HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering
Laboratory of Computer and Information Science

Mikko Multanen

OUTLIER DETECTION IN CELLULAR NETWORK DATA EXPLORATION

Master's Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.

Espoo, 2nd May 2007

Supervisor: Professor Olli Simula
Instructor: Kimmo Raivio, D.Sc. (Tech.)

HELSINKI UNIVERSITY OF TECHNOLOGY Department of Computer Science and Engineering	ABSTRACT OF MASTER'S THESIS
Author Mikko Multanen	Date 2nd May 2007 Pages 63
Title of thesis Outlier detection in cellular network data exploration	
Professorship Computer and information science	Professorship Code T-115
Supervisor Prof. Olli Simula	
Instructor Kimmo Raivio, D.Sc. (Tech.)	

A cellular network like a GSM network contains many base stations to cover a large geographical area. The area covered by one base station can be seen as a cell in the network. Regardless of the location of a user the network should be able to provide the services it was build for. So each cell should have enough resources to succeed in meeting the demand. Arising from this the analysis of the network can to a certain extent be divided into the analysis of separate cells.

In this thesis, unsupervised methods for outlier detection are examined with a thought of applying them on the analysis of cellular network data. Both the case where the outliers are sought inside one cell and the case where some cells are identified as outliers are considered.

Some statistical and neural methods that are suitable to the task are tested with synthetic and real data. The real cell-specific data is from a GSM network. In the case of the GSM network data the two different outlier detection tasks are considered. As little as possible a prior knowledge of the network is used in the analysis as the methods are that way more portable into use with other data. The results of the methods are analyzed comparing them to each other on the general level but the final conclusions regarding the analysis of the network are left for network experts to draw.

Keywords

outlier detection, k-nearest neighbors, neural network, Parzen windows, Gaussian mixture model, distance measure, cellular network

TEKNILLINEN KORKEAKOULU Tietotekniikan osasto	DIPLOMITYÖN TIIVISTELMÄ
Tekijä Mikko Multanen	Päiväys 2.5.2007 Sivumäärä 63
Työn nimi Poikkeavuuksien etsintä matkapuhelinverkon datasta	
Professuuri Informaatiotekniikka	Koodi T-115
Työn valvoja Prof. Olli Simula	
Työn ohjaaja Kimmo Raivio, TkT	

Matkapuhelinverkot kuten GSM-verkko rakentuvat erillisistä soluista. Yksi solu verkossa on yhden tukiaseman kantoalue. Riippumatta siitä minkä tukiaseman alueella käyttäjä on pitäisi verkon pystyä huolehtimaan käyttäjän vaatimista palveluista. Jokaisella tukiasemalla pitäisi siis olla tarpeeksi resursseja käytössään palvellakseen oman alueensa asiakkaita. Täten myös verkon analysointi voidaan jossain määrin jakaa tukiasemien yksilölliseen tutkimiseen.

Tässä työssä tutkitaan ohjaamattomaan oppimiseen perustuvia menetelmiä poikkeavien mittausten etsimisessä. Valittaessa menetelmiä on pyritty huomioimaan niiden soveltuvuus soluverkosta kerätyn datan analysointiin. Verkon analysointi on jaettu yksittäisen solun tutkimiseen ja solujen väliseen vertailuun.

Tarkempaan tutkimukseen valitut tilastolliset ja neuraalit menetelmät testataan synteettisellä datalla ja oikeasta GSM-verkosta kerätyllä datalla. GSM-verkon datan tapauksessa käsitellään edellä mainutut kaksi eri tapausta soluverkolle. Testeissä pyritään käyttämään menetelmiä hyödyntämällä mahdollisimman vähän käytettävissä olevaa prioritietoa verkosta, jotta menetelmien siirtäminen uuden datan käsittelyyn olisi helpompaa. Saatuja tuloksia analysoidaan vertaamalla menetelmiä sekä keskenään että yleisellä tasolla mutta lopulliset päätökset niiden soveltuvuudesta verkon analysointiin jätetään verkkoasiantuntijoiden päätettäväksi.

Avainsanat

poikkeavuuksien etsintä, k:n lähimmän naapurin menetelmä, neuroverkko, Parzen ikkunat, Gaussin mikstuura, etäisyysmitta, matkapuhelinverkko

Acknowledgments

This Master's thesis has been done in the Laboratory of Computer and Information Science of the Department of Computer Science and Engineering at the Helsinki University of Technology.

First, I want to thank my supervisor Olli Simula and instructor Kimmo Raivio for all the advice they have given me during this work and previous projects.

I wish to thank Pasi Lehtimäki for the clarifying and constructive discussions which helped me to concentrate on the relevant things.

My gratitude also goes to the people sharing room A322 with me, and to other people in the laboratory: you are an important part of a nice and relaxed work environment.

Finally, I would like to thank my family for the invaluable support they have given me in the course of my life and studies.

Otaniemi, 2nd May 2007

Mikko Multanen

Abbreviations and notations

2G	Second generation
3G	Third generation
AuC	Authentication Center
BSC	Base Station Controller
BSS	Base Station Subsystem
BTS	Base Transceiver Station
EDGE	Enhanced Data rates for GSM Evolution
EIR	Equipment Identity Register
EM	Expectation Maximization
FDMA	Frequency-Division Multiple Access
GMM	Gaussian Mixture Model
GPRS	General Packet Radio Services
GSM	Global System for Mobile communications (originally: Groupe Spécial Mobile)
HLR	Home Location Register
HO	Handover
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
KPI	Key Performance Indicator
MS	Mobile Station
MSC	Mobile service Switching Center
NG	Neural Gas
NSS	Network Subsystem
PDC	Personal Digital Cellular
PDF	Probability Density Function
ROC	Receiver Operating Characteristic curve
SIM	Subscriber Identity Module
SOM	Self-Organizing Map
SDCCH	Stand-alone Dedicated Control Channel
SMS	Short Message Service
TCH	Traffic Channel
TDMA	Time-Division Multiple Access

UMTS	Universal Mobile Telecommunications System
VLR	Visitor Location Register
VQ	Vector Quantization
$p(x)$	Continuous PDF evaluated at x
$P(x)$	Discrete PDF evaluated at x
x	Scalar
$\boldsymbol{x}$	Vector
$\boldsymbol{X}$	Matrix or a set of variables
$\|a\|$	Absolute value of a
$\\|\boldsymbol{x}\\|$	Euclidean norm of $\boldsymbol{x}$
$\|A\|$	Number of elements in a set A
$\boldsymbol{I}$	Unity matrix
$\mu, \boldsymbol{\mu}$	Mean
σ^2	Variance
$\boldsymbol{\Sigma}$	Covariance matrix
$\boldsymbol{\Theta}$	Set of parameters
$\boldsymbol{\Theta}^*$	Estimate of $\boldsymbol{\Theta}$

Contents

Chapter 1

Introduction

The development of digital data acquisition and storage technology has enabled us to collect huge databases. In all fields more and more information is moving and mobile networks are not an exception. There is a vast amount of information that is crucial to the operation of a network, for example, monitoring of the network elements to detect hardware problems and billing to collect revenues from the use of the services provided by the network. As the information is already stored into a database it would be beneficial for the operator to use it for other tasks too. The performance of the network could be improved by adjusting the software parameters or the antenna configurations of the network in the best case avoiding the need for costly hardware upgrades. Also finding fraudulent usage patterns and other attacks against the system and the business can perhaps be detected by using the same database. These improvements can also reflect to the satisfaction of the customers. It may have a positive effect on their experiences and improve their view of the service as it becomes more reliable and the customers experience less problems with it.

The databases are usually too large for a direct human analysis so we need intelligent computer aided data mining and exploration tools to help us in the task. Hand et al. [13] define data mining as:

> *Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner.*

Typical tasks in data mining are classification, regression, clustering of the data, determining parameter dependencies and anomaly detection.

The analysis of the data starts with the detection of the problem or interesting part of the data [39]. After the problem is detected we can move to identify and diagnose it and then perhaps take some appropriate actions in the process from which the data was collected. The classical statistical techniques are designed to test some definite problem. They can be called confirmatory methods. But in practical situations the problems may not be so well defined and the classical statistical tests can behave badly. The exploratory methods give us a way to approach the data with flexible probing. The methods are robust and resistant and not designed for any narrow situation. They are usually the methods that give the first look to the data [16].

1.1 Objectives and scope

The aim of the thesis is to investigate how anomaly detection methods can be utilized in the exploratory analysis of mobile network data. Traditionally confirmatory methods are used in the monitoring of the network. They give alarms when some definite test fails. The methods in this thesis are suited to the exploratory analysis of the data, to get a good first look of the data and possibly to find some novel properties of the data which aren't evident to the confirmatory methods. The usual tasks where anomaly detection is useful in the analysis of the data include preprocessing the data to clean it for other analysis methods and monitoring and detecting changes in the operation to find problematic parts. Because the detection methods depend heavily on the data available there is no clear basic method or analysis process that could be used in all the cases. Different kinds of methods are reviewed and examined how they can be used for diverse data.

Some methods are tested with data which is collected from a real GSM network. The data is not pre-labeled so the investigated methods are limited to those who do not need in advance training but can operate directly with the data at hand. The basic theory of the methods is presented in a way that the reader can understand the results of the experiments that has been made.

1.2 Organization of the thesis

The rest of the thesis is structured as follows. In Chapter 2, the background of analyzing mobile network data and the architecture of a mobile network is introduced. Also the features of the data that is collected from the network are presented there. The outlier detection task in general and the theory of the outlier detection methods are presented in Chapter 3. In Chapter 4, the methods are tested with synthetic and real data. Concluding remarks are given in Chapter 5.

Chapter 2

Mobile network

During the early 80s analog mobile networks were deployed. Every country had its own national system and there were no international standards. The development of digital second generation (2G) networks fixed this problem partly. A pan-European network standard, GSM, is one that was developed for international usage. But there were also others like PDC which is used in Japan and cdmaOne which is used in the Americas and Asia. The vastly increased demand of mobile phones and need for more capacity created pressure to deploy new networks, where the frequency bandwidth of the cellular system would be used more efficiently and the bit rate would be higher. Third generation (3G) systems like UMTS were designed to answer this demand. An international standard was again a target but unfortunately an unsuccessful one.

The complexity of the networks has grown along the development. In the 2G and earlier networks speech was the main feature the network offered. Today multimedia and data services, like video calls, Internet and email, have taken place from the traditional speech communication and they need to be taken into account when new networks are developed. 2G systems were extended with techniques like GPRS and EDGE to better answer the demand of data traffic. In the 3G systems the proportion of data traffic has increased. Another feature that increases the complexity of the overall systems is that 2G and 3G networks should work well together because the client base on 2G networks is vast and the smooth cooperation helps the transition from the old networks to the new ones.

Because the complexity of the network systems has grown new more automated methods are needed for the configuration and maintenance of the systems. It can be too vast and difficult task even for an expert to do this using the old methods used in the earlier network systems. Operators also desire to keep the efficiency of the network on a high level and the new method could provide ways to help in the performance optimization. Usually important tasks to the maintenance of the network include monitoring of the network elements, analysis of the network for optimization purposes and fraud detection to avoid losses.

The purpose of monitoring is to detect problems in the network in order to keep the quality of the service for customers high and the usage of the network resources on a decent level. Monitoring can be done using simple indicators of the network status or with more complex methods which could find problems that aren't so clear in the indicators. Laiho et al. [23] presented models for the network monitoring and analysis of UMTS networks. Because of the visualization properties of the self-organizing map (SOM) they used it as a basic method in their analysis. Some methods using also other competitive neural networks for monitoring 3G networks were presented by Barreto et al. [3]. They extend some of the anomaly detection ideas of Laiho et al. [23].

Analyzing the network can reveal opportunities to optimize the resource usage of the network. The capacity of the network or the quality of the service for customers can perhaps be increased. A knowledge-based method for analyzing GSM network performance using model fitting was presented by Lehtimäki and Raivio [24]. They divide a GSM network system into subsystems and then analyze and visualize the performance of the subsystems using models.

To keep the paying customers happy and to avoid losses in revenue the detection of fraudulent usage of the network is essential. A fraud happens when someone uses the network without an intent to pay for the usage [19]. One way to perform this kind of fraud is to use the account of a paying customer. This is perceived latest when the paying customer complains about the billing. Hollmén [18] presents methods for fraud detection using probabilistic and neural methods. The fraud detection problem in telephone network is little similar to the intrusion detection problem in computer systems. Ye and Chen [45] presented an anomaly detector which was based on chi-square statistic. The detector handles all the features of the data independently and by doing so decreases the computational needs compared to the basic multivariate anomaly detector where the full covariance matrix and its inverse are calculated.

In this chapter an overview of the GSM network [29, 40] and the data collected from a real operating GSM network are represented.

2.1 Architecture of the GSM network

An overview of the architecture of a basic GSM (Global System for Mobile communications) network is given in this section. The network can be di-

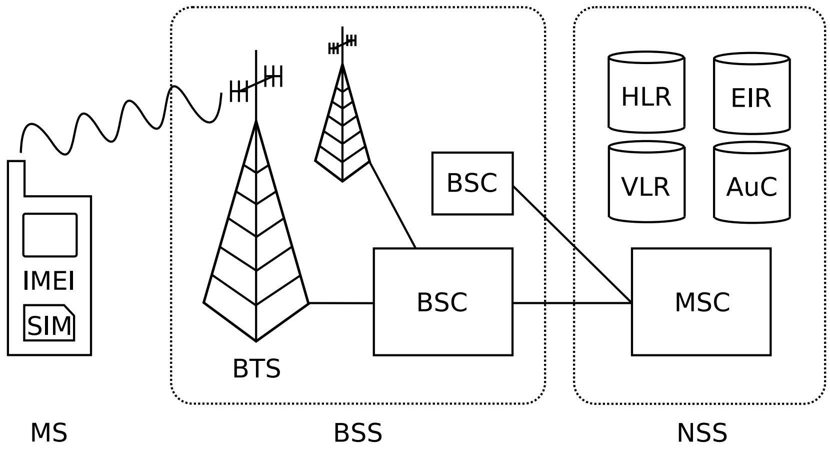


Figure 2.1: Architecture of the GSM network.

vided into three parts, the Mobile Station (MS), the Base Station Subsystem (BSS) and the Network Subsystem (NSS). The subscriber carries the MS. The BSS controls the radio link to the MS. The NSS handles the switching of calls inside the mobile network to other telephone networks. An overview of the network architecture is shown in figure 2.1.

2.1.1 Mobile station

What the subscriber carries is a mobile equipment and a Subscriber Identity Module (SIM) card. These are the parts of the MS. The SIM card makes it possible for the user to use her subscribed services independent of the mobile equipment. The SIM card contains the International Mobile Subscriber Identity (IMSI) that is used in the identification of the user when connecting to the network. Mobile equipments have the International Mobile Equipment Identity (IMEI) which is unique for each mobile equipment.

2.1.2 Base station subsystem

The BSS is composed of two parts, The Base Transceiver Station (BTS) and the Base Station Controller (BSC). The BTS contains radio transceivers and handles the radio-link protocols. The BSC controls the radio resources for one or more BTSs.

2.1.3 Network subsystem

The Mobile service Switching Center (MSC) is the main component of the NSS. The network is connected to other telephone networks through the MSC. It handles the switching and other functionality of the NSS in conjunction with several functional entities, the Home Location Register (HLR), the Visitor Location Register (VLR), the Equipment Identity Register (EIR) and the Authentication Center (AuC).

