Fine-grained and Continual Visual Recognition for Assisting Visually Impaired People

MARCUS KLASSON

Doctoral Thesis
Stockholm, Sweden, 2022

KTH Royal Institute of Technology
School of Electrical Engineering and Computer Science
Division of Robotics, Perception, and Learning
SE-10044 Stockholm, Sweden

TRITA-EECS-AVL-2020:4
ISBN 100-

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av Teknologie doktorexamen i elektroteknik fredagen den 18 januari 2022 klockan 14.00 i Sal F3, Lindstedtsvägen 26, Kungliga Tekniska Högskolan, Stockholm.

Tryck: Universitetsservice US AB

Abstract

In recent years, computer vision-based assistive systems have enabled visually impaired people to use automatic object recognition on their mobile phones. These systems should be capable of recognizing objects that are important for the user on a fine-grained level. To this end, we have focused on the particular application of classifying food items which can be challenging for blind/low-vision people since visual information is often required for distinguishing between similar items. In Paper A, we present a challenging image dataset of groceries taken in grocery stores where each item is hierarchically labeled to capture the fine-grained structure of the various items. Furthermore, we demonstrate in Paper B how more easily accessible information about the items, such as web-scraped images and text descriptions, can be utilized for enhancing the classification performance of groceries compared to only using the real-world images for training.

A valuable feature of assistive vision systems is the capability of adapting to new object classes. The main challenge here is to avoid catastrophically forgetting previously learned knowledge when the classifier is updated with new classes. In Paper C, we propose a new continual learning setting for replay-based methods that aligns well with real-world needs where constraints are placed on processing time rather than the storage capacity of old samples. We then study the timing of replaying certain tasks and show that learning replay schedules over which tasks to replay can be critical for the final classification performance in our proposed setting. Finally, in Paper D, we present a method based on reinforcement learning for learning a policy for selecting which tasks to replay at different times. The benefit of our learned replay scheduling policy is that it can be applied to any new continual learning scenario for mitigating catastrophic forgetting in a classifier without additional computational cost.

To conclude, I will discuss some potential future directions for the development of the next generation of computer vision-based assistive technologies.

Keywords: Visual Recognition, Fine-grained Classification, Continual Learning, Visually Impaired People, Assistive Technologies

Sammanfattning

hej

List of Papers

A ***A Hierarchical Grocery Store Image Dataset with Visual and Semantic Labels***
Marcus Klasson, Cheng Zhang, Hedvig Kjellström
In *IEEE Winter Conference on Applications of Computer Vision (2019)*

B ***Using Variational Multi-view Learning for Classification of Grocery Items***
Marcus Klasson, Cheng Zhang, Hedvig Kjellström
In *Patterns, Volume 1(8) (2020)*

C ***Learn the Time to Learn: Replay Scheduling for Continual Learning***
Marcus Klasson, Hedvig Kjellström, Cheng Zhang
Under submission

D ***Meta Policy Learning for Replay Scheduling in Continual Learning***
Marcus Klasson, Hedvig Kjellström, Cheng Zhang
Under preparation

Acknowledgements

hej

Acronyms

List of commonly used acronyms:

CL	Continual Learning
CNN	Convolutional Neural Network
FGIR	Fine-Grained Image Recognition
LSTM	Long Short-Term Memory
MCTS	Monte Carlo Tree Search
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
VAE	Variational Autoencoder
VCCA	Variational Canonical Correlation Analysis
VI	Vision Impairment

Contents

Part I

Overview

Chapter 1

Introduction

Vision is probably the most important of all senses that humans possess. Our society is built on having this ability. For example, if we would like to cross a street, there are thick colored stripes on the road or signs above head height that indicate where the cross walk is located such that we can cross the street in an appropriate way. Another example is how we use text to communicate with each other, where words and sentences are composed by structured sequences of symbols that constitute a specific language. Furthermore, it has been shown that learning from both images and text can improve comprehension over learning from text only [1,2]. Possessing normal vision capabilities basically make everyday tasks easier when it comes to reaching destinations in the world, communicating with other people, and learning new concepts.

In 2020, it was estimated to be 43.3 million people who are blind and 295 million people with moderate to severe visual impairment in the world [3]. To enhance the mobility of visually impaired (VI) people, there exist various kinds of assistive devices and tools, such as screen readers and Braille typewriter machines, for supporting them with receiving information and communicating through text. More recently, several computer vision-based assistive vision tools have emerged in the form of wearable devices and mobile applications for helping VIs with tasks where visual information is a must, for example, object recognition [4–6] and wayfinding in natural environments [7–9] and .

Despite the recent successes in computer vision [10–12], these methods can face several challenges when deployed in the real-world which makes their recognition performance suffer. For example, it can be difficult for the methods to distinguish between similar items on a fine-grained level, such as different brands of apples and pears, as well as performing robustly in environments with noisy backgrounds and poor lighting. Part of the reason for such challenges is that specifying a model of the visual world that has been injected with knowledge about the rich complexity that can exist in images is very difficult [13]. Therefore, there is a necessity for developing computer vision methods that can recognize different appearances of objects, adapt to changes of known objects, and learn what new objects look like. At the same time, these tasks should be possible to execute in a time-efficient and robust manner.

In this thesis, we address the challenges on robustness in fine-grained classification as well as how the method can learn to recognize new object classes. We will begin this introduction by briefly describing vision impairments in Section 1.1, followed by a summary of assistive vision technologires in Section 1.2. Then we describe the scope of the thesis in Section 1.3 and summarize the contributions of the included papers in Section 1.4. Finally, in Section 1.5, we give the outline to the rest of the contents in this thesis.

1.1 Vision Impairments

Vision impairment (VI) is defined as the decrease of one's ability to see from various distances [14]. There are different types of VIs ranging from various degrees of blindness to having issues with seeing from far or near distances. The visual capabilities are in general assessed by measuring the *visual acuity* (sharpness) of seeing, for example, a letter or symbol, from some fixed distance. The visual acuity measured differently based on whether near- or far-sighted VI is assessed. For far-sighted VI, the visual acuity is calculated by the ratio between the distance that the subject can see the item and the distance a normal-sighted person could recognize the item. When assessing near-sighted VI, one checks the font size of letters that the subject can see using a standardized point system for measuring the symbol size [15].

In 2020, it was estimated that 338 million people possess moderate to severe VI globally, including 43 million people that are blind [3]. Furthermore, the World Health Organization (WHO) have estimated that at least 2.2 billion people live with a near or distance VI, where at least 1 billion cases could have been prevented or yet has to be addressed [15]. The untreated cases are projected to grow to 1.7 billion people by 2050 mainly due to population growth in the world as well as increased aging among the populations [3]. The leading causes for vision loss are uncorrected refractive errors, untreated cataracts, age-related macular degenerationm, glaucoma, diabetic retinopathy, where 90% of such cases are preventable and treatable [16]. The causes for vision loss also differs between countries and areas with different incomes.

There exists several tools for assisting VI people with everyday tasks. The *white cane* is probably the most common tool among VI people which is used for wayfinding to help the user anticipate what is present in their near surroundings. Also, guiding dogs are used for enhancing mobility by helping VI people to maintain a direct route, avoid obstacles, and prepares owner by stopping at curbs and stairways until they are told to proceed [17]. There also exist several tools for recognition tasks. For example, currency markers are used for keeping track of different bills in wallets, color indicators can be used to tell the user of the color of clothes, and labeling apparatus are used for distinguishing between similar items. Means for communication also exists in the form of Braille keybords and screen readers that are used in both computers and mobile phones to provide nearly equal opportunities for VI people when it comes to office-related tasks. There has been a recent emergence of various devices that are aimed to assist VI people with object recognition tasks which we will discuss next.

1.2 Assistive Vision Technologies

Cameras are used by people with VIs, including blindness, to record events and memories similarly as normal-sighted people [18]. This has opened up for opportunities where VI people can use their cameras for more than recording events, for example, object recognition, document text recognition, and color identification. Object recognition has been shown to be considered an everyday challenge, where VI people would like to ask questions about objects where visual information is necessary for identification [19]. For example, it can be very difficult to distinguish between different containers and packages that have similar shapes but different content without being able to see. These findings have encouraged development of technical aids that use computer vision for assisting VI people.

In the last decade, we have seen several variants of assistive vision technologies emerging on the market. There exist many applications for mobile phones where various visual tasks have been cramped in into the app, such as object and face recognition, barcode scanning, color and currency identification, and text recognition [20–23]. Moreover, there exists wearable devices with similar capabilities as the mobile phone apps [24,25] that also use computer vision for assistance. An alternative to the computer vision-based apps there are other mobile applications where VI users can have a video call with sighted volunteers that help them with any kind of task requiring visual capabilities [26,27]. Despite that these assistive vision technologies has opened up for VI people being more independent, there remains several challenges to tackle regarding system requirements [28–30] and privacy concerns [31–33].

Current assistive vision technologies face several challenges that needs to be tackled to enhance their utility for VI people. In the past decade, machine learning techniques have been applied successfully to various computer vision tasks such as object recognition [10,34,35], generating scene descriptions [12,36,37], and visual question answering [38–40]. In addition to better computer hardware, the main reason for these successes is the immense data collection and annotation that is required for obtaining large-scale computer vision datasets. However, the annotation becomes even more costly if the object classes should be separated based on fine-grained details about the objects, which makes it challenging for assistive vision systems to provide users with further information about objects than the general object class. Another challenge is how to update the assistive vision devices with information about new objects to recognize and ensuring that the system is still able to recognize the previous known items correctly. Furthermore, assistive vision devices should have the ability to answer questions about the surroundings of the user, should perform in real-time and be robust when applied in different environments, as well as ensuring privacy for the user.

1.3 Scope of Thesis

This thesis is focused on two applications for machine learning and computer vision-based assistive technologies, namely *fine-grained classification* [41] and *continual learning* [42, 43]. Fine-grained classification involves identifying subcategories and details of general object classes, which can be important when distinguishing between visually similar items. An example is when one has to distinguish between two juice packages from the same brand where the main ingredients are apples and oranges in the packages. The general setting in fine-grained classification is that all data and classes to learn are given all at once to the classifier to learn, but can be extended to the continual learning setting where the classes to learn are divided into tasks that are learned at different points in time. Continual learning methods are used for updating the classifier's current knowledge with information about the new classes and making sure that the classifier remembers the previously learned classes. The common denominator of these fields is classification, but both have challenges of their own that has to be addressed before adding such features into assistive vision devices. Next, we describe the challenges that we have focused on in this thesis.

Fine-grained Classification

One main challenge for fine-grained classification is the data collection procedure and there are several reasons for this. Firstly, the annotation of the collected data becomes more time-consuming as the annotators must know specific details about the objects to label the data as accurately as possible. Secondly, as fine-grained classes might be rare, there might be few examples per class that the classifier can learn from to discriminate between the objects. An application where an assistive vision device would need to learn fine-grained classes from sparse datasets is grocery shopping for helping VI people [5, 44]. Grocery items usually require visual information to distinguish between them, for example, when one needs to know how the ingredients differ in two juice packages. This also goes for raw grocery items where it might be difficult for a VI customer to tell the difference between two different brands of green apples unless the customer knows how the apples smell or how the texture of their peel differs when touching them. Furthermore, situations in the grocery store environment can disturb the recognition performance of the assistive vision device, for example, when multiple and misplaced items appear in the camera view and also when there are poor lighting settings in some areas of the store. Collecting training data that covers all possible scenarios that can occur in the store would be a cumbersome procedure. Our goal is to reduce the need for training data in the grocery stores by collecting web-scraped information about the items and using this for easing the learning of the classifier.

Continual Learning

The main challenge in continual learning is called *catastrophic forgetting* [45] which means that the classifier will overwrite previously learned knowledge with information about the new objects of interest during learning. Therefore, we must use additional training techniques that prevents this forgetting effect to maintain the recognition performance on all classes during the lifespan of the classifier. A simple yet efficient approach in continual learning for mitigating catastrophic forgetting is replay-based methods [46, 47]. The main assumption is that we are allowed to keep a low number of examples from every seen class in a small memory buffer. The idea is then to mix the old examples with the training data from new classes, such that we learn the new classes and aim to retain the performance on the old classes by replaying the memory examples for the classifier.

Most previous works on replay-based continual learning ignores the time to replay certain tasks. However, the timing of rehearsal has been shown to be very important for humans to retain memory on various tasks [48–52]. Moreover, in contrast to the constraint on the small memory size, machine learning systems used in real-world applications may be limited by processing times rather than data storage capacity [Add REFs]. In such settings, there is a need for methods that select what data from the huge storage to replay as the problem of catastrophic forgetting still remains. Our goal is to demonstrate that scheduling over which tasks to replay can be crucial for continual learning performance in this setting. Hence, we will need to develop efficient methods that can automatically propose replay schedules that mitigate catastophic forgetting in classifiers to enable this strategy in real-world settings.

1.4 Thesis Contributions

In this section, we provide summaries of the included papers as well as stating the contributions of each author to the manuscripts.

Paper A: A Hierarchical Grocery Store Image Dataset with Visual and Semantic Labels

Marcus Klasson, Cheng Zhang, Hedvig Kjellström. In *IEEE Winter Conference on Applications of Computer Vision (WACV) 2019.*

Summary We collect a dataset with natural images of raw and refrigerated grocery items taken in grocery stores in Stockholm, Sweden, for evaluating image

Table 1.1: Examples of grocery item classes in the Grocery Store dataset. We display four different items (coarse-grained class in parenthesis), followed by two natural images taken with a mobile phone inside grocery stores. Next comes the web-scraped information of the items consisting of an iconic image and a text description. We have highlighted ingredients and flavors in the text description that are characteristic for the specific item.

Class Labels	**Natural Images**	**Iconic Images**	**Text Descriptions**
Granny Smith (Apple)			*"...**green** apple with **white, firm** pulp and a **clear acidity** in the flavor."*
Tropicana Mandarin (Juice)			*". . . is a **ready to drink** juice **without pulp** pressed on **orange**, **mandarin** and **grapes**. Not from concentrate. Mildly **pasteurized**."*

classification models on a challenging real-world scenario. The data collection was performed by taking photos of groceries with a mobile phone to simulate a scenario of grocery shopping using an assistive vision app. Furthermore, we downloaded iconic images and text descriptions of each grocery item by web-scraping a grocery store website to enhance the dataset with information describing the semantics of each individual item. The items are grouped based on their type, e.g., apple, juice, etc., to provide the dataset with a hierarchical labeling structure. We show two examples of grocery item classes and their corresponding web-scraped information in Table 1.1.

We provide benchmark results evaluated using pre-trained and fine-tuned CNNs for image classification. Moreover, we take an initial step towards utilizing the rich product information in the dataset by training the classifiers with representations where both natural and iconic images have been combined through a multi-view VAE.

Author Contributions CZ and HK presented the idea and the data collection procedure for the natural images and web-scraped information. MK performed the data collection including visiting the grocery stores for taking the natural images and the web-scraping of the grocery store website for iconic images and text descriptions. MK performed all the experiments. All authors contributed to discussing the results and contributed to writing the manuscript.

