h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Hochschule Darmstadt

– Fachbereich Mathematik und Naturwissenschaften–

Data-Efficient and Iterative Metric Learning for Open Set Classification in an Industrial Setting

Thesis for obtaining the academic degree

Master of Science (M.Sc.)

for the major in Data Science

submitted by

Samuel Kees

Matrikelnummer: 729402

Referent	:	Prof. Dr. Andreas Thümmel
Korreferent	:	Prof. Dr. Markus Döhring

Ausgabedatum	:	28.09.2020
Abgabedatum	:	15.03.2021

Samuel Kees : *Data-Efficient and Iterative Metric Learning for Open Set Classification in an Industrial Setting*, © 15. March 2021

ERKLÄRUNG

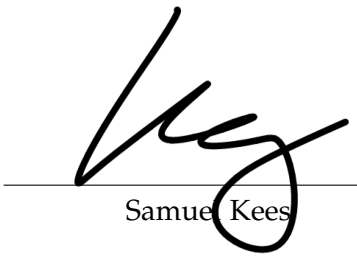
Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 15. March 2021



Samuel Kees

ABSTRACT

Over the last decades many mechanical tasks have been automated to a certain degree by machines. In recent years, the trend has shifted to automate more complex tasks, while imitating the human ability to make intuitive decisions using Machine Learning (ML). Many ML algorithms are known to need a lot of curated and labeled data to be trained. The industry can often easily provide raw data, but lacks the knowledge needed to apply it to ML. This poses the challenge of acquiring relevant data and processing it to a degree where it becomes usable for ML tasks. For the industry, labeling is much harder than just providing the raw data. The reason for this is, that labeling in the industry often requires human experts, who are always scarce and expensive. This gives an incentive to improve the labeling process so that it can be done by non-experts or machines, with minimal expert supervision. This work proposes a concept, namely the Data Refinery (DR), to efficiently label data by exploring and exploiting semantic information, based on Deep Metric Learning (DMeL). This is not achieved without human expertise, but instead with a decreased effort, thus simplifying an quickening the process of acquiring relevant data. For this purpose two DRs were implemented which only differ in their selection strategy. One DR selected data samples to show to a human based on a score (DR-S) and the other randomly without considering the score at all (DR-R). The idea behind the score was to know in advance which data samples are valuable for training. The results indicate that DR-S, in contrast to DR-R, specifically adds new data samples for training in classes where it underperformed. Contrary to the expectations, this did not lead to an overall improvement of DR-S when compared with DR-R, but to a similar performance for a classification task. Regardless, the goal of the case study was not to train a state of the art classification model for a given task, but to efficiently mine data for such a model. Even though the success of the case study is only moderate performance wise, the new approach of the DR as a concept proves to be a promising way to acquire specific data that can be used to create curated data sets. Such curated data sets, are not only valuable for the industry but can be used for all kinds of ML tasks throughout all industries.

ZUSAMMENFASSUNG

In den letzten Jahrzehnten wurden viele mechanische Aufgaben bis zu einem gewissen Grad durch Maschinen automatisiert. In den letzten Jahren hat sich der Trend dahingehend verschoben, komplexere Aufgaben zu automatisieren und dabei die menschliche Fähigkeit, intuitive Entscheidungen zu treffen, mit Machine Learning (ML) zu imitieren. Es ist bekannt, dass viele ML Algorithmen eine Menge kuratierter und gelabelte Daten benötigen, um trainiert zu werden. Die Industrie kann meist leicht Rohdaten zur Verfügung stellen, aber es fehlt ihr an Wissen, um diese auf ML anzuwenden. Daraus ergibt sich die Herausforderung, relevante Daten zu beschaffen und sie so weit zu verarbeiten, dass sie für ML Aufgaben nutzbar werden. Für die Industrie ist das labeln viel schwieriger als die Bereitstellung der Rohdaten. Der Grund dafür ist, dass das labeln in der Industrie oft menschliche Experten erfordert, die immer knapp und teuer sind. Dies gibt einen Anreiz, den Labelings-prozess so zu verbessern, dass er von Nicht-Experten oder Maschinen mit minimaler Expertenaufsicht durchgeführt werden kann. Diese Arbeit schlägt ein Konzept vor, die sogenannte Data Refinery (DR), um Daten effizient zu labeln, indem semantische Informationen erforscht und genutzt werden, basierend auf Deep Metric Learning (DMeL). Dies geschieht nicht ohne menschliches Fachwissen, sondern mit einem verringerten Aufwand, wodurch der Prozess relevante Daten zu beschaffen, vereinfacht und beschleunigt wird. Zu diesem Zweck wurden zwei DR implementiert, die sich nur in ihrer Selektionsstrategie unterscheiden. Die eine DR wählt Datenmuster, die einem Menschen gezeigt werden sollen, anhand eines Scores aus (DR-S), die andere zufällig, ohne den Score überhaupt zu berücksichtigen (DR-R). Die Idee hinter dem Score war, im Voraus zu wissen, welche Datenproben für das Training wertvoll sind. Die Ergebnisse zeigen, dass DR-S im Gegensatz zu DR-R gezielt neue Datenproben für das Training in Klassen hinzufügt, in denen es unterdurchschnittlich abschneidet. Entgegen den Erwartungen führte dies nicht zu einer Gesamtverbesserung von DR-S im Vergleich zu DR-R, sondern zu einer ähnlichen Leistung bei einer Klassifikationsaufgabe. Unabhängig davon war das Ziel der Fallstudie nicht das Trainieren eines State-of-the-Art-Klassifikationsmodells für eine bestimmte Aufgabe, sondern das effiziente Mining von Daten für ein solches Modell. Auch wenn der Erfolg der Fallstudie in Bezug auf die Performance nur mäßig ist, erweist sich der neue Ansatz des DR als Konzept als vielversprechender Weg, um spezifische Daten zu gewinnen, die zur Erstellung kuratierter Datensätze verwendet werden können. Solche kuratierten Datensätze sind nicht nur für die Industrie wertvoll, sondern können für alle Arten von ML Aufgaben in allen Branchen verwendet werden.

Die Unterschiedlichkeit der Herangehensweisen ist nie der Anlaß, das Gespräch abzubrechen, sondern bildet die Basis wechselseitiger Faszination.

— Heinz Von Foerster - Physiker, Kybernetiker und Philosoph [31]

ACKNOWLEDGMENTS

The author of this paper would like to thank all those who made this work possible. First of all Sascha Desch, without him this work would not have been written. Thanks for that and thanks for all the heated and relaxed discussions, which contributed immensely to this work.
Next, the author would like to thank Schulz Systemtechnik GmbH and especially the head of the software department Daniel Magin.
The author also expresses deepest gratitude to the department located in Kassel for their trust and courage to go new ways. Thank you Martin Brückemann, Udo Richter, Helmut Heyne and Eckart Willich.
For the scientific supervision, the author expresses gratitude to Prof. Dr. Andreas Thümmel and Prof. Dr. Markus Döhring.
The author would also like to thank the many label workers who found the time to support this work.
Finally, the author thanks all the proofreaders, without you and your feedback this work would be neither readable nor understandable. Thank you Dan, Betül and Elena.
A special thank you for repeated, almost endless criticisms of the author's interpretation of the English language is addressed to Melanie. Dankeschön.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACRONYMS

CNN Convolutional Neural Network
ML Machine Learning
K-NN K-Nearest Neighbors Algorithm
MeL Metric Learning
DMeL Deep Metric Learning
FS Feature Space
FE Feature Extractor
ANN Artificial Neural Network
CV Computer Vision
DCV Deep Computer Vision
CrI Crate Inspector
CAD Computer-Aided Design
PET Polyethylene Terephthalate
UV Ultraviolet
UUID Universally Unique Identifier
CI Confidence Interval
BML-R Bottle-Marketing-Label Recognition
BT-R Bottle-Type Recognition
DR Data Refinery
DR-S Data Refinery Score Sampling
DR-R Data Refinery Random Sampling
TP True Positive
TN True Negative
FP False Positive
FN False Negative
MiR Micro Recall
MaR Macro Recall
MiP Micro Precision
MaP Macro Precision
SD Standard Deviation
HITL Human-in-the-Loop
EU Expected Utility

DNA Desoxyribonucleic Acid

FGVC Fine-Grained Visual Categorisation

MIC Most Interesting Classes

SC Spot Check

PPV Positive Predictive Value

TPR True Positive Rate

T-ACC Triplet Accuracy

NABU Naturschutzbund Deutschland

Part I

THESIS

1 INTRODUCTION

1.1 MOTIVATION

Over the last decades many mechanical tasks have been automated to a certain degree by machines [4]. In recent years, the trend has shifted to automate more complex tasks, while imitating the human ability to make intuitive decisions using ML e.g. in autonomous driving [35], breast cancer classification [18] or predictive maintenance [3]. These decisions are based on semantic information, referring to the contextual meaning of data. Humans in fact are excellent at extracting semantic information out of data, they do it all the time. Humans learn to make intuitive decisions by experience and training, increasing their expertise on a topic and leading them to become experts in said topic. A well-trained radiologist for instance can detect small but potentially dangerous abnormalities on an x-ray image. Hence, the radiologist has learned to distinguish and extract important semantic information from the data encoded in the image. The medical sector is not the only area where such intuitive decisions are eminent parts of daily tasks. There are other sectors, like the industry, where extracting semantic information for decision-making is valuable.

Note: Even by reading this work semantic information is extracted

Many ML algorithms are known to need a lot of curated and labeled data to be trained. The industry often could easily provide raw data, but lacks the knowledge needed to apply it to ML [16]. This poses the challenge of acquiring relevant data and processing it to a degree where it becomes usable for ML [34]. A common way to provide this information is through labeling. For the industry, labeling is much harder than just providing the raw data. The reason for this is, that labeling in the industry often requires human experts, who are always scarce and expensive. This gives an incentive to improve the labeling process so that it can be done by non-experts or machines, with minimal expert supervision.

This work proposes a concept, namely the Data Refinery (DR), to efficiently label data by exploring and exploiting semantic information, based on Deep Metric Learning (DMeL). The goal is to reduce the labeling effort from experts and non-experts alike. The proposed concept is first applied in an industrial setting to create Feature Extractors (FE), which use Deep Computer Vision (DCV) to detect semantic information in images with the aim to then discriminate objects via clustering. Hence, this thesis provides a scheme of experts, non-experts, and machines working successfully in tandem to generate curated data sets for ML tasks.

1.2 SETTING

This section gives an overview of the legal framework of the German "Pfandsystem" (deposit-refund system) and part of the resulting industry in which the concept of this work was conceived.

Germany has a deposit-refund system called "Pfandsystem" for collecting and returning package materials, especially bottles. It is defined in the German law "Verpackungsgesetz (VerpackG)" (packaging act). The idea of the act is to reduce trash, reuse material, and, hence, to protect the environment[1]. Basically, for some glass or Polyethylene Terephthalate (PET) bottles there is a deposit ("Pfand") to be payed as a fee on top of the regular price. The fee ranges from 0.08€-0.25€ depending on the type of bottle. The consumer pays the fee with the product price and later gets the fee back when returning the bottle.

Note: This is only the basic idea of the "Pfandsystem". There is more to it and for curious minds reading the "Verpackungsgesetz" is recommended.

The "Pfandsystem" is applied to two main types of packaging: (1) "Mehrwegverpackungen" (reusable packaging). (2) "Einwegverpackungen" (single use packaging), which means the packaging is not reusable but the materials of the packing are, to a certain percent, recycled.[2]

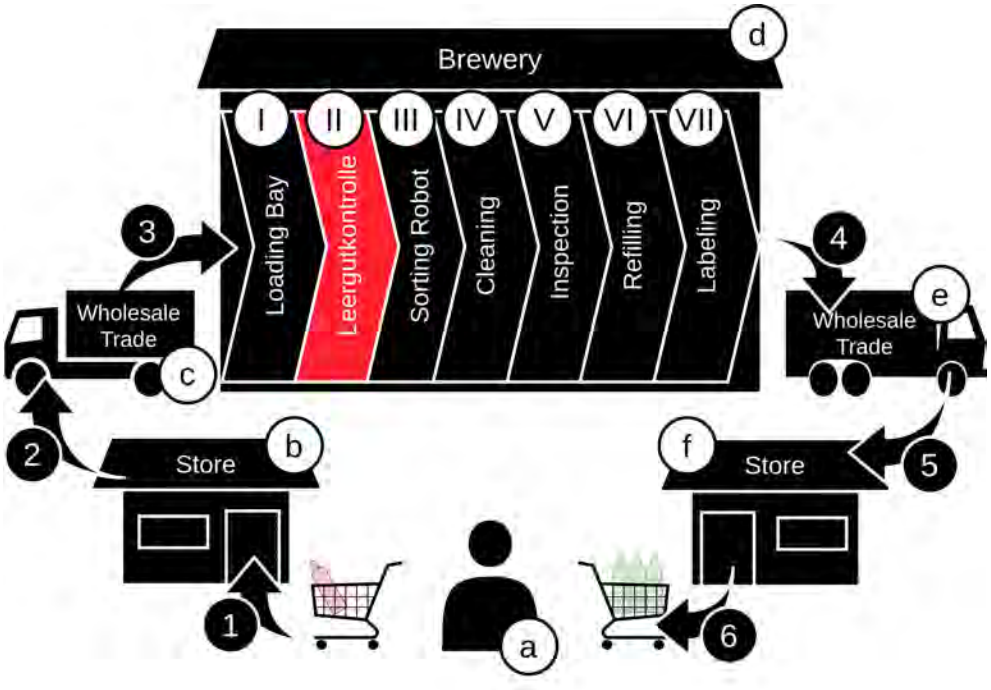


Figure 1.1: **"Mehrwegkreislauf" (reusable cycle) using the example of beer bottles.**

1 § 1 VerpackG
2 § 3 VerpackG

The concept of "Mehrwegverpackungen" comes with a logistical effort. Figure 1.1 shows the "Mehrwegkreislauf" (reusable cycle) using the example of beer bottles in more detail. ❶ The cycle starts with the customer ⓐ, returning an empty bottle to a store ⓑ. ❷ The store ⓑ passes the bottle on to a market for bottle trading ⓒ. ❸ Eventually, the bottle is delivered to a brewery ⓓ. Inside the brewery ⓓ a new cycle begins: (I) bottles are delivered and stored in the brewery. Then a (II) "Leergutkontrolle" (empties inspection) will be held, which is the process of inspecting the bottle and determining the type of the bottle. In (III) the sorting robot is located, where the bottle is sorted with the information from (II). After the sorting from (III), the bottle is cleaned (IV), inspected (V), refilled (VI), labeled (VII), etc.[3] ❹ The refilled bottle is shipped from the brewery ⓓ to a wholesale market ⓔ. ❺ At some point the refilled bottle is returned to a store ⓕ. ❻ A new customer ⓐ buys the refilled bottle from the store ⓕ and the cycle starts over again. Every part of the cycle comes with its own challenges, one of the many challenges is the correct identification of bottles, since there are many different types.

Schulz Systemtechnik GmbH[4], a German automation company, provides a solution for the "Leergutkontrolle", namely the Crate Inspector (CrI), that performs a Bottle-Type Recognition (BT-R) task. The next section will lay down the technical details of the CrI and its capabilities.

1.2.1 *Crate Inspector*

Schulz Systemtechnik GmbH has developed a solution for the BT-R task namely, the Crate Inspector (CrI). The CrI was designed by Schulz Systemtechnik GmbH, not only to recognize bottles ("Leergutkontrollen"), but also to be able to perform empty crate controls ("Leerkastenkontrollen"), full crate controls ("Vollkastenkontrollen") and crate logo controls ("Kastenlogokontrollen"). By doing so, the CrI reaches a performance of up to $6K$ crates per hour.

Note: Depending on the type of crates, a performance of up to $144K$ bottles per hour is possible.

The task of the CrI, with respect to the "Leergutkontrolle" is: *to predict the type of bottle inside a crate without unpacking the bottle from the crate*. In this work, this task will be referenced to as the Bottle-Type Recognition (BT-R) task. The type of bottle is determined by the shape, color, and size of the bottle. The paper logo on the bottle is not a determining factor. There is a widely used variety of different standard bottle shapes, like euro-bottle, NRW-bottle, longneck, and many more. Most of them have white, green or brown as standard colors and a standard volume of $0.33L$ or $0.5L$. Although the "Mehrwegkreislauf" encourages standards, some breweries favor their own designs. The brewery Beck's e.g., uses a standard white bottle with their logo as a relief on it. But there are more unique bottles then the ones

Note: Beck's relief can lead to strange things. [29]

3 Not all steps inside the brewery are shown in Figure 1.1.
4 https://www.schulz.st/

from Beck's. Breweries sometimes use colors like blue (Veltins V+ Energy) or unique volumes like 0.4*L* (Heineken). Every geographical region can have their own variation of unique bottles. This complicates the sorting task, especially when new bottle-types are released in the future.

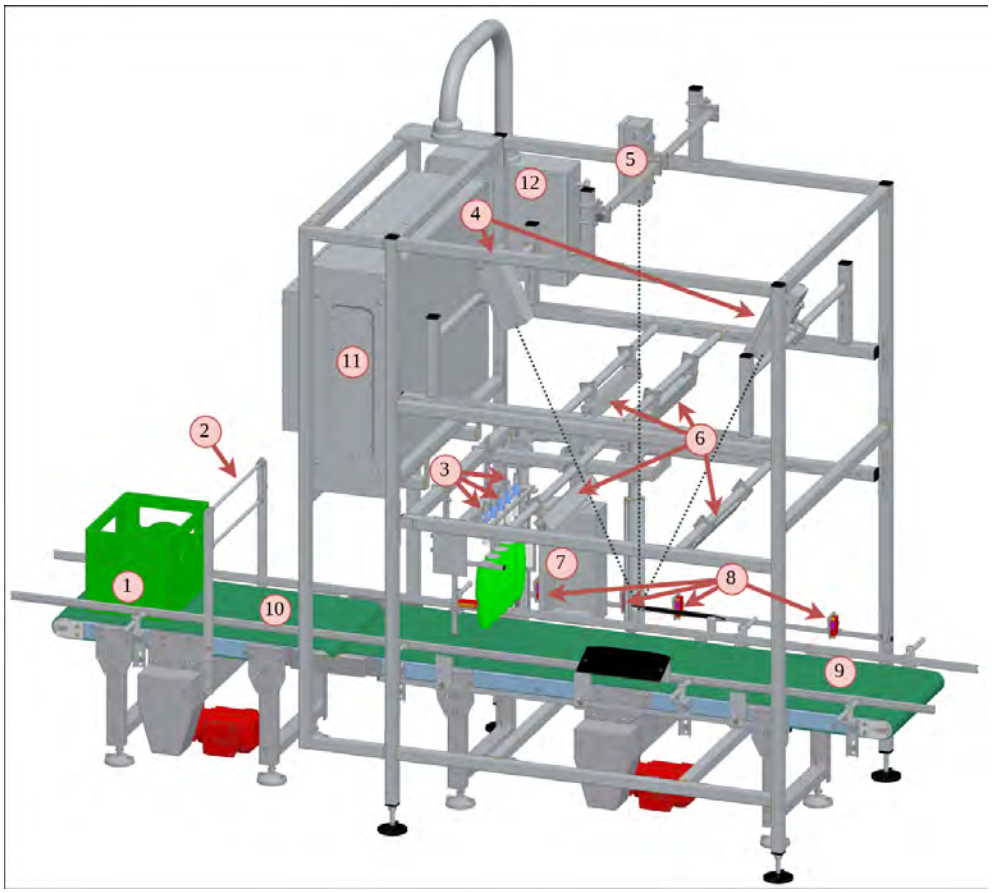


Figure 1.2: **Schematic Drawing of the Crate Inspector.** A technical Computer-Aided Design (CAD) of the CrI provided by Schulz Systemtechnik GmbH.

Figure 1.2 gives a deeper understanding of what the CrI is in technical terms and how it is constructed. In Figure 1.2 a crate (1) is transported to the CrI via the front conveyor belt (10), which manages the amount of crates transferred to the Confidence Interval (CI). A height detector (photoelectric barriers) (2) checks if something is in the crate that could damage the machine. If (2) detects no foreign object, the crate is transferred to the back conveyor belt (9). Ultrasonic sensors (3) measure the height of the individual bottles. The cameras (4) and (5) take pictures from every row of the crate. The cameras in Figure 1.2 at (4) provide pictures from the diagonal and camera (5) from the vertical perspective. All cameras take pictures, first with normal light conditions and then with Ultraviolet (UV) light. The light is provided by an illumination system (6). Additional pictures are taken from the logo of the crate by another camera (7). The photoelectric sensors located at (8) measure the length of the crate and provide additional information about the position

of the crate on the conveyor belt. All collected information is processed by a computer located at (11). The computer can display information on a screen for a operator (12).

Another way to look at the Crate Inspector (CrI) is as a system with in- and outputs as shown in Figure 1.3. Inputs to a system have different levels of importance to the given system in respect of the output. In Figure 1.3 only the most important inputs to the CrI are illustrated and Table 1.1 ranks those inputs.

INPUTS	TYPE	IMPORTANCE
Bottle-Crate	Object of Interest	***
Images via Cameras	Sensor	***
Ultrasonic Data	Sensor	***
Photoelectric Data	Sensor	*
Calibration	Config	***
Configuration	Config	**
Light	Environment	***
Temperature	Environment	**

Table 1.1: A ranking of the most important inputs for the CrI. The importance is represented by stars. The most important input is assigned three stars (***).

The main input, as shown in Figure 1.3 to the CrI is the crate (orange arrow on the left). The crate can contain up to 24 bottles. Each bottle in the crate can potentially be any bottle-type. The bottles can also have all kinds of conditions: polluted, moldy, broken, etc. But usually most of them are intact. The CrI uses the sensors (camera, ultrasonic, photoelectric) to observe the crate and the bottles inside (blue arrows). The sensors generate information from the bottles like measurements and images. The collected information is then used to determine properties like height, color, basic shapes, etc.

Note: The bottle-type distribution is mainly dependent on the geographic region

An algorithm determines which properties are needed for a certain classification regarding the BT-R task. The algorithm (yellow box) is written in C++ and all classification rules have to be defined and written in that algorithm by a human expert. This comes with a lot of effort to (1) determine the right properties, (2) implement appropriate extraction of those properties from the available data, and (3) generalize (1) and (2) for all relevant bottle-types. But there is even more to consider: because of the nature of the algorithm, the classification is sensitive to external influences. Those influences can be changes in light, temperature, vibration, etc (red arrows on the bottom). On top of that, there is the calibration and configuration (green arrows) of the

CrI and its parts (i.e., slight changes in the camera angle or position can have immense influence on the classification). Also, the configuration is often dependent on the customer and which bottles the customer needs.

