Faculty of Science and Engineering

Department of Electrical and Computer Engineering

Classification, Novelty Detection and Clustering for Point Pattern Data

Quang Nhat Tran

This thesis is presented for the Degree of

Doctor of Philosophy

of

Curtin University

April 2017

DECLARATION

To the best of my knowledge and belief, this thesis contains no material previously published by any other person, except where due acknowledgment has been made. This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

[illegible]

Quang Nhat Tran
21 April 2017

ACKNOWLEDGMENTS

I would like to deeply thank *my parents* and *sister*, who are always by my side, supporting and encouraging me. I believe that one of the most fortunate things in the world is having a wonderful family like them.

I would also like to express my sincere gratitude to my supervisors, *Professor Ba-Ngu Vo, Professor Dinh Phung,* and *Professor Ba Tuong Vo.* Their great knowledge, dedication and passion in work, not only assisted me a lot in my research, but also strongly inspired me to pursue my ambitions in the future.

I would also like to thank *thay Tran Tien Duc*, my undergraduate supervisor, who inspired and encouraged me to continue with my higher studies. Without his great inspiration and encouragement, it is possible that I would not have gone this far. My sincere thanks also goes to *thay Thuong Nguyen*, who introduced me to my PhD supervisors. Without his kind introduction, I may miss the chance of working with such great supervisors. I would also like to thank *thay Dang Truong Son*, the retired Dean of the Faculty of Information Technology at my previous university, for all his kind support to me which enabled me to pursue this higher study.

I would also like to thank *anh Thin, anh Huy, anh Dang,* and *bac Hai* for their warm welcome and kind help from my very first days in Australia. Their compassion and care are heartwarming and I am extremely grateful.

I would also like to thank my wonderful friends: *anh Gia Hung, anh Duc Hung, chi Vy, chi Linh, chi Thanh (Madeleine), anh Binh, Jeff, Yuthika, Lara, Angel, Francesco, Rajah* and *many others*. They are a huge source of kindness, knowledge and ideas, which have helped and inspired me on countless occasions. I feel so lucky and grateful to have them as my friends.

Last but not least, I gratefully acknowledge the generous financial support

I have received from the *Department of Electrical and Computer Engineering of Curtin University*. I would also like to sincerely thank *Robyn* and *the staff* from the department and university for their kind and professional administrative support. I would also like to thank *Michele* for her very helpful language corrections on my thesis.

Once again, I would like to express my heartfelt gratitude to all of you! I wish all the best to *all*!

ABSTRACT

A point pattern (PP) is a set or multi-set of unordered points, where each point is a (fixed-length) vector representing the state or features of the object of study. PP data are abundant in nature and applications. For example, locations of trees, positions of stars and galaxies, neurons in brain tissue can be considered as PPs. In machine learning, where PP data are more commonly known as bags or multiple instance data, many data sources are indeed PP data or can be represented by PPs, such as point cloud data, transactional data, text data (in bag-of-words representation), images (bag-of-visual-words), audio signals (bag-of-audio-words), and sparse data.

Despite the abundance of PP data, the fundamental machine learning problems for PPs — including classification (supervised learning), novelty detection (semi-supervised learning), and clustering (unsupervised learning) — have not received much attention. In classification, of the three approaches for this learning task, namely the Instance-Space, Embedded-Space, and Bag-Space paradigm, only the Bag-Space paradigm solves PP classification at its fundamental level. The other approaches (i.e., Instance-Space, and Embedded-Space) attempt to convert the PP learning problem into a point (or vector) learning problem, hence the potential for information loss and additional computation. In clustering, only two algorithms have been developed for PP data, namely BAMIC and M^{3}IC, but none of them are model-based, hence they lack the ability to exploit statistical trends in the data, not to mention computational problems with high dimensional inputs and large datasets. For novelty detection, there are no solutions for PP data in the literature.

In this work, we solve three fundamental learning problems, namely classification, novelty detection, and clustering, for PP data using two approaches: one with knowledge of the underlying data model (model-based approach), and one without (distance-based approach). In particular, for model-based classification, we propose to model PP data using point processes and apply the maximum likelihood estimation method to learn the data model. For distance-based classification, we use the Optimal Sub-Pattern Assignment (OSPA) distance with the k-nearest neighbour (k-NN) algorithm to classify PP data.

For model-based novelty detection, the likelihood evaluation of PP data is fundamentally different from that of point data. To this end, we propose a novel ranking function for PP data based on the IID-cluster point process density. This ranking function outperforms the naive Bayes likelihood as well as the IID-cluster point process probability density in PP novelty detection. For distance-based novelty detection, we propose a solution based on the set distance between the candidate PP and its nearest neighbour in the normal training set.

For model-based clustering, we propose to model PP clusters by mixture of point processes. Then the expectation-maximization (EM) technique is applied to learn the mixture and cluster the data. For distance-based clustering, we combine the Affinity Propagation (AP) clustering algorithm with set distances as dissimilarity measures. The proposed learning methods perform well for both simulated and real data.

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AP	affinity propagation
EM	expectation-maximization
F1	F_1 score
i.i.d.	independent and identically distributed
MAP	maximum a posteriori
ML	maximum likelihood
MLE	maximum likelihood estimate
MI	multiple instance
NB	naive Bayes
NN	nearest neighbours
NNN	nearest normal neighbour
NMI	normalized mutual information
OSPA	optimal sub-pattern assignment
PP	point pattern
pdf	probability density function
Pu	purity
RI	rand index
RFS	random finite set

LIST OF NOTATIONS

$\underline{d}$	base distance
p_c	cardinality distribution
$\lvert X\rvert$	cardinality of X
$E[\cdot]$	expected value
p_f	feature density
$d_{\mathtt{H}}$	Hausdorff distance
π_k	k^{th} component weight
$\delta_i[j]$	Kronecker delta (is 1 if $i = j$, and 0 otherwise)
$\lvert\lvert \cdot \rvert\rvert_2$	L_2-norm
y_n	label of n^{th} observation
X_n	n^{th} observation
N_{test}	number of observations in test dataset
N_{train}	number of observations in training dataset
$d_{\mathtt{O}}^{(p,c)}$	OSPA distance of order p, and cutoff c
ξ	parameters of cardinality distribution
φ	parameters of feature distribution
θ_k	parameters of k^{th} component distribution
ψ	parameters of mixture model $\psi \triangleq (\pi_{1:K}, \theta_{1:K})$
$\mathcal{D}_{\text{test}}$	test set
$\mathcal{D}_{\text{train}}$	training dataset
$d_{\mathtt{W}}^{(p)}$	Wasserstein distance of order p

LIST OF PUBLICATIONS

The following papers have been published or will be submitted:

Tran, Q. N., Vo, B.-N., Phung, D., and Vo, B.-T. (2016). Clustering for point pattern data. In *23rd International Conference on Pattern Recognition (ICPR)*, Cancún, México, December 2016. (Chapter 5, and Chapter 6)

Vo, B.-N., **Tran, Q. N.**, Phung, D., and Vo, B.-T. (2016). Model-based classification and novelty detection for point pattern data. In *23rd International Conference on Pattern Recognition (ICPR)*, Cancún, México, December 2016. (Chapter 3, and Chapter 4)

Tran, Q. N., Vo, B.-N., Phung, D., Vo, B.-T., and Nguyen, T. (2017). Multiple instance learning with the optimal sub-pattern assignment metric. (to be submitted) (Chapter 6)

Vo, B.-N., **Tran, Q. N.**, Phung, D., and Vo, B.-T. (2017). Model-based multiple instance learning. (to be submitted) (Chapter 3, Chapter 4, and Chapter 5)

My key contributions in these papers include:

- Performed a comprehensive literature survey of point pattern data sources and applications.
- Performed a comprehensive review of existing methods for learning point pattern data.
- Taking part in developing novel learning methods for point pattern data.
- Code up the proposed point pattern data learning methods in Matlab.
- Preparation of data and evaluation of the performance of the proposed methods.

CHAPTER 1
INTRODUCTION

Chapter's key points:

- Importance of fundamental learning tasks
- Importance of point pattern data
- Problem formulation of point pattern learning
- Example of learning point patterns
  using point learning method
- Contributions of this research
- Structure of this thesis

In this chapter, we first describe the problems of interest — fundamental machine learning problems for point pattern data: what they are and why they matter. Then we summarize our key contributions in solving these problems. Finally, a roadmap of the thesis is given.

1.1 Brief review of machine learning

Letting computers learn from experience without the need for detailed programming (Samuel, 1959) is considered to be one of the first broad descriptions of **machine learning** in its rudimentary stage, as proposed by Arthur Samuel, a pioneer in the field. In 1983, in an attempt to clarify the machine learning concept, Herbert Simon defined **learning** as changes in a system (e.g., an animal, a human, or a machine) which allow the system to perform the same task (or other tasks drawn from the same population) better the next time (Simon, 1983, p. 28).[1] This definition was then stated more formally by Tom Mitchell: "A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997, p. 2).

Nowadays, machine learning has become a vast field of study and development, therefore, it is even more difficult for authors to agree on one definition. Some, such as Ethem Alpaydin[2], retain the definition by Simon (1983) or Mitchell (1997), whereas others prefer a definition which relates to the particular field to which it applies, such as Kevin Murphy, who defined machine learning as a set of automated methods for data analysis (Murphy, 2012). Others neglect the need for a (theoretical or philosophical) definition for machine learning and see machine learning as a practical tool to solve problems of interest, such as Ian Witten and Eibe Frank, who stated that data mining (their field of interest) is involved with machine learning in a practical sense and left the question "What is machine learning?" for philosophers (Witten and Frank, 2005).

Although the debate on the definition of machine learning continues, there is widespread agreement that machine learning has increased in importance in many fields with a plethora of useful applications. In the field of *computer network*, a well-known example is **web search services** (such as Google, Bing, or Baidu) which enable us to find what we want from over 130 trillion web pages on the Internet (Google, 2017). This kind of tool makes use of many machine learning techniques (Boyan et al., 1996; Agichtein et al., 2006; Yoganarasimhan, 2016). Another example is **recommender systems** (Adomavicius and Tuzhilin, 2005; Pazzani and Billsus, 2007) which can recommend things in which we may be interested, e.g., video recommendations on YouTube, Netflix, and product

[1] Interestingly, this definition, to some extent, agrees with a widely accepted definition of human learning in psychology: "Learning is a relatively permanent change in behavior due to experience." (Coon and Mitterer, 2007; Gross, 2014).

[2] In his book, Alpaydin defined machine learning as follows: "Machine learning is programming computers to optimize a performance criterion using example data or past experience" (Alpaydin, 2004, p. 3).

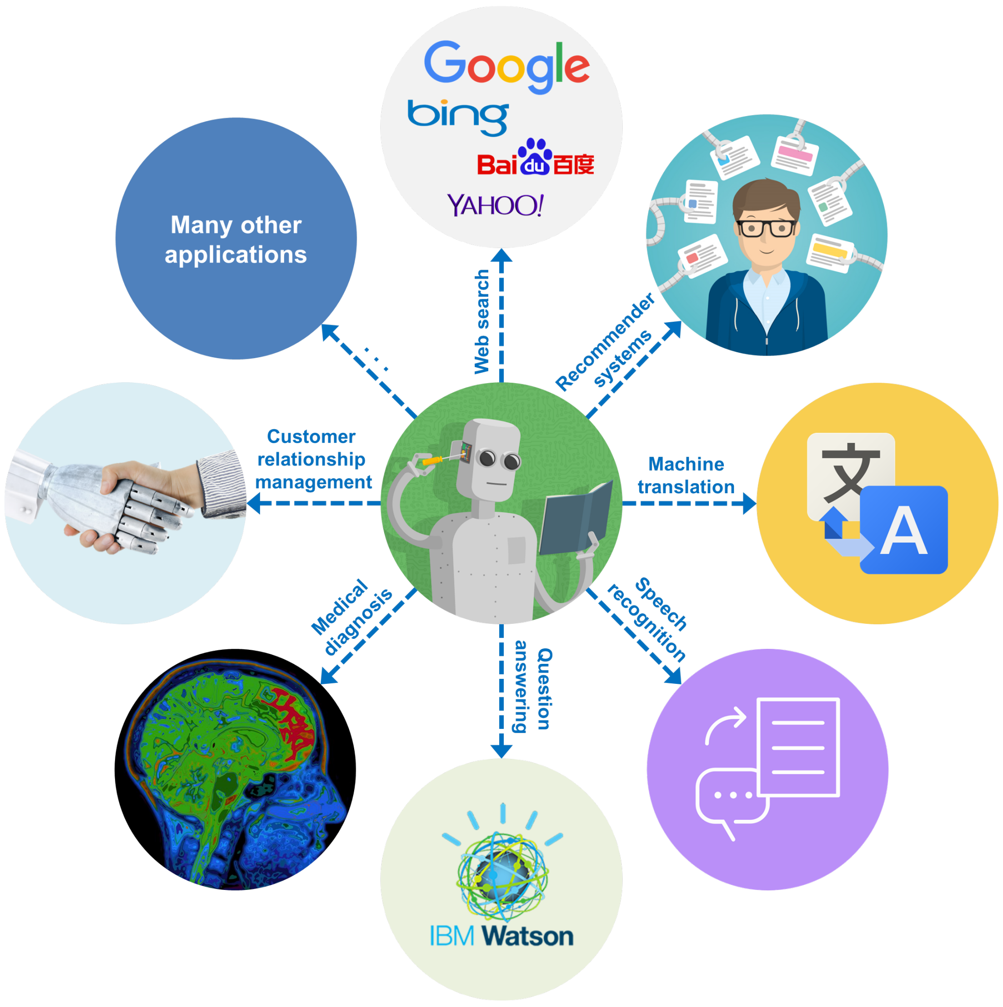


Figure 1.1: Examples of machine learning applications. Image sources: toptal.com, wikimedia.org, logonoid.com, mybluemix.net, watson.devpost.com, medicaldaily.com, economist.com.

suggestions on Amazon, eBay. In contrast, **spam filters** (or **information filters** in general) (Guzella and Caminhas, 2009; Lee et al., 2010) can be considered as information doormen, keeping us away from redundant or unwanted information, e.g., junk email filters in Gmail, Hotmail, social spam filters on Facebook, Twitter.

In *natural language processing*, machine learning is applied to many important tasks, e.g., **speech recognition** which is the task of translating spoken language into text (Hinton et al., 2012), **handwriting recognition** which provides computers with the ability to interpret a handwritten document (LeCun et al., 1989), **machine translation** serving the needs of language translation (Bahdanau et al., 2014), and **question answering** which aims at building systems to answer questions posed in human language (Ferrucci et al., 2010).

Medicine is another field where machine learning is playing an increasingly important role, for example, it is applied in **medical diagnosis** (Kononenko, 2001), **drug design** (Burbidge et al., 2001), and **disease prognosis** (Cruz and Wishart, 2006). In *business*, there are even more machine learning applications, such as in **customer relationship management** (Ngai et al., 2009), **customer segmentation** (Cheng and Chen, 2009), **fraud detection** (Bolton and Hand, 2002), **stock trading** (Kuo et al., 2001), and **online advertising** (Graepel et al., 2010).

The aforementioned is a non-exhaustive list of the applications of machine learning. With the rise of the Internet as well as the growth in data storage and computing power, undoubtedly, machine learning will play an even more important role in many fields in the future. One example of this trend can be seen in the field of *information retrieval*: in the 1990s, people preferred information given by other people rather than by information retrieval systems, however, nowadays most people prefer and are satisfied (most of the time) with information provided by web search engines (Manning et al., 2008).

Although there are a plethora of applications of machine learning, they all are based on or make use of a small number of **fundamental machine learning tasks**. These tasks are also the focus of this research and will be presented in the next section.