The location of a subscriber is stored in the HLR and VLR. The MSC uses these registers when routing calls. The administrative information of the subscribers is stored in the operators HLR. The VLR is used when the subscriber is not using her own operator's network. The use of VLR makes roaming [1] possible.

When a user is connecting to the network the validity of the IMEI code of the mobile equipment is checked from the EIR. The access is denied if the IMEI is registered stolen or the IMEI is not type approved. The AuC contains the same secret keys that each subscriber's SIM card contains. The key is used for authentication and encryption over the radio channel.

2.2 Radio link of the GSM network

The radio link uses two separate frequency bands, one for uplink and the other for downlink. Because frequency spectrum is a limited resource the frequency bands are divided into smaller parts to provide access to the network for many subscribers simultaneously. In the GSM network this division is done with a combination of Time- and Frequency-Division Multiple Access (TDMA/FDMA) methods. In the FDMA part the frequency band is divided into smaller subbands. These subbands are then divided to smaller blocks in time using TDMA. Channels are composed of these blocks.

In the GSM network speech is transferred in digital form packed with a lossy codec. As the radio transmission link can suffer from interference error protection is used.

Multipath fading also induces problems. At the 900/1800/1900 MHz range where GSM networks today operate, radio waves bounce off everything: buildings, trees, hills etc. An inverse filtering method is used to alleviate this problem. Slow frequency hopping also helps to alleviate the multipath fading because the multipath fading is dependent of the carrier frequency.

Another positive effect of the frequency hopping is that the co-channel

[1] Roaming occurs when a subscriber of one mobile network operator uses the network of another operator

interference is in effect randomized by it. The co-channel interference can also be alleviated by discontinuous transmission. Because in normal conversation a person speaks approximately less than 40 percent of the time it is not wise to send something all the time.

The battery power in the mobile unit is also saved by using discontinuous transmission. Other ways to conserve power are discontinuous reception and power control. When the mobile unit is not in call mode it does not need to monitor the network all the time. It just checks from time to time if new calls are coming and the rest of the time it can be sleeping. In the power control the idea is to use minimal power levels necessary for an acceptable signal quality between the mobile unit and the BTS. The minimal power levels have also a decreasing effect on co-channel interference.

2.3 Data collected from the network

The data have been collected from a real operative network. It was stored as counters of different events in the network elements. The counters were stored hourly and then reset for the next cycle. The database contained many different counters but less than hundred were used in this project. The counters used in the experiments below concentrate to traffic channel, control channel and handover information.

One BSC from the network system was selected and data samples were collected separately from each BTS. There were 92 BTSs under the selected BSC. The data contains samples from a period of 40 days. There are on average 730 samples from each BTS so there are quite many missing samples over the time line. There are few BTSs which have only little over 200 samples. But there isn't any BTS that has a sample for each time slot.

2.3.1 Key Performance Indicators

A Key Performance Indicator (KPI) is calculated from one or more counters. The KPIs were defined by the network manufacturer. The KPIs are much more intuitive than the counters so they were used instead of the counters. It is much easier to understand what one KPI represents than what the plain counters from which the KPI is calculated represent. On the other hand some information is also lost with KPIs because one KPI value combines the information of many counters and the true problem may not show up in the KPI.

All of these selected KPIs are quality related. They indicate how well the system is performing. The unit of the KPIs is %.

- SDCCH Availability – Indicates the availability of control channels[2]. The value of the KPI should always be near 100%.
- SDCCH Blocking – Indicates the blocking of access to SDCCH which is used for control and SMS [3]. The value of the KPI should be below 0.5%.
- SDCCH Usage – Usage of control channels. This KPI should stay under 10%.
- TCH Availability – Indicates the availability of traffic channels [4] which means that the MS can step into the network. The value of the KPI should be always near 100%.
- TCH Blocking – Indicates the blocking of access. Call is not started. The value of the KPI should be below 0.5%.
- TCH Drop – Indicates the dropping of active connections. The value of the KPI should be below 3%.
- TCH Usage – Indicates the usage of traffic channels. The desired value of this KPI is 50%.
- HO In success – Indicates successful incoming handovers [5]. The value of the KPI should be near 100%.
- HO Out success – Indicates successful outgoing handovers. The value of the KPI should be near 100%.

These KPIs except the usage ones are related to call progress. The most critical KPIs are TCH Drop Rate, which indicates abnormal service interruptions, and TCH Blocking, which indicates service denials. These KPIs have a direct effect to the quality experienced by the subscribers [22]. Problems in the SDCCH represent signaling problems so they may have a connection to other possible problems happening at the same time.

[2] SDCCH (Stand-alone Dedicated Control Channel) is used for transmission of control information [22].

[3] SMS (Short Message Service) is a service that permits the sending of short text messages.

[4] TCH (Traffic Channel) carries encoded speech and data.

[5] HO (Handover) is an event in the network in which a call is transferred from one BTS to another BTS.

Chapter 3

Outlier detection

Anomaly detection is detecting anomalous observations from data. Anomalies can arise due to different reasons such as mechanical faults, other changes in the system, fraudulent behavior, instrument error, human error or natural deviation. Usually anomalous observations are more interesting and need excess examination. It is not easy to define exactly what is an anomaly or an outlier. Grubbs [12] definition for outlier is:

> *An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.*

Barnett and Lewis [2] refined it:

> *An observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data.*

There are many techniques for anomaly detection. The name can vary depending on the task at hand. For example, novelty detection, outlier detection, anomaly detection, noise detection, deviation detection or exception mining are all fundamentally the same. But there are little differences. For example novelty detection is more of detecting new or unknown data that was not available during training and outlier detection is finding observations that do not seem to fit into the data. The data that is available determines strongly what kinds of techniques can be used. Methods are divided roughly to supervised and unsupervised methods according to the available data. Many methods can be used in both cases with little modifications. There also exists semi-supervised methods that are kind of a mixture of supervised and unsupervised methods.

In the case that we have pre-labeled data (label for observations is either normal or abnormal) we can classify both normality and abnormality. This is analogous to supervised classification task. The supervised analysis is illustrated in Figure 3.1. The hard part in supervised classification is to choose

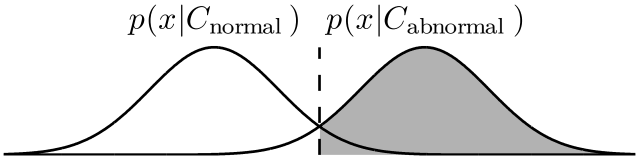


Figure 3.1: Differential analysis. The density function of the normal class C_{normal} is $p(x|C_{\text{normal}})$ and the density function of the abnormal class C_{abnormal} is $p(x|C_{\text{abnormal}})$. The gray area is classified as abnormal.

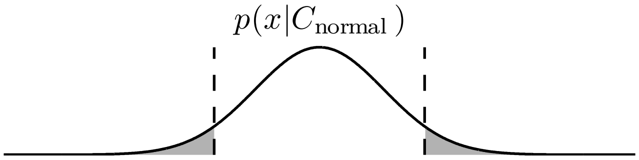


Figure 3.2: Absolute analysis. The density function of the normal class C_{normal} is $p(x|C_{\text{normal}})$. The gray area is classified as abnormal.

the boundary between the normal class and the abnormal class. There exist many general algorithms for supervised classification tasks, such as, linear classifiers, support vector machines, *k*-nearest neighbors and neural networks [15, 6]. Countless improvements have also been developed to the basic methods. For example, Titsias and Likas [43] combined probabilistic aspects with neural ones and Goldberger et al. developed a novel method for improving *k*-nearest neighbors classification [10].

If the data contains samples from one class only normality or in few cases abnormality is modeled. This task is usually called novelty detection as the new data samples are compared to the old data. The semi-supervised analysis is illustrated in Figure 3.2 . The task is to find a boundary for the normal class. For good performance the data must cover the whole gamut of normality but no abnormal data is required as in the supervised classification. This approach is good in the cases when it is costly to get abnormal observations. For example, if we had to break a big jet engine [42] just to get some abnormal data it would be beneficial to collect only normal data and model only the normal operation of the engine. Barreto et al. [3] used also only the normal data in their methods for the monitoring of a cellular network. Markou and Singh [26, 27] give a very good introduction to different methods that are useful in novelty detection.

If we do not have any prior knowledge of the data we can think that the task of finding outliers is equivalent to unsupervised clustering. In this case the only thing we can do is to find those observations that are most

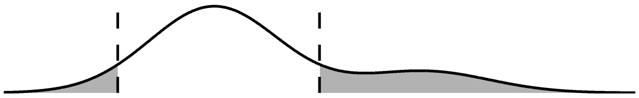


Figure 3.3: Absolute analysis. The combined density function of the normal class and the abnormal class is $p(x|C_{\text{normal\&abnormal}})$. The gray area is classified as abnormal.

remote or improbable compared to the whole data. The analysis method of a distribution which is a mixture of normal and abnormal classes is illustrated in Figure 3.3. The task of deciding a boundary for normality is here similar to the case in semi-supervised analysis. Many semi-supervised methods can also be used with little modifications as unsupervised methods too. Usually we assume that the normal class dominates the distribution of the training data, the number of abnormal samples is much smaller than the number of normal samples. Eskin [8] presented a probabilistic method for finding anomalies using a mixture model and applying a statistical test to detect the anomalies. The method leverages the fact that the anomalies are rare within data compared to the number of normal samples.

The rest of this chapter covers the theory of outlier detection methods that are used in the experiments later. All the methods are exploratory as they do not need strong assumptions of the distribution of data. Smoothness of the distribution is the main assumption made. The methods use unsupervised training because the data that was available for experiments did not contain any pre-labeled training data and thus prevented the use of supervised or semi-supervised methods. Although with minor modifications some of the methods presented here are usable in a supervised manner, too.

The approaches to outlier detection can be divided into statistical and neural ones. In the statistical approaches the data is modeled based on its statistical properties and the estimated model is used to test whether a test sample comes from the same distribution. Statistical approaches can be divided to non-parametric ones and parametric ones. Nearest neighbor and Parzen windows belong to the non-parametric approaches and the Gaussian mixture model is a parametric one. The idea behind the competitive neural models covered last in this chapter is very close to these statistical approaches.

3.1 Nearest neighbors

Nearest neighbors algorithms are based on the local distances between the data points. If the neighbors of a data point are close it is quite probable that the point belongs to the same class as its neighbors. But if all the neighbors are remote the point might be an outlier.

Byers and Raftery [5] introduced an algorithm for using the kth nearest neighbor in detecting clutter. Ramaswamy et al. [34] formulated the same idea to an exact algorithm for outlier detection. All the points are ranked based on their distance to the kth nearest neighbor. The n most distant points according to this ranking are considered outliers.

When using this algorithm the only requirement for the data is that there is a way to define a distance between two data points. The major problem with this kind of approaches is that the computational complexity grows quite fast with the size of the data set as all the distances between data points must be calculated. If the dimensionality of the data set is small the complexity of finding the kth nearest neighbor can be alleviated by using advanced data structures such as kd-trees [4].

3.2 Probability distribution estimation

In unsupervised probabilistic outlier detection we don't know the underlying probability distribution of either normal data or abnormal (outlier) data. We only have a sample set which is drawn from the mixture of the distributions. However we can usually assume that most of the samples come from the distribution of normal data and so the sample set represents quite well the distribution of normal data.

An estimate $p(\boldsymbol{x})$ for the distribution of normal data can be build by using the sample set. Those samples that are least probable to come from the estimated distribution are treated then as outliers.

$$\begin{array}{rl} \text{If} & p(\boldsymbol{x}) > K_\alpha \\ \text{Then} & \boldsymbol{x} \text{ is normal} \\ \text{Else} & \boldsymbol{x} \text{ is abnormal} \end{array} \tag{3.1}$$

where the threshold K can be set by using a confidence level. K_α means that α percentage of samples is below the threshold and if a new sample is above the threshold we can say that with $100(1-\alpha)\%$ confidence it is normal. The use of a threshold is illustrated in Figure 3.4.

For estimating the probability distribution two methods are previewed next, Parzen windows and a mixture model. They have a very similar base

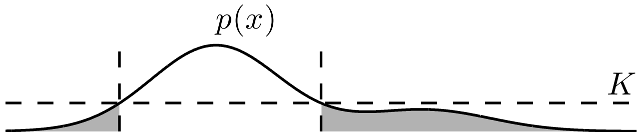


Figure 3.4: Probability density function

idea, the probability distribution is modeled with a mixture of simple probability density functions. The difference between them is that in the Parzen windows the number of components in the mixture is the same as the number points in the sample set and in the mixture model the number of components is usually much smaller.

3.2.1 Parzen windows

Parzen windows is a technique for estimating the unobservable underlying probability density function according to which all the data is distributed using the data at hand. The data at hand are seen as a random samples drawn from the distribution. Tarassenko et al. [41] used Parzen windows in identification of masses in mammograms. Parzen windows can be regarded as a generalization of nearest neighbor techniques. All the data points are taken into account rather than using only the nearest ones.

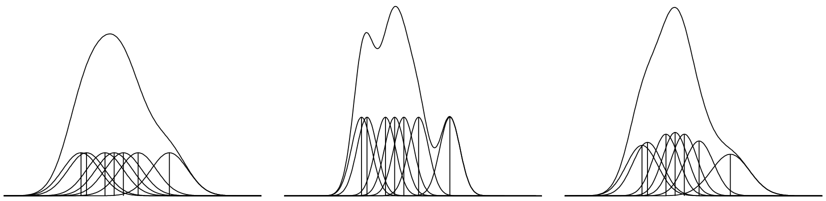
The probability density function is built by placing a kernel function for each data point and taking an average over all the kernels.

$$p(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} K_i(\boldsymbol{x}) \tag{3.2}$$

where $p(\boldsymbol{x})$ is the modeled probability density function, N is the number of data points and $K_i(\boldsymbol{x})$ is some kernel function, i.e., some probability density function when we want our model to be a probability density function. A Gaussian function is a common choice for the kernel function.

$$K_i(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_i|^{1/2}} \mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x}-\boldsymbol{x}_i)} \tag{3.3}$$

where $\boldsymbol{x}_i$ is a data point, $\boldsymbol{\Sigma}_i$ is a covariance matrix and n is the dimension of the data. The covariance matrix $\boldsymbol{\Sigma}_i$ controls the smoothness of the density function. In the most simple form $\boldsymbol{\Sigma}_i$ can be a scalar σ^2 which is constant over i. Choosing of σ^2 is critical. If it is too large all detail is obscured and if it tends to zero the estimate will be a sum of Dirac delta function spikes at the observations.



(a) Constant $\sigma_i^2 = 1$ (b) Constant $\sigma_i^2 = 0.3$ (c) Adaptive nearest neighbors smoothing $h = 0.24$

Figure 3.5: Parzen windows estimate

When the smoothness parameter $\mathbf{\Sigma}_i$ is a constant the main problem is that it is not always possible to choose a good value for it. When the parameter is set so that the main part of the distribution is shown in detail there can be spurious noise in the tails and if the distribution is smoothed to hide the noise the detail in the main part is lost. So an adaptive smoothing method is needed.

Adaptive nearest neighbors smoothing

Neighborhood of a kernel function can be used in defining its width. The distance d_i of data point i to its kth nearest neighbor is calculated as described in Section 3.1. For data point i the variance

$$\sigma_i^2 = h\Big(\frac{d_i}{\mu_{\mathrm{Chi}_n}}\Big)^2 \tag{3.4}$$

where h is a parameter for overall smoothness and μ_{Chi_n} is a mean of n-dimensional Chi distribution. The length of a n-dimensional vector whose components are independent and normally distributed with zero mean has a Chi distribution.