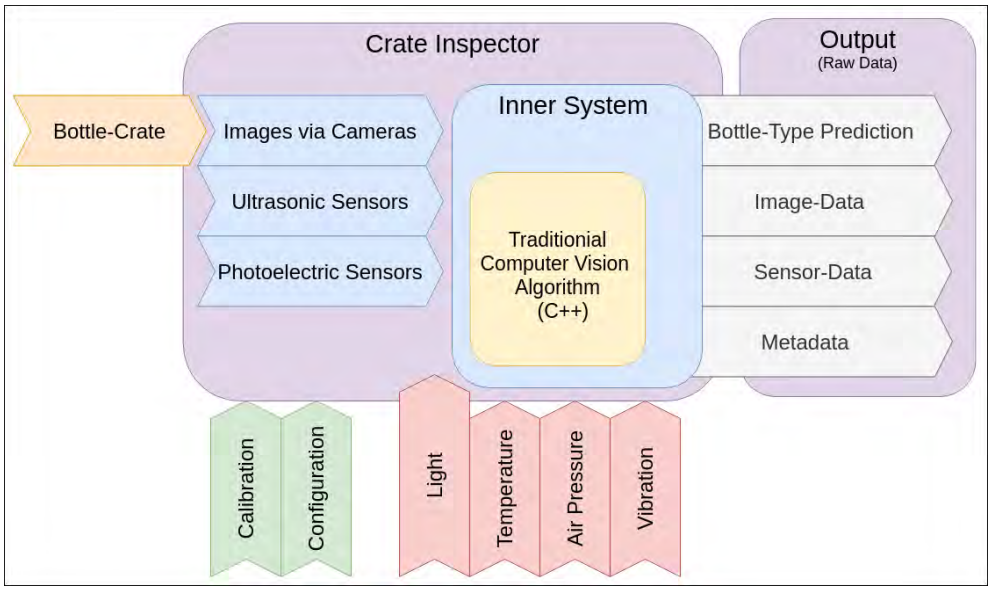


Figure 1.3: **System Analysis of the Crate Inspector.** The Crate Inspector (CrI) as a system with in- and outputs

Schulz Systemtechnik GmbH has managed these challenges and installed the CrI at over 34 customer sites worldwide. This comes with a lot of effort though and poses challenges on adapting the CrI to ever increasing customer demands. Such challenges include the sensitivity of the system to changes in the camera and lighting setup and the ever increasingly variety of different bottle brands and types. In contrast to the current software for the CrI, which heavily relies on traditional Computer Vision (CV), this work proposes that an alternative approach, utilizing ML and Deep Computer Vision (DCV) techniques has the potential to alleviate some of the challenges described above.

The Section 1.2.2 Raw Data, takes a closer look at the outputs from the CrI, which can also used as an input for a ML approach. In Section 2.2 Concept, the process of acquiring relevant data for such a ML approach is presented. In Section 2.1 Fundamentals, the necessary basic principles underlying the concept are explained.

1.2.2 *Raw Data*

The CrI described in Section 1.2.1 produces a lot of raw data which can be used for ML. This section provides a exploration of the produced raw data. Table 1.2 provides a list with the available raw data for one bottle-crate. The data is split into four types: image-data, sensor-data, metadata and given prediction of the CrI for the BT-R tasks.

FILE-NAME	TYPE	DESCRIPTION
*-Autostore-1.png	Image-Data	Configuration 1
*-Autostore-2.png	Image-Data	Configuration 2
*-Autostore-3.png	Image-Data	Configuration 3
*-Autostore-4.png	Image-Data	Configuration 4
*-Autostore-5.png	Image-Data	Configuration 5
*-Autostore-6.png	Image-Data	Configuration 6
*-Autostore-7.png	Image-Data	Configuration 7
*-Autostore-8.png	Image-Data	Configuration 8
*-Autostore.ini	Metadata	Additional Information
*-Autostore-9.csv	Sensor-Data	Ultrasonic Measurements
*-Autostore-SaveData.ini	Prediction	Classification Prediction

Table 1.2: List of Raw Data Files generated by the Crate Inspector

In Figure 1.4 all images from all the cameras and lighting settings taken by the CrI are shown. Figure 1.4a and Figure 1.4b show the crate from a vertical perspective, with two different standard settings for lighting, with Figure 1.4a slightly brighter than Figure 1.4b. In these images, the mouths of the bottles are clearly visible. For the attentive observer of both images, it is additionally recognizable that Figure 1.4b shows the crate has slightly advanced on the conveyor belt. It is also of note that the images are composed of four individual images, one for every row of the crate. This is true for all images presented in Figure 1.4 except for Figure 1.4e, which consists of just a single image, which shows the logo of the crate.

In Figure 1.4c and Figure 1.4g the crate is shown in two different diagonal perspectives and with standard lighting settings. The neck and some parts of the bottles shoulders are especially well visible here. For bottles with a paper label on the neck, this label is also visible.

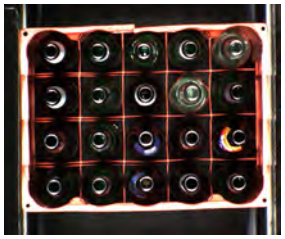
Figure 1.4d and Figure 1.4h show the crate with the two identical diagonal perspectives just as shown in Figure 1.4c and Figure 1.4g but with Ultraviolet (UV) lighting. The UV light shows properties from the bottles that were hidden before. In the upper right corner on Figure 1.4h there are two bottles glowing green. This implies the bottles have a special UV protective finish. UV protection is required for beer in white bottles. Otherwise, the beer could continue to ferment.

Another, previously hidden property can be observed on the left side in the Figure 1.4h. There are three bottles with a green glowing mark in the region of the upper neck. This mark is an identification mark applied by the

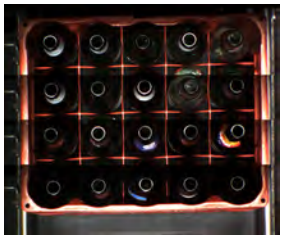
manufacturer to unambiguously identify the bottle. Although UV can reveal hidden properties on a bottle, most bottles don't have UV specific hidden properties.

Note: In Figure 1.4h, the mark for Krombacher Pilz 0.5L and Köstritzer 0.5L are visible.

There are two more images illustrated in Figure 1.4, which will not be used in this work: Figure 1.4e which, as mentioned before, shows the logo of the crate, and Figure 1.4f which shows all bottles in a "fish eye optic". The special fish eye optic helps to better identify properties on the shoulders, e.g. identify Beck's relief.



(a) Image Configuration 1



(b) Image Configuration 2

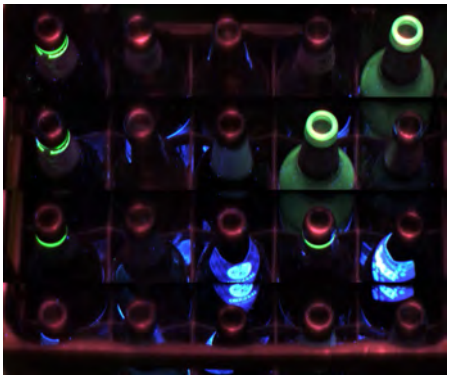
(c) Image Configuration 4

(d) Image Configuration 6

(e) Image Configuration 7

(f) Image Configuration 8

(g) Image Configuration 3



(h) Image Configuration 5

Figure 1.4: **Image-Data provided by the Crate Inspector.** All images depict the same crate in different angles. All Images were taken by the CrI.

The sensor-data from the ultrasonic sensors are used to calculate the height of the bottles in the crate. This process is error-prone and the calculation rather complicated. For the CrI the height is one of the main properties used to classify the bottles.

This Section 1.2.2 has given an overview of the available raw data created by the CrI. In the Section 3.2, this work will explain how the raw data was prepared for the proposed approach of obtaining curated data sets by applying the Data Refinery (DR). In the next Section 2.1, Fundamentals, the necessary basics for understanding the concept (Section 2.2) behind the DR are presented.

2 THEORY

In this chapter, the fundamentals (Section 2.1) which build the basis for this work, the concept of the Data Refinery (DR) (Section 2.2), and related work (Section 2.3) are introduced.

2.1 FUNDAMENTALS

2.1.1 *Machine Learning Program Paradigm*

A. L. Samuel, a pioneer on Machine Learning (ML), foresaw and explained the ML program paradigm in a paper from 1959: "Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort" [23]. Nowadays we distinguish between "traditional programming" and Machine Learning (ML). The main difference is, as depicted in Figure 2.1 that in traditional programming, a human is programming a detailed program, and in combination with input (data) an output is generated. In contrast to this, ML receives the output plus the input and a program, a function, or an algorithm is generated or "learned" by the machine.

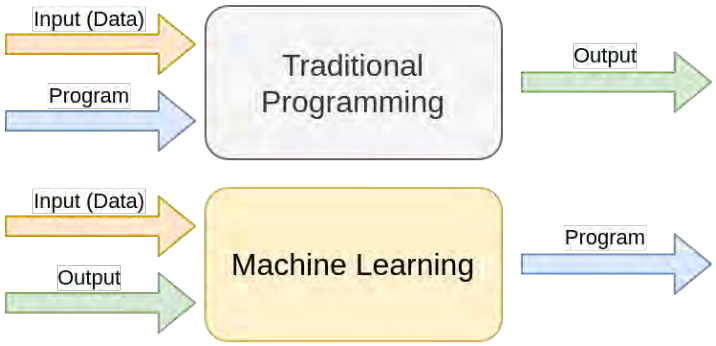


Figure 2.1: **Scheme of the Machine Learning Program Paradigm.**

In the context of this work, a human expert is not needed to explicitly program an algorithm which can recognize different bottle-types and instead has to provide a sufficient amount of input and corresponding output data (labels).

2.1.2 *Computer Vision*

The concept of Computer Vision (CV) aims to "[...] come up with a computational model of the human visual system" to automatically "[...] perform some tasks which the human visual system can perform" [22]. Larry Roberts is considered the father of CV because of his PhD thesis from 1960. Hence, CV is not new, a fairly extensive topic and yet quite difficult. One reason CV is considered difficult is that the human visual system is remarkably good at certain tasks, like face-recognition, so that a CV system can not keep up. [22]

Nowadays, CV is distinguished between traditional or analytical CV and Deep Computer Vision (DCV), which uses Machine Learning (ML). A lot of "low level" vision tasks for analytical CV like edge detection are incorporated into Deep Computer Vision (DCV) over time, which made the whole Computer Vision (CV) process easier to handle. One of the most known and successful developments of DCV are Convolutional Neural Networks (CNN).

2.1.3 *Convolutional Neural Network*

A Convolutional Neural Network (CNN) allows to automatically analyze an image with less effort than analytical CV. The architecture of the Convolutional Neural Network (CNN) consists of two main parts: a convolutional part, also called Feature Extractor (FE) and a fully connected part, also called classification part. In Figure 2.2 a simple architecture of a CNN is given.

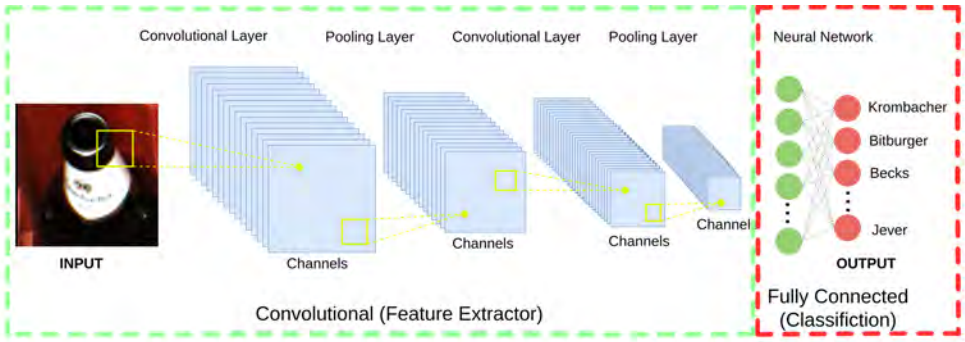


Figure 2.2: **Simple Convolutional Neural Network Architecture.** Simplified representation of the architecture of an Convolutional Neural Network.

An image fed as a input to a CNN is first passed through a convolutional layer as shown in Figure 2.2. The convolutional layer traverses the image via different filters. Every filter creates a version of the image (a channel). All channels go through a pooling layer which reduces the dimensions of the channels. This alternating between convolutional layer and pooling layer continues for a predetermined number of times. After the last pooling layer, the resulting channels are flattened (i.e., reshaped into a regular vector). This vector, sometimes also called feature embedding, is then fed to the fully

connected part, which is another basic type of Artificial Neural Network (ANN). In the end the output neurons are used to classify the image into one out of a number of predetermined classes.

2.1.4 *Open-Set*

For most classification/recognition tasks a ML algorithm is trained on data where all possible target classes are known from the beginning. This can be problematic for real world use cases however, where the task may change over time and thus not all target classes are known at training time. The Bottle-Type Recognition (BT-R) task, explained in Section 1.2.1 is once such use case. Depending on shifts in the market, such as the introduction of newer bottle-types a particular customer of the CrI may want to adjust to, may change which bottle-types need to be classified. If a standard CNN were to be trained for said customer, that later wants to recognize a new bottle-type, the CNN would have to be fitted with a new output layer with an additional neuron for the new bottle-type. This in turn requires retraining the CNN as a whole. In effect this would lead to an architecture adjustment and retraining process for every new bottle-type or other change in the task. Open-Set detection aims at dealing with this problem when not all classes are known at training time by changing the training regiment. [6].

One approach to solve this problem is to learn a Feature Space (FS). Instead of learning a discrete classification, a continuous space is learned which encodes features. The FE part of a CNN can be used to create a FS for images. Section 2.1.5 explains how to train a FS by using Metric Learning (MeL) in more detail.

To give an intuition of how a FS can be interpreted, one can use the concept of Desoxyribonucleic Acid (DNA) as an analogy. DNA encodes the information that determines the phenotype (appearance) of a creature. The information that determines the color of the iris (eyes) is not encoded in one area of the DNA, but rather on different parts. The combination of these features determines the final phenotype of a certain property, like the eye color [33]. A property like color, which can be interpreted by a human, is encoded in the FS in a similar way it is encoded in DNA, where the combination of different features account for a certain color. It is important to note that the individual features from the FS are not interpretable by humans.

The benefit of an continuous FS versus a discrete classification is that theoretically a endless amount of classes can be encoded. Furthermore, if one wants to add a new class, retraining is not necessarily required. The downside is that for the actual classification another algorithm like K-Nearest Neighbors Algorithm (K-NN) is needed.

2.1.5 *Metric Learning*

The purpose of Metric Learning (MeL) or metric embedding learning is to learn a function $f(\cdot) : \mathbb{R}^Q \rightarrow \mathbb{R}^F$; which maps semantically similar data samples from the data space $\mathbb{R}^Q$ closer together in the target space $\mathbb{R}^F$ (Figure 2.3). The function $f(\cdot)$ can be anything ranging from a linear transformation [15, 32], to a non-linear transformation usually represented by a deep Artificial Neural Network (ANN) [26]. [9]
The result $\mathbb{R}^F$ from $f(\cdot)$ is further referenced as Feature Space (FS) and semantically similar data samples are grouped into one class c_i all classes belong to the set C.

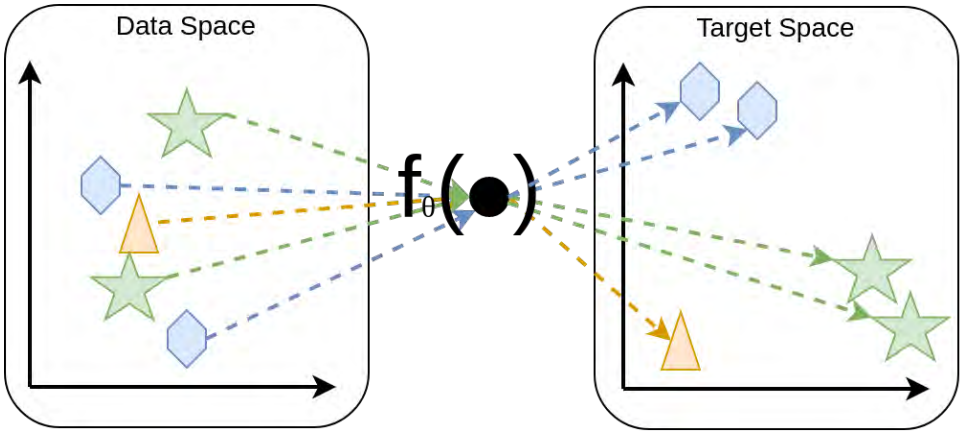


Figure 2.3: **Concept of Metric Learning.** Simplified presentation of the concept of Metric Learning.

In recent academic research Deep Metric Learning (DMeL), where $f(\cdot)$ is a deep ANN, showed great success specifically for Computer Vision (CV) tasks [12]. Especially for face recognition [26], where the number of different classes (faces) is high and the number of images per class is low. This is also known as "few-shot learning". The next Section 2.1.5.1 introduces the triplet loss, a way to learn or train the function $f(\cdot)$ for DMeL.

2.1.5.1 *Triplet Loss*

A loss function L is an essential part of Machine Learning (ML) tasks. In essence, ML tasks are optimization problems (i.e., basically, "a number" becomes either maximized or minimized). The loss resp. loss function, defines exactly "the number" to be optimized.

Let $d : \mathbb{R}^{\mathbb{F}} \times \mathbb{R}^{\mathbb{F}} \rightarrow \mathbb{R}^+$ be a function which measures the distance between two samples in the FS. For clarity the shortcut $d(x_i, x_j) = d(f(x_i), f(x_j))$ is used. Furthermore let x_a (anchor) be a sample in $\mathbb{R}^{\mathbb{F}}$, with the class $c_i \in C$, x_p (positive) a sample in $\mathbb{R}^{\mathbb{F}}$ which also has the class c_i, and x_n (negative) a sample in $\mathbb{R}^{\mathbb{F}}$ which has the class $c_{j,j\neq i} \in C$ (i.e., a different class then the class from x_a and x_p). [9] Now a loss function is needed $L : \mathbb{R}^{\mathbb{F}} \times \mathbb{R}^{\mathbb{F}} \times \mathbb{R}^{\mathbb{F}} \rightarrow [0, \infty)$ that: (1) yields a big number if $d(x_a, x_p) \gg d(x_a, x_n)$, (2) a low number

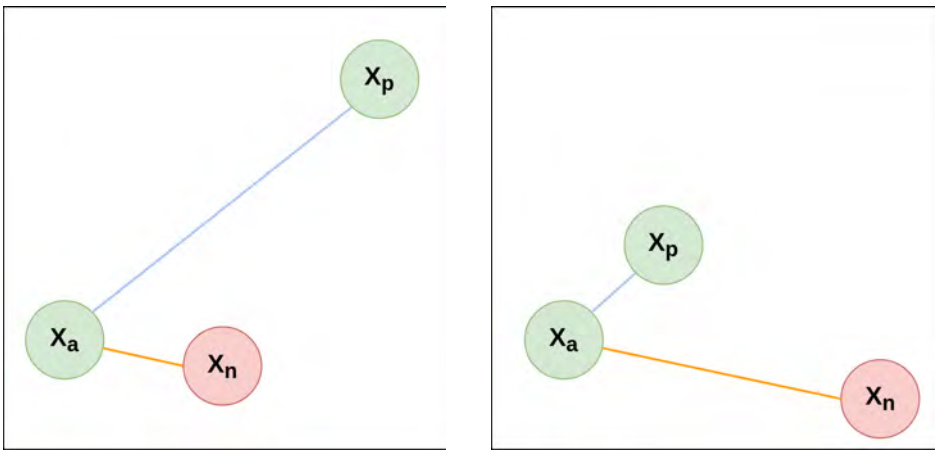
if $d(x_a, x_p) > d(x_a, x_n)$, and (3) 0 if $d(x_a, x_p) \leq d(x_a, x_n)$. Equation 2.1 defines such a function; the triplet loss.

$$\mathsf{L}(x_a, x_p, x_n) = max(0, d(x_a, x_p) - d(x_a, x_n)) \tag{2.1}$$

One extension of the triplet loss function is the margin parameter m. It is added to the $d(x_a, x_p)$, so that in some cases, even if $d(x_a, x_p) < d(x_a, x_n)$ the triplet loss yields a value > 0. The triplet margin loss is shown in Equation 2.2 and is applied in this work.

$$\mathsf{L}(x_a, x_p, x_n) = max(0, m + d(x_a, x_p) - d(x_a, x_n)) \tag{2.2}$$

Over the learning period, the ML task tries to minimize the output of the Equation 2.2, in order to learn an optimal function $f(\cdot)$. This is successful when $d(x_a, x_p)$ is minimized and $d(x_a, x_n)$ is maximized (i.e., x_a and x_p are mapped close together and x_n further away from x_a in the FS ($\mathbb{R}^{\mathbb{F}}$)). A visual intuition of this is given in Figure 2.4.

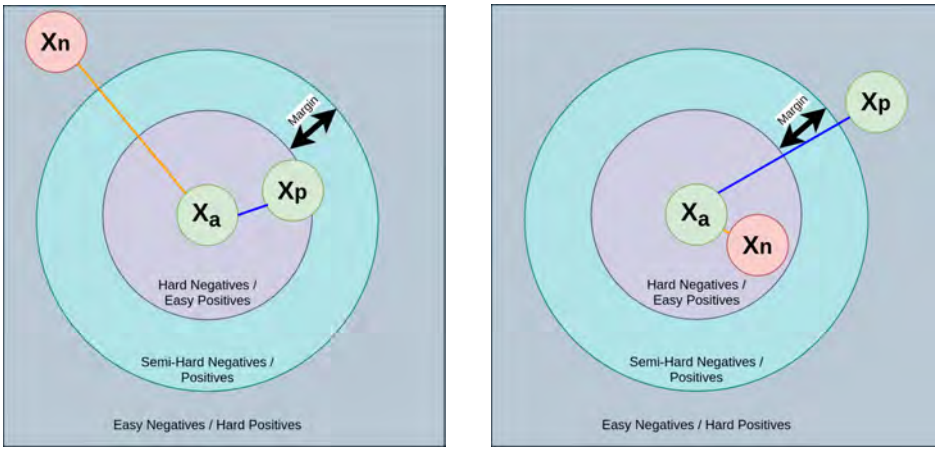


(a) Before Training (b) After Training

Figure 2.4: **Feature Space and Triplets.** In 2.4a the distance $d(x_a, x_p)$ ( ) between points in the same class ( ) is higher, than the distance $d(x_a, x_n)$ ( ) from different classes (and). After learning the metric embedding from the data; the $d(x_a, x_p)$ is reduced and the $d(x_a, x_n)$ is increased. If the training is successful, $d(x_a, x_p) < d(x_a, x_n)$, as shown in 2.4b.

It is important to highlight the following: a ML model in a training phase learns more if the triplet loss (L) is high. Over the course of training, the L should converge towards 0. But all this is dependent on which triplets were build from the data samples. The number of all possible triples for a data set with n samples is $\mathcal{O}(n^3)$. Using all triplets for a large n is impracitcal for ML model training [5]. Hence the idea is to build mostly so called "hard" or "semi-hard" triplets (i.e., triplets with $L \gg 0$ or $L > 0$).

The Figure 2.5 shows an example of hard, semi-hard and easy triplets. In Figure 2.5a, a easy triplet is shown. The negative sample x_n is further away



(a) Example of one easy Triplet where $L = 0$

(b) Example of one hard Triplets where $L \gg 0$

Figure 2.5: **Example of hard, semi-hard and easy Triplets.** 2.5a shows a easy triplet, with no new information to learn. In 2.5b a hard triplet is shown, which has new information to learn. For an optimal function $f(\cdot)$; the sample x_p has to be mapped closer to x_a and x_n further away from x_a. This figure is inspired by [12].

to the anchor x_a than the positive sample x_p. The triplet loss would yield $L = 0$ for such an example (i.e., nothing can be learned). Figure 2.5b shows an example for a hard triplet. The negative sample x_n is closer to the anchor sample x_a than the positive sample x_p. The triplet loss yields $L \gg 0$. Hard triplets are valuable for training because the impact on learning is high.

2.1.6 *Human-in-the-Loop*

Human-in-the-Loop (HITL) describes the process of utilizing humans high-level cognitive power to tackle particularly difficult tasks a computer system can't solve on its own [20]. This is often used for difficult visual object recognition tasks [11, 20] and will also be applied in this work.