1.1.1 Research scope: fundamental machine learning tasks

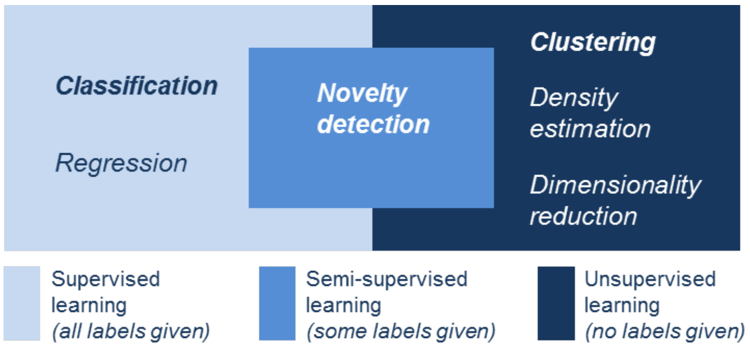

Classification
Regression
Novelty detection
Clustering
Density estimation
Dimensionality reduction
Supervised learning (all labels given)
Semi-supervised learning (some labels given)
Unsupervised learning (no labels given)

Figure 1.2: Fundamental machine learning tasks divided into 3 broad categories: supervised learning, semi-supervised learning, and unsupervised learning. The bold tasks (namely classification, novelty detection, and clustering) are the focus of this research.

The fundamental tasks of machine learning include classification, regression, clustering, density estimation, dimensionality reduction, and novelty detection (Bishop, 2006; Markou and Singh, 2003). The first two tasks, i.e., *classification*, and *regression*, belong to a class of machine learning problems called **supervised learning** which aims to find a function mapping between given inputs and outputs (called *training data* and *labels*, respectively) (Russell and Norvig, 2003). When the outputs are discrete and finite, the task is called **classification**, when the outputs are continuous, it is called **regression** (Bishop, 2006; Murphy, 2012).

The next three tasks, i.e., *clustering*, *density estimation*, and *dimensionality reduction*, are **unsupervised learning** tasks which, as opposed to supervised learning tasks, are not provided with the outputs (Russell and Norvig, 2003). The unsupervised tasks differ in their targets. **Clustering** aims at finding the structure of input data, whereas **density estimation** learns the distribution representing the input data, and **dimensionality reduction** compresses high-dimensional data for the sake of visualization (Bishop, 2006) or feature extraction (Guyon et al., 2008).

The last task, *novelty detection*, belongs to **semi-supervised learning**, which is an area of emerging interest in machine learning (Chapelle et al., 2006). In semi-supervised learning, only a part of the training data is labeled, in other words, it is halfway between unsupervised learning (where labels are completely absent) and supervised learning (where labels are fully observed in training data) (Chapelle et al., 2006). Specifically, in novelty detection, training data consist of only normal data and the task is to infer some 'characterization' of these data which is then used to identify novel data.

In this research, we focus on the most common tasks in three learning categories, namely **classification** in supervised learning, **clustering** in unsupervised learning, and **novelty detection** in semi-supervised learning. These tasks are briefly described in Table 1.1. More details on these learning tasks will be given in Chapter 2.

TASK	TYPE	OBJECTIVE	TRAINING DATA
Classification	Supervised learning	Assign each observation to a category	Labeled data
Clustering	Unsupervised learning	Find groups in data	Non-labeled data
Novelty detection	Semi-supervised learning	Detect novel observations	Normal data only

Table 1.1: Three learning tasks which are the focus of this research.

1.2 Importance of point pattern data

A **point pattern (PP)** is a *set or multi-set*[3] *of unordered points*, where each point is a (fixed-length) vector representing the state or features of the object of study (Moller and Waagepetersen, 2003). Note that PPs is a mathematical term whereas among the machine learning community, terminology such as **bags** (Joachims,

[3] Note that a set does not contain repeated elements while a multi-set can.

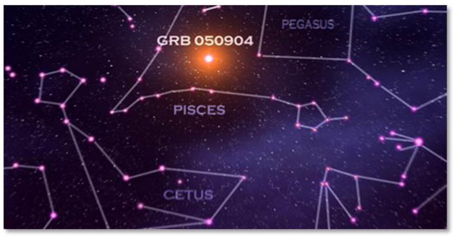
1996; Csurka et al., 2004) and **multiple instance data** (Dietterich et al., 1997; Amores, 2013) are commonly used. *Bag* is a synonym of *multi-set* (Knuth, 1998, p. 694). A *multiple instance* datum is defined as a (multi-)set of instances (Foulds and Frank, 2010; Amores, 2013), where the term *instance* (proposed by Dietterich et al. (1997)) is equivalent to the term *point* in *point pattern*.

PPs are ubiquitous in nature. For example, in the field of statistical ecology, a set of coordinates of objects, such as trees, bird nests in a forest (Figure 1.3a), houses in a city, or people in a crowd, is a PP (Diggle, 1983). Other examples of PPs can be found in various fields, such as the positions of stars and galaxies (Figure 1.3b) (astrostatistics (Babu and Feigelson, 1996)); neurons in brain tissue (Figure 1.3c); blood cells in a haemocytometer square; point-like defects in a silicon crystal wafer (Figure 1.3d) (materials science (Ohser and Mücklich, 2000)); or the home addresses of individuals diagnosed with a rare disease (spatial epidemiology (Elliot et al., 2000)). Example datasets of these types include: significant tree point of Hobart (City-of-Hobart, 2016), hickory trees, Norwegian spruces, weed plants, mucous membrane cells (Moller and Waagepetersen, 2003); amacrine cells, gastroenteric disease (Diggle, 2014); and longleaf pines (Cressie and Wikle, 2011).

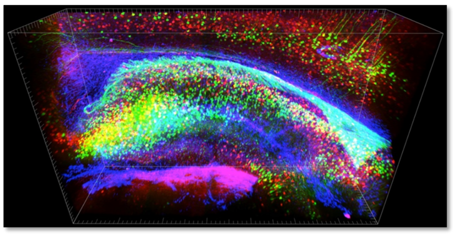
In machine learning, many data sources are indeed PP data or can be represented by PPs. As the first example, **point cloud data** (Figure 1.4) are sets of points mainly used to represent three-dimensional (3D) objects' surfaces (Linsen, 2001). A point cloud is a basic and frequently used format for 3D object representations (Ioannou et al., 2012). With the development of 3D sensing hardware and 3D data processing techniques, 3D perception has become of increasing importance in robotics and other fields (Rusu and Cousins, 2011). There are many point cloud datasets which are publicly available, such as Robotic 3D Scan Repository (Nüchter and Lingemann, 2016), Canadian Planetary Emulation Terrain 3D Mapping Dataset (Tong et al., 2013), Stanford 3D Scanning Repository (Stanford Graphics Laboratory, 2014), ISPRS Test On Extracting DEMs From Point Clouds (ISPRS, 2006). Point cloud data are used in

WEST HOBART
BATTERY POINT
SOUTH HOBART
CASCADE ROAD
LIVERPOOL STREET
DAVEY STREET
BRISBANE STREET

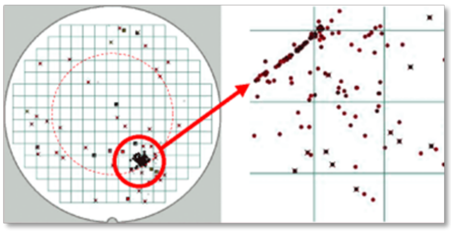
(a) Locations of street trees *(Source: Significant tree point, City of Hobart, Australia, www.data.gov.au)*



(b) Locations of stars *(Image: Constellation Pisces, www.nasa.gov)*



(c) Neurons in a brain *(Image: See-through brains clarify connections, www.nature.com)*



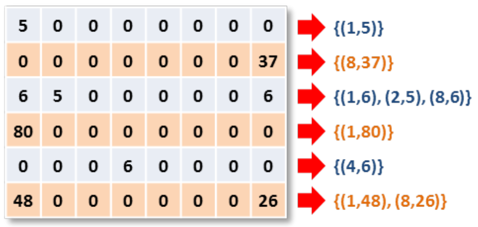
(d) Defects on a wafer *(Image: Defect/Yield Analysis and Metrology, micromagazine.fabtech.org)*

(e) SIFT keypoints of an image *(Photo: A rice field in Vietnam, by Huan Minh)*

(f) Market basket data are sets of transaction items *(Image: www.expeditiongoodlife.com)*

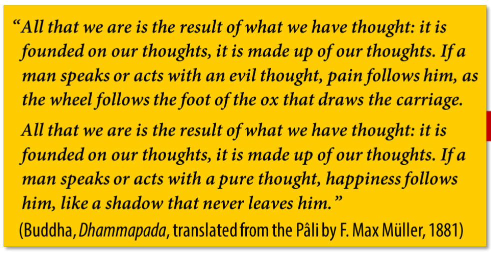
```
Log Type: syslog
Log Length: 4258
Showing 4096 bytes of 4258 total. Click here for the full log.
[main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
2014-06-17 08:45:57,331 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: MapTask metrics system started
2014-06-17 08:45:57,411 INFO [main] org.apache.hadoop.mapred.YarnChild: Executing with tokens:
2014-06-17 08:45:57,412 INFO [main] org.apache.hadoop.mapred.YarnChild: Kind: mapreduce.job, Service: job_1402783580700_0009, Ident: (org.apache.hadoop.ma
2014-06-17 08:45:57,659 INFO [main] org.apache.hadoop.mapred.YarnChild: Sleeping for 0ms before retrying again. Got null now.
2014-06-17 08:45:58,120 INFO [main] org.apache.hadoop.mapred.YarnChild: mapreduce.cluster.local.dir for child: /tmp/hadoop-yarn/nm-local-dir/usercache/kaw
2014-06-17 08:45:59,863 WARN [main] org.apache.hadoop.conf.Configuration: mapred.task.id is deprecated. Instead, use mapreduce.task.attempt.id
2014-06-17 08:45:59,868 WARN [main] org.apache.hadoop.conf.Configuration: mapred.task.is.map is deprecated. Instead, use mapreduce.task.ismap
2014-06-17 08:45:59,868 WARN [main] org.apache.hadoop.conf.Configuration: mapred.tip.id is deprecated. Instead, use mapreduce.task.id
2014-06-17 08:45:59,888 WARN [main] org.apache.hadoop.conf.Configuration: mapred.task.partition is deprecated. Instead, use mapreduce.task.partition
2014-06-17 08:45:59,889 WARN [main] org.apache.hadoop.conf.Configuration: mapred.local.dir is deprecated. Instead, use mapreduce.cluster.local.dir
2014-06-17 08:45:59,901 WARN [main] org.apache.hadoop.conf.Configuration: job.local.dir is deprecated. Instead, use mapreduce.job.local.dir
2014-06-17 08:45:59,901 WARN [main] org.apache.hadoop.conf.Configuration: mapred.job.id is deprecated. Instead, use mapreduce.job.id
2014-06-17 08:46:00,936 WARN [main] org.apache.hadoop.conf.Configuration: session.id is deprecated. Instead, use dfs.metrics.session-id
2014-06-17 08:46:02,041 INFO [main] org.apache.hadoop.mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2014-06-17 08:46:03,542 INFO [main] org.apache.hadoop.mapred.MapTask: Processing split: hdfs://hakunamapdata:8020/user/kawaa/QuasiMonteCarlo_TMP_3_141592
2014-06-17 08:46:03,515 INFO [main] org.apache.hadoop.mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
2014-06-17 08:46:09,816 INFO [main] org.apache.hadoop.mapred.MapTask: (EQUATOR) 0 kvi 52428796(209715184)
2014-06-17 08:46:09,924 INFO [main] org.apache.hadoop.mapred.MapTask: mapreduce.task.io.sort.mb: 200
2014-06-17 08:46:09,924 INFO [main] org.apache.hadoop.mapred.MapTask: soft limit at 167772160
2014-06-17 08:46:09,924 INFO [main] org.apache.hadoop.mapred.MapTask: bufstart = 0; bufvoid = 209715200
2014-06-17 08:46:09,924 INFO [main] org.apache.hadoop.mapred.MapTask: kvstart = 52428796; length = 13107200
2014-06-17 08:46:15,258 INFO [main] org.apache.hadoop.mapred.MapTask: Starting flush of map output
2014-06-17 08:46:15,262 INFO [main] org.apache.hadoop.mapred.MapTask: Spilling map output
2014-06-17 08:46:15,262 INFO [main] org.apache.hadoop.mapred.MapTask: bufstart = 0; bufend = 18; bufvoid = 209715200
2014-06-17 08:46:15,262 INFO [main] org.apache.hadoop.mapred.MapTask: kvstart = 52428796(209715184); kvend = 52428792(209715168); length = 5/13107200
2014-06-17 08:46:15,314 INFO [main] org.apache.hadoop.mapred.MapTask: Finished spill 0
2014-06-17 08:46:15,393 INFO [main] org.apache.hadoop.mapred.Task: Task:attempt_1402783580700_0009_m_000000_0 is done. And is in the process of committing
2014-06-17 08:46:16,473 INFO [main] org.apache.hadoop.mapred.Task: Task 'attempt_1402783580700_0009_m_000000_0' done.
2014-06-17 08:46:16,474 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Stopping MapTask metrics system...
2014-06-17 08:46:16,476 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: MapTask metrics system stopped.
2014-06-17 08:46:16,476 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: MapTask metrics system shutdown complete.
```

(g) Web usage logs are sets of user transactions *(Image: YARN Hadoop clusters, www.ibm.com)*



5	0	0	0	0	0	0	0	{(1,5)}
0	0	0	0	0	0	0	37	{(8,37)}
6	5	0	0	0	0	0	6	{(1,6), (2,5), (8,6)}
80	0	0	0	0	0	0	0	{(1,80)}
0	0	0	6	0	0	0	0	{(4,6)}
48	0	0	0	0	0	0	26	{(1,48), (8,26)}

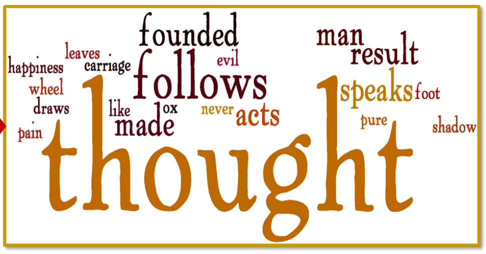
(h) Sparse data can be stored by sets of indices and values.



"All that we are is the result of what we have thought: it is founded on our thoughts, it is made up of our thoughts. If a man speaks or acts with an evil thought, pain follows him, as the wheel follows the foot of the ox that draws the carriage.

All that we are is the result of what we have thought: it is founded on our thoughts, it is made up of our thoughts. If a man speaks or acts with a pure thought, happiness follows him, like a shadow that never leaves him."

(Buddha, *Dhammapada*, translated from the Pâli by F. Max Müller, 1881)



(i) A document can be represented by a set of words.

Figure 1.3: Examples of point pattern data.

various applications, for example: medical imaging (Sitek et al., 2006), environment modeling (Rusu et al., 2008; Rusu, 2010), 3D object benchmarking (Geiger et al., 2012), 3D object recognition (Zhong, 2009; Garstka and Peters, 2015), 3D object retrieval (Mian et al., 2010), reverse engineering (Woo et al., 2002; Schnabel et al., 2007).

The second example of PP data in machine learning is **transactional data**, including market basket data (Figure 1.3f) (Guha et al., 1999; Yang et al., 2002b), web log data (Figure 1.3g) (Cadez et al., 2000; Iváncsy and Vajk, 2006), surveys and contests (Rygielski et al., 2002). Each datum in this data type is a set or multi-set of transaction records, such as sets of shopping items from customers at a supermarket or an online store, sets of requests from a web client, and maintenance data of a web server, etc. With the expeditious growth in data storage and computing power, these data are being increasingly collected and analyzed and are growing in importance in relation to business decision-making processes. Public sources of this data type can be found at, e.g., Retail Market Basket Data (Brijs et al., 1999), Online Retail Data (Chen et al., 2012), UNIX User Data (Aeberhard et al., 1994), Wikipedia Page Traffic Statistics (Skomoroch, 2009). This type of data is used in a wide range of applications, such as product recommendation (Liu and Shih, 2005), customer profiles building (Adomavicius and Tuzhilin, 2001), customer relationship management (Rygielski et al., 2002), marketing communication (Zahay et al., 2004), users' interests extraction (Murata and Saito, 2006), online advertising (Nanopoulos and Manolopoulos, 2000), and server performance improvement (Iváncsy and Vajk, 2006).