Paper B: Using Variational Multi-View Learning for Classification of Grocery Items

Marcus Klasson, Cheng Zhang, Hedvig Kjellström. In *Patterns, Volume 1(8) (2020).*

Summary We investigate whether training image classifiers can benefit from learning joint representations of grocery items using multi-view learning over the natural images and web-scraped information of the grocery items in the Grocery Store dataset (see Paper 1.4). We employ a deep multi-view model based on VAEs called Variational Canonical Correlation Analysis (VCCA) [53] for learning joint representations of the different data types, i.e., natural images, iconic images, and text descriptions. We performed a thorough ablation study over all data types to demonstrate how they contribute individually to enhancing the classification performance. Furthermore, we apply two classification approaches where we (i) train the classifier on the joint latent representations, and (ii) using a generative classifier by incorporating a class decoder to the VCCA model that can be used for classifying images.

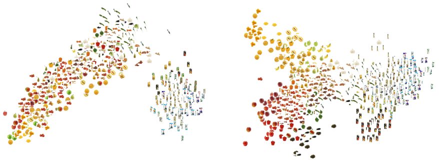

(a) VAE$_{\boldsymbol{x}}$ (b) VCCA$_{\boldsymbol{xiwy}}$

Figure 1.1: Visualizations of the latent representations projected in 2D space with PCA from models VAE$_{\boldsymbol{x}}$ and VCCA$_{\boldsymbol{xiwy}}$, where we plot the corresponding iconic image for each latent representation. We observe that VCCA$_{\boldsymbol{xiwy}}$ structures the items based on visual similarities by incorporating the web-scraped information into the learning.

We performed a thorough ablation study over all data types to demonstrate how they contribute individually to enhancing the classification performance. To gain further insights into our results, we visualized the learned representations of the grocery items from VCCA and discussed how the iconic images and text descriptions help the model to better distinguish between the groceries. Our results show that the iconic images help to group the items based on their color and shape while text descriptions separate the items based on differences in ingredients and flavor. Figure 1.1 shows visualizations of the latent representations projected in 2-dimensional space using Principal Component Analysis (PCA), where we illustrate how the latents change when adding the iconic image and text description into the VCCA model. Finally, we concluded that utilizing the iconic images and text descriptions yielded better classification results than only using natural images.

Author Contributions CZ and HK presented the idea and all authors contributed to formalizing the methodology. MK performed all the experiments and created the visualizations. All authors took part in discussing the results. All authors contributed to writing the manuscript.

Paper C: Learn the Time to Learn: Replay Scheduling for Continual Learning

Marcus Klasson, Hedvig Kjellström, Cheng Zhang. Submitted to *International Conference on Machine Learning (ICML) 2022.*

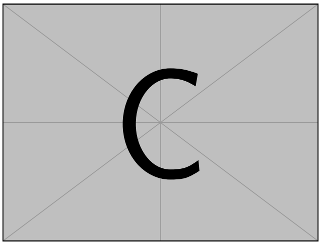


Figure 1.2: **MK: Paper C could illustrate the new CL setup with a figure that shows a network and a scheduler that needs to fetch a small replay memory from a huge pool of historical data.**

Summary In this paper, we show that learning the time to replay different tasks can be critical for continual learning (CL) performance in replay-based methods. As the main assumption in replay-based CL is that only a small set of historical data can be re-visited for mitigating catastrophic forgetting, most works have focused on improving the sample quality of the replay memory. However, in many real-world applications, historical data is accessible at all times, e.g., by storing it on the cloud. But although all historical data could be stored, retraining machine learning systems on a daily basis is prohibitive due to processing times and operational costs. Therefore, small replay memories are still needed in CL to mitigate catastrophic forgetting when learning new tasks. To this end, we propose to learn the time to learn for a CL system, in which we learn schedules over which tasks to replay at different times. Inspired by human learning, we demonstrate that scheduling over the time to replay is critical to the final CL performance with finite memory resources. We then illustrate our idea with scheduling over which tasks to replay by learning such policy with Monte Carlo tree search. We perform extensive evaluation showing that learning replay schedules can significantly improve the performance compared to baselines without learned scheduling. We also show that our method can be combined with any replay-based method and memory selection technique. Finally, our results indicate that the learned schedules are also consistent with human learning insights.

Author Contributions CZ presented the idea and MK and CZ contributed to formalizing the methodology. MK performed all the experiments. All authors took part in discussing the results and contributed to writing the manuscript.

Paper D: Meta Policy Learning for Replay Scheduling in Continual Learning

Marcus Klasson, Hedvig Kjellström, Cheng Zhang. Under preparation for conference submission.

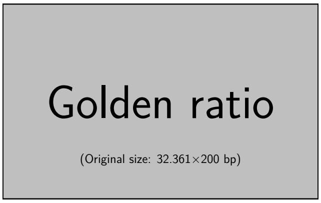


Figure 1.3: **MK: Paper D could show illustration of the RL agent scheduler that gets performance measures as input and outputs an action proportion of how to select the replay memory. RL agent could also have a replay buffer where data is collected from several environments. Maybe it can also show the test case, so that it would be two separated ”at training/test phase”.**

Summary In this paper, we propose a reinforcement learning-based method for learning policies for replay scheduling that can be applied in new continual learning scenarios. We demonstrated in Paper C that learning the time to replay different tasks is important in continual learning. However, a replay scheduling policy that can be applied in any continual learning scenario is currently absent, which makes replay scheduling infeasible in real-world scenarios. To this end, we propose using reinforcement learning to enable learning general policies that can generalize across different data domains. The learned policy can then propose replay schedule that efficiently mitigate catastrophic forgetting to improve the continual learning performance without any additional computational cost in the new domain. We compare the learned policies to several replay scheduling baselines and show that the learned policies can improve the continual learning performance given task orders and datasets unseen during training.

Author Contributions CZ presented the idea and MK and CZ contributed to formalizing the methodology. MK performed all the experiments. All authors took part in discussing the results and contributed to writing the manuscript.

1.5 Thesis Outline

The rest of the thesis is organized as follows. We provide some background and preliminaries on the machine learning methodology used in this thesis in Chapter 2. Chapter 3 is focused on our contributions in fine-grained classification where we first describe the related work in this field followed by explaining the frameworks used in our work. Similarly, in Chapter 4, we focus on our contributions in continual learning by first describing the related work to place our contributions in context and then describing the framework we used for enabling replay scheduling in continual learning. Finally, in Chapter 5, we provide our conclusions of the presented works and present some research directions that we believe would be interesting to look deeper into in the future.

MK: todo: fix references to chapters!

Chapter 2

Background

The goal with this chapter is to provide the reader with preliminaries that are useful for comprehending the included papers. We assume that the reader has some knowledge in calculus, linear algebra, and probability theory, but we intend to keep it on a basic level. First, we give the notation that will be used throughout the thesis. Then, we will introduce a selection of related works to place the thesis into context.

2.1 Notation and Terminology

We will begin by providing some algebraic notation that will be used for representing various types of data in the thesis. Scalars (both integer and real) are denoted by italic letters such as a. Vectors are denoted by lowercase bold italic letters such as $\boldsymbol{x}$, where all vectors are assumed to be column vectors. A superscript T denotes the transpose of a vector or matrix, such that $\boldsymbol{x}^T$ becomes a row vector. Matrices are denoted as uppercase bold italic letters such as $\boldsymbol{W}$. The notation $(w_1, \ldots, w_m)$ denotes a row vector with m elements, where the corresponding column vector is denoted as $(w_1, \ldots, w_m)^T$.

A dataset is denoted by the set $\mathcal{D} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$, where $\boldsymbol{x}^{(i)}$ is the i-th example among the N data points. Each data point is assumed to belong in a space of vectors denoted by $\mathcal{X}$, such that $\boldsymbol{x} \in \mathcal{X}$. The data generating distribution is denoted by $p_{data}(\mathcal{X})$ which is usually unknown. To provide an example, we let the vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ represent a flattened image of m pixels. In this case, all possible images that can exist belong to the space $\mathcal{X}$ and the data generating distribution $p_{data}(\mathcal{X})$ gives the probability of how likely each image is to occur in the world. In supervised learning, there is also a target, either denoted as $y^{(i)}$ or $\boldsymbol{y}^{(i)}$, associated with $\boldsymbol{x}^{(i)}$. The target belongs to the target space $\mathcal{Y}$, where the space is discrete $\mathcal{Y} = \{1, \ldots, C\}$ for classification tasks over C number of classes, or continuous $\mathcal{Y} = (-\infty, \infty)$ over an interval of real values for regression tasks.

Throughout this thesis, we take a machine learning approach to solve tasks by tuning an adaptive model using a dataset called the *training set*. In our case, we will represent the model with a function $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ that allows us to predict outcomes of events/data $\boldsymbol{x}$ from the task of interest. The parameters $\boldsymbol{\theta}$ expresses the function and we use machine learning algorithms for tuning parameters with the given dataset during the training phase. Once the model is trained, we often enter the *test phase* where we want to evaluate the model by predicting outcomes on an unseen dataset called the *test set*. The ability to predict outcomes of new data that is different from the examples seen during training is called *generalization*, which is a central goal for most applications in machine learning and pattern recognition.

2.2 Problem Settings in Machine Learning

Machine learning problems can be divided into three main fields, namely, *supervised learning*, *unsupervised learning*, and *reinforcement learning* (RL). Since this thesis includes work from each of these problem settings, we will briefly introduce these topics to provide the reader with context on the tasks that we are trying to solve.

MK: TO-DO: Add references to papers and sections!

Supervised Learning. In this setting, we are given a dataset $\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N}$ where each data example $\boldsymbol{x}^{(i)}$ is accompanied with a target $\boldsymbol{y}^{(i)}$. The goal is to estimate a function $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ that assigns the correct target to each example in the training set as accurately as possible. In classification problems, each target belongs to one of K discrete categories, such that $\boldsymbol{y} = \{1, \dots, K\}$, and we want to predict which of the categories that new data belongs to. Classification tasks will be involved in all included papers of this thesis wherein Paper A and B we focus on assigning the correct product category to images of grocery items. Another problem type in supervised learning is regression where the targets are continuous and real-valued. An example of a regression task is to predict the outdoors temperature tomorrow given the observed temperature today.

Unsupervised Learning. Here, we are given a dataset $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ without access to any corresponding targets. The goal in this unsupervised setting may then be to find hidden structures in the given dataset. For example, we might be interested in discovering groups of similar examples with *clustering* techniques, or we may want to use *density estimation* where we approximate the true data distribution p_{data} with a parametric distribution $p_{\boldsymbol{\theta}}$ using the collected dataset, or we may want to project high-dimensional data into two or three dimensions for *visualization* purposes. We will get back to these goals when we introduce representation learning in Section X.

Reinforcement Learning. For these problems, we have a *learning agent* that wants reach a goal in an environment by performing a given set of actions. After performing an action, the agent observes the state of the environment and receives a reward from the environment saying how good or bad the taken action was to reach the goal. The objective for the agent is to maximize the reward signal within the time the agent reaches the goal. The agent then has to learn a policy for deciding which actions to perform in certain situations in the environment. The policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s})$ is a mapping from perceived states $\boldsymbol{s}$ in the environment to actions $\boldsymbol{a}$ that should maximize the reward signal. An example of a task that can be framed as a RL problem is the so called Mountain Car problem, where the agent is a car that is trying to drive up to the top of a hill. The state represents the position and velocity of the car and the agent must take actions that will move the car forward or backwards. The objective is to reach the goal with as little time as possible, and the agent is encouraged to do so by the environment by sending the agent a negative reward for every time step that passes without reaching the top of the hill. We will return to the RL framework later when we describe the prerequisites for Paper D.

There exist many different methods for solving problems within these three fields. In this thesis, we employ deep learning methods which has been successfully applied in each field by representing the models with deep neural networks [10, 54, 55].

2.3 Deep Learning

In this section, we give a brief overview of deep learning [56] which is the main building block for the models we use in this thesis. Deep learning contains a family of machine learning models based on neural networks that are parametric function approximators used for representing some function of interest. Much of the successes of deep learning methods have been in supervised learning settings, especially in applications where there are large amounts of labelled data and sufficiently large model in terms of number of parameters in the network. In the following sections, we will introduce the deep learning frameworks and models that we have used in this thesis.

Deep Neural Networks

Neural networks is a class of machine learning models which popularity have grown immensely due to their ability to learn from large and high-dimensional datasets. Moreover, neural networks have been successfully applied in various number of fields in computer vision [10, 34], natural language processing [57], and reinforcement learning [55, 58]. These models are constructed by stacking layers of parameters that extract intermediate representations of the input data. The last layer outputs the target answer from the queried input and is specific for the task. For example, in image classification, the last layer outputs class scores representing which class the image is most likely to belong to.

Next, we will describe three popular types of neural networks, namely, multilayer perceptrons (MLPs), convolutional neural networks (CNNs), and recurrent neural networks (RNNs). **MK: TO-DO: It would be nice to have some simple illustrations of how the networks process the data, I can take inspiration from how other people have done it.**

Multilayer Perceptrons. The simplest form of feedforward neural networks is the MLP. Let the vector $\boldsymbol{x} = (x_1, \dots, x_d)^T$ be some form of data where x_i is the i-th feature for $i = 1, \dots, d$. Every layer in the MLP constitutes of weights $\boldsymbol{W}$ that are used for transforming the input such that the output reveals some hidden structure useful for solving the task of interest. The transformation is performed with a matrix multiplication, i.e., $\boldsymbol{h} = \boldsymbol{W}\boldsymbol{x}$, to receive the intermediate representation $\boldsymbol{h}$. An essential part for enabling neural networks to learn non-linear functions is to add an activation function right after the matrix multiplication of each layer. Otherwise, the neural network would only be capable of learning linear functions since the matrix multiplication is a linear mapping. A common activation function is the $a(\boldsymbol{x}) = \max(0, \boldsymbol{x})$, or the so called Rectified Linear Unit (ReLU) activation, which outputs $\boldsymbol{x}$ when $\boldsymbol{x} > 0$ or otherwise zero. By stacking two layers together in a neural network with a ReLU activation, we then obtain the representation $\boldsymbol{h} = \boldsymbol{W}_2 \max(0, \boldsymbol{W}_1\boldsymbol{x})$. Note here that this representation could be predicted class scores, such that $\boldsymbol{h} = \hat{\boldsymbol{y}}$, if we would use two-layer MLP for a classication task.

Convolutional Neural Networks. For data where the spatial order of each feature can be salient for prediction tasks such as image classification, we need a network that can capture relationships between features. CNNs are special kinds of neural networks that can process data with grid-like structures. Convolutional layers constitutes of a set of filters with adaptable weight parameters. To produce the output, we slide each filter across the input across the width and height of the input and compute dot products between the filter weights and the input at any position. Each filter will then produce a 2-dimensional feature map that gives the responses of that filter at every spatial position. The 2-dimensional feature maps from all filters are then stacked depth-wise to obtain the output volume. The obtained feature maps are often downsampled along their spatial dimensions using a pooling operation after the activation function. The parameter sharing in convolutional layers where each weight in a filter is applied to every position of the input comes from the idea that if some visual features are important in one part of the image, it should intuitively be useful at some other location as well. Furthermore, this design choice also makes the model require fewer parameters and a lower number of operations to compute the outputs.