To give an example of how such a process can work, one can examine Hutchsion David: *Visual Recognition with Humans in the Loop*. In his own words: "We present an interactive, hybrid human-computer method for object classification. The method applies to classes of objects that are recognizable by people with appropriate expertise (e.g., animal species or airplane model), but not (in general) by people without such expertise. It can be seen as a visual version of the 20 questions game, where questions based on simple visual attributes are posed interactively. The goal is to identify the true class while minimizing the number of questions asked, using the visual content of the image." [11]

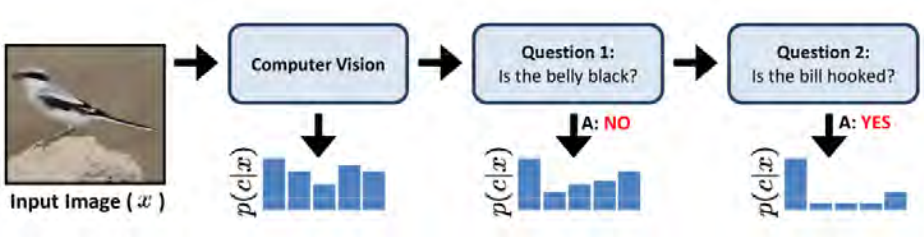


Figure 2.6: **Visualization of a Human-in-the-Loop System.** "Visualization of the basic algorithm flow. The system poses questions to the user, which along with computer vision, incrementally refine the probability distribution over classes" [11]. (Figure taken from [11])

The Figure 2.6 (out of [11]) visualizes the process; first a CV system analyzes the image of a bird, to calculate a probability distribution over the bird classes. Then the system asks specific questions a human will then give answers to. Every answer refines the probability distribution over the bird classes. Thus a computer system and a human solve a task together.

2.1.7 *Expected Utility*

When using a HITL approach, decisions have to be made on which questions are presented to a human worker. It is often not possible to have a human worker answer every question due to the sheer amount of potential questions. This introduces a decision-making problem that has to be resolved somehow. This work uses the concept of Expected Utility (EU) for evaluating decisions under uncertainty. It can incorporate preferences over decisions and originates in economics where it is used to model human decision-making.

The concept of expected utility is best illustrated by example. Suppose there is a lottery with a probability (p) of 50% to win 100€ and 50% to win 0€. In the context of EU the "win" is called outcome (o) (i.e., 50% probability for an outcome of 100€ and so on). Furthermore there is a second lottery with a probability of 50% to yield an outcome of 50€ and 50% probability to yield an outcome of 50€. Lottery one is called X and lottery two X', which lottery is better?

$$\mathsf{X} = \langle o, p \rangle = \left\langle \begin{bmatrix} 100€ \\ 0€ \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \right\rangle \tag{2.3}$$

$$\mathsf{X}' = \langle o, p \rangle = \left\langle \begin{bmatrix} 50€ \\ 50€ \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \right\rangle \tag{2.4}$$

The expected value E is calculated as a weighted sum in Equation 2.5 and Equation 2.6, for both lotteries E is the same: 50€.

$$\mathsf{E}(\mathsf{X}) = \sum_i p_i * o_i = 0.5 * 100€ + 0.5 * 0€ = 50€ \tag{2.5}$$

$$\mathsf{E}(\mathsf{X}') = \sum_i p_i * o_i = 0.5 * 50€ + 0.5 * 50€ = 50€ \tag{2.6}$$

But still there are some people who will prefer lottery X' over X, because it is a safe bet to gain 50€. Those people can be called "risk averse", they will always prefer the safe option. Some other people will prefer lottery X over X', because they like the thrill of gambling. Those people can be called "risk seeking".[1] EU provides a way, in form of a utility function U, to incorporate those preferences; risk averse, risk seeking and so on. For this the outcomes are not directly used in the weighted sum, but are re-evaluated. To calculate an example of a risk averse preference, the square root can be used as utility function U to re-evaluate the outcomes. Equation 2.7 calculates this for X and Equation 2.8 for X'.

$$\mathsf{EU}(\mathsf{X}) = \sum_i p_i * U(o_i) = 0.5 * \sqrt{100€} + 0.5 * \sqrt{0€} = 5 \tag{2.7}$$

$$\mathsf{EU}(\mathsf{X}') = \sum_i p_i * U(o_i) = 0.5 * \sqrt{50€} + 0.5 * \sqrt{50€} \approx 7.07 \tag{2.8}$$

Now $EU(X)$ yields a utility of 5 and $EU(X')$ a utility of 7.07. With this, lottery X' has a higher utility then X and thus lottery X' is the natural choice for a risk averse person.

1 There are also people who are "risk neutral" which see lottery X and X' as equal.

It is important to note that the outcome is no longer a value in euro, but in utility and that the utility function U defines how something is valued. There are various commonly used utility functions, able to capture all kinds of preferences. Equation 2.9 shows a more general definition of EU.

$$\mathsf{EU} = \sum_i p_i * U(o_i) \tag{2.9}$$

This section just gave a simple introduction to the EU suitable for its use in this work. There's a lot more literature [2, 25] which gives a more complete picture of this topic for the interested reader.

2.1.8 *Multi-Class Classification Validation*

To validate the performance of a ML model, the confusion matrix, also known as error-matrix is often used [28]. Table 2.1 shows the confusion matrix for a binary classification task. Assumed a model is trained for a classification task, which can detect if a cat is depicted in a picture or not. The amount of times the model predicts a cat is in a picture and this is actually true is called True Positive (TP). The amount of times the model predicts there is no cat in a picture and this is actually true is called True Negative (TN). The amount of times the model predicts a cat is depicted in a picture, but in truth, there is no cat in the picture is called False Positive (FP). FP is also known as Type I error. The amount of times the model predicts there is no cat in the picture, but in truth, a cat is in the picture is called False Negative (FN). FN is also known as Type II error.

		True Condition	
		Condition Positive	Condition Negative
Predicted Condition	Predicted Condition Positive	True Positive (TP)	False Positive (FP) Type I Error
	Predicted Condition Negative	False Negative (FN) Type II Error	True Negative (TN)

Table 2.1: The Confusion Matrix for a Binary Classification Task.

To validate the cat-model, TP, TN, FP and FN are calculated on a test set to form the confusion matrix (i.e., on pictures the model has not seen before). The confusion matrix functions as a base for all kind of validation metrics. One such metric is accuracy as given in the Equation 2.10.

$$\mathsf{ACC} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.10}$$

Accuracy is a good metric if the test set is balanced but it cannot handle in-balanced data. Assumed 90% of pictures in the test set are pictures of cats, a model which just predicts that there is a cat in every picture in the test set, would reach 90% accuracy. Better metrics for in-balanced data are precision and recall.

Precision is the ratio between TP and all the times the model predicted a cat on a picture (TP+FP). Precision is also called Positive Predictive Value (PPV).

$$\mathsf{PPV} = \frac{TP}{TP + FP} \tag{2.11}$$

Recall, also called sensitivity or True Positive Rate (TPR), is the ratio between TP and all the times a cat actually was in a picture (TP+FN).

$$\mathsf{TPR} = \frac{TP}{TP + FN} \tag{2.12}$$

For multi-class classification validation, there are two types of precision and recall; micro and macro averaging [28]. Micro averaging is calculated over the total test set and all test samples are treated equally. Supposed a classification model, which classifies in three classes[2]; cat (c), dog (d) and mouse (m), is used. The Table 2.2 shows the prediction of a given test set for such a classification model.

True Condition	c	d	m	d	m	m	c	d	d
Predicted Condition	d	d	c	d	c	m	d	m	d

Table 2.2: Predicted Condition for the Multi-Class Classification Example which Classifies in Cat (c), Dog (d), and Mouse (m).

TP are all true predictions the model has given; TP= 4. FP are all the times a certain animal is predicted, but in truth, there is a different animal; FP= 5. FN are all the times an animal should have been predicted, but was not. In column one, a cat should have been predicted, but was not. In column three, a mouse should have been predicted, but was not and so on. Hence the amount of FN are also 5. In other words, there is always the same amount of FN and FP which leads to Micro Precision (MiP) and Micro Recall (MiR) being equal for micro averaging.

Macro averaging calculates precision and recall for each class and then averages over all classes. This is illustrated for the example in Table 2.3, macro averaging weighs every class equally, independent of the amounts of samples per class.

The right metric to validate a model is always dependent on the use case. Often one metric is not enough to fully capture the performance of a model.

2 This example is inspired from a blog post created by Simon Hessner. [10]

Class	TP	FN	FP	PPV	TPR
Cat (c)	0	2	2	0	0
Dog (d)	3	1	2	$\frac{3}{5}$	$\frac{3}{4}$
Mouse (m)	1	2	1	$\frac{1}{2}$	$\frac{1}{3}$
Macro Average				0.3667	0.3611

Table 2.3: The Confusion Matrix, Macro Precision and Recall for all Example Classes; Cat (c), Dog (d) and Mouse (m).

In [28], comprehensive analysis of different performance measurements is executed by Marina Sokolova.

2.2 CONCEPT

In the Section 1.1 two main problems in the industrial setting were identified. (1) The industry has a lot of untidy data (Section 2.2.1), that can not be easily used with state of the art ML algorithms. (2) Acquiring semantic information is valuable for a lot of applications, but is hard to extract for a machine. In Section 1.2 one such application; namely the Bottle-Type Recognition (BT-R) task for the German "Pfandsystem", is described. The problem with the approach described in the mentioned section is, that the semantic information and how the machine should interpret them, is given by humans. This leads to an increasingly complex program, that the human must keep in order. This work proposes a novel concept, in which a machine learns to extract semantic information and while doing so, is also capable of refining the untidy data provided by the industry. This is not achieved without human expertise, but instead with a decreased effort, thus simplifying an quickening the process of acquiring relevant data (Figure 2.7).

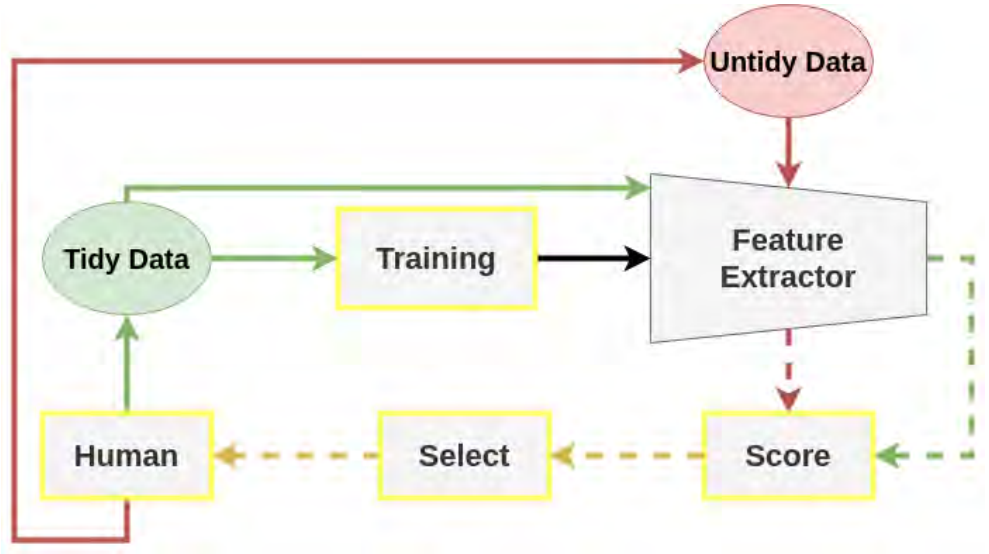


Figure 2.7: **Schematic Concept of the Data Refinery.** Tidy data is used to train a Feature Extractor (FE). The FE is used to find interesting untidy data samples to show to a human. The human gives feedback regarding the shown samples. As a result, a FE can be trained with more tidy data in the next iteration.

In order to optimize the process and to reduce the effort required from the human expert, a FE is trained with tidy data, as shown in Figure 2.7. The trained FE maps data samples with similar features in a Feature Space (FS) close to each other. With the help of the provided tidy data, the FE is capable of finding interesting untidy data samples, which are worth showing to a human. A score function (Section 2.2.3) determines how interesting a data sample is, the higher the score, the more interesting. A select function (Section 2.2.4) is used as a filter to control the score. The goal of this process is to simplify the task of the human, which is to label only the interesting data,

tidying it in the process. So that in the next iteration, a FE can be trained with more tidy data. The whole process can also be performed with no tidy data provided in the first iteration. For this a FE starts with a random initialisation. In every step errors can be made so the proper implementation of all steps determine the performance of the DR.

2.2.1 *Tidy and Untidy Data*

The DR distinguishes between tidy and untidy data. Tidy data in the context of the DR is data a human has examined and given semantic information to (i.e., labeled the data accordingly). This can also include additional information. For example: that a data sample does not belong to a specific class, or that it looks similar to data examples from other classes. Untidy data is all the data that was not examined by a human and so the properties are unknown. The untidy data is explored, while the tidy data is exploited. In this work tidy data is sometimes referred to as labeled or known data, while untidy data is referred to as unlabeled or unknown data.

2.2.2 *Training Process*

The training process uses the tidy data to create an FE. This FE aims to capture the essence of a data sample in a feature embedding. For image-data advances in CNNs like AlexNet [14], VGGNet [27] GoogLeNet-Inception [30] and ResNet [7] produced excellent FEs for images. In Section 3.3 an example for such a training process and the corresponding FE is given.

2.2.3 *Score*

The idea behind the score is to know in advance which untidy data samples are valuable for training in the next iteration. Valuable implies how much can be learned from a untidy data sample regarding the training process. For training a loss function is used, and in general the higher the loss, the more can be learned. The score aims to simulate the expected loss for a given untidy data sample. This is not easy, because some assumptions about the untidy data sample have to be made. In order to make these assumptions, the feature space of the tidy data can be used. One way to exploit the feature space is to put the feature embedding of the untidy data sample in the feature space and look at the neighborhood. In Section 3.5 one possible implementation of such a score is given.

2.2.4 *Selection*

The selection step is responsible for the selection of the untidy data samples shown to a human. The most straightforward way could be by selecting untidy data samples with the highest score. This is not always the best way.

For example: there could be a class, one is more interested in compared to other classes. As a result untidy data samples which could belong to this class should be selected predominantly. Another possible way to select is to restrict how often a sample is chosen that could belong to a certain class per iteration (i.e., repetitively asking about the same data is not efficient). Weighing untidy data samples by occurrence over iterations by predicted properties can also be beneficial. In general, selecting a portion with a selection strategy and the rest randomly can account for unwanted biases.

2.2.5 *Human*

The human in this process has the job to give feedback to the machine and by doing so, guiding the training process. The task the machine provides for a human should be as easy as possible (e.g. a simple binary question if an untidy data sample belongs to a class or not). The simpler the task, the risk of error-prone answers is reduced and the human does not necessarily need to be an expert. An example for the implementation of such a task is given in Section 3.4.

In Chapter 3 the concept of the iterative Data Refinery (DR) process is applied to the Bottle-Marketing-Label Recognition (BML-R) task which is similar to the Bottle-Type Recognition (BT-R) task from Section 1.2.

2.3 RELATED WORK

In the aspect of Deep Metric Learning (DMeL) and Human-in-the-Loop (HITL), Yin Cui's et. al. work "Fine gradient categorization and data set bootstrapping using deep metric learning with humans in the loop" fits best to the approach followed in this work. Yin Cui's work aimed to tackle the three challenges imposed on Fine-Grained Visual Categorisation (FGVC) by using bootstrapping and a generic iterative framework for FGVC. Similar to this work, Yin Cui extended a data set (bootstraps) with the help of human workers. A data set of flowers was obtained from Instagram. Then a model was trained via DMeL using triplet loss. In every round new images with a high confidence score on their model were sent to humans for labeling.

The main differences to this work lied in the way the images were chosen to be shown to a human and in the training process. Yin Cui's confidence score indicated a probability that an image belonged to a certain class. All data samples with a confidence score higher than a threshold of 50% were shown to a human. The human labeler answered a binary question: if the prediction was true (true positive) or not (false positive). The false positives were used as hard negatives for triplet generation. In contrast this work showed data samples to a human which indicated a possible high triplet loss for the next iteration as explained in the previous section (Section 2.2). Another difference was that the data sets Cui used were a self mined set originating from Instagram and a benchmark data set for FGVC, whereas this work used a data set obtained from the industry.

Yin Cui demonstrated that a random triplet generation strategy (triplet-naive) performed poorly in comparison to a triplet sampling strategy with hard negatives. Furthermore, incorporating the hard negatives found by humans in training boosted the performance of a model by 3.5%. An additional performance boost of 3.5% could be observed when the new data samples which were labeled as true positives by humans were added to the training process.

3

METHODOLOGY

This section presents a case study to evaluate the Data Refinery (DR) mentioned in Section 2.2 on a similar Bottle-Type Recognition (BT-R) task as described in Section 1.2; the Bottle-Marketing-Label Recognition (BML-R) task. The one essential difference to the BT-R task from the mentioned section was, that the classes for the recognition were enlarged so that not only the bottle-type, but also the bottle-marketing-label, could be recognized. With the bottle-marketing-label this work refers to the paper label a bottle has in a grocery store (i.e.; "Beck's Gold no Alk.", "Bitburger Radler", "Bitburger Pils no Alk." and so on). The following sections describe the realized case study and the implementation of the DR and its elements.

3.1 CASE STUDY

The case study aimed to show that a DR á la Section 2.2, in combination with a corresponding score and selection step could efficiently mine untidy data for the given BML-R task. For this purpose two DRs were implemented which only differ in their selection step. One DR selected untidy data samples to show to a human based on a score and the other randomly without considering the score at all. In this work the DR, which used a score sampling strategy is referred to as Data Refinery Score Sampling (DR-S) and the DR which used a random sampling strategy is referred to as Data Refinery Random Sampling (DR-R).

Both DRs consisted of 10 iterations. The iterations $1 - 9$ passed through all DR steps; training, score, select, and human. The iteration 10 passed through an additional training step to see the effects of the last labels provided by the human step from iteration 9. For the purpose of evaluating the performance of DR-S and DR-R, a simple K-NN based classifier was used on the Feature Space (FS) of the Feature Extractors (FE), trained by the DRs for each iteration. A detailed explanation of the performance evaluation is given by Section 3.7. Section 3.2 Data Preparation, Section 3.3.1 Feature Extractor, Section 3.3 Feature Extractor Training, Section 3.4 Human Task, Section 3.5 Score Calculation, and Section 3.6 Selection explaining the implementation details of the different DR elements for this case study.

3.2 DATA PREPARATION

With the ML program paradigm the main work shifted from writing a program, to preparing data for the ML process. Therefore this section provides

an overview of the data preparation process.

Initially, the raw data described in Section 1.2.2 were indexed; every crate and bottle was given a Universally Unique Identifier (UUID). The Index also included meta-information from each bottle, most importantly the height of the bottle. Afterwards, the bottles in the raw images given by the CrI were cropped out. This was done for the image configurations 1-6.

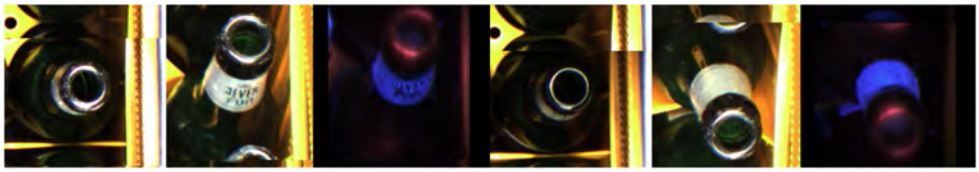


Figure 3.1: **Example of a Data Sample.** Example for a data sample without the height. A sample consisted of 6 different images of one bottle. Each image of the bottle had the dimensions 3x260x260 and was cropped from the image configurations 1-6 (Figure 1.4).

The 6 images from one bottle together with the height from that bottle created a sample, illustrated in Figure 3.1. In total 183084 samples were created in this way from data provided by a CrI, located in a brewery in Germany. The raw data needed for this was obtained in approximately $2h$ by the CrI. Subsequently to the creation of the samples, a core data set was obtained from the samples.

3.2.1 *Core Data Set*

The result of the data preparation process was the "core data set". The core data set provided mainly two types of information (1) a list of bottle samples with their corresponding labels "true labels". (2) A list of bottle samples with explicit negative examples for a corresponding class "explicit negatives". The true labels were later used as anchors or positives and the explicit negatives were used as hard negatives for Metric Learning (MeL) with triplets.

Previously the bottle-marketing-label was mostly ignored by the CrI, now the bottle-marketing-label determines the given true label for a data sample. The process of obtaining the core data set took about three weeks and domain knowledge was needed. Utmost care was taken, that the bottles picked for a class, captured the variants of the class. In Figure 3.2 an example of a "diverse" class is given. To accelerate the entire process, meta-information provided from the CrI and a pre-trained ResNet 18 were used to obtain the core data set. Additionally a web-app was created to easily explore the raw data.

Figure 3.2: **Example of a "Diverse" Class.** Each yellow square shows a bottle with image configuration 3. All bottles belong to the same target class. Important for all target classes was a certain diversity (i.e., all bottles from one target class have to be in different crates, in different positions, have a different rotation, and so on). The bottles from the target class shown in the figure fulfill this request.

- **True Labels**
  As a result, 4294 true label samples from 140 target classes were obtained. This was about 2.4% of all samples. The mean of samples in all target classes was 31, with a high standard deviation of 24. Figure 3.3 shows the distribution of true label samples per target class.

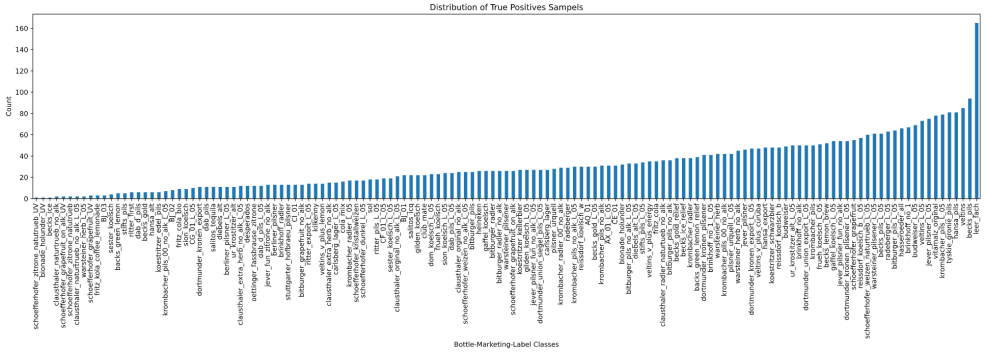


Figure 3.3: **The Class Distribution of the True Labels.** A enlarged version of this figure is included in the Appendix D (Figure D.1).

- **Explicit Negatives**
  5308 explicit negative samples from 88 target classes were obtained. The mean for samples in all target classes was 62, with an even higher standard deviation of 45. Figure 3.4 shows the distribution of explicit negative samples per target class. The reason for the discrepancy of target classes from explicit negative samples and true label samples lied in the fact that subclasses were introduced after the process of obtaining the core data set was completed. For training, the true label target classes for one class, could also be used as a negative for all other classes.

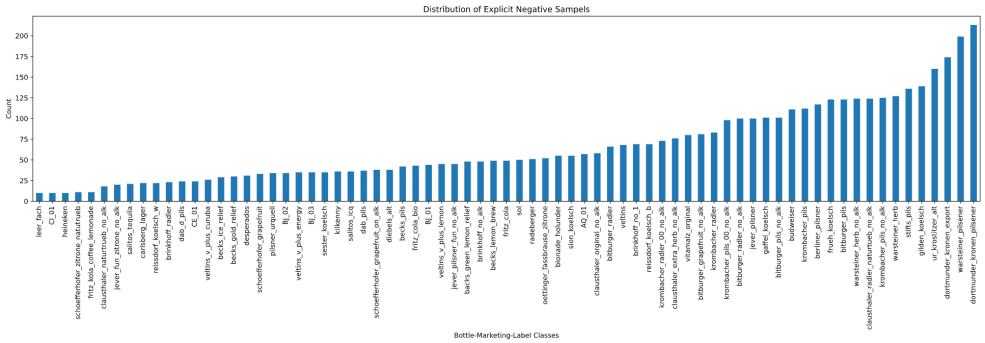


Figure 3.4: **The Class Distribution of the Explicit Negative.** A enlarged version of this figure is included in the Appendix D (Figure D.2).

