VRIJE UNIVERSITEIT AMSTERDAM

MASTERS THESIS

Robust structure learning among weak labels

Author:
N.H.P. Torremans

Supervisor:
Dr. V. François-Lavet

Second reader:
Prof. Dr. S. Bhulai

External Supervisors:
S. Brugman, M. Sc
Dr. F. Jansen
Dr. M. Baak
Dr. I. Fridman Rojas

Faculty of Science
Master Business Analytics

August 29, 2021

Robust structure learning among weak labels

N.H.P. Torremans
Student number: 2542409

Graduation Thesis
Internship report

Vrije Universiteit Amsterdam
Faculty of Science
Business Analytics
De Boelelaan 1105
1081 HV Amsterdam

ING
Bijlmerplein 880
1102 MG Amsterdam

August 2021

VRIJE UNIVERSITEIT AMSTERDAM

Abstract

Faculty of Science
Master Business Analytics

Labeling training data is increasingly the largest bottleneck in deploying machine learning systems since labeling a lot of training data by hand is very expensive. The goal of this research is to show to what extent Robust PCA can be combined with other methods to estimate the dependency structure among binary weak labels when the true label is unknown. This dependency structure can then be used in a program to automatically label training data. In Varma et al.[25], Robust PCA is applied on the observable part of the inverse of the covariance matrix of weak labels. This method decomposes it into a sparse component and a low-rank component. The block structure of the sparse component should then represent the dependency structure among the weak labels since the dependency structure is expected to be sparse. In our approach we try to improve on this method by estimating the non-observed part of the covariance matrix. We are interested under which conditions our approach is able to retrieve the dependency structures of weak labels. This leads to examining the following research question:

To what extent can Robust PCA be applied and combined with other methods to find the dependency structure among weak labels?

In our approach we create a subset of four weak labels that are the most likely to be conditional independent and fit a Bayesian Network on it. Then, we use the parameters of the Bayesian Network and assumptions to fill the non-observed part of the covariance matrix. Subsequently, Robust PCA is applied on the inverse of the covariance matrix to retrieve the sparse component. As a final step, we combine our sparse component with the estimation of the approach of Varma et al[25] to make a final estimation of the dependency structure.

Based on the performed experiments we can conclude that a combination of Robust PCA and the usage of a Bayesian Network for estimating the unobserved part of the covariance matrix can be used to estimate the dependency structure among weak labels. When the number of cliques in a synthetic dataset increases or when the datset becomes imbalanced the approach can still be used to estimate the dependency structure correctly in most cases. However, when the size of the cliques becomes large, the performance of our approach decreases. The results have shown that a subset of a variable size, that depends on the detected dependency between weak labels in the inverse covariance matrix, might improve the performance of our approach.

Acknowledgements

This project serves as the graduation project, which is a compulsory part of the Masters degree program Business Analytics at VU University in Amsterdam. The graduation thesis has been produced as part of the final product of the mandatory internship. The internship and research have been executed at ING WBAA in Amsterdam, although most of the time we have been working remotely.

I am grateful that ING WBAA gave me the opportunity to conduct this research and to be part of a professional and advanced organization during my internship. I enjoyed working with experienced data scientists who are critical and always aiming for the best results. I would like to express my gratitude to MSc. Simon Brugman from ING WBAA who has been closely involved in this project. Additionally, I would like to thank Dr. Fabian Jansen, Dr. Max Baak and Dr. Ilan Fridman Rojas from ING WBAA for their cooperation and the contribution they have made during the last months. Furthermore, I would like to thank my university supervisor Dr. Vincent François-Lavet for the support and guidance throughout this research. Finally, a special thanks goes to my friends and family for their loving support during my studies.

Contents

Introduction

ING bank is a well-known (originally) Dutch bank that is part of the ING group. It is an autonomous company that works independently of the insurance- and investment part of ING. Wholesale Banking is one of the two parts of ING bank and delivers banking services as lending, payments and cash managements to companies, governments and financial institutions. Retail Banking is the other part. The Wholesale Banking Advanced Analytics team (WBAA) of ING is a mix of around 120 data scientists, data engineers, software developers, business developers, UX experts and other specializations needed to build products based on data science and machine learning. The department works on data-driven algorithmic products that aim to create substantial value for the Wholesale Banking clients. Next to this they share knowledge, actively contribute to open-source software and communities and maintain strong connections to academia.

During the last decade, the usage of classifiers gained popularity due to their predictive powers. Classifiers are used widely. For example: banks use classifiers to classify transactions or loans and social media platforms use them to determine whether a comment is appropriate or not. These classifiers require labeled training data to be trained with. Now that the popularity of using classifiers is increasing, the demand of labelled data that can be used to train them is increasing as well. However, labeling training data can be complicated when there is a huge amount of data. In 2020 the Dutch banks processed 6.2 billion transactions and on social media like Facebook people write billions of comments each day [19], [18]. It would take a lot of time to label those transactions or comments by hand. Labeling these datasets automatically would be more productive. In 2017, researchers from Stanford University developed the so-called Snorkel program [23]. The Snorkel programming framework can be used to automatically label unlabeled training data. One of the things that the framework needs as input is the dependency structure among the observable variables of the data. The question is: how do we find that dependency structure when the true label is unknown? In 2017, Wu et al [28] showed that Robust PCA (RPCA) and the observable inverse covariance matrix can be used to find the underlying dependency structure among weak labels. A weak label can be used to express various weak supervision sources such as patterns in the form of a (binary) variable. The recovered structure can be used by the Snorkel program to form a strong label which can represent the missing true label. In this research we try to improve on the method of Varma et al.[25] to make an estimation of the dependency structure.

1.1 Research question

In the research of this thesis it has been tried to improve on the structure learning methods of Varma et al. [25]. The goal is to find out to what extent we can combine Varma's approach with other methods to make a good estimation of the dependency

structure among weak labels. So, the research question that is answered in this research is:

To what extent can Robust PCA be applied and combined with other methods to find the dependency structure among weak labels?

The main take-away for WBAA will be a tested, reproducible and documented account of how and under which circumstances Robust PCA can help to retrieve the dependency structure among weak labels. This dependency structure could be used to form a representative label for improving training data quality that can in the end improve the performance of classifiers.

First, the relevant literature will be addressed in Chapter 2. Then, in Chapter 3 the conceptual background will be discussed. After that, the approach will be presented in Chapter 4, followed by the experiments in Chapter 5. Then, the results of the experiments are discussed in Chapter 6. Finally, the thesis will end with the conclusion and discussion.

Relevant literature

This chapter briefly discusses some relevant literature that is available in the domain of applying Robust PCA and finding dependency structures. As explained in the introduction, the goal of this work is to investigate whether Robust PCA can be combined with other methods to make an estimation of the dependency structure among weak labels. In this section the three works that function as a guideline in our research will be discussed.

We will start with the Snorkel program [23]. In this section it will be explained how the Snorkel program can be used to automatically label training data. After that, the Loh & wainwright paper [17] about estimating the depwendency structure from a generalised inverse covariance matrix will be described. Finally, the paper of Varma et al [25] and their application of Robust PCA will be discussed. By combining the information of these three papers, it is possible to make an estimation of the dependency structure among weak labels and form a label that can represent the true label. In our work we try to improve on these methods.

2.1 Snorkel

The process of labeling training data has been done by hand for a long time. But, now that the amount of data is increasing, the demand of labelled data that can be used for machine learning projects is increasing as well. Labelling millions of data samples by hand is very time consuming. Rich companies can hire large teams to do that, but small ones cannot. They are increasingly turning to weak supervision: cheaper sources of labels that are noisier or heuristic [23]. These sources often have limited accuracy and coverage. For that reason, Ratner et al. [23] from Stanford University created Snorkel, a system where the users use labeling functions that express the weak supervision sources. The Snorkel system consists of three stages:

1. Writing labeling functions;
2. Applying the labeling function over unlabeled data and learning a generative model to combine these functions into probabilistic labels;
3. Use these labels to train a classification- or machine learning model.

The Snorkel users model the true class label for a data point as a latent variable in a probabilistic model. In the simplest way, each weak label can be modeled as a "voter" that is uncorrelated with the other weak labels and use a majority vote. Then, to improve the predictive performance of the generative model, you can model the (conditional) dependencies between the labels. To do that, the dependency structure among the labels must be known or estimated. The next section explains how this dependency structure can be found.

2.2 Dependency structure and precision matrix

Loh & Wainwright [17] have investigated the relationship between the dependency structure of a discrete graphical model and the inverse of a generalized covariance matrix. Their main result has a corollary for tree-structured graphs: for such models, the inverse of a generalized covariance matrix is always (block) graph-structured when the variables are binary. In that case, the inverse of the covariance matrix may be used to recover the edge structure of the tree. The structure can be found by looking at the values of the inverse of the covariance matrix. When the corresponding term in the matrix is zero, there is no edge between the variables x_2 and x_3 in the tree-structured graph. In other words: x_2 and x_3 are conditionally independent on all of the other terms. An example of this is shown graphically in Figure 2.1.

FIGURE 2.1: Example of the inverse covariance matrix of a dataset that contains two conditional independent variables.

In Figure 2.1 it can be seen that the entry of the inverse covariance matrix that corresponds with x_2 and x_3 is zero. Therefore, it can be assumed that x_2 and x_3 are conditional independent. So, the inverse of the covariance matrix of the true label and all the dependent variables can be used to find the conditional dependencies between the variables and the true label. However, in our case the true label is unknown. This means that we can only invert the observable part of the covariance matrix. When the inverse of the observable covariance matrix is calculated (while the true label is unknown) the resulting matrix contains statistical noise since the information of the true label is missing. In their paper, Varma et al.[25] show that the inverse of the observable part of the covariance matrix Σ_O^{-1} can be written as:

$$\Sigma_O^{-1} = K_O - zz^T \tag{2.1}$$

In 2.1 K_O is the inverse covariance matrix that represents the dependency structure and is sparse. The other component, zz^T is the noise and is low-rank. Therefore, to filter the noise from the inverse covariance matrix, we should decompose the inverse covariance matrix in a sparse component and a low-rank component. This can be done by using Robust PCA which is explained below.

2.3 Matrix noise removal

As explained in the previous section, when the inverse of the observable covariance matrix is retrieved it contains statistical noise because the true label is missing. Because of this noise the dependency structure might not be fully detectable from the matrix. An example of this can be seen in Figure 2.2.

(A) Including the true label. (B) Without the true label.

FIGURE 2.2: Example of the inverse covariance matrix when the true label is included and when it is not.

In Figure 2.2 the same dataset is used for both sub-figures. It can be seen that when the true label is included and the covariance matrix is inverted, the dependency structure is clearly visible: the conditional independent entries are approximately zero and the conditional dependent entries are non-zero. However, in Figure 2.2b it can be seen that when only the observable part of the covariance matrix is inverted, so without the true label, the structure is still visible, but it is a lot noisier. Varma et al. [25] used a Robust PCA algorithm to filter out this noise by dividing the observed inverse covariance matrix M into a low-rank matrix L and a sparse component S. Therefore, a full matrix M can be written as $M = L + S$. After the application of Robust PCA the dependency structure should be more visible from the sparse component. The nonzero entries of the sparse component can be passed to the Snorkel program as dependencies to generate labels for the train set. This technique of applying Robust PCA for these cases was introduced by Wu et al[28]. However, the work of Varma et al. is the closest related to our case.

Conceptual background

This chapter introduces the central concepts used throughout this work and builds them up towards the approach. The methods itself will be partially described in the next chapter.

First, we introduce the concepts related to dependency this work is concerned with. After that, we discuss the covariance and inverse covariance matrix, which structural properties are of essence for our approach.

3.1 Dependency

As stated in the introduction of this thesis, the goal of this research was to show under which circumstances it is possible to find the dependency structure among multiple weak labels. Dependency means that the value of a variable will change depending on the value of another variable [5]. In statistics, two random variables A and B are said to be independent if and only if equation 3.1 holds [5].

$$P(A \cap B) = P(A)P(B) \tag{3.1}$$

When 3.1 does not hold, the random variables A and B are not independent, so they are dependent variables. Besides this direct dependency there is also indirect dependency, which is called conditional dependency.

3.1.1 Conditional dependency

In probability theory, conditional dependency is about the relationship between two or more random variables when an extra random variable occurs. Two random variables A and B are said to be conditionally independent given a third random variable C if:

$$P(A \cap B|C) = P(A|C)P(B|C) \tag{3.2}$$

The definition of conditional probability for two variables A and B follows from Bayes' rule [20] and says that:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} = \frac{P(A \cap B)}{P(B)} \tag{3.3}$$

Then, if definition 3.3 is conditioned on a third variable C:

$$P(A|B \cap C) = \frac{P(A \cap B|C)}{P(B|C)} \tag{3.4}$$

If A and B are said to be conditionally independent given a third variable C, that means that A only depends on C [11]. This is can be expressed by using 3.2 in 3.4:

$$\begin{aligned} P(A|B,C) &= \frac{P(A \cap B|C)}{P(B|C)} \\ &= \frac{P(A|C)P(B|C)}{P(B|C)} \\ &= P(A|C) \end{aligned} \tag{3.5}$$

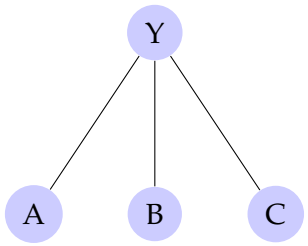
So, in 3.5 it can be seen that when A is conditionally independent of B given C:

$$P(A|B,C) = P(A|C) \tag{3.6}$$

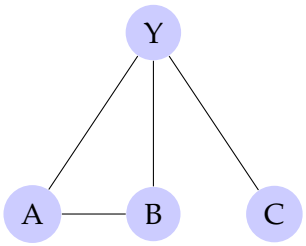
In Equation 3.6 it can be seen that when A and B are conditionally independent, it does not matter if B is mentioned in the probability of A given C. Since A and B are conditionally dependent, formula 3.6 also holds when A and B are exchanged. This is defined in 3.7 [5].

$$\begin{aligned} \text{if: } P(A|B,C) &= P(A|C) \\ \text{then: } P(B|A,C) &= P(B|C) \end{aligned} \tag{3.7}$$

The conditional dependency and the conditional independency can be explained graphically [27]. In Figure 3.1 two examples of a graphical network of four nodes A, B, C and Y can be seen. The graphical network represents the dependency structure between nodes, where each node represents a label. In the case of this research, node Y represents the true label and the nodes A, B and C represent weak labels. When there is a line drawn between two nodes it means that the nodes, and so the variables, depend on each other. In Figure 3.1a it can, for example, be seen that if you start from node A you can not go directly to node B. You can only reach node B indirectly via node Y. This is a good example of conditional independency since it means that $P(A|B,Y) = P(A|Y)$. In Figure 3.1b it can be seen that there is a line between node A and node B. This means that A and B are not conditionally independent given Y since you can go directly from A to B and you do not have to go indirectly via Y. So, it can be stated that $P(A|B,Y) \neq P(A|Y)$. In this study the conditional independence of Y is out of scope. All variables should depend on Y directly to be called a weak label. More information about weak labels and their relationship with the true label will be explained in section 3.2.1.



(A) Labels A, B and C are conditionally independent given Y.



(B) Labels A and B are conditionally dependent given Y.

FIGURE 3.1: Example of graphical networks with conditional dependency (B) and conditional independency (A).

3.1.2 Cliques

In Section 2.1 it was explained that in the Snorkel program a way of "majority voting" can be used. In this approach each weak label can be seen as a voter. However, when two weak labels are conditional dependent their votes are both counted, while in fact they express similar heuristics. To avoid this "double counting" problem the dependency structure can be included in the Snorkel program [23]. When multiple variables are conditionally dependent on each other they form a group. Groups like these are called cliques [22].

An example of a clique can be seen in Figure 3.2. In this Figure it can be seen that all nodes can be reached from node Y. Additionally, starting from node Y, it is possible to reach each of the four cliques of nodes. The nodes $1A$, $1B$ and $1C$ form a clique, the nodes $2A$ and $2B$ form a second clique and the nodes $3A$ and $3B$ form a third clique. The fourth clique is formed by the single node $4A$ since it can be seen that node $4A$ is conditionally independent of all other nodes, because $4A$ can only be reached indirectly via node Y.

For example, node $4A$ is not in the same clique as nodes $3A$ and $3B$. So, to go from $4A$ to either $3A$ or $3B$ you must go via node Y. This means that $4A$ is conditionally independent of $3A$ and $3B$ given Y. In this research the nodes, or variables, as in Figure 3.2 are known. The question is: between which nodes should a line be drawn?

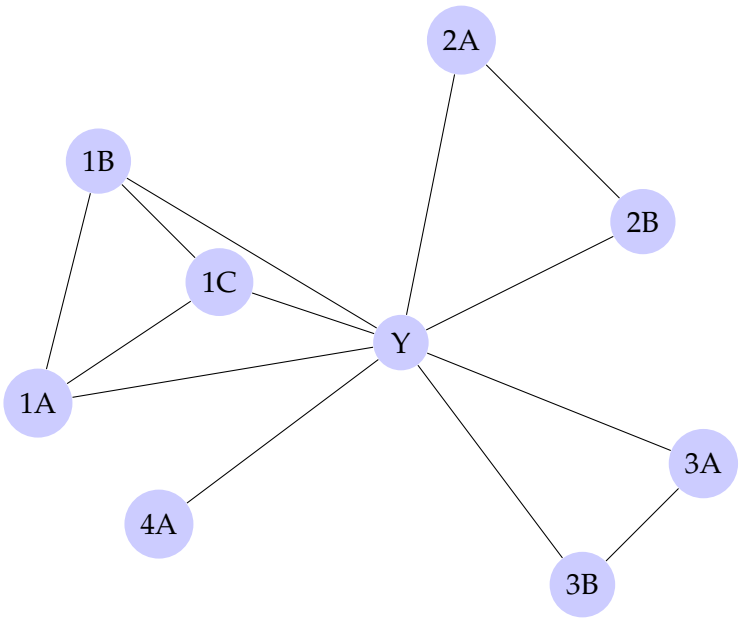


FIGURE 3.2: Example of a graphical network with 4 cliques.

3.2 Labels

In machine learning, supervised learning is a task of learning a function how to map an input into an output based on examples. In our case we do not have these examples since our data is not labeled. Therefore, we use labels based on the theory of weak supervision [5],[23]. This section explains what these labels are.

