Doctoral theses at NTNU, 2024:130

Doctoral thesis

Eirik Lund Flogard

Improving Labour Inspection Efficiency via Machine Learning

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical
Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Eirik Lund Flogard

Improving Labour Inspection Efficiency via Machine Learning

Thesis for the Degree of Philosophiae Doctor

Trondheim, June 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ISBN 978-82-326-7854-9 (printed ver.)
ISBN 978-82-326-7853-2 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2024:130

Printed by NTNU Grafisk senter

Dedicated to my family

Abstract

Labour inspections are carried out nationwide by governmental agencies in countries that have ratified the International Labour Organization's Labour Inspection Convention (1947), to enforce decent working conditions and prevent injuries in workplaces. The inspections are conducted by individual inspectors, typically using checklists to survey inspected workplaces for non-compliance to health, environment, and safety-related regulations. Carrying out inspections efficiently is becoming increasingly more difficult, as workplaces are becoming more diversified and complex.

This thesis therefore investigates the potential for improving the efficiency of labour inspections via machine learning (ML). Current research into this topic is very limited, so we first investigate what kind of data and ML methods that could be used to support labour inspection tasks. The investigation also involves assessing different baseline ML methods for selecting workplaces for inspection, and for selecting relevant (predefined) labour inspection checklists. We also assess various feature selection methods to maximize the performance of the baselines. Although the initial results are promising, we found that it was difficult to achieve good prediction accuracy even for the best-performing methods.

We also propose ML methods for generating new checklists that could efficiently aid inspectors in identifying working environment violations. One of these methods can be used to generate dynamic checklists, which can be continuously adapted to any new information that surfaces during inspections. Our work also includes proposed explanation approaches to make the dynamic checklists more interpretable for inspectors. We then look further into how such ML-based checklists should be evaluated, by comparing the results from cross-validation performance estimates on existing data to the results from a field study where the checklists are tested in real-world labour inspections. The results of the comparison suggest that the cross-validation performance may not reflect the real-world field performance of the checklists. However, the results from the field study also show that ML-based dynamic checklists significantly increase the number of violations found in the inspections, improving inspection efficiency.

The overall results from this Ph.D. suggest a great potential for using ML in labour inspection tasks. Therefore, our work could promote more independent research on the topic.

Preface

This thesis is submitted to satisfy the requirements for the degree of Philosophiae Doctor in computer science at the Faculty of Information Technology and Electrical Engineering at the Norwegian University of Science and Technology. The Ph.D. project was funded by the Norwegian Labour Inspection Authority and the Norwegian Research Council, grant number 299928. The main supervisor for the Ph.D project was Ole Jakob Mengshoel. The co-supervisors were Kerstin Bach, Helge Langseth, and Heri Ramampiaro.

The thesis is based on a collection of papers and describes the research objectives for the Ph.D. It also describes how the papers relate to both each other and these objectives. The paper collection consists of five peer-reviewed articles, which have been reformatted for consistency and readability. Two of the papers are published in the most leading publication channels in artificial intelligence and machine learning, ranked at the highest level in the Norwegian Scientific Index. One paper has also received a best paper award.

Acknowledgements

I would like to thank my supervisor Ole Jakob Mengshoel for all his help during the Ph.D, especially for the valuable advises on writing papers for the top publication channels in artificial intelligence. Thanks to Kerstin Bach for helping out with her expertise on Case-Based Reasoning, in particular during the early stages of my Ph.D. studies. I would also like to thank the Department of Computer Science and its leader Heri Ramampiaro, especially for the extra efforts in facilitating decent working conditions during the pandemic.

I am also grateful to the Norwegian Labour Inspection Authority and my colleagues there for the collaboration and support that I received during my Ph.D. I would like to thank Ole Brevik, Marius Søberg, Ole Magnus Theisen, Jan Ingar Trangen, Tore Kristiansen, Cristina Krogh, Monica Seem, Vigdis Johnsen, and others for their support and involvements.

I would like to thank my family for their support and understanding throughout these years.

Contents

Nomenclature

Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
BCBR	Bayesian Case-Based Reasoning
BN	Bayesian Network
CBCBR	Context-aware Bayesian Case-Based Reasoning
CBR	Case-Based Reasoning
DAG	Directed Acyclic Graphs
FS	Feature Selection
HSE	Health, Environment and Safety
ILO	International Labour Organization
k-NN	*k*-Nearest Neighbour
KPI	Key Performance Indicator
LIME	Local Interpretable Model-Agnostic Explanations
ML	Machine Learning
NBI	Naive Bayesian Inference
NLIA	Norwegian Labour Inspection Authority
NLP	Natural Language Processing
OSH	Occupational Safety and Health
PGM	Probabilistic Graphical Model
SDG	Sustainable Development Goal
SHAP	Shapley Additive Explanations
SLS	Stochastic Local Search
SLS4FS	Stochastic Local Search for FS

Part I

THESIS

Chapter 1

Introduction

1.1 Machine Learning and Labour Inspections

Workplaces generally carry many risks to human health, safety, and environment (HSE). These risks vary from workplace to workplace, depending on many factors. These factors can include the industrial sector of the workplace, workplace culture and the geographical location [43]. To effectively address and mitigate these risks, labour inspection authorities carry out inspections among business and organisations on a large scale. These inspections are conducted in pursuant to the International Labour Organisation's (ILO) Labour Inspection Convention from 1947, and are important to enforce compliance to national and international labour standards and promote United Nation's sustainable development goal of decent work (SDG 8).

A simplified abstraction of what the inspection process can look like is shown in Figure 1.1.[1] The inspections are carried out differently depending on the target organisation, but common for all inspections is that the targeted organisations are surveyed for non-compliance with the working environment regulations. Any violations found in the inspections are cited individually in inspection reports. These reports are sent back to the inspected organisations, eventually with orders to rectify the violations. Further sanctions may be taken against organisations that fail to comply.

A challenge with labour inspections is that they are complex tasks and executing them can be time consuming. Labour inspection authorities therefore face difficult decision problems: How should labour inspections be carried out to maximize efficiency? In which organisations should inspection efforts be concentrated? What should the inspector look for in each individual target organisation? Since the goal of labour inspections is to ensure decent occupational health, safety, and environment, inspections should rectify as many HSE risks and violations as possible [14, 44]. However, doing so efficiently is becoming increasingly more difficult, and therefore the use of labour inspections as a regulation strategy has been declining in recent years [83, 119, 121]. One of the reasons for this is that workplaces are increasingly becoming more diversified and complex with significant variations in HSE risks [122], as illustrated by Figure 1.2. Labour legislation is also increasing in complexity and volume in many nations [118]. In recent years, researchers have searched for solutions to these problems in new strategic policies and principles for regulation enforcement [14, 121], and by increasing the understanding of HSE risks in certain occupations and contexts [20, 25, 99, 123]. However, there has been far less research into how technology could be utilized to improve labour inspection efficiency.

[1] In reality the labour inspection process can vary from nation to nation, and can also vary depending on the context.

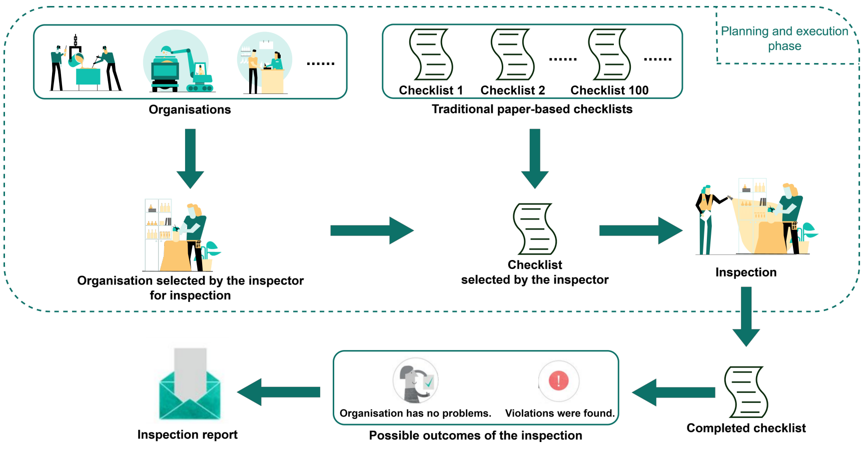

Planning and execution phase
Organisations
Checklist 1
Checklist 2
Checklist 100
Traditional paper-based checklists
Organisation selected by the inspector for inspection
Checklist selected by the inspector
Inspection
Completed checklist
Organisation has no problems.
Violations were found.
Possible outcomes of the inspection
Inspection report

Figure 1.1. A simplified conceptual overview of how labour inspections are planned and executed. In the planning phase, a target organisation is selected for an inspection. This could for instance be a factory, a construction company or a hair dresser. The inspector also selects the checklist (from many pre-defined checklists) that they consider to be most appropriate for their inspection target. The next phase is the execution of the inspection, which may be carried out differently depending on the selected target organisation and checklist. During the inspection, the targeted organisation is surveyed for non-compliance to working environment regulations. After the inspection (execution phase) is completed, any violations that were found are cited individually in an inspection report. These reports are sent back to the inspected organisations, eventually with orders to rectify the violations. Further sanctions may be taken against organisations that fail to comply.

One of the promising technologies that have not received much research is machine learning (ML), which has seen a significant increase in popularity over the recent years. Especially traditional areas of ML such as natural language processing (NLP) and computer vision have seen much progress and growth recently [94, 102]. ML is also being researched and used in many application domains to improve task efficiency, such as law enforcement [33, 68]. Therefore, this thesis aims to investigate how ML can be used to improve labour inspection efficiency.

1.2 Scope of the Thesis

This Ph.D.-project looks at how ML can be used to improve labour inspections, but is limited to the planning and execution phase in Figure 1.1. The project is based on data from inspections carried out by Norwegian authorities, but the research is likely relevant and can be used in other nations as well, by adapting or replacing the data. The project is also limited to the ML-technical aspects of improving labour inspections. Related topics, such as how labour inspections should be organised to function optimally (with or without ML), are considered to be outside the scope. More specifically, the project looks at how ML can be used to create or select checklists that are optimal for a specific context. The project also investigates the use of ML to predict non-compliance to HSE regulations among organisations, which could be utilized to

Figure 1.2. Different workplaces carry different risks to employee's health, environment, and safety. Employees at an industrial butchery or a construction company (top row) may for instance be exposed to physical injuries, while office workers and bartenders (bottom row) may have increased risks of certain muscle-skeletal disorders. The introduction of new technologies, machines, hazardous substances, processes and also new employment structures has made HSE risks more diversified and increased the complexity of labour inspections in recent years [118]. The photos are owned and distributed by the Norwegian Labour Inspection Authority.

select organisations for inspections [34, 43, 61]. It is also important to understand the data and processing methods that are necessary for these tasks, and these factors are also considered in this work.

1.3 Research Questions

This section provides an overview of the research goal and research questions for this thesis. The research goal represents the overall objective that the research seeks to address, and is listed below.

Research Goal

Understanding how ML can be used to improve labour inspection efficiency.

The scope of this research goal is large, as there are potentially many different ways to improve labour inspection efficiency via ML. Therefore, the goal is broken down into four research questions in order to focus the research efforts. The research questions below are also designed to reflect research contributions and scientific advances in both labour inspections and ML.

Research question 1

What kind of data and ML methods are effective for improving labour inspections?

The effectiveness of using ML in any task can depend heavily on the data and methods used, which is a motivation for the first research question. Research question 1 therefore relates to how ML can be optimized for labour inspection tasks. This is important considering that there are many different ML and data processing methods available.

Research question 2

Can ML be used to create optimized checklists for real-world tasks?

Research question 2 relates to the checklists that labour inspection authorities use for their inspections. The checklists are needed to survey individual organisations for non-compliance to working environment regulations [35, 44, 64]. One of our hypotheses in our research is that using ML to create new optimized checklists could also improve labour inspection efficiency (research goal), by increasing the number of violations found in the inspections. Since checklists are used for a wide range of different tasks in addition to labour inspections [21, 55], the research could also be relevant for other domains and therefore have a broader impact than only the labour inspection domain [44, 45]. Thus, the research question reflects that its scope could be considered to be wider. The research question also relies on the insights from Research question 1, regarding what data and ML methods should be used to create optimized checklists.

Research question 3

How should ML-based checklists be evaluated?

Research question 3 can be seen as an extension of Research question 2, since ML-generated checklists need to be evaluated in order to understand how they impact task performance [46]. Therefore, finding a reliable evaluation approach is important to establish whether ML-based checklists can improve labour inspection efficiency (research goal). Answering the research question may also be important in establishing good evaluation practices for future research on ML-based checklists in other applications, such as in medicine [77, 126].

Research question 4

How can we ensure that ML models for labour inspections are interpretable?

Research question 4 addresses the interpretability of ML models. Since this thesis considers ML models that are meant to be used in regulation enforcement, it is essential that they are interpretable [46, 110]. There are already many ways to ensure

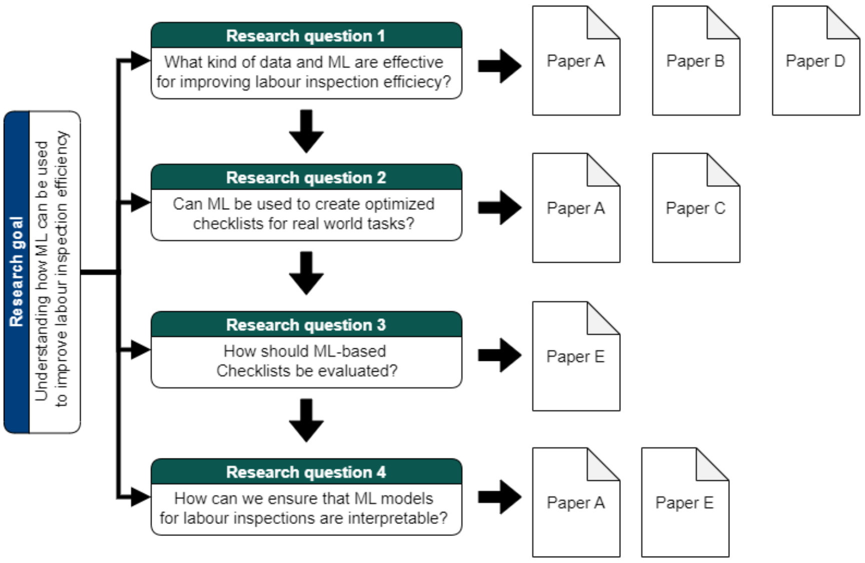

Research goal
Understanding how ML can be used to improve labour inspection efficiency
Research question 1
What kind of data and ML are effective for improving labour inspection efficiecy?
Research question 2
Can ML be used to create optimized checklists for real world tasks?
Research question 3
How should ML-based Checklists be evaluated?
Research question 4
How can we ensure that ML models for labour inspections are interpretable?
Paper A
Paper B
Paper D
Paper A
Paper C
Paper E
Paper A
Paper E

FIGURE 1.3. An overview of the research conducted as part of the Ph.D.-project. The research goal is broken down into four research questions. The arrows between the questions indicate that, to some extent, they build on each other's outcomes. Each of the research questions is covered by a set of papers, which are listed in the figure on the right-hand side. The papers are numbered from A to E in chronological order.

interpretability and provide explanations for ML models [36, 41, 66]. However, most of the existing methods yield detailed information that can be understandable by ML or domain experts, but not necessarily by other users. In particular, there is a lack of research into how ML models can be made interpretable to end users in the labour inspection domain, such as inspectors. Therefore, more research is needed on this topic.

1.4 RESEARCH OVERVIEW

The research conducted in this Ph.D.-project is presented as a collection of five papers that answer the research questions in Section 1.3. The papers are listed with letters from A-E and can be found in Part 2 of this book. The papers have been reformatted for readability and are also listed with the name and year of their publication channel, as well as the official Norwegian Scientific Index (NVI) for the channel. The index starts at level 0, which means that the publication channel does not satisfy the necessary standards for scientific quality according to the index. A publication channel listed as level 1 means that it satisfies the requirements and standards of scientific quality and that publications are sufficiently peer reviewed. Most conferences and journals are listed as level 1. Level 2 is the highest level, reserved for the most leading international publication channels that publish the most significant or ground-breaking papers within their respective fields of science. Thus, the standards of the peer reviews in level 2 channels tend to be stricter than level 1, and getting papers accepted into level 2 channels is therefore usually much more difficult.

Figure 1.3 shows an overview of the papers written as part of this Ph.D.-project, as well as how the papers relate to the research questions. A more detailed description of how the papers relate to the research questions follows below.

Paper A

Title: Bayesian Feature Construction for Case-Based Reasoning: Generating Good Checklists.
Published in: Proceedings of the International Conference on Case-Based Reasoning (ICCBR), 2021.
Norwegian Scientific Index: level 1.

Paper A addresses Research question 1 by introducing a new dataset from NLIA and testing a selection of baseline ML methods on a novel problem: constructing checklists that optimize the execution of specific tasks in real-world applications. The paper focuses on labour inspections, but the problem could potentially be adapted to other domains as well. Paper A addresses Research question 2 by introducing and formally defining the problem of constructing checklists, and by introducing a new ML method (BCBR) to attack it. The paper also addresses Research question 4 by focusing on transparency (white-box ML methods).

Paper B

Title: Stochastic Local Search Heuristics for Efficient Feature Selection: An Experimental Study.
Published in: Norsk IKT konferanse for forskning og utdanning (NIKT), 2021.
Norwegian Scientific Index: level 1.

Paper B addresses Research question 1 by introducing a new method for feature selection that performed well with ML models on datasets from different domains. Among these datasets is an NLIA-dataset for predicting the most relevant checklists for a particular labour inspection. A more refined version of this dataset is introduced in Paper D.

Paper C

Title: Creating Dynamic Checklists via Bayesian Case-Based Reasoning: Towards Decent Working Conditions for All.
Published in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2022.
Norwegian Scientific Index: level 2.

Paper C addresses Research question 2 by introducing a new ML method for generating dynamic checklists, which are adapted according to information that inspectors obtain during the inspections. The benefit of such checklists is that they can automatically adapt to changes in context throughout the inspections. The research could also be useful to other domains as well. Dynamic checklists are currently used in medicine [38, 39], but there are no readily available ML methods to create them [45].

Paper D

Title: A Dataset for Efforts Towards Achieving the Sustainable Goal of Safe Working Environments.
Published in: Advances in Neural Information Processing Systems (NeurIPS), 2022.
Norwegian Scientific Index: level 2.

Paper D addresses Research question 1 by introducing a dataset and two new ML problems. The paper addresses the research question by exploring different approaches for feature selection, and also by testing a varied selection of baseline ML methods on the two problems. The paper also discusses alternative definitions of the problems, demonstrates complexity, and motivates directions for future work.

Paper E

Title: Creating Explainable Dynamic Checklists via Machine Learning to Ensure Decent Working Environment for All: A Field Study with Labour Inspections.
Published in: Prestigious Applications of Intelligent Systems 2023.
Norwegian Scientific Index: level 1.
Note: Received best paper award.

Paper E addresses Research question 3, as it presents the results of a field study where the dynamic checklists generated by the ML method from Paper C (CBCBR) are evaluated. Inspectors from NLIA participated in the field study, carrying out inspections among Norwegian organisations. The paper also addresses Research question 4 by introducing new user-oriented methods for explaining the content of the dynamic checklists.

1.5 Thesis Structure

The rest of the thesis is structured as follows: Chapter 2 presents background information that is useful to understand the thesis. Chapter 3 summarizes related work by other authors. Chapter 4 presents and analyzes the thesis research results and explains in more detail how these relate to the research questions presented earlier in this section. A summary of the research contributions made by this thesis is also presented. Chapter 5 presents the conclusion of the thesis and potential directions for future work.

Chapter 2

Background

This chapter provides a brief introduction to some of the topics related to this thesis. The first topic is labour inspections and the use of checklists for such inspections. We then provide an overview and a general introduction to ML techniques and data processing, since a wide range of different baseline methods for addressing labour inspection tasks are explored in the thesis. We also give an introduction to interpretable ML methods, which we focused on investigating for this thesis.

2.1 Labour Inspections

Labour inspections are an essential part of a nation's labour administration system. The main goal of the inspections is to enforce labour laws and regulations among workplaces (organisations with employees). This can include employment-related conditions such as wages, working hours and child labour, and also general health, environment, and safety standards [118, 119]. One of the key foundations for modern labour inspections is tripartism, which means communications and interactions between the labour inspection authority, employers and workers [118]. This could for instance be consultations, negotiations or decision making and can happen on national level (government, workers' unions, and employers' unions) or at business level (inspector, employer, and workers). The goal of tripartism is to promote cooperation between parties and increase the legitimacy of labour inspections. Thus, the cooporation can make regulation enforcement easier for inspectors in the field [57, 118].

Labour inspections are usually carried out at individual workplaces with preventative a perspective, for instance by addressing potential risks of long or short-term injuries [44, 118, 119]. The way inspections are executed can vary, depending on the organisation of the labour inspection authority [118], the individual inspectors and the inspected organisations [46]. Inspections also tend to be industry-oriented. As shown in Table 2.1, there are many different areas of industries subjected to inspections. Each main industry area consists of many different industries and the inspection approaches vary accordingly. Some inspections can be physical in nature, where inspectors walk around workplaces and visually assess the conditions there. Other inspections can take place as a conversation in the office of the owner, leader, or management of the workplace. The contents and agendas of inspections can also vary. When inspectors find non-compliance to certain legal provisions in their inspections, an order to rectify the problem(s) is often given to the inspected business. Consulting and advising the inspected organisation on how to improve itself is also important to ensure that they have the ability to cooperate and make any necessary corrections. Thus, inspectors generally have to find a good balance between advisory and regulation enforcement [118].

Industry	***Industry main area code***	N
Agriculture, forestry and fishing	A	65 118
Mining and quarrying	B	1 683
Manufacturing	C	22 690
Electricity, gas, steam and air conditioning supply	D	2 085
Water supply	E	2 396
Construction	F	72 756
Wholesale and retail trade	G	73 135
Transportation and storage	H	30 940
Accommodation and food service activities	I	18 504
Information and communication	J	26 851
Financial and insurance activities	K	4 880
Real estate activities	L	64 813
Professional, scientific and technical activities	M	69 759
Administrative and support service activities	N	29 923
Public administration and defence	O	4 674
Education	P	21 184
Human health and social work activities	Q	58 689
Arts, entertainment and recreation	R	33 757
Other service activities	S	24 339

TABLE 2.1. An overview of industries within Norway. The far right column (N) shows the number of organisations within each industry. Data and numbers are retrieved from Statistics Norway [97]. The industry main area code represents the highest level of a four-level hierarchy. Thus, each industry main area consists of many different industries. The high number of industries and the number of organisations highlight the large scale, diversity, and complexity of labour inspections. Any organisation can also be subject to inspections, but thorough inspection of all of them is infeasible. Thus, inspection efforts tend to lean towards certain high-risk industry areas such as manufacturing, accommodation and food, or construction [46, 64]. Industries that are considered high-risk may also vary from nation to nation.

In some serious cases and usually as a last resort, the inspected business may be issued administrative penalties or fines [118].

Selecting organisations for inspections can be difficult, as inspectors seek to avoid workplaces that comply with the regulations of the working environment and focus on those that are non-compliant [120]. Therefore, overall inspection activities and efforts tend to lean toward certain industry areas that are considered high-risk, as shown in Table 2.1. Additionally, inspectors tend to select organisations for inspections randomly or based on their own knowledge about local businesses (*proactive*). Inspectors may also select inspection targets and carry out their inspections *reactively* based on certain criteria, such as public input or accident reports [118]. Sometimes, the inspection targets may also be selected randomly based on certain regulatory campaigns or activities against specific working environment problems or risks. An example of such activity in NLIA is inspections of workers' living quarters during the corona pandemic to ensure that they were compliant with the quarantine regulations. In addition to these "traditional" approaches, some labour inspection authorities have recently started developing ML models for selecting inspection targets [34]. Some of these ML approaches are described in more details in Section 3.1. The research into the use of ML to support other aspects of labour inspections is limited.

2.2 CHECKLISTS

Another challenge for labour inspections, in addition to deciding which organisations to inspect, is determining which regulations inspectors should check in each workplace. Each workplace that receives an inspection may be subject to hundreds of different regulations. However, going through all of them at once is not feasible [45]. For this reason, many labour inspection authorities use checklists as aids and reminders of what inspectors should go through during an inspection [21, 55]. Checklists are also widely used in many other high-stakes decision-making applications as well, such as food inspections, surgery or aviation [40, 55, 58].

The use of checklists for labour inspections faces some problems. One of them is that hundreds of different pre-defined checklists need to be maintained to cover the most relevant working environment risks in the different types of workplaces, which is currently done manually by domain experts. Some of the checklists are industry-specific [46, 64], intended to be used in certain industries (see Table 2.1). There are various formal approaches for creating and maintaining checklists such as the Delphi method [9, 28, 92], where a checklist is created, and then repeatedly presented to a panel of domain experts and modified as an iterative process. An overview of the approach is presented in Figure 2.1. There are many different variants of the Delphi method, as well as other similar application-specific approaches [7, 48, 107]. Common to all these methods is that they are time consuming, resource demanding, and the results rely heavily on the expertise and consensus of the domain experts [52]. The large number of checklists also makes them difficult to use, since inspectors have to manually find and determine the best match for their inspections [43].

Another problem is that checklists also tend to be limited to only a few specific thematic topics in terms of content (such as ergonomics or legal work contracts). As a consequence, the checklists do not necessarily cover all relevant risks in an inspected workplace very well. Thus, the use of checklists has been criticized by ILO for limiting

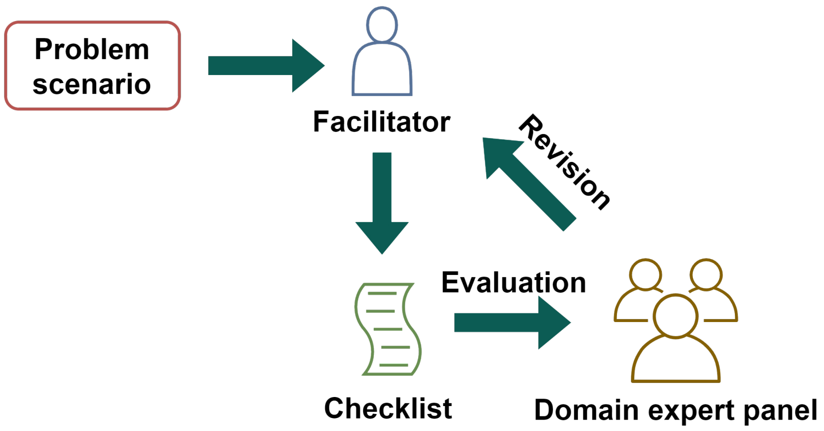


Figure 2.1. An overview of the Delphi method, which is a well-known approach for creating checklists. First, a facilitator (typically a domain expert) takes a problem scenario and creates a checklist. The checklist is then presented to a panel of domain experts who give an assessment of its content. The checklist is then revised via the facilitator and could then be presented to the panel again if necessary. Thus, the Delphi method is typically an iterative process, and the same approach can be used to revise or update existing checklists when needed. The current approach creating traditional labour inspection checklists is likely somewhat similar to the Delphi method but simpler and less formal, as the Delphi method is mostly used to create checklists for safety-critical tasks and can be very resource-demanding [52].

the scope of inspections, despite the advantages they may have as cognitive aids [101].

Addressing the above problems may require more research. Current research efforts on checklists are often case-studies that focus on analyzing and identifying advantages or disadvantages of using them [21, 55]. There has been little research into how checklists should be created, used and maintained in general, besides what is reported in this thesis and in medicine [38, 39].

2.3 Machine Learning Techniques

The general objective of ML is typically to train models that solve task(s) or problem(s), by learning from existing experiences (data). ML methods are usually described as supervised learning, unsupervised learning, or something in between these two. The main difference between them is whether or not the ML method relies on labeled data. This section gives a brief overview of the approaches.

2.3.1 Supervised Learning

Supervised learning methods are used to build models that generally rely on a dataset $\mathcal{D}$, where each instance $d_i \in \mathcal{D}$ is a tuple $d_i = (x_i, y_i)$ that consists of independent feature(s) x_i and a label y_i (dependent feature). For supervised learning, we wish to predict unobserved values of the dependent feature y, using the independent feature(s)

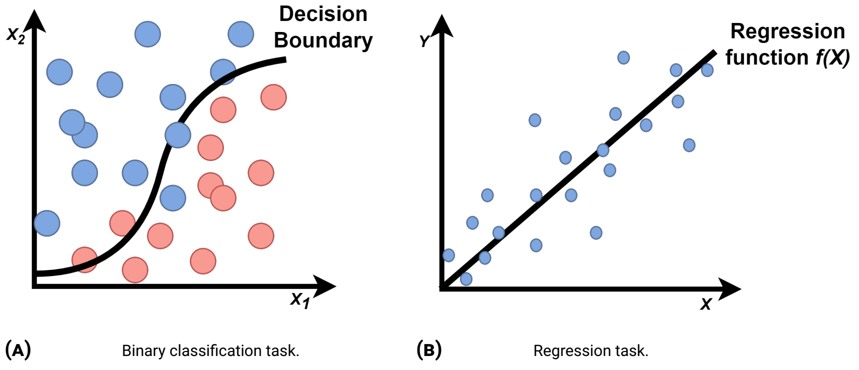


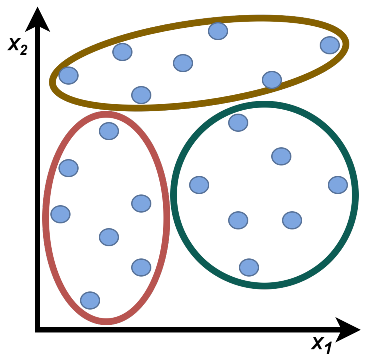
FIGURE 2.2. Two examples of supervised learning tasks; classification and regression. For the decision task, an ML model relies on a decision boundary to classify data-points (red/blue dots) as correctly as possible, given the input (X_1 and X_2). For the regression task, the ML model can simply be a function ($f(X)$) that approximates the value of the data-points (Y) as accurately as possible, given the input (X). Thus, the tasks are similar and the main difference between them is the representation of the predicted values.

x and an ML model. Training an ML model means that a method or algorithm is used to *learn* some map or function $f : X \longrightarrow Y$, used to obtain estimates $\hat{y} = f(x)$ of label values y. The approach for finding f varies, but most methods use some optimization process that aims to find a f that minimizes the difference between $\hat{y}_i = f(x_i)$ and the ground truth label y_i for every record $d_i = (x_i, y_i)$ in the training data $\mathcal{D}$. After finding f, the function can be used to predict the values of the ground truth labels for future observations of x.

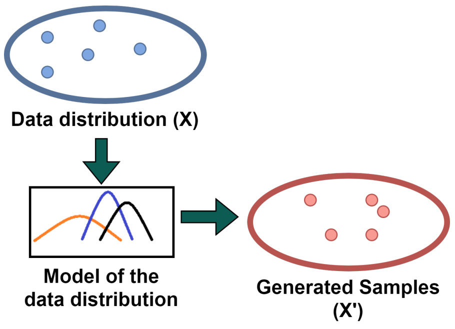
There are many different ML methods for supervised learning, depending on the task or problem to be solved. Figure 2.2 shows an illustration of binary classification and regression, which are examples of well-known ML tasks. Binary classification is typically associated with decision making, such as email spam classification or churn prediction. Regression is typically associated with predicting quantities, such as housing or stock prices. Since supervised learning tasks generally relies on labeled data, it is possible to develop ML models for very specifically defined purposes and goals.

2.3.2 Unsupervised Learning

In contrast to supervised learning, unsupervised learning is ML methods that do not use labeled data. While most supervised methods seek to learn to predict the value of annotated labels (y), unsupervised methods often seek to learn representations of input data for different purposes and tasks. One such task is clustering, as shown in Figure 2.3a. The goal of clustering is usually to group data such that the data-points in each group are more similar to each other than to data-points in other groups. Thus, the clusters themselves become a simpler representation of the data-points within. Popular clustering methods include K-means clustering, hierarchical clustering, den-



(A) Clustering task.



(B) Generation task.

FIGURE 2.3. Two conceptual examples of unsupervised learning tasks; clustering and generation. In the clustering task, the goal is to learn a set of optimal representations (colored rings) for some data points (blue dots). These can then be used to represent both existing and new data points in a simplified manner. In the generation task, the goal is also to learn a representation of data points (X). The learned representation is then used to generate samples X' that have distributions similar to X, so that any data points in X' appear as if they could belong to X.

sity based clustering, and more. Clustering can be used for dimensionality reduction (see Section 2.3.4), and also to model similarity in tasks such as anomaly detection [2]. Another popular unsupervised learning task is generation, as shown in Figure 2.3b. This also involves learning a representation of data, but the goal here is to generate new data-points by sampling from the representation. Popular methods include Bayesian Networks (sampling from the underlying joint probability distribution), Variational Autoencoders, and Generative Adverserial Networks [32, 117, 124]. Generative methods can also be regarded as supervised learning, if they use labeled data for learning representations.

2.3.3 *Other Techniques*

There may not always be a clear distinction between supervised and unsupervised learning. For instance, methods that rely on self-supervised learning can be regarded as both supervised and unsupervised. In self-supervised learning, ML is used to generate synthetic labels from an unlabeled dataset [78]. The labeled data can then be used for supervised learning. Thus, in self-supervised learning the task is converted from being unsupervised to being supervised. Reinforcement learning is also an example of ML that does not necessarily fit the description of either supervised or non-supervised learning. In reinforcement learning, an agent is tasked to select a certain set of actions (policy) in an environment in a way that maximizes a reward function [10]. One of the problems in reinforcement learning is that the rewards of future actions are usually not known to a certain extent in advance. ML is therefore used to find out which actions that lead to the highest rewards, based on past experiences. Neither self-supervised learning nor reinforcement learning are a part of the research in this Ph.D.-project, but they are still mentioned as they are important areas of ML in general.

2.3.4 Data Acquirement and Processing

In general, the quality of any ML model is heavily dependent on the data it is trained on. Acquiring the necessary data for ML can sometimes be tedious manual work, and a dataset often has to be integrated from many different sources [31]. In some cases, data can also be collected by "scraping" information from various sources such as social media or news agencies. This can be a less labour intensive process, but the quality of such data may vary and can for example contain racial bias or false information [69]. There may also be certain legal restrictions on what or how data can be used, such as privacy or copyrights. Using or expanding existing datasets may therefore generally be a good solution. In this sense, working with labour inspection tasks is an advantage, as data is usually generated and quality-assured as a part of the case management process in labour inspection authorities [6, 43].

Before using data for ML, it is usually necessary to perform cleaning operations by removing noise and inconsistent elements. It may also be necessary to normalize the data or address class imbalance. Such operations are often done directly on the data by for instance re-balancing the dataset [6], but can sometimes also be implemented by making corrections to the ML models [43]. Selecting and extracting features for the ML model may also be relevant to reduce potential noise, redundancy, or irrelevant information from a dataset [73, 87]. This process is often referred to as dimensionality reduction. Feature selection methods are often divided into different types of approaches. One of them is filter methods, which involve assessing feature importance and filtering out the least important features [73]. Examples of filter methods include χ^2 and ANOVA [88, 89]. These methods are typically relatively computationally inexpensive, but may not be as effective compared to other types of approaches. Another approach is wrapper-based methods, which rely on searching for an optimal subset of features. A drawback with this approach is that it can be computationally expensive, since each candidate feature subset (search step) is evaluated on a classifier that needs to be trained on the subset [43, 73]. Some ML methods may also have "built-in" feature selection as a part of the model training, such as Decision Trees or many Artificial Neural Networks variants (ANN). These feature selection approaches are often referred to as embedded feature selection methods [73]. However, they may still benefit from additional pre-training feature selection in terms of lower computational costs and slightly higher model accuracy [88, 89].

2.4 Interpretable Machine Learning Methods

A potential problem with many ML methods is that the predictions they make can be difficult to understand. Many methods such as deep learning (ANN), random forests, or boosting methods tend to be called "black-box" methods, because their size, complexity, low-level nature, and "hidden layers" (in ANNs) effectively hide their internal logic [15]. The lack of transparency in these models makes them difficult to interpret and explain, and such methods tend to rely on post-hoc explanations to

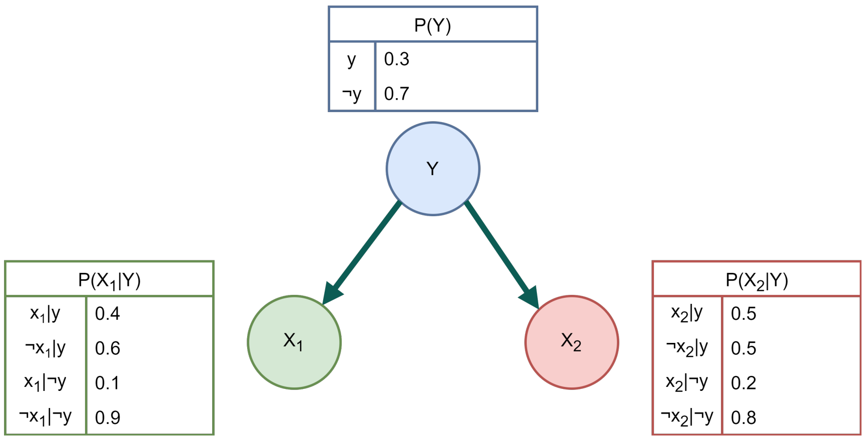


FIGURE 2.4. A small and simple Bayesian network (BN) with binary variables Y, X_1 and X_2. This particular network can also be seen as a Naive Bayes' Classifier. The parameters for the network are listed in the tables next to the graph. The symbols x_1, x_2 and y are short-hand notations for the events $X_1 = 1, X_2 = 1$ and $Y = 1$, respectively. Collectively, the tables form an underlying joint probability distribution that can be used for querying. For example, it is possible to find $(y|x_1, x_2)$ by using the tables to solve $P(y|x_1, x_2) = \frac{P(x_1,x_2|y)P(y)}{p(x_1,x_2)}$. Thus, BNs are very flexible ML model representations.

achieve some degree of interpretability [110]. In contrast, "white-box" methods are transparent and can provide a better understanding of how predictions are made, since it is feasible to track the logic behind predictions that the methods make [111]. One of the motivations for using black-box methods is often prediction accuracy. However, the lack of interpretability can negatively impact accuracy due to a lack of trust from users [111]. In many cases, white-box methods may also be more accurate than black-box methods, despite the lower model complexities [110]. For governmental agencies, especially regulators such as labour inspectorates, making transparent decisions is vital for maintaining trustworthiness. Forthcoming EU-regulations will also require more use of interpretable ML models [29].

In this Ph.D.-project we therefore investigated interpretable (and relatively transparent) ML methods for solving labour inspection tasks. The rest of this section discusses the most important areas we looked into.

2.4.1 Probabilistic Machine Learning

Probabilistic ML methods are used to create models that represent uncertainty, relying on probability distributions and probability theory to make interpretable predictions about future data [37, 49]. A commonly used probabilistic method for ML is Generalized Linear Models (GLMs), which is a set of (inverse) linear models for predicting the mean values on probability distributions from the exponential family [13, 93]. Both linear and logistic regression are examples of GLM models.

Bayesian Network (BN) is another popular probabilistic method for addressing uncertainty with many application areas, such as fault diagnosis, medical diagnosis, circuits modeling, and sensor validation [24, 37, 85, 86, 109]. The method is typically used in classification tasks and can also handle incomplete input data [37]. Formally, BNs are directed acyclic graphs (DAG) with nodes and edges. The nodes represent variables (features), while the edges represent dependencies between the variables. The variables in the network represent a joint probability distribution, with independencies defined via the edges. Inference on the network is done by applying probability calculus to the joint distribution, such as conditioning or marginalization. An example of a BN is shown in Figure 2.4. The figure shows a Naive Bayes' Classifier, which is a popular variant of BNs for predicting or classifying the cause (dependent variable) of observed effects (independent variables). The method is called "naive" because it is naively assumed that all independent variables are conditionally independent, given the dependent variable.

Creating and learning graph structures (DAGs) for BNs can be difficult and also very computationally expensive, and it is an offline process in practice [113]. Finding and accurately estimating the parameters for the underlying probability distributions (conditional probability tables) of BNs can also be difficult, depending on whether the training dataset is complete or not [16, 59].

BNs can be considered to be relatively interpretable compared to black-box models such as ANNs. Since reasoning in BNs is based on probability calculus, inferences or predictions from such models can usually be traced and explained easily. Although BNs can be relatively large [30, 109], they are typically much smaller than ANNs in terms of number of nodes and edges. Nodes in BNs also often represent real-world phenomena, unlike the hidden nodes in an ANN.

2.4.2 Case-Based Reasoning

Case-Based Reasoning (CBR) is a methodology used to solve many different problems and tasks in ML, especially decision support for human users [1, 116]. Case-based reasoning is used, for example, in recommendation systems [8, 62, 115] and for making decisions in medical applications [11, 42]. The core idea in CBR is to solve (new) problems by retrieving and reusing past experiences. This idea is different from many other modern ML methods, which are typically based on learning models or functions of data to solve new problems.

Figure 2.5 shows a generic overview of the CBR process, in terms of how the methodology can be used to solve problems and learn from experiences. Central to the CBR process is a case base that contains cases, which are representations of past experiences. The CBR process can be broken down into four phases: *retrieve* the most similar case(s) to the input problem, *reuse* the retrieved case on the input problem, *revise* the case to solve the problem better, and finally *retain* the solved case. A brief discussion of each of the four phases follows below.

1. **Retrieve.** The first phase consists of retrieving a past case that is similar to the input case, shown in Figure 2.5. This is usually done by creating an input case that contains the description of the problem that needs to be solved, and then retrieving the past case(s) with the most similar description. The implementation of the retrieval phase varies. In some CBR methods, ML models are used

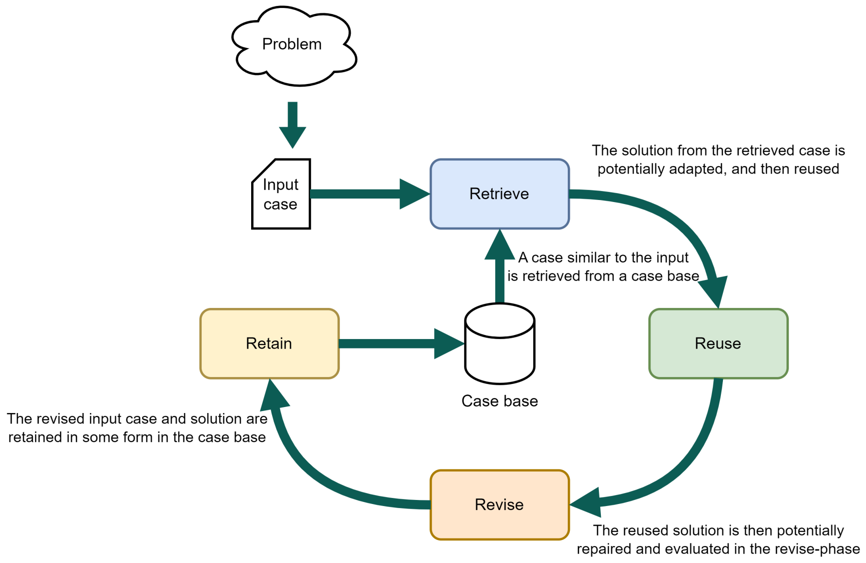

Problem
Input case
Retrieve
The solution from the retrieved case is potentially adapted, and then reused
A case similar to the input is retrieved from a case base
Retain
Case base
Reuse
The revised input case and solution are retained in some form in the case base
Revise
The reused solution is then potentially repaired and evaluated in the revise-phase

FIGURE 2.5. The figure is based on an illustration by Aamodt et Plaza. [1], and shows a conceptual overview of the CBR methodology. CBR can be broken down into four phases: retrieve, reuse, revise, and retain.

to perform or enhance case retrievals, improving the match between input and retrieved cases [96].

2. **Reuse.** The retrieved case(s) usually consists of a problem description-part and a solution-part. Thus, the reuse phase consists of taking the solution-part of the retrieved case and applying it to the input problem. This is often done by simply copying the solution from the retrieved case, and then applying it [1, 72]. However, more elaborate approaches also exist. Some CBR methods can derive a single solution from multiple cases [100] or modify the retrieved solution via an ML model before it is used [72, 74, 125].

3. **Revise.** After the solution has been applied to the problem in the real-world environment (reused), it is evaluated. Typically, feedback or an indication of whether the solution was successful or not is given, such as a ground truth label [44, 96]. If the solution was successful, it may be passed on for retention. If the solution fails or does not work sufficiently to solve its problem, a repair operation on the solution may be performed in addition to the evaluation [1]. This operation typically modifies the failed solution to better fit the problem, in order to prevent future solution failures. Thus, one of the purposes of the revision phase is to facilitate learning in the CBR system from both successful or failed solutions.

4. **Retain.** In the last phase, the reused and revised solution and input case are stored in the case-base, as shown in Figure 2.5. This means that the case can be used to solve future problems. The implementation of the retain-phase usually depends on the representation of cases in the case-base, and may therefore vary.

The simplest implementation is to just retain every case indiscriminately, which can include cases of successful and failed or repaired solutions [1]. Some CBR methods also retain exclusively successful solutions [80]. Other methods may rely on the construction of generalized cases for their retain-phases, where information in the new case can be integrated into one of the existing cases [1].