3.3 FEATURE EXTRACTOR

As mentioned in Section 2.2, FEs were trained in every iteration with the tidy data from the DR process. For the presented case study the first iteration of DR-R and DR-S were trained with the same sub-set of the core data set. After the first iteration, the training set for DR-R and DR-S distinguished themselves from each other due to the different selection strategies. The architecture of all trained FEs and the overall training process were identical for DR-R and DR-S. In this section the architecture of the FEs and the training process with its parameters is explained in the context of the case study.

3.3.1 *Feature Extractor Architecture*

In ML-driven Deep Computer Vision (DCV), Convolutional Neural Networks (CNN) are popular Feature Extractors (FE) for image processing. Hence, the FE used was in essence a CNN. The architecture of the FE is given in Figure 3.5. The image data of a sample was fed to three ResNet 18 [7] which do not share any weights and were cut before their full connected part. Image configuration 1 and 2 were fed into the first ResNet, image configuration 3 and 4 were fed to the second ResNet and image configuration 5 and 6 to the third ResNet. The result of all ResNets were concatenated and the resulting vector was normalized. After that three full connected layers with a PReLU [8] as their activation function followed, which reduced the dimension of the calculated vector to 300. In the beginning of the second part of the FE, the height for a data sample was concatenated with the result vector of the first part. Then two fully connected layers and a L2 loss followed. The full connected layers encoded the height and the L2 loss was responsible that the height information was only used when necessary. In the end the vectors with and without the height information were added together to form the output vector of the FE.

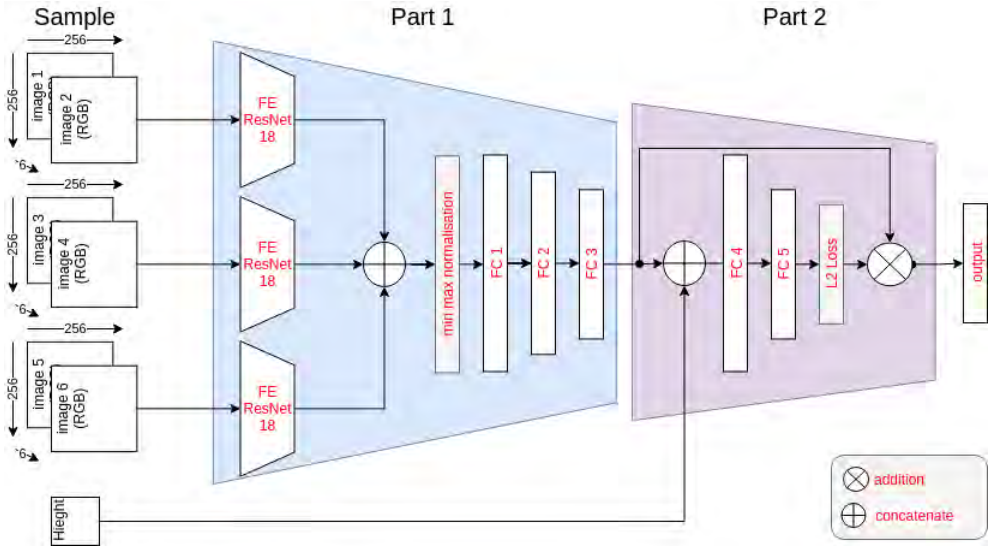


Figure 3.5: **The Architecture of the Feature Extractor.**

For this case study all three ResNets used were pre-trained and their parameters were frozen.

3.3.2 *Feature Extractor Training*

The training of a FE is based on ML and the triplet loss function. The training process tries to create a Feature Space (FS), where for every triplet the anchor and positive data sample are closer together than the anchor and the negative data sample, as described in Section 2.1.5.

Algorithm 1 describes the process necessary to train one FE, as implemented for the case study. In step 1 (line 1.6) a validation and training set was created from the core data set D.[1] After that in step 2 (line 1.7) the triplets were built. This was done for the validation and training set to build validation and training triplets. For clarity the process of building triplets is only described once. First, a class c_i out of all classes C was uniformly drawn (line 1.9). Afterwards an anchor (x_a) and positive (x_p) sample with the class c_i was randomly selected out of the true label list provided by the core data set $D_{\text{true labels}}$ (line 1.10). Next, the negative (x_n) data sample was selected. If explicit negative samples were known for the given class c_i in $D_{\text{explicit negatives}}$, a random data sample from $D_{\text{explicit negatives}}$ with class c_i was selected (line 1.14). The parameter p determined the probability with which an explicit negative sample was seeked. Else a different data sample with a class $c_j \neq c_i \in C$ was randomly drawn from $D_{\text{true labels}}$ (line 1.20). The triplet generation repeated until a maximum number of n triplets were built. Finally a FE f was trained until an early stopping criteria was eventually met (line 1.24). Early stopping periodically checks an early stopping criteria and if no improvement after a

1 In the first iteration the core data set, in the following iterations the updated core data set was used.

given amount of periods (also known as patience) is observed, the training is stopped.

For training the anchor (x_a), positive (x_p), and negative(x_n) samples were fed to the FE f to get their feature embedding. The feature embedding, $f(x_a)$, $f(x_p)$, and $f(x_n)$ were then used by the triplet loss. The resulting loss is back propagated to all layers of the FE f and their corresponding parameters are updated through stochastic gradient descent [13].

Algorithm 1 Data Refinery Feature Extractor Training Process (Simplified)

Input: D ▷ core data set with $|C|$ classes (Section 3.2.1)
Parameter: n ▷ max number of triplets
Parameter: p ▷ probability for picking a explicit negative sample
Output: f ▷ feature extractor (Section 3.3.1)
function TRAIN(D)
    Step 1 → split D in train and validation set
    Step 2 → build triplets ▷ this is done for validation and train set
    repeat
        Step 2.1 → select a random class $c_i \in C$
        Step 2.2 → select x_a, x_p from $D_{\text{true labels}}$ with class c_i
        Step 2.3 → select x_n
        if *case 1*: (pick x_n from $D_{\text{explicit negatives}}$) **then** ▷ has probability p
            if explicit negative known for c_i **then**
                Step 2.3.1 → select x_n from $D_{\text{explicit negatives}}$ with class c_i
            else
                Step 2.3.2 → go to *case 2*!
            end if
        else *case 2*: (pick x_n from $D_{\text{true labels}}$) ▷ has probability $(1 - p)$
            Step 2.3.3 → choose other class $c_j \neq c_i \in C$
            Step 2.3.4 → select x_n from $D_{\text{true labels}}$ with class c_j
        end if
    until max number (n) of triplets are build
    while early stopping criteria not met **do**
        Step 3 → train f with train triplets ▷ deep metric learning
    end while
    return f
end function

In this case study $500k$ training triplets and $3k$ validation triplets were built. The probability p to select a explicit negative sample was set to 80%. Early stopping was conducted with a patience of 8 and the early stopping criteria was the Triplet Accuracy (T-ACC), as given in Equation 3.1. T-ACC indicates the relative frequency with which an anchor is closer to its positive data sample than to its negative data sample for all triplets in the validation set. The T-ACC was calculated and compared after every $10K$ triplets trained. The training process was done for 16 FEs in every iteration for DR-R and DR-S. Each iteration new FEs were trained. To parallelize the training Azure Ma-

chine Learning Pipeline with the HyperDrive [19] functionality was used.

$$\text{T-ACC} = \frac{1}{n}\sum_{i=0}^{n}(d(f(x_a^{(i)}), f(x_p^{(i)})) < d(f(x_a^{(i)}), f(x_n^{(i)}))) \tag{3.1}$$

3.4 HUMAN TASK

In this section, the human task or the "question" a DR was asking a human is explained in the context of the case study. The task was identical for DR-S and DR-R.

Figure 3.6: **Human Labeling Task.** An example for the human task. The "question" was: does the bottle in the middle (red box) belong to: (A) "Hansa Pils" (yellow box), (B) "Club Mate" (blue box) or neither to A nor to B? The answer was: the bottle belongs to (A) "Hansa Pils".

Figure 3.6 shows an example from a human labeling task regarding the BML-R. The image is separated into three color-coded parts. The part on the top (yellow) is marked with the letter A and the part on the bottom (blue) with the letter B. The part in the middle (red) consists of six pictures of one bottle, all taken in different angles and lighting conditions. Each part A and B consists of two bottles depicted in the same way as the bottle in the red part. A and B are example bottles for two different bottle classes (or groups) which are known. The class of the bottle in the middle is unknown. On the right hand side the measured height for the bottle in the middle and the

mean height for group A, respectively group B, is given.

The "question" the system was asking the human was: "To what group does the bottle in the middle belong to?" The human was presented with three answer options[2]: (1) the bottle belongs to group A, (2) the bottle belongs neither to group A nor to group B, or (3) the bottle belongs to group B.

The task was kept simple, so that a non-expert could execute the task. A human can quickly look at the bottles and then answer if he believed whether the bottle in the middle looks like a bottle from group A, B, or neither. More information about all possible classes would help, but was not needed to answer the task. After the human made a decision, the bottle in the middle automatically became labeled and the core data set updated. If (1) or (3) was picked as an answer, the bottle in the middle got the respective true label and the explicit negative label. For answer (2) the bottle in the middle got the label from the bottle group A and B as a explicit negative label.

For the case study a video introduction, a summary of the task and occasionally updates per mail were provided to the human workers. In total 30 workers with different experience levels executed 500 tasks per iteration for DR-R and DR-S each.[3] The answers given by the human workers were not systematically reviewed, before they were used to create new labels for the unknown data samples.

3.5 SCORE CALCULATION

As mentioned in Section 2.2, the score determined how interesting a data sample is. Interesting data samples are data samples a DR "believes" it can learn the most from. This section explains how the score in the context of the case study was implemented.

The DR uses ML to train FEs, for this a triplet loss function is used. A triplet consists of an anchor, positive and negative data sample. The most valuable triplets are those who yield a high triplet loss, namely hard or semi-hard triplets where the most can be learned, as explained in Section 2.1.5.1. The scores purpose is to find unlabeled (untidy) data samples which can be used to generate hard triplets for the next iteration. For example, data samples that belong to a certain class, but might look a bit different when compared to the tidy data known to belong to that class. Another example are data samples that might look similar to a class, but do not belong to it.

In Algorithm 2 the inner workings of the score function is illustrated. In step 1 (line 2.7) the features for all tidy samples were calculated by the FE f. F_L

2 Technically a human could also skip a task if he is unsure.
3 To organize the workers the online annotation service www.zillin.io was used.

denoted the set of all resulting feature vectors. In step 2 (line 2.8) the features for all unknown (untidy) samples were calculated by f. F_U denotes the set of all resulting feature vectors. Step 3 (line 2.9) calculated the score for every element $x \in F_u$. First the K nearest neighbors x had in F_L, were determined (line 2.11). After that, all different classes K_c in K were identified (line 2.12). If there were at least two different classes in K_c, a human task as described in Section 3.4 could be generated (line 2.13).

Algorithm 2 Data Refinery Unlabeled Data Scoring (Simplified)

```
1:  Input: U                          ▷ set of unlabeled (untidy) data samples
2:  Input: f                          ▷ feature extractor Section 3.3.1
3:  Input: L                          ▷ set of labeled (tidy) data samples D_true label
4:  Parameter: k                      ▷ number of k nearest neighbors
5:  Output: S                         ▷ list of EUs (score) for all tasks
6:  function SCORE(L, U, f)
7:      Step 1 → F_L = f(l)           ▷ calculate features for all l ∈ L
8:      Step 2 → F_U = f(u)           ▷ calculate features for all u ∈ U
9:      Step 3 → calculate Expected Utility (EU)          ▷ score
10:     for x in F_U do               ▷ for every sample x ∈ F_U
11:         Step 3.1 → K = k-NN(x, F_L, k)   ▷ k-nearest neighbors algorithm
12:         Step 3.2 → get all different classes in neighborhood K: K_c
13:         if |K_c| >= 2 then        ▷ if at least 2 classes in neighborhood K
14:             for every possible human task(x) do          ▷ Section 3.4
15:                 Step 3.1.1 → calculate expected probabilities p̂_1, p̂_2, p̂_3
16:                 Step 3.1.2 → simulate expected outcome ô_1, ô_2, ô_3
17:                 Step 3.1.3 → extend S by Σ_{i=1}^{3} p̂_i * U(ô_i)   ▷ EU
18:             end for
19:         else
20:             Step 3.x → extend S by 0                    ▷ |K_c| < 2
21:         end if
22:     end for
23:     Return S
24: end function
```

Technically there could be more than one task for an untidy sample x, but in this case study only one task per x was considered in the calculation. Only the task with the two most frequent classes in the neighborhood K were used.

Every human task was considered as a simple lottery in the context of the score. A lottery is defined as a set of outcomes (o) and their corresponding probabilities (p) (Section 2.1.7). The outcomes for the score were the different losses a FE can expect in the next iteration, depending on the different answers a human could give in the current iteration. The probabilities were how likely a certain answer was.

There were three probabilities for a given lottery (task): $p_1 = P(x$ belonged to class $K_c^{(i)})$; $p_2 = P(x$ belonged to $K_c^{(j)})$; or $p_3 = P(x$ does not belong to either $K_c^{(i)}$ or $K_c^{(j)})$. The true probabilities were not known, but could be estimated by the DR. For p_1 the relative frequency of class $K_c^{(i)}$ in the neighborhood K was the estimated probability $\hat{p}_1$. For p_2 the relative class frequency of $K_c^{(j)}$ in the neighborhood K was $\hat{p}_2$. For p_3 the combined relative class frequencies of all classes neither $K_c^{(i)}$ or $K_c^{(j)}$ in the neighborhood K was $\hat{p}_3$ (line 2.15).

For the outcomes one needed to consider the different triplet losses the DR could expect. For outcome o_1, where x was $K_c^{(i)}$, x could be used for a triplet as an anchor or positive for the class $K_c^{(i)}$ and as a negative for class $K_c^{(j)}$. For outcome o_2, where x was $K_c^{(j)}$, x could be used as a anchor or positive for class $K_c^{(j)}$ and as a negative for the class $K_c^{(i)}$. Outcome o_3, where x does not have the class $K_c^{(i)}$ or $K_c^{(j)}$, x could be used in a triplet as a negative for $K_c^{(i)}$ and $K_c^{(j)}$. To estimate the different outcomes, the different possible losses for $o_1; o_2;$ and o_3 were calculated s times, with corresponding random labeled (tidy) data samples. The average over the different simulated losses for $o_1; o_2$ and o_3 were the estimated outcomes $\hat{o}_1; \hat{o}_2;$ and $\hat{o}_3$ (line 2.16).

To calculate how valuable an unknown data sample x was for a DR: the estimated probabilities and estimated outcomes were inserted in the Expected Utility (EU) function (line 2.17).

$$\mathsf{EU} = \sum_i p_i * U(o_i) = \hat{p}_1 * U(\hat{o}_1) + \hat{p}_2 * U(\hat{o}_2) + \hat{p}_3 * U(\hat{o}_3) \tag{3.2}$$

In summary, the score considered the human task a simple lottery. To compare lotteries, the score calculated the individual EU for every lottery (task). The outcomes were the expected losses the FE could expect for the different possible answers. The probability for an answer was estimated by the neighborhood an unlabeled sample had in the Feature Space (FS) of the known samples. In the end, the human worker played the "Lotto-Fee[4]" (lotto-fairy) and announces the "winning" answer.

In the case study the parameter K for the number of nearest neighbors was 12, the number of simulations s for the estimated outcomes was 250. The number of unknown (untidy) samples was $25K$ and the utility function U was the square root in order to follow a risk averse strategy. The process of calculating a score was done for 16 FEs for DR-R and DR-S in each iteration. All parameters were equal (as described), only the $25K$ untidy samples and the tidy samples were different for DR-R and DR-S and varied in every iteration.

4 Lotto-Fee (lottery-fairy) is the German name for a person that announces the winning numbers for the state lottery in national TV.

In the next section the different selection strategies for DR-R and DR-S are explained.

3.6 SELECTION

As mentioned before, the goal of the case study was to show that a DR á la Section 2.2 in combination with a corresponding score and selection step could efficiently mine untidy data. In the last section the score step was explained, which was identical for DR-S and DR-R. This section explains how untidy data samples, which were shown to a human, were selected.

For DR-R the score was not used. DR-R picked random untidy data samples and the corresponding human tasks were created. For DR-S all scores from all 16 FEs for one iteration were normalized in the interval $[0 - 1]$. For every untidy data sample x the normalized score was summed up. DR-S used the top n of the summed up values to determine for what samples a human task should be generated. Hence, the different FEs created a quorum which samples were valued the most among all FEs.

In this case study 500 human tasks per iteration were generated for DR-R and DR-S each. There was one limitation for DR-S: it could not ask the same question more than 30 times per iteration because without this limitation DR-S would ask the same question multiple times. The reason for this was that per iteration, there was only one training period and the selection of the questions took place after that.

3.7 PERFORMANCE VALIDATION

For the validation of the experiments, a classification based on a confidence score was performed. For this a test set was created out of the core data set (Table 3.1). After the creation of the core data set 25% of samples per class were randomly picked for the test set if the total number of samples in one class was > 10. All classes with a total number of samples < 10, were not used for the test or training set. In total, the test set consisted of 920 samples and is referred to as core data test set.

NAME	N	BOTTLE-TYPE	BOTTLE-MARKETING-LABEL
Core Data Test Set	973	33	121
Core Data Training Set	3130	33	121

Table 3.1: Test and Training Data Set

3.7.1 *Classification*

The confidence score described in [5] was used as the base for the classification of the core data test set. The main difference to [5] was, that the confidence score was not performed on pre-trained anchor points based on manifolds for all classes, but on all classes observed in the K nearest neighbors for a given test sample.

Suppose N categories (classes), limited by the K nearest neighbors for a given input query sample, are identified in a feature space of a test set. The j-th neighbor for category i is represented as u_{ij}, where $i = 1, 2, \ldots, N$, $j = 1, 2, ..., K$. First, the feature embedding from the input query sample is extracted from a model $f(x)$, then the confidence score for category i is generated (Equation 3.3)

$$p_i = \frac{\sum_{j=1}^{K} e^{-\gamma \| f(x) - u_{i,j} \|}}{\sum_{N}^{l=1} \left(\sum_{j=1}^{k} e^{-\gamma \| f(x) - u_{l,j} \|} \right)} \tag{3.3}$$

The predicted label for x was the category with the highest confidence score $argmax_i\ p_i$. γ was a parameter controlling the "softness" of label assignment and closer neighbors played more significant roles in soft voting. If $\gamma \to \infty$, only the nearest neighbor was considered and the predicted label was "hard" assigned to be the same as the nearest neighbor. On the other hand, if $\gamma \to 0$, all the neighbors were considered to have the same contribution, regardless of their distances between $f(x)$. [5]

Notice that because classification took place in a high dimensional Feature Space (FS) the category selected for classification could be any collectively known information of a sample in the FS (i.e., if the color of the samples in the FS is known, the classification could yield a color prediction for a query sample). All this works, without explicit training on the category on which the prediction was performed on or with any extra effort. The only condition is; that the information for the category used for calcification was encoded in the FS.

In this work classification was based on two main categories; "bottle-marketing-label" and "bottle-type". The category bottle-marketing-label was, as mentioned, the label a bottle has in a grocery store (i.e., "Beck's Gold no Alk.", "Bitburger Radler", "Bitburger Pils no Alk.", and so on). The category bottle-type was the type of bottle mainly defined by its shape, color, and size, as mentioned in Section 1.2. The bottle-marketing-label was used for training and bottle-type is the classification category used by the Crate Inspector (CrI) to sort bottles. Categories like; "bottle-color", "bottle-size" and "bottle-height" can also be used for classification. Examples for this can be viewed in Appendix B.

4 RESULTS

The Crate Inspector (CrI) recognizes bottle-types mainly by height measurements and looking at images from bottle-crates using traditional CV. This is complicated and a lot of human effort and highly specific knowledge is needed. In the last decades, new CV methods have been invented and mainly optimized in a scientific environment and with artificial benchmark data sets. Those methods show great potential in lowering the effort of humans, to generalize and to potentially surpass the performance of traditional CV in highly specific domains. But there's one problem; applying these kinds of new methods to "real world data" is not trivial. One reason for this is that real world data is untidy. Hence the real world data has to be processed before it can be applied to new methods (i.e., like mineral oil has to be processed in a refinery before it can be used in an engine to drive a car). In Section 2.2 this work proposed a concept for such a "refinery" namely the Data Refinery (DR). Chapter 3 showed how the DR was implemented in the context of a case study, and this section described the results from this case study.

4.1 OVERALL PERFORMANCES PER ITERATION

As mentioned before, the goal of the case study was to show that the DR á la Section 2.2, in combination with the corresponding score and select step, can effectively mine untidy data for the given BML-R task. For this the Data Refinery Score Sampling (DR-S) and Data Refinery Random Sampling (DR-R) which only differ in the selection step, were implemented. Both DRs had ten iterations and for every iteration 16 different Feature Extractor (FE)s were trained. The performance measurements can be viewed in the tables contained in Appendix A where the mean and the Standard Deviation (SD) from the 16 FEs for every iteration are included.

Figure 4.1a shows the mean (solid line) and the mean $\pm$ the SD (dotted line) in every iteration (x-axis) for the Micro Precision (MiP) (y-axis) over the trained FEs for the DR-R (blue) and DR-S (red). In the first iteration the DR-R started with about 58% and the DR-S with 61%. In iteration two DR-R and DR-S dropped about $[3 - 4\%]$. DR-S fully recovered from the drop in the next iteration and DR-R recovered in iteration four. After iteration five, DR-S and DR-R had no significant in- or decreases over the following iterations.

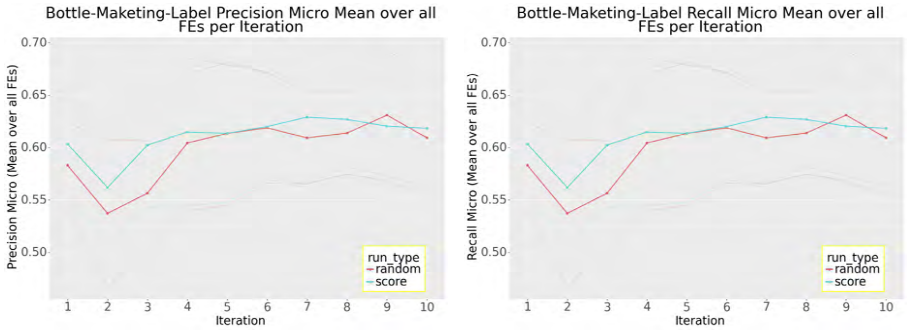
Figure 4.1b shows the mean (solid line) and the mean $\pm$ the SD (dotted line) in every iteration (x-axis) for the Micro Recall (MiR) (y-axis) over the trained FEs for the DR-R (blue) and DR-S (red). The plot is identical to the previously

described plot Figure 4.1a, because precision and recall are identical for micro averaging. This phenomena is explained in Section 2.1.8 and originates from the fact that for multi-classification micro averaging FP=FN.

Figure 4.1c shows the mean and the mean $\pm$ the SD for Macro Precision (MaP), in the same fashion as established in Figure 4.1a and Figure 4.1b. In the first iteration DR-R started at 53% and DR-S started slightly higher at about 56%. In the second iteration a decrease of $[3-4\%]$ was observed for DR-R and DR-S. The time DR-S required to recover was one iteration. DR-R needed two iterations to fully recover and surpass its previously best performance. The highest MaP score of DR-S was observed in the seventh iteration, being around 60%. DR-R dropped a bit in the same iteration and achieved its highest MaP score in iteration nine with about 58%.