A common type of data which can be represented by PPs is documents or **text data**. In particular, a document (or a part of it such as a sentence or a paragraph) can be treated as a collection (i.e., multi-set) of its words (Figure 1.3i). This representation is known as the **bag-of-words** model (Joachims, 1996; McCallum and Nigam, 1998; Maron, 1961). This model can be considered as a special case of a more general model used in language modeling – the n-**gram** model (Cavnar et al., 1994; Fürnkranz, 1998). In the (word-level) n-gram model,

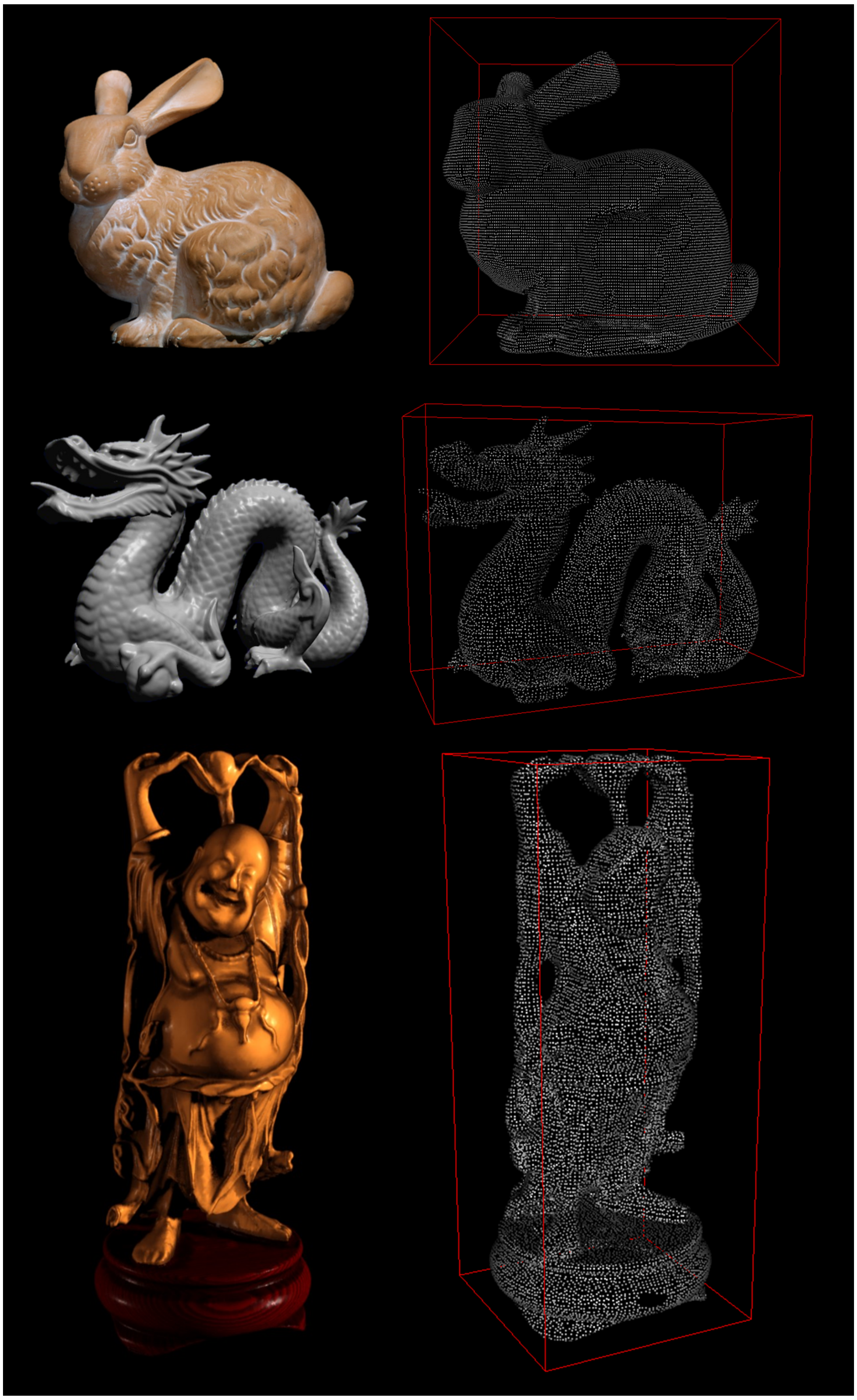
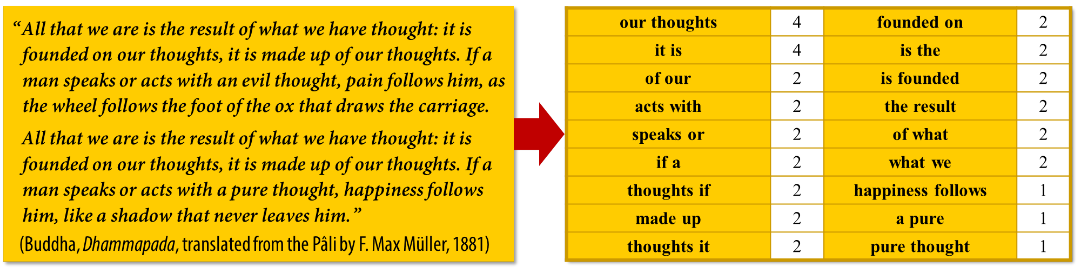


Figure 1.4: Examples point cloud data. Left: real objects, Right: point cloud representation. From top to bottom: Stanford Bunny, Stanford Dragon, Happy Buddha *(Source: Stanford 3D Scanning Repository* (Stanford Graphics Laboratory, 2014)*)*.



"All that we are is the result of what we have thought: it is founded on our thoughts, it is made up of our thoughts. If a man speaks or acts with an evil thought, pain follows him, as the wheel follows the foot of the ox that draws the carriage.

All that we are is the result of what we have thought: it is founded on our thoughts, it is made up of our thoughts. If a man speaks or acts with a pure thought, happiness follows him, like a shadow that never leaves him."

(Buddha, *Dhammapada*, translated from the Pâli by F. Max Müller, 1881)

our thoughts	4	founded on	2
it is	4	is the	2
of our	2	is founded	2
acts with	2	the result	2
speaks or	2	of what	2
if a	2	what we	2
thoughts if	2	happiness follows	1
made up	2	a pure	1
thoughts it	2	pure thought	1

Figure 1.5: Example of 2-gram model. Left: the document; Right: some 2-word phrases and the number of their occurrences in the document (2-gram representation).

a document (or a part of it) is treated as a multi-set of n-word phrases (Figure 1.5). When $n = 1$, the n-gram model is equivalent to the bag-of-words. Examples of bag-of-words and n-gram datasets include: Farm Ads (Mesterharm and Pazzani, 2011), DBWorld e-mails (Filannino, 2011), Amazon Commerce reviews (ZhiLiu, 2011a), Reuter 50 50 (ZhiLiu, 2011b), and Google Books Ngrams (Google, 2016). There are many applications with bag-of-words and n-gram data, including text categorization (Joachims, 1996; McCallum and Nigam, 1998; Cavnar et al., 1994), information retrieval (Ruch et al., 2002), spam filtering (Delany et al., 2005; Cormack et al., 2007), file type categorization (Li et al., 2005), malicious code detection (Abou-Assaleh et al., 2004), authorship attribution (Kešelj et al., 2003), machine translation evaluation (Papineni et al., 2002), and automated summarization evaluation (Lin and Hovy, 2003).

In computer vision, the well-known **bag-of-visual-words** representation — an analogue of bag-of-words — treats **images** as PPs of their key patches (Csurka et al., 2004; Fei-Fei and Perona, 2005). Depending on the application, the key patches can be extracted from images by various feature detection methods, such as edge detection, corner detection, ridge detection, and scale-invariant feature transform (SIFT) (Lowe, 2004). The bag-of-visual-words representation is used in numerous applications, including image categorization (Csurka et al., 2004; Tirilly et al., 2008; Uijlings et al., 2009), scene recognition (Fei-Fei and Perona, 2005; Li et al., 2011; Botterill et al., 2008), image retrieval (Shekhar and Jawahar, 2012; Lavoué, 2011), target detection (Sun et al., 2012), image annotation (Wu et al., 2010), 3D object categorization (Toldo et al., 2009), object segmentation (Larlus et al., 2010), and facial expression recognition (Li et al., 2009; Ionescu et al., 2013).

Another example, **audio signals**, can be also treated as PPs with the **bag-of-audio-words** (Pancoast and Akbacak, 2012) or the **bag-of-frames** (Aucouturier et al., 2007) model. In these representations, a signal is decomposed into a set of local features such as mel-frequency cepstral coefficients (MFCC) (Aucouturier et al., 2007; Pancoast and Akbacak, 2012) or octave-scale spectral contrast (West and Cox, 2004). Note that this can be considered as a special case of a more general model: the **audio-word n-grams** where an audio signal is treated as a multi-set of sequences of n audio words (Pancoast and Akbacak, 2013). The bag-of-audio-words model has been proved very effective in soundscape classification (Ma et al., 2003; Aucouturier et al., 2007) that classifies audio signals into environmental scenes such as railway stations, streets, lectures, offices, etc. This representation for audio data has also been used in many other applications, such as acoustic event classification (Pancoast and Akbacak, 2012; Plinge et al., 2014), musical genre classification (West and Cox, 2004; Zeng et al., 2009), audio retrieval (Chechik et al., 2008), video retrieval (Jin et al., 2012), audio-video copy detection (Ngo et al., 2009; Liu et al., 2010), and artist identification (Mandel and Ellis, 2005).

Another example is **sparse data** which has attracted considerable interest in recent years. Sparse data are data in the form of arrays with most elements taking the same value(s) (Chickering and Heckerman, 1999; Jing et al., 2007), e.g., the arrays in the left table of Figure 1.3h of which most elements are zeros. This data can be efficiently represented by PPs, for example, a sparse array can be treated as a set of non-zero values and their indices (Figure 1.3h). The sparseness of data is commonly found in text data (and the like), for example, when a document is represented by a N-dimensional array of the numbers of word occurrence (i.e., its word histogram), with N is the dictionary size (Joachims, 1996). Since a dictionary usually contains many more words than those used by a document, the array representing the document is usually sparse, i.e., most elements are zeros. Instead of the full (sparse) array, we can store only pairs (word ID, number of occurrence), i.e., a multi-set of word IDs.[4] In many modern applications, **high-dimensional data**, such as gene data (Carvalho et al., 2012), and recommendation data (Pavlov and Pennock, 2002), are also often sparse (Jing et al., 2007) and thus are promisingly treated as PPs.

The aforementioned is a non-exhaustive list of examples of PP data, other examples can be found in books such as (Moller and Waagepetersen, 2003; Daley and Vere-Jones, 2003; Diggle, 2014) as well as from many sources on the Internet.

1.3 Problem formulation

In this section, we define our problem of study — **point pattern learning**[5] — and compare it with a traditional learning problem — **point learning** — in order to point out their differences and the significance of developing point pattern learning.

As introduced in Section 1.2, a **point** is a single (fixed-length) vector, which is also named as an **instance** or a single-instance datum to differentiate from multiple instance datum (Foulds and Frank, 2010). Learning data consisting of independent points — so-called single-instance learning — is a traditional and well-studied learning framework (Foulds and Frank, 2010; Amores, 2013; Cheplygina et al., 2015), whereas learning PP data — so called **point pattern learning**, or more commonly known as **multiple instance learning**[6] — is a developing field of research and receives growing interest in machine learning (Wang and Zucker, 2000).

Formally, if we denote the point space by $\mathcal{X}$ and the set of all finite multi-

[4] This is indeed the aforementioned bag-of-words representation.
[5] In particular, we focus on three fundamental learning tasks for PP, namely clustering, classification, and novelty detection (see Section 1.1.1).
[6] Named by Dietterich et al. (1997).

subsets of $\mathcal{X}$ by $\mathcal{M}(\mathcal{X})$, we can define a point learning task[7] as $f_P : \mathcal{X} \to L$, and a PP learning task as $f_{PP} : \mathcal{M}(\mathcal{X}) \to L$, where L is the set of labels (Foulds and Frank, 2010).[8]

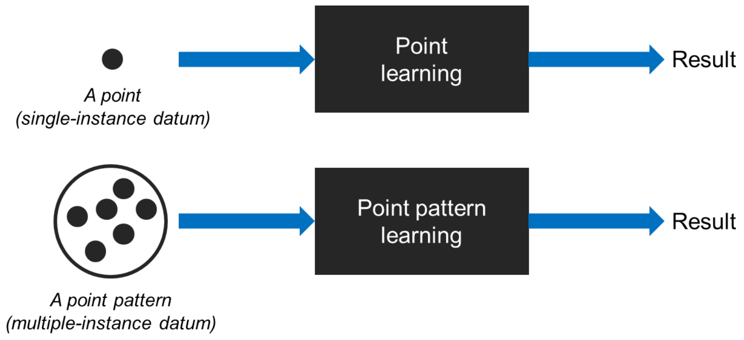


Figure 1.6: Point learning versus point pattern learning (based on a similar illustration in (Foulds and Frank, 2010)).

	INPUT DATA	**INPUT LABEL**[9]	**OUTPUT**
Point learning	Points (single-instance data)	Point-level labels	Point-level labels
PP learning	PPs (multiple-instance data)	PP-level labels	PP-level labels

Table 1.2: Point learning versus PP learning.

There are three main differences between point learning and PP learning. First, the input of PP learning are (multi-)sets of dependent points, i.e., each input observation of PP learning should to be treated as a whole object rather than independent points as in point learning (see Figure 1.6 for an illustration). Second, while (supervised) point learning requires labels of individual points, PP learning can use only labels of PPs rather than labels of their constituent points.[10] Third, the output of PP learning are labels of whole PPs, not indi-

[7] Restricted in our learning tasks of interest, i.e., clustering, classification, and novelty detection.

[8] In (Foulds and Frank, 2010), MI learning is defined as $f_{PP} : \mathbb{N}^{\mathcal{X}} \to L$, where $\mathbb{N}^{\mathcal{X}}$ is the set of all multi-subsets of $\mathcal{X}$, however, in practice $\mathcal{M}(\mathcal{X})$ — set of all *finite* multi-subsets of $\mathcal{X}$ — is enough for real datasets.

[9] (for supervised learning)

[10] There are also approaches to PP learning requiring point labels, however, since obtaining point labels not only is expensive but also can break the label structure of PP data (Cheplygina et al., 2015), we do not consider them in this work.

vidual points as in point learning.[11] Table 1.2 summarizes these differences between point learning and PP learning.

The original (and currently popular) approach for PP learning is to exploit the point-level labels together with some *assumption* between point-level and PP-level labels to infer the PP labels (Dietterich et al., 1997; Foulds and Frank, 2010). This approach allows point learning techniques to be used directly for PP learning, however, it omits a nice property of PP learning: using only PP-level labels which helps to reduce the extensive cost of labeling individual points. In addition, there are some undesirable properties of this approach, as discussed through the following example.

Consider an example of image classification where the input images are represented as PPs of their key points. The target is to classify whether an image includes tree(s) (called a *positive image*) or not. To apply the approach using point-level labels, it is first required (a possibly computational expensive process) to label all the key points in the training images as either belonging to trees (called *positive key points*) or not. Using these key points and their labels, we train a point classifier assessing a key point as either positive or not (i.e., belongs to trees or not). In the testing phase, given a new image, we use the learned point classifier to detect positive key points. If there is at least one positive key point in the image, we conclude that the image is positive (i.e., includes tree(s)). Here, we imposed an *assumption* between point labels (key point labels) and the PP label (image label): *a PP is labeled as positive if and only if one or more of its points are positive*. This is known as the **standard MI assumption** which originated in Dietterich et al. (1997)'s work and is commonly used in PP learning algorithms (Foulds and Frank, 2010).