Using the adaptive smoothing with the Gaussian kernel estimates the underlying distribution with N spherical multivariate normal distributions whose variance changes according to the density of the data. The use of adaptive smoothing is illustrated in Figure 3.5. The adaptive method gives better fineness of detail in the parts where there are more data points without being too sensitive in the tails of the distribution.

3.2.2 Mixture model

A mixture model is a technique for estimating a complex probability density function with a convex combination of simple density functions. One

(a) (b)

Figure 3.6: Gaussian mixture model. Sample points in the graphs were drawn from a mixture distribution of two Gaussians. The initial number of components in the mixture model has a vast effect to the robustness of the model which can be seen by comparing the above results.

popular version is a Gaussian mixture model (GMM). In GMM the estimate of the modeled probability density function

$$p(\boldsymbol{x}) = \sum_{i=1}^{K} w_i N(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{3.5}$$

in which K is the number of Gaussians, w_i is the weight of Gaussian i ($0 < w_i < 1 \; \forall \; i = 1, \ldots, K$ and $\sum_{i=1}^{K} w_i = 1$) and $N(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ is a density function of a normal distribution parametrized by mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$.

$$N(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_i|^{1/2}} \mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)} \tag{3.6}$$

To get different versions of the Gaussian mixture model the number of free variables in the covariance matrix $\boldsymbol{\Sigma}_i$ is decreased. The simplest case is to use a scalar variance $\boldsymbol{\Sigma}_i = \sigma_i^2$, next step is to use diagonal covariance matrix where the variances of different features are only used. There are many methods which base on how the number of free parameters in the covariance matrix is decreased from the full covariance matrix [11, 9, 32].

One much used way to estimate the parameters of a Gaussian mixture model is to use the expectation maximization algorithm. The expectation maximization algorithm is an algorithm for finding maximum likelihood estimates of parameters in a probabilistic model. The idea is that we want to estimate some unknown parameters $\boldsymbol{\Theta}$, given data $\boldsymbol{X}$. However there are also latent variables $\boldsymbol{L}$ which need to be integrated out. We want to find a maximum for the posterior probability of the parameters $\boldsymbol{\Theta}$ given the data $\boldsymbol{X}$ and marginalizing over the latent variables $\boldsymbol{L}$.

$$\boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}} \sum_{\boldsymbol{L}} P(\boldsymbol{\Theta}, \boldsymbol{L}|\boldsymbol{X}) \tag{3.7}$$

The intuition behind the expectation maximization algorithm is to alternate between estimating the unknown parameters $\boldsymbol{\Theta}$ and the latent variables $\boldsymbol{L}$.

In the Gaussian mixture model the unknown parameters $\boldsymbol{\Theta}$ we would like to know are the means $\boldsymbol{\mu}_i$, the covariances $\boldsymbol{\Sigma}_i$ and the weights w_i. We can assume that each data point $\boldsymbol{x}_j$ comes from one of the distributions in the mixture. This membership can be seen as latent variables $\boldsymbol{L}$ in the model. To start the algorithm some initial values for the parameters are needed. After that the algorithm proceeds by alternating between the expectation and the maximization steps. The probabilities $P(g_i|\boldsymbol{x}_j, \theta_t)$ which indicate the membership of the data point $\boldsymbol{x}_j$ to Gaussian g_i are computed in the expectation step .

$$P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t) = \frac{N(\boldsymbol{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)w_i}{\sum_{k=1}^{K} N(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)w_k} \tag{3.8}$$

New estimates for the mean $\boldsymbol{\mu}_i$, the covariance $\boldsymbol{\Sigma}_i$ and the weight w_i parameters are calculated by maximizing the expected log-likelihood of the joint event $\mathrm{E}_{\boldsymbol{g}}\left[\ln \prod_{j=1}^{N} p(\boldsymbol{x}_j, \boldsymbol{g}|\theta)|\boldsymbol{x}_j\right]$.

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)\boldsymbol{x}_j}{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \tag{3.9}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \tag{3.10}$$

$$w_i = \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)}{\sum_{j=1}^{N}\sum_{i=1}^{K} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \tag{3.11}$$

These new parameters $\boldsymbol{\Theta}_{t+1}$ are then used again in the expectation step and the process is iterated until some acceptable level is achieved [7, 30]. The derivation of the update rules is presented in more detail in Appendix A.

This basic update rule has a problem with covariance matrices becoming singular. Ormoneit and Tresp [33] proposed the use of Bayesian regularization. Conjugate priors are set for the mixture parameters so that the solution of the posterior density is also analytic and EM-update rules can be derived to obtain the MAP parameter estimates. In the simplest case when focusing on the variance prior the update rule for the covariance matrices becomes:

$$\boldsymbol{\Sigma}_i = \frac{\beta \boldsymbol{I} + \sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \theta_t)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \theta_t)} \tag{3.12}$$

where β controls how much large variances are favored. This method also helps with overfitting when β is not too small. If β is chosen too large the model can become too constraining to model well the underlying distribution.

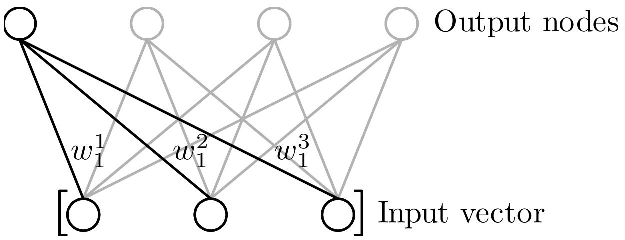


Figure 3.7: Competitive neural network

A practical task that can also be done between the rounds of the iteration is to decrease the number of components in the mixture if necessary. The robustness of the model suffers if there are too many components in the model (Figure 3.6). One heuristic way to do the reduction is to drop those components i whose weight w_i is very small because we can think that their influence on the final mixture is minimal. When using spherical components, covariance matrix $\mathbf{\Sigma}_i$ is just a scalar variance σ_i^2, a way to decrease the number of components is to remove components whose variance σ^2 becomes zero because these components try to model only one separate point of the data set.

3.3 Competitive neural models

A competitive neural network contains two layers of nodes, an input layer and an output layer (Figure 3.7). Weights $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$ of the output nodes can be seen as prototype vectors. The dimension of prototype vectors is the same as the dimension of the input vectors. The goal in the competitive neural models is to divide the data into a number of clusters such that the inputs in a cluster are in some sense similar. After learning, each prototype vector will represent some part of the data (Figure 3.8). The number of clusters is the same as the number of nodes in the output layer. An input belongs to the cluster whose prototype vector is closest to the input. The closeness is usually measured in the sense of Euclidean distance. The prototype vectors define a codebook for vector quantization, a process where the vectors in the codebook are used to represent the data. The process can be seen as a lossy data compression method. The differences between the methods covered in this section are in the learning algorithms.

3.3.1 Training of a competitive neural network

A general sequential training algorithm for a competitive neural network is as follows

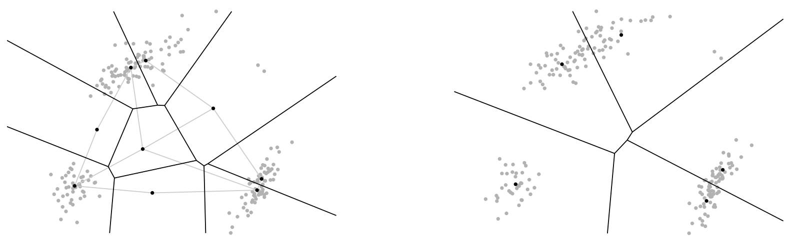
1. Choose the number of clusters K.
2. Initialize the prototypes $\boldsymbol{w}_1, \dots, \boldsymbol{w}_K$, i.e. by taking random vectors from the input data.
3. Pick randomly an input vector $\boldsymbol{x}$ from the training data.
4. Calculate the distances $\boldsymbol{d} = [d_1, \dots, d_K]$ between the prototypes $\boldsymbol{w}_i$ and the input $\boldsymbol{x}$.
$$d_i = \|\boldsymbol{w}_i - \boldsymbol{x}\| \tag{3.13}$$
5. Update the prototype vectors $\boldsymbol{w}_i$ according to the distances $\boldsymbol{d}$.
$$\boldsymbol{w}_i^{\text{new}} = \boldsymbol{w}_i^{\text{old}} + \eta \mathrm{h}_i(\boldsymbol{d})(\boldsymbol{x} - \boldsymbol{w}_i^{\text{old}}) \tag{3.14}$$
6. Check for a stopping criteria or return to step 3.

where $\mathrm{h}_i(\boldsymbol{d})$ is a function that determines which ones of the prototype vectors are updated and η is a learning rate. Several different methods are possible depending on whether the learning rate is constant or decays with time. When the learning rate is constant the system stays adaptive and is therefore able to follow also non-stationary distributions but if the distribution is stationary a decaying learning rate is better. The convergence of the prototypes is better with decaying learning rate because then they can be fine-tuned with smaller learning rates. One way to decrease the learning rate over time is an exponential decay [37] $\eta(t) = \eta_0(\eta_T/\eta_0)^{t/T}$ where η_0 and η_T are initial and final values of $\eta(t)$ and T is the total number of training rounds.

The simplest training method is a competitive neural vector quantization. "Winner-take-all" describes the learning rule of vector quantization well. The prototype vectors compete for activation and only the winner, the closest prototype is updated.

$$\mathrm{h}_i(\boldsymbol{d}) = \begin{cases} 1 & \text{if } i = \operatorname{argmin}_j d_j \\ 0 & \text{else} \end{cases} \tag{3.15}$$

In the self-organizing map (SOM) [21] the nodes on the output layer are ordered as a regular low dimensional grid, usually two dimensional. The prototype vectors form an elastic net. When the SOM is trained the net settles so that the prototype vectors represent the training data as well as possible without losing the net structure, so the topology is preserved. The net



(a)

(b)

Figure 3.8: Self-organizing map and neural gas. The gray lines represent the grid of the SOM.

structure of a trained SOM is shown in Figure 3.8(a). Because of the two dimensional grid structure it is straightforward to visualize multidimensional data in two dimensions with a SOM. In the training the prototype vectors compete for activation and the winner, the closest prototype vector, and its neighbors are updated. The neighborhood function $\mathrm{h}_i(\boldsymbol{d})$ defines how the neighbors are updated. A Gaussian weighting function can be used as a neighborhood function.

$$\mathrm{h}_i(\boldsymbol{d}) = \mathrm{e}^{-\frac{|\boldsymbol{r}_i - \boldsymbol{r}_k|^2}{\sigma^2}}, \quad k = \operatorname*{argmin}_j d_j \tag{3.16}$$

where $\boldsymbol{r}_i$ and $\boldsymbol{r}_k$ are the positions of nodes i and k on the output grid, σ^2 defines the radius of the neighborhood and σ^2 should decay with time just like η.

In the training phase of a neural gas (NG) network [28] output nodes are ranked according to the distance to the input. The ranking effects so that the adaptation of the closest nodes is stronger. The prototype vectors compete for activation and all the prototypes are updated according to their ranking of closeness to the input. $\mathrm{h}_i(\boldsymbol{d})$ plays a similar role as the neighborhood function in a self-organizing map.

$$\mathrm{h}_i(\boldsymbol{d}) = \mathrm{e}^{-\frac{k-1}{\lambda}} \tag{3.17}$$

where k is the position of the node i in the ranking of the distances $\boldsymbol{d}$ and λ defines the radius of effect and λ should also decay with time just like η.

The neural methods introduced above used sequential training. Training can also be done in a batch mode. In each iteration round the whole data set is processed at first and the weights are updated after that as one batch and this is iterated until the weights converge to some satisfactory state.

An obvious downside with the batch mode is that the whole training set is needed for the training at the start. An advantage that the batch mode is usually said to have is that it converges faster than the sequential mode [21] and is thus computationally more efficient. There are also proposals to use sequential batch mode where the training data is processed in smaller chunks [1]. This gives to a certain extent the advantages of the batch mode though we don't necessarily need to have the whole training set at the beginning.

3.3.2 Outlier detection using prototype vectors

Regardless of the neural training method the prototype vectors can be used in the outlier detection task. After the training of one of the neural networks presented above, the quantization error of an input can be used in detecting if the input is an outlier. The quantization error e^i is the distance between the winner prototype $\boldsymbol{w}_k^i$ and the input $\boldsymbol{x}^i$.

$$e^i = \|\boldsymbol{E}^i\| = \|\boldsymbol{w}_k^i - \boldsymbol{x}^i\| \tag{3.18}$$

where $\boldsymbol{E}^i$ is the quantization error vector.

For the outlier detection Höglund et al. [17] and Laiho et al. [23] used a simple threshold method where a threshold e_p is defined for the quantization error e.

$$\begin{array}{rl} \text{If} & e^i < e_p \\ \text{Then} & \boldsymbol{x}^i \text{ is normal} \\ \text{Else} & \boldsymbol{x}^i \text{ is abnormal} \end{array} \tag{3.19}$$

The decision threshold can be found in a non-parametric way using a confidence level. Threshold e_p means that $p = 1 - \alpha$ percentage of samples is below the threshold and if a new sample is below the threshold we can say that with $100p\%$ confidence it is normal. This approach has a problem if there are outliers in the training data because it is possible that some of the nodes of the network represent outliers.

A little more complex two threshold method was presented by Barreto et al. [3]. The outlier detection is done in two phase by using decision intervals. A first phase detector is named as a global detector and the quantization error e is used in it.

$$\begin{array}{rl} \text{If} & e^i \in [e_p^-, e_p^+] \\ \text{Then} & \boldsymbol{x}^i \text{ is normal} \\ \text{Else} & \boldsymbol{x}^i \text{ is abnormal} \end{array} \tag{3.20}$$

Thresholds e_p^- and e_p^+ are set so that p percentage of samples are inside the interval $[e_p^-, e_p^+]$. If the global detector comes to the conclusion that the new input is an outlier the input is fed to a second phase detector named as a local detector. In the local detector the individual components of the quantization error vector $\boldsymbol{E^i}$ are used.

$$\begin{array}{rl} \text{If} & |E_j^i| \in [|E_j^-|, |E_j^+|] \\ \text{Then} & \boldsymbol{x}^i \text{ is normal} \\ \text{Else} & \boldsymbol{x}^i \text{ is abnormal} \end{array} \tag{3.21}$$

where E_j^i is the j:th component of the quantization error vector $\boldsymbol{E^i}$ and the interval for the j:th component is $[|E_j^-|, |E_j^+|]$. This approach sorts out those components which are responsible of the abnormal state. Also some false alarms of the global detector can be detected if none of the components in the local detector are classified as abnormal.