3.2.1 Weak labels

In statistics a variable is any characteristics, number, or quantity that can be measured or counted. In this research, we use weak labels. A weak label is a specific type of variable: it represents a labeling function. A labeling function is used to express various weak supervision sources such as patterns, heuristics, external knowledge bases, and more [23]. These labeling functions are written by subject matter experts (SMEs). For example: we can write a labeling function that checks if the word "house" appears in a sentence. If it does, it outputs True and if it does not it outputs False.

Since the labeling functions are written by SMEs it is assumed that there is a certain positive relationship between the true class and the weak label.

In this work, for simplicity, we only consider binary weak labels. Since we assume that there is a positive relationship between a weak label and the true label, the conditional probability of a certain binary weak label λ given the true class can be expressed by the inequalities 3.8 and 3.9.

$$P(\lambda = 1|Y = 1) > P(\lambda = 1|Y = 0) \tag{3.8}$$

$$P(\lambda = 0|Y = 0) > P(\lambda = 0|Y = 1) \tag{3.9}$$

It can be seen that the probability of Y being equal to weak label λ is larger than the probability of Y being unequal to weak label λ. In other words: weak labels have a positive relation with the true label Y. Expressing this dependency numerically can be done by using the *True Positive Rate* and the *True Negative Rate* for binary cases. The True Positive Rate (TPR) and the True Negative Rate (TNR) can be calculated by using the formulas in 3.10. They express the fraction of times where the true label Y and the weak label λ have the same value between 0 and 1.

$$\begin{aligned} TPR &= \frac{P(\lambda = 1 \cap Y = 1)}{P(Y = 1)} \\ TNR &= \frac{P(\lambda = 0 \cap Y = 0)}{P(Y = 0)} \end{aligned} \tag{3.10}$$

Since a weak label is expected to have a certain positive relationship with the true label, the True Positive- and Negative rate of a weak label should be equal to or larger than 0.5.

An overview of the True Positive Rates and True Negative Rates for four weak labels could, for example, can be seen in Table 3.1.

	TPR	TNR
λ_1	0.86	0.89
λ_2	0.57	0.73
λ_3	0.83	0.94
λ_4	0.69	0.75

TABLE 3.1: Example of the True Positive- and Negative Rates of 4 weak labels with the true class.

It can be seen that all values are larger than 0.5. Therefore, it can be said that there is a positive relationship between the true class and the weak labels. However, in this research we do not observe the true label Y, so the significant True Positive- and Negative rates can not be calculated. Instead we try to find a replacement of Y, called a strong label.

3.2.2 Strong labels

Besides the weak labels there is also a strong label, also known as the ground truth in supervised learning. We want the strong label to be as much as similar as possible with the true label.

We can think of an example of a strong label by considering a situation where we would like to predict if the grass outside will be wet next morning or not. In this case the fact that it is going to rain tonight can be seen as a strong label. It does not immediately mean that the grass will be wet, but there is a high likelyhood.

When the true label is unknown, but there is conditional dependency between weak labels, the dependency structure among these labels can be used by the Snorkel data programming to form a strong label [23]. In other words: the dependency structure among the weak labels can be used to combine them to form a strong label that can represent the true label.

3.2.3 Bayesian Network

The structure among weak labels can be adequately described by a Bayesian Network [15]. A Bayesian Network is a probabilistic graphical model that consists of nodes and edges. Each node corresponds to a label and an edge between nodes represents the conditional probability for the corresponding labels. A Bayesian Network summerizes the joint probability of all nodes. An example of a Bayesian Network has been given earlier in this chapter in Figure 3.1a. This Figure could be the graphical representation of a Bayesian Network. In the Figure it can be seen that node Y is connected with the nodes A, B and C. Hence, the Bayesian Network summarizes the joint probability $P(Y, A, B, C)$. It can be seen that the nodes A, B and C are conditionally independent given Y. Therefore, the joint probability of the network can be calculated with 3.11.

$$P(Y, A, B, C) = P(Y) \cdot P(A|Y) \cdot P(B|Y) \cdot P(C|Y) \tag{3.11}$$

In this research, a Bayesian Network will be used to describe the structure among a subset of weak labels. More about this can be read in Section 4.3.

3.3 Covariance

In this research, the inverse of the covariance matrix is an important tool for finding the dependency structure among the weak labels. In this section the covariance matrix and its inverse will be discussed to see what they mean and how they can help during the research.

3.3.1 Covariance matrix

The covariance matrix is a symmetric matrix that represents the covariances between weak labels [5]. An example of the covariance matrix of four randomly generated weak labels that form a dataset D can be seen in Figure 3.3.

FIGURE 3.3: Example of the covariance matrix of four weak labels.

Each non-diagonal entry in the matrix represents the covariance between the two corresponding labels. That covariance is a measure of how much two weak labels vary together. It can be seen that the matrix is symmetric because of commutativity. This is because $Cov(A, B) = Cov(B, A)$. The diagonal entries simply represent the variance of the corresponding label. Where a variance tells you how much a label varies, the covariance tells you how much two labels vary together [5]. The covariance between two weak labels X and Y can be calculated by the equations in 3.12.

$$Cov(X, Y) = E(X \cdot Y) - E(X) \cdot E(Y) \tag{3.12}$$

To compose the covariance matrix, the covariances need to be calculated between the estimated true label Y and itself, the conditional independent weak labels, and the other weak labels. The covariance between Y and itself, a conditionally independent weak label and an other weak label can be calculated by using the formula in 3.13, 3.14 and 3.15 respectively.

$$\begin{aligned} Cov(Y = 1, Y = 1) &= p(Y = 1, Y = 1) - P(Y = 1) \cdot P(Y = 1) \\ &= P(Y = 1) - P(Y = 1)^2 \\ &= P(Y = 1) \cdot [(1 - P(Y = 1))] \\ &= P(Y = 1) \cdot P(Y = 0) \end{aligned} \tag{3.13}$$

$$\begin{aligned} Cov(Y = 1, \lambda_{ci} = 1) &= P(Y = 1, \lambda_{ci} = 1) - P(Y = 1) \cdot P(\lambda_{ci} = 1) \\ &= P(\lambda_{ci} = 1|Y = 1) \cdot P(Y = 1) - P(Y = 1) \cdot P(\lambda_{ci} = 1) \\ &= P(Y = 1)P(Y = 0) \cdot [P(\lambda_{ci} = 1|Y = 1) - P(\lambda_{ci} = 1|Y = 0)] \end{aligned} \tag{3.14}$$

$$\begin{aligned} Cov(Y = 1, \lambda = 1) &= P(Y = 1, \lambda = 1) - P(Y = 1) \cdot P(\lambda = 1) \\ &= P(\lambda = 1|Y = 1) \cdot P(Y = 1) - P(Y = 1) \cdot P(\lambda = 1) \\ &= [P(\lambda = 1|Y = 1) - P(\lambda = 1)] \cdot P(Y = 1) \end{aligned} \tag{3.15}$$

Let's take Figure 3.3 as an example. If we observe from the dataset D that $P(\lambda_1 = 0) = 1 - P(\lambda_1 = 1) = 0.1$ we can use 3.13 to calculate the covariance between weak label 1 and itself:

$$Cov(\lambda_1 = 1, \lambda_1 = 1) = P(\lambda_1 = 1) \cdot P(\lambda_1 = 0) = 0.9 \cdot 0.1 = 0.09 \tag{3.16}$$

Then, if we want to calculate the covariance between weak label 1 and weak label 2 we can use 3.15. We already observed $P(\lambda_1 = 1$ from the dataset D. The other probabilities that we have to observe from D are: $P(\lambda_2 = 1|\lambda_1 = 1)$ and $P(\lambda_2 = 1)$. Let's say we observe for them the values 0.08 and 0.013 respectively. Then, the covariance between weak label 1 and weak label 2 can be calculated by:

$$\begin{aligned} Cov(\lambda_1 = 1, \lambda_2 = 1) &= [P(\lambda_1 = 1|\lambda_2 = 1) - P(\lambda_2 = 1)] \cdot P(\lambda_1 = 1) \\ &= (0.08 - 0.013) \cdot 0.9 \approx 0.06 \end{aligned} \tag{3.17}$$

When the formulas have been used to fill in the covariance matrix, the matrix can be inverted to retrieve the precision matrix.

3.3.2 Precision matrix

The inverse of the covariance matrix can also be called the precision matrix [5]. To invert the covariance matrix the Moore–Penrose method is used [3]. Where the non-diagonal entries of the covariance matrix show how much the corresponding weak labels co-vary, the non-diagonal entries of the precision matrix show how much the corresponding weak labels do not co-vary. Loh & Wainwright state that when entry i, j in the precision matrix is zero, the corresponding weak labels are conditionally independent given all other labels [17]. An example of the precision matrix can be seen in Figure 3.4. This is not the inverse of the covariance matrix in Figure 3.3.

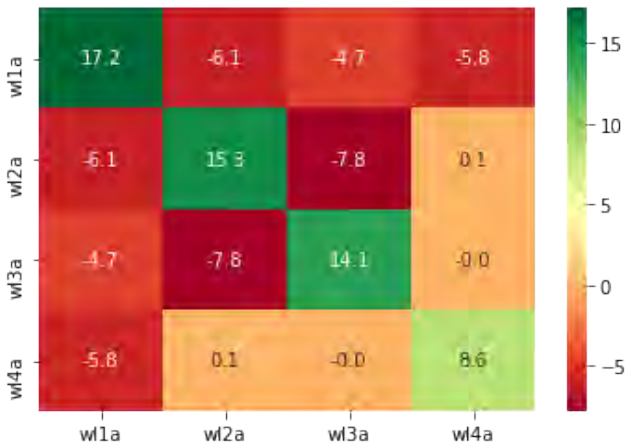


FIGURE 3.4: Example of the precision matrix of four weak labels.

It can be seen that only the entries that correspond with weak label 3*a* and weak label 4*a* are approximately zero. That means that weak label 3*a* and weak label 4*a* can be assumed to be conditionally independent given the other weak labels. It can also be seen that the entries between weak label 2*a* and weak label 4*a* are equal to 0.1. Since that value is close to zero it is hard to say whether weak label 2*a* and weak label 4*a* are conditionally independent or not. In that case, a threshold can be stated

to determine whether the value of an entry of the precision matrix is small enough to assume that the corresponding weak labels are conditionally independent.

3.4 Robust PCA

In this work we use Robust Principal Component Analysis (RPCA) to filter statistical noise from the inverse covariance matrix [28]. Since we calculate the covariances from a finite dataset they will not be exactly equal with the true covariances. These small differences are called noise. Robust PCA is a modification of regular Principal Component Analysis and can be used to divide a matrix into a low-rank component, which is the noise, and a sparse component that represents the "cleaned" inverse covariance matrix. The rank of a matrix is the number of linearly independent vectors. So, when a matrix is low-rank it means that there is a low number of linearly independent vectors. When a matrix is sparse, it means that the matrix contains mostly values that are zero [29]. There are multiple approaches for Robust PCA, for example the one used by Chandrasekaran et al. [8]. In this work, we use a Robust PCA algorithm called Principal Component Pursuit introduced by Candes et al.[7] to divide the inverse of the covariance matrix into a sparse component and a low-rank component.

3.4.1 Principal Component Pursuit

The Principal Component Pursuit (PCP) algorithm can be used to divide a matrix into a low-rank and a sparse component. The low-rank component represents a stationary background and the sparse component represents the values that we are actually interested in. Candes et al. [7] used a nice example of the application of Robust PCA on the variation between video frames of security cameras that can be seen in Figure 3.5.

FIGURE 3.5: Example of the division of a matrix by using Robust PCA.

In Figure 3.5 we can see three images. The left image is a videoframe that is recorded by a security camera. The middle and the right image are respectively the low-rank and the sparse component that have been retrieved by applying Robust PCA on the left image. In this videoframe the office can be seen as the stational background, while the walking man is the element that users are interested in. By using a sequence of these video frames, the stationary background can be seperated from the movements in the front. In the middle image it can be seen that the man is filtered from the image, while on the right image only the man can be seen.

The PCP algorithm can be stated as a minimization problem that can be seen in 3.18.

$$\begin{aligned} \min \; & ||L||_* + \lambda ||S||_1 \\ \text{s.t. } & L + S = M \end{aligned} \tag{3.18}$$

In this problem, M is the matrix that must be divided, L is the low-rank matrix and S is the sparse matrix. To solve this problem they used a augmented Lagrange multiplier.

The approach of the PCP algorithm is to first minimize the augmented Lagrangian l with respect to L (fixing S), then minimize l with respect to S (fixing L), and then finally update the Lagrange multiplier matrix Y based on the residual M = L - S, a strategy that can be seen in the Algorithm 1 below [16], [30].

Algorithm 1 (Principal Component Pursuit by Alternating Directions)

1: **initialize** $S_0 = Y_0 = 0, \mu > 0$.
2: **while** not converged **do**
3: compute $L_{k+1} = D_\mu(M - S_k - \mu^{-1}Y_k)$;
4: compute $S_{k+1} = S_{\lambda\mu}(M - L_{k+1} + \mu^{-1}Y_k)$;
5: compute $Y_{k+1} = Y_k + \mu(M - L_{k+1} - S_{k+1})$;
6: **end while**
7: **output:** L,S

Approach

This section outlines all aspects of the research description, starting with a description of the approach. After this, the corresponding research questions and success criteria will be discussed.

The goal of this research was to see under which circumstances the graph dependency structure can be retrieved from data.

Our proposed approach to make an estimation of the dependency structure can be divided in five steps:

1. Find a (minimal) subset;
2. Fit a Bayesian Network to find the class balance;
3. Fill the non-observed part of the covariance matrix;
4. Apply Robust PCA on the inverse of the covariance matrix;
5. Create a combination matrix.

The following sections describe for each of the steps why they are needed and how precisely they work.

4.1 Observable covariance matrix

If you have a labeled data set, one can - theoretically - retrieve the dependency structure directly from the inverse of the covariance matrix. However, since the true label of the dataset is unknown in our case, the full covariance matrix is not available. We only know the covariance between the dependent weak labels since you can observe the dependent weak labels. That means that it is not possible to calculate the complete covariance matrix directly. We can only calculate the covariance between the observed weak labels. That said, the covariance matrix Σ of a dataset can be expressed as in 4.1.

$$\mathbf{Cov}[O \cup S] := \Sigma = \begin{bmatrix} \Sigma_S & \Sigma_{OS} \\ {\Sigma^T}_{OS} & \Sigma_O \end{bmatrix} \tag{4.1}$$

Note that Σ has been divided in an observable component O and three non-observable components. The three non-observable components are related to the unknown true label. The part OS consists of the covariances between the true label and the weak labels and the part SO equals the transposed part OS. The last part S is simply the covariance between the true label and itself. But as said, the non-observable parts of the covariance matrix cannot be calculated yet. Varma et al.[25] used an alternative approach to find the dependency structure. They applied Robust PCA on the observable part of the inverse covariance matrix. In this research we try to estimate (or fill) the non-observed part of the covariance matrix. The expectation

is that when you do that you add information to the covariance matrix. This is because now, the true label will also be taken into account when the inverse covariance matrix is calculated. The inverse of the covariance matrix will be more similar with the true inverse covariance matrix. This will result in a dependency structure that is more visible then when only the observable part of the covariance matrix is used. To fill the non-observed part of the covariance matrix the covariances between the weak labels and the true label must be estimated. In section 3.3 it was explained that the conditional probability of a weak label given the true label and the class balande are used to calculate the covariance between a weak label and the true label. We note that from a subset of conditional independent weak labels, we can estimate these probabilities and the class balance. When the data is generated we try to find at least 4 conditionally independent weak labels in the data set.

4.2 Find a (minimal) subset

To estimate the class balance and the conditional probabilities between the true label and the conditional independent weak labels at least four conditional independent weak labels must be found. The reason why we need at least four weak labels will be explained in Section 4.3.2.

We cannot conclude yet which weak labels are conditionally independent since we do not know the true label. However, we can make an estimation of the weak labels that are the most likely(!) to be conditionally independent based on the inverse of the observable part of the covariance matrix.

Loh & Wainwright [17] showed that when you look at the inverse covariance matrix the corresponding weak labels of the entries that equal zero are said to be conditionally independent given all other weak labels. This works both ways, so when two weak labels are conditionally independent their corresponding entry in the inverse of the covariance matrix should equal zero. The entries of the inverse of the observable part of the covariance matrix might not be exactly zero, but they can be close to zero. However, two weak labels can be conditionally indepedent although their corresponding value in the inverse of the inverse of the covariance matrix is small, but not exactly zero. To find the weak labels that are the most likely to be conditionally independent we use a greedy method. We pick the smallest absolute value of the inverse of the observed covariance matrix since a value that approximates zero implies conditionaly independence of the two corresponding weak labels. As an example you can imagine an observable part of the inverse covariance matrix that consists of 5 weak labels. In Figure 4.1 an example of finding the first two conditional independent weak labels can be seen. In this Figure it can be seen that the smallest absolute value is at the entries between weak label 2 and weak label 3. So, these two are in this case the two weak labels that are the most likely to be conditionally independent.

The third weak label that is the most likely to be conditionally independent can again be found by looking at the inverse of the observed covariance matrix. The third weak label must be the one that is the most likely to be conditionally independent with respect to the first two conditionally independent weak labels that we found. In other words: a linear combination of the first two weak conditionally independent weak labels is compared to the other weak labels. To do this, the sum of the absolute values of the corresponding entries of the first two weak labels has been calculated. An example of this process can be seen in Figure 4.2.

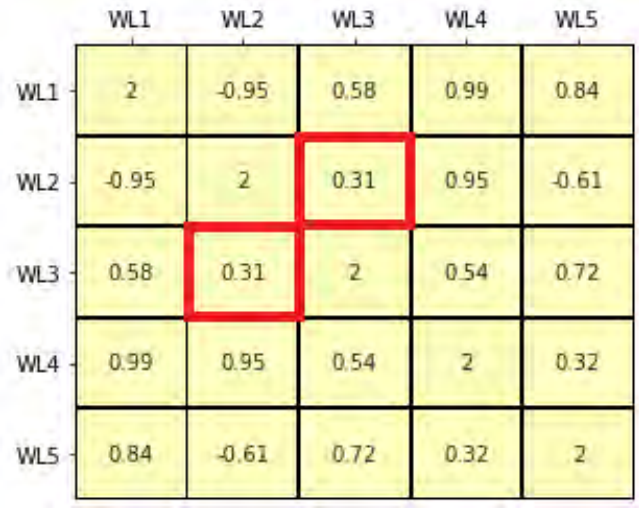


FIGURE 4.1: Example of the inverse covariance matrix with the smallest absolute value selected.