One of the biggest advantages with CBR-based methods is that most of them are transparent and interpretable (white-box) [65]. This is because each phase of the CBR cycle is usually implemented with logic or algorithms that are relatively easy to trace, which is crucial for systems used for decision support [1]. For example, case-retrieval is often implemented as a variant of k-Nearest Neighbours (k-NN), retrieving cases based on a distance function [82]. CBR also provides intuitive example-based reasoning and has a long history of implementations as explanatory frameworks, where retrieved cases with predicted solutions are presented along with explanations to users [1, 65, 116]. The implementation of such explanations may vary depending on the goals they have [116], and what task or problem the CBR method addresses. A widely used approach is to present information from retrieved cases so that users can compare them with input cases, which can justify the solutions in the retrieved cases [46, 98]. A closely related technique is to also display cases that are similar to the retrieved case, and similar cases with solutions that contradict the retrieved case [71, 116]. CBR can also be used to provide post-hoc example-based explanations for black-box methods, such as ANNs [65].

Chapter 3

Related Work

This chapter is intended to summarize research related to some of the most important research topics for the Ph.D.-project. These are meant to provide motivation for the research conducted as part of the Ph.D.

3.1 Machine Learning in Labour Inspections

Despite the growing popularity of ML, the publicly available research on the use of ML for labour inspection tasks is limited. In general, there is also a lack of publicly available data to support research and development of ML models for labour inspections [43]. Yet, some ML models for predicting inspection targets have been proposed [43]. One of them is an ensemble-based model called Super Learner [61]. The model predicts the number of injuries from work accidents that can be averted for businesses that receive inspections [61]. The businesses with the highest predicted numbers are typically selected for inspections. A drawback with the approach is that the data the model relies on may not be reliably used for this purpose due to bias and under-reporting of accidents and injuries [79, 106]. NLIA has also developed its own ML model for predicting inspection targets using logistic regression [34]. Their model predicts the probability for finding serious working environment violations instead of the number of injuries. This is similar to one of the ML tasks addressed in Paper D [43]. ML models for carrying out inspections addressing undeclared work have also been proposed using association rule mining and associative classification [4–6].

There is also some research on the use of ML models to understand or predict specific HSE or OSH related hazards or problems, such as hearing impairment [76, 128]. However, the scope of such models may be too narrow to be practically useful for labour inspections [34]. Thus, there is not much publicly available research on the use of ML for labour inspection tasks, besides this Ph.D.-project and the ML models for prioritizing targets for labour inspections. Therefore, the research carried out in this Ph.D. could potentially generate more interest in the topic from independent researchers.

3.2 Approaches to Checklists

There are many different ways to create and use checklists, as they can have different objectives and purposes [21]. Figure 3.1 shows a simple classification of approaches to checklists based on how they are created (manual versus ML-based creation) and

	Static	***Dynamic***
Manually created	Checklists are created manually by domain experts. Refered to as "traditional checklists" in this thesis. Used in medicine, aviation, food inspections, labour inspections and more [21, 35, 55, 58].	Checklists are created manually by domain experts. Dynamic adaptations to the checklists are made via manually defined logic or models. Used for surgeries and to provide guidelines for creative tasks [12, 39, 70].
ML-based	Checklists are created via ML. Used for medical diagnosis and construction site quality inspections [19, 77, 126]. Limited research besides this Ph.D.-project [44].	Both creation and dynamic adaptations to the checklists are done via ML. No known use-cases or research besides this Ph.D-project [45, 46].

Table 3.1. An overview of different approaches to checklists. Checklists can be created manually by domain experts or through ML and other data-driven approaches. Checklists can also be static or dynamic. Static checklists typically remain fixed in size and content after they have been created. Any changes to them are made manually. Dynamic checklists are checklists that can automatically adapt and change in size and content during use.

whether they can be automatically adapted after creation (static versus dynamic checklists). Some examples of what the checklists in each category are used for are listed in the figure. This Ph.D.-project investigates the potential for static or dynamic ML-based checklists, particularly for labour inspections. Thus, a more detailed discussion of the related work on ML-based as well as dynamic checklists follows.[1]

3.2.1 Machine Learning Based Checklists

One of the research questions for this Ph.D.-project is whether ML can be used to create optimized checklists for real-world tasks, since creating them manually can be difficult and time-consuming [44, 45]. A generic illustration of how ML can be used to generate checklists is shown in Figure 3.1, but the architectural and functional details of the ML methods vary across the application domains. Zhang et al. propose an ML method for generating predictive checklists for medical diagnosis [126]. They define a medical diagnosis checklist as a binary M-of-N decision problem. That is, a certain diagnosis is predicted as positive if at least M of the N items on the checklist (symptoms) are checked as positive. If less than M items are checked, the diagnosis is predicted to be negative. An integer program is used to select a set of N items for an optimal checklist

[1]The background on traditional static checklists in context of labour inspections is discussed in Section 2.2.

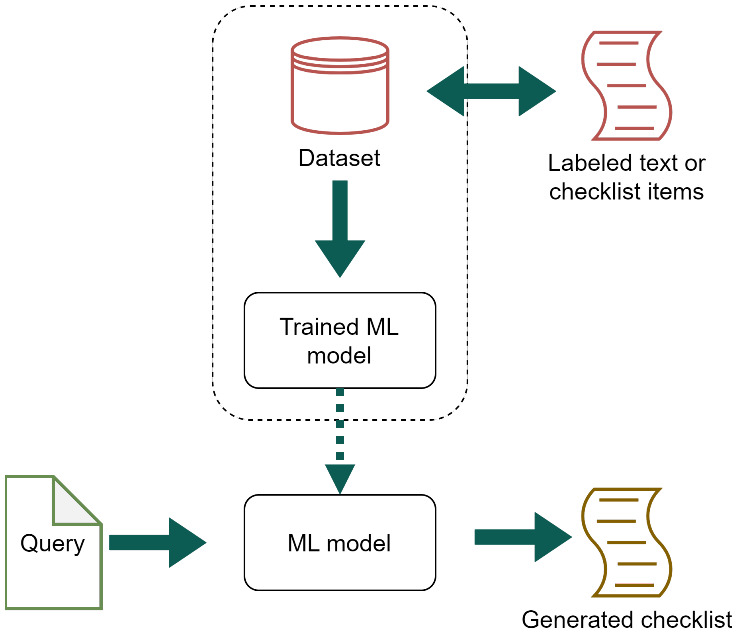


FIGURE 3.1. A conceptual overview of how ML can be used to generate checklists. First, an ML model for generating checklists is trained on a dataset that contains some form of labeled text or checklist items. The objective of the model training is to learn how to assemble optimal checklists for any given criteria or settings, based on the labeled components in the data. The model can then be queried after training, generating new checklists that match the queries. Checklists can also be generated using optimization methods (search) instead of a trained ML model [60, 126].

that most accurately predicts a diagnosis when at least M items are checked. Various extensions of the method have also been proposed [60, 77]. However, a drawback with this approach is that it relies heavily on the fact that a checklist represents a binary M-of-N decision problem. This is likely not the case for applications outside of medical diagnosis, where each individual checklist item may represent a separate decision [45]. Another drawback is that the method is computationally expensive, as the problem definition can be seen as a variant of a knapsack problem. There are also no case studies that evaluate how well the checklists generated by their method perform in the real world.

Cai et al. propose an approach for generating checklist items for quality assurance inspections on construction sites [19]. The approach uses NLP to generate checklist items from regulatory texts. Checklists are then assembled through SQL queries generated via a user interface. The checklists can then be manually customized by users after creation via the interface. The approach could be useful for applications where checklists need to be generated from scratch and there are no relevant data on existing checklists that could be used [45]. However, the authors do not present any experimental evaluations and the approach may be less relevant for labour inspections, since there are available data on checklist items [44]. Also, a problem with NLP models

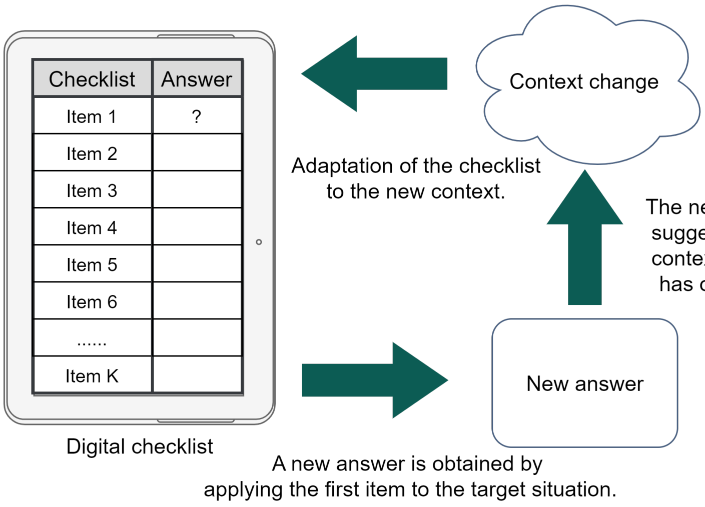

Checklist
Answer
Item 1
?
Item 2
Item 3
Item 4
Item 5
Item 6
......
Item K
Digital checklist
Context change
Adaptation of the checklist to the new context.
The new answer suggests that a context change has occurred.
New answer
A new answer is obtained by applying the first item to the target situation.

FIGURE 3.2. The figure [45] illustrates how dynamic checklists can adapt to changes in environment or context during use. Such adaptations to the checklists can increase their relevance and efficiency in aiding users for their intended tasks. For example, checklist answers can be used to automatically detect that something goes wrong during a surgery (context change), and use this information to adapt the checklist accordingly [38]. This can reduce cognitive load on the surgeons and improve checklist compliance and patient safety [39].

is that they are usually black boxes and it can therefore be difficult for domain experts to make any corrections or adjustments to them. Another problem is ensuring that the generated texts are factually correct and truthful because NLP models potentially use information from many different independent sources [22, 127].

In summary, checklists are being used for many different applications, but research into using ML for generating or optimizing checklist content is limited. The ML methods discussed above only generate static checklists, and none of them are used to adapt or improve the checklists while they are being used [45, 46].

3.2.2 Dynamic Checklists

Dynamic checklists are checklists that can adapt to changes in context during use [38], and they are used in medicine for safety-critical tasks such as surgeries and intensive care [39, 70]. A conceptual view of a dynamic checklist is shown in Figure 3.2, showing how such checklists can be adapted during use based on how they are answered. An advantage of dynamic checklists is that they may remain relevant under unexpected events or circumstances, while static checklists may need to be replaced or manually adjusted [112]. A variety of different designs of dynamic checklists in digital formats have been proposed. One of the simplest designs is to use simple logic to hide, reveal, or automatically check off checklist items [12, 112]. A similar design that relies on a rule-based decision model has also been proposed for intensive care units (surgeries) [38].

A process model has also been proposed and tested in different clinical settings [26]. Most of these designs also claim to improve checklist compliance and reduce errors. Dynamic checklists are also proposed to manage the complexity of design guidelines in creative tasks, by progressively removing or hiding irrelevant checklist items [12].

Dynamic checklists may also be useful for labour inspections, since the inspection approach can often (dynamically) change over time based on the information obtained [45]. For the same reason, they may also be relevant for similar regulation enforcement tasks such as food inspections [55]. However, models for traditional dynamic checklists are created manually, which can be time-consuming [39, 70]. For diverse and complex tasks, such as labour inspections, it is practically infeasible to manually capture and model every possible unfolding inspection scenario [45, 70]. This effectively limits the use of dynamic checklists to narrow and repetitive tasks that are performed in relatively controlled environments. This underscores the need for research on ML and other data-driven methods for generating dynamic checklists.

3.3 BAYESIAN CASE-BASED REASONING

The ML methods that we proposed for generating checklists in this Ph.D.-project are based on Bayesian Case-Based Reasoning (CBR). There are also other methods that combine Bayesian models or methods with CBR. One of them is a framework for prototype classification and clustering [67]. The model learns prototype observations (cases) that represent clusters of a dataset, by performing Gibbs' sampling on cluster labels, prototypes, and most important cluster features. The method has relatively good performance on multi-class classification tasks such as recognition of hand-written digits. Another method uses a combination of BNs and CBR as a two-stage model to perform user profiling, where CBR is used to store and retrieve cases for calculating probability values (parameters) of the BNs [114]. A similar method has been proposed for tasks that rely on large datasets, as a way to improve the classification accuracy for BN models [51]. There is also an approach that aims to improve the accuracy of CBR case retrievals by using posterior probability distributions to modify or add features to query cases [96]. The same approach can also be used to provide explanations for case failures in various applications [95]. Combinations of Bayesian methods and CBR have also been proposed to improve case-retrieval, and also to provide explanations in other applications as well [50, 66].

A common motivation for most of the methods discussed above seems to be that the combination of Bayesian methods and CBR could improve the handling of uncertainty or incompleteness in data [17]. While there are many methods that rely on Bayesian Case-Based Reasoning, they are usually specifically designed for many different tasks and purposes. The methods are therefore not related to each other, besides that they combine Bayesian methods with CBR. Further research into the topic can be motivated by potential improvements in ML performance on certain tasks. Especially for tasks where ML model interpretability is necessary, such as creating checklists [44, 46]. Another motivation to investigate Bayesian Case-Based Reasoning for creating checklists is that labour inspection data can be subjected to aleatoric or stochastic uncertainty, due to a large number of various factors and complex relations in the real world that are infeasible to capture sufficiently in the data [44, 46, 56].

Chapter 4

Research Results

This chapter provides an overview of the research conducted in this Ph.D.-project, outlining how the research questions from Chapter 1 are answered by the results from the research. Some of the limitations of the research contributions are also discussed here.

4.1 Summary of Research Contributions

This section summarizes the research contributions from this thesis, which are broken down based on the four research questions from Chapter 1. The research contributions are discussed in more details in Section 4.2.

Research question 1

What kind of data and ML methods are effective for improving labour inspections?

- Paper A introduces a new dataset for generating labour inspection checklists via ML.
- Paper B shows that feature selection can increase ML performance on the task of selecting a relevant checklist for an inspection in a given target organisation.
- Paper D introduces a new dataset that supports two novel tasks. (1) predicting an optimal relevant checklist that inspectors can use for a given inspection, and (2) predicting non-compliance to working environment regulations among organisations. The dataset could be useful for future ML-research in these tasks.

Research question 2

Can ML be used to create optimized checklists for real-world tasks?

- Paper A formally defines the problem of creating optimized checklists for a certain task and introduces an ML method to solve the problem, focusing on labour inspections.
- Paper C formally defines the problem of creating dynamic checklists and introduces an ML method to do so, based on the method from Paper A. Dynamic checklists are also used in medicine [26, 39, 70], but this is the first ever ML-based approach for generating such checklists [45].

Research question 3

How should ML-based checklists be evaluated?

- Paper A and Paper C propose to evaluate ML-based checklists based on cross-validation on estimated ground truth labels based on empirical distributions.
- Paper E presents the results from a labour inspection field study where the dynamic checklists from Paper C are evaluated. The results from the field study indicate that the evaluation approach from Paper A and C is somewhat optimistic, even when only true ground truth labels are considered in the experiment.
- Paper E also shows that dynamic checklists increase the number of violations found in the inspection, compared to traditional static checklists used in labour inspection. The results therefore indicate that using dynamic checklists increases labour inspection efficiency, since there were also fewer items on these checklists.

Research question 4

How can we ensure that ML models for labour inspections are interpretable?

- The ML method (BCBR) introduced in Paper A is an interpretable "white-box" method. The paper also discusses the importance of this property, and focuses on using interpretable ML methods as baselines in the experiments.
- Paper E introduces two new approaches for explaining the content of dynamic checklists (generated by CBCBR from Paper C) to their end-users (inspectors). The explanation methods are implemented in a prototype used for the field study described in the paper.

4.2 Research Results and Discussion

The research results and contributions from this thesis are presented and discussed here. The research is described in full detail in the papers (A-E), included in Part II. The research results are broken down based on the research questions from Chapter 1, in the same fashion as in the summary above.

4.2.1 *Research Question 1 - What Kind of Data and ML Methods are Effective for Improving Labour Inspections?*

The first research question concerns the exploration of useful datasets and ML methods for labour inspection tasks, taking into consideration the ML research on the topic besides this Ph.D.-project is limited (see Section 3.1). Due to the poor public availability of datasets, the knowledge of what kind of ML methods that can support such tasks is also limited. Many labour inspection authorities, such as NLIA, collect data as part of their daily operations [6]. Therefore, we looked into how existing data could be leveraged to support tasks that are related to planning and executing labour inspections. In particular, Paper A investigates how data on checklists used in past inspections can be used to create new checklists that could enable inspectors to carry out their inspections more efficiently. To support and promote further research on this topic, a dataset has been published. The paper also explores a set of baseline ML-methods in a preliminary experiment, to find out which baseline methods work most effectively for the task. The experiment answers the research question and serves as a stepping stone for Research question 2. More contents and contributions from Paper A are discussed in more details in Section 4.2.2.

Paper D also addresses Research question 1 by introducing a new dataset, supporting two new ML problems for the labour inspection domain. Solving these problems is currently an integral part of the workflow for planning and executing inspections in NLIA and other labour inspection agencies. The first problem is to select an optimal relevant checklist (among existing, predefined checklists) to use for an inspection. The second problem is to predict non-compliance to working environment regulations among businesses, which could be potential inspection targets. The paper identifies and discusses some of the key independent features in the dataset for the two problems. The paper also addresses some data processing issues, such as how the imbalanced distribution of target labels in training and evaluation data should be addressed. We also tested different baseline ML-methods for solving the two problems. Although the Ph.D.-project mainly focuses on interpretable methods, it may be useful for future research to also consider black-box baselines. Therefore, we tested both interpretable baselines such as k-NN and regression methods, and black-box methods like AdaBoost and Multi-layer perceptron (ANN). We identify Decision trees and AdaBoost as the best-performing methods for solving the first and second problem, respectively. This shows that black-box methods do not always outperform white-box methods with much less complexity [110, 111].[1]

We also test some feature selection methods in paper D, where the results indicate that filter-based methods such as Chi2 and Anova F work well. Paper B looks further

[1]The AdaBoost-baseline combines many decision trees models into a single predictive model.

into the feature selection problem using an earlier version of the dataset from Paper D. In the paper, we propose a new feature selection method (SLS4FS) based on Stochastic Local Search (SLS). SLS an algorithm that performs a greedy search with stochastically added noise and restarts [90]. One of the advantages with SLS4FS is that it can potentially search for optimal feature sets more effectively than linear search methods like forward or backward selection, without getting stuck in local optima. The experiments show promising results for SLS4FS configurations using fixed or adaptive noise in terms of measured accuracy and number of features found. This can be both in terms of increased model accuracy and reduced model training time and complexity as a result of fewer features [88, 89]. We also used Markov chains to analyse the computational costs of SLS in a different paper, not included as a part of this thesis [90]. Thus, both paper B and D contribute to Research question 1 by showing how feature selection can be used to improve ML model performance for labour inspection tasks. The papers also briefly discuss the top selected features for the tasks, giving an insight into what kind of features are useful in ML models for solving them. The highest ranked features vary depending on feature selection method and task but tend to include industry codes, business location, business age, number of employees, and financial data such as unpaid public fees, revenues, costs, and assets [43].

4.2.2 Research Question 2 - Can ML be Used to Create Optimized Checklists for Real-World Tasks?

The research goal of this thesis is to understand how ML can be used to improve labour inspection efficiency. Therefore, the Ph.D.-project looks into the possibility of using ML to create improved labour inspection checklists, since they implicitly govern the inspection procedures (what the inspectors should do in the inspections). The research also has a broader impact potential than just labour inspections, as checklists are commonly used for many different tasks in other domains such as healthcare and safety applications [38, 40, 55, 58]. At the beginning of this Ph.D.-project in 2019, there was no available research on the use of ML to create checklists in general.

Paper A formally defines the problem of constructing checklists, enabling the use of ML to solve it. This can be viewed as a type of generation task (see Section 2.3.2), which involves the generation of new optimized checklists for any given target organisation. The problem is to select the best possible set of K items, from N available items (small K, large N), that maximizes the number of violations (non-compliance) found at a given inspection target. To address the problem, Paper A also introduces a method called Bayesian Case-Based Reasoning (BCBR) for creating checklists. BCBR is a hybrid method based on a combination of Bayesian inference on empirical distributions (NBI) and CBR. One of the application-specific motivations for BCBR is to create checklists that are effective (prioritizing checklist items that have high probability for non-compliance) and relevant for each inspection (items are retrieved from checklists used in similar past inspections).

BCBR works by first augmenting past CBR cases with probability estimates for positive target labels (non-compliance to working environment regulations) to enhance the accuracy of case-retrievals from the case base. BCBR then builds a checklist by retrieving K augmented CBR cases with checklist items that have the highest probability to be found non-compliant for the given inspection target. This principle of

Checklist	
Does the employer ensure that the employees work within the framework in Chapter 10 of the Working Environment Act?	**[✓ Yes] / No**
Has the employer ensured that the employment agreements are in line with the minimum requirements set out in the Working Environment Act §14-6?	**Yes / [✓ No]**
Does the company have routines for how violence, threats and other unfortunate burdens as a result of contact with others are to be prevented, reported, handled and followed up?	**Yes / No**
Has the employer implemented the necessary measures and/or prepared a plan describing measures to remove or reduce hazards and problems at work?	**Yes / No**
Has the employer mapped the dangers and problems the employees may be exposed to in the company and on this basis assessed the risk of injury to or danger to the employees?	**Yes / No**
Added Items	
Does the company pay at least a 40% supplement to the salary for overtime work?	**Yes / No**
Does the employer have control over the working hours of the employees within the framework of the Working Environment Act, Chapter 10?	**Yes / No**

TABLE 4.1. The table [45] shows a dynamic checklists with 5 initial items, generated for a hotel inspection in Oslo. Two items on the checklist have been answered (as indicated with blue circles). Two items have also been dynamically added to the checklist (at the bottom), based on the answered items.

augmenting cases to enhance case-retrieval in CBR is novel and may also be relevant to use in other applications as well [84]. The decision to use NBI to generate probability estimates for the CBR cases is based on a preliminary experiment with a simpler problem [44]. The objective of the problem is to predict the correct answer of any checklist item (classification problem). The results of the experiment show that NBI performed best overall out of the methods considered. Paper A also introduces a cross-validation approach, which is used in an experiment where BCBR and other baselines are evaluated. The results of the experiment indicate that the checklists generated by BCBR are superior to traditional static checklists used in labour inspections, increasing the expected average number of violations found per checklist item (precision) [44].

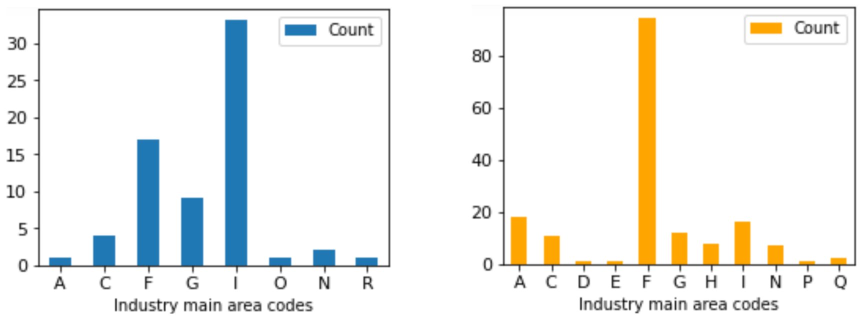
Paper C proposes an extension of BCBR called CBCBR, which creates dynamic checklists that can self-adapt during inspections. CBCBR does so by using the checklist answers provided by inspectors to recommend additional items that fit the context. The recommendations are carried out by first updating the probability estimates in the augmented CBR cases, conditioning the estimates on the known checklist answers. Any CBR cases that have received sufficiently increased estimates are then retrieved and the embedded items are presented to the users. The new items can then be appended to the checklists at the users' discretion. As far as we know, CBCBR is the first AI- or ML-based approach for generating dynamic checklists. The method is also versatile, so the dynamic checklists can be implemented in many different ways. For example, in addition to adding new items, it is possible to mark or even remove checklist items that become less relevant during inspection. However, such functionalities could also become confusing for users and may also complicate evaluations [45, 46], so this is something that may be considered for future work.

Checklist	
Does the employer ensure that a written working agreement is made with the employees?	**Yes / No**
Does the employer have a continuous overview of how much the individual employee works?	**Yes / No**
Has the employer ensured that the employee has received a written statement of the calculation method for salary, the calculation basis for holiday pay and deductions that have been made (payslip)?	**Yes / No**
Has the employer arranged to pay wages, including holiday pay and other compensation in cash, via bank to the employee's account?	**Yes / No**
Does the employer pay wages in accordance with regulations on generalization of collective agreements for accommodation, catering and catering businesses?	**Yes / No**
Has the employer paid wages and any other compensation in accordance with the current public policy decision?	**Yes / No**
Has the employer deducted the employee's wages for accommodation in the company in accordance with regulations on the generalization of collective agreements for accommodation, serving and catering businesses?	**Yes / No**

TABLE 4.2. The table [46] shows a traditional static checklist of 7 items that are used for inspections within the hospitality industry (hotels, restaurants and catering) by the Norwegian Labour Inspection Authority. This represents only one of multiple checklists that are used in such inspections.

A demonstration of a dynamic checklist generated via CBCBR for a hotel inspection in Oslo is shown in Table 4.1. As a result of the two answers provided in the checklist (blue rings), two more items related to those answers have been added at the bottom. The added items cover additional topics and prompt inspectors to check that employees receive overtime payments and do not work too much overtime. In contrast, Table 4.2 shows a traditional static checklist that could also be used in the same hotel inspection. This checklist remains the same throughout the inspection and covers a much narrower set of topics compared to the dynamic checklist in Table 4.1. The static checklist in Table 4.2 covers working agreements and that workers receive the legal minimum wages. The dynamic checklist in Table 4.1 seems to cover most of the same topics and more, such as routines for handling threats and violence. We made the same observation when comparing more examples of dynamic versus traditional checklists for other types of inspections as well. The comparisons indicate that the CBCBR model from Paper C works as intended and is capable of generating optimized checklists. The experiments conducted in Paper C indicate that the ability to adapt the checklists while they are used by inspectors gives ML-based dynamic checklists (CBCBR) a clear advantage over static checklists (BCBR as well as traditional checklists currently used by NLIA).

All in all, Paper A and Paper C answer Research question 2 by introducing and testing ML methods for creating checklists. These methods could also potentially be used to generate checklists for tasks in other domains by using different training datasets. In particular, CBCBR from Paper C could be relevant for applications where



(A) Distribution of inspections in the test group, using dynamic checklists.

(B) Distribution of inspections in the control group, using paper-based checklists.

FIGURE 4.1. The diagrams [46] show the number of inspections conducted in organisations belonging to different industries in the field study from Paper E (indicated by letters A-Q). For the test group, most of the inspections were conducted within the Accommodation & Food and Construction industries (I). The majority of the inspections of the control group were carried out in construction businesses (F). Interpretations for the letters shown the figures can be found in Table 2.1.

dynamic checklists are already being used, such as in healthcare [38, 39, 70] (see Section 3.2.2). In these applications, dynamic checklists rely on models that must be created manually by domain experts, which can be time consuming [26, 39, 45, 46].

4.2.3 Research Question 3 - How Should ML-Based Checklists be Evaluated?

This research question is an extension of Research question 2. Both Paper A and Paper C address Research question 2 by evaluating ML methods using cross-validation on existing data. However, the effect of checklists on inspectors' task performance can be unpredictable [21, 39, 55] and evaluation methods on existing data may not account for certain real-world factors, such as interaction effects between inspectors (human), inspected organisations, and the ML-based checklists [81, 103, 104]. Thus, Paper E addresses Research question 3 by presenting the results from a field study where the ML-based checklists are tested in real-world environments by labour inspectors. To our knowledge, this is the first published study on the use of ML-based checklists [46]. The purpose of the field study is: (1) to find out how checklists should be evaluated, and (2) to confirm that ML-based checklists do indeed increase labour inspection performance in the field. The field study was conducted with a test and control group, with inspectors carrying out inspections using ML-based dynamic checklists in the test group and traditional static checklists in the control group. As shown in Figure 4.1, the inspections were carried out in a wide range of industries, which caused some selection bias. This was done to avoid disrupting inspectors from their daily tasks, and shows that it can be practically challenging to design real-world experiments for complex applications such as labour inspections. To correct bias and allow fair comparisons, the results from the study are weighted.

The results from the study in Paper E indicate that the evaluation methods proposed in Paper A and C do not reflect the real-world field performance of the ML-

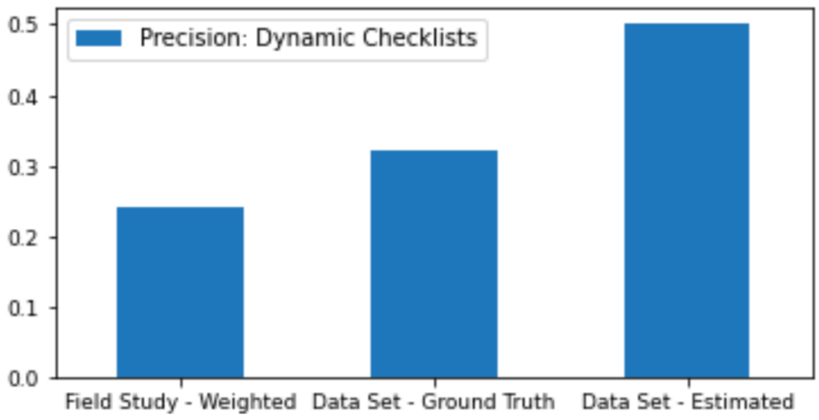


FIGURE 4.2. Overall weighted average precision scores from the field study in Paper E (left) versus ground truth precision (middle) and estimated precision scores (right) from cross-validation in Paper C. The figure indicates that the cross-validation approaches (middle and right) are too optimistic.

checklists being evaluated. This is shown in Figure 4.2, where the score of the dynamic checklists in the field study is significantly lower than the cross-validation scores. This insight may have implications for future research in other domains where ML is used to create checklists, such as medicine [77, 126]. The results also show that labour inspection efficiency increases when dynamic checklists are used because more violations are found (up to 27% increase), using shorter checklists (up to 20% reduction). Therefore, these results make a significant contribution to the research goal of understanding how ML can be used to improve labour inspection efficiency. However, some of the inspectors also reported that they spent more time following up on violations. The implications of this are discussed in more detail in Section 4.3.2.

4.2.4 *Research Question 4 - How can we Ensure that ML Models for Labour Inspections are Interpretable?*

This research question addresses the fact that many of the recent ML methods are "black boxes", with the consequence that models (and their predictions) are difficult to interpret or explain [110, 111]. To promote users' trust and reliance on ML models, it is important that ML models are interpretable so that their predictions can be understood. Model-agnostic explanation methods for black boxes exist, such as LIME or SHAP [47, 91, 108], but these methods do not fully explain models' reasoning processes that lead to their predictions. They can also be difficult to understand for users who are not ML-experts [46, 110]. Inspectors and domain experts working with labour inspections may be unable to understand detailed ML model characteristics and explanations as well as ML experts. The research question is therefore a natural progression towards the research goal from the other research questions because even the best, most accurate ML models, can be useless if users cannot rely on them [110]. Furthermore, forthcoming EU-regulations will require a level of interpretability of ML technologies [29].

Paper A seeks to address Research question 4, showing that interpretable ML methods can be achieved by focusing the research on ML methods that are naturally interpretable or transparent. For example, the paper highlights BCBR as a transparent method, since both of its components (NBI and CBR) are based on processes that can be traced manually [116]. The NBI-component calculates probability estimates for each CBR-case simply by counting instances in the dataset. The CBR-component assigns scores to every case in the case-base via a similarity function and retrieves the K cases that have the highest scores. The experiments in Paper A are also carried out exclusively with relatively interpretable baselines such as logistic regression and the naive Bayes' classifier. In paper C, we also tested some black-box methods against both BCBR and CBCBR, such as ANN and Random Forests. We found that the black-box methods perform on par with NBI in terms of model accuracy, but worse than BCBR and CBCBR. Time-wise, the black-box methods also perform significantly worse than NBI, BCBR and CBCBR. Thus, the results in Paper C indicate that interpretability in ML does not necessarily come at a cost to model accuracy [110].

Although Paper A and C introduce new ML methods that are transparent and interpretable for ML experts, the papers do not consider how to ensure interpretability for inspectors. However, this topic is covered in Paper E and is important because inspectors are unlikely to have the same level of technical skills as ML experts [46]. The paper discusses how explanations can be provided to ensure that ML-based checklists are interpretable for inspectors, with the goal of providing a certain level of justification and transparency [116]. Paper E introduces two methods for explaining the content of dynamic checklists (generated by CBCBR from Paper C) to inspectors [46]. The first method is to show the estimated probability for non-compliance on each checklist item to inspectors, to justify the use of the items selected for each checklist. The second method is to highlight the checklist answers that had the most impact when recommending new items for a dynamic checklist. The method could help inspectors understand why certain extra checklist items are being recommended during inspections. Both explanation methods are implemented in a prototype that is used for the field study of labour inspections described in the paper. The explanations received positive feedback from the inspectors who participated in the study. However, the inspectors also requested explanations for why certain items did not make it on their checklists [46]. It is currently possible to provide such explanations by examining the cases in CBCBR's case-base, but this has to be done manually. Finding a way to generate such explanations automatically could be investigated in future work.

4.3 Limitations of the Research

This section discusses some of the limitations of the research contributions from the Ph.D.-project, in context of the research questions discussed in the previous sections.

4.3.1 Implementation of Machine Learning in Labour Inspection Authorities

As mentioned in Section 1.2, the scope of the research in the Ph.D.-project does not cover how ML methods should be implemented and operationalized into labour inspections. This is still an important question, as some of the ML methods that we proposed may need some adaptations and improvements to fit labour inspection processes and operations. In our work, we have also assumed that one of the key performance indicators (KPIs) for labour inspections is the number of violations found in the inspections. Our assumption is that the number of violations found in an inspection should statistically correlate with the number of serious violations found as well. However, while this KPI may make sense for countries like Norway, it may be less important in countries such as Bangladesh or India where combating child labour has a much bigger focus than addressing general working environment violations. This is usually related to limited resources, which forces labour inspection authorities to make very specific priorities [57]. This issue could potentially be solved by adapting or replacing the datasets we used to accommodate other KPIs.

4.3.2 *Measuring Increases in Labour Inspection Efficiency*

Another limitation of the research is measuring labour inspection efficiency, which can be difficult [118]. Efficiency can be defined as the amount of effort that must be made to get something done. In Paper E, we presented a field study showing that using dynamic checklists increases the number of violations found in the inspections, with the dynamic checklists generally being shorter than traditional static checklists. We assume that the combination of fewer things to check for (shorter checklists), combined with an increased number of violations being found in the inspections, indicates increased labour inspections efficiency. However, as mentioned earlier, inspectors report that they also spent more time following up the inspections when using dynamic checklist. We have not measured how much time the inspectors spend on case management in our study. Therefore, it is not certain whether the total time inspectors spend on planning, executing, and following up inspections increases or decreases as a consequence of using dynamic checklists. However, studies in this area may be considered in future work.

4.4 Source Code and Data

In addition to the research discussed above, the Ph.D.-project also generated data and source code. This is important considering that the publicly available data on labour inspections is very limited. An overview of the resources is provided below, including URLs.

- The Checklist Items dataset, used to create checklists in Paper A, C and E: https://dx.doi.org/10.21227/m1t7-hg51. The dataset consists of a collection of almost 2 000 unique items from 369 checklists used in 59 988 labour inspections between 2013 and 2019.

- The Labour Inspection Checklist Dataset (LICD), analyzed and used for the experiments in Paper D: https://doi.org/10.18710/7U6TZP. The dataset consists of 63 634 inspections conducted between 2012 and 2019 and has 577 features and labels. The dataset can be used to predict the most relevant checklists that inspectors could use for their inspections or to predict non-compliance to working environment regulations among organisations. An earlier version of the dataset is used for the experiments in Paper B.

- Source code for the experiments in Paper C: https://github.com/ntnu-ai-lab/cbcbr. The experiment compares the checklist generated by CBCBR from Paper C and BCBR from Paper A to baselines such as ANN, NBI, DT and traditional static checklists. The results are measured in terms of precision, accuracy, recall and execution time. Installation instructions can be found in the repository.

- Source code for the prototype software to generate dynamic checklists, presented in Paper E: https://github.com/ntnu-ai-lab/cbcbr-prototype. The software was installed on tablets and used by labor inspectors in the field study discussed in the paper. The repository includes an executable file that can be downloaded and installed in a Windows environment. The installation instructions can be found in the repository. The code may be unavailable at the time of submission of this thesis.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The goal of this research is to study how ML can be used to improve the efficiency of labour inspections. These inspections are an integral part of the enforcement and promotion of the United Nations' decent work agenda (SDG 8) [101]. Pursuant to the International Labour Organization's (ILO) Labour Inspection Convention (1947), labour inspections are carried out nationwide in ratifying states among organisations that employ workers with the goal of addressing non-compliance to international and national labour standards, including occupational health, environment and safety (OSH) [119]. However, due to the increasing diversification of HSE risks in workplaces as well as diminishing budgets, the effectiveness and number of labour inspections being conducted have been declining in recent years [83, 119, 122]. To address these challenges, this thesis focuses on the use of ML to improve how inspections are planned and executed. More specifically, we use ML to create or select checklists for inspections, and to predict non-compliance to HSE regulations among organisations that could potentially be targeted for inspections. Addressing these tasks via ML may mean that inspectors could find violations in their inspections more effectively and potentially conduct and complete their inspections faster.

In our work, we started by investigating prior ML research on labour inspection tasks and found that there was not much available. There were very few datasets openly available for research as well. Therefore, we publish two new datasets to support and promote ML research, one in Paper A and one in Paper D. These datasets support the use of ML to solve at least the three labour inspections tasks mentioned above. Paper A covers the use of ML to create checklists, while Paper D covers the use of ML to select an existing checklist for inspection (out of 369 available pre-established checklists). Paper D also covers the task of predicting non-compliance to working environment regulations among organisations, which could be potential targets for inspections. The paper investigates potential ML methods that could be used to solve the tasks. Both Paper B and D also show that using feature selection could increase model performances and reduce computational costs on the tasks. Thus, all three papers contribute to the establishment of labour inspection tasks for ML, by introducing new datasets and knowledge of how the data should be leveraged.

ILO is also concerned that the current use of pre-established checklists limits the scope and impact of inspections [101]. This research focuses mostly on the task of using ML to create new checklists for each inspection, so that labour inspections can be carried out with higher efficiency without relying on pre-established checklists. Paper A investigates a variation of baselines that could potentially be used to solve the checklist creation task, and finds that Naive Bayesian Inference on empirical distributions (NBI) is the best-performing method. The paper also introduces a new method called Bayesian Case-Based Reasoning (BCBR), which is a hybrid method based on Case-Based Reasoning and the NBI model. In a cross-validation experiment, the checklists generated by BCBR outperform the existing traditional static checklists used by the Norwegian Labour Inspection Authority (NLIA), as well as the checklists from the other ML baseline methods. Paper C also presents an extension of BCBR called Context-aware Bayesian Case-Based Reasoning (CBCBR), which also makes it possible to dynamically adapt the checklists based on how they are answered during inspections. In practice, this means that the checklists are context-aware and can adapt to findings throughout inspections. The cross-validation results in Paper C show that CBCBR's dynamic checklists perform better than BCBR's static checklists. Thus, both Paper A and C contributes to establishing an understanding of how ML can be used to create checklists that potentially increase labour inspection performance.

To further evaluate the dynamic checklists generated by CBCBR and understand the impact they have on the efficiency of the labour inspections, the checklists are tested in a field study with real-world labour inspections. The results are described in Paper E and show that the dynamic checklists increase labour inspection performance compared to traditional inspection checklists. The results also show that the cross-validation performance estimates in Paper A and C are somewhat optimistic regarding the real-world performance of the dynamic checklists. Therefore, the field study in paper E validates that ML-based checklists can improve labour inspection efficiency and also advances the understanding of how ML-based checklists should be evaluated. Paper E also introduces two explanation methods for explaining the content of the dynamic checklists to inspectors, exploiting CBCBR being a "white-box" method. The first explanation method is to highlight the probability for finding non-compliance on each item on the checklist, while the second method explains how dynamically added checklist items are related to the inspectors' interactions with the checklists during the inspections. The methods were well received by the inspectors participating in the study, and could contribute to establishing an understanding of how dynamic checklists can be explained.

Thus, the thesis has established a foundation for understanding how ML can be used in labour inspection tasks, to improve efficiency. However, the scope of this thesis is relatively limited and there is still much potential for future work.

5.2 Future Work

First, the ML models discussed in the thesis have not yet been adopted by any labour inspection agencies and there will likely be challenges that need to be addressed. For example, introduction of new labour regulations would require corresponding new checklist items to be incorporated into the CBCBR model and the dynamic checklists. In recommender systems, this challenge is known as the cold-start problem [75]. Another challenge is how records of completed checklists should be retained in the case-base of the CBCBR model. There is also limited research into the design practices for dynamic checklists, in terms of how to create model and interfaces that provide the best possible user experiences. Finding new ways to explain checklist content to inspectors is also another option. An area to consider is counterfactual explanations such as: What would happen if certain inspected checklist items are not inspected and instead replaced with some other checklist items? These are just a few examples of areas that could be explored in more detail with respect to dynamic checklists.

Another possibility for future work is to look for ways to expand the datasets introduced in Paper A and D with more features. The dataset introduced in Paper A contains around 10 features, which are categorical. Currently, CBCBR only supports categorical features, but it could be possible to use binning techniques, potentially combined with kernels or transformation functions, to support numerical features as well [23, 53, 77], and then expand the dataset with more numerical features. The ML performance in both of the tasks introduced in Paper D is promising but not great, and expanding with new features could help boosting performance in these tasks. Scraping data from the web or other large data sources could be an option to collect more relevant data on potential inspection targets (organisations). Another option is to collect data via physical sensors, for instance, to get visual or audio data from inspections. Such data could potentially be used to capture dangerous situations in certain workplaces, such as construction sites. However, collecting such data could also be challenging due to privacy regulations. It may still be possible to use sensors to capture useful information while maintaining privacy [54, 63], and doing so for labour inspections could be a potential area for future research.

Another direction for future work is to investigate data processing methods further, such as feature selection and feature extraction methods. Such methods are also essential if data scraping is used to collect large amounts of data. Optimization-based feature selection show promising results in both Paper B and D, but finding the optimal set of features is an NP-hard problem [27]. It may be possible to achieve better ML model performance by optimizing the search strategy for the optimal feature set(s), where local search methods could be a starting point [27, 88, 89]. Feature extraction methods and transformations are also topics that could be looked into.

There are also many potential tasks besides those mentioned in this thesis that could be addressed with ML or AI. One of them could be to use NLP models to write the inspection reports that are sent to the inspected organisations after inspections. Another use-case could be to use NLP models to analyze and summarize tips that the labour inspection authorities receive, which for instance could be used to create new features in the datasets from Paper A and D. Such use-cases could potentially be addressed by fine-tuning large-scale language models like GPT for the tasks, to make generated texts more accurate and true to the sources used by the models [3, 18, 105].

References

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1): 39–59, 1994.

[2] Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708–713, 2015.

[3] Nikolich Alexandr, Osliakova Irina, Kudinova Tatyana, Kappusheva Inessa, and Puchkova Arina. Fine-tuning GPT-3 for Russian text summarization. In *Data Science and Intelligent Systems: Proceedings of 5th Computational Methods in Systems and Software 2021, Vol. 2*, pages 748–757. Springer, 2021.

[4] Eleni Alogogianni and Maria Virvou. Association rules and machine learning for enhancing undeclared work detection. In *2020 11th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–8. IEEE, 2020.

[5] Eleni Alogogianni and Maria Virvou. Data mining for targeted inspections against undeclared work by applying the CRISP-DM methodology. In *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pages 1–8. IEEE, 2021.

[6] Eleni Alogogianni and Maria Virvou. Undeclared work prediction using machine learning: Dealing with the class imbalance and class overlap problems. In *2022 13th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pages 1–8. IEEE, 2022.

[7] Aimee Alphonso, Marc Auerbach, Kirsten Bechtel, Kyle Bilodeau, Marcie Gawel, Jeannette Koziel, Travis Whitfill, and Gunjan Kamdar Tiyyagura. Development of a child abuse checklist to evaluate prehospital provider performance. *Prehospital emergency care*, 21(2):222–232, 2017.

[8] Gharbi Alshammari, Jose L. Jorro-Aragoneses, Stelios Kapetanakis, Miltos Petridis, Juan A. Recio-García, and Belén Díaz-Agudo. A hybrid CBR approach for the long tail problem in recommender systems. In *International Conference on Case-Based Reasoning, ICCBR*, pages 35–45. Springer, 2017.

[9] Marly Ryoko Amaya, Danieli Parreira da Silva Stalisz da Paixão, Leila Maria Mansano Sarquis, and Elaine Drehmer de Almeida Cruz. Construction and content validation of checklist for patient safety in emergency care. *Revista gaucha de enfermagem*, 37, 2017.

[10] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[11] Kerstin Bach, Tomasz Szczepanski, Agnar Aamodt, Odd Erik Gundersen, and Paul Jarle Mork. Case representation and similarity assessment in the self back decision support system. In *Case-Based Reasoning Research and Development: 24th International Conference, ICCBR 2016, Atlanta, GA, USA, October 31-November 2, 2016, Proceedings 24*, pages 32–46. Springer, 2016.