When using neural networks to quantize inputs some of the prototype vectors can represent outliers or nothing at all which is the case of a dead neuron. Those prototypes should be filtered from the network structure to improve the outlier detection task. One way that is quite similar with the probabilistic methods described in Section 3.2 is to use Gaussians with the prototype vectors. The prototype vectors are seen as the centers of Gaussians and a probability density function is built.

$$p(\boldsymbol{x}) = \sum_{i=1}^{K} w_i K_i(\boldsymbol{x}) \tag{3.22}$$

A Gaussian kernel is placed for each prototype vector.

$$K_i(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_i|^{1/2}} \mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{w}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{w}_i)} \tag{3.23}$$

where the covariance $\boldsymbol{\Sigma}_i$ and the weight w_i for the prototype i are calculated using those training samples A_i that have the prototype i as the closest prototype.

$$\boldsymbol{\Sigma}_i = \frac{1}{|A_i|} \sum_{\boldsymbol{x} \in A_i} (\boldsymbol{x} - \boldsymbol{w}_i)(\boldsymbol{x} - \boldsymbol{w}_i)^T \tag{3.24}$$

$$w_i = \frac{1}{|A_i|} \tag{3.25}$$

$$A_i = \left\{ \boldsymbol{x} \in D \mid \|\boldsymbol{w}_i - \boldsymbol{x}\| \leq \|\boldsymbol{w}_j - \boldsymbol{x}\| \; \forall\, j = 1, \ldots, K \right\} \tag{3.26}$$

where D is the set of training samples.

Using the density function $p(\boldsymbol{x})$ we can calculate the least probable training data points as in Section 3.2 and define them as outliers. If only outliers hit a prototype that prototype should be removed from the structure of the neural network or marked to only classify outliers. After the filtering of outlier prototypes the simple quantization error based method presented above can be utilized in the final outlier detection.

3.4 Distance between two data sets

When classifying entities that are composed of many data points we need methods to measure the dissimilarities or the distances between the data point sets. After the distances are known the outlying entities can be detected using for example the methods presented in Section 3.1 and Section 3.2.1.

There are three properties a real distance measure must fulfill.

$$d(p,q) \geq 0, \quad \text{and} \quad d(p,q) = 0 \quad \text{iif.} p = q \tag{3.27}$$

$$d(p,q) = d(q,p) \tag{3.28}$$

$$d(p,q) + d(q,h) \geq d(p,h) \tag{3.29}$$

where p, q and h are probability density functions or some other model of the entities. The distance between p and q is denoted by $d(p,q)$.

3.4.1 Nearest neighbors distance metric

The simplest way to measure the dissimilarity between two data point sets is to build a dissimilarity measure which is directly based on the data points. Nearest neighbor classification can be used here. For every data point k nearest neighbors are found. The number of points from data set one and the number of points from data set two are counted. The closer the ratio between these numbers is the ratio of the sizes of the data sets the smaller the dissimilarity is between the data sets. So the dissimilarity measure is:

$$d_{k-\mathrm{nn}} = \frac{\sum_i^N \left(|n_i^{c1} - l^{c1}| + |n_i^{c2} - l^{c2}|\right)}{l^{c1}N_2 + l^{c2}N_1} \tag{3.30}$$

$$N = N_1 + N_2$$

$$l^{c1} = k\frac{N_1}{N}$$

$$l^{c2} = k\frac{N_2}{N}$$

where n_i^{c1} and n_i^{c2} are the number of nearest data points from classes 1 and 2, N_1 and N_2 are the number of data points in classes 1 and 2. The distance measure is scaled so that it is always $d_{k-\mathrm{nn}} \in [0,1]$. This dissimilarity measure fulfills at least the two first properties of a distance measure the third one is harder to prove.

3.4.2 Distance metrics for probability density functions

Another way to model an entity is to estimate a density distribution from the data. Two ways to build a probability density function are presented in Section 3.2.1 and Section 3.2.2. Mixture models are a popular choice for the estimate because they are usually quite simple and compress well the essential information in the data. So for each entity a mixture model is computed. These models are then used when evaluating the similarity between two entities. In speech processing Gaussian mixture models are found to be a good method [36, 35, 31].

One straightforward distance measure between two probability density functions is

$$d_{L^r}(p,q) = \Big(\int |p(\mathbf{x}) - q(\mathbf{x})|^r \,\mathrm{d}\mathbf{x}\Big)^{1/r} \tag{3.31}$$

where common values for r are 1 and 2. This satisfies the three properties of distance measure. As a negative property this one cannot be usually computed analytically so we must lean on numerical methods.

Importance sampling

The main idea in importance sampling is to choose a good distribution from which to simulate random variables. Any integral can be computed using sampling. An integral can be written as an expectation with respect to some suitable sample distribution $g(x)$.

$$\int f(x)\,\mathrm{d}x = \int \frac{f(x)}{g(x)} g(x)\,\mathrm{d}x = \mathrm{E}_g\Big[\frac{f(x)}{g(x)}\Big] \tag{3.32}$$

We get an estimate for the expectation by calculating the sample mean.

$$\tilde{f}_n^g(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{f(X_i)}{g(X_i)} \quad , X_i \sim g(x) \tag{3.33}$$

where the samples X_i follow $g(x)$. As $g(x)$ obviously has an effect on the performance of computing the integral some care should be used when choosing it. It can be shown that $\mathrm{Var}[\tilde{f}_n^g(x)]$ is minimized when $g(x) \propto |f(x)|$ [38]. Some good rules for choosing the sample probability distribution $g(x)$ are:

1. $g(x) > 0$ whenever $f(x) \neq 0$.
2. $g(x)$ should be as close as possible of being proportional to $|f(x)|$.
3. It should be easy to simulate samples from $g(x)$.

4. It should be easy to compute $g(x)$ with any x.

The distance measure $d_{L^r}(p,q)$ can be computed with

$$\tilde{d}_{L^r}(p,q) = \left(\frac{1}{n}\sum_{i=1}^{n}\frac{|p(\boldsymbol{x}_i) - q(\boldsymbol{x}_i)|^r}{g(\boldsymbol{x}_i)}\right)^{1/r} \quad ,\boldsymbol{x}_i \sim g(\boldsymbol{x}) \tag{3.34}$$

where $g(\boldsymbol{x})$ should follow well $|p(\boldsymbol{x}) - q(\boldsymbol{x})|^r$. Combined mixture probability density function $1/2(p(\boldsymbol{x}) + q(\boldsymbol{x}))$ is a possible candidate for $g(\boldsymbol{x})$. When $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ are clearly separate $1/2(p(\boldsymbol{x}) + q(\boldsymbol{x}))$ is proportional to $|p(\boldsymbol{x}) - q(\boldsymbol{x})|$ and the sampling goes well. But if $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ are very similar except their variances differ the samples are concentrated on the density centrals of $1/2(p(\boldsymbol{x}) + q(\boldsymbol{x}))$ and therefore the importance of the tails may be underestimated. So the results of this distance measure might emphasize more the differences in the locations of the density centrals than the differences in the variances.

3.4.3 Parametric distance metric for mixture models

Liu and Huang [25] presented a parametric distance metric for mixture probability density functions. Their idea is to calculate first all the distances between the different components of two mixtures and then construct the distance between the mixtures as a weighted sum of the distances between the components.

Mixture models $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ are of type,

$$p(\boldsymbol{x}) = \sum_{i=1}^{K}\alpha_i p_i(\boldsymbol{x}), \quad q(\boldsymbol{x}) = \sum_{j=1}^{L}\beta_j q_j(\boldsymbol{x}) \tag{3.35}$$

where $p(\boldsymbol{x})$ is composed of components $p_i(\boldsymbol{x})$ and $q(\boldsymbol{x})$ is composed of components $q_j(\boldsymbol{x})$. The corresponding weights α_i and β_j satisfy $\sum_{i=1}^{K}\alpha_i = 1$ and $\sum_{j=1}^{L}\beta_j = 1$ so that $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ are real probability density functions. The distance between any two components p_i and q_j is denoted by $d(p_i, q_j)$. If these distances are known the overall distance between $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ is defined as

$$d_M(p,q) = \min_{w=[w_{ij}]}\sum_{i=1}^{K}\sum_{j=1}^{L} w_{ij} d(p_i, q_j) \tag{3.36}$$

$$w_{ij} \geq 0, \quad 1 \leq i \leq K, 1 \leq j \leq L$$

$$\sum_{i=1}^{K} w_{ij} = \beta_j, 1 \leq j \leq L, \quad \sum_{j=1}^{L} w_{ij} = \alpha_i, 1 \leq i \leq K$$

Finding the optimal weights w_{ij} is a linear programming problem and there are many algorithms for solving it such as simplex tableau. In this parametric distance metric the distance metric between the components is not fixed so the same framework can therefore be used with different kind of mixture probability density functions. It is also shown that if the component distance measure $d(p_i, q_j)$ satisfies the three properties of a distance measure the parametric distance measure $d_M(p, q)$ satisfies them too.

A distance measure which can be used when the Gaussian mixtures $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ use diagonal covariance matrices is

$$d_{L^r}(p, q) = \Big(\sum_{i=1}^{N} |\mu_i^p - \mu_i^q|^r + \sum_{i=1}^{N} |\sigma_i^p - \sigma_i^q|^r \Big)^{1/r} \tag{3.37}$$

where N is the number of features, μ_i is the mean and σ_i is the corresponding i:th variance. The usual values for r are 1 and 2. This distance measure satisfies the three properties of a distance measure. It should be noticed that in $d_{L^r}(p, q)$ mean and variance are treated equally though their dynamic range may be very different.

Although the parametric distance measure was built for mixture probability density functions it does not restrain us from using the same concept with different types of mixture models. The prototype vectors of the neural methods can be seen as a mixture model. A distance measure between two prototype vector collections is

$$d_{L^r}(\boldsymbol{m}, \boldsymbol{n}) = \Big(\sum_{i=1}^{N} |m_i - n_i|^r \Big)^{1/r} \tag{3.38}$$

where $\boldsymbol{m}$ and $\boldsymbol{n}$ are prototype vectors, N is the number of features and m_i and n_i are the i:th components of vectors. The usual values for r are 1 and 2.

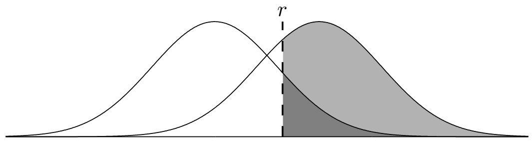
3.5 Receiver operating characteristic curve

The performance of a binary classifier can be visualized with a receiver operating characteristic (ROC) curve [44]. The curve visualizes the trade-off between positively classified samples and false positively classified ones.

If the normal and the abnormal classes are one dimensional normal distributions we can write the definition of the curve as follows:

$$\mathrm{ROC} = \left\{ (u, v) \,\middle|\, u = \int_r^\infty p_{\mathrm{normal}}(x)\,\mathrm{d}x ; v = \int_r^\infty p_{\mathrm{abnormal}}(x)\,\mathrm{d}x \right\} \tag{3.39}$$

where r is the decision threshold of the abnormal class defined by a detector, p_{normal} and p_{abnormal} are the probability distributions of the normal and the



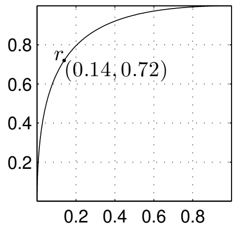


Figure 3.9: The densities of a variable are shown on the left figure. The left density is of the normal class and the right one is of the abnormal class. The dashed line at r is the decision threshold. The light gray shows the probability of detecting the abnormal class and the dark gray shows the probability of false detection. On the right figure is the ROC curve of the decision. The decision point r that is marked into the figures corresponds with a false positive detection probability of 0.14 and a positive detection probability of 0.72.

abnormal class. By modifying the threshold variable r of the detector we get a set of points (u, v) for the ROC curve. A ROC curve of Gaussian classes is shown in Figure 3.9.

ROC curves have some nice properties. They are indifferent to the sample ratio between the normal and the abnormal class as both classes are evaluated separately. This removes the problem that usually the normal class is dominant in data and samples from the abnormal class are rare. The area under a ROC curve shows how well the detector works in overall. The nearer the curve goes the upper left corner of the plot the better the detector works. The performance is at the lowest when the curve goes near the diagonal. Hanley and McNeil [14] have shown that the area under the ROC curve corresponds to the probability of classifying a random sample into a correct class. In the case of two Gaussian classes the more the density functions overlap the closer the ROC curve will be the diagonal since it becomes harder to classify samples correctly. A downside is that when we want to use ROC curves in performance visualizing we need to know the distribution densities of the classes or the real labels of the data points.

Chapter 4

Experiments and results

Methods introduced in Chapter 3 are tested in this chapter first with synthetic two dimensional data and then with real GSM network data. The purpose of using the synthetic data is to compare the methods with well known data and to illustrate the differences between the methods. Applicability of the methods to a real use is tested with the GSM network data introduced in Chapter 2.

4.1 Synthetic data

In this section the performance of the methods will be analyzed using synthetic data where the labels of the data points are know. The data is two dimensional for the comfort of visualization. The purpose of this testing is to illustrate with well controlled data the different properties these methods have. Tests were run with single cluster data and with multiple cluster data. The performance of the detectors was visualized with a receiver operating characteristic curve. The single cluster data represent those cases where we know that the data contains only one cluster but the shape of the cluster is strange. Many mobile network related performance indicators are such that the data is single clustered. The multiple cluster data is for testing the outlier detector′s ability to adapt for multiple clusters. In both cases parameters for the detectors were kept the same for the sake of good generalization properties.

4.1.1 Single cluster data

With the single cluster data the idea is to experiment how the outlier detectors handle data that contain only one peculiar shape cluster. The data was generated with following steps:

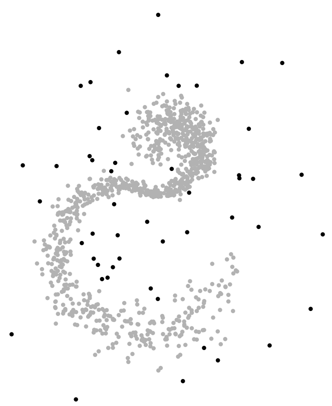


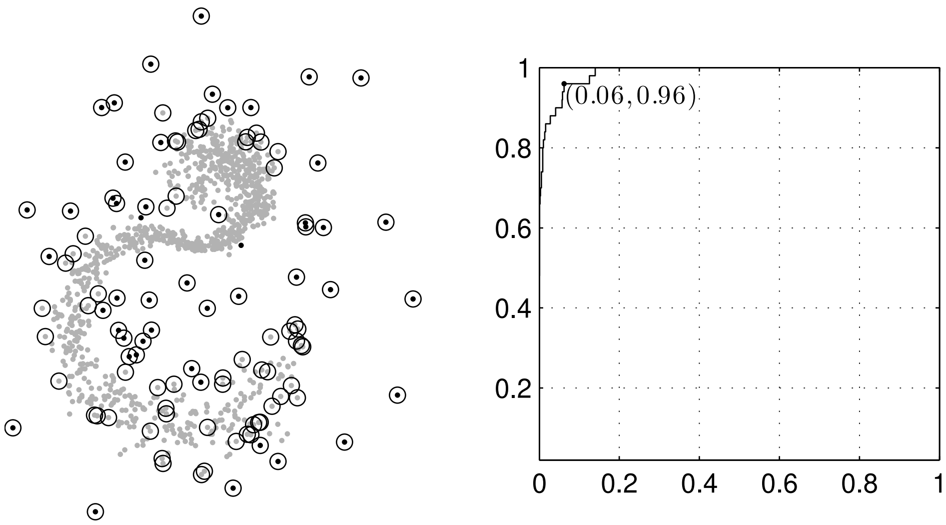
Figure 4.1: Single cluster data. Grey points are from normal class and black ones are from abnormal class.

1. Normal class data points were generated uniformly from an S-shape line added with noise from normal distribution. The variance of the noise varies along the line. The noise is largest in the ends of the line.