Recurrent Neural Networks. RNNs are a family of neural network models specialized for processing sequences of data. Similar to CNNs, these models use parameter sharing by applying the same weights across several time steps. The parameter sharing is important in RNNs as it enables the model to handle different sequence lengths as well as being capable of recognizing relevant information that can appear at different locations in the sequence. Many RNNs follow the same procedure through the equation $\boldsymbol{h}^{(t)} = f_{\boldsymbol{\theta}}(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)})$, where the RNN produces the current state $\boldsymbol{h}^{(t)}$ by incorporating the input data $\boldsymbol{x}^{(t)}$ at time t into the previous hidden state $\boldsymbol{h}^{(t-1)}$. Hence, the hidden state $\boldsymbol{h}^{(t)}$ will now contain information about the whole past sequence up to time t. In most applications, there will be an extra output layer that reads the information from state $\boldsymbol{h}^{(t)}$ to make predictions. An example application for RNNs is predicting the next word in a sentence given previous words, where the RNN should store the necessary information about previous words to predict the rest of the sentence. A common choice of RNN model is the Long Short-Term Memory [59] (LSTM) which mitigates problems with vanishing gradients during the training phase.

For training deep networks, *loss* functions are used for measuring how well the network performs to solve the task of interest. In classification tasks, the cross-entropy loss is commonly used where the predicted class scores $\hat{\boldsymbol{y}}$ are compared against the true target class $\boldsymbol{y}$,

$$\mathcal{L}_{CE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{k=1}^{K} \boldsymbol{y}[k] \log \hat{\boldsymbol{y}}[k], \tag{2.1}$$

where the true target vector $\boldsymbol{y}$ uses a one-hot representation where the true class i is denoted in the vector by setting the i-th element in $\boldsymbol{y}$ to one, as in $\boldsymbol{y}[i] = 1$, and zero elsewhere.

Probably the most common optimization algorithm for deep learning is stochastic gradient descent (SGD). The model parameters are updated by first computing the gradient of the loss function with respect to the weights of the network, as in $\nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y})$, for a single input-output pair with backpropagation [60]. We can then update the parameters with the equation

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y}), \tag{2.2}$$

where η is the learning rate which is an important parameter for SGD determining how much the weights should be updated with the computed gradient.

Autoencoders

A common architecture type for deep learning in unsupervised learning are autoencoders for learning hidden representations of unlabeled data. Autoencoders are commonly used for dimensionality reduction of high-dimensional data, where the lower-dimensional representation can be used for classification tasks, or to visualize hidden structures in the data that are hard to reveal from the original input data. The objective of the model is to reconstruct the original input data. The network architecture is built using two networks called *encoder* and *decoder* with a bottleneck layer between the networks for extracting the hidden representation $\boldsymbol{h}$. The encoder and decoder architectures can be of any neural network type, such as MLPs, CNNs, or RNNs, that fits the given dataset. The encoder is used for obtaining the hidden representation of the input data, while the decoder tries to reconstruct the original input from the obtained representation. Therefore, the idea is that the learned representation should contain the relevant information for reconstructing the data.

Mathematically, we denote the decoder as $f_{\boldsymbol{\theta}}$ and the encoder as $g_{\boldsymbol{\phi}}$. The encoder extracts the hidden representation $\boldsymbol{h} = g_{\boldsymbol{\phi}}(\boldsymbol{x})$ from the input $\boldsymbol{x}$, then the decoder produces a reconstruction $\hat{\boldsymbol{x}} = f_{\boldsymbol{\theta}}(\boldsymbol{h})$ from $\boldsymbol{h}$. We train the encoder and decoder simultaneously by minimizing a reconstruction loss $\mathcal{L}(f_{\boldsymbol{\theta}}(g_{\boldsymbol{\phi}}(\boldsymbol{x})), \boldsymbol{x})$, for instance mean-squared error loss, using SGD similarly as for the feedforward networks described above. There exist various kinds of methods for improving the quality of the learned representations in autoencoders. For example, we can adjust target task by adding noise to the inputs and let the decoder reconstruct the original input from noise variants [61], or we can induce different constraints in the bottleneck layer to, for example, obtain a sparse lower-dimensional representation of the data. Next, we will introduce the variational autoencoder which originates from latent variable models.

Variational Autoencoders

The variational autoencoder [62] (VAE) is a variant of autoencoders where learning is viewed from the perspective of probabilistic modeling. These models come from the family of deep generative models, where the goal is to approximate some underlying data distribution p_{data} with a parametric distribution $p_{\boldsymbol{\theta}}$ learned from a dataset $\mathcal{D} \sim p_{data}$. A common approach for estimating $p_{\boldsymbol{\theta}}$ is to use a latent variable model that infers hidden structures in the data to facilitate learning the distribution. VAEs is a deep latent variable model that uses neural networks for learning $p_{\boldsymbol{\theta}}$ making the training scalable to large high-dimensional datasets.

The main idea with introducing latent variables is that they should encode some semantically meaningful information about the observed data. Latent variable models are usually expressed by the joint distribution

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}), \tag{2.3}$$

where $\boldsymbol{z}$ denotes the latent variables and $\boldsymbol{x}$ the observed variables that represents the observed data points. The distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is the likelihood of the data and $p(\boldsymbol{z})$ is the prior distribution for the latents. This model describes the generative process of the data $\boldsymbol{x}$ by following the steps 1) sample the latent vector $\boldsymbol{z} \sim p(\boldsymbol{z})$ from the prior, and 2) generate data point $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{z})$ from the sampled latent $\boldsymbol{z}$. We are now interested in learning the model $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ that best fits a given dataset $\mathcal{D}$, as well as inferring the posterior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ over the latent variables $\boldsymbol{z}$ given the data $\boldsymbol{x}$.

The overall goal with latent variable models is to maximize the marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ given a dataset $\mathcal{D} \sim p_{data}$. However, computing $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ by with marginalizing out the $\boldsymbol{z}$ from the model $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z}$ is in general intractable due to the many settings of $\boldsymbol{z}$ we would need to evaluate. Consequently, the posterior distribution also becomes intractable since $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})/p_{\boldsymbol{\theta}}(\boldsymbol{x})$ from Bayes' rule. Variational inference [63, 64] is a technique for enabling learning of latent variable models. The idea of variational inference is to provide means for calculating the marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ by selecting a parameterized distribution $q_{\boldsymbol{\phi}}$ for approximating the true posterior distribution. In VAEs, the approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is represented as a neural network with parameters $\boldsymbol{\phi}$ that outputs the latents $\boldsymbol{z}$ given data points $\boldsymbol{x}$. With this approach, we can now form a lower bound on the marginal log-likelihood given by

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq \mathbb{E}_{z \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - D_{\mathrm{KL}}[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})]. \tag{2.4}$$

The right-hand side is called the evidence lower bound (ELBO) and comprises of two quantities that we can evaluate to train the model. The expectation over the log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ can be estimated with Monte Carlo sampling. The KL divergence between $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{z})$ can be computed analytically depending on how we choose these distributions. The standard choice for the prior is to use a zero-mean unit-variance Gaussian distribution $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I})$ where $\mathbf{I}$ is the identity matrix. The approximate posterior is also selected to be a Gaussian distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \mathrm{diag}(\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x})))$, where the encoder network parameterized by $\boldsymbol{\phi}$ outputs the the means $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x})$ and standard deviations $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x})$ for the latent dimensions. The latent vector is sampled using the "reparametrization trick" [62, 65] by computing $\boldsymbol{z} = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}) + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{x})$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\odot$ denotes element-wise multiplication, which enables backpropagating gradients through the sampling operation. The likelihood $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is the decoder network that tries to reconstruct the original input to the encoder. The likelihood distribution depends on the type of data $\boldsymbol{x}$ we wish to generate. If $\boldsymbol{x}$ is a continuous variable, then we can let the decoder output Gaussian parameters for the likelihood similar as for the encoder.

VAEs have been used in various applications for modeling images, text, and audio data, as well as when combining data from different modalities. Next, we will briefly introduce how autoencoders can be used when learning representations from multiple data types from different modalities.

Multimodal Learning using Autoencoders

Learning representations from different types of data is a highly active research field in deep learning [66]. Combining visual data with other modalities such as natural language and audio signals for learning more rich representations of data has been studied frequently the last decade [67–70] [Add REFs]. A common framework in such applications is to use autoencoders for incorporating information from the different data types into a single joint representation. Learning from multiple sources then opens up for capturing correspondences between the data types and obtaining better representations that can be used for downstream tasks such as classification.

Let $\boldsymbol{x}$ and $\boldsymbol{y}$ originate from two different data sources but share some high-level information. For example, $\boldsymbol{x}$ could be an image of a living room and $\boldsymbol{y}$ is text describing the appearance of the room, where objects are located etc. Constructing a joint representation is then done by projecting both $\boldsymbol{x}$ and $\boldsymbol{y}$ with separate encoder networks into the same latent space. The joint multimodal representation is then passed through two separate decoder networks used for predicting the original input data individually. The advantage of multimodal autoencoders is that they can be trained end-to-end for both learning representations as well as making predictions of the used modalities. However, a major challenge is how to handle scenarios where modalities might be missing. One option is to only encode the data modality that we know will be available at both training and test phases and then establish a joint representation by decoding two both modalities [68].

Multimodal autoencoders have frequently been extended to deep generative models, mainly VAEs [53, 71–74]. These models are capable of generating new data through sampling from the latent space in addition to learning joint representations. Furthermore, they can handle missing modalities for the encoders which enables cross-modal data generation between the modalities. In Paper B [Add REF], we employ Variational Canonical Correlation Analysis (VCCA) for learning joint representations of natural images and web-scraped information of grocery items to facilitate learning image classifiers.

Deep Reinforcement Learning

MK: TO-DO: Related works for Paper D. After writing Chapter 4 on CL

Chapter 3

Fine-grained Recognition

This chapter presents an approach for enhancing fine-grained classification performance of grocery items by using web-scraped information. We focus on classification of grocery items due to applicability in assistive vision and its potential to enhance the independence of visually impaired (VI) people [ADD groceries/shopping/object recognition for VI REFs]. Initially, we were interested in learning classifiers with natural images taken in the grocery stores combined with web-scraped information about the grocery items, such as iconic images and text descriptions from supermarket websites. Using iconic images have been used in grocery image classification earlier [ADD grocery paper REFs], however, utilizing text descriptions was as far we know absent for this application even if it has been successfully applied in other image classification problems [75–77]. Thus, we collected our own dataset of grocery items images using a mobile phone camera as well as web-scraped images and text descriptions to study whether this multi-view approaches would benefit training the classifiers (Section 2). We then select a multi-view learning framework based on the Variational Autoencoder (VAE) for investigating how the different data views affect the fine-grained classification performance (Section 3).

3.1 Related Work

In this section, we will briefly discuss the related work on fine-grained image recognition [41], particularly when learning from external information, and multi-view learning.

Fine-grained Image Recognition

The goal with fine-grained image recognition (FGIR) is to distinguish between images with multiple visually similar sub-categories that belong to a super-category. For example, various attempts have been made to discriminate between sub-categories of different animals [78], cars [79], fruits [80], retail products [81], etc. The challenge is to recognize differences that are sufficient for discriminating between objects that are generally similar but differ in fine-grained visual details. In recent years, the successes with deep learning in computer vision have encouraged researchers to explore various approaches for FGIR that can broadly be divided into three directions for recognition by utilizing (i) localization-classification subnetworks, (ii) end-to-end feature encoding, and (iii) external information. In (i), the goal is to find object parts that are shared across the sub-categories for discovering details that make the part representations different. This can be achieved by utilizing feature maps from the activations of convolutional layers as local descriptors [82–84], employing detection and segmentation techniques for localizing object parts[REFs], or leveraging attention mechanisms when common object parts are difficult to represent or even define [REFs]. With (ii), the goal has been to learn features that are better at capturing subtle and local differences by, for instance, performing high-order features interactions [85] as well as designing novel loss functions [Add REFs]. In the third approach (iii), the goal is to leverage external information, for example, web data and multimodal data, in FGIR as additional supervision to the images. We will put more focus on the approach on FGIR with external information next, as we use this approach in Paper A and B.

Recognition with External Information

Learning fine-grained details about objects often requires large amounts of labeled data. To ease the need for large amounts of accurately labeled images, there have been several attempts to let either web-scraped or multimodal data influence learning the fine-grained features of the sub-categories to boost the FGIR performance. Web-scraped images may be noisy in the sense that retrieved images may have high-variations of the objects. For example, the objects of interest can look different in appearance, and there could also be other irrelevant objects in the images that potentially occlude the category to recognize. Hence, incorporating web-scraped data into the training set may establish a domain gap between the easily acquired web data and the original training set which we need to overcome by reducing the domain gap or reducing the negative effects of the noisy web data that can disturb the learning. Another direction than using web-scraped data is to utilize multimodal data, for example, images, text and knowledge bases, for boosting the classification performance. In FGIR, the goal is to establish a joint representation between the images and additional data sources, where the additional data should act like extra guidance for learning useful representations that capture the fine-grained details of objects. Text descriptions have been a popular data type to combine with images, which can be both easy and cheap to collect as they can be accurately generated by non-experts. High-level knowledge graphs of objects have also been used and can contain rich knowledge useful for fine-grained recognition. In addition to FGIR, both web-scraped and multimodal external information has been used for zero-shot learning to transfer knowledge from annotated categories to new fine-grained categories. In Paper A, we collect web-scraped images and text descriptions of grocery items to accompany real-world images of groceries for FGIR. Then, in Paper B, we perform a study using multi-view learning to investigate how the external information can enhance the classification performance. Next, we will cover the related work for the multi-view learning approach that we used.

Multi-view Learning

MK: TO-DO: Learning from several data sources and modalities. Briefly on multi-view learning approchaes and VCCA.

3.2 Dataset Collection

In this section, we describe our procedure for collecting the image dataset of grocery items. As the target use case is grocery shopping with an assistive vision device, we visited several supermarkets and collected natural images of the groceries with a mobile phone camera to imitate such scenarios. Hence, the collected images will capture situations that can be challenging for the assistive device, such as, various lighting conditions, multiple instances and classes present, hand occlusions, and misplaced items. All images were taken with a single targeted item in mind, such that each image is paired with a single label. For items which belong to a clear super-class, for example, various kinds of apples and milk packages, we also provided the general class of the items to establish a hierarchical labeling structure of the data. Collecting natural images of the grocery items is unfortunately a time-consuming process. Furthermore, as the surroundings in every grocery store varies, it may be difficult to build accurate classifiers that can recognize fine-grained details solely from natural images. Hence, we need some cheaper procedure that can complement the collection of real-world images for boosting the classification performance of the groceries.

We have complemented the image dataset with external information from the web of each grocery item that can be used for training classifiers. In the past years, most supermarket chains have the option for consumers to purchase groceries online from their websites. The website usually provides each grocery item with an iconic image of the item on a white background, a text description that describes the flavor and ingredients of the item, as well as nutrition values if applicable. We downloaded these information types of all grocery item classes by web-scraping the online shopping website of a supermarket chain. We show four examples of grocery items and their web-scraped information in Table 3.1. Since these data types are on a class-based level, we can use the web-scraped information as weak supervision to guide the classifier to learn fine-grained details that helps discriminating between visually similar items.

Table 3.1: Examples of grocery item classes in the Grocery Store dataset. We display four different items (coarse-grained class in parenthesis), followed by two natural images taken with a mobile phone inside grocery stores. Next comes the web-scraped information of the items consisting of an iconic image and a text description. We have highlighted ingredients and flavors in the text description that are characteristic for the specific item.