[12] Aditya Bharadwaj, Pao Siangliulue, Adam Marcus, and Kurt Luther. Critter: Augmenting creative work with dynamic checklists, automated quality assurance, and contextual reviewer feedback. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

[13] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[14] Julia Black and Robert Baldwin. Really responsive risk-based regulation. *Law & policy*, 32(2):181–213, 2010.

[15] Francesco Bodria, Fosca Giannotti, Riccardo Guidotti, Francesca Naretto, Dino Pedreschi, and Salvatore Rinzivillo. Benchmarking and survey of explanation methods for black box models. *Data Mining and Knowledge Discovery*, pages 1–60, 2023.

[16] Guy Van den Broeck, Karthika Mohan, Arthur Choi, and Judea Pearl. Efficient algorithms for Bayesian network parameter learning from incomplete data. *arXiv preprint arXiv:1411.7014*, 2014.

[17] Tore Bruland, Agnar Aamodt, and Helge Langseth. Architectures integrating case-based reasoning and Bayesian networks for clinical decision support. In *Intelligent Information Processing V: 6th IFIP TC 12 International Conference, IIP 2010, Manchester, UK, October 13-16, 2010. Proceedings 6*, pages 82–91. Springer, 2010.

[18] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*, 2023.

[19] Hubo Cai, JungHo Jeon, Xin Xu, Yuxi Zhang, Liu Yang, et al. Automating the generation of construction checklists. Technical report, Purdue University. Joint Transportation Research Program, 2020.

[20] Antonio José Carpio-de Los Pinos, María de las Nieves González-García, Ligia Cristina Pentelhão, and J. Santos Baptista. Zero-risk interpretation in the level of preventive action method implementation for health and safety in construction sites. *International journal of environmental research and public health*, 18(7):3534, 2019.

[21] Ken Catchpole and Stephanie Russ. The problem with checklists. *BMJ quality & safety*, 24(9):545–549, 2015.

[22] Subhajit Chaudhury, Sarathkrishna Swaminathan, Chulaka Gunasekara, Maxwell Crouse, Srinivas Ravishankar, Daiki Kimura, Keerthiram Murugesan,

Ramón Fernandez Astudillo, Tahira Naseem, Pavan Kapanipathi, et al. X-factor: A cross-metric evaluation of factual correctness in abstractive summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7100–7110, 2022.

[23] Wenlin Chen, Yixin Chen, Yi Mao, and Baolong Guo. Density-based logistic regression. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 140–148, 2013.

[24] Arthur Choi, Adnan Darwiche, L. Zheng, and Ole Jakob Mengshoel. A tutorial on Bayesian networks for system health management. *Machine Learning and Knowledge Discovery for Engineering Systems Health Management*, 10(1):1–29, 2011.

[25] Jae Kwang Choi and Yeon Hwan Lee. Effects of emotional labor on musculoskeletal disorders among physical therapists in south korea. *International Journal of Occupational Safety and Health*, 13(2):190–198, 2023.

[26] Stefan C. Christov, Heather M. Conboy, Nancy Famigletti, George S. Avrunin, Lori A. Clarke, and Leon J. Osterweil. Smart checklists to improve healthcare outcomes. In *International Workshop on SEHS*, pages 54–57, 2016.

[27] Ahmet Cevahir Cinar. A novel adaptive memetic binary optimization algorithm for feature selection. *Artificial Intelligence Review*, pages 1–58, 2023.

[28] Mark J. Clayton. Delphi: a technique to harness expert opinion for critical decision-making tasks in education. *Educational psychology*, 17(4):373–386, 1997.

[29] European Commission. Proposal for a regulation of the European parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts, 2021. URL https://digital-strategy.ec.europa.eu/en/library/proposal-regulation-laying-down-harmonised-rules-artificial-intelligence.

[30] Michele Cossalter, Ole Jakob Mengshoel, and Ted Selker. Visualizing and understanding large-scale Bayesian networks. In *Scalable Integration of Analytics and Visualization*, 2011.

[31] Ania Cravero, Sebastian Pardo, Samuel Sepúlveda, and Lilia Muñoz. Challenges to use machine learning in agricultural big data: A systematic literature review. *Agronomy*, 12(3):748, 2022.

[32] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[33] Andrea S. N. Curman, Martin A. Andresen, and Paul J. Brantingham. Crime and place: A longitudinal examination of street segment patterns in Vancouver, bc. *Journal of Quantitative Criminology*, 31(1):127–147, 2015.

[34] Øyvind Dahl. The future role of big data and machine learning for health and safety inspection efficiency. *EU-OSHA: Bilbao, Spain*, 2019.

[35] Øyvind Dahl and Marius Søberg. Labour inspection and its impact on enterprises' compliance with safety regulations. *Saf. Sci. Monit*, 17:1–12, 2013.

[36] Jesus M. Darias, Marta Caro-Martínez, Belén Díaz-Agudo, and Juan A. Recio-Garcia. Using case-based reasoning for capturing expert knowledge on explanation methods. In *International Conference on Case-Based Reasoning*, pages 3–17. Springer, 2022.

[37] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009.

[38] A. J. R De Bie, S. Nan, L. R. E. Vermeulen, P. M. E. Van Gorp, R. A. Bouwman, A. J. G. H. Bindels, and H. H. M. Korsten. Intelligent dynamic clinical checklists improved checklist compliance in the intensive care unit. *BJA: British Journal of Anaesthesia*, 119(2):231–238, 2017.

[39] Ashley J. R. De Bie, Eveline Mestrom, Wilma Compagner, Shan Nan, Lenneke van Genugten, Kiran Dellimore, Jacco Eerden, Steffen van Leeuwen, Harald van de Pol, Franklin Schuling, et al. Intelligent checklists improve checklist compliance in the intensive care unit: a prospective before-and-after mixed-method study. *British Journal of Anaesthesia*, 126(2):404–414, 2021.

[40] Asaf Degani and Earl L. Wiener. Human factors of flight-deck checklists: the normal checklist. Technical report, 1990.

[41] Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019.

[42] Ciara Feely, Brian Caulfield, Aonghus Lawlor, and Barry Smyth. A case-based reasoning approach to predicting and explaining running related injuries. In *Case-Based Reasoning Research and Development: 29th International Conference, ICCBR 2021, Salamanca, Spain, September 13–16, 2021, Proceedings 29*, pages 79–93. Springer, 2021.

[43] Eirik Lund Flogard and Ole Jakob Mengshoel. A dataset for efforts towards achieving the sustainable development goal of safe working environments. In *Advances in Neural Information Processing Systems 35 - NeurIPS*, 2022.

[44] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Bayesian feature construction for case-based reasoning: Generating good checklists. In *International Conference on Case-Based Reasoning*, pages 94–109. Springer, 2021.

[45] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Creating dynamic checklists via Bayesian case-based reasoning: Towards decent working conditions for all. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5108–5114, 2022. doi: 10.24963/ijcai.2022/709.

[46] Eirik Lund Flogard, Ole Jakob Mengshoel, Ole Magnus Theisen, and Kerstin Bach. Creating explainable dynamic checklists via machine learning to ensure decent working environment for all: A field study with labour inspections. In *Prestigious Applications of Intelligent Systems (PAIS)*, 2023.

[47] Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of LIME. In *International conference on artificial intelligence and statistics*, pages 1287–1296. PMLR, 2020.

[48] Tom Gelhausen, Mathias Landhäußer, and Sven J. Körner. Automatic checklist generation for the assessment of UML models. In *International Conference on Model Driven Engineering Languages and Systems*, pages 387–399. Springer, 2008.

[49] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[50] Venkatsampath Raja Gogineni, Sravya Kondrakunta, Danielle Brown, Matthew Molineaux, and Michael T. Cox. Probabilistic selection of case-based explanations in an underwater mine clearance domain. In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 110–124, 2019. doi: 10.1007/978-3-030-29249-2_8.

[51] Yuan Guo, Wei Chen, Ying-Xia Zhu, and Yu-Qin Guo. Research on the integrated system of case-based reasoning and Bayesian network. *ISA transactions*, 90:213–225, 2019.

[52] Felicity Hasson, Sinead Keeney, and Hugh McKenna. Research guidelines for the Delphi survey technique. *Journal of advanced nursing*, 32(4):1008–1015, 2000.

[53] Yujie He, Wenlin Chen, Yixin Chen, and Yi Mao. Kernel density metric learning. In *2013 IEEE 13th international conference on data mining*, pages 271–280. IEEE, 2013.

[54] Saad Himmi, Oguzhan Ilter, François Pailleau, Roland Siegwart, Berta Bescos, and Cesar Cadena. Don't share my face: Privacy preserving inpainting for visual localization. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12506–12511. IEEE, 2022.

[55] Daniel E. Ho, Sam Sherman, and Phil Wyman. Do checklists make a difference? a natural experiment from food safety enforcement. *Journal of Empirical Legal Studies*, 15(2):242–277, 2018.

[56] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.

[57] Vera Jatobá et al. Labour inspection within a modernized labour administration. *ILO*, 2002. URL https://ilo.primo.exlibrisgroup.com/discovery/fulldisplay/alma993608813402676/41ILO_INST:41ILO_V2.

[58] Srdjan Jelacic, Andrew Bowdle, Bala G. Nair, Kei Togashi, Daniel J. Boorman, Kevin C. Cain, John D. Lang, and E. Patchen Dellinger. Aviation-style computerized surgical safety checklist displayed on a large screen and operated by the anesthesia provider improves checklist performance. *Anesthesia & Analgesia*, 130(2):382–390, 2020.

[59] Zhiwei Ji, Qibiao Xia, and Guanmin Meng. A review of parameter learning methods in Bayesian network. In *Advanced Intelligent Computing Theories and Applications: 11th International Conference, ICIC 2015, Fuzhou, China, August 20-23, 2015. Proceedings, Part III 11*, pages 3–12. Springer, 2015.

[60] Qixuan Jin, Haoran Zhang, Thomas Hartvigsen, and Marzyeh Ghassemi. Fair multimodal checklists for interpretable clinical time series prediction. In *NeurIPS 2022 Workshop on Learning from Time Series for Health*, 2022.

[61] Matthew S. Johnson, David I. Levine, and Michael W. Toffel. Improving regulatory effectiveness through better targeting: Evidence from OSHA. *Harvard Business School Technology & Operations Mgt. Unit Working Paper*, (20-019), 2019.

[62] Jose Luis Jorro-Aragoneses, Marta Caro-Martínez, Belén Díaz-Agudo, and Juan A. Recio-García. A user-centric evaluation to generate case-based explanations using formal concept analysis. In *International Conference on Case-Based Reasoning, ICCBR*, pages 195–210. Springer, 2020.

[63] Vijaya Kumar Kambala and Jonnadula Harikiran. Privacy preserving human activity recognition framework using an optimized prediction algorithm. *IAES International Journal of Artificial Intelligence*, 11(1):254, 2022.

[64] Nektarios Karanikas and Sikder Mohammad Tawhidul Hasan. Occupational health & safety and other worker wellbeing areas: Results from labour inspections in the Bangladesh textile industry. *Safety Science*, 146:105533, 2022.

[65] Eoin M. Kenny and Mark T. Keane. Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In *Twenty-Eighth International Joint Conferences on Artifical Intelligence (IJCAI), Macao, 10-16 August 2019*, pages 2708–2715, 2019.

[66] Eoin M. Kenny, Elodie Ruelle, Anne Geoghegan, Laurence Shalloo, Micheál O'Leary, Michael O'Donovan, and Mark T. Keane. Predicting grass growth for sustainable dairy farming: A CBR system using Bayesian case-exclusion and post-hoc, personalized explanation-by-example (XAI). In *International Conference on Case-Based Reasoning*, pages 172–187. Springer, 2019.

[67] Been Kim, Cynthia Rudin, and Julie A. Shah. The Bayesian case model: A generative approach for case-based reasoning and prototype classification. *Advances in neural information processing systems*, 27, 2014.

[68] Suhong Kim, Param Joshi, Parminder Singh Kalsi, and Pooya Taheri. Crime analysis through machine learning. In *Proceedings of the IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 415–420, 2018.

[69] Vlad Krotov, Leigh Johnson, and Leiser Silva. Tutorial: Legality and ethics of web scraping, 2020.

[70] Leah Kulp, Aleksandra Sarcevic, Megan Cheng, and Randall S. Burd. Towards dynamic checklists: Understanding contexts of use and deriving requirements for context-driven adaptation. *ACM TOCHI*, 28(2):1–33, 2021.

[71] David Leake, Larry Birnbaum, Kristian Hammond, Cameron Marlow, and Hao Yang. An integrated interface for proactive, experience-based design support. In *Proceedings of the 6th international conference on Intelligent user interfaces*, pages 101–108, 2001.

[72] David Leake, Xiaomeng Ye, and David J. Crandall. Supporting case-based reasoning with neural networks: An illustration for case adaptation. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, volume 2, 2021.

[73] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[74] Chieh-Kang Liao, Alan Liu, and Yu-Sheng Chao. A machine learning approach to case adaptation. In *2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 106–109. IEEE, 2018.

[75] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert systems with applications*, 41 (4):2065–2073, 2014.

[76] Sara Maheronnaghsh, H. Zolfagharnasab, M. Gorgich, and J. Duarte. Machine learning in occupational safety and health: protocol for a systematic review:(protocol). *International Journal of Occupational and Environmental Safety*, 5(1):32–38, 2021.

[77] Yukti Makhija, Edward De Brouwer, and Rahul G. Krishnan. Learning predictive checklists from continuous medical data. *arXiv preprint arXiv:2211.07076*, 2022.

[78] Håkon Måløy. *Learning neural representations for the processing of temporal data in deep neutral networks*. PhD thesis, NTNU, 2023.

[79] David Maré and Kerry L. Papps. The effects of occupational safety and health interventions. *Labour Market Bulletin*, 2:101–131, 2002.

[80] Aitor Mata and Juan Manuel Corchado. Forecasting the probability of finding oil slicks using a CBR system. *Expert Systems with Applications*, 36(4):8239–8246, 2009.

[81] Aditya Mate, Lovish Madaan, Aparna Taneja, Neha Madhiwalla, Shresth Verma, Gargi Singh, Aparna Hegde, Pradeep Varakantham, and Milind Tambe. Field study in deploying restless multi-armed bandits: Assisting non-profits in improving maternal and child health. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12017–12025, 2022.

[82] Bjørn Magnus Mathisen, Agnar Aamodt, Kerstin Bach, and Helge Langseth. Learning similarity measures from data. *Progress in Artificial Intelligence*, 9(2): 129–143, 2020.

[83] Päivi Mattila-Wiro, Yogindra Samant, Wiking Husberg, Magnus Falk, Annemarie Knudsen, and Eyjolfur Saemundsson. Work today and in the future: perspectives on occupational safety and health challenges and opportunities for the nordic labour inspectorates. Technical report, Nordic labour inspectorates, 2020.

[84] David H. Ménager and Dongkyu Choi. Hybrid event memory as a case base for state estimation in cognitive agents. In *International Conference on Case-Based Reasoning*, pages 134–149. Springer, 2023.

[85] Ole Jakob Mengshoel, Adnan Darwiche, Keith Cascio, Mark Chavira, Scott Poll, and Serdar Uckun. Diagnosing faults in electrical power systems of spacecraft and aircraft. In *AAAI*, pages 1699–1705, 2008.

[86] Ole Jakob Mengshoel, Adnan Darwiche, and Serdar Uckun. Sensor validation using Bayesian networks. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 2008.

[87] Ole Jakob Mengshoel, Raj Desai, Andrew Chen, and Brian Tran. Will we connect again? machine learning for link prediction in mobile social networks. In *Eleventh workshop on mining and learning with graphs*, 2013.

[88] Ole Jakob Mengshoel, Eirik Flogard, Jon Riege, and Tong Yu. Stochastic local search heuristics for efficient feature selection: An experimental study. In *Norsk IKT-konferanse for forskning og utdanning*, pages 58–71, 2021.

[89] Ole Jakob Mengshoel, Tong Yu, Jon Riege, and Eirik Flogard. Stochastic local search for efficient hybrid feature selection. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 133–134, 2021.

[90] Ole Jakob Mengshoel, Eirik Lund Flogard, Tong Yu, and Jon Riege. Understanding the cost of fitness evaluation for subset selection: Markov chain analysis of stochastic local search. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 251–259, 2022.

[91] Karim El Mokhtari, Ben Peachey Higdon, and Ayşe Başar. Interpreting financial time series with SHAP values. In *Proceedings of the 29th annual international conference on computer science and software engineering*, pages 166–172, 2019.

[92] Pamela J. Morgan, Jenny Lam-McCulloch, Jodi Herold-McIlroy, and Jordan Tarshis. Simulation performance checklist generation using the Delphi technique. *Canadian journal of anaesthesia*, 54(12):992–997, 2007.

[93] Kevin P. Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022.

[94] Tatwadarshi P. Nagarhalli, Vinod Vaze, and N. K. Rana. Impact of machine learning in natural language processing: A review. In *2021 third international conference on intelligent communication technologies and virtual mobile networks (ICICV)*, pages 1529–1534. IEEE, 2021.

[95] Hoda Nikpour and Agnar Aamodt. Fault diagnosis under uncertain situations within a Bayesian knowledge-intensive CBR system. *Progress in Artificial Intelligence*, pages 1–14, 2021. doi: 10.1007/s13748-020-00227-x.

[96] Hoda Nikpour, Agnar Aamodt, and Kerstin Bach. Bayesian-supported retrieval in bncreek: a knowledge-intensive case-based reasoning system. In *International Conference on Case-Based Reasoning, ICCBR*, pages 323–338. Springer, 2018.

[97] Statistics Norway. Statistics on establishments, Retrieved 3. September 2023. URL https://www.ssb.no/en/virksomheter-foretak-og-regnskap/virksomheter-og-foretak/statistikk/virksomheter.

[98] Conor Nugent and Pádraig Cunningham. A case-based explanation system for black-box systems. *The Artificial Intelligence Review*, 24(2):163, 2005.

[99] Anindita Tasnim Onni, Asela Kumar Perera Dodanwalage, Magne Bråtveit, and Bente Elisabeth Moen. Prevalence of occupational injuries in selected coir industries in Sri Lanka: a cross-sectional study. *International Journal of Occupational Safety and Health*, 13(2):206–213, 2023.

[100] Santiago Ontanón and Enric Plaza. Amalgams: A formal approach for combining multiple case solutions. In *Case-Based Reasoning. Research and Development: 18th International Conference on Case-Based Reasoning, ICCBR, Alessandria, Italy, July 19-22*, pages 257–271. Springer, 2010.

[101] International Labour Organization. Labour inspection convention, 1947 (no. 81), and labour inspection (agriculture) convention, 1969 (no. 129) - general observation, publication, 2020. URL https://www.ilo.org/global/standards/subjects-covered-by-international-labour-standards/labour-inspection/WCMS_752439/lang--en/index.htm.

[102] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*, pages 128–144. Springer, 2020.

[103] Judea Pearl. The seven tools of causal inference, with reflections on machine learning. *Communications of the ACM*, 62(3):54–60, 2019.

[104] Andi Peng, Besmira Nushi, Emre Kiciman, Kori Inkpen, and Ece Kamar. Investigations of performance and bias in human-AI teamwork in hiring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12089–12097, 2022.

[105] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with GPT-4. *arXiv preprint arXiv:2304.03277*, 2023.

[106] Glenn Pransky, Terry Snyder, Allard Dembe, and Jay Himmelstein. Under-reporting of work-related disorders in the workplace: a case study and review of the literature. *Ergonomics*, 42(1):171–182, 1999.

[107] Prema Rassiah, Fan-Chi Frances Su, Jessica Huang, Dan Spitznagel, Vikren Sarkar, Martin W. Szegedi, Hui Zhao, Adam B. Paxton, Geoff Nelson, and Bill J. Salter. Using failure mode and effects analysis (FMEA) to generate an initial plan check checklist for improved safety in radiation treatment. *Journal of Applied Clinical Medical Physics*, 21(8):83–91, 2020.

[108] Juan A. Recio-García, Belén Díaz-Agudo, and Victor Pino-Castilla. CBR-LIME: a case-based reasoning approach to provide specific local interpretable model-agnostic explanations. In *Case-Based Reasoning Research and Development: 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8–12, 2020, Proceedings 28*, pages 179–194. Springer, 2020.

[109] Brian Ricks and Ole Jakob Mengshoel. Diagnosis for uncertain, dynamic and hybrid domains using Bayesian networks and arithmetic circuits. *International Journal of Approximate Reasoning*, 55(5):1207–1234, 2014.

[110] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[111] Cynthia Rudin and Joanna Radin. Why are we using black box models in AI when we don't need to? a lesson from an explainable AI competition. *Harvard Data Science Review*, 1(2):10–1162, 2019.

[112] Aleksandra Sarcevic, Brett J. Rosen, Leah J. Kulp, Ivan Marsic, and Randall S. Burd. Design challenges in converting a paper checklist to digital format for dynamic medical settings. In *International Conference on Pervasive Computing Technologies for Healthcare:[proceedings]. International Conference on Pervasive Computing Technologies for Healthcare*, volume 2016, page 33. NIH Public Access, 2016.

[113] Mauro Scanagatta, Antonio Salmerón, and Fabio Stella. A survey on Bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8:425–439, 2019.

[114] Silvia N. Schiaffino and Analia Amandi. User profiling with case-based reasoning and Bayesian networks. In *IBERAMIA-SBIA 2000 open discussion track*, pages 12–21, 2000.

[115] Barry Smyth, Aonghus Lawlor, Jakim Berndsen, and Ciara Feely. Recommendations for marathon runners: on the application of recommender systems and machine learning to support recreational marathon runners. *User Modeling and User-Adapted Interaction*, 32(5):787–838, 2022.

[116] Frode Sørmo, Jörg Cassens, and Agnar Aamodt. Explanation in case-based reasoning–perspectives and goals. *Artificial Intelligence Review*, 24(2):109–143, 2005.

[117] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. *Advances in neural information processing systems*, 33:19667–19679, 2020.

[118] Wolfgang Von Richthofen. *Labour inspection: a guide to the profession*. International Labour Organization, 2002. URL https://www.ilo.org/global/topics/safety-and-health-at-work/resources-library/publications/WCMS_108665/lang--en/index.htm.

[119] David Walters. Labour inspection and health and safety in the eu. *The European Trade Union Institute's (ETUI) Health and Safety at Work Magazine,(14)*, pages 12–17, 2016.

[120] David Weil. If OSHA is so bad, why is compliance so good? *RAND Journal of Economics*, 27(3):620, 1996.

[121] David Weil. A strategic approach to labour inspection. *International Labour Review*, 147(4):349–375, 2008.

[122] David Weil. Improving workplace conditions through strategic enforcement. *Boston U. School of Management Research Paper*, 20, 2010.

[123] Stig Winge. *Occupational Safety in the Construction Industry. Identifying Important Accident Types, Barrier Failures, Causal Factors and Safety Management Factors*. PhD thesis, NTNU, 2019.

[124] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, Jie Chen, Zhaogang Wang, and Honglin Qiao. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference*, pages 187—-196, 2018.

[125] Xiaomeng Ye, David Leake, Vahid Jalali, and David J. Crandall. Learning adaptations for case-based classification: A neural network approach. In *Case-Based Reasoning Research and Development: 29th International Conference, ICCBR 2021, Salamanca, Spain, September 13–16*, pages 279–293. Springer, 2021.

[126] Haoran Zhang, Quaid Morris, Berk Ustun, and Marzyeh Ghassemi. Learning optimal predictive checklists. *Advances in Neural Information Processing Systems 34, NeurIPS*, 2021.

[127] Ruochen Zhao, Xingxuan Li, Yew Ken Chia, Bosheng Ding, and Lidong Bing. Can ChatGPT-like generative models guarantee factual accuracy? on the mistakes of new generation search engines. *arXiv preprint arXiv:2304.11076*, 2023.

[128] Yanxia Zhao, Jingsong Li, Meibian Zhang, Yao Lu, Hongwei Xie, Yu Tian, and Wei Qiu. Machine learning models for the hearing impairment prediction in workers exposed to complex industrial noise: a pilot study. *Ear and hearing*, 40 (3):690, 2019.

Part II

Papers

Paper A: Bayesian feature construction for case-based reasoning: Generating good checklists

Published in: *Proceedings of the International Conference on Case-Based Reasoning,* 2021.
Norwegian Scientific Index Level (NVI): *1*

Notes. This paper is published in the International Conference on Case-Based Reasoning (ICCBR), which is one of the top conferences in the CBR-field. The conference is ranked at level 1 by the Norwegian Scientific Index (NVI), which means that the publication channel satisfies the necessary requirements for being recognized as a scientific publication channel. Since the paper is a level 1 conference paper, it is therefore worth 0.7 publication points. The paper has been reformatted to ensure readability and consistency with the other papers included in the thesis.

Contribution Statement. My contribution to the paper includes the ideas presented in the paper, writing the first draft and carrying out the experiments in the paper. The co-authors of the paper are my supervisors, and they helped with the problem formulations and refining the paper.

BAYESIAN FEATURE CONSTRUCTION FOR CASE-BASED REASONING: GENERATING GOOD CHECKLISTS

Eirik Lund Flogard[1,2], **Ole Jakob Mengshoel**[2] and **Kerstin Bach**[2]
[1]Norwegian Labour Inspection Authority
[2]Norwegian University of Science and Technology
eirik.flogard@arbeidstilsynet.no, {ole.j.mengshoel, kerstin.bach}@ntnu.no

ABSTRACT

Checklists are used to aid the fulfillment of safety critical activities in a variety of different applications, such as aviation, health care or labour inspections. However, optimizing a checklist for a specific purpose can be challenging. Checklists also need to be trustworthy and user friendly to promote user compliance. With labour inspections as a starting point, we introduce the Checklist Construction Problem. To address the problem, we seek to optimize the content of labour inspection checklists in order to improve the working conditions in every organisation targeted for inspections. To do so, we introduce a hybrid framework called BCBR to construct trustworthy checklists. BCBR is based on case-based reasoning (CBR) and Bayesian inference (BI) and constructs new checklists based on past cases. A key novelty of BCBR is the use of BI for constructing new features in past cases. The augmented past cases are retrieved via CBR to construct new checklists, which ensures justification for the content of the checklists and promotes trust. Experiments suggest that BCBR is more effective than any other baseline we tested, in terms of constructing trustworthy checklists.

A.1 INTRODUCTION

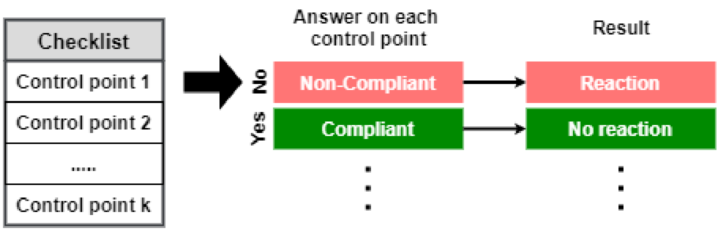


FIGURE A.1. Conceptual view of NLIA's procedure

Context. Every year more than three million workers are victims of serious accidents causing more then 4000 deaths due to poor working conditions in EU alone.[1] World-wide, it has been estimated that there are at least 9.8 million people in forced labour (2005) [1]. The most important measure to prevent poor working conditions is regulations. Regulations are usually enforced through labour inspections, which make them a vital part of the strategy employed by many countries to ensure good health,

[1]https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52014DC0332

safety, decent work conditions and well-being for workers (see UN's SDGs 3, 8 and 16[2]). Hence it is important to carry out labour inspections efficiently at large scale.

To identify poor working conditions, labour inspection agencies use surveys to check individual organisations for non-compliance [2]. Such procedures vary between different countries and we will use the Norwegian Labour Inspection Authority (NLIA) as an example. NLIA's inspection procedure is shown in Figure A.1. It consists of a checklist which is a set of control points that are answered during the inspection. Every control point is a question that corresponds to a specific regulation. The answer to each question indicates whether the inspected organisation is compliant or not. These answers provide a basis for reactions if non-compliance is found. Checklists for ensuring health and safety are also used in other domains such a surgery or flight procedures to ensure high accuracy of due diligence, and success often relies on correctly applying checklists [3].

Challenges with Checklists. Currently, labour inspection agencies operate with a limited, fixed number of static procedures or checklists targeting specific industries that organisations belong to. The inspectors select the checklist they subjectively believe is most relevant to the organisation they are visiting. A drawback with this approach is that the selected checklist can be poorly optimized for its target, while also being limited in terms of scope. This may prevent the inspections from fulfilling their purpose of addressing high risks to the workers' health, environment and safety. Checklists used for other applications such as aviation and health care may have similar problems where poorly optimized checklists can suffer from compatibility issues with users or contexts [3, 4]. This can have a negative effect on the users' motivation to use the checklists.

Contributions. We introduce the Checklist Construction Problem (CCP): Suppose that we have N unique questions with yes/no answers, where the answer to each question has an unknown probability distribution. Given the questions, construct a checklist for a target entity by selecting K unique questions that maximize the likelihood for obtaining no-answers to every selected question.

This problem could be applied to any domain where checklist optimization is an issue, such as healthcare or aviation. In these domains, the N unique questions may be designed to accomplish a specific task such as surgery or flight check and the target entity may be a patient or an aircraft. Any question with a likely no-answer should then be on the surgery or flight checklist so that yes-answers are obtained instead. However, this work focuses on solving CCP for labour inspections and introduces a new data set as a starting point to do so.

To solve CCP, we introduce BCBR, which is a framework based on Bayesian inference (BI) and case-based reasoning (CBR) for constructing new checklists optimized for a target organisation (entity). BCBR uses CBR to retrieve questions from checklists which have been used in past cases to survey organisations similar to the target organisation. BI is used to construct features in past cases which ensures that the retrieved questions have high probabilities for non-compliance. The approach starts with a data set of cases containing organisations and questions from previously used checklists. New features are then constructed by means of BI and added to each row in the data set to create augmented cases. The augmented cases are added to a case base which is queried using similarity based retrieval. The query contains a target probability

[2]https://sdgs.un.org/

and organisation, which is used to retrieve cases containing the questions for a new checklist (solution).

From a technical perspective, the use of augmented cases is a key novelty of BCBR that can be viewed as a data-driven approach that uses feature construction to embed solution knowledge in cases for case retrieval in CBR [5–7]. The use of BI to estimate probability ensures transparency because the estimates are made by counting cases in the data set. The use of similarity based retrieval also promotes trustworthiness and ensures justification of the BI estimates because they are related to past cases. Trustworthiness is important to ensure user compliance with the checklists. The core contributions of this paper are:

- We introduce a formal definition of the Checklist Construction Problem and a new data set of previously used questions (control points) collected from NLIA's labour inspections between 2012 and 2019.
- We present the details for BCBR, which is designed for constructing checklists based on CBR and Bayesian inference.
- We establish an approach for evaluating the checklists constructed by BCBR. The framework is then empirically compared to baselines. The results show that BCBR constructs more efficient checklists than the baselines.

A.2 Related Work

Hybrid Frameworks Based on CBR and BI. There are multiple examples of frameworks with combinations of CBR and BI to address uncertainty for applications where some prior belief or information is available. Such frameworks also provide explanations, where CBR has been used to achieve explanation goals [8] (such as transparency and justification) or generate explanations [9]. Nikpour et al. [7] use Bayesian posterior distributions to modify or add features to input case descriptions to increase accuracy of similarity assessments in case retrieval. They also use the same approach to provide explanations for case failures in different domains [10]. This approach is similar to BCBR, but BCBR constructs new features which are also added to the case base-cases rather than modifying input cases. Kenny et al. [11] also use a combination of BI and CBR to exclude outlier cases from case retrieval and to provide explanations by examples. The purpose of the framework is to predict grass growth for sustainable dairy farming. Gogineni et al. [12] combines CBR and BI to retrieve and down-select explanatory cases for underwater mine clearance.

Similarity Based Retrieval for Trustworthiness. Lee et al [13] replaced the output layer of a neural network with k-nearest neighbour (kNN) to generate voted predictions and find the nearest neighbour cases to explain the predictions. This also guarantees that every prediction can be justified by a relevant past explanatory case. The justification via explanatory cases increases the reliability of the neural network predictions and promotes trustworthiness. BCBR is also based on the same principle where BI predictions are justified by being embedded in past cases as features.

Trustworthy Case-Based Recommender Systems. BCBR aims to select a subset of all possible questions for a new checklist. Similarily, in recommender systems, a user is recommended a subset of items from the space of all possible items. Such systems can be divided into two classes: collaborative and case-based (content or user-based) recommender systems [14], where the latter approach could relate to our work. The case-based approach has been used to predict running-paces for different stages in ultra races, based on cases from similar runners in past cases [15]. CBR has also been used to provide explanatory cases for black-box recommender systems to achieve justification [16, 17]. Explanations for such systems can also be created through relations between features (concepts) [18]. However, the quality of explanations for black-box systems in terms of transparency, interpretability and trustworthiness can still be questionable [19]. Some authors also suggest to avoid explainable black box models in cases where they are not needed [20] and to use transparent, interpretable models for high-stakes decision making [19].

A.3 CASE AND PROBLEM DEFINITION

In this section we introduce the formal case and problem definition used for the rest of the paper.

Data Set and Cases. A data set $\mathcal{D}$ for variables $\mathbf{Z}$ is a finite length tuple where a case $\mathbf{d}_j \in \mathcal{D}$ is an instantiation of $\mathbf{Z}$ [21]. A case is a tuple $\mathbf{d} = (e, \mathbf{x}, l)$ where e denotes a question from a checklist, $\mathbf{x}$ is an entity and $l \in \{0, 1\}$ denotes the answer of the question. A case in the data set is a past experience where a question e has been applied to $\mathbf{x}$ to obtain the answer l. A case description is shown in Table A.1.

Name	*Description*	*Type*
x_{isc}	Industry subgroup code	Ordinal
x_{igc}	Industry group code	Ordinal
x_{ic}	Industry code	Ordinal
x_{iac}	Industry area code	Ordinal
x_{imac}	Industry main area code	Nominal
x_{mnr}	Municipality number	Ordinal
x_{fyl}	Fylke (county)	Nominal
e	Question	Nominal
l	Non-compliance	Binary

TABLE A.1. Description of a case in the data set

Entity. Every case d in the data set contains an entity description in the form of an organisation $\mathbf{x}$, defined by its location and industry. The features are organised according to Figure A.2. An organisation can be implicitly defined as $\mathbf{x} = (x_{mnr}, x_{isc})$, since the other features of $\mathbf{x}$ are located higher in the hierarchies.

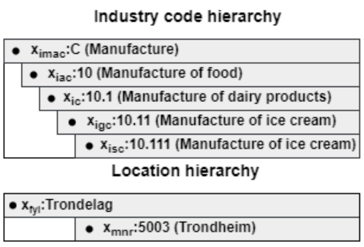

Industry code hierarchy
• x_imac:C (Manufacture)
• x_iac:10 (Manufacture of food)
• x_ic:10.1 (Manufacture of dairy products)
• x_igc:10.11 (Manufacture of ice cream)
• x_isc:10.111 (Manufacture of ice cream)
Location hierarchy
• x_fyl:Trondelag
• x_mnr:5003 (Trondheim)

FIGURE A.2. Industry and location hierarchies of an organisation

Question. Each case in the data set contains a question (control point) e with a yes/no answer. The question is used to survey the entity $\mathbf{x}$ in the case. A specific question can appear in multiple checklists.

Checklist. A checklist $\mathbf{y}$ is defined as a set of yes/no questions constituted by cases in the data set, so that $\mathbf{y} = (e_1 \in \mathbf{d}_1, e_2 \in \mathbf{d}_2 ... e_{nd} \in \mathbf{d}_{nd})$. A question can only appear once per checklist such that $e_i \neq e_j$ for every $e_i \wedge e_j \in \mathbf{y}$.

Answer. The label l of a case is the observed answer from applying the question e to the entity $\mathbf{x}$. The answer $l = 1$ means that non-compliance has been found, while $l = 0$ means that $\mathbf{x}$ is compliant.

The Checklist Construction Problem. The problem is shown on Figure A.3. Let there be a set of N unique questions and a new target entity $\mathbf{x}^{cnd}$. Each question has an unobserved answer l about $\mathbf{x}^{cnd}$ that belongs to an unknown distribution. Given the N questions, a model $\mathcal{M}$ first needs to correctly estimate the probability for observing $l = 1$ for each question. $\mathcal{M}$ then needs to select K unique questions $(e_1, e_2, ..., e_K)$, with the highest estimated probability, for a candidate checklist $\mathbf{y}^{cnd}$. The goal is to observe as many $l = 1$ answers as possible when applying $\mathbf{y}^{cnd}$ to $\mathbf{x}^{cnd}$.

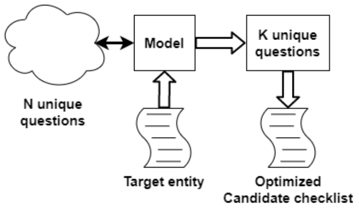


FIGURE A.3. An overview of CCP.

A.4 BCBR FRAMEWORK

An overview of the BCBR framework is shown in Figure A.4. The motivation for the framework is to solve the CCP problem while also ensuring that every question $e_i \in \mathbf{y}^{cnd}$ can be justified by a relevant past experience (see Section A.5.3). The framework can be described by the following three steps: (1) A naive Bayesian inference method is used to generate two probability estimates ($\theta^{be}_{x_{isc}}$ and $\theta^{be}_{x_{mnr}}$) for every case $\mathbf{d}_j \in \mathcal{D}$. The estimates are generated by counting the cases in the data set with the same question and entity description as $\mathbf{d}_j$. This is done because many of the cases in the data set contains identical questions and/or identical target entities. Using Bayesian inference also ensures transparency for the estimates. (2) A case base $\mathcal{CB}$ of augmented CBR cases $\mathbf{c}_j$ is created. Each case $\mathbf{c}_j \in \mathcal{CB}$ is created by adding both estimates as features to each $\mathbf{d}_j \in \mathcal{D}$. (3) A query $\mathbf{q}$ is defined, which contains a target entity $\mathbf{x}^{cnd}$ and target values for the probability estimates. The query is used to retrieve a selection of K cases from $\mathcal{CB}$. Each case contains a question e_i for the candidate checklist $\mathbf{y}^{cnd}$.

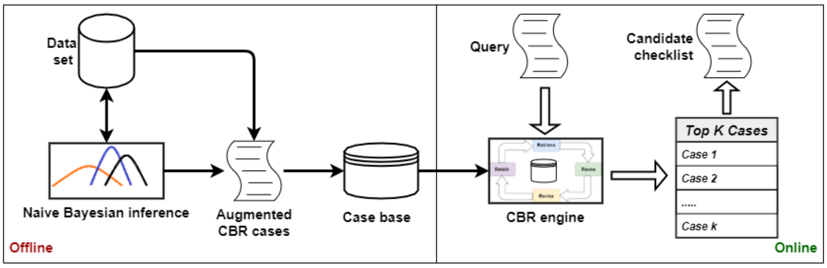


FIGURE A.4. An overview of the BCBR framework. The creation of augmented cases and the case base happens offline. The case base is used for the construction of checklists in the online-part.

A.4.1 Bayesian Inference

We use empirical distributions of the data set $\mathcal{D}$ to estimate the probability for observing $l = 1$, to achieve transparency for the BCBR framework. When prior knowledge or belief about l is available, BI can be used instead of the standard maximum likelihood method. An advantage with BI is that it (to some extent) can be used to address inaccurate empirical estimates caused low or zero case counts ("Zero count problem") [21]. The problem may have a negative impact on the quality of the K answers selected by BCBR. To further deal with this problem we use Naive Bayesian inference (NBI) which generates two probability estimates instead of just one. A derivation for this follows below.

Estimating the Empirical Probability for Non-compliance (l). By using the definitions from Section A.3, the empirical distribution of the data set $\mathcal{D}$ can be defined as:

$$\theta_D(\alpha) = \frac{\mathcal{D}\#(\alpha)}{\mathcal{N}} \tag{A.1}$$

where $\mathcal{D}\#(\alpha)$ is the number of cases in the data set $\mathcal{D}$ which satisfy the event α and $\mathcal{N}$ is the number of cases in $\mathcal{D}$ [21]. We denote the event $L = 1$ as observing the outcome $l = 1$ and $L = 0$ for $l = 0$. From the expression above, the probability for $L = 1$ can then be calculated given $\mathbf{x}$ and e:

$$\theta_D(L = 1|\alpha) = \frac{\theta_D(L = 1 \wedge \alpha)}{\theta_D(\alpha)} = \frac{\mathcal{D}\#(L = 1 \wedge \mathrm{X} = \mathbf{x} \wedge E = e)}{\mathcal{D}\#(\mathrm{X} = \mathbf{x} \wedge E = e)} \tag{A.2}$$

where $\alpha = (\mathrm{X} = \mathbf{x}) \wedge (E = e)$. That is, the event where the entity description is given as $\mathbf{x}$ and the question is given as e.

Naive Bayesian Inference for Estimating Empirical Probability (l). The posterior probability for an event $L = 1|\alpha$ can be expressed as the mean of a Beta distribution [21]:

$$\theta^{be}(L = 1|\alpha) = \frac{\mathcal{D}\#(L = 1 \wedge \alpha) + \psi_{L=1|a}}{\mathcal{D}\#(L = 1 \wedge \alpha) + \psi_{L=1|a} + \mathcal{D}\#(L = 0 \wedge \alpha) + \psi_{L=0|\alpha}} \tag{A.3}$$

where ψ is a set of prior belief parameters and where $(\mathcal{D}\#(L = 1 \wedge \alpha) + \psi_{L=1|a})$ and $(\mathcal{D}\#(L = 0 \wedge \alpha) + \psi_{L=0|a})$ are the parameters for a Beta distribution.

From the components x_{isc} and x_{mnr} of $\mathbf{x}$, two NBI probability estimates $\theta^{be}_{x_{isc}}$ and $\theta^{be}_{x_{mnr}}$ can be obtained from Equation A.3 by substituting α: $\theta^{be}_{x_{isc}} = \theta^{be}(L = 1|(X_{isc} = x_{isc} \wedge E = e))$ and $\theta^{be}_{x_{mnr}} = \theta^{be}(L = 1|(X_{mnr} = x_{mnr} \wedge E = e))$. Using two probability estimates instead of one is an effective measure against low case counts because $\mathcal{D}\#(X_{isc} = x_{isc} \wedge E = e) \geq \mathcal{D}\#(\mathrm{X} = \mathbf{x} \wedge E = e)$ and $\mathcal{D}\#(X_{mnr} = x_{mnr} \wedge E = e) \geq \mathcal{D}\#(\mathrm{X} = \mathbf{x} \wedge E = e)$. The approach is "naive" since it assumes that x_{mnr} and x_{isc} are independent given l and e.

A.4.2 Case Base Creation and CBR Engine

This section defines the details for the augmented CBR cases, case base and similarity based retrieval from Figure A.4.

Algorithm 1 Creation of a case base $\mathcal{CB}$ with cases $\mathbf{c}_j$

```
Input  : $\mathcal{D}$;
Output: $\mathcal{CB} \leftarrow ()$;
foreach $\mathbf{d}_j \in \mathcal{D}$ do
    $//(x_{isc,j}, x_{mnr,j}, e_j) \in \mathbf{d}_j$
    $\theta^{be}_{x_{isc}} \leftarrow \theta^{be}(L = 1|(x_{isc,j}, e_j))$;
    $\theta^{be}_{x_{mnr}} \leftarrow \theta^{be}(L = 1|(x_{mnr,j}, e_j))$;
    $\kappa_{x_{mnr}} \leftarrow \mathcal{D}\#(L = 1 \wedge X_{mnr} = x_{mnr,j} \wedge E = e_j)$;
    $\kappa_{x_{isc}} \leftarrow \mathcal{D}\#(L = 1 \wedge X_{isc} = x_{isc,j} \wedge E = e_j)$;
    $\mathbf{c}_j \leftarrow Join(\mathbf{d}_j, \theta^{be}_{x_{mnr}}, \theta^{be}_{x_{isc}}, \kappa_{x_{mnr}}, \kappa_{x_{isc}})$;
    $\mathcal{CB} \leftarrow Join(\mathcal{CB}, \mathbf{c}_j)$;
return $\mathcal{CB}$;
```

Augmented CBR Case and Case Base. Algorithm 1 shows the creation of a case base $\mathcal{CB}$ with augmented cases $\mathbf{c}$. The algorithm includes two additional features: $\kappa_{x_{mnr}}$ and $\kappa_{x_{isc}}$. The features are included to adjust for the case counts of the probability estimates when retrieving cases. The values for the θ^{be} and the κ-features are estimated from $\mathcal{D}$, given $x_{mnr,j}$, $x_{isc,j}$ and e_j from $\mathbf{d}_j \in \mathcal{D}$. The features are added to $\mathbf{d}_j$ to form a case $\mathbf{c}_j$ for $\mathcal{CB}$. An example showing the specific features of the augmented cases can be found in Section A.4.3.