The aforementioned approach has two undesirable properties. First, it is usually difficult and costly to obtain point labels (Cheplygina et al., 2015), e.g., in the above example, labeling all the key points of the training images (mark-

[11] There also exists applications where the output labels at point-level are required, such as image annotation (Cheplygina et al., 2015), however, this is out of the scope of this research.

ing whether they belong to trees or not) can be a time-consuming task. Second, the aforementioned standard assumption about the relationship between point labels and PP labels can be not relevant in many cases. Revisiting the aforementioned example, but now we want to classify whether an image contains *both tree(s) and mountain(s)*. Following the described approach, we first mark the key points of trees or mountains as positive key points. Then, an image containing positive key points can be *not* our desired image since it can be an image of only trees or only mountains, not both.

Moreover, using point learning methods for PP learning can lead to some problematic issues (discussed in the next section). These highlight the need to develop PP learning techniques which work directly on PP data using PP labels only (i.e., solutions for the PP learning problem formulated above). These learning methods, if they can be developed, will have several advantages, such as saving point-level labeling cost, and avoiding making assumptions between point-level and PP-level labels (which can be irrelevant in some cases, such as in the aforementioned example).

1.4 Example of learning PPs using point learning method

The purpose of this section is to illustrate some problematic issues which can arise when using point learning methods for PP learning. Specifically, the example in this part shows that the solution for PP novelty detection (a semi-supervised PP learning task, see Section 1.1.1) cannot be simply extended from its counterpart for point data, such as using the *naive Bayes (NB) likelihood*.

Generally, a statistical data model is specified by the **likelihood** function which can be interpreted as *how likely* an observation is, given the parameters of the underlying model. In novelty detection, the likelihood can be used to detect novelties (Markou and Singh, 2003). Let us consider an intuitive example of detecting abnormal apple falling patterns.

Suppose that apples, which have fallen from an apple tree, land on the

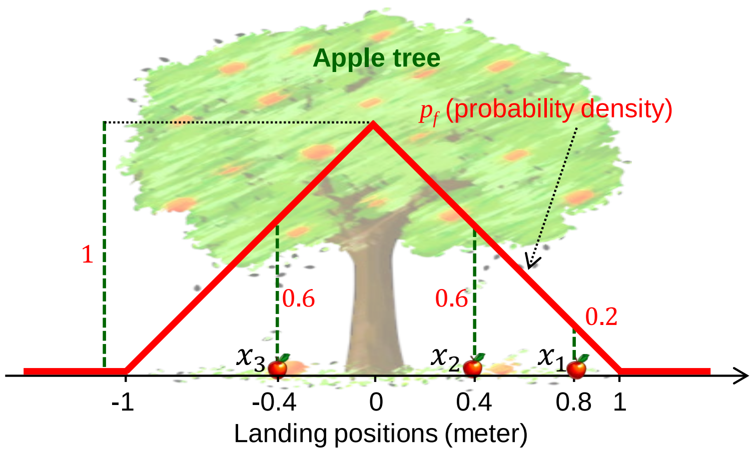


Figure 1.7: Distribution of apple landing positions. Position $x_1 = 0.8\,\mathrm{m}$ is 3 times less likely than $x_2 = 0.4\,\mathrm{m}$ and $x_3 = -0.4\,\mathrm{m}$ in having fallen apples. Credit: clipartbest.com (apple tree clipart)

ground *independently* of each other, and that the daily PPs of landing positions are independent from day to day. Further, the probability density p_f of the landing position, learned from normal training data, is shown in Figure 1.7. Since the apple landing positions are independent, following common practice (see e.g., (Maron, 1961; Joachims, 1996; McCallum and Nigam, 1998; Cadez et al., 2000; Csurka et al., 2004)) the likelihood that the apples land at positions $x_1, ..., x_m$ is given by the NB likelihood (Russell and Norvig, 2003; Bishop, 2006):

$$p(x_1, ..., x_m) = \prod_{i=1}^{m} p_f(x_i) \tag{1.1}$$

Suppose on day 1, we observe one apple landing at x_1, and on day 2 we observe two apples landing at x_2 and x_3 which of these daily landing patterns is *more likely to be a novelty*? Since there is no novel training data in novelty detection, the common practice (see e.g., (Markou and Singh, 2003)) is to examine the *normal data likelihoods* of the landing patterns, i.e., to compare

$$p(x_1) = p_f(x_1) = 0.2, \tag{1.2}$$

$$p(x_2, x_3) = p_f(x_2)\, p_f(x_3) = 0.36, \tag{1.3}$$

to identify outliers. From Eq. (1.2)-Eq. (1.3), one can conclude that the day 1 pattern *is more likely* to be a novelty since $p(x_1) < p(x_2, x_3)$. However, had we measured distance in centimeters, then

$$p(x_1) = 0.002 > p(x_2, x_3) = 0.000036, \tag{1.4}$$

thereby, *contradicting* the previous conclusion! This phenomenon arises from the incompatibility in the *measurement units of the likelihoods* because $p(x_1)$ is measured in "m^{-1}" or "cm^{-1}" whereas $p(x_2, x_3)$ is measured in "m^{-2}" or "cm^{-2}", i.e., we are *not* "comparing apples with apples."

In addition, the NB likelihood of the landing positions also suffers from another problem. To eliminate the effect of unit incompatibility, we assume that there are only 201 positions numbered from -100 to 100, evenly spaced on the interval $[-1\,\mathrm{m}, 1\,\mathrm{m}]$. Thus, instead of a probability density on $[-1\,\mathrm{m}, 1\,\mathrm{m}]$ we now have a *(unit-less) probability mass function* on the discrete set $\{-100, \dots, 100\}$, as shown in Figure 1.8.

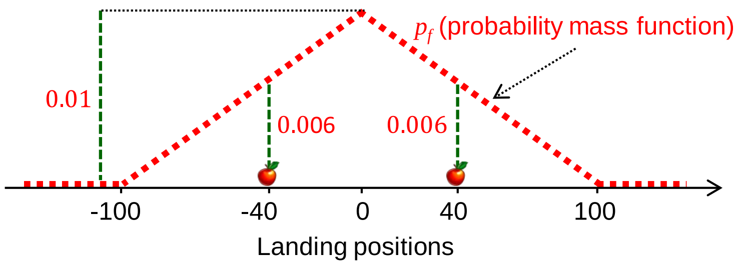
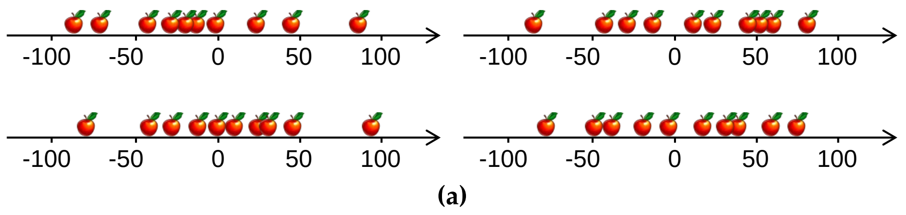


Figure 1.8: Distribution of discrete landing positions.

Four PPs from the normal training data set are shown in Figure 1.9a, while Figure 1.9b shows 2 new observations X_1 and X_2. Since X_2 has only 1 feature, whereas X_1 and the normal observations each has around 10 features, it is intuitive that X_2 is novel. However, its likelihood is much *higher* than that of X_1 (0.009 versus 2×10^{-23}). This counter intuitive phenomenon arises from the lack of appropriate *cardinality information* in the likelihood.

The simple example above demonstrates that the NB likelihood of the constituent points is not a good likelihood of a PP. In particular, it suffers from incompatibility in the unit of measurement and does not appropriately account



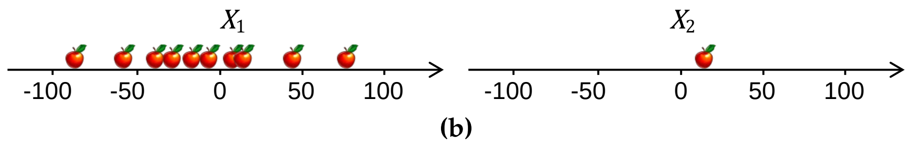


Figure 1.9: (a) Examples of normal observations. (b) Input observations: $p(X_1) \approx 2 \times 10^{-23}$ and $p(X_2) = 0.009$.

for the number of elements in each PP.

1.5 Contributions

In this work, we solve three PP learning problems, namely classification, clustering and novelty detection, using two approaches: one with knowledge of the underlying data model (**model-based approach**), and one without (**distance-based approach**). The proposed methods are listed in Table 1.3 and briefly described in the following sections.

TASK	MODEL-BASED APPROACH	DISTANCE-BASED APPROACH
Classification	MLE of IID-cluster point process	k-NN with OSPA distance
Novelty detection	Ranking function based on IID-cluster point process density	NN-based novelty detection with OSPA distance
Clustering	EM with mixture of IID-cluster point processes	AP with OSPA distance

Table 1.3: Proposed methods for PP learning problems.

1.5.1 Proposed methods for PP classification

To address the classification problem for PP data, we propose two approaches. In the *model-based approach*, we propose to model PP data using **IID-cluster point processes** (Daley and Vere-Jones, 1988; Moller and Waagepetersen, 2003), and develop a **maximum likelihood (ML) estimation** to learn the data model.

The proposed method delivers good performance on both simulated and real data experiments (see Chapter 3 for details).

In the *distance-based approach*, we apply the **Optimal Sub-Pattern Assignment (OSPA)** distance (Schuhmacher et al., 2008) together with the **k-nearest neighbour (k-NN)** algorithm (Cover and Hart, 1967) to classify PP data. The advantage over existing approaches lies in the versatility of the OSPA distance over other set distances, such as the Hausdorff (Huttenlocher et al., 1993b), Chamfer (Gavrila and Philomin, 1999) and Earth mover's (Zhang et al., 2007) distances (see Chapter 6 for details).

1.5.2 Proposed methods for PP novelty detection

In the *model-based approach*, instead of using the IID-cluster point process density as the likelihood for PP data (which may provide poor performance in some cases), we propose a novel **ranking function** for PP data based on this density. This ranking function outperforms the naive Bayes likelihood as well as the IID-cluster point process density in novelty detection for PP data (see Chapter 4 for details).

In the *distance-based approach*, we propose a solution based on the **OSPA distance** between the candidate PP and its **nearest neighbour** in the normal training set. We also propose to use normal training data to find the suitable parameter for the OSPA distance. This PP novelty detection method is simple, effective and versatile across various applications (see Chapter 6 for details).

1.5.3 Proposed methods for PP clustering

In the *model-based approach*, we propose to use finite **mixture models of IID-cluster point processes** as the generative models for PP clusters. Then the **expectation-maximization (EM)** technique is deployed to learn the mixture parameters and cluster the data. The proposed method performs well on both simulated and real data (see Chapter 5 for details).

In the *distance-based approach*, we combine the **affinity propagation** clustering algorithm (Frey and Dueck, 2007) with the **OSPA distance** as a dissimilarity measure. Compared to the existing technique using the k-medoids algorithm and Hausdorff distance, AP can find clusters faster with much lower error, and does not require the number of clusters to be specified (Frey and Dueck, 2007). In addition, the OSPA distance is more versatile than the Hausdorff distance, hence the proposed method can be suitable to more applications (see Chapter 6 for details).

1.6 Thesis structure

The remainder of this thesis is organized as follows. Chapter 2 provides the necessary background for the work in this thesis. In this chapter, three learning tasks of interest, namely classification, novelty detection, and clustering, are described in more detail in terms of their definitions, common methods, and performance measures. This chapter also provides background on point process theory: important definitions, successful applications, common point process models and their probability density functions. The last sections of this chapter describe three distances for sets used in this work, namely the Hausdorff, Wasserstein, and OSPA distances.

In three chapters, Chapter 3, Chapter 4, and Chapter 5, we describe our proposed methods for PP classification, novelty detection, and clustering, respectively, under the model-based approach. Each of these chapters has a similar structure, including a brief review on existing methods for the problem of interest, details of our proposed method, and numerical experiments on both simulated and real data.

In Chapter 6, we propose alternative solutions for the three learning problems of interest, using the distance-based (model-free) approach. In this chapter, for each learning problem, an overview on the existing distance-based methods for the learning problem is first given, followed by details of the proposed method, and numerical experiments on both simulated and real data. This chapter concludes with a comparison between the performance of the proposed distance-based methods and that of the model-based methods.

Finally, Chapter 7 concludes the thesis with a summary and discussion on the work, as well as suggesting some potential future research directions.

CHAPTER 2
BACKGROUND

Chapter's key points: background on

- Classification
- Novelty detection
- Clustering
- Point process theory
- Set distances

In the previous chapter, we gave an overview on machine learning as well as its fundamental learning problems. In this chapter, three learning tasks of interest, namely classification, novelty detection, and clustering, will be described in more detail in relation to their definitions, common methods, and performance measures (Section 2.1, Section 2.2, and Section 2.3). In Section 2.4, we provide background on point process theory which is used to develop model-based PP learning methods in the following chapters, focusing on its important definitions, successful applications, common point process models and their probability density functions. Finally, Section 2.5 describes distances of sets, which are used in distance-based PP learning methods.

2.1 Classification

2.1.1 Definition

Classification is a supervised learning task of assigning a class label $y \in \{1, ..., K\}$

to each input observation X (Bishop, 2006), where K is the number of classes.[1] Classification relies on **training data**, which are fully observed input-output pairs $\mathcal{D}_{\text{train}} = \{(X^{(n)}, y^{(n)})\}_{n=1}^{N_{\text{train}}}$, hence the name *supervised learning* (Murphy, 2012).

2.1.2 Applications

Classification is arguably the most widely used machine learning form (Murphy, 2012). As an example, let us briefly describe an early application of machine learning: Soybean disease classification — a classic machine learning success (Witten and Frank, 2005). The target of this application is to classify diseases of soybean plants by identifying diagnosing rules. Impressively, the rules generated by computer (in this 1970s research) outperform human-expert-derived rules with an accuracy rate of 97.5% versus 72% (Witten and Frank, 2005).

Nowadays, classification can be found in various fields of study, from computer vision (Pham et al., 2000; Javed and Shah, 2002), natural language processing (Collobert et al., 2011; Pang et al., 2002), speech recognition (Graves et al., 2006; Nwe et al., 2003), handwriting recognition (Xu et al., 1992; Belongie et al., 2002), information retrieval (Lewis, 1998; Croft et al., 2010), to drug discovery (Dietterich et al., 1997; Ellinger-Ziegelbauer et al., 2008).

2.1.3 General approach

The classification process consists of two main steps: the **learning step** and the **classifying step** (Han et al., 2012). In the first step, using training data, a *mapping* between training observations and their labels is constructed. In the classifying step, the class labels of new (test) observations are predicted using the learned mapping. The classification performance then can be evaluated using the indicators described in Section 2.1.5.

[1] There exists a learning problem called *multi-label classification*, where an observation can be assigned with more than one label (Tsoumakas and Katakis, 2006). However, it is out of the scope of this work.

2.1.4 Classification methods

In this section, we briefly describe several typical classification methods, namely classification rules, decision trees, neural networks, support vector machines, statistical models, and k-nearest neighbors. For a more comprehensive review, refer to textbooks such as (Bishop, 2006; Murphy, 2012; Han et al., 2012).

Classification rules are simply *if-then rules* (Han et al., 2012; Witten and Frank, 2005). For example, to classify low-risk/high-risk customers, a bank can use a rule such as (Alpaydin, 2004):

if income>A *and* savings>B *then* low-risk *else* high-risk

These rules can be either given by human-experts or generated by computers using, for example, the *separate-and-conquer* technique (Witten and Frank, 2005). In many cases, computer-generated rules can outperform expert-derived rules, e.g., in the aforementioned soybean disease classification.