Class Labels	**Natural Images**	**Iconic Images**	**Text Descriptions**
Granny Smith (Apple)			*“...**green** apple with **white, firm** pulp and a **clear acidity** in the flavor.”*
Royal Gala (Apple)			*“...**crispy** and **very juicy** apple, with **yellow-white** pulp. The peel is **thin** with a **red yellow** speckled color.”*
Tropicana Mandarin (Juice)			*“. . . is a **ready to drink** juice **without pulp** pressed on **orange**, **mandarin** and **grapes**. Not from concentrate. Mildly **pasteurized**.”*
Yoggi Vanilla (Yoghurt)			*“...**creamy vanilla** yoghurt original... added **sugar** than regular flavored yoghurt. Great for both **breakfast and snacks**.”*

3.3 Fine-grained Classification of Grocery Items

This section describes the approaches we used for classification of grocery items from the available data types in the collected dataset. We begin by introducing the problem setting, followed by describing the methods for learning representations from the available data views.

Problem Setting

The application we are focusing on is grocery shopping with an assistive vision device. The device could for instance be a mobile phone app where the groceries are recognized by an in-built image classifier from natural images taken with the camera. Training such image classifier to be robust in grocery store environments would typically require an immense amount of labeled training examples of all available groceries. To reduce the need for labeled training data, we aim to combine the collected natural images with web-scraped information about the groceries when training the classifier. The goal is that incorporating the web-scraped information should help the classifier to learn fine-grained details about the items to enhance the classification performance and robustness.

The available data views that are available for training image classifiers is denoted as follows:

- $\boldsymbol{x}$: Natural images of the grocery items in the grocery stores.
- $\boldsymbol{y}$: Class labels of grocery items from corresponding natural images.
- $\boldsymbol{i}$: Iconic images of the grocery items scraped from a supermarket website.
- $\boldsymbol{w}$: Text description of the grocery items scraped from the same supermarket website as $\boldsymbol{i}$.

The simplest approach is to take a standard supervised approach and train a CNN from the natural image and class label pairs. An alternative is leverage from CNNs pre-trained on a large dataset, such as Imagenet [86], and fine-tune the final classification layer to the grocery item recognition task [87]. We use ideas from multi-view learning [88] and VAEs [62] for learning joint representations from the available data views that can be used for training the image classifiers, which we present in the next section.

Multi-view Representation Learning of Grocery Items

This section describes the approach we took for learning representations of grocery items that are shared across the available data types. We employ a deep latent variable model called Variational Canonical Correlation Analysis [53] (VCCA) for learning the shared representation. The main assumption in VCCA is that each data view have been generated from the same latent space. The goal then is to learn this latent space that captures the correspondences between all views into representations shared across the views for the grocery items. This representation can then be utilized for enhance the learning more accurate classifiers as well as for performing tasks such as synthesis and prediction of novel images. Next, we describe how to enable learning the shared latent space.

Capturing variations from each view in the learned representation is performed by predicting the original views from the latent space. To obtain the latent representation, we extract the representation by encoding the natural images with neural network. The extracted representation is then used for predicting each view individually by inputting the representation through separate neural networks. Note that we only use the natural images for extracting the latent representation here since it is the only view that is available at test time when we want to use the learned classifier in the grocery store. We have two options for exploiting the new representation to train classifiers. The first option is to train the classifier with the latent representations after we have learned the latent space as described above. The second option is to train the classifier and learning the latent space simultaneously by adding an additional classifier network predicting the class label with the latent representation as input.

MK: TO-DO: I should add a figure of the architecture for extra clarity on the method.

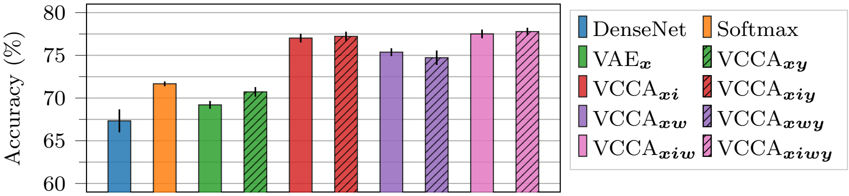


Figure 3.1: Fine-grained accuracies on the Grocery Store dataset for all classification methods. We show the means and standard deviations averaged over 5 seeds. Adding the iconic image $\boldsymbol{i}$ and text description $\boldsymbol{w}$ for learning joint representations with VCCA improves the classification performance over approaches that only utilize the natural images and class labels.

3.4 Experiments

In this section, we summarize the main results from the experimental study in Paper B. We performed an ablation study with VCCA over the combinations of available data views to investigate how each data view contributes to the classification performance. First, we present results on fine-grained classification performance between the compared methods. Then, we provide insights in how the web-scraped icnoc images and text descriptions contribute to the boosting the classification performance by visualizing their the joint latent spaces. Finally, we demonstrate how the iconic image decoder in VCCA can be used for explaining the misclassifications by generating iconic images from natural images at test time.

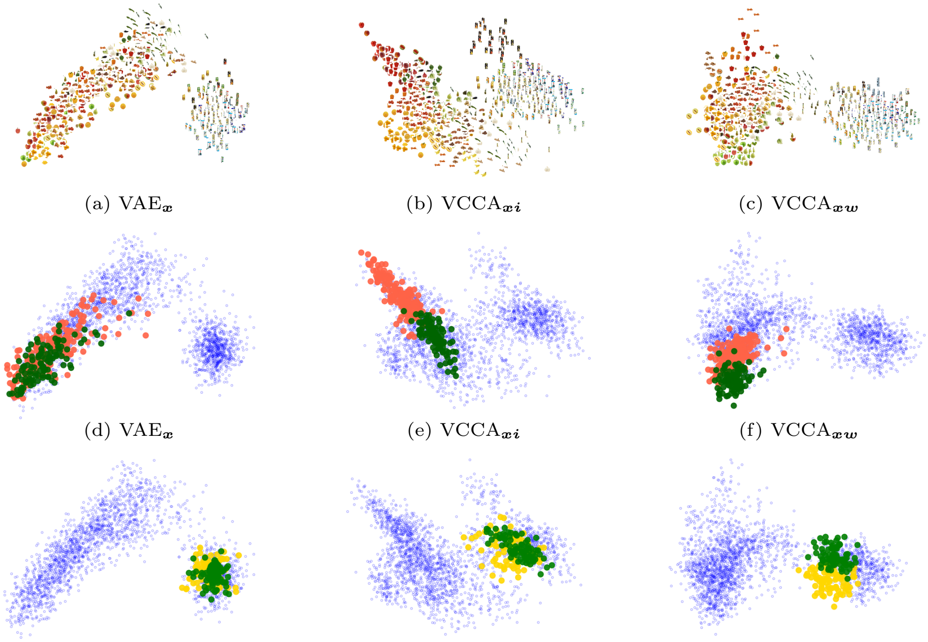
We compare the VCCA models against two CNN baselines that uses the DenseNet [89] architecture. The first baseline is a DenseNet trained from scratch on the dataset, and the second baseline is a Softmax classifier trained from image features extracted from a DenseNet pre-trained on ImageNet. We denote which data views that are used by VCCA using subscripts. For instance, VCCA$_{\boldsymbol{xiw}}$ means that the natural images $\boldsymbol{x}$, iconic images $\boldsymbol{i}$, and text descriptions $\boldsymbol{w}$ are utilized for learning the joint latent representation. These VCCA models uses the two-stage classifier setup with steps 1) train VCCA on the data views, and 2) train 1-layer MLP classifier using the extracted latent representations from VCCA and the corresponding class labels. We also compare against VAE$_{\boldsymbol{x}}$ only using the natural images $\boldsymbol{x}$ trained in this setting. The VCCA models with class label decoders are denoted by using $\boldsymbol{y}$ in the subscript, such as VCCA$_{\boldsymbol{xiwy}}$.

Fine-grained Classification Results. Adding the web-scraped views in VCCA improves the classification performance over approaches that only utilize the natural images and class labels. Figure 3.1 shows a bar plot over the fine-grained accuracies achieved by all classification methods. Among the methods only using natural images and class labels, we see that the Softmax baseline performs best, which could be due to some information loss when compressing the images into as low-dimensional latent representations with VAE$_{\boldsymbol{x}}$ and VCCA$_{\boldsymbol{xy}}$. The performance of VCCA significantly improves over Softmax when the iconic image $\boldsymbol{i}$

and text description $\boldsymbol{w}$ are used, which shows that both data views are useful for enhancing the fine-grained classification performance. Comparing VCCA$_{\boldsymbol{xi}}$ and VCCA$_{\boldsymbol{xw}}$, we see that utilizing the iconic image has an advantage over using the text description for improving the performance. This could be due to the fact that the text descriptions are providing information on ingredients and flavors rather than visual appearance. Combining both $\boldsymbol{i}$ and $\boldsymbol{w}$ achieves on par performance as only utilizing the iconic image, which could potentially be improved by filtering the text descriptions for obtaining words relevant for describing the fine-grained details of the items. Finally, we observe that both classification options for VCCA performs similar, which could potentially be since $\boldsymbol{i}$ and $\boldsymbol{w}$ acts as labels since there is only a single instance of these views for all classes.

Visualization of Latent Space. The web-scraped iconic images and text descriptions structures the grocery items based on view-specific similarities in the latent space that are beneficial for fine-grained classification. In Figure 3.2, we illustrate how adding either the iconic image $\boldsymbol{i}$ or the text description $\boldsymbol{w}$ changes the structure of the latent space, where we have used PCA to project the latent representations into a 2-dimensional space. In Figure 3.2(a-c), we have plotted the corresponding iconic image for all latent representations of models VAE$_{\boldsymbol{x}}$, VCCA$_{\boldsymbol{xi}}$, VCCA$_{\boldsymbol{xw}}$ for visualization purposes. The latent space for VAE$_{\boldsymbol{x}}$ separates raw and packaged grocery items into two separate clusters. When adding the iconic image in VCCA$_{\boldsymbol{xi}}$, we observe that the latent space becomes structured according to the color of the items. For VCCA$_{\boldsymbol{xw}}$, the latent space becomes structured according to the ingredients of the items, where we see in the raw food cluster that bell peppers are placed in the upper region while apples are in the lower region.

Next, we focus on a certain set of classes to inspect to gain more insights in how the additional data views affect the latent space. First, we focus on the green and red apples classes to inspect how the views handle visually different items in color. In Figure 3.2(d-f), we plot the latent representations for the three models but highlight the green and red apple classes by plotting them as green and red dots respectively. All other classes are plotted as smaller blue dots. We see that both VCCA$_{\boldsymbol{xi}}$ and VCCA$_{\boldsymbol{xw}}$ manages to separate the apple classes better than VAE$_{\boldsymbol{x}}$, where adding the iconic images yields the most clear separation. Next, we want to study the benefits of the text descriptions. We focus on some yoghurt and juice package classes that are visually similar but have very different ingredients and flavors. In Figure 3.2(g-i), we plot the latent representations for the three models again where the yoghurt and juice classes are plotted in green and yellow colored dots respectively. Here, we see that VCCA$_{\boldsymbol{xw}}$ manages to separate these items better than VCCA$_{\boldsymbol{xw}}$ due to the differences in the text descriptions between the selected package classes.



(a) VAE$_{\boldsymbol{x}}$ (b) VCCA$_{\boldsymbol{xi}}$ (c) VCCA$_{\boldsymbol{xw}}$

(d) VAE$_{\boldsymbol{x}}$ (e) VCCA$_{\boldsymbol{xi}}$ (f) VCCA$_{\boldsymbol{xw}}$

(g) VAE$_{\boldsymbol{x}}$ (h) VCCA$_{\boldsymbol{xi}}$ (i) VCCA$_{\boldsymbol{xw}}$

Figure 3.2: Visualizations of the latent representations from VAE$_{\boldsymbol{x}}$, VCCA$_{\boldsymbol{xi}}$, and VCCA$_{\boldsymbol{xw}}$ projected in 2-dimensional space with PCA. In (a-c), we show the latent representations plotted using the iconic images of the corresponding object class. In (d-f), we illustrate how the iconic images structures the items based on visual similarities by focusing on the green and red apple classes in the dataset plotted in their corresponding colors. Similarly, in (g-i), we show how the text descriptions structure items based on ingredients and flavor by focusing on visually similar yoghurt (green) and juice (yellow) packages. The blue dots correspond to all other grocery item classes.

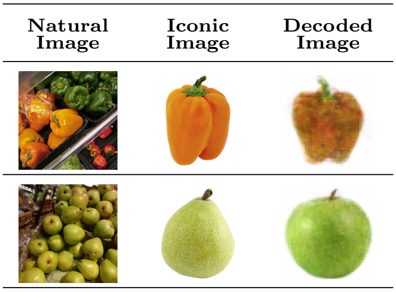


Figure 3.3: Examples of decoded iconic images from VCCA$_{\boldsymbol{xiwy}}$ with their corresponding natural image and true iconic image.

Iconic Image Generation. The iconic image decoder can provide explanations for the predicted classes. Figure 3.3 shows two examples of decoded iconic images from two natural images where the class labels are *Orange Bell Pepper* and *Anjou Pear*. On the first row, we see that VCCA$_{\boldsymbol{xiwy}}$ has recognized the green bell peppers in the natural image and generated a mixed orange and green bell pepper in the iconic image. On the second row, we see that the decoded iconic image is a *Granny Smith* apple instead of a pear which was the true class. The classifier consequently predicts the natural image to be a *Granny Smith* apple. Hence, the iconic image decoder can be used as a tool for providing an intuition of why the classifier made an error.

3.5 Discussion

In the experiments, we showed that utilizing the web-scraped information with VCCA can enhance the performance of grocery item classifiers. This shows that the cheaper web-scraped views can serve as a good alternative for improving classification performance over collecting more natural images in the grocery stores. Furthermore, we illustrated how the iconic images and text descriptions affects the structure of the latent space based on view-specific information. More specifically, the iconic images structures the latent space after visual similarities such as colors and shapes, while the text descriptions pushes items with similar ingredients and flavors closer to each other in the latent space. Finally, we demonstrated how the iconic images can be used for providing potential explanations for misclassifications, which could help us detect hard classes and give us indications of how robust the latent representations are to classifying images with different classes.

We observed that utilizing the iconic images in VCCA affects the classification performance significantly better than the text descriptions. Potentially, this is due to the text description view to be more noisy than the iconic images as there are few words that are relevant to the recognition task. However, the text descriptions could be utilized more efficiently, for example by pre-processing the text to keep words describing fine-grained details about the items as well as removing stop words ('the', 'it', 'and', etc.) and other irrelevant words. A second option would be to use attention mechanisms [90,91] that helps the model to learn which words to emphasize on when learning the joint latent representations. We could also encode the text into a single-vector embedding with various methods [92–94] and replace the RNN with an MLP predicting in the embedding space which the text embedding the natural image is closes to.

Regarding the dataset collection, we have suggested extensions on how to provide more useful information about the items to improve the recognition task. Firstly, it would be valuable to download instances of iconic images and text descriptions from more supermarket websites to potentially allow the VCCA model to capture more view-specific variations into the learned representations. Secondly, the model should be extended to handle video data rather than still images. This would require record videos with the mobile phone camera in the grocery stores to properly evaluate the classifiers. Nevertheless, the extension to video would be beneficial for user experience of the recognition app since the classifier would receive more chances to classify the items correctly by utilizing multiple frames.

Finally, the Grocery store dataset could be extended to zero/few-shot learning [95,96] and continual learning [42,43] settings. Such applications are important for assistive vision devices to build data-efficient and adaptable systems that improves their usability in real-world scenarios.