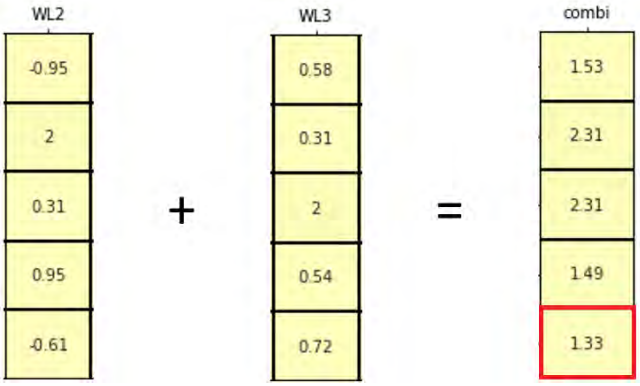


FIGURE 4.2: A linear combination of the entries of the first two conditional independent weak labels to find a third.

Calculating the sum of the absolute values for each entry results into a new vector. The closer each value in this vector is to zero, the more likely the corresponding weak label is to be conditionally independent of the first two conditionally independent weak labels [17]. So, again the smallest absolute value of the vector is chosen, as long as the corresponding weak label is not equal to one of the first two weak labels. This results in weak label 5 as the third weak label that is the most likely to be conditional independent. To find the fourth weak label again the smallest absolute value of can be taken from a linear combination of the first three weak labels. Again, as long as the corresponding weak label is not already in the set of conditionally independent weak labels. A visual example of this can be seen in Figure 4.3.

It can be seen that from all of the weak labels that have not been assumed to be conditionally independent the value that is the closest to zero is the index that corresponds to weak label 4. So, the four weak labels that are the most likely to be conditionally independent are weak labels 2, 3, 5 and 4.

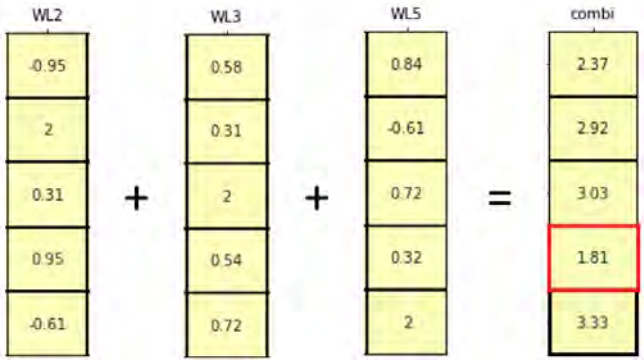


FIGURE 4.3: A linear combination of the entries of the first three conditional independent weak labels to find the third one.

4.3 Bayesian Network fit

In the first step we have created a subset of the four weak labels that were the most likely to be conditional independent. Now that we can assume that these weak labels are conditional independent of each other and that they all depend on the true label, we have already estimated a part of the dependency structure. This dependency structure can be described by a Bayesian Network. A graphical representation of this can be seen in Figure 4.4.

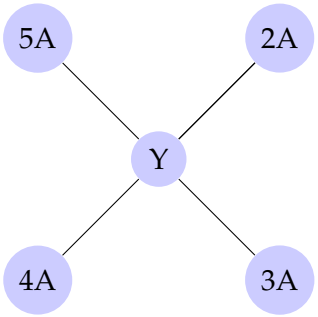


FIGURE 4.4: Example of the estimated dependency structure of the subset created in step 1.

In Figure 4.4 it can be seen that all weak labels are conditional independent of each other given the true label. In that case, when a weak label α can be written as λ_α the structure of this Bayesian Network can be written as in 4.2.

$$P(Y) \cdot P(\lambda_{2A}|Y) \cdot P(\lambda_{3A}|Y) \cdot ... \cdot P(\lambda_{5A}|Y) \tag{4.2}$$

In 4.2 it can be seen that the class balance $P(Y)$ and the conditional probability of each weak label in the subset are used to describe the structure of the Bayesian Network. The question is: what values should these probabilities have? To find that out, we can fit the Bayesian Network structure on our subset[24]. In other words: by fitting a Bayesian Network to the subset, we can estimate the class balance and conditional probabilities of the four conditionally independent weak labels given the true class. Later, we can use these probabilities to fill the unobservable part of the covariance matrix.

To fit the Bayesian Network we will focus on the counts of each possible state in our subset. Since there are four binary weak labels in our subset, each data sample

can be one of $2^4 = 16$ states. For example: the first two weak labels can have a value of 1 and the last two weak labels can have a value of 0, which gives a state of $[1, 1, 0, 0]$. From the data of the weak labels the number of occurrences of each state can be counted. The goal of the fit of a Bayesian Network is to find a Bayesian structure, consisting of probabilities, that would describe these counts. So, if the data for example consists of 100.000 samples the distribution of the counts of all 16 states could be represented as in list 4.1 below:

State				**Counts**
λ_2	λ_3	λ_4	λ_5	
0	0	0	0	10.104
0	0	0	1	4.329
0	0	1	0	4.331
...	...	...	...	...
1	1	1	1	28.201

TABLE 4.1: Example of the counts of each possible state of a subset of 4 binary weak labels.

In the left column of the Table each possible states that a sample of the subset can be is described, while on the right side the number of occurrences of these states in the subset can be seen. What we want is to fill the probabilities in 4.2 so that when we draw 100.000 samples from our Bayesian Network we get exactly the same counts as in Table 4.1. To do that, we use a Binned Likelihood function.

4.3.1 Binned Likelihood

As said, the goal is to fill in these probabilities in a way that if you take N random samples from the Bayesian Network you would get the same counts of each possible state is in the real data. Filling in these probabilities so that the Bayesian Network describes the data can be done by using a Poisson Binned Likelihood function L, since we are dealing with counts. If we let f be the predicted counts per state and y be the observed number of counts we can write L as in 4.3 [2].

$$L = f - y \cdot \log f + \ln |\Gamma(y + 1)| \tag{4.3}$$

To calculate the predicted counts the parameters of the Bayesian Network are used. An example of these parameters can be seen below in 4.2 :

Parameter
$P(\lambda_2 = 0 \mid Y = 0)$
$P(\lambda_2 = 0 \mid Y = 1)$
$P(\lambda_3 = 0 \mid Y = 0)$
$P(\lambda_3 = 0 \mid Y = 1)$
...
P(Y = 0)

TABLE 4.2: Example of the parameters of the Bayesian Network that is fit on the subset of the data.

It can be seen that the parameters describe the probabilities of the weak labels given Y and the probability of Y being zero. Since all weak labels are binary, all

conditional probabilities given Y and the class balance can be estimated with these estimations. For example:

$$P(\lambda_1 = 1|y = 0) = 1 - P(\lambda_1 = 0|y = 0)$$
$$\textit{and}$$
$$P(Y = 1) = 1 - P(Y = 0)$$

For optimizing the Binned Likelihood function the Minuit optimizer will be used [1].

4.3.2 Goodness of fit test

After the conditional probabilities have been estimated by using the optimizer, we would like to test the quality of our fit. The parameters of the fit will be used in the next step of our approach. Therefore, we want their quality to be high. The quality of our fit depends on how much the parameters of the fit are similar with the true conditional probabilities. To summarise, we want to test whether the parameters of our fit are equal with the true conditional probabilities of the corresponding weak labels. To test whether multiple probabilities are equal we can use a χ^2 goodness of fit test [13], [12]. This test can answer the question if there is a significant difference between the parameters of the Bayesian Network and the true conditional probabilities of the four conditional independent weak labels. The hypotheses of this test can be stated as:

H_0 : The conditional independent structure is correct, based on the fit;
H_α : The conditional independent structure is not correct, based on the fit.

To see whether the null hypothesis should be rejected, the χ^2 test statistic can be calculated as explained in the paper of Pearson [13] and compared to the critical value. To calculate the test statistic the true counts and the expected counts are compared. The true counts can be observed by looking at the number of appearances of each possible state of the four conditional independent labels, combined with the true label. The estimated counts can be found by multiplying each parameter of the fit with the total number of samples. When the test statistic is larger than the critical value it means that the differences between the counted bins and the counted samples from the fitted Bayesian Network are too big. In other words: the Bayesian Network does not fit the data well. In this work a confidence level of 95% is used.

To use this technique to test the quality of the fit the size of the subset must be at least four. Since we are using binary weak labels, the number of different states in our subset of size N can be written by 4.4:

$$\#states = 2^N \tag{4.4}$$

For each weak label in the subset the conditional probabilities $P(\lambda_1 = 0|Y = 0)$ and $P(\lambda_1 = 0|Y = 1)$ are used as parameters in the fit. Additionally, $P(Y = 0)$ is a parameter and there is a scale parameter representing the number of data points. Therefore, the number of parameters of the fit can be calculated by:

$$\#parameters = 2N + 2 \tag{4.5}$$

The development of the number of states and the number of parameters when N is increasing can be seen in Table 4.3.

N	States	Parameters
1	2	4
2	4	6
3	8	8
4	16	10
5	32	12

TABLE 4.3: Example of the development of the number of states and parameters when N increases.

In Table 4.3 it can be seen that when N is smaller than 3 there are more parameters than states. In terms of linear algebra, this means that there are fewer equations than unknowns which is called: underdetermined. The system can always be solved [14] [10]. This results in a χ^2-test statistic that will always be zero. When $N = 3$ the number of parameters and states is equal so then, the system is perfectly solvable. However, when N is larger than 3 there are more states than parameters. This means that the system is not always solvable and the χ^2-test statistic will not always be zero. Therefore, the size of our subset must be at least 4 to be able to test the quality of our fit.

The critical value of our test can be calculated by looking at a χ^2 distribution with a certain amount of degrees of freedom. These degrees of freedom can, in this case, be calculated as follows:

$$df = \text{Number of probabilities} - 1 \tag{4.6}$$

When the χ^2 distribution with the degrees of freedom has been set up, a significance level has to be stated. This significance level represents the strength of the evidence that must be present in the test statistic before you can draw a conclusion about the null hypothesis. A significance level of 95% has been used in the experiments. This means that when the null hypothesis is rejected, it has been done with 95% certainty that the correct decision has been made. A significance level of 95% means that when you want to find the critical value you select the value of the 95% quantile of the χ^2 distribution with the calculated degrees of freedom [13]. If the χ^2 test statistic is smaller than the critical value the null hypothesis is not rejected and it can be concluded that the Bayesian Network fits the data. So, it can be assumed that the estimated conditional probabilities of the weak labels and the estimated class balance are valid. These can now be used to fill in the non-observable part of the covariance matrix.

4.4 Fill the non-observed part of the covariance matrix

When the conditional probabilities of the assumed to be conditional independent weak labels and the class balance have been calculated, they can be used to fill in the the covariance matrix. As a small recap: in 4.1 it was shown that the covariance matrix could be divided in an observable and three non-observable parts. Another example of this can be seen in Figure 4.5. In this Figure the green part represents

the observable part of the covariance matrix. The white part represents the non-observable part. It can be seen that the covariance between the weak labels can be calculated from the observed data, but the covariance between Y and a weak label cannot, since Y is unknown. Each cell in the Figure represents the source where the probabilities have been retrieved from to calculate the covariance between the corresponding labels.

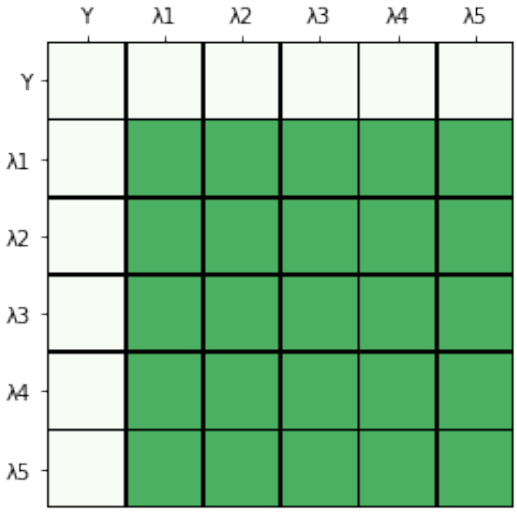


FIGURE 4.5: Example of the covariance matrix entries of the true class Y and 5 weak labels.

The covariance between Y and a weak label and the covariance between Y and itself can be calculated by respectively using the formulas 4.7 and 4.8 [9].

$$\begin{aligned} Cov(Y = 1, \lambda = 1) &= [P(\lambda = 1|Y = 1) - P(\lambda = 1)] \cdot P(Y = 1), \\ Cov(Y = 0, \lambda = 0) &= [P(\lambda = 0|Y = 0) - P(\lambda = 0)] \cdot P(Y = 0). \end{aligned} \tag{4.7}$$

$$Cov(Y, Y) = P(Y = 1) \cdot P(Y = 0) \tag{4.8}$$

Now that the class balance $P(Y)$ and the conditional probabilities of the weak labels that were most likely to be conditional independent have been estimated, it is possible to fill in these unknown part of the covariance matrix by using the equations above in 4.7. In the example of section 4.2 at the end the weak labels 2, 3, 5 and 4 were chosen as the most likely to be conditionally independent. If these weak labels indeed have been used for fitting the Bayesian Network their estimated conditional probabilities $P(\lambda|Y)$ and the estimated class balance $P(Y)$ can be used to fill in the covariance matrix. In Figure 4.6 it can be seen what the covariance matrix will look like after the probabilities from the fit have been used for calculating the covariances of Y with itself and the weak labels that were most likely to be conditional independent. The covariance entries in Figure 4.6 that are calculated by using the parameters of the fit are colored yellow.

It can be seen that the covariance matrix now has been filled almost completely since there are still two red entries. In this case, only the covariance between Y and weak label 1 is still unknown. Since this weak label was not one of the weak labels that were the most likely to be conditional dependent the conditional probability $P(\lambda|Y)$ are still unknown. To estimate this probability, the average of the conditional

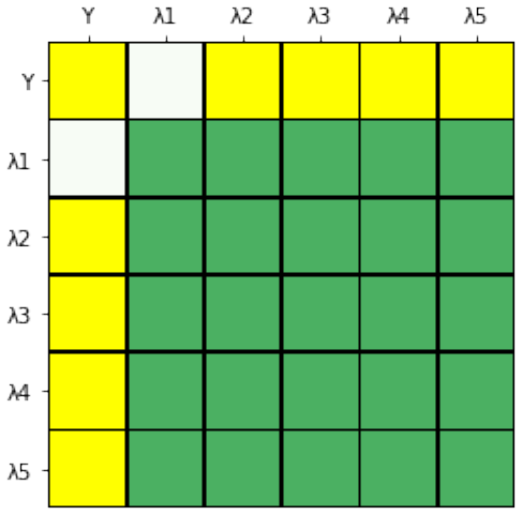


FIGURE 4.6: Example of the covariance matrix entries sources after the probabilities from the fit have been used.

probabilities from the fit has been calculated. So, if weak label i is not in the set of conditional independent weak labels N then $P(wl|Y)$ can be calculated with the formula in 4.9.

$$P(\lambda_i|Y) = \frac{\sum_{P(\lambda_j|Y)}}{N}, j \in N \tag{4.9}$$

Now that an assumption has been made for the conditional probability of the weak labels that were not in the fit, the covariance matrix can be filled in completely as in Figure 4.7. In this Figure it can be seen that the covariance entries that are calculated by using the assumptions are colored grey.

Now, the covariance matrix is filled completely and can be inverted to find the graph dependency structure. But, the inverse of the covariance matrix might contain noise, because of the assumptions that have been made in the previous step. To filter out the noise from the inverse covariance matrix Robust PCA can be used.

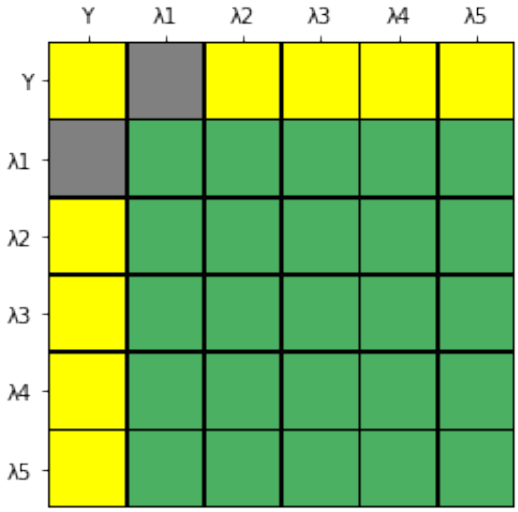


FIGURE 4.7: Example of the covariance matrix entries sources after the remaining conditional probabilities have been assumed.

4.5 Filter out noise

By following the steps in the previous sections the inverse of the covariance matrix has been found and can be used to read the graph structure from. However, the conditional probabilities that have been retrieved by using the fit of the Bayesian Network and by making the assumption of taking the average probabilities of the fit might have caused some noise in the covariance matrix. Additionally, we calculated the covariances from a finite dataset, so there will generally be noise. For example: the covariance between λ and itself will not be exactly $P(\lambda = 0) \cdot (1 - P(\lambda = 0))$. In other words: the estimated values in the covariance matrix are not exact. That means that when the covariance matrix is inverted, the inverse of the covariance matrix is also not correct. Due to the errors the correct graph structure might not be detectable from the inverse of the covariance matrix. A common weak supervision assumption is that the graph structure that can be red from the inverse covariance matrix is sparse. Let's say we take the inverse K of the true covariance matrix in 4.1. Then, K can again be divided in four parts as shown in 4.10 [25].

$$\Sigma^{-1} := K = \begin{bmatrix} K_S & K_{OS} \\ K^T{}_{OS} & K_O \end{bmatrix} \tag{4.10}$$

From the block chain inversion formula [4],

$$K_O = \Sigma_O^{-1} + c\Sigma_O^{-1}\Sigma_{OS}\Sigma_{OS}^T\Sigma_O^{-1} \tag{4.11}$$

In this equation c can be written as in 4.12.

$$c = (\Sigma_S - \Sigma_{OS}^T\Sigma_O^{-1}\Sigma_{OS})^{-1} \tag{4.12}$$

If we let $z = \sqrt{c}\Sigma_O^{-1}\Sigma_{OS}$, then we can write 4.11 as:

$$\Sigma_O^{-1} = K_O - zz^T \tag{4.13}$$

In equation 4.13 the observable part of the inverse covariance matrix is represented by Σ_O^{-1}, while the sparse K_O is the part that we are interested in because that is where the graph structure should be readable from. The noise is represented by zz^T and is expected to be a low-rank matrix since it is a multiplication of a matrix with its transposed self. Hence, the inverse of the filled covariance matrix should be divided in a sparse part and a low-rank part. To do that, we will use the Robust PCA algorithm Principal Component Pursuit (PCP). This is the same Robust PCA algorithm that was used in the paper of Varma et al. [25], although they used the loss function of Wu et al.[28]. After the usage of Robust PCA the retrieved sparse component should represent the desired dependency structure.

