessary to investigate whether the proposed system also works well for more complex image classification problems (e.g., 100 class object recognition). Also we need to test the system on different types of object recognition tasks, such as face recognition for the abovementioned university professor, to show its applicability in various situations.

All of these steps are important to verify that the system could support users' daily memory-related tasks. Thus, we are considering distributing DeepEye to a group of Google Glass users, and to collect more diverse image data from their everyday lives. This will let us conduct additional experiments to gauge the scalability and applicability of the proposed system.

Effectiveness
We believe general users are unlikely to have difficulty using the proposed system because they are only asked to perform some simple operations via Google Glass (e.g., image labelling through Google Voice Input). However, we need to check the effectiveness of the system with target users who have special needs. More specifically, we need to quantitatively and qualitatively assess the usability of the system for people with memory disorders (e.g., mild cognitive impairment), possibly including their caregivers. Their feedback may allow us to improve our user interface. More importantly, a longitudinal study needs to be conducted to verify whether the proposed system can improve their memory and cognitive abilities. To that end, we might need to collaborate with medical professionals.

Conclusion
In this paper, we proposed and prototyped a novel wearable system DeepEye that is aimed at augmenting human memory. To that end, the system builds personalized deep learning models for recognizing objects of interest to a user. To the best of our knowledge, this is the first attempt to train machine (deep) learning models for personalized object recognition, via camera-equipped wearable computers like Google Glass. The proposed system works as a client-server model: the Google Glass client collects images from a user's everyday life and sends them to a GPU-equipped Linux server. The server then trains a deep convolutional neural network (CNN) on the user-specific image data. We utilized finetuning to efficiently update the pre-trained network on newly-added image data. In the custom 10 class object recognition task, DeepEye trained a personalized CNN within 7 minutes on our GPU and showed a 97% classification accuracy. We plan to test the system with more complex object recognition tasks and to verify its effectiveness in augmenting human memory and perception.

Acknowledgements

Part of this work was done while Hosub Lee was a summer intern at Samsung Research America, Mountain View, CA.

References

1. Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. 2012. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 3642–3649.
2. Andrew G Howard. 2013. Some improvements on deep convolutional neural network based image classification. *arXiv preprint arXiv:1312.5402* (2013).
3. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*. ACM, 675–678.
4. Sergey Karayev, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. 2014. Recognizing image style. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press.
5. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
6. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
7. Robert LiKamWa, Zhen Wang, Aaron Carroll, Felix Xiaozhu Lin, and Lin Zhong. 2014. Draining our glass: An energy and heat characterization of google glass. In *Proceedings of 5th Asia-Pacific Workshop on Systems*. ACM, 10.
8. Steve Mann. 1997. Wearable computing: A first step toward personal imaging. *Computer* 30, 2 (1997), 25–32.
9. Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on* 22, 10 (2010), 1345–1359.
10. Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61 (2015), 85–117.
11. Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. 2014. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*. CBLS.
12. Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*. CBLS.
13. Thomas Way, Adam Bemiller, Raghavender Mysari, and Corinne Reimers. 2015. Using Google Glass and Machine Learning to Assist People with Memory Deficiencies. In *Proceedings on the International Conference on Artificial Intelligence (ICAI)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 571.
14. Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer vision–ECCV 2014*. Springer, 818–833.