Chapter 4

Continual Learning

This chapter introduces the idea of replay scheduling for mitigating catastrophic forgetting in continual learning (CL). The problem setting of CL is on learning tasks of recognizing a new set of classes with a dataset given at the current time step. In the standard setting, one main assumption is that the data from past tasks can never be fully revisited by the model. However, in the real-world, many organizations record data from incoming streams for storage rather than deleting it [97, 98] [Add at least 1 more REF]. In contrast to the assumption on data storage in standard CL, we suggest a new setting where we assume that all seen data is accessible at any time for the model to revisit. The challenge then becomes how to select which tasks that needs to be remembered via replay as the data is still incoming from a stream. We propose to learn the time when replaying a certain task is necessary when the model is updating its knowledge with new incoming tasks. In Paper C, we propose the new CL setting where historical data is accessible and introduce the idea of replay scheduling and how it can be used in CL. In Paper D, we propose a framework based on reinforcement learning [99] (RL) for learning replay scheduling policies that can be applied in new CL scenarios.

MK: TO-DO: Motivate replay scheduling from mobile phone perspective, perhaps from perspective that phones can store lots of data but how to select which classes to replay. Maybe it should also be from the computational perspective, that we want such scheduling policy to work in many scenarios without additional compute cost for the policy learning.

4.1 Related Work

In this section, we give an overview of previous works related to Paper C and D. We begin by describing different approaches in CL, especially replay-based approaches, and then discuss meta-learning policies and generalization in RL.

Continual Learning. There exist many different approaches in CL for mitigating catastrophic forgetting in neural networks. In general, these approaches can be divided into three main cores, namely, *regularization-based*, *architecture-based*, and *replay-based* methods. Regularization-based methods are mainly focused on applying regularization techniques on parameters important for recognizing old tasks and fit the remaining parameters to new tasks [100–102]. Knowledge distillation methods [103] also belong to these approaches where classification logits are used for regularizing the output units for previous tasks in the network [104,105]. More recently, there are some works that uses projection-based approaches for constraining the parameter updates to subspaces which avoid interference with previous tasks [106,107]. Architecture-based approaches focuses on adding task-specific network modules for every seen task [108–111], or isolating parameters for predicting specific task in fixed-size networks [112–114]. Replay-based methods re-trains on samples of old tasks when learning new tasks. The old samples are either stored in an external memory [46,115,116], or synthesized with a generative model [72,117–119]. Both regularization- and architecture-based methods can be combined with replay for improving the models capability of remembering tasks [Add REFs]. Our replay scheduling idea is originated from replay-based methods which we will cover more in detail next.

Replay-based Continual Learning. In this thesis, we focus on replay-based methods with external memories for storing historical data. The most common selection strategy for filling the memory is random sampling from the used datasets. There exist several works focusing on selecting high quality samples for storing in the memory [Add REFs]. However, in image classification problems, random sampling has been shown to often perform on par with more elaborate selection strategies [115,120]. In contrast to using various memory selection methods, there has been proposals of retrieval policies over which samples to select for replay from the memory, for instance, selecting the samples that will mostly interfere with the parameter update with batches of new data [121]. Our replay scheduling approach differs from this method as we focus on selecting which tasks to select for replay rather than the individual samples to retrieve from the memory. More recent works have focused on evolving the memory samples through data augmentation to avoid overfitting to the memory [Add REFs], and also by using contrastive learning to improve discriminating between tasks. Another direction has been to increasing the storage capacity to store more samples by compressing raw data into features that are more memory-cheap [Add REFs]. The above mentioned methods assume that the memory is small and allocates equal storage amount for all tasks. Our new problem setting for memory-based CL is different from this assumption as we argue that data storage is cheap in many real-world applications. Hence, we compose a replay memory with data from historical tasks before learning new tasks because the amount of compute is limited. However, replay scheduling can be combined with of the mentioned methods as it only differs with the standard memory-based CL setting in that the replay memory has to be selected at every new task.

Meta-Policy Learning?

Similarities and Differences between Continual Learning and other fields. **MK: A short and cozy table fo this, similar to survey by Delange etal 2021**

4.2 Replay Scheduling in Continual Learning

In this section, we introduce a slightly new CL setting considering the real-world needs where all historical data can be available since data storage is cheap. However, the amount of compute is limited when the model is updated on new data due to operational costs. Hence, it is impossible for the model to leverage from all the available historical data to mitigate catastrophic forgetting. The goal then becomes to learn how we can select subsets of historical data for replay to efficiently reduce forgetting of the old tasks. We will refer to these subsets of historical data as the *replay memory* throughout this chapter. The size of the replay memory affects the processing time when learning new tasks as well as the allowed time for the training phase. When composing the replay memory, we focus on determining the number of samples to draw from the seen tasks in the historical data rather than selecting single stored instances. Next, we introduce the problem setting in more detail as well as the notation of the new CL setting.

Problem Setting

We introduce the notation of our problem setting which resembles the traditional CL setting for image classification. We let a neural network $f_{\boldsymbol{\theta}}$, parameterized by $\boldsymbol{\theta}$, learn T tasks from the datasets $\mathcal{D}_1, \dots, \mathcal{D}_T$ arriving sequentially one at a time. The t-th dataset $\mathcal{D}_t = \{(\boldsymbol{x}_t^{(i)}, \boldsymbol{y}_t^{(i)})\}_{i=1}^{N_t}$ consists of N_t samples where $\boldsymbol{x}_t^{(i)}$ and $\boldsymbol{y}_t^{(i)}$ are the i-th data point and class label respectively. The training objective at task t is given by

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N_t} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}_t^{(i)}), \boldsymbol{y}_t^{(i)}), \tag{4.1}$$

where $\mathcal{L}(\cdot)$ is the loss function, which in our case is the cross-entropy loss. When learning task t, the network $f_{\boldsymbol{\theta}}$ is at risk of catastrophically forgetting the previous $t-1$ tasks. The forgetting effect shows as the decrease in task accuracy between time steps, for example, $A_{t,i} < A_{t,i-1}$ where $A_{t,i}$ is the accuracy for task t at time step i. Replay-based methods mitigate catastrophic forgetting by storing a few number of examples from historical tasks in an external memory. The network

$f_{\boldsymbol{\theta}}$ is then allowed to fetch old examples from the memory and mix these with the current task dataset to remind itself about the previous tasks during training.

In the new problem setting, we assume that historical data from old tasks are accessible at any time step. However, we can only fill a small replay memory $\mathcal{M}$ with M historical samples for replaying old tasks due to processing time constraints prohibiting re-using all historical data at the same time. The challenge then becomes how to know which tasks to include in the replay memory that efficiently retain the previous knowledge when learning new tasks. We decide to fill the replay memory with M historical samples using sequence of task proportions $(p_1, \dots, p_{t-1})$ where $\sum_{i=1}^{t-1} p_i = 1$ and $p_i \geq 0$. The number of samples from task i to place in $\mathcal{M}$ is given by $p_i \cdot M$. In the next section, we introduce a method for selecting the task proportions of which old tasks to replay.

Comparison to Traditional CL. The new setting has several similarities to the traditional CL setting. Both settings share the fundamental setting that the data arrive in streams and re-training on all historical data is prohibited. Also, the goal that the model should perform well both historical tasks and tasks associated with new data remains the same. In replay-based CL, we also share the same constraints that the memory size is limited. However, we argue that this limitation is mainly associated with compute rather than of storage. Our assumption aligns with the real-world where data storage is cheap and easy to maintain, but retraining large machine learning models is computationally expensive. The only difference is that we allow filling the limited replay memory from historical data or some other external memory. Here, we argue that historical data is stored rather than deleted in many real-world settings [97]. Thus, we should keep the limited memory assumption for training but allow access to historical data to fill the replay memory to make CL align with real-world needs. **MK: this paragraph can be written in a more humble way, more like "this is something worth to investigate"**

Replay Scheduling for Mitigating Catastrophic Forgetting

In this section, we describe our replay scheduling method for selecting the replay memory at different time steps. A replay schedule is defined as a sequence $S = (\boldsymbol{p}_1, \dots, \boldsymbol{p}_{T-1})$, where $\boldsymbol{p}_i = (p_1, \dots, p_{T-1})$ for $1 \leq i \leq T-1$ is the sequence of task proportions for determining how many samples per task to fill the replay memory with at time step i. To make the selection of task proportions tractable, we construct an action space with a discrete number of choices for the task proportions from historical tasks.

We show the procudre for creating the discrete action space in Algorithm 1. At task i, we have $i-1$ historical tasks that we can choose from. We then generate all possible bin vectors $\boldsymbol{b}_i = [b_1, \dots, b_i] \in \mathcal{B}_i$ of size i where each element are a task index $1, ..., i$. We sort all bin vectors by the order of task indices and only keep the unique bin vectors. For example, at $i = 2$, the unique choices of

Algorithm 1 Discretization of action space with task proportions

```
Require: Number of tasks $T$
$\mathcal{T} = ()$                                    ▷ Initialize sequence for storing actions
for $i = 1, \dots, T-1$ do
    $\mathcal{P}_i = \{\}$                            ▷ Set for storing task proportions at $i$
    $\mathcal{B} = \texttt{combinations}([1:i], i)$   ▷ Get bin vectors of size $i$ with bins $1, ..., i$
    $\bar{\mathcal{B}} = \texttt{unique}(\texttt{sort}(\mathcal{B}))$   ▷ Only keep unique bin vectors
    for $\boldsymbol{b}_i \in \hat{\mathcal{B}}$ do
        $\boldsymbol{p}_i = \texttt{bincount}(\boldsymbol{b}_i)/i$   ▷ Calculate task proportion
        $\mathcal{P}_i = \mathcal{P}_i \cup \{\boldsymbol{p}_i\}$   ▷ Add task proportion to set
    end for
    $\mathcal{T}[i] = \mathcal{P}_i$                  ▷ Add set of task proportions to action sequence
end for
return $\mathcal{T}$                                  ▷ Return action sequence as discrete action space
```

vectors are $[1,1], [1,2], [2,2]$, where $[1,1]$ indicates that all samples in the replay memory should be from task 1, $[1,2]$ indicates that half memory is from task 1 and the other half are from task etc. The task proportions are then computed by counting the number of occurrences of each task index in $\boldsymbol{b}_i$ and dividing by i, such that $\boldsymbol{p}_i = \texttt{bincount}(\boldsymbol{b}_i)/(i)$. From this specification, we have built a tree $\mathcal{T}$ with different task proportions that can be selected at different time steps. We construct a replay schedule S by traversing through $\mathcal{T}$ and select a task proportion on every level to append to S. We can then evaluate the replay schedule S by training a network on the CL task sequence and use S to compose the replay memory to use for mitigating catastrophic forgetting at every task. **MK: Question, Wrap algorithm around this paragraph and remove the comments? Algorithm with comments could be placed in Appendix of Paper D.**

We are interested in studying whether the time to replay different tasks is important in the new CL setting. One option is to use a brute-force approach, for example, breadth-first search, and evaluate every possible replay schedule in the tree. However, as the tree grows fast with the number of tasks, we need a scalable method that can perform searches in large action spaces. We suggest using Monte Carlo tree search [122] (MCTS) due to its previous successes in applications with similar conditions as ours [58, 123, 124]. The use of MCTS enables performing search for datasets with longer task horizons. Furthermore, MCTS encourages searches in promising paths based on the selected reward function, which we set as the average accuracy of all tasks after learning the final task achieved by the network. We provide the full details on how MCTS is used to search for replay schedules in Paper C. **MK: Question: More info on MCTS?**

4.3 Meta Policy Learning for Replay Scheduling

In this section, we present an RL-based framework to learn policies for selecting which tasks to replay in CL scenarios. We are interested in learning such policy that can be transferred to new CL scenarios, such as new task orders and new datasets, without any additional computational cost for updating on the new domain. We take a meta-learning approach where the policy learns from episodes of experience collected from training a classifier in CL settings. The experience from the environment is represented as the classification performance on each seen task in the dataset. The policy receives the task performances for basing its action on which task that needs to be replayed at the next time step. Our goal is to obtain a policy that can generalize to be used for replay scheduling in new CL scenarios to mitigate catastrophic forgetting. Next, we describe in more detail how the framework for learning this policy works.

MK: Is this how it should be motivated? Imagine the scenario that we can collect experiences from many users applying their phone to CL scenarios for learning different objects to recognize sequentially. Assume that we can store the collected data (limitation here is privacy!), the models will suffer from catastrophic forgetting as they are trained in CL scenario. But we can then use the collected data to train a replay scheduling policy. The learned policy can then be transferred to new users using their phone in CL settings and the policy is used for mitigating catastrophic forgetting in their environment without additional computational cost.

Limitations here are of course that we potentially need lots of data for learning a policy that generalizes. Additionally, we need lots of training time and hyperparameter tuning as we are dealing with RL. Also, we need to store the data somewhere which is cheap, but it must be secure due to privacy concerns. An alternative there could be to store features instead of raw data, which is not completely flawless (I think that it's possible to revert features back to the real data to some extent) but at least it is a safer alternative. Another option for the data needed can be to gather experience from simulated environments and benchmark datasets. As the policy only takes in states with task performances, we can make use mixes of benchmark datasets and data from real contributing users.

Problem Setting

We let the environment where the agent gathers experiences represent a network and a dataset $\mathcal{D}_{1:T}$ of T tasks that f_{ϕ} should learn in a CL setting. The dataset is split into training, validation, and test sets as $\mathcal{D}_{1:T} = \{\mathcal{D}_{1:T}^{(train)}, \mathcal{D}_{1:T}^{(val)}, \mathcal{D}_{1:T}^{(test)}\}$ respectively. The training sets $\mathcal{D}_{1:T}^{(train)}$ are for the network to learn all T tasks sequentially, while the $\mathcal{D}_{1:T}^{(val)}$ are for evaluating how well the network performs on each task during training. The task performances on the validation sets can be used for dense rewards to the RL agent. The test sets are for final evaluation and are unseen during training as standard practice to avoid overfitting .

4.4 Experiments

4.5 Discussion

Chapter 5

Conclusions and Future Directions

5.1 Conclusions

5.2 Future Directions

- Video data for object recognition instead of images for making systems easier to use. And use a disability-first approach when collecting the data
- Federated Learning for decentralizing model updates

Disability-first Approaches

Federated Learning

Bibliography

[1] Alexander Eitel, Katharina Scheiter, Anne Schüler, Marcus Nyström, and Kenneth Holmqvist. How a picture facilitates the process of learning from text: Evidence for scaffolding. *Learning and Instruction*, 28:48–63, 2013.

[2] Anne Nielsen Hibbing and Joan L. Rankin-Erickson. A picture is worth a thousand words: Using visual images to improve comprehension for middle school struggling readers. *The Reading Teacher*, 56, 2003.

[3] Rupert Bourne, Jaimie D Steinmetz, Seth Flaxman, Paul Svitil Briant, Hugh R Taylor, Serge Resnikoff, Robert James Casson, Amir Abdoli, Eman Abu-Gharbieh, Ashkan Afshin, et al. Trends in prevalence of blindness and distance and near vision impairment over 30 years: an analysis for the global burden of disease study. *The Lancet global health*, 9(2):e130–e143, 2021.