Decision trees are tree structures supporting decision-making processes based on the concept of conditional information gain from information theory. In a decision tree, each node represents a test on a feature (or attribute) of input data, the outcomes of a test are represented by edges connected to the note, and the decisions (or class labels, in the case of classification) are represented by leaves (Han et al., 2012). For example, the classification rule above can be (partly) represented by the decision tree in Figure 2.1. A decision tree can be built using a *greedy optimization* (Bishop, 2006) or *C4.5 algorithm* (Quinlan, 2014).

A **neural network**, in general, is a computational model simulating the way a human brain processes information. In classification, a class of neural networks, called *feed-forward neural networks* or *multilayer perceptrons*, is commonly used due to its efficiency to practical applications (Bishop, 2006). A multilayer perceptron consists of processing units (nodes), organized in layers, connecting with units of other layers to form a directed graph (see Figure 2.2 for il-

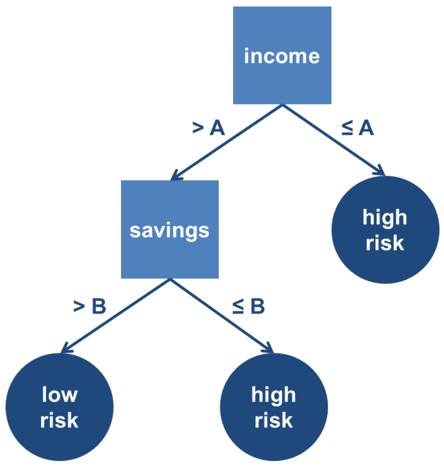


Figure 2.1: Example of a decision tree.

lustration). The multilayer perceptron can be efficiently trained by using the *backpropagation* technique (Bishop, 2006). For more details on neural networks, refers to books such as (Ripley, 1996; Bishop, 1995).

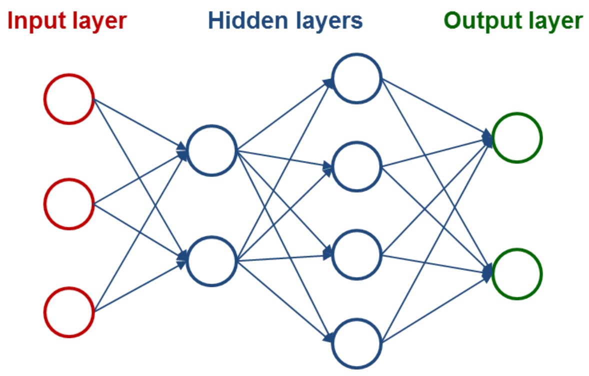


Figure 2.2: Example of a feed-forward neural network or multilayer perceptron.

A **support vector machine (SVM)** is a binary classifier, i.e., the classifier assigns data into two classes. The key idea is constructing an *optimal hyperplane* (also known as *maximum-margin hyperplane*) which is chosen so that the distances between the hyperplane and the closest observations from either classes (i.e., the margins) are maximal (Vapnik, 1995) (see Figure 2.3 for illustration). The SVM classifier can be extended to multiclass classification using schemes, such as One-vs-All or All-vs-All (Rifkin and Klautau, 2004). Refer to books, such as (Vapnik, 1995; Alpaydin, 2004), for more detail on SVM.

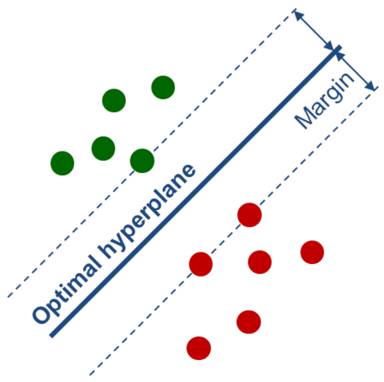


Figure 2.3: Example of an optimal hyperplane in the SVM classifier (based on a similar illustration in (Vapnik, 1995)).

Classification approaches with **statistical models** are based on the assumption that data are generated from some underlying distribution (Murphy, 2012), e.g., a Gaussian distribution. This model can be learned using *maximum likelihood* (ML) estimation or *maximum a posteriori* (MAP) estimation (Murphy, 2012). A well-known example of this type of classifiers is the *naive Bayes* (Russell and Norvig, 2003; Bishop, 2006), which imposes an independence assumption among data features. More detail on model-based classifiers will be given in Chapter 3.

k-**nearest neighbors (**k**-NN)** classifier (Cover and Hart, 1967; Keller et al., 1985) belongs to the class of *lazy learning*. Contrary to *eager learning* algorithms (e.g., the aforementioned classifiers) in which a classifier is learned from training data in the training phase, the k-NN algorithm delays most of its computational effort to the classifying (or test) phase. In the test phase, a queried observation will be assigned the most popular label among its k nearest observations in the training set. This classifier will be discussed in more detail in Chapter 6.

2.1.5 Performance indicators

The evaluation for a classification algorithm is usually performed by running the algorithm several times on different *test sets* and then averaging the performance measures, such as *accuracy*, and *uncertainty coefficient*. For datasets that are not too large, the *N-fold cross validation* technique (Bishop, 2006) is often used to generate multiple training and test sets for the evaluation.

In general, a **test set** of input-output pairs $\mathcal{D}_{\text{test}} = \{(X^{(n)}, y^{(n)})\}_{n=1}^{N_{\text{test}}}$, different from the training set, is used to evaluate the performance of classifiers. Following (Manning et al., 2008), we can use **accuracy** defined as

$$\text{Accuracy} \triangleq \frac{\text{No. of correct classifications}}{\text{No. of observations in the test set}}. \tag{2.1}$$

To ensure a meaningful accuracy measure, it is recommended that the test sets contain classes of similar sizes, otherwise other indicators, such as *uncertainty coefficient* (Press et al., 2007) should be used. In this work, since the size of classes in the used test sets are the same, we use accuracy when evaluating classifiers in order to easily interpret their performance.

2.2 Novelty detection

2.2.1 Definition

Novelty detection is the task of identifying new or strange data that are significantly different from normal training data (Markou and Singh, 2003; Pimentel et al., 2014). Novelty detection thus belongs to a broader class: **outlier detection**. To better understand the task, we dedicate the next section to locate novelty detection in outlier detection.

2.2.2 Novelty detection versus outlier detection

The general task of identifying observations that are significantly different from the rest of the data (according to some criteria) is referred to as **outlier detection** or **anomaly detection**. For comprehensive surveys on outlier detection, refer to (Hodge and Austin, 2004; Chandola et al., 2009).

Based on training data requirements, outlier detection can be divided into three categories: unsupervised, supervised, and semi-supervised (Chandola et al., 2009; Hodge and Austin, 2004). **Unsupervised outlier detection** does not require any training data, and as a result can be easily applied to any problem, albeit at poorer performance compared to the other two categories.

On the other hand, **supervised outlier detection**, which requires both normal and anomalous training data, achieves better performance. However, its applicability is limited because anomalies do not necessarily follow any model, and anomalous data are often unavailable in practice, for example, the many types of malfunctions of a newly developed machine cannot be foreseen.

Semi-supervised outlier detection, or simply **novelty detection**, which requires *only normal training data*,[2] provides a good compromise between applicability and performance (Hodge and Austin, 2004). Novelty detection is a separate problem in its own right, whereas unsupervised and supervised outlier detection problems can be considered as special cases of the clustering and classification problems, respectively (Hodge and Austin, 2004).

2.2.3 Applications

Novelties provide information that can lead to critical decisions/actions in many applications, for example, fraud detection (Bolton and Hand, 2002), intrusion detection (Jyothsna et al., 2011), medical diagnosis (Tarassenko et al., 1995; Quinn and Williams, 2007), structural health monitoring (Surace and Worden, 2010), video surveillance (Diehl and Hampshire, 2002; Markou and Singh, 2006), and automated inspection (Neto and Nehmzow, 2007).

2.2.4 General approach

Similar to classification, there are two key steps in novelty detection, namely the **training step** and **detecting step**. In the first step, using a training dataset (containing only normal data), a *normal data characterization*, e.g., the normal data likelihood (in the model-based approach, see Chapter 4), or the intra-class distances between normal data (in the distance-based approach, see Chapter 6),

[2] As previously discussed in Chapter 1, in general, semi-supervised learning uses a training set consisting of two parts: *observations with labels* and *observations without labels* (Chapelle et al., 2006). Our considered scenario (training set consisting of only normal data) is a special case of semi-supervised learning.

is constructed. In the detecting step, the queried observations are ranked according to how well they fit the normal data characterization and those not well-fitted are deemed novel (Chandola et al., 2009; Hodge and Austin, 2004). The detection performance then can be evaluated using the indicators described in Section 2.2.6.

2.2.5 Novelty detection methods

In this section, we briefly describe several techniques used in novelty detection, namely negative selection, support vector domain description, statistical models, and nearest neighbours.

The **negative selection** method was first introduced by Forrest et al. (1994), inspired by the human immune system (Pimentel et al., 2014). In the immune system, *T cells* are used to detect foreign proteins (e.g., viruses, transplanted organs) by the binding test: any cells that successfully bind with a T cell are marked as foreign cells. T cells are generated by a random genetic rearrangement process and are selected through *negative selection*: T cells that successfully bind with body proteins will be destroyed. Following this idea, Forrest et al. (1994) proposed a method for detecting novel strings (e.g., computer viruses, malware codes) by using *detector strings*. The detector strings are generated and selected so that they will not match normal strings, thus any strings matching the detector strings can be classified as novelties (Forrest et al., 1994). This approach was later extended to data other than binary strings, such as the work by Dasgupta and Majumdar (2002); González and Dasgupta (2003); Taylor and Corne (2003); Esponda et al. (2004).

Support vector domain description, proposed by Tax and Duin (1999), is inspired by the method of support vector machines in classification (see Section 2.1.4). In this approach, the problem of novelty detection is solved by finding a *minimum volume hypersphere* which encloses (almost) all training data (consisting of only normal data) (Tax and Duin, 1999). Then, the observations with distances to the center of the hypersphere larger than the radius of the hypersphere are deemed novelties. In order to deal with non-spherical data, a generalization of this method using kernel functions, such as polynomial kernel or Gaussian kernel, is also proposed (Tax and Duin, 1999, 2001).

Statistical model approaches for novelty detection assume that novelties are distributed outside the normal data distribution (Tarassenko et al., 1995). Hence the novelties have significantly lower likelihoods compared to normal data according to the normal data distribution. Commonly, for point data, *Gaussian mixture models* are used to represent normal data (Markou and Singh, 2003; Pimentel et al., 2014). This model can be learned using the *expectation-maximization* (EM) methods (Dempster et al., 1977). For sequential data, *hidden Markov models* (HMM) can be used. Each HMM consists of a distribution representing hidden states transition (called *transition distribution*), and a set of distributions for generating observations (called *emission distributions*). The parameters of these distributions can be learned by the *Baum-Welch algorithm* (Baum et al., 1970; Rabiner and Juang, 1986). The statistical approach for novelty detection method will be discussed in more detail in Chapter 4.

The **nearest neighbour** approach is arguably the most common technique for novelty detection (Pimentel et al., 2014). This method is based on the assumption that normal observations are closer to the training (normal) data than novelties (with respect to some distance measure) (Hautamäki et al., 2004). Depending on data type, different distance measures can be applied, such as Euclidean distance and p-norm distance (for vectors), Mahalanobis distance and Earth mover's distance (for distributions), Hausdorff distance and Jaccard distance (for sets). This novelty detection method will be discussed in more detail in Chapter 6.

2.2.6 Performance indicators

Similar to classification (see Section 2.1.5), the evaluation of novelty detection is also done by running the algorithm several times on different test sets and then averaging the performance measures, such as *precision*, *recall*, and F_1 *score*.

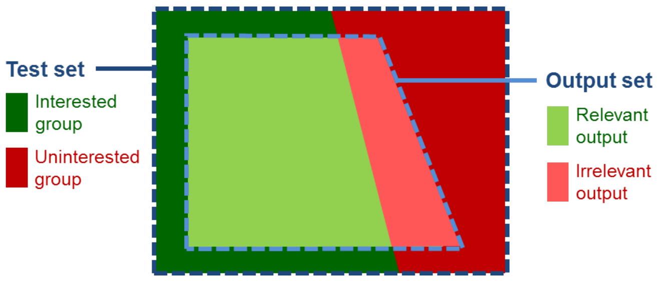


Figure 2.4: Test set and output set.

Consider a problem of identifying some data of interest out of a dataset consisting of two groups: interested and uninterested data (see Figure 2.4), e.g., novel and normal data in the context of novelty detection. The relevance of an output set by a algorithm can be determined using **precision** and **recall** (Manning et al., 2008):

$$\text{Precision} \triangleq \frac{|\text{Relevant output}|}{|\text{Output set}|}, \quad \text{Recall} \triangleq \frac{|\text{Relevant output}|}{|\text{Interested group}|},$$

where $|S|$ is the number of observations in set S. In words, *precision* is the proportion of relevant output data among data in the output set, and *recall* is the proportion of relevant output data among all relevant data in the test set (i.e., the interested group, see Figure 2.4).

Precision and recall evaluate different aspects of the algorithm. Precision shows how relevant (or correct) the output set is, whereas recall shows how good the algorithm is at getting the data of interest. If we attempt hardly to increase recall, precision may decrease and vice versa. In some applications, such as novelty detection, a balance between precision and recall is necessary. **$\mathbf{F_1}$ score** is a performance measure providing such balance between precision and recall (Manning et al., 2008):

$$\text{F}_1 \text{ score} \triangleq 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{2.2}$$

Note that the F_1 score, as a function of precision and recall, is discontinuous at $(0,0)$. To ensure continuity of the F_1 score, we define its value at $(0,0)$ to be its limit at $(0, 0)$, i.e., F_1 score$(0, 0) \triangleq 0$.

Other performance measures, such as *uncertainty coefficient* (Press et al., 2007), *adjusted rand index* (Santos and Embrechts, 2009), *Matthews correlation coefficient* (Matthews, 1975), *receiver operating characteristic (ROC)* or *precision-recall* curve (Murphy, 2012), can be also used to quantify other aspects of novelty detection performance. For the purpose of this work, the F_1 score are sufficient.

2.3 Clustering

2.3.1 Definition

Compared to supervised learning tasks (e.g., classification), unsupervised tasks (e.g., clustering) are much less well-defined (Murphy, 2012; Estivill-Castro, 2002). A loose definition of **clustering** (also known as **cluster analysis**) could be: the process of organizing objects into groups whose members are similar in some way (Murphy, 2012). A **cluster** is therefore a collection of objects which are similar to each other and are dissimilar to the objects belonging to other clusters. A **clustering** or **partitioning** of a dataset $\{X_1,...,X_N\}$ is often represented by the (latent) cluster assignment $y_{1:N}$, where y_n denotes the cluster label for datum X_n.

Clustering is an *unsupervised learning* problem since the class (or cluster) labels are not given (Jain et al., 1999; Russell and Norvig, 2003).

2.3.2 Applications

Clustering is a fundamental machine problem with a long history dating back to the 1930s in psychology (Tryon, 1939). Today, clustering is widely used in a host of application areas, including text mining (Dhillon and Modha, 2001), network traffic classification (Erman et al., 2006), medical imaging (Yang et al., 2002a), genetics (Collins et al., 2007), immunology (Lund et al., 2004), market research (Arimond and Elfessi, 2001), social network analysis (Mishra et al., 2007), and mobile robotics (Nguyen et al., 2005).

2.3.3 Clustering types

In terms of the output, clustering algorithms can be broadly categorized as *hard* or *soft* (also referred to as *fuzzy*) (Jain et al., 1999). In **hard clustering**, each datum can only belong to one cluster, i.e., a hard clustering algorithm outputs a partition of the dataset. More specifically, the hard clustering could be dichotomized as either hierarchical or partitional. The **hierarchical clustering** outputs a nested tree of partitions, whereas the output of **partitional clustering** is only one partition (Murphy, 2012; Jain et al., 1999). *k-means* is a typical example of hard (partitional) clustering.