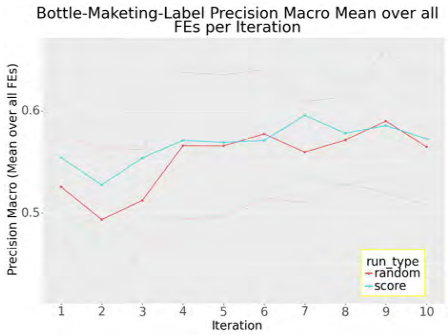
Figure 4.1d shows the Macro Recall (MaR) for DR-R and DR-S in the same manner as in the previously described plots. It showed the same characteristics for the first four iterations as Figure 4.1c, but with slightly decreased performance values. The best MaR score is now achieved by DR-R in iteration nine with 56%. For DR-S no significant changes were observed after iteration four.

In Summary: all Plots showed different performance values in the initial iteration for DR-R and DR-S. For the second iteration all plots showed a decrease in performance for DR-R and DR-S. After iteration four, respectively five, not much of an improvement of the performance of DR-R or DR-S was observed in any plot. The next section looks into the new data samples that were added in every iteration for DR-S and DR-R.

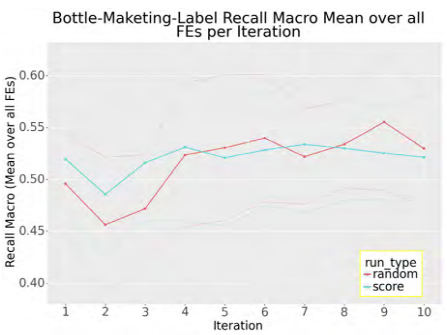


(a) Micro Precision

(b) Micro Recall



(c) Macro Precision



(d) Macro Recall

Figure 4.1: **Overall Performances per Iteration for Data Refinery Score Sampling and Data Refinery Random Sampling.** The four plots show the Micro Precision (MiP), Micro Recall (MiR), Macro Precision (MaP), and Macro Recall (MaR) for DR-S and DR-R per iteration for the BML-R task. Enlargend Figures can be viewed in Appendix D (Figure D.6, Figure D.5).

4.2 NEW DATA SAMPLES PER CLASS OVER THE ITERATIONS

In the end of every DR iteration, humans answered "questions" in form of tasks as described in Section 3.4. The answers to those "questions" were used to label previously unlabeled data. This newly labeled data was then added to the tidy data used in the next iteration to train new FEs. Over all iterations a total of about 2600 new data samples were added to true labels and about $6K$ were added to explicit negatives in the core data set for each DR. Figure 4.2 and Figure 4.3 show to which class the most fresh data samples (true labels) were added per iteration for the DR-R and DR-S.

In Figure 4.2, the y-axis shows the relative class frequency and the x-axis the different classes. The red dot represents the class frequency of iteration one, which was used to determine the order of the classes on the x-axis from lowest to highest class frequency. Hence, iteration one functioned as a baseline to show differences in class frequencies over the iterations. Every iteration is represented by a unique symbol and color (as shown in the legend).

As an example for an increase of a relative class frequency over the iteration, one can examine class "Leerfach" (empty compartment), on the far right side of Figure 4.2. In iteration one (red dot) the relative class frequency of class "Leerfach" was 0.04 for the training set of this iteration. Iteration two (brown triangle) lied above the red dot which indicated an increase in relative class frequency for "Leerfach". This observation continued and became more pronounced with every following iteration. This trend concluded in a relative class frequency of 0.053 in iteration ten (pink hexagon) for "Leerfach".

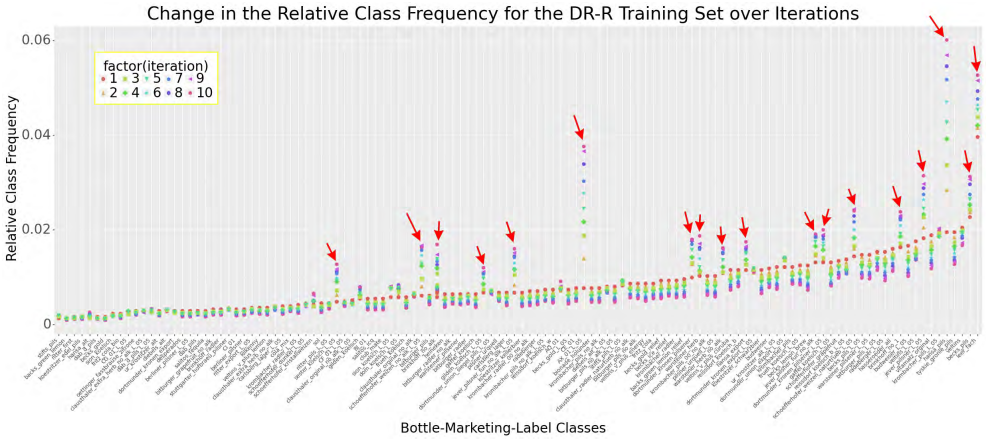


Figure 4.2: **Change in the Relative Class Frequency of Bottle-Marketing-Label Classes in the Random Training Set over Iteration.** The red arrows mark the classes that were examined in the following chapters (Most Interesting Classes (MIC)). An enlarged representation of the image is shown in Appendix D (Figure D.3)

As an example for a decrease of relative class frequency over the iteration, one can examine class "Leerfach", on the far right side of Figure 4.3. In iteration one (red dot) the relative class frequency of class "Leerfach" is, as expected, 0.04 for the training set of this iteration. Iteration two (brown triangle) lied beneath the red dot which indicated a decrease in relative class frequency for "Leerfach". This observation continued and became more pronounced with every following iteration. This trend concluded in a relative class frequency of 0.022 in iteration ten (pink hexagon) for "Leerfach".

The classes that showed significant increases in the relative class frequency over every iteration will be further referred to as Most Interesting Classes (MIC). In Figure 4.2, which shows the new data obtained by DR-R, the classes: "Sester Koelsch 0.5L", "Brinkhoff no Alk", "Schoefferhofer Weizen no Alk 0.5L", "Gilden Koelsch 0.5L", "Jever Pilsner Fun no Alk 0.5L", "Wicküler 0.5L", "Dortmunder Kronen Pilsener", "Warsteiner Herb no Alk", "Hansa Export", "Dortmunder Kronen Pilsener 0.5L", "Jever Pilsner Fun no Alk", "Schoefferhofer Weizen Naturtrueb 0.5L", "Brinkhoff No.1", "Jever Pilsner 0.5L", "Hansa Pils", "Becks Pils", "Leerfach", "Warsteiner Herb" are the MIC for DR-R. In Figure 4.3, which shows the new data obtained by DR-S, the classes: "Ritter Export 0.5L", "Ritter Pils 0.5L", "Bitburger Pils", "Krombacher

Pils no Alk", "Krombacher Radler", "Warsteiner Herb no Alk", "Hansa Export", "Dortmunder Kronen Export 0.5*L*", "Krombacher Pils", "Dortmunder Kronen Pilsener 0.5*L*", "Brinkhoff No.1", "Veltins 0.5*L*", "Hansa Pils", "Becks Pils" are the MIC for DR-S. The performance of the mentioned classes over the iterations are presented in the next section.

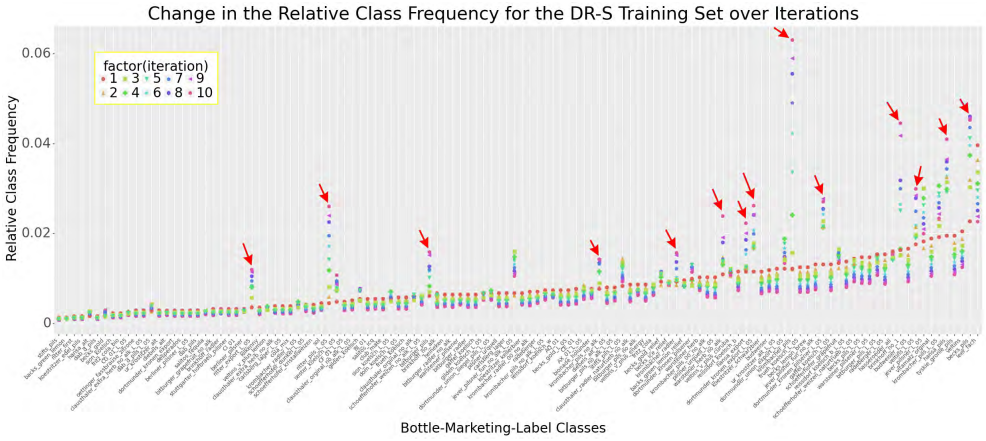


Figure 4.3: **Change in the Relative Class Frequency of Bottle-Marketing-Label Classes in Score Training Set over Iteration.** The red arrows mark the classes that were examined in the following chapters (MIC). An enlarged representation of the image is shown in Appendix D (Figure D.4)

4.3 INDIVIDUAL CLASS PERFORMANCE FOR SOME SELECTED CLASSES

This section takes a closer look at the individual performances of the different MIC for DR-R and DR-S.

Figure 4.4 depicts the precision from the 18 different classes which were the MIC for DR-R (i.e., the classes DR-R was most interested in adding new data samples to). The x-axis depicts the iterations, while the y-axis shows the mean precision over all trained FE in one iteration. The performance comparison between DR-R and DR-S regarding the MIC chosen by DR-R was presented. The left plot shows the performance for DR-R on those classes, while the right plot shows the performance of DR-S on those classes.

The left plot shows that the precision for all classes started in the first iteration within the range of $[35\% - 100\%]$. For some classes, a dip in iteration two, similar to the one seen in overall class performance (Figure 4.1) was observed. There was not much of an improvement for nearly all of the classes over the ten iterations. Except for "Warsteiner Herb no Alk" (pink), which started at 51% and ended at 61%. One class, namely "Leerfach" (dark blue), stayed at 100% over all iterations.

The right plot depicts for the majority of classes there was a slight increase in performance over all iterations. A slight decrease or no change was ob-

served in the remaining classes. The drop in iteration two was also noticeable in some classes. This was especially profound in "Jever Pilsner Fun no Alk 0.5L" (blue).

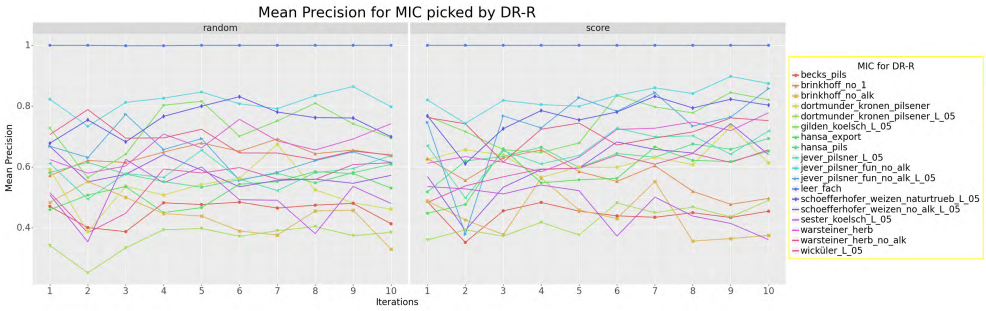


Figure 4.4: **Mean Precision for Most Interesting Classes chosen by Data Refinery Random Sampling regarding the Bottle-Marketing-Label task.** A enlarged figure is included in Appendix D (Figure D.7).

Figure 4.5 shows the precision from the 14 different classes which were the MIC for DR-S (i.e., the classes DR-S was most interested in to add new data samples to). The x-axis shows the iterations, while the y-axis depicts the mean precision over all trained FEs in one iteration. The performance comparison between DR-R and DR-S regarding the MIC chosen by DR-S is shown. The left plot shows the performance for DR-R on those classes, while the right plot shows the performance of DR-S on those classes.

The left plot displays that the performances started within the range of $[20 - 60\%]$. Compared to Figure 4.4, a lot of variation for the performance of the classes was observed between the iterations. For the majority of classes shown, there was no in- or decrease of performance noticeable, when comparing iteration one and ten. Only for a few classes there was an increase in performance, this was most noticeable for "Warsteiner Herb no Alk" (red).

The right plot shows for most classes that there was an increase in performance from iteration one to ten. Most notably were the cases of "Ritter Export 0.5L" and "Ritter Pils 0.5L" which had a high volatility in regards to performance over the ten iterations and yet ended with a significant spike in performance in its last run. From $\sim$ 24% in the first iteration to $\sim$ 58% in the final iteration. The most noticeable decrease in performance was observed in the class "Krombacher Pils no Alk" (yellow) from $\sim$ 23% in the first iteration to $\sim$ 16% in the final iteration.

In Summary, DR-R showed not much of an improvement in performance of either classes DR-R or DR-S considered MIC. DR-S on the other hand, showed an improvement in performance for MIC chosen by itself.

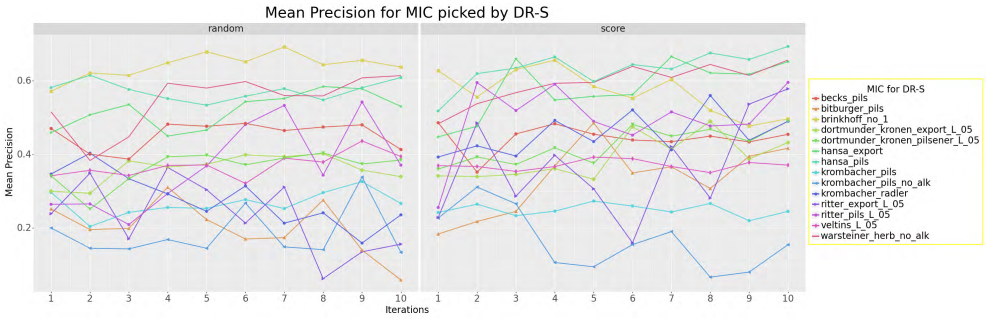


Figure 4.5: **Mean Precision for Most Interesting Class chosen by Data Refinery Score Sampling regarding the Bottle-Marketing-Label-Recognition task.** A enlarged figure is included in Appendix D (Figure D.8).

4.4 HUMAN ANNOTATION TIME OVER ITERATIONS

The focus of the previous sections lied in the evaluation of the different FEs performances for DR-R and DR-S. This section focused on the human step of the DR. To be more specific, it examined how long it took a human to answer a "question" in the form of a task (Section 3.4), DR-R, respectively DR-S, was asking.

Figure 4.6 shows the average time (y-axis) it took a human to answer a task for DR-R (red) or DR-S (blue), per iteration (x-axis). The x-axis was cut off after 35*s*. It is important to note, that the human workers did this task as a side task along with their normal workload if they had time to spare. Because of this, there were many extreme outliers. To account for this 5% of the highest measured times were not included in this evaluation.

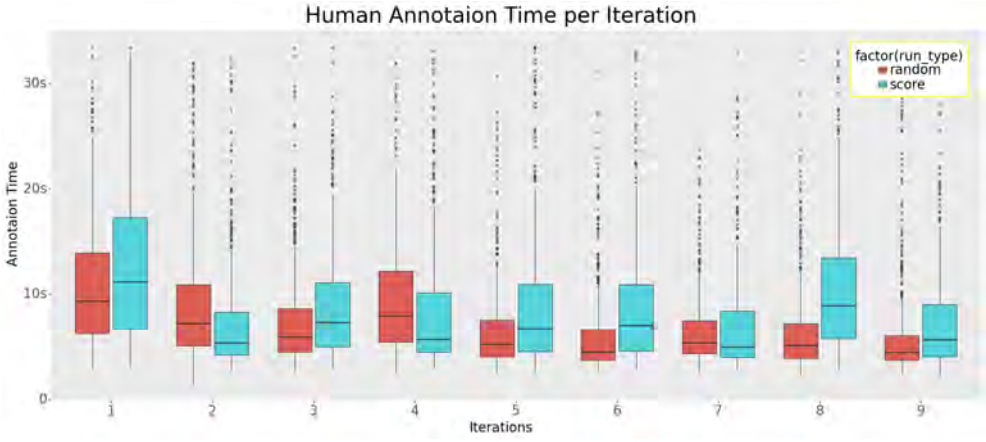


Figure 4.6: **Human Annotation Time Over Iterations.**

The annotation time for the first iteration of DR-R shows a median of about 9*s*. For the following iterations annotation time was decreased with the fourth iteration showing a slight increase, before reducing again. Until iteration four, DR-R showed quite a bit of variance. For the iterations five through nine, the

variance was reduced, compared to the previous iterations. The mean annotation time lied around 5*s* for iterations five through nine.

For the first iteration the median annotation time for DR-S lied around 12*s* and dropped beneath 6*s* for the following, second iteration. In contrast to DR-R, the variance in annotation time was quite high in every iteration, with the highest variation in iteration one. After iteration four the median lies above the median of DR-R for all following iterations, with iteration seven being the only exception.

This chapter presented the results of the case study. An improvement of individual classes DR-S determines MIC could be noted. This is in contrast to DR-R where almost no such individual improvements were observed. It could also be observed that the time humans needed to answer a "question" asked by DR-R was decreased over the iterations. This could not be shown for "questions" asked by DR-S where the annotation time fluctuated. In the most cases, the median annotation time required to answer a "question" asked by DR-S was higher than the compared annotation time of DR-R. An overall improvement for Micro Precision (MiP), Micro Recall (MiR), Macro Precision (MaP), and Macro Recall (MaR) over the iterations was not shown in either DR-S or DR-R.

5 DISCUSSION AND CONCLUSIONS

The results discussed in Chapter 4, indicate that there is a problem in the DR process as executed for the presented case study. The three following indicators, and the reason why this problem may have occurred, are discussed in the following section.

The overall perfomance plot Figure 4.1 in Section 4.1 showcases that DR-R and DR-S start with different perfomances in the first iteration, while both of them had the exact same training data set at this point of the case study. This could be caused by the fact that only a maximum of $500K$ triplets were generated from the initial training set. Another reason could be that due to early stopping, training times for the FEs differ.

The average training times in the first iteration for DR-R (6:00h) and DR-S (6:20h) differ slightly which does not provide sufficient reasoning to solely explain the discrepancy in performance. It is more likely that training time was not long enough for the FEs to fully converge and that DR-R and DR-S did not have a lot of overlap in the generated triplets besides the fact that the generated triplets came from the same training data set. Hence, further investigation is needed.

The plot Figure 4.1 from Section 4.1 also shows no significant improvement over the iterations. Over the iterations, the DR process generates more data, this should increase performance because one commonly accepted principle of machine learning is: "More data is better".[1] One cause could be the fact, that the training time and triplets created were not enough to use the full potential of the newly acquired data. It is also possible, that the new data which was generated by human workers is error-prone, because the answers of the humans were not validated. This would resonate with another ML principle: "garbage in, garbage out (GIGO)" [24].

Finally, another cause, although it might play a smaller role compared to the other reasons previously discussed, could be that the three pre-trained ResNet 18, used in the FEs were frozen. This means that the weights of those ResNets were not updated in training. The reason for this could be that specific features from the training image data might not have been incorporated by the FEs.

The plot Figure 4.1 from Section 4.1 shows another interesting, yet unexpected result: The dip in performance observed in iteration two for DR-R and

1 This principle is not always true, the wrong data can lead to worse performance.

DR-S. In the second iteration DR-R and DR-S had more data for training, but still drop in performance. In the presented case study it was not fully possible to evaluate why this happens. Most likely the explanation lies in the structure of the core data set and the algorithm used to build triplets. In Section 3.3 the training algorithm, which includes the triplet generation is explained.

In the training algorithm the step 2.3 (1.11) shows how to select a negative for a triplet. The parameter p, which was set to 80%, determines how likely it is that an explicit hard negative is chosen. But p was not the only factor, there also had to be an explicit negative sample for a given class in the core data set. As described in Section 3.2, explicit negatives were not available for every class in the core data set. As a result of this the amount of triplets with hard negatives was 0% instead of 80% for some classes in the first iteration. At the end of the first iteration, new explicit negative samples were added for all classes. This made the training process more difficult. Adding only a few explicit negative samples for the classes in question resulted in an increasing frequency in which they were used as hard negative examples in the second iteration. So every error a human caused, had a significant impact in this iteration. The influence of human error on the classes in question would decrease over the iterations, because of the increase in data added.

Even though the cause of the dip cannot be fully explored, it poses as an opportunity to evaluate if the goal of the case study is met. If the goal is met, DR-S should ask for more examples in the classes where it under performed. One indication for the successfully met goal is, that DR-S is able to recover from the dip in performance after one additional iteration in contrast to DR-R which needs two iterations.

To further strengthen this point, one can examine the class "Jever Pilsner Fun no Alk 0.5L". In Figure 4.4 from Section 4.3, the precision from DR-S over the iterations for selected classes is shown in the right plot. The dip is clearly visible for "Jever Pilsner Fun no Alk 0.5L" (blue) and drops from 75% in iteration one to 39% in iteration two, finally recovering to 76% in iteration three. If DR-S behaves as proposed, it should ask for more examples of "Jever Pilsner Fun no Alk 0.5L" in iteration two which will change the relative class frequency in iteration three for the training set of this class. If one looks at the Figure 4.3 for relative class frequency, an increase in iteration three (green square), followed by a decrease in the following iterations for the class "Jever Pilsner Fun no Alk 0.5L" can be observed.

The same effect can be seen in other classes for DR-S with and without the dip in performance. Most prominent is the class "Leerfach" where DR-S adds almost no new data samples. This is in a stark contrast to DR-R, which adds more data in every iteration, ignoring the fact that "Leerfach" already has a

precision and recall of 100% since iteration one.

It is not yet explored why there is no overall improvement over the iterations for DR-S observed (Figure 4.1). It is most likely that a faulty training process, which is unable to utilize the provided data, insufficient model parameter optimization and poorly optimized training time play a major role as discussed previously.

Figure 4.6 of Section 4.4 shows the average annotation time a human needs to complete a task given by DR-R and DR-S. After iteration four, the iterations before can be viewed as a "training period" for the human to learn the task, the average time and the variance for tasks given by DR-S is higher in almost all iterations when compared to tasks given by DR-R. This indicates that the human required more time to answer the task given by DR-S, because the question was harder.

Figure 5.1 shows an example of a task asked by DR-S after iteration four. The task asks if the bottle in the middle is a "Krombacher Pils 0.33L" bottle (A) or a "Bitburger Pils 0.33L" bottle (B). The answer to the question is not easy, especially with the label not fully visible and the bottle located at the side of the crate.

Figure 5.1: **Human Task Example 1.** Example 1 of a question asked by DR-S after iteration four: does the bottle in the middle belong to "Krombacher Pils 0.33L" (A) or to "Bitburger Pils 0.33L" (B) or neither? Answer: the bottle belongs to (B).

In Figure 5.2 another question asked by DR-S after the fourth iteration is shown. The question is, whether the bottle belongs to "Sol 0.33*L*" (A) or "Veltins V+ Curuba 0.33*L*" (B). The bottle in the middle has no logo, so it is possible it is a "Veltins V+ Curuba 0.33*L*" with a missing logo or an entirely different bottle. This backs the question if a "Veltins V+ Curuba 0.33*L*" without a logo does belong in the category "Veltins V+ Curuba 0.33*L*" or in a different, new category. This depends on the use case. For the CrI, "Veltins V+ Curuba 0.33*L*" and the bottle in middle are standard white bottles and "Sol 0.33*L*" is a special white bottle not to be confused with the standard white bottles.