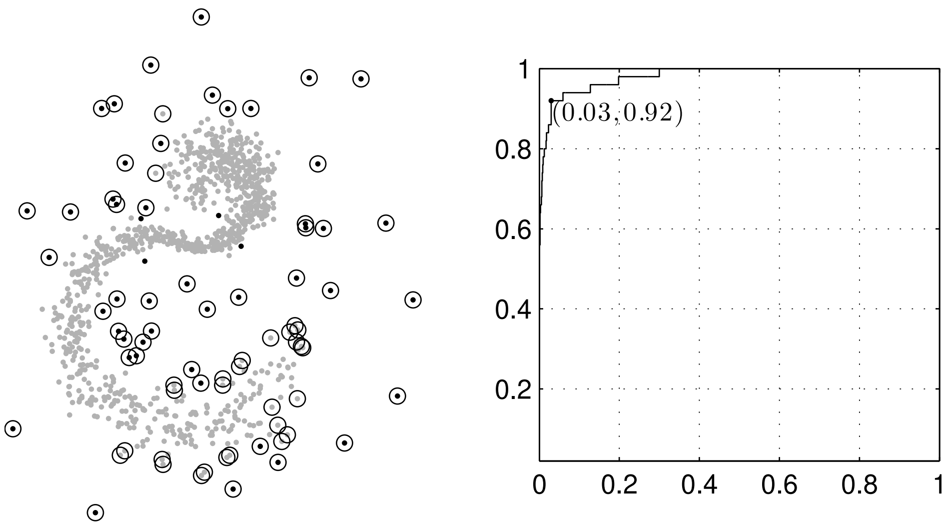
2. Abnormal data points were added to the data using normal distribution. To prevent the overlapping of the normal and the abnormal class those abnormal points which were closer to a normal data point than any other normal data point were removed.

The number of abnormal data points was set to 5 percent. Using the process described above 1000 samples were generated. The data is shown in Figure 4.1. The results of the outlier detectors are shown in Figure 4.2 and Figure 4.3. A scatter plot and a ROC curve is drawn for each detector. The scatter plot is there to show visually the differences between the methods and the ROC curve is there to show the overall performance of the detectors. The thresholds for the abnormal class in the scatter plots is selected by choosing a point from the ROC curve where P(positive detection probability)$(1 - P$(false positive detection probability)$)$ is biggest. The overall areas under the ROC curves are shown in Table 4.1.

In neural gas outlier detection method the prototype vectors try to settle down evenly according to the data points. These prototype vectors are used for the classification of abnormal data points. The number of prototype vectors was initially set to 40 but the method selected only 34 of them for the normal class. The neural gas method has difficulties when there are density variations in the data set. The threshold for outliers is globally constant and does not depend on the local density of the data points near a prototype. The effect of this behavior can be seen in Figure 4.2(a). The

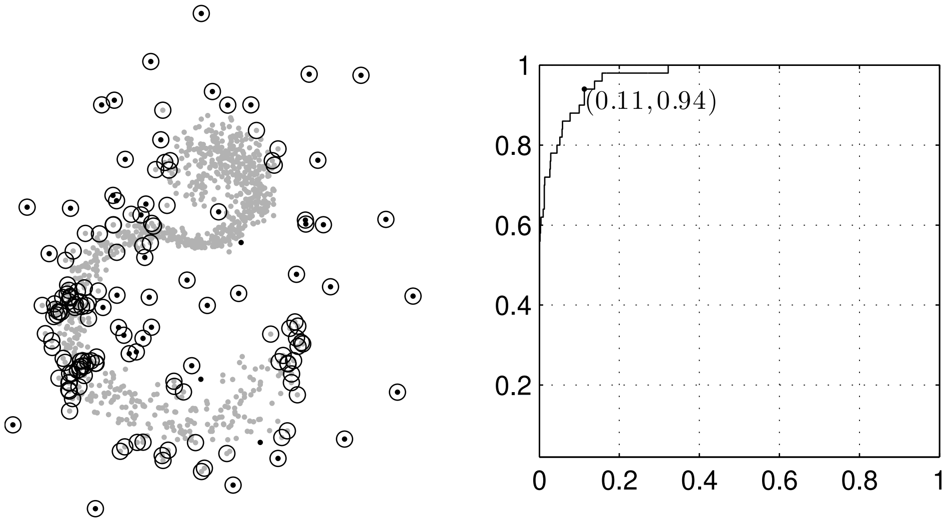


(a) Neural gas

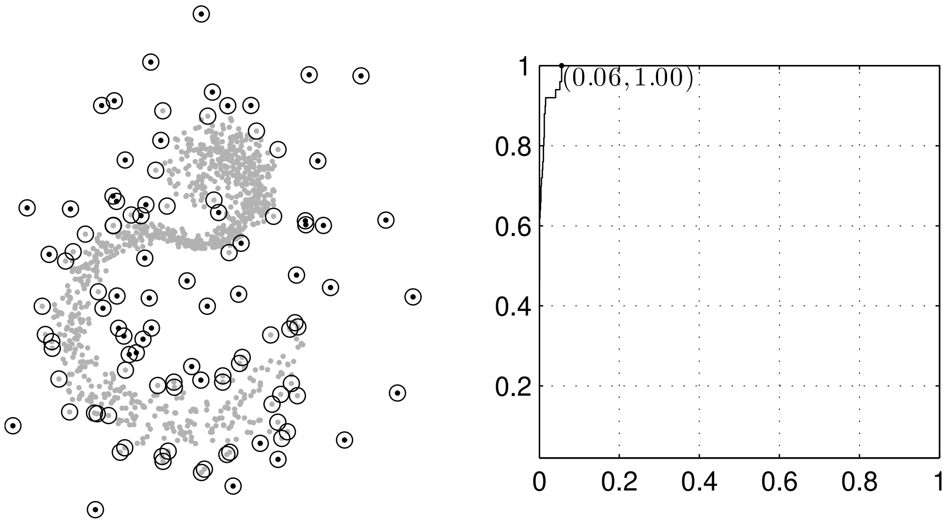


(b) Nearest neighbors

Figure 4.2: Single cluster outlier detectors. On the left side are shown the scatter plots of the classifying. Outliers are marked with circles. In the plots the outliers were selected according to the ROC curve. On the right side are the ROC curves.



(a) Gaussian mixture model



(b) Parzen windows

Figure 4.3: Single cluster outlier detectors. On the left side are shown the scatter plots of the classifying. Outliers are marked with circles. In the plots the outliers were selected according to the ROC curve. On the right side are the ROC curves.

Table 4.1: Area under the ROC curve with single and multiple cluster data

	Single	Multiple
Neural gas	0.988	0.950
Nearest neighbor	0.982	0.974
Gaussian mixture model	0.972	0.969
Parzen windows	0.993	0.974

tails of the S-shape cluster are modeled inferior to the central part where the density of the data points is higher. The prototype vectors do not cover the sparse regions well enough and there are gaps in the normal class between the prototype vectors.

Also the nearest neighbor method suffers from the same problem as the neural gas method that is the global constant threshold which does not depend on local densities. But nearest neighbor method has an advantage it covers much better the whole data set because it does not do any kind of information reduction. The global threshold problem can be seen in Figure 4.2(b) some of the outliers get undetected on the higher density area because the threshold distance must be set so that it works on the lower density areas too.

The Gaussian mixture model method tries to model the probability density of the data with a mixture of Gaussians. The initial number of Gaussians was set to 40 but the method ended up to using 39 of them. The use of probability density function should help with the problem of global threshold for outliers. But the problem in this method seems to be that the fitting of Gaussian to the data is not very successful. In Figure 4.3(a) we can see that the method breaks the normal class into parts.

With Parzen windows method a better estimate for the probability density is build because all the data points get similar attention and none of the information is lost. The Parzen windows method improves the nearest neighbors method by taking the local densities of the data into account. The classification results are shown in Figure 4.3(b).

4.1.2 Multiple cluster data

In the multiple cluster case the idea is to experiment how the outlier detectors manage data that contains multiple abnormal shaped clusters and to examine if there are differences to the single cluster case. The data was generated with following steps:

1. Normal class data points were generated uniformly from an S-shape

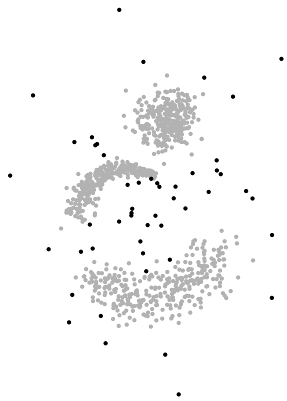


Figure 4.4: Multiple cluster data. Grey points are from normal class and black ones are from abnormal class.

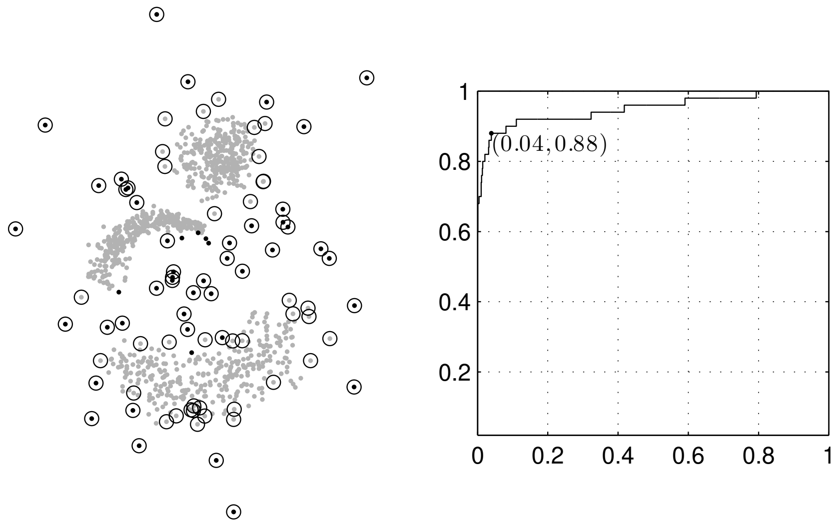
line added with noise from a normal distribution. The S-shape line was cut into three pieces so that the normal class contained three separate clusters. The variance of the noise varies along the line. In the ends of the line the noise is largest.

2. Abnormal data points were added to the data using normal distribution. To prevent the overlapping of the normal and the abnormal class those abnormal points which were closer to a normal data point than any other normal data point were removed.

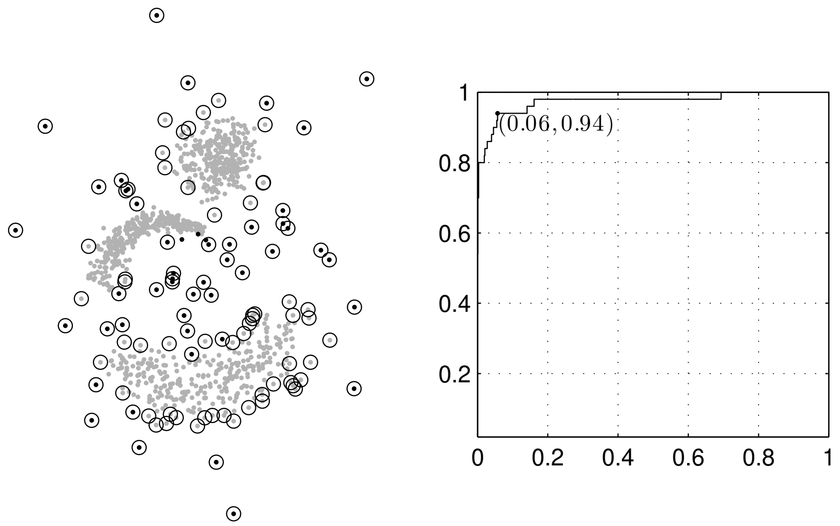
The number of abnormal data points was set to 5 percent. Using the process described above 1000 samples were generated. The data is shown in Figure 4.4. The results of the outlier detectors are shown in Figure 4.5 and Figure 4.6. A scatter plot and a ROC curve is drawn for each detector. The scatter plot is there to show visually the differences between the methods and the ROC is there to show the overall performance of the detectors. The thresholds for the abnormal class in the scatter plots is selected by choosing a point from the ROC curve where $P(\text{positive detection probability}) * (1 - P(\text{false positive detection probability}))$ is biggest. The areas under the ROC curves are shown in Table 4.1.

The initial number of prototype vectors in the neural gas was set to 40 but the method selected only 33 of them for the normal class. The neural gas method has again problems with varying densities. This can be seen in Figure 4.5(a). There seem to be gaps between prototype vectors in the lower sparse cluster. When compared to single cluster case the result is quite similar.

The Gaussian mixture model method handles the multiple cluster data better than it did with the single cluster data. This could arise from the

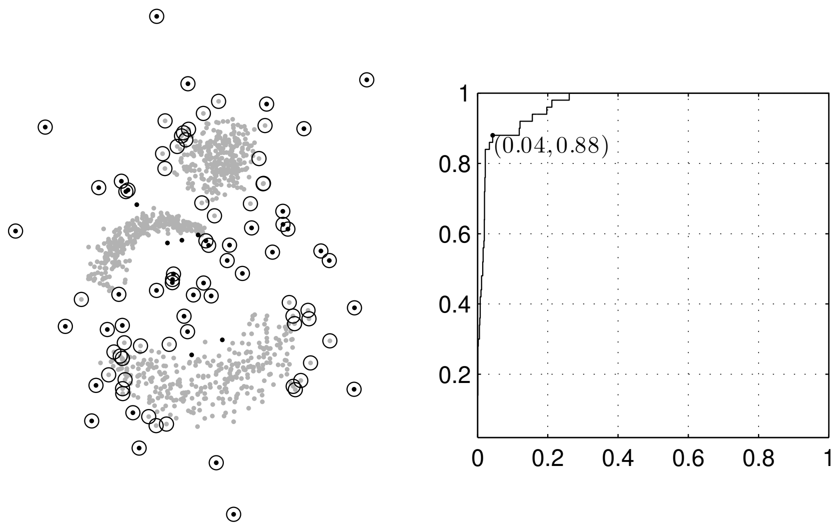


(a) Neural gas

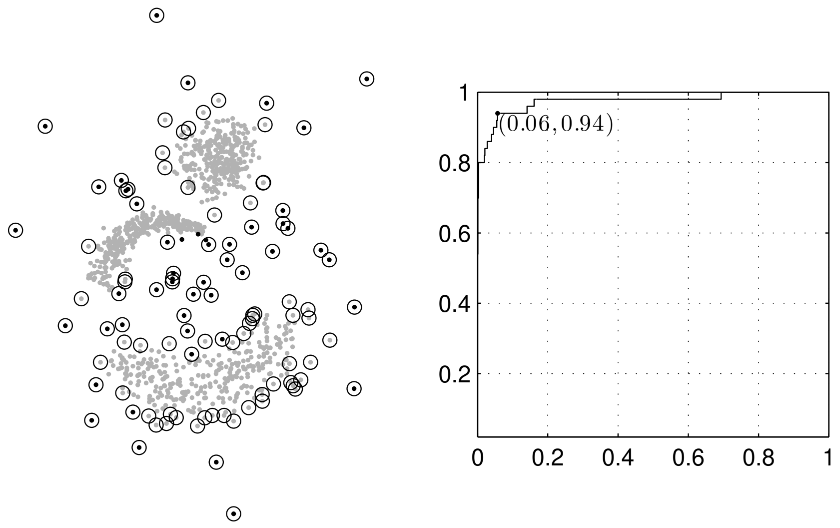


(b) Nearest neighbors

Figure 4.5: Multiple cluster outlier detectors. On the left side are shown the scatter plots of the classifying. Outliers are marked with circles. In the plots the outliers were selected according to the ROC curve. On the right side are the ROC curves.