Soft clustering (or **fuzzy clustering**), on the other hand, allows each datum to belong to more than one cluster with certain degrees of membership (Kaufman and Rousseeuw, 1990; Xu and Wunsch, 2005). The *Gaussian mixture model* is an example of soft clustering wherein the degree of membership of a datum to a cluster is given by its likelihood with respect to the mixture density. Hard clustering can be also obtained from soft clustering by simply assigning the cluster with the highest membership degree to each datum (Jain et al., 1999).

2.3.4 Clustering algorithms

Since the problem of clustering is too broad (due to its ambiguous definition), to the best of our knowledge, there is no general approach for the clustering problem. Instead, we represent in this section several widely used approaches to the clustering problem, namely *centroid-based approach* (such as k-means, k-medoids), *statistical model-based approach* (such as Gaussian mixture models), and *message passing approach* (such as affinity propagation).

In the **centroid-based** approach, each cluster is represented by a *centroid* which can be either a datum belonging to the dataset, called an exemplar, or a computed mean. An example is the well-known **k-means** algorithm (MacQueen, 1967) which consists of two main steps. First, each datum is assigned to the group whose center is closest to the datum (with respect to some distance measure). Second, the group center is computed using the current member observations of the group. These two steps are repeated until convergence. As another example, the k-**medoids** algorithm (Kaufman and Rousseeuw, 1987) is similar to k-means except that in the second step, the group center is chosen among the existing observations in the dataset. For more detail on these algorithms as well as their extensions, refer to materials such as (Jain, 2010; Kaufman and Rousseeuw, 1990).

Clustering using **statistical models** is based on the assumption that data are generated by some underlying distribution. A well-known algorithm is **expectation-maximization (EM)** with Gaussian mixture models. There are two main steps in this clustering method. First, the parameters of the underlying model are estimated using the EM technique (Dempster et al., 1977). Second, data are assigned to groups using *maximum a posteriori (MAP)* estimation. We will discuss this clustering method in more detail in Chapter 5.

Message passing is another approach to the clustering problem, with a well-known algorithm: **affinity propagation (AP)** (Frey and Dueck, 2007). The AP algorithm uses the *similarity* values (e.g., negative of distances) between pairs of observations as input and returns the 'best' set of exemplars. AP first considers all observations as potential exemplars. Progressively better sets of exemplars and corresponding clusters are then determined by passing *responsibility* and *availability* messages based on similarities between observations. Further details on the AP algorithm can be found in (Frey and Dueck, 2007; Dueck and Frey, 2007). We will discuss this clustering method in more detail in Chapter 6.

2.3.5 Performance indicators

A good way to evaluate the performance of a clustering algorithm is using *ground truth data*, i.e., a dataset with true cluster labels given (Färber et al., 2010). Several indicators can be used to measure the quality of a clustering, such as *purity*, *normalized mutual information* (NMI), *rand index*, *F_1 score* (Manning et al., 2008). Let $\mathsf{O} = \{O_1, \ldots, O_n\}$ be the clustering output by an algorithm, O_i be the

i^{th} cluster in O, $\mathsf{G} = \{G_1, \ldots, G_m\}$ be the ground truth data, G_j be the j^{th} cluster in G, and N be the number of observations in the dataset, then these indicators can be defined as follows.

Purity is defined as follows:

$$\text{purity}(\mathsf{O}, \mathsf{G}) \triangleq \frac{1}{N} \Sigma_i \max_j |O_i \cap G_j|. \tag{2.3}$$

Purity is simple to compute and straightforward to interpret (Manning et al., 2008), however it tends to increase (to a maximum of 1) with the number of clusters in the clustering output. Another indicator which can address this issue is the **normalized mutual information (NMI)** (Manning et al., 2008):

$$\text{NMI}(\mathsf{O}, \mathsf{G}) \triangleq \frac{2 \times I(\mathsf{O}; \mathsf{G})}{H(\mathsf{O}) + H(\mathsf{G})}, \tag{2.4}$$

where I is the *mutual information*

$$I(\mathsf{O}; \mathsf{G}) = \sum_i \sum_j \frac{|O_i \cap G_j|}{N} \log \frac{N \times |O_i \cap G_j|}{|O_i| \times |G_j|}, \tag{2.5}$$

and H is the *entropy*

$$H(\mathsf{O}) = -\sum_i \frac{|O_i|}{N} \log \frac{|O_i|}{N}. \tag{2.6}$$

Note that because NMI is normalized, it can be used to compare clustering outputs of different numbers of clusters (Manning et al., 2008).

Rand index is an indicator which measures the accuracy of the clustering output (Manning et al., 2008):

$$\text{Rand index} \triangleq \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}, \tag{2.7}$$

where TP (*true positive*) is the number of assignments of two observations having the same true label as the same cluster, TN (*true negative*) is the number of assignments of two observations having different true labels to different clusters, FP (*false positive*) is the number of assignments of two observations having different true labels as the same cluster, and FN (*false negative*) is the number of assignments of two observations having the same true label to different clusters (see Table 2.1 for summary).

	Same cluster	**Different clusters**
Same true labels	TP (*true positive*)	FN (*false negative*)
Different true labels	FP (*false positive*)	TN (*true negative*)

Table 2.1: Computing TP (true positive), FN (false negative), FP (false positive), and TN (true negative).

F_1 score is another commonly used indicator to measure clustering performance. Recall from Section 2.2.6

$$F_1 \text{ score} \triangleq 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{2.8}$$

Note that in the context of clustering, precision and recall is computed using TP, FN, and FP as follows

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

All the mentioned indicators have a range of $[0, 1]$ (1 for the best performance). For more detail and discussion, refer to (Manning et al., 2008).

2.4 Point process

This section outlines the elements of **point process** theory and presents some basic models for PP data. For further detail on point processes, we refer the reader to textbooks such as (Stoyan et al., 1995; Daley and Vere-Jones, 1988; Moller and Waagepetersen, 2003).

2.4.1 Definitions

A PP can be characterized as a **counting measure** on the space $\mathcal{X}$ of features. Given a PP X, a counting measure N is defined, for each (compact) set $A \subseteq \mathcal{X}$, by

$$N(A) = \text{number of points of } X \text{ falling in } A. \tag{2.9}$$

The values of the counting variables $N(A)$ for all subsets A provide sufficient information to reconstruct the PP X (Stoyan et al., 1995; Daley and Vere-Jones, 1988). The points of X are the set of $\underline{x}$ such that $N(\{x\}) > 0$. A PP is said to be *finite* if it has a finite number of points, i.e., $N(\mathcal{X}) < \infty$; and *simple* if it contains no repeated points, i.e., $N(\{x\}) \leq 1$ for all $x \in \mathcal{X}$.

Formally a **point process** is defined as a *random counting measure*. A random counting measure N may be viewed as a collection of random variables $N(A)$ indexed by $A \subseteq \mathcal{X}$. A point process is *finite* if its realizations are finite almost surely, and *simple* if its realizations are simple almost surely. For all practical purposes, we only consider **finite point processes**.

2.4.2 Applications

Point process theory has proven to be effective in diverse areas such as image processing (Baddeley and Lieshout, 1993; Perrin et al., 2005), neuroscience (Truccolo et al., 2005), physiology (Barbieri et al., 2005), finance and econometrics (Engle and Russell, 1998; Zhang et al., 2001a; Bowsher, 2007), communication (Baccelli and Blaszczyszyn, 2010), forestry (Stoyan and Penttinen, 2000), and robotics (Mullane et al., 2011). Point process theory has attracted a lot of attention worldwide over the last decade due to its success in the area of multi-sensor multitarget tracking (Mahler, 2014). In particular, the *simple-finite point process* (used in this work), also known as the *random finite set*, has been successfully applied to the field of multitarget tracking as briefly reviewed in the following.

The random finite set was used to develop various effective multitarget estimation techniques, such as the Probability Hypothesis Density (PHD) filter, Cardinalized PHD (CPHD) filter, and multi-Bernoulli filters. The *PHD filter* is a computationally inexpensive multitarget filter developed by Mahler (2003c). In addition, the PHD filter is then generalized to be the *CPHD filter* (Mahler, 2007a) which can capture changes in the number of targets better, and hence provide better performance. Both the PHD and CPHD filters can be effectively implemented using Gaussian mixture or sequential Monte Carlo (SMC) (Vo et al., 2005; Vo and Ma, 2006; Vo et al., 2007). Further a Gaussian mixture PHD smoother was developed in (Vo et al., 2012). The PHD/CPHD filters can be extended to jointly estimate the multitarget state, false alarm parameters, and state-dependent detection probability (Beard et al., 2013; Mahler et al., 2011; Mahler and Vo, 2014).

For non-linear dynamic scenarios, the *multi-Bernoulli filters* with SMC implementation can outperform the CPHD filter (Vo et al., 2009, 2010). These filters have been successfully applied in solving many problems from various fields, such as computer vision (Maggio et al., 2008; Hoseinnezhad et al., 2012, 2013), cell biology (Rezatofighi et al., 2015), autonomous vehicles (Mullane et al., 2011; Lundquist et al., 2011), automotive safety (Battistelli et al., 2008; Meissner et al., 2013), sensor scheduling (Mahler, 2003b; Ristic and Vo, 2010; Ristic et al., 2011; Hoang and Vo, 2014; Gostar et al., 2013; Hoang et al., 2015; Gostar et al., 2015, 2016), and sensor networks (Zhang, 2011; Battistelli et al., 2013).

A generalization of the aforementioned filters is the *Generalized Labeled Multi-Bernoulli (GLMB) filter* (Vo and Vo, 2013; Vo et al., 2014, 2017b), which also outputs the targets' trajectories. The GLMB filter is an exact solution of the Bayes multitarget filter (Vo and Vo, 2013; Vo et al., 2014). This filter has several good analytical properties, such as the void probability functional of a GLMB, the Cauchy-Schwarz divergence between two GLMBs, the L_1-distance between a GLMB and its truncation, all can be computed in closed form (Beard et al., 2015b; Vo et al., 2014). Since its appearance, it has been deployed in many applications, including merged measurements (Beard et al., 2015a), extended targets (Beard et al., 2016), maneuvering targets (Punchihewa et al., 2016; Jiang et al., 2016), track-before-detect (Papi et al., 2015; Papi and Kim, 2015), computer vision (Kim et al., 2013; Punchihewa et al., 2014; Rathnayake et al., 2015; Kim et al., 2016), sensor scheduling (Beard et al., 2015b; Gostar et al., 2014), field robotics (Deusch et al., 2015), and distributed multi-object tracking (Fantacci et al., 2015).

2.4.3 Probability

In this work, we are interested in likelihood functions for finite PP observations. For a countable feature space $\mathcal{X}$, the likelihood function f is simply the **probability** of the PP. More concisely,

$$f(\{x_1, ..., x_m\}) = p_c(m) \sum_{\pi} p(x_{\pi(1)}, ..., x_{\pi(m)} \mid m), \tag{2.10}$$

where π denotes a permutation of $\{1, 2, \ldots, m\}$; $p(x_1, \ldots, x_m \mid m)$ is the joint probability of the features $x_1, \ldots, x_m$, given that there are m features; and $p_c(m)$ is the probability that there are m features.

For an uncountable feature space, the likelihood function is the *probability density* of the PP (described in the next section). Hereon, we consider point processes on a compact subset $\mathcal{X}$ of $\mathbb{R}^d$, unless otherwise stated.

2.4.4 Probability density

The **probability density** of a point process is the Radon-Nikodym derivative of its probability distribution with respect to a dominating measure μ, usually an unnormalised probability distribution of a Poisson point process.

Let ν be a (non-atomic σ-finite) measure on $\mathcal{X}$, a **Poisson point process**[3] on $\mathcal{X}$ with intensity measure ν is a point process such that

- for every (compact) set $A \subset \mathcal{X}$, the random variable $N(A)$ is Poisson distributed with mean $\nu(A)$,
- if $A_1, ..., A_n \subset \mathcal{X}$ are disjoint (compact) sets, then the random variables $N(A_1), ..., N(A_n)$ are independent.

In general the probability density of a point process *may not exist* (van Lieshout, 2000; Baddeley et al., 2007). To ensure that probability densities are available, we restrict ourselves to finite point processes (Baddeley et al., 2007).

[3] Later in this thesis, for brevity, sometimes the Poisson point process model is briefly called the **Poisson model**.

Further, in many applications involving uncountable feature spaces, the observed PPs do not have repeated elements, and hence can be modeled as a simple point process. A *simple finite* point process is equivalent to a **random finite set (RFS)** (Baddeley et al., 2007), i.e., a random variable taking values in $\mathcal{F}(\mathcal{X})$ — the space of *finite subsets* of $\mathcal{X}$.

The probability density $f : \mathcal{F}(\mathcal{X}) \to [0, \infty)$ of a RFS is usually taken with respect to the *dominating measure* μ, defined for each (measurable) $\mathcal{T} \subseteq \mathcal{F}(\mathcal{X})$, by (see e.g., (Geyer et al., 1999; Moller and Waagepetersen, 2003; Vo et al., 2005)):

$$\mu(\mathcal{T}) = \sum_{m=0}^{\infty} \frac{1}{m! U^m} \int \mathbf{1}_{\mathcal{T}}(\{x_1, ..., x_m\}) d(x_1, ..., x_m), \tag{2.11}$$

where U is the unit of hyper-volume in $\mathcal{X}$, $\mathbf{1}_{\mathcal{T}}(\cdot)$ is the indicator function for $\mathcal{T}$, and by convention, the integral for $i = 0$ is $\mathbf{1}_{\mathcal{T}}(\emptyset)$. The measure μ is the unnormalized distribution of a Poisson point process with unit intensity $1/U$ when $\mathcal{X}$ is bounded. For this choice of reference measure, it was shown in (Vo et al., 2005) that the integral of f is given by

$$\int f(X)\mu(dX) = \sum_{m=0}^{\infty} \frac{1}{m! U^m} \int f(\{x_1, ..., x_m\}) d(x_1, ..., x_m), \tag{2.12}$$

is equivalent to Mahler's set integral (Mahler, 2003a, 2007b) and that densities relative to μ can be computed using Mahler's set derivative (Mahler, 2003a, 2007b). Note that the reference measure μ, and the integrand f are all dimensionless.

The **probability density** of an RFS, with respect to μ, evaluated at $\{x_1, ..., x_m\}$ can be written as (van Lieshout, 2000, p. 27) ((Eqs. (1.5), (1.6), and (1.7)):

$$f(\{x_1, ..., x_m\}) = p_c(m)\, m!\, U^m f_m(x_1, ..., x_m), \tag{2.13}$$

where $p_c(m)$ is the **cardinality distribution**, and $f_m(x_1, ..., x_m)$ is a symmetric function[4] denoting the joint probability density of $x_1, ..., x_m$ given cardinality

[4] Since f_m is symmetric, the notations $f_m(x_1, ..., x_m)$ and $f_m(\{x_1, ..., x_m\})$ can be used in-

m. Note that by convention $f_0 = 1$ and hence $f(\varnothing) = p_c(0)$. It can be seen from Eq. (2.13) that the probability density f captures the cardinality information as well as the dependence between the features. Also, U^m cancels out the unit of the probability density $f_m(x_1, ..., x_m)$, and hence f is unitless.

2.4.5 Intensity and conditional intensity

The **intensity function** λ of a point process is a function on $\mathcal{X}$ such that for any (compact) $A \subset \mathcal{X}$

$$\mathbb{E}\left[N(A)\right] = \int_A \lambda(x)dx. \tag{2.14}$$

The intensity value $\lambda(x)$ can be interpreted as the instantaneous expected number of points per unit hyper-volume at x.

For a *hereditary* probability density f, i.e., $f(X) > 0$ implies $f(Y) > 0$ for all $Y \subseteq X$, the **conditional intensity** at a point u is given by (Baddeley et al., 2007)

$$\lambda(u, X) = \frac{f(X \cup \{u\})}{f(X)}, \tag{2.15}$$

Loosely speaking, $\lambda(u, X)du$ can be interpreted as the conditional probability that the point process has a point in an infinitesimal neighbourhood du of u given all points of X outside this neighbourhood.