It can be said that the inverse of the covariance matrix M can be decomposed in two parts so that:

$$M = L + S \tag{4.14}$$

In this decomposition L is the low rank matrix of noise that corresponds to zz^T while S is the sparse underlying true inverse covariance matrix K_O that we are interested in. To divide the inverse covariance matrix into these low rank and sparse components Robust PCA algorithm Principal Component Pursuit (PCP) can be used. After the use of this algorithm the sparse component that has been retrieved can be used to read the graph dependency structure from.

4.6 Combination matrix

When the covariance matrix is filled and inverted, the dependency structure might already be notable from it. However, when the PCP algorithm is applied and the noise is removed from the inverse covariance matrix and the structure might be even more visible. An example of this can be seen in Figure 4.8.

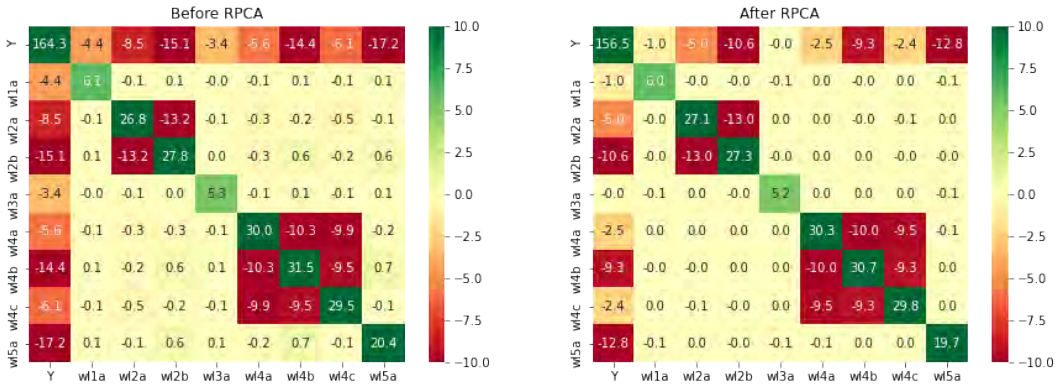


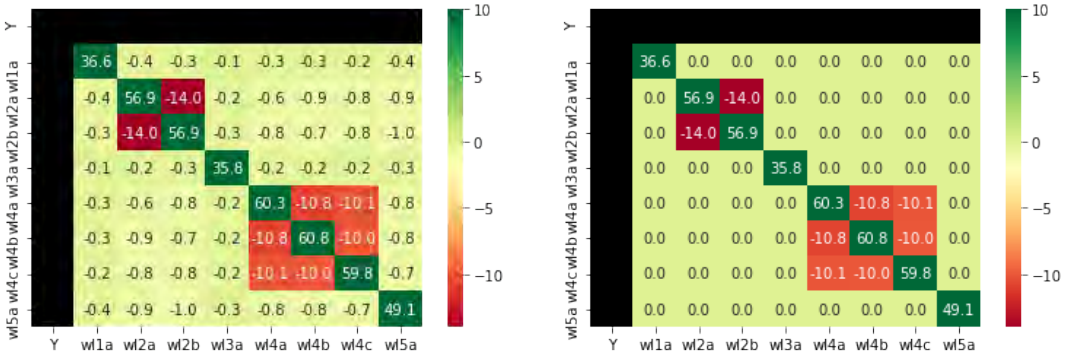
FIGURE 4.8: Example of the inverse covariance matrix before and after Robust PCA.

In Figure 4.8 it can be seen in the left matrix that before the usage of Robust PCA the dependency structure might already be visible. But, after the usage of Robust PCA the structure is even more visible since most of the entries that should approximate zero actually changed into zero. In other words: after Robust PCA we can say about an entry that it approximates zero with more certainty. However, some entries might still not approximate zero while they should. Therefore, we try to find the dependency structure by using a combination of our approach and the approach of Varma et al [25]. We use Varma's approach of making a conservative estimation of the dependency structure with our approach that has more certainty about entries approximate zero. We form a combination matrix. If the inverse covariance matrix after Robust PCA is noted with M and the inverse covariance matrix of Varma et al.[25] with V, the entries of the combination matrix C are filled as in 4.15.

$$C_{ij} = \begin{cases} 0 & \text{if } M_{ij} < t_m. \\ 0 & \text{if } V_{ij} < t_v. \\ V_{ij} & \text{otherwise .} \end{cases} \qquad \textit{where} \qquad t_v = \max V_1, ..V_N \tag{4.15}$$

In 4.15 N is the collection of entries of M that are smaller than a certain threshold t_m. In that way, we collect the entries of M that are approximately zero. Of these entries we know almost for sure that they are zero. Then, we compare the matching entries from V and calculate the maximum value of those entries to form the threshold t_v. In other words: we use the accurate estimation of our approach to get a threshold for the estimation of Varma et al.[25] In that way, the entries that are classified as approximately zero by our approach remain zero, but some zero entries that our approach might have missed can be compensated by the approach of Varma et al. An example can be seen in Figure 4.9.

In Figure 4.9a the dependency structure estimation by the approach of Varma et al.[25] can be seen. When we apply 4.15 on this matrix and our estimation after



(A) Estimation by Varma et al.

(B) Combination of Varma's and our approach.

FIGURE 4.9: Inverse covariance matrices

Robust PCA in Figure 4.8, the result can be seen in Figure 4.9b. It can be seen that in Figure 4.9b the dependency structure is perfectly readable from the matrix.

Experiments

In the previous chapter the approach to estimate the dependency structure was explained. In this chapter the experiments that will be done to see to what extent Robust PCA can be combined with other methods to estimate the dependency structure. This chapter outlines all aspects of the experiments, starting with a description of the experimental parameters and the generation of datasets. After this, the evaluation methods will be discussed and finally, the 4 experiments will be described in detail.

5.1 Parameters and setup

The experimental setup is determined by seven parameters:

- Structure S;
- Equality rate r;
- Class balance p;
- Number of samples N;
- Runs R;
- Size of subset nci;
- Maximum threshold t_m.

The first four parameters are used for generating a dataset and will be discussed later in 5.2. The runs parameter R represents the number of times the experiment will be repeated with a different seed to determine whether a dataset was a fluke, or it represents a normal case. After these runs the average of the results will be calculated for evaluation. The size of subset parameter nci represents the number of weak labels that will be chosen in the first step of the approach as the weak labels that are most likely to be conditional independent. These weak labels will be used in step two to create a subset that will be used to fit a Bayesian Network on. The parameter t_m is the maximum threshold that is used for determing whether an entry of the resulting inverse covariance matrix of our approach is zero or not. For our Robust PCA algorithm we will use a maximum of 10.000 iterations.

In total, four experiments will be done to test whether our approach works under different circumstances:

- Structure recovering using synthetic data;
- Data structure changes;
- Varying interaction strength;

- Varying subset size.

In the first experiment, the approach will be tested for a dataset where all weak labels are conditional independent. This is supposed to be the simplest dataset that our approach should work for. After that, more dependency will be introduced to make the data more complicated. In the second experiment the influence of changes of the data structure will be tested. In the third experiment the influence of changes of the class balance and the number of fitted weak labels will be tested. The final experiment will test how the number of chosen weak labels in the first step affects the final result. Each experiment consists of three steps:

1. Data generation;
2. Structure learning;
3. Evaluation.

In the next sections, these steps will be discussed.

5.2 Data generation

For the experiments of this research multiple synthetic data sets of weak labels and a true label will be generated. Generating a dataset consists of two steps: generating the true label and adding cliques. To generate a dataset, a data generation method will be used. This data generation method uses four parameters that were shortly introduced in the beginning of this section:

- Structure S;
- Equality rate r;
- Class balance p;
- Number of samples N;

The structure parameter S respresents the structure of the weak labels in the dataset and is expressed by a sequence of integers. Each of these integers represents the size of a clique. An example of a value for the structure parameter is $(1, 2, 1, 3)$. In this example, the data consists of a true label, followed by four cliques that consist of 1, 2, 1 and 3 weak labels respectively. With this parameter the structure can be specified to make it easier to see whether the retrieved structure at the end of our approach is correct. In addition, it can be easily adjusted to generate a dataset with an other structure to test our approach on.

The equality rate r is a parameter that ranges between 0 and 1 and is used to control the dependency between the weak labels in a clique and the true label. More about the equality rate will be explained later in this section when the generation of cliques is discussed.

The third parameter is the class balance p. This is again a value that can range between 0 and 1 and represent the probability of the true label being one. In other words: $P(Y = 1)$.

The final parameter is the number of samples N. By making weak labels that are sufficiently large the true dependency structure among the generated weak labels should be visible in the inverse covariance matrix of the dataset. In the experiments we will use a value of 100.000 for N. For each experiment that will be discussed in this chapter a dataset with unique parameters will be used.

5.2.1 Generate true Label

To generate a dataset the true label Y will be generated as a start. The true label Y will be generated by drawing N samples from a binomial distribution with one trial and a probability of success p. So, the probability of success is the probability that an element of Y will have a value of one and can be called the class balance. The true label Y represents the target variable we are interested in predicting.

5.2.2 Add cliques

After the true label has been generated we will add cliques to the dataset one by one.

S clique of size one consists of one single conditional independent weak label λ. This weak label should be dependent of the true label Y but independent of the other weak labels in the dataset. To do this, the values j of λ can be generated by using 5.1.

$$\lambda_j \sim Bin(1, p_j) \qquad where: \qquad p_j \sim \begin{cases} \mathcal{U}(0.70, 0.95) & \text{if } Y_j = 1. \\ \mathcal{U}(0.05, 0.40) & \text{if } Y_j = 0. \end{cases} \tag{5.1}$$

For each weak label λ_i the value of each element j of the weak label λ_i is generated by using 5.1. It can be seen that each element of λ_i is a drawing from a binomial distribution with one trial and where the probability of success p depends on Y. When Y_j equals one, p_{ij} is a uniform distributed variable between 0.70 and 0.95. On the other hand, when Y_j equals zero, p_{ij} is again a uniform distributed value, but between 0.05 and 0.40. In other words: when Y_j equals one, the probability of λ_{ij} being one is larger than the probability of λ_{ij} being zero. This makes weak label λ_i dependent on Y. Since this process has been repeated for each weak label, they are all dependent on Y, but independent of each other. In other words: they are all conditionally independent of each other, given Y. Each of these conditionally independent weak labels form a clique of size one. But, a clique might contain more weak labels.

When a clique is added that has a size larger than one a latent variable Y' will be introduced. This latent variable is a variable that is for a certain part equal to the true label Y, but for the other part completely random. This is where the equality rate parameter r will be used. The equality parameter represents the fraction of Y' that will be equal to Y. The other part of Y' is random. Therefore, Y' is a noisy version of the true label Y. To decide which values of Y' should be equal to Y a variable z is introduced. This variable consist of N samples of a Uniform distribution with parameters (0,1). When z_j is smaller or equal to r, then the corresponding Y'_j will be equal to Y. Otherwise, Y'_j will be a sample of a Binary distribution with a success rate of 0.5. Therefore, the values j of Y' can be expressed with 5.2.

$$Y'_j = \begin{cases} Y_j & \text{if } z_j \leq r. \\ \text{Bin(1,0.5)} & \text{otherwise .} \end{cases} \qquad \text{where :} \qquad z_j = \mathcal{U}(0,1) \tag{5.2}$$

For each clique that is added z and Y' will be generated again. By doing this, the weak labels that are in a certain clique will all be drawn from the same latent variable, but not from the same latent variable as weak labels in an other clique. When the latent variable Y' is generated, the weak labels of the clique to add will be sampled from Y'. We will do this the same way as for the generation of Y'. Hence, the values j of weak label i of a certain clique can be generated by 5.3.

$$\lambda_{ij} = \begin{cases} Y'_j & \text{if } z_j \leq r. \\ \text{Bin(1,0.5)} & \text{otherwise .} \end{cases} \qquad \text{where :} \qquad z_j = \mathcal{U}(0,1) \tag{5.3}$$

When a clique is generated this way, each weak label in the clique is dependent of the true label Y. On top of that, each weak label is dependent of the other weak labels in the clique since they were all drawn from the same source, namely Y'. This way of generating cliques is equivalent to jointly generating them. After the dataset is generated we use the approach described in Chapter 4 to estimate the dependency structure among the weak labels of the dataset.

5.3 Baseline

We benchmark our structure learning approach as described in the approach chapter against the approach of Varma et al. [25]. To estimate the dependency structure, the steps as described in Chapter 4 have been followed:

1. Find conditional independent labels;
2. Fit a Bayesian Network and retrieve the class balance;
3. Fill in the covariance matrix using the network's parameters;
4. Apply Robust PCA on the inverse of the covariance matrix;
5. Create a combination matrix.

After these steps, there should be a "clean" inverse covariance matrix that could be used to estimate the dependency structure. When the structure is estimated we evaluate the quality of our estimation. Since we generated the dataset we can find out whether the estimated dependencies equal the true dependencies.

5.4 Evaluation

To evaluate the results of an experiment, the result after each of the four steps will be measured.

This step-wise evaluation can be used to assess to what extent each of the stages is causing error on the dependency structure recovery. After all, since the result of a step depends on the result of the previous step there might be cascading failures. Below, each performance metric per step will be discussed.

5.4.1 End-to-end evaluation

We benchmark our structure learning approach as described Chapter 4 against the approach of Varma et al [25]. To do this, the structure will first be estimated by using our approach. Then, the structure will be estimated by the approach of Varma et al. By using a performance metric both approaches will be compared to see whether they can be used to make a good estimation of the dependency structure.

In step 4 of our approach Robust PCA has been applied to remove noise from the inverse of the covariance matrix. After this step, it should be readable from the inverse of the covariance matrix which weak labels can be assumed to be conditional independent. When an entry of the inverse covariance matrix approximates zero, the corresponding weak labels can be assumed to be conditional independent. Since the dataset was generated, it is known which weak labels are conditional independent. The question was if, after the five steps, the same weak labels were said to be conditional independent according to the inverse of the covariance matrix. In other words: for each entry of the inverse of the covariance matrix the true value can be compared with the estimated value. The entries of the weak labels that are known to be conditional independent should (approximately) be equal to zero.

This can be formulated as a classification problem: the approach of the five steps finally classifies each entry of the inverse covariance matrix as a zero or as a nonzero value. To decide whether an entry can be classified as zero a certain threshold must be used. The threshold is the maximum value of which can be stated that it approximates zero. If the threshold is too low, too little entries of the inverse of the covariance matrix will be classified as zero, while some of them might actually be nonzero. On the other hand, if the threshold is too high, too many entries of the inverse of the covariance matrix will be classified as zero and weak labels will be classified as conditional independent, while in fact they are not. In other words, using a good threshold is important for the approach to make a good estimation of the dependency structure.

In a real case, finding the best threshold is hard since you do not know the true dependency structure. But, in the experiments we do know the true dependency structure. To compare our approach with the approach of Varma et al.[25], the best threshold for each approach will be used. To find this best threshold a grid search will be used by letting the threshold vary between 0 and the maximum threshold t_m and see for which threshold each approach performs at its best [31]. In that way it can be seen for which threshold our approach gets the best performance and for which threshold the approach of Varma et al.[25] gets the best performance. These two best performances will then be compared to see if our approach outperforms the approach of Varma et al.[25]. More about the performance metrics will be described later in this section.

Since the entries of the inverse of the covariance matrix will be classified as zero or nonzero, the difference between true values and classified values can be expressed as a true positive, false positive, false negative or true negative.

		True value	
		Zero	Nonzero
Classified value	Zero	TP	FP
	Nonzero	FN	TN

TABLE 5.1: Example of all 4 possible labels that can be assigned to each entry of the inverse covariance matrix.

In Table 5.1 it can be seen that when an entry of the inverse covariance matrix has a value of approximately zero and the corresponding weak labels are indeed conditional independent, the entry is marked as a true positive. In section 3.2.1 an introduction to true positives and true negatives has been given. Next to these true metrics there also exist false positives and false negatives. In this case: when after the 4 steps an entry in the inverse covariance matrix has a value of approximately zero, but in fact the two corresponding weak labels are not conditional independent, then that entry could be marked as a false positive and a false negative vice versa.

When the number of true positives, true negatives, false positives and false negatives have been calculated, the performance metrics can be calculated.

Since labelling the entries of the inverse covariance matrix as zero or nonzero can be seen as a binary classification problem, common performance metrics as the precision and recall can be used. However, since we are interested if the estimated dependency structure is correct, the Accuracy will be used. The Accuracy ranges between 0 and 1 and simply represents the fraction of predictions that are made correctly. It can be calculated by dividing the number of correct predictions by the total number of predictions[21]. This can be seen in 5.4.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.4}$$

The accuracy has issues when the classes are unbalanced. Nevertheless, we are interested if a single estimation of the block structure was correct, so it is used as the performance metric. The usage of the Accuracy can give insights to what extent the true dependency structure can be found by using the method as described in Chapter 4. A high value of the Accuracy means a good performance of our approach. It can also be used to compare the performance of our approach with the performance of the approach of Varma et al [25].

5.4.2 Step-by-step evaluation

Identify conditional independent labels In the first step of the approach we make a subset of at least four conditionally independent weak labels, based on their value in the inverse of the observable covariance matrix and of the linear combinations. To evaluate whether this has been done correctly, it must be checked if the chosen weak labels indeed are conditional independent. Since the data is generated, the true dependency structure is known. Therefore, it is known which weak labels are in the same clique. In a real situation this would of course not be the case.

To see whether the algorithm that selects the four weak labels that are the most likely to be conditional independent works, **the chosen weak labels are compared with the known cliques.** If none of the chosen weak labels is in the same clique as an other chosen weak label, it means that in this step indeed a subset of four conditional independent weak labels as been made. If not, the step failed.

