CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

LICENTIATE THESIS

Supervised Learning is a popular machine learning paradigm that requires labeled data. Data Labeling refers to the process of annotating data to be used for supervised Learning. Implementing a robust and efficient process can be difficult for many reasons. In-house and third-party data labeling have their pros and cons. This thesis addresses the primary problems faced in the industry. It outlines the challenges and mitigation strategies in the industry and provides new and improved mitigation strategies based on Active Learning and Semi-Supervised Learning. Active Learning is a machine learning paradigm that selects the instances to be labeled based on a query strategy. Semi-Supervised Learning utilizes both labeled and unlabeled instances to train a classifier. The thesis provides practitioners with important guidelines for developing a data labeling process based on empirical simulation studies utilizing the most commonly used semi-supervised algorithms and benchmark datasets. First, it provides the optimal algorithms that achieve the highest accuracy based on the dataset's characteristics and w.r.t number of manually labeled instances. Second, it tells when the algorithms have a high probability of achieving an accuracy of 90%. Third, it shows how the algorithms will perform on real-life data. Lastly, it provides practitioners with datasets suitable for evaluating semi-supervised learning algorithms. Based on the results presented in this thesis, practitioners in the industry will know how many labels they need and which labeling algorithm to choose based on desired accuracy for their use cases.

CHALMERS
UNIVERSITY OF TECHNOLOGY

TEODOR FREDRIKSSON • Problems, Opportunities and Solutions for Automatic Labeling of Data Using Machine Learning • 2023

Problems, Opportunities and Solutions for Automatic Labeling of Data Using Machine Learning

TEODOR FREDRIKSSON

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Opportunities, Challenges and Solutions for Automatic Labeling of Data Using Machine Learning

Teodor Fredriksson

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2023

Opportunities, Challenges and Solutions for Automatic Labeling of Data Using Machine Learning

Teodor Fredriksson

Technical Report No. 5313
ISBN:978-91-7905-847-0
ISSN 0346-718X
This thesis has been prepared using LaTeX.

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000
www.chalmers.se

Printed by Chalmers Reproservice
Gothenburg, Sweden, April 2023

To my parents, Ileana, Maria and Anders.

Abstract

Context: Supervised learning is the most common machine learning paradigm and requires labeled data. Because much data in the industry is unlabeled, data labeling is an essential step in the data preparation process. As data labeling can take time and require domain knowledge, many companies need more resources for in-house personnel with domain-specific knowledge to perform labeling. Therefore it is relevant for companies to explore cheaper, more accurate, and automated approaches to labeling.
Objective: This research aims to identify industry challenges and mitigation strategies for Data Labeling.
Method: The research was conducted using multidisciplinary research. We performed a systematic mapping study to understand and identify the main approaches to labeling and their application domains. We performed a case study with two companies to understand the challenges and mitigation strategies in the industry. The case study consisted of an internship with one of the companies and interviews with data scientists from both companies. A thematic analysis was then utilized to formulate challenges based on collected data. For each of the challenges, a mitigation strategy was formulated. The rest of the research consists of simulations and the Bayesian Bradley-Terry Model and Item Response Theory to study what labeling approaches are best in accuracy and evaluate the sustainability of the datasets used to evaluate the labeling approaches.
Results: In this thesis, we present four main findings. First, we present an overview of the most popular data labeling approaches used in different applications, and we provide an overview of the datasets used to evaluate these. Second, we define and categorize the different industry challenges that data scientists face. We then define and formulate mitigation strategies for these challenges. Third, we present the best automated labeling approaches for accuracy and how much manual effort these algorithms need to achieve the best accuracy. Fourth, we present the best benchmark datasets for evaluating automatic labeling approaches.
Outlook: In future work, we want to examine safe and deep semi-supervised learning and how they are used in practice, as we have noticed that semi-supervised learning based on Deep Learning has become more prevalent in recent years.

Keywords: Data Labeling, Semi-supervised learning, active learning.

List of Publications

This thesis is based on the following publications:

[A] **Teodor Fredriksson**, David Issa Mattos, Jan Bosch, Helena Holmström Olsson, "Machine Learning Algorithms for Labeling: Where and How They are Used?". Published in 2022 IEEE 16th International Systems Conference (SYSCON).

[B] **Teodor Fredriksson**, David Issa Mattos, Jan Bosch, Helena Holmström Olsson, "Data Labeling: An Empirical Investigation into Industrial Challenges and Mitigation Strategies". Published in 2020 21st International Conference on Product-Focused Software Process Improvement (PROFES) p. 202-206.

[C] **Teodor Fredriksson**, David Issa Mattos, Jan Bosch, Helena Holmström Olsson, "An Empirical Evaluation of Graph-Based Semi-Supervised Learning Algorithms". Submitted to Elsevier Computers in Industry Journal.

[D] **Teodor Fredriksson**, David Issa Mattos, Jan Bosch, Helena Holmström Olsson, "Assessing the Suitability of Semi-Supervised Learning Datasets using Item Response Theory". Published in 2021 47th Euromicro Conference on Software Engineering and Advanced Applications, (SEAA), p. 326-333.

Other publications by the author, not included in this thesis, are:

[E] **Teodor Fredriksson**, David Issa Mattos, Jan Bosch, Helena Holmström Olsson, "An Empirical Evaluation of Algorithms for Data Labeling". Published in IEEE 45th Annual Computers, Software, and Applications Conference, (COMPSAC) p. 201-209.

[F] **Teodor Fredriksson**, Jan Bosch, Helena Holmström Olsson, "Machine Learning Models for Automatic Labeling: A systematic literature review". Published in 2020 15th International Conference on Software Technologies (ICSOFT), p. 552-566.

Individual Contributions

I am the first author of all publications included in this thesis. I planned, executed, analyzed, and reported all the research in this thesis with some help from David Issa Mattos. The individual contributions are specified below according to the CRediT (Contributor Roles Taxonomy)

Conceptualisation: I formulated the research goals in Paper E alone and collaborated with David Issa Mattos in Papers A-D, and F.

Methodology: I formulated and developed the research methodology in Papers B, F alone and collaborated with David Issa Mattos in Papers A, C-E.

Software Programming: I programmed the machine learning and statistical models in Paper F. In Papers C, and D, the machine learning models were programmed by me and David Issa Mattos programmed the statistical models. No programming was done in Papers A, B, and E.

Formal analysis: The formal analysis was performed by me alone in Papers A, E, F, and David Issa Mattos assisted in Papers B-D.

Investigation: I performed the investigation process of screening papers in Papers A, and E, the interviews of Paper B, and the experiments of Papers C, D, and F David Issa Mattos collected the data from the statistical models in Papers C and D.

Data Curating I managed all the data except for the data from the statistical models of Papers C and F.

Writing: I am the main author and wrote the drafts for Papers A-F except for the parts where the statistical models are described in Papers C and D.

Project administration: I administered all contact with the companies involved in the studies.

Acknowledgments

First, thanks to my supervisors, Professor Jan Bosch and Professor Helena Holmström Olsson, for giving me this opportunity and believing in me.

Secondly, I want to thank my industry partners for providing me with insights into their practices.

Third I want to thank my colleague Dr. Sakib Sistek from Chalmers for always supporting me in situations of stress and doubt.

Acronyms

ML:	Machine Learning
DL:	Deep Learning
SL:	Supervised Learning
SSL:	Semi-supervised learning
AL:	Active Learning

Contents

CHAPTER 1

Introduction

With the rise of Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL) have become popular techniques in the industry. ML and DL models have proven to improve quality when trained on larger datasets, especially for DL models. Because the performance of algorithms depends on the dataset size [1], companies are required to collect large datasets for successfully training high-quality ML and DL models. Data collecting is usually not the problem, as many companies have massive datasets. The main problem is how many instances of the datasets are labeled.

The three main machine learning paradigms are *supervised learning* (SL), *unsupervised learning* (UL), and *reinforcement learning* (RL). Out of these three, SL is the most commonly used. SL is different from UL and RL because it requires labeled instances. Many of the datasets collected from the industry are incomplete because labels are missing either partially or entirely. Missing labels make it challenging to train SL algorithms of high quality. Because obtaining labels is essential, companies must allocate manual labeling tasks to employees. According to research, more than 80% of engineering tasks in ML projects are spent preparing and labeling data [2]. The important questions that companies have to ask are, "Who will perform the labeling?",

"How long will the labeling take?" and "how much will the labeling cost?

Data scientists usually have the best knowledge of how datasets should be labeled. In many instances, however, the labeling has to be done manually. Manual labeling of instances can be a timely and tedious task, causing mistakes to be made because of the human factor. Furthermore, data scientists are specialists in building models and requiring them to spend time on tasks such as annotating data like images, and video will waste resources.

Data Labeling is important in applications such as autonomous driving [3], and medical applications [4]. Autonomous vehicles must detect objects with high accuracy to guarantee the safety of passengers and pedestrians. Medical data annotation is required to enable DL models for detecting cancer cells in patients and saving lives. Because data labeling is important for companies, the market for third-party data labeling services has risen and is supposed to triple by 2024 [5], [6]. Crowdsourcing is one such third-party service [7]. Crowdsourcing is a common strategy to acquire labels using human supervision [8], [9]. An example of such a service is Amazon Mechanical Turk[10]. Useful as it might be, there are some reasons why it can not be used in all cases. First of all, companies are unable to share data that contain confidential information, such as data that is used for military purposes or surveillance. Secondly, there might not be any people who understand the labeling task, meaning the data will be mislabeled. Therefore, in-house labeling is still utilized to avoid security issues and the risks of getting mislabeled data while being half as expensive as crowdsourced labels [2].

Crowdsourcing and in-house labeling have pros and cons as they are both expensive but in different ways, and companies want to rely on more automated approaches for labeling. *Human-in-the-loop* (HITL) machine learning [11] approaches such as *Active Learning* (AL) [12] help increase the efficiency of data labeling for different tasks such as computer vision and natural language processing [13]. The primary purposes of HITL are to improve ML models' accuracy, help reach a target accuracy faster, and maximize accuracy by combining human and machine intelligence.

AL [12] is compared to Passive Learning (PL) [12]. In a passive learning (PL) scenario, the learner randomly chooses instances to be labeled by some oracle (e.g., a human annotator). These newly labeled instances will then be used to train the ML model. If the model accuracy is insufficient, the user randomly chooses another batch of labels to be annotated, adds the

newly labeled instances to the training dataset, and re-evaluates the model accuracy. This procedure is repeated until some pre-defined threshold has been reached. PL is inefficient, as this might include training samples that are not *informative* and hence do not increase model accuracy. On the other hand, AL allows the underlying ML model to choose the instances for which it is trained. This is done by imposing so-called *query strategies* that help choose the most informative sample to be labeled by the oracle.

Another approach is to use semi-supervised learning (SSL) algorithms [14]. Supervised learning algorithms utilize labeled data to classify new instances, and unsupervised learning algorithms utilize unlabeled data to find patterns in data. Semi-supervised learning algorithms utilize both labeled and unlabeled data to classify either the unlabeled data at hand (*transductive* SSL) or new instances (*inductive* SSL). Furthermore, SSL algorithms operate under the assumption that the number of unlabeled instances is far more significant than that of labeled instances.

A large portion of a Data Scientist's job is extracting data from various databases, the rest is spent training ML models. Both tasks require specialized knowledge of software such as Python and SQL. Because data scientists possess specialized knowledge, having them spend time labeling data is a waste of time and resources.

Although there have been many publications on active learning techniques and semi-supervised learning algorithms [15], there needs to be more research to help practitioners in the industry choose the best algorithms for their application domain. ML projects can contain many complicated tasks that have to be planned carefully to account for costs and risks. First, understanding a dataset's structure is important for choosing the optimal algorithm that will yield the highest accuracy. Many algorithms tend only to perform well for datasets with a certain structure or datatype. It is also important to know how the algorithm (SL or SSL) works as it affects the choice of AL strategy. The accuracy of many ML and DL algorithms increases as the size of the dataset increases. Small datasets can yield a low accuracy. There are ways to generate data but that can lead to other problems, such as underfitting and overfitting [16]. If practitioners need to label data manually, they need to know how much data they should label and how to perform quality checks of the labels. When researchers in the industry need to develop their algorithm, it can be benefical if the algorithm will perform well on many different datasets.

In order to ensure that the algorithm is transferable to as many datasets as possible, practitioners need to know how to evaluate their algorithm and include the optimal datasets. This thesis strives to help practitioners reduce the costs and risks that arise when creating their labeling infrastructure by providing practitioners with guidelines based on the results of the following studies.

First, we present a systematic mapping study where we performed a literature search for many types of AL och SSL algorithm. Based on specified criteria, we included a significant number of papers. From each paper, we identified the algorithms, the application domain, and the datasets used to evaluate the algorithm. Secondly, we present an empirical investigation into industry challenges and mitigation strategies. The investigation is based on data gathered from a case study involving two software companies in the embedded system-domain, one less experienced and one with vast experience in labeling. In the study's first phase, we spent time at company A, 2-3 times/week. During the internship, we attended meetings with stakeholders and workshops. We were granted access to datasets that we analyzed using Python. After the internship, we interviewed participants from companies A and B. During the entirety of the study, we took note of problems the companies face when labeling data. After the interviews, we performed a thematic analysis and formulated three challenges based on the observed problems. A separate mitigation strategy was then formulated for each of the challenges. Third, to provide practitioners with a guide on what algorithm to use for different situations, we apply simulations to evaluate what algorithms and methods provide the best accuracy and how many labels need to be available to achieve the highest accuracy for data labeling. We limited the algorithms and datasets that, according to our mapping study, are the most popular. We ran the simulations where we varied the number of available labeled instances to investigate how much manual effort was required. In order to determine what algorithm is better than another, we applied Bayesian statistical inference in the Bradley-Terry model to rank the algorithms according to the highest accuracy. The results of the study indicated that the algorithms learned some datasets better. Therefore, our last contribution is a study in which we utilized Item Response Theory to evaluate what benchmarked machine learning datasets are best suited to evaluate the ability of SSL algorithms.

CHAPTER 2

Background

In this chapter, the background for the labeling problem is described. First, the labeling problem is discussed in the context of machine learning. Second, the machine learning approaches for solving the labeling problem are presented.

2.1 Basic Machine Learning Paradigms

This section discusses the main machine learning paradigms described in this thesis. We start by discussing supervised learning, unsupervised learning and reinforcement learning. Afterwards we discuss data labeling techniques that can reduce manual labeling such as Active Learning scenarios and different query strategies. Lastly we discuss semi-supervised learning and how it extends supervised learning.

Supervised Learning

In *supervised classification*, we have an arbitrary set $\mathcal{X}$ of objects we wish to label. The set $\mathcal{X}$ is called the *features*, and each element $x \in \mathcal{X}$ is called an

instance. The set $\mathcal{Y}$ is known as the *label set* and is the set of possible labels. Supervised learning aims to find a *classifier* $h\colon \mathcal{X} \to \mathcal{Y}$ based on some training data

$$S = \{(x_1, y_1), ..., (x_m, y_m)\},$$

where $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y},\ i = 1, ..., m$. Furthermore, we assume that the data comes from a data distribution $\mathcal{D}$, where all points $s \in S$ are generated from $\mathcal{D}$ and then labeled by a *labeling function* f. Neither $\mathcal{D}$ nor f are observed by the learner. The end goal is to estimate the *true error*

$$L_{\mathcal{D},f}(h) := \mathop{\mathbb{P}}_{x\sim\mathcal{D}}[h(x) \neq f(x)],$$

by some *loss* function $L_S(h)$ that is empirically estimated with the help of the training data. The goal is to find a classifier h_S that minimizes the loss function,

$$h_S \in \operatorname*{argmin}_{h\in\mathcal{H}} L_S(h).$$

This procedure is referred to as *Empirical Risk Minimization* (ERM), and the choice of $L_S(h)$ will vary based on the data and application. In many cases h_S is found by solving an optimization problem and it is important to use an appropriate L_S so that a chosen learning algorithm converges to a solution.

Unsupervised Learning

In *unsupervised learning*, we only have access to features $\mathcal{X}$ and no labeled data. There are many unsupervised tasks such as *clustering* and *dimensionality reduction* [17].

Clustering is the task of grouping so that similar objects belong to the same group and is an important step in exploratory data analysis. A clustering algorithm takes input features $\mathcal{X}$ and requires a distance function d such that

$$d\colon \mathcal{X} \to \mathcal{X}, \quad d(x, x) = 0, \quad d(x, y) = d(y, x),\ \forall x, y \in \mathcal{X},$$

or a *similarity function* $s\colon \mathcal{X} \times \mathcal{X} \to [0, 1]$ such that $s(x, x) = 1$. The output of the algorithm is a partition of C of $\mathcal{X}$ such that,

$$C = (C_1, ..., C_k), \quad \bigcup_{i=1}^{k} C_i = \mathcal{X}, \quad C_i \cap C_j = \varnothing, \text{ for all } i \neq j.$$

where each $C_i,\ i = 1, ..., k$ represents a cluster. The k-means algorithm is an example of a clustering algorithm that requires the distance measure

$$d(x, y) = \|x - y\|,$$

and a parameter for the number of clusters k. The goal of the algorithm is to find a parition $C = (C_1, ..., C_k)$ by minimizing the objective function $G = G(\mathcal{X}, d, C)$. Each $C_i, i = 1, 2, ..., k$ has a centroid μ_i defined as

$$\mu(C_i) = \underset{\mu_i}{\operatorname{argmin}} \sum_{x \in C_i} d(x, y)^2,$$

and the objective function is

$$G = \min_{\mu_1, ..., \mu_k} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu)^2.$$

Dimensionality reduction algorithms map high-dimensional data to a lower-dimensional space. Studying data in lower dimensions is helpful because it is easier to visualize. Higher dimensional data have many computational challenges and can give a bad generalization. The dimensionality reduction is performed by applying a linear transformation,

$$T \colon \mathbb{R}^m \to \mathbb{R}^n, \quad T(\mathbf{x}) = W\mathbf{x}.$$

A dimensionality reduction algorithm aims to find W by solving an optimization problem. A second matrix U can recover the original vector $\mathbf{x}$. An example of a dimensionality reduction algorithm is *Principal Component Analysis (PCA)*, in which U and W are found by solving the optimization problem.

$$\underset{W,U}{\operatorname{argmin}} \sum_{i=1}^{m} \|x_i - UWx_i\|_2^2.$$

Reinforcement Learning

RL differs from SL and UL because it does not utilize labeled data. Applications of RL-based systems include making robots walk, defeating world champions at backgammon, chess, and Go, managing investment portfolios,

and playing Atari video games [18]. Reinforcement Learning relies on mathematical control theory and dynamic systems.

The key ingredient in Reinforcement Learning is *Markov Decision Processes*. We say that a sequence of states $S_1, S_2, ..., S_t$, is Markov if the probability of moving to the next state S_{t+1} depends only on the present state S_t and not its previous states. In other words the probability of S_{t+1} is only dependent the previous state S_t,

$$\mathbb{P}(S_{t+1}|S_t) = \mathbb{P}(S_{t+1}|S_1, ..., S_t).$$

A Markov process is a tuple $(\mathcal{S}, \mathcal{P})$ where $\mathcal{S}$ is a set of states and $\mathcal{P}$ is a state transition probability matrix, here the element

$$\mathcal{P}_{ss'} = \mathbb{P}(S_{t+1} = s'|S_t = s)$$

represents the probability of moving from state s' to s.

The dynamics of a Markov process can be outlined as follows: We start at some state s_0 and then transition next state s_1, where s_1 is drawn from $\mathcal{P}_{s_0 s_1}$. Then we transition from s_1 to s_2 and so on.

A Markov Decision process is extended from a Markov process by introducing a reward, actions, and discount. Furthermore, a *Markov Decision Process* (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{R})$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P}$ is the state transition probability matrix where

$$\mathcal{P}^a_{ss'} = \mathbb{P}(S_{t+1} = s'|S_t = s, A_t = a),$$

and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a *reward function*. In the MDP, the transition between states depends on the previous state and a taken action A_t. The action taken at a state is also associated with a reward R_t. The dynamics of the MDP are: Start at state s_0 and choose an action $a_0 \in \mathcal{A}$. As a result the MDP randomly transits to a successor state s_1, drawn from $\mathcal{P}^{a_0}_{s_0 s_1}$, Then for state s_1 we pick another state and on it goes.

RL aims to choose the most optimal actions to maximize *return*. The return G_t is the total discounted reward from time-step t.

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

The *policy* π is the probability of taking A_t given state S_t.

$$\pi(a|s) = \mathbb{P}(A_t = a|S_t = s).$$

In order to find the optimal policy, we utilize the *state-value* and *action-value* functions. The state-value function is defined as

$$v_\pi(s) := \mathbb{E}_\pi(G_t|S_t = s),$$

the expected return value starting from state$s$ when following a policy π. The action-value function is defined as the expected return starting from state s, taking action a and then following a policy π

$$q_\pi(s, a) := \mathbb{E}(G_t|S_t = s, A_t = a).$$

Furthermore, there is a relationship between $v_\pi(s)$ and $q_\pi(s, a)$ through the *Bellman Equations*,

$$\begin{aligned} v_\pi(s) &= \sum_{a\in\mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right), \\ q_\pi(s, a) &= \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a \pi(a'|s') q_\pi(s', a'), \end{aligned}$$

where

$$\mathcal{R}_s^a = \mathbb{E}(R_{t+1}|S_t = a, A_t = a).$$

We care for the *optimal state-value* and *action-value functions* defined as the maximum state-value and action-value functions over policies, respectively,

$$\begin{aligned} v_*(s) &= \max_\pi v_\pi(s), \\ q_*(s, a) &= \max_\pi q_\pi(s, a). \end{aligned}$$

It is now relevant to find the optimal policy π_*. By definition, a policy π is said to be better than a policy π, denoted $\pi \geq \pi'$, if $v_\pi(s) \geq v_{\pi'}(s)$, $\forall s$. Therefore the optimal policy π_* is defined as the policy that is better than every other policy. According to the theory of Markov Decision Processes,

such an optimal policy exists and is given by

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \text{argmax}_{a \in \mathcal{A}}\, q_*(s,a) \\ 0 & \text{otherwise} \end{cases} .$$

The remaining problem will be to find the optimal value functions. The value functions can be expressed in terms of each other,

$$v_*(s) = \max_a q_*(s,a),$$
$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s'),$$

and by performing substitutions, we can formulate the *Bellman optimality equations*,

$$v_*(s) = \max_a \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s') \right),$$
$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \max_{a'} q_*(s',a').$$

In summary the objective in solving an RL is to solve the Bellman optimally equations. Because these equations are non-linear, they cannot be represented in closed form and must be solved using dynamical programming.

2.2 Data Labeling

SL algorithms are sometimes challenging to use as labeled data might be hard to collect automatically, and a human labeler (*or supervisor*) might be needed to collect labels [19]. ML and DL models are known to perform better on large datasets [1], so obtaining large labeled datasets is of grave importance. Companies might have massive datasets that contain many features. However, it is often the case that these datasets are incomplete because they are missing labels, either partially or entirely. Because of this, companies might have in-house personnel perform the data labeling manually.

One such solution is to use *crowdsourcing*. Crowdsourcing in machine learning entails the hiring of annotators at a large scale. Many crowdsourcing plat-

forms such as *Scale, Clickworker, Lionbridge AI, Isahit, MarsCrowd, Amazon Mechanical Turk* and *Cloud Factory* offer data labeling for different tasks such as *sentiment analysis, entity extraction, text classification*, *text translation*, *image classification, object detection* and *image segmentation* among others. Crowdsourcing offers a 24/7 workforce and it is easy and affordable. Some cons of crowdsourcing are that quality control is not guaranteed, and it is challenging to maintain consistent results over time. Companies that need to label sensitive and confidential data can not share their data with third-party organizations, and many companies still utilize in-house labeling. A consistent annotation process can yield long-term reliability and success if a company has a consistent annotation process. Companies can set up feedback loops for the annotation procedure, constantly improve their labeling and set up robust quality control. However, building the annotation process is expensive and time-consuming if the company is small.

Both crowdsourcing and in-house labeling have their pros and cons. To the best of the author's knowledge and experience with the industry, most companies are not interested in utilizing any of these approaches. Companies are more interested in investigating automated approaches to reduce manual effort or to automate the process entirely.

2.3 Active Learning

One way to reduce manual effort is to use AL. Machine Learning systems usually implement *passive learning* (PL) in which the learner randomly chooses instances to be labeled and used to train a classifier $h\colon X \to Y$ to predict labels for new unlabeled data. Selecting instances at random is problematic as these instances are not guaranteed to be *informative*, which is bad for training the classifier. An active learning system develops and learns new classifiers as the interactive learning process continues. Unlike PL, the AL approach will not choose instances randomly but according to some *query strategy*. The learner would then train a classifier on the training data. Suppose the classifier does not reach the desired accuracy. In that case, the learner will query more instances to be labeled or re-labeled, add these newly labeled instances to the training data and then retrain the model. This procedure would iterate until the desired accuracy is reached. When using active learning, it is assumed that large amounts of unlabeled instances are available and easy to collect.

Let us illustrate this by a simple example where we have access to features $\mathcal{X}$ describing the health of a test subject in a clinical trial for evaluating the effects of a new medication and labels $\mathcal{Y} = \{0, 1\}$ representing the "good" or "bad" outcome of the trial. The objective is to find a classifier $f \in \mathcal{H}$ to predict whether the medication is safe for a test subject based on its health. Assume we can rank the participant's health on a scale of 1,...,10. In other words, we can order the features and it is logical to define a threshold θ to determine whether using the medication based on your health is safe. Let the hypothesis space be defined as

$$\mathcal{H} := \{h = h(x; \theta)\},$$

such that

$$h(x; \theta) = \begin{cases} 1, & x < \theta \\ 0, & \text{otherwise.} \end{cases}$$

In a supervised setting we must collect unlabeled samples, administer the medication to each individual, and see the outcome. Based on the effect of the medication, we now have a labeled training set for which we have acquired an estimate $\widehat{\theta}$. According to the theoretical framework of PAC learning theory [20], we need $\frac{1}{\epsilon} = 100$ instances to guarantee an error rate of $\epsilon = 0.01$.

Instead, suppose we impose a strategy, *binary search* algorithm to find θ. The time complexity goes from $O(1/\epsilon)$ to $O(\log_2 1/\epsilon)$, so instead, it would only take $\log_2 1/\epsilon = \log_2 100 \approx 7$ instances to achieve the same accuracy but it will be much less expensive to acquire features.

Above is an example of an Active Learning scenario where the instances to be labeled is queried according to a strategy. The example is very simplistic because, in reality, the features can be much more complex and expensive to collect, dependent on the application. Furthermore, there are different ways a learner can ask queries. The three main senarios are *query synthesis, stream-based selective sampling* and *pool-based sampling* [12].

Stream-based selective sampling

This scenario draws an unlabeled instance[21], [22] one at a time from some input source, an actual distribution, and then decides whether to query it or

not [23]. Although this query type is cheap regarding computational cost, querying only one instance at a time only considers part of the distribution in the decision process. Another disadvantage is that the learner needs to set a decision boundary that decides if an instance should be queried. The learner might discard valuable instances if the decision boundary is set poorly.

Query Synthesis

Unlike stream-based selective sampling, query synthesis generates data points that it thinks are most informative from a synthetic distribution[24] and queries these to be labeled by the oracle [25]. This approach has the issues of stream-based selective sampling in the sense that it might not be able to model some areas of the actual distribution that is unknown and will therefore be unable to query labels from the entire distribution. In some cases, query synthesis has generated incomprehensive instances. In[26], a DL model was trained to recognize handwritten characters. The query synthesis would generate artificial hybrid characters that do not make sense.

Pool-based sampling.

In this scenario, it is assumed that a large pool of unlabeled data and a small pool of labeled samples are available. The unlabeled data is queried according to some query strategy. Pool-based sampling is the most popular of the scenarios and has been applied in many different applications [27]–[29]. Furthermore, pool-based sampling is computationally expensive since it needs to evaluate each instance in the unlabeled pool. Still, pool-based sampling is optimal for choosing the most informative sample. However, query synthesis and stream-based sampling will be better if the memory and processing power is limited, such as in mobile and embedded devices.

Measuring informativeness:

Once the Query Type has been selected, the learner needs to decide on what measure to use for measuring informativeness. The most prevalent techniques *uncertainty sampling, query-by-committee* and *Error and Variance reduction.*

Uncertainty sampling:

The motivation for sampling the most uncertain instances is that the more uncertain instances are, the more they can reveal the ground truth. The most simple is the *least confident* instance, which is found by calculating

$$x^*_{LC} = \arg\max_x 1 - P_\theta(\widehat{y}|x)$$

where $\widehat{y} = \arg\max_y P_\theta(y|x)$ is the instance with the highest posterior probability. It becomes intuitive that x^*_{LC} will be the instance with the lowest probability, hence the most uncertain instance. The problem with the LC estimate is that it only considers the label with the highest probability. To mitigate that, we can use the *margin sampling*[12] that considers the two labels with the highest probabilities. The margin estimate is given by the instance that minimizes the difference.

$$x^*_M = \arg\min_x P_\theta(\widehat{y}_1|x) - P_\theta(\widehat{y}_2|x).$$

The larger the margin, the more confident we are about assigning that label. The more popular approach that considers all the labels is the *entropy* query strategy defined as:

$$x^*_E = \arg\max_x -\sum_i P(\widehat{y}_i|x) \log P(\widehat{y}_i|x)$$

Entropy measures the average information of x.

Query-by-Committee

A hypothesis f is said to be *consistent* if it fits the training instances perfectly but does not have a perfect generalization error. We define the subset $\mathcal{V} \subseteq \mathcal{H}$ denoted *version space* [30] as the set,

$$\mathcal{V} = \{h : \mathcal{X} \to \mathcal{Y} : \text{consistent with the training data}\},$$

and let $|\mathcal{V}|$ denote the size of the versions space. Assuming we can find hypotheses that fit the training data perfectly, we add more labeled data and retrain the model. Hopefully, the generalization error will increase as we retrain, and the size of the version space $|\mathcal{V}|$ will decrease. Therefore instances

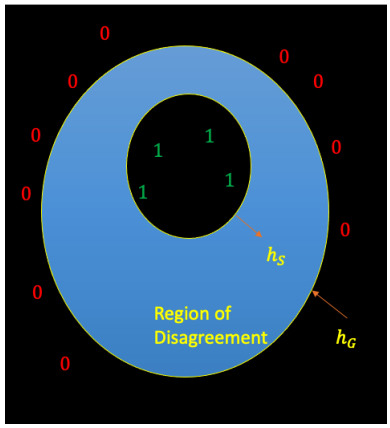


Figure 2.1: The set of most specific and most general hypothesises h_S and h_G.

that reduce the version space size quickly should be queried.

First, consider the *Query by Disagreement* (QBD) query strategy [22], which assumes the stream-based sampling query type. QBD can be illustrated with the *region of disagreement* DIS($\mathcal{V}$). There are however shortcomings to this approach. If $\|\mathcal{V}\|$ is infinite, then the version space can not be stored in memory. To mitigate the issue, we could specify two speculative hypotheses h_1, h_2 trained on $S \cup (x, 0)$ and $S \cup (x, 1)$ respectively. On the other hand, if we generalize $\mathcal{Y}$ to contain n different classes, this procedure will be computationally expensive as $n \to \infty$. Another approach is to consider two subsets $\mathcal{S}$ and $\mathcal{G}$ where

$$\mathcal{S} = \{\text{set of } \textit{most specific} \text{ hypothesis}\}$$
$$\mathcal{G} = \{\text{set of } \textit{most general} \text{ hypothesis}\}$$

QBD is then applied to $h_1 \in \mathcal{S}$ and $h_2 \in \mathcal{G}$. This approach does not force us to maintain the entire hypothesis as we summarize DIS($\mathcal{V}$) with $\mathcal{S}$ and $\mathcal{G}$, but maintaining these sets can be expensive still. There is yet another approach that only maintains two hypothesises, h_1 and h_2. If for some instance, xDIS($\mathcal{V}$) and $y = 1$, then h_1 must become more general. If $y = 0$ then h_2 needs to be more specific. see Figure 2.1. Unfortunately, this approach can also be computationally expensive in terms of computation as the size of S increases.