The intensity function is related to the conditional intensity by

$$\lambda(u) = E\left[\lambda(u, X)\right]. \tag{2.16}$$

For a Poisson point process the conditional intensity equals the intensity.

The probability density of a finite point process is completely determined by its conditional intensity (Stoyan et al., 1995; Moller and Waagepetersen, 2003). Certain point process models are convenient to formulate in terms of the conditional intensity rather than probability density. Using the conditional intensity also eliminates the normalizing constant needed for the probability density.

terchangeably.

However, the functional form of the conditional intensity must satisfy certain consistency conditions.

2.4.6 IID-cluster point process

Imposing the *independence assumption* among the features, the model in Eq. (2.13) reduces to the **IID-cluster point process** model[5] with density given by (Daley and Vere-Jones, 1988; Stoyan et al., 1995):

$$f(X) = p_c(|X|)\,|X|!\,[Up_f]^X, \tag{2.17}$$

where $|X|$ denotes the cardinality (number of elements) of X, p_f is a probability density on $\mathcal{X}$, referred to as the **feature density**, and $h^X \triangleq \prod_{x \in X} h(x)$, with $h^{\emptyset} = 1$ by convention, is the finite-set exponential notation.

Sampling from an IID-cluster can be accomplished by first sampling the number of points from the cardinality distribution p_c, and then sampling the corresponding number of points independently from the feature distribution p_f .

When the cardinality distribution p_c is Poisson with rate (or mean) ρ, we have the celebrated **Poisson point process** (Daley and Vere-Jones, 1988; Stoyan et al., 1995) with the density given by

$$f(X) = \rho^{|X|}\,e^{-\rho}\,[Up_f]^X. \tag{2.18}$$

The Poisson point process model is completely determined by the intensity function $\lambda = \rho p_f$ (Daley and Vere-Jones, 1988). Note that the Poisson cardinality distribution is described by a single non-negative number ρ, hence there is only one degree of freedom in the choice of cardinality distribution for the Poisson point process model.

[5] Later in this thesis, for brevity, sometimes the IID-cluster point process model is briefly called the **IID-cluster model**.

2.4.7 Finite Gibbs model

A well-known general model that accommodates dependence between its elements is a **finite Gibbs process**, which has probability density of the form (Stoyan et al., 1995; Moller and Waagepetersen, 2003).

$$f(X) = \exp\left(V_0 + \sum_{i=1}^{|X|} \sum_{\{x_1,...,x_i\} \subseteq X} V_i(x_1,...,x_i) \right), \tag{2.19}$$

where V_i is called the i^{th} **potential**, given explicitly by

$$V_i(x_1,...,x_i) = \sum_{Y \subseteq \{x_1,...,x_i\}} (-1)^{|\{x_1,...,x_i\}|-|Y|} \log f(Y).$$

Gibbs models arise in statistical physics, where $\log(f(X))$ may be interpreted as the potential energy of the PP. The term $-V_1(x)$ can be interpreted as the energy required to create a single point at a location x, and the term $-V_2(x_1, x_2)$ can be interpreted as the energy required to overcome the force between the points x_1 and x_2.

Note that any hereditary probability density of a finite point process can be expressed in the Gibbs form (Baddeley et al., 2007), e.g., the Poisson point process is indeed a first order Gibbs model.

2.5 Set distances

2.5.1 Distance measures

A **distance** is a fundamental measure of dissimilarity between two objects. Hence, the notion of distance or metric is important to learning approaches without models (Jain, 2010; Amores, 2013; Pimentel et al., 2014).

Recall the definition of a distance function or metric on a non-empty space $\mathcal{S}$, a function $d : \mathcal{S}\times\mathcal{S} \to [0;1)$ is called a metric if it satisfies the following three axioms:

1. (Identity) $d(x,y) = 0$ if and only if $x = y$;

2. (Symmetry) $d(x,y) = d(y,x)$ for all $x, y \in \mathcal{S}$;
3. (Triangle inequality) $d(x,y) \leq d(x,z) + d(z,y)$ for all $x, y, z \in \mathcal{S}$.

Our interest lies in the **set distances** between two finite subsets $X = \{x_1, ..., x_m\}$ and $Y = \{y_1, ..., y_n\}$ of a metric space $(\mathcal{W}, \underline{d})$, where $\mathcal{W}$ is closed and bounded observation window, and $\underline{d}$ denotes the **base distance** between the elements of $\mathcal{W}$. Note that $\underline{d}$ is usually taken as the Euclidean distance when $\mathcal{W}$ is a subset of $\mathbb{R}^n$.

There are several set distances which can be used for PP data[6], namely the Hausdorff (Huttenlocher et al., 1993b), Wasserstein (Hoffman and Mahler, 2004), and OSPA (Schuhmacher et al., 2008) distances.

2.5.2 Hausdorff distance

The **Hausdorff distance** between two non-empty sets X and Y is defined by

$$d_{\mathrm{H}}(X,Y) = \max\left\{\max_{x\in X}\min_{y\in Y}\underline{d}(x,y), \max_{y\in Y}\min_{x\in X}\underline{d}(x,y)\right\}, \tag{2.20}$$

Note that the Hausdorff distance is not defined when either X or Y is empty.

In addition to being a metric, the Hausdorff distance is easy to compute and was traditionally used as a measure of dissimilarity between binary images. It gives a good indication of the dissimilarity in the visual impressions that a human would typically perceive between two binary images (Huttenlocher et al., 1993a; Rucklidge, 1995).

2.5.3 Wasserstein distance

The **Wasserstein distance** (also known as *Optimal Mass Transfer distance* (Schuhmacher et al., 2008)) of order $p \geq 1$ between two sets X and Y is defined by (Hoffman and Mahler, 2004)

[6] A multi-set can be equivalently expressed as a set by augmenting the multiplicity of each element, i.e., a multi-set with elements x_1 repeated N_1 times, ..., x_m repeated N_m times, can be represented as the set $\{(x_1, N_1), ..., (x_m, N_m)\}$.

$$d_{\mathtt{W}}^{(p)}(X,Y) = \min_{C} \left(\sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j} \underline{d}\left(x_i, y_j\right)^p \right)^{\frac{1}{p}}, \tag{2.21}$$

where $C = (c_{i,j})$ is an $m \times n$ transportation matrix (recall that m and n are the cardinalities of X and Y, respectively), i.e., $c_{i,j}$ are non-negative and satisfies:

$$\sum_{j=1}^{n} c_{i,j} = \frac{1}{m} \text{ for } 1 \leq i \leq m, \tag{2.22a}$$

$$\sum_{i=1}^{m} c_{i,j} = \frac{1}{n} \text{ for } 1 \leq j \leq n. \tag{2.22b}$$

Note that similar to the Hausdorff distance the Wasserstein distance is a metric (Hoffman and Mahler, 2004) and is not defined when either X or Y is empty.

The Wasserstein distance can be considered as the *Earth Mover's distance* (Rubner et al., 1998) *adapted for PPs* (Hoffman and Mahler, 2004). Consider the sets $X = \{x_1, ..., x_m\}$ and $Y = \{y_1, ..., y_n\}$ as collections of earth piles at x_i each with mass $1/m$ and y_j each with mass $1/n$, i.e., the total mass of each collection is 1, and suppose that the cost of moving a mass of earth over a distance is given by the mass times the distance. Then the Wasserstein distance can be considered as the *minimum cost* needed to build one collection of earth piles from the other. This is illustrated in Figure 2.5 and Figure 2.6, where the arrows correspond to the optimal movements of the earth piles.

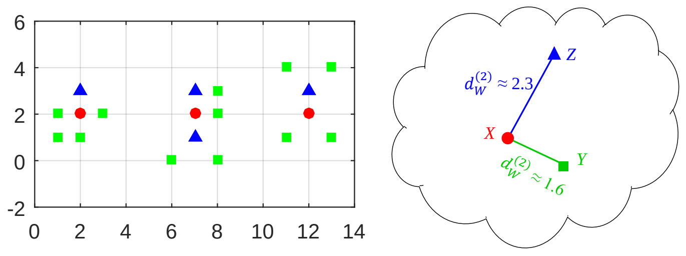
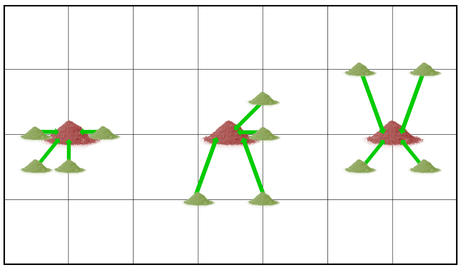


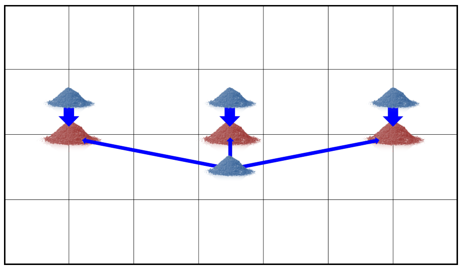
Figure 2.5: Left: Sets X (red ●), Y (green ■), and Z (blue ▲) in $\mathbb{R}^2$. Right: Abstract impression of the Wasserstein distances between the finite sets X, Y, and Z.

2.5.4 OSPA distance

The **Optimal SubPattern Assignment (OSPA)** (Schuhmacher et al., 2008) distance of order $p \geq 1$, and cutoff $c > 0$, is defined by



(a) To compute $d_{\mathtt{W}}^{(p)}(X,Y)$: move Y's earth piles (green) to form X's earth piles (red).



(b) To compute $d_{\mathtt{W}}^{(p)}(X,Z)$: move Z's earth piles (blue) to form X's earth piles (red).

Figure 2.6: Earth mover's interpretation of the Wasserstein distance. Sets X, Y, and Z in Figure 6.2 are considered as collections of earth piles. The blue/green arrows represent the amounts and directions of the transportations of the blue/green earth piles.

$$d_{\mathtt{O}}^{(p,c)}(X,Y) = \left(\frac{1}{n}\left(\min_{\pi\in\Pi_n}\sum_{i=1}^{m}\underline{d}^{(c)}\left(x_i,y_{\pi(i)}\right)^p + c^p\left(n-m\right)\right)\right)^{\frac{1}{p}}, \tag{2.23}$$

if $n \geq m > 0$ (recall that m and n are the cardinalities of X and Y, respectively), and $d_{\mathtt{O}}^{(p,c)}(X,Y) = d_{\mathtt{O}}^{(p,c)}(Y,X)$ if $m > n > 0$, where Π_n is the set of permutations of $\{1,2,...,n\}$, $\underline{d}^{(c)}(x,y) = \min\left(c, \underline{d}\left(x,y\right)\right)$. Further $d_{\mathtt{O}}^{(p,c)}(X,Y) = c$ if one of the set is empty; and $d_{\mathtt{O}}^{(p,c)}(\varnothing,\varnothing) = 0$. The two adjustable parameters p, and c, are interpreted as the outlier penalty and the cardinality sensitivity, respectively.

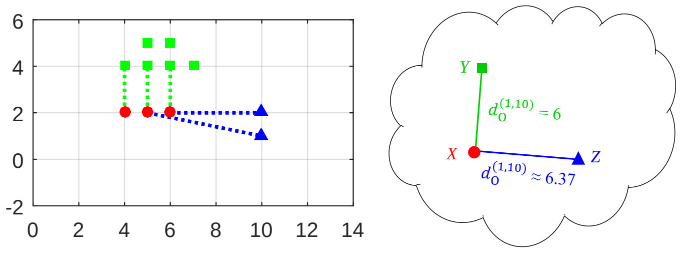


Figure 2.7: Computing OSPA distance. Left: Sets X (red ●), Y (green ■), and Z (blue ▲) in $\mathbb{R}^2$; the dotted lines are optimal assignments between the elements of X and Y, Z respectively. Right: Abstract impression of the OSPA distances between the sets X, Y, and Z.

Assuming $p = 1$, to compute Eq. (2.23), we assign m elements of Y to the m elements of X so as to minimize the total adjusted distance $\underline{d}^{(c)}$ (see Figure 2.7 for illustration). This can be achieved via an optimal assignment procedure such as the Hungarian method (Kuhn, 1955, 1956). For each of the $(n-m)$ elements in Y which are not assigned, we set a fixed distance of c. The OSPA distance is simply the average of these n distances (i.e., m optimal adjusted distances and $(n - m)$ fixed distances c). Thus, the OSPA distance has a physically intuitive interpretation as the "per element" dissimilarity that incorporates both features and cardinality (Schuhmacher et al., 2008).

CHAPTER 3

MODEL-BASED CLASSIFICATION FOR POINT PATTERN DATA

Chapter's key points:

- Existing methods for PP classification
- Model-based PP classification
- Learning point process models
- Numerical experiments

In the previous chapter, we reviewed the problem of classification in general, and introduced the background of point process theory. In this chapter, we develop a **maximum likelihood estimate** of the IID-cluster model to use in PP classification. Numerical experiments show that the proposed method performs well on both simulated and real data.

3.1 Existing methods for PP classification

As previously discussed, classification is the supervised learning task that uses fully observed training input-output pairs $\mathcal{D}_{\text{train}} = \{(X_n, y_n)\}_{n=1}^{N_{\text{train}}}$ to determine the output class label $y \in \{1, \ldots, K\}$ of each input input observation (Bishop, 2006; Murphy, 2012). This fundamental machine learning task is the most widely used form of supervised machine learning, with applications spanning many fields of study (see Section 2.1).

The classification for PP data is more commonly known as **multiple in-**

stance classification (**MIC**) (Amores, 2013). According to Amores (2013), approaches for solving PP learning problems in general can be divided into three paradigms, namely Instance-Space, Embedded-Space, and Bag-Space paradigm. These paradigms differ in the way they exploit data at the *point-level or PP-level*. **Instance-Space** is the paradigm exploiting data at the point-level. Specifically, in this paradigm, point-level labels are required to construct a point classifier. Then, given a test PP, we use this point classifier to classify points in the PP, and use some assumption (such as the standard MI assumption mentioned in Chapter 1) to infer the PP label (Dietterich et al., 1997; Maron and Lozano-Pérez, 1998; Zhang et al., 2001b; Andrews et al., 2003).[1]

At the global level, the **Embedded-Space** paradigm maps all PPs to vectors of a fixed dimension, which are then also processed by some point classifiers. Examples of this approach include (Dong, 2006; Bunescu and Mooney, 2007; Gärtner et al., 2002). On the other hand, the **Bag-Space** paradigm addresses the problem at the most fundamental level by operating directly on the PPs. The philosophy of the Bag-Space paradigm is to preserve the information content of the data, which could otherwise be compromised through the data transformation process (as in the Embedded-Space approach). In this work, the proposed methods for PP classification belong to this last paradigm — the *Bag-Space paradigm*.

Existing methods in the Bag-Space paradigm include **distance-based techniques** (such as k-nearest neighbour (k-NN)) using set-distances, e.g., the Hausdorff (Huttenlocher et al., 1993b), Chamfer (Gavrila and Philomin, 1999), and Earth Mover's (Zhang et al., 2007; Rubner et al., 1998) distance.[2] Such classifiers do not require any underlying data models and are simple to use. However, they may perform poorly with high dimensional inputs due to the curse

[1] Note that many of existing methods for PP classification in the Instance-Space paradigm are developed for solving only two-class classification (i.e., *binary classification*) (Cheplygina et al., 2015). In practice, however, many applications are indeed multiclass classification, hence these binary classifiers need to be extended by applying some scheme, such as One-vs-All or All-vs-All (Rifkin and Klautau, 2004).

[2] The distance-based technique for PP classification will be discussed further in Chapter 6.

of dimensionality, and are often computationally intractable for large datasets (Murphy, 2012), not to mention that the decision procedure is unclear. On the other hand, knowledge of the underlying data model can be used to exploit statistical patterns in the training data, and to devise optimal decision procedures. However, to the best of our knowledge, **model-based classifiers** for PPs have not been investigated, despite the fundamental role of statistical models in machine learning.