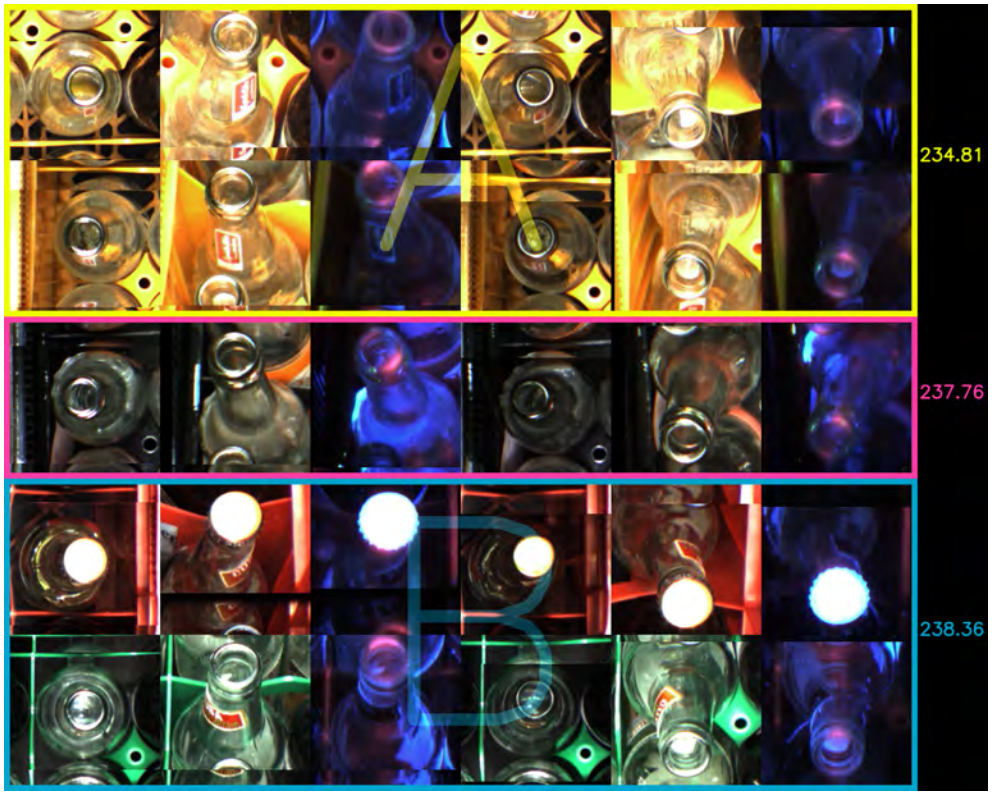


Figure 5.2: **Human Task Example 2.** Example 2 of a question asked by DR-S after iteration four: does the bottle in the middle belong to "Sol 0.33*L*" (A) or to "Veltins V+ Curuba 0.33L" (B) or neither? Answer: it is unclear, as the answer depends on the use case.

Figure 5.3 shows a peculiar case which could be easily missed. If one would only look at the image, it is easy to see that the image in the middle does not contain a bottle and thus belongs to the class "Leerfach" (A). However, a second look and focus on the height measurements (right side) indicates an error in the machines measurement apparatus. The Crate Inspector (CrI) measured a height of 238, but the height of a "Leerfach" is always around 55. With this question asked, DR-S challenges the given information and so detects errors in the data generation process. This can not only be used for

error detection, but also to account for content shift problems.

Figure 5.3: **Human Task Example 3.** Example 3 of a question asked by DR-S after iteration four: does the "bottle" in the middle belong to "Leerfach" (A) or to "Veltins 0.33L" (B) or neither? Answer: it belongs to A.

Some more examples of "questions" are provided in the Appendix C.

5.1 SIDE NOTE DATA REFINERY VERSUS CRATE INSPECTOR

During the evaluation of this study one further interesting question arose. "Would a Feature Extractor (FE) with a K-NN classifier attached, reach a similar or even better performance than the Crate Inspector (CrI) regarding the Bottle-Type Recognition (BT-R) task?" This may be up for further research adding to the outcomes of this work.

Before examining the results in more detail, there is one problem that has to be addressed. There is no strict statistically founded baseline for the performance of the CrI. At the end of the case study, there was an attempt, even though no solid baseline was given, to at least estimate the performance of the CrI, which was used as a comparison to DR trained FEs. For this purpose, two FEs from the last iteration of the DR-R and DR-S with the respectively highest performance in that iteration were evaluated on $10K$ data samples obtained before the generation of the core data set.

Suppose, the CrI is 100% accurate in the prediction of a bottle-type for all $10K$ given bottles, then the Micro Precision (MiP) of the DR-R trained FE was 77.46% and 70.91% for the DR-S trained FE. But to estimate the real performance of the CrI and the two FEs, two Spot Checks (SC); for the DR-R trained FE (SC 1) and for the DR-S trained FE (SC 2) were performed. With the goal of a 95% Confidence Interval (CI) (i.e., a margin of error $e = 0.05$) and a confidence level of 95% (i.e., a z-score of z = 1.96) by a total population N of $10K$ samples and a SD $p = 0.5$, Equation 5.1 returns a sample size for a Spot Check (SC) of about 370. This was generously rounded up to a sample size of 400 for both SCs.

$$\text{Sample size} = \frac{\left[z^2 * p(1-p)\right] / e^2}{1 + \left[z^2 * p(1-p)\right] / e^2 * N} \tag{5.1}$$

The SC manually checked for each of the 400 samples per SC; if the CrI and if the FEs (DR-R or DR-S) recognized the bottle-type correctly, or if CrI and the FEs fail to recognize the bottle-type of a given sample. The result of both spot checks is given in Table 5.1. It can be seen, that the estimated performance of the CrI for SC 1 lies around 89.59% (86.57 ± 92.61), for spot check SC 2 around 87.66% (84.38 ± 90.93) and that the DR-R trained FE seems to reach the CrI performance-wise with 87.40% (84.10 ± 90.70). In contrast to the DR-S trained FE, which shows the worst performance among the three.

Notice, that because each CrI is configured for a customer who needs to recognize specific bottle-types, the performance for those bottle-types is much

higher. For easier compatibility the overall performance (micro precision) of all known bottle-types (classes) is used in this section.

SPOT CHECK	MICRO PRECISION	CONFIDENCE INTERVAL	N
(SC 1) DR-R FE	87.40%	84.10 ± 90.70	389
(SC 1) CrI	87.66%	84.38 ± 90.93	389
(SC 2) DR-S FE	78.43%	74.42 ± 82.49	394
(SC 2) CrI	89.59%	86.57 ± 92.61	394

Table 5.1: SC out of the $10K$ test samples to estimate the performance for the CrI and the best FE for DR-R and DR-S from the last iteration of the case study. Per SC 400 samples were checked, some of them could not be unambiguously identified. N specifies how many samples could be unambiguously identified. In addition to the Micro Precision (MiP) the 95% CI is given.

5.2 CONCLUSION

There is often a gap between the untidy data the industry has and the tidy data needed to efficiently train novel Machine Learning (ML) algorithms. In this work, a novel concept, namely the Data Refinery (DR) is proposed to tackle this gap by using the human ability to extract important semantic information and the ability of Deep Computer Vision (DCV) in combination with Deep Metric Learning (DMeL) to quickly examine large amounts of data. A case study was conducted to compare two different DRs which only differed in the aspect of which untidy data samples were selected to show to a human, so they could be labeled.

One DR randomly picked data samples (DR-R), the other DR calculated a score which reflects how valuable this data sample for a training process is (DR-S). The results indicate that DR-S, in contrast to DR-R, specifically adds new data samples for training in classes where it underperformed. Contrary to the expectations, this did not lead to an overall improvement of DR-S when compared with DR-R, but to a similar performance of $\sim$ 60% for Micro Precision (MiP). When the results of this case study are compared with other deep metric learning works, it underperformed [5]. The main reasons for this, could be that in the Related Work (Section 2.3), only experts labeled the human tasks and the training was up to 10 times longer. Regardless, the goal of the case study was not to train a state of the art classification model for a given task, but to efficiently mine data for such a model.

Even though the success of the case study is only moderate performance wise, the new approach of the DR as a concept proves to be a promising way to acquire specific data that can be used to create curated data sets. Such curated data sets, are not only valuable for the industry (i.e., Bottle-Type Recognition (BT-R)), but can be used for all kinds of ML tasks throughout all industries.

Before this concept can be applied, the parts of the case study that proved to be problematic should be revised. Also, certain aspects of the general concept, as an example: the score function, should be optimized or alternative options reviewed.

6 OUTLOOK

The implementation of the DR, in the context of the presented case study represents a prototype. The case study indicates that the principle idea of the DR can work. It also showed there are problems with the implementation. The most obvious issue is with the training process, as stated before. Besides the optimization of the training process, optimization is required for other elements of the DR as well.

One such element requiring optimization, is the score function. In the case study Expected Utility (EU) is used to estimate the utility of a given untidy data sample. In the case study, there was no investigation on which parameters for EU would be optimal. For example: in order to follow a risk averse strategy, the utility function used, was the square root. It is not clear whether the risk averse strategy is the most effective for reaching the goal of the case study or not. In future studies, it should be investigated which strategy is most promising and which function is best to map this strategy. Furthermore, if there is a better way than a simple lottery and EU to estimate the value of untidy data samples, should be illuminated.

Another element to improve is the human step. In this case study, the human was assumed to be infallible but in truth, making errors is unavoidable for a human. The answers given by humans must be reviewed, but this in not all: if curated data is the enabler for state of the art machine learning for a wide range of tasks on a big scale, the labeling job should be valued more. This means label workers need to be trained more extensively and given enough time to adequately answer the tasks. A future study could lay the focus on the human step in the environment of the DR. The examples of optimization discussed are only a fraction of possible opportunities to improve the concept and implementation of the DR.

6.1 DATA REFINERY CONCEPT EXTENSION FOR OPEN SET CLASSIFICATION

The DRs main job was to detect interesting data samples a human should examine and label. In this work, a non-optimized K-NN based classifier, which was not part of the DR concept, was used to classify data samples. This was done to measure if the different underlying data sets showed an impact on the performance. For open set classification the DR was just thought of as the first step to get new data and their feature embeddings for classification. The next step is a "transformation" which transforms the Feature Space (FS) of the DR to a FE for a specific task. In the context of this work, this could

mean that a FS is optimized for either a Bottle-Type Recognition (BT-R) or Bottle-Marketing-Label Recognition (BML-R) task.

After that, there is another step: which mainly deals with uncertainty after a classification took place in the specialised FS. This step incorporates external information not known to the previous steps. Such information in context to the BML-R could be, the geographical location of the CrI and the resulting drinking habits of the consumers there. Other examples are: in what crate a bottle stands, to what class neighbouring bottles were predicted to and so on. This information statistically analyzed has the potential to guide the classification decision in case of uncertainty.

6.2 FURTHER FIELDS OF APPLICATION AND IMPROVEMENTS FOR THE CRI

The job of the CrI is to recognize glass bottles so that they can be sorted, cleaned and refilled. For this the CrI is using traditional CV and expert knowledge. As stated before, this is quite complex and costly especially when new bottle-types are added and must be recognized. In this work it could be shown that the DR with the random sampling strategy can reach the estimated overall performance of the CrI examined in this case study by using Deep Computer Vision (DCV). This indicated that DCV is applicable to the BT-R task performed by the CrI. In future implementations this could result in less complex, less costly and easy scaleable and generalized solutions regarding the BT-R task.

To improve the symbiosis of ML and CrI, the sensors of the CrI could be upgraded. For example the cameras used by the CrI have a relatively low resolution because of restrictions in the traditional CV process that is used. Those restrictions do not apply in the same sense to the DCV approach. With this upgrade more details are visible on the images which may generate more features, that could lead to a better discrimination between similar classes.

Thinking beyond the BT-R task, the CrI in combination with DCV could be used to recognize other objects that fit on the conveyor belt. In combination with the DR, curated data sets for all kind of objects could be created.

6.3 FURTHER APPLICATIONS REGARDING THE PFANDSYSTEM

According to Statista, 7 European countries plan to implement a "Pfandsystem" until 2023. Additionally, 8 more countries are discussing such an implementation [1]. In Germany, the idea of the expansion of the "Pfandsystem" to wine and juice bottles is considered [36]. With the protection of the environment in mind, more deposit systems will be introduced in the future, where the "Pfand-objects" must be classified and sorted upon return.

A more precise classification of "Pfandflaschen" could also help to create more transparency in the "Pfandsystem". The unclaimed "Pfand" (Pfandschlupf) from the year 2015 for "Einwegpfand" (single use Pfand) alone, accumulates to 180€ Mio according to Naturschutzbund Deutschland (NABU) [21]. The study assumes that of 18 billion disposable bottles per year (4.5€ billion deposit), only about 4% are not returned. The accumulated "Pfandschlupf" in the sector of "Einwegpfand" since its implementation up to 2015 is estimated to be about 3.5€ Billion. The presented numbers are only estimates, because of the lack of transparency. There is a ongoing dispute on who should profit from this money.

6.4 DATA REFINERY AS A TOOL BEYOND IMAGE RECOGNITION

The DR is not restricted to the "Pfandsystem" or mainly using images. The concept of the DR can also be applied to other types of data. For example: audio, where the question the DR asks a human is in form of audio samples. Obtaining curated data sets in this way can be used to train state of the art machine learning algorithms under real world conditions. To validate this, new case studies with different data types must be conducted.

Part II

APPENDIX

A

PERFORMANCE TABLES

The complete data collected for this work is contained on the included CD. The Tables that are included in the Appendix for completeness, show the overall performance (Micro Precision (MiP), Micro Recall (MiR), Macro Precision (MaP), Macro Recall (MaR), Micro F1-Score, and Macro F1-Score) of Data Refinery Score Sampling (DR-S) and Data Refinery Random Sampling (DR-R).

Iteration	MiP Mean	MiP SD	MiR Mean	MiR SD	F1 Mean	F1 SD
1	0.5829	0.0403	0.5829	0.0403	0.5829	0.0403
2	0.5372	0.0704	0.5372	0.0704	0.5372	0.0704
3	0.5565	0.0499	0.5565	0.0499	0.5565	0.0499
4	0.6043	0.0648	0.6043	0.0648	0.6043	0.0648
5	0.6134	0.0680	0.6134	0.0680	0.6134	0.0680
6	0.6187	0.0526	0.6187	0.0526	0.6187	0.0526
7	0.6091	0.0438	0.6091	0.0438	0.6091	0.0438
8	0.6138	0.0392	0.6138	0.0392	0.6138	0.0392
9	0.6308	0.0630	0.6308	0.0630	0.6308	0.0630
10	0.6093	0.0546	0.6093	0.0546	0.6093	0.0546

Table A.1: Micro Performance for Data Refinery Random Sampling.

Iteration	MaP Mean	MaP SD	MaR Mean	MaR SD	F1 Mean	F1 SD
1	0.5255	0.0510	0.4957	0.0507	0.4916	0.0509
2	0.4935	0.0706	0.4562	0.0651	0.4523	0.0680
3	0.5122	0.0496	0.4717	0.0518	0.4646	0.0531
4	0.5658	0.0718	0.5235	0.0694	0.5201	0.0719
5	0.5656	0.0696	0.5304	0.0700	0.5259	0.0707
6	0.5773	0.0628	0.5397	0.0615	0.5354	0.0626
7	0.5595	0.0494	0.5218	0.0460	0.5189	0.0470
8	0.5714	0.0424	0.5338	0.0423	0.5273	0.0423
9	0.5899	0.0699	0.5552	0.0655	0.5482	0.0703
10	0.5647	0.0573	0.5297	0.0562	0.5203	0.0583

Table A.2: Macro Performance for Data Refinery Random Sampling.

Iteration	MiP Mean	MiP SD	MiR Mean	MiR SD	F1 Mean	F1 SD
1	0.6032	0.0480	0.6032	0.0480	0.6032	0.0480
2	0.5619	0.0549	0.5619	0.0549	0.5619	0.0549
3	0.6023	0.0589	0.6023	0.0589	0.6023	0.0589
4	0.6147	0.0701	0.6147	0.0701	0.6147	0.0701
5	0.6135	0.0656	0.6135	0.0656	0.6135	0.0656
6	0.6199	0.0513	0.6199	0.0513	0.6199	0.0513
7	0.6290	0.0636	0.6290	0.0636	0.6290	0.0636
8	0.6268	0.0452	0.6268	0.0452	0.6268	0.0452
9	0.6203	0.0471	0.6203	0.0471	0.6203	0.0471
10	0.6183	0.0559	0.6183	0.0559	0.6183	0.0559

Table A.3: Micro Performance for Data Refinery Score Sampling.

Iteration	MaP Mean	MaP SD	MaR Mean	MaR SD	F1 Mean	F1 SD
1	0.5541	0.0516	0.5196	0.0495	0.5170	0.0496
2	0.5274	0.0610	0.4855	0.0565	0.4828	0.0584
3	0.5537	0.0587	0.5160	0.0565	0.5116	0.0588
4	0.5711	0.0754	0.5310	0.0711	0.5281	0.0731
5	0.5692	0.0715	0.5207	0.0657	0.5207	0.0683
6	0.5710	0.0505	0.5282	0.0531	0.5241	0.0530
7	0.5955	0.0623	0.5337	0.0655	0.5372	0.0659
8	0.5779	0.0509	0.5299	0.0503	0.5279	0.0512
9	0.5856	0.0496	0.5252	0.0440	0.5262	0.0460
10	0.5725	0.0608	0.5213	0.0592	0.5202	0.0613

Table A.4: Macro Performance for Data Refinery Score Sampling.

B

FEATURE SPACE PLOTS

The following plots show a t-SNE [17] representation of the FS from the score model 14 iteration 8. The data samples obtained from the core data set are presented. All plots highlight different properties.

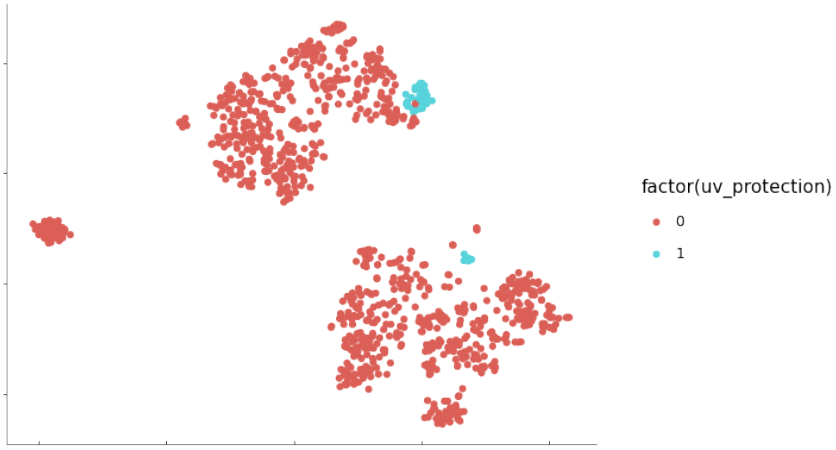


Figure B.1: **Feature Space with the Property UV Protection Highlighted.** Red implies the bottle has no UV protective finish. Blue implies the bottle has a UV protective finish.

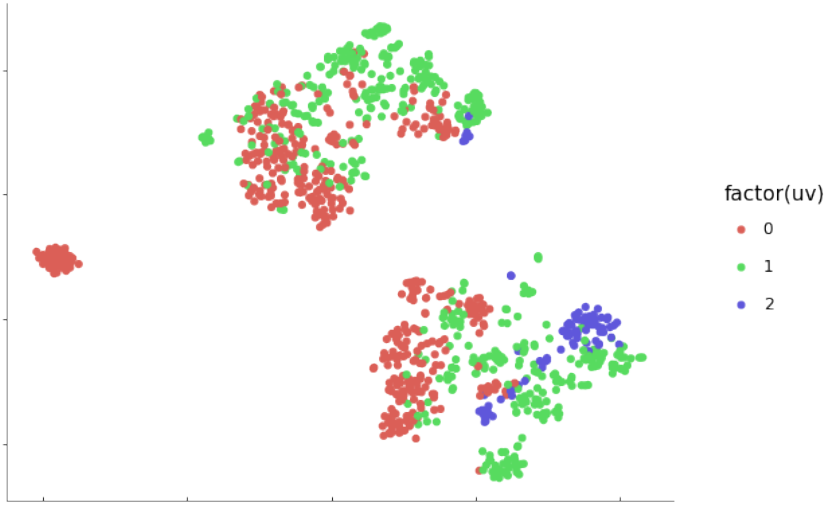


Figure B.2: **Feature Space with the Property UV Marker Highlighted.** Red implies that the image is only black. Blue implies the bottle has a special UV marker from the manufacturer to unambiguously identify the bottle. Green represents a image that is not black but no specific marker can be identified.

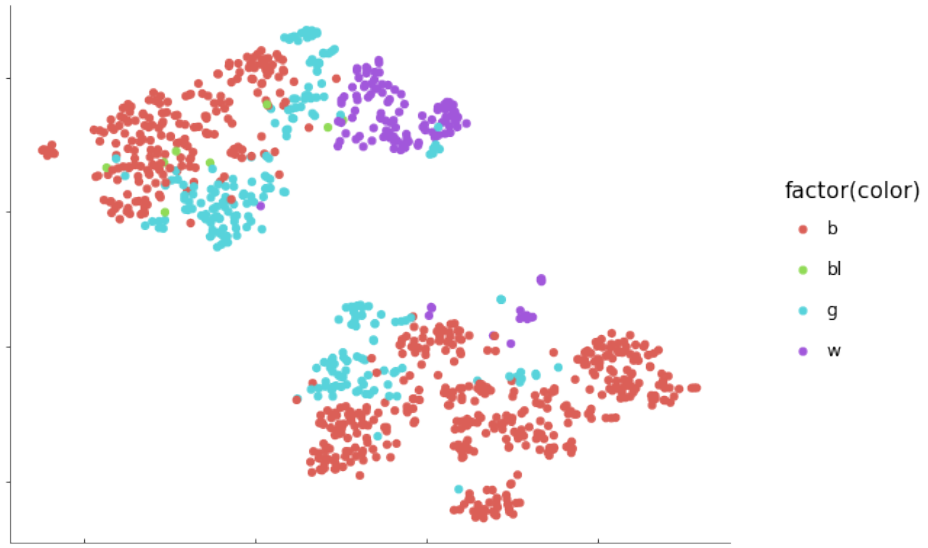


Figure B.3: **Feature Space with the Property Color Highlighted.** Red represents brown bottles, green represents blue bottles, blue represents green bottles and purple represents white bottles.

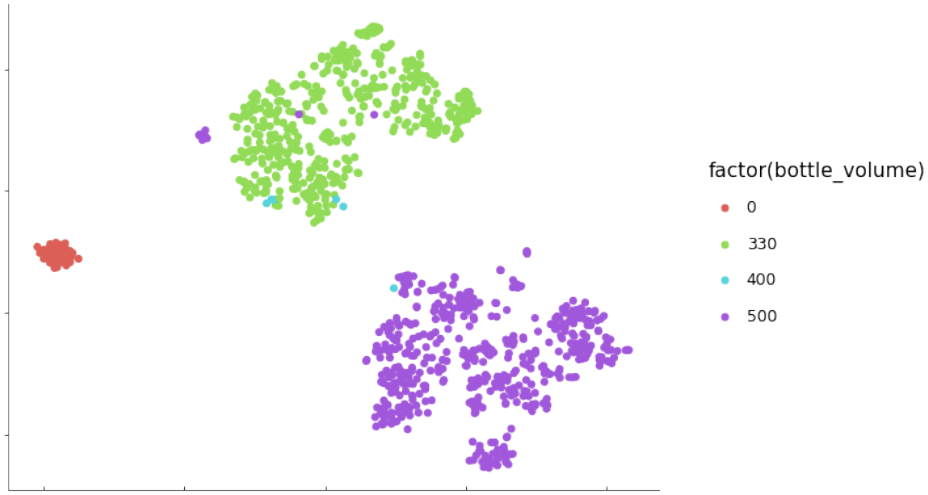


Figure B.4: **Feature Space with the Property Volume Highlighted.** Red represents empty compartments, green represents the volume $0.33L$, blue represents the volume $0.4L$ and purple represents the volume $0.5L$.