[4] Dragan Ahmetovic, Daisuke Sato, Uran Oh, Tatsuya Ishihara, Kris Kitani, and Chieko Asakawa. Recog: Supporting blind people in recognizing personal objects. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[5] Rabia Jafri, Syed Abid Ali, Hamid R Arabnia, and Shameem Fatima. Computer vision-based object recognition for the visually impaired in an indoors environment: a survey. *The Visual Computer*, 30(11):1197–1222, 2014.

[6] Hernisa Kacorri. Teachable machines for accessibility. *ACM SIGACCESS Accessibility and Computing*, (119):10–18, 2017.

[7] James Coughlan and Roberto Manduchi. Functional assessment of a camera phone-based wayfinding system operated by blind and visually impaired users. *International Journal on Artificial Intelligence Tools*, 18(03):379–397, 2009.

[8] Hernisa Kacorri, Eshed Ohn-Bar, Kris M Kitani, and Chieko Asakawa. Environmental factors in indoor navigation based on real-world trajectories of blind users. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2018.

[9] Jack M Loomis, Reginald G Golledge, Roberta L Klatzky, and James R Marston. Assisting wayfinding in visually impaired travelers. In *Applied Spatial Cognition*, pages 179–202. Psychology Press, 2020.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[12] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[13] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[14] World Health Organization. International Classification of Diseases 11th Revision (ICD-11), 2022. URL https://icd.who.int/en. Accessed 2022-03-22.

[15] World Health Organization. *World report on vision*. World Health Organization, 2019.

[16] Jaimie D Steinmetz, Rupert RA Bourne, Paul Svitil Briant, Seth R Flaxman, Hugh RB Taylor, Jost B Jonas, Amir Aberhe Abdoli, Woldu Aberhe Abrha, Ahmed Abualhasan, Eman Girum Abu-Gharbieh, et al. Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to vision 2020: the right to sight: an analysis for the global burden of disease study. *The Lancet Global Health*, 9(2):e144–e160, 2021.

[17] Roberto Manduchi and James Coughlan. (computer) vision without sight. *Communications of the ACM*, 55(1):96–104, 2012.

[18] Chandrika Jayant, Hanjie Ji, Samuel White, and Jeffrey P Bigham. Supporting blind photography. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, pages 203–210, 2011.

[19] Erin Brady, Meredith Ringel Morris, Yu Zhong, Samuel White, and Jeffrey P Bigham. Visual challenges in the everyday lives of blind people. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2117–2126, 2013.

[20] Microsoft Corporation. Seeing AI, 2017. URL https://www.microsoft.com/en-us/ai/seeing-ai. Accessed 2022-03-14.

[21] Patrick Clary. Lookout: an app to help blind and visually impaired people learn about their surroundings, 2018. URL https://blog.google/outreach-initiatives/accessibility/lookout-app-help-blind-and-visually-impaired-people-learn-about-their-surrou Accessed 2022-04-05.

[22] Inc Cloudsight. TapTapSee, 2013. URL https://taptapseeapp.com/. Accessed 2022-03-22.

[23] Envision. Envision App, 2018. URL https://www.letsenvision.com/envision-app. Accessed 2022-03-22.

[24] OrCam. OrCam MyEye 2, 2019. URL https://www.orcam.com/sv/myeye2/. Accessed 2022-03-22.

[25] Envision. Envision Glasses, 2020. URL https://www.letsenvision.com/envision-glasses. Accessed 2022-03-22.

[26] Be My Eyes. Be My Eyes, 2017. URL https://www.bemyeyes.com/. Accessed 2022-03-22.

[27] Aira Tech Corp. Aira, 2017. URL https://aira.io/. Accessed 2022-03-14.

[28] Tai-Yin Chiu, Yinan Zhao, and Danna Gurari. Assessing image quality issues for real-world problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3646–3656, 2020.

[29] Hernisa Kacorri, Kris M Kitani, Jeffrey P Bigham, and Chieko Asakawa. People with visual impairment training personal object recognizers: Feasibility and challenges. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 5839–5849, 2017.

[30] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning. *arXiv preprint arXiv:1912.01100*, 2019.

[31] Tousif Ahmed, Roberto Hoyle, Kay Connelly, David Crandall, and Apu Kapadia. Privacy concerns and behaviors of people with visual impairments. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3523–3532, 2015.

[32] Danna Gurari, Qing Li, Chi Lin, Yinan Zhao, Anhong Guo, Abigale Stangl, and Jeffrey P Bigham. Vizwiz-priv: A dataset for recognizing the presence and purpose of private visual information in images taken by blind people. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 939–948, 2019.

[33] Roberto Hoyle, Robert Templeman, Steven Armes, Denise Anthony, David Crandall, and Apu Kapadia. Privacy behaviors of lifeloggers using wearable cameras. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 571–582, 2014.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[35] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[36] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4565–4574, 2016.

[37] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[38] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[39] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[40] Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. Language-conditioned graph networks for relational reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10294–10303, 2019.

[41] Xiu-Shen Wei, Yi-Zhe Song, Oisin Mac Aodha, Jianxin Wu, Yuxin Peng, Jinhui Tang, Jian Yang, and Serge Belongie. Fine-grained image analysis with deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[42] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[43] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[44] Patrick E Lanigan, Aaron M Paulos, Andrew W Williams, Dan Rossi, and Priya Narasimhan. Trinetra: Assistive technologies for grocery shopping for the blind. In *2006 10th IEEE International Symposium on Wearable Computers*, pages 147–148. IEEE, 2006.

[45] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[46] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[47] Tyler L Hayes, Giri P Krishnan, Maxim Bazhenov, Hava T Siegelmann, Terrence J Sejnowski, and Christopher Kanan. Replay in deep learning: Current approaches and missing biological elements. *Neural Computation*, 33(11):2908–2950, 2021.

[48] Frank N Dempster. Spacing effects and their implications for theory and practice. *Educational Psychology Review*, 1(4):309–330, 1989.

[49] Hermann Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4):155, 2013.

[50] Karri S Hawley, Katie E Cherry, Emily O Boudreaux, and Erin M Jackson. A comparison of adjusted spaced retrieval versus a uniform expanded retrieval schedule for learning a name–face association in older adults with probable alzheimer's disease. *Journal of Clinical and Experimental Neuropsychology*, 30(6):639–649, 2008.

[51] T. Landauer and Robert Bjork. Optimum rehearsal patterns and name learning. *Practical aspects of memory*, 1, 11 1977.

[52] Paul Smolen, Yili Zhang, and John H Byrne. The right time to learn: mechanisms and optimization of spaced learning. *Nature Reviews Neuroscience*, 17(2):77, 2016.

[53] Weiran Wang, Xinchen Yan, Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*, 2016.

[54] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[55] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[56] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[57] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[58] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[59] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[60] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[61] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[62] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[63] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

[64] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[65] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[66] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[67] Andrew Owens and Alexei A Efros. Audio-visual scene analysis with self-supervised multisensory features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, 2018.

[68] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *ICML*, 2011.

[69] Carina Silberer and Mirella Lapata. Learning grounded meaning representations with autoencoders. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 721–732, 2014.

[70] Michelle A Lee, Yuke Zhu, Peter Zachares, Matthew Tan, Krishnan Srinivasan, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Jeannette Bohg. Making sense of vision and touch: Learning multimodal representations for contact-rich tasks. *IEEE Transactions on Robotics*, 36(3):582–596, 2020.

[71] Mike Wu and Noah Goodman. Multimodal generative models for scalable weakly-supervised learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[72] Yuge Shi, Brooks Paige, Philip Torr, et al. Variational mixture-of-experts autoencoders for multi-modal deep generative models. *Advances in Neural Information Processing Systems*, 32, 2019.

[73] Ramakrishna Vedantam, Ian Fischer, Jonathan Huang, and Kevin Murphy. Generative models of visually grounded imagination. *arXiv preprint arXiv:1705.10762*, 2017.

[74] Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. Joint multimodal learning with deep generative models. *arXiv preprint arXiv:1611.01891*, 2016.

[75] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, 2011.

[76] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[77] Sebastian Bujwid and Josephine Sullivan. Large-scale zero-shot image classification from rich and diverse textual descriptions. *arXiv preprint arXiv:2103.09669*, 2021.

[78] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[79] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[80] Yushan Feng Saihui Hou and Zilei Wang. Vegfru: A domain-specific dataset for fine-grained visual categorization. In *IEEE International Conference on Computer Vision*, 2017.

[81] Xiu-Shen Wei, Quan Cui, Lei Yang, Peng Wang, and Lingqiao Liu. Rpc: A large-scale retail product checkout dataset. *arXiv preprint arXiv:1901.07249*, 2019.

[82] Xiaopeng Zhang, Hongkai Xiong, Wengang Zhou, Weiyao Lin, and Qi Tian. Picking deep filter responses for fine-grained image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1134–1142, 2016.

[83] Yaming Wang, Vlad I Morariu, and Larry S Davis. Learning a discriminative filter bank within a cnn for fine-grained recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4148–4157, 2018.

[84] Yao Ding, Yanzhao Zhou, Yi Zhu, Qixiang Ye, and Jianbin Jiao. Selective sparse sampling for fine-grained image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6599–6608, 2019.

[85] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015.

[86] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[87] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[88] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

[89] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[90] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[91] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[92] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[93] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[94] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[95] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.

[96] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[97] Peter Bailis, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. Macrobase: Prioritizing attention in fast data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 541–556, 2017.

[98] Tom M. Mitchell. Machine learning and data mining. *Communications of the ACM*, 42:30–36, 1999.

[99] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[100] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[101] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[102] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

[103] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[104] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[105] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR, 2018.

[106] Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

[107] Ta-Chu Kao, Kristopher Jensen, Gido van de Ven, Alberto Bernacchia, and Guillaume Hennequin. Natural continual learning: success is a journey, not (just) a destination. *Advances in Neural Information Processing Systems*, 34, 2021.

[108] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[109] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

[110] Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition. *arXiv preprint arXiv:1902.09432*, 2019.

[111] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. *arXiv preprint arXiv:2003.09553*, 2020.

[112] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[113] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[114] Jonathan Schwarz, Siddhant Jayakumar, Razvan Pascanu, Peter E Latham, and Yee Teh. Powerpropagation: A sparsity inducing weight reparameterisation. *Advances in Neural Information Processing Systems*, 34:28889–28903, 2021.

[115] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pages 466–483. Springer, 2020.

[116] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P Lillicrap, and Greg Wayne. Experience replay for continual learning. *arXiv preprint arXiv:1811.11682*, 2018.

[117] Gido M van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.

[118] Gido M van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):1–14, 2020.

[119] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.

[120] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[121] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems*, 32, 2019.

[122] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.

[123] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[124] Muhammad Umar Chaudhry and Jee-Hyong Lee. Feature selection for high dimensional data using monte carlo tree search. *IEEE Access*, 6:76036–76048, 2018.

Part II

Included Papers

Paper A

A Hierarchical Grocery Store Image Dataset with Visual and Semantic Labels

Marcus Klasson
Robotics, Perception, and Learning, EECS
KTH Royal Institute of Technology, Stockholm, Sweden

Cheng Zhang
Microsoft Research
Cambridge, United Kingdom

Hedvig Kjellström
Robotics, Perception, and Learning, EECS
KTH Royal Institute of Technology, Stockholm, Sweden

Abstract

Image classification models built into visual support systems and other assistive devices need to provide accurate predictions about their environment. We focus on an application of assistive technology for people with visual impairments, for daily activities such as shopping or cooking. In this paper, we provide a new benchmark dataset for a challenging task in this application – classification of fruits, vegetables, and refrigerated products, e.g. milk packages and juice cartons, in grocery stores. To enable the learning process to utilize multiple sources of structured information, this dataset not only contains a large volume of natural images but also includes the corresponding information of the product from an online shopping website. Such information encompasses the hierarchical structure of the object classes, as well as an iconic image of each type of object. This dataset can be used to train and evaluate image classification models for helping visually impaired people in

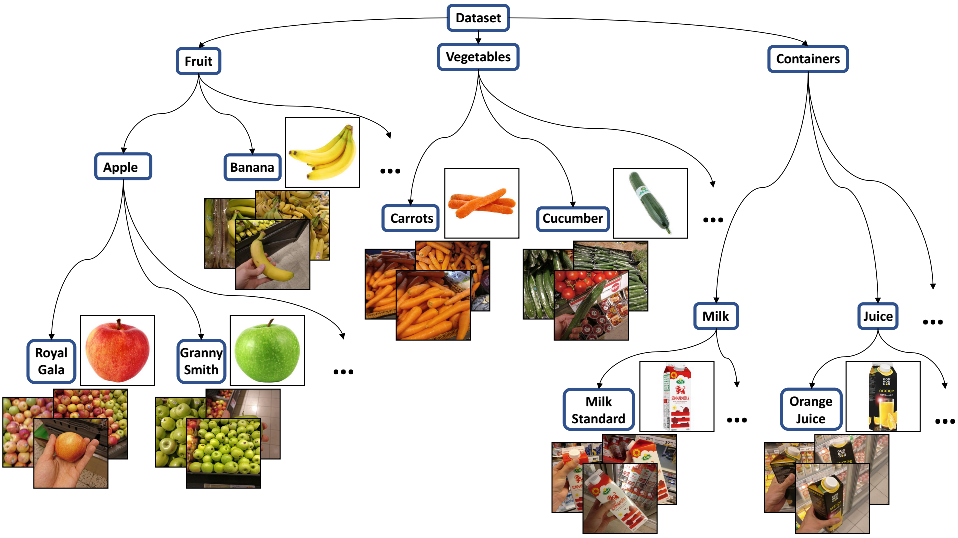


Figure 1: The primary contribution of this paper is a dataset of grocery items, for the purpose of training a visual recognition system to aid visually impaired people. The dataset is organized according to a hierarchical class structure, as illustrated above. A novel aspect of the dataset is that each class, apart from the semantic label, also has a visual label in the form of an iconic image.

natural environments. Additionally, we provide benchmark results evaluated on pretrained convolutional neural networks often used for image understanding purposes, and also a multi-view variational autoencoder, which is capable of utilizing the rich product information in the dataset.

1 Introduction

In this paper, we focus on the application of image recognition models implemented into assistive technologies for people with visual impairments. Such technologies already exist in the form of mobile applications, e.g. Microsoft's Seeing AI [1] and Aipoly Vision [2], and as wearable artificial vision devices, e.g. Orcam MyEye [3] and the Sound of Vision system introduced in [4]. These products have the ability to support people with visual impairments in many different situations, such as reading text documents, describing the user's environment and recognizing people the user may know.

We here address a complementary scenario not handled by current systems on the market: visual support when shopping for grocery items considering a large range of eatable objects, including fruits, vegetables, milk, and juices. In the case of fruits and vegetables, these are usually stacked in large bins in grocery stores as shown in Figure **??**(a-f). A common problem in grocery stores is that similar items are often stacked next to each other; therefore, items are often misplaced into neighboring bins. Figure **??** shows a mix of red and green apples, where it might be difficult for the system to determine which kind of apple is the actual target. Humans can distinguish between groceries without vision to some degree, e.g. by touching and smelling them, but it requires prior knowledge about texture and fragrance of food items.

Moreover, in addition to raw grocery items, there are also items that can only be differentiated with the help of visual information, e.g. milk, juice, and yogurt cartons, see Figure **??**(g-i). Such items usually have barcodes, that are readable using the existing assistive devices described above. However, the barcodes are not easily located by visually impaired persons. Thus, an assistive vision device that fully relies on natural image understanding would be of significant added value for a visually impaired person shopping in a grocery store.