Bayesian Network fit The second step is to fit a Bayesian Network to the data of the chosen weak labels in step 1. The parameters of this fit can be used to estimate the conditional probabilities of the weak labels given the true label. Since the data has been generated, these conditional probabilities are observable. Hence, the observable conditional probabilities can be compared with the parameters of the fit, to see whether they differ.

To do this, the χ^2 test will be used[13]. Using the χ^2 test has the advantage that this test returns a simple yes or no to the question: can we assume that the parameters of the fit are equal to the observable conditional probabilities? In that way it is easy to decide whether the fit of the Bayesian Network has been a success. To perform the χ^2 test we will compare the expected counts of each state with the true counts of each state. The true counts O_i of each state i can be observed from the data. The expected counts E_i of each state i can be retrieved by multiplying the estimated conditional probabilities with the number of samples n. Then, the test χ^2 test statistic for N parameters can be calculated by using 5.5.

$$\chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i} \tag{5.5}$$

The degrees of freedom df for this test could be calculated with $df = N - 1$. If the difference between the estimated counts and the observed counts is not significant, it can be assumed that the parameters of the Bayesian Network are sufficient and a good estimation of the conditional probabilities has been made.

Sum of squared errors covariance matrix In the third step of the approach, the unobservable part of the covariance matrix is filled in. Since the data is generated, the true covariance matrix and its inverse are known. So, the filled unobservable part of the covariance matrix can be compared with the true inverse of the covariance matrix to see whether the unobservable part has been filled correctly.

To do this, the mean squared errors (MSE) of all the unobservable entries of the covariance matrix can be calculated [26]. Let the covariance matrix be an $N \times N$ matrix with indexes $i,j \in \{0, 1, 2.., N-1\}$.

Then, O_{ij} is the observed entry of the inverse of the covariance matrix for the labels i and j and E_{ij} is the estimated entry of the inverse of the covariance matrix, then the mean of squared errors can be calculated by using 5.6. In this formula, k is the set of the names of all weak labels.

$$MSE = \frac{\sum_{i=1}^{N}[(O_{i0} - E_{i0})^2] + \sum_{j=1}^{N}[(O_{0j} - E_{0j})^2] + (O_{00} - E_{00})^2}{2N + 1} \tag{5.6}$$

The MSE has the advantage that it punishes outliers [6]. If an entry of the covariance matrix is filled with a value that differs a lot from the true value, it might influence the inverse of the covariance matrix and so the estimated dependency structure. Therefore, we want the filled entries of the covariance matrix to be as close as possible to their true values and introduce punishments when they are not.

As explained in 4.4, the assumptions that have been made to fill in the unobservable part of the covariance matrix might cause noise. In other words: the filled in part might be not perfectly correct. So, it is hard to compare the filled in unobservable part with the true unobservable part since there will be an expected difference due to the noise. Therefore, the MSE will only be used as an indication of how well the unobservable part of the covariance matrix is filled.

5.5 EXP1: Structure recovery using synthetic data

Parameter	**Value**
Structure (S)	$[1, 2, 1, 2]$
Equality rate (r)	0.9
Class balance (p)	0.62
Number of samples (N)	100.000
Runs (R)	20
Size of subset (nci)	4
Maximum threshold (t_m)	0.1

TABLE 5.2: Overview of the parameter setup of experiment 1

The goal of this work is to see to what extent Robust PCA can be used together with other methods to estimate the dependency structure among data. In this first experiment it will be tested if the approach that is described in Chapter 4 can find the dependency structure for a small dataset. The dataset contains of four cliques of size one or two. By adding cliques of size two we add some conditional dependency to the dataset. Otherwise, there would not be any nonzero entries in the inverse covariance matrix to classify. In the next experiments there will be more cliques of different sizes be added to the data to see to what extension the appraoch works successfully. Additionally, an equality rate of 0.9 is used. Since $0.9 \cdot 0.9 = 0.81$ this means that the weak labels that are in the same clique will be equal with the true label for 81%. The other 19% of each weak label will be a random binary value. On average, this means that all weak labels that are in the same clique will be equal for $81 + (0.5 \cdot 19) \approx 91\%$. The class balance is randomly set to 0.62 and we use a sample size of 100.000. Furthermore, the number of runs is set to 20 to generate a sufficient amount of estimations for each structure. The subset size in the first step will be four as explained and the maximum threshold that will be used is 0.1. The reason for this low threshold is that for our approach the threshold can be low. Therefore, we would also like to see the performance of the approach of Varma et al.[25] when a low threshold is used.

For this experiment the research question is:

To what extent can our approach be used to estimate the dependency structure for a dataset that consists of four cliques of size one and two?

Given the fact that at least 4 cliques are required to test the fit of the Bayesian Network, this should be a simple case that our approach should work for. A summary of the experimental parameters can be seen in Table 5.2.

We will do an inductive approach. In this experiment we start with a base case and then in the next experiments the data will be extended and changed to see to what extent the approach provides a result.

Success criteria

The success of this experiment depends on the results of the evaluation metrics that are discussed in 5.4. The goal of the method that is described in Chapter 4 was to make an estimation of the dependency structure by estimating conditional independency between weak labels. For this experiment a simple dataset that the method is expected to work for is generated. Namely, one true label and 4 cliques of size one or two. If that structure can be observed from the inverse of the covariance matrix that is retrieved after performing the steps described in Chapter 4, the experiment was a succes. In that case, it can be concluded that the method works for the simple dataset and the method can be tested on more advanced datasets.

5.6 EXP2: Data structure changes

Parameter	**Value**
Structure (S)	$[1,2,1,2,1]$, $[1,2,1,2,1,2]$, $[1,2,1,2,1,2,1]$..
Equality rate (r)	0.9
Class balance (p)	0.62
Number of samples (N)	100.000
Runs (R)	20
Size of subset (nci)	4
Maximum threshold (t_m)	0.1

TABLE 5.3: Overview of the parameter setup of experiment 2.1

Parameter	**Value**
Structure (S)	$[1,3,1,3]$, $[1,4,1,4]$, $[1,5,1,5]$..
Equality rate (r)	0.9
Class balance (p)	0.62
Number of samples (N)	100.000
Runs (R)	20
Size of subset (nci)	4
Maximum threshold (t_m)	0.1

TABLE 5.4: Overview of the parameter setup of experiment 2.2

In this experiment the effect of changing the dataset structure will be tested. In the previous experiment it was tested if the dependency structure can be estimated correctly when the dataset consists of only 4 cliques of size one or two. Assuming that the first experiment was a success, the experiment will now be repeated, but for datasets with more weak labels and larger cliques. So, the research question of this experiment is:

To what extent is our approach able to estimate the dependency structure correctly for a dataset that consists of a large amount weak labels or large cliques?

As explained in 3.1.2 the data consists of several cliques of different sizes. The structure of the dataset can be adjusted by changing the number of cliques or changing the size of those cliques. When the number of cliques or the size of the cliques

are increasing, the dependency structure will become more complicated and hence harder to correctly estimate with our approach. This is because the number of weak labels in the dataset will increase. The more weak labels our dataset contains, the more predictions should be made. Hence, our approach has to perform well to classify all entries correctly when there are many weak labels. The question is if the approach will still be able to make a good estimation of the graph dependency structure when the size of the dataset is enlarged.

The assumption is made that in a real situation there probably will not be more than 50 good weak labels. Knowing that a dataset can consist of at least 4 and at most 50 weak labels, makes that there is a large amount of possible datasets. A dataset can consist of 4 cliques of size one, but also of, for example, 26 cliques that have variable sizes as long as the total number of weak labels is not larger than 50. The total number of possible combinations can be calculated by using 5.7.

$$\#\text{ Possible datasets} = \sum_{i=4}^{50} \sum_{j=1}^{i} \binom{i-1}{j-1} \tag{5.7}$$

Solving this formula results in a number of possible datasets of 10^{15}. It is too expensive to test our approach on all these datasets, so in part 1 of this experiment it will be tested on a certain selection of datasets that are enlarged compared to the dataset of experiment 1.

To enlarge the size of the dataset, we start with a dataset that consists of 4 cliques of size one or two, which was used in experiment 1. Then, a clique of size two and a clique of size one will be constantly added alternately to the dataset until the dataset consists of a maximum of 50 cliques. By adding cliques of size 2 there will be conditional dependency added to the structure, so that the data not only consists of conditional independent weak labels. After each addition, the approach will be applied and the performance will be measured. In that way, the approach for a number of weak labels that is between 4 and 50 will be tested. An overview of the experimental parameters can be seen in Table 5.3.

An other way to enlarge the dataset is to enlarge the size of a clique. To do that, we again start with a dataset of 4 cliques of size 1. Then, the size of two of the cliques will be enlarged by one and the approach will be tested on the dataset and evaluated. This will be continued until the total number of weak labels is 50. In that way, there is a dataset that consists of 2 weak labels of size 1 and 2 cliques of a constantly growing size. In the end, the structure will consist of two cliques of size one and two cliques of size 24, which brings the total number of weak labels to 50. In that way, it can be tested how the size of a cliques influences the performance of our approach. An overview of the experimental parameters can be seen in Table 5.4.

Success criteria To decide whether this experiment has been a success, the evaluation metric that are discussed in 5.4 will be taken into account. Based on these metrics for each step of the approach it can be said whether the result after that step was a success. If the experiment fails it will be good to see after which step it fails and for which data structures it fails.

Parameter	Value
Structure (S)	$[1, 2, 3, 1]$
Equality rate (r)	0.9
Class balance (p)	$\frac{1}{2}, \frac{1}{10}, \frac{1}{100}, \frac{1}{1000}$
Number of samples (N)	100.000
Runs (R)	20
Size of subset (nci)	4
Maximum threshold (t_m)	0.1

TABLE 5.5: Overview of the parameter setup of experiment 3.

5.7 EXP3: Varying interaction strength

In this experiment it will be tested whether the degree of balanced data has influence on the estimation of the dependency structure.

To detect a transaction that might be fraudulent classifiers can be used. When data of transactions is used as training data to supervise these classifiers there is a very imbalanced dataset. This is because banks process a large number of transactions each year and only a small amount of them can be labeled as fraudulent. For classification problems where the data is unbalanced, like transaction classification, it will be nice to see how our approach performs. Therefore, it is interesting to see how the approach performs on a dataset with a low class balance. In this experiment, a class balance of $\frac{1}{2}$, $\frac{1}{10}$, $\frac{1}{100}$ and $\frac{1}{1000}$ will be tested. So, the research question of this experiment is as follows:

To what extent can our approach be used to estimate the dependency structure among weak labels when the dataset becomes unbalanced?

Success criteria The success of this experiment will be evaluated by using the evaluation metrics that are discussed in 5.4. The experiment will be a success if it is shown that for several different class balances the approach is still able to retrieve a good estimation of the dependency structure. If there are class balances that the approach is not able to find a good estimation of the class balance for, it will be nice to see for which of those parameters the approach exactly fails.

5.8 EXP4: Varying subset size

In this experiment the influence of the number of chosen weak labels to use for the fit of the Bayesian Network on the estimation of the dependency structure will be tested.

In the first step of the approach four weak labels that are the most likely to be conditionally independent are chosen. Then, in step 2, a Bayesian Network is fitted on the subset of these weak labels to estimate their conditional probabilities given the true class. As explained in Chapter 4.3, if less than four weak labels are selected then the goodness of the fit can not be tested. When at least four weak labels are chosen the goodness of the fit can be tested. Also, when the number of samples is

large the fitted Bayesian Network will probably give a good estimation of the conditional probabilities and the class balance. Since the Bayesian Network is able to find estimations for the conditional probabilities of conditional independent weak labels, we want to use as many weak labels for the fit as possible. After all, the more conditional probabilities that are estimated correctly, the more the filled covariance matrix will be similar with the true covariance matrix. Additionaly, the inverse covariance matrix will be more similar with the true inverse covariance matrix, so the dependency structure will be more visible.

The risk of using many weak labels for the fit is that we might use a weak label in the subset that is conditional dependent of an other weak label in the fit. It is assumed that all weak labels that are used for the fit are conditionally independent. If one of them is not, the solver will search for a non existing independency and will perform badly. When that happens, the fit will not give correct estimations of the conditional probabilities and the final performance of our approach might become low. So, the research question of this experiment is:

How can we choose the optimal number of weak labels in step 1 to form a subset?

As explained in Chapter 3 the values of the inverse covariance matrix are used to find the weak labels that are the most likely to be conditionally independent. The first two chosen weak labels are those that correspond to the smallest absolute value of the matrix. To find the next weak labels that are the most likely to be conditional independent, linear combinations are made by calculating the sum of the absolute columns of the weak labels that already have been selected. Of these linear combinations, the smallest value that corresponds to a weak label that is not already in the subset is chosen as the next most likely to be conditional independent weak label. Hence, after each weak label that has been selected the linear combination is enlarged. Therefore, the value that is chosen as the smallest absolute value to locate the next conditional independent label will increase when the number of chosen weak labels increases. For example: the smallest absolute value that is found by selecting the first two conditional independent weak labels will be close to zero. The second smallest value is found in the sum of the entries of the first two weak labels in the inverse covariance matrix and hence and hence will be larger than the first value. The difference between the first and the second values that are chosen is what we call the **Addition value**. The addition value represents the conditional dependency between the weak label that is added to the subset and the weak labels that already are in the subset. An example of the addition value can be seen in the Figures 4.1 and 4.2 in Chapter 4. In Figure 4.1 it can be seen that the first addition value equals 0.31. Then, in Figure 4.2 it can be seen that the second addition value is $1.33 - 0.31$, the difference between the first value and the second value. The expectation is that as long as we add weak labels to the subset that indeed are conditional independent the addition value will remain low. The question is: what value should the addition value take to give the impression that a weak label that is not conditional independent is added to the subset?

Success criteria To investigate this, a data structure of at least C cliques will be used. Each time we will select the $C + 1$ weak labels that are the most likely to be conditional independent and see what happens with the addition value when we select the last one. In other words: we observe what the addition value does when a weak label that is not conditional independent is added. For each structure, this will be repeated 50 times with different seeds. Then, the number of cliques, the

class balance and the dependency between weak labels in the cliques will be slightly adjusted to see how it influences the addition value. Based on these experiments we will make an estimation of a threshold that can indicate whether the latest chosen weak label can still be considered to be conditionally independent. We will use this threshold on full runs of the approach on three different data structures and compare the performance with the approach when only four weak labels have been chosen to form a subset. The experiment will be successful if we can find a threshold and the usage of it will result in a better performance than when only four weak labels are chosen to form a subset.

Results

This section describes the obtained results of the experiments of the previous chapter. The results will be discussed for each experiment individually. To do this, the explanation of the results consist of two parts:

- Results of a single run;
- Mean result of all runs.

To account for randomness in the experiments, each experiment has been ran with 20 different seeds. In that way, it can be seen how the approach performs on the long term when there are small differences in the dataset. To discuss the results of a single run, the focus will be on the performance for each step of the approach, clarified with plots. Then, the average results of the 20 runs will be discussed to evaluate the overall performance of the approach for certain parameters.

6.1 Experiment 1

The goal of this experiment was to show that our approach works for a dataset that contains a true label Y and two cliques of size one and two cliques of size two.

Single run In the first step of the approach, four conditional indepedent weak labels had to be identified from the observable inverse covariance matrix of all the weak labels. The weak labels that have been chosen (1a, 2b, 3a and 4b) were indeed conditional independent. In the second step, The Bayesian Network is used to estimate the conditional probabilities of the chosen weak labels given the true label Y. Then, the χ^2-test has been used to check whether the parameters of the fit and the true conditional probabilities can be assumed to be equal. The p-value of the test was 0.99, so the null hypothesis is not rejected. It can be assumed that the parameters of the fit are equal to the true conditional probabilities of the conditional independent weak labels, given Y.

In the third step, the fitted parameters from step 2 have been used to fill in the unobservable part of the covariance matrix. After that, the filled covariance matrix can be compared with the true covariance matrix by calculating the absolute difference between each of the corresponding entries of both matrices. 6.1.

It can be seen that the non-observable part of the differences is red for the most part. These red entries are the entries that correspond with the weak labels that have been chosen in the first step. The red color means that there are barely any differences between the non-observable parts of the true covariance matrix and the estimated covariance matrix. The mean squared error of the non-observable part is 4×10^{-5} which is very small. These small differences can also be noticed by looking at the unobservable part in the Figure. It can be seen that there are tiny differences,

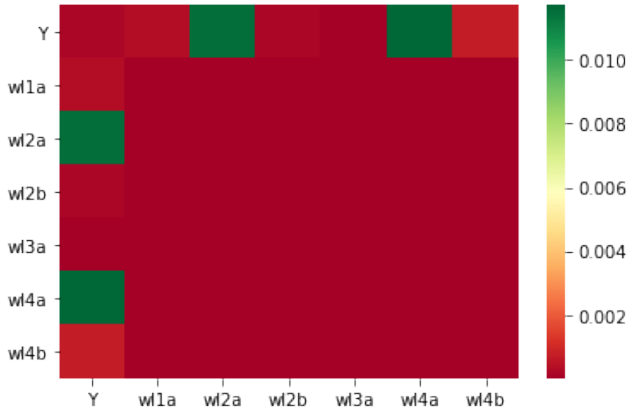


FIGURE 6.1: The absolute differences between the entries of the estimated covariance matrix and the true covariance matrix.

so it can be said that the estimated covariance matrix looks similar with the true covariance matrix.

In the fourth step, Robust PCA has been applied on the inverse of the estimated covariance matrix to filter out noise. Then, the retrieved matrix is combined with the matrix retrieved by the approach of Varma et al. Additionally, the Robust PCA algorithm of Varma et al. [25] has been used on the inverse of the observable part of the covariance matrix. These result in two estimations of the inverse covariance matrix. One according to our approach and one according to the approach of Varma et al[25]. These two inverse covariance matrices together with the true inverse covariance matrix can be seen in Figure 6.2. The true inverse covariance matrix is the inverse of the full covariance matrix of the generated data.