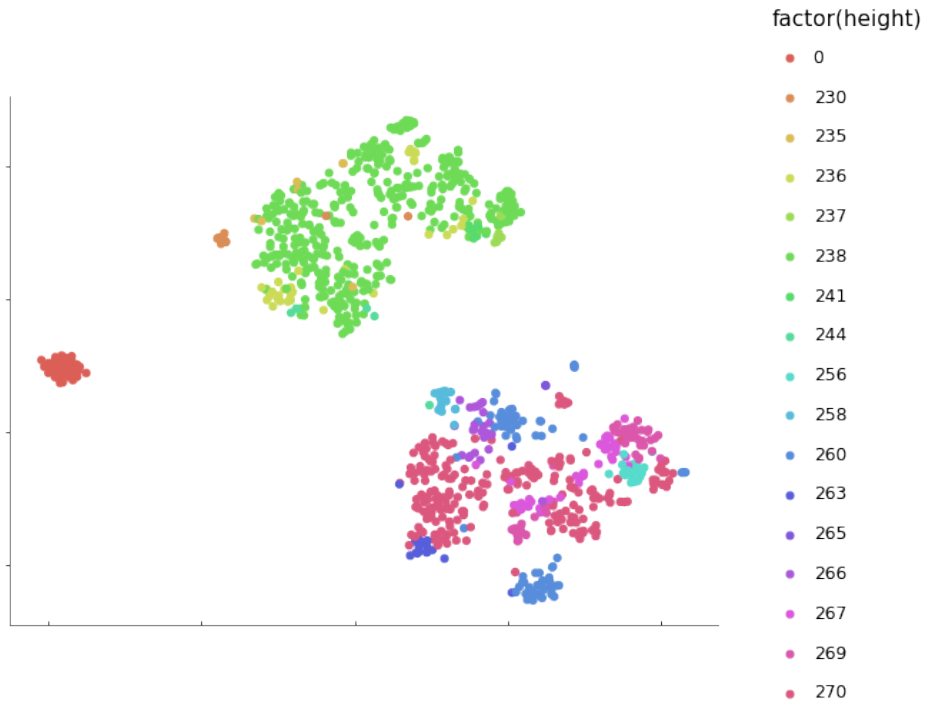


Figure B.5: **Feature Space with the Property Height Highlighted.** The different colors represent the different heights measured by the Crate Inspector.

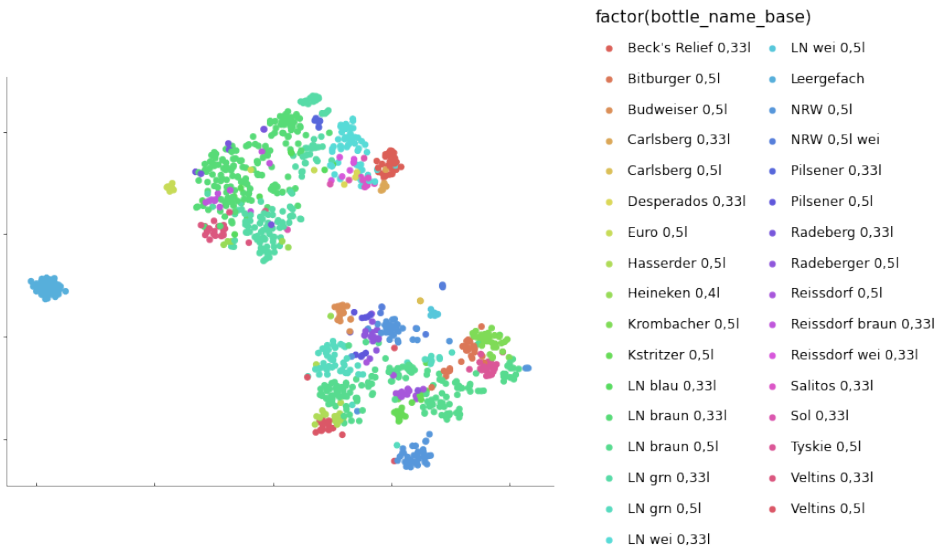


Figure B.6: **Feature Space with the Property Bottle-Type Highlighted.** The different colors represent the different properties of bottle-type.

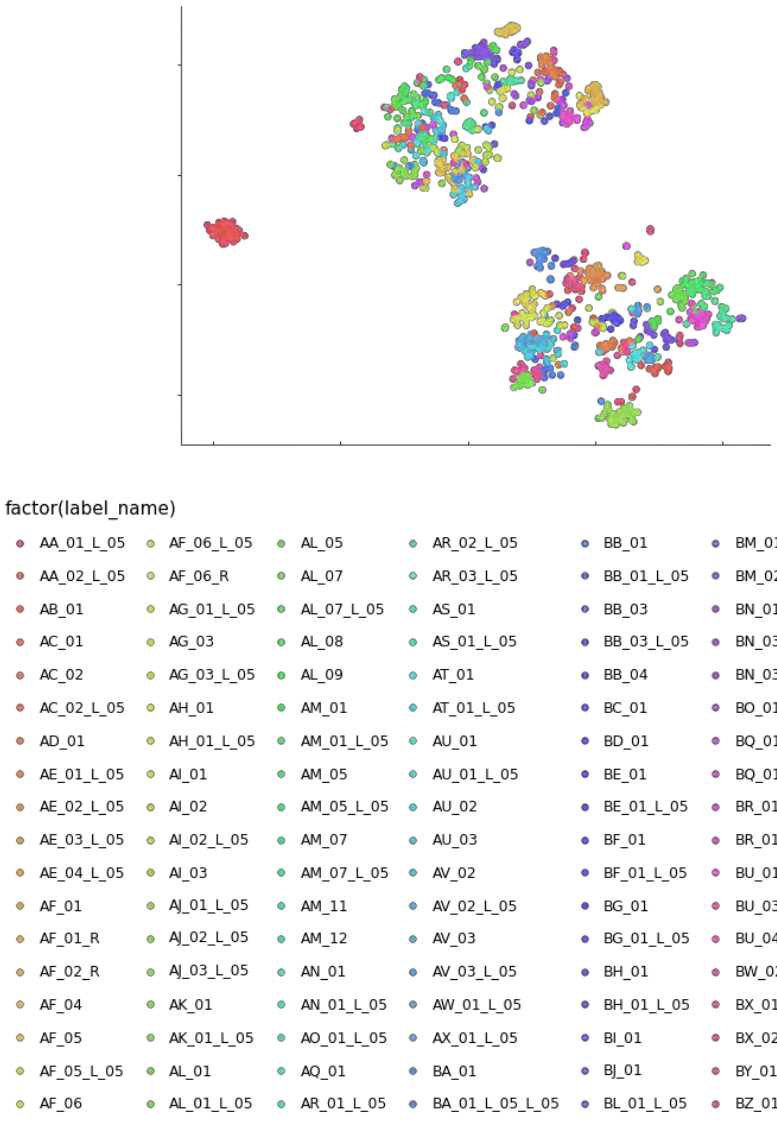


Figure B.7: **Feature Space with the Property Bottle-Marketing-Label Highlighted.** The different colors represent the different properties of Bottle-Marketing-Label.

C

QUESTIONS ASKED BY THE DATA REFINERY

The following figures show more examples from the human tasks or "questions" asked by the Data Refinery Score Sampling (DR-S). Some of the shown questions were also asked by Data Refinery Random Sampling (DR-R), but in less frequent matter.

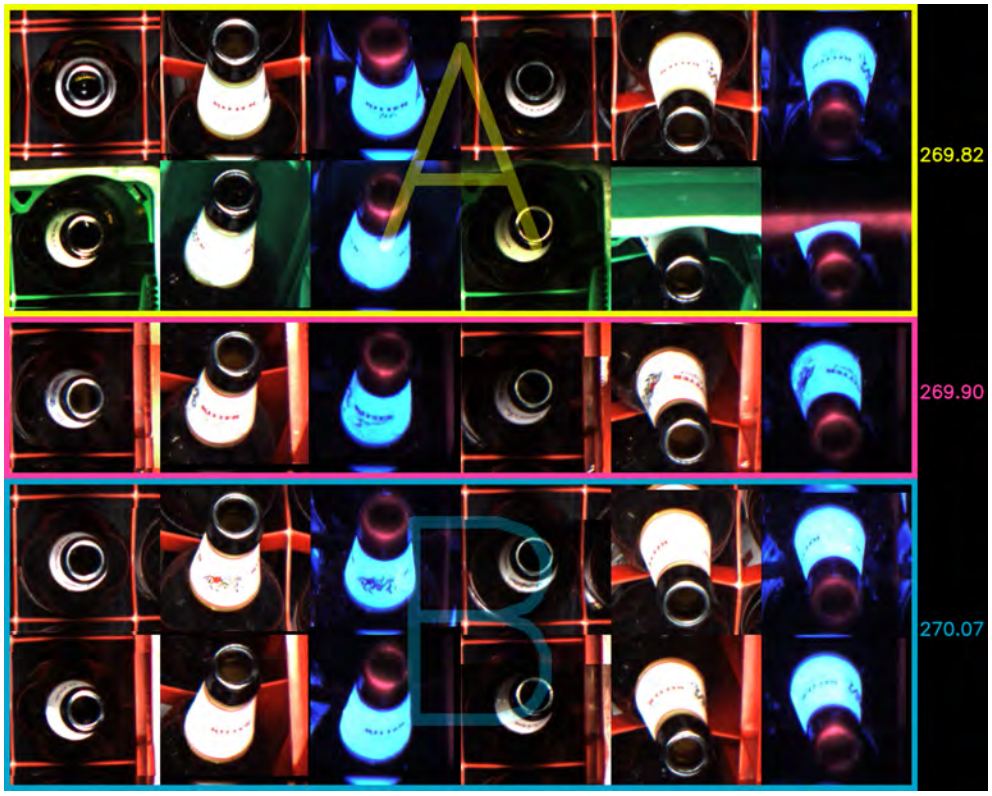


Figure C.1: **Human Task Example 4.** Example 4 of a question asked by DR-S: does the bottle in the middle belong to "Ritter Pils 0.5*L*" (A) or to "Ritter Export 0.5" (B) or neither? Answer: the bottle belongs to (B). The only difference between the bottles are the green, resp. red rings on the paper label on the neck of the bottles.

Figure C.2: **Human Task Example 5.** Example 5 of a question asked by DR-S: does the bottle in the middle belong to "Riter Export 0.5*L*" (A) or to "Krombacher Pils no Alk. 0.5*L*" (B) or neither? Answer: the bottle belongs to (B). The logos of the bottles are different, but they both have red rings on the paper label on the neck.

Figure C.3: **Human Task Example 6.** Example 6 of a question asked by DR-S: does the bottle in the middle belong to "Becks Gold Relief 0.33*L*" (A) or to "Becks Gold 0.33*L*" (B) or neither? Answer: the bottle belongs to (A). Both bottles are "Becks Gold". The only difference is one has the Becks relief on the shoulders, the other one doesn't.

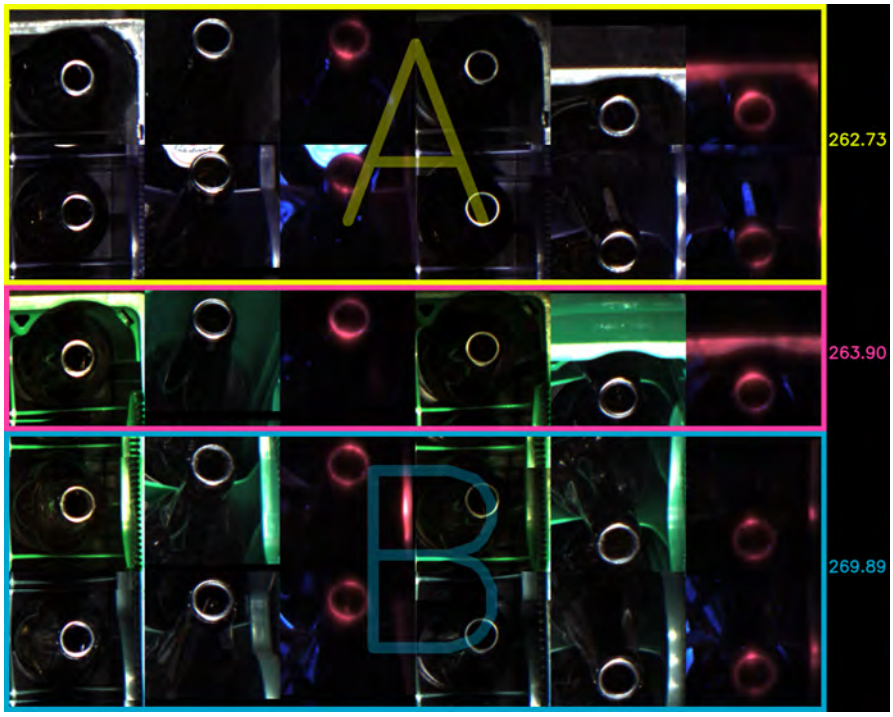


Figure C.4: **Human Task Example 7.** Example 7 of a question asked by DR-S after iteration four: does the bottle in the middle belong to "Veltins 0.5L" (A) or to "Hasseröder 0.5L" (B) or neither? Answer: the bottle belongs most likely to (A). Because of the difficulty of this question, some of the label workers referred to this question as the "Endgegner" (final enemy).

D

ENLARGED FIGURES

This Appendix contains some enlarged figures shown in this work.

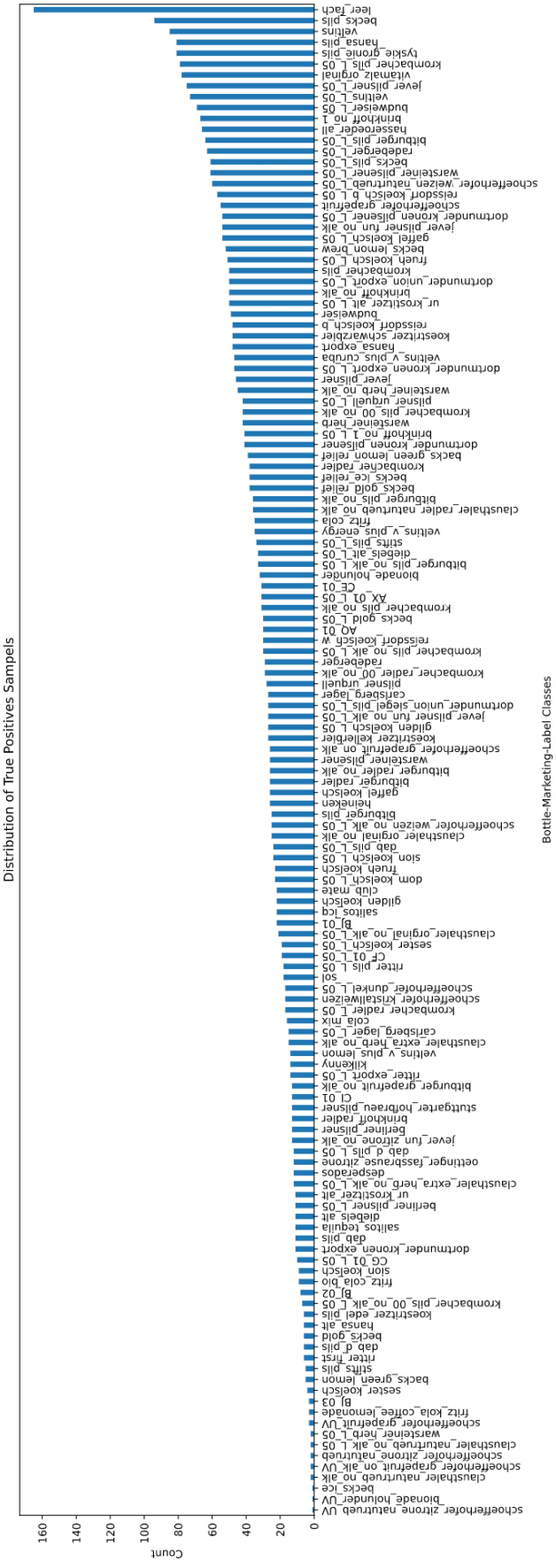


Figure D.1: **The Class Distribution of the True Labels.**

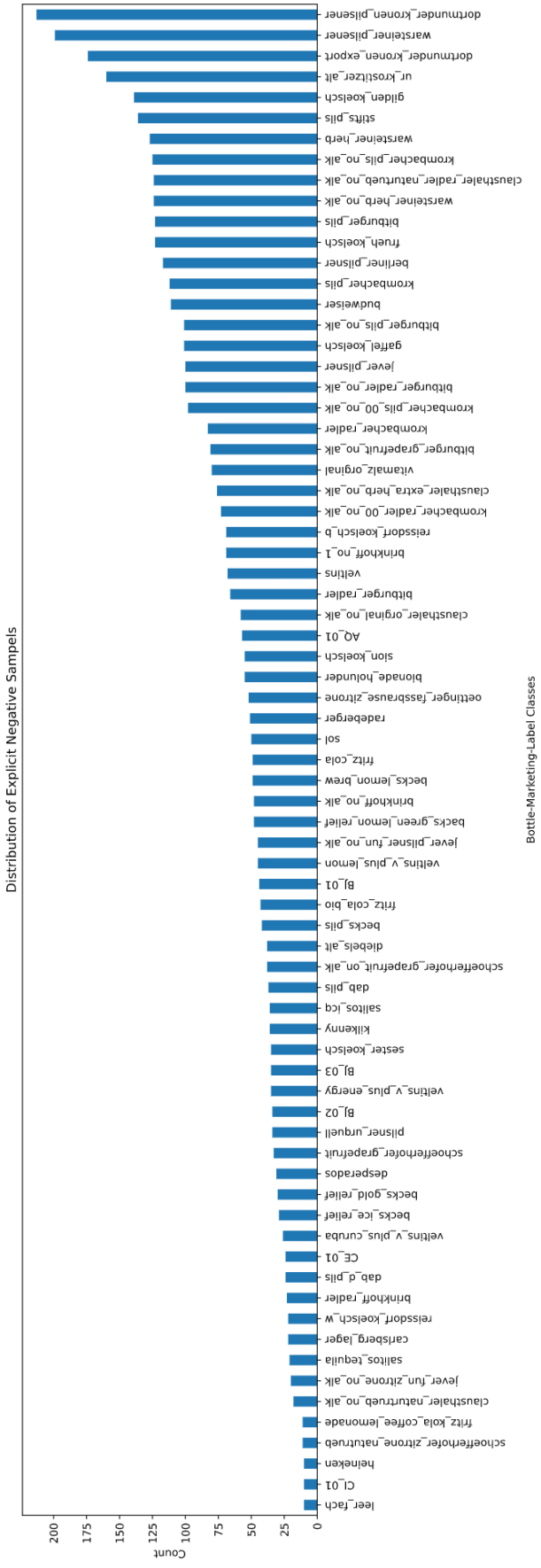


Figure D.2: **The Class Distribution of the Explicit Negatives.**

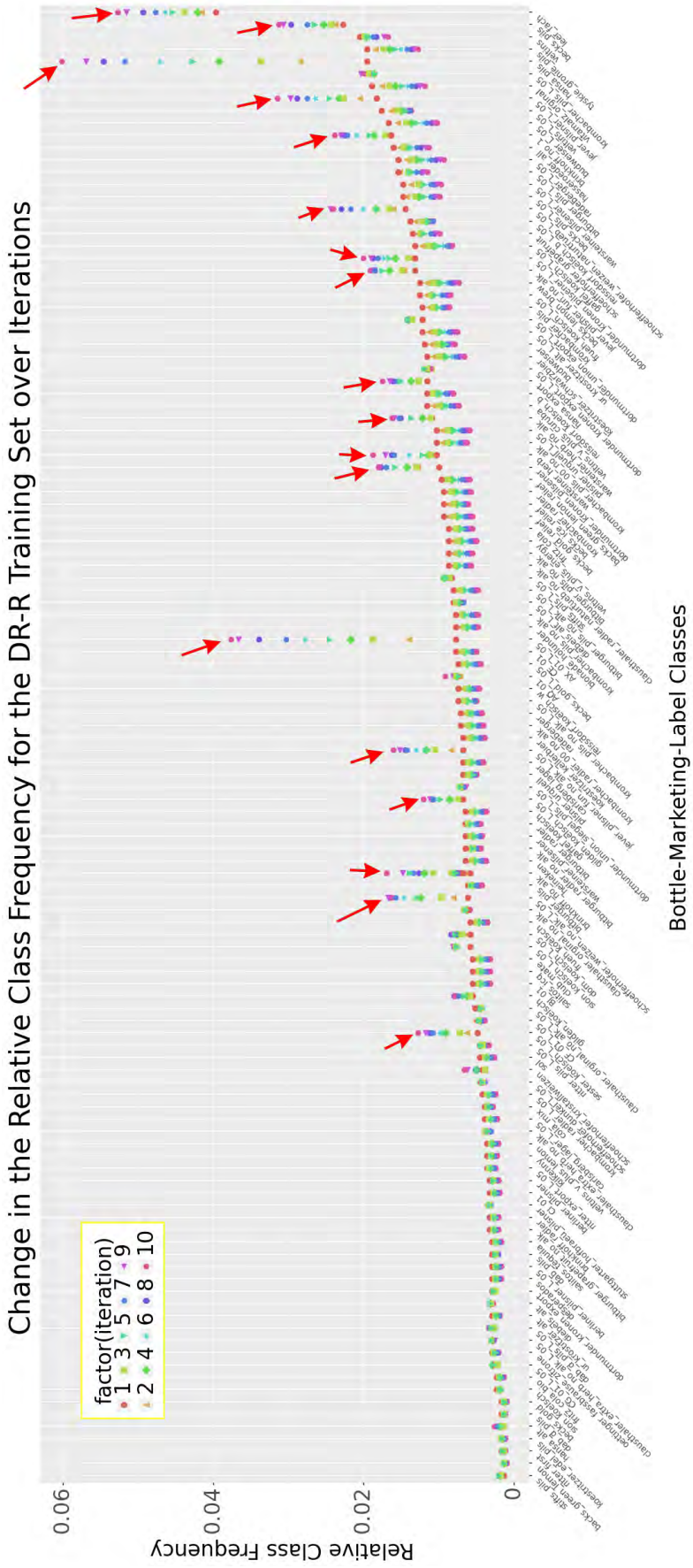


Figure D.3: **Change in the Relative Class Frequency of Bottle-Marketing-Label Classes in the Random Training Set over Iterations.**