Another approach that keeps only two hypotheses, h_S and h_G, is roughly the most specific and general in $\mathcal{V}$. We then approximate

$$\text{DIS}(\mathcal{V} \approx \{x \sim \mathcal{D} \colon h_S(x) \neq h_G(x)\},$$

and sample *background points* $\mathcal{B} \sim \mathcal{D}$ and add them to $\mathcal{S}$ with artificial labels- To train the h_S hypothesis, give the instances the "0" label so that it becomes more conservative. To train h_G. add the "1" label to make it more willing to label unlabeled data as "1".

Using QBD instead of uncertainty sampling is tempting as it poses queries for different hypotheses. However, QBD is measured among all $h \in \mathcal{H}$, which can be problematic. The data could also be noisy, and $\mathcal{V}$ could be challenging to define. Furthermore, QBD needs to be generalized to the pool-based scenario.

So far we have relied on two assumptions

1. Disagreement is measured among all hypothesis $h \in \mathcal{V}$, or two extremes h_S and h_G.

2. Disagreement is a binary measure. No controversial instance matters more than another.

Query-by-committee relaxes these assumptions and makes pool-based active learning possible. QBC rerefers to all disagreement approaches that use a "committee" or ensembles C of hypotheses. All QBC approaches requires a method for obtaining a hypothesis in the committee and a heuristic for measuring disagreement among them. Some examples of measuring disagreement include *Vote entropy*

$$x^*_{VE} = \arg\max_x \left(-\sum_y \frac{\text{vote}_{\mathcal{C}}(y,x)}{|\mathcal{C}|} \log \frac{\text{vote}(y,x)}{|\mathcal{C}|} \right)$$

where y are all possible labelings and

$$\text{vote}_C(y,x) = \sum_{\theta \in C} 1_{h_\theta(x)=y},$$

is the number of votes. A second disagreement measure is *soft vote entropy*

defined as:

$$x^*_{SVE} = \arg\max_x \left(-\sum_y P_{\mathcal{C}}(y|x) \log P_{\mathcal{C}}(y|x) \right)$$

where

$$P_{\mathcal{C}}(y|x) = \frac{1}{|\mathcal{C}|} \sum_{\theta \in \mathcal{C}} P_\theta(y|x)$$

is the average probability that the committee agrees that y is the correct label, according to the committee. Last, the Kullback-Leiber measurement measures the difference between probability distributions. We query the sample x that maximizes the average divergence of each committee member of θ's prediction for that consensus $\mathcal{C}$

$$x^*_{KL} = \arg\max \frac{1}{|\mathcal{C}|} \sum_{\theta \in \mathcal{C}} \mathrm{KL}\left(P_\theta(y|x) | P_{\mathcal{C}}(y|x)\right)$$

. Other query strategies for QBC include *Jensen-Shannon divergence*[31] and *F-compliment* [32].

2.4 Semi-Supervised Learning

AL can reduce the labeling effort by telling the learner what instances are informative, thus reducing manual labeling effort. However, it will only remove the manual labeling partially. SSL can be viewed as an extension to SL as it learns from both labeled data $S_L = \{(x_i, y_i), i = 1, 2, ..., l\}$ and unlabeled data $S_U = \{x_i, i = l + 1, .., m\}$ and it is assumed that there are much more unlabeled instances than there are labeled ones. SSL can increase accuracy by utilizing unlabeled data, requiring fewer labeled instances. We define two types of SSL below [14].

Transductive Semi-Supervised Learning

Transductive SSL algorithms strive to fit a hypothesis h on the labeled data S_L and unlabeled data S_U to predict labels for new incoming data. Like in SL we divide the dataset into a training set and test, although the test set contains only labeled data and no unlabeled data.

Inductive Semi-Supervised Learning

Like transductive SSL algorithms, the inductive SSL algorithms also strive to fit a hypothesis h on the labeled and unlabeled data, but only to fit labels to the unlabeled input data S_U.

Self-Training

Self-training is a simple SSL approach where a supervised classifier h learns from its predictions. Train the classifier h on labeled data S_L and predict the labels $\mathcal{Y}_{predicted} = h(S_U)$ for the unlabeled instances S_U. Remove the most confident predictions from the unlabeled set and add them to the labeled training set. Retrain the classifier based on the updated training set and repeat the steps until satisfactory results are reached.

Generative Models

This section discusses the *generative* approach to learning. The generative approach assumes that the underlying distribution can be expressed in a specific parametric form. The task of learning such parametric form is known as *parametric density estimation*.

Maxmimum Likelihood Estimation

Let $S = (x_1, ..., x_m)$ be the training set with distribution $\mathcal{P}_\theta$. The *log-likelihood function* is defined as the log of the probability of S,

$$\begin{aligned} L(S;\theta) &= \log \mathbb{P}(S = (x_1, ..., x_m)) \\ &= \sum_{i=1}^{m} \log(\mathcal{P}_\theta(x_i)) \end{aligned}$$

The *Maximum Likelihood Estimator* is defined as

$$\widehat{\theta} \in \operatorname*{argmax}_\theta L(S;\theta).$$

A mixture model is a probabilistic model that represents several subgroups. In SL and SSL, a mixture model contains each group for each label. A trained hypothesis h aims to input an instance $x \in \mathcal{X}$ and output a label $y \in \mathcal{Y}$.

Mixtures of Gaussians

When $\mathcal{P}(x)$ can be represented using a linear combination of Gaussian densities

$$\mathcal{P}(x) = \sum_{i=1}^{K} \pi_k \,\text{Normal}(x|\mu_k, \mathbf{\Sigma}_k),$$

$p(x)$ is called a *Gaussian Mixture Model* (GMM). Choosing a significantly large K with appropriate parameters makes it possible to approximate any continuous distribution with a GMM. π_k are called *mixing coefficient* and the sum of all mixing coefficients is 1,

$$\int \mathcal{P}(x)\, dx = \sum_{k=1}^{K} \pi_k = 1.$$

and because $\text{Normal}(x|\mu_k, \sigma_k) \geq 0$, we have that $0 \leq \pi_k \leq 1$, hence π_k are probabilities. It is possible, therefore to express $\mathcal{P}(x)$ as a marginal probability distribution, $\mathcal{P}(x) = \sum_{k=1}^{K} \mathcal{P}(k)\mathcal{P}(x|k)$. Take $\pi_k = \mathcal{P}(k)$ as the prior probability and $\mathcal{P}(x|k) = \text{Normal}(x|\mu_k, \Sigma_k)$ as the likelihood probability. Using Bayes Theorem we calculate the posterior probabilities $\mathcal{P}(k|x)$:

$$\gamma_k(x) = \mathcal{P}(k|x) = \frac{\mathcal{P}(k)\mathcal{P}(x|k)}{\sum_l \mathcal{P}(l)\mathcal{P}(x|l)} = \frac{\pi_k \,\text{Normal}(x|\mu_k, \Sigma_k)}{\sum_l \pi_l \,\text{Normal}(x|\mu_l, \Sigma_l)}.$$

The GMM is controlled by the parameters π, μ, Σ. The parameters are usually computed using MLE but in the multivariate case it can be problematic as the solution cannot be given in the closed form.

Now, let $z = (z_1, ..., z_K)$ be a vector such that $z = 1$ and $z_j = 0$ for all other $j \neq k$. The distribution over z is $\mathcal{P}(z_k = 1) = \pi_k$ which can be written as a product,

$$\mathcal{P}(z) = \prod_{k=1}^{K} \pi_k^{z_k},$$

and

$$\mathcal{P}(x|z_k = 1) = \text{Normal}(x|\mu_k, \Sigma_k),$$

$$\mathcal{P}(x|z) = \prod_{k=1}^{K} \text{Normal}(x|\mu_k, \Sigma_k)^{z_k},$$

so the marginal distribution is then given by,

$$\begin{aligned}\mathcal{P}(x) &= \sum_{z} \mathcal{P}(z)\mathcal{P}(x|z) \\ &= \sum_{k=1}^{K} \pi_k \,\text{Normal}(x|\mu_k, \Sigma_k)\end{aligned}$$

By introducing z, we can now work with $\mathcal{P}(x, z)$, which leads to simplifications in the EM-algorithm. Another important quantity is $\mathcal{P}(z|x)$ which according to Bayes Theorem we can be computed as the posterior distribution,

$$\begin{aligned}\gamma(z_k) &= \mathcal{P}(z_k = 1|x), \\ &= \frac{\mathcal{P}(z_k = 1)\mathcal{P}(x|z_k = 1)}{\sum_{j=1}^{K} \mathcal{P}(z_j = 1)\mathcal{P}(x|z_j = 1)}, \\ &= \frac{\pi_k \,\text{Normal}(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \,\text{Normal}(x|\mu_j, \Sigma_j)},\end{aligned}$$

where π_k is the prior probability of $z_k = 1$ and $\gamma(z_k)$ is the posterior distribution after observing x. Suppose there are observations $\mathbf{x}_1, ..., \mathbf{x}_m$ we wish to model by a GMM, $\mathbf{x}_1, ..., \mathbf{x}_m$ are represented by a matrix $\mathbf{X} \in \mathbb{R}^{m \times D}$ and the latent variables $\mathbf{z}_1, ...\mathbf{z}_K$ can be represented as $\mathbf{Z} \in \mathbb{R}^{m \times K}$. If the data points are drawn i.i.d then we can express a Gaussian Mixture Model for this problem and we can express the log-likelihood as,

$$\begin{aligned}\log \mathcal{P}(\mathbf{X}|\pi, \mu, \mathbf{\Sigma}) &= \sum_{n=1}^{m} \log \sum_{k=1}^{K} \pi_k \,\text{Normal}(\mathbf{x}_n|\mu_k, \Sigma_k) \\ &= \sum_{n=1}^{m} \log \sum_{k=1}^{K} P(\mathbf{z}_k = 1|\pi, \mu, \mathbf{\Sigma})\mathcal{P}(\mathbf{x}_n|\mathbf{z}_k = 1, \pi, \mu, \mathbf{\Sigma}) \\ &= \sum_{n=1}^{m} \log \sum_{k=1}^{K} \mathcal{P}(\mathbf{x}_n, \mathbf{z}_k = 1|\pi, \mu, \mathbf{\Sigma}) \\ &= \log \sum_{k=1}^{K} \mathcal{P}(\mathbf{X}, \mathbf{z}_k|\pi, \mu, \mathbf{\Sigma})\end{aligned}$$

The EM Algorithm

The EM-algorithm aims to find ML solutions for models with latent variables. Let Z denote latent variables, the observed data $\mathbf{X}$ and model parameters θ and a likelihood

$$\log \mathcal{P}(\mathbf{X}|\theta) = \log \left(\sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}, \mathbf{Z}|\theta) \right)$$

The set $\{\mathbf{X}, \mathbf{Z}\}$ is called the *complete* dataset and $\mathbf{X}$ is the *incomplete*. The likelihood for the complete dataset is $\log \mathcal{P}(\mathbf{X}, \mathbf{Z}|\theta)$. The complete dataset is never given in practice, hence the knowledge of the complete dataset. Therefore we consider the expected value under the posterior distribution of the latent variable. If the current estimate is θ^{old} the renewed estimate will be θ^{new}. Initially we choose some random value θ_0. In the E-step calculate the posterior distribution of the latent variables given by $\mathcal{P}(\mathbf{Z}|\mathbf{X}, \theta^{old})$. This quantity is used to calculate the expectation of the log-likelihood function

$$Q(\theta, \theta^{old}) = \sum_{\mathbf{z}} \mathcal{P}(\mathbf{Z}|\mathbf{X}, \theta^{old}) \log \mathcal{P}(\mathbf{X}, \mathbf{Z}|\theta).$$

In the M-step we define the reused parameter estimate θ^{new} by minimizing the function,

$$\theta^{new} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta, \theta^{\theta^{old}}).$$

The general EM-algorithm

1. Choose an initial setting for the parameter θ^{old}.
2. E-step: Evaluate
$$\mathcal{P}(\mathbf{Z}|\mathbf{X}, \theta^{old}).$$
3. M-step: Evaluate θ^{new} given by
$$\theta^{new} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta, \theta^{old}),$$
where
$$Q(\theta, \theta^{old}) = \sum_{\mathbf{z}} \mathcal{P}(\mathbf{Z}|\mathbf{X}, \theta^{old}) \log \mathcal{P}(\mathbf{X}, \mathbf{Z}|\theta).$$
4. Check for convergence of either the log-likelihood or the parameter val-

ues. If the convergence criterion is not satisfied let

$$\theta^{old} \leftarrow \theta^{new},$$

and return to step 2.

EM algorithm for GMM

If we were to estimate the parameters of a GMM, we need to find the parameter $\theta = (\pi, \mu, \Sigma)$. The procedure is outlined below:

1. Initiate parameters $\theta_0 = (\pi_0, \mu_0, \Sigma_0)$ and evaluate the initial value of the log-likelihood.

2. E-step. Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \,\mathrm{Normal}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \,\mathrm{Normal}(\mathbf{x}|\mu_j, \Sigma_k)}.$$

3. M-step. Re-estimate the parameters using the current responsibilities

$$\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n,$$
$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x} - \mu_k^{new})(\mathbf{x} - \mu_k^{new})^T,$$
$$\pi_k^{new} = \frac{N_k}{N},$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}).$$

Support Vector Machines

Here is a brief overview of supervised support vector machines to explain how to extend the idea of semi-supervised learning. For a more detailed description the reader is referred to other literature [33].

The idea of support vector machines is to find a *decision boundary* that separates the label distribution into different classes. Let $f(x) = w^T x + b$ be the decision boundary defined by the set

$$\{x|f(x) = 0\}$$

where $w \in \mathbb{R}^D$ is the parameter vector and $b \in \mathbb{R}$ is an offset parameter. Given an input of labeled examples (x_i, y_i), the objective is to find the decision boundary that maximizes the distance from the decision boundary to the closest labeled instance,

$$\max_{w,b} \min_{i=1}^{l} \frac{y_i f(x_i)}{\|w_i\|}.$$

$$\begin{aligned} \min_{w,b,\xi} \quad & \sum_{i=1}^{l} \xi_i + \lambda \|w\|^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1, ..., l. \end{aligned}$$

The equation above can be formulated as a regularized risk minimization problem

$$\min_{w,b} H(x, y) + \lambda \|w\|^2,$$

where $H(x)$ is the *hinge* loss

$$H(x, y) = \sum_{i=1}^{l} \max(1 - y_i(w^T x_i + b), 0).$$

After solving the optimization problem, we predict the labels by computing $\widehat{y} = \text{sign}(f(x))$. We can then derive the *hat* loss function

$$\widehat{H}(x) = H(x, \widehat{y}) = \sum_{x} \max(1 - |w^T x + b|, 0)$$

The Hinge loss is used to incorporate unlabeled data into the loss function. In order to avoid all the unlabeled classes being predicted into the same class, we also incorporate a constraint on the optimization problem and the final

optimization problem will be

$$\min_{w,b} \quad \sum_{i=1}^{l} \max(1 - y_i(w^T x_i + b), 0) + \lambda_1 \|w\|^2 + \lambda_2 \sum_{j=l+1}^{l+u} \max(1 - |w^T x_j + b|, 0)$$

$$\text{subject to} \quad \frac{1}{u} \sum_{j=l+1}^{l+u} w^T x_j + b = \frac{1}{l} \sum_{i=1}^{l} y_i.$$

Co-training

Co-training is an algorithm that can be used if we can view the labels $\in \mathcal{Y}$ in at least two different *views*. In other words each $y \in \mathcal{Y}$ can be predicted using features that can be divided into two $\mathcal{X} = [\mathcal{X}^{(1)}, \mathcal{X}^{(2)}]$. We then train two different classifiers on each of these views separately and then have them teach each other by checking how they agree on the labels.

The input in the co-training algorithm is the labeled data S_L and unlabeled data S_U. Initially the training sets for both classifiers are identical, i.e., $S_L = S_L^{(1)} = S_L^{(2)}$, as well as an integer learning speed k. We then train h_1 on $S_S^{(1)}$ and h_2 on $S_L^{(2)}$. Then separately classify the rest of the unlabeled instances using both classifiers Add the k best predictions of h_1 to $S_L^{(2)}$, and the k best predictions of h_2 to $S_L^{(1)}$ and then remove those instances from the unlabeled set. Repeat this until the unlabeled data has run out. Co-training is generalized into *multi-view learning* where multiple views are considered. The assumption for multiple views is that x split into a finite number of m views $\mathcal{X} = [\mathcal{X}^{(1)}, ..., \mathcal{X}^{(m)}]$.

Graph-Based Semi-Supervised Learning

Graph-based Semi-Supervised Learning (GSSL) algorithms are the only inductive SSL algorithms presented in this thesis. GSSL assumes that labeled and unlabeled data are embedded on a low-dimensional manifold [34], [35] that can be approximated by a weighted graph. Each instance of the dataset corresponds to a vertex in the weighted graph. The graph's weight w_{ij} measures the similarity of edge e_i and e_j. The graph-based SSL algorithms are divided into three steps

1. Graph-construction

2. Seed label injection
3. Label inferring

Graph-based algorithms are attractive for many reasons. First, graphs are a natural representation of many datasets. Most graph-based SSL algorithms optimize a convex loss function. Unlike TSVMs, graph-based SSL algorithms are scalable [36] and efficient in practice [35].

Conclusion

This chapter has outlined the different machine learning paradigms. The three most common are RL, UL and SL. RL aims for an agent to learn an optimal policy that maximizes a reward function by utilizing a set of states and actions. UL algorithms use unlabeled data for tasks such as clustering and dimensionality reduction.

SL utilizes and labeled training data to find a classifier to label future incoming unlabeled data points. There are two types of SSL, Transductive and Inductive SSL. In transductive learning we are only interested in labeling the given unlabeled samples. Inductive learning has the same goal as supervised classification but includes unlabeled data in the training set. Many SSL algorithms are based on supervised learning. Self-Training algorithms treat supervised predictions as pseudo labels, add them o the training set, and retrain the supervised classifier until satisfactory results are achieved. Co-training and multi-view learning learn using several supervised classifiers. Furthermore, many supervised algorithms such as mixture models and SVMs have their semi-supervised counterparts. Both SL and inductive SSL are usually trained passively. In other words, the unlabeled instances to be labeled are chosen randomly. In AL the instances to be labeled are chosen according to a query strategy and have proven o outperform random sampling in many cases.

Both Active and Semi-Supervised learning are tools to minimize data labeling. The studies in this thesis discuss how different tools and techniques based on AL and SSL can be used in industry and the pros and cons of utilizing these methods.

CHAPTER 3

Research objective and method

This chapter outlines the research objective of this thesis and the research methodology utilized to achieve said objectives. The thesis strives to reach this objective by answering four research questions. The choice of research methods is based on the fact that there is a lack of empirical research studies on labeling within the industry. To address this gap in research, we performed a literature review and mapping study to gather data on the available algorithms, how they were evaluated, and what datasets and datatypes they were evaluated. Second, we performed a case study with the industry to determine the challenges and mitigation strategies. Lastly, we designed benchmark experiments to evaluate the most popular set of algorithms at scale. We utilized Bayesian Data Analysis and Item Response Theory to rank the algorithm according to the optimal accuracy and determine the optimal datasets for evaluating data labeling algorithms. At the end of the chapter, threats to validity are presented.

3.1 Objective

Obtaining labels for supervised classification tasks is expensive financially and in terms of manual labor. The objective of the doctorate research in this thesis is to enable automated data labeling algorithms for industry practitioners, given the challenges that are present within the industry. These challenges involve getting high-quality labeled data to train ML and DL models that achieve high performance. In order to accomplish the research objective, the following research questions are answered.

RQ1: What Data Labeling challenges exist in the industry and how can they be mitigated using Machine Learning algorithms for Data Labeling?

RQ2: What Machine Learning algorithms are available for Data Labeling?

RQ3: What Machine Learning algorithms for Data Labeling are optimal for achieving high accuracy while maintaining low labeling cost?

RQ4: Do benchmark datasets contribute to a fair evaluation of Graph-based Semi-Supervised algorithms?

RQ1: What Data Labeling challenges exist in the industry and how can they be mitigated using Machine Learning algorithms for Data Labeling?

Much time spent in an ML project is allocated to data preparation. Many mitigation strategies are available for the labeling problem. However, there is a lack of research investigating the challenges and mitigation strategies deployed in the industry. Many approaches only work well in some practical scenarios, or they only work in theory or on specific benchmarks and not in real-world applications. How do companies structure their labeling process? This research question strives to fill this gap in research for company-specific applications.

RQ2: What Machine Learning algorithms are available for aiding in Data Labeling?

It is often difficult and costly in terms of time to design algorithms and make them work on industrial data as it requires excellent theoretical knowledge of algorithms and practical knowledge of programming. Adapting domain-knowledge and changing an algorithm in some scenarios might be necessary.

This research question strives to help and guide practitioners in the industry that are new to data labeling algorithms to understand what state-of-the-art algorithms are available for data labeling or to be used to learn with less data. If practitioners need to develop their algorithms, they must know what datasets are available for evaluating their algorithms based on what datatype and from what applications their industrial datasets come.

RQ3: What Machine Learning algorithms for Data Labeling are optimal for achieving high accuracy while maintaining low labeling cost?

Determining whether an algorithm will work well on a particular application and dataset is often difficult and expensive. Furthermore, data labeling can be expensive in terms of time and financial costs. It is, therefore important to learn how different algorithms perform on different datasets of different types and domains through benchmark experiments and case studies to help practitioners choose the algorithm with the highest performance accuracy while reducing manual labeling effort.

RQ4: Do benchmark datasets contribute to a fair evaluation of Graph-based Semi-Supervised algorithms?

Because practitioners in the industry do not have time to analyze and develop algorithms from scratch, they need to learn from benchmark studies to choose the algorithm that will give the highest accuracy for their specific application. However, many industrial datasets lack properties in these benchmark datasets, and practitioners might waste effort trying to implement an algorithm on their dataset only to find that the algorithm only performs well on the benchmark. This research question aims to evaluate what datasets are suitable for evaluating semi-supervised learning algorithms.

3.2 Research Context

The research presented in this thesis was conducted in the context of the Wallenberg AI, Autonomous Systems Program (WASP) and Software Center (SC).

WASP conducts research collaborations with the industry and eight Swedish universities. WASP research is conducted within AI, Autonomous Systems

and Software. Areas strongly affect individuals, society, and industry. The research is focused on solving common challenges to autonomous systems within areas such as *Machine Learning and Knowledge Representation* and *Software Technologies and Methods*.

SC conducts close and long-term research projects in collaboration with academia and industry. The SC research is divided into three applications themes, *Software, Data* and *AI*, and five technology themes *Continuouis Deliviery*, *Continuous Architecture*, *Metrics*, *Customer Data- and Ecosystems* and *AI Engineering*. Paper B reports on a case study conducted in collaboration with an SC company.

3.3 Method

To explore the objectives of this study, we adopted a mixed methods research approach involving a mapping study, case study research in close collaboration with industrial partners and experiments in which simulations were run.

There are many algorithms and publications concerning algorithms for data labeling, but few publications concern how to use how these applications are used for industrial and domain-specific use. To fill the research gap between industrial use and theory, we first performed a literature review a mapping study that outlines the available algorithms that have been available through the ages to give practitioners an idea of how they might adapt these algorithms for their use. We note and present what algorithms are used based on the application and what datasets are used to evaluate the said application. In parallel to the mapping study, we conducted a case study research with companies in the embedded systems domain to identify the challenges and mitigation strategies of the two companies. We formulate new mitigation strategies based on the data gathered from these companies and the data collected from the mapping study. After the aforementioned studies we performed an empirical experiment. Based on the mapping study, we gathered the most commonly used datasets and algorithm types and ran various experiments to answer RQ3 and RQ4. By running simulations on datasets with few labeled instances, we wish to guide and encourage practitioners on how to use and evaluate semi-supervised algorithms on their datasets.

Case Study

In parallel with the mapping study we performed a case study. Case studies are an appropriate research method for investigating how different practices are used in an industrial context [37]. A case study is relevant for studying industry data labeling challenges and mitigation strategies. Paper B reports on data collected from a case study where an internship and semi-structured interviews with practitioners from two companies. The study consists of two phases. First, an Exploration Phase and then a Validation Phase. During the exploration phase we spent time at the office of Company A, 2-3 days/week. The data was collected by observing how data scientists are working with machine learning and labeling. We held discussions with data scientists about datasets. They granted us access to these datasets that we analyzed with the help of Python. Based on the data collected during the first phase, we formulated seven challenges the data scientist faced. Second came the Validation Phase in which we conducted 25-55 minute interviews with four data scientists from company A and one from company B. During the interviews, we asked the data scientists the purpose of their labels, how they got the data annotated, and how they assessed the quality of the labels.

Case study companies

The case study presented in this thesis contains results collected from two companies based in Sweden. At the request of these companies, we have agreed to withhold information about their identities. Thus, only a brief description of the companies is presented here.

Company A is a worldwide telecommunication provider and one of the leading information and Communication Technology (ICT) providers. The company operates in many countries and has 100000 employees. The company has transitioned from agile software development into DevOps in the past few years. Company A started implementing machine learning algorithms into its pipelines but has yet to create a well-defined labeling infrastructure.

Company B is a company specializing in labeling and has nearly 100 employees. They have developed an annotation platform to provide the autonomous vehicles industry with a labeling infrastructure that yields top-quality labeled training data. Company B has many clients from both academia and software companies.

Case study participants

In order to get an insight into how labeling is performed in the industry, we had access to five people from the industry that works with machine learning and know about labeling. Four came from Company A and one from Company B. Their roles were "Data Scientist" and "Senior Data Scientist". The participants had work experience between 2-8 years and the participant from Company B had good knowledge of labeling practices compared to the participants from Company A.

Table 3.1: Overview of participants in the interview study

Company	**Participant Nr**	**Title/Role**	**Experience**
A	I	Data Scientists	4 years
A	II	Senior Data Scientist	8 years
A	III	Data Scientist	3 years
A	IV	Senior Data Scientist	2 years
B	V	Senior Data Scientist	7 years

Systematic Mapping Study

In parallel to the case study we performed a systematic mapping study to understand what data labeling algorithms have been developed through the ages. Systematic Mapping Studies are important tools to find and analyze research in a specific field [38]. Utilizing systematic mapping studies allows one to get a deep insight into algorithms for data labeling and to find research gaps to be filled. This mapping study was conducted to find tools to formulate mitigation strategies for the challenges identified during the case study.

In this study we collected data through Google Scholar, as it is considered an unbiased source [39] and contains papers from arxiv which contains many machine learning papers. To start we looked at introductory textbooks on active learning and semi-supervised learning to get an idea of different categories of active and semi-supervised learning algorithms. We then applied

two different search strings: "semi-supervised learning + <category of semi-supervised learning algorithm" OR active machine learning + <category of active learning>. The word "machine" had to be included when searching for research within active learning so that we did not get studies based on the "active learning" subfield of teaching. The theory of active and semi-supervised learning has existed since the 1980s. However, we expected early papers to be mainly theoretical, containing asymptotic analysis that only works in theory rather than practice. To avoid receiving these papers we searched for papers published between 2000-2020 when simulations had become more common.

Simulation Studies

Papers C and D are simulation studies based on benchmark experiments according to [40]. A benchmark study aims to evaluate how well certain algorithms perform on specific problems and why they fail in certain problems. We have chosen to evaluate graph-based semi-supervised learning algorithms as they are the most commonly used algorithms according to the mapping study[41]. The second reason for choosing these algorithms is that they are all implemented in *GraphLearning* Python package. Thanks to this Python library, practitioners can easily use the algorithms examined in this thesis.

Experimental Setup

We collected a total of 24 of the datasets that we found during our systematic mapping study. These were the ones that were available online and accessible through software packages such as sci-kit learn and TensorFlow. Initially we only used twelve datasets, but as Papar C is an extension of [42] we added twelve more datasets later. The additional 12 algorithms are not contained within Paper D. Similarly the number of already labeled instances was either set to 10% or 50% in Paper D. In Paper C we choose $10\%, 25\%, 50\%, 75\%$, and 90%. We ran each algorithm for ten different random seeds on all datasets using different portions of labeled data. For each iteration we saved the accuracy defined by,

$$\varepsilon = \frac{100}{n-m} \max\left(\sum_{i=1}^{n} I(y_i = \widehat{y}_i) - m, 0 \right),$$

where $I(x)$ is **indicator function** defined as,

$$I(x) = \begin{cases} 1 \text{ if } x \text{ is true} \\ 0 \text{ otherwise} \end{cases},$$

n is the total number of instances and m is the number of labeled instances.

In Paper C, the goal of the benchmark study is to utilize Bayesian Data Analysis to rank the algorithms according to the highest accuracy w.r.t manual effort and to see on what benchmark datasets the algorithms perform optimally. The study aims to determine what algorithm has the optimal trade-off between optimal labeling accuracy and the lowest manual labeling effort. These are relevant questions as they will assist practitioners in choosing the optimal algorithm for labeling.

The benchmark study in Paper D evaluates what datasets are well-suited for evaluating semi-supervised learning algorithms using Item Response Theory (IRT). According to IRT a dataset is not well-suited for evaluating semi-supervised learning algorithms if they all yield the same accuracy. If all algorithms have low accuracy, the datasets are too difficult to learn, and if they all perform very well, they are too easy to learn. Therefore, if the accuracy of the algorithms varies the dataset is well-suited for the evaluation of semi-supervised learning. This experiment is important as it will help practitioners to choose what datasets to include when evaluating a graph-based SSL algorithm. Data processing takes up a lot of a practitioner's time, and the results of this experiment will help practitioners save time on the evaluation of graph-based SSL algorithms.

Summary

We first performed the mapping study in parallel with the case study. Based on the results from the mapping study we could formulate mitigation strategies for the challenges identified during the case study. The most popular algorithms and datasets we found during the mapping study were evaluated in papers C and D, respectively.

Table 3.2 summarizes the research method used during our research. Figure 3.1 illustrates how each paper addresses the Research Questions, and Figure 3.2 illustrates the complete timeline of the Papers included in this thesis.

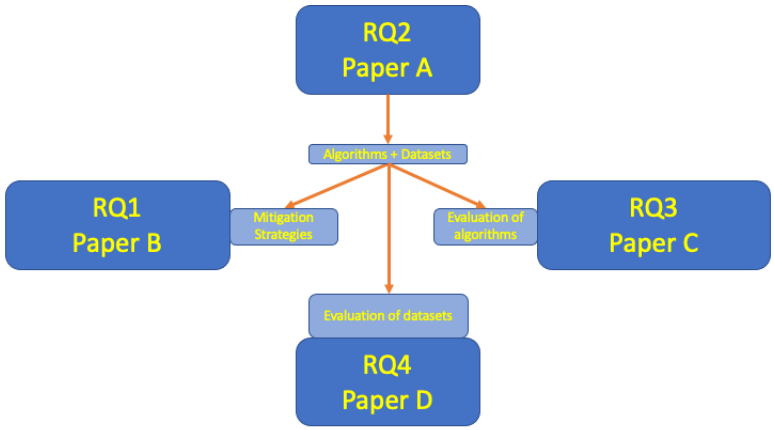
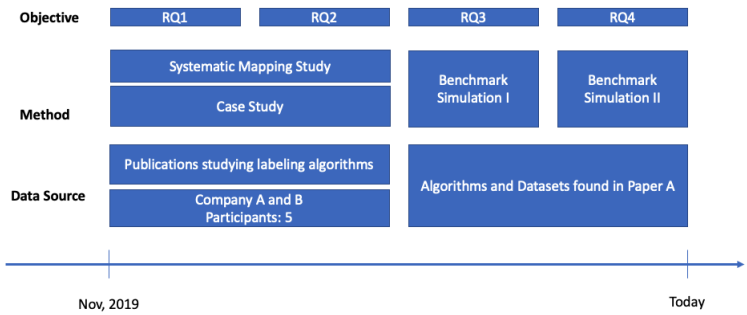


Figure 3.1: An overview of how each paper contributed to the research questions

Paper	Research Strategy	Comments
A	Systematic Mapping Study	Reviewing literature to find algorithms, datasets and application domains.
B	Case Study	Data collected from an internship with Company A and interviews with Company A and B.
C	Benchmark Simulation Study	Benchmark simulations to find optimal algorithms based on three dimensions.
D	Benchmark Simulation Study	Benchmark simulations to investigate what datasets are suitable for the evaluation of GSSL algorithms

Table 3.2: Summary of the research strategies used in the included papers.