Case Retrieval and Similarity Function. To retrieve questions e_i for the candidate checklist $\mathbf{y}^{cnd}$, a query case $\mathbf{q}$ and similarity function is used. The query consists of the target entity $\mathbf{x}^{cnd}$ and the desired values for both the probability estimates and the case count features. A similarity function assigns a score $Sim(\cdot,\cdot) \in [0, 1]$ to every pair $(\mathbf{q}, \mathbf{c}_j \in \mathcal{CB})$. A set of unique e_i for $\mathbf{y}^{cnd}$ is then retrieved from the K cases with the highest similarity score. The similarity function is defined according to the equation below:

$$Sim(\mathbf{q}, \mathbf{c}_j) = \frac{1}{\sum w_i} \sum_i w_i \cdot sim_i(\mathbf{q}, \mathbf{c}_j). \tag{A.4}$$

Where w_i is a weight, sim_i is a local similarity function and i denotes a feature common to the query and the case. Each local similarity function in Equation (A.4), yields a score $[0, 1]$ for each feature (i) according to the similarity $sim_i(\mathbf{q}, \mathbf{c}_j)$ between the cases $\mathbf{q}$ and $\mathbf{c}_j$. The local similarity functions and the weights are defined by a domain expert for the purpose of this work (see Section A.5.1).

A.4.3 Example: NBI Estimates, Case Retrieval and CBR Case

NBI Estimates. Let $x_{isc} = 22.230$, $x_{mnr} = 1507$ be features of an entity description $\mathbf{x}$ and e =“Did the employer make sure to equip all employees who carry out work at the construction site with a HSE card?” be a question of a case $\mathbf{d} \in \mathcal{D}$. The prior parameters are $\psi_{L=1|\alpha} = 1$ and $\psi_{L=0|\alpha} = 5$ because $l = 1$ is observed in approximately 1 of 6 cases. Given this information, $\theta^{be}_{x_{isc}}$ is estimated by counting cases $\mathbf{d}$ in data set $\mathcal{D}$

Feature	w	*Query 1*	*Case 1*	*Query 2*	*Case 2*
x_{isc}	1	22.230	22.230	22.230	22.230
x_{igc}	2	22.23	22.23	22.23	22.23
x_{ic}	2	22.2	22.2	22.2	22.2
x_{iac}	2	22	22	22	22
x_{imac}	2	C	C	C	C
x_{mnr}	2	1507	1507	1507	1507
x_{fyl}	2	MoM	MoM	MoM	MoM
l	0	-	0	-	0
e	0	-	e_1	-	e_2
$\theta^{be}_{x_{isc}}$	9	100%	22%	-	7%
$\theta^{be}_{x_{mnr}}$	4	100%	32%	-	7%
$\kappa_{x_{isc}}$	1	70	1	-	0
$\kappa_{x_{mnr}}$	1	70	89	-	30
Sim		-	0.546	-	0.448

TABLE A.2. Description of case features, similarity weights, query and retrieved case for the example.

which satisfy $X_{isc} = x_{isc}$ and $E = e$. Applying $\alpha = (X_{isc} = x_{isc} \wedge E = e)$ to Equation A.3 yields: $\theta^{be}_{x_{isc}} = \frac{1+1}{1+2+6} \approx 22\%$.

This estimate is more accurate than the empirical probability estimate, which is $\theta_{x_{isc}} = \frac{1}{1+2} \approx 33\%$ (Eq. A.2). The difference can be explained by low case count, which affect the quality of both the Bayesian and empirical estimates.

The same procedure is used to calculate: $\theta^{be}_{x_{mnr}} = \frac{89+1}{89+186+6} \approx 32\%$. In this case the Bayesian estimate is approximately the same as the empirical probability estimate, since the case count is high. The estimates are used to create an augmented CBR case **c**.

Case Retrieval and Augmented CBR Case. For this example we assume that a case base of CBR cases has been created and that $K = 1$, for the sake of brevity. The case retrieval starts by defining a query case (Query 1), shown in Table A.2. $\theta^{be}_{x_{isc}}$ and $\theta^{be}_{x_{mnr}}$ are set to 100%, which is the target value for the retrieved cases. Both $\kappa_{x_{isc}}$ and $\kappa_{x_{mnr}}$ are set to 70 so that case counts of 70 or higher yield full similarity scores, according to Figure A.5.

After applying the similarity function to every pair $(\mathbf{q}, \mathbf{c} \in \mathcal{CB})$, the top $K = 1$ case with highest similarity (Case 1) is retrieved for the candidate checklist $\mathbf{y}^{cnd}$.

For comparison, we also define Query 2 in Table A.2 where $\theta^{be}_{x_{isc}}$, $\theta^{be}_{x_{mnr}}$, $\kappa_{x_{isc}}$ and $\kappa_{x_{mnr}}$ are undefined. The $K = 1$ case returned from $\mathcal{CB}$ is Case 2. Case 2 fully matches Query 2 in terms of $\mathbf{x}$, but $\theta^{be}_{x_{isc}}$ and $\theta^{be}_{x_{mnr}}$ suggest

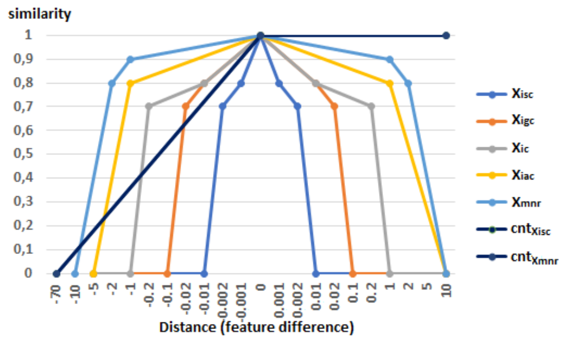


FIGURE A.5. Local similarity functions.

that it is unlikely to observe $l = 1$ when e_2 is applied to $\mathbf{x}$. This is expected because we removed the part of the query that maximizes the probability for observing $l = 1$.

A.5 EXPERIMENTS

In this section three experiments are presented. In the first experiment a simple label classification problem is introduced to establish a starting point for comparing ML methods as baselines for the labour inspection CCP. The second experiment aims to measure the justification of checklists constructed by BCBR and the two best-performing baselines from the first experiment. The third experiment aims to measure the performance of BCBR against the baselines from the second experiment.

A.5.1 Experimental Setup

Measure of Justification. We introduce Equation A.5 to measure the justification ($J \in [0, 100\%]$) of a checklist $\mathbf{y}$ for a given entity $\mathbf{x}$, according to the proportion of questions $e_i \in \mathbf{y}$ which also exist in past cases $(e_i, \mathbf{x}, \cdot) \in \mathcal{D}$.

$$J(\mathbf{y}, \mathbf{x}, \mathcal{D}) = \frac{|\{e_i \in \mathbf{y} : (e_i, \mathbf{x}, \cdot) \in \mathcal{D}\}|}{|\{e_i \in \mathbf{y}\}|} \tag{A.5}$$

The expression can be seen as an adaptation of Massie alignment score [22] that measures the percentage of questions $e_i \in \mathbf{y}$ with full alignment to the nearest neighbour case in $\mathcal{D}$.

BCBR Configuration. For the experiments, BCBR uses the same configuration as in Section A.4.3. The only difference is that $K = 15$ is used instead of $K = 1$, so that the constructed checklists consist of 15 questions.

The weights and local similarity functions are set based on domain knowledge and are shown in Table A.2 and Figure A.5 respectively. The weights are set according to the importance of each feature, while the similarity functions are defined to model the similarity according to the hierarchical relationship between the ordinal features of the entity $\mathbf{x}$ (see Section A.3). For the other features not shown in Figure A.5, the default option in the myCBR tool is used to define the local similarity functions.

Baselines for the Experiments. The baseline methods used for the experiments are: CBR (CBR-BL), Logistic Regression(LR), Decision tree (DT) and Naive Bayes classifier (NBC), Conditional probability estimates (CP), Bayesian inference (BI), Naive conditional probability (NCP) and NBI.

CBR-BL generates predictions from the label of the closest neighbour case in the training data. CP generates predictions for any pair $(e, \mathbf{x})$ according to Equation A.2. BI uses Equation A.3 with $\psi_{L=1|\alpha} = 1$, $\psi_{L=0|\alpha} = 5$ and $\alpha = (\mathbf{X} = \mathbf{x} \land E = e)$. NCP is based on Equation A.2 and is defined as: $\theta\,(L = 1|e, \mathbf{x}) = \frac{\theta_{x_{isc}} + \theta_{x_{mnr}}}{2}$. The baseline NBI estimates are calculated using $\psi_{L=1|\alpha} = 1$ and $\psi_{L=0|\alpha} = 5$ according to: $\theta\,(L = 1|e, \mathbf{x}) = \frac{\theta^{be}_{x_{isc}} + \theta^{be}_{x_{mnr}}}{2}$.

Environment. A Dell XPS 9570 with Intel i9 8950hk, 32GB RAM and Windows 10 were used for the experiments. Every experiment is conducted in a Python environment

Method	*Acc*	*Prec*	*Rec*	*Avg*	*Time*
CBR-BL	0.677	0.178	0.246	0.367	60238
Random	0.500	0.161	0.500	0.387	-
CP	0.680	0.210	0.357	0.416	**3.84**
BI	**0.760**	**0.270**	0.288	0.439	3.89
DT	0.644	0.233	0.529	0.469	122.6
NCP	0.592	0.250	0.761	0.534	9.0
NBC	0.588	0.251	0.778	0.539	67.33
LR	0.591	0.252	0.782	0.542	68.4
NBI	0.605	0.261	**0.790**	**0.552**	10.4

Table A.3. Results from the experiment. Time is measured in seconds per validation fold.

using Jupyter Notebook. NBI for BCBR, NBI, BI, CP and NCP are implemented as MSSQL17 queries via PYODBC. The similarity based retrieval for BCBR and CBR-BL are implemented via MyCBR [23]. The rest of the methods are implemented via Scikit-learn 0.24.

Data Set. For the experiments we introduce a new data set of questions used in previous inspections conducted by NLIA.[3] The data set is denoted as $\mathcal{D}$ for the rest of this section and consists of 1,111,502 entries from inspections conducted between 01/01/2012 and 01/06/2019. Embedded in these entries are $N = 1,967$ unique questions from checklists used in 59,988 inspections. Each entry (case) in $\mathcal{D}$ is also associated with an *id*[4] which maps to a checklist $\mathbf{y}$ (past solution) used to survey the organisation $\mathbf{x}$ in one of the 59,988 inspections within $\mathcal{D}$.

A.5.2 Experiment 1: Answer Classification Performance (Baselines)

The goal of this experiment is to compare ML methods and select two of the best as baselines for the labour inspection CCP. Because CCP is a complex problem, we here study a new, simple classification problem as a stepping stone.

The Answer Classification Problem. Let each $\mathbf{d}_j \in \mathcal{D}$ be a case with a two-class ground truth label l_j. A model $\mathcal{M}$ is trained on the cases in $\mathcal{D}$. For any new case $\mathbf{d} = (e, \mathbf{x}, l)$ where $l = 0$ (compliance) or $l = 1$ (non-compliance), the problem goal is for $\mathcal{M}$ to correctly classify the value of l based on $(e, \mathbf{x})$.

Method. Each model is validated on the data set $\mathcal{D}$, using 8-fold cross validation with the same partitioning of data for every model. Each model $\mathcal{M}$ outputs a class prediction score for every $(e, \mathbf{x})$. Thus, the classification threshold is set to the median of $\mathcal{M}$'s scores for each validation fold. The results are measured in terms of accuracy, precision and recall which are calculated for per validation fold: $Acc = \frac{TP+TN}{TP+FP+TN+FN}$, $Prec = \frac{TP}{TP+FP}$ and $Rec = \frac{TP}{TP+FN}$.

Results and Discussion. The results are shown in Table A.3 where the baselines are sorted according to *Avg*, which is the average score of the preceding columns. In

[3]The data set is available at https://dx.doi.org/10.21227/m1t7-hg51

[4]The *id* is a "key" for identifying a past checklist/organisation pair (value) in $\mathcal{D}$.

terms of the *Avg*-score NBI performs better then standard ML methods such as LR, DT and NBC. NBI also has the best recall and an average runtime of 10.4 seconds per validation fold, which is significantly less than NBC, DT, LR and CBR-BL. BI has the best performance in terms of accuracy and precision, but it also has poor recall which results in a low average score. The worst performing method was CBR-BL where the size of the training data was reduced to 100,000 cases due to long running time.

The results indicate that NBI yields the best average performance, which motivates us to combine NBI with CBR. LR, NBC and NCP also perform well, but we select NBI and LR as baselines for the next experiments. A limitation for this experiment is that it cannot be used to evaluate BCBR, as BCBR is designed for CCP and not ACP.

A.5.3 Experiment 2: Trustworthiness of Constructed Checklists

The goal of this experiment is to measure justification of constructed checklists $\mathbf{y}^{cnd}$ for the CCP. This is done by measuring the average proportion of questions $e_i \in \mathbf{y}^{cnd}$ which are justified by past cases. The experiment is based on Lee et al.'s use of past cases to justify predictions and promote trust [13]. The experiment is conducted on checklists constructed by BCBR and two of the baselines from Section A.5.2, NBI and LR.

Method. Each model $\mathcal{M}$ is trained on the data set $\mathcal{D}$ containing 1,111,502 entries. An evaluation data set $\mathcal{D}_V$ of 59,988 tuples $(\mathbf{x}^{cnd}, \mathbf{y})$ of past entity/checklist pairs is created using every unique *id* from $\mathcal{D}$. For each $\mathbf{x}^{cnd} \in \mathcal{D}_V$, $\mathcal{M}$ constructs a checklist $\mathbf{y}^{cnd}$ for $\mathbf{x}^{cnd}$ as following depending on the model in question. For $\mathcal{M} = NBI$ or $\mathcal{M} = LR$: $\mathcal{M}$ generates a prediction score for every unique $e_j \in \mathcal{D}$. The $K = 15$ questions with the highest prediction scores are selected as the candidate checklist $\mathbf{y}^{cnd}$ for $\mathbf{x}^{cnd}$. For $\mathcal{M} = BCBR$: a query containing $\mathbf{x}^{cnd}$ is defined to retrieve past cases, containing $K = 15$ unique questions for $\mathbf{y}^{cnd}$.

Each $\mathbf{y}^{cnd}$ constructed by one of the models $\mathcal{M}$ then forms an evaluation pair $(\mathbf{y}^{cnd}, \mathbf{x}^{cnd})$ with each corresponding $\mathbf{x}^{cnd}$ from $\mathcal{D}_V$. Based on Equation A.5, the average justification $(J_\mathcal{M})$ for every pair $(\mathbf{y}^{cnd}, \mathbf{x}^{cnd})$ given $\mathcal{M}$ is:

$$J_\mathcal{M}(\mathcal{D}, \mathcal{D}_V) = \frac{\sum_{(\mathbf{y}^{cnd}, \mathbf{x}^{cnd})} J(\mathbf{y}^{cnd}, \mathbf{x}^{cnd}, \mathcal{D})}{|\mathcal{D}_V|} \tag{A.6}$$

$J_\mathcal{M}$ measures the average percentage of questions $e_i \in \mathbf{y}^{cnd}$ where at least one corresponding explanatory case $(e_i, \mathbf{x}^{cnd}, \cdot)$ exists in $\mathcal{D}$. The purpose of the $J_\mathcal{M}$ score is to enable a fair comparison between the three models. A higher relative score means higher justification of the checklists constructed by $\mathcal{M}$.

Results and Discussion. The results are: $J_{NBI} = 0.6\%$, $J_{LR} = 4.8\%$ and $J_{BCBR} = 64\%$. This suggests that both LR and NBI perform poorly in terms of justification of their constructed checklists. Qualitative assessments of some of the checklists also reveal that many of their questions ($e_i \in \mathbf{y}^{cnd}$) are unrelated to and incompatible with the target entities. Because of the incompatibility issues and that less than 5% of the items on the checklists are justified, LR and NBI are not trustworthy. BCBR scored 64% which is significantly higher. Incompatible questions also seam to appear less frequently in BCBR's checklists.

A.5.4 Experiment 3: Evaluation of Constructed Checklists

The goal of this experiment is to evaluate the performance of the BCBR framework against LR, NBI and the original past checklists from the data set. Since BCBR uses similarity based retrieval, NBI and LR serve as non-similarity based baselines to compare with. Due to the results in Section A.5.3, a filter is applied to both LR and NBI to ensure that every checklist can be justified by past cases. This is necessary for the evaluation procedure, as it assumes that the questions on the checklists can be justified by past similar cases.

Method. The evaluation approach is done on the data set $\mathcal{D}$ which contains 1,111,502 entries. The approach can be summarized as following: The data set $\mathcal{D}$ is partitioned into a training fold ($\mathcal{D}_T$) and validation fold ($\mathcal{D}_{C\mathcal{B}}$), where the training fold is used to calculate probability estimates for the validation cases. The validation fold is used as the case base and for performance evaluation. A model $\mathcal{M}$ is trained on $\mathcal{D}_T$ and the evaluation is done on every checklist $\mathbf{y}^{cnd}$ constructed by $\mathcal{M}$.

A problem with the validation is that since every $\mathbf{y}^{cnd}$ is a new checklist, the ground truths l needed to evaluate $\mathbf{y}^{cnd}$ can be missing. A common solution to this problem is to collect the ground truth empirically [24], but this is not an option for us. To get a meaningful validation result, the performance statistics for the evaluation need to be estimated. To accomplish this, the following assumption is made: Let $\mathbf{d}^{cnd} = (-, \mathbf{x}^{cnd}, -)$ be a case without question component or observed ground truth answer and $\mathbf{d} = (e, \mathbf{x}, l)$ be any validation case with ground truth. If $\mathbf{x}^{cnd}$ and $\mathbf{x}$ are content-wise equal or similar, we assume that the unobserved ground truth answer l^{cnd} from applying e to $\mathbf{x}^{cnd}$ is correctly estimated from an empirical distribution of l, conditioned on $\mathbf{x}$, e and the validation data fold. This is based on the assumption that similar problems have similar solutions [5].

Based on the assumption, we introduce the following procedure to estimate accuracy (Acc), precision (Prec)[5] and recall (Rec) for every model $\mathcal{M}$.

1. Let $\mathcal{D}_T$ be the training fold and $\mathcal{D}_{C\mathcal{B}}$ be both the validation fold and case base(for BCBR). Let $\mathcal{D}_V$ be a set of past entity/checklist pairs $(\mathbf{x}^{cnd}, \mathbf{y})$ from $\mathcal{D}_{C\mathcal{B}}$, created using every unique *id* in $\mathcal{D}_{C\mathcal{B}}$. A model $\mathcal{M}$ is trained on $\mathcal{D}_T$.

2. For every $\mathbf{x}^{cnd} \in \mathcal{D}_V$, $\mathcal{M}$ selects K unique questions (e_i) for a checklist $\mathbf{y}^{cnd}$ to form a validation pair $(\mathbf{x}^{cnd}, \mathbf{y}^{cnd})$. The questions are selected from $\mathcal{D}_{C\mathcal{B}}$.

3. For each pair $(\mathbf{x}^{cnd}, \mathbf{y}^{cnd})$ the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) are estimated by evaluating each $e_i \in \mathbf{y}^{cnd}$(predicted positives) and $e_j \notin \mathbf{y}^{cnd}$(predicted negatives).

4. For every question $e_i \in \mathbf{y}^{cnd}$, both TP_{e_i} and FP_{e_i} are estimated using the following function: $f(l, \mathbf{x}_0, e_i) = \frac{\mathcal{D}_{C\mathcal{B}}\#(L=l \wedge \mathrm{X}=\mathbf{x}_0 \wedge E=e_i)}{\mathcal{D}_{C\mathcal{B}}\#(\mathrm{X}=\mathbf{x}_0 \wedge E=e_i)}$, so that $TP_{e_i} = f(1, \mathbf{x}_0, e_i)$ and $FP_{e_i} = f(0, \mathbf{x}_0, e_i)$. If $\mathcal{D}_{C\mathcal{B}}\#(\mathrm{X} = \mathbf{x}^{cnd} \wedge E = e_i) > 0$, then $\mathbf{x}_0 = \mathbf{x}^{cnd}$ is applied to f. If $\mathcal{D}_{C\mathcal{B}}\#(\mathrm{X} = \mathbf{x}^{cnd} \wedge E = e_i) = 0$, then $\mathbf{x}_0 = \mathbf{x}_i$ from the case $(e_i, \mathbf{x}_i, l_i)$, retrieved by BCBR[6] for $\mathbf{y}^{cnd}$, is used because there is no data to

[5]An additional statistic Prec(gt) is included, which is precision calculated (step 4-8) using only $e_i \in \{\mathbf{y}^{cnd} \cap \mathbf{y}\}$ from cases containing the original ground truth labels.

[6]The condition $\mathcal{D}_{C\mathcal{B}}\#(\mathrm{X} = \mathbf{x}^{cnd} \wedge E = e_i) = 0$ only occurs if BCBR is used.

Method	**Acc**	**Prec (gt)**	**Prec**	**Rec**	***Avg***
Org. CL	0.337	0.170	0.181	0.622	0.328
LR	0.484	0.226	0.267	0.694	0.418
NBI	0.486	0.229	0.270	0.698	0.421
BCBR	**0.574**	**0.259**	**0.343**	**0.718**	**0.474**

TABLE A.4. 8 fold cross validation results of the constructed vs. the original checklists (Org. CL).

evaluate $(e_i, \mathbf{x}^{cnd})$. Each TP_{e_i} and FP_{e_i} is assigned a value $v \in [0, 1]$ via f so that $TP_{e_i} = 1 - FP_{e_i}$.

5. For every unique question $e_j \notin \mathbf{y}^{cnd}$ in $\mathcal{D}_{C\mathcal{B}}$, both TN_{e_j} and FN_{e_j} are estimated using the function: $g(l, e_j \notin \mathbf{y}^{cnd}) = \frac{\mathcal{D}_{C\mathcal{B}}\#(L=l \wedge X=\mathbf{x}^{cnd} \wedge E=e_j)}{\mathcal{D}_{C\mathcal{B}}\#(X=\mathbf{x}^{cnd} \wedge E=e_j)}$. The function is used to obtain $TN_{e_j} = g(0, e_j)$ and $FN_{e_j} = g(1, e_j)$, so that each TN_{e_j} and FN_{e_j} receives a value of $v \in [0, 1]$ and that $TN_{e_j} = 1 - FN_{e_j}$.

6. TP, FP, FN and TN for each candidate checklist $\mathbf{y}^{cnd} \in (\mathbf{x}^{cnd}, \mathbf{y}^{cnd})$ are calculated as following: $TP = \sum_{e_i} TP_{e_i}$, $FP = \sum_{e_i} FP_{e_i}$, $TN = \sum_{e_j} TN_{e_j}$ and $FN = \sum_{e_j} FN_{e_j}$ for every unique $e_i \in \mathbf{y}^{cnd}$ and $e_j \notin \mathbf{y}^{cnd}$ from $\mathcal{D}_{C\mathcal{B}}$.

7. Statistics are then calculated for each $\mathbf{y}^{cnd}$: $Acc_{\mathbf{y}^{cnd}} = \frac{TP+TN}{TP+FP+TN+FN}$, $Prec_{\mathbf{y}^{cnd}} = \frac{TP}{TP+FP}$ and $Rec_{\mathbf{y}^{cnd}} = \frac{TP}{TP+FN}$. Repeat from Step 2 until every pair $(\mathbf{x}^{cnd}, \mathbf{y}^{cnd})$ is evaluated.

8. The average Acc, Prec and Rec of every checklist $\mathbf{y}^{cnd}$ constructed by $\mathcal{M}$ is: $Acc = \frac{\sum_{\mathbf{y}^{cnd}} Acc_{\mathbf{y}^{cnd}}}{|\mathcal{D}_V|}$, $Prec = \frac{\sum_{\mathbf{y}^{cnd}} Prec_{\mathbf{y}^{cnd}}}{|\mathcal{D}_V|}$ and $Rec = \frac{\sum_{\mathbf{y}^{cnd}} Rec_{\mathbf{y}^{cnd}}}{|\mathcal{D}_V|}$.

The procedure is used to evaluate BCBR and the other baselines. To evaluate the original checklists, the procedure is applied to the past checklists in the validation fold so that $\mathbf{y}^{cnd} = \mathbf{y}$ for $\mathbf{y} \in \mathcal{D}_V$ in Step 2. Step 2 for NBI and LR is done by generating predictions for every unique question (see Sect. A.5.3). Then a filter is applied after prediction and before the selection of the questions for $\mathbf{y}^{cnd}$. The filter excludes any question (e) from selection if $(e, \mathbf{x}^{cnd}, \cdot) \notin \mathcal{D}_{C\mathcal{B}}$. This means that every $e_i \in y^{cnd}$ is justified by a past case so that J_{NBI} and J_{LR} is 100% (Eq. A.6). The filter is necessary for the evaluation to ensure that NBI and LR construct checklists that satisfy the assumption above. The models use $K = 15$ and are validated using 4,8 and 16-fold cross validation.

Results and Discussion. The results are shown in in Table A.4. The *Avg* column shows the average of the four preceding columns, where the results suggest that the checklists constructed by NBI, LR and BCBR are more effective than the original checklists. BCBR scores 0.474 which is significantly higher than the original checklists and also higher than NBI and LR. Figure A.6 shows the results for different numbers of validation folds. The figure suggests that BCBR consistently outperforms NBI and LR in accuracy and precision. Also, both accuracy and precision statistics tend to increase with the size of the validation data sets. We believe this is caused by the fact that TP

and TN increases compared to FP and FN as the quality of the retrieved questions increases when more cases are available. Recall also decreases with the size of the validation data sets as the number of predicted positives is fixed ($K = 15$), which entails that FN increases more than TP when the size of the validation set increases. The experiment suggests that BCBR is more effective for constructing checklists than LR or NBI.

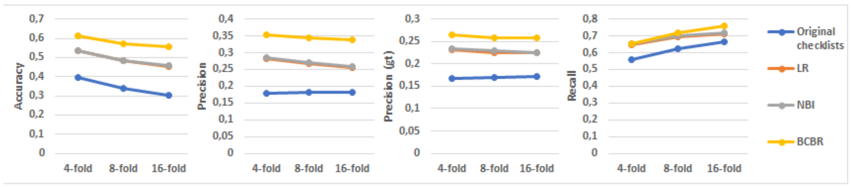


Figure A.6. Crossvalidation results for different validation fold sizes

A limitation of this experiment is that the results are based on estimates of Acc, $Prec$ and Rec. For CBR frameworks, the validity of the evaluation results partially depends on high similarity between the $\mathbf{x}$-part of the query and retrieved cases. This could be problematic when evaluating and comparing multiple CBR-based frameworks and should be investigated in future work.

A.6 Conclusion

In this paper we studied the problem of constructing checklists for safety critical applications, in particular labour inspections where constructing good high-performance checklists manually is difficult. Thus, we proposed the CCP where we consider the automatic construction of good, justifiable checklists. To address the CCP we introduced BCBR, which uses naive BI to construct features in CBR cases for retrieving questions for the checklists. We conducted three experiments on a data set of past labour inspections, which we introduced for the paper. Because CCP is a fairly complex problem, we conducted our first experiment on a simple answer classification problem. The goal of the experiment was to select two baselines for CCP, which was NBI and LR. In the second experiment we measured the justification of the checklist constructed by BCBR, NBI and LR, where we found that only BCBR constructs checklists which are justified by past cases. Another conclusion from the experiment is that questions selected for the constructed checklists should be justified in terms of prior use in similar entities, because some questions may be closely related to the entities that they originally were designed for. The results from the last experiment also suggest that BCBR is the most effective method for constructing checklists to address poor working conditions in inspected organisations. The checklists constructed by BCBR also perform significantly better than the original checklists.

One of the things that could be addressed in future work is solution adaptation, such as adapting questions after they have been retrieved for a checklist. Another option is to explore data-driven approaches to derive the weights and local functions for BCBR. It could also be interesting to see how BCBR perform in other CCPs such as surgery or preflight checklists.

A.7 References

[1] Patrick Belser. Forced labour and human trafficking: Estimating the profits. 2005.

[2] David Weil. If osha is so bad, why is compliance so good? *RAND Journal of Economics*, 27(3):620, 1996.

[3] Ken Catchpole and Stephanie Russ. The problem with checklists. *BMJ quality & safety*, 24(9):545–549, 2015.

[4] Asaf Degani and Earl L Wiener. *Human factors of flight-deck checklists: the normal checklist*. Ames Research Center, 1990.

[5] Bjørn Magnus Mathisen, Agnar Aamodt, Kerstin Bach, and Helge Langseth. Learning similarity measures from data. *Progress in Artificial Intelligence*, 2020.

[6] Thomas Gabel and Eicke Godehardt. Top-down induction of similarity measures using similarity clouds. *ICCBR 2015: Case-Based Reasoning Research and Development*, pages 149–164, 2015. doi: 10.1007/978-3-319-24586-7_11.

[7] Hoda Nikpour, Agnar Aamodt, and Kerstin Bach. Bayesian-supported retrieval in bncreek: A knowledge-intensive case-based reasoning system. In *Case-Based Reasoning Research and Development*, pages 323–338, 2018.

[8] Frode Sørmo, Jörg Cassens, and Agnar Aamodt. Explanation in case-based reasoning–perspectives and goals. *Artificial Intelligence Review*, 24(2):109–143, 2005.

[9] Juan A Recio-García, Belén Díaz-Agudo, and Victor Pino-Castilla. Cbr-lime: A case-based reasoning approach to provide specific local interpretable model-agnostic explanations. In *International Conference on Case-Based Reasoning*, pages 179–194, 2020.

[10] Hoda Nikpour and Agnar Aamodt. Fault diagnosis under uncertain situations within a bayesian knowledge-intensive cbr system. *Progress in Artificial Intelligence*, pages 1–14, 2021. doi: 10.1007/s13748-020-00227-x.

[11] Eoin M. Kenny, Elodie Ruelle, Anne Geoghegan, Laurence Shalloo, Micheál O'Leary, Michael O'Donovan, and Mark T. Keane. Predicting grass growth for sustainable dairy farming: A cbr system using bayesian case-exclusion and post-hoc, personalized explanation-by-example (xai). In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 172–187, 2019.

[12] Venkatsampath Raja Gogineni, Sravya Kondrakunta, Danielle Brown, Matthew Molineaux, and Michael T. Cox. Probabilistic selection of case-based explanations in an underwater mine clearance domain. In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 110–124, 2019. doi: 10.1007/978-3-030-29249-2_8.

[13] Ritchie Lee, Justin Clarke, Adrian Agogino, and Dimitra Giannakopoulou. Improving trust in deep neural networks with nearest neighbors. In *AIAA Scitech 2020 Forum*, page 2098.

[14] Derek Bridge, Mehmet H. Goker, Lorraine McGinty, and Barry Smyth. Case-based recommender systems. *The Knowledge Engineering Review*, 20(3):315–320, 2005.

[15] Cathal McConnell and Barry Smyth. Going further with cases: Using case-based reasoning to recommend pacing strategies for ultra-marathon runners. In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 358–372, 2019.

[16] Jose Jorro-Aragoneses, Marta Caro-Martinez, Juan Antonio Recio-Garcia, Belen Diaz-Agudo, and Guillermo Jimenez-Diaz. Personalized case-based explanation of matrix factorization recommendations. In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 140–154, 2019.

[17] Marta Caro-Martinez, Juan A. Recio-Garcia, and Guillermo Jimenez-Diaz. An algorithm independent case-based explanation approach for recommender systems using interaction graphs. In Kerstin Bach and Cindy Marling, editors, *Case-Based Reasoning Research and Development*, pages 17–32, 2019. doi: 10.1007/978-3-030-29249-2_2.

[18] Jose Luis Jorro-Aragoneses, Marta Caro-Martínez, Belén Díaz-Agudo, and Juan A Recio-García. A user-centric evaluation to generate case-based explanations using formal concept analysis. In *International Conference on CBR*, pages 195–210, 2020.

[19] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. doi: 10.1038/s42256-019-0048-x.

[20] Cynthia Rudin and Joanna Radin. Why are we using black box models in ai when we don't need to? *Harvard Data Science Review*, 1(2), 2019.

[21] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge University Press, 2009. doi: 10.1017/CBO9780511811357.

[22] Stewart Massie, Nirmalie Wiratunga, Susan Craw, Alessandro Donati, and Emmanuel Vicari. From anomaly reports to cases. In *International Conference on Case-Based Reasoning*, pages 359–373, 2007.

[23] Kerstin Bach, Bjørn Magnus Mathisen, and Amar Jaiswal. Demonstrating the mycbr rest api. In *ICCBR Workshops*, pages 144–155, 2019.

[24] Charlie Wang, Arpita Agrawal, Xiaojun Li, Tanima Makkad, Ejaz Veljee, Ole Mengshoel, and Alvin Jude. Content-based top-n recommendations with perceived similarity. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*.

Paper B: Stochastic local search heuristics for efficient feature selection: An experimental study

Published in: *Norsk IKT konferanse for forskning og utdanning (NIKT),* 2021.
Norwegian Scientific Index Level (NVI): 1

Notes. This paper is published in the NIKT conference, which is a Norwegian national conference in information technology. The conference is ranked at level 1 in the Norwegian Scientific Index (NVI) for 2021. Thus, the paper has been peer-reviewed according to the standards required for scientific publications and is worth 0.7 publication points as a level 1 conference paper. The paper has been reformatted to ensure readability and consistency with the other papers included in the thesis.

Contribution Statement. The first author of the paper is my supervisor, Ole Jakob Mengshoel. He wrote the first draft of the paper. My contribution to the paper is carrying out the experiments and writing the final version with the other authors. In particular, i was responsible for all the experiments and paper content related to the labour inspection domain and the checklist-dataset.

Stochastic local search heuristics for efficient feature selection: An experimental study

Ole Jakob Mengshoel[1], **Eirik Lund Flogard**[1,2], **Jon Riege**[3] and **Tong Yu**[4]
[1]Norwegian University of Science and Technology (NTNU).
[2]Norwegian Labour Inspection Authority.
[3]Boston Consulting Group.
[4]Carnegie Mellon University.

Abstract

Feature engineering, including feature selection, plays a key role in data science, knowledge discovery, machine learning, and statistics. Recently, much progress has been made in increasing the accuracy of machine learning for complex problems. In part, this is due to improvements in feature engineering, for example by means of deep learning or feature selection. This progress has, to a large extent, come at the cost of dramatic and perhaps unsustainable increases in the computational resources used. Consequently, there is now a need to emphasize not only accuracy but also computational cost in research on and applications of machine learning including feature selection. With a focus on both the accuracy and computational cost of feature selection, we study stochastic local search (SLS) methods when applied to feature selection in this paper. With an eye to containing computational cost, we consider an SLS method for efficient feature selection, SLS4FS. SLS4FS is an amalgamation of several heuristics, including filter and wrapper methods, controlled by hyperparameters. While SLS4FS admits, for certain hyperparameter settings, analysis by means of homogeneous Markov chains, our focus is on experiments with several realworld datasets in this paper. Our experimental study suggests that SLS4FS is competitive with several existing methods, and is useful in settings where one wants to control the computational cost.

B.1 Introduction

Context. Feature selection (FS), i.e., finding the best features or attributes among a large number of them, plays an important role in machine learning (ML), knowledge discovery, and data mining [1–4]. Reasons for feature selection include improved accuracy, explainability, understandability, and computational efficiency of the resulting machine learning model [1, 2]. There is an important distinction between filter and wrapper methods for feature selection [1]. The filter approach selects features in a preprocessing step, and the features selected do not depend on the ML algorithm used. The wrapper approach, in contrast, uses an ML algorithm as an integral part of the FS process. Variants of local search (such as backward selection and forward selection) have traditionally been employed for wrapper-based FS [2]. Methods such as genetic algorithms [5], regression [6], stochastic local search [7] and item sets [8] have also been used.

This paper encourages cross-fertilization between research on FS and stochastic local search (SLS). An SLS algorithm is a generalization of local search where stochasticity (or randomization) is also applied. Thus, SLS algorithms make occasional random search steps in order to avoid getting stuck or trapped in local but non-global optima. SLS is among the best methods to solve many computationally hard problems [9]. For example, SLS performs well in solving the satisfiability (SAT) problem [9, 10] as well as in computing the maximum a posteriori (MAP) hypothesis [11] and the most probable explanation (MPE) [12, 13] in Bayesian networks (BNs). SLS and its variants have also been widely used in other applications, such as sparse signal recovery [14], neural architecture search [15], sentence summarization [16], and subset selection [17, 18].

Problems. Clearly, recent advances in AI and ML have been impressive. At the same time, one may want to reflect on the massive computational, energy, and human resources brought to bear in today's AI and ML efforts, and how such resource consumption has recently increased. For example, OpenAI, a prominent AI development and deployment company, made the following observations on May 16, 2018:[1]

> [S]ince 2012, the amount of compute used in the largest AI training runs has been increasing exponentially with a 3.4-month doubling time (by comparison, Moore's Law had a 2-year doubling period). Since 2012, this metric has grown by more than 300,000x (a 2-year doubling period would yield only a 7x increase).

While a 300,000x growth in compute over a 6-year period is impressive, there may also reason to be concerned. If this growth continues, where does it lead to? What are the sustainability implications? And which individuals and organizations can afford to participate in and drive AI and ML research forward, if such massive compute resources are dramatically beneficial or (even worse) required to stay competitive?

Contributions. The contributions in this paper are in part motivated by concerns about the dramatic increases in resources being consumed in AI or ML. We make resource utilization, in particular computational cost measured in terms of compute time, more of a consideration compared to much previous research. At the same time, we acknowledge that FS research has, with notable exceptions [2, 19], often been heavily experimental.

In this paper, we integrate filter and wrapper methods for FS via SLS4FS, "Stochastic Local Search (SLS) for (4) Feature Selection (FS)," and provide experimental results. Compared to the most closely related work [7], SLS4FS adds and integrates three heuristics, detailed in this paper: soft greedy search, FS filters, and a randomized neighborhood relation [20]. Key intuitions underlying the results with our hybrid SLS4FS method include (i) in FS, the computational cost of a wrapper's greedy search step is typically much greater than the computational cost of a noisy or an initialization (or filter) search step and (ii) the greedy steps are still useful in FS for refining a filter's results when optimizing a feature subset. Our experiments with SLS4FS with three different classifiers and several real-world datasets provide further details about these intuitions and demonstrate the competitiveness of SLS4FS. SLS4FS is formulated such that a Markov Chain analysis is possible (see [7, 12]), however such analysis is not pursued here.

[1] https://openai.com/blog/ai-and-compute/

B.2 The Challenges of Feature Selection

We focus here on two goals for SLS: maximizing fitness and controlling computational cost. In SLS for FS specifically, maximizing fitness corresponds to maximizing ML model accuracy, and controlling computational cost corresponds to controlling the training time of ML models. We further discuss these two FS goals below.

Maximizing Fitness. We study search spaces consisting of bit-strings $\mathbb{B} = \{0, 1\}^n$. Fitness f is a pseudo-boolean function (PBF) that maps from $\mathbb{B}$ to the non-negative real numbers $\mathbb{R}_{\geq 0}$. Our focus is to optimize (without loss of generality, maximize) the fitness function f and find a global optimum $\boldsymbol{b}^*$:

$$\boldsymbol{b}^* = \arg\max_{\boldsymbol{b} \in \mathbb{B}} f(\boldsymbol{b}) . \tag{B.1}$$

When considering FS problems, a bit b in a bitstring $\boldsymbol{b}$ indicates whether a corresponding feature in a dataset is present ($b = 1$) or absent ($b = 0$) for purposes of learning. One can generalize (B.1) in order to handle multiple optima $\boldsymbol{b}_1^*, \boldsymbol{b}_2^*, \ldots$ and multiple fitness (or objective) functions $f_1, f_2, \ldots$. But we here keep it simple, in order not to complicate the notation and discussion.

Controlling Computational Cost. What is the time it takes for an algorithm to search for and find $\boldsymbol{b}^*$ or a good approximation? There are several factors, but let us highlight these. First, it depends on the cost g, for example compute time, it takes to compute $f(\boldsymbol{b})$ for a state $\boldsymbol{b} \in \mathbb{B}$: $g(\boldsymbol{b})$. Here, g maps from $\mathbb{B}$ to $\mathbb{R}_{\geq 0}$. Second, it depends on the complexity of the search landscape induced by f, and how that landspace interacts with the search algorithm. An SLS algorithm will in general evaluate $g(\boldsymbol{b})$ for many $\boldsymbol{b} \in \mathbb{B}$ when running. Given these factors, controlling computational cost of FS is an important goal. We are motivated by the fact that the amount of compute used in the largest AI training runs has recently increased exponentially with a 3.4-month doubling time[2] and with runtimes of days or weeks. In contrast, the compute time per experiment in Section B.4 is cut off at a maximum of 100 seconds, thus enabling an exploratory and human-centric workflow that includes ML as a component.

Hyperparameters and Heuristic Settings. In order to maximize fitness and control cost as discussed above, we need to optimize SLS hyperparameter and heuristic settings. This could be done via recent hyperparameter tuning methods including Bayesian optimization (BO) [21–23]. Unfortunately, while such methods are often general and mathematically elegant, they do not always scale well [24]. We focus in this paper on fast experiments with SLS under limited computing resources, on the order of seconds or minutes, while BO typically takes hours or days.

Multi-objective optimization algorithms (MOOAs), for example multi-objective evolutionary algorithms [25, 26], may seem like another alternative to adress the above two goals related to fitness f and cost g. A MOOA would compute a Pareto front that approximates Pareto optimality, trading off the two objectives of fitness f and computational cost g of the resulting model, for example a classifier. However, we are in this paper interested in a different problem. We study the computational cost g of the SLS process itself when searching for $\boldsymbol{b}^*$, a bitstring of maximal fitness $f^* = f(\boldsymbol{b}^*)$.

Specifically, we study the effect of varying several heuristics and hyperparameters for real-world FS problems. This is a challenge, as both the FS problems and the

[2] See above and https://openai.com/blog/ai-and-compute/.

configuration of SLS4FS are potentially high-dimensional and complex multi-modal search spaces.

B.3 SLS4FS: SLS for Feature Selection

We now discuss our algorithm for SLS-based feature selection, "Stochastic Local Search (SLS) for (4) Feature Selection (FS)" (or SLS4FS). The algorithm is presented briefly in previous work [20]. In this paper we provide a detailed discussion of SLS4FS, including pseudo-code, and its performance in three experiments with real-world datasets. In this section we first discuss the components or search steps of SLS4FS in Section B.3.1 before presenting the algorithm's overall structure in Section B.3.2.

B.3.1 SLS Search Steps

Local search takes place in the proximity, or in the neighborhood, of the current state $\boldsymbol{b}$, and we introduce these definitions.

Definition 1 (Neighborhood) *Let $\boldsymbol{b} = b_1 \dots b_i \dots b_n \in \{0,1\}^n$ and $\boldsymbol{b}' = b'_1 \dots b'_i \dots b'_n \in \{0,1\}^n$. Further, use $\oplus$ for exclusive or and define Hamming distance H as $H(\boldsymbol{b}', \boldsymbol{b}) = \sum_{i=1}^{n}(b'_i \oplus b_i)$. The neighborhood $N(\boldsymbol{b}) \subset \{0,1\}^n$ of $\boldsymbol{b}$ is defined as all bitstrings with a Hamming distance H of one to $\boldsymbol{b}$: $N(\boldsymbol{b}) = \{\boldsymbol{b}' \in \{0,1\}^n \mid H(\boldsymbol{b}', \boldsymbol{b}) = 1\}$.*

In FS [3] and other problems with non-trivial g, working with the neighborhood $N(\boldsymbol{b})$ may be too compute-intensive and we therefore introduce a subset of it, $N(\boldsymbol{b}, N_r)$, as follows.

Definition 2 (Randomized Neighborhood) *Let $N_r \in \mathbb{N}^+$ with $0 < N_r \leq n$. A randomized neighborhood is defined as a set $N(\boldsymbol{b}, N_r) \subseteq N(\boldsymbol{b})$:*

$$N(\boldsymbol{b}, N_r) = \{\boldsymbol{b}' \in N(\boldsymbol{b}) | \boldsymbol{b}' \text{ is picked randomly from } N(\boldsymbol{b})\}, \tag{B.2}$$

such that $|N(\boldsymbol{b}, N_r)| = N_r$.

"Picked randomly" in (B.2) means "picked uniformly at random without replacement." The computational benefit of $N(\boldsymbol{b}, N_r)$ compared to $N(\boldsymbol{b})$ is perhaps clearer when considering SLS4FS search, which we do now.

SLS4FS seeks to find or approximate an optimal state $\boldsymbol{b}^*$. This is done via repeated application of a greedy step GreedyStep, a noise step NoiseStep, and a restart step RestartStep, given the current state $\boldsymbol{b}$. These search steps are formally defined as follows.

Definition 3 (Greedy Step) *A greedy step GreedyStep($\boldsymbol{b}$, N_r, f, L, D) computes from bitstring $\boldsymbol{b}$ a bitstring $\boldsymbol{b}' \in N(\boldsymbol{b}, N_r)$ while maximizing the objective function $f(\boldsymbol{b}')$ by using the learning algorithm L with dataset D. If there is a tie in $f(\boldsymbol{b}')$ among neighbors $N(\boldsymbol{b}, N_r)$, one of these neighbor is picked uniformly at random. A* strict *greedy step* GreedyStep_s *stays with $\boldsymbol{b}$ if $f(\boldsymbol{b}) \geq f(\boldsymbol{b}')$ for all $\boldsymbol{b}' \in N(\boldsymbol{b}, N_r)$, while a* loose *or* soft *greedy step* GreedyStep_ℓ *always moves to the best-fit neighbor.*

The GreedyStep highlights the wrapper nature [1] of SLS4FS, via its use of the learning algorithm L. Definition 3 introduces two variants of the GreedyStep. For both variants we move to a neighbor $\boldsymbol{b}' \in N(\boldsymbol{b})$. The first variant, for $N_r = n$, is a *complete* greedy step. The second variant, for $N_r < n$, is a *randomized* greedy step. This randomized variant's purpose is to reduce the computational cost of a complete greedy step. Thus, when $N_r < n$, GreedyStep($\boldsymbol{b}$, N_r, f) in Definition 3 employs randomization as follows. First, it randomly chooses N_r neighbors among $N(\boldsymbol{b})$ and then picks a neighbor maximizing the objective function $f(\boldsymbol{b}')$ among them. Intuitively, with high-dimensional datasets (large n) and $N_r \ll n$, this provides substantial computational saving relative to using $N(\boldsymbol{b})$, at the obvious drawback of not necessarily finding the best neighbor.