3.2 Model-based PP classification

Using the notion of probability density for the point process from Section 2.4, the standard model-based classification formulation can extend to PP classification:

- In the *training phase*, we seek likelihoods that "best" fit the training data. Specifically, for each class $k \in \{1, \ldots, K\}$, we seek a likelihood function $f(\cdot|y = k)$ that best fits the training input PPs in $\mathcal{D}_{\text{train}}^{(k)} = \{X : (X, k) \in \mathcal{D}_{\text{train}}\}$, according to criteria such as **maximum likelihood**, or **Bayes optimal** if suitable priors on the likelihoods are available.
- In the *classifying phase*, the likelihoods (learned in the training phase) are used to classify test observations. When a PP X is passed to query its label, the Bayes classifier returns the **mode of the class label posterior** $p\,(y = k \mid X)$ computed from the likelihood and the class prior p via Bayes' rule:
$$p\,(y = k \mid X) \propto p(y = k) f(X \mid y = k). \tag{3.1}$$

The simplest choices for the class priors are the uniform distribution, and the categorical distribution, usually estimated from the training data via

$$p(y = k) = \frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} \delta_{y_n}[k], \tag{3.2}$$

where $\delta_i[j]$ is the Kronecker delta, which takes on the value 1 when $i = j$, and zero otherwise. Hence, the main computational effort in model-based classification lies in the training phase.

3.3 Learning point process models

Learning the likelihood function for class k boils down to finding the value(s) of the parameter θ_k such that the (parameterized) probability density $f(\cdot \mid y = k, \theta_k)$ best explains the observations $X_1, ..., X_N$ in $\mathcal{D}_{\text{train}}^{(k)}$. In this section, we consider a fixed class label and its corresponding observations $X_1, ..., X_N$.

Methods for learning point process models have been available since the 1970s, see e.g., (Moller and Waagepetersen, 2003; Baddeley et al., 2007). We briefly summarize some recognized techniques, then develop the methods of **maximum likelihood (ML)**, and **maximum a posteriori probability (MAP)** for learning **IID-cluster models** as tractable PP classification solutions (Section 3.3.5 and Section 3.3.6).

3.3.1 Model fitting via summary statistics

The *method of moments* seeks the parameter θ such that the expectation of a given statistic of the model point process parameterized by θ is equal to the statistic of the observed PPs (Baddeley et al., 2007). However, this approach is only tractable when the solution is unique and the expectation is a closed form function of θ, which is usually not the case in practice, not to mention that moments are difficult to calculate.

The *method of minimum contrast* seeks the parameter θ that minimizes some dissimilarity between the expectation of a given summary statistic (e.g., the K-function) of the model point process and that of the observed PPs (Baddeley et al., 2007). Provided that the dissimilarity functional is convex in the parameter θ, this approach can avoid some of the problems in the method of moments. However, in general, the statistical properties of the solution are not well understood, not to mention the numerical behaviour of the algorithm used to determine the minimum.

3.3.2 Maximum likelihood

In the *maximum likelihood (ML)* approach, we seek the ***ML estimate (MLE)*** of θ:

$$\mathrm{MLE}(f(\cdot \mid \theta); X_{1:N}) \triangleq \operatorname*{argmax}_{\theta} \left(\prod_{n=1}^{N} f(X_n \mid \theta) \right) \tag{3.3}$$

The MLE has some desirable statistical properties such as asymptotic normality and optimality (Baddeley et al., 2007). However, in general, there are problems with non-unique maxima, and moreover, analytic MLEs for point process models are not available because the likelihood Eq. (2.19) of many Gibbs models contains an intractable normalizing constant which is a function of θ (Moller and Waagepetersen, 2003).

To the best of our knowledge, currently there is *no general ML technique* for learning generic models such as Gibbs from real data. Numerical approximation methods such as (Ogata and Tanemura, 1984) and Markov Chain Monte Carlo (MCMC) (Geyer and Møller, 1994) are highly specific to the chosen model, computationally intensive, and require careful tuning to ensure good performance. Nonetheless, simple models such as the IID-cluster model Eq. (2.17), admit an analytic MLE (detailed in Section 3.3.5).

Remark: The method of moment replaces the ML estimation equation

$$\frac{d}{d\theta} \left(\sum_{n=1}^{N} log(f(X_n \mid \theta)) \right) = 0 \tag{3.4}$$

by an unbiased sample approximation of the general equation $\mathbb{E}_{\theta}\left[\Psi(\theta, X)\right] = 0$, i.e.,

$$\sum_{n=1}^{N} \Psi(\theta, X_n) = 0 \tag{3.5}$$

For example, $\Psi(\theta, X_n) = \frac{d}{d\theta} log(f(X_n \mid \theta))$ results in ML since it is well-known from classical ML theory that Eq. (3.4) is an unbiased estimating equation. Setting $\Psi(\theta, X_n) = T(X_n) - \mathbb{E}_{\theta}[T(X)]$ (i.e., the difference between the empirical value and the expectation of the summary statistic) results in the method of moments since $T(X_n)$ is an unbiased estimator of $\mathbb{E}_\theta[T(X)]$. *Takacs-Fiksel* is another well-known family of estimating equations (Takacs, 1986; Fiksel, 1988).

3.3.3 Maximum pseudolikelihood

Maximum pseudolikelihood (MPL) estimation, proposed by Besag (1975, 1977), is a powerful approach that avoids the intractable normalizing constant present in the likelihood and retains desirable properties, such as consistency and asymptotic normality in a large-sample limit (Baddeley et al., 2007). The key idea is to replace the likelihood of a point process (with parameterized conditional intensity $\lambda_\theta(u; X)$) by the **pseudolikelihood**:

$$\mathrm{PL}(\theta; X_{1:N}) = \prod_{n=1}^{N} e^{-\int \lambda_\theta(u;X_n)du} \left[\lambda_\theta(\cdot; X_n)\right]^{X_n}. \tag{3.6}$$

The rationale behind this strategy is discussed in (Besag, 1975). Up to a constant factor, the pseudolikelihood is indeed the likelihood if the model is Poisson, and is approximately equal to the likelihood if the model is close to Poisson. The pseudolikelihood may be regarded as an approximation to the likelihood which ignores the inter-point dependence.

An MPL algorithm has been developed by Baddeley and Turner (2000) for point processes with sufficient generality, such as Gibbs whose conditional intensity has the form

$$\lambda(u, X) = \exp\left(\sum_{i=1}^{|X|+1} \sum_{\{x_1,\dots,x_{i-1}\}\subseteq X} V_i(u, x_1, \dots, x_{i-1}) \right). \tag{3.7}$$

By turning the pseudolikelihood of a general point process into a classical Poisson point process likelihood, MPL can be implemented with standard generalised linear regression software (Baddeley and Turner, 2000). Due to its versatility, the Baddeley-Turner algorithm is the preferred model fitting tool for point processes.

The main hurdle in the application of the Baddeley-Turner algorithm to PP classification is the computational requirement. While this may not be an issue in spatial statistics applications, the algorithm's computational requirement is still prohibitive with large data sets often encountered in PP learning. On the other hand, the disadvantages of MPL (relative to ML) such as small-sample bias and inefficiency (Besag, 1977; Jensen and Møller, 1991) become less significant with large data. Efficient algorithms for learning general point process models is an ongoing area of research. Nonetheless, the Baddeley-Turner algorithm is a promising tool for complex PP classification problems.

3.3.4 Bayesian approach

Apart from the aforementioned ML method, which belongs to the class of frequentist approach, the *Bayesian approach* is based on the *posterior distribution* of the model parameter, which incorporates prior knowledge of the parameters of interest. In the Bayesian approach, the optimal estimate of parameter θ can be either the posterior mean, median, or mode, however, the mode of the posterior is the most common choice due to its computational efficiency (Murphy, 2012).

In the *maximum a posteriori probability (MAP)* method, we seek an estimate of θ which maximizes the **posterior distribution** $p(\theta \mid X_{1:N})$, i.e.,

$$\mathrm{MAP}(p(\theta), f(\cdot \mid \theta); X_{1:N}) \triangleq \operatorname*{argmax}_{\theta} \left(p(\theta) \prod_{n=1}^{N} f(X_n \mid \theta) \right), \tag{3.8}$$

where $p(\theta)$ is the **prior distribution** which represents our knowledge of the parameter θ before any data are obtained (Wasserman, 2004).

It can be observed from Eq. (3.8) that the MAP estimate of θ is equivalent to the MLE (Section 3.3.2) if the prior $p(\theta)$ is uniform. In other words, the ML estimation can be seen as a special case of the MAP estimation. For sufficiently large datasets, the MAP (with an arbitrary prior) and ML estimation yield approximately the same results, although in general they need not agree (Wasserman, 2004).

When the prior $p(\theta)$ and posterior $p(\theta \mid X_{1:N})$ are in the same family of distributions, $p(\theta)$ is called a *conjugate prior* with respect to the likelihood $f(X \mid \theta)$ (Wasserman, 2004). In addition, if a conjugate prior is also in the same family as the likelihood, then it is called a *natural conjugate prior* (Gelman et al., 2004). For example, consider a MAP estimation for the mean of a Gaussian likelihood (whose covariance is known); if the prior is Gaussian, then the posterior is also Gaussian. Conjugate priors can bring mathematical and computational convenience, since it gives a parametric form for the posterior distribution (Gelman et al., 2004).

MAP estimates can be computed analytically if the mode of the posterior has a closed form solution, such as when conjugate priors are used. Otherwise, numerical approximation methods such as the conjugate gradient method (Saad, 2003) and Markov Chain Monte Carlo (MCMC) (Geyer and Møller, 1994) have to be deployed. These methods, however, are computationally intensive and require careful tuning to ensure good performance.

The appeal of MAP estimation is that it is a natural way to incorporate the prior information about the parameters (if this exists) with the observed data (Wasserman, 2004). However, MAP estimation has several drawbacks, such as problems with non-unique maxima, no measure of uncertainty, untypical summary of the posterior, and is dependent on the parameterization (Murphy, 2012).

There is scant literature on Bayesian learning for point processes and most studies rely on numerical approximation methods, such as MCMC, rather than analytical solutions. For example, Byers and Raftery (2002) propose to use MCMC to estimate a piecewise constant rate point process specified by a Voronoi tiling model. In (Moller and Waagepetersen, 2003), the authors review a few contributions on Bayesian inference for Markov point processes using numerical techniques such as Metropolis-Hastings and MCMC.

To the best of our knowledge, currently there is *no general analytical MAP* *technique* for learning point process models. Nonetheless, it can be shown that simple models, such as the IID-cluster model Eq. (2.17), admit an analytic MAP estimate (detailed in Section 3.3.6).

3.3.5 ML learning for IID-clusters

Computationally efficient algorithms for learning point process models are important because PP learning usually involves large data sets (compared to applications in spatial statistics). Since learning a general point process is computationally prohibitive, the IID-cluster model Eq. (2.17) provides a good trade-off between tractability and versatility by neglecting interactions between the points.

Since an IID-cluster model is uniquely determined by its cardinality and feature distributions, we consider a parameterization of the form:

$$f(X \mid \xi, \varphi) = p_\xi(|X|)\,|X|!\,U^{|X|} p_\varphi^X . \tag{3.9}$$

where p_ξ and p_φ, are the parameterized *cardinality distribution* and *feature distribution* respectively. Learning the underlying parameters of an IID-cluster model amounts to estimating the parameter $\theta = (\xi, \varphi)$ from training input data.

The form of the IID-cluster likelihood function allows the MLE to *separate* into the MLE of the cardinality parameter ξ and MLE of the feature parameter φ. This is stated more concisely in Proposition 1 (the proof is straightforward, but is included for completeness).

Proposition 1. *Let $X_1, ..., X_N$ be N i.i.d. realizations of an IID-cluster with parameterized cardinality distribution p_ξ and feature density p_φ. Then the MLE of (ξ, φ), is given by*

$$\hat{\xi} = \text{MLE}\left(p_\xi; |X_1|, ..., |X_N|\right), \tag{3.10}$$

$$\hat{\varphi} = \text{MLE}\left(p_\varphi; \biguplus_{n=1}^{N} X_n\right), \tag{3.11}$$

where $\uplus$ denotes disjoint union.

Proof. Using Eq. (3.9), we have

$$\prod_{n=1}^{N} f(X_n \mid \xi, \varphi) = \prod_{n=1}^{N} p_\xi(|X_n|)\,|X_n|!\,U^{|X_n|} p_\varphi^{X_n} \tag{3.12}$$

$$= \prod_{n=1}^{N} |X_n|!\,U^{|X_n|} \prod_{n=1}^{N} p_\xi(|X_n|) \prod_{n=1}^{N} p_\varphi^{X_n} \tag{3.13}$$

Hence, to maximize the likelihood we simply maximize the second and last products in the above separately. This is achieved with Eq. (3.10) and Eq. (3.11). QED

Observe from Proposition 1 that the MLE of the feature density parameter is identical to that used in the *naive Bayes (NB) model* (Bishop, 2006, pp. 380–381). For example, if the feature density is a *Gaussian* $\mathcal{N}(\cdot \mid \mu, \Sigma)$, then the MLEs of the mean and covariance are:

$$\hat{\mu} = \frac{1}{N} \textstyle\sum_{n=1}^{N} \sum_{x \in X_n} x, \tag{3.14}$$

$$\hat{\Sigma} = \frac{1}{N} \textstyle\sum_{n=1}^{N} \sum_{x \in X_n} (x - \hat{\mu})(x - \hat{\mu})^{\mathrm{T}}. \tag{3.15}$$

If the feature density is a mixture model of the form:

$$p_\varphi(x) = \sum_{k=1}^{K} \pi_k\, p_{\theta_k}(x) \tag{3.16}$$

where parameters $\varphi = \{\pi_{1:K}, \theta_{1:K}\}$, $\pi_{1:K}$ are prior probabilities, $\theta_{1:K}$ are parameters of component distributions, then MLE of φ can be found using the *expectation–maximization (EM)* algorithm (Dempster et al., 1977; Little and Rubin, 2002). The mixture feature density, e.g., Gaussian mixture, is suitable for PP data with complex shapes in features, such as the Texture image data described in Section 3.4.2.

Consequently, compared to learning NB model, learning the IID-cluster model requires only one additional task of computing the MLE of the cardinality parameters, which is relatively inexpensive.

For a *categorical* cardinality distribution, i.e., $\xi = (\xi_1, ..., \xi_M)$ where $\xi_k = \Pr(|X| = k)$ and $\sum_{k=1}^{M} \xi_k = 1$, the MLE of the cardinality parameter is given by

$$\hat{\xi}_k \;\; = \;\; \frac{1}{N} \sum_{n=1}^{N} \delta_k[|X_n|]. \tag{3.17}$$

Since there are M parameters $\xi_1, ..., \xi_M$, a sufficiently large training dataset (significantly larger than M) is needed to avoid over-fitting. Alternatively, the standard practice of placing a Laplace prior on the cardinality distribution can be applied, i.e., replacing the above equation by $\hat{\xi}_k \propto \epsilon + \sum_{n=1}^{N} \delta_k[|X_n|]$, where ϵ is a small number.

If the training dataset is significantly small, a cardinality distribution with a small number of parameters should be used, e.g., a *Poisson* distribution parameterized by the rate $\xi = \rho$, in which case the MLE is given by

$$\hat{\rho} = \frac{1}{N} \sum_{n=1}^{N} |X_n|. \tag{3.18}$$

It is also possible to develop MLEs for other families of cardinality distributions such as *binomial*, *multi-Bernoulli*, etc.

3.3.6 MAP learning for IID-clusters

The form of the IID-cluster likelihood function also allows the MAP estimate to *separate* into the MAP estimates of the cardinality parameter ξ and MAP estimate of the feature parameter φ, if the prior on (ξ, φ) separates into priors on ξ and φ. This is stated more concisely in Proposition 2 (the proof is straightforward, but is included