Figure 3.2: Overview of the timeline of the research questions, research methods and activities



3.4 Data Analysis

Thematic Analysis

Thematic analysis is a flexible method for identifying, analyzing and reporting patterns within data [43]. In [43], thematic analysis is presented in six phases.

In Phase 1 we get familiar with data to the extent that we immerse ourselves in the data to the extent that the researcher is familiar with the depth and breadth of the content. Data from interviews must be transcribed to perform a thematic analysis. Phase 2 entails finding codes from the data by identifying interesting key features. The coding can be done manually by analyzing text using pens to highlight as many patterns as possible. In phase 3 we start analyzing the codes and investigate how these codes combine to shape a theme. Codes form main themes and others will make sub-themes. Once one has candidate themes and sub-themes, Phase 4 will be to review the themes. Some candidate themes will be discarded, and some will be merged or divided into separate themes. If the themes form a coherent pattern, consider the validity of individual themes in relation to the dataset. Reread the dataset to make sure the themes work w.r.t the dataset and perform re-coding. In Phase 5, the themes are defined and named by identifying the essence of each theme by deciding what viewpoint of the data the theme considers. Be careful not to make a theme too wide, complex, or overlapping. Phase 6 reports results and consists of the final analysis and report writing. Tell the story convincingly

so that the reader can see the validity of the research.

Not that the analysis is not linear, and the phases are to be processed in a recursive process where it is allowed to jump back and forward between the phases. Furthermore, the phases are to be viewed as guidelines rather than rules.

The procedure of Thematic Analysis was utilized in Paper B to identify and analyze patterns in the data collected from the Exploration and Validation phases of the study.

Bayesian Data Analysis

Statistical Inference allows for the interpretation and analysis of the past and future of phenomena using probabilistic modeling. Probability modeling utilizes the uncertainty of given some collected data. A probabilities model consists of data sample x from some underlying distribution $f(x|\theta)$ where theta is an unknown parameter. Given the data, the goal is to provide inference on the parameter theta. The information provided from data is processed using the likelihood function,

$$\ell(\theta|x) = f(x|\theta).$$

In the frequentistic view of statistics, probabilities are expressed in frequencies. However, there are phenomena where events can not be repeatable. To mitigate the issue of non-repeatable events, we infer an uncertainty on the model parameter θ. We call these methods Bayesian statistical models as they utilize the Bayes Theorem to update the distribution of θ when new data is observed. The prior information of θ is known as the *prior distribution*. The updated information is expressed through the *posterior distribution* $\pi(\theta|x)$ calculated with Bayes Theorem,

$$\pi(\theta|x) = \frac{\ell(\theta|x)\pi(\theta)}{\mathcal{P}(x)},$$

where $\mathcal{P}(x)$ is known as the *marginal distribution*,

$$\mathcal{P}(x) = \int \ell(\theta|x)\pi(\theta)\, d\theta.$$

Item-Response Theory

Item Response Theory is a tool utilized in educational measurement to measure students' ability and attitude related to academic achievement using psychological scales [44]. Among other things, IRT can be used to assess whether an exam is suitable for assessing a student's ability. Previously the reliability and validity of test scores were defined in the CTT framework. Consider the test score X as a random variable that can be written as a sum

$$X = T + E = C + S + E,$$

where T is the true score and E is the random error, C reflects the construct being measured by the test, and S is the systematic error. We then define *reliability* ρ [45] as,

$$\rho = \frac{\mathrm{Var}(T)}{\mathrm{Var}(X)},$$

representing the small measurement error compared to the total score variance [45] suggest the definition of validity V as,

$$V = \frac{\mathrm{Var}(C)}{\mathrm{Var}(X)}.$$

Sample dependence and item dependence

Consider two participants A and B who will be evaluated on their mathematics skills. Suppose A takes test X and B takes test Y. Both A and B get a test score of 60. Even if two students get the same score on exams, we cannot tell if one exam is easier than another because the difficulty of the test is based on each participant's skill in mathematics. This property is called *sample(group) dependence*.

Another problem is that test scores depend on the difficulty of test items. If the test items are easy for students, their test scores will be high. This property is called *item(test) dependence*. Because of sample and item dependence, we cannot compare test scores obtained from different examinee groups or those obtained using different test items. Hence sample and item dependence is problematic when we compare examines with various ability levels. CCT cannot solve these issues, which motivates the use of IRT.

IRT is a class of latent variable models that solve the sample and item

dependence problems [46]–[48] and are now considered a practical tool in educational measurement. IRT can simultaneously estimate the examinee item difficulty and the examinee's ability.

Suppose that

$$X_{i,j} = \text{"response of person } j \text{ to item } i \text{ "} \sim \text{Bernoulli}(p_i).$$

In other words $X_{i,j} = 1$ with probability p_i (Person j got the right answer at item i) and $X_{i,j} = 0$ with probability p_i and $X_{i,j} = 1$ with probability $1 - p_i$. (Person j got the wrong answer at item i). Additionally, assume that the responses depend on a *latent(hidden) variable* θ, representing a person's ability level. The objective of IRT is to model the probability parameter p as a function

$$p_i = P(X_{i,j} = 1|\theta).$$

There are different ways to model p_i, such as the 2PL model [47]

$$p_i = P(X_{ij} = 1|\theta) = \frac{\exp(\theta - b_i)}{1 + \exp(a_i(\theta - b_i))}.$$

In the 2PL model a_i is the *discrimination parameter* of item i and b_i is the *difficulty level parameter* of item i.

IRT has previously been utilized for assessing the sustainability of supervised learning datasets [49]. In Paper D, we utilize IRT using the 2PL model and Bayesian analysis to asses how appropriate 12 datasets from Paper C are for evaluating semi-supervised learning algorithms.

3.5 Threats to Validity

We shall discuss validity types for the case study [37] and for the simulations studies [50]. For both types of studies we discuss *Construct Validity*, *External Validity*, and *Internal Validity*. In addition we discuss *Conclusion Validity* for the simulations studies. Construct validity refers to how well the studied operational measures are suited for answering the research questions. External validity refers to how generalized the results are and to what extent the results are transferable to other domains. Internal validity refers to when the researchers fail to account for unexpected factors that affect the results of the investigated factor. Conclusion validity refers to how well-suited the methods

and procedures are to conclude correlations between variables. It involves the use of inappropriate assumptions and statistical tests and an insufficient number of iterations.

Case Study

Construct validity was achieved by preparing the interview participants before conducting the interviews. These preparations consisted of a lecture where we introduced the participants to definitions and concepts of data labeling techniques. Additionally, we sent out an email containing sample questions so that the participants reflect and ask questions in order for us to address any concerns they had before the interview was conducted. We achieved external validity by formulating each research question so they were not domain-specific. Internal validity was achieved through data triangulation. Data Triangulation combines data from different sources to study a social phenomenon [51]. In the case study context we utilize data triangulation to corroborate and identify weaknesses in the data collected from different people. Thereby increasing the validity of results and strengthening our conclusions as every participant in the interviews was experienced with labeling. Hence, we received all relevant information during the data collection procedure.

Simulation Study

In our studies, the algorithms used in these studies have been previously validated, and the experiments are well-defined according to the needs of the industry. GSSL algorithm does not impose any assumption on the probability distribution. For the BDA, we performed reliable tests such as model checking and inspections of parameters and indicators to ensure the validity of the results [52]. To consider the uncertainty of stochastic simulation, we ran ten iterations of each. All experiments are replicable as the implementation software is available online [53]. Internal Validity was accounted for by including every available factor thought to be relevant, and we did not exclude any. No simplification in the simulation model was done to force desired outcomes, and we included many diverse datasets with the same model parameters. External Validity was also achieved as we used many different datasets for each category. Therefore the results should be generalized to other datasets as well.

CHAPTER 4

Summary of included papers

This chapter provides a summary of the included papers.

4.1 Paper A

Teodor Fredriksson, David Issa Mattos, Jan Bosch, Helena Holmström Olsson
Machine Learning Algorithms for Labeling: Where and How They are Used?
 DOI: 10.1109/APSEC53868.2021.00031 .

SL is the most common machine learning algorithm paradigm and requires labeled data. Data Labeling is a technique for annotating data depending on the contents of the data [54]. It is an important step in the data preparation step as much data from the industry is unlabeled. Much of Data Labeling in software-intensive companies in the online and embedded system's domain is still performed manually [54]. Manual labeling is sometimes preferred as it allows for easy maintenance and data quality checks. The downside of manual Data Labeling is that it is costly in time and manual effort. SSL and AL are

two types of algorithms that reduce manual labeling. AL algorithms utilize query strategies to query an oracle to label more instances based on their informativeness, thus reducing manual effort. SSL algorithms use labeled and unlabeled instances to train a classifier and can eliminate manual effort to some extent or completely. Building a labeling infrastructure around active and SSL can be time-consuming as you might have to test many algorithms until finding a suitable one. We performed a systematic mapping study to identify state-of-the-art semi-supervised and active learning algorithms. We identify algorithms, the datasets used for evaluating the algorithms, and application domains. Based on the results we created a taxonomy for the algorithms and a classification scheme for the datasets based on datatype and application domain. The contributions of this study will provide guidelines for practitioners in industry and academia to find appropriate active and SSL algorithms for their particular use-case and the most common datasets to evaluate their algorithms.

4.2 Paper B

Teodor Fredriksson, David Issa Mattos, Jan Bosch, Helena Holmström Olsson
Data Labeling: An Empirical Investigation into Industrial Challenges and Mitigation Strategies
 DOI: 10.1109/APSEC53868.2021.00031 .

A large portion of a Data Scientist's job in a machine learning project is dedicated to preparing data in which data labeling can be a key component. Data Scientists may have the strongest domain-knowledge but are often busy with other specialized tasks, such as using software such as Python and SQL to extract data from databases and build ML models in Python. Third-party labeling services such as Amazon Mechanical Turk are available, but many companies can not use them as their data might contain confidential information. The risks of getting mislabeled data can increase with crowdsourcing as the labelers might lack the required domain-knowledge. Therefore many companies still utilize in-house labeling. Much research has been conducted within crowdsourcing and machine learning to overcome the data quality issue. To the best of our knowledge, there is a lack of research that investigates the challenges and mitigation strategies present in the industry. We focus mainly

on applications where labeling is non-trivial and requires a level of domain-knowledge. We collected data by performing semi-structured interviews with practitioners from two companies and an internship with one of the companies. Based on the collected data we identified key challenges companies face when labeling data and mitigation strategies the companies employ to address some or all of the challenges. We also present new strategies to improve and complement old strategies.

4.3 Paper C

Teodor Fredriksson, David Issa Mattos, Jan Bosch, Helena Holmström Olsson

An Empirical Evaluation of Graph-Based Semi-Supervised Learning Algorithms

 DOI: 10.1109/APSEC53868.2021.00031 .

SL is the most common learning paradigm and requires labeled data. The accuracy of ML and DL models is strongly correlated to the size of the labeled training set. In many cases, the more labeled data available, the better. Many companies cannot utilize crowdsourcing as data might be confidential. In-house labeling might not be an option as it is difficult to allocate manual labeling to personnel with the required time and domain knowledge. Because of the aforementioned reasons in-house labeling is not always attractive and companies would rather rely on more automated approaches. According to previous research SSL is a popular ML algorithm when labeled data is scarce. SSL has been used on several datasets, including image, video, text, sound and other numerical datasets. Choosing the correct algorithm for a specific application and dataset can be difficult and time-consuming. According to previous studies GSSL algorithms are the most popular to use and there are many to choose from. This paper ranks the optimal GSSL semi-supervised-learning algorithms using the Bayesian Bradley-Terry Model on data collected from a benchmark simulation study. In the study we evaluated 13 different GSSL algorithms for 24 datasets that are the most commonly used to evaluate GSSL algorithms. The algorithms are evaluated based on three dimensions. *Performance:* to measure how well the algorithms predict labels and calculate the probability of an algorithm reaching an accuracy above 90%. *Effort:* to measure how many labeled instances are required to achieve the optimal

performance. The last dimension is *Datatype:* To examine if the algorithms perform better on a particular type of data.

4.4 Paper D

Teodor Fredriksson, David Issa Mattos, Jan Bosch, Helena Holmström Olsson
Assessing the Suitability of Semi-Supervised Learning Datasets using Item Response Theory
 DOI: 10.1109/APSEC53868.2021.00031 .

Paper A reports on the most commonly used labeling algorithms and the datasets associated with evaluating algorithms. Paper B provides a taxonomy of challenges in the industry and a mitigation strategy for each challenge. The mitigation strategies are based on practices and algorithms found in Paper A. In Paper A, we found that Graph-based algorithms are the most popular algorithms, so we ran simulations using many such algorithms on many of the datasets found in Paper A. Based on the results of Paper C, we observed that many algorithms had a high performance on many of the datasets. We started to question if the datasets used to evaluate these algorithms are suitable for evaluating SSL algorithms. Based on the simulation results of Paper C, we utilized a Bayes congeneric item response theory model to asses how suitable the datasets are for evaluating GSSL algorithms.

CHAPTER 5

Machine Learning Algorithms for Labeling: Where and How They are Used?

5.1 Introduction

In software-intensive companies in the online and the embedded systems domain, vast sets of data are being processed and labeled manually [54]. Manual labeling is an expensive approach for a company, but it allows easy maintenance of the data's quality. One of the downsides is that the task is tedious and time-consuming for highly qualified professionals, often leading to prohibitively expensive costs. Data labeling is a way of annotating data depending on the data's content [54]. The labels each data instance receives are decided after information about the entry has been processed. Research in machine learning and artificial intelligence led to the development of multiple algorithms for fully automating (semi-supervised learning) or assisting humans in labeling the data (active learning).

Semi-supervised learning is a set of machine learning algorithms used when the majority of instances are unlabeled. The semi-supervised classification objective is to train a classifier on both unlabeled and labeled data. This

classifier is used to label the instances of the dataset that are not labeled [55].

Active learning is a machine learning framework that utilizes several labeled instances to query an oracle (often a human) to label some desired instances. Active learning is used to assist humans in selecting a smaller subset of the best instances to label [12].

In this paper, we conducted a systematic mapping study [38] to identify the state-of-the-art literature on machine learning algorithms that are used for assisted or automatically labeling and where they are used.

This paper provides three main contributions. First, we identify the machine learning algorithms for labeling. We present a taxonomy of the algorithms. Second, we identify the datasets that are used to evaluate the algorithms. We create a classification scheme for the datasets based on the type of data and the application area. Third, we present guidance to industry practitioners on optimally getting their data labeled and using labeled datasets for machine learning and data labeling practices. The results presented in this paper can be used by both researchers and practitioners to find missing labels with the aid of machine algorithms or to select appropriated datasets to compare new state-of-the-art algorithms in their respective application areas.

The remainder of this paper is organized as follows. Section 5.2 provides an overview of related work and presents key concepts used in semi-supervised and active machine learning algorithms. In section 5.4 we provide a concise description of the problem that we seek to address in this paper, followed by an overview of our research method. We present the systematic literature mapping results in section 5.5 and discuss these in section 5.6. Finally, we end the paper with an overview of open research questions and a conclusion in section 5.7.

Many studies were included in this paper. To preserve space, the references to the papers containing the algorithms and datasets were moved from the paper into a separate reference list that uses a different referencing style. The list of references for algorithms and datasets can be found in an online appendix: https://github.com/teodorf-bit/Systematic-Mapping-Study/blob/main/SMS_SYSCON.pdf.

5.2 Background

Supervised learning is used for classification and regression tasks. Supervised classification requires that each instance in a dataset is associated with a label. In practice, this becomes a problem as they have large amounts of data, but the data is often incomplete as labels are missing partially or entirely. If labels are available, these can still be of lousy quality, affecting a model's performance. Hence, acquiring labels of high quality is paramount to train high-accuracy models.

The most popular strategy for achieving labels with human supervision is crowdsourcing [56]. However, a problem with crowdsourcing is that it requires third-party companies to access sensitive and confidential data. Another problem is when the label distribution is skewed. Imbalanced label distributions tend to make machine learning perform poorly [57].

The distribution independent model of concept learning (supervised learning), known as PAC-learning, a framework for mathematical analysis machine learning, was introduced by Leslie Valiant in [20]. The PAC learning framework was later extended for semi-supervised learning in [55].

The survey [58] introduce researchers and practitioners to the main semi-supervised learning algorithm, such as Graph-based algorithms, Mixture models and EM, self-training, co-training, and multi-view learning. Surveys on active learning such as [59] introduce the three main approaches to active learning. Membership queries, as well as pool-based and stream-based active learning. Furthermore, they define query strategies such as uncertainty sampling, query-by committee, error reduction, and variance strategies. Issues with regards to skewed label distributions, unreliable oracles, and costs associated with labeling are discussed as well.

Semi-supervised learning

Semi-supervised learning is a set of machine learning algorithms that can be used if most instances are unlabeled, but a small subset of them has labels. In technical terms, we have access to a set of data points that can be divided into two disjoint subsets, one containing the labeled instances and the other containing the unlabeled instances. The objective of semi-supervised classification is to train a classifier on both unlabeled and labeled data so that it is better than a supervised classifier trained only on the labeled data.

Co-training and multi-view learning Co-training was first used to classify web pages [60], [61], and it was shown that co-training could improve classification accuracy. Furthermore, a PAC generalization bound was created in [62]. Co-training assumes that the features can be divided into two sets. We say that the features have at least two views. The corresponding labels are then predicted using both of the views using the co-training algorithm. The algorithm takes both labeled and unlabeled data as input, as well as a learning speed k. The goal is to train two classifiers. The first classifier is trained on view 1 and the second classifier on view 2. Furthermore, we must assume that the classifiers alone have a high classification accuracy and that the two views must be conditionally independent given the class label. Both training sets consist initially of the same labeled data, $L_1 = L_2$. Then train the first classifier on L_1 and the second on L_2. Classify the remaining unlabeled data with classifier one and classifier two separately. Take the k most-confident predictions of classifier 1 and add them to L_2 and k most-confident predictions of the second classifier and add them to L_1. Remove these instances from the unlabeled data. Repeat this procedure until we are out of unlabeled data. Multi-view learning was first used in [63] and generalized co-training to n number of views.

Mixture models and the EM-algorithm A (Generative) Mixture Model (MM) is a weighted sum of densities from M components

In the supervised setting, the parameters of the densities are calculated using maximum likelihood estimation (MLE). Using MLE in the supervised setting is straightforward. However, in the semi-supervised setting, we must utilize the unlabeled data as well. Hence, we have to solve a different optimization problem to find the parameters of the densities.

The missing labels are referred to as hidden variables and make the log-likelihood difficult to optimize. To optimize the log-likelihood, the Expectation-Maximization (EM) [64] algorithm is used.

Theoretical aspects of mixture models have been justified in [65], [66]. The only assumption needed for mixture models is that data comes from a mixture model. This assumption is difficult to assess if the labels are scarce. However, it is usually assessed by domain knowledge or mathematical convenience. If the assumption is violated, then the unlabeled data will worsen the accuracy of the predicted labels [67]. Another issue with generative mixture models is

that the model that describes the unlabeled data must be unique.

Semi-supervised Support Vector Machines (S3VM) models Supervised support vector machines (SVM) strive to classify instances by creating a decision boundary. Such a decision boundary is found by solving an optimization problem. If we have unlabeled instances, then there is no way of knowing whether the unlabeled instance is put on the right side of the decision boundary. S3VM strives to mitigate this issue by incorporating a loss function on the unlabeled data to the SVM objective function. Furthermore, S3VM assumes that there is a low-density region that separates the labels. If such a region does not exist, S3VMs might not perform as well as expected. S3VMs or TSVM (Transductive SVMs), as these were originally called, were first introduced in [68].

Graph-based models Graph-based methods construct graphs from the training data that consist of labeled and unlabeled instances. These instances constitute the graphs' vertices, which means that the more unlabeled data, the bigger the graph will become. The learning procedure will then result in labels assigned to the vertices. There are two types of graph constructing algorithms.

Task-independent algorithms do not use labeled data and are hence unsupervised. Popular methods include k-NN,ϵ-neighborhood, b-matching [69] and, hard and soft α-graphs [70].

Task-dependent algorithms do use labeled data such as Inference-drive Metric Learning [71] and Kernel-alignment based spectral kernel design [72].

The next step is to inject seeds and infer labels on the unlabeled data. These algorithms are divided into Transductive and Inductive methods. The goal of transductive algorithms is to predict labels only for the unlabeled data. These algorithms include Graph cut [73], Gaussian random fields (GRF)[74], Local and Global consistency (LGC)[75], Adsorption [76], Modified Adsorption (MAD)[77], Quadratic Criteria (QC) [78], Transduction and Confidence (TACO) [79], Information Regularization [80]–[82] and Measurement Propagation (MP)[35]. Inductive learning estimates a function that can be applied to new data instances. Inductive algorithms are few [56], an example of an inductive algorithm is Manifold regularization. [83].

Active learning

Historically machine learning algorithms usually try to fit a model according to currently labeled data, and we refer to these models as "passive" learning models. Active learning systems, on the other hand, create new models as they iterative learn. Similar to how a scientist plans several experiments to conclude a hypothesis, an active learning method imposes query strategies to help select the most informative examples to be labeled by an oracle.

In some cases, an active learning system might not be optimal if the model does not require a considerable number of labels. Instead, use it when there is a massive set of unlabeled examples, and there is a need to label a massive amount of data to train the system.

If active learning is appropriate, then we need to specify in what way we want to query the examples [12]. The three most common scenarios are:

1. *Query synthesis:* This scenario allows the learner to request labels for any unlabeled example. Query synthesis also works for instances the learner itself has generated. All that is required is knowledge about how the inputs are constructed. Query synthesis is sometimes helpful, but in some cases, it is not reasonable to use. For example, in natural language processing, one might generate a text string that is in-comprehensive [12].

2. *Stream-based selective sampling:* Also known as stream-based sampling, this scenario involves sampling one sample at a time from the actual distribution, and then the learner should decide whether to query it or not[12].

3. *Pool-based sampling:* In many scenarios, an extensive set of unlabeled data must be processed at once, and this is where pool-based sampling is appropriate. The scenario involves having a large set of unlabeled examples and a small pool of labeled examples as well[12].

Uncertainty sampling and Density weighted methods models The uncertainty sampling approach [84] aims to select the instances that we are least certain about and label these. Labels that we are certain about will probably not contribute to informativeness. Two common strategies include *entropy* [85] and *least confident*[86]. These strategies are among the most popular

and work especially well for probabilistic models [86]–[90] but has also been successfully applied to non-probabilistic models [86], [91]–[94]. The downside is that they only consider information about the one prediction and not the entire distribution. Density-weighted methods strive to model the input distribution in the query strategy. Thus, not only querying by uncertainty but also by how representative each instance is. *Information density*[87] does exactly that, other similar strategies have also been proposed [91], [95]–[97].

Query-based models Query by Committee [98] involves a committee of classifiers. Each classifier is trained on the same training data. The *Version Space* is the set of classifiers that are consistent with the labeled training data. The smaller the version space is, the more confident we are about the version space classifiers. Therefore, a smaller version space means that we do not need to test many different classifiers to find the most accurate model. The goal of QBC is to minimize the version space. To produce a QBC algorithm, we first have to construct the committee of models representing the entire version space. Secondly, we need a measurement to determine disagreement between the committee members. Constructing the committee can be done by sampling a committee of random hypotheses [98]. When using generative models, this can be done by sampling models from some posterior distribution [23], [95]. For discriminate and non-probabilistic models, query-by-boosting and query-by-bagging are proposed [99]. Two common measurements for disagreement between committee members are *vote entropy*[23] and *KL divergence*[95].

In practice, QBC is relatively simple to implement and works with any basic model. The downsides are that these are difficult to maintain, and just like uncertainty sampling, it only looks at one instance at a time and does not consider the entire distribution.

5.3 Algorithms

This section describes the main sub-categories of algorithms based on the two categories: semi-supervised and active learning.

Semi-supervised learning

Semi-supervised learning is a set of machine learning algorithms that can be used if most instances are unlabeled, but a small subset of them has labels.

In technical terms, we have access to a set of data points that can be divided into two disjoint subsets, one containing the labeled instances and the other containing the unlabeled instances. The objective of semi-supervised classification is to train a classifier on both unlabeled and labeled data so that it is better than a supervised classifier trained only on the labeled data.

Co-training and multi-view learning Co-training was first used to classify web pages [60], [61], and it was shown that co-training could improve classification accuracy. Furthermore, a PAC generalization bound was created in [62]. Co-training assumes that the features can be divided into two sets. We say that the features have at least two views. The corresponding labels are then predicted using both of the views using the co-training Arithm. The Arithm takes both labeled and unlabeled data as input, as well as a learning speed k. The goal is to train two classifiers. The first classifier is trained on view 1 and the second classifier on view 2. Furthermore, we must assume that the classifiers alone have a high classification accuracy and that the two views must be conditionally independent given the class label. Both training sets consist initially of the same labeled data, $L_1 = L_2$. Then train the first classifier on L_1 and the second on L_2. Classify the remaining unlabeled data with classifier one and classifier two separately. Take the k most-confident predictions of classifier 1 and add them to L_2 and k most-confident predictions of the second classifier and add them to L_1. Remove these instances from the unlabeled data. Repeat this procedure until we are out of unlabeled data. Multi-view learning was first used in [63] and generalized co-training to n number of views.

Mixture models and the EM-Arithm A (Generative) Mixture Model (MM) is a weighted sum of M component densities.

In the supervised setting, the parameters of the densities are calculated using maximum likelihood estimation (MLE). Using MLE in the supervised setting is straightforward. However, in the semi-supervised setting, we must utilize the unlabeled data as well. Hence, we have to solve a different optimization problem to find the parameters of the densities.

The missing labels are referred to as hidden variables and make the log-likelihood difficult to optimize. To optimize the log-likelihood, the Expectation-Maximization (EM) [64] Arithm is used.

Theoretical aspects of mixture models have been justified in [65], [66]. The only assumption needed for mixture models is that data comes from a mixture model. This assumption is difficult to assess if the labels are scarce. However, it is usually assessed by domain knowledge or mathematical convenience. If the assumption is violated, then the unlabeled data will worsen the accuracy of the predicted labels [67]. Another issue with generative mixture models is that the model that describes the unlabeled data must be unique.

Semi-supervised Support Vector Machines (S3VM) models Supervised support vector machines (SVM) strive to classify instances by creating a decision boundary. Such a decision boundary is found by solving an optimization problem. If we have unlabeled instances, then there is no way of knowing whether the unlabeled instance is put on the right side of the decision boundary. S3VM strives to mitigate this issue by incorporating a loss function on the unlabeled data to the SVM objective function. Furthermore, S3VM assumes that there is a low-density region that separates the labels. If such a region does not exist, S3VMs might not perform as well as expected. S3VMs or TSVM (Transductive SVMs), as these were originally called, were first introduced in [68].

Graph-based models Graph-based methods construct graphs from the training data that consist of labeled and unlabeled instances. These instances constitute the graphs' vertices, which means that the more unlabeled data, the bigger the graph will become. The learning procedure will then result in labels assigned to the vertices. There are two types of graph constructing algorithms.

Task-independent algorithms do not use labeled data and are hence unsupervised. Popular methods include k-NN,ϵ-neighborhood, b-matching [69] and, hard and soft α-graphs [70].

Task-dependent algorithms do use labeled data such as Inference-drive Metric Learning [71] and Kernel-alignment based spectral kernel design [72].

The next step is to inject seeds and infer labels on the unlabeled data. These algorithms are divided into Transductive and Inductive methods. The goal of transductive algorithms is to predict labels only for the unlabeled data. These algorithms include Graph cut [73], Gaussian random fields (GRF)[74], Local and Global consistency (LGC)[75], Adsorption [76], Modified Adsorp-

tion (MAD)[77], Quadratic Criteria (QC) [78], Transduction and Confidence (TACO) [79], Information Regularization [80]–[82] and Measurement Propagation (MP)[35]. Inductive learning estimates a function that can be applied to new data instances. Inductive algorithms are few [56], an example of an inductive Arithm is Manifold regularization. [83].

Active learning

Historically machine learning algorithms usually try to fit a model according to currently labeled data, and we refer to these models as "passive" learning models. Active learning systems, on the other hand, create new models as they iterative learn. Similar to how a scientist plans several experiments to conclude a hypothesis, an active learning method imposes query strategies to help select the most informative examples to be labeled by an oracle.

In some cases, e.g., an active learning system might not be optimal if the model does not require a considerable number of labels. Instead, use it when there is a massive set of unlabeled examples, and there is a need to label a massive amount of data to train the system.

If active learning is appropriate, then we need to specify in what way we want to query the examples [12]. The three most common scenarios are:

1. *Query synthesis:* This scenario allows the learner to request labels for any unlabeled example. Query synthesis also works for instances the learner itself has generated. All that is required is knowledge about how the inputs are constructed. Query synthesis is sometimes helpful, but in some cases, it is not reasonable to use. For example, in natural language processing, one might generate a text string that is in-comprehensive [12].

2. *Stream-based selective sampling:* Also known as stream-based sampling, this scenario involves sampling one sample at a time from the actual distribution, and then the learner should decide whether to query it or not[12].

3. *Pool-based sampling:* In many scenarios, an extensive set of unlabeled data must be processed at once, and this is where pool-based sampling is appropriate. The scenario involves having a large set of unlabeled examples and a small pool of labeled examples as well[12].

Uncertainty sampling and Density weighted methods models The uncertainty sampling approach [84] aims to select the instances that we are least certain about and label these. Labels that we are certain about will probably not contribute to informativeness. Two common strategies include *entropy* [85] and *least confident*[86]. These strategies are among the most popular and work especially well for probabilistic models [86]–[90] but has also been successfully applied to non-probabilistic models [86], [91]–[94]. The downside is that they only consider information about the one prediction and not the entire distribution. Density-weighted methods strive to model the input distribution in the query strategy. Thus, not only querying by uncertainty but also by how representative each instance is. *Information density*[87] does exactly that, other similar strategies have also been proposed [91], [95]–[97].

Query-based models Query by Committee [98] involves a committee of classifiers. Each classifier is trained on the same training data. The *Version Space* is the set of classifiers that are consistent with the labeled training data. The smaller the version space is, the more confident we are about the version space classifiers. Therefore, a smaller version space means that we do not need to test many different classifiers to find the most accurate model. The goal of QBC is to minimize the version space. To produce a QBC Arithm, we first have to construct the committee of models representing the entire version space. Secondly, we need a measurement to determine disagreement between the committee members. Constructing the committee can be done by sampling a committee of random hypotheses [98]. When using generative models, this can be done by sampling models from some posterior distribution [23], [95]. For discriminate and non-probabilistic models, query-by-boosting and query-by-bagging are proposed [99]. Two common measurements for disagreement between committee members are *vote entropy*[23] and *KL divergence*[95].

In practice, QBC is relatively simple to implement and works with any basic model. The downsides are that these are difficult to maintain, and just like uncertainty sampling, it only looks at one instance at a time and does not consider the entire distribution.

5.4 Research Method

In order to reach the objectives of this paper, we conducted a systematic mapping study. Systematic mapping studies seek to identify, analyze and interpret all relevant research on a particular topic [38]. In this study, the topic of interest is automatic labeling in machine learning, and thus the purpose of this SMS is to find and analyze relevant literature on automatic labeling. We conducted this systematic mapping study according to [100]. The procedure consists of four steps:

1. Definition of research questions.
2. Identification of search terms and searching for papers.
3. Screen of papers based on inclusion and exclusion criteria.
4. Data extraction and mapping.

We detail each step below.

Definition of research questions

The purpose of this paper is to provide a systematic overview of the existing research on automatic labeling of data using machine learning algorithms. This paper aims to examine previous research and explore the possibility of contributing to new research.

Three research questions are defined below:

RQ1: What types of machine learning algorithms are used for assisted or automatically labeling?

RQ2: What are the datasets used to evaluate these algorithms?

RQ3: What algorithm(s) should be used based on application?

Identification of search terms and conducting search

To source relevant studies, we utilized a keyword-based database search. The main search string was constructed iteratively. At first, we used keywords such as "automatic labeling," but it gave a too large variety of algorithms specific to a particular type of application. We then changed the keywords to

methods based on active machine learning and semi-supervised learning since these algorithms are applied to various applications.