The goal in FS is to find a global optimum $\boldsymbol{b}^*$ (a best feature subset) or an approximation thereof. However, in many cases there are in FS problems local but non-global optima, as demonstrated in Section B.4.2, where search can get stuck. SLS4FS contains two search operators, the NoiseStep and the RestartStep, to handle this problem.

Definition 4 (Noise Step) *A noise step NoiseStep($\boldsymbol{b}$) randomly jumps from bitstring $\boldsymbol{b}$ to a neighbor $\boldsymbol{b}' \in N(b)$; note that $\boldsymbol{b}' \neq \boldsymbol{b}$.*

While there are different ways to randomize noise steps [12, 27], we focus in this paper on the simple and easy-to-analyze NoiseStep of picking a neighbor uniformly at random.

Definition 5 (Restart Step) *A restart step RestartStep(F,D) randomly computes a bitstring $\boldsymbol{b} \in \{0, 1\}^n$, using a filter algorithm F with a dataset D.*

Clever restart and initialization algorithms can have a very positive impact on SLS optimization [13, 27]. For FS specifically, a filter F should ideally start search close to $\boldsymbol{b}^*$ at low computational cost. For the RestartStep, we study in this paper both naive methods and more advanced filter algorithms from the FS literature [2–4]. We consider the following three naive filter methods.

Definition 6 (Zeros Filter) *The all-zeros filter F_{0s} creates an initial feature subset in this way: We set each bit b_i to zero, $b_i = 0$, $1 \leq i \leq n$.*

Definition 7 (Ones Filter) *The all-ones filter F_{1s} creates an initial feature subset in this way: We set each bit b_i to one, $b_i = 1$, $1 \leq i \leq n$.*

Definition 8 (Uniform (at Random) Filter) *The uniform at random filter F_U creates an initial feature subset in this way: For each bit b_i, where $1 \leq i \leq n$, we flip an unbiased coin. If the coin comes up heads, we set $b_i = 1$. If it comes up tails, we set $b_i = 0$.*

The advanced filters studied can be defined as follows.

Definition 9 (Score-Based Filter) *A score-based filter creates an initial feature subset by assigning a score to every feature. For each bit b_i, if the corresponding feature's score is in the 90th percentile or above, $b_i = 1$, else $b_i = 0$.*

The following 8 score-based filters are used in this paper: mutual information (F_{MI}), ANOVA (F_{A}), χ^2 (F_{χ^2}), variance (F_{V}), random forrest impurity (F_{RFI}), random forrest permutations (F_{RFP}), lasso regression (F_{LR}) [28], and ridge regression (F_{RR}) [29]. Both the advanced, score-based filters as well as the naive F_{U}, $F_{0\mathrm{s}}$, and $F_{1\mathrm{s}}$ filters are studied experimentally in Section B.4.

Algorithm 1: SLS for FS (SLS4FS).

```
Input  : Probability of restart $P_r$, noise step $P_n$, dataset $D$ with number of
         instances $d$ and features $n$, machine learner $L$, filter $F$, accuracy
         $f(\boldsymbol{b}, D, L)$ for $L$ on dataset $D$ with subset $\boldsymbol{b}$, termination threshold $\tau$,
         strict or soft GreedyStep $X$, number of neighbors $N_r$ for the
         GreedyStep.
Output : Optimized feature subset $\boldsymbol{b}^+$
$\boldsymbol{b} \leftarrow$ RestartStep($F$,$D$), $c \leftarrow 0$, $f^+ \leftarrow 0$, $\boldsymbol{b}^+ \leftarrow \boldsymbol{b}$
while ¬ Terminate() do
    $c \leftarrow c + 1$
    if $Rand(0,1) < P_r$ then
        $\boldsymbol{b} \leftarrow$ RestartStep($F$,$D$) { Def. 5 }
    else
        if $rand(0,1) < P_n$ then
            $\boldsymbol{b} \leftarrow$ NoiseStep($\boldsymbol{b}$) { Def. 4 }
        else
            $old\_b \leftarrow \boldsymbol{b}$
            $\boldsymbol{b} \leftarrow$ GreedyStep$_X$($\boldsymbol{b}, N_r, f, L, D$) { Def. 3 }
            if $old\_b = \boldsymbol{b}$ then
                { We may be stuck in a local optimum }
                $P_r = P_r + \bar{P}_r\alpha_r$ { Increase $P_r$ }
                $P_n = P_n + \bar{P}_n\alpha_n$ { Increase $P_n$ }
            else
                $P_r = P_r(1 - \alpha_r/2)$
                $P_n = P_n(1 - \alpha_n/2)$
    if $f(\boldsymbol{b}) > f^+$ then
        { Update the current best subset $\boldsymbol{b}^+$ }
        $f^+ \leftarrow f(\boldsymbol{b}, D, L)$
        $\boldsymbol{b}^+ \leftarrow \boldsymbol{b}$
return $\boldsymbol{b}^+$
```

B.3.2 The SLS4FS Algorithm

The SLS4FS algorithm, summarized in Algorithm 1, searches the bitstring space $\{0, 1\}^n$ representing feature subsets while optimizing accuracy f as discussed below. An optimized feature subset, represented as a bitstring $\boldsymbol{b}^+$, is the output of SLS4FS. SLS4FS is tailored to FS [20] but is based on previous SLS algorithms [7, 12].

Input and Output. Given a machine learner L, the objective function f is computed in wrapper fashion [1]. For a feature subset $\boldsymbol{b}$, we run L on D using the features indicated by $\boldsymbol{b}$; $f(\boldsymbol{b})$ gives the estimated accuracy of L [1, 20]. The estimated accuracy is obtained by cross-validation (CV) on a training set or by validation on a separate test set.

Initialization and Search. SLS4FS starts, using RestartStep(F,D), from a random

ID	Name	# Features n	# Instances d
1	breast cancer (UCI)	9	700
2	m-of-n-3-7-10 (UCI)	10	1,324
3	checklists	575	63,634
4	cleve (UCI)	9	202
5	madelon [30]	500	2,000
6	bioresponse [31]	1,776	3,751
7	gas-drift (UCI)	128	13,910
8	crime (UCI)	124	2,215

Table B.1. Datasets for experimental evaluation.

initial state $\boldsymbol{b}$ among the 2^n possible bitstrings.[3] RestartStep(F,D) initializes a feature subset using a filter method F operating on the dataset D. In each search step, SLS4FS performs (i) a GreedyStep with probability $(1 - P_r)(1 - P_n)$; (ii) a NoiseStep with probability $(1 - P_r)P_n$; or (iii) a RestartStep with probability P_r. During search, SLS4FS keeps track of a best-so-far $\boldsymbol{b}^+$. If, for the i-th search step $f(\boldsymbol{b}) > f(\boldsymbol{b}^+)$, then $\boldsymbol{b}$ is the new best-so-far. Upon termination, SLS returns $\boldsymbol{b}^+$ as an approximation to $\boldsymbol{b}^*$. Hyperparameters α_r and α_n can be used to dynamically adapt P_r and P_r.[4]

Termination. Different termination criteria can be used in SLS4FS, as suggested by Terminate() in Algorithm 1. In this work, SLS4FS terminates upon reaching an upper bound on compute time τ or when a local optimum is found.

We identify a special case of SLS4FS when only initialization is randomized:

Definition 10 (Purely greedy SLS4FS) *SLS4FS run with input parameters $P_r = 0$, $P_n = 0$, $N_r = n$, GreedyStep$_s$, $F = F_U$, and Terminate() at a local optimum is denoted* purely greedy SLS4FS.

Remark 1. A key advantage of the SLS4FS algorithm is its ability to handle local optima due to its randomized NoiseStep and RestartStep. Further, the RestartStep can provide fruitful starting points due to the use of a filter F in RestartStep(F, D). For simplicity, one can use an all-zeroes filter F_{0s}. Alternatively, a more advanced filter F may result in SLS4FS computing better feature sets.

Remark 2. When applying SLS to FS, a significant difference from many other applications of SLS is the greatly varying computational cost of search steps.

We study both of these points, and others, in experiments in Section B.4.

B.4 Experimental Results

We conduct three experiments to analyze SLS4FS. First, in Section B.4.2, we analyze several FS problems for real-world datasets to find out if they exhibit local optima. One

[3] Note that (i) a deterministic initialization, for example at $\boldsymbol{b} = 0...0$ using F_{0s}, is a special case of a randomized initialization and (ii) randomized initialization may or may not be uniformly at random.

[4] To enable a homogeneous Markov chain analysis [7, 12], one can use probabilistic restart and not adapt the probability parameters P_r and P_n when running SLS4FS [7, 24].

of the advantages of SLS4FS is the possibility of it using noise and restart steps to escape local optima in the search space. In the second experiment, in Section B.4.3, we evaluate the performance of SLS4FS when varying its noise parameter, restart parameter, and the filter. Third, in Section B.4.4, we compare and analyze the performance of SLS4FS against other FS methods on several real-world datasets.

B.4.1 Datasets and Methods

Using Naive Bayes, Decision Tree, and Support Vector Machine classifiers, we conduct experiments on 8 real-world datasets to validate our hybrid SLS4FS method. For this work, we focus on small and medium sized datasets to ensure that meaningful results can be obtained within a reasonable amount of time.[5] Most of these are also real-world datasets, which are often limited in size. Table B.1 lists the datasets with the number of features and instances used in our experiments. The datasets are taken from the UCI repository[6] or the literature. We have also included a new dataset called checklists.[7] A similar dataset has also been used for constructing new checklists [32]. Experiments 2 and 3 are executed in NTNU's IDUN cluster environment, using one CPU and 24 GB of memory per run. Implementations are in Python using Scikit-learn.[8]

B.4.2 Experiment 1: Multiple Local Optima in Feature Selection

Goal. The challenge of local but non-global optima in FS motivated us to design the noise and restart steps in SLS4FS. To what extent does FS for real-world datasets exhibit such local optima?

Method and Data. To investigate this question, we run purely greedy SLS4FS (see Definition 10) 1,000 times for three FS problems. SLS4FS terminates upon finding a bitstring $\boldsymbol{b}^+$ with higher accuracy than all neighbors $N(\boldsymbol{b}^+)$. We use a Decision Tree model as L in SLS4FS and compute accuracy $f(\boldsymbol{b}^+)$ of the model. For each problem, we report a histogram reflecting the model's accuracy, see the plots in Figure B.1.

Results and Discussion. We report the results for three small-scale problems (breast cancer, m-of-n-3-7-10, and cleve) to get a comprehensive picture. The results are shown in Figure B.1. Among the 1,000 experiments, about 600 find suboptimal solutions on the breast cancer dataset. Similar observations can be made for the other two datasets; clearly m-of-n-3-7-10 is most difficult in that about 730 experiments terminate with suboptimal solutions. These results suggest that real-world FS problems contain local but non-global optima. Local optima are problematic for traditional greedy FS algorithms, like ForwardSelection and BackwardSelection. At the same time,

[5] Some FS methods can take hours or even days to fully complete on one of our medium-sized datasets.

[6] UCI repository: http://archive.ics.uci.edu/ml/

[7] The checklists dataset consists of 63,634 inspections conducted by the Norwegian Labour Inspection Authority. The dataset contains 575 features related to the economical and organisational information about the target organisation. Each instance also has a binary target label, denoting whether the target organisation was found non-compliant or not at the inspection.

[8] https://scikit-learn.org/stable/.

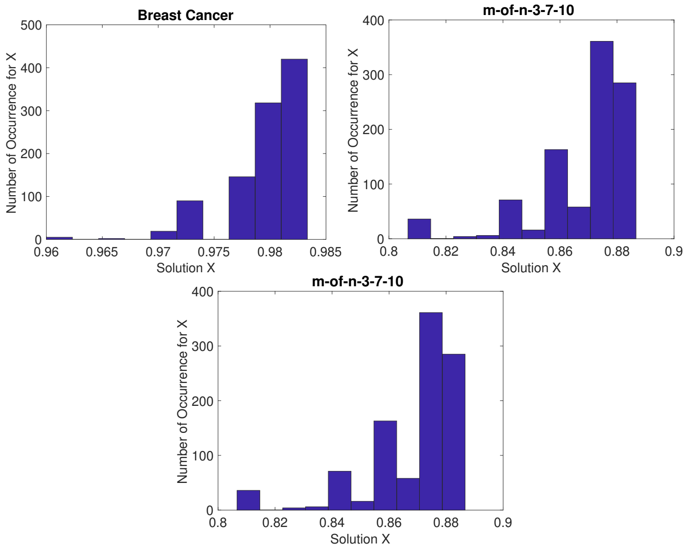


FIGURE B.1. Experimental FS results for three datasets breast cancer, m-of-n-3-7-10, and cleve. In each panel, the x-axis shows the accuracies achieved by solutions (feature subsets) computed by SLS4FS for a dataset. The y-axis counts the number of occurrences at each accuracy level. This clearly demonstrates that there are multiple local optima in each of these FS problems.

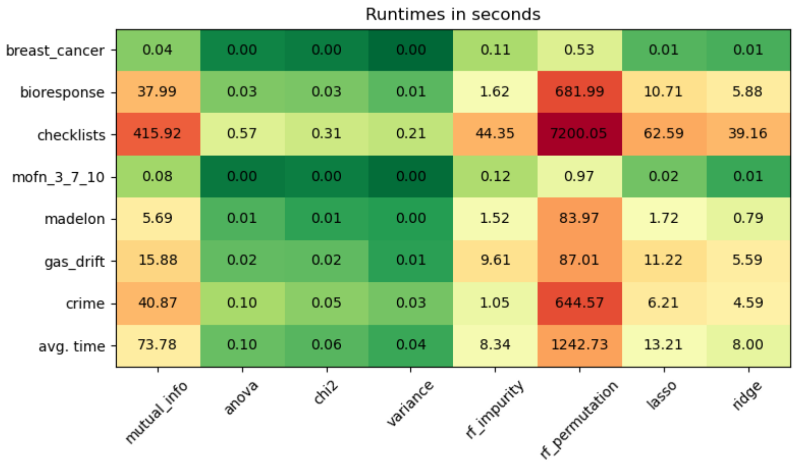
the randomization in SLS algorithms such as SLS4FS are able to handle local optima, using $P_r > 0$ or $P_n > 0$. Motivated by these results, we carefully study the impact and optimization of P_r, P_n, and other SLS4FS heuristics in Section B.4.3.

B.4.3 Experiment 2: Varying SLS4FS Heuristics and Settings

Goal. How do different settings of heuristics and hyperparameters impact SLS4FS's accuracy? We assess how SLS4FS performance is affected by varying these heuristics: strict versus soft Greedy, F, P_n, and P_r.

Dataset	Classifier	Time τ (sec)	N_rn
breast cancer	SVM	0.2	1.0
m-of-n-3-7-10	SVM	2.0	1.0
madelon	DT	60.0	1.0
bioresponse	NB	60.0	0.1
checklists	DT	60.0	0.1
gas-drift	DT	60.0	0.2
crime	NB	10.0	1.0

TABLE B.2. Experimental setting in Section B.4.3. The ratio of neighbors considered in the greedy step is N_r/n.



	mutual_info	anova	chi2	variance	rf_impurity	rf_permutation	lasso	ridge
breast_cancer	0.04	0.00	0.00	0.00	0.11	0.53	0.01	0.01
bioresponse	37.99	0.03	0.03	0.01	1.62	681.99	10.71	5.88
checklists	415.92	0.57	0.31	0.21	44.35	7200.05	62.59	39.16
mofn_3_7_10	0.08	0.00	0.00	0.00	0.12	0.97	0.02	0.01
madelon	5.69	0.01	0.01	0.00	1.52	83.97	1.72	0.79
gas_drift	15.88	0.02	0.02	0.01	9.61	87.01	11.22	5.59
crime	40.87	0.10	0.05	0.03	1.05	644.57	6.21	4.59
avg. time	73.78	0.10	0.06	0.04	8.34	1242.73	13.21	8.00

Figure B.2. Runtimes for 8 different FS filters (on x-axis) on 7 different datasets (on y-axis). Runtimes are classified from extremely low (dark green) to extremely high (dark red). The filters are, left to right, mutual information (F_{MI}), ANOVA (F_{A}), χ^2 (F_{χ^2}), variance (F_{V}), random forrest impurity (F_{RFI}), random forrest permutations (F_{RFP}), lasso regression (F_{LR}) and ridge regression (F_{RR}). Per-filter runtime averages, which vary quite dramatically, are in the bottom row. Figure B.3 contains corresponding accuracies.

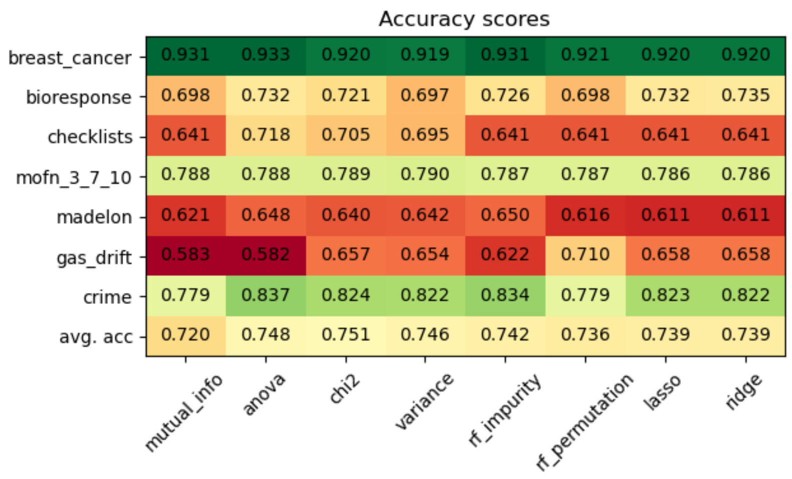
Method and Data. To inform the F-parameter of SLS4FS, we test different FS filters and record runtime and accuracy. These filters are either well-known or prominent in the FS literature [4, 33]; see Section B.3. Runtimes are shown in Figure B.2; F_{χ^2} or F_{V} have the best performances on almost every dataset. F_{χ^2} also has the highest recorded mean accuracy when applied to the datasets, see Figure B.3. Thus, for our experiment below we use $F = F_{\chi^2}$ in our SLS4FS algorithm. Further, we include $F = F_{0\text{s}}$ as a simple baseline.

Based on these filter results, we test each configuration of SLS4FS using several datasets, see Table B.2. Each problem consists of a dataset, an ML method L, a compute time bound τ, and a neighborhood size N_r. The neighborhood size determines the fraction of neighbors included in the randomized neighborhood. The time limit and neighborhood size are set to limit the overall running time of SLS4FS. SLS4FS hyperparameters and heuristics that are varied are: P_n, P_r, F, and GreedyStep$_s$ versus GreedyStep$_\ell$.

Every evaluation in the experiment is done by restricting the dataset based on the SLS4FS-selected features, initializing the ML method, training the ML method on 2/3 of the dataset and calculating the accuracy based on the other 1/3.

Results and Discussion. Four configurations of SLS4FS, each tested with 20 different values of noise probability P_n, are shown in Figure B.4.[9] The plots are consistent with findings about local optima in Section B.4.2 as well as previous Markov chain

[9] Other configurations were also tested, but the ones in Figure B.4 illustrate the main findings well.


Accuracy scores

	mutual_info	anova	chi2	variance	rf_impurity	rf_permutation	lasso	ridge
breast_cancer	0.931	0.933	0.920	0.919	0.931	0.921	0.920	0.920
bioresponse	0.698	0.732	0.721	0.697	0.726	0.698	0.732	0.735
checklists	0.641	0.718	0.705	0.695	0.641	0.641	0.641	0.641
mofn_3_7_10	0.788	0.788	0.789	0.790	0.787	0.787	0.786	0.786
madelon	0.621	0.648	0.640	0.642	0.650	0.616	0.611	0.611
gas_drift	0.583	0.582	0.657	0.654	0.622	0.710	0.658	0.658
crime	0.779	0.837	0.824	0.822	0.834	0.779	0.823	0.822
avg. acc	0.720	0.748	0.751	0.746	0.742	0.736	0.739	0.739

FIGURE B.3. Accuracy scores for 8 different FS filters (on x-axis) on 7 different datasets (on y-axis). Per-filter accuracy averages are in the bottom row. While χ^2 (F_{χ^2}) has the highest average accuracy, the differences in averages for the best methods are very small and not statistical significant. This table corresponds to the table with runtimes in Figure B.2.

analyses and experiments with SLS [12, 13, 34]: P_n being varied clearly has a significant impact on performance. And the optimal noise probability varies between different problem instances. Plots B.4(a) and B.4(b) do not show large differences between strict GreedyStep$_s$ and soft GreedyStep$_\ell$. However, soft outperforms strict on the m-of-n-3-7-10 dataset, and in general seems to be at least as good. As seen in Plot B.4(d), restart has little effect on most problems, and a slight negative effect on a few. This is a small surprise, given previous research highlighting the benefit of restart [12, 13], and an area for future research.

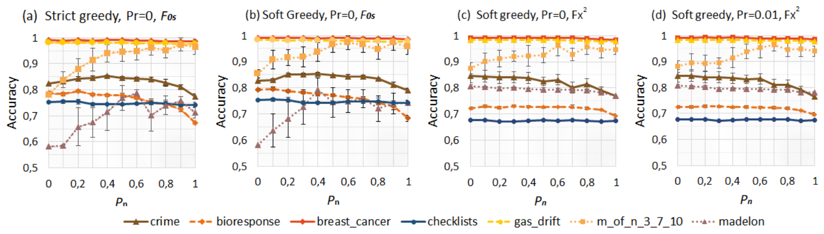


FIGURE B.4. Accuracy (on y-axis) versus P_n (on x-axis) for four configurations of SLS4FEVE (in plots (a), (b), (c), and (d)) as applied to six datasets. Each datapoint in a plot is the average over 10 runs, with the error bars showing the 95% confidence interval.

Subjectively, we deem these to be the most promising configurations of SLS4FS: soft Greedy, $P_r = 0$, $P_n \in [0.1, 0.5]$, and with either $F = F_{0s}$ or $F = F_{\chi^2}$ as filter depending on the selected dataset. These results inform our experiments with four different SLS4FS configurations in Section B.4.4.

Algorithm	Configuration
RFE	N/A
ForwardSelection (F_{0s})	N/A
BackwardSelection (F_{1s})	N/A
AdaptiveNoise	$\phi = 0.2$, $\theta = 1/6$
AdaptiveSLS	$P_n = 0.0$, $P_r = 0.0$, $\alpha_n, \alpha_r = 0.32$
SoftSLS	$P_n = 0.5$, $P_r = 1/n$, $\alpha_n, \alpha_r = 0.0$
SLS4FS (F_{χ^2})	$P_n = 0.5$, $P_r = 0.0$, $\alpha_n, \alpha_r = 0.0$, F_{χ^2}
SLS4FS ($P_r = 0.1$)	$P_n = 0.5$, $P_r = 0.1$, $\alpha_n, \alpha_r = 0.0$, F_{χ^2}
SLS4FS (F_{0s})	$P_n = 0.5$, $P_r = 0.0$, $\alpha_n, \alpha_r = 0.0$, F_{0s}
SLS4FS (adaptive)	$P_n = 0.0$, $P_r = 0.0$, $\alpha_n, \alpha_r = 0.32$, F_{χ^2}

Table B.3. Algorithms and configurations used in Section B.4.4. RFE [35], AdaptiveNoise [34], AdaptiveSLS [7], and SoftSLS [7] are from the literature, the other algorithms are well-known or described in this paper.

B.4.4 Experiment 3: Comparing SLS4FS to Other Algorithms

Goal. How does SLS4FS compare to other FS wrappers using local search?

Method and Data. We test four configurations of SLS4FS and six other algorithms, recording the accuracy of the best feature subset for each algorithm (see Table B.3). The algorithms are tested on problem instances from Table B.1 and evaluations are performed similar to in Section B.4.3. However, a time limit of $\tau = 100$ sec is used in every problem instance, Here, $N_r = \lceil n/10 \rceil$ and GreedyStep$_\ell$ are used for SLS4FS while $\alpha_n = \alpha_r = 0.32$ are used for AdaptiveSLS and SLS4FS (adaptive).[10]

Results and Discussion. The results are summarised in Table B.4.[11] The top three performers, ranked by mean accuracy (in the right-most column), are all variations of SLS4FS. SLS4FS is quite robust across all problems compared to existing algorithms. For example, ForwardSelection achieves the highest accuracy for three problems but is far behind for other problems, leading it to be ranked as one of the last overall.

For the checklists dataset, there were differences in the number of features that were found by the best performing FS configurations. On average, ForwardSelection selected 8 features while AdaptiveSLS and SoftSLS selected 7 features. SLS4FS (F_{0s}) selected 47 features with similar performance in terms of accuracy. The same pattern was also observed for the other large datasets when comparing SLS4FS to ForwardSelection and the existing SLS algorithms. Generally, by using a randomized neighborhood, SLS4FS is able to take more steps per time unit and explore a larger fraction of the search space which yields more features with approximatly the same computational cost.

[10]The hyperparameters α_n and α_r were optimized empirically in pilot studies. In the pilot studies both synthetic and real-world problems were used. In the results reported here, α_n and α_r are kept constant.

[11]The main conclusion, but not the dataset-specific results, of this table has been presented earlier [20].

	breast cancer	m-of-n-3-7-10	madelon	bioresponse	checklists	gas-drift	crime	Mean	Rank
RFE	0.978 ± 0.000	**1.000** ± **0.000**	0.738 ± 0.000	0.636 ± 0.000	NA	0.976 ± 0.000	0.808 ± 0.000	0.734	9
ForwardSelection	0.991 ± 0.000	0.780 ± 0.000	0.582 ± 0.000	**0.797** ± **0.000**	**0.751** ± **0.000**	**0.982** ± **0.000**	0.818 ± 0.000	0.803	8
BackwardSelection	**0.996** ± **0.000**	**1.000** ± **0.000**	0.726 ± 0.000	0.628 ± 0.000	0.675 ± 0.000	0.970 ± 0.000	0.784 ± 0.000	0.826	6
AdaptiveNoise	**0.996** ± **0.000**	**1.000** ± **0.000**	0.731 ± 0.029	0.620 ± 0.017	0.667 ± 0.000	0.973 ± 0.002	**0.854** ± **0.004**	0.834	5
AdaptiveSLS	**0.996** ± **0.000**	**1.000** ± **0.000**	0.732 ± 0.024	0.623 ± 0.025	**0.751** ± **0.000**	0.974 ± 0.003	0.830 ± 0.007	0.844	4
SoftSLS	**0.996** ± **0.000**	**1.000** ± **0.000**	0.796 ± 0.005	0.626 ± 0.023	0.749 ± 0.001	0.974 ± 0.003	0.848 ± 0.007	0.856	3
SLS4FS (F_{χ^2})	**0.996** ± **0.000**	**1.000** ± **0.000**	0.803 ± 0.006	0.734 ± 0.008	0.678 ± 0.002	**0.982** ± **0.002**	0.845 ± 0.002	0.862	2
SLS4FS ($P_r = 0.1$)	**0.996** ± **0.000**	**1.000** ± **0.000**	0.795 ± 0.006	0.717 ± 0.008	0.673 ± 0.000	**0.982** ± **0.001**	0.829 ± 0.003	0.856	3
SLS4FS (F_{0s})	**0.996** ± **0.000**	**1.000** ± **0.000**	**0.804** ± **0.012**	0.774 ± 0.005	0.749 ± 0.003	0.980 ± 0.002	0.843 ± 0.004	**0.878**	**1**
SLS4FS (adaptive)	**0.996** ± **0.000**	0.790 ± 0.019	0.796 ± 0.008	0.724 ± 0.007	0.677 ± 0.001	**0.982** ± **0.001**	0.684 ± 0.009	0.807	7

TABLE B.4. Mean accuracy and standard deviation for 4 versions of SLS4FS and 6 other algorithms, applied to 7 datasets. The two right-most columns show Mean accuracy (higher is better) and Rank (lower is better) for all 10 algorithms across the 7 datasets. Overall, SLS4FS (F_{0s}) is the best algorithm among the 10 for these datasets.

B.5 CONCLUSION AND FUTURE WORK

In this paper, we adapt and apply stochastic local search (SLS) to the problem of feature selection. We study an SLS algorithm SLS4FS for feature selection; it is a hybrid approach that integrates the well-known filter and wrapper approaches. Relative to the most closely related research [7], SLS4FS adds and integrates three heuristics: soft greedy search, filters, and a randomized neighborhood relation [20]. Experimentally, motivated by constraining computational resources, we study different FS filter algorithms with SLS4FS and three ML classifiers: Decision Tree, Naive Bayes, and Support Vector Machine. SLS4FS produces competitive results on several different datasets in experiments, at modest computational cost, thus reflecting our goals of maximizing model accuracy while controlling training time.

These are a few areas for future work: First, it would be useful to more systematically vary SLS4FS hyperparameters including P_r and P_n. Second, we plan to investigate other classifiers and even larger datasets with more features. Third, it would be interesting to automatically optimize the hyperparameters of SLS4FS [7, 24, 36], considering the varying computational costs of search steps for different computers and datasets. Fourth, it could be interesting to compare SLS4FS to other methods, such as tabu search, simulated annealing, and evolutionary algorithms. Fifth, we plan to further study SLS4FS from a theoretical perspective, using Markov chain hitting time analysis.

B.6 References

[1] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[2] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *JMLR*, 3:1157–1182, 2003.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. *Feature Selection for High-Dimensional Data*. Springer, 2015.

[4] Andrea Bommert, Xudong Sun, Bernd Bischl, Jörg Rahnenführer, and Michel Lang. Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, 143:1–19, 2019. doi: 10.1016/j.csda.2019.106839.

[5] Md Monirul Kabir, Md Shahjahan, and Kazuyuki Murase. A new local search based hybrid genetic algorithm for feature selection. *Neurocomputing*, 74(17): 2914–2928, 2011.

[6] S. Y. Kim and E. Xing. Feature selection via block-regularized regression. In *Proc. UAI*, pages 325–332, 2008.

[7] O. J. Mengshoel, Y. Ahres, and T. Yu. Markov chain analysis of noise and restart in stochastic local search. In *Proc. IJCAI*, pages 639–646, 2016. URL http://www.ijcai.org/Abstract/16/097.

[8] A. J. Knobbe and E. K. Y. Ho. Maximally informative k-itemsets and their efficient discovery. In *Proc. KDD*, pages 237–244, 2006.

[9] H. H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann, San Francisco, 2005.

[10] Emre Yolcu and Barnabás Póczos. Learning local search heuristics for boolean satisfiability. In *Proc. NeurIPS*, pages 7990–8001, 2019.

[11] J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *JAIR*, 21:101–133, 2004.

[12] O. J. Mengshoel. Understanding the role of noise in stochastic local search: Analysis and experiments. *Artificial Intelligence*, 172(8-9):955–990, 2008.

[13] O. J. Mengshoel, D. C. Wilkins, and D. Roth. Initialization and restart in stochastic local search: Computing a most probable explanation in Bayesian networks. *IEEE Transactions on Knowledge and Data Engineering*, 23(2):235–247, 2011.

[14] D. K. Pal and O. J. Mengshoel. Stochastic CoSaMP: Randomizing greedy pursuit for sparse signal recovery. In *ECML-PKDD*, pages 761–776, 2016.

[15] C. White, S. Nolen, and Y. Savani. Exploring the loss landscape in neural architecture search. In *Proc. UAI*. AUAI Press, 2021.

[16] Raphael Schumann, Lili Mou, Yao Lu, Olga Vechtomova, and Katja Markert. Discrete optimization for unsupervised sentence summarization with word-level extraction. In *Proc. ACL*, pages 5032–5042, 2020.

[17] C. Bian, C. Feng, C. Qian, and Y. Yu. An efficient evolutionary algorithm for subset selection with general cost constraints. In *Proc. AAAI*, pages 3267–3274, 2020.

[18] Ke Shang, Hisao Ishibuchi, and Weiyu Chen. Greedy approximated hypervolume subset selection for many-objective optimization. In *Proc. GECCO*, pages 448–456, 2021.

[19] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, 34(3):483–519, 2013.

[20] Ole Jakob Mengshoel, Tong Yu, Jon Riege, and Eirik Flogard. Stochastic local search for efficient hybrid feature selection. In *Proc. GECCO*, pages 133–134, 2021.

[21] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. ICML*, pages 1015—-1022, 2010.

[22] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Proc. NIPS*, pages 2951–2959, 2012.

[23] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proc. ICML*, pages 1436–1445, July 2018.

[24] T. Yu, B. Kveton, and O. J. Mengshoel. Thompson sampling for optimizing stochastic local search. In *Proc. ECML-PKDD*, pages 493–510, 2017.

[25] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., 2001.

[26] C. A. C: Coello, G. B. Lamont, and D. A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag, 2006.

[27] O. J. Mengshoel, D. Roth, and D. C. Wilkins. Portfolios in stochastic local search: Efficiently computing most probable explanations in Bayesian networks. *Journal of Automated Reasoning*, 46(2):103–160, 2011.

[28] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[29] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[30] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the NIPS 2003 feature selection challenge. In *Proc. NIPS*, pages 545–552, 2004.

[31] Boehringer Ingelheim. Predicting a biological response. Mar 2012. URL https://www.kaggle.com/c/bioresponse/data.

[32] E. L. Flogard, O. J. Mengshoel, and K. Bach. Bayesian feature construction for case-based reasoning: Generating good checklists. In *Proc. ICCBR*, pages 94–109. Springer, 2021.

[33] A. Jović, K. Brkić, and N. Bogunović. A review of feature selection methods with applications. In *Proc. MIPRO*, pages 1200–1205, 2015.

[34] H. H. Hoos. An adaptive noise mechanism for WalkSAT. In *Proc. AAAI*, pages 655–660, 2002.

[35] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002. doi: 10.1023/A:1012487302797.

[36] S. Ermon, C. P. Gomes, A. Sabharwal, and B. Selman. Designing fast absorbing Markov chains. In *Proc. AAAI*, pages 849–855, 2014.

Paper C: Creating Dynamic Checklists via Bayesian Case-Based Reasoning: Towards Decent Working Conditions for All

Published in: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI),* 2022.
Acceptance rate: 14.9% (2022)
Norwegian Scientific Index Level: *2 (highest)*

Notes. This paper is published in the Proceedings of the International Joint Conference on Artificial Intelligence, which is one of the top publication channels in the field of artificial intelligence (AI). The channel receives contributions from the best AI researchers and is highly selective, with an acceptance rate of just 14.9% for 2022. The publication channel is ranked at the highest level (2) in the Norwegian Scientific Index, placing it above most other conferences and journals in the field of AI. Level 2 is generally reserved for the most leading publication channels, publishing the most significant research within their respective fields of science. The conference proceedings has its own ISSN number and is therefore regarded as a journal in the Norwegian publication channel register, meaning that the paper is worth 3 publication points. The paper has been reformatted to ensure readability and consistency with the other papers in the thesis.

Contribution Statement. My contribution to the paper includes the ideas presented in the paper, writing the first draft and carrying out the experiments in the paper. The co-authors of the paper are my supervisors, and they helped with the problem formulations and refining the paper.

Creating Dynamic Checklists via Bayesian Case-Based Reasoning: Towards Decent Working Conditions for All

Eirik Lund Flogard[1,2], **Ole Jakob Mengshoel**[2] and **Kerstin Bach**[2]
[1]Norwegian Labour Inspection Authority
[2]Norwegian University of Science and Technology
eirik.flogard@arbeidstilsynet.no, {ole.j.mengshoel, kerstin.bach}@ntnu.no

Abstract

Every year there are 1.9 million deaths worldwide attributed to occupational health and safety risk factors. To address poor working conditions and fulfill UN's SDG 8,"protect labour rights and promote safe working environments for all workers", governmental agencies conduct labour inspections, using checklists to survey individual organisations for working environment violations. Recent research highlights the benefits of using machine learning for creating checklists. However, the current methods only create static checklists and do not adapt them to new information that surfaces during use. In contrast, we propose a new method called Context-aware Bayesian Case-Based Reasoning (CBCBR) that creates dynamic checklists. These checklists are continuously adapted as the inspections progress, based on how they are answered. Our evaluations show that CBCBR's dynamic checklists outperform static checklists created via the current state-of-the-art methods, increasing the expected number of working environment violations found in the labour inspections.

C.1 Introduction

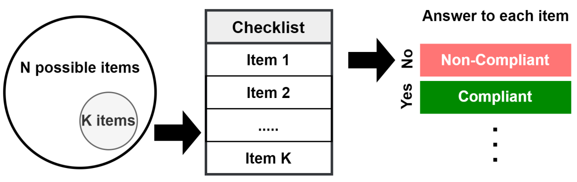


Figure C.1. A conceptual view of a checklist for labour inspections. For a given inspection, a checklist ideally contains a subset of the K items most likely to be non-compliant, out of N possible items.

Checklists are extensively used in high-stakes decision making, such as in surgery or food inspections [1, 2]. They are also used by government agencies in labour inspections, to survey individual organisations for non-compliance to working environment regulations [3, 4]. Such inspections are important to globally achieve UN's SDG 8, "protect labour rights and promote safe working environments for all workers", specifically SDG 8.8. Despite the inspection efforts, there are still 1.9 million registered deaths world-wide each year that are attributed to occupational health and safety risk factors [5]. Our goal in this paper is to attack this problem by introducing dynamic checklists created via machine learning.

A conceptual view of a labour inspection checklist is shown in Figure C.1. Here, a checklist consists of a subset of K of N items where each item corresponds to a specific regulation. Each item has a yes or no answer which indicates the inspected organisation's compliance to the corresponding regulation. Any non-compliant item found at an inspection is cited individually in a report which is sent to the inspected organisation, with an order to rectify the violations.

Each inspected organisation may be subjected to several hundred different regulations, which vary according to its size, location and industry. Prioritizing the correct regulations for inspections, in terms of risks to workers' health and safety, is very difficult. Assessing compliance to regulations is also time-consuming, so a checklist should ideally contain a small subset K of N possible items that are most likely to be found non-compliant at the inspection. Creating or using such checklists is difficult, partly because they are situation dependent [2, 6].

In order to create optimized checklists efficiently, Flogard et al. [7] propose to use machine learning (ML) to construct checklists and a new ML problem called the Checklist construction problem (CCP). They introduce a method called BCBR, which constructs checklists for labour inspections. BCBR uses Bayesian inference (BI) to construct features for cases used in case-based reasoning (CBR) [8]. The input to BCBR is an organisation targeted for a labour inspection. BCBR then selects cases that contain a set of the K out of N possible unique items that are most likely to be found non-compliant at the given organisation. The output is an optimized checklist that consists of the selected K items.

Motivation. A shortcoming with BCBR is that it only creates static checklists, which can be inaccurate in many real-world situations where the context changes over time [9]. For instance, new information obtained during an inspection could suggest that inquiries should be made into regulations that are not represented by the K items currently on the checklist on Figure C.1. A possible solution for this is to dynamically adapt the checklists during use. Figure C.2 shows how a dynamic checklist can be adapted to a situation, inferred from how a user answers the checklist. This idea has been proposed and studied in clinical surgery trials [10], where the current approaches for adapting checklists are based on process [11] or rule-based models [12]. However, these models need to be built and maintained manually by domain experts, which is infeasible for creating the ideal high-detailed models needed for many complex tasks. Kulp et al. [10] also discuss this limitation and highlight the need for a technical solution that can adapt checklists without relying on manually created models.

Scientific Contributions. There are two main contributions in this work. The first contribution is to establish ML as a means to create dynamic checklists. As far as we know, this is a difficult problem where ML has not been used before. A challenge here is to develop a method that is accurate and fast enough to make real-time dynamic adaptations of checklists.

The second contribution is a new method called Context-aware Bayesian Case-Based Reasoning (CBCBR), which is an extension of BCBR. The novelty, compared to BCBR, is a context-aware naive Bayesian inference model that enables dynamic adaptations to the constructed checklists during use. This means that CBCBR both creates new checklists and adapts them. The adaptations are done as recommendations of new items for the checklists. The recommendations are based on answers of the checklists, which can be considered a part of the temporal context of the situations where the checklists are used [13]. CBCBR is also a fully transparent, online model that enables

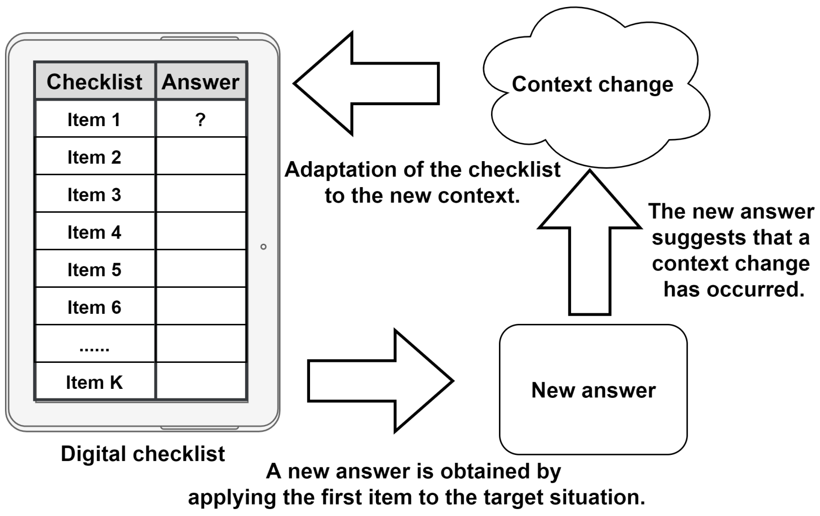

Checklist
Answer
Item 1
?
Item 2
Item 3
Item 4
Item 5
Item 6
......
Item K
Digital checklist
Context change
Adaptation of the checklist to the new context.
The new answer suggests that a context change has occurred.
New answer
A new answer is obtained by applying the first item to the target situation.

FIGURE C.2. A dynamic checklist is a digital checklist that can adapt to changes in environment or context during use. Such adaptations to the checklists can increase their relevance and efficiency in aiding users for their intended tasks.

real-time creation and adaptation of checklists, which is crucial for their usability in real-world situations. Online learning models are known for their effectiveness in dynamic or non-stationary situations [14, 15], such as labour inspections.

Social Impact. Our work focuses on improving checklists used for labour inspections, which will have a direct impact on the promotion of decent work (see SDG 8). We show that CBCBR's dynamic checklists can improve task completion and increase the number of violations (non-compliance) found during inspections, which in turn will increase levels of compliance with working environment and labour rights and also reduce injuries (SDG indicator 8.8.1 and 8.8.2). More effective checklists could also reduce social dumping (SDG 8.5), human trafficking and forced labour (SDG 8.7).

C.2 RELATED WORK

Checklist Construction. Until recently, checklists have been created manually by domain experts via approaches like the Delphi method [16, 17]. However, creating checklists manually is difficult and time consuming [18]. Instead of relying on human experts, Flogard et al. [7] propose BCBR for constructing static checklists. In experiments, they show that labour inspection checklists constructed by BCBR outperform human domain experts and other ML methods. Zhang et al. [19] propose an ML method for constructing checklists for medical diagnosis, assuming that a checklist is a binary M-of-N decision problem. They use an integer program to find an optimal checklist of N items that most accurately predict a diagnose, given that at least M items are checked. However, their method only creates static checklists and is also not usable for creating labour inspection checklists, as they are not binary M-of-N

decision problems. An approach for generating checklist items for construction site inspections is proposed by Cai et al. [20]. The approach is based on Natural Language Processing and generates new items from regulation document texts. Checklists are generated via SQL queries, by matching generated items with keywords specified by a user. The checklists are described as dynamic, but only in the sense that they are presented digitally, enabling users to manually customize them.

Context-Aware Recommender Systems. Research shows that dynamically elevating items on checklists can be effective for adapting them to context changes [12]. CBCBR's recommendations can also be seen as a way to elevate context-relevant items and is inspired by context-aware recommender systems (CARS) and contextual modelling [21].

Case-Based Reasoning. In recommender systems, CBR is used to address explainability and the long tail problem, which are known issues in collaborative filtering [22, 23]. Reasoning in CBR (and CBCBR) is based on retrieving the best past cases to solve a given problem [8]. To improve reasoning, Bayesian methods are used to infer missing case features or information in CBR systems [24, 25]. For the same reason, CBCBR's cases are augmented with probability estimates to improve case retrieval. CBCBR also learns from answered checklists items, by using them for checklist adaptations or retaining them to create future checklists.

Summary. To our knowledge, there is no previous work where ML is used to dynamically adapt checklists. In our work, we show that checklists can be significantly improved by doing so, using answers to past items in a checklist to adapt future items in the same checklist. The approach is also generic and can likely be used in many other domains, since most checklists have some kind of answers recorded in them. Thus, our work can potentially be seen as a starting point for future ML research into dynamic context-aware checklists.