(a) Gaussian mixture model



(b) Parzen windows

Figure 4.6: Multiple cluster outlier detectors. On the left side are shown the scatter plots of the classifying. Outliers are marked with circles. In the plots the outliers were selected according to the ROC curve. On the right side are the ROC curves.

fact that the clusters are this time nearer to convex clusters which are easier to model with Gaussians. The method used 36 Gaussians of the initial 40. The results of the Gaussian mixture model detector are represented in Figure 4.5(b).

The results of the nearest neighbors detector (Figure 4.6(a)) and the Parzen windows detector (Figure 4.6(b)) are similar to the single cluster case.

4.1.3 Conclusion

The parameters for the methods were not tuned up for the data. Some assumptions had to be made like how many prototype vectors or Gaussians would be enough for the data in the neural gas method or in the Gaussian mixture model method, which neighbor should be used in the nearest neighbor method or in the Parzen windows method and how smooth should the probability density function be in the Parzen windows method. As general purpose values as possible were chosen for the parameters of the methods because the idea was to study how well the methods work if there isn't much prior knowledge of the data.

The Parzen windows method gives clearly the best results of the four tested outlier detection methods. The major drawback it has is that it is computationally very demanding. The computational requirements grow exponential to the size of the data set. Although the test data set was small for the comfort of visualization the Parzen windows method was the slowest methods and the gap to others would become wider if the size of the data set was grown. Though the requirements of the nearest neighbors method also grow exponentially it is still much faster with small data sets than the Parzen windows method because it is much more simple. In the Parzen windows method in addition to the same distance calculations as in the nearest neighbor method also the values of the Gaussians must be calculated.

The idea in the neural gas method and the Gaussian mixture method is to decrease computational requirements by approximating the data set with a model that has less free variables than the data set. This can not be done so well that none of the information is lost and this explains why their results are usually inferior to the Parzen windows method and the nearest neighbors method. In the neural gas method the computational requirements grow linearly to the size of the data set. The learning time of the Gaussian mixture model method is more random and depends on the initialization. The randomness in these method is another drawback because you can never be sure how good the result is. This is why these methods should be run several times with different initializations.

4.2 GSM network data

The GSM network data is build of counter values. These counters are from different events in the network for each BTS. Samples are taken hourly from these counters. A smaller number of key performance indicators are calculated from these counter samples. There were 9 KPIs which were introduced in Section 2.3.1. All the KPIs were treated with same importance because there were no prior expert knowledge of how the KPIs should have been weighted. 92 BTSs were selected for these experiments and the overall number of data samples was then 66858. Although the data contain time information it wasn't used in these experiments.

Two different outlier detection tasks were examined, the classification of the outlier samples inside a BTS and the detection of outlier BTSs. In the task of detecting outliers inside a BTS the BTSs were examined separately. The main idea in the second task, the detection of outlier BTSs, was to calculate distances between the BTSs and use that data to detect outliers or to cluster the BTSs.

4.2.1 BTS-specific outlier detection

The data of one BTS is examined separately from the other BTSs. The same methods which were used with the synthetic data in Section 4.1 can be used with this data. Any higher level knowledge of the data like time series properties or prior information of the features is not taken into account. This kind of outlier detection that handles one BTS at a time can be used to preprocess the data for other methods or to find possibly interesting data samples for further study.

The data was first preprocessed by subtracting the mean and normalizing the variances of the separate key performance indicators. After these the principal component analysis [20] was used to find the main components in the data. These components were fed to the outlier detectors. Seven of the nine main components of the data are plotted in Figure 4.7. Two of the components were dropped from the figure because their magnitude was considerable smaller than the other's.

4.2.2 Inter-BTS outlier detection

The idea in this experiment is to build a method which calculates an outlier value for each BTS. This outlier value should tell us how far the BTS is from the other BTSs. The outlier value for each BTS was calculated using the Parzen windows method introduced in Sections 3.2 and 3.2.1. To calculate the values of the density function that is built with the Parzen windows

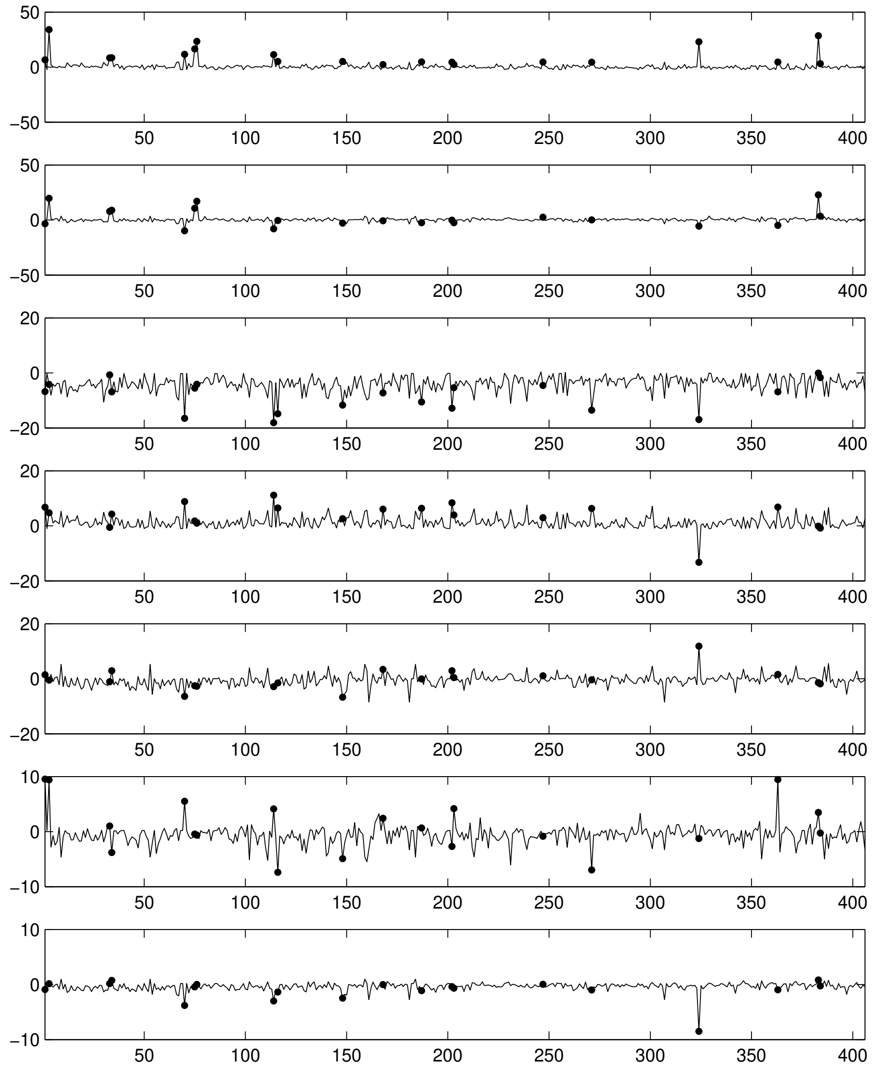


Figure 4.7: Outliers in the data of a BTS. Samples classified as outliers are marked with dots. For the visualization of the plots 5 percentage of the samples were selected to be outliers.

we only need to know the distances between the BTSs. Also a similar but little simpler nearest neighbor method introduced in Section 3.1 can be used when determining the outlier values for the BTSs.

The main method to build a distance matrix between the BTSs can be represented with following steps:

1. Preprocess the data.
2. Teach a model for each BTS.
3. Calculate the distances between the BTSs using the models.

Where the model for a BTS can for example be a density function or vector quantization compression of the data. Also the raw data is a valid model for a BTS.

In the preprocessing step the mean of the data vectors is subtracted and the variance of each feature is normalized. So each feature is considered equally important. After these the principal component analysis [20] is used to find the main components in the data. Here is a chance to reduce the dimensionality of the data by dropping those components whose eigenvalues are the smallest.

Ways to calculate a distance between two data sets which in this case are the data from two BTSs are presented in Section 3.4. Five different methods were tested, nearest neighbors, neural gas, Gaussian mixture model with diagonal covariance matrix, Gaussian mixture model with full covariance matrix and Parzen windows. The parametric distance metric presented in Section 3.4.3 is used with the neural gas method and the GMM with diagonal covariance matrix method. With the GMM with full covariance matrix method and the Parzen windows method importance sampling is used. In the importance sampling samples are taken from the data of the BTSs.

The outlier values of the BTSs are shown in Figure 4.8. There is plotted a bar graph of the outlier values for each method. In each graph there is a bar for every BTS. Bigger bar means that the BTS is more likely an outlier. On the rows the bars and the BTSs are sorted according to a different method. This gives us a view to compare the results of the methods. On the last row the bars are sorted according to a committee of all the methods. In the committee the outlier values which are given by each method are ranked and the place numbers in the ranking are then used in combining the results. Each method gives a place number for a BTS. The most distant BTS gets the highest number. The place numbers are then summed together to get combined distances for the BTSs.

The nearest neighbors method and the Parzen windows method give very similar results which is understandable because the methods are fundamentally quite similar though they look externally quite different. The

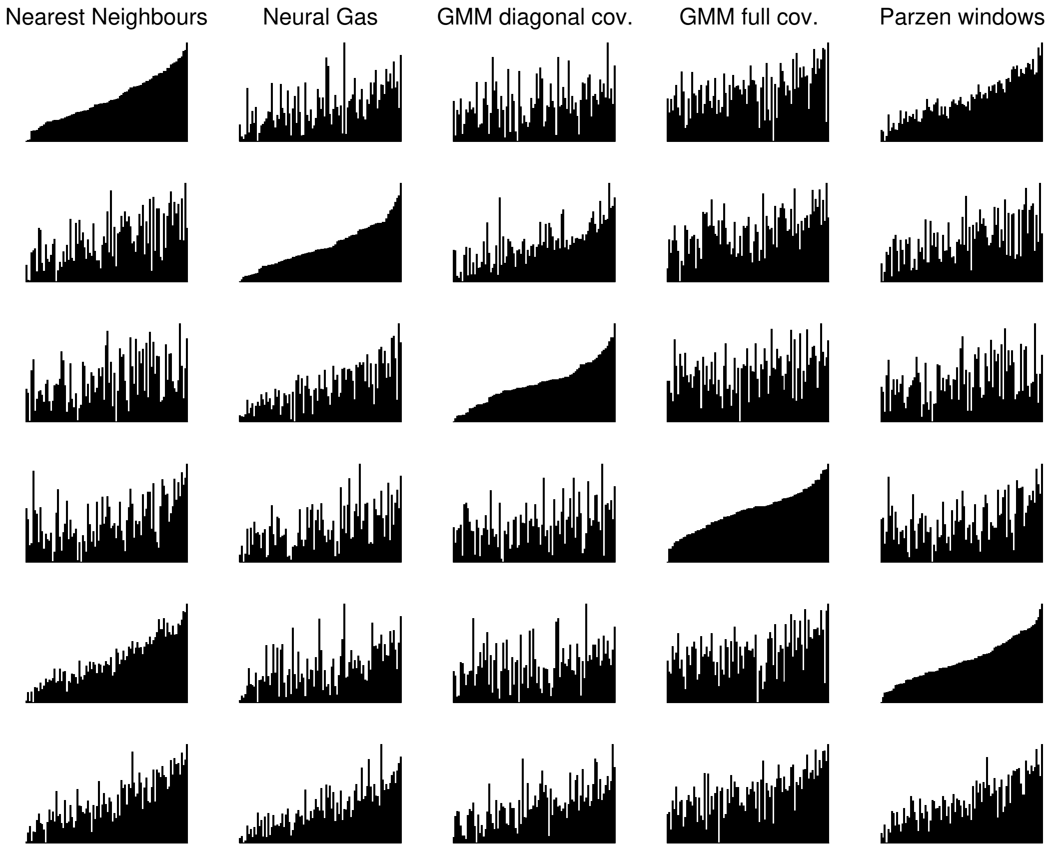


Figure 4.8: Bar graphs of outlier values. On every row there is one bar graph per method. The outlier values of the BTSs are plotted in these graphs with vertical bars. The BTS which has the longest bar in a graph is the most likely outlier in the graph. On a row the order of the BTSs is the same in all the graphs. The order of the BTSs is sorted according to a different method on each row. On the last row the sorting is done according to a committee of all the methods.

overall performance in calculating the distances between the BTSs seems to be the best compared to other tested methods. These methods are the only ones which give the same result on each run. They are based only on the data of the BTSs and do not use any kind of random training.

The results of the neural gas method and the Gaussian mixture model with diagonal covariance matrix are also quite similar. In these the distance measure between the BTSs is alike. In these the step where the distances between the BTSs are calculated is also the fastest of the tested methods because the data of the BTSs is transformed into a compressed models and only the models are used in the calculation of the distances between the BTSs. The average training time of a neural gas network is slower compared to the GMM with a small number of data like in this test but the training time of the GMM varies quite much.

The Gaussian mixture model with full covariance matrix method tries to model the BTSs with probability density functions like the Parzen windows method but the final results are quite different. The Gaussian mixture model tries to compress the data but there may happen some unfavorable things in the process. The basic method for finding the parameters of the GMM does not guarantee that the probability density functions are smooth like with the Parzen windows. The model may overfit some parts of the data while other parts are modeled well. This propagates most to the distances where the density functions of the BTSs are overlapping and least to the cases where the BTSs are more separate so the GMM method can find the outlying BTSs but has problems defining the distances when the BTSs are near each other.

Another way to use the distance matrix and to search outliers is to use clustering. It gives us an overview of the distances between groups of BTSs. A hierarchical agglomerative clustering of the BTSs is given in Appendix B.

4.2.3 Conclusion

When examining the data of a BTS it should be noted that the results of the synthetic data in Section 4.1 hold here also. It is important to think which properties of the methods are crucial for the BTS data. All the methods get acceptable results when finding outliers from the data. Harder part is to decide the proportion of the outliers in the data. This can be done by testing different thresholds for the outliers.

The inter-BTS outlier detection methods give each BTS a value which tells its potential of being an outlier. They don't give any information which properties of the BTSs lead to this conclusion just a number that tells which ones of the BTSs should perhaps be examined more. The results of the tested methods vary a little when calculating the distances between the BTSs. They weight the BTSs differently. The methods which are based on the parametric distance metric (Section 3.4.3), the neural gas method and the Gaussian mixture model with diagonal covariance method, tend to emphasize the most distant BTSs. The Gaussian mixture model with full covariance method which uses the same importance sampling based distance measure as the Parzen windows method finds well only the BTSs which are most likely outliers and tends to scramble the distance of BTSs that are close to each other. The Nearest neighbors method and the Parzen windows method seem to give the most uniform distribution to the distances. All of the tested methods work when unsupervised exploratory outlier analysis of the BTSs is wanted. The methods which are based on the parametric distance metric would become more interesting if the sizes of the data sets were grown as their way of calculating the distances between the BTSs is faster than in the other methods.

Chapter 5

Summary and conclusions

The potential of using outlier detection algorithms for analysis of a GSM network data was studied. Very few assumptions were made from the data and little prior information was used in the analysis of the data. The goal was to use the outlier detectors with generic parameters so that applying the methods for new data would be a straightforward process.

The GSM network data was not pre-labeled and did not contain any information which could have been used for training outlier detectors so only unsupervised methods were tested with the data. Four different types of detectors were studied. The methods were based on nearest neighbors, neural network, Gaussian mixture model and Parzen windows algorithms. Although the bases of the methods are quite different the outlier detectors share many similarities.