To find papers based on active learning, we searched for "active machine learning" + "<*CATEGORY OF ACTIVE LEARNING*>". If we dismissed the "machine" in the string, we would get results related to "education". Similarly, for semi-supervised learning," we searched for "semi-supervised learning " + "<*CATEGORY OF SEMI-SUPERVISED LEARNING*>". The active learning categories found were Uncertainty, Query by committee, Error and Variance Reduction, and Density-weighted algorithms. The semi-supervised categories found were Co-training and Multi-view learning, EM-algorithms and Mixture models, Semi-supervised Support Vector Machines, and Graph-based semi-supervised learning.

We used Google Scholar (https://scholar.google.com) as the source where we applied our search string. There are three reasons as to why. The first reason is that we expected many relevant articles from the search because the search terms are so general. Secondly, Google Scholar is perceived as an unbiased source [39]. The third reason is that Google scholar includes papers from arxiv, where many research papers on machine learning are submitted.

The theoretical considerations of machine learning started in the 1980s and the first computations on machine learning algorithms did not start until the 2000s. Therefore all papers between 1980 and 1999 should concern theoretical aspect of machine learning and are unnecessary to include in our study. Hence, only paper between 2000 and 2020 will be included.

The search strings were applied in December 2020 to the selected electronic database to retrieve articles that include the keywords in their title, abstracts, and instructions. The retrieval stopped after the abstracts and introductions became less relevant to avoid an infinite number of papers. In the end, 312 articles were retrieved for further screening and processing of inclusion and exclusion criteria.

Screening of papers on the basis of inclusion and exclusion criteria.

All retrieved studies were examined for inclusion and exclusion based on pre-established criteria. The exclusion and inclusion criteria considered in our study are presented below:

Inclusion Criteria

- Papers that includes AL/SSL techniques for labeling unlabeled and or partially unlabeled data form the industry.
- Papers that compare several AL/SSL techniques with each other.
- Papers that include a hybrid between AL/SSL learning.
- Papers that compare AL/SSL techniques with other non-AL/SSL methods.
- Papers that has a title that describes the application.

Exclusion Criteria

- Papers concerning theoretical proofs of AL/SSL methods.
- Papers concerning simulation studies.
- Absence of industrial validation.

Data Extraction and Analysis

Data extraction involved the collection of information related to the RQs of the study. For each paper, we identified the research field, what kind of datatypes, and what method the paper focused on.

5.5 Results

In this section, we provide the results of the systematic mapping study concerning the research questions proposed in section 5.4. First, we provide a list of the algorithms we found from analyzing the papers. Second, we provide a classification of the datasets used in the algorithm evaluation. Finally, we present what type of algorithm should be used for specific applications.

RQ1: What types of machine learning algorithms are used for assisted for autimatic labeling?

Co-training and multi-view learning The following were algorithms found: Semi-supervised discrete hash model (SSNDH) [101], Ensemble co-training

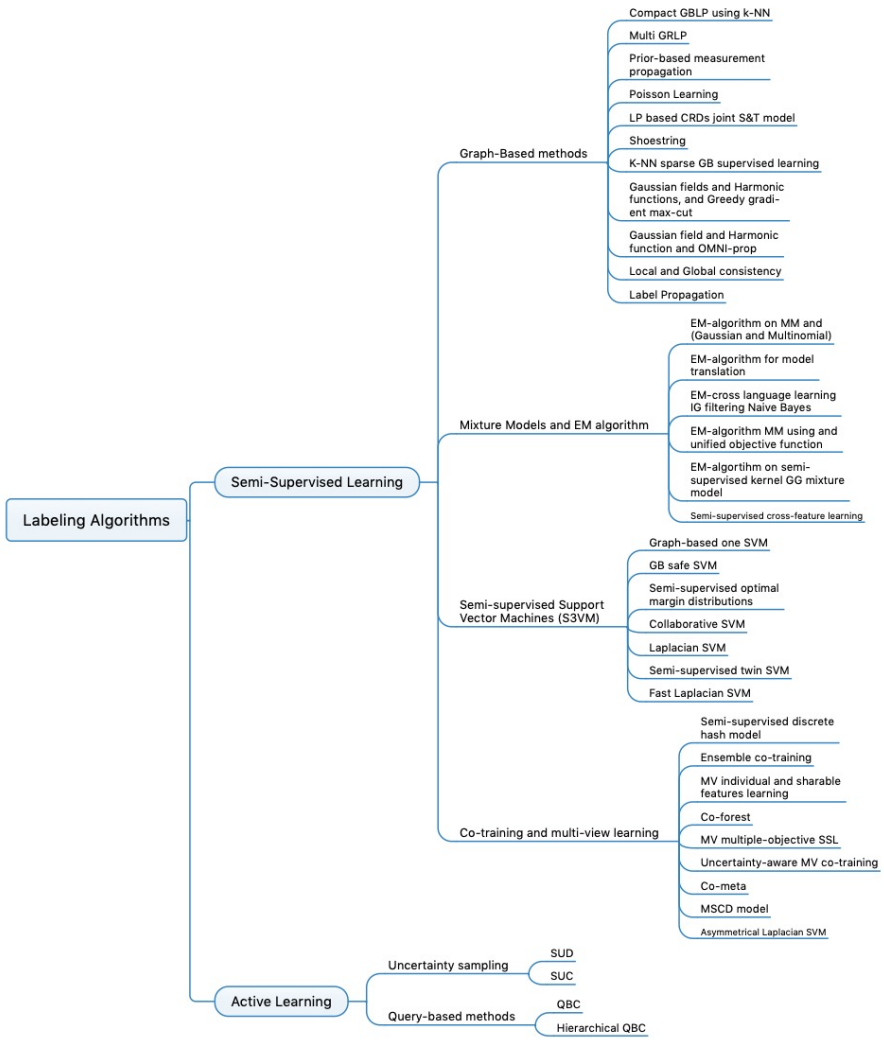


Figure 5.1: Tree diagram illustrating the different types of algorithms in our taxonomy.

(En-Co-training) [102], Multi-view individual and shareable features learning [103], Co-forest [104], Multi-view multiple-objective SSL [105], Uncertainty-aware multi-view co-training [106], Co-meta [107], the MSCD model [108] and Semi-supervised cross-feature learning [109].

Mixture models and the EM-algorithm The following algorithms were found in the literature: EM-algorithms on mixture modes (Gaussian, Multinomial), [110], [111], EM-algorithm for model translation [112], EM-cross language learning using IG filering Naive Bayes [112], EM-algorithm on mixture model using an unified objective function [113], EM-algorithm on semi-supervised kernel GG mixture model [114].

Semi-Supervised Support Vector Machines (S3VM) models We found the following algorithms: Graph-based one class support vector machine (OC-SSVM)[115], Graph-based safe support vector machines [116], Semi-supervised optimal margin distributions (ssODM)[117], Collaborative Support Vector Machines (ColSVM) [118], Laplacian Support Vector Machine (LapSVM) [119], Semi-Supervised Twin Support Vector Machine (TWSVM) [120], Fast Laplacian twin Support Vector Machines (FLapTWSVM)[121] and Asymmetric Laplacian Support Vector Machines (AsyLapSVM)[122].

Graph-based models The following algorithms were found: Compact graph-based label propagation using k-NN [123], Multi graph-based label propagation [124], k-NN sparse graph-based supervised learning [125], Prior-base measurement propagation (pMP)[126], Label propagation based CRFs joint S&T model [127], Spectral graph transducer and Gaussian fields [128], Poisson learning [129], Shoestring [130], Sentiment value propagation [131], Label Propagation [126], Local and Global consistency [132], [133], Gaussian fields and Harmonic functions [128], [133], and OMNI-Prop [132], Greedy gradient max-cut [133], ntegrated Graph-based Semi-supervised Multiple/SingleInstance Learning [134].

Active learning

Uncertainty sampling and Density weighted methods models The algorithms that we identified were, Sampling by uncertainty and density (SUD) and Sampling by clustering (SBC) [135].

Table 5.1: Categories for datasets

Datatype	Name	Application	Reference
Image Input	Brazilian Coffee Scenes	Image classification	[132]
	Cats vs Dogs	Cats or dogs classification	[132]
	CIFAR10	Image classification	[101],[129]
	COREL	Image classification, Image retreival	[124],[94]
	CORAL	Image classification	[134]
	Coil-20	Image classification	[122], [116]
	Coil-100	Image classification	[123]
	CMU	face recognition	[116]
	Diabetic Retinopath	Image segmentation, Image sequence recognition	[114]
	Digestive-Tract-Cancer	Image segmentation, Image sequence recognition	[114]
	Digit	Image classification	[123]
	ETH80	Image classification	[123]
	EMNIST	Handwritten digits classification	[132]
	FashionMNIST	Image classification	[129], [132]
	Flickr	Image annotation	[125]
	LiTS (Liver Tumour dataset)	Medical image segmentation	[106]
	MNIST	Handwritten digits classification, Image segmentation, Image sequence recognition	[114],[123],[129]
	MF (Multiple Features)	Image classification	[122]
	miniImageNet	Image classification	[130]
	NUS-WIDE-OBJECT	Image classification	[122]
	Scene	Face recognition	[122]
	tieredImageNet	Image classification	[130]
	TRECVID	Semantic concept detection in videos	[109]
	USPS	Image classification	[122],[123],[116],[119]
	UMIST	Face recognition	[116]
	WIKI	Image classification	[101]
	YaleB	Face recognition	[122], [116]
Text Inputs	AD	Webpage classification	[103]
	Comp2	Text classification	[135]
	DBWorld	E-mail classification	[119]
	Interest	Word Sense Disambiguation	[135]
	Reuters	Text classification	[136]
	Spambase	Email spam classification	[119]
	WebKB	Text classification	[135]
	Yahoo RSS News	Cross language text classification	[112]
	CTB-7	Part-of-speech tagging	[127]
	MSR	Part-of-speech tagging	[127]
	Wall Street Journal text from PTB	Part-of-speech tagging	[137]
	English poetry from BNC	Part-of-speech tagging	[137]
	Universal Dependency 2.3	Part-of-speech tagging	[107]
Sound inputs	TIMIT	Phone and segment classification, Phonetic classification	[126],[113]
Numeric Inputs	Bupa	Liver-disorder classification	[121], [123]
	Adult	Salary over 50k/yr classification	[119]
	Austra	credit lard application classification	[117],[119]
	Banknote	authentication of banknotes	[119]
	Cellcycle	Optimal caching in edge networks	[138]
	DFT database	Power factor prediction in diamond-like thermo	[139]
	Derisi	Optimal caching in edge networks	[138]
	Echodiagram	heart attack survival classification	[117]
	ECG5000	Dynamic network classification	[108]
	ECGFiveDay	Dynamic network classification	[108]
	Eisen	Optimal caching in edge networks	[138]
	Enron	Optimal caching in edge networks	[138]
	Expr	Optimal caching in edge networks	[138]
	Exprindiv_ara	Optimal caching in edge networks	[138]
	Fertility	Fertility classification	[119]
	Gasch1	Optimal caching in edge networks	[138]
	Gasch2	Optimal caching in edge networks	[138]
	German	Credit risk classification	[117]
	House-Votes	U.S. Senate and House of Representatives votes classification	[117]
	Heart-statlog	Heart decease classification	[117], [123]
	Haberman	Survival status classification of breast surgery patitents	[117],[119]
	interpro_ara	Optimal caching in edge networks	[138]
	IONOSPHERE	Radar returns classification	[117], [119], [121]–[123]
	Iris	Iris plan classification	[123]
	liver-dicorders	Presense of liver-disorder classification	[117]
	LSVT	Voice rehabilitation classification	[119]
	Mushroom	Edibility of muchrooms classification	[119]
	KDD	Network intrusion detection	[128]
	krvskp	win or lose classification in chess	[117]
	MUSK	Musk classification	[119]
	Pima	Diabetes prediction	[119], [121], [122]
	QSAR	Bioconcentration classification	[122]
	Scop_ara	Optimal caching in edge network	[138]
	Seq	Optimal caching in edge network	[138]
	Seq_ara	Optimal caching in edge network	[138]
	Sonar	Sonar signal classification	[119], [121]
	SpectHeart	Single Proton Emission Computed Tomography (SPECT) diagnosis classification	[119]
	Spo	Optimal caching in edge network	[138]
	Vote	US election vote classification	[122]
	WCBC	Breast cancer classification	[119]
	WDBC	Breast cancer diagnostics classification	[117], [119], [121]
	WDBC	Breast cancer prognostics classification	[121]
	Wine	Dynamic Network Classification	[123]
	Yoga	Dynamic network classification	[108]
	waveform	Phonetic classification	[113]

Query-based models The algorithms found were: Query by committee [136], [137], [139]–[141] and Hierarchical Query by committee [138]. The different types of algorithms in our taxonomy are illustrated in Fig. 5.1

RQ2: What are the datasets used to evaluate these algorithms?

It is crucial to select a suitable dataset when evaluating machine learning methods. Datasets need to be real, reflecting real-life scenarios. The datasets must be well-studied and have documentation containing information regarding their features. For classification tasks, it is vital to have the same number of labels from each class. Otherwise, one has to deal with the "class imbalance" problem [142]. This is a problem because many of the basic machine learning algorithms assume that the same number of instances of each class is available [143].

The datasets were categorized based on two main characteristics, the application area and the type of input. Below we present, a definition of our classification for the type of input.

Image-based datasets

(Digital) images consists of pixels that are represented by a two-dimensional numerical array [144].

Text-based datasets

Text-based data contains ore more feature columns that contain text.

Sound-based datasets

Sound-based data contains features that come from an audio file (e.g, .wav).

Numerical datasets

Numerical data contains features that are numerical.

Every dataset and its application can be found in table 5.1.

RQ3: What algorithm(s) should be used based on application?

Practitioners from the industry need to know what labeling algorithms are optimal for their application data. Based on the papers included in this study, we found several applications that utilized the algorithms discussed. Co-training and multi-view learning can be used for fast image search, medical image segmentation, activity recognition, classification of dynamic networks, webpage classification, question classification and, natural language processing. Graph-based algorithms can be used for image annotation, data augmentation, network intrusion detection, natural language processing, text classification, document classification, and speech recognition. The EM-algorithm and mixture models can be applied for web images and text classification, microalgae classification, anomaly detection in medical images, cross-language text classification, phonetic classification, and data-driven structural health monitoring. Semi-supervised support vector machines are used for face recognition, object detection, human facial emotion detection, human activity recognition, malware detection, and lung diagnostics classification from lung sounds. AL with uncertainty sampling is used in image sequence recognition, content-based information retrieval, word sense disambiguation, text classification, part-of-speech tagging, and named entity recognition. Finally, AL using QBC is applied to part-of-speech tagging, sentiment classification, document classification, power-factor prediction, and edge caching in mobile data traffic.

Based on the findings of this study, we formulate the following guidelines for choosing the optimal labeling algorithm.

- If automatic labeling is possible, choose a semi-supervised learning algorithm. Otherwise, if the only possible choice is manual labeling, choose an active learning algorithm.

- Examine whether your application is the same or similar to the applications above. Choose your algorithm based on this information.

- Since each algorithm has different assumptions to perform optimally, evaluate your data to see what assumptions are fulfilled. It is important if two types of algorithms can work for the same application.

5.6 Discussion

Several algorithms for data labeling were identified during this study. These algorithms are based on two classes i.e, semi-supervised learning and active learning. More specifically, the semi-supervised algorithms were based on graph-based, mixture models, multi-view learning and S3VMs. Active learning algorithms were based on uncertainty sampling, density weighted methods, expected error and variance reduction and QBC. The details regarding these algorithm are summarized in section 5.2.

Theoretically, these different semi-supervised algorithms have different assumptions in order to perform in the best possible way [58]. This fact is most likely known to practitioners as none of the papers compared semi-supervised learning algorithms of different types.

On the other hand, active learning algorithms can be compared to each other, which we have also seen in the papers. We cannot draw conclusions regarding which algorithm works best as the results vary. One thing that can be said for sure is that all algorithms outperform random sampling.

Eighty-seven datasets were found and categorized into four categories, image, text, sound, and numerical inputs. Each dataset was then labeled according to its application domain. A list of applications domains for each datatype can be found in Table 5.1.

Co-training and multi-view learning has often been used on image, text, and numerical data. We found no application to sound-based data. For most cases, graph-based semi-supervised learning has been used on image data, but it has also been used on text and numerical data. Only one application to sound data has been found. Models based on the EM-algorithm and mixture models have been used primarily on image and text data. Only one application was found for each numeric and sound data. Semi-supervised support vector machines have been widely used on image data, but we only found one application for text, numeric, and sound-based datasets. We only found applications from image and text data for Active Learning using Uncertainty and Density weighted algorithms. No application was found to numerical and sound-based data. For Active learning based on QBC, the majority of the applications were based on text data, a few on numerical datasets. No applications were found from sound-based data.

5.7 Conclusion

This study aims to provide a detailed overview of what machine learning algorithms exist for labeling and present common datasets that are used to evaluate these algorithms. The study also presents applications where semi-supervised learning and active learning algorithms are applied and a procedure for determining what algorithm should be used for what application.

Semi-supervised learning is a tool to use for automatic labeling. When automatic labeling is impossible, active learning is useful to determine which instances are most informative and best to label manually. This is useful when manual labeling is costly. Furthermore, we found a total of 87 datasets that are used to evaluate labeling algorithms. The datasets are distributed across four datatypes. The majority of the applications were based on image data and numerical data. There were fewer applications based on text data and only two applications were found for sound-based data. The results of this paper help researchers and practitioners to choose data labeling algorithms based on popularity, datatype and application. Furthermore, these results will help select the best dataset for evaluating newly developed algorithms based on what data and application labels are needed. In future investigations, we intend to evaluate more specific algorithms on both simulated and real-world data and investigate what datasets are better to use for evaluating certain algorithms and applications.

CHAPTER 6

Data Labeling: An Empirical Investigation into Industrial Challenges and Mitigation Strategies

6.1 Introduction

Current research estimates that over 80% of engineering tasks in a machine-learning ML project concern data preparation and labeling. The third-party data labeling market is expected to almost triple by 2024 [5], [6]. This massive effort spent in data preparation and labeling often happens because, in industry, datasets are often incomplete. After all, some or all instances are missing labels. Also, the available labels are of low quality in some cases, meaning that the label associated with a data entry is incorrect or only partially correct. Labels of sufficient quality are a prerequisite to perform supervised machine learning as the performance of the model in operations is directly influenced by the quality of the training data [54].

Crowdsourcing has been a common strategy for acquiring quality labels with human supervision [8], [9], particularly for computer vision and natural language processing applications. However, crowdsourcing has several limitations for other industrial applications, such as allowing unknown third-party

access to company data, lack of people with an in-depth understanding of the problem, or the business to create quality labels. In-house labeling can be half as expensive as crowdsourced labels while providing higher quality [2]. Due to these factors, companies still perform in-house labeling. Despite the large body of research on crowdsourcing and machine learning systems that can overcome different label quality problems, to the best of our knowledge, no research investigates the challenges faced and strategies adopted by data scientists and human labelers in the labeling process of company-specific applications. In particular, we focus on the problems seen in applications where labeling is non-trivial and requires an understanding of the problem domain.

Utilizing case study research based on semi-structured interviews with practitioners in two companies, one of which has extensive labeling experience, we study the challenges and the adopted mitigation strategies in the data labeling process that these companies employ. The contribution of this paper is twofold. First, we identify the key challenges that these companies experience concerning labeling data. Second, we present an overview of the mitigation strategies that companies employ regularly or potential solutions to address these challenges.

The remainder of the paper is organized as follows. In the next section, we provide a more in-depth overview of the background of our research. Subsequently, in section 6.3, we present the research method that we employed in the paper and an overview of the case companies. Section 6.4 presents the challenges that we identified during the case study, observations, and interviews at the company, the results from the expert interviews to validate the challenges as well as the mitigation strategies. Finally, the paper is concluded in section 6.6.

6.2 Background

Crowdsourcing is defined as a process of acquiring required information or results by request of assistance from a group of many people available through online communities. Thus crowdsourcing is a way of dividing and distributing a large project among people. After each process is completed, the people involved in the process are rewarded [145]. According to [6], crowdsourcing is the primary way of achieving labels. In the context of machine learning, crowdsourcing has its own set of problems. The primary problem is annotators

that produce bad labels. An annotator might not be able to label instances correctly. Even if an annotator is an expert, the labels' quality will potentially decrease over time due to the human factor [54]. Examples of crowdsourcing platforms are the *Amazon Mechanical Turk* and the *Lionbridge AI* [146].

Allowing a third-party company to label your data has its benefits, such as not developing your annotation tools and labeling infrastructure. In-house labeling also requires investing time training your annotators, which is not optimal if you don't have enough time and resources. A downside is that sensitive and confidential company data has to be shared with the crowdsourcing platforms. Before selecting crowdsourcing platforms, there are essential factors, such as how many and what kind of projects has the platform been successful with previously? Does the platform have high-quality labeling technologies so that high-quality labels can be obtained? How does the platform ensure that the annotators can produce labels of sufficient quality? What are the security measures taken to ensure the safety of your data?

A tool to be used in crowdsourcing when noisy labels are cheap to obtain is *repeated-labeling*. According to [147] repeated labeling should be exercised if labeling can be repeated and the labels are noisy. This approach can improve the quality of the labels which leads to improved quality in the machine learning model. This seems to work especially well when the repeated-labeling is done selectively, taking into account label uncertainty and machine learning model uncertainty. However, this approach does not guarantee that the quality is improved. Sheshadri and Lease [148] provides an empirical evaluation study that compares different algorithms that computes the crowd consensus on benchmark crowdsourced data sets using the *Statistical Quality Assurance Robustness Evaluation* (SQUARE) benchmark [148]. The conclusions of [148] is that no matter what algorithm you choose, there is no significant difference in accuracy. These algorithms includes *majority voting* (MV), *ZenCrowd* (ZC), *David and Skene (DS)/ Naive Bayes (NB)* [147]. There are also other ways to handle noisy labels. For example, in [56], they improve accuracy when training a deep neural network with noisy labels by incorporating a noise layer. So rather than correcting noisy labels, there are ways to change the machine learning models to handle noisy labels. The downside to this approach is that you need to know which instances are clean and which instances are noisy. This can be difficult with industrial data. Another strategy to detect noisy labels is *confident learning* which can be used to identify noisy labels and learn

from noisy labels. [149].

6.3 Research Method

In this paper, we report on case study research. We explored the challenges of labeling data for machine learning and what strategies can be employed to mitigate them. This section will present the data we collected and how we analyzed it to identify the challenges.

A case study is a research method that investigates real-world phenomena through empirical investigations. These studies aim to identify challenges and find mitigation strategies through action, reflection, theory, and practice, [37], [150], [151].

A case study suits our purpose well because of its exploratory nature, and we are trying to learn more about specific processes at Company A and B. The two main research questions we have are:

- **RQ1:** *What are the key challenges that practitioners face in the process of labeling data?*
- **RQ2** *What are the mitigation strategies that practitioners use to overcome these challenges?*

Data Collection

Our case study was conducted in collaboration with two companies. Company A is a worldwide telecommunication provider and one of the leading providers in Information and Communication Technology (ICT). Company B is a company specialized in labeling. They have developed an annotation platform to provide the autonomous vehicles industry with labeled training data of top quality. Their clients include software companies and research institutes.

- **Phase I: Exploration** - The empirical data collected during this phase is based on an internship from November 18 2019 to February 28 2020 in which the first author spent time at Company As office two-three days a week. The data was collected from the data scientist by observing how the they were working with machine learning and how they deal with data where labels are missing as well as having access to data sets. We held discussions with each of the data scientist working with each

Table 6.1: List of the interview participants of phase II

Company	Participant Nr	Title/role	Experience
A	I	Data Scientist	4 years
A	II	Senior Data Scientist	8 years
A	III	Data Scientist	3 years.
A	IV	Senior Data Scientist	2 years
B	V	Senior Data Scientist	7 years

particular dataset to collect data regarding the origin of the data, what they wish to use it for in the future, and how often it is updated. Using Python we could investigate how skew the label distribution is of the label distribution as well as examine the data to potentially find any clustering structure in the labels. The datasets studied in phase I came from participant I and II.

- **Phase II: Validation** - After the challenges had been identified during phase I, both internal and external confirmation interviews were conducted to validate if the previous phase's challenges were general. Four participants in the interviews where from company A and one participant was from company B. Company A had several data scientists, but we only included scientists that had issues with labeling. Each participant was interviewed separately, and the interviews lasted between 25-55 minutes. All but one interview was conducted in English. The one interview was conducted in Swedish and then translated to English by the first author. During the interview, we asked questions such as *What is the purpose of your labels?*, *How do you get annotated data?* and *How do you assess the quality of the data/labels?*

Based on meetings and interviews, we managed to evaluate and plan strategies to mitigate the challenges we observed during our study.

Data analysis

The interviews were analyzed by taking notes during the interviews and internship. We then performed a *thematic analysis* [43]. Thematic analysis is

defined as "a method for identifying, analyzing and reporting patterns" and was used to identify the different themes and patterns in the data we collected. From the analysis, we were able to identify themes and define the industrial challenges based on the notes. For each interview, we identified different themes, such as topics that came up during the interviews. Several of these themes were present in more than one interview, so we combined the data for each of the interviews, and based on that, we could draw conclusions based on the information on the same theme.

Threats to Validity

According to [37] there are four different concepts of validity to consider, *construct validity*, *internal validity*, *external validity* and *reliability*. To achieve construct validity, we provided every participant of company A with an e-mail containing all the definitions of concepts and some sample questions to be asked during the interview. We also provided a lecture on how to use machine learning to label data before the interviews so that the participant's could reflect and prepare for the interview. We can argue that we achieved internal validity through data triangulation since we interviewed every person at Company A that had experience with labels. Therefore it is implausible that we missed any necessary information when collecting data.

6.4 Results

In this section, we shall present the results of our study. We begin by listing the fundamental problems found from phase I of the study. Coming up next, we state the problems we encountered from Phase II. The interview we held with participant V was then used as an inspiration for formulating mitigation strategies for the data scientist's problems from Company A.

Phase I: Exploration

Here we list the problems that we found during Phase I of the case study.

1. **Lack of a systematic approach to labeling data for specific features:** It was clear that automated labeling processes was needed. The data scientists working at Company A had all kinds of needs for

automatic labeling. Currently, they have no idea how to approach the problem.

2. **Unclear responsibility for labeling:** Data scientists do not have the time to label instances manually. Their stakeholders can label the data by hand, but they do not want to do it either. Thus the data scientist is expected to come up with a way to do the labeling.

3. **Noisy labels:** Participant I has a small subset of his data labeled. These labels come from experiments conducted in a lab. The label noise seems to be negligible, but that is not the case. There is a difference between the generated data and the true data. The generated data will have features that are continuous, while the generated data will be discrete. Participant II works on a data set that contains tens of thousands of rows and columns. The column of interest includes two class labels, "Yes" and "No". The first problem with the labels is that they are noisy. The "Yes" is dependent on two errors, I and II. Only "Yes" based on error I is of interest. If the "Yes" is based on error II. then it should be relabeled as a "No". Furthermore, the stakeholders do not know if the "Yes" instances are due to error I or error II.

4. **Difficulty to find a correlation between labels and features:** Participant I works with a dataset whose label distribution contains five classes that describe grades from "best" to "worst". Where 1 is "best" and 5 is "worst". Cluster analysis reveals that there is no particular cluster structure for some of the labels. Labels of grade 5 seem to be in one cluster, but the other 1-4 seem to be randomly scattered in one cluster. Analysis of the data from participant II reveals no way of telling whether the "Yes" is based on error I or error II. This means that many of the "Yes" are mislabeled. .

5. **Skewed label distributions:** The label distribution from both datasets is highly skewed. The dataset from participant I has fewer instances that has a high grade compared to low grades. For participant II the number of instances labeled "No" is greater than the number of labels set as

"Yes". When training a model on this data, it will overfit.

6. **Time dependence:** Due to the nature of participant IIs data, it is possible that some of the "No" can become "Yes" in the future and so the "No" labels are possibly incorrect too.

7. **Difficulty to predict future uses for datasets.** The purpose of the labels in both datasets was to predict new labels for future instances provided by the stakeholder on an irregular basis. For participant I, the labels might be used for other purposes later. There are no current plans to use the label for different machine learning purposes.

Phase II: Validation

The problems that appeared during the interviews can be categorized as follows:

1. *Label distribution related.* Question regarding the distribution.
2. *Multiple-task related.* Questions regarding the purpose of the labels.
3. *Annotation related.* Questions regarding the oracle and noisy labels.
4. *Model and data reuse related.* Questions regarding reuse of trained model on new data.

   Below we discuss each category in more detail.

1. **Label Distribution:** We found several issues related to the label distribution. Participant Is data has an unknown label distribution. The current labels are measured in percentages and need to be translated into at least two classes, but if more labels are needed, that can be done. Participant II has a label distribution that contains two classes, "Yes" and "No". Participant IIIs data has a label distribution that includes at least three labels. Participants IV has more than three-thousand labels, so it is hard to get a clear picture of its distribution. Participant I-III all have skewed label distributions. If a dataset has a skew label distribution, then the machine learning model will overfit. This means that

if you have a binary classification problem and you have 80% of class A and 20% of class B, the model might predict A most of the time even when an actual case is labeled as B [152].

2. **Multiple tasks:** Participant I, II,and III say that that for now, the only purpose of their labels is to find labels for new data, but the chances are that it will be reused for something else later on. Participant IV does not use its labels for machine learning purposes but other practical reasons. If you do not plan ahead and only train a model concerning one task, then if you need to use the labels for something else later, you will have to relabel the instances for each new task.

3. **Annotation:** Participant I has some labeled data that comes from laboratory experiments. However, these labels are only used to help label new instances to be labeled manually. Participant II has its labels coming from the stakeholders, but these instances need to be relabeled since they are noisy. Participant III has labeled data coming from stakeholders, and these are expected to be 100% correct. Participant IV defines all labels by itself and does not consult the stakeholders at all. The problem here is that the data scientists are often tasked to do labeling on their own. Even if the data scientists get instances from the stakeholders, the amount of labels are often of insufficient quantity and/or quality.

4. **Data Reuse:** Participant III has had problems with reusing a model. First the data was labeled into two classes "Yes" and "No. Later the "Yes" category would be divided into sub-categories "YesA" and "YesB". When running the model on this new data, it would predict the old "Yes" instance as "No" instance. Participant III has no idea as to why this happens.

Summary from Company B

Participant V of Company B has earlier experience with automatic labeling. Therefore interview V was used to verify some actual labeling issues from the industry. According to participant V, Company B has worked and studied

automatic labeling for at least seven years. Company B uses crowdsourcing to label data using 1000 people. Participant V confirms that the labeling task takes 200 times less time thanks to active learning than if active learning was not used. The main problem company B has with the labeling is that it is hard to evaluate the quality labels and access the human annotator's quality. A final remark from Company B is that they have experienced a correlation between automation and quality. The more automation included in the process, the less accurate will the labels be. Three of the authors of this paper performed a systematic literature review on automated labeling using machine learning [153]. Thanks to that paper, we can conclude that active learning and semi-supervised learning can be used to label instances.

Machine Learning methods for Data Labeling

Here we present and discuss Active Learning and Semi-supervised learning methods in terms of how they can be used in practice with labeling problems.

Active Learning:

Traditionally labels would be chosen randomly to be labeled and used with machine learning. However, choosing instances to be labeled randomly could lead to a model with low predictive accuracy since non-informative instances could be selected for labeling. To mitigate the issue of choosing non-informative instances, active learning (AL) is proposed. Active learning queries instances by informativeness and then labels them. The different methods used to pose queries are known as *query strategies* [12]. According to [153] the most commonly used query strategies are *uncertainty sampling, error/variance reduction, query-by-committee (QBC)* and *query-by-disagreement (QBD)*. After instances are queried and labeled, they are added to the training set. A machine learning algorithm is then trained and evaluated. If the learner is not happy with the results, more instances will be queried, and the model will be retrained and evaluated. This iterative procedure will proceed until the learner decides it is time to stop learning. Active learning has proven to outperform passive learning if the query strategy is properly selected based on the learning algorithm [12]. Most importantly, active learning is a great way to make sure that time is not wasted on labeling non-informative instances, thus saving time and money in crowdsourcing [6].

Semi-supervised learning:

Semi-supervised learning (SSL) is concerned with algorithms used in the scenario where most of the data is unlabeled, but a small subset of it is labeled. Semi-supervised learning is mainly divided into *semi-supervised classification* and *constrained clustering* [34].