Image recognition models used for this task typically require training images collected in similar environments. However, current benchmark datasets, such as ImageNet [5] and CIFAR-100 [6], do contain images of fruits and vegetables, but are not suitable for this type of assistive application, since the target objects are commonly not presented in this type of natural environments, with occlusion and cluttered backgrounds. To address this issue, we present a novel dataset containing natural images of various raw grocery items and refrigerated products, e.g. milk, juice, and yogurt, taken in grocery stores. As part of our dataset, we collect images taken with single and multiple target objects, from various perspectives, and with noisy backgrounds.

In computer vision, previous studies have shown that model performance can be improved by extending the model to utilize other data sources, e.g. text, audio, in various machine learning tasks [7–10]. Descriptions of images are rather common to computer vision datasets, e.g. Flickr30k [11], whereas the datasets in [8, 12] includes both descriptions and a reference image with clean background to some objects. Therefore, in addition to the natural images, we have collected iconic images with a single object centered in the image (see Figure **??**) and a corresponding product description to each grocery item. In this work, we also demonstrate how we can benefit from using additional information about the natural images by applying the multi-view generative model.

To summarize, the contribution of this paper is a dataset of natural images of raw and refrigerated grocery items, which could be used for evaluating and training image recognition systems to assist visually impaired people in a grocery store. The dataset labels have a hierarchical structure with both coarse- and fine-grained classes (see Figure 1). Moreover, each class also has an iconic image and a product description, which makes the dataset applicable to multimodal learning models. The dataset is described in Section 3.

We provide multiple benchmark results using various deep neural networks, such as Alexnet [13], VGG [14], DenseNet [15], as well as deep generative models, such as VAE [16]. Furthermore, we adapt a multi-view VAE model to make use of the iconic images for each class (Section 4), and show that it improves the classification accuracy given the same model setting (Section 5). Last, we discuss possible future directions for fully using the additional information provided with the dataset and adopt more advanced machine learning methods, such as visual-semantic embeddings, to learn efficient representations of the images.

2 Related Work

Many popular image datasets have been collected by downloading images from the web [5, 6, 8, 12, 17–21]. If the dataset contains a large amount of images, it is convenient to make use of crowdsourcing to get annotations for recognition tasks [5, 6, 22]. For some datasets, the crowdsourcers are also asked to put bounding boxes around the object to be labeled for object detection tasks [8, 17, 20]. In [18] and [6], the target objects are usually centered and takes up most content of the image itself. Another significant characteristic is that web images usually are biased in the sense that they have been taken with the object focus in mind; they have good lighting settings and are typically clean from occlusions, since the collectors have used general search words for the object classes, e.g. *car*, *horse*, or *apple*.

Some datasets include additional information about the images beyond the single class label, e.g. text descriptions of what is present in the image and bounding boxes around objects. These datasets can be used in several different computer vision tasks, such as image classification, object detection, and image segmentation. Structured labeling is another important property of a dataset, which provides flexibility when classifying images. In [8, 12], all of these features exist and moreover they include reference images to each object class, which in [12] is used for labeling multiple categories present in images, while in [8] these images are used for fine-grained recognition. Our dataset includes a reference image, i.e. the iconic image, and a product description for every class, and we have also labeled the grocery items in a structured manner.

Other image datasets of fruits and vegetables for classification purposes are the FIDS30 database [23] and the dataset in [24]. The images in FIDS30 were downloaded from the web and contain background noise as well as single or multiple instances of the object. In [24], all pixels belonging to the object are extracted from the original image, such that all images have white backgrounds with the same brightness condition. There also exist datasets for detecting fruits in orchards for robotic harvesting purposes, which are very challenging since the images contain plenty of background and various lighting conditions, and the targeted fruits are often occluded or of the same color as the background [25, 26].

Another dataset that is highly relevant to our application need is presented in [27]. They collected a dataset for training and evaluating the image classifier by extracting images from video recordings of 23 main classes, which are subdivided into 98 classes, of raw grocery items (fruits and vegetables) in different grocery stores. Using this dataset, a mobile application was developed to recognize food products in grocery store environments, which provides the user with details and health recommendations about the item along with other proposals of similar food items. For each class, there exists a product description with nutrition values to assist the user in shopping scenarios. The main difference between this work and our dataset is firstly the clean iconic images (visual labels) for each class in our dataset, and secondly that we have also collected images of refrigerated items, such as dairy and juice containers, where visual information is required to distinguish between the products.

3 Our Dataset

We have collected images from fruit and vegetable sections and refrigerated sections with dairy and juice products in 18 different grocery stores. The dataset consists of 5125 images from 81 fine-grained classes, where the number of images in each class range from 30 to 138. Figure 2 displays a histogram over the number of images per class. As illustrated in Figure 1, the class structure is hierarchical, and there are 46 coarse-grained classes.

Figure **??** shows examples of the collected natural images. For each fine-grained class, we have downloaded an iconic image of the item and also a product description including origin country, an appreciated weight and nutrient values of the item from a grocery store website. Some examples of downloaded iconic images can be seen in Figure **??**.

Our aim has been to collect the natural images under the same condition as they would be as part of an assistive application on a mobile phone. All images have been taken with a 16-megapixel Android smartphone camera from different distances and angles. Occasionally, the images include other items in the background or even items that have been misplaced in the wrong shelf along with the targeted item. It is important that image classifiers that are used for assisting devices are capable of performing well with such noise since these are typical settings in a grocery store environments. The lighting conditions in the images can also vary depending on where the items are located in the store. Sometimes the images are taken while the photographer is holding the item in the hand. This is often the case for refrigerated products since these containers are usually stacked compactly in the refrigerators. For these images, we have consciously varied the position of the object, such that the item is not always centered in the image or present in its entirety.

We also split the data into a training set and test set based on the application need. Since the images have been taken in several different stores at specific days and time stamps, parts of the data will have similar lighting conditions and backgrounds for each photo occasion. To remove any such biasing correlations, all images of a certain class taken at a certain store are assigned to either the test set or training set. Moreover, we balance the class sizes to as large extent as possible in both the training and test set. After the partitioning, the training and test set contains 2640 and 2485 images respectively. Predefining a training and test set also makes it easier for other users to compare their results to the evaluations in this paper.

The task is to classify natural images using mobile devices to aid visually impaired people. The additional information such as the hierarchical structure of the class labels, iconic images, and product descriptions can be used to improve the performance of the computer vision system. Every class label is associated with a product description. Thus, the product description itself can be part of the output for visually impaired persons as they may not be able to read what is printed on a carton box or a label

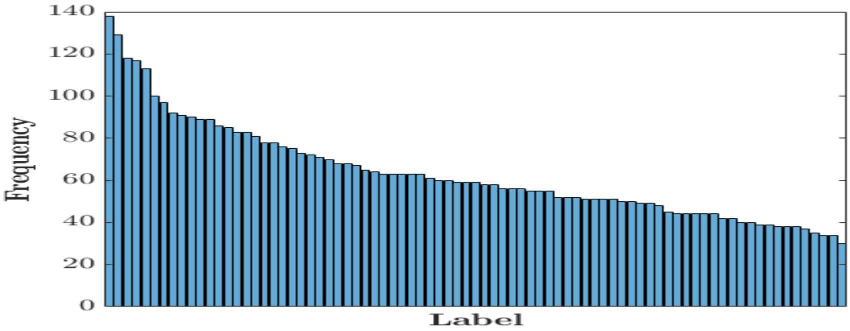


Figure 2: Histogram over the number of images in each class in the dataset.

tag on a fruit bin in the store.

The dataset is intended for research purposes and we are open to contributions with more images and new suitable classes. Our dataset is available at https://github.com/marcusklasson/GroceryStoreDataset. Detailed instructions on how to contribute to the dataset can be found on our dataset webpage.

4 Classification Methods

We here describe the classification methods and approaches that we have used to provide benchmark results to the dataset. We apply both deterministic deep neural networks as well as a deep generative model used for representation learning to the natural images that we have collected. Furthermore, we utilize the additional information – iconic images – from our dataset with a multi-view deep generative model. This model can utilize different data sources and obtain superior representation quality as well as high interpretability. For a fair evaluation, we use a linear classifier with the learned representation from the different methods.

Deep Neural Networks. CNNs have been the state-of-the-art models in image classification ever since AlexNet [34] achieved the best classification accuracy in ILSVRC in 2012. However, in general, computer vision models require lots of labeled data to achieve satisfactory performance, which has resulted in interest for adapting CNNs that have already been trained on a large amount of training data to other image datasets. When adapting pretrained CNNs to new datasets, we can either use it directly as a feature extractor, a.k.a use the off-the-shelf features, [28, 29], or fine-tune it [30–34]. Using off-the-shelf features, we need to specify which feature representation we should extract from the network and use these for training a new classifier. Fine-tuning a CNN involves adjusting the pretrained model parameters, such that the network can e.g. classify images from a dataset different from what the CNN was trained on before. We can either choose to fine-tune the whole network or select some layer parameters to adjust while keeping the others fixed. One important factor on deciding which approach to choose is the size of the new dataset and how similar the new dataset is to the dataset which the CNN was previously trained on. A rule of thumb here is that the closer the features are to the classification layer, the features become more specific to the training data and task [33].

Using off-the-shelf CNN features and fine-tuned CNNs have been successfully applied in [28, 29] and [30, 31, 34] respectively. In [28, 29], it is shown that the pretrained features have sufficient representational power to generalize well to other visual recognition tasks with simple linear classifiers, such as Support Vector Machines (SVMs), without fine-tuning the parameters of the CNN to the new task. In [30, 34], all CNN parameters are fine-tuned, whereas in [31] the pretrained CNN layer parameters are kept fixed and only an adaptation layer of two fully connected layers are trained on the new task. The results from these works motivate why we should evaluate our dataset on fine-tuned CNNs or linear classifiers trained on off-the-shelf feature representations instead of training an image recognition model from scratch.

Variational Autoencoders with only natural images. Deep generative models, e.g. the variational autoencoder (VAE) [16, 35, 36], have become widely used in the machine learning community thanks to their generative nature. We thus use VAEs for representation learning as the second benchmarking method. For efficiency, we use low-level pretrained features from a CNN as inputs to the VAE.

The latent representations from VAEs are encodings of the underlying factors for how the data are generated. VAEs belongs to the family of latent variable models, which commonly has the form $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, where $p(\mathbf{z})$ is a prior distribution over the latent variables $\mathbf{z}$ and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is the likelihood over the data $\mathbf{x}$ given $\mathbf{z}$. The prior distribution is often assumed to be Gaussian, $p(\mathbf{z}) = \mathcal{N}(\mathbf{z} \,|\, \mathbf{0}, \mathbf{I})$, whereas the likelihood distribution depends on the values of $\mathbf{x}$. The likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is referred to as a decoder represented as a neural network parameterized by $\boldsymbol{\theta}$. An encoder network $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ parameterized by $\boldsymbol{\phi}$ is introduced as an approximation of the true posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, which is intractable since it requires computing the integral $p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\, d\mathbf{z}$. When the prior distribution is a Gaussian, the approximate posterior is also modeled as a Gaussian, $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z} \,|\, \boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}^2(\mathbf{x}) \odot \mathbf{I})$, with some mean $\boldsymbol{\mu}(\mathbf{x})$ and variance $\boldsymbol{\sigma}^2(\mathbf{x})$ computed by the encoder network. The goal is to maximize the marginal log-likelihood by defining a lower bound using $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned} \log p_{\boldsymbol{\theta}}(\mathbf{x}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = & \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \right] \\ & - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z})). \end{aligned} \tag{1}$$

The last term is the Kullback-Leibler (KL) divergence of the approximate posterior from the true posterior. The lower bound $\mathcal{L}$ is called the evidence lower bound (ELBO) and can be optimized with stochastic gradient descent via backpropagation [16, 37]. VAE is a probabilistic framework. Many extensions such as utilizing structured priors [38] or using continual learning [39] have been explored. In the following method, we describe how to make use of the iconic images while retaining the unsupervised learning setting in VAEs.

Utilizing iconic images with multi-view VAEs. Utilizing extra information has shown to be useful in many applications with various model designs [38, 40–43]. For computer vision tasks, natural language is the most commonly used modality to aid the visual representation learning. However, the consistency of the language and visual embeddings has no guarantee. As an example with our dataset, the product description of a Royal Gala apple explains the appearance of a red apple. But if the description is represented with word embeddings, e.g. word2vec [44], the word 'royal' will probably be more similar to the words 'king' and 'queen' than 'apple'. Therefore, if available, additional visual information about objects might be more beneficial for learning meaningful representations instead of text. In this work, with our collected dataset, we propose to utilize the iconic images for the representation learning of natural images using a multi-view VAE. Since the natural images can include background noise and grocery items different from the targeted one, the role of the iconic image will be to guide the model to which features that are of interest in the natural image.

The VAE can be extended to modeling multiple views of data, where a latent variable $\mathbf{z}$ is assumed to have generated the views [40, 42]. Considering two views $\mathbf{x}$ and $\mathbf{y}$, the joint distribution over the paired random variables ($\mathbf{x}$, $\mathbf{y}$) and latent variable $\mathbf{z}$ can be written as $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\mathbf{z}) p_{\boldsymbol{\theta}^{(1)}}(\mathbf{x} \,|\, \mathbf{z}) p_{\boldsymbol{\theta}^{(2)}}(\mathbf{y} \,|\, \mathbf{z})$, where both $p_{\boldsymbol{\theta}^{(1)}}(\mathbf{x} \,|\, \mathbf{z})$ and $p_{\boldsymbol{\theta}^{(2)}}(\mathbf{y} \,|\, \mathbf{z})$ are represented as neural networks with parameters $\boldsymbol{\theta}^{(1)}$ and $\boldsymbol{\theta}^{(2)}$. Assuming that the latent variable $\mathbf{z}$ can reconstruct both $\mathbf{x}$ and $\mathbf{y}$ when only $\mathbf{x}$ is encoded into $\mathbf{z}$ by the encoder $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, then the ELBO is written as

$$\begin{aligned} \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y}) \geq & \, \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{y}) \\ = & \, \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\boldsymbol{\theta}^{(1)}}(\mathbf{x}|\mathbf{z}) + \log p_{\boldsymbol{\theta}^{(2)}}(\mathbf{y}|\mathbf{z}) \right] \\ & - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z})). \end{aligned} \tag{2}$$

This model is referred to as variational autoencoder canonical correlation analysis (VAE-CCA) and was introduced in [42]. The main motivation for using VAE-CCA is that the latent representations need to contain information about reconstructing both natural and iconic images. The main motivation for using VAE-CCA is that the latent representation needs to preserve information about how both the natural and iconic images are reconstructed. This also allows us to produce iconic images from new natural images to enhance the interpretability of the latent representation of VAE-CCA (see Section 5) [40].

5 Experimental Results

We apply the three different types of models described in Section 4 to our dataset and evaluate their performance. The natural images are propagated through a CNN pretrained on ImageNet to extract feature vectors. We experiment with both the off-the-shelf features as well as fine-tuning the CNN. When using off-the-shelf features, we simply extract feature vectors and train an SVM on those. For the fine-tuned CNN, we report both results from the softmax classifier used in the actual fine-tuning procedure and training an SVM with extracted fine-tuned feature vectors.