When the tested methods are looked through the experimental work done some key points can be said. The parameters are not very sensitive in any of the tested methods. Some common sense should be used when deciding the parameters. For example, in the nearest neighbors algorithm the size of the neighborhood should be chosen so that it is comparable with the size of the whole data set. All the methods could find the most severe outliers. The distinguishing area is the border between the normal class and the abnormal class. The most of the differences in the results originates from there. Some of the methods can also classify normal data as outliers. This happens mostly with the methods which use lossy compression on the data, the neural network vector quantization and the Gaussian mixture model. The number of free parameters in the model can sometimes be set so low that the compression is too high and the model cannot detect outliers reliably or the training has not gone well and the places of the codebook vectors or the Gaussian are not optimal for the outlier detection task.

5.1 Future work

Interesting future work would be to find out how some of the prior knowledge of the data could be included in the studied methods and if this had an effect on how usable the methods are with this extra information. For example the time series properties of the BTS data could be taken into account.

The examined methods give now only a classification for the data. They do not give any clue which features have the most influence to the results. The methods could perhaps be improved so that they would give more information about why the classification was done like it was.

Appendix A

Expectation maximization

In the Gaussian mixture model the unknown parameters $\boldsymbol{\Theta}$ are the means $\boldsymbol{\mu}_i$, the covariances $\boldsymbol{\Sigma}_i$ and the weights w_i. We can assume that each data point $\boldsymbol{x}_j$ comes from one of the distributions in the mixture. This membership can be seen as latent variables $\boldsymbol{L}$ in the model. To start the algorithm some initial values for the parameters are needed. After that the algorithm proceeds by alternating between the expectation and the maximization steps.

Expectation step

The probabilities $P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)$ which indicate the membership of the data point $\boldsymbol{x}_j$ to Gaussian g_i are computed in the expectation step.

$$P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t) = \frac{p(\boldsymbol{x}_j|g_i, \boldsymbol{\Theta}_t)P(g_i|\boldsymbol{\Theta}_t)}{p(\boldsymbol{x}_j|\boldsymbol{\Theta}_t)}$$

$$\begin{aligned}
p(\boldsymbol{x}_j|g_i, \boldsymbol{\Theta}_t) &= N(\boldsymbol{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \\
P(g_i|\boldsymbol{\Theta}_t) &= w_i \\
p(\boldsymbol{x}_j|\boldsymbol{\Theta}_t) &= \textstyle\sum_{k=1}^{K} p(\boldsymbol{x}_j|g_k, \boldsymbol{\Theta}_t)P(g_k|\boldsymbol{\Theta}_t)
\end{aligned}$$

$$P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t) = \frac{N(\boldsymbol{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)w_i}{\sum_{k=1}^{K} N(\boldsymbol{x}_j|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)w_k}$$

Maximization step

Here the expected log-likelihood of the joint event is maximized.

$$\begin{aligned}
q(\boldsymbol{\Theta}) &= E_{\boldsymbol{g}}\left[\ln \textstyle\prod_{j=1}^{N} p(\boldsymbol{x}_j, \boldsymbol{g}|\boldsymbol{\Theta})|\boldsymbol{x}_j\right] \\
&= \textstyle\sum_{j=1}^{N}\sum_{i=1}^{K} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)\ln\big(p(\boldsymbol{x}_j|g_i, \boldsymbol{\Theta})P(g_i|\boldsymbol{\Theta})\big) \\
&= \textstyle\sum_{j=1}^{N}\sum_{i=1}^{K} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)\ln\big(w_i N(\boldsymbol{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\big)
\end{aligned}$$

There is a constraint

$$\sum_{i=1}^{K} w_i = 1$$

which can be added into the expression with a Lagrangian.

$$Q(\boldsymbol{\Theta}) = q(\boldsymbol{\Theta}) - \lambda(\sum_{i=1}^{K} w_i = 1)$$

The new estimated $\boldsymbol{\Theta}_{t+1}$ is found by calculating the maximum of $Q(\boldsymbol{\Theta})$. New estimates for the means:

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\Theta})}{\partial \boldsymbol{\mu}_i} &= 0 \\ \boldsymbol{\mu}_i &= \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)\boldsymbol{x}_j}{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \end{aligned}$$

New estimates for the covariances:

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\Theta})}{\partial \boldsymbol{\Sigma}_i} &= 0 \\ \boldsymbol{\Sigma}_i &= \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \end{aligned}$$

New estimates for the weights:

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\Theta})}{\partial w_i} &= 0 \\ w_i &= \frac{\sum_{j=1}^{N} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)}{\sum_{j=1}^{N} \sum_{i=1}^{K} P(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}_t)} \end{aligned}$$

These are now the new estimate $\boldsymbol{\Theta}_{t+1}$ of the parameters and they are used on the next round.

Appendix B

Dendrograms of BTS distances

The Distances between BTSs can be used also to cluster the BTS. One clustering algorithm that uses the distances is hierarchical agglomerative clustering. At first each BTS is assigned to its own cluster. Then distances between all the clusters are calculated and two nearest ones are merged. The distance d between two clusters C_k and C_l is calculated by using the average linkage:

$$d(C_k, C_l) = \frac{\sum_{i,j} d_{ij}}{N_k N_l},$$

where $i \in C_k$, $j \in C_l$, d_{ij} is the distance between BTS i and BTS j, $k \neq l$ and N_k and N_l are the number of data points in the clusters C_k and C_l. The merging of the clusters is continued until there is only one cluster left which contains all the BTSs. These operations build a clustering tree (dendrogram). The distance between two clusters in a dendrogram is shown by the height of the connecting line. Several different clustering of the data can be made by cutting the tree from different levels.

The dendrograms when using the distance matrices calculated by using the different methods represented in Section 4.2.2 are shown in Figures B.1, B.2, B.3, B.4 and B.5. The widths of the bottom most lines are determined by the distances of the BTSs as calculated in Section 4.2.2. The more distant the BTS is the thinner the line is. The ten most distant BTSs are colored with gray in the dendrograms. The distances used for the coloring are taken from the methods used in Section 4.2.2. The heights of the connecting lines between the clusters in the dendrograms vary from the distances of the BTSs. This stems from the fact that the distance of a BTS tells us more about how far the BTS is altogether from the other BTSs than how well it fits in the group of nearby BTSs.