Constrained clustering is an extension to unsupervised clustering. Constrained clustering requires unlabeled instances as well as some supervised information about the clusters. The objective of constrained clustering is to improve upon unsupervised clustering[154]. The most popular semi-supervised classification methods are *mixture models using the EM-algorithm*, *co-training/multi-view learning*, *graph-based SSL* and *semi-supervised support vector machines (S3VM)* [153].

Below we list eight practical considerations of Active Learning.

1. **Data exploration to determine which algorithm is best.** When starting on a new project involving machine learning, it is hard to know which algorithm will yield the best result. Often there is no way of knowing beforehand what the best choice is. There are empirical studies on which one to choose, but the results are relatively mixed [87], [155], [156]. Since the selection of algorithms varies so much, it is essential to understand the problem beforehand. If it is interesting to reduce the error, then expected error or variance reduction is the best query strategies to choose from [12]. If the sample's density is easy to use and there is strong evidence that support correlation between cluster structure to the labels, then use density-weighted methods [12]. If using extensive probabilistic models, uncertainty sampling is the only viable option [12]. If there is no time testing out different query strategies, it is best to use the more simple approaches based on uncertainty [12]. From our investigation, it is clear that company A needs labels in their projects. However, since they have never implemented an automatic labeling process before, it is important to do right from the beginning. The data scientists must carefully examine the distribution of data set, check whether there are any cluster structures and if there are any relationships between the clusters and the labels. If the data exploration is done in a detailed, correct way, then finding the correct machine learning approach is easy, and we don't need to spend time testing different

machine learning algorithms.

2. **Alternative query types:** A traditional active learner queries instances to be labeled by an oracle. However, there are other querying ways, e.g *human domain knowledge*, incorporated into machine learning algorithms. This means the learner builds models based on human advice, such as rules and constraints, and labeled and unlabeled data. An example of domain knowledge with active learning is to use information about the features. This approach is referred to as *tandem learning* and incorporates feature feedback in traditional classification problems. *Active dual supervision* is an area of active learning where features are labeled. Here oracles label features that are judged to be good predictors of one or more classes. The big question is how to query these feature labels actively.

3. **Multi-task active learning:** From our interview we can see that there are cases where labels are needed to predict labels for future instances. In other cases the labels aren't even needed for machine learning. In one case the data scientist thinks that the labels will be used for other prediction task but is unsure. The most basic way in which active learning operates is that a machine learner is trying to solve a single task. From the interviews it is clear the same data needs to annotated in several ways for several future tasks. This means that the data scientist will have to spend even more time annotating at least one time for each task. It would be more economical to label a single instance for all sub-tasks simultaneously. This can be done with the help of multi-task active learning [157].

4. **Data reuse and the unknown model class:** The labeled training set collected after performing active learning always has a bias distribution. The bias is connected to the class of the model used to select the queries. If it is necessary to switch learners to a more improved learner, it might be troublesome to reuse the training data with models of a different class. This is an essential issue in practical use for active learning. If you know the best model class and feature set beforehand,

then active learning can safely be used. Otherwise, active learning will be outperformed by passive learning.

5. **Unreliable oracles:** It is essential to have access to top-quality labeled data. If the labels come from some experiments, there is almost always some noise present. In one of the data sets from company A, a small subset of the data was labeled. The labels of that particular data set come from experiments conducted in a lab. The label noise seems to be negligible, but that is not the case. There is a difference between the generated data and the actual data. The actual data will have continuous features, while the generated data will have discrete features. Another dataset that we studied has labels that came from customer data. The labels were coded "Yes" and "No". However, the "Yes" was due to factors A and B. So the problem here is to find a model that can predict the labels, but we are only interested in the "Yes" that is due to factor A. The "Yes" due to factor B needs to be relabeled to a "No". Since the customer data does not provide whether the "Yes" are due to factor A or B. The second problem was that some of the "No" could develop into a "Yes" over time. It was up to the data scientist to find a way to relabel the data correctly. The data scientist had a solution to the problem but realized that it was faulty and asked us for help. We took a look at the data and the current solution. We saw two large clusters, but no significant relationship existed between the different labels and the features. We found two clusters, but both contained almost equally many "Yes" and "No". Let's say that the first cluster contained about 60% "Yes" and 40% "No" and in the second cluster we had 60% "No" and 40% "Yes". After doing this, all of the first cluster instances were relabeled as "Yes" and all instances in the second cluster were relabeled as "No". We conclude that this is an approach that will yield noisy labels. The same goes if the labels come from a human annotator because some of the instances might be difficult to label. People can easily be distracted and tired over time, so the labels' quality will vary over time. Thanks to crowdsourcing, several people can annotate the same data, and that it is easier to determine which label is the correct one and produce "gold-standard quality training sets". This approach can also be used to evaluate learning algorithms on training sets that

are non-gold-standard. The big question is: How do we use noisy oracles in active learning? When should the learner query new unlabeled instances rather than update currently labeled instances if we suspect an error. Studies, where estimates of both oracle and model uncertainty were taken into account, show that data can be improved by selectively repeated labeling. How do we evaluate the annotators? How might the effect of payment influence annotation quality? What to do if some instances are noisy no matter what oracle you use and repeated labeling does not improve the situation?

6. **Skewed label distributions:** In two of the data sets we studied, the distributions of the labels are skewed. That is, there is more of one label than there is of another. In the "Yes" and "No" labeled example, there are way more "No" instances. When the label distribution is skewed, active learning might not give much better results than passive learning. If the labels are not balanced, active learning might query more of one label than another. The skewed distribution is a problem, but the lack of labeled data is also a problem. In one of the datasets, we have instances labeled from an experiment. Very few labels are labeled from the beginning, and new unlabeled data is coming every fifteen minutes. "Guided learning" is proposed to mitigate the slowness problem. Guided learning allows the human annotator to search for class-representative instances in addition to just querying for labels. Empirical studies indicate that guided learning performs better than active learning as long as it's annotation costs are less than eight times more expensive than labeling queries.

7. **Real labeling costs and cost reduction:** From observing the data scientists at Company A, we would say that they will spend about 80% of the time they spend on data science prepossessing the data. Therefore we recognize that they do not have time to label too many instances, and it is crucial to reduce the time it takes to label things manually. If the possibility exists, avoid manual labeling. Assume that the *cost* of labeling is uniform. The smaller the training set used, the lower will the associated costs be. However, in some applications, the cost might be varying, so simply reducing the labeled instances in the training data

does not necessarily reduce the cost. This problem is studied within *cost-sensitive active learning*. To reduce the effort in active learning, *automatic pre-annotation* can help. In automatic pre-annotation the current model predictions helps to query the labels [90], [158]. This can often help the laboring efforts of the learner. If the models make many classification mistakes, then there will be extra work for the human annotator to correct them. To mitigate these problems *correlation propagation* can be used. In correlation propagation, the local edits are used to update the prediction interactively. In general automatic pre-annotation and correction propagation do not deal with labeling costs themselves. However, they do try to reduce the costs indirectly by minimizing the number of labeling actions performed by human oracle. Other cost-sensitive active learning methods take varying labeling costs into account. The learner can incorporate both current labeling costs and expected future errors in classification costs [159]. The costs might not even be deterministic but stochastic. In many applications, the costs are not known beforehand. However, they might be able to be described as a function over annotation time [160]. To find such a function, train a regression cost-model that predicts the annotation costs. Studies involving real human annotation cost shows the following results.

- Annotation costs are not constant across instances [161]–[164].
- Active learners that ignore costs might not perform better than passive learners [12].
- The annotations costs may vary on the person doing the annotation [161], [165].
- The annotation costs can include stochastic components. *Jitter* and *pause* are two types of noise that affect the annotation speed.
- Annotation can be predicted after seeing only a few labeled instances. [163], [164].

8. **Stopping criteria:** Since active learning is an iterative process, it is relevant to know when to stop learning. Based on our empirical findings, the data scientists have no interest in doing any manual labeling, and if they have to, they want to do it as little as possible. So when

the cost of gathering more training data is higher than the cost of the current system's errors, then it is time to stop extending the training set and hence stop training the machine learning algorithm. From our experience at company A the data scientist have so little time free from doing other tasks than data prepossessing, so time is the most common stopping factor.

Challenges and Mitigation Strategies:

Many of the problems identified during phase I and phase II overlap to a certain degree, so we took all the problems and summarized them into three challenges (C1-C3) that were later mapped to three mitigation strategies (MS1-MS3). These mitigation strategies are derived from the practical consideration above. Finally, we map MS1 to C1, MS2 to C2, and MS3 to C3.

C1: **Pre-processing:** This challenge represents all that needs to be done during the planning stage of the labeling procedure. This would include creating a systematic approach for labeling (problem 1 of phase I), doing an exploratory data analysis to find the correlation between labels and features (problem 4 of phase I), as well as choosing a model that can be reused on new data (problem 6 of phase I) and label instances concerning multiple tasks (problem 7 of phase I, problem 4 of phase II).

MS1: **Planning**: This strategy contains all the solution frameworks from practical consideration 1, 2, 3, 4, 7 and 8 as they all involve the steps necessary to plan an active learning strategy for labeling.

C2: **Annotation:** This challenge represents the problems concerning choosing an annotator as well as evaluating and reduce the label noise (problems 2,3 from phase I and problem 3 from phase II).

MS2: **Oracle selection**: This strategy contains only solution frameworks from practical consideration 5. It describes how we can choose oracles to produce top quality labels.

C3: **Label Distribution:** This challenge represents all the problems concerning the symmetry of the label distributions such as learning with a skew label distribution (problem 5 of Phase I and problem 1 o Phase II).

MS3: **Label distribution**: This strategy contains solution frameworks from practical consideration 6. It describes how we can do labeling when the label distribution is skew.

6.5 Discussion

We learned that active learning is a popular tool for acquiring labels from our verification interview with Company B. Thanks to active learning, the labeling task takes 200 times less than if active learning was not used.

In the background, we presented some current practices that can help with labeling. The most popular practice being crowdsourcing. However, crowdsourcing has its own set of problems. The primary concern is those bad annotators will produce noisy labels due to inexperience or human factors. Secondly, The benefit of allowing third-company to label data is that you don't have to spend time training your employees to do the job, nor do you need to develop your own annotation tools and infrastructure. The big downside is that you have to share confidential company data with the crowdsourcing platform. Repeated labeling can improve the quality of the labels, but there are no guarantees that this will enhance the quality. Rather than correcting noisy labels, there are ways in which you can change the machine learning models to handle noisy labels. The downside to this is that you need to know which instances are bad, and this can be difficult in an industrial setting.

None of the techniques discussed in the background utilizes automated labeling using machine learning. Thanks to our efforts, we formulated three labeling challenges and provided mitigation strategies based on active machine learning. These challenges are related to questions such as, How can labeling processes be structured?, who and how do we label the instances? Can the correlation between labels and features be found, so that labels can be determined from the features? Both manual and automatic labeling involves some noise in the labels. How should these noisy labels be used? What do we do if the distribution of the labels is skewed? How do we consider the fact that some of the labels might change over time, due to the nature of the

data? How do we label instances so that the labels can be useful for several future tasks?

Three mitigation strategies that could possibly solve the three challenges were presented.

6.6 Conclusion

This study aims to provide a detailed overview of the challenges that the industry faces with labeling and outline mitigation strategies for these challenges.

To the best of our knowledge 95%of all the machine learning algorithms deployed in the industry are supervised. Therefore, every dataset must be complete with labeled instances. Otherwise, the data would be insufficient, and supervised learning would not be possible.

It proves to be challenging to find and structure a labeling process. You need to define a systematic approach for labeling and examine the data to choose the optimal model. Finally, you need to select an oracle to produce top-quality labels as well as plan how to handle skewed label distributions.

The contribution of this paper is twofold. First, based on a case study involving two companies, we identified problems that companies experience in relation to labeling data. We validated these problems using interviews at both companies and summarized all problems into challenges. Second, we present an overview of the mitigation strategies that companies employ (or could employ) to address the challenges.

In our future work, we aim to further develop the challenges and mitigation strategies with more companies. In addition, we intend to develop solutions to simplify the use of automated labeling in industrial contexts.

CHAPTER 7

An Empirical Evaluation of Graph-based Semi-Supervised Learning for Data Labeling

7.1 Introduction

Most industries have recently started implementing machine learning algorithms for various tasks. Among these tasks, supervised classification algorithms are used to solve classification tasks such as classifying images and text. For companies to use supervised classification, datasets need to be fully labeled. However, datasets are rarely fully labeled in industry, and achieving labels is troublesome for many reasons. The first reason is that the data needs to be manually labeled by data scientists. However, data scientists are often busy performing more specialized tasks and do not have time for labeling. A solution to this problem is crowdsourced labeling. The second problem is that crowdsourcing is expensive, and companies might have to share confidential data. These two problems make manual labeling unappealing, and companies prefer to implement automated learning approaches [8], [9]. According to [153], a systematic literature review investigating machine learning approaches that reduce the labeling effort, it concluded that semi-supervised

learning is a popular tool for automated labeling. Semi-supervised learning is applicable to image, video, sound, text, and numerical datasets, yet not widely applied in industry [153]. Furthermore, new users of semi-supervised learning will need to spend much time learning what algorithms fit well for their particular problem [166].

This paper is an extension of a simulation study [42] where four semi-supervised learning and three active learning algorithms were evaluated in terms of two dimensions. All algorithms from [167] are removed because we decided that comparing SSL to AL would be unsuitable. This paper extends the benchmark of semi-supervised algorithms by studying 13 different graph-based algorithms for 25 datasets. We restrict ourselves to graph-based algorithms as they are the most commonly used [153]. The algorithms were collected from the *GraphLearning* package. We took the 15 datasets from [167] and added ten more commonly used datasets for evaluating semi-supervised learning. We evaluate the algorithms across three dimensions. In the original empirical evaluation [153], we evaluate the algorithms in terms of two dimensions, *Performance:* How well does the algorithm predict labels. *Effort*, what number of available labels are required for best performance. In addition, the new empirical evaluation of the taxonomy's algorithms will add a third dimension, *Datatype*. Do the algorithms perform better on datasets of a different datatype? Furthermore, this paper will address whether using a certain algorithm will increase the probability of achieving a certain accuracy and will the algorithm perform worse in the presence of noise.

Thanks to the results of this study, practitioners will know what algorithms to explore in order for them to achieve a certain accuracy when applying automatic labeling, how much manual effort is required to achieve such accuracy and whether to expect worse results when applying the algorithms to their dataset.

The paper is organized in the following manner. Section 7.2 describes the theory behind semi-supervised learning. Section 7.3 outlines the research method we used, how we performed the simulations, what software packages we used and how we evaluated the algorithms. Section 7.4 presents the results and section 7.5 discusses the results. Finally, the paper is concluded in section 7.6. The tables that were generated to describe the results are available at https://github.com/deeplearner788/An-EmGraph-Based-Semi-Supervised-Learning-Algorithms-for-Data-Labeling/blob/main/appendix.pdf

7.2 Background

This section gives a theoretical overview of the machine learning and statistical tools utilized in this study.

Labeling challenge in Software Engineering

Machine learning has a vast range of applications such as self-driving vehicles. When engineering machine learning based software, training deep learning models for object detection and scene perception in self-driving vehicles faces many challenges [168]. One problem is localization which is solved using maps. Mapping however is a costly task [169] that can be addressed using sensors with limited range and coverage. Therefore, unmanned aerial vehicles (UAVs), satellites, and other aerial vehicles have been used to find mappings sideways. These datasets need to be labeled for the supervised classification of unseen data.

Many different labeling types exist, such as bounding boxes, polygons, and image segmentation. Semantic image segmentation involves annotation on a pixel level. The procedure transforms an image into different colors, each with a different label. The images obtained from the image segmentation are then used to train a deep learning algorithm that can automatically segment an input image and ensure the vehicle does not hit any obstacles while driving. This labeling procedure is time-consuming as there are many labeling categories. The most common are *accessible road areas, barriers, traffic signs, and roadside buildings.* According to research [167], 80% of the time spent in a machine learning project is on data labeling. Because of the time it takes to label the data, the task should not be allocated to data scientists busy doing more specialized tasks. A person performing this detailed labeling job might also make mistakes that will lead to low-quality labels, which directly influences the performance of the machine learning algorithm.

Another solution to the labeling problem is to use third-party data labeling services. Due to the high demand for labeling, the data labeling market is expected to triple by 2024 [5], [6]. Crowdsourcing is an example of a third-party labeling service and the primary way of getting labels [6]. Crowdsourcing distributes and divides a task among several parties. The parties involved in this task will be rewarded once said task is completed [145]. Through crowdsourcing, companies can obtain labels by requests from a group or online

communities. *Amazon Mechanical Turk* and *Lionbridge AI* are examples of crowdsourcing services [146]. Using crowdsourcing means that the companies do not have to develop their labeling infrastructure and tools necessary to perform the labeling. The downside of crowdsourcing is that companies cannot share sensitive data. When choosing a crowdsourcing service, one important question is, how can we ensure annotators produce high-quality labels?

A tool that can reduce the labeling is *Active Learning*. Active learning queries what instances should be labeled according to a *query strategy* that selects the instances based on how informative they are. The newly labeled instances are then included in the training data, and the ML model is trained and evaluated. We then interactively add, remove or relabel instances in the training dataset until we reach a sufficiently high accuracy.

Semi-Supervised Learning

This section only gives a brief overview of SSL. The reader is referred to [34] for a more detailed view of SSL.

Machine Learning and Deep Learning algorithms require large amounts of data to achieve high-performance accuracy. For the industry to apply supervised learning, large labeled datasets are necessary. In many scenarios, the datasets are missing labels either entirely or partially. In order to achieve high-performance classification algorithms without using costly tools such as crowdsourcing and active learning for manual laboring, they could utilize *semi-supervised learning* [34]. While supervised and unsupervised learning algorithms have been designed to learn from labeled and unlabeled data, semi-supervised learning algorithms have been designed to learn from unlabeled and labeled data respectively. Therefore semi-supervised learning can be a more realistic scenario in industrial settings. Semi-supervised learning algorithms strive to improve the decision boundary acquired by supervised learning with the help of unlabeled data.

There are four main assumptions in semi-supervised learning. The main assumption is that many unlabeled and few labeled instances are available. The three other assumptions put constraints on the distribution. These are the *smoothness*, *cluster* and the *manifold* assumptions [170]. The smoothness assumption says that if two features lie close to each other in a high-density region, their output labels also lie close. The cluster tells us that if two features lie in the same cluster, they most likely have the same class label.

The manifold assumption, often considered a generalization of the two fore mentioned assumptions, states that each datapoint lies on a manifold [34].

In this study, we have chosen to evaluate *graph-based* semi-supervised learning algorithms as they are the most favorable for our setting and the most popular semi-supervised algorithms [153]. Our analysis utilizes *Bayesian Data Analysis* (BDA) previously used to analyze other benchmarks [42], [171]. The *classical* view of probability expresses it in terms of random repeatable events. Some events can not be repeatable, so the classical view of viewing probability becomes useless. The existence of non-repeatable events motivates the *Bayesian* viewpoint to express probability as a measurement of uncertainty. This uncertainty is updated through new evidence. Suppose we have some prior hypothesis H before observing our evidence E. This prior hypothesis is expressed in a prior probability distribution $p(H)$. The probability $p(E|H)$ represents the effect of the evidence. With the help of Bayes formula [172], we calculate the updated probability, known as the *posterior probability distribution* $p(H|E)$ [172]. BDA has proven advantageous to the classical view regarding modeling and model assumptions. Many Bayesian tools are available for answering research questions [171].

Graph-based SSL algorithms are *transductive* semi-supervised algorithms [58], meaning that they only predict labels for the unlabeled instances that the learner provides [58]. Because the transductive algorithms do not model the input space, Graph-based algorithms construct a graph for the data points and label the unlabeled instances by measuring the distances between nodes in the graph [170].

The Bradley Terry model

The Bayesian version [171], [173] of the Bradley-Terry model [174], [175] is frequently used for ranking and comparison of objects. Each outcome $y_{i,j}$ of the comparisons are binary variables, either taking value 1 with probability $p_{i,j}$ if i beats j and value 0 with probability $1 - p_{i,j}$ otherwise. This means that the outcomes $y_{i,j}$ are Bernoulli distributed, in other words:

$$y_{i,j} \sim \text{Bernoulli}(p_{i,j}).$$

Furthermore, we assume that the outcomes are independent. To rank n objects, we first estimate the strength parameter $\mu \in \mathbb{R}$ of each object and then

calculate the probability of object i beating object j as

$$p_{i,j} := P(i \text{ over } j) = \text{logit}^{-1}(\mu_i - \mu_j).$$

Bradley-Terry model's ability to calculate the probability of objects beating each other and access the reliability of ranks through uncertainty estimation makes it preferable to other models. We have utilized a *Generalized Linear Mixed Model* [176] to account for the random effect on each dataset.

Logit GLMM model for binomial samples

The Generalized Linear Mixed Model for Binomial samples [171], [176] calculates the probability of success (an algorithm yields a specific accuracy). Let y_i be an observation, with

$$y_i = \begin{cases} 1 & \text{if success} \\ 0 & \text{if failure} \end{cases}$$

i.e, $y_i \sim \text{Bernoulli}(p)$.

For n samples $y_1, y_2, ..., y_n$, the sum of all outcomes will be binomial distributed,

$$y = \sum_{i=1}^{n} y_i \sim \text{Binomial}(n, p).$$

Hence, we will use the binomial distribution as likelihood. The probability of success will be modeled as follows:

$$p = \text{logit}(P(y = 1)) = a + bx + u,$$
$$u \sim \text{Normal}(0, \sigma^2).$$

where a is the fixed effect, b is the log-odds ratio and u is the random effect.

The parameters have the following distributions:

$$\begin{aligned} a_{alg,i} &\sim N(0,5), \\ b_{noise,i} &\sim N(0,5), \\ a_{bm,j} &\sim N(0,s), \\ s &\sim \text{Exponential}(0.1). \end{aligned}$$

7.3 Research Method

This section outlines how the simulations were conducted and what data was extracted.

This study is an empirical evaluation of graph-based semi-supervised algorithms that aid in automatically labeling data. We utilize benchmark experiments according to [40] but use Bayesian Data Analysis instead of frequentist statistics. In benchmark experiments, a contrived environment is set up to analyze and measure the differences in various techniques. Two common goals of Benchmarking Studies include *Algorithm Comparison* and *Characterizing Algorithms' Performance by Problem Features.*

The first goal is to compare the performance of many algorithms to understand the strengths and weaknesses of different algorithms for different types of problems. In the context of this study, we wish to determine what algorithms achieve the best accuracy. The second goal is to link features of the problem with the algorithms' performance. In this study the features are the datatypes, and the objective is to investigate whether some algorithms perform better on a particular dataset and datatype. In addition, we investigate how much manual effort is required to achieve a certain accuracy. To accomplish these goals we study the following research questions below.

- **RQ1:** *What is the ranking of algorithms in terms of highest accuracy*
- **RQ2:** *How do the algorithms rank differently according to a specific datatype?*
- **RQ3:** *Do the algorithms rank differently depending on the number of labeled instances in the dataset?*
- **RQ4-a:** *What is the probability of each algorithm yielding an accuracy* $\varepsilon \geq 0.9$

- **RQ4-b:** *What is the impact of noise in the probability of success of each algorithm's accuracy* $\varepsilon \geq 0.9$

Algorithms

Graph-based semi-supervised learning algorithms are transductive. Given that our datasets consist of labeled and unlabeled instances, the algorithm will only provide labels for the unlabeled instances in this dataset. Graph-based algorithms are the second most popular algorithms that are used in practice. Only multi-view learning algorithms are the more popular. However, these rely on different assumptions we do not consider in this study.

Thirteen different graph-based algorithms were used in this study. All simulations were run in Python, using the GraphLearning package [53], which includes *Poisson Learning* algorithms. Poisson Learning can achieve high accuracy at low label rates and has proven to outperform other SSL algorithms on common datasets [129]. The GraphLearning [53] package also uses *Laplace Learning* algorithms, commonly used for comparison with Poisson Learning. The are algorithms in the GraphLearning package were all used with $w_{ij} = \exp(-4|x_i - x_j|^2/d_k(x_i)^2),\ i \neq j,\ k = 10$. The weight matrix is made symmetric by $w = w^T + w$. For the Poisson learning algorithms we set $w_{ii} = 0$ for all i. The choice of w_{ii} does not affect the solution of the Poisson learning algorithm but increases the convergence speed.

Every algorithm used is listed below.

- Laplace Learning **(laplace):**
- Mean Shifted Laplace **(mean_shifted_laplace):**
- Centered kernel method **(centeredkernel):**
- Poisson Learning **(poisson):**
- Poisson Learning, alternate version **(poisson_2):**
- Poisson Learning, Balanced **(poissonbalanced):**
- Poisson MBO, **(poissonmbo)**
- Poisson MBO, Balanced **(poissonmbobalanced):**
- Poisson MBO with volume constraints **(poissonmbo_old:**

- Poisson learning with volume constraints **(poissonvolume):**
- Random walk **(randomwalk):** is implemented with $\epsilon = 0.05$
- Sparse Label Propagation **(sparselabelpropagation):**
- Weighted non-local Laplacian **(wnll):**

Datasets

All 25 datasets can be found in the list below. We used eight datasets for each datatype. The datatypes are *image, text* and *numerical* defined according to [167].

- **Image data:**
  - **Caltech-256:** Contains 30607 images divided into 256 categories [177]. The dataset was downloaded from Kaggle [178], and the original dataset is located at [179].
  - **Cifar-10:** This dataset originally contains 60000 32x32 images that can be divided into ten classes, airplane, automobile, bird, car, deer, dog, frog, horse, ship, and truck [180]
  - **Corel:** (Database for Content based Image Retreival) [181]. We choose ten classes of images: beaches, bus, dinosaurs, elephants, flowers, foods, horses, monuments, mountains and snow, people and villages in Africa. Each class contains 90 images. The dataset was downloaded from Kaggle [182]
  - **Digits:** Each dataset instance is an image containing a handwritten digit from 0 up to 9 [183]. Furthermore, there are ten label classes 0 to 9 and 1797 instances.
  - **MNIST:** This dataset is a modified NIST (National Institute of Standards and Technology) dataset to better suit machine learning testing. The dataset contains 60000 instances distributed across ten classes.
  - **MiniImageNet:** This dataset is a smaller version of the ImageNet dataset [184]. The dataset is constructed according to a hierarchy provided by WordNet and is used for object detection. The number of instances included is above fourteen million and the number of

categories is 20000. Mini ImageNet [185] is a smaller version of ImageNet.

- **TieredImageNet:** Like mini ImageNet, the tiered ImageNet [186] is a smaller version of ImageNet. It contains 608 categories for a total of 779165 instances.

- **Text data:**
  - **20news:** This dataset contains 18846 instances divided into 20 classes that describe the 20 different types of news [187]. Each instance of this dataset contains a newsgroup document. A total of 18846 instances in the dataset are evenly distributed across 20 newsgroups.
  - **Amazon:** This dataset contains reviews from Amazon. Originally it contained two features and 3 million instances, but we have selected only 5000. The labels represent the review scores going from 1 to 5.
  - **DBworld:** This dataset contains 64 emails collected from the DBWorld newsletter. The dataset was already pre-processed using a binary bag-of-words representation and stopword removal [188].
  - **Fake and true news:** Each instance contains a news article labeled according to its subject. The purpose of the dataset is to predict whether the news article is considered "truthful" or "false" [189], [190]. There are 44596 instances in the dataset.
  - **IMDB:** The IMDB dataset [191] contains 50000 movie reviews and their sentiment: positive or negative.
  - **Ohsumed:** This dataset is part of the National Library of Medicine's (MEDLINE) database. This database contains millions of instances. Each instance contains a reference to a life science journal. The purpose of the dataset is to classify what medical subject heading it belongs to (MeSH) [192]. The Ohsumed contains 23 subject headings and in this study we use 12 of these [192].
  - **Spambase:** This dataset contains 4601 instances of emails labeled as "spam" and "non-spam". The dataset aims to classify the emails as "spam" or "non-spam".

  - **Reuters:** Also known as Reuters-21578. This dataset was collected from the "Reuters financial newswire service" in 1987. Each instance represents a news article. The purpose of the dataset is to classify the class of news articles. The dataset contains 10788 instances and has 90 class labels.
  - **Spambase:** This dataset contains 4601 instances of emails labeled as "spam" and "non-spam". The dataset aims to classify the emails as "spam" or "non-spam".

- **Numerical data**
  - **Statlog:** This dataset contains financial data of 1000 Germans. Each instance represents information about one individual. The purpose of the dataset is to classify an individual's credit score as "good" or "bad" [193].
  - **Ionosphere:** The Ionosphere dataset [194] was collected from 16 high-frequency antennas in Goose Bay, Labrador. The purpose of the dataset is to classify whether a radar return is "good" or "bad". The dataset contains 351 instances and 34 features.
  - **Iris:** Each instance represents measurements of different Iris flowers. The dataset contains 150 samples, and the purpose is to classify the species of Iris flower [195].
  - **MUSK:** Each instance represents information about a molecule. The dataset consists of 166 features that describe 476 molecules. The purpose of the dataset is to classify whether a molecule is a "musk" or "non-musk". [195]
  - **PIMA:** The PIMA dataset [196] originates from the National Institute of Diabetes and Digestive and Kidney Diseases. All participants in this study were 21-year-old females of Pima Indian heritage. The purpose of the dataset is to predict whether an individual has diabetes. The features contain information such as MBI, insulin level, age, and the number of pregnancies.
  - **Sonar:** The Sonar dataset [197] contains 208 instances of sonar signals that bounce off a metal cylinder or rocks. The purpose of the dataset is to determine if the signal bounces off a metal cylinder or a rock. The data contains 208 features.

- **WDBC:** The Breast Cancer Wisconsin (Diagnostics) [198],[199] dataset contains features from images of breast mass. The purpose of the dataset is to classify cancer as "malignant" or "benign". There are 569 instances and 32 features in total.
- **Wine:** The wine dataset is also a classic example of multi-class classification. It contains 178 instances across three classes [200]. This dataset was obtained from chemical analysis of wines cultivated from three different sources. The features describe the quantity of 30 different substances found in the wines. The purpose of the dataset is to classify the source of the wines.

Table 7.1: Summary table for the datasets.

Datatype	Dataset
Image	Caltech-256
	Cifar-10
	Coil-100
	COREL
	Digits
	FashionMNIST
	MNIST
	MiniImageNet
	TieredImageNet
Text	20news
	Amazon
	DBworld
	Fake and true news
	IMDB
	Ohsumed
	Reuters
	Spambase
Numeric	Ionosphere
	Iris
	German
	MUSK
	PIMA
	Sonar
	WDBC
	Wine

Simulations

We need to vary the number of available labels in the dataset to answer questions regarding manual effort. This paper considers the cases where we have 10%, 25%, 50%, 75%, and 90% of available labels.

We measured accuracy by

$$\varepsilon = \frac{100}{n-m} \max\left(\sum_{i=1}^{n} I(y_i = \widehat{y}_i) - m, 0\right),$$

where $I(x)$ is **indicator function** defined as

$$I(x) = \begin{cases} 1 \text{ if } x \text{ is true} \\ 0 \text{ otherwise} \end{cases},$$

n is the number of labeled and unlabeled instances and m is the number of labeled instances.

The simulations were computed for ten iterations using different random seeds. The pseudocode is found below.

```
for s in iteration:
    for a in algorithm:
        for ℓ in available_labels:
            for d in dataset:
                ε = accuracy(a, ℓ, d, s)
```

The simulations were saved in a .csv file containing one column with numerical accuracy, one column with dataset name, one with datatype, one with iteration number, and one with manual effort.

7.4 Results

In this section we present the results from our simulations.

RQ1: Aggregated results

Figure 7.1 illustrates the descriptive statistics of the accuracy for each algorithm as a boxplot.

We dropped the data frame columns containing the number of available labels and datatype to analyze the aggregated results. A column that contains the algorithms label was added. Table 7.2 lists the median ranks of the

aggregated data and shows that sparselabelpropagation is the highest-ranking algorithm, followed by poissonmbo, poissonmbo_old, and Balanced poissonmbo. The rankings of the algorithms are uncertain since the variance of the ranks is large. The HDPIs for the strength parameters distribution are found in the appendix.

RQ2: Datatype comparison

To analyze the results based on the datatype we ignored the column containing manual effort but kept the datatype column. A boxplot for the descriptive statistics of datatypes is located in Figure 7.2.