C.3 Definitions

Data Set and Cases. A data set $\mathcal{D}$ for variables $\mathbf{Z} = \{E, X, L\}$ is a tuple $(\mathbf{d}_1, ..., \mathbf{d}_N)$ where a case $\mathbf{d}_j \in \mathcal{D}$ is an instantiation of $\mathbf{Z}$ [26]. A case can be defined as a tuple $\mathbf{d} = (e, x, l)$ where e denotes a single item of a checklist, x is the target organisation for the item and $l \in \{0, 1\}$ denotes the binary answer to the item. A case in the data set can be viewed as a past experience where an item e has been applied to x to obtain the answer l. Any e, x and l are instantiations of the variables E, X and L, respectively.

Organisation. Every case in the data set contains a target organisation description x, that consists of multiple sub-features. These sub-features are all categorical and describe the organisation's location and industry [7]. For brevity we treat x and X as categorical in our work.

Item. Each case in the data set contains an answerable item e, used to survey the organisation x. There are N unique items in the data set, so an item e is a categorical value that may appear in multiple cases. Thus, E is also categorical.

Checklist. The item in each case of the data set belongs to a checklist $\mathbf{y}$, which is a solution applied to the organisation x. A checklist consists of a set of items such that $\mathbf{y} = (e_1 \in \mathbf{d}_1, e_2 \in \mathbf{d}_2, ..., e_n \in \mathbf{d}_n)$. An item can only occur once per checklist, such that $e_i \neq e_j$ for every $e_i \wedge e_j \in \mathbf{y}$. Thus, $\mathbf{y}$ consists of items from multiple cases.

Answer. The label $l \in \{0, 1\}$ of each case $\mathbf{d}_j \in \mathcal{D}$ is the recorded answer from applying the item e to the organisation x. A positive answer (non-compliance) is observed if $l = 1$.

The Checklist Construction Problem. Given a candidate target organisation x^{cnd}, a model $\mathcal{M}$ needs to select K out of N unique items $(e_1, e_2, ..., e_K)$ for an initial candidate checklist $\mathbf{y}^{cnd}$. Each item $e_i \in \mathbf{y}^{cnd}$ needs to be selected so that it maximizes the probability for observing the answer $l_i = 1$ when applied to the targeted x^{cnd}.

Dynamic Adaptation of Checklists. Let $(l_1, ..., l_i)$ be the observed answers to the items $(e_1, ..., e_i) \in \mathbf{y}^{cnd}$ where $i \leq K$. Given the target organisation x^{cnd} and the answered items, adapt the checklist so that the posterior probability for observing a positive answer to any unanswered items on the changed checklist is maximized. The adaptation could be done by adding or removing items to the checklist in many ways. In this work adaptation is done by recommending any additional M of N items $\hat{e} \notin y^{cnd}$ that have higher posterior probability for observing $\hat{l} = 1$ (given $(l_1, ..., l_i)$), compared to any existing items on y^{cnd}. The M recommendations are appended to the existing K items of $\mathbf{y}^{cnd}$.

C.4 CBCBR Framework

The purpose of CBCBR is to create a dynamic checklist for any given x^{cnd}, by retrieving past cases with items used in similar organisations, and with high estimated probabilities for non-compliance. Answering the checklist may change the estimates, triggering dynamic updates to the checklist.

An overview of CBCBR is shown in Figure C.3 and consists of the following steps: (1) a naive Bayesian inference (NBI) model generates probability estimates (θ^{be}) for answers based on empirical distributions of the data set $\mathcal{D}$. (2) A case base $\mathcal{CB}$ of augmented CBR cases $\mathbf{c}_j$ is created by adding θ^{be} as feature to the instances $\mathbf{d}_j \in \mathcal{D}$. (3) Similarity based retrieval is used to retrieve K cases from $\mathcal{CB}$, given a query $\mathbf{q}$ that contains the target organisation x^{cnd} and a fixed target value for θ^{be} in the cases. The retrieved cases contain the items for the initial candidate checklist $\mathbf{y}^{cnd}$, as illustrated by the white arrows on Figure C.3. (4) The novel recommendation part for adapting $\mathbf{y}^{cnd}$ is shown by the blue arrows in Figure C.3. First, an item on $\mathbf{y}^{cnd}$ is answered, which prompts CBCBR to refresh the $\mathcal{CB}$ with updated posterior θ^{be} estimates (via NBI). CBCBR then retrieves any new cases from $\mathcal{CB}$ with sufficiently increased θ^{be}, containing M recommended items which are appended to $\mathbf{y}^{cnd}$.

Further details for how θ^{be} is calculated by the NBI model in steps 1 and 4 are covered in Section C.4.1 and C.4.2, respectively. The other details for step 2-4 are found in Section C.4.3.

C.4.1 *Naive Bayesian Inference for Checklist Construction*

NBI is based on using empirical distributions of the data set $\mathcal{D}$ to estimate the probability for the event $L = 1|x, e$, expressed as the mean of a Beta distribution [26]:

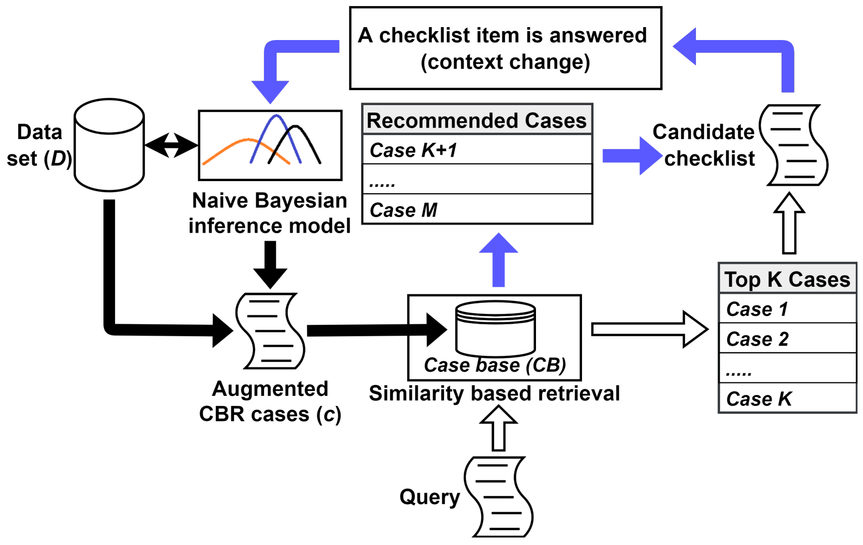


FIGURE C.3. An overview of CBCBR. The black arrows show how the case base is created or updated. The white arrows show the creation of a candidate checklist. The checklist is dynamically updated via the blue (and black) arrows, starting from the candidate checklist.

$$\theta^{be}(L = 1|x, e) = \frac{\mathcal{D}\#(L = 1 \wedge X = x \wedge E = e) + \psi_{L=1|x,e}}{\sum_{l=0}^{1} \mathcal{D}\#(L = l \wedge X = x \wedge E = e) + \psi_{L=l|x,e}} \tag{C.1}$$

where $L = l, X = x$ and $E = e$ denote the event where the outcomes l, x and e are observed. Both $(\mathcal{D}\#(L = 1 \wedge X = x \wedge E = e)$ and $(\mathcal{D}\#(L = 0 \wedge X = x \wedge E = e)$ are parameters for the Beta distribution, and $\psi_{L=l|x,e}$ denotes the prior parameters. Equation C.1 is used to calculate the probability estimates needed for selecting the optimal items when constructing the initial checklist y^{cnd}.

C.4.2 *Naive Bayesian Inference for Item Recommendations*

After answering an item e_i on the checklist y^{cnd}, the obtained answer l_i holds evidence that can be used to update the posterior belief in any unobserved answer $\hat{l}$ for a candidate item $\hat{e}$.[1] This assumption can be used to update the probability estimates from Equation C.1, by estimating new additional Beta distribution parameters with the following equation:

[1]In this setting, $\hat{e}$ is a potential candidate for recommendation.

$$p(\hat{l}, x, \hat{e}, e_i, l_i) = \mathcal{D}\#(L = \hat{l} \wedge X = x \wedge E \in \{\hat{e} : (\hat{e}, e_i) \in \mathbf{y} \wedge (x, e_i, l_i) \in \mathcal{D}\}). \quad \text{(C.2)}$$

The parameters are made by counting cases in $\mathcal{D}$. Each case in $\mathcal{D}$ is counted if it contains the given x, $\hat{l} \in \{0, 1\}$, $\hat{e}$ and if there exist at least one other case $\mathbf{d} = (x, e_i, l_i)$ and a past checklist $\mathbf{y}$ in $\mathcal{D}$ such that both e_i and $\hat{e}$ exists in that checklist. The posterior estimates can now be updated by inserting the parameters above into the Beta distribution from Equation C.1:

$$\theta^{be}(L = 1|x, \hat{e}, \mathbf{y}^{cnd}) = \frac{\beta_{L=1|x,\hat{e}} + \sum_{(l_i, e_i \in \mathbf{y}^{cnd})} p(1, x, \hat{e}, e_i, l_i)}{\sum_{\hat{l}=0}^{1} \beta_{L=\hat{l}|x,\hat{e}} + \sum_{(l_i, e_i \in \mathbf{y}^{cnd})} p(\hat{l}, x, \hat{e}, e_i, l_i)} \quad \text{(C.3)}$$

where $\beta_{L=\hat{l}|x,\hat{e}} = \mathcal{D}\#(L = \hat{l} \wedge X = x \wedge E = \hat{e}) + \psi_{L=\hat{l}|x,\hat{e}}$. The θ^{be} estimate in Equation C.3 is calculated by summing the parameters for every $e_i \in \mathbf{y}^{cnd}$ that has an observed answer l_i. It should be noted that the equation above "naively" assumes that all applied items e_i are mutually independent of each other given x. This is done to decrease the amount of θ^{be} estimates based on low or zero case counts [26]. Equation C.3 is an important technical contribution of this paper, since it enables online adaptations of checklists.

C.4.3 Case Base and Retrieval of Checklist Items

This section defines the details for the augmented CBR cases, case base and similarity based retrieval from Figure C.3.

Creating Augmented CBR Cases and Case Base. Algorithm 1 shows the creation of a case base $\mathcal{CB}$ with augmented cases $\mathbf{c}$. The κ feature is included to adjust for the case counts of the probability estimates when retrieving cases [7]. The algorithm iterates through each case $\mathbf{d}_j \in \mathcal{D}$ and creates κ and θ^{be} estimates for the case via Equation C.1. Both θ^{be} and κ are conditioned on the case features x and e_j and added as features to each $\mathbf{d}_j$, to create cases $\mathbf{c}$ for $\mathcal{CB}$.

Case Retrieval for the Candidate Checklist. To retrieve items e_i for the candidate checklist $\mathbf{y}^{cnd}$, a query case $\mathbf{q}$ and a similarity function are used. The query consists of the target organisation x^{cnd} and the desired values for both the probability estimates and the case count features. The similarity function assigns a score $Sim(\cdot, \cdot) \in [0, 1]$ to every pair $(\mathbf{q}, \mathbf{c}_j \in \mathcal{CB})$. The function is the same linear weight similarity function used by Flogard et al. [7]. The function is applied to every $\mathbf{c}_j \in \mathcal{CB}$ and a set of unique items $(e_1, ..., e_K)$ is then retrieved from the K cases with the highest similarity score, to create an initial checklist $\mathbf{y}^{cnd}$ with K items.

Case Base Update and Retrieval for Recommendations. Algorithm 1 also shows how the case base is refreshed after answering items on the checklist. This procedure is done after an item $e_i \in \mathbf{y}^{cnd}$ has been applied to x^{cnd}. First the existing $\mathcal{CB}$ is cleared for cases. Then updated probability estimates are calculated for each case $\mathbf{d}_j \in \mathcal{D}$

Algorithm 1 Creating or updating $\mathcal{CB}$ with cases $\mathbf{c}$ containing new θ^{be} estimates.

```
Input   : $\mathcal{D}$, $y^{cnd}$; //$y^{cnd}$ is only included when updating $\mathcal{CB}$ with new posterior $\theta^{be}$ estimates after answering items.
Output : $\mathcal{CB} \leftarrow ()$; //Initialize $\mathcal{CB}$.
foreach $\mathbf{d}_j \in \mathcal{D}$ do
    //Both $x_j$ and $e_j$ are found in the current case $\mathbf{d}_j$.
    if $y^{cnd}$ is empty then
        $\theta \leftarrow \theta^{be}(L = 1|x_j, e_j)$; //Eq. C.1
    else
        $\theta \leftarrow \theta^{be}(L = 1|x_j, e_j, y^{cnd})$; //Eq. C.3
    $\kappa \leftarrow \mathcal{D}\#(L = 1 \wedge X = x_j \wedge E = e_j)$;
    $\mathbf{c} \leftarrow Join(\mathbf{d}_j, \theta, \kappa)$; //merge $\mathbf{d}_j$, $\theta$, $\kappa$ to a single case.
    $\mathcal{CB} \leftarrow Insert(\mathbf{c})$; //Add the new case $\mathbf{c}$ to $\mathcal{CB}$.
return $\mathcal{CB}$; //The $\mathcal{CB}$ is now ready for any case retrievals.
```

via Eq. C.3, conditioned on x_j, e_j and every $e_i \in y^{cnd}$ with an obtained l_i. The newly calculated θ and κ features are concatenated to each $\mathbf{d}_j$ to create the updated cases $\mathbf{c}$ for $\mathcal{CB}$.

The updated cases are then retrieved for recommendation. First, the function $Sim(\mathbf{q}, \mathbf{c}_j)$ is again applied to $\mathbf{q}$ and every $\mathbf{c}_j \in \mathcal{CB}$. A filter is then applied to remove all cases with similarity scores less than any of the K cases that were selected for the initial y^{cnd}. Duplicated items are also removed so that M cases with unique items remain. The size of M is usually very small (between 0 and 4) and is not known or fixed, but simply depends on how many eligible cases that remain after applying the filter. If there are $M > 0$ remaining cases, the items from these cases are recommended for the checklist.

C.5 Demonstration and Assessment of CBCBR

Data Set. We use the Checklist data set introduced by Flogard et al. [7], which has 1 111 502 entries $\mathbf{d}_j$. The entries constitute 63 634 inspections conducted with 369 different unique checklists, which in turn contain $N = 1\,947$ unique items in total. The checklists cover a wide range of industries and the items on each checklist typically have an inspection topic that specifically relates to a few intended target industries. Mengshoel et al. [27] also use a similar data set.

Query. We use the configuration from Section C.6.1. A recommendation is done after answering each item. The query is $\mathbf{q} = (x^{cnd}, \theta, \kappa)$ and contains the desired values of the retrieved cases, where θ is set to 100% and κ to 70. The target organisation for the inspection x^{cnd} is a hotel in Oslo.

Checklist	
Does the employer ensure that the employees work within the framework in Chapter 10 of the Working Environment Act?	[✓ Yes] / No
Has the employer ensured that the employment agreements are in line with the minimum requirements set out in the Working Environment Act §14-6?	Yes / [✓ No]
Does the company have routines for how violence, threats and other unfortunate burdens as a result of contact with others are to be prevented, reported, handled and followed up?	Yes / No
Has the employer implemented the necessary measures and/or prepared a plan describing measures to remove or reduce hazards and problems at work?	Yes / No
Has the employer mapped the dangers and problems the employees may be exposed to in the company and on this basis assessed the risk of injury to or danger to the employees?	Yes / No
Added Items	
Does the company pay at least a 40% supplement to the salary for overtime work?	Yes / No
Does the employer have control over the working hours of the employees within the framework of the Working Environment Act, Chapter 10?	Yes / No

TABLE C.1. The state of a dynamic checklist y^{cnd} with $K = 5$ initial items, generated for a hotel inspection in Oslo. Two items on the checklist have been answered (as indicated with blue circles). Two items have also been dynamically added to the checklist (at the bottom), based on the answered items.

Constructed Checklist and Item Recommendations. Table C.1 shows the constructed checklist y^{cnd} for the target hotel x^{cnd} where two of the items have been answered. The answers were randomly selected. The initial size of the checklist is $K = 5$ items, however two items have been added as recommendations based on the answered items. Further items may be added after more answers are given. The added items increases the checklist length, but allow the inspector to focus more on highly relevant risks in the inspected organisation.

Qualitative Assessment. The items in Table C.1 cover a variety of topics such as working hours, overtime payment and violence. These are all common risks for the hotel industry, especially in a large city such as Oslo. The added items shown in the table are also relevant and closely related to the two answered items on the top of the checklist. Thus, the recommendations align well with the findings at the inspection. CBCBR also performed well when we, with domain experts, assessed a variety of other inputs (x^{cnd}) and lengths (K).

C.6 EXPERIMENT

In this section we conduct an experiment to measure the performance of CBCBR against BCBR and other baselines.

C.6.1 Experimental Setup

CBCBR and BCBR Configuration. Both CBCBR and BCBR generate the exact same initial checklists, but BCBR does not perform any additional recommendations or dynamic adaptations. The two methods use the same configuration as Flogard et al. [7]. The NBI models for CBCBR and BCBR use fixed priors $\psi_{L=1|x,e} = 1$ and $\psi_{L=0|x,e} = 5$. For every query $\mathbf{q}$, the target matching value for θ^{be} is set to 100%. The target value for κ is set to 70. The NBI models are created and updated via MSSQL17 queries and stored as Python data frames. The similarity based retrieval is implemented via MyCBR [28] for both methods. For performance reasons, CBCBR recommendations are made each time 5 items on the checklist have been answered.

Generating Checklist Answers for Predictive CBCBR Recommendations. To generate predictive recommendations for CBCBR in the experiment, we introduce a simulator. The approach is similar to user-behaviour simulations in recommender systems [29], as the goal of the simulator is to perform a realistic walk-through of an initial checklist to obtain an answer l_i for each item $e_i \in \mathbf{y}^{cnd}$. These answers are then used to generate realistic item recommendations in the experiment below. This is done as follows:

1. For a given target organisation x^{cnd}, CBCBR (model $\mathcal{M}$) first constructs an initial checklist $\mathbf{y}^{cnd}$ with items e_i. Each e_i is retrieved from a case $\mathbf{c}_i \in \mathcal{CB}$ and has an unobserved answer l_i.
2. For each $e_i \in \mathbf{y}^{cnd}$, a value for l_i needs to be generated by drawing from the set $\{0, 1\}$.
3. To do so, the parameters $\mathbf{b}_i$ of a Bernoulli distribution $\mathcal{B}$ for l_i are estimated empirically from $\mathcal{CB}$ using the following expression: $\mathbf{b}_i = \frac{\mathcal{CB}\#(L=1 \wedge X=x^{cnd} \wedge E=e_i)}{\mathcal{CB}\#(X=x^{cnd} \wedge E=e_i)}$.
4. If e_i is an item that has been recommended earlier in the simulation, the recommendation context is taken into account when calculating $\mathbf{b}_i$. This is done by applying Eq. C.3 to $\mathcal{CB}$ (instead of $\mathcal{D}$): $\mathbf{b}_i = \theta^{be}(L = 1|x^{cnd}, e_i, \mathbf{y}^{cnd})$.
5. The value $l_i \in \{0, 1\}$ is now drawn from a Bernoulli distribution, given by $\mathcal{B}(\mathbf{b}_i, 1 - \mathbf{b}_i)$. If 5 values have been generated since the last recommendation, then proceed to the next step. Otherwise, go to Step 2.
6. After 5 l_i-values are generated, CBCBR is updated with the new values in order to perform a recommendation. Any recommended items are then automatically appended to $\mathbf{y}^{cnd}$. After the recommendation, go to Step 2 if there are any unanswered items left in $\mathbf{y}^{cnd}$.

After the procedure above has been applied, the extended checklist is ready for evaluation.

Baselines for the Experiment. We use these following baselines to generate static checklists: Multi layer perceptron (NN), Random forest (RF), Naive Bayes inference (NBI), Decision tree (DT) and Logistic regression (LR).

Each baseline ($\mathcal{M}$) above generates a checklist of K items as follows: (1) $\mathcal{M}$ is first trained on a training set $\mathcal{D}_T$ of $\mathcal{D}$, using l from each instance $\mathbf{d} = (e, x, l)$ as the target label. Each $\mathcal{M}$ is trained with the goal of correctly classifying the value of l, given x and

e. (2) Given a validation set $\mathcal{D}_{CB}$ of $\mathcal{D}$ and an input x^{cnd}, $\mathcal{M}$ generates a prediction score ($[0, 1]$) for every N possible items. Predictions are only generated for items (e) with at least one corresponding instance $(e, x, l) \in \mathcal{D}_{CB}$ where $x = x^{cnd}$. (3) The K items with the highest scores are selected for the candidate checklist y^{cnd}.

We also use the original, domain expert created checklists (ECL) from the Checklist data set as baseline.

Baselines Configuration. The NBI-baseline is calculated using Equation C.1 and is implemented via MSSQL17. The other baselines are implemented via Sklearn, using the default configurations for most of them. For the NN and RF baselines we used GridsearchCV (Sklearn) for hyperparameter tuning. The optimal configuration for NN was 20 layers, logistic activation function and L2 penalty of 0.0001. The optimal configuration for RF was 50 estimators, bootstrapping sample size of 50% and minimum sample size of 10 for node splits.

Environment. The experiment is conducted on a fully upgraded Dell Precision 5560 with Intel i9 11950h and 64GB RAM, in a Python environment using Scikit-learn (Sklearn).

C.6.2 Evaluation of CBCBR's Dynamic Checklists

The goal of this experiment is to evaluate the performance of CBCBR against BCBR and other baselines.

Method and Data. The experiment is done on the data set $\mathcal{D}$ from Section C.5. An 8-fold cross-validation is used where $\mathcal{D}$ is partitioned into training folds ($\mathcal{D}_T$) and validation folds ($\mathcal{D}_{C\mathcal{B}}$). $\mathcal{D}_T$ is used to calculate any probability estimates needed to select or recommend items for a checklist y^{cnd}. $\mathcal{D}_{C\mathcal{B}}$ is used for the case base ($C\mathcal{B}$) and for performance evaluation. Each validation fold has 138 938 instances that constitutes 7 954 inspections. Each inspection contains a target organisation x that is used as input (x^{cnd}) to each checklist construction model $\mathcal{M}$. Each model $\mathcal{M}$ then generates a candidate checklist y^{cnd} for each given x as described in Section C.4 and C.6.1. The target length of y^{cnd} is set to $K = 15$.[2]

Statistics for all checklists created by each $\mathcal{M}$ are calculated as follows: For each generated y^{cnd}, all items $e_i \in \mathrm{y}^{cnd}$ are considered as having predicted positive answers $l_i = 1$. The rest of the N possible items where $e \notin \mathrm{y}^{cnd}$ are considered as having predicted negative. The number of true positive (TP)/false positive (FP) answers and true negative (TN)/false negative (FN) answers of y^{cnd} are estimated from empirical distributions of $\mathcal{D}_{C\mathcal{B}}$, as described by Flogard et al. [7]. This is done because every y^{cnd} is a new checklist, so most items $e_i \in \mathrm{y}^{cnd}$ do not have an observed ground truth answer l_i. The estimates are used to calculate accuracy, precision and recall statistics for each y^{cnd} via: $Acc_{\mathrm{y}^{cnd}} = \frac{TP+TN}{TP+FP+TN+FN}$, $Prec_{\mathrm{y}^{cnd}} = \frac{TP}{TP+FP}$ and $Rec_{\mathrm{y}^{cnd}} = \frac{TP}{TP+FN}$. The final mean *Acc*, *Prec* and *Rec* for each validation fold are found by averaging each statistic over all 7954 generated y^{cnd}. We also included *Prec (gt)*, which is mean precision calculated by only using the subset of items on the checklists where ground truth

[2]For CBCBR, each generated y^{cnd} also includes recommended items. The initial length of CBCBR's checklists are $K = 15$, before adding any extra items. The code for the experiment is published at https://github.com/ntnu-ai-lab/cbcbr.

Method	**Acc**	**Prec (gt)**	**Prec**	**Rec**	**Avg**	***Time (sec)***
ECL	0.365	0.177	0.176	0.592	0.328	-
DT	0.494	0.229	0.258	0.610	0.398	3 641
LR	0.513	0.247	0.272	0.664	0.424	205
RF	0.518	0.253	0.284	0.676	0.433	4 222
NBI	0.520	0.254	0.286	0.686	0.437	**7.2**
NN (MLP)	0.521	0.256	0.290	0.688	0.439	14 280
BCBR	**0.675**	0.313	0.479	0.764	0.558	8.7
CBCBR	**0.675**	**0.322**	**0.497**	**0.808**	**0.576**	8.6 (4.5)

Table C.2. The mean accuracy *Acc*, ground truth precision *Prec (gt)*, precision *Prec* and recall *Rec* for the content of the checklists, created via the various methods in the table. The *Avg* column shows the average scores of the four preceding columns. The last column shows the training times in seconds for each method.

answers are available. The statistics are comparable to mean average (MA), used to evaluate recommender systems [30].

Results and Discussion. The results are shown in Table C.2. CBCBR has the highest *Avg* score and has higher scores than BCBR on most of the statistics. This is impressive as CBCBR's checklists are longer and should generally have lower *Acc*, *Prec* and *Prec (gt)*, since only 17.7% of the instances in the data set have positive labels ($l = 1$) and longer checklists means more predicted positives. NN has the third highest *Avg* score, only marginally higher than NBI. It may be possible to improve the results for NN with more advanced parameter tuning methods, but this will unlikely be enough to match BCBR or CBCBR's performance. The domain experts' checklists (ECL) have the lowest score.

On average, the checklists constructed by CBCBR contain 9.13 violations (true positives) against 7.19 for BCBR. CBCBR's checklists also contain 18.1 items against 15.0 for BCBR. This means that CBCBR recommends 3.1 items on average, because CBCBR creates the same initial checklists as BCBR. There are 1.94 violations found among the 3.1 recommended items, which corresponds to a precision score of 0.63. Thus, CBCBR's recommendations have much higher precision than static checklists created by BCBR or other baselines in Table C.2. Overall, the results suggest that using CBCBR's dynamic checklists for labour inspections will increase efficiency and the amount of violations to working environment regulations found at the inspected organisations.

Time-wise, the slowest method is NN with an average training time of 14 280 seconds, excluding hyper parameter optimization. The fastest method is NBI with 7.2 seconds training time. CBCBR is nearly as fast, as the combined time for model training and generating an initial checklist is 8.6 seconds. This is much faster than LR and DT, which are known for their low training times. It also takes 4.5 seconds to update CBCBR and recommend new items after answering a checklist. However, the cross-validation took more than 20 days to complete because CBCBR must be trained and updated individually for each validation case. Still, the short individual training and update times for each checklist means that CBCBR can be used as an online model and we believe that further speed-ups can be achieved via parallelization.

C.7 Conclusion

In this work we show that dynamic answer-based adaptations to checklists can significantly increase the number of violations found in labour inspections. We introduce CBCBR for real-time generation and adaption of checklists, which could be employed by labour inspection agencies to increase the efficiency of their inspections. This will likely increase national and global levels of compliance with labour rights and reduce injuries (see SDG indicator 8.8.1 and 8.8.2). We are currently testing the real-world performance of CBCBR in a trial.

This paper only explores one of many possible approaches for dynamically adapting checklists. Future work could investigate other designs as well. An example is to dynamically build checklists from ground-up, starting with one item and adding K more, one-by-one as answers are obtained. Another direction for future work is to explore approaches for creating checklist items, where Cai et al. [20] could be a starting point. It could also be interesting to look into other ML methods for dynamically adapting checklists, such as RNNs. CBCBR's dynamic checklists could also be tested in other relevant tasks, such as food inspections, aviation or surgeries.

C.8 References

[1] Srdjan Jelacic, Andrew Bowdle, Bala G Nair, Kei Togashi, Daniel J Boorman, Kevin C Cain, John D Lang, and E Patchen Dellinger. Aviation-style computerized surgical safety checklist displayed on a large screen and operated by the anesthesia provider improves checklist performance. *Anesthesia & Analgesia*, 130(2):382–390, 2020.

[2] Daniel E Ho, Sam Sherman, and Phil Wyman. Do checklists make a difference? a natural experiment from food safety enforcement. *Journal of Empirical Legal Studies*, 15(2):242–277, 2018.

[3] Øyvind Dahl and Marius Søberg. Labour inspection and its impact on enterprises' compliance with safety regulations. *Safety Science Monitor*, 17(2):1–12, 2013.

[4] Nektarios Karanikas and Sikder Mohammad Tawhidul Hasan. Occupational health & safety and other worker wellbeing areas: Results from labour inspections in the bangladesh textile industry. *Safety Science*, 146:105533, 2022.

[5] W̃orld Health Organization. Joint estimates of the work-related burden of disease and injury, 2000-2016: Global monitoring report. Technical report, 2021.

[6] Ken Catchpole and Stephanie Russ. The problem with checklists. *BMJ quality & safety*, 24(9):545–549, 2015.

[7] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Bayesian feature construction for case-based reasoning: Generating good checklists. In *ICCBR*, pages 94–109. Springer, 2021.

[8] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[9] Eliot Grigg. Smarter clinical checklists: how to minimize checklist fatigue and maximize clinician performance. *Anesthesia & Analgesia*, 121(2):570–573, 2015.

[10] Leah Kulp, Aleksandra Sarcevic, Megan Cheng, and Randall S Burd. Towards dynamic checklists: Understanding contexts of use and deriving requirements for context-driven adaptation. *ACM TOCHI*, 28(2):1–33, 2021.

[11] Stefan C Christov, Heather M Conboy, Nancy Famigletti, George S Avrunin, Lori A Clarke, and Leon J Osterweil. Smart checklists to improve healthcare outcomes. In *International Workshop on SEHS*, pages 54–57, 2016.

[12] AJR De Bie, S Nan, LRE Vermeulen, PME Van Gorp, RA Bouwman, AJGH Bindels, and HHM Korsten. Intelligent dynamic clinical checklists improved checklist compliance in the intensive care unit. *BJA: British Journal of Anaesthesia*, 119(2): 231–238, 2017.

[13] Khalid Haruna, Maizatul Akmar Ismail, Suhendroyono Suhendroyono, Damiasih Damiasih, Adi Cilik Pierewan, Haruna Chiroma, and Tutut Herawan. Context-aware recommender system: A review of recent developmental process and future research direction. *Applied Sciences*, 7(12):1211, 2017.

[14] Wasim Huleihel, Soumyabrata Pal, and Ofer Shayevitz. Learning user preferences in non-stationary environments. In *AISTATS*, pages 1432–1440. PMLR, 2021.

[15] Mobin M Idrees, Leandro L Minku, Frederic Stahl, and Atta Badii. A heterogeneous online learning ensemble for non-stationary environments. *Knowledge-Based Systems*, 188:104983, 2020.

[16] Pamela J Morgan, Jenny Lam-McCulloch, Jodi Herold-McIlroy, and Jordan Tarshis. Simulation performance checklist generation using the delphi technique. *Canadian journal of anaesthesia*, 54(12):992–997, 2007.

[17] Marly Ryoko Amaya, Danieli Parreira da Silva Stalisz da Paixão, Leila Maria Mansano Sarquis, and Elaine Drehmer de Almeida Cruz. Construction and content validation of checklist for patient safety in emergency care. *Revista gaucha de enfermagem*, 37, 2017.

[18] Felicity Hasson, Sinead Keeney, and Hugh McKenna. Research guidelines for the delphi survey technique. *Journal of advanced nursing*, 32(4):1008–1015, 2000.

[19] Haoran Zhang, Quaid Morris, Berk Ustun, and Marzyeh Ghassemi. Learning optimal predictive checklists. *NeurIPS*, 34, 2021.

[20] Hubo Cai, JungHo Jeon, Xin Xu, Yuxi Zhang, Liu Yang, et al. Automating the generation of construction checklists. Technical report, Purdue University. Joint Transportation Research Program, 2020.

[21] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. 2011.

[22] Gharbi Alshammari, Jose L Jorro-Aragoneses, Stelios Kapetanakis, Miltos Petridis, Juan A Recio-García, and Belén Díaz-Agudo. A hybrid cbr approach for the long tail problem in recommender systems. In *ICCBR*, pages 35–45. Springer, 2017.

[23] Jose Luis Jorro-Aragoneses, Marta Caro-Martínez, Belén Díaz-Agudo, and Juan A Recio-García. A user-centric evaluation to generate case-based explanations using formal concept analysis. In *ICCBR*, pages 195–210. Springer, 2020.

[24] Hoda Nikpour and Agnar Aamodt. Fault diagnosis under uncertain situations within a bayesian knowledge-intensive cbr system. *Progress in Artificial Intelligence*, pages 1–14, 2021.

[25] Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *NeurIPS*, pages 1952–1960, 2014.

[26] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009.

[27] Ole Jakob Mengshoel, Eirik Flogard, Jon Riege, and Tong Yu. Stochastic local search heuristics for efficient feature selection: An experimental study. In *Norsk IKT-konferanse for forskning og utdanning*, pages 58–71, 2021.

[28] Kerstin Bach, Bjørn Magnus Mathisen, and Amar Jaiswal. Demonstrating the mycbr rest api. In *ICCBR Workshops*, pages 144–155, 2019.

[29] Shuo Zhang and Krisztian Balog. Evaluating conversational recommender systems via user simulation. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1512–1520, 2020.

[30] Garima Natani and Satoru Watanabe. Knowledge graph-based data transformation recommendation engine. In *IEEE BigData*, pages 4617–4623. IEEE, 2021.

Paper D: A Dataset for Efforts Towards Achieving the Sustainable Development Goal of Safe Working Environments

Published in: *Advances in Neural Information Processing Systems (NeurIPS),* 2022.
Acceptance rate: *25.6% (2022)*
Norwegian Scientific Index Level: 2 (highest)

Notes. This paper is published in Advances on Neural Information Processing Systems, which is one of the most leading publications in the field of machine learning (ML). The channel receives contributions from the top ML researchers and has an acceptance rate of 25.6% for 2022. Neurips is ranked among the top 10 most influential publication channels in science by Google Scholar in 2022. The publication channel is also ranked at the highest level (2) in the Norwegian Scientific Index in 2022, placing it above most other conferences and journals in the AI field. Level 2 is generally reserved for the most leading publication channels, publishing the most significant research within their respective fields of science. Advances on Neural Information Processing Systems is registered with an ISSN number and is regarded as a journal in the Norwegian publication channel register, meaning that the paper is worth 3 publication points. The paper has been reformatted to ensure readability and consistency with the other papers in the thesis.

Contribution Statement. My contribution to the paper includes the ideas presented in the paper, writing the first draft, data collection and carrying out the experiments described in the paper. The co-author of the paper is my supervisor who helped to refine the paper. He also helped to write the final version of the abstract and introduction of the paper.

A Dataset for Efforts Towards Achieving the Sustainable Development Goal of Safe Working Environments

Eirik Lund Flogard[1,2] and **Ole Jakob Mengshoel**[2]
[1]Norwegian Labour Inspection Authority
[2]Norwegian University of Science and Technology
eirik.flogard@arbeidstilsynet.no, ole.j.mengshoel@ntnu.no

Abstract

Among United Nations' 17 Sustainable Development Goals (SDGs), we highlight SDG 8 on Decent Work and Economic Growth. Specifically, we consider how to achieve subgoal 8.8, "protect labour rights and promote safe working environments for all workers [. . .]", in light of poor health, safety and environment (HSE) conditions being a widespread problem at workplaces. In EU alone, it is estimated that more than 4000 deaths occur each year due to poor working conditions. To handle the problem and achieve SDG 8, governmental agencies conduct labour inspections and it is therefore essential that these are carried out efficiently. Current research suggests that machine learning (ML) can be used to improve labour inspections, for instance by selecting organisations for inspections more effectively. However, the research in this area is very limited, in part due to a lack of publicly available data. Consequently, we introduce a new dataset called the Labour Inspection Checklists Dataset (LICD), which we have made publicly available. LICD consists of 63634 instances where each instance is an inspection conducted by the Norwegian Labour Inspection Authority. LICD has 577 features and labels. The dataset provides several ML research opportunities; we discuss two demonstration experiments. One experiment deals with the problem of selecting a relevant checklist for inspecting a given target organisation. The other experiment concerns the problem of predicting HSE violations, given a specific checklist and a target organisation. Our experimental results, while promising, suggest that achieving good ML classification performance is difficult for both problems. This motivates future research to improve ML performance, inspire other data analysis efforts, and ultimately achieve SDG 8.

D.1 Introduction

Background. Poor health, safety and environment (HSE) conditions in workplaces is a widespread problem that negatively affects both individuals and society. Every year in EU alone, more than three million workers are victims of serious accidents causing more than 4000 deaths due to poor working conditions [1]. World-wide, it has been estimated that at least 9.8 million people are in forced labour (2005) [2]. Labour inspections, a part of the International Labour Organization's "Decent Work Agenda," seek to preventing this and enforce regulations that protect workers' health, environment and safety [3]. Labour inspections are also important to globally achieve UN's Sustainable Development Goal (SDG) 8.8, "protect labour rights and promote safe working environments for all workers [. . .]."

To identify poor HSE conditions, labour inspection agencies use checklists to survey organisations for non-compliance [4–6]. Each checklist contains questions that relate to common working environment violations within one or more industries. The answers to the questions indicate whether non-compliance to HSE and working environment regulations are found within the targeted organisations. When non-compliance is found, the agency follows up with reactions against the organisations. Checklists are also used in other high-stakes domains including aviation, food inspection and surgery [7, 8]. They are also used in machine learning (ML) for various tasks, such as testing and evaluating NLP models [9, 10] or fairness assessments [11].

Since labour inspection agencies usually have limited resources, it is vital that the inspections are carried out efficiently. This is challenging, as inspection strategies are difficult to adapt to a world that is constantly changing [12, 13]. Therefore, some agencies have recently started to use ML to select organisations for inspections [14, 15]. This is shown to increase inspection efficiency, as violations are far more common in the organisations that are selected using ML. However, ML research in this areas has been very limited. To our knowledge, and other than our previous work [15, 16], there is a lack of publicly available datasets to enable such research.

The Labour Inspection Checklists Dataset. In order to address SDG 8, specifically SDG 8.8, we present the Labour Inspection Checklists dataset (LICD) [17]. We aim to support and inspire ML research on labour inspections. Our current LICD dataset complements previous SDG-related NeurIPS datasets and benchmarks [18, 19], since none of them covers SDG 8. LICD is a Norwegian dataset, translated to English, that consists of results from inspections conducted by the Norwegian Labour Inspection Authority (NLIA) between January 2012 and June 2019. The dataset contains 63634 instances and 575 features. Each instance contains organisational and financial information about the organisation targeted for the inspection, a description of the checklist used and a binary variable that indicates whether violations were found. To demonstrate possible LICD use-cases, we introduce and show initial experimental ML results for these two problems:

The Checklist Selection Problem (CLSP): *Let there be a selection of N checklists and a target organisation* $\mathbf{x}$*. Given the N possible checklists, select the best checklist* y *to survey the target organisation* $\mathbf{x}$*. In this setting, the best checklist is the checklist that its user (the inspector) considers to be most relevant for surveying* $\mathbf{x}$*. The problem can be seen as a classification problem.*

The Non-compliance Classification Problem (NCP): *Given a checklist* y *and a target organisation* $\mathbf{x}$*, classify the target organisation's compliance* l *to any of the regulations given by the content of* y*. The value of* l *is unknown until the completion of the inspection and belongs to a Bernoulli distribution where* $l = 1$ *means that the target organisation is non-compliant and* $l = 0$ *means that the organisation is compliant.*

Both problems are non-trivial and important to promote safe workplaces. For CLSP, there are $N = 369$ different checklists that can be used for any given organisation. Table D.1 demonstrates how labour inspection checklists can vary significantly, even among similar organisations. The reason for this is that organisations are subjected to numerous regulations and only a few of these are covered by each checklist. Selecting a checklist that covers all the most relevant subset of regulations is no trivial task, so CLSP is therefore difficult to solve. The task is also important, as a consequence of selecting wrong checklists may be that working environment violations are left unaddressed [15]. NCP is also important as it can be useful for selecting the best

Checklist Content	Industry	County	Result
Working agreements, HSE and working environment training, Working hour schedules, Occupational health service, Building and equipment conditions, Risk assessment and measures, ...	Accommodation business	Oslo	Compliant
Market control - chemicals, Substance register.	Manufacture of metal products	Rogaland	Non-compliant
HSE and working environment training, Occupational health service, Mapping and risk assessment of chemicals and biological factors, Personal protective gear - technical, Noise exposure, ...	Manufacture of metal products	Viken	Non-compliant

Table D.1. Summary of three inspections of three different organisations. The inspections are conducted in two different industries, with three different checklists. The last two rows show different checklists being used for similar organisations, illustrating that checklists are situation dependent.

combinations of checklists and organisations for inspections, by predicting whether violations at a potential organisation is found when using a potential checklist. Alternatively, the checklist-component in NCP can be omitted so that the problem only focuses on predicting violations in organisations [14]. Both CLSP and NCP could also be solved as a multi-objective problem of selecting the most relevant checklist (CLSP) that also maximizes the likelihood for finding non-compliance in the inspected organisation (NCP). However, for simplicity we treat CLSP and NCP as single-objective problems in this paper.

Paper Overview. The rest of the paper has the following structure. In Section D.2 we present an overview of related work. A formal description of LICD is provided in Section D.3. Section D.4 describes how the data was collected and processed. An analysis of the dataset is provided in Section D.5. Two baseline CLSP and NCP experiments are conducted in Section D.6. In Section D.7, we discuss implications of this work, including ideas for future work.

D.2 Related Work

Regulation Enforcement Datasets. Earlier, we used a dataset similar to LICD to construct new checklists via ML [15, 16]. However, the previous dataset only contains 4 independent features and cannot easily be used for other tasks beyond constructing new checklists. In contrast, LICD contains as many features as we have been able to collect (that can also be shared to the public within legal and ethical limits). Further, LICD is designed for checklist selection and non-compliance prediction rather than checklist construction. Since selecting existing checklists may be much simpler than creating new ones, an ML method for doing so may be easier to adopt for labour inspection agencies.

Superficially similar labour inspection datasets from other countries are publicly available, such as the Danish Smiley dataset [20]. Unfortunately, the Smiley dataset only contains instances where organisational non-compliance is found (positive labels); there are no instances where the organisation is compliant (negative labels) that also include inspection details. Another example is a dataset published by the American OSHA, which is frequently used in health and safety research [21, 22]. However, the dataset is not complete from an ML perspective since it only contains records about the regulations that were found non-compliant at the inspections (positive labels). The OSHA dataset also contains injury records, so Johnson et al. [23] propose to use ML on them to prioritize organisations for inspections. However, injury data may not be reliable for this purpose due to bias and under-reporting of accidents and injuries to authorities [24]. Other labour inspection datasets have been analysed [5, 6], but none are published openly. A mining safety inspection dataset is openly available [25], but such inspections are highly specialized; health and safety hazards and regulations in other industries are quite different. Datasets from other enforcement domains have also been published. These datasets include environmental inspections where ML has been used to predict violations in facilities [26]. There is also the Vancouver Crime dataset for law enforcement, which has been used for both criminology and ML research [27, 28]. Another example is the Chicago Food Inspection dataset, which includes an ML classification model [29, 30]. These datasets are fundamentally different from LICD, but highlight the importance of ML research within other branches of regulation enforcement.

Checklists in Other Domains. In addition to labour inspections, checklists are used in other situations where ensuring human health and safety is critical. In surgeries, checklists are used to ensure compliance to safety standards. For example, the WHO Surgical Safety Checklist from 2008 has substantial positive effects on patients' safety [31]. However, there are challenges related to implementation of the checklist in daily use. Some of the challenges are communication errors, lack of user compliance and lack of flexibility since standards of medicine varies from country to country [32]. Cockpit checklists in the aviation industry also face similar challenges, as improper checklist usage can lead to accidents [33]. The success of using a checklist may depend on having the correct content for a given context. Selecting the most relevant checklist for a given context is also one of the motivations for our work.

Long-tail Classification. In Section D.5 we observe that the distribution of checklists in LICD is long-tailed and since the dataset contains $N = 369$ unique checklists, the dataset could be relevant for methods which address long tail classification problems. Dealing with long tail distributions, where classes tend to be distributed according to Zipf's law, can be challenging as models usually perform better when dealing with head-classes than the tail-classes [34]. Some well known long tail distributed datasets besides LICD already exist, such as CIFAR-100-LT, Fashion-MNIST or ImageNet-LT [35, 36], but these are not directly SDG-relevant. The most well-known approaches to deal with long tailed distributions are balancing methods, such as under- or oversampling [37]. Various ML and feature extraction-based methods to improve classification performance on long-tailed datasets have also been proposed [38–41]. However, the scope of this paper is to describe LICD rather than developing new methods for addressing problems such as classification of long-tail distributions.