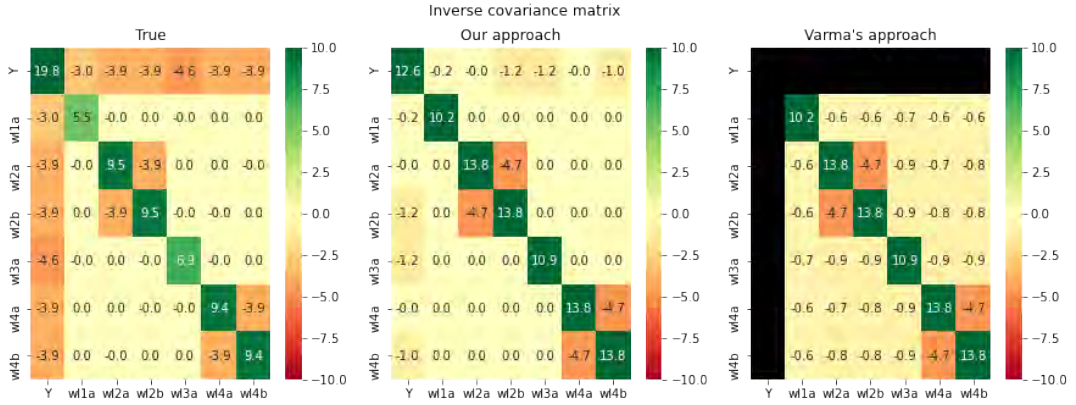


FIGURE 6.2: The true inverse covariance matrix and two estimations of it with different approaches.

It can be seen that in the inverse covariance matrix of the approach of Varma et al the row and column that correspond with Y are black since Y is not estimated in their approach. To do an end-to-end evaluation of our approach, we compared the performance of our approach with the performance of the approach of Varma et al[25]. To calculate the performance, we focussed on the observable part of the inverse covariance matrix. In this part, the values of the entries tell whether the corresponding weak labels can be assumed to be conditional independent or not.

Therefore, it is tested for each approach if they classified the entries of the true inverse covariance matrix that equal zero indeed as zero. The performance of both approaches is measured by calculating the accuracy. Our approach had an accuracy of 1 and the approach of Varma et al. had an accuracy of 0.28. It can be said that our approach classified the conditional independent weak labels well. This can also be seen by looking at the absolute differences between the true inverse covariance matrix and the two inverse covariance matrices of the approaches. They can be seen in Figure 6.3.

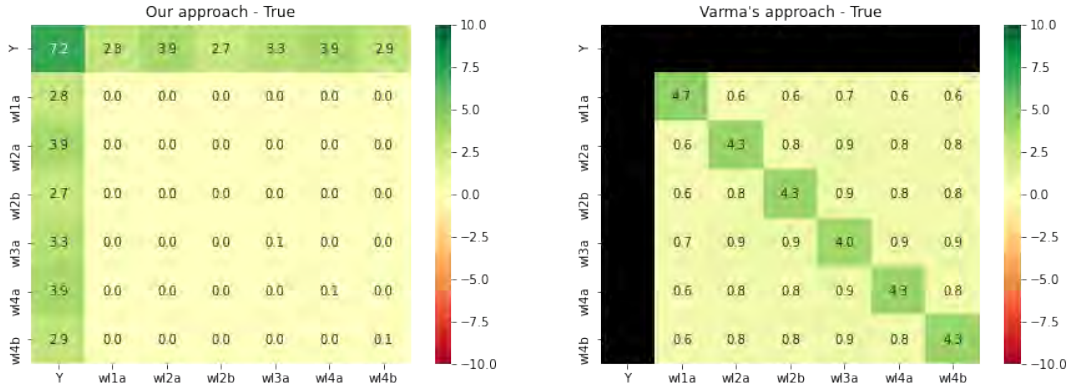


FIGURE 6.3: The absolute differences between the inverse covariance matrices of both approaches and the true inverse covariance matrix.

It can be seen that the observable part of the differences between our approach and the true inverse covariance matrix is totally yellow. Which means that there is hardly any difference. The inverse covariance matrix that has been retrieved by applying the method of Varma et al. contains a little more differences. Now it is interesting to see if this is also the case when 20 runs have been done.

All runs To include randomness in the experiments, each experiment has been ran with 20 different seeds. The mean results for each step can be seen in Table 6.2.

Step	**Performance metric**	**Mean result**
Step1: Create subset	Part of correctly chosen labels	1.0
Step2: fit Bayesian Network	P-value $\chi^2 < 0.05$	1.0
Step3: fill covariance matrix	MSE non-observed cov. matrix	0.000438
End-to-end	Accuracy our approach	0.91
End-to-end	Accuracy Varma	0.28

TABLE 6.1: Mean results end-to-end and per step after 20 runs.

Step	**Mean result**
Accuracy our approach	0.91
Accuracy Varma	0.28

TABLE 6.2: Mean results end-to-end and per step after 20 runs.

In Table 6.2 it can be seen that the first step of our approach was a success in each run. In each run the four weak labels that were identified as conditionally independent were indeed conditionally independent. In the second step, in all of the runs

the result of the χ^2-test said that the parameters of the fit and the true conditional probabilities could be assumed to be equal. As a result of that, it can be seen that the average mean squared error of the filled covariance matrix after step 3 is small. When we look at the end-to-end results by considering the accuracy, it can be seen that after each run our approach had a accuracy of 0.92. In other words: for most of the 20 datasets our approach estimated the dependency structure correctly. By using the approach of Varma et al. a mean accuracy of 0.28 over all runs was retrieved. It can be said that for a data structure that consists of 4 cliques of size one or two our approach is able to estimate the structure correctly.

6.2 Experiment 2

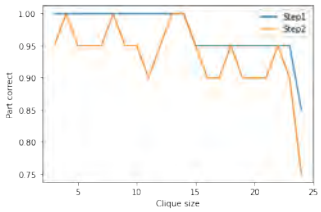
In this experiment the goal was to test how our approach would perform when the structure of the dataset changed. Changing in this case meant: the number of cliques changes or the size of the cliques changes. First, a single run of our approach with a data structure than consists of two cliques of size one and two cliques of size 24 will be worked out. Then, a single run of our approach with a data structure that consists of 50 weak labels that form cliques of size one or two will be discussed. Finally, the mean results of our approach when the number of cliques or the size of the cliques becomes large will be described.

Single run To give an example of the performance of our approach when the size of cliques becomes large, the results of a single run will be discussed. In the first step the four weak labels that have been chosen for the subset were all conditional independent. In the second step the χ^2-test returned a P-value of 0.99. Therefore, it can be stated that the fit of the Bayesian Network went well and the conditional probabilities have been estimated correctly. In the third step, the entries of the covariance matrix that were filled had a mean squared error of 0.0018. An additional plot of the differences between the true covariance matrix and our filled covariance matrix can be found in Appendix A. After the application of Robust PCA and the setup of the combination matrix, the performance of our approach could me measured. The accuracy of our approach was 0.57, so it classified a little more than half of the entries correctly. Before the Robust PCA algorithm was applied the inverse covariance matrix looked quite similar with the true inverse covariance matrix. However, after the usage of Robust PCA the retrieved inverse covariance matrix looked less similar with the true inverse covariance matrix. This can be seen graphically in Appendix B. So, in this single case the usage of Robust PCA did not seem to improve the performance of our approach.

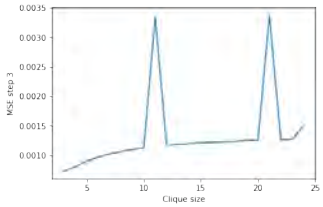
An other interesting situation is when the datset becomes large by containing a lot of cliques. Now, the results of a single run of our approach with a data structure that consists of 50 weak labels that form cliques of size one or two will be discussed. In the first step of our approach, the four weak labels that were chosen were indeed conditional independent. In the second step the χ^2-test returned a P-value of 0.95. Therefore, it can be stated that the fit of the Bayesian Network went well and the conditional probabilities have been estimated correctly. This could also be seen when the unobservable part of the covariance matrix is filled in the third step. The entries of the covariance matrix that were filled had a mean sqaured error of 0.0028. An additional plot of the differences between the true covariance matrix and our filled covariance matrix can be found in Appendix C. In the last step Robust PCA was applied and the combination matrix was set up. Then, the performance of our

approach could me measured. The accuracy of our approach was 0.99, so it classified almost all of the entries correctly. Graphical representations of the recovered structure can be found the Appendix D.

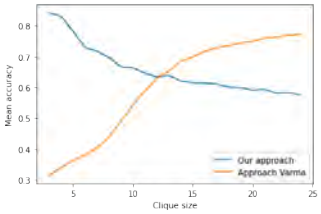
All runs The mean results of all runs will be described using plots that describe the development of the performance of our approach when the data structure changes. The results of when the clique sizes become large can be seen in Figure 6.4. The graphs represent the mean results when the size of two cliques remains one and the size of two other cliques enlarges.



(A) Development of the performance of the first two steps of our approach over the clique size.



(B) Development of the performance of the third step of our approach over the clique size.



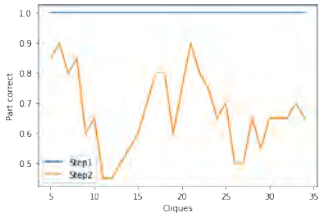
(C) Development of the accuracy over the clique size.

FIGURE 6.4: Performances when the size of two cliques is increasing.

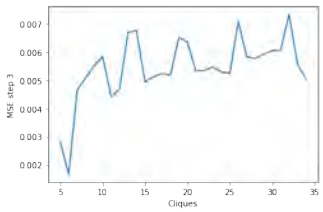
In Figure 6.4a it can be seen that the performance in step 1 and step 2 of our approach fluctuate and then decrease when the size of the clique becomes larger than 25. In the first step it was measured whether the chosen weak labels for the subset were indeed conditional independent. In the second step a χ^2-test was used to test whether the parameters of the fit were equal with the true conditional probabilities.

Also, in Figure 6.4b it can be seen that the average mean squared error in the third step of the approach slowly increases when the number of cliques increases. In Figure 6.4c it can be seen that the accuracy of our approach decreases when the clique size becomes large.

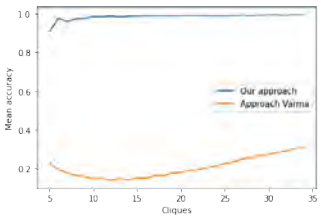
The results of when the number of cliques becomes large can be seen in Figure 6.5. The graphs represent the mean results when a structure that consists of two cliques of size one or two is repeatedly extended with a clique of size one or two.



(A) Development of the performance of the first two steps of our approach over the number of cliques.



(B) Development of the performance of the third step of our approach over the number of cliques.



(C) Development of the accuracy over the number of cliques.

FIGURE 6.5: Performances when the number of cliques is increasing.

In Figure 6.5a it can be seen that the performance of the first step equals 1.0 for all number of cliques. For all numbers of cliques there were four weak labels selected in the first step of our approach that were conditional independent. The performance of the second step varies between 0.9 and 0.4. In the third step the MSE slowly

increases when the number of cliques increases as can be seen in Figure 6.5b. This makes sense since we use a subset of only four weak labels. Therefore, when the number of weak labels in our dataset increases, more assumptions are needed to fill the non-observed part of the covariance matrix. The varying performance in the second step and the decreasing performance in the third step did not influence the final performance of our approach. The development of the final accuracy of our approach can be seen in Figure 6.5c. It can be seen that it remains stable around 1.0 regardless of the number of cliques.

It can be stated that when the size of the cliques becomes large, the performance of our approach decreases. On the other hand, our approach performs well regardless of the number of cliques.

6.3 Experiment 3

In this experiment the goal was to investigate the influence of the class balance on the performance of our approach of estimating the dependency structure. When banks want to classify transactions to detect fraudulent transactions, the data is probably imbalanced. For these cases it is interesting to see how our approach performs compared to the approach of Varma et al. when the data becomes increasingly imbalanced. The class balances that are used are $\frac{1}{2}$, $\frac{1}{10}$, $\frac{1}{100}$ and $\frac{1}{1000}$. First a single run for a class balance of $\frac{1}{1000}$ will be discussed. Then, the mean results for all class balances will be described.

Single run In the first step of the approach, four conditional indepedent weak labels were chosen from the observable inverse covariance matrix of all the weak labels to create a subset. The four chosen weak labels were indeed conditional independent, so step 1 was a success. In the second step, the Bayesian Network is fitted on the subset. Although the chosen four weak labels were indeed conditional independent, the parameters of the fitted Bayesian Network have not been qualified as significantly equal with the true conditional probabilities. The used χ^2-test returned a P-value of zero. Therefore, it would be interesting to see how well the covariance matrix was filled in step 3.

In the third step the unobservable part of the covariance matrix is filled. The MSE that is used to measure the performance of step 3 returns a value of 1.2×10^{-8} for the non-obersved part of the covariance matrix which is very small. We can state that despite of the error in the fit of the Bayesian Network the unseen part of the covariance matrix was still filled well. This can also be seen in Figure 6.6 where the differences between the filled covariance matrix and the true covariance matrix are plotted.

After the covariance matrix had been filled and inverted the Robust PCA and the inverse covariance matrix by the approach of Varma et al. have been used to create a matrix that can be used to read the estimated dependency structure from. These inverse covariance matrices can, together with the true covariance matrix, be seen in Figure 6.7.

In Figure 6.7 it can be seen that the dependency structure is perfectly readable from the estimation of our approach.

All runs The results of all runs can be seen in Table 6.3. As explained, for each class balance the approach is repeated 20 times with different seeds and the mean performances are calculated.

FIGURE 6.6: The absolute differences between the filled covariance matrix and the true covariance matrix.

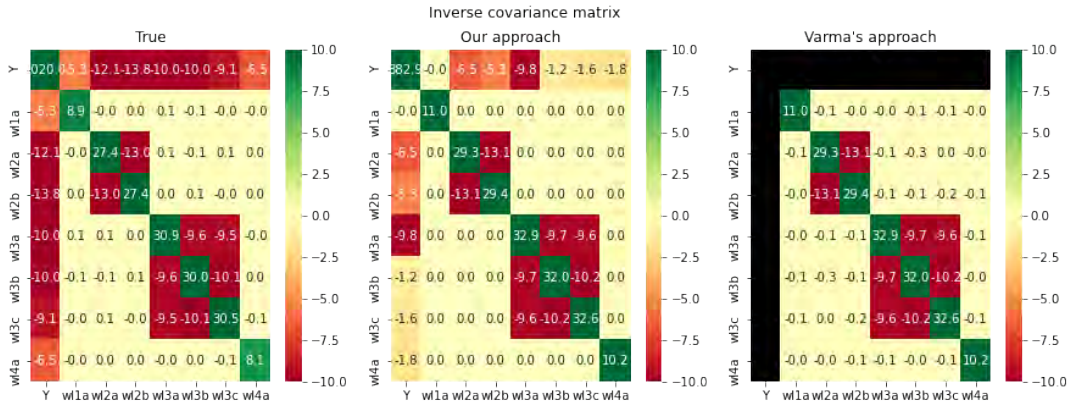


FIGURE 6.7: The true covariance matrix and the estimations according to our approach and the approach of Varma et al.

It can be seen that in the first step of the approach the four weak labels that were chosen were always conditional independent. However, in the second step the parameters of the fitted Bayesian Network were less accurate when the class balance dropped. When the data was balanced the parameters were always significantly equal with the true conditional probabilities according to the χ^2-test. But, when the class balance was $\frac{1}{100}$ and $\frac{1}{1000}$ the parameters there were no cases where the parameters were equal with the conditional probabilities. However, this did not influence the performance of step 3. It can be seen that the MSE in the third step is small for each class balance. After all, it can be seen that the performance of our approach increases when the data becomes more imbalanced. But, when the data is balanced our approach still performs well. According to these results it can be stated that our approach works for balanced and imbalanced datasets in the most cases. However, it must be said that the structure that this was tested on was simple: it consisted of only four cliques. It would be interesting to see how our approach performs when the data becomes imbalanced and at the same time more structure is added.

Class balance	Step1	Step2	Step3	Accuracy	Accuracy Varma
$\frac{1}{2}$	1.0	1.0	0.000493	0.90	0.28
$\frac{1}{10}$	1.0	0.5	0.000064	0.99	0.29
$\frac{1}{100}$	1.0	0.0	0.000007	0.98	0.32
$\frac{1}{1000}$	1.0	0.0	0.000611	0.99	0.75

TABLE 6.3: Mean results after 20 runs of each approach on different class balances.

6.4 Experiment 4

The goal of this experiment was to find a threshold for the addition value to chose the number of weak labels that should be chosen in step 1 of the approach to form a subset and evaluate the performance when it is applied. The addition value is the increase of the of the smallest value of the linear combination when a new weak label is added to the subset. The last addition value is when a weak label is added that is not conditional independent.

In the first part of this experiment we started with a dataset that consists of four cliques of size 1, 2, 3 and 4. Then, we continuously added cliques to the structure to see the influence on the addition value. For all these structures, the algorithm of step 1 that was discussed in Chapter 4 created a subset of weak labels that were indeed conditional independent. The results for all structures can be seen in Figure 6.8.

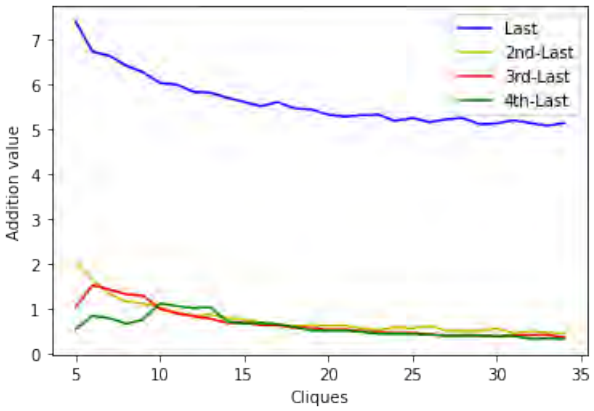


FIGURE 6.8: The average behaviour of the last four addition values against the number of cliques.

In Figure 6.8 it can be seen that when the second-last, third-last and fourth-last weak labels that are the most likely to be conditional independent are added to the subset, the addition value does not change a lot. The yellow, red and green lines remain close to each other. However, when the last weak label is added, which is not conditional independent, the addition value shoots up. When the number of cliques increases, the blue line slowly converges to five. This would indicate that 5 could be a nice threshold for the addition value: when the addition value is smaller than 5, the chosen weak label can be assumed to be conditional independent and added to the subset.

In the second part of this experiment the influence of the class balance on the addition value is observed. In this part, again a dataset that consists of four cliques

of size 1, 2, 3 and 4 has been used. We start with a class balance of 0.5 and slightly reduce it to see how it influences the addition value. The results can be seen in Figure 6.9.