Tables 7.3, 7.4, and 7.6 contains the median ranks for datatypes.

According to Table 7.3, the highest-ranking algorithm for image datasets is centeredkernel, the second-highest-ranking algorithm is poissonmbo, the third-ranking is shared with Laplace learning and Sparse label propagation, and the fourth-highest algorithm is Balance Poisson. The high variance of the third highest-ranking algorithm indicates the uncertainty in their ranks.

According to Table 7.4, the highest-ranking algorithm for text datasets is randomwalk, the second-highest algorithm is Poisson Balanced, the third-highest is poissonmbo, and the fourth-highest is poissonmbo_old.

According to Table 7.6, the highest ranking algorithm is poissonmbo, the second highest is Sparse label propagation, the third highest is Balanced Poisson and the fourth highest is poissonmbo_old.

Figures that illustrate the distribution of the strength parameters and their High Posterior Density Intervals (HPDI) are found in the appendix.

RQ3: Manual effort

We dropped the datatype column and kept the column containing manual effort to analyze the results based on the manual effort. Thus we could analyze the accuracy of the algorithms with respect to the number of available labels. A boxplot for the descriptive statistics w.r.t number of available labels is located in Figure 7.3

Tables 7.7, 7.8, 7.9, 7.10 and 7.11 contains the median ranks for manual effort.

Table 7.7 illustrates that the highest-ranking algorithm with access to 10% available labels is poissonmbo, the second-highest rank is shared with Bal-

anced Poisson and randomwalk, the third-highest rank is shared with poissonmbo_old and sparselabelpropagation. The four algorithms above have high variance in their rank, which explains the rank uncertainty.

According to 7.8, the highest-ranking algorithm with access to 25% available labels is sparselabelpropagation followed by poissonmbo, Balance poissonmbo, and poissonmbo_old.

Table 7.9 illustrates that the highest-ranking algorithm with access to 50% available labels is poissonmbo, and the second-highest algorithms spot is shared between centered kernel, Balance Poisson, and Sparse label propagation. The third highest is poissonmbo_old and the fourth-highest algorithm is randomwalk. The second highest ranking algorithms have a higher variance in their ranks, which is why they are uncertain.

According to Table 7.10, the highest-ranking algorithm with access to 75% available labels is sparselabelpropagation followed by poissonmbo, poissonmbo_old, and randomwalk.

According to Table 7.11, the highest-ranking algorithm when having access to 90% labels is shared by poissonmbo, randomwalk, and sparselabelpropagation. The algorithm with the second-highest ranking is Balanced poissonmbo, the third-highest is poissonmbo_old and the fourth-highest algorithm is centered kernel. The highest-ranking algorithms have much higher variance in their ranks, hence the uncertainty in their median rank.

Figures that illustrate the distribution of the strength parameters and their HPDIs are found in the appendix.

RQ4: Probability of success

This research question was answered w.r.t to both aggregated results, datatype comparison, and manual effort. Hence, to answer RQ4 we performed the following operations on all three of the datasets that were used to answer the previous RQs. First, we made a copy of the dataset and added a new column called "SD" (for standard deviation) in both copied and original variants. In the original dataset, we put $SD = 0$ to indicate the absence of noise. In the copied dataset we put $SD = 3$ to indicate noise in the data. To add noise for each instance we replace the accuracy y with a simulated value of normal distribution with mean y and standard deviation 3. After the operations on the copied dataset, both datasets were concatenated by row into a new dataset. The odds ratios for each algorithm's intercept a_{alg} and noise b_{noise}

are computed using a new dataset.

Figures 7.4,7.5(a),7.5(b),7.5(c),7.5(d),7.5(e),7.5(f),7.5(g),7.5(h) contain the HPDIs for the OR of the intercept.

Figures that contain the HPDIs for the OR of noise and tables containing the numerical estimates of the intercepts, noise, and standard deviation can be found in the appendix.

According to Figure 7.4, all algorithms have a high mean OR above one except for Poisson2. OR > 1 means that Poisson2 is the only algorithm whose intercept parameter does not increase the probability of success. When it comes to the noise parameters, they all have an OR ≈ 0.2 or less. Hence every algorithm performs poorly in the presence of noise. This is especially true for randomwalk, poissonmbo, and poissonnmbo_old

For image datatypes Figure 7.5(a) shows that all algorithms have a high mean OR > 1 so all algorithms increase the probability of success. Every algorithm has an OR of less than 0.09 so all algorithms perform poorly in the presence of noise.

For text datatypes Figure 7.5(b) shows that all algorithms have OR less than one, so these algorithms do not increase the probability of success. The mean OR of the noise parameters is around 0.2 hence every algorithm performs poorly in the presence of noise.

For numerical datatypes, Figure 7.5(c) shows that laplace, mean_shifted_laplace, poissonmbo, poissonmbobalanced are the algorithms with mean OR > 1, so these are the algorithms that will have a high probability of achieving an accuracy higher than 90%. The mean OR of the noise parameters is around 0.2 hence every algorithm performs poorly in the presence of noise.

When having access to 10% available labels, Figure 7.5(d) shows that all algorithms have mean OR less than one, so every algorithm does not increase the pro ability of success. Every algorithm has a mean OR around 0.2, so every algorithm performs poorly in the presence of noise.

When having access to 25% of available labels, Figure 7.5(e) shows that centeredkernel, poisson, poisson2, poissonbalanced and poissonvolume are the only algorithms that do not have mean OR greater than one. These algorithms do not increase the probability of success. All noise parameters have a mean OR around 0.1, so every algorithm performs poorly in the presence of noise.

When having access to 50% of available labels, Figure 7.5(f) shows that every algorithm except poisson2 has a mean OR above one, so every algorithm

except poisson2 increases the probability of success. Every noise parameter has a mean OR around 0.1 so every algorithm performs poorly in the presence of noise.

When having access to 75% of available labels, Figure 7.5(g) shows that poisson, poisson2, and poissonbalanced are the only parameters with mean OR less than 1. Hence these are the only algorithms that do not increase the probability of success. Every noise parameter has a mean OR around 0.1 hence every algorithm performs poorly in the presence of noise.

When having access to 90% of available labels, Figure 7.5(h) shows that laplace, mean_shifted_laplace, poisson2, poissonbalanced, poissonvolume and wnll have mean OR less than 1. Hence, all of these algorithms do not increase the probability of success. Every noise parameter has a mean OR close to 0, most around 0.15 so every algorithm performs poorly in the presence of noise.

Based on the observations above we can see the following: For aggregated data, the highest-ranking algorithm is sparselabelpropagation. This algorithm also increases the probability of achieving an accuracy of at least 90%. Regarding datatypes, the top-ranked algorithms for image, text, and numeric are centered kernel, randomwalk, and poissonmbo. These algorithms, however, only increase the probability of getting an accuracy above 90% for image and numeric data. For text data, no algorithm increases the probability of achieving an accuracy above 90%. For manual effort the highest-ranking algorithm for 10% of available labels is poissonmbo but no algorithm increases the probability of achieving an accuracy higher than 90%. For 25%, 50%, 75%, and 90% of available labels, the top-ranking algorithm that increases the probability of achieving 90% accuracy is Sparse label propagation or poissonmbo.

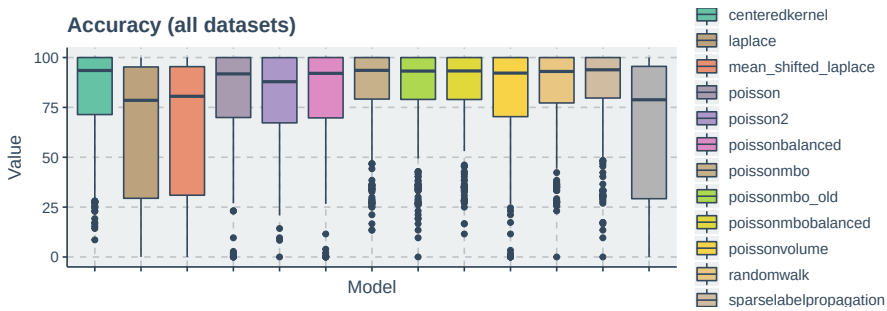


Figure 7.1: Boxplot illustrating the accuracy of all algorithms

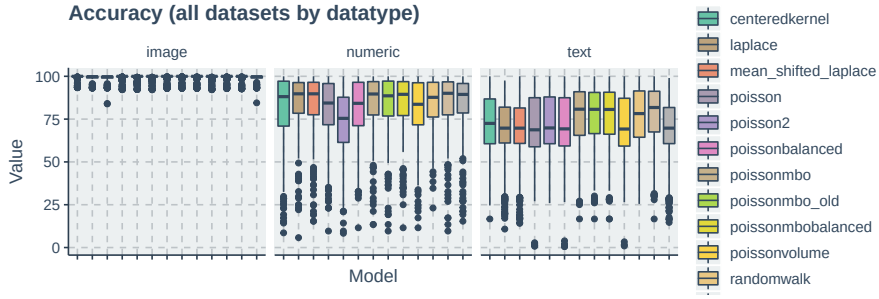


Figure 7.2: Boxplot illustrating the accuracy of all algorithms based on datatype

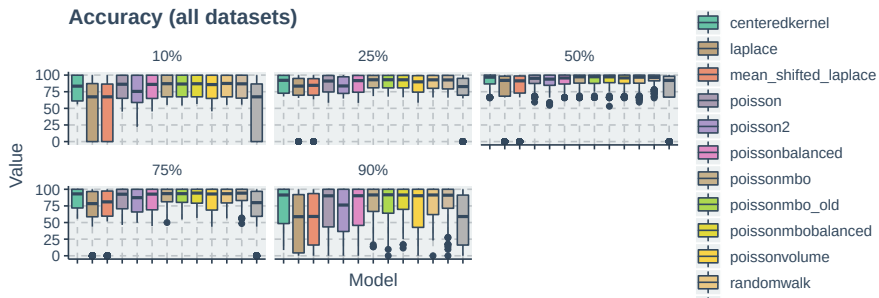


Figure 7.3: Boxplots illustrating the accuracy of all algorithms. From left to right in the upper row, we have boxplots for 10%,25% and 50%. From left to right in the lower right, we have 75% and 90% available labels

7.5 Discussion

The fast pace of new method development and publication dictates the need for continuous benchmarking. Furthermore, benchmarking is only able to evaluate methods implemented in a current release of the software. New releases of a method can differ in accuracy and runtime, suggesting a wide need for a permanent benchmarking effort. In addition to accounting for new method development, the benchmarking practice also needs to incorporate changes in the datasets. Routinely updating a benchmarking study may require develop-

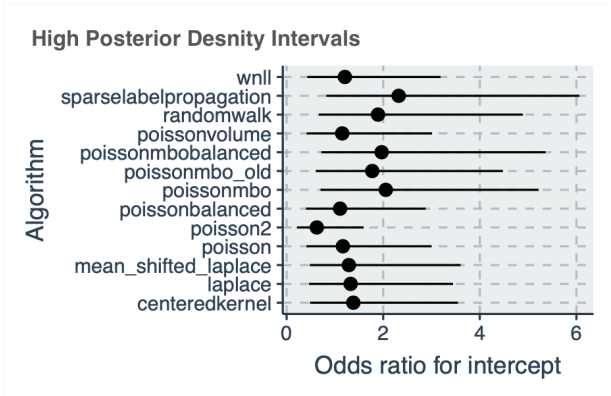
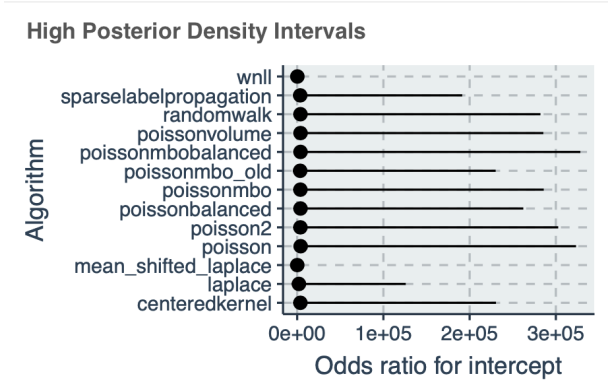
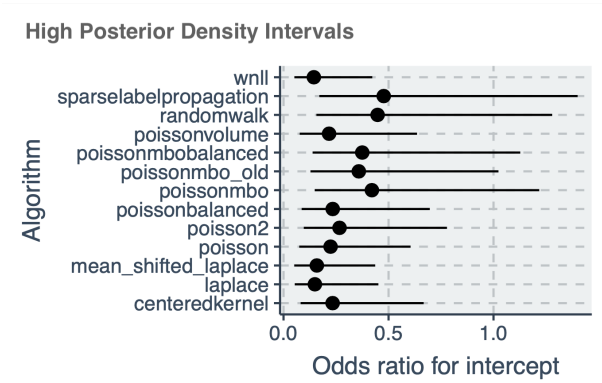


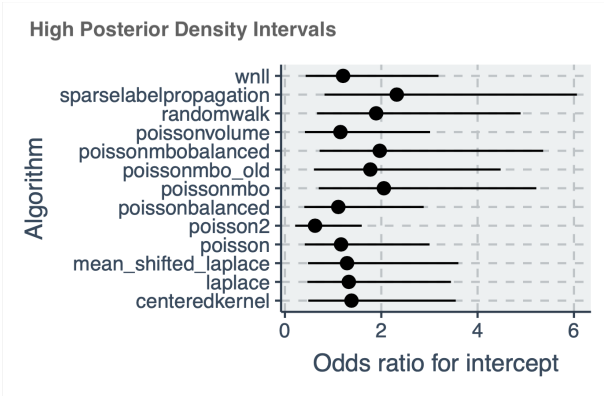
Figure 7.4: HPDIs of the ORs for each algorithm intercept for aggregated data.



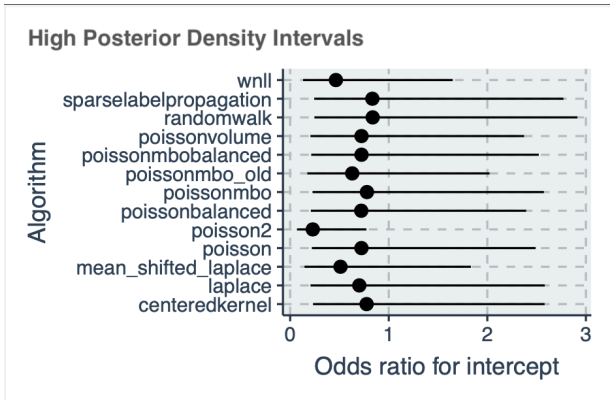
(a) HPDIs of the ORs for each algorithm intercept for image datatype.



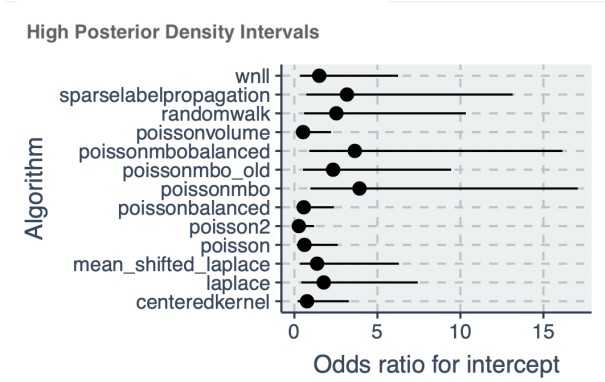
(b) HPDIs of the ORs for each algorithm intercept for text datatype.



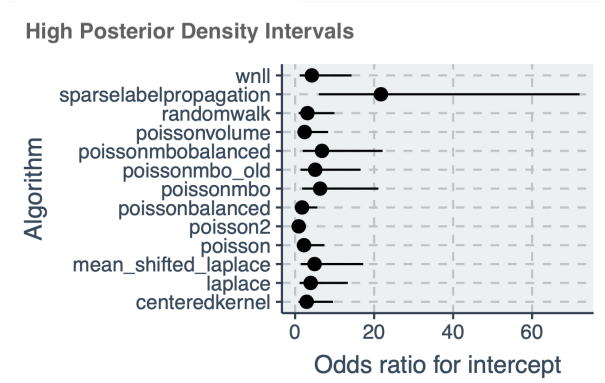
(c) HPDIs of the ORs for each algorithm intercept for numeric datatype.



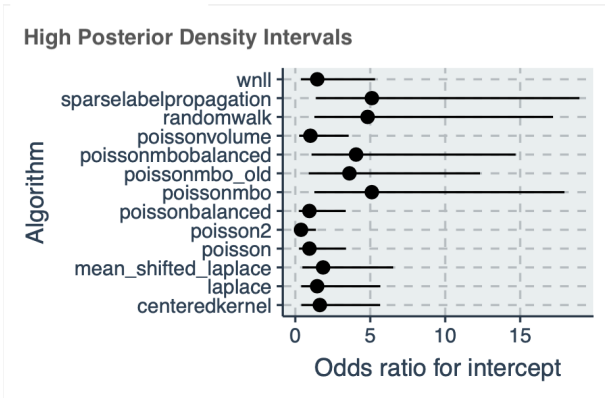
(d) HPDIs of the ORs for each algorithm intercept 10% available labels.



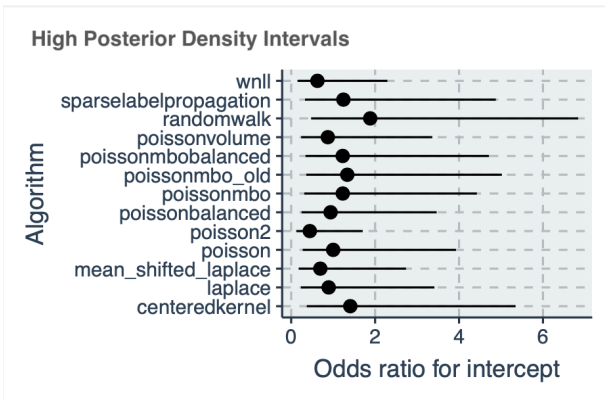
(e) HPDIs of the ORs for each algorithm intercept 25% available labels.



(f) HPDIs of the ORs for each algorithm intercept for 50% available labels.



(g) HPDIs of the ORs for each algorithm intercept for 75% available labels.



(h) HPDIs of the ORs for each algorithm intercept for 90% available labels.

Table 7.2: Ranking of the algorithms

Models	Median Rank	Variance of the Rank
sparselabelpropagation	1	0.276
poissonmbo	2	0.297
poissonmbo_old	3	0.293
poissonmbobalanced	4	0.269
randomwalk	5	0.000
centeredkernel	6	0.000
poissonbalanced	7	0.185
poisson	8	0.341
poissonvolume	9	0.212
laplace	11	0.595
mean_shifted_laplace	11	0.668
wnll	11	0.653
poisson2	13	0.000

Table 7.3: Ranking of the algorithms of image datatype

Models	Median Rank	Variance of the Rank
centeredkernel	1	0.000
poissonmbo_old	2	0.443
laplace	4	2.012
sparselabelpropagation	4	1.576
poissonmbobalanced	5	1.504
wnll	6	1.907
poissonmbo	7	1.214
mean_shifted_laplace	8	0.773
randomwalk	9	0.076
poisson	10	0.610
poissonvolume	11	0.644
poissonbalanced	12	0.445
poisson2	13	0.000

ers to determine the intersection between the previous and current datasets. Datasets can be updated or discarded with time due to their ability to eval-

Table 7.4: Ranking of the algorithms of text datatype

Models	Median Rank	Variance of the Rank
randomwalk	1	0.297
poissonmbobalanced	2	0.448
poissonmbo	3	0.755
poissonmbo_old	4	0.678
sparselabelpropagation	5	0.608
poissonbalanced	6	0.124
poisson	7	0.310
poissonvolume	8	0.193
centeredkernel	9	0.533
poisson2	10	0.714
laplace	11	0.735
mean_shifted_laplace	12	0.233
wnll	13	0.021

Table 7.5: From left to right, columns contain the abbreviation of the algorithm, the median rank and the variance of the ranks.

uate the algorithms [201]. No labeling problem should be considered solved at any given time, and continuous benchmarking needs to be performed to inform the user about the best algorithms available for a problem.

7.6 Conclusion

This study aims to analyze automatic labeling algorithms based on machine learning, laying out a framework of what algorithms should be used in industrial situations.

In general, practitioners should primarily investigate the Poisson MBO algorithm as it is the one algorithm that always lies in the top-3 highest ranking algorithms for every scenario except for image datasets, where there is high uncertainty in what algorithm is best as they all perform well. For aggregated results, Poisson MBO is only outperformed by Sparse Label Propagation. For 10%, 50%, 90%, the Poisson MBO is the highest-ranking algorithm, followed by Sparse Label Propagation. For 25% and 75%, Sparse Label Propagation

Table 7.6: Ranking of the algorithms of numeric datatype

Models	Median Rank	Variance of the Rank
poissonmbo	1	0.133
sparselabelpropagation	2	0.133
poissonmbobalanced	3	0.221
poissonmbo_old	4	0.357
wnll	5	0.660
laplace	6	0.747
mean_shifted_laplace	7	0.923
randomwalk	8	0.945
centeredkernel	9	0.529
poisson	11	0.656
poissonbalanced	11	0.583
poissonvolume	11	0.638
poisson2	13	0.000

Table 7.7: Ranking of the algorithms with 10% of available labels

Models	Median Rank	Variance of the Rank
poissonmbo	1	0.133
poissonmbobalanced	3	1.354
randomwalk	3	1.252
poissonmbo_old	4	1.302
sparselabelpropagation	4	1.261
poisson	6	0.493
poissonbalanced	7	0.743
centeredkernel	8	0.747
poissonvolume	9	0.578
wnll	10	0.474
laplace	11	0.579
mean_shifted_laplace	12	0.520
poisson2	13	0.000

is the highest-ranking algorithm, followed by Poisson MBO. Both algorithms have a high probability of achieving an accuracy higher than 90% in every

Table 7.8: Ranking of the algorithms with 25% of available labels

Models	Median Rank	Variance of the Rank
sparselabelpropagation	1	0.582
poissonmbo	2	0.699
poissonmbobalanced	3	0.741
poissonmbo_old	4	0.490
randomwalk	5	0.258
centeredkernel	6	0.258
laplace	7	0.371
mean_shifted_laplace	8	0.514
wnll	9	0.492
poisson	10	0.269
poissonbalanced	11	0.369
poissonvolume	12	0.270
poisson2	13	0.000

Table 7.9: Ranking of the algorithms with 50% of available labels

Models	Median Rank	Variance of the Rank
poissonmbo	2	1.090
centeredkernel	3	1.217
poissonmbobalanced	3	1.201
sparselabelpropagation	3	1.190
poissonmbo_old	5	0.071
randomwalk	6	0.000
poissonvolume	7	0.098
poissonbalanced	8	0.346
poisson	9	0.344
mean_shifted_laplace	10	0.372
laplace	11	0.423
wnll	12	0.215
poisson2	13	0.003

scenario except for text datasets and when having access to 10% data. The majority of times, Poisson MBO outperforms Sparse Label Propagation, but

Table 7.10: Ranking of the algorithms with 75% of available labels

Models	Median Rank	Variance of the Rank
sparselabelpropagation	1	0.147
poissonmbo	2	0.405
poissonmbo_old	3	0.389
randomwalk	4	0.314
poissonmbobalanced	5	0.277
centeredkernel	6	0.001
poisson	7	0.227
poissonbalanced	8	0.273
poissonvolume	9	0.069
poisson2	10	0.640
laplace	11	0.938
wnll	11	0.840
mean_shifted_laplace	13	0.464

Table 7.11: Ranking of the algorithms with 90% of available labels

Models	Median Rank	Variance of the Rank
poissonmbo	2	0.773
randomwalk	2	0.802
sparselabelpropagation	2	0.761
poissonmbobalanced	4	0.383
poissonmbo_old	5	0.392
centeredkernel	6	0.000
poissonvolume	7	0.069
poisson	8	0.313
poissonbalanced	9	0.324
poisson2	10	0.013
laplace	11	0.458
mean_shifted_laplace	12	0.539
wnll	12	0.548

both algorithms perform well, which is why we recommend practitioners try both algorithms. Lastly, every algorithm performs worse when there is noise

in the datasets, so practitioners should expect the algorithms not to perform equally well when applied to real-life datasets.

The results of this study help machine learning specialists to determine what algorithm will get the highest accuracy and most likely achieve an accuracy of at least 90%. Furthermore, the results also show the minimal manual effort needed to achieve the highest possible accuracy. In future simulation studies, we wish to examine these algorithms using other statistical models [171] to answer related RQs and other types of semi-supervised learning algorithms. Another interesting topic is to compare semi-supervised learning to transfer learning.

CHAPTER 8

Assessing the Suitability of Semi-Supervised Learning Datasets using Item Response Theory

8.1 Introduction

In the past ten years, machine learning has increased in usage across companies that have implemented or are in the process of implementing machine learning. Supervised learning is used for classification problems. In order to perform supervised learning, huge amounts of data is required. Each instance in the dataset must be associated with a label. Companies usually have access to large amounts of data, but the data is often incomplete in the sense that the data is partially missing labels [54]. Several labeling issues were identified in a case study performed with industry [167]. Labeling is costly as companies have to spent money on services such as crowdsourcing or in-house labeling [8], [9]. These are costs that they would rather spend on automated approaches. In a systematic literature review [153], several machine learning algorithms for labeling were investigated. One of the learning paradigms found in this study was semi-supervised learning. Graph-based algorithms, Mixture models and EM, Co-training and multi-view learning are the most popular semi-

supervised algorithms [153]. Semi-supervised learning algorithms are trained using both labeled and unlabeled instances to label unlabeled data.

Even if semi-supervised learning algorithms have been around for some time, they are unknown to most companies and as a consequence the usage of such algorithms is rare in industry. In order for practitioners to know what semi-supervised learning algorithms are the best to use, these algorithms needs to be evaluated on the right datasets. To the best of our knowledge, there is no taxonomy of datasets for evaluating semi-supervised learning algorithms.

Utilizing a simulation study we evaluated twelve datasets across thirteen different graph-based semi-supervised learning algorithms. The datasets where equally distributed across three different types, namely numerical, text and image data. The datasets where evaluated using a Bayesian congeneric item response theory model.

The contributions of this paper is two-fold. First, we propose the use of Bayesian congeneric item response theory model to assess the suitability of commonly used datasets. Second, we compare the different SSL algorithms using these datasets. The results show that with except of three datasets, the others have very low discrimination factors and are easily solved by the current algorithms. Additionally, we show that the SSL algorithms perform similarly under a 90% credible interval.

The remainder of this paper is organized as follows. In the following section we provide an overview of how graph-based semi-supervised learning and how the item-response theory works. In section 8.2 we describe the method that was used during this study. What datasets we used, how we performed the simulations and the metrics that was used to evaluate the accuracy of the algorithms and how we implemented the Item response theory. The results of our simulations are presented in section 8.5. The interpretations of these results are presented in section 8.6 the paper is concluded in 9. The online appendix provides the data and the reproducible code for the model fitting, figures and tables presented in this paper. The online appendix can found at: `https://davidissamattos.github.io/congeneric-irt-ssl/`

8.2 Background

In this section, we provide an overview of graph-based semi-supervised learning

Datatypes

In this paper we study datasets of three different types, numeric, image, and text-based datasets. See definitions below.

- **Image-based:** Datasets where each instance is represented by two-dimensional numerical arrays, known as pixels. Applications include face recognition, image retreival and image segmentation.
- **Text-based:** Datasets where the feature columns contain text. Applications include named entity recognition, information extraction and word sense disambiguation.
- **Numerical datasets:** Datasets where the features are numerical. Applications include activity recognition, network intrusion detection and structural health monitoring.

Semi-Supervised learning

Supervised learning algorithms only utilize labeled data. Semi-supervised learning utilizes both labeled and unlabeled data. In some cases a semi-supervised learning algorithm can outperform supervised learning algorithms. For more information on semi-supervised learning algorithms we refer the reader to [34]. We decided to study graph-based semi-supervised learning algorithms as these are the most popular according to [153].

The graph-based semi-supervised learning procedure can be summarized into three steps. First, a graph is constructed, secondly, seed labels are injected on a subset of nodes, the last step is to infer labels on the unlabeled nodes.

Given a set of labeled instances $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{n_u}$ and unlabeled instance $\mathcal{U} = \{\mathbf{x}\}_{i=1}^{n_l}$. The objective of the graph construction step is to find a graph $G = (V, E, W)$, where V are the verticies, E are the edges and $W = \{w_{ij}\}$ are the weights. The weights can be calculated using different algorithms such as kNN and ϵN methods as well as b-matching methods such as Binary, Gaussian Kernel and Localy Linear Reconstruction [69].

The second step is the seed injection step, several different algorithms has been proposed for this such as, graph cut [202], Gaussian fields [74], local and global consistency[75], adsorption[76], modified adsorption[77], quadratic criteria[78] and measurement propagation[35] are examples of different seed inferring methods.

8.3 Item Response Theory

Item response theory (IRT) corresponds to a family of statistical models developed to evaluate how latent traits of students (such as intelligence) when evaluated by a set of items (an exam). The foundations of item response theory comes from the idea of utilizing latent variables in education research from Binet (1905) and Thurstone (1925) and further developed by Lord (1952) and Rasch (1960) [203]. Since then, item response theory has been standard practice in the development of psychometric scales, national exams such as the Programme for International Student Assessment (PISA) and the National Assessment of Educational Progress (NAEP) [204]. Recent research has also suggested its use for evaluating the applicability of IRT for assessing datasets for machine learning research [49].

In this paper, we take the analogy from education to evaluate how SSL algorithms perform in different datasets. We consider the different SSL algorithms as students taking an exam. The exam consists of a selection of datasets. Each dataset corresponds to an item in the exam and each SSL model have a score for each item.

Next, we describe the simple dichotomous two-parameter IRT model to introduce some IRT concepts and then we proceed to introduce the congeneric IRT model used in this paper as well as estimation method.

The two-parameter logistic model

The two-parameter logistic model (2PL) response for dichotonomous items was introduced by Birnbaum (1968) [47] to model students abilities when taking an exam. Each item of the exam accepts only binary responses, correct or wrong.

The model assumes a latent trait variable (the ability θ) that will influence the probability of a test taker (student p) to correctly answer an item (i). In the 2PL, we model each item based on their difficulty level (b_i) and on the discrimination of the item (a_i) [203].

The model takes a logistic regression curve to estimate the probability of the test taker p to correctly answer the item i. The difficulty level shifts the logistic curve either to the left (easy items) or right (hard items). Easier items have higher probability to being correctly answered regardless of the difficulty level. Hard items require a much higher ability level of the respondent p to correctly

answer the question. The discrimination coefficient indicates the maximum slope of the logistic curve. A higher slope allows a shift in probability of correctly differentiating the respondents when their ability matches the item's difficulty.

The 2PL model is represent by the equations below [205]:

$$\pi_{p,i} = \frac{\exp a_i(\theta_p - b_i)}{1 + \exp a_i(\theta_p - b_i)} \tag{8.1a}$$

$$y_{p,i} \sim \text{Bernoulli}(\pi_{p,i}) \tag{8.1b}$$

In this model, we have the following notation:

- $\pi_{p,i}$ is the probability of an item i being correctly answered by test taker p.
- $y_{p,i}$ is the dichotomous response from test taker p on item i. The value of 1 is for a correct answer and 0 for a wrong answer.
- a_i is the discrimination parameter of item i
- b_i is the difficulty level of item i
- θ_p is the latent trait of the test taker p.

Despite the large applicability of this model in including in ML research [49], dichotomous models are not suitable for evaluating the accuracy of ML models since these variables are inherently continuous and any transformation on those can add significant bias to the results.

The congeneric model

To address the problem of using dichotomous variables, we utilize the Jöreskog's model for congeneric measurements, also called the congeneric model [206], [207].

The congeneric model assumes that the regression of the item score is modeled by a linear function on the latent variables. If assumed that only a single latent variable is present (as in the 2PL) this model takes the form of:

$$y_{p,i} \sim \mathcal{N}(\mu_{i,p}, \sigma^2) \tag{8.2a}$$
$$\mu_{i,p} = b_i + a_i\theta_p \tag{8.2b}$$

In this model, we have the following notation and interpretation of the parameters:

- $\mu_{p,i}$ is the average observation of score of an item i being answered by test taker p.
- $y_{p,i}$ is actual observation of the continuous response from test taker p on item i.
- a_i is the discrimination parameter of item i
- b_i is the difficulty level of item i. Constraining the b_i parameter as a positive value shifts the interpretation from difficulty to easiness of the item.
- θ_p is the latent trait of the test taker p.