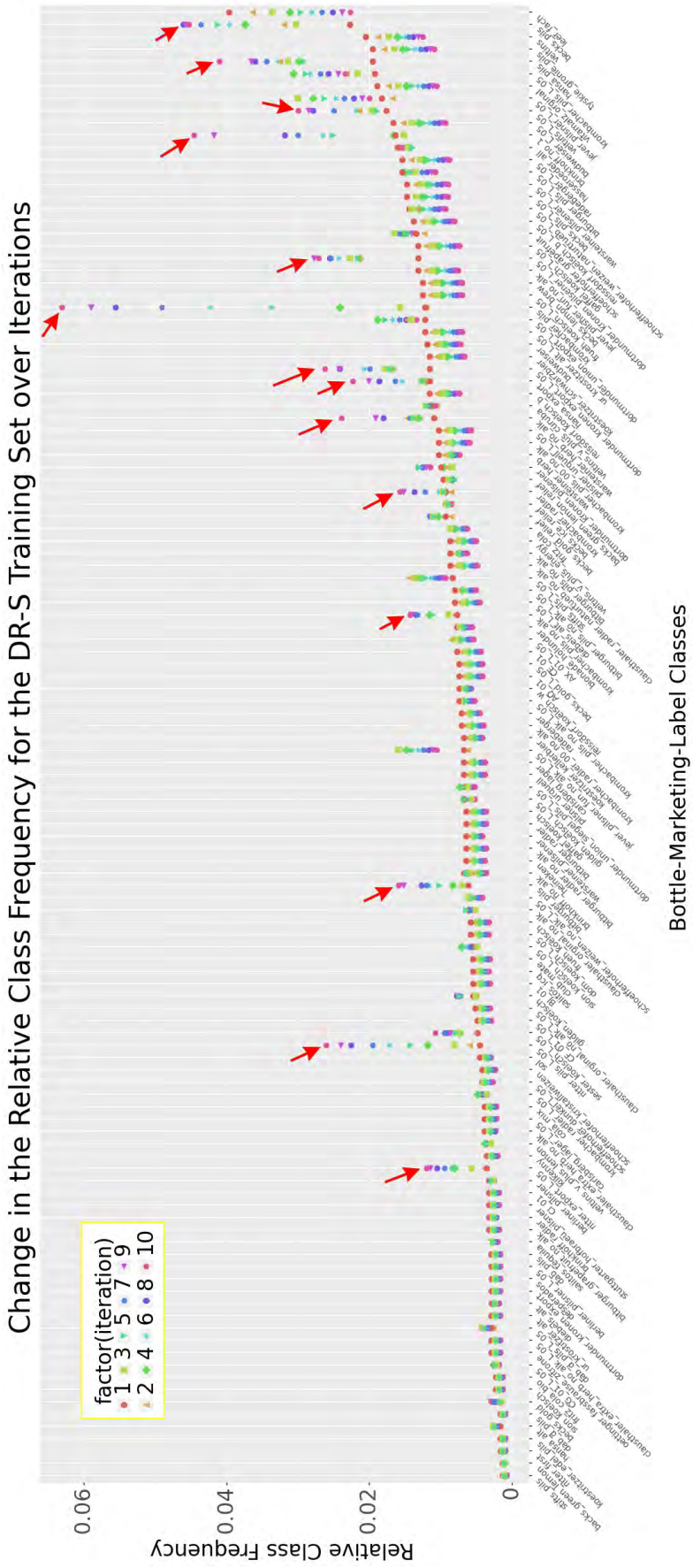
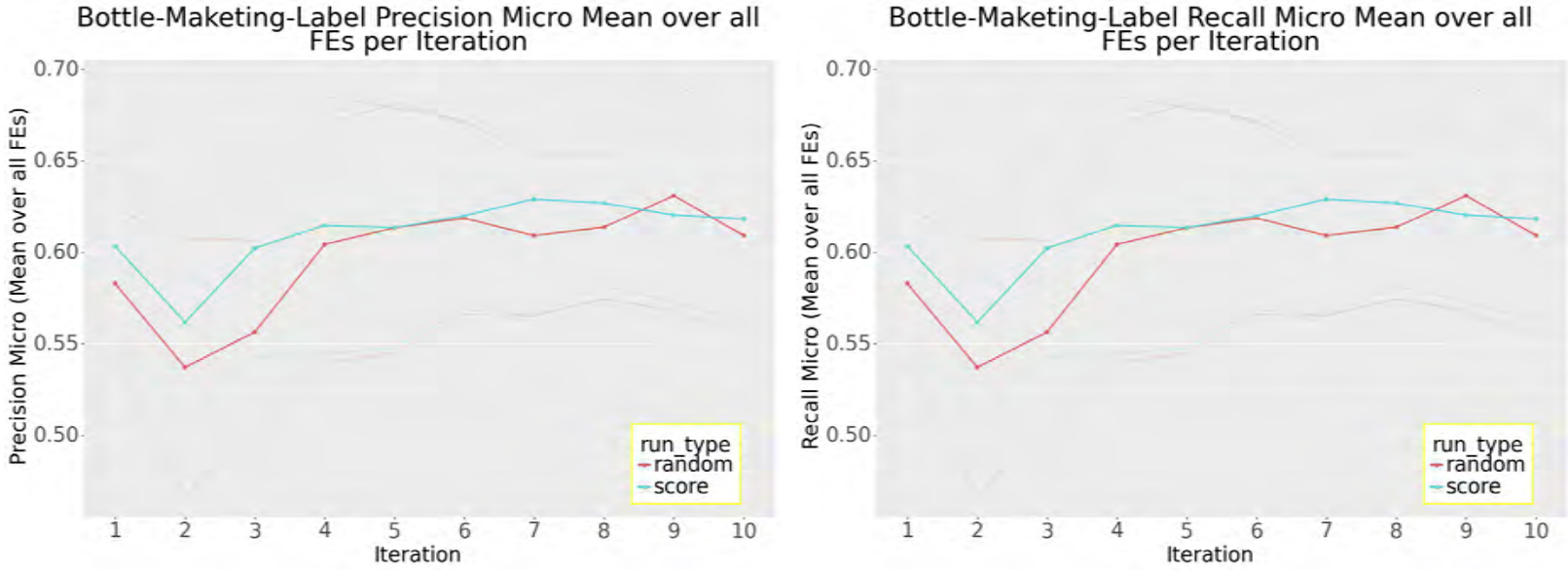


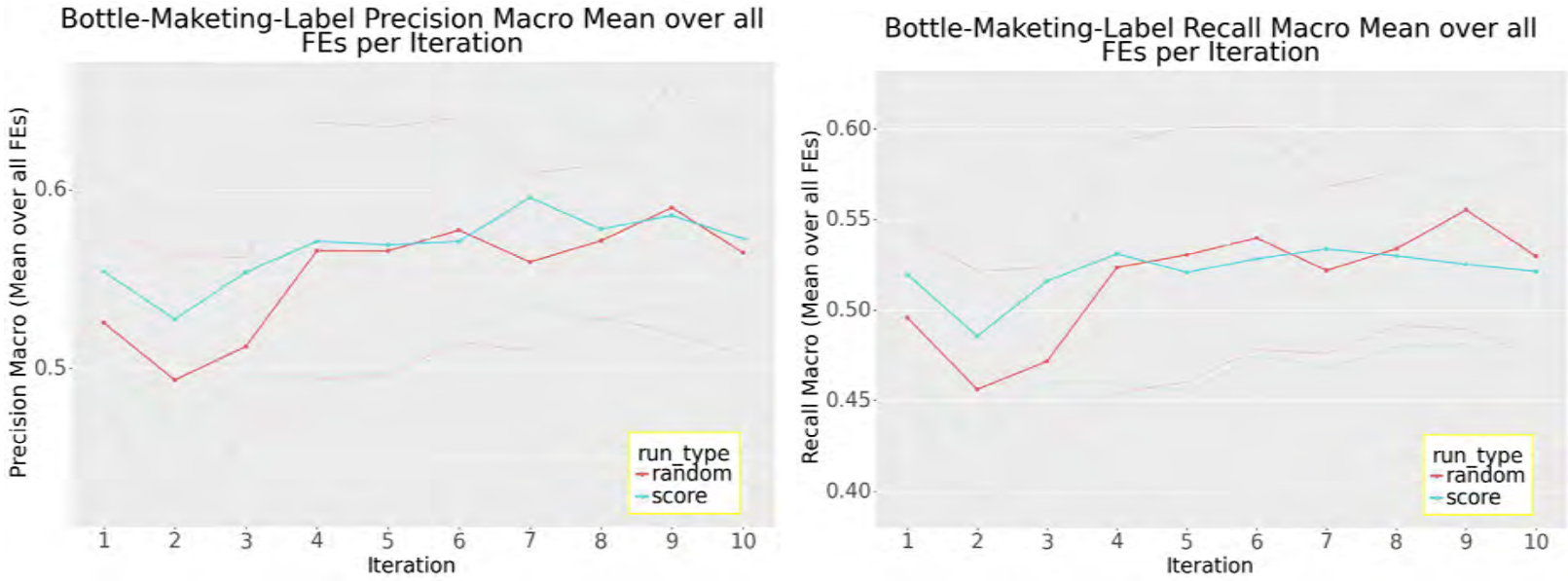
Figure D.4: **Change in the Relative Class Frequency of Bottle-Marketing-Label Classes in the Score Training Set over Iterations.**



(a) Micro Precision (MiP)

(b) Micro Recall (MiR)

Figure D.5: **Overall Performances per Iteration for Data Refinery Score Sampling and Data Refinery Sampling.** The two plots show the Micro Precision (MiP), Micro Recall (MiR) for DR-S and DR-R per iteration for the BML-R task.



(a) Macro Precision (MaP)

(b) Macro Recall (MaR)

Figure D.6: **Overall Performances per Iteration for Data Refinery Score Sampling and Data Refinery Sampling.** The two plots show the Macro Precision (MaP), and Macro Recall (MaR) for DR-S and DR-R per iteration for the BML-R task.

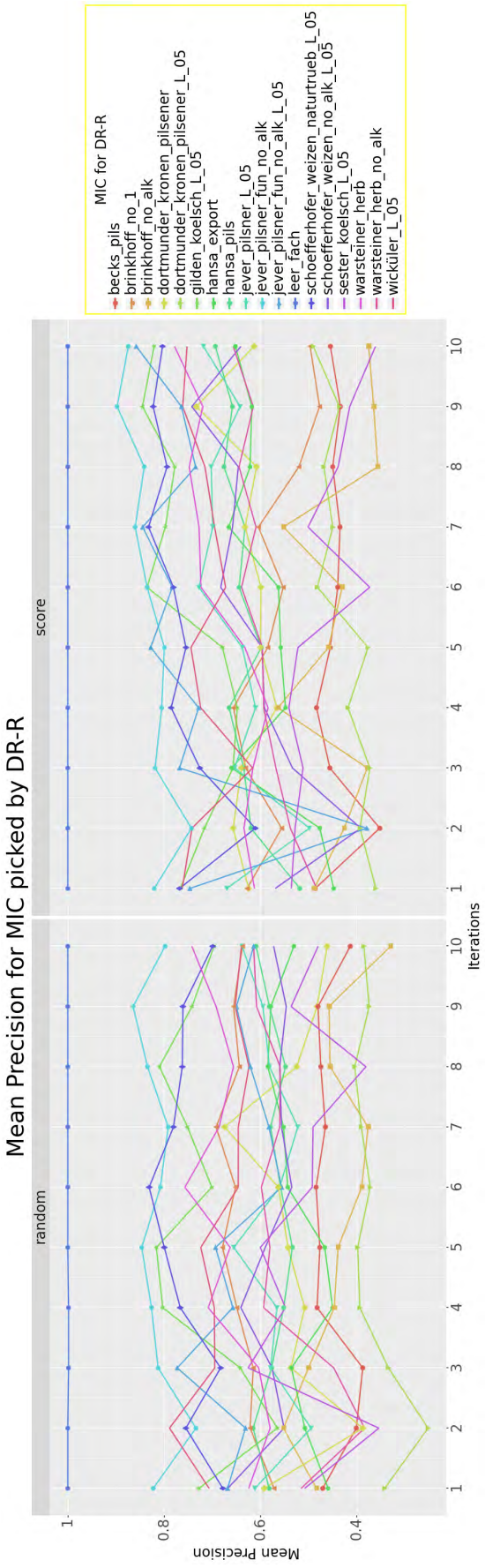


Figure D.7: **Mean Precision for Most Interesting Classes chosen by Data Refinery Random Sampling regarding the Bottle-Marketing-Label task.**

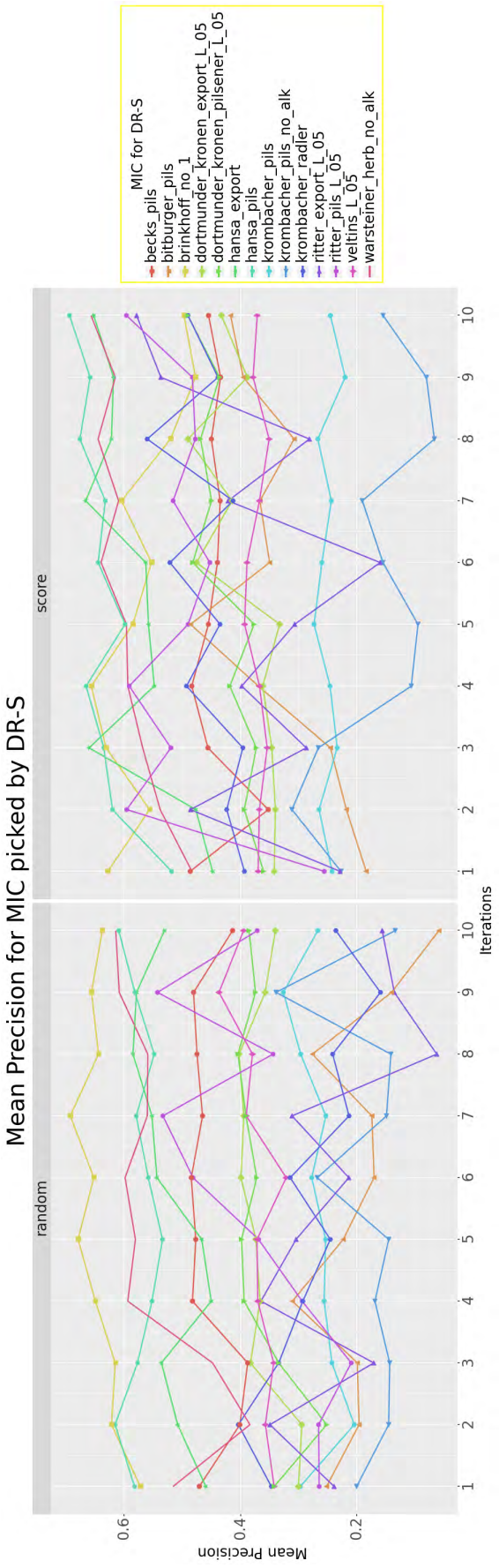


Figure D.8: **Mean Precision for Most Interesting Classes chosen by Data Refinery Sampling Score regarding the Bottle-Marketing-Label task.**

BIBLIOGRAPHY

[1] René Bocksch. *Infografik: Pfandsysteme in Europa*. Statista Infografiken. 2020. URL: https://de.statista.com/infografik/21881/aktive-und-geplante-einweg-pfandsysteme-in-europa/ (visited on 03/06/2021).

[2] R. A. Briggs. "Normative Theories of Rational Choice: Expected Utility." In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Fall 2019. Metaphysics Research Lab, Stanford University, 2019. URL: https://plato.stanford.edu/archives/fall2019/entries/rationality-normative-utility/ (visited on 02/25/2021).

[3] Thyago P. Carvalho, Fabrízzio A. A. M. N. Soares, Roberto Vita, Roberto da P. Francisco, João P. Basto, and Symone G. S. Alcalá. "A systematic literature review of machine learning methods applied to predictive maintenance." In: *Computers & Industrial Engineering* 137 (Nov. 2019), p. 106024. ISSN: 03608352. DOI: 10.1016/j.cie.2019.106024. URL: https://linkinghub.elsevier.com/retrieve/pii/S0360835219304838 (visited on 03/10/2021).

[4] Edward Cone and James Lambert. *How Robots Change the World*. June 2019. URL: https://resources.oxfordeconomics.com/how-robots-change-the-world.

[5] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge Belongie. "Fine-Grained Categorization and Dataset Bootstrapping Using Deep Metric Learning with Humans in the Loop." In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, June 2016, pp. 1153–1162. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.130. URL: http://ieeexplore.ieee.org/document/7780499/ (visited on 01/07/2021).

[6] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. "Recent Advances in Open Set Recognition: A Survey." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020), pp. 1–1. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2020.2981604. arXiv: 1811.08581. URL: http://arxiv.org/abs/1811.08581 (visited on 03/07/2021).

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition." In: *arXiv:1512.03385 [cs]* (Dec. 10, 2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (visited on 02/25/2021).

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification." In: *arXiv:1502.01852 [cs]* (Feb. 6, 2015). arXiv:

1502.01852. URL: http://arxiv.org/abs/1502.01852 (visited on 03/03/2021).

[9] Alexander Hermans, Lucas Beyer, and Bastian Leibe. "In Defense of the Triplet Loss for Person Re-Identification." In: *arXiv:1703.07737 [cs]* (Nov. 21, 2017). arXiv: 1703.07737. URL: http://arxiv.org/abs/1703.07737 (visited on 10/12/2020).

[10] Simon Hessner. *Why are precision, recall and F1 score equal when using micro averaging in a multi-class problem? – Simon's blog*. July 19, 2018. URL: https://simonhessner.de/why-are-precision-recall-and-f1-score-equal-when-using-micro-averaging-in-a-multi-class-problem/ (visited on 02/19/2021).

[11] David Hutchison et al. "Visual Recognition with Humans in the Loop." In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Vol. 6314. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 438–451. ISBN: 978-3-642-15560-4 978-3-642-15561-1. DOI: 10.1007/978-3-642-15561-1_32. URL: http://link.springer.com/10.1007/978-3-642-15561-1_32 (visited on 02/25/2021).

[12] Kaya and Bilge. "Deep Metric Learning: A Survey." In: *Symmetry* 11.9 (Aug. 21, 2019), p. 1066. ISSN: 2073-8994. DOI: 10.3390/sym11091066. URL: https://www.mdpi.com/2073-8994/11/9/1066 (visited on 03/11/2021).

[13] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In: *arXiv:1412.6980 [cs]* (Jan. 29, 2017). arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980 (visited on 02/24/2021).

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." In: *Communications of the ACM* 60.6 (May 24, 2017), pp. 84–90. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/3065386. URL: https://dl.acm.org/doi/10.1145/3065386 (visited on 03/08/2021).

[15] Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Z. Li. "Person Re-identification by Local Maximal Occurrence Representation and Metric Learning." In: *arXiv:1406.4216 [cs]* (May 6, 2015). arXiv: 1406.4216. URL: http://arxiv.org/abs/1406.4216 (visited on 10/22/2020).

[16] Stephen C-Y. Lu. "Machine learning approaches to knowledge synthesis and integration tasks for advanced engineering automation." In: *Computers in Industry* 15.1 (Jan. 1, 1990), pp. 105–120. ISSN: 0166-3615. DOI: 10.1016/0166-3615(90)90088-7. URL: https://www.sciencedirect.com/science/article/pii/0166361590900887 (visited on 03/10/2021).

[17] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE." In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[18] Abdullah-Al Nahid and Yinan Kong. "Involvement of Machine Learning for Breast Cancer Image Classification: A Survey." In: *Computational and Mathematical Methods in Medicine* 2017 (2017), pp. 1–29. ISSN: 1748-670X, 1748-6718. DOI: 10.1155/2017/3781951. URL: https://www.hindawi.com/journals/cmmm/2017/3781951/ (visited on 03/10/2021).

[19] Jeff Rasley, Yuxiong He, Feng Yan, Olatunji Ruwase, and Rodrigo Fonseca. "HyperDrive: exploring hyperparameters with POP scheduling." In: *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference*. Middleware '17: 18th International Middleware Conference. Las Vegas Nevada: ACM, Dec. 11, 2017, pp. 1–13. ISBN: 978-1-4503-4720-4. DOI: 10.1145/3135974.3135994. URL: https://dl.acm.org/doi/10.1145/3135974.3135994 (visited on 10/28/2020).

[20] Orod Razeghi and Guoping Qiu. "Object Recognition with Human in the Loop Intelligent Frameworks." In: *arXiv:1912.05575 [cs]* (Dec. 11, 2019). arXiv: 1912.05575. URL: http://arxiv.org/abs/1912.05575 (visited on 02/25/2021).

[21] Sascha Roth and Sibille Heine. *Das Geschäft mit dem Einweg-pfand*. Jan. 2017. URL: https://www.nabu.de/imperia/md/content/nabude/abfallpolitik/170207_nabu_infopapier_einwegpfand.pdf.

[22] Sumit Saha. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. Medium. Dec. 17, 2018. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (visited on 03/08/2021).

[23] A. L. Samuel. "Some Studies in Machine Learning Using the Game of Checkers." In: *IBM Journal of Research and Development* 3.3 (July 1959), pp. 210–229. ISSN: 0018-8646, 0018-8646. DOI: 10.1147/rd.33.0210. URL: http://ieeexplore.ieee.org/document/5392560/ (visited on 01/07/2021).

[24] Hillary Sanders and Joshua Saxe. "GARBAGE IN, GARBAGE OUT: HOW PURPORTEDLY GREAT ML MODELS CAN BE SCREWED UP BY BAD DATA." In: (July 27, 2017), p. 6.

[25] P. J. H Schoemaker. *Experiments on Decisions under Risk*. OCLC: 1059410782. Dordrecht: Springer Netherlands, 2013. ISBN: 978-94-017-5040-0. URL: https://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=5555560 (visited on 02/26/2021).

[26] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A Unified Embedding for Face Recognition and Clustering." In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 815–823. DOI: 10.1109/CVPR.2015.7298682. arXiv: 1503.03832. URL: http://arxiv.org/abs/1503.03832 (visited on 10/22/2020).

[27] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." In: *arXiv:1409.1556 [cs]* (Apr. 10, 2015). arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556 (visited on 03/08/2021).

[28] Marina Sokolova and Guy Lapalme. "A systematic analysis of performance measures for classification tasks." In: *Information Processing & Management* 45.4 (July 2009), pp. 427–437. ISSN: 03064573. DOI: 10.1016/j.ipm.2009.03.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0306457309000259 (visited on 02/18/2021).

[29] D. Sommerfeld. *Kuriose Zwitterwesen: Nanu, wie kommt das Kölsch in die Pils-Flasche?* Express.de. Mar. 13, 2010. URL: https://www.express.de/koeln/kuriose-zwitterwesen-nanu--wie-kommt-das-koelsch-in-die-pils-flasche--18065166 (visited on 11/12/2020).

[30] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going Deeper with Convolutions." In: *arXiv:1409.4842 [cs]* (Sept. 16, 2014). arXiv: 1409.4842. URL: http://arxiv.org/abs/1409.4842 (visited on 03/08/2021).

[31] Heinz Von Foerster and Bernhard Pörksen. *Wahrheit ist die Erfindung eines Lügners: Gespräche für Skeptiker*. Elfte Auflage. OCLC: 936800414. Heidelberg: Carl-Auer-Systeme-Verl, 2016. 167 pp. ISBN: 978-3-89670-646-1.

[32] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. "Distance Metric Learning for Large Margin Nearest Neighbor Classification." In: (2009), p. 8.

[33] Désirée White and Montserrat Rabago-Smith. "Genotype–phenotype associations and human eye color." In: *Journal of Human Genetics* 56.1 (Jan. 2011). Number: 1 Publisher: Nature Publishing Group, pp. 5–7. ISSN: 1435-232X. DOI: 10.1038/jhg.2010.126. URL: https://www.nature.com/articles/jhg2010126 (visited on 03/09/2021).

[34] Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. "Machine learning in manufacturing: advantages, challenges, and applications." In: *Production & Manufacturing Research* 4.1 (Jan. 1, 2016). Publisher: Taylor & Francis, pp. 23–45. ISSN: null. DOI: 10.1080/21693277.2016.1192517. URL: https://doi.org/10.1080/21693277.2016.1192517 (visited on 03/10/2021).

[35] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. "A Survey of Autonomous Driving: *Common Practices and Emerging Technologies*." In: *IEEE Access* 8 (2020), pp. 58443–58469. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2983149. URL: https://ieeexplore.ieee.org/document/9046805/ (visited on 03/10/2021).

[36] tagesschau.de. *Verpackungen in Gastronomie: Schulze plant Mehrweg-Pflicht*. tagesschau.de. Nov. 19, 2020. URL: https://www.tagesschau.de/wirtschaft/mehrweg-schulze-101.html (visited on 03/06/2021).