D.3 Dataset Description

The LICD consists of 63634 instances with past inspections conducted by NLIA between 01/01/2012 and 01/06/2019 [17]. Each instance in LICD is described via 575 features and two target variables. Each feature represents either organisational or financial information about the inspected organisation. The first target variable is an identifier for the checklist that was used to inspect the organisation. The second target variable denotes whether non-compliance was found at the inspection. The features, target variables and column names of the dataset have been translated from Norwegian

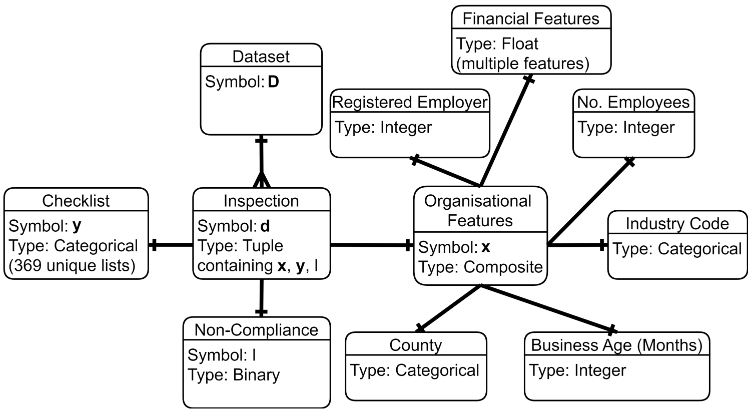


Figure D.1. Diagram showing relations between different features and entities in the dataset. All of these are contained within a single table.

to English.

Data Collection. Let D be LICD (table) where each instance (row) consists of an inspection **d**. The initiation of an inspection usually happens as a result of risk assessments. Each inspection takes 1-4 hours and is carried out using a checklist (y), which is a form that consists of yes/no questions. A no-answer to any of the questions means that the organisation is non-compliant ($l = 1$) to one of the regulations enforced by the agency. After the completion of the inspection, the completed checklist is uploaded digitally to NLIA's case management system. A report is then generated and delivered to the target organisation, regardless of the findings at the inspection. For any violations that are found, the corresponding questions and answers in the completed checklist are quoted in the report.

The information from each report is used to create an instance **d** in the dataset **D**. Thus, each instance in the dataset is collected organically as part of NLIA's daily operations without additional interventions. Figure D.1 shows the relations between various entities of the dataset. A description of these follows below.

Inspection. An inspection is a single instance in LICD and can be viewed as a tuple $\mathbf{d} = (\mathbf{x}, y, l)$ where **x** is a target organisation, y is a checklist and l is the outcome of the inspection after using y to survey **x**. **x** consists of the 575 features in the dataset, while y and l represent the target variables.

Organisation. An organisation $\mathbf{x} \in \mathbf{d}$ is described by the 575 features in LICD. Each feature contains either organisational or financial information about the target organisation. Figure D.1 shows how organisational information is contained in these features: Industry codes[1] (categorical), presence in the employer and VAT register (binary), county (categorical) and number of employees (integer). The other features, which are real numbers, contain financial information about the inspected organisation. Many of the instances in LICD have missing values ("NULL") for the financial features whenever these are irrelevant for the organisations' daily operations or fiscal reports. We therefore recommend replacing these missing values with 0 for the purpose of training ML models on the dataset.

[1]Interpretation table for industry codes: https://www.ssb.no/en/klass/klassifikasjoner/6

Checklist. Each instance $\mathbf{d}$ in LICD contains a checklist y, used to survey an organisation $\mathbf{x}$. The content of y is described by a column named Checklist Content, which lists of all the topics that are being investigated during the inspection. Every checklist y is also associated with an identifier (Checklist ID). The Checklist ID is categorical and is one of the target variables in LICD. Since there are 369 unique checklists in the dataset, the Checklist ID can take on 369 different values.

Non-Compliance. Each inspection has an outcome l in LICD, where $l = 1$ denotes that at least one of the questions on the checklist were found to be non-compliant (violation) at the inspection. The value $l = 0$ means that no checklist questions were found to be non-compliant. The outcome l is also considered to be one of the potential target variables in the dataset.

D.4 Dataset Acquisition

Processing and Validation. LICD is a new dataset that is now being made publicly available, with exclusive permission from NLIA. The dataset was retrieved from the agency's databases using MSSQL17. Quality and integrity assurance is dealt with by NLIA's case management system, since this is essential for the system's operations (see Section D.3). This assurance includes validation of data type, range and constraints. Consistency is also ensured by saving "information snapshots" of the target organisations at the time of the inspection, so that inspection records are unaffected by any updates to the organisations' registered information.

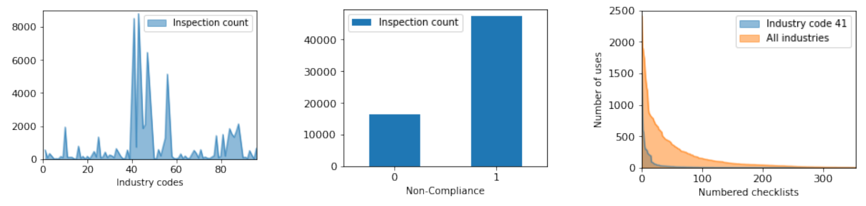
Data Source Availability and Harm Prevention. LICD contains information from sources that are publicly available, to some extent. Some of the organisational and financial features are available through Norway's official Central Coordinating Register, but the register has no historical records. The checklists and inspection outcomes are only available from inspection reports. Some of these may be available to the public, but access is granted only on a case-by-case basis via requests.

Since it is difficult to prevent identification of organisations in LICD, our strategy for harm prevention is to ensure that the dataset only contains information that is safe to make publicly accessible. For this reason we have not included details such as which regulations were found to be non-compliant. As an extra precaution we also deliberately make it difficult to identify organisations in the dataset. To do so we have excluded the organisation names, identifiers and precise locations from the dataset.

D.5 Analysis of the Dataset

In this section we conduct some analyses on LICD to highlight how the target labels and some of the key features of the dataset are distributed. We focus on industry and location (county), because these features are known to be important for labour inspections [12, 15, 16].

Distribution of Inspections Over Industries. Labour inspections are industry-oriented [12], so Figure D.2a shows a histogram of inspections over different industry codes in the dataset. The horizontal axis represents the industry codes, which can be regarded as an ordinal feature where each number represents a specific industry.



(A) Histogram of inspections over industry codes.

(B) Distribution of non-compliance in LICD.

(C) Histogram of checklists. Each unique checklist is denoted as a number on the x-axis.

Figure D.2. Histograms of inspections, non-compliance and checklists with discrete unit bins on the horizontal axes. The vertical axes on the figures represent the number of occurrences in LICD.

Industries with codes in close proximity to each other are often related. The vertical axis shows how many inspections that have been carried out in a particular industry code. As seen on the figure, most of the inspections are focused on industry codes from 50 to 60, which correspond to most of the building and road construction industries.

Distribution of Non-Compliance. As seen in Figure D.2b, non-compliance ($l = 1$) is found in more than 40000 inspections in LICD. In other words, at least one violation is found in 74% of the inspections in the dataset. The fact that the majority of the inspected organisations are found to be non-compliant reflects the importance of labour inspections, in terms of correcting health, environment and safety problems in workplaces.

Distribution of Checklists. Figure D.2c shows a histogram of the observation count of each unique checklist for all industries. The checklists are shown on the horizontal axis, ordered according to their number of observations in LICD. The vertical axis shows the number of observations in the dataset. The figure suggests that checklists usage follows a long-tailed distribution [34], since only a small fraction of the checklists are used very often.

Figure D.2c also shows the distribution of checklists used within industry code 41, which is long-tailed. Industry code 41 corresponds to “building construction” and is one of the industries with the highest number of inspections in LICD. The figure reveals that more than 140 different unique checklists are used for inspections within that industry. However, most of the inspections are carried out using 20 of the available checklists. The high number of checklists used within a single industry code points to the fact that there is a significant diversity in the health, environmental and safety risks for organisations, even within the same industry.

County	Count #
Agder	3375
Innlandet	5179
Møre & Romsdal	4368
Nordland	3555
Oslo	7688
Rogaland	5358
Svalbard	59
Troms & Finnmark	4148
Trøndelag	5463
Vestfold & Telemark	4922
Vestland	5773
Viken	13746

Table D.2. Overview of the distribution of inspections for different locations in the dataset.

Distribution of Inspections Over Regions. Table D.2 shows the distribution of inspections across different counties or regions (fylke). Most inspections are carried out in Viken and Oslo, the two regions with the highest populations in Norway. The inspection counts seem to correlate with the population count of each region.

D.6 Demonstration Experiments

In this section we conduct two simple experiments to demonstrate potential use-cases for LICD. The first experiment addresses the problem of predicting the best checklist for a given organisation. The second experiment addresses the problem of predicting non-compliance, when a specific organisation and checklist is given. Due to the high number of features in LICD, we also evaluate some simple feature selection algorithms. Feature selection is known to promote explainability for ML in tasks involving high-stakes decision making, such as labour inspections [15, 42]. Feature selection can also improve computational and model performance via data dimensionality reductions [43, 44] and could therefore be another use-case for LICD. We used an earlier, unpublished version of LICD to evaluate feature selection methods in our previous work [43].

D.6.1 Data Preprocessing

For the experiments, we use one-hot encoding for categorical features. The other features are used directly in their original form. Many of the financial features contain missing values, which are denoted as NULL (see Section D.3). We replaced these missing values with 0 for the experiments, because this is usually the most correct interpretation. We also excluded financial features with less than 10 observations, since these are unlikely to be useful for ML. Since we are using feature selection for the experiments, we decided to set this threshold fairly low.

D.6.2 Setup and Environment

The feature selection and ML methods used for the experiments are implemented via Scikit-learn. We study the following methods for feature selection: Anova F, χ^2, Model Coefficients, Mutual Information, Forward Selection and Recursive Elimination. The ML methods used for the experiments are: Decision tree (DT), Logistic regression (LR), Naive Bayes' Classifier (NBC), K-Nearest-Neighbor (k-NN), AdaBoost, GradientBoost and Multi layered Perceptron (MLP). We are using GridSearchCV for hyper parameter tuning for k-NN, AdaBoost, GradientBoost and MLP. For each method, prediction threshold is set to 0.5 (NCP) or the class that has the highest prediction score (CLSP). We decided to not set the prediction threshold individually for each method, in order to keep the experiment simple and to avoid introducing bias.

Each ML method is always evaluated on 8 different feature set sizes that are selected via feature selection. The set sizes are 0.1%, 0.5%, 1%, 5%, 10%, 20%, 30%, 40% and 50% of the 575 features in the dataset, rounded up to the closest integer.

A Dell Precision 5560 laptop with Intel i9 11950h at 5Ghz, 64GB RAM at 3200Mhz, Nvidia Quadro RTX A2000 and Windows 10 are used for the experiments. The experiments are conducted in a Python environment using Scikit-learn 0.24 and Jupyter Notebook.

D.6.3 Experiment 1: The Checklist Selection Problem (CLSP)

The goal of this demonstration experiment is to establish baselines for solving the CLSP problem described in Section D.1. The experiment is broken down in two phases: a pilot experiment where we evaluate feature selection methods for the problem, and the main demonstration experiment.

Evaluation of Feature Selection Methods. Since LICD contains many features, we decided to assess a range of feature selection methods for the main experiment. The feature selection methods and evaluation details are discussed in Section D.6.2. We use DT for the evaluation, since it is fast and compatible with all the feature selection methods. After assessing each feature selection method, we then take the top three best performing methods in terms of accuracy and use the two fastest performing methods for our experiment.

The results from testing the feature selection methods are listed in Table D.3. There are only minor differences between most of the methods in the test. Forward Selection, Anova F and Mutual Information have the best recorded accuracy scores. Forward Selection has the highest accuracy but the score is recorded from one run where only 0.1% of the features are selected. Forward Selection was unable to complete within two hours for any of the larger feature sets. We decided to move forward with Anova F and Mutual Information for the main experiment, since they are the fastest performing methods among the top three with highest accuracy.

Method	***Acc***	***Time***
Anova F	**.106**	**.75**
χ^2	.070	.78
Model coefficients	.099	19.1
Mutual Info	**.106**	296
Forward Selection	**.108**	6146
Recursive elimination	.105	306

Table D.3. Result from the feature selection evaluation for CLSP using DT, with time measured in seconds.

Design of the Main Experiment. We are using the ML methods and setup described in Section D.6.2. Each feature selection algorithm is applied to LICD before model training and evaluation, using the 8 configurations of feature set sizes described in Section D.6.2.

After performing hyper parameter tuning, the optimal configuration for AdaBoost is 0.5 learning rate and 50 estimators. The optimal configuration for GradientBoost is 0.1 learning rate and 20 estimators. For k-NN, the optimal configuration is distance-based weights and $k = 100$ neighbors. The best settings for MLP are logistic activation function, 100 hidden layers and constant learning rate.

Each configuration is evaluated using 5-fold cross-validation with randomly selected training-evaluation sets from LICD. The performance is measured in terms of balanced accuracy[2], accuracy, precision and recall scores using the available methods in the Scikit-learn library. The average standard deviation for each cross-validation is also recorded. The results for each method are recorded by calculating the average of the scores reported from the 8 feature selection configurations.

[2]Balanced accuracy is accuracy score calculated via Sklearn with class-balanced sample weights.

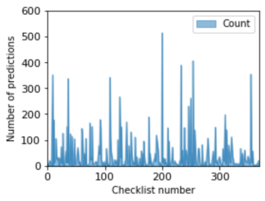
Method	*Mutual Info*				*Anova F*				*Time*
	Bal. Acc	***Acc***	***Prec***	***Rec***	***Bal. Acc***	***Acc***	***Prec***	***Rec***	
LR	.01±0	.02±.01	0±0	0±0	.01±0	.03±.01	0±0	.01±0	381
NBC	.01±0	.01±0	.01±0	.01±0	.02±0	.01±.01	.01±0	.02±0	**13.6**
DT	**.06±.01**	**.09±.01**	**.05±0**	**.04±.01**	**.06±.01**	**.09±.02**	**.05±0**	**.05±.01**	33.2
k-NN	.05±.01	.08±.01	**.05±.01**	.03±0	.04±0	.07±.01	.04±0	.03±0	40.0
AdaBoost	.01±0	.05±.02	0±0	.01±0	.02±0	.04±.02	0±0	.02±0	619
GradientBoost	.03±.01	.06±.02	.03±.01	.02±.01	.04±.01	.08±.02	.03±0	.03±0	22316
MLP	.01±.01	.04±.02	.01±.01	.01±0	.02±.01	.05±.02	.02±.01	.02±.01	338

TABLE D.4. Prediction performance with average standard deviations and run times from the CLSP main experiment on LICD. Times are measured in seconds.

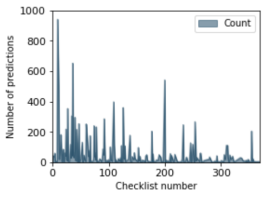
Results and Discussion. The results from the main demonstration experiment are shown in Table D.4, where the best score in each column is highlighted. In overall, it seems that achieving high prediction performance for CLSP is challenging. The prediction performance scores are reasonably low since the target variable consists of 369 different classes. When comparing accuracy and balanced accuracy in the table, the accuracy is in most cases greater than the balanced accuracy. This is probably caused by the long tailed distribution of the target variable classes (unique checklists) in the dataset. Out of the ML-methods we tested, DT had the highest balanced accuracy, accuracy, precision and recall scores. The results are somewhat surprising, especially since both AdaBoost and GradientBoost are based on using Decision trees as weak learners to improve model predictions. However, these methods are also susceptible to overfitting, which could be caused by the long-tailed class distribution of the target variable [34]. k-NN also performed well in comparison, ranked as second best on nearly all the scores in Table D.4. However, the results show that there is not significant differences in balanced accuracy, accuracy and precision between the two best configurations of k-NN and DT. By naively predicting the most frequently used checklist from Figure D.2c as positive, one can expect an accuracy and precision score of $\frac{2414}{63634} \approx 0.04$. Both k-NN and DT perform better than this, which is promising. It is still questionable whether these methods also would outperform an inspector or domain expert. Time-wise, DT has an average cross-validation time of 33.2 seconds. This is slightly better than k-NN's 40.0 seconds.

Figure D.3a and D.3b show the number predictions that each of the 369 checklists receives from DT and k-NN, and offer additional insights regarding the performance of the two best configurations in Table D.4. Ideally, the distributions in Figure D.3a and D.3b should be similar to the distribution of ground truth labels in Figure D.3c. Compared to the distribution of predictions from k-NN, the distribution for DT is more similar. In particular, the checklist with number 200 has the highest number of observations in both Figure D.3a and D.3c. For k-NN, checklist number 9 has 940 predictions which is over 3 times as many as the number of ground truth labels. Also, the predictions are much more concentrated between checklist number 0-100 in comparison to DT and the ground truths. Despite the similarities between Figure D.3a and D.3c, the amount of true positives (matches between predicted checklists and ground truths) for DT is low. One possible reason for this discrepancy is discussed in Section D.6.5.

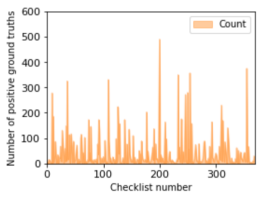
The overall difference in prediction performance between Mutual Information and Anova F in Table D.4 is minor, but may slightly be in favour of Anova F for all the methods except k-NN and AdaBoost. Despite this, the highest ranked features for Mutual Info seem to be more relevant than for Anova F. The highest ranked features for Mutual Info include business age, public fees, payroll costs, industry codes and counties.



(A) Distribution of predictions for DT, using feature selection with Anova F and a 10% set size.



(B) Distribution of predictions for k-NN, using Mutual Info and a 10% feature set size.



(C) Distribution of ground truth labels.

Figure D.3. Distributions on the evaluation set of a random paired 80-20 training-evaluation split. The horizontal axes represent the identifiers for 369 possible checklists (classes). The vertical axes on the figures represent the number of observations for each class in the evaluation set.

For Anova F, the highest ranked features are business age and financial features such as assets, equity and bank deposits, dept, costs (including payroll), revenues, investments and profits. Industry codes are not ranked high by Anova F, which is surprising considering that checklists aim to address HSE risks which typically are industry-specific [15]. Thus, the results indicate that feature selection for ML is not always intuitive.

The goal of this experiment is to solve the NCP problem described in Section D.1. The experiment is structured in two phases in the same manner as in Section D.6.3, with an evaluation of feature selection methods before the main experiment.

Evaluation of Feature Selection Methods. The assessment and selection of feature selection methods is done as in the previous experiment, where the two fastest performing methods out of the three methods with the highest accuracy are selected for the main experiment. We are using the feature selection methods and the feature set sizes described in Section D.6.2. The feature selection methods are evaluated using DT.

D.6.4 *Experiment 2: The Non-Compliance Classification Problem (NCP)*

The results from the evaluation are shown in Table D.5. The best performing method was Forward Selection, but we were only able to run it on the 0.1% and 0.5% feature set sizes within the time limit of two hours. Thus, the recorded accuracy is the average for only these two feature sets. The method with the second best accuracy score is Anova F and χ^2 is the third best scoring method. Time-wise χ^2 is the best performing method with an average completion time of 0.29 seconds for all the feature set sizes. The worst performing method was

Method	***Acc***	***Time***
χ^2	**.667**	**.29**
Anova F	**.684**	.58
Mutual Info	.661	229
Model coefficients	.657	12.3
Forward Selection	**.750**	5464
Recursive elimination	.658	300

Table D.5. Result from the feature selection evaluation for NCP with DT.

Method	χ^2					*Anova F*					*Time*
	Bal. Acc	***Acc***	***Prec***	***Rec***	***AUC***	***Bal. Acc***	***Acc***	***Prec***	***Rec***	***AUC***	
LR	.42±.02	.44±.02	.68±.02	.46±.02	.41±.02	.46±.02	.45±.02	.72±.02	.43±.02	.47±.02	3.56
NBC	.56±.04	.49±.11	.72±.20	.42±.18	.57±.04	.57±.02	.53±.02	.81±.02	.48±.02	.59±.02	**1.06**
DT	.54±.01	.45±.02	.59±.01	.35±.03	.57±.01	.51±0	.30±0	.20±0	.08±.01	.56±.01	14.2
k-NN	**.58±.02**	**.53±.04**	.81±.01	**.49±.07**	.61±.02	.57±.02	.52±.02	.79±.06	.47±.03	.61±.02	100
AdaBoost	**.58±.01**	.51±.03	**.82±.01**	.44±.07	**.63±.02**	**.62±.01**	**.57±.02**	**.84±.01**	**.52±.04**	**.68±.02**	241
GradBoost	**.58±.01**	.50±.03	**.82±.01**	.43±.06	**.63±.02**	**.62±.01**	**.57±.02**	**.84±.01**	.51±.04	**.68±.02**	1352
MLP	.53±.01	.40±.03	.78±.02	.27±.05	.53±.01	.54±.01	.39±.02	.82±.02	.23±.05	.56±.02	27.7

TABLE D.6. Results with average standard deviations from the NCP main experiment. The average time in seconds per cross-validation is shown on the far right column.

Forward Selection with almost 1.5 hours. Even though Forward Selection is the best performing method in terms of accuracy, the long running time makes it unfeasible for large feature sets or ML-methods with high computational complexity. Thus, we decide to use χ^2 and Anova F for the main experiment.

Design of the Main Experiment. For the experiment, we use the ML methods and configurations described in Section D.6.2. We also performed hyper parameter tuning for AdaBoost, GradientBoost, MLP and k-NN. After performing hyper parameter tuning, the optimal configuration for AdaBoost is 1.0 learning rate and 200 estimators. The optimal configuration for GradientBoost is 0.1 learning rate and 500 estimators. For MLP, the optimal configuration is logistic activation, 20 hidden layers, 0.0001 alpha and adaptive learning rate. The best configuration for k-NN is $k = 500$ neighbors and uniform weights.

All the ML methods in the experiment are evaluated using 5-fold cross validation where each fold consists of separate training (72%), test (8%) and evaluation sets (20%). Since the target variable is unbalanced with 74% positive labels (see Figure D.2b), the prediction thresholds for the evaluation sets are set to the median prediction-scores of the corresponding test sets. As a result there should be an approximately equal number of predicted positives and predicted negatives for each evaluation set. The performance of each ML method is measured using the same statistics with standard deviations as in Section D.6.3. Area under receiver operating characteristic curve (AUC) is also used.

Results and Discussion. The results from the experiment are shown in Table D.6. These results are a bit more nuanced compared to the previous experiment. The best configurations are AdaBoost and GradientBoost with Anova F for feature selection, with significantly higher classification performance scores compared to χ^2. For χ^2, the best performing methods are AdaBoost, GradientBoost and k-NN. k-NN has slightly higher accuracy and recall scores than AdaBoost and GradientBoost, but AUC and precision scores are slightly lower. However, these differences are not significant. Time-wise, k-NN has the best performance with an average time of 100 seconds. AdaBoost outperforms GradientBoost with an average time of only 241 seconds, compared to 1352 for GradientBoost. Overall, AdaBoost seems to be the best method in this experiment, in terms of both time and classification performance.

The AUC score for AdaBoost is only 0.68, which indicates that there is room for improvements in terms of classification performance. On a first glance, the precision score of 0.84 looks better than the AUC score. However, the dataset is imbalanced 74:26 towards positives (non-compliance), so predicting all labels in the dataset as positive would yield a precision of 0.74. The difference is only 0.12 points, but this also translates into a 66% increase (3.16 to 5.25) in the odds of finding non-compliance in an organisation that is predicted as positive by the model. The difference is therefore an important improvement from a labour inspection perspective.

The highest ranked features from feature selection with Anova F are business age, number of employees, unpaid public fees, revenue, costs, industry codes and county (location). For χ^2, the highest ranked features are somewhat limited in terms of variety and only include financial features related to taxes, revenue, costs, loans and equity. Thus, we would argue that Anova F selects a better, more informed feature set compared to χ^2. Anova F also yields higher performance than χ^2 for most of the methods in Table D.6.

D.6.5 *Machine Learning Performance for CLSP and NCP*

While some of the results from the experiments look promising, none of the methods we tested perform very well in terms of classification performance on either CLSP or NCP. This is also a motivation to use LICD in future research that aims to develop new, better performing ML methods. For this domain, a high precision score is arguably the most important statistic because it means that more effective checklists are being selected (CLSP) or a higher rate of non-compliance is found per inspection (NCP). A high recall is also desirable, but less important. Accuracy or AUC is also important, in order to ensure that ML methods have decent classification performance.

For CLSP, the highest recorded precision score in Table D.4 is only 0.05 (DT). This is better than the naive selection strategy (0.04), but it is still questionable whether DT or any of the other methods perform well enough to be useful from a labour inspection perspective. However, it may be possible to improve performance by treating CLSP as a recommendation problem where a fixed number of checklists with the highest prediction scores are selected, leaving the user to decide which of them to use. This approach could be considered since the results in Figure D.3 may suggest that there often are multiple optimal checklists for a given organisation, while CLSP assumes that there is only one. Collecting more information or features for LICD could be another way to improve performance, but more research is needed in order to understand what kind of information that should be collected.

There may also be ways to increase ML classification performance for NCP. The results for Forward Selection in Table D.5 indicate that wrapper-based methods for feature selection may increase accuracy. Although using Forward Selection on LICD seems to be time-wise infeasible for feature sets larger than 0.1%, some of the more recent wrapper-based algorithms such as Stochastic Local Search or Differential Evolution could be more viable as they have comparably lower computational costs [43, 45–47].

D.7 Conclusion and Future Work

In this paper we have proposed LICD, a new dataset with labour inspection checklists. The dataset can be used to address working environment violations in organisations. Addressing such violations is important for efforts towards achieving SDG 8.8, to "protect labour rights and promote safe working environment for all workers". Research on ML for labour inspection is currently limited, so our motivation for this work is to promote further research on this subject.

LICD consists of 63634 instances with past inspections conducted by NLIA. The dataset contains 575 features and 2 possible target variables: Non-compliance and Checklist. Based on the target variables, the dataset could potentially be used for the following tasks: a) To select the most relevant checklist to inspect a given target organisation (CLSP). b) To classify whether non-compliance is found at an inspection, for a given organisation and checklist (NCP). Two demonstration experiments with CLSP and NCP suggest that they are promising but difficult problems, thus motivating future research.

A potential direction for future work is to explore ways to solve a combination of both CLSP and NCP as a multi-objective optimizations problem: to select the most relevant inspection checklist that maximizes the number of violations found in a given organization. A simple way to solve the problem could be to use a two-stage approach for selecting checklists. The first stage could be to select a subset of relevant checklists as candidates for the second stage. The second stage could then be to select the candidate that is most likely to be classified as non-compliant. Addressing a combination of CLSP and NCP could provide valuable decision support for inspectors and the approach can potentially be easier to adopt for inspection agencies, compared to our previous work where ML is used to create new checklists [15, 16]. Another direction is to develop more accurate ML methods for solving the CLSP and NCP problems, especially since none of the methods in our demonstration experiments achieved outstanding results in terms of accuracy, precision and recall scores. For the same reason, LICD could be relevant for benchmarking ML methods. The dataset could for instance be used in an SDG framework like SustainBench [18], which currently lacks a dataset that addresses the SDG on decent work and economic growth (SDG 8).

D.8 References

[1] Communication from the commission to the European parliament, the council, the European economic and social committee and the committee of the regions. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52014DC0332. Accessed: 2022-08-15.

[2] Patrick Belser. Forced labour and human trafficking: Estimating the profits. *Available at SSRN*, 2005.

[3] David Weil. A strategic approach to labour inspection. *International Labour Review*, 147(4):349–375, 2008.

[4] David Weil. If OSHA is so bad, why is compliance so good? *RAND Journal of Economics*, 27(3):620, 1996.

[5] Øyvind Dahl, Torbjørn Rundmo, and Espen Olsen. The impact of business leaders' formal health and safety training on the establishment of robust occupational safety and health management systems: Three studies based on data from labour inspections. *International Journal of Environmental Research and Public Health*, 19 (3):1269, 2022.

[6] Nektarios Karanikas and Sikder Mohammad Tawhidul Hasan. Occupational health & safety and other worker wellbeing areas: Results from labour inspections in the Bangladesh textile industry. *Safety Science*, 146:105533, 2022.

[7] Srdjan Jelacic, Andrew Bowdle, Bala G. Nair, Kei Togashi, Daniel J. Boorman, Kevin C. Cain, John D. Lang, and E. Patchen Dellinger. Aviation-style computerized surgical safety checklist displayed on a large screen and operated by the anesthesia provider improves checklist performance. *Anesthesia & Analgesia*, 130 (2):382–390, 2020.

[8] Daniel E. Ho, Sam Sherman, and Phil Wyman. Do checklists make a difference? a natural experiment from food safety enforcement. *Journal of Empirical Legal Studies*, 15(2):242–277, 2018.

[9] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, 2020.

[10] Shaily Bhatt, Rahul Jain, Sandipan Dandapat, and Sunayana Sitaram. A case study of efficacy and challenges in practical human-in-loop evaluation of NLP systems using checklist. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 120–130, 2021.

[11] Michael A. Madaio, Luke Stark, Jennifer Wortman Vaughan, and Hanna Wallach. Co-designing checklists to understand organizational challenges and opportunities around fairness in AI. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2020.

[12] David Weil. Improving workplace conditions through strategic enforcement. *Boston U. School of Management Research Paper*, (2010-20), 2010.

[13] Päivi Mattila-Wiro, Yogindra Samant, Wiking Husberg, Magnus Falk, Annemarie Knudsen, and Eyjolfur Saemundsson. Work today and in the future: perspectives on occupational safety and health challenges and opportunities for the nordic labour inspectorates. Technical report, Nordic labour inspectorates, 2020.

[14] Øyvind Dahl and A. Starren. The future role of big data and machine learning in health and safety inspection efficiency. Technical report, European Agency for Safety and Health, 2019.

[15] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Bayesian feature construction for case-based reasoning: Generating good checklists. In *International Conference on Case-Based Reasoning*, pages 94–109. Springer, 2021.

[16] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Creating dynamic checklists via Bayesian case-based reasoning: Towards decent working conditions for all. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5108–5114, 2022. doi: 10.24963/ijcai.2022/709. URL https://doi.org/10.24963/ijcai.2022/709.

[17] Eirik Lund Flogard. Labour Inspection Checklist Dataset. https://doi.org/10.18710/7U6TZP, 2022. URL https://doi.org/10.18710/7U6TZP.

[18] Christopher Yeh, Chenlin Meng, Sherrie Wang, Anne Driscoll, Erik Rozi, Patrick Liu, Jihyeon Lee, Marshall Burke, David B. Lobell, and Stefano Ermon. Sustainbench: Benchmarks for monitoring the sustainable development goals with machine learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[19] Samriddhi Singla, Ayan Mukhopadhyay, Michael Wilbur, Tina Diao, Vinayak Gajjewar, Ahmed Eldawy, Mykel Kochenderfer, Ross Shachter, and Abhishek Dubey. Wildfiredb: An open-source dataset connecting wildfire spread with relevant determinants. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[20] Danish labour inspection authority's smiley dataset. https://at.dk/tilsyn/efter-tilsynsbesoeget/viden-om/roed-gul-groen-og-kronesmiley/smiley-soegning/. Accessed: 2022-08-15.

[21] Alison D. Morantz. Has devolution injured American workers? state and federal enforcement of construction safety. *The Journal of Law, Economics, & Organization*, 25(1):183–210, 2009.

[22] Don J. Lofgren. Results of inspections in health hazard industries in a region of the state of Washington. *Journal of Occupational and Environmental Hygiene*, 5(6): 367–379, 2008.

[23] Matthew S. Johnson, David I. Levine, and Michael W. Toffel. Improving regulatory effectiveness through better targeting: Evidence from OSHA. *Harvard Business School Technology & Operations Mgt. Unit Working Paper*, (20-019), 2020.

[24] Glenn Pransky, Terry Snyder, Allard Dembe, and Jay Himmelstein. Under-reporting of work-related disorders in the workplace: a case study and review of the literature. *Ergonomics*, 42(1):171–182, 1999.

[25] Olivia Milam, Haroon Malik, and Sarah Surber. Digital canaries: Identifying hazardous patterns in MSHA data using a machine learner. *Procedia Computer Science*, 177:227–233, 2020.

[26] Miyuki Hino, Elinor Benami, and Nina Brooks. Machine learning for environmental monitoring. *Nature Sustainability*, 1(10):583–588, 2018.

[27] Suhong Kim, Param Joshi, Parminder Singh Kalsi, and Pooya Taheri. Crime analysis through machine learning. In *Proceedings of the IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 415–420, 2018.

[28] Andrea S.N. Curman, Martin A. Andresen, and Paul J. Brantingham. Crime and place: A longitudinal examination of street segment patterns in Vancouver, bc. *Journal of Quantitative Criminology*, 31(1):127–147, 2015.

[29] Keegan McBride, Gerli Aavik, Tarmo Kalvet, and Robert Krimmer. Co-creating an open government data driven public service: The case of Chicago's food inspection forecasting model. In *Proceedings of the 51st Hawaii International Conference on System Sciences*, pages 2453–2462, 2018.

[30] Vinesh Kannan, Matthew A. Shapiro, and Mustafa Bilgic. Hindsight analysis of the Chicago food inspection forecasting model. *arXiv preprint arXiv:1910.04906*, 2019.

[31] I.A. Walker, S. Reshamwalla, and I.H Wilson. Surgical safety checklists: do they improve outcomes? *British journal of anaesthesia*, 109(1):47–54, 2012.

[32] Amit Vats, C.A. Vincent, Kamal Nagpal, R.W. Davies, Ara Darzi, and Krishna Moorthy. Practical challenges of introducing who surgical checklist: Uk pilot experience. *British Medical Journal (BMJ)*, 340, 2010.

[33] Asaf Degani and Earl L. Wiener. Cockpit checklists: Concepts, design, and use. *Human factors*, 35(2):345–359, 1993.

[34] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. *Advances in Neural Information Processing Systems*, 33:1513–1524, 2020.

[35] Caidan Zhao and Yang Lei. Intra-class cutmix for unbalanced data augmentation. In *13th International Conference on Machine Learning and Computing*, pages 246–251, 2021.

[36] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[37] Chris Drummond and Robert C Holte. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8, 2003.

[38] Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. In *European Conference on Computer Vision*, pages 694–710. Springer, 2020.

[39] Yoon-Joo Park and Alexander Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 11–18, 2008.

[40] Xiangxin Zhu, Dragomir Anguelov, and Deva Ramanan. Capturing long-tail distributions of object subcategories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 915–922, 2014.

[41] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 7032–7042, 2017.

[42] Christie M. Fuller, David P. Biros, and Dursun Delen. Exploration of feature selection and advanced classification models for high-stakes deception detection. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS)*, pages 80–80. IEEE, 2008.

[43] Ole Jakob Mengshoel, Eirik Flogard, Jon Riege, and Tong Yu. Stochastic local search heuristics for efficient feature selection: An experimental study. In *Norsk IKT-konferanse for forskning og utdanning*, pages 58–71, 2021.

[44] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[45] Ole Jakob Mengshoel, Tong Yu, Jon Riege, and Eirik Flogard. Stochastic local search for efficient hybrid feature selection. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 133–134, 2021.

[46] Ole Jakob Mengshoel, Eirik Lund Flogard, Tong Yu, and Jon Riege. Understanding the cost of fitness evaluation for subset selection: Markov chain analysis of stochastic local search. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 251–259, 2022.

[47] Omid Tarkhaneh, Thanh Thi Nguyen, and Samaneh Mazaheri. A novel wrapper-based feature subset selection method using modified binary differential evolution algorithm. *Information Sciences*, 565:278–305, 2021.

Paper E
Creating Explainable Dynamic Checklists via Machine Learning to Ensure Decent Working Environment for All: A Field Study with Labour Inspections

Published in: *Conference on Prestigious Applications of Artificial Intelligence,* 2023.
Norwegian Scientific Index Level (NVI): 1.
Received best paper award.

Notes. The paper has been accepted to the conference, held 30. September 2023 to 5. October 2023, and is receiving the best paper award. The publication venue is one of the leading conferences in applied artificial intelligence, and is held as a part of the European Conference on Artificial Intelligence (ECAI). All submissions to the conference are peer reviewed in accordance with scientific standards required for level 1 in the Norwegian Scientific Index. The paper has been reformatted to ensure readability and consistency with the other papers in this thesis.

Contribution Statement. My contribution includes writing the first draft of the paper, the ideas and development of the explanation methods and the prototype described in the paper, and processing all the reported results. The field study was organised by me and Ole Magnus Theisen (third author), who also helped to write the final version of the paper. The other co-authors of the paper (Ole Jakob Mengshoel and Kerstin Bach) are my supervisors and helped with revising the paper.

Creating Explainable Dynamic Checklists via Machine Learning to Ensure Decent Working Environment for All: A Field Study with Labour Inspections

Eirik Lund Flogard[1,2], **Ole Jakob Mengshoel**[2], **Ole Magnus Theisen**[1]
and **Kerstin Bach**[2]
[1]Norwegian Labour Inspection Authority
[2]Norwegian University of Science and Technology

Abstract

To address poor working conditions and promote United Nations' sustainable development goal 8.8, "protect labour rights and promote safe working environments for all workers [...]", government agencies around the world conduct labour inspections. To carry out these inspections, inspectors traditionally use paper-based checklists as a means to survey individual organisations for working environment violations. Currently, these checklists are created by domain experts, but recent research indicates that machine learning (ML) could be used to generate dynamic checklists to increase inspection efficiency. A drawback with the dynamic checklists is that they are complex and could be difficult to understand for inspectors. They have also never been field-tested. In this paper, we therefore propose user-oriented explanation methods for Context-aware Bayesian Case-Based Reasoning (CBCBR), which is the current state-of-art ML method for generating dynamic checklists. We also introduce a prototype of CBCBR and present a field study where we test it in real-world labour inspections. The results from the study indicate that using the explainable dynamic checklists increases the efficiency of the labour inspections, and inspectors also report that they find the checklists useful. The results also suggest that current ML evaluation methods, where model prediction performance is evaluated on existing data, may not fully reflect the real-world field performance of checklists.

E.1 Introduction

Labour inspections are conducted by government agencies around the world to address poor working conditions and promote United Nations' sustainable development goal (SDG) 8.8 "to protect labour rights and promote safe working environments for all workers". The inspections are carried out in workplaces (organisations) on a large scale to enforce national and international labour laws and standards, pursuant to the International Labour Organization's (ILO) Labour Inspection Convention (1947). Despite the inspection efforts, there are still 1.9 million registered deaths worldwide annually attributable to occupational health and safety risks [1]. Increasing the efficiency of inspections is therefore important. To carry out these inspections, inspectors often use checklists to survey individual organisations for non-compliance to health, safety, and environment (HSE) regulations [2, 3]. A labour inspection checklist consists of a non-fixed-sized subset of K out of N items, where each item has a binary response (non-compliant/compliant) and corresponds to a specific health and safety regulation.

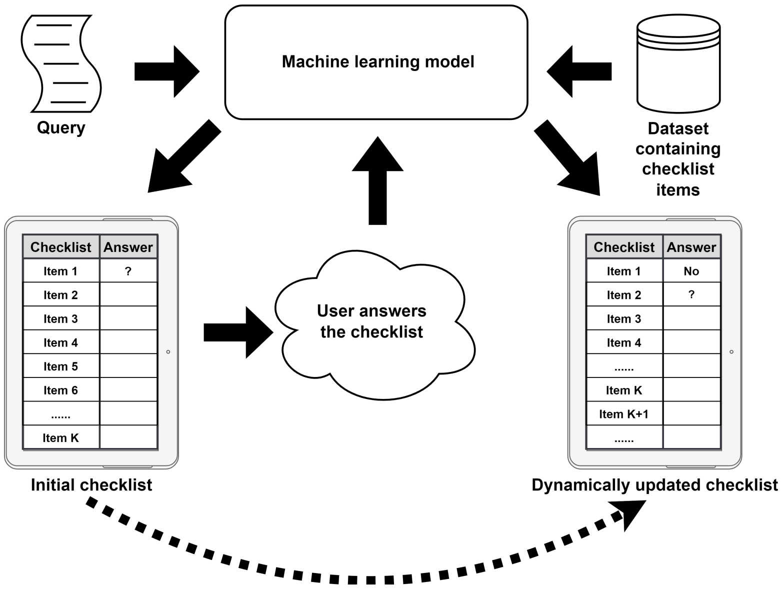


FIGURE E.1. An illustration of how dynamic checklists work. First, the initial checklist to the left is created via a query specified by the user. After the user answers the checklist, it is then dynamically updated by the ML model. The updated checklist is shown on the right side.

Each inspected organisation may be subjected to hundreds of different regulations [4], but a checklist ideally contains between 5 and 30 items. Creating such checklists manually is difficult since checklists are situation-dependent and labour inspections can be executed in many different ways depending on the context [5, 6]. This means that the contents of the checklists need to vary between individual inspected organisations. Traditional static, paper-based checklists are still often too long, making it difficult to differentiate between critical and less important tasks for their users [7].

Instead of relying on traditional checklists, it is possible to use ML to generate dynamic checklists. Figure E.1 shows an overview of a dynamic checklist, where an ML model is given an inspection target (workplace) as input. The model then creates an initial checklist containing a set of K out of N possible items (small K, large N). Based on how the user answers the checklist, it is dynamically updated with additional items to make it more contextually relevant to the situation it is being used in. To our knowledge, the state-of-the-art method for generating dynamic checklists is Context-aware Bayesian Case-Based Reasoning (CBCBR) [8]. CBCBR aims to create checklists that maximize the number of violations found during inspections, by generating and dynamically updating checklists specifically for each inspection target. It is assumed that the dynamic checklists can increase the detection and rectification of working environment violations in the inspections compared to traditional paper-based checklists, thereby increasing efficiency [8]. The main purpose of this paper is to test this assumption by conducting an empirical field study, to determine if ML-based checklists are indeed superior for real-world inspections.

Motivation. It is hard to tell how effective current ML-generated checklists are, since they have never been tested in real-world environments [9, 10]. Flogard et al. propose a cross-validation approach that shows promising results for the dynamic checklists in labour inspections. However, lacking ground truth cases, the approach is essentially a simulation that is mostly based on labels that are generated from existing data [8, 11]. Since the approach relies on existing data, it also does not account for real-world factors that could impact labour inspection performance, such as intervention effects from replacing the current domain expert-designed checklists [12]. This is potentially problematic as inspections are complex tasks, and the success of using checklists could depend on many factors, such as implementation details or how users interact with them [6, 13]. Another problem is that CBCBR currently lacks explanation methods. Dynamic checklists are complex constructs, rendering it difficult for inspectors to understand the dynamic changes to their checklists during inspections. If the dynamic checklists are not understood or justified, they may be difficult to use, undermining any advantages they may have on task performance [6]. Moreover, forthcoming EU regulations will require a certain level of explanation in ML and related technologies [14].

Contributions. To address the problems mentioned above, our scientific contributions in this paper are as follows:

(1) Technical: We propose methods for explaining the content of dynamic checklists to their end-users (inspectors), focusing on justification and transparency as explanation goals [15]. We also developed a prototype based on the state-of-the-art method for generating checklists (CBCBR), implementing the explanation methods.[1]

(2) Social: As far as we know, ML-based checklists remain untested in the field, let alone implemented or adopted. Collaborating with the Norwegian Labour Inspection Authority (NLIA), we conducted a field study, testing the prototype in dynamic real-world environments with seven inspectors carrying out 69 valid inspections across various industries. Both qualitative and quantitative results are presented and compared to inspections conducted by the same inspectors, using ordinary static checklists created by domain experts. The results show that dynamic checklists increase the efficiency and number of violations being addressed in labour inspections. This insight is essential in order to determine whether the adoption of the dynamic checklists is worth the investment, as most labour inspection authorities have limited resources [16].

(3) Analytical: Our analyses of the results from the study show significant discrepancies between field performance and existing cross-validation performance estimates of dynamic checklists. Current ML evaluation practices may therefore be insufficient for estimating field performance of checklists. This insight could have implications for research in other domains where ML or AI is used to create checklists, such as medicine [17–19].

Social Impact. Due to the results from our field study, the Norwegian Labour Inspection Authority (NLIA) plans to adopt the dynamic checklists into their inspections. We believe that our work could inspire labour inspection authorities in other countries to do the same. A widespread adoption of dynamic checklists could increase levels of compliance with working environment laws and labour rights in society and reduce both long and short-term injuries (SDG indicators 8.8.1 and 8.8.2), as inspections become more efficient in addressing and reducing violations in workplaces.

[1]The source code for the prototype can be found at: https://github.com/ntnu-ai-lab/cbcbr-prototype.

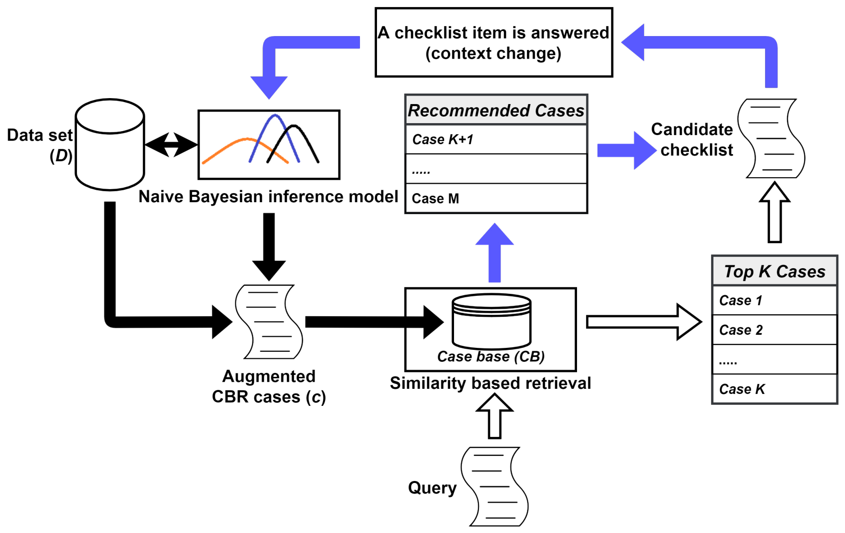


FIGURE E.2. The figure [8] shows an overview of CBCBR. The black arrows show how the case base is created or updated. The white arrows show the creation of a candidate checklist. The checklist is dynamically updated via the blue (and black) arrows, starting from the candidate checklist on the right hand side.