It is worth noting that the items are modeled with a linear regression and a normal distribution of the errors. While this approach simplifies the interpretation of the model, it does not add constraint bounds on the observed values.

Bayesian estimation

The congeneric model can be both estimated using the maximum likelihood estimator procedure described in [207] or utilizing a Markov Chain Monte Carlo (MCMC) sampler in a Bayesian estimation procedure. In this paper, we utilize a Bayesian estimation method. Bayesian Data Analysis (BDA) has multiple advantages over the frequentist counterpart such as easier interpretation of the credible intervals, transparency of the model assumptions. The benefits of BDA have been widely discussed in research and an in-depth analysis is beyond the scope of this paper [52], [208]–[210].

We utilize as basis for our Bayesian congeneric IRT model, the model presented in equations 8.2. By adding normally distributed and weakly informa-

tive proper priors for all parameters [211] being estimate we arrive at following model:

$$y_{p,i} \sim \mathcal{N}(\mu_{i,p}, \sigma^2) \quad \text{[Likelihood]} \tag{8.3a}$$
$$\mu_{i,p} = b_i + a_i\theta_p \tag{8.3b}$$
$$a_i \sim \text{Half-Normal}(0, 1) \quad \text{[Prior]} \tag{8.3c}$$
$$b_i \sim \text{Half-Normal}(0, 1) \quad \text{[Prior]} \tag{8.3d}$$
$$\theta_p \sim \text{Half-Normal}(0, 3) \quad \text{[Prior]} \tag{8.3e}$$
$$\sigma \sim \text{Half-Normal}(0, 1) \quad \text{[Prior]} \tag{8.3f}$$

In this model, we use the following notation:

- $\mu_{p,i}$ is the average observation of score of an item i being answered by test taker p.
- $y_{p,i}$ is actual observation of the continuous response from test taker p on item i.
- a_i is the discrimination parameter of item i
- b_i is the easiness level of item i. Constraining the b_i parameter as a positive value shifts the interpretation from difficulty to easiness of the item.
- θ_p is the latent trait of the test taker p.

The presented model in equations 8.3 is implemented in Stan [212] and estimated with the No U-Turn Hamiltonian Monte Carlo algorithm [213]. The code related to the implementation and the data used can be found at the online appendix.

Assessment of the convergence and suitability of the model with predictive posterior checks [211] are presented in the online appendix. In section 8.5, we present the results of this model with credible intervals and median to summarize the posterior distribution.

8.4 Experimental Setup

The purpose of this paper to empirically evaluate the suitability of the datasets commonly used to evaluate and compare different SSL algorithms. We performed a simulation study using fifteen datasets of three different datatypes (numerical, text, image) on thirteen different SSL algorithms. The fifteen datasets are equally distributed across three the different types of data.

We explore the following research questions:

- **RQ1:** *What datasets are suitable to compare different graph-based SSL algorithms.*
- **RQ2:** *How can different graph-based SSL algorithms be compared.*

To compare the different graph-based SSL algorithms using IRT, we run the algorithms using a fixed percentage of labels each iteration. We let 10% of the data be labeled and the remaining of the 90% unlabeled.

Simulations

The datasets

We selected our datasets based on a systematic mapping study that is currently in proceedings. In this study the authors listed several data labeling algorithms such as active learning and semi-supervised learning algorithms. The study also contains 79 datasets that were commonly used to evaluate these algorithms. We choose twelve benchmarked datasets from the mapping study to be used in our study. These twelve datasets were the most popular and had the best availability. We choose four datasets of of each type.

- **Image:**
  - **Cifar-10:** The Cifar-10 dataset consists of 60000 images with 32×32 resolution. Each image contains one object that can be divided into ten categories, "airplane", "automobile", "bird", "cat", "deer", "dog", "frog", "horse", "ship", and "truck". The dataset has been used in many studies [214]–[216].
  - **Digits:** The digits dataset consists of 1797 images with 8×8 resolution. Each image contains a digit and there is a total of ten different classes representing each digit 0,1,2,...9 [217], [218].

  - **MNIST:** This is a subset of the NIST, a databese for handwritten text-recognition. Each image contains a handrwritten digit of class 0,1,2,...,9. The dataset consists of 60000, 28×28 resolution images. [219], [220].
  - **FashionMNIST:** This dataset contains Zalando's article images. The dataset consists of totally 60000 images with 28×28 resolution. Each image depicts an article from Zalando. There are ten classes of articles, T-shirt, Trouser, Pullover, Dress, Coat, Sneaker, Bag and Ankle boot. This dataset was indendent as a replacement to the classical MNIST dataset [221], [222]. .

- **Text:**
  - **Fake news:** This dataset contains news articles that can be divided into two classes, false and truthful articles. The dataset contains 44594 instances. There are three features, the "title", "subject" and the "text"[189], [190].
  - **20 Newsgroups:** This dataset consists of approximately 18000 instances of twnety classes. In this dataset we have selected only four categories "alt.atheism", "talk.religion.misc", "comp.graphics" and "sci.space" resulting in a dataset consisting of 2034 instances. [223]–[226].
  - **Ohsumed:** This dataset is a subset of the MEDLINE database and contains peer-review medical literature. The are 23 classed of medical subject headings and 50216 medical abstracts. In this paper we use a subset four medical subject headings and 5379 instances.
  - **Reuters:** This dataset consist of news documents divided into 90 categories and 9598 instances. In this paper we use a subset of six categories "Neg-", "Pos-acq", "Pos-coffee", "Pos-earn", "Pos-gold", "Pos-heat" [227]–[230].

- **Numeric:**
  - **Iris:** The purpose of this dataset was to distinguish different Iris flower species. The data consist of 150 instances equally distributed across thre classes, "Iris setosa", "Iris virginica" and "Iris versicolor" [231]–[233].

- **Wine:** This data is used for wine classification. It consist of 1797 instances and 64 features. The labels are divided across seventeen instances [234], [235].
- **MUSK:** This dataset is a subset of the MUSK dataset. It is used to classify molecules as "musks" or "non-musks". There are 476 instances and 166 features [236]–[238].
- **German:** This dataset describes german credit scores. It contains 522 instances and nine features. There are two classes "good" and "bad". The purpose of the dataset is to determine the quality of a persons credit score. [239]–[241].

Table 8.1: Summary table for the datasets.

Datatype	Dataset
Image	Cifar-10 Digits MNIST FashionMNIST
Text	Fake and truthful news 20news Ohsumed Reuters
Numeric	Iris Wine MUSK German

Each dataset was preprocessed so that the 10% of the labels were available. Each algorithm was run on the datasets ten times utilizing different seeds for each iteration. To store the results, a data frame with ten rows was created from running an algorithm on a dataset. The accuracy of the predictive labels were logged in each iteration and stored in a total of 195 data frames.

The SSL algorithms

It is recognized in [153] that graph-based semi-supervised learning is one of the most popular type of semi-supervised learning algorithms. We have included

thirteen different graph-based semi-supervised learning that are listed below. In bold we represent the short name used in the tables and figures.

The algorithms utilized in this study are based on Laplace learning [74], lazy random walks [75], multiclass MBO [242], weighted non-local Laplacian [243], volume contrained MBO [244], Centered kernel method, Sparse Label Propagation[245], Poisson and PoissonMBO algorithms [129]. All algorithms are implemented in Python using the GraphLearning package [1].

- Laplace learning **(laplace)** [74]
- Mean Shifted Laplace **(mean_shifted_laplace)** [129]
- Centered kernel method **(centeredkernel)** [246]
- Poisson learning **(poisson)** [129]
- Poisson learning, alternate version **(poisson2)**
- Balanced Poisson learning **(poissonbalanced)**
- Poisson MBO **(poissonmbo)** [129]
- Poisson MBO with volume contraints **(poissonvolumembo)**
- Balanced Poisson MBO [129]
- Poisson with volume constraints **(poissonvolume)**,[129]
- Random Walk **(randomwalk)** [75]
- Sparse Label Propagation **(sparselabelpropagation)** [245]
- Weighted non-local Laplacian **(wnll)** [243]

To calculate the accuracy of each algorithm we assume that a fixed percentage of the instances are already labeled and the rest is unlabeled. The accuracy is calculated by

$$\varepsilon = \frac{100}{n-m} \max\left(\sum_{i=1}^{n} I(y_i = \widehat{y}_i) - m, 0\right),$$

[1] https://github.com/jwcalder/GraphLearning/blob/master/graphlearning/graphlearning.py

where $I(x)$ is **indicator function** defined as

$$I(x) = \begin{cases} 1 \text{ if } x \text{ is true} \\ 0 \text{ otherwise} \end{cases} ,$$

n is the number of both labeled and unlabeled instances and m is the number of labeled instances. The purpose of the algorithms is to predict the labels of all of these unlabeled instances.

Threats to Validity

A treat to validity is that the accuracy of our semi-supervised algorithms might be compromised as we have not considered whether the class labels are balanced or not for all datasets [247].

8.5 Results

In this section, we present the results obtained from the parameter estimation of the Bayesian congeneric model represented by equations 8.3. We first present the estimated easiness and the discrimination parameters of the items (i.e. the datasets). Next, we present the ability level of the test taker (i.e. the SSL models)

Items parameters

Utilizing the data collected and discussed in section 8.4, table 8.2 shows the parameters for the easiness level and the discrimination level. The columns CI 5% and CI 95% represent the lower and higher of the 90% credible intervals, respectively. This table is can be visualized in figure 8.1.

We can see from table 8.2 and figure 8.1 that the easiness parameter is almost 1 for most datasets. These datasets are solved by most SSL algorithms. Analogously, most datasets have a discrimination parameter close to zero indicating that the SSL models obtain very similar results under these datasets. These datasets with high level of easiness and very low level of discrimination are not suitable for the comparison between different SSL models.

Table 8.2: Posterior summary values of the discrimination and easiness level parameters for the datasets

Dataset	Median	CI 5%	CI 95%
Discrimination value (a)			
20news	0.015	0.004	0.047
cifar	0.004	0.000	0.018
corpus	0.004	0.000	0.018
digits	0.005	0.000	0.020
fashionmnist	0.004	0.000	0.018
german	0.001	0.000	0.008
iris	0.005	0.001	0.022
mnist	0.011	0.002	0.035
musk	0.157	0.066	0.413
ohsumed	0.186	0.078	0.491
reuters	0.462	0.193	1.201
wine	0.005	0.000	0.020
Easiness level (b)			
20news	0.852	0.837	0.861
cifar	0.996	0.988	1.001
corpus	0.578	0.569	0.582
digits	0.990	0.981	0.995
fashionmnist	0.996	0.988	1.001
german	0.594	0.590	0.598
iris	0.993	0.983	0.998
mnist	0.977	0.964	0.985
musk	0.637	0.546	0.688
ohsumed	0.396	0.289	0.456
reuters	0.312	0.047	0.462
wine	0.993	0.984	0.998

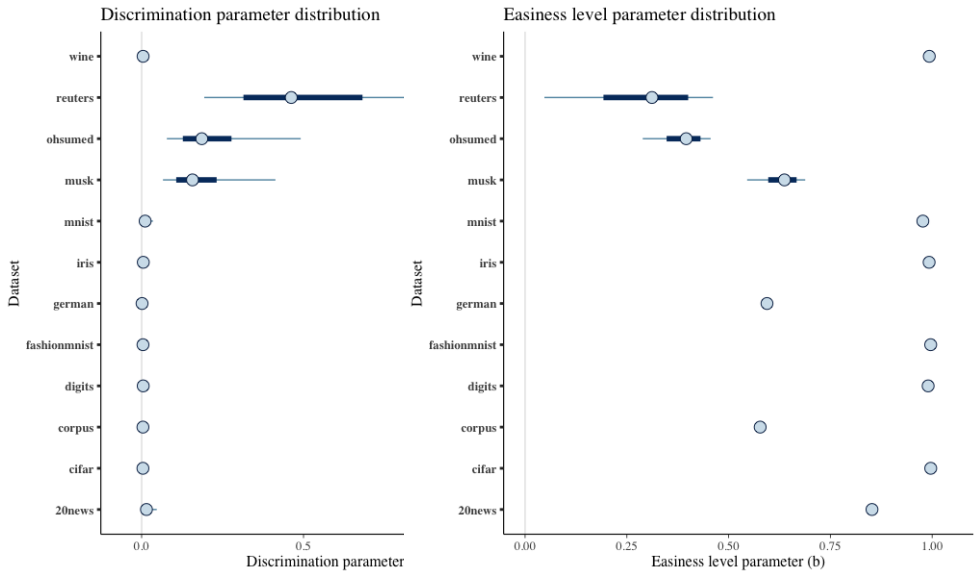


Figure 8.1: The posterior distribution of the discrimination and easiness parameters per dataset. The circles represent median value, the thick dark blue lines represent the 50% probability mass and the thin light blue line represent the 90% probability mass.

The ability parameters

Table 8.3 shows the ability parameters SSL models. The columns CI 5% and CI 95% represent the lower and higher the 90% credible intervals, respectively. This table can be visualized in figure 8.2.

We can see in table 8.3 and figure 8.2 that while there are differences between the ability parameters of the SSL models, they have large overlapping intervals, in particular in the 50% probability mass (thick dark blue line). This overlapping indicates an uncertainty in the difference between the accuracy of these SSL models.

However, it is worth noting that this large uncertainty comes from the fact that the choice of the datasets with high easiness and low discrimination parameters does not help in the proper selection of the correct SSL model.

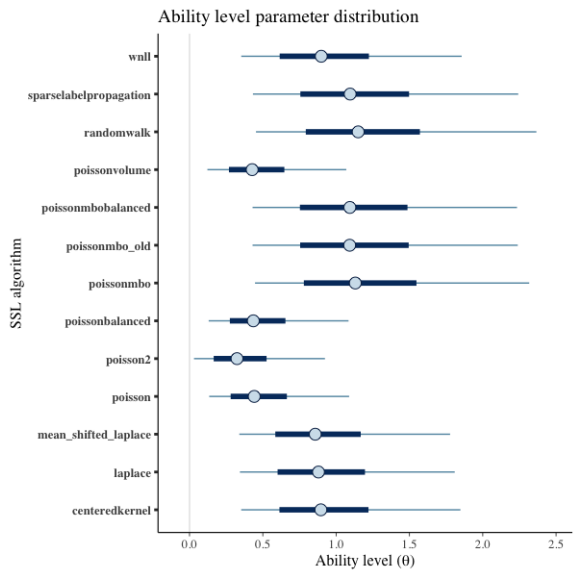


Figure 8.2: The posterior distribution of the ability parameter of the SSL models. The circles represent median value, the thick dark blue lines represent the 50% probability mass and the thin light blue line represent the 90% probability mass.

Table 8.3: Posterior summary values of the ability level of the SSL models

Model	Median	CI 5%	CI 95%
centeredkernel	0.896	0.353	1.848
laplace	0.880	0.344	1.809
mean_shifted_laplace	0.858	0.340	1.777
poisson	0.441	0.134	1.089
poisson2	0.324	0.031	0.924
poissonbalanced	0.435	0.131	1.085
poissonmbo	1.131	0.447	2.317
poissonmbo_old	1.092	0.430	2.239
poissonmbobalanced	1.094	0.430	2.234
poissonvolume	0.427	0.122	1.069
randomwalk	1.150	0.453	2.366
sparselabelpropagation	1.096	0.433	2.242
wnll	0.898	0.353	1.856

8.6 Discussion

As pointed out in section 8.1, companies are in need for automatic labeling algorithms. When performing data exploration to determine what algorithm is the best to use, it is essential to evaluate different algorithms on the the most optimal datasets.

IRT has previously been used to evaluate datasets for supervised learning classifiers [248]. In [248], 60 datasets from the well-known OpenML-CC18 benchmark are evaluated. The results show that 88% of the datasets are easy and 60% are not discriminating, hence not suitable for evaluating supervised learning algorithms.

According to Table 8.1, most of the datasets in our study have a high easiness parameter and a low discrimination parameter. This means that these datasets are not suitable for evaluating semi-supervised learning algorithms. The Corpus and German datasets have low discrimination parameter despite a medium easiness parameter. This means that algorithms with different ability parameters will have the same accuracy on these datasets.

According to Table 8.3, the algorithms with the highest ability parameters are random walk, Poisson MBO and sparse label propagation. In Figure 8.2

we can see that the C.I intervals of the algorithms overlap. This is because the datsets have low discrimination values. If the datasets would have high discrimination value, then it would be easy to differentiate the algorithms by estimating their ability parameter.

8.7 Conclusion

The goal of these simulations is to provide an overview over what datasets should be used to evaluate semi-supervised machine learning algorithms to automatically label data in an industrial setting. According to our results Reuters, Ohsumed and Musk are the only datasets suitable for evaluating semi-supervised learning algorithms. Diverse datasets with good discrimination reduces the uncertainty of the algorithms ability. This means that it is easy to determine what algorithms are better. According to our results we can say that it does not matter what semi-supervised learning algorithm we choose for labeling because our datasets are not very discriminitative.

Among the suitable datasets, Reuters and Ohsumed are text-based datasets and Musk is numerical. We have not identified any suitable image-based datasets. Therefore, in future research we want to investigate more image-based datasets to determine what datasets should be used to evaluate semi-supervised learning algorithms.

CHAPTER 9

Concluding Remarks and Future Work

In this chapter we present concluding remarks and the future direction of the research. In this thesis we present challenges and mitigation strategies for data labeling. In order to enable high-accuracy machine learning algorithms without spending unnecessary resources on data labeling, we provide a detailed overview of what semi-supervised learning algorithms are the most optimal for different applications. To make it easier for the industry to evaluate their semi-supervised learning algorithm, we provide the most suitable datasets for evaluating semi-supervised learning algorithms. Four papers are included in this thesis, each discussing aspects of the challenges and mitigation strategies for data labeling. The end of the chapter presents the future direction of doctoral research.

9.1 Conclusion

In this thesis exploratory research is done to discover and analyze challenges and mitigation strategies for data labeling. After formulating challenges and mitigation strategies, a subset of semi-supervised learning algorithms is empirically validated. Furthermore, we provide an overview of algorithms applicable

to many datasets and suggest suitable datasets for evaluating semi-supervised learning algorithms. The background and theory for the discussed algorithms are presented. The research objective and methods utilized are presented, as well as a discussion of threats to validity.

RQ1: What data labeling challenges exist in the industry, and how can they be mitigated using Machine Learning for Data Labeling?

This research question was answered by utilizing a case study in which data was collected from two companies. The data collecting procedure was divided into two phases. First was the exploration phase, in which time was spent with company A. The second phase was the validation phase. Here, we conducted interviews with company participants. During phase I and II we observed different problems that we summarized into three challenges. A different mitigation strategy was formulated for each challenge based on active and semi-supervised learning practices. Thanks to the mitigation strategies, companies have been given the tools necessary to solve challenges concerning data labeling. The three different challenges are *Pre-processing Annitation* and *label Distribution* (C1-C3). For each challenge we plan mitigation strategies called *Planning, Oracle selection* and *Label Distribution* (MS1-MS3). The first challenge involves the planning of the labeling procedure. Practitioners are required to perform suitable exploratory analysis to do labeling with respect to different tasks and choose the correct model based on the data structure. The choice of query strategy is dependent on the underlying ML algorithm. If the underlying ML algorithm reduces expected error or variance, the query strategy with the same name should be used. If detecting a cluster structure in the labels is possible, choose a density-weighted AL or GSSL. For probabilistic models such as GMM, uncertainty sampling should be used. If the labels are to be used for different tasks, then multi-task active learning should be utilized. To account for labeling costs, cost-sensitive active learning can be used. Labeling costs might vary over time and cost-sensitive AL allows us to model the labeling cost as a deterministic or stochastic function. Suppose that data is generated from an experiment due to actual data being too expensive to acquire. Predicting "actual" data based on generated data will therefore predict noisy labels. In many cases some instances will be difficult to label

manually. Some people have a limited attention span and therefore the quality of the labels will decrease over time. Crowdsourcing can be a good solution as it allows several people to label the instances, making it easier to detect errors. Another solution is to use repeated labeling to reduce uncertainty in the oracle and model. The label distribution challenge (C3) involves problems concerning uncertainty in the label distribution such as skewness. If the label distribution is skewed then AL might not outperform PL. A solution for this is to search for class representative instances using guided learning which can create a more balanced dataset.

RQ2: What Machine Learning algorithms are available for Data Labeling?

This research question was answered by studying previous research on machine learning algorithms that will reduce or eliminate manual data labeling. We identified the most popular machine learning algorithms by conducting a systematic mapping study. During the literature search, we identified 87 datasets that are used to evaluate these algorithms. The datasets are distributed across four datatypes, image, text, sound, and numerical. The majority of these datasets are images and numerical. We provide a taxonomy of algorithms consisting of active learning and semi-supervised learning. We outline the applications where these algorithms are used and conclude what algorithms are used for a specific application. The classes of SSL algorithms are *GSSL, GMM, MVL* and *S3VM*, and the classes of AL algorithms are *uncertainty sampling, density-weighted method, expected variance reduction* and *QBC*. SSL algorithms all rely on different assumptions on the dataset. Theoretically, if the dataset only satisfies a few assumptions, then not every algorithm class will work on the dataset. Because of the difference in assumptions could be why algorithms of different classes are not evaluated together. Al algorithms do not rely on different assumptions on the data but on the algorithm. The optimal AL algorithm varies but in almost every case AL outperforms random sampling. All algorithms have been primarily evaluated on image data and secondly on text data. Fewer studies have been found involving numerical data. Only two studies involved sound data, one utilizing GSSL and the other S3VM. No AL algorithms were applied to sound data. Uncertainty sampling was applied to images and text datasets but not to numerical data. QBC algorithms were mostly used on text and a few studies

involved numeric data.

RQ3: What Machine Learning algorithms for Data Labeling are optimal for achieving high accuracy while maintaining low labeling costs?

We performed a benchmark study to investigate which algorithms yield the highest accuracy while maintaining low labeling costs. From the 87 datasets found in our systematic mapping study, we collected the 24 most commonly used datasets publicly available for download to use in the simulations. The datasets were equally distributed among image, text, and numeric data. According to a systematic literature review, graph-based algorithms are the most commonly used SSL algorithm. We collected 13 graph-based algorithms to use in the simulations. To account for how much manual labeling is required, we varied the number of training the number of labeled instances used during training. Furthermore, we ran each simulation ten times using different seeds. During each simulation we recorded the accuracy of the algorithms. We ranked the algorithms according to their accuracy by utilizing the Bayesian Bradley-Terry Model. Based on the results the top algorithms are Sparse Label Propagation and Poisson MBO. These algorithms will reach an accuracy higher than 90% for most datasets when at least 25% of the instances are labeled except for text datasets. The results will help practitioners in the industry to choose the algorithm that will most likely achieve an accuracy of at least 90% while minimizing the manual effort required to achieve such accuracy.

RQ4: Do benchmark datasets contribute to a fair evaluation of Graph-based Semi-Supervised algorithms?

When answering RQ3, we notice that the algorithms performed exceptionally well on particular image datasets. Are some of these datasets too easy for the algorithms to learn? To answer this research question we took the data collected from the simulations of RQ3. The data was analyzed using item response theory. Item Response Theory is a common tool in educational research to investigate student responses to questions on exams. Our analysis views datasets as the items and the algorithms as the students. Therefore the analysis will measure the algorithm's ability to learn different datasets.

According to our results, Reuters, Ohsumed and Musk are the only suitable datasets. The first two datasets are text-based and the third is numerical. The other datasets are too easy to learn and it is difficult to determine which algorithm is better based on these datasets.

9.2 Future Work

So far, we have studied many algorithms that will help practitioners reduce labeling costs. The convergence properties of these algorithms have been theoretically proven through asymptotical analysis. Based on experience, we have noticed that semi-supervised learning algorithms do not always outperform supervised learning and that negative results tend not to be published. It is unclear whether unpublished negative results are the cause of semi-supervised learning not receiving much attention in the industry. According to our experience with the industry, some companies know about active learning but do not know about semi-supervised learning. Therefore it is difficult to say why the industry does not utilize semi-supervised learning. Many companies are not interested in active learning as it involves manual data labeling.

A large interest in deep semi-supervised learning has risen in the few years since we started researching semi-supervised ML. One of the most popular deep SSL algorithms is FixMatch, a robust algorithm that has proven to yield an accuracy of 86% utilizing 40 labeled instances of the CIFAR10 dataset. Other deep semi-supervised learning has taken inspiration from FixMatch to transfer its strengths to other tasks. The PseCo algorithm transfers the ideas of FixMatch to object detection tasks. According to a survey on deep semi-supervised learning algorithms, there are five types of deep semi-supervised learning algorithms, Deep Generative Models, Graph-based methods, Consistency Regularization methods, Pseudo-Labeling methods, and hybrid methods. The survey compares the latter three types of algorithms and finds that Hybrid methods have the best in performance. FixMatch is among these hybrid methods. Although the survey summarizes the performance of algorithms, the comparisons are not fair as it is unclear how they are compared to their supervised counterpart. Some of these algorithms are not evaluated on the same datasets or the same number of labeled instances. New software packages implementing these high-performance deep semi-supervised learning algorithms have been publicly available. Another new interesting topic is

safe semi-supervised learning. SSL can improve SL under some conditions. Several empirical and theoretical results show that SSL could also degrade performance. Even if SSL algorithms can degrade performance, they are still relevant for practitioners in the industry as they sometimes are willing to sacrifice accuracy for less manual labeling effort. In future research we wish to provide the industry with more labeling tools and to apply safe SSL and deep SSL on datasets from the industry. Utilizing statistical analysis and item response theory we intend to provide a more fair evaluation of algorithms across many dimensions and assess whether the datasets used to evaluate these algorithms are suitable for evaluating safe and deep SSL algorithms.

References

[1] A. K. Tyagi *et al.*, "Machine learning with big data," in *Machine Learning with Big Data (March 20, 2019). Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India*, 2019.

[2] H. Cloud Factory, *Crowd vs. managed team: A studo on quality data processing at scale*, https://go.cloudfactory.com/hubfs/02-Contents/3-Reports/Crowd-vs-Managed-Team-Hivemind-Study.pdf, 2020.

[3] S. Shafaei, S. Kugele, M. H. Osman, and A. Knoll, "Uncertainty in machine learning: A safety perspective on autonomous driving," in *Computer Safety, Reliability, and Security: SAFECOMP 2018 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Västerås, Sweden, September 18, 2018, Proceedings 37*, Springer, 2018, pp. 458–464.

[4] J. Zheng, D. Lin, Z. Gao, S. Wang, M. He, and J. Fan, "Deep learning assisted efficient adaboost algorithm for breast cancer detection and early diagnosis," *IEEE Access*, vol. 8, pp. 96 946–96 954, 2020.

[5] Cognilytica Research, "Data Preparation & Labeling for AI 2020," Cognilytica Research, Tech. Rep., 2020, pp. 1–35.

[6] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: A big data-ai integration perspective," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[7] H. S. Alenezi and M. H. Faisal, "Utilizing crowdsourcing and machine learning in education: Literature review," *Education and Information Technologies*, vol. 25, no. 4, pp. 2971–2986, 2020.

[8] J. C. Chang, S. Amershi, and E. Kamar, "Revolt: Collaborative crowdsourcing for labeling machine learning datasets," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 2334–2346.

[9] J. Zhang, V. S. Sheng, T. Li, and X. Wu, "Improving crowdsourced label quality using noise correction," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 5, pp. 1675–1688, 2017.

[10] P. G. Ipeirotis, "Analyzing the amazon mechanical turk marketplace," *XRDS: Crossroads, The ACM magazine for students*, vol. 17, no. 2, pp. 16–21, 2010.

[11] E. Mosqueira-Rey, E. Hernandez-Pereira, D. Alonso-Rios, J. Bobes-Bascaran, and A. Fernandez-Leal, "Human-in-the-loop machine learning: A state of the art," *Artificial Intelligence Review*, pp. 1–50, 2022.

[12] B. Settles, "Active learning, volume 6 of synthesis lectures on artificial intelligence and machine learning," *Morgan & Claypool*, 2012.

[13] X. Wu, L. Xiao, Y. Sun, J. Zhang, T. Ma, and L. He, "A survey of human-in-the-loop for machine learning," *Future Generation Computer Systems*, 2022.

[14] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 3, no. 1, pp. 1–130, 2009.

[15] T. Fredriksson, J. Bosch, and H. H. Olsson, "Machine learning models for automatic labeling: A systematic literature review.," *ICSOFT*, pp. 552–561, 2020.

[16] H. Zhang, L. Zhang, and Y. Jiang, "Overfitting and underfitting analysis for deep learning based end-to-end communication systems," in *2019 11th international conference on wireless communications and signal processing (WCSP)*, IEEE, 2019, pp. 1–6.

[17] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[18] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[19] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*. MIT press Cambridge, MA, USA, 2017, vol. 1.

[20] L. G. Valiant, "A theory of the learnable," *Communications of the ACM*, vol. 27, no. 11, pp. 1134–1142, 1984.

[21] L. Atlas, D. Cohn, and R. Ladner, "Training connectionist networks with queries and selective sampling," *Advances in neural information processing systems*, vol. 2, 1989.

[22] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Machine learning*, vol. 15, no. 2, pp. 201–221, 1994.

[23] I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Machine Learning Proceedings 1995*, Elsevier, 1995, pp. 150–157.

[24] D. Angluin, "Queries and concept learning," *Machine learning*, vol. 2, no. 4, pp. 319–342, 1988.

[25] ——, "Queries revisited," in *International Conference on Algorithmic Learning Theory*, Springer, 2001, pp. 12–31.

[26] E. B. Baum and K. Lang, "Query learning can work poorly when a human oracle is used," in *International joint conference on neural networks*, Beijing China, vol. 8, 1992, p. 8.

[27] D. E. Graff, E. I. Shakhnovich, and C. W. Coley, "Accelerating high-throughput virtual screening through molecular pool-based active learning," *Chemical science*, vol. 12, no. 22, pp. 7866–7881, 2021.

[28] N. Khoshnevis and R. Taborda, "Application of pool-based active learning in physics-based earthquake ground-motion simulation," *Seismological Research Letters*, vol. 90, no. 2A, pp. 614–622, 2019.

[29] A. Lang, C. Mayer, and R. Timofte, "Best practices in pool-based active learning for image classification," 2021.

[30] T. M. Mitchell, "Generalization as search," *Artificial intelligence*, vol. 18, no. 2, pp. 203–226, 1982.

[31] P. Melville and R. J. Mooney, "Diverse ensembles for active learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 74.

[32] G. Ngai and D. Yarowsky, "Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking," *arXiv preprint cs/0105003*, 2001.

[33] N. Cristianini, J. Shawe-Taylor, *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[34] X. J. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2005.

[35] A. Subramanya and J. Bilmes, "Semi-supervised learning with measure propagation.," *Journal of Machine Learning Research*, vol. 12, no. 11, 2011.

[36] M. Karlen, J. Weston, A. Erkan, and R. Collobert, "Large scale manifold transduction," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 448–455.

[37] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical software engineering*, vol. 14, no. 2, p. 131, 2009.

[38] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*, 2008, pp. 1–10.

[39] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.

[40] T. Bartz-Beielstein, C. Doerr, D. v. d. Berg, *et al.*, "Benchmarking in optimization: Best practice and open issues," *arXiv preprint arXiv:2007.03488*, 2020.

[41] T. Fredriksson, J. Bosch, H. H. Olsson, and D. I. Mattos, "Machine learning algorithms for labeling: Where and how they are used?" In *2022 IEEE International Systems Conference (SysCon)*, IEEE, 2022, pp. 1–8.

[42] T. Fredriksson, D. I. Mattos, J. Bosch, and H. H. Olsson, "An empirical evaluation of algorithms for data labeling," in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, IEEE, 2021, pp. 201–209.

[43] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative research in psychology*, vol. 3, no. 2, pp. 77–101, 2006.

[44] C. Tatsuno, "Kyouiku hyouka no igi, rekishi [significance and history of educational evaluation]," *Kyouiku hyouka jiten*, pp. 18–19, 2006.

[45] T. Yoshida, H. Ishii, and T. Haebara, "Syakudo no sakusei shiyou to datousei no kentou [construction, use, and validation of psychological scales]," *Kyouiku Shinrigaku Nenpou [Annu. Rep. Educ. Psychol. Jpn.]*, vol. 51, pp. 213–217, 2012.