These extracted feature vectors are also used for VAE and VAE-CCA which makes further compression. We perform classification for those VAE based models by training a classifier, e.g. an SVM, on the data encoded into the latent representation. We use this classification approach for both VAE and VAE-CCA. In all classification experiments, except when we fine-

Table 1: Fine-grained classification (81 classes) accuracies with the methods described in Section 5.1. Each row displays from which network architecture and layer that we extracted the feature vectors of the natural images. The columns show the result from the classifiers that we used (see Section 5.1).

	SVM	SVM-ft	VAE+SVM	VAE+SVM-ft	VAE-CCA+SVM	VAE-CCA+SVM-ft
AlexNet$_6$	69.2	72.6	65.6	70.7	67.8	71.5
AlexNet$_7$	65.0	70.7	63.0	68.7	65.0	70.9
VGG16$_6$	62.1	73.3	57.5	71.9	60.7	73.0
VGG16$_7$	57.3	71.7	56.8	67.8	56.8	71.3
DenseNet-169	72.5	85.0	65.4	79.1	72.6	80.4

Table 2: Coarse-grained classification (46 classes) accuracies with an SVM for the methods described in Section 5.1 that uses off-the-shelf feature representations. Each row displays from network architecture and layer that we extracted the feature vectors of the natural images and the columns show the result for the classification methods.

	SVM	VAE+SVM	VAE-CCA+SVM
AlexNet$_6$	78.0	74.2	76.4
AlexNet$_7$	75.4	73.2	74.4
VGG16$_6$	76.6	74.2	74.9
VGG16$_7$	72.8	71.7	72.3
DenseNet-169	85.2	79.5	82.0

Table 3: Fine-grained classification accuracies from fine-tuned CNNs pretrained on ImageNet, where the column shows which architecture that has been fine-tuned. A standard softmax layer is used as the last classification layer.

	AlexNet	VGG16	DenseNet-169
Fine-tune	69.3	73.8	84.0

tune the CNN, we use a linear SVM trained with the one-vs-one approach as in [29].

We experiment with three different pretrained CNN architectures, namely AlexNet [34], VGG16 [14] and DenseNet-169 [89]. For AlexNet and VGG16, we extract feature vectors of size 4096 from the two last fully connected (FC) layers before the classification layer. The features from the n^{th} hidden layer are denoted as AlexNet$_n$ and VGG16$_n$. As an example, the last hidden FC layer in AlexNet is denoted as AlexNet$_7$, the input of which is output from AlexNet$_6$. For DenseNet-169, we extract the features of size 1664 from the average pooling layer before its classification layer.

5.1 Experimental Setups

The following setups were used in the experiments:

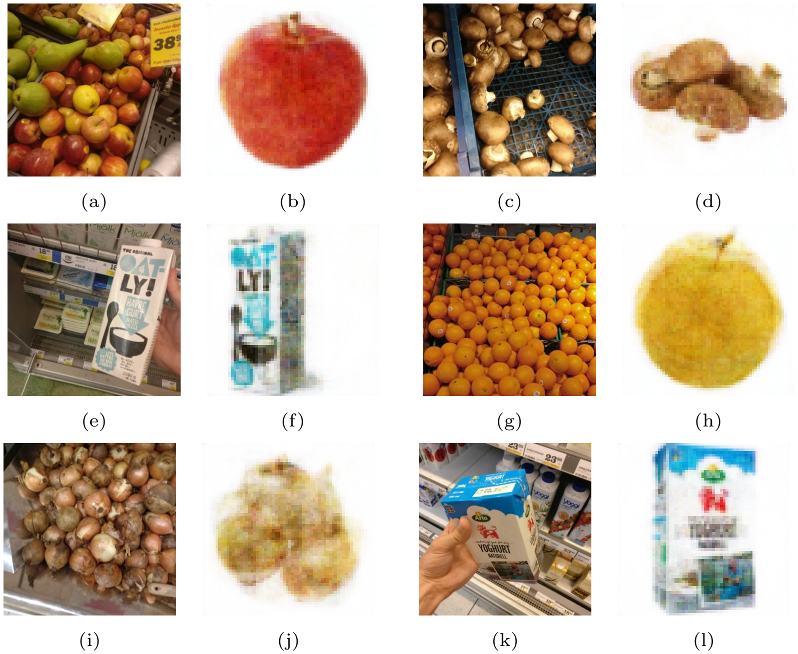
Setup 1. Train an SVM on extracted off-the-shelf features from a pretrained CNN, which is denoted as SVM in the results. We also fine-tune the CNN by replacing the final layer with a new softmax layer and denote these results as Fine-tune. We denote training an SVM on extracted finetuned feature vectors as SVM-ft.

Setup 2. Extract feature vectors with a pretrained CNN of the natural images and learn a latent representation $\mathbf{z}$ with a VAE. Then the data is encoded into the latent space and we train an SVM with these latent representations, which used for classification. We denote the results as VAE+SVM when using off-the-shelf feature vectors, whereas using the fine-tuned feature vectors are denoted as VAE+SVM-ft. In all experiments with the VAE, we used the architecture from [**?**], i.e. the latent layer having 200 hidden units and both encoder and decoder consisting of two FC layers with 1,000 hidden units each.

Setup 3. Each natural image is paired with its corresponding iconic image. We train VAE-CCA similarly as the VAE, but instead, we learn a joint latent representation that is used to reconstruct the extracted feature vectors $\mathbf{x}$ and the iconic images $\mathbf{y}$. The classification is performed with the same steps as in Setup 2 and denotes the results similarly with VAE-CCA+SVM and VAE-CCA+SVM-ft. Our VAE-CCA model takes the feature vectors $\mathbf{x}$ as input and encodes them into a latent layer with 200 hidden units. The encoder and the feature vector decoder uses the same architecture, i.e. two FC layers with 512 hidden units, whereas the iconic image decoder uses the DCGAN [**?**] architecture.

Figure **??** displays the three experimental setups described above. We report both fine-grained and coarse-grained classification results with an SVM in Table 1 and 2 respectively. In Table 3, we report the fine-grained classification results from fine-tuned CNNs.

When fine-tuning the CNNs, we replace the final layer with a softmax layer applicable to our dataset with randomly initialized weights drawn from a Gaussian with zero mean and standard deviation 0.01 [34]. For AlexNet and VGG16, we fine-tune the networks for 30 epochs with two different learning rates, 0.01 for the new classification layer and 0.001 for the pretrained layers. Both learning rates are reduced by half after every fifth epoch. The DenseNet-169 is fine-tuned for 30 epochs with momentum of 0.9 and an initial learning rate of 0.001, which decays with 10^{-6} after each epoch. We report the classification results from the softmax



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 3: Examples of six natural images in the test set that have been decoded into product iconic images by the iconic image decoder $p_{\boldsymbol{\theta}^{(2)}}(\mathbf{y} \mid \mathbf{z})$ using VAE-CCA model as in Figure ??. This result is obtained with the fine-tuned DenseNet-169 features, which corresponds to VAE-CCA+SVM-ft in Table 1. We show the natural image and corresponding decoded iconic image next to each other. The classes for all images are (a, b) Royal Gala Apple, (c, d) Brown Cap Mushroom, (e, f) Oatly Oatgurt, (g, h) Orange, (i, j) Onion, and (k, l) Arla Natural Yogurt.

activation after the fine-tuned classification layer. We also report classification results from an SVM trained with feature representations from a fine-tuned CNN, which are extracted from FC6 and FC7 of the AlexNet and VGG16 and from the last average pooling layer in DenseNet-169.

The VAE and VAE-CCA models are trained for 50 epochs with Adam [?] for optimizing the ELBOs in Equation 1 and 2 respectively. We use a constant learning rate of 0.0001 and set the minibatch size to 64. The extracted feature vectors are rescaled with standardization before training the VAE and VAE-CCA models to stabilize the learning.

5.2 Results

The fine-grained classification results for all methods using an SVM as classifier are shown in Table 1. We also provide coarse-grained classification results for some of the methods in Table 2 to demonstrate the possibility of hierarchical evaluation that our labeling of the data provides (see Figure 1). The accuracies in the coarse-grained classification are naturally higher than the accuracies in the corresponding columns in Table 1. Table 3 shows fine-grained classification accuracies from a softmax classifier in the fine-tuned CNNs. We note that fine-tuning the networks gives consistently better results than training an SVM on off-the-shelf features (see Table 1).

Fine-tuning the entire network results improves the classification performance consistently for each method in Table 1. The performance is clearly enhanced for features extracted from fine-tuned VGG16 and DenseNet-169, which improves the classification accuracy by 10% in most cases for SVM-ft, VAE+SVM-ft, and VAE-CCA+SVM-ft. For AlexNet and VGG16, we see that the performance drops when extracting the features from layer FC7 instead of FC6. The reason might be that the off-the-shelf features in FC7 are more difficult to transfer to other datasets since the weights are biased towards classifying objects in the ImageNet database. The performance drops also when we use fine-tuned features, which could be due to the small learning rate we use for the pretrained layers, such that the later layers are still ImageNet-specific. We might circumvent this drop by increasing the learning rate for the later pretrained layers and keeping the learning rate for earlier layers small.

The VAE-CCA model achieves mostly higher classification accuracies than the VAE model in both Table 1 and 2. This indicates that the latent representation separates the classes more distinctly than the VAE by jointly learning to reconstruct the extracted feature vectors and iconic images. However, further compressing the feature vectors with VAE and VAE-CCA will lower the classification accuracy compared to applying the feature vectors to a classifier directly. Since both VAE and VAE-CCA compresses the feature vectors into the latent representation, there is a risk of losing information about the natural images. We might receive better performance by increasing the dimension of the latent representation at the expense of speed in both training and classification.

In Figure 3, we show results from the iconic image decoder $p_{\boldsymbol{\theta}^{(2)}}(\mathbf{y} \,|\, \mathbf{z})$ when translating natural images from the test set into iconic images with VAE-CCA and a fine-tuned DenseNet-169 as feature extractor. Such visualization can demonstrate the quality of the representation using the model, as well as enhancing the interpretability of the method. Using VAE-CCA in the proposed manner, we see that with challenging natural images, the model is still able to learn an effective representation which can be decoded to the correct iconic image. For example, some pears have been misplaced in the bin for Royal Gala apples in Figure 3a, but still the image decoder manages to decode a blurry red apple seen in Figure 3b. In Figure 3h, a mix of an orange and an apple are decoded from a bin of oranges in Figure 3g, which indicates these fruits are encoded close to each other in the learned latent space. Even if Figure 3e includes much of the background, the iconic image decoder is still able to reconstruct the iconic images accurately in Figure 3f, which illustrates that the latent representation is able to explain away irrelevant information in the natural image and preserved the features of the oatgurt package. Thus, using VAE-CCA with iconic images as the second view not only advances the classification accuracy but also provides us with the means to understand the model.

6 Conclusions

This paper presents a dataset of images of various raw and packaged grocery items, such as fruits, vegetables, and dairy and juice products. We have used a structured labeling of the items, such that grocery items can be grouped into more general (coarse-grained) classes and also divided into fine-grained classes. For each class, we have a clean iconic image and a text description of the item, which can be used for adding visual and semantic information about the items in the modeling. The intended use of this dataset is to train and benchmark assistive systems for visually impaired people when they shop in a grocery store. Such a system would complement existing visual assistive technology, which is confined to grocery items with barcodes. We also present preliminary benchmark results for the dataset on the task of image classification.

We make the dataset publicly available for research purposes at https://github.com/marcusklasson/GroceryStoreDataset. Additionally, we will both continue collecting natural images, as well as ask for public contributions of natural images in shopping scenarios to enlarge our dataset.

For future research, we will advance our model design to utilize the structured nature of our labels. Additionally, we will design a model that use the product description of the objects in addition to the iconic images. One immediate next step is to extend the current VAE-CCA model to three views, where the third view is the description of the product.

References

[1] Microsoft Seeing AI app. https://www.microsoft.com/en-us/seeing-ai/. Accessed on 2018-02-22.

[2] Aipoly Vision app. https://www.aipoly.com/. Accessed on 2018-02-28.

[3] Orcam. https://www.orcam.com/en/. Accessed on 2018-02-28.

[4] S. Caraiman, A. Morar, M. Owczarek, A. Burlacu, D. Rzeszotarski, N. Botezatu, P. Herghelegiu, F. Moldoveanu, P. Strumillo, and A. Moldoveanu. Computer vision for the visually impaired: the sound of vision system. In *IEEE International Conference on Computer Vision Workshops*, 2017.

[5] J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[6] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[7] Andrea Frome, Greg Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems*, 2013.

[8] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. Fine-grained car detection for visual census estimation. In *AAAI Conference on Artificial Intelligence*, 2017.

[9] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[10] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. In *International Conference on Machine Learning*, 2011.

[11] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *IEEE International Conference on Computer Vision*, 2015.

[12] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[16] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[17] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, Jun 2010.

[18] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[19] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[20] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[21] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[22] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

[23] Škrjanec Marko. Automatic fruit recognition using computer vision. (Mentor: Matej Kristan), Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2013. FIDS30 dataset was accessed 2018-02-24 at http://www.vicos.si/Downloads/FIDS30.

[24] Horea Muresan and Mihai Oltean. Fruit recognition from images using deep learning. Technical report, Babes-Bolyai University, 2017.

[25] Suchet Bargoti and James Patrick Underwood. Deep fruit detection in orchards. In *IEEE International Conference on Robotics and Automation*, 2017.

[26] Inkyu Sa, Zongyuan Ge, Feras Dayoub, Ben Upcroft, Tristan Perez, and Chris McCool. Deepfruits: A fruit detection system using deep neural networks. *Sensors*, 16(8):1222, 2016.

[27] Georg Waltner, Michael Schwarz, Stefan Ladstätter, Anna Weber, Patrick Luley, Horst Bischof, Meinrad Lindschinger, Irene Schmid, and Lucas Paletta. Mango - mobile augmented reality with functional eating guidance and food awareness. In *International Workshop on Multimedia Assisted Dietary Management*, 2015.

[28] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, 2014.

[29] Ali Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[30] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014.

[31] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[32] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.

[33] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, 2014.

[34] Ning Zhang, Jeff Donahue, Ross B. Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European Conference on Computer Vision*, 2014.

[35] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

[36] Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. Advances in variational inference. *arXiv preprint arXiv:1711.05597*, 2017.

[37] Carl Doersch. Tutorial on variational autoencoders. *CoRR*, abs/1606.05908, 2016.

[38] Judith Butepage, Jiawei He, Cheng Zhang, Leonid Sigal, and Stephan Mandt. Informed priors for deep representation learning. In *Symposium on Advances in Approximate Bayesian Inference*, 2018.

[39] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[40] Ramakrishna Vedantam, Ian Fischer, Jonathan Huang, and Kevin Murphy. Generative models of visually grounded imagination. In *International Conference on Learning Representations*, 2018.

[41] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.

[42] Weiran Wang, Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *CoRR*, abs/1610.03454, 2016.

[43] Cheng Zhang, Hedvig Kjellström, and Carl Henrik Ek. Inter-battery topic representation learning. In *European Conference on Computer Vision*, pages 210–226. Springer, 2016.

[44] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 2013.

Paper B

Name for Paper B

Marcus Klasson, Cheng Zhang, Hedvig Kjellström

Abstract

Abstract aby stract

1 Introduction

hej hej här är en artikel

what do you think this is?