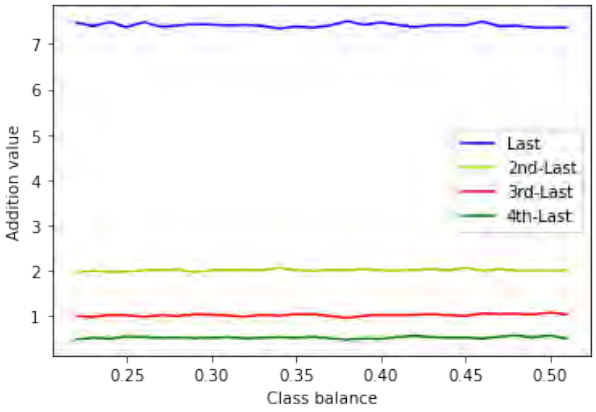


FIGURE 6.9: The average behaviour of the last four addition values against the class balance.

In this Figure it can be seen that again the addition value increases a lot when the last weak label, from which we know that it is not conditional independent, is added to the subset. This addition value remains constant just above 7, so 5 can still be used as a threshold for the addition value.

In the third part of this experiment the influence of the dependency between the weak labels in a clique on the addition value is researched. The expectation is that when the weak labels in a clique are less dependent on each other, the harder it is to make a separation between the weak labels in a clique and the other labels in the dataset. The results of this part can be seen in Figure 6.10.

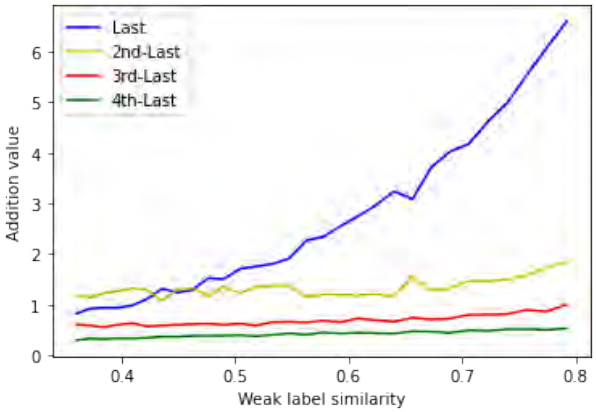


FIGURE 6.10: The average behaviour of the last four addition values against the similarity between weak labels in a clique.

In Figure 6.10 the similarity represents for which part of the weak labels in a clique are equal with each other. The blue line represents the addition value of the first weak label that is added to the subset that is not conditional independent. It can be seen that when the similarity increases it is easier to separate this last weak label from the rest. When the similarity is low the question is to what extent the algorithm that we use in step 1 of our approach is able to find four conditional independent variables. This is show in Figure 6.11.

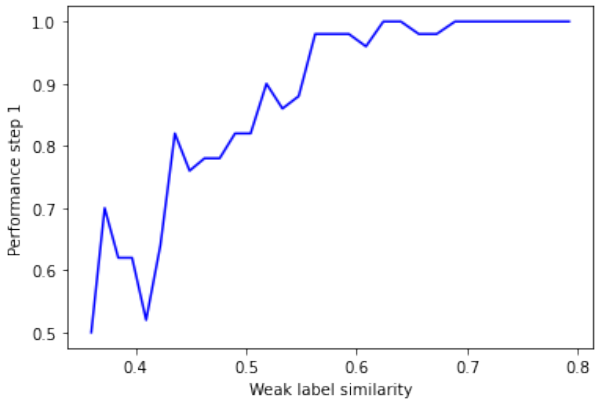


FIGURE 6.11: Performance of step 1 of our approach against the similarity among weak labels in a clique.

In 6.11 the vertical axis represents the fraction of times that the algorithm that has been used in step 1 of our approach to find four conditional independent weak labels performed well. A good performance means that the four weak labels that were selected for the subset were indeed conditional independent. Each fraction is calculated over 50 runs with different seeds. It can be seen that when the similarity is smaller than 0.6 the algorithm is having trouble with finding four conditional independent weak labels. In other words: when the weak labels in a clique are equal for less than 60% of their elements they start to be considered as conditional independent instead of conditional dependent.

One of the assumptions that are made is that the weak labels that form a clique are conditional dependent of each other. We assume that the similarity between weak labels in a clique is at least 0.6. If we then have a look at Figure 6.10 it can be seen that when the similarity is 0.6 the addition value of the added conditional dependent label to the clique is around 3. This is lower than the addition value of 5 that was found earlier in the section, so based on the first three parts of this experiment it can be said that 3 would be a good threshold for the addition value. In other words: when a weak label is added to the subset and the difference between the smallest absolute value of the current linear combination and the smallest absolute value of the previous linear combination is equal or larger than 3 the weak label should not be added to the subset. In the next part the performance of the approach when this threshold is applied will be compared to our approach where only four weak labels are chosen.

6.4.1 Threshold test

To test the influence of selecting more than four weak labels in the first step of the approach, three different structures have been used:

- structure 1: [1,2,3,1,4]
- structure 2: [1,2,3,1,4,2,5]
- structure 3: [1,2,3,1,2,2,1,1,2,1]

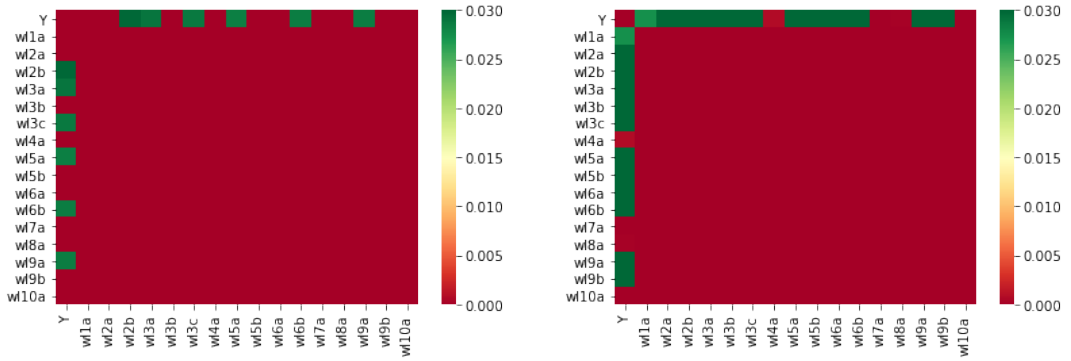
It can be seen that the number of cliques in each structure is increasing. Since the usual approach uses only four weak labels in the subset this adjusted approach

might use more weak labels in the subset. Therefore, it will be interesting to see the performance when the number of cliques increases. The expectation is that if more conditional independent weak labels are added to the subset, the the Bayesian Network fit on that subset will estimate the conditional probabilities of the weak labels well. If that is the case, the probabilities can be used to fill the unobservable part of the covariance matrix so that it will look more like the true covariance matrix. Then, the precision matrix will also be more similar with the true precision matrix. The results of the tests on the three structures can be seen in Table 6.4.

Structure	**4 weak labels**	**4+ weak labels**
[1,2,3,1,4]	0.98	0.98
[1,2,3,1,4,2,5]	0.97	0.98
[1,2,3,1,2,2,1,1,2,1]	0.96	0.99

TABLE 6.4: Comparison of the precision of the approach with 4 weak labels and the approach with +4 weak labels in the subset on three structures.

In Table 6.4 it can be seen that when the number of cliques increases the performance of the approach where only four weak labels are chosen for the subset decreases. Simultaneously , it can be seen that the performance of the approach where more than four weak labels are chosen increases. For both approaches a sample of the differences between the filled covariance matrix and the true covariance matrix can be seen in Figure 6.12.



(A) More than 4 weak labels chosen.

(B) 4 weak labels chosen.

FIGURE 6.12: Differences between the filled covariance matrix and the true covariance matrix.

In Figure 6.12 it can be seen that when a subset of four weak labels is made the difference between the filled covariance matrix and the true matrix is larger than when more than four weak labels are used for the subset. Therefore, it can be said that for large datasets it is preferable to use the addition value threshold to find a maximum amount of conditional independent weak labels for the subset.

Conclusion

This section answers the research question presented in the introduction and the sub questions of the experiments that were presented in Chapter 5. Each sub question is answered individually, so it contributes to answering the main research question at the end of this chapter.

To what extent can our approach be used to estimate the dependency structure for a dataset that consists of four cliques of size one and two? The answer to this questions reveals whether our approach is able to estimate the dependency structure correctly for a simple small dataset that consists of two cliques of size one and two cliques of size two. The results of the experiment showed that on average the algorithm in step 1 for finding four weak labels that are conditionally independent performed well. The same holds for the performance of step 2. In the third step, the average mean squared error is small. The final accuracy of our approach was 0.91. So, on average more than 90% of the dependency structure was classified correctly. It can be stated that our approach is able to decently estimate the dependency structure for a dataset that consists of two cliques of size one and two cliques of size two.

To what extent is our approach able to estimate the dependency structure correctly for a dataset that consists of a large amount weak labels or large cliques?
To see whether our approach will be applicable for different datasets, it was interesting to see how it would perform when the data structure changed. Considering a maximum number of 50 weak labels, we repeatedly extended the dataset with a clique of size one or two. An other way was to enlarge the size of two cliques repeatedly by one. After each change of the structure the approach was applied again and the results were measured. The results of the experiment where the clique size is increased showed that when the clique size becomes large the performance of the first two steps of our approach slowly decreases. The same holds for the third step and it could also be seen in the final performance of the approach. When the size of the cliques increase, the average accuracy of our approach slowly decreases and then stabilizes around 0.55. We can conclude that when the size of cliques becomes large, the performance of our approach decreases.

The results of the experiment where the number of cliques is slowly increased showed that the mean performance of the first step remains stable at 1.0, while the mean performance of the second step varies between 0.4 and 0.9. The performance of the third step slowly decreases as expected when the number of cliques increases. The mean final performance of our approach, the accuracy, remains stable around 1.0 regardless of the number of cliques in the dataset.

After the experiments we can conclude that our approach performs well regardless of the amount of cliques that are in the dataset. On the other hand, the performance of our approach becomes smaller when the size of the cliques increases.

To what extent can our approach be used to estimate the dependency structure among weak labels when the dataset becomes unbalanced?
To see whether our approach will be applicable in practice, it was interesting to see how our approach performs when the data becomes imbalanced. To test this, we tested our approach on datasets that had four different class balances: $\frac{1}{2}$, $\frac{1}{10}$, $\frac{1}{100}$, $\frac{1}{1000}$. The performance of the first step of our approach remained stable around 1.0. This indicates that regardless of the class balance, the algorithm of finding four weak labels that are the most likely to be conditional independent always finds four labels that indeed are conditional independent. However, in the second step of the approach the performance decreases when the class balance becomes smaller. The performance of the third step increases when the class balance becomes smaller, although the average mean squared error suddenly becomes large when the class balance is $\frac{1}{1000}$. The final accuracy of our approach increases when the data becomes more imbalanced and stabilizes around 1.0.

Based on the performed experiments it can be stated that our approach is able to estimate the dependency structure well for datasets that are imbalanced. The performance remains stable, despite of the decreasing performance of the second step of our approach. However, the data structure that this was tested on was simple. It would be interesting to see how our approach performs when the data becomes imbalanced and at the same time more structure is added.

How can we choose the optimal number of weak labels in step 1 to form a subset?
In the first step of our approach a certain number of weak labels is chosen to form a subset. These weak labels are considered as the weak labels that are the most likely to be conditionally independent. The labels are chosen based on their corresponding values in the observable inverse covariance matrix. The subset is then used to fit a Bayesian Network on, to use its parameters to fill the unobservable part of the covariance matrix. The expectation is that the more weak labels that are in the subset, the more accurate the covariance matrix will be filled in the end. As long as the weak labels in the subset are all conditionally independent. When they are not, the Bayesian Network assumes conditional independency while in fact they are not. The consequence is that the fit will not be done well and the covariance matrix will be filled badly.

In the process of adding weak labels to the subset we introduced a threshold for the addition value. The addition value represents the conditional dependency between the weak label that is added and all the weak labels that are already in the subset. Several tests showed that when conditional independent weak labels are added to the subset, the addition value remains low, which is an indication of conditional independency. However, when a weak label that is conditional dependent is added the addition value strongly increases. To decide whether a weak label should be added to the subset a threshold had to be set. When the addition value was larger than the threshold, the corresponding weak label was not added. To find the best threshold several tests have been done by looking at the behaviour of the addition value when the number of cliques, the class balance or the dependency between weak labels changed. Based on the tests, it can be stated that the threshold for the addition value should be a value of which we can be sure that it will make our algorithm add conditional independent weak labels to that subset, but no conditional dependent weak labels. According to the tests, this value should be 3. Tests with this threshold showed that when the number of cliques becomes large the performance of our approach increases while the performance of the approach that uses only four

weak labels decreases. It can be stated that a threshold of 3 can be used when the dataset consists of a large number of weak labels.

Based on the sub questions, the main research question will be answered below. The main research question was:

To what extent can Robust PCA be applied and combined with other methods to find the dependency structure among weak labels?

Based on the performed experiments we can conclude that a combination of Robust PCA and the usage of a Bayesian Network for estimating the unobserved part of the covariance matrix can be used to estimate the dependency structure among weak labels. When the number of cliques in a dataset increases or when the dataset becomes imbalanced the approach can still be used to estimate the dependency structure correctly in most cases. However, when the size of the cliques becomes large, the performance of our approach decreases. The results have shown that a subset of a variable size, that depends on the detected dependency between weak labels in the inverse covariance matrix, might improve the performance of our approach.

Discussion

The aim of this master thesis was to investigate whether Robust PCA could be combined with other methods to estimate the dependency structure among weak labels. This research has experienced some limitations towards achieving better results. In this chapter, these limitations are discussed.

Subset size in step 1
The results of the fourth experiment showed that the number of chosen weak labels in the first step of our approach might influence the final estimation that is made. In all the other experiments that have been done we have chosen a number of four conditional independent weak labels in the first step of our approach to form a subset. Reason for this was that a subset of size four was the minimal size to be able to test the fit of a Bayesian Network on it. However, the results of the fourth experiment showed that using a variable subset size might have improved the achieved results. Therefore, it might be interesting to redo the experiments with a variable number of chosen weak labels in the first step and see if it improves the quality of the final estimation of the dependency structure.

Chi-squared test in step 2
In the second step of our approach a χ^2-test has been used to test whether the parameters of the fitted Bayesian Network on our subset could be considered as equal with the true conditional probabilities of the corresponding weak labels. In many cases the test returned a P-value that was smaller than the stated confidence level, so the null-hypothesis was rejected. This meant that the parameters of the fit could not be considered to be equal with the true conditional probabilities. However, in the third step of our approach the parameters of the fit were used to fill the unobservable part of the covariance matrix. In many cases where the χ^2-test returned a result that did not indicate equality, the fill of the coviarnace matrix went well, according to the performance of step 3. In those cases, the parameters of the fit were apparently not that different from the true conditional probabilities, but they were simply not close enough to make the χ^2-test return a higher P-value. Therefore, an other method for evaluating the performance of the second step of our approach might give a better representation of the true performance in that step.

Use RPCA or not
To make an estimation of the dependency structure we used Robust PCA to filter out noise from our retrieved inverted filled covariance matrix. However, in some cases, for example experiment two, the inverse covariance matrix represented the dependency structure well before the Robust PCA algorithm has been applied. So, in some cases Robust PCA does not seem to necessarily be applied to get an inverse covariance matrix that can be used to read the dependency structure from. In fact, the usage of the Robust PCA algorithm might in some cases even make the result worse. This could also be seen in the graphical representation of the results of experiment 2. Therefore, it is important to consider whether it would be useful to apply Robust PCA on the inverted filled covariance matrix or not.

Threshold inverse covariance matrix

After the application of Robust PCA on the inverted filled covariance matrix we determined of each matrix entry whether it could be considered as zero or nonzero. As explained, we used a maximum threshold t_m to do this. In each experiment we initialized this threshold with a value of 0.1. In this way we did not consider any threshold that was larger than 0.1. an entry of the inverse covariance matrix should have a very low value to be considered as zero. There might be other values for t_m that would improve the estimated dependency structure by our approach.

Combination matrix

In the final part of our approach we created a combination matrix by using a combination of the inverse covariance matrix that we retrieved and the inverse covariance matrix retrieved by the approach of Varma et al.[25] We used the retrieved inverse covariance matrix of the approach of Varma et al. to make a combination with our own estimation. Therefore, it might be dubious the compare our approach with the approach of Varma et al. since we use the solution of Varma et al. for our own solution. However, the final goal of this work was to make an estimation of the dependency structure among weak labels. By making a combination matrix we showed that in many cases our approach can be used to find a good estimation of the dependency structure.

Covariance matrix when cliques are large

This appendix provides information about the third step of a single run of our approach on a dataset that consists of two cliques of size one and two cliques of size 24. In the third step we filled the unobservable part of the covariance matrix and compared our filled covariance matrix with the true covariance matrix. The absolute differences can be seen in the plot.

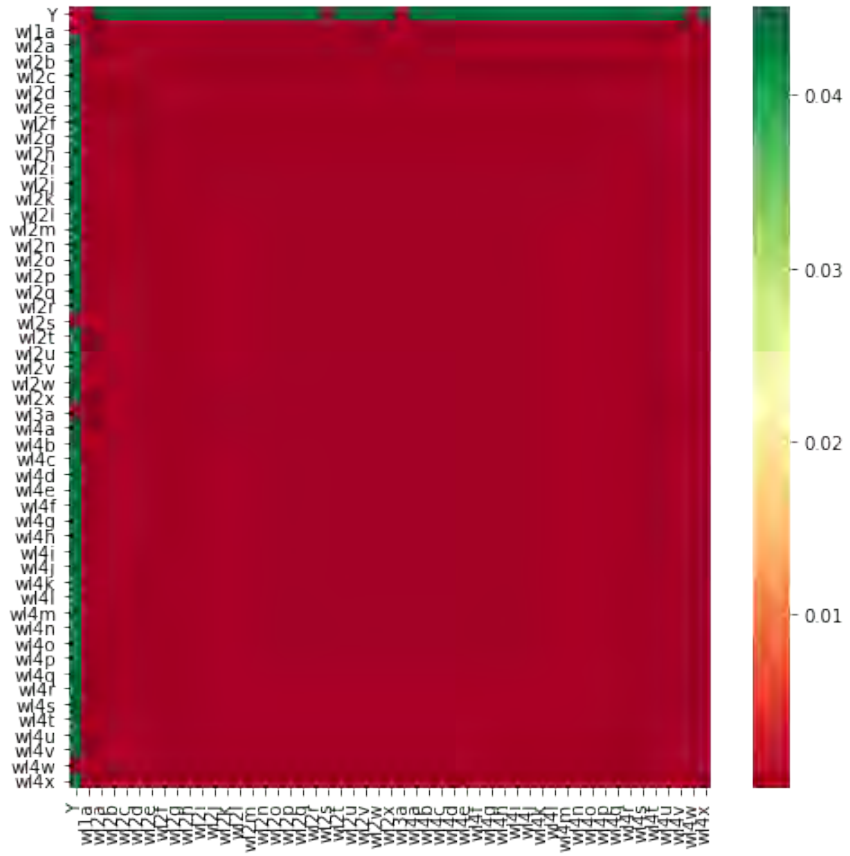


FIGURE A.1: The absolute differences between the true covariance matrices and our filled covariance matrix for a structure of [1,24,1,24].