[46] F. Lord, "A theory of test scores.," *Psychometric monographs*, 1952.

[47] A. L. Birnbaum, "Some latent trait models and their use in inferring an examinee's ability," *Statistical theories of mental test scores*, 1968.

[48] G. Rasch, *Probabilistic models for some intelligence and attainment tests.* ERIC, 1993.

[49] F. Martinez-Plumed, R. B. Prudencio, A. Martinez-Uso, and J. Hernandez-Orallo, "Item response theory in ai: Analysing machine learning classifiers at the instance level," *Artificial intelligence*, vol. 271, pp. 18–42, 2019.

[50] B. B. N. de França and G. Travassos, "Simulation based studies in software engineering: A matter of validity.," in *CIbSE*, 2014, pp. 308–321.

[51] D. S. Triangulation, "The use of triangulation in qualitative research," in *Oncol Nurs Forum*, vol. 41, 2014, pp. 545–7.

[52] D. I. Mattos, J. Bosch, and H. H. Olsson, "Statistical models for the analysis of optimization algorithms with benchmark functions," *arXiv preprint arXiv:2010.03783*, 2020.

[53] J. Calder, *Graph-based clustering and semi-supervised learning*, https://github.com/j 2023.

[54] AzatiSoftware, *AzatiSoftware automated data labeling with machine learning*, https://azati.ai/automated-data-labeling-with-machine-learning, 2019.

[55] R. Board and L. Pitt, "Semi-supervised learning," *Machine Learning*, vol. 4, no. 1, pp. 41–65, 1989.

[56] S. Sukhbaatar and R. Fergus, "Learning from noisy labels with deep neural networks," *arXiv preprint arXiv:1406.2080*, vol. 2, no. 3, p. 4, 2014.

[57] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[58] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.

[59] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and information systems*, vol. 35, no. 2, pp. 249–283, 2013.

[60] T. M. Mitchell, "The role of unlabeled data in supervised learning," in *Language, Knowledge, and Representation*, Springer, 2004, pp. 103–111.

[61] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, 1998, pp. 92–100.

[62] S. Dasgupta, M. L. Littman, and D. McAllester, "Pac generalization bounds for co-training," *Advances in neural information processing systems*, vol. 1, pp. 375–382, 2002.

[63] V. R. de Sa, "Learning classification with unlabeled data," in *Advances in neural information processing systems*, Citeseer, 1994, pp. 112–119.

[64] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[65] V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recognition Letters*, vol. 16, no. 1, pp. 105–111, 1995.

[66] J. Ratsaby and S. S. Venkatesh, "Learning from a mixture of labeled and unlabeled examples with parametric side information," in *Proceedings of the eighth annual conference on Computational learning theory*, 1995, pp. 412–417.

[67] F. G. Cozman, I. Cohen, M. C. Cirelo, *et al.*, "Semi-supervised learning of mixture models," in *ICML*, vol. 4, 2003, p. 24.

[68] V. Vapnik and V. Vapnik, *Statistical learning theory 156–160*, 1998.

[69] T. Jebara, J. Wang, and S.-F. Chang, "Graph construction and b-matching for semi-supervised learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 441–448.

[70] S. I. Daitch, J. A. Kelner, and D. A. Spielman, "Fitting a graph to vector data," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 201–208.

[71] P. S. Dhillon, P. P. Talukdar, and K. Crammer, "Inference-driven metric learning for graph construction," in *4th North East Student Colloquium on Artificial Intelligence*, 2010.

[72] X. Zhu, J. S. Kandola, J. Lafferty, and Z. Ghahramani, *Graph kernels by spectral transforms.* 2006.

[73] A. Blum, J. Lafferty, M. R. Rwebangira, and R. Reddy, "Semi-supervised learning using randomized mincuts," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 13.

[74] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the 20th International conference on Machine learning (ICML-03)*, 2003, pp. 912–919.

[75] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in neural information processing systems*, 2004, pp. 321–328.

[76] S. Baluja, R. Seth, D. Sivakumar, *et al.*, "Video suggestion and discovery for youtube: Taking random walks through the view graph," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 895–904.

[77] P. P. Talukdar and K. Crammer, "New regularized algorithms for transductive learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2009, pp. 442–457.

[78] Y. Bengio, O. Delalleau, and N. Le Roux, "Label propagation and quadratic criterion," in *Semi-Supervised Learning*, Semi-Supervised Learning. MIT Press, Jan. 2006, pp. 193–216.

[79] M. Orbach and K. Crammer, "Graph-based transduction with confidence," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2012, pp. 323–338.

[80] M. S. T. Jaakkola and M. Szummer, "Partially labeled classification with markov random walks," *Advances in neural information processing systems (NIPS)*, vol. 14, pp. 945–952, 2002.

[81] A. A. D. Corduneanu, "The information regularization framework for semi-supervised learning," Ph.D. dissertation, Massachusetts Institute of Technology, 2006.

[82] A. Corduneanu and T. S. Jaakkola, "On information regularization," *arXiv preprint arXiv:1212.2466*, 2012.

[83] M. Belkin, P. Niyogi, and V. Sindhwani, "On manifold regularization.," in *AISTATS*, vol. 1, 2005.

[84] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *SIGIR'94*, Springer, 1994, pp. 3–12.

[85] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[86] D. D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," in *Machine learning proceedings 1994*, Elsevier, 1994, pp. 148–156.

[87] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 1070–1079.

[88] R. Hwa, "Sample selection for statistical parsing," *Computational linguistics*, vol. 30, no. 3, pp. 253–276, 2004.

[89] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[90] A. Culotta and A. McCallum, "Reducing labeling effort for structured prediction tasks," in *AAAI*, vol. 5, 2005, pp. 746–751.

[91] A. Fujii, K. Inui, T. Tokunaga, and H. Tanaka, "Selective sampling for example-based word sense disambiguation," *arXiv preprint cs/9910020*, 1999.

[92] M. Lindenbaum, S. Markovitch, and D. Rusakov, "Selective sampling for nearest neighbor classifiers," *Machine learning*, vol. 54, no. 2, pp. 125–152, 2004.

[93] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[94] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proceedings of the ninth ACM international conference on Multimedia*, 2001, pp. 107–118.

[95] A. K. McCallumzy and K. Nigamy, "Employing em and pool-based active learning for text classification," in *Proc. International Conference on Machine Learning (ICML)*, Citeseer, 1998, pp. 359–367.

[96] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 79.

[97] Z. Xu, R. Akella, and Y. Zhang, "Incorporating diversity and density in active learning for relevance feedback," in *European Conference on Information Retrieval*, Springer, 2007, pp. 246–257.

[98] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 287–294.

[99] N. Abe, "Query learning strategies using boosting and bagging," *Proc. of 15ˆ< th> Int. Cmf. on Machine Learning (ICML98)*, pp. 1–9, 1998.

[100] S. Keele *et al.*, "Guidelines for performing systematic literature reviews in software engineering," Technical report, Ver. 2.3 EBSE Technical Report. EBSE, Tech. Rep., 2007.

[101] C. Zhang and W.-S. Zheng, "Semi-supervised multi-view discrete hashing for fast image search," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2604–2617, 2017.

[102] D. Guan, W. Yuan, Y.-K. Lee, A. Gavrilov, and S. Lee, "Activity recognition based on semi-supervised learning," in *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, IEEE, 2007, pp. 469–475.

[103] F. Wu, X.-Y. Jing, J. Zhou, *et al.*, "Semi-supervised multi-view individual and sharable feature learning for webpage classification," in *The World Wide Web Conference*, 2019, pp. 3349–3355.

[104] Z. Yu, L. Su, L. Li, Q. Zhao, C. Mao, and J. Guo, "Question classification based on co-training style semi-supervised learning," *Pattern Recognition Letters*, vol. 31, no. 13, pp. 1975–1980, 2010.

[105] X. Cui, J. Huang, and J.-T. Chien, "Multi-view and multi-objective semi-supervised learning for hmm-based automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 1923–1935, 2012.

[106] Y. Xia, F. Liu, D. Yang, *et al.*, "3d semi-supervised learning with uncertainty-aware multi-view co-training," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 3646–3655.

[107] K. Lim, J. Y. Lee, J. Carbonell, and T. Poibeau, "Semi-supervised learning on meta structure: Multi-task tagging and parsing in low-resource scenarios," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 8344–8351.

[108] C. Chen, Y. Li, H. Qian, Z. Zheng, and Y. Hu, "Multi-view semi-supervised learning for classification on dynamic networks," *Knowledge-Based Systems*, vol. 195, p. 105 698, 2020.

[109] R. Yan and M. Naphade, "Semi-supervised cross feature learning for semantic concept detection in videos," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, vol. 1, 2005, pp. 657–663.

[110] R. G. Colares, P. Machado, M. de Faria, A. Detoni, V. Tavano, *et al.*, "Microalgae classification using semi-supervised and active learning based on gaussian mixture models," *Journal of the Brazilian Computer Society*, vol. 19, no. 4, pp. 411–422, 2013.

[111] L. Bull, K. Worden, and N. Dervilis, "Towards semi-supervised and probabilistic classification in structural health monitoring," *Mechanical Systems and Signal Processing*, vol. 140, p. 106 653, 2020.

[112] L. Shi, R. Mihalcea, and M. Tian, "Cross language text classification by model translation and semi-supervised learning," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2010, pp. 1057–1067.

[113] J.-T. Huang and M. Hasegawa-Johnson, "On semi-supervised learning of gaussian mixture models for phonetic classification," in *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, Association for Computational Linguistics, 2009, pp. 75–83.

[114] N. Kumar and S. P. Awate, "Semi-supervised robust mixture models in rkhs for abnormality detection in medical images," *IEEE Transactions on Image Processing*, vol. 29, pp. 4772–4787, 2020.

[115] R. Lang, R. Lu, C. Zhao, H. Qin, and G. Liu, "Graph-based semi-supervised one class support vector machine for detecting abnormal lung sounds," *Applied Mathematics and Computation*, vol. 364, p. 124 487, 2020.

[116] S. Wang, X. Guo, Y. Tie, I. Lee, L. Qi, and L. Guan, "Graph-based safe support vector machine for multiple classes," *IEEE Access*, vol. 6, pp. 28 097–28 107, 2018.

[117] T. Zhang and Z.-H. Zhou, "Semi-supervised optimal margin distribution machines.," in *IJCAI*, 2018, pp. 3104–3110.

[118] K. Zhang, C. Li, Y. Wang, X. Zhu, and H. Wang, "Collaborative support vector machine for malware detection," *Procedia Computer Science*, vol. 108, pp. 1682–1691, 2017.

[119] Z. Yang and Y. Xu, "A safe screening rule for laplacian support vector machine," *Engineering Applications of Artificial Intelligence*, vol. 67, pp. 309–316, 2018.

[120] M. P. Kumar and M. K. Rajagopal, "Detecting facial emotions using normalized minimal feature vectors and semi-supervised twin support vector machines classifier," *Applied Intelligence*, vol. 49, no. 12, pp. 4150–4174, 2019.

[121] R. Rastogi and S. Sharma, "Fast laplacian twin support vector machine with active learning for pattern classification," *Applied Soft Computing*, vol. 74, pp. 424–439, 2019.

[122] H. Pei, Q. Lin, L. Yang, and P. Zhong, "A novel semi-supervised support vector machine with asymmetric squared loss," *Advances in Data Analysis and Classification*, pp. 1–33, 2020.

[123] M. Zhao, T. W. Chow, Z. Zhang, and B. Li, "Automatic image annotation via compact graph based semi-supervised learning," *Knowledge-Based Systems*, vol. 76, pp. 148–165, 2015.

[124] J. Tang, H. Li, G.-J. Qi, and T.-S. Chua, "Image annotation by graph-based inference with integrated multiple/single instance representations," *IEEE Transactions on Multimedia*, vol. 12, no. 2, pp. 131–141, 2009.

[125] J. Tang, R. Hong, S. Yan, T.-S. Chua, G.-J. Qi, and R. Jain, "Image annotation by knn-sparse graph-based label propagation over noisily tagged web images," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 2, pp. 1–15, 2011.

[126] Y. Liu and K. Kirchhoff, "Graph-based semi-supervised learning for phone and segment classification.," in *INTERSPEECH*, 2013, pp. 1840–1843.

[127] X. Zeng, D. F. Wong, L. S. Chao, and I. Trancoso, "Graph-based semi-supervised model for joint chinese word segmentation and part-of-speech tagging," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, pp. 770–779.

[128] C. Chen, Y. Gong, and Y. Tian, "Semi-supervised learning methods for network intrusion detection," in *2008 IEEE international conference on systems, man and cybernetics*, IEEE, 2008, pp. 2603–2608.

[129] J. Calder, B. Cook, M. Thorpe, and D. Slepcev, "Poisson learning: Graph based semi-supervised learning at very low label rates," in *International Conference on Machine Learning*, PMLR, 2020, pp. 1306–1316.

[130] W. Lin, Z. Gao, and B. Li, "Shoestring: Graph-based semi-supervised classification with severely limited labeled data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4174–4182.

[131] B. Yoon, Y. Jeong, and S. Kim, "Detecting a risk signal in stock investment through opinion mining and graph-based semi-supervised learning," *IEEE Access*, vol. 8, pp. 161 943–161 957, 2020.

[132] W. D. G. de Oliveira, O. A. Penatti, and L. Berton, "A comparison of graph-based semi-supervised learning for data augmentation," in *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, IEEE, 2020, pp. 264–271.

[133] S. M. K. Zaman, X. Liang, and H. Zhang, "Graph-based semi-supervised learning for induction motors single-and multi-fault diagnosis using stator current signal," in *2020 IEEE/IAS 56th Industrial and Commercial Power Systems Technical Conference (I&CPS)*, IEEE, 2020, pp. 1–10.

[134] J. Tang, H. Li, G.-J. Qi, and T.-S. Chua, "Integrated graph-based semi-supervised multiple/single instance learning framework for image annotation," in *Proceedings of the 16th ACM international conference on Multimedia*, 2008, pp. 631–634.

[135] J. Zhu, H. Wang, T. Yao, and B. K. Tsou, "Active learning with sampling by uncertainty and density for word sense disambiguation and text classification," in *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, 2008, pp. 1137–1144.

[136] R. Liere and P. Tadepalli, "Active learning with committees for text categorization," in *AAAI/IAAI*, 1997, pp. 591–596.

[137] E. Ringger, P. McClanahan, R. Haertel, *et al.*, "Active learning for part-of-speech tagging: Accelerating corpus annotation," in *Proceedings of the Linguistic Annotation Workshop*, 2007, pp. 101–108.

[138] F. K. Nakano, R. Cerri, and C. Vens, "Active learning for hierarchical multi-label classification," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1496–1530, 2020.

[139] Y. Sheng, Y. Wu, J. Yang, W. Lu, P. Villars, and W. Zhang, "Active learning for the power factor prediction in diamond-like thermoelectric materials," *npj Computational Materials*, vol. 6, no. 1, pp. 1–7, 2020.

[140] S. Li, Y. Xue, Z. Wang, and G. Zhou, "Active learning for cross-domain sentiment classification," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[141] S. Bommaraveni, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Active content popularity learning and caching optimization with hit ratio guarantees," *IEEE Access*, vol. 8, pp. 151 350–151 359, 2020.

[142] R. C. Prati, G. E. Batista, and M. C. Monard, "Data mining with imbalanced class distributions: Concepts and methods.," in *IICAI*, 2009, pp. 359–376.

[143] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[144] R. C. Gonzalez and R. E. Woods, *Digital image processing, hoboken*, 2018.

[145] J. Zhang, X. Wu, and V. S. Sheng, "Learning from crowdsourced labeled data: A survey," *Artificial Intelligence Review*, vol. 46, no. 4, pp. 543–576, 2016.

[146] hackernoon.com, *Crowdsourcing data labeling for machine learning projects*, https://hackernoon.com/crowdsourcing-data-labeling-for-machine-learning-projects-a-how-to-guide-cp6h32nd, 2020.

[147] P. G. Ipeirotis, F. Provost, V. S. Sheng, and J. Wang, "Repeated labeling using multiple noisy labelers," *Data Mining and Knowledge Discovery*, vol. 28, no. 2, pp. 402–441, 2014.

[148] A. Sheshadri and M. Lease, "Square: A benchmark for research on computing crowd consensus," in *First AAAI conference on human computation and crowdsourcing*, 2013.

[149] C. G. Northcutt, L. Jiang, and I. L. Chuang, "Confident learning: Estimating uncertainty in dataset labels," *arXiv preprint arXiv:1911.00068*, 2019.

[150] P. Reason and H. Bradbury, *Handbook of action research: Participative inquiry and practice*. Sage, 2001.

[151] M. Staron, *Action Research in Software Engineering: Theory and Applications*. Springer Nature, 2019.

[152] T. DataScience, *What to do when your classification data is imbalanced?* https://towardsdatascience.com/what-to-do-when-your- classification-dataset-is-imbalanced-6af031b12a36, 2019.

[153] T. Fredriksson., J. Bosch., and H. H. Olsson., "Machine learning models for automatic labeling: A systematic literature review," in *Proceedings of the 15th International Conference on Software Technologies - Volume 1: ICSOFT,*, INSTICC, SciTePress, 2020, pp. 552–561, ISBN: 978-989-758-443-5.

[154] E. Bair, "Semi-supervised clustering methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 5, pp. 349–361, 2013.

[155] C. Körner and S. Wrobel, "Multi-class ensemble-based active learning," in *European conference on machine learning*, Springer, 2006, pp. 687–694.

[156] A. I. Schein and L. H. Ungar, "Active learning for logistic regression: An evaluation," *Machine Learning*, vol. 68, no. 3, pp. 235–265, 2007.

[157] A. Harpale, "Multi-task active learning," Ph.D. dissertation, Carnegie Mellon University, 2012.

[158] J. Baldridge and M. Osborne, "Active learning and the total cost of annotation," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 9–16.

[159] A. Kapoor, E. Horvitz, and S. Basu, "Selective supervision: Guiding supervised learning with decision-theoretic active learning.," in *IJCAI*, vol. 7, 2007, pp. 877–882.

[160] B. Settles, M. Craven, and L. Friedland, "Active learning with real annotation costs," in *Proceedings of the NIPS workshop on cost-sensitive learning*, Vancouver, CA: 2008, pp. 1–10.

[161] S. Arora, E. Nyberg, and C. Rose, "Estimating annotation cost for active learning in a multi-annotator environment," in *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, 2009, pp. 18–26.

[162] E. K. Ringger, M. Carmen, R. Haertel, *et al.*, "Assessing the costs of machine-assisted corpus annotation through a user study.," in *LREC*, vol. 8, 2008, pp. 3318–3324.

[163] S. Vijayanarasimhan and K. Grauman, "What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 2262–2269.

[164] B. C. Wallace, K. Small, C. E. Brodley, J. Lau, and T. A. Trikalinos, "Modeling annotation time to reduce workload in comparative effectiveness reviews," in *Proceedings of the 1st ACM International Health Informatics Symposium*, 2010, pp. 28–35.

[165] R. A. Haertel, K. D. Seppi, E. K. Ringger, and J. L. Carroll, "Return on investment for active learning," in *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, vol. 72, 2008.

[166] A. Kumar, M. Boehm, and J. Yang, "Data management in machine learning: Challenges, techniques, and systems," in *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017, pp. 1717–1722.

[167] T. Fredriksson, D. I. Mattos, J. Bosch, and H. H. Olsson, "Data labeling: An empirical investigation into industrial challenges and mitigation strategies," in *Product-Focused Software Process Improvement: 21st International Conference, PROFES 2020, Turin, Italy, November 25–27, 2020, Proceedings 21*, Springer, 2020, pp. 202–216.

[168] A. Gupta, A. Anpalagan, L. Guan, and A. S. Khwaja, "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," *Array*, vol. 10, p. 100 057, 2021.

[169] M. S. Ramanagopal, C. Anderson, R. Vasudevan, and M. Johnson-Roberson, "Failing to learn: Autonomously identifying perception failures for self-driving cars," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3860–3867, 2018.

[170] A. Subramanya and P. P. Talukdar, "Graph-based semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 4, pp. 1–125, 2014.

[171] D. I. Mattos, J. Bosch, and H. H. Olsson, "Statistical models for the analysis of optimization algorithms with benchmark functions," *IEEE Transactions on Evolutionary Computation*, 2021.

[172] C. P. Robert *et al.*, *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer, 2007, vol. 2.

[173] D. Issa Mattos and É. Martins Silva Ramos, "Bayesian paired comparison with the bpcs package," *Behavior Research Methods*, pp. 1–21, 2022.

[174] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[175] M. Cattelan, "Models for paired comparison data: A review with emphasis on dependent data," *Statistical Science*, pp. 412–433, 2012.

[176] A. Agresti, *Categorical data analysis*. John Wiley & Sons, 2003, vol. 482.

[177] P. Kleinschmidt and R. Mock, "Sensor systems in industrial applications," in *COMPEURO 89 Proceedings VLSI and Computer Peripherals.*, IEEE Computer Society, 1989, pp. 3–9.

[178] `https://www.kaggle.com/jessicali9530/caltech256`.

[179] `http://www.vision.caltech.edu/Image_Datasets/Caltech256/`.

[180] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[181] *Corel database for content-based image retrieval*, `https://sites.google.com/site/dctresearch/Home/content-based-image-retrieval`.

[182] `https://www.kaggle.com/elkamel/corel-images`.

[183] A. K. Seewald, "Digits-a dataset for handwritten digit recognition," *Austrian Research Institut for Artificial Intelligence Technical Report, Vienna (Austria)*, 2005.

[184] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[185] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, pp. 3630–3638, 2016.

[186] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots, "One-shot learning for semantic segmentation," *arXiv preprint arXiv:1709.03410*, 2017.

[187] Newsgroup dataset, *20 newsgroups dataset*, 2012.

[188] M. Filannino, "Dbworld e-mail classification using a very small corpus," *The University of Manchester*, 2011.

[189] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, Springer, 2017, pp. 127–138.

[190] ——, "Detecting opinion spams and fake news using text classification," *Security and Privacy*, vol. 1, no. 1, e9, 2018.

[191] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 142–150.

[192] W. Hersh, C. Buckley, T. Leone, and D. Hickam, "Ohsumed: An interactive retrieval evaluation and new large test collection for research," in *SIGIR'94*, Springer, 1994, pp. 192–201.

[193] O. Ekin, P. L. Hammer, A. Kogan, and P. Winter, "Distance-based classification methods," *INFOR: Information Systems and Operational Research*, vol. 37, no. 3, pp. 337–352, 1999.

[194] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker, "Classification of radar returns from the ionosphere using neural networks," *Johns Hopkins APL Technical Digest*, vol. 10, no. 3, pp. 262–266, 1989.

[195] C. C. Aggarwal and S. Sathe, "Theoretical foundations and algorithms for outlier ensembles," *Acm Sigkdd Explorations Newsletter*, vol. 17, no. 1, pp. 24–47, 2015.

[196] J. W. Smith, J. E. Everhart, W. Dickson, W. C. Knowler, and R. S. Johannes, "Using the adap learning algorithm to forecast the onset of diabetes mellitus," in *Proceedings of the annual symposium on computer application in medical care*, American Medical Informatics Association, 1988, p. 261.

[197] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural networks*, vol. 1, no. 1, pp. 75–89, 1988.

[198] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, "Nuclear feature extraction for breast tumor diagnosis," in *Biomedical image processing and biomedical visualization*, International Society for Optics and Photonics, vol. 1905, 1993, pp. 861–870.

[199] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," *Operations Research*, vol. 43, no. 4, pp. 570–577, 1995.

[200] M. Forina, S. Lanteri, C. Armanino, *et al.*, "Parvus-an extendible package for data exploration, classification and correlation, institute of pharmaceutical and food analysis and technologies, via brigata salerno, 16147 genoa, italy (1988)," *Av. Loss Av. O set Av. Hit-Rate*, 1991.

[201] T. Fredriksson, D. I. Mattos, J. Bosch, and H. H. Olsson, "Assessing the suitability of semi-supervised learning datasets using item response theory," in *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, IEEE, 2021, pp. 326–333.

[202] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," 2001.

[203] W. J. van der Linden, *Handbook of Item Response Theory: Volume 1: Models*. CRC Press, 2016.

[204] K. Hori, H. Fukuhara, and T. Yamada, "Item response theory and its applications in educational measurement part i: Item response theory and its implementation in r," *Wiley Interdisciplinary Reviews: Computational Statistics*, e1531, 2020.

[205] W. J. van der Linden, "Unidimensional logistic response models," in *Handbook of Item Response Theory, Volume One: Models*, Chapman and Hall/CRC, 2016, pp. 11–30.

[206] K. G. Jöreskog, "Statistical analysis of sets of congeneric tests," *Psychometrika*, vol. 36, no. 2, pp. 109–133, 1971.

[207] G. J. Mellenbergh, "Models for continuous responses," in *Handbook of Item Response Theory, Volume One: Models*, Chapman and Hall/CRC, 2016, pp. 153–163.

[208] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: A tutorial for comparing multiple classifiers through bayesian analysis," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2653–2688, 2017.

[209] J. K. Kruschke and T. M. Liddell, "Bayesian data analysis for newcomers," *Psychonomic bulletin & review*, vol. 25, no. 1, pp. 155–177, 2018.

[210] C. A. Furia, R. Feldt, and R. Torkar, "Bayesian data analysis in empirical software engineering research," *IEEE Transactions on Software Engineering*, 2019.

[211] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian data analysis*. CRC press, 2013.

[212] B. Carpenter, A. Gelman, M. D. Hoffman, *et al.*, "Stan: A probabilistic programming language," *Journal of statistical software*, vol. 76, no. 1, 2017.

[213] M. D. Hoffman and A. Gelman, "The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo.," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1593–1623, 2014.

[214] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[215] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[216] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*, Springer, 2016, pp. 630–645.

[217] F. Alimoglu, E. Alpaydin, and Y. Denizhan, "Combining multiple classifiers for pen-based handwritten digit recognition," 1996.

[218] F. Alimoglu and E. Alpaydin, "Methods of combining multiple classifiers based on different representations for pen-based handwritten digit recognition," in *Proceedings of the Fifth Turkish Artificial Intelligence and Artificial Neural Networks Symposium (TAINN 96*, Citeseer, 1996.

[219] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition using state-of-the-art techniques," in *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*, IEEE, 2002, pp. 320–325.

[220] K. Huang, I. King, and M. R. Lyu, "Constructing a large node chow-liu tree based on frequent itemsets," in *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, IEEE, vol. 1, 2002, pp. 498–502.

[221] T. Liu, J. Bao, J. Wang, and Y. Zhang, "A hybrid cnn–lstm algorithm for online defect recognition of co2 welding," *Sensors*, vol. 18, no. 12, p. 4369, 2018.

[222] C. Xiang, L. Zhang, Y. Tang, W. Zou, and C. Xu, "Ms-capsnet: A novel multi-scale capsule network," *IEEE Signal Processing Letters*, vol. 25, no. 12, pp. 1850–1854, 2018.

[223] C. Wang, Y. Song, A. El-Kishky, D. Roth, M. Zhang, and J. Han, "Incorporating world knowledge to document clustering via heterogeneous information networks," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1215–1224.

[224] F. Gama, G. Leus, A. G. Marques, and A. Ribeiro, "Convolutional neural networks via node-varying graph filters," in *2018 IEEE Data Science Workshop (DSW)*, IEEE, 2018, pp. 1–5.

[225] I. Gallo, S. Nawaz, and A. Calefati, "Semantic text encoding for text classification using convolutional neural networks," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, vol. 5, 2017, pp. 16–21.

[226] P. Bafna, D. Pramod, and A. Vaidya, "Precision based recommender system using ontology," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, IEEE, 2017, pp. 3153–3160.

[227] E. Frank, C. Chui, and I. H. Witten, "Text categorization using compression models," 2000.

[228] K. H. Lee, J. Kay, and B. H. Kang, "Active learning: Applying rinscut thresholding strategy to uncertainty sampling," in *Australasian Joint Conference on Artificial Intelligence*, Springer, 2003, pp. 922–932.

[229] Z. Xu, X. Xu, K. Yu, and V. Tresp, "A hybrid relevance-feedback approach to text retrieval," in *European Conference on Information Retrieval*, Springer, 2003, pp. 281–293.

[230] K. Toutanova, F. Chen, K. Popat, and T. Hofmann, "Text classification in a hierarchical mixture model for small training sets," in *Proceedings of the tenth international conference on Information and knowledge management*, 2001, pp. 105–113.

[231] Ö. Yılmaz, L. E. Achenie, and R. Srivastava, "Systematic tuning of parameters in support vector clustering," *Mathematical biosciences*, vol. 205, no. 2, pp. 252–270, 2007.

[232] L. Singh, S. Singh, and P. K. Dubey, "Applications of clustering algorithms and self organizing maps as data mining and business intelligence tools on real world data sets," in *2010 International Conference on Methods and Models in Computer Science (ICM2CS-2010)*, IEEE, 2010, pp. 27–33.

[233] H.-L. Li and Y.-H. Huang, "A diamond method of inducing classification rules for biological data," *Computers in biology and medicine*, vol. 41, no. 8, pp. 587–599, 2011.

[234] S. Dilmaç and M. Korürek, "Using abc algorithm for classification and analysis on effects of control parameters," in *2014 18th National Biomedical Engineering Meeting*, IEEE, 2014, pp. 1–4.

[235] J.-F. Liu, D.-R. Yu, Q.-H. Hu, and X.-D. Li, "Granular entropy based hybrid knowledge reduction using uniform rough approximations," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*, IEEE, vol. 3, 2004, pp. 1878–1883.

[236] X. Xu and E. Frank, "Logistic regression and boosting for labeled bags of instances," in *Pacific-Asia conference on knowledge discovery and data mining*, Springer, 2004, pp. 272–281.

[237] D. M. Tax and R. P. Duin, "Learning curves for the analysis of multiple instance classifiers," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, Springer, 2008, pp. 724–733.

[238] X. Yuan, X.-S. Hua, M. Wang, G.-J. Qi, and X.-Q. Wu, "A novel multiple instance learning approach for image retrieval based on adaboost feature selection," in *2007 IEEE International Conference on Multimedia and Expo*, IEEE, 2007, pp. 1491–1494.

[239] J. Li, X. Li, and X. Yao, "Cost-sensitive classification with genetic programming," in *2005 IEEE congress on evolutionary computation*, IEEE, vol. 3, 2005, pp. 2114–2121.

[240] D. Pedreschi, S. Ruggieri, and F. Turini, "Measuring discrimination in socially-sensitive decision records," in *Proceedings of the 2009 SIAM international conference on data mining*, SIAM, 2009, pp. 581–592.

[241] L.-l. Zhang, X.-f. Hui, and L. Wang, "Application of adaptive support vector machines method in credit scoring," in *2009 International Conference on Management Science and Engineering*, IEEE, 2009, pp. 1410–1415.

[242] C. Garcia-Cardona, E. Merkurjev, A. L. Bertozzi, A. Flenner, and A. G. Percus, "Multiclass data segmentation using diffuse interface methods on graphs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 8, pp. 1600–1613, 2014.

[243] Z. Shi, S. Osher, and W. Zhu, "Weighted nonlocal laplacian on interpolation from sparse data," *Journal of Scientific Computing*, vol. 73, no. 2, pp. 1164–1177, 2017.

[244] M. Jacobs, E. Merkurjev, and S. Esedoḡlu, "Auction dynamics: A volume constrained mbo scheme," *Journal of Computational Physics*, vol. 354, pp. 288–310, 2018.

[245] A. Jung, A. O. Hero III, A. Mara, and S. Jahromi, "Semi-supervised learning via sparse label propagation," *arXiv preprint arXiv:1612.01414*, 2016.

[246] X. Mai and R. Couillet, "Random matrix-inspired improved semi-supervised learning on graphs," in *International Conference on Machine Learning*, 2018.

[247] B. N. De França and G. H. Travassos, "Reporting guidelines for simulation-based studies in software engineering," 2012.

[248] L. F. Cardoso, V. C. Santos, R. S. K. Francês, R. B. Prudêncio, and R. C. Alves, "Decoding machine learning benchmarks," in *Brazilian Conference on Intelligent Systems*, Springer, 2020, pp. 412–425.