E.2 RELATED WORK

Dynamic Checklists. Digital dynamic checklists have also been proposed to deal with context changes in medical applications, such as emergency care or surgeries. However, these checklists are currently created via rule [20] or process-based models [21] that are not based on ML, requiring manual construction and maintenance that limits the models' complexity and nuance [7, 8]. Nevertheless, some of these have been successfully tested in medical trials. De Bie et al. show that dynamic checklists improved user compliance, compared to traditional paper checklists for intensive care units in a trial [22]. Kulp et al. propose a dynamic digital checklist for trauma resuscitation modeled as an iterative process via user interviews, and test it in a trial [7]. The results are promising, but the checklists' impact on task execution and performance in many medical situations turned out differently than expected, underscoring the complexity of checklists and the importance of analyzing and testing them in the field before adoption. This is also a motivation for our field study, especially considering that ML-based checklists have never been field-tested before.

Machine Learning Methods for Creating Checklists. Research on ML for creating checklists is currently limited. Besides CBCBR and BCBR proposed by Flogard et al. [8, 11], there are other methods for checklist creation [17, 18, 23]. These are unsuitable for creating dynamic checklists or checklists for labour inspections [8]. As far as we know, CBCBR is currently the only readily available ML method for creating dynamic checklists for labour inspections. CBCBR is a hybrid method that uses a Naive Bayesian inference (NBI) model to construct features for cases used in Case-Based Reasoning (CBR) [24]. CBCBR creates a dynamic checklist for a given organisation targeted for an inspection by retrieving and reusing past cases with checklist items used in similar organisations, that also have high estimated probabilities for non-compliance. CBCBR, see Figure E.2, operates in two phases. (1) In the first phase, an initial checklist is created. A naive Bayesian inference (NBI) model is used to generate probability estimates for non-compliance (θ^{be}), based on empirical distributions from the dataset $\mathcal{D}$. The probability estimates are added to dataset instances $\mathbf{d}_j \in \mathcal{D}$ as new features, to create new augmented CBR cases $\mathbf{c}_j$ for a case base $\mathcal{CB}$. Similarity based retrieval is then used to create an initial candidate checklist by retrieving K CBR cases with unique items, using a query $\mathbf{q} = (x^{cnd}, \theta, \kappa)$ that contains feature values for the target organisation (x^{cnd}), a fixed target value (θ) for the probability estimate embedded in each of the CBR cases and a target value (κ) for the number of observed instances that are used to calculate the probability estimate. (2) The second phase consists of dynamic updates to the candidate checklist via the case-base. After the user answers a checklist item, the NBI model updates the CBR cases $\mathbf{c}_j \in \mathcal{CB}$ with new posterior probability estimates for non-compliance. CBCBR then retrieves any additional cases that have sufficiently increased estimates, which are appended to the checklist as a dynamic update. Depending on the setup, this phase is repeated after a certain number of checklist items are answered. A complete formal definition of dynamic checklists and a more detailed description and analysis of the CBCBR framework is given by Flogard et al. [8].

Explanations for Dynamic Checklists. As far as we know, no one has proposed any method that offers user-oriented explanations of dynamic checklists. However, CBCBR is a good starting point for new explanation approaches since it is based on two transparent methods: CBR and parameter estimates from empirical distributions (NBI) [24, 25]. There are many examples of CBR systems being used to provide explanations, often as post-hoc or in twin configurations with black box systems [26, 27]. Many methods based on model agnostic approaches also exist, such as LIME or SHAP [28, 29]. However, most of the current explanation methods address other explanation goals than ours and are not good starting points for explaining the content of dynamic checklists, and are therefore not considered within the scope of our work. Explanations should generally be goal-oriented and serve a specific target audience [15, 29]. Thus, we propose approaches for providing user-oriented explanations with justification and transparency goals in mind, both for initially created checklists (before inspection starts) and for any dynamic updates to the checklists that are made during inspections.

E.3 Explanation Methods for Dynamic Checklists

To reach the explanation goals mentioned earlier, we propose two approaches. Both approaches are based on showing traces of model logic to the users [15].

Showing Estimated Probabilities for Non-Compliance. The first approach is to show CBCBR's estimated probability of finding non-compliance on each checklist item to the users. The probability estimates are calculated via the NBI model that was proposed by Flogard et al. [8], but these have not been used for explanations in previous work. As each item on a checklist has its own estimate that depends on the inspection target, the purpose of the explanations is to provide prediction transparency and justify the use of the items. Since the estimates are based on sufficient statistics using empirical distributions [25], they should in theory reflect the probabilities observed in the real world if unbiased data is used and all prior parameters are known. This property should ensure that the estimates are as consistent as possible with the real world, within the limitation of the dataset being used, which is necessary to promote long-term trust [30]. Probability estimates are also an intuitive way to communicate uncertainty [25].

Showing the Most Important Answer for a Dynamic Update. The purpose of this second approach is to make the users aware of why additional items are dynamically added to their checklist (justification), and how these are related to the answered part of the checklist (transparency), which could promote trustworthiness [31, 32]. A formula for finding the most important answer is derived as follows: Let x be a target organisation for an ongoing inspection and y^{cnd} be a candidate checklist that a user interacts with during the inspection. Let's assume that an item $\hat{e} \notin \mathrm{y}^{cnd}$ is considered as a candidate to be dynamically added to the checklist. Let $(e_i, l_i) \in \mathrm{y}^{cnd}$ be pairs of existing items and given answers in the checklist, respectively, with the position in y^{cnd} indexed by i. The probability for finding non-compliance ($L = 1$) for any candidate item $\hat{e}$, given x and every pair (e_i, l_i), can be estimated via [8]:

$$\theta^{be}(L = 1|x, \hat{e}, \mathrm{y}^{cnd}) = \frac{\beta_{L=1|x,\hat{e}} + \sum_{(l_i, e_i \in \mathrm{y}^{cnd})} p(1, x, \hat{e}, e_i, l_i)}{\sum_{\hat{l}=0}^{1} \beta_{L=\hat{l}|x,\hat{e}} + \sum_{(l_i, e_i \in \mathrm{y}^{cnd})} p(\hat{l}, x, \hat{e}, e_i, l_i)}, \tag{E.1}$$

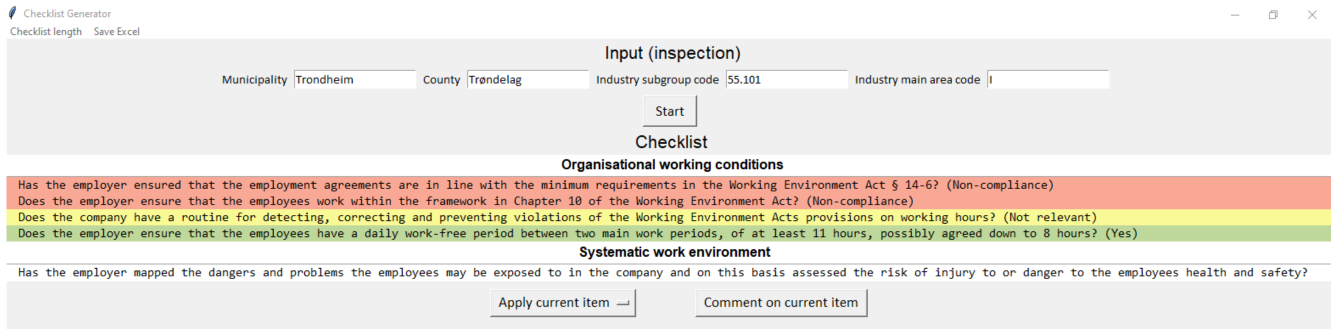
which CBCBR relies on to dynamically update the checklists and is the mean of a posterior beta distribution (see Flogard et al. [8] for more details). Given this information, we seek to find the index of the pair (e_i, l_i) that has the most impact on an $\hat{e}$ being selected for a dynamic update to the checklist. The index i can be found by altering Equation E.1 to depend on only single pairs (e_i, l_i) as follows:

$$\arg\max_i \theta^{be}(L = 1|x, \hat{e}, e_i, l_i) = \arg\max_i \frac{\beta_{L=1|x,\hat{e}} + p(1, x, \hat{e}, e_i, l_i)}{\sum_{\hat{l}=0}^{1} \beta_{L=\hat{l}|x,\hat{e}} + p(\hat{l}, x, \hat{e}, e_i, l_i)}. \tag{E.2}$$

The right hand side of Equation E.2 can be reduced to $\arg\max_i \frac{p(1,x,\hat{e},e_i,l_i)}{\sum_{\hat{l}=0}^{1} p(\hat{l},x,\hat{e},e_i,l_i)}$. In some cases $\arg\max_i \theta^{be}$ may have multiple solutions. In that case, we select one of them randomly. We compute $\arg\max_i \theta^{be}$ via sequential search in the checklist y^{cnd}, when a dynamic update takes place. This runs quite fast as checklists are relatively short and because we calculate the parameters p and store them in tables immediately each time an item on the checklist is answered, to reduce computational costs. After finding i, an explanation text for the item $\hat{e}$ in the dynamic update is generated. A demonstration of the text is presented in Section E.4.

E.4 Implementation of Dynamic Checklists

For the field study, we have created a prototype based on the CBCBR framework introduced by Flogard et al. [8]. The prototype is developed based on the Minimum Viable Product (MVP) scheme, which is a lean and cost-effective way to confirm or refute hypotheses about a product's benefits or values [33]. The prototype is also designed according to the Human-AI interaction guidelines proposed by Amershi et al. [32]. The details regarding the interface, functionality, and configuration of the prototype are described below. We also demonstrate a comparison between a dynamic and traditional checklist.



Input (inspection)

Municipality Trondheim County Trøndelag Industry subgroup code 55.101 Industry main area code I

Checklist

Organisational working conditions

```
Has the employer ensured that the employment agreements are in line with the minimum requirements in the Working Environment Act § 14-6? (Non-compliance)
Does the employer ensure that the employees work within the framework in Chapter 10 of the Working Environment Act? (Non-compliance)
Does the company have a routine for detecting, correcting and preventing violations of the Working Environment Acts provisions on working hours? (Not relevant)
Does the employer ensure that the employees have a daily work-free period between two main work periods, of at least 11 hours, possibly agreed down to 8 hours? (Yes)
```

Systematic work environment

```
Has the employer mapped the dangers and problems the employees may be exposed to in the company and on this basis assessed the risk of injury to or danger to the employees health and safety?
```

FIGURE E.3. A screenshot of the CBCBR prototype with a short checklist of $K = 5$ items, generated for a labour inspection of a hotel in Trondheim, Norway. As a demonstration, we answered four of the items, marked in red, yellow and green. The colors are used to highlight the answers so that they are easy to recognize.

E.4.1 Prototype Interface and Functionality

A screenshot of a short checklist generated with the prototype is shown in Figure E.3. The graphical interface of the software consists of three parts: the input fields, the checklist and pop-up windows with dynamic updates to the checklist. The functionality of these are described below.

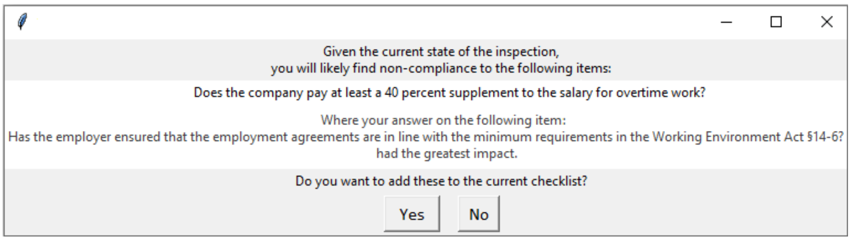
Input Fields. The input fields, shown at the top in Figure E.3, are used to specify the features of the organisation that is targeted for the inspection. The features describe the location (municipality) and industry[2] of the target organisation. The fields are used to build the query in Figure E.2. The query is executed once the user (inspector) presses the start button. After the user presses the start button, an initial checklist that matches the query appears below the input fields. Different input values to the prototype can yield very different checklists.

Checklist. The checklist-part of Figure E.3 corresponds to the candidate checklist in Figure E.2. The checklist is generated for a hotel (ISC 55.101) located in Trondheim municipality, with an initial length of $K = 5$ items. It is possible to adjust the initial checklist length in the upper left corner. To enhance readability for longer checklists, items are grouped under headlines according to the main working environment factor each belongs to. The groups and items within are also initially sorted alphabetically, but users can easily reorder them. The checklist can be saved and opened in Excel format (Save Excel button), where the estimated probability for finding non-compliance is listed for each item to serve as an explanation. We decided not to list the estimates (explanations) on the main GUI to avoid cluttering. As mentioned in Section E.3, the purpose of the estimates is to provide ML model transparency and justification regarding the selected items.

The checklist is answered chronologically from top-to-bottom, as shown by the partially filled-out checklist in Figure E.3. The user can select answers from a drop-down menu by clicking on the "apply current item" button. The options are "non-compliance" (red color), "yes" (green), "not relevant" (yellow), "not controlled" (yellow), "follow up later" (yellow), and "regulation already checked" (yellow). A yes-answer means that the regulation for the corresponding item is compliant; "not relevant" is used for items that do not relate to the target organisation's operations; and "not controlled" or "follow up later" are used if the inspector does not have time or lacks the knowledge/information to follow up the item immediately during the inspection. Finally, "regulation already checked" means that the checklist contains another item that corresponds to the same regulation. Since CBCBR relies on binary target labels for training and prediction, we have designated the "non-compliance" answer as 1 (positive) and the rest of the answers as 0 (negative). The logic is that 1 means that non-compliance is found and 0 means that non-compliance is not found. The resulting binary values, mapped from the answers, are also used to update CBCBR and provide dynamic updates with additional items for the checklist.

Dynamic Updates to the Checklist. Dynamic updates to the checklists are implemented as dynamic recommendations for additional items to use during the inspections. Figure E.4 demonstrates a recommendation of an additional item, based on all the answers from Figure E.3. The recommended item is appended to the bottom of the

[2]An overview of Norwegian industry codes can be found at: https://www.ssb.no/en/klass/klassifikasjoner/6


Given the current state of the inspection,
you will likely find non-compliance to the following items:

Does the company pay at least a 40 percent supplement to the salary for overtime work?

Where your answer on the following item:
Has the employer ensured that the employment agreements are in line with the minimum requirements in the Working Environment Act §14-6?
had the greatest impact.

Do you want to add these to the current checklist?

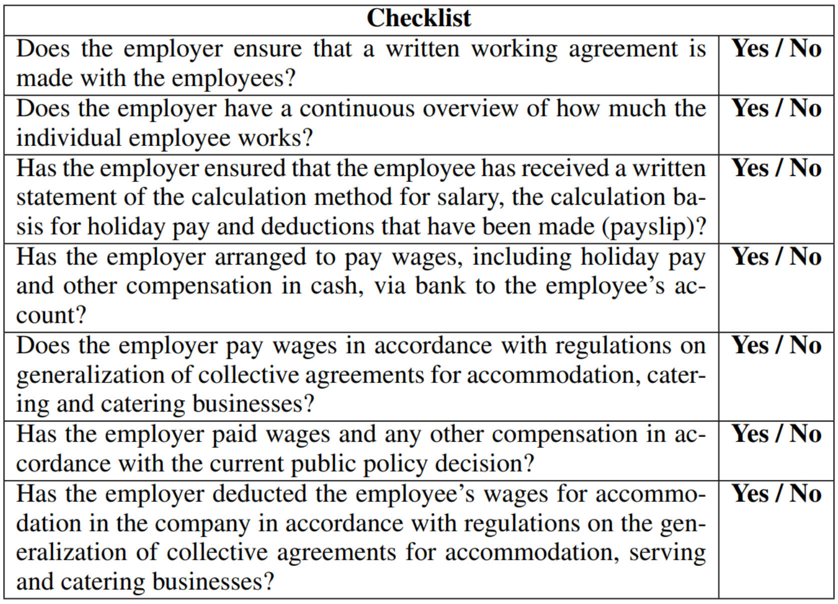
FIGURE E.4. A pop-up with recommendation of an additional item and an explanation, based on the answers from the checklist in Figure E.3.

checklist if the user presses the "yes" button in the dialogue, and it is answered in the same manner as ordinary checklist items. In our implementation, dynamic updates are attempted each time all the checklist items grouped under a headline have been answered. This is the case in Figure E.3, where all items under "organisational working conditions" have been answered. It is also worth noting that sometimes no recommendations are made if there are no eligible items outside the checklist that have received sufficiently increased posterior probability estimates to appear in a recommendation. Below the recommended item in Figure E.4, an explanation is also provided. The explanation shows the answer on the checklist (Figure E.3) that had the most impact on the recommendation of the item, as described in Section E.3. For the purpose of the field study, we have encouraged inspectors to accept any dynamic items that are recommended for the checklists. We have chosen not to dynamically remove checklist items, or forcibly append new items to the checklists in the software without user approval as this could strain or confuse users [8].

Using the Prototype in Inspections. We intentionally designed the prototype to operate in a wide range of labour inspections. This is important as the execution of labour inspections is contextual and varies [34, 35]. Inspections can be based on conversations in an office, or perception-based where inspectors walk around and inspect working areas or equipment. Before an inspection starts, an initial checklist is created with the prototype. As the inspection progresses, the checklist (including any dynamic items) is filled out as described above. After the inspection is completed, the inspector saves the checklist to an Excel spreadsheet and then manually uploads it into the case management system where a draft for an inspection report is automatically generated.

E.4.2 Comparison of Dynamic vs. Traditional Checklist

An example of a traditional checklist is presented in Figure E.5, to highlight the difference from the dynamic checklist in Figure E.3. Figure E.5 shows one of many different checklists that are typically used for an inspection at a hotel (ISC 55.101). The checklist contains 7 items that focus only on working agreements, working hours, and wage requirements. The dynamic checklist in Figure E.3 is generated for ISC 55.101 and has 5 items, plus the dynamic recommended item in Figure E.4. It covers working agreements, working hours, systematic HSE risk assessments, and overtime payments. The content of the dynamic checklist is broader and more relevant to the inspected



Checklist	
Does the employer ensure that a written working agreement is made with the employees?	**Yes / No**
Does the employer have a continuous overview of how much the individual employee works?	**Yes / No**
Has the employer ensured that the employee has received a written statement of the calculation method for salary, the calculation basis for holiday pay and deductions that have been made (payslip)?	**Yes / No**
Has the employer arranged to pay wages, including holiday pay and other compensation in cash, via bank to the employee's account?	**Yes / No**
Does the employer pay wages in accordance with regulations on generalization of collective agreements for accommodation, catering and catering businesses?	**Yes / No**
Has the employer paid wages and any other compensation in accordance with the current public policy decision?	**Yes / No**
Has the employer deducted the employee's wages for accommodation in the company in accordance with regulations on the generalization of collective agreements for accommodation, serving and catering businesses?	**Yes / No**

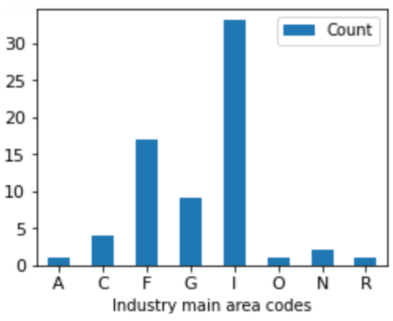
FIGURE E.5. A simplified version of a traditional checklist currently used by NLIA for inspections in hotels and restaurants. A yes-answer on the checklist means that the inspected organisation is compliant with the regulation in question, while no means that it is non-compliant.

organisation and addresses more specific violation risks than the traditional checklist. More specifically, both checklists cover working agreements and salaries but the dynamic checklist also focuses on routines to minimize risk of injuries and ensure decent working hours.

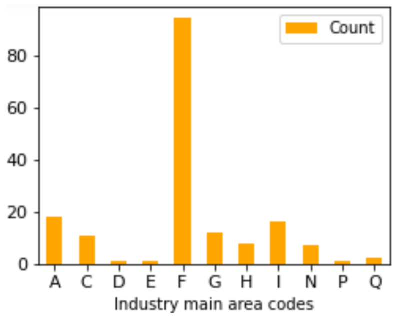
It should be noted that the relevant content of a checklist varies depending on the inspected organisation, as the applicable HSE risks and regulations also vary between organisations. This is a challenge with the current traditional checklists, as hundreds of predefined checklists need to be maintained (NLIA has 369) [4]. Selecting the correct traditional checklist for each specific inspection can therefore be difficult. Such variations are not a problem for dynamic checklists, since they can be created on-demand specifically for each inspected organisation.

E.4.3 Prototype Configurations and Setup

We are using the same configurations as Flogard et al. proposed for their example demonstration of CBCBR, where the components of the query q are $\theta = 100\%$, $\kappa = 70$. The NBI model is implemented via MSSQL2019 and uses the same fixed prior parameter values as Flogard et al. [8]. CBCBR's similarity-based retrieval is implemented via myCBR [36]. To generate the dynamic checklists, the prototype relies on the dataset that Flogard et al. introduced for generating labour inspection checklists [11]. The dataset contains $N = 1967$ different unique checklist items from checklists used in



(a) Test group distribution of inspections using dynamic checklists.



(b) Control group distribution of inspections using paper-based checklists.

Figure E.6. The diagrams show the number of inspections conducted within each industry. For the test group, most of the inspections were conducted within the Accommodation & Food (I) and Construction industries (F). The majority of the inspections of the control group were carried out in Construction businesses (F).

59989 past labour inspections. Another related dataset exists [4], but it is not suitable for creating new checklists from scratch. We are using the municipality/county and industry code hierarchy from the dataset as features to represent organisations (x) [8, 11]. The GUI of the prototype is implemented via TKinter and installed on Microsoft Surface Pro tablets. CBCBR also runs locally on each tablet. Response times for generating an initial checklist and for dynamic updates are circa 10 and 5 seconds, respectively.

E.5 Field Study with Dynamic Checklists

In this section, we present the results from testing our implementation of explainable dynamic checklists in labour inspections. The purpose of the field study is to test the assumption that the checklists increase labour inspection efficiency and the number of violations found by inspectors.

E.5.1 *Method and Design.*

Design of a Test and Control Group. For the study, seven of NLIA's inspectors volunteered to participate. The field study is conducted as a paired test where the same seven inspectors participate in both a test group and a control group. The test group consists of 69 inspections conducted between March 1. 2022 and October 1. 2022, using the CBCBR prototype. The control group consists of 171 ordinary inspections that the same inspectors conducted within the same period using NLIA's standard paper-based checklists, without any interventions from us. Figure E.6a shows how the 69 inspections in the test group are distributed. Most of the efforts are concentrated on Accommodation & Food (I, with 33 inspections), Construction (F, with 17 inspections), and Wholesale & Retail (G, with 9 inspections).[2] Figure E.6b shows a different distribution for the control group, where most of the inspections are conducted within the Construction industry. For both the test and control groups, the inspections in Accommodation & Food (F) were carried out by four of the seven inspectors. The inspections in Construction were distinctly carried out by further two of the seven of inspectors. The inspections in Wholesale & Retail and the remaining industries were carried out by the one remaining inspector, in addition to three of the inspectors from Accommodation & Food. The inspections were divided in this manner to avoid disrupting NLIA's inspection efforts and allow inspectors to freely select their targets (discussed in the ethical statement), as inspectors tend to be specialists and therefore carry out most of their inspections within a few industries that are familiar to them. We reviewed other factors such as the size (number of employees) and location of the inspected organisations and found no significant differences between the test and control groups.

Measuring Results. Because labour inspections are industry oriented and due to the differences between the distributions of inspections in the test and control group in Figure E.6, we report the results from the study by the following categories to address bias: Accommodation & Food (I), Construction (F), Others and All inspections. This is important as there are substantial differences in how checklists are used and inspections are carried out between Construction and Accommodation & Food. There are relatively few inspections in Wholesale & Retail (G) and the other industries in our study. We therefore grouped these industries into the category named Others, as the inspection results in these industries are similar. The results are reported for each category in terms of average relative frequency of checklist answers per inspection, average number of discovered violations (Avg_v), and average length (Avg_l) of the checklists used. We also use two different average precision scores [8, 11]. These are calculated from a set of N completed inspections as follows: $Prec_v = \frac{1}{N}\sum_{i=1}^{N}\frac{v_i}{|y_i|}$ and $Prec_r = \frac{1}{N}\sum_{i=1}^{N}\frac{r_i}{|y_i|}$. $|y_i|$ is the number of items in each completed checklist y_i (predicted positives), $v_i \in \mathbb{R}_{\geq 0}$ and $r_i \in \mathbb{R}_{\geq 0}$ is the number of violations and the number of reactions in the i-th inspection (true positives), respectively. Each checklist can at most have one violation/reaction per checklist item, so that $v_i \leq |y_i|$ and $r_i \leq |y_i|$ always holds. In words, $Prec_v$ can be explained as the average number of violations per checklist item and $Prec_r$ as the average number of reactions per checklist item. We also use an additional statistic $D\,Prec_v$, which is $Prec_v$ calculated exclusively on the dynamically added part of the checklists. Ideally, it would be beneficial to have more statistics such as recall or accuracy in the study. However, the ground truth (negatives) needed to calculate these is not feasible to obtain [11]. To compare the overall results from the study with current cross-validation scores, we use an industry-weighted average precision score from all inspections in the study to remove bias (see Fig. E.6): $Weighted\,Precision = \sum_j w_j\,Prec_v(j)$, where $Prec_v(j) = \frac{1}{N_j}\sum_{i=1}^{N_j}\frac{v_i}{|y_i|}$ is calculated on the set of all (N_j) completed inspections within each industry j (see $Prec_v$). The weights w satisfy $\sum_j w_j = 1$ and each w_j is calculated using Flogard et al.'s dataset [11] $\mathcal{D}$ as follows: $w_j = \frac{S_j}{S_{\mathcal{D}}}$, where S_j is the number of inspections in $\mathcal{D}$ within industry j and $S_{\mathcal{D}}$ is the total number of inspections in $\mathcal{D}$.

Test Group - Dynamic Checklists				
	Acc&Food	Construction	Others	All
Avg_v	9.03 ± 0.52	2.94 ± 0.76	3.53 ± 0.59	6.02 ± 0.54
Avg_I	17.0 ± 0.49	17.5 ± 0.49	16.2 ± 0.58	17.0 ± 0.31
$Prec_v$	0.53 ± 0.02	0.17 ± 0.02	0.22 ± 0.02	0.35 ± 0.01
$Prec_r$	0.37 ± 0.02	0.11 ± 0.02	0.18 ± 0.02	0.25 ± 0.01
D $Prec_v$	0.71 ± 0.10	0.24 ± 0.09	0.12 ± 0.06	0.49 ± 0.08
Control Group - Traditional Checklists				
	Acc&Food	Construction	Others	All
Avg_v	7.50 ± 0.86	2.88 ± 0.26	2.78 ± 0.38	3.70 ± 0.26
Avg_I	17.6 ± 1.05	22.1 ± 0.78	19.1 ± 0.86	20.6 ± 0.56
$Prec_v$	0.42 ± 0.03	0.14 ± 0.01	0.15 ± 0.01	0.15 ± 0.01
$Prec_r$	0.30 ± 0.03	0.09 ± 0.01	0.10 ± 0.01	0.12 ± 0.01

TABLE E.1. Quantitative results from the test and control group of inspections conducted in the study.

E.5.2 Qualitative Results and Discussions.

We conducted both conversational and structured qualitative interviews with the inspectors in the study, which are summarized due to space restrictions: Overall, the CBCBR prototype is mostly well-received. Most of the inspectors reported an increase in the number of significant working environment violations they found in the inspections when using dynamic checklists, but they also had to spend a bit more time on case management afterward to follow up on the extra violations. The inspectors also perceive dynamic checklists as more relevant to the target organisations, in comparison to the existing paper-based checklists. This is because the content of each checklist is tailored to match the working environment risks in each organisation. The inspectors also reported that they found violations on items that they normally would not think of, especially among the dynamically added items. They found the explanatory probability estimates (explanation method 1) helpful to understand the model's confidence in finding violations to items on the checklists. The explanations for the dynamic checklist updates (explanation method 2) were also useful, both for understanding how and why they should be used. On the negative side, the inspectors reported that dynamic checklists are more difficult to memorize than their paper-based counterparts due to their uniqueness. The checklists also require more attention from the inspectors when operated due to the dynamic updates. The prototype's GUI also needs some improvements.

E.5.3 Quantitative Results and Discussions.

Overall Results from the Field Study. The results in Table E.1 show significant increases in the average number of violations found per inspection (Avg_v) between the test and control groups. For all inspections, the increase is 62.7% (6.02 vs 3.70), but some of the difference can be explained by the fact that the test group contains more

	Acc&Food	Construction	Others	All
Non-compliance	0.53	0.17	0.22	0.35
Yes	0.42	0.52	0.59	0.49
Not controlled	0	0.06	0.04	0.03
Not relevant	0.04	0.25	0.15	0.13
Follow up later	0.01	0	0	0
Total	1.00	1.00	1.00	1.00

Table E.2. Average relative frequency distribution of checklist answers per inspection from the test group in the field study.

inspections conducted in Accommodation & Food than the control group (see Figure E.6), where inspectors find more violations than any other industries. For Accommodation & Food and Others, the increases are 20.5% (9.03 vs 7.50) and 27% (3.53 vs 2.78) respectively, which are significant and likely attributed to the dynamic checklists. It seems that the dynamic checklists are less effective in Construction, as the increase in Avg_v is insignificant. However, the average precision per checklist item for violations ($Prec_v$) for Construction is still significantly higher in the test group, as the average checklist size (Avg_l) is lower. $Prec_v$ is also significantly higher in Accommodation & Food and in Others as well, as more violations are found. The benefits of finding more violations are discussed in Section E.1, and shorter checklists may also decrease time spent on the inspections and the cognitive load for the inspectors [22]. Thus, the overall increases in $Prec_v$ in the test group can therefore be seen as indicators for increased labour inspection efficiency [8, 11]. The results in Table E.1 also support Flogard et al.'s claim that dynamic updates to the checklists increase overall precision [8]. $D\ Prec_v$ is the precision score exclusively for the dynamic part of checklists and the overall score is 49% (0.49), which is significantly higher than the $Prec_v$ score of 35% for the full dynamic checklists in the test group. Without the checklist updates, the overall $Prec_v$ would have been 33%. The effectiveness of the dynamic items varies among the industries, and they seem to be less effective in the Others category (test group) as $D\ Prec_v$ is 9 percentage points less than $Prec_v$.

Distribution of Checklist Answers. Table E.1 also shows variations in the scores between different industries, which are similar for both the test and control groups. Therefore, we look more closely into how the checklists are used in the different industries for the test group only. Table E.2 shows the distribution of checklist answers in the test group. There are clear differences in how inspectors interact with the checklists, based on the industries. For all industries combined, 35% of the checklist items are non-compliant, 49% are compliant, and the rest are either not controlled or not relevant. The Construction industry has the highest share of checklist items marked as "not relevant" and "not controlled" (31%), which is interesting as Construction also has the lowest $Prec_v$ scores in Table E.1 for both the test and control groups. In contrast, the shares of these answers in the Others and Accommodation & Food categories are only 19% and 4% respectively. Therefore, it seems that inspectors generally find checklists less relevant for inspections in Construction, compared to other industries. There are also considerably less yes-answers and more non-compliance-answers in Accommodation & Food compared to the Others category, which indicates lower compliance with overall working environment regulations for this industry. However,

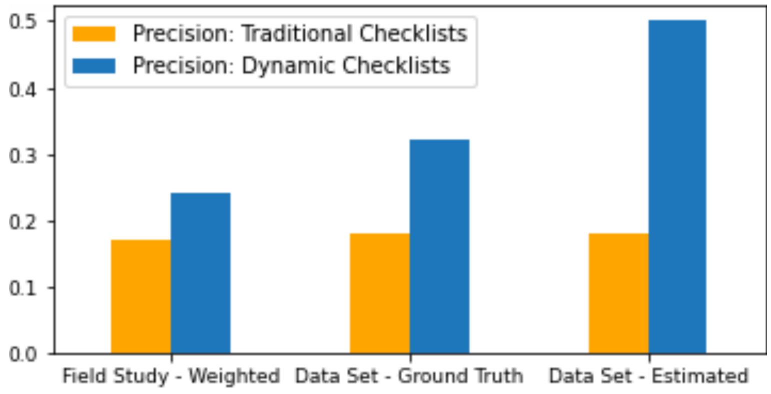


FIGURE E.7. Weighted average precision scores from this field study (left) versus ground truth precision (middle) and estimated precision scores (right) from cross-validations done in previous work [8]. The standard error is 0.01 for the field study and 0 for the rest.

some of these observed differences between the industries may not be industry-specific but could be caused by individual differences between inspectors of the different industries. This is because inspection efforts in each industry are distinctly divided between the 7 inspectors participating in the study, pair-wise for both test and control groups as mentioned earlier.

Field Study versus Cross-Validation Performance. Figure E.7 shows the average precision performance scores for dynamic vs. traditional checklists for different experimental setups. The right plots show $Prec_v$ scores based on estimates, using empirical distributions from the validation parts of the dataset [8]. The middle plots show cross-validation scores using only available ground truth labels [8]. The left plots show the overall $Prec_v$ scores from this field study, which are weighted according to how inspections are distributed among industries in Flogard et al.'s dataset, for comparisons with the other plots. For the baseline traditional checklists (orange), all the scores are nearly identical with 0.17 for the field study and 0.18 for the data sets. For the dynamic checklists (blue), the weighted score from the field study (0.24) is much lower than the cross-validation scores of 0.32 and 0.50. This indicates that the cross-validations are unable to accurately estimate the field performance of dynamic checklists. The discrepancies might be attributed to confounding factors related to the use of checklists in the real world that are not accounted for in the cross-validation, such as how users interact with their checklists [37]. Cross-validation may still not be completely unreliable, as Figure E.7 shows that dynamic checklists consistently outperform traditional checklists in both the cross-validations and field study. Yet, the differences between the field study and cross-validation results highlight the importance of field-testing ML methods.

E.6 Conclusion

In this paper, we developed a prototype based on the current state-of-the-art ML method for generating dynamic checklists (CBCBR). We also propose two different approaches for explaining the content of the checklists to inspectors, which are implemented into the prototype. The prototype is tested in a field study of real-world labour inspections. The results indicate that the *efficiency of labour inspections significantly increases with explainable dynamic checklists*. The way checklists are answered varies based on the industry where the inspections are carried out. Some of these variations could be caused by individual differences in inspection practices between inspectors. Our findings also suggest that current cross-validation methods [8] do not accurately reflect real-world performances of ML-based checklists. Field testing is therefore essential for obtaining fully reliable estimates of checklist performance.

Research on using ML for generating checklists is in an early stage. Despite this, we believe that the results from the field study are strong enough to encourage labour inspection authorities to adopt and further develop ML methods for creating checklists, which could increase national and global levels of compliance with labour rights and reduce injuries (SDG indicators 8.8.1 and 8.8.2). NLIA has already plans to further develop our prototype into a system that can be used nationwide in Norway, replacing the 369 different traditional checklists currently being used [4]. To accomplish this, improving the user interface of the prototype is important. Based on the results from the study and the feedback from the inspectors, it is likely that doing so could further increase inspection performance. The dataset used for the prototype does not have many features, so adding more features and using feature selection could be an option to optimize performance [4, 38–40]. Another direction for future work is to take advantage of the transparency of the CBCBR method, and develop more explanation methods to promote and increase the inspectors' trust in the system. In particular, inspectors have requested methods that provide counterfactual explanations for why certain items have not made it into their checklists.

Ethical Statement

We seek to avoid conducting research in ways that can have negative impacts. An ethical concern for this field study is that the time the inspectors spend on inspections for the study could be spent on something else. Thus, we designed the study to avoid disrupting inspectors from their daily tasks or degrading the quality of the inspections. Some of the design choices may therefore not be optimal from a scientific point of view, such as letting the inspectors select organisations for inspections based on their own decisions (in both test and control groups) or that appending extra dynamic items to the checklists is not done automatically without approval from the user. Privacy for the participating inspectors is also a possible concern, and we have therefore collected an informed consent from the participating inspectors for the purpose of this paper. We have also taken care to not provide any results or information in this paper that can be used to identify any businesses subjected to the labour inspections.

E.7 References

[1] World Health Organization. Joint estimates of the work-related burden of disease and injury, 2000-2016: Global monitoring report. Technical report, 2021.

[2] Øyvind Dahl and Marius Søberg. Labour inspection and its impact on enterprises' compliance with safety regulations. *Safety Science Monitor*, 17(2):1–12, 2013.

[3] Nektarios Karanikas and Sikder Mohammad Tawhidul Hasan. Occupational health & safety and other worker wellbeing areas: Results from labour inspections in the bangladesh textile industry. *Safety Science*, 146:105533, 2022.

[4] Eirik Lund Flogard and Ole Jakob Mengshoel. A dataset for efforts towards achieving the sustainable development goal of safe working environments. In *Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS)*, 2022. URL https://openreview.net/forum?id=93cqcWFpTex.

[5] Daniel E Ho, Sam Sherman, and Phil Wyman. Do checklists make a difference? a natural experiment from food safety enforcement. *Journal of Empirical Legal Studies*, 15(2):242–277, 2018.

[6] Ken Catchpole and Stephanie Russ. The problem with checklists. *BMJ quality & safety*, 24(9):545–549, 2015.

[7] Leah Kulp, Aleksandra Sarcevic, Megan Cheng, and Randall S Burd. Towards dynamic checklists: Understanding contexts of use and deriving requirements for context-driven adaptation. *ACM TOCHI*, 28(2):1–33, 2021.

[8] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Creating dynamic checklists via bayesian case-based reasoning: Towards decent working conditions for all. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5108–5114, 2022. doi: 10.24963/ijcai.2022/709. URL https://doi.org/10.24963/ijcai.2022/709.

[9] Andi Peng, Besmira Nushi, Emre Kiciman, Kori Inkpen, and Ece Kamar. Investigations of performance and bias in human-ai teamwork in hiring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12089–12097, 2022.

[10] Aditya Mate, Lovish Madaan, Aparna Taneja, Neha Madhiwalla, Shresth Verma, Gargi Singh, Aparna Hegde, Pradeep Varakantham, and Milind Tambe. Field study in deploying restless multi-armed bandits: Assisting non-profits in improving maternal and child health. In *AAAI Conference on Artificial Intelligence*, pages 12017–12025, 2022.

[11] Eirik Lund Flogard, Ole Jakob Mengshoel, and Kerstin Bach. Bayesian feature construction for case-based reasoning: Generating good checklists. In *International Conference on Case-Based Reasoning*, pages 94–109. Springer, 2021.

[12] Judea Pearl. The seven tools of causal inference, with reflections on machine learning. *Communications of the ACM*, 62(3):54–60, 2019.

[13] Baptiste Vasey, Myura Nagendran, Bruce Campbell, David A Clifton, Gary S Collins, Spiros Denaxas, Alastair K Denniston, Livia Faes, Bart Geerts, Mudathir Ibrahim, et al. Reporting guideline for the early stage clinical evaluation of decision support systems driven by artificial intelligence: Decide-ai. *bmj*, 377, 2022.

[14] European Commission. Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts, 2021. URL https://digital-strategy.ec.europa.eu/en/library/proposal-regulation-laying-down-harmonised-rules-artificial-intelligence.

[15] Frode Sørmo, Jörg Cassens, and Agnar Aamodt. Explanation in case-based reasoning–perspectives and goals. *Artificial Intelligence Review*, 24(2):109–143, 2005.

[16] David Walters. Labour inspection and health and safety in the eu. *The European Trade Union Institute's (ETUI) Health and Safety at Work Magazine,(14)*, pages 12–17, 2016.

[17] Haoran Zhang, Quaid Morris, Berk Ustun, and Marzyeh Ghassemi. Learning optimal predictive checklists. *NeurIPS*, 34, 2021.

[18] Yukti Makhija, Edward De Brouwer, and Rahul G Krishnan. Learning predictive checklists from continuous medical data. *arXiv preprint arXiv:2211.07076*, 2022.

[19] Qixuan Jin, Haoran Zhang, Thomas Hartvigsen, and Marzyeh Ghassemi. Fair multimodal checklists for interpretable clinical time series prediction. In *NeurIPS 2022 Workshop on Learning from Time Series for Health*, 2022.

[20] AJR De Bie, S Nan, LRE Vermeulen, PME Van Gorp, RA Bouwman, AJGH Bindels, and HHM Korsten. Intelligent dynamic clinical checklists improved checklist compliance in the intensive care unit. *BJA: British Journal of Anaesthesia*, 119(2): 231–238, 2017.

[21] Stefan C Christov, Heather M Conboy, Nancy Famigletti, George S Avrunin, Lori A Clarke, and Leon J Osterweil. Smart checklists to improve healthcare outcomes. In *International Workshop on SEHS*, pages 54–57, 2016.

[22] Ashley JR De Bie, Eveline Mestrom, Wilma Compagner, Shan Nan, Lenneke van Genugten, Kiran Dellimore, Jacco Eerden, Steffen van Leeuwen, Harald van de Pol, Franklin Schuling, et al. Intelligent checklists improve checklist compliance in the intensive care unit: a prospective before-and-after mixed-method study. *British Journal of Anaesthesia*, 126(2):404–414, 2021.

[23] Hubo Cai, JungHo Jeon, Xin Xu, Yuxi Zhang, Liu Yang, et al. Automating the generation of construction checklists. Technical report, Purdue University. Joint Transportation Research Program, 2020.

[24] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[25] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009.

[26] Eoin M Kenny and Mark T Keane. Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ann-cbr twins for xai. In *Twenty-Eighth International Joint Conferences on Artifical Intelligence (IJCAI), Macao, 10-16 August 2019*, pages 2708–2715, 2019.

[27] Jesus M Darias, Marta Caro-Martínez, Belén Díaz-Agudo, and Juan A Recio-Garcia. Using case-based reasoning for capturing expert knowledge on explanation methods. In *International Conference on Case-based Reasoning*, pages 3–17. Springer, 2022.

[28] Juan A Recio-García, Belén Díaz-Agudo, and Victor Pino-Castilla. Cbr-lime: a case-based reasoning approach to provide specific local interpretable model-agnostic explanations. In *International Conference on Case-based Reasoning*, pages 179–194. Springer, 2020.

[29] Christoph Molnar, Gunnar König, Julia Herbinger, Timo Freiesleben, Susanne Dandl, Christian A Scholbeck, Giuseppe Casalicchio, Moritz Grosse-Wentrup, and Bernd Bischl. General pitfalls of model-agnostic interpretation methods for machine learning models. In *xxAI-Beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020*, pages 39–68. Springer, 2022.

[30] Guglielmo Papagni, Jesse de Pagter, Setareh Zafari, Michael Filzmoser, and Sabine T Koeszegi. Artificial agents' explainability to support trust: considerations on timing and context. *AI & SOCIETY*, pages 1–14, 2022.

[31] Chathurika S Wickramasinghe, Daniel L Marino, Javier Grandio, and Milos Manic. Trustworthy ai development guidelines for human system interaction. In *2020 13th International Conference on Human System Interaction (HSI)*, pages 130–136. IEEE, 2020.

[32] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N Bennett, Kori Inkpen, et al. Guidelines for human-ai interaction. In *Chi conference on human factors in computing systems*, pages 1–13, 2019.

[33] Dobrila Rancic Moogk. Minimum viable product and the importance of experimentation in technology startups. *Technology Innovation Management Review*, 2(3), 2012.

[34] Roberto Pires. Promoting sustainable compliance: Styles of labour inspection and compliance outcomes in brazil. *International Labour Review*, 147(2-3):199–229, 2008.

[35] David Weil. Improving workplace conditions through strategic enforcement. *Boston U. School of Management Research Paper*, (2010-20):2–3, 2010.

[36] Kerstin Bach, Bjørn Magnus Mathisen, and Amar Jaiswal. Demonstrating the mycbr rest api. In *International Conference on Case-based Reasoning Workshops*, pages 144–155, 2019.

[37] Rodrigo J Daly Guris and Meghan B Lane-Fall. Checklists and cognitive aids: underutilized and under-researched tools to promote patient safety and optimize clinician performance. *Current Opinion in Anaesthesiology*, 35(6):723–727, 2022.

[38] Ole Jakob Mengshoel, Tong Yu, Jon Riege, and Eirik Flogard. Stochastic local search for efficient hybrid feature selection. In *Genetic and Evolutionary Computation Conference*, pages 133–134, 2021.

[39] Ole Jakob Mengshoel, Eirik Flogard, Jon Riege, and Tong Yu. Stochastic local search heuristics for efficient feature selection: An experimental study. In *Norsk IKT-konferanse for forskning og utdanning*, pages 58–71, 2021.

[40] Ole Jakob Mengshoel, Eirik Lund Flogard, Tong Yu, and Jon Riege. Understanding the cost of fitness evaluation for subset selection: Markov chain analysis of stochastic local search. In *Genetic and Evolutionary Computation Conference*, pages 251–259, 2022.

ISBN 978-82-326-7854-9 (printed ver.)
ISBN 978-82-326-7853-2 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

NTNU

Norwegian University of
Science and Technology