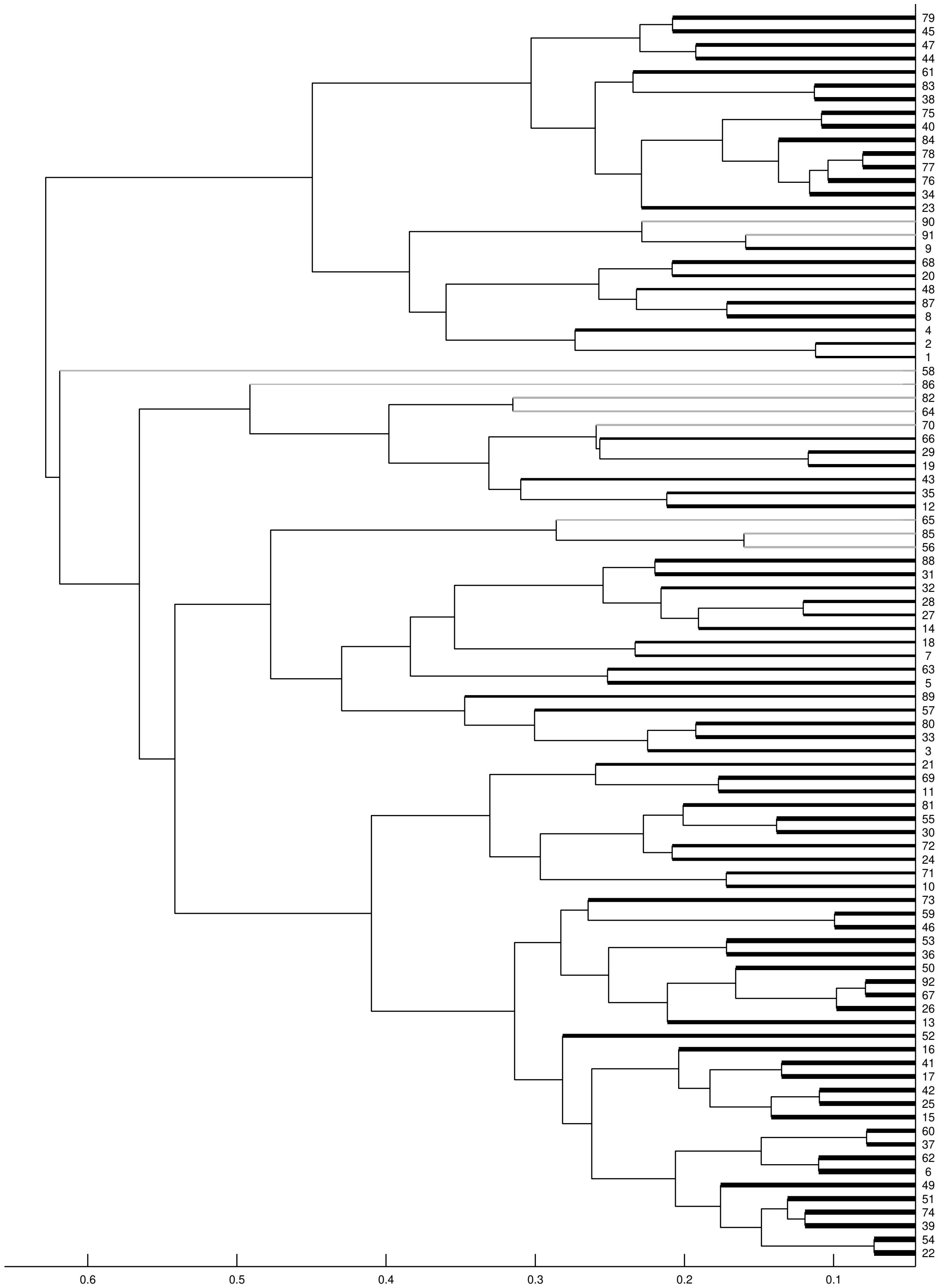


Figure B.1: Nearest neighbors

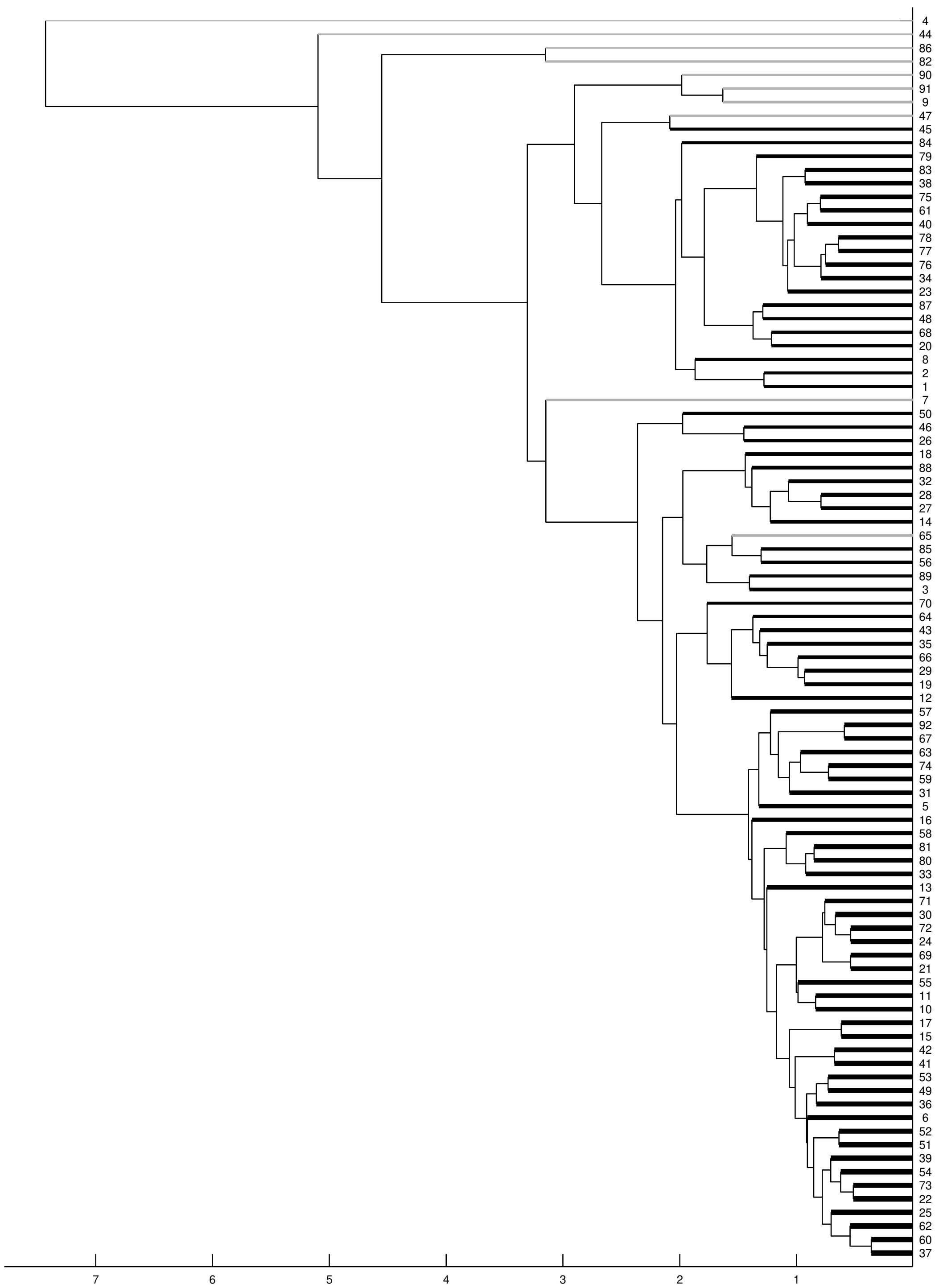


Figure B.2: Parametric distance metric using neural gas

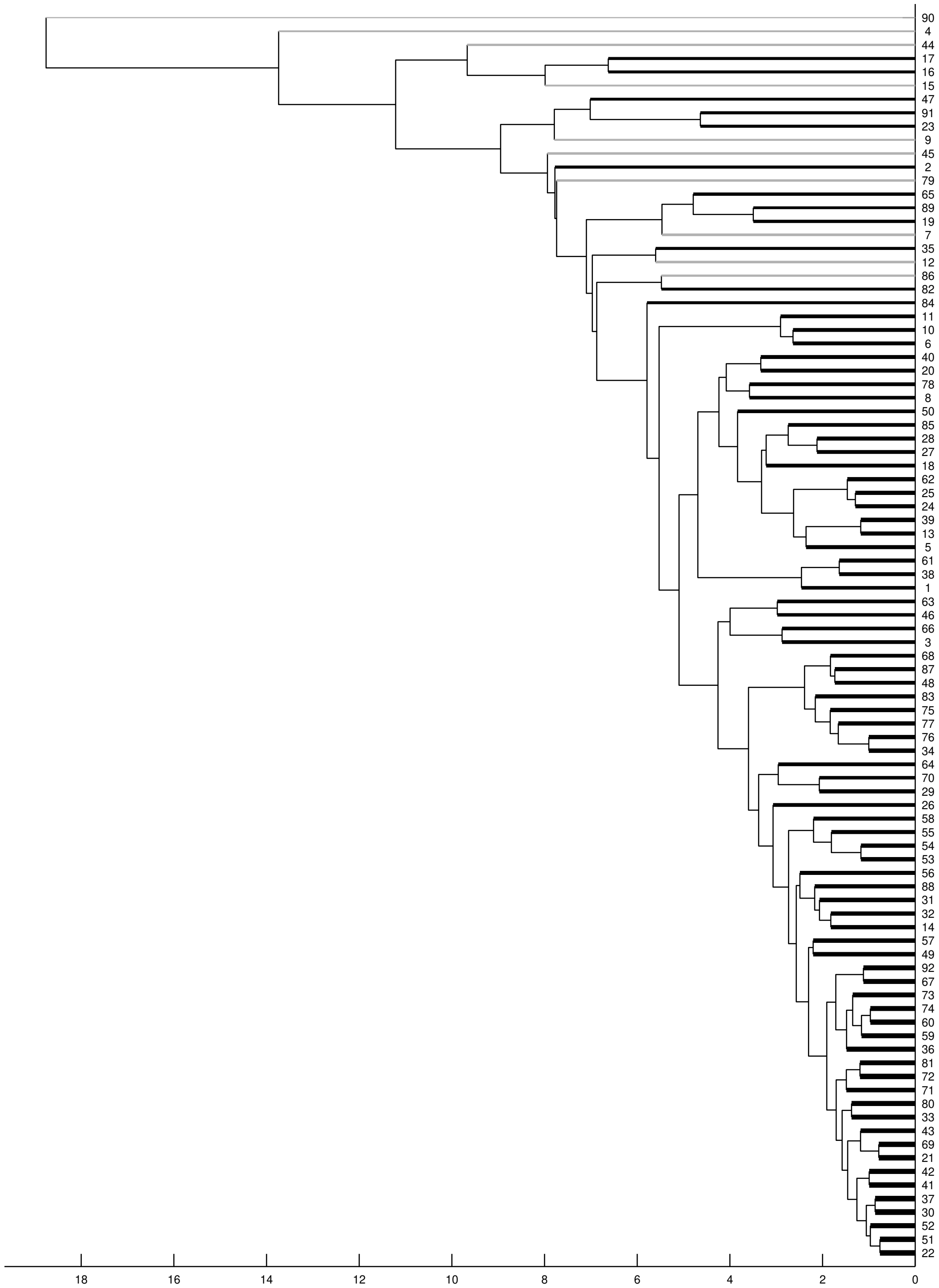


Figure B.3: Parametric distance metric using GMM with diagonal covariance matrix

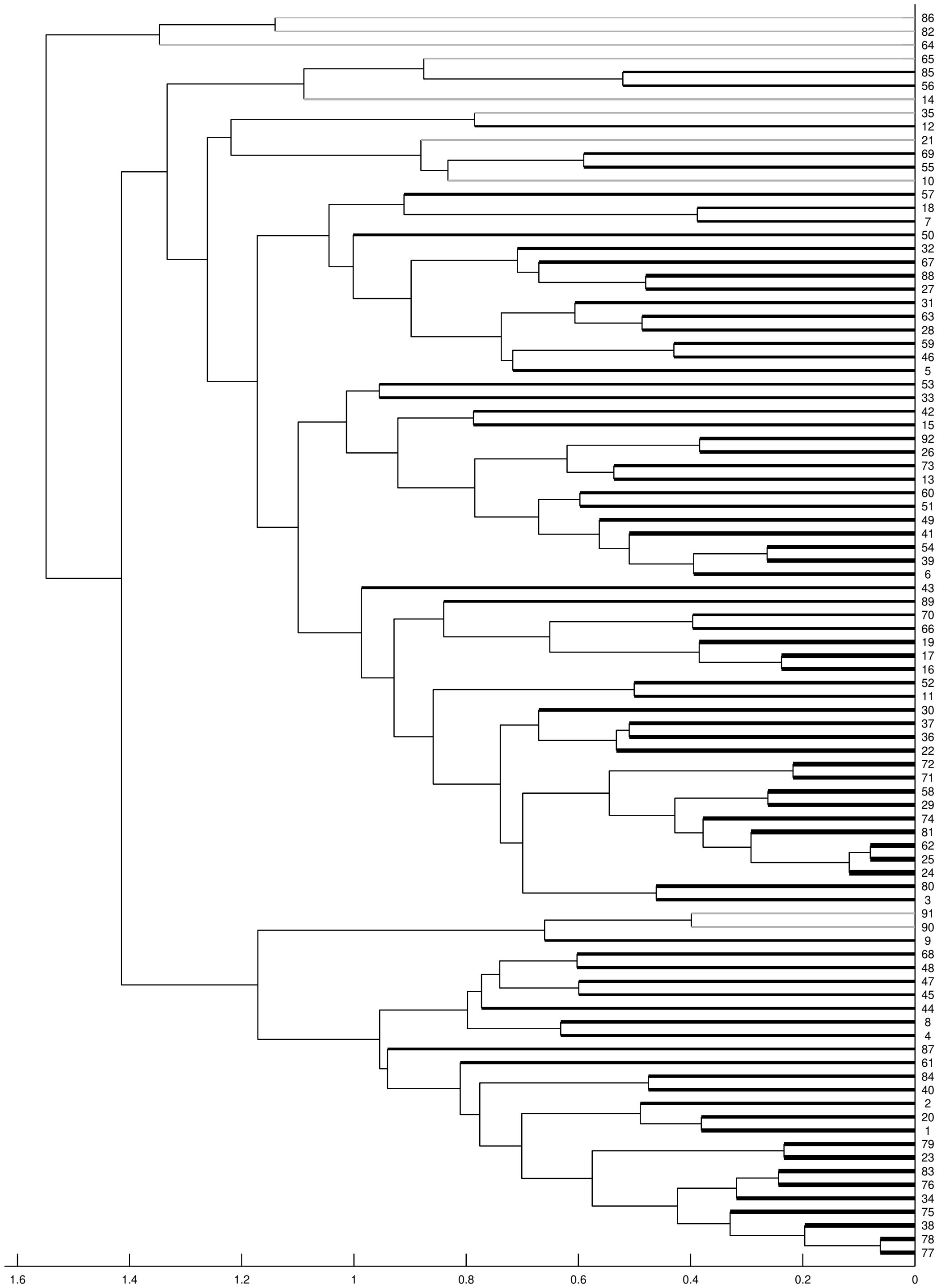


Figure B.4: Data importance sampling using GMM with full covariance matrix

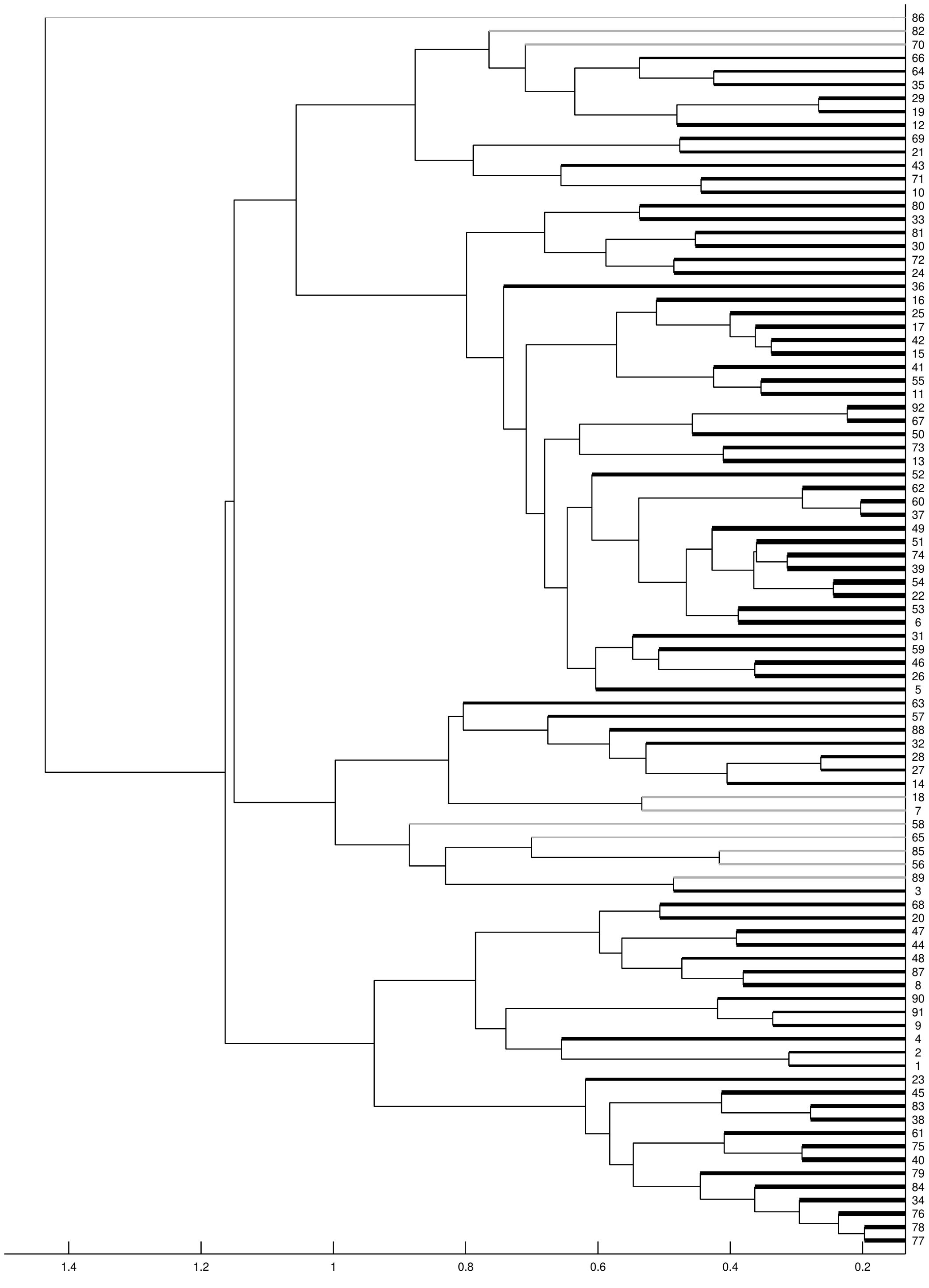


Figure B.5: Data importance sampling using Parzen windows

Bibliography

[1] V. Baez-Monroy and S. O'Keefe. Modelling incremental learning with the batch som training method. *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, 2005.

[2] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley & Sons, 1994.

[3] G. Barreto, J. Mota, L. Souza, R. Frota, and L. Aguayo. Condition monitoring of 3g cellular networks through competitive neural models. *IEEE Transactions on Neural Networks*, 16(5):1064–1075, 2005.

[4] J. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[5] S. Byers and A. Raftery. Nearest neighbor clutter removal for estimating features in spatial point processes. *Journal of the American Statistical Association*, 93(442):577–584, 1998.

[6] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines (and other Kernel-based learning methods)*. Cambridge University Press, 2000.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[8] E. Eskin. Anomaly detection over noisy data using learned probability distributions. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000)*, pages 02–255, 2000.

[9] M. Gales. Semi-tied covariance matrices for hidden markov models. *IEEE Transactions on Speech and Audio Processing*, 7(3), 1999.

[10] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. *NIPS 2004*, pages 13–18, 2004.

[11] R. Gopinath. Maximum likelihood modeling with gaussian distributions for classification. *Proc. ICASSP*, 2, 1998.

[12] F. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[13] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. The MIT Press, 2001.

[14] J. Hanley and B. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982.

[15] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[16] D. Hoaglin, F. Mosteller, and J. Tukey. *Understanding robust and exploratory data analysis*. Wiley New York, 1983.

[17] A. Höglund, K. Hätönen, and A. Sorvari. A computer host-based user anomaly detection system using the self-organizing map. *IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, 5:411–416, 2000.

[18] J. Hollmén. *User Profiling and Classification for Fraud Detection in Mobile Communications Networks*. PhD thesis, Helsinki University of Technology, 2000.

[19] M. Johnson. Cause and effect of telecoms fraud. *Telecommunication (International Edition)*, 30(12):80–84, 1996.

[20] I. Jolliffe. *Principal Component Analysis*. Springer, 2002.

[21] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, 1995.

[22] S. Kyriazakos and G. Karetsos. *Practical radio resource management in wireless systems*. Artech House, 2004.

[23] J. Laiho, M. Kylväjä, and A. Höglund. Utilization of advanced analysis methods in UMTS networks. *55th IEEE Vehicular Technology Conference (VTC) Spring*, 2:726–730, 2002.

[24] P. Lehtimäki and K. Raivio. A knowledge-based model for analyzing gsm network performance. *In Proceedings of The 6th International Symposium on Intelligent Data Analysis*, 2005.

[25] Z. Liu and Q. Huang. A new distance measure for probability distribution function of mixture type. *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '00)*, 1:616–619, 2000.

[26] M. Markou and S. Singh. Novelty detection: a review: part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 2003.

[27] M. Markou and S. Singh. Novelty detection: a review: part 2: neural network based approaches. *Signal Processing*, 83:2499–2521, 2003.

[28] T. Martinetz and K. Schulten. A "neural-gas" network learns topologies. In *Artificial Neural Networks*, pages 397–402, 1991.

[29] M. Mouly and M. Pautet. *The GSM System for Mobile Communications.* Telecom Publishing, 1992.

[30] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368, 1999.

[31] P. Olsen, S. Axelrod, K. Visweswariah, and R. Gopinath. Gaussian mixture modeling with volume preserving nonlinear feature space transforms. *Automatic Speech Recognition and Understanding (ASRU ´03)*, pages 285–290, 2003.

[32] P. Olsen and R. Gopinath. Modeling inverse covariance matrices by basis expansion. *IEEE Transactions on Speech and Audio Processing*, 12(1):37–46, 2004.

[33] D. Ormoneit and V. Tresp. Improved gaussian mixture density estimates using bayesian penalty terms and network averaging. In *Advances in Neural Information Processing Systems*, pages 542–548, 1996.

[34] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438, 2000.

[35] D. Reynolds, T. Quatieri, and R. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1–3):19–41, 2000.

[36] D. Reynolds and R. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3(1):72–83, 1995.

[37] H. Ritter, T. Martinetz, and K. Schulten. *Neuronale Netze*. Addison-Wesley, 1991.

[38] R. Y. Rubinstein. *Simulation and the Monte Carlo Method*. Wiley-Interscience, 1981.

[39] E. Russell, L. Chiang, and R. Braatz. *Fault Detection and Diagnosis in Industrial Systems*. Springer, 2001.

[40] J. Scourias. Overview of the global system for mobile communications. Technical report, Department of Computer Science, University of Waterloo, 1996.

[41] L. Tarassenko, P. hayton, n. Cerneaz, and M. Brady. Novelty detection for the identification of masses in mammograms. *Artificial Neural Networks*, pages 442–447, 1995.

[42] L. Tarassenko, A. Nairac, N. Townsend, and P. Cowley. Novelty detection in jet engines. *IEE Colloquium on Condition Monitoring: Machinery, External Structures and Health (Ref. No. 1999/034)*, pages 4/1–4/5, 1999.

[43] M. Titsias and A. Likas. A probabilistic RBF network for classification. *IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, 4, 2000.

[44] H. L. V. Trees. *Detection, Estimation and Modulation Theory*. John Wiley & Sons, 1967.

[45] N. Ye and Q. Chen. An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems. *Quality and Reliability Engineering International*, 17(2):105–112, 2001.