Robust PCA effect when cliques are large

This appendix provides extra graphical information about the fourth step of a single run of our approach on a dataset that consists of two cliques of size one and two cliques of size 24. In the fourth step we applied Robust PCA on the inverted filled covariance matrix to filter out statistical noise. In Figure B.1 the inverted filled covariance matrix before the usage of Robust PCA is compared with the true inverse covariance matrix and the retrieved inverse covariance matrix by the approach of Varma et al. In Figure B.2 the inverted filled covariance matrix after the usage of Robust PCA is compared with the true inverse covariance matrix and the retrieved inverse covariance matrix by the approach of Varma et al.

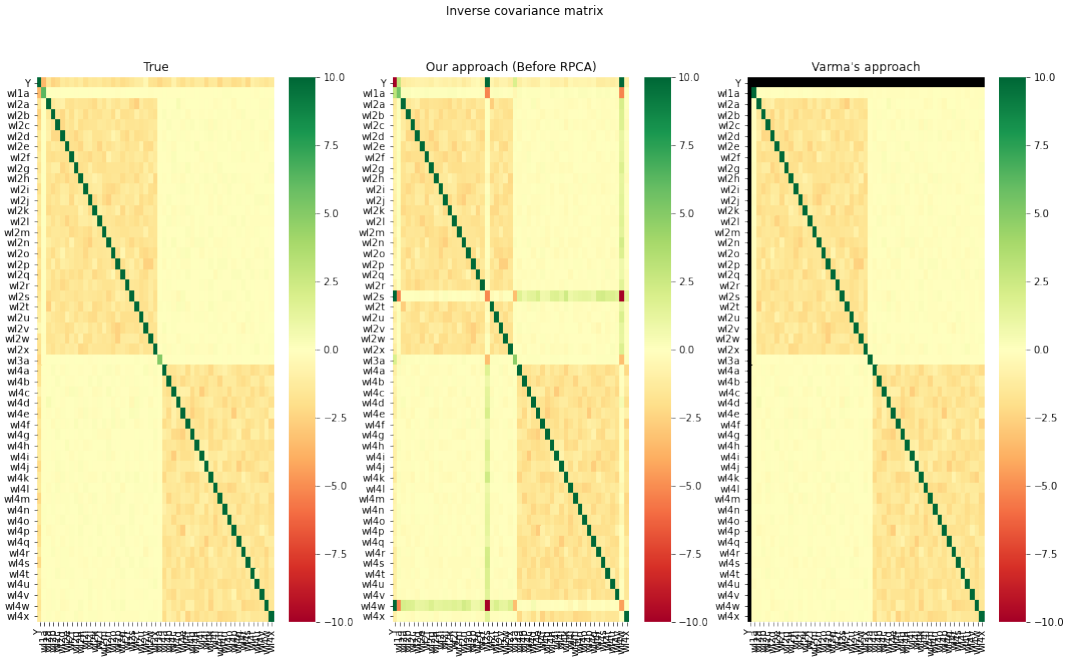


FIGURE B.1: The inverse covariance matrix retrieved by our approach before Robust PCA compared with the true inverse covariance matrix and the one retrieved by the approach of Varma et al.

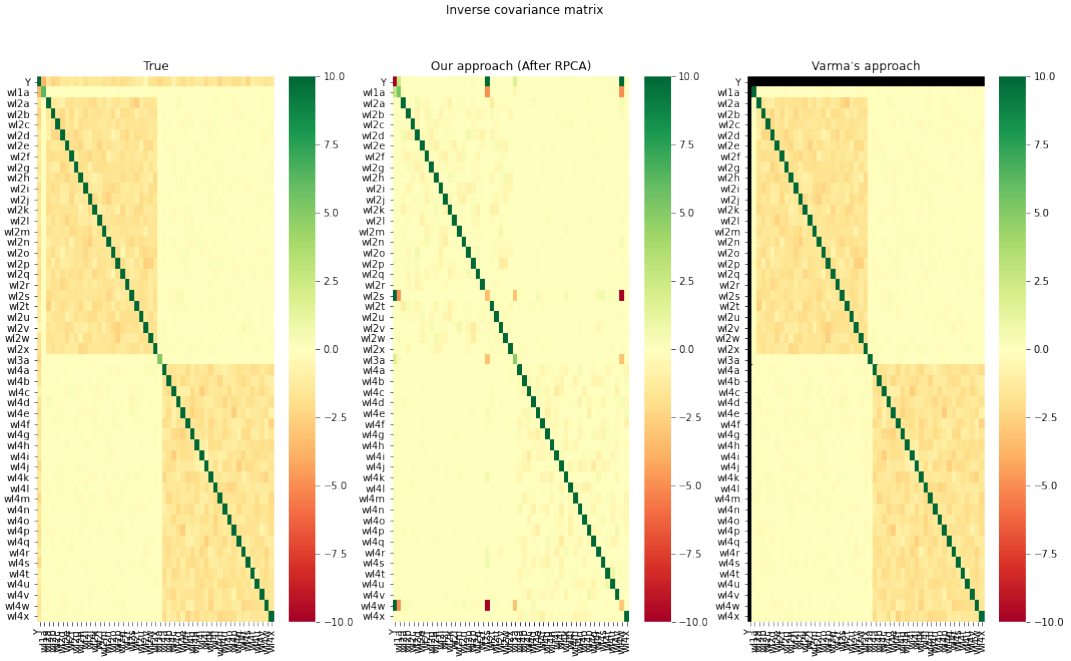


FIGURE B.2: The inverse covariance matrix retrieved by our approach compared with the true inverse covariance matrix and the one retrieved by the approach of Varma et al.

Covariance matrix for large number of cliques

This appendix provides information about the third step of a single run of our approach on a dataset that consists of 50 weak labels that are divided in multiple cliques of size one or two. In the third step we filled the unobservable part of the covariance matrix and compared our filled covariance matrix with the true covariance matrix. The absolute differences can be seen in the plot.

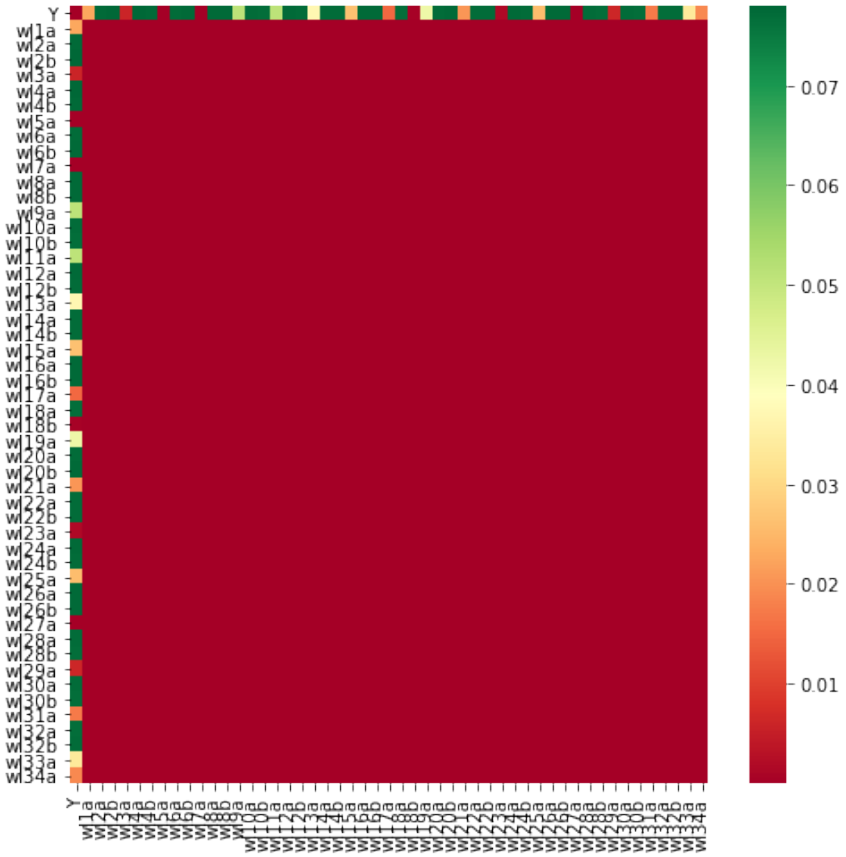


FIGURE C.1: The absolute differences between the true covariance matrices and our filled covariance matrix.

Robust PCA effect for large number of cliques

This appendix provides extra graphical information about the fourth step of a single run of our approach on a dataset that consists 50 weak labels that are divided in cliques of size one or two. In the fourth step we applied Robust PCA on the inverted filled covariance matrix to filter out statistical noise. In Figure D.1 the retrieved inverse covariance matrix by our approach is compared with the true inverse covariance matrix and the retrieved inverse covariance matrix by the approach of Varma et al.

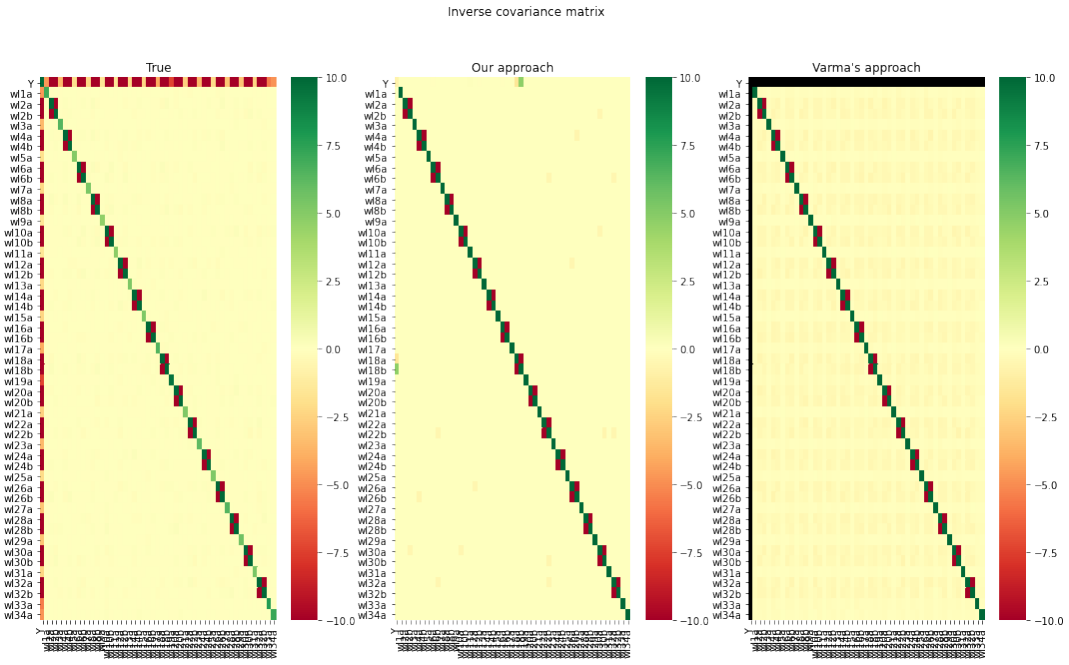


FIGURE D.1: The inverse covariance matrix retrieved by our approach compared with the true inverse covariance matrix and the one retrieved by the approach of Varma et al.

Bibliography

[1] Piti Ongmongkolkul et al. *Iminuit*. https://iminuit.readthedocs.io/en/v1.5.4/reference.html#iminuit.Minuit. 2020.

[2] Dr. M. Baak. *Weak labels*. https://github.com/mbaak/probfit/blob/weak_labels/probfit/weaklabels.py. 2021.

[3] Adi Ben-Israel and Thomas N.E. Greville. *Generalized Inverses: Theory and Applications*. Springer-Verlag New York, 2001.

[4] Dennis S. Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas (Second Edition)*. Princeton University Press, 2009. ISBN: 9780691140391. URL: http://www.jstor.org/stable/j.ctt7t833.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York, NY: Springer Science+Business Media, LLC, 2006.

[6] Alexei Botchkarev. "A New Typology Design of Performance Metrics to Measure Errors in Machine Learning Regression Algorithms". In: *Interdisciplinary Journal of Information, Knowledge, and Management* 14 (2019), 045–076. ISSN: 1555-1237. DOI: 10.28945/4184. URL: http://dx.doi.org/10.28945/4184.

[7] Emmanuel J. Candes et al. *Robust Principal Component Analysis?* 2009. arXiv: 0912.3599 [cs.IT].

[8] Venkat Chandrasekaran et al. "Rank-Sparsity Incoherence for Matrix Decomposition". In: *SIAM Journal on Optimization* 21.2 (Apr. 2011), 572–596. ISSN: 1095-7189. DOI: 10.1137/090761793. URL: http://dx.doi.org/10.1137/090761793.

[9] "Covariance and linear independence". In: *Understanding Regression Analysis*. Boston, MA: Springer US, 1997, pp. 31–35. ISBN: 978-0-585-25657-3. DOI: 10.1007/978-0-585-25657-3_7. URL: https://doi.org/10.1007/978-0-585-25657-3_7.

[10] B.N. Datta. *Numerical Linear Algebra and Applications, Second Edition*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 2010. ISBN: 9780898716856. URL: https://books.google.nl/books?id=1V9PbyYGZIIC.

[11] A. P. Dawid. "Conditional Independence in Statistical Theory". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 41.1 (1979), pp. 1–31. ISSN: 00359246. URL: http://www.jstor.org/stable/2984718.

[12] David Freedman, Robert Pisani, and Roger Purves. *Graphical Models in Applied Multivariate Statistics*. W. W. Norton & Company, 2007.

[13] Karl Pearson F.R.S. "X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 50.302 (1900), pp. 157–175. DOI: 10.1080/14786440009463897.

[14] Daniel Y. Fu et al. *Fast and Three-rious: Speeding Up Weak Supervision with Triplet Methods*. 2020. arXiv: 2002.11955 [stat.ML].

[15] Khalid Iqbal et al. "An Overview of Bayesian Network Applications in Uncertain Domains". In: *International Journal of Computer Theory and Engineering* 7 (2015), pp. 416–427.

[16] Oleg Kuybeda et al. "A collaborative framework for 3D alignment and classification of heterogeneous subvolumes in cryo-electron tomography". In: *Journal of Structural Biology* 181.2 (Feb. 2013), 116–127. ISSN: 1047-8477. DOI: 10.1016/j.jsb.2012.10.010. URL: http://dx.doi.org/10.1016/j.jsb.2012.10.010.

[17] Po-Ling Loh and Martin J. Wainwright. "Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses". In: *The Annals of Statistics* 41.6 (Dec. 2013). ISSN: 0090-5364. DOI: 10.1214/13-aos1162. URL: http://dx.doi.org/10.1214/13-AOS1162.

[18] Matt Mcgee. *Facebook: 3.2 Billion Likes & Comments Every Day*. Aug. 2012. URL: https://martech.org/facebook-3-2-billion-likes-comments-every-day/.

[19] Betaalvereniging Nederland. *Facts and figures on the Dutch payment system in 2020*. URL: https://factsheet.betaalvereniging.nl/en/.

[20] Maurice G Kendall; Alan Stuart; J K Ord; Steven F Arnold; Anthony O'Hagan. *Kendall's advanced theory of statistics.* London : Edward Arnold ; New York : Halsted Press, 1994.

[21] David L. Olson and Dursun Delen. *Advanced Data Mining Techniques*. 1st. Springer Publishing Company, Incorporated, 2008. ISBN: 3540769161.

[22] Edmund R. Peay. "Hierarchical Clique Structures". In: *Sociometry* 37.1 (1974), pp. 54–65. ISSN: 00380431. URL: http://www.jstor.org/stable/2786466.

[23] Alexander Ratner et al. "Snorkel". In: *Proceedings of the VLDB Endowment* 11.3 (Nov. 2017), 269–282. ISSN: 2150-8097. DOI: 10.14778/3157794.3157797. URL: http://dx.doi.org/10.14778/3157794.3157797.

[24] Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. "Who Learns Better Bayesian Network Structures: Constraint-Based, Score-based or Hybrid Algorithms?" In: *Proceedings of the Ninth International Conference on Probabilistic Graphical Models*. Ed. by Václav Kratochvíl and Milan Studený. Vol. 72. Proceedings of Machine Learning Research. Prague, Czech Republic: PMLR, Sept. 2018, pp. 416–427. URL: http://proceedings.mlr.press/v72/scutari18a.html.

[25] Paroma Varma et al. *Learning Dependency Structures for Weak Supervision Models*. 2019. arXiv: 1903.05844 [stat.ML].

[26] D. Wackerly, W. Mendenhall, and R.L. Scheaffer. *Mathematical Statistics with Applications*. Cengage Learning, 2014. ISBN: 9781111798789. URL: https://books.google.nl/books?id=lTgGAAAAQBAJ.

[27] J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley, 1991.

[28] Changjing Wu et al. "Graphical model selection with latent variables". In: *Electronic Journal of Statistics* 11.2 (2017), pp. 3485 –3521. DOI: 10.1214/17-EJS1331. URL: https://doi.org/10.1214/17-EJS1331.

[29] Tao Wu et al. "An efficient sparse-dense matrix multiplication on a multicore system". In: Oct. 2017, pp. 1880–1883. DOI: 10.1109/ICCT.2017.8359956.

[30] Xiaoming Yuan and Junfeng Yang. "Sparse and low rank matrix decomposition via alternating direction method". In: *Pacific Journal of Optimization* 9 (Jan. 2009).

[31] Leila Zahedi et al. *Search Algorithms for Automated Hyper-Parameter Tuning*. 2021. arXiv: 2104.14677 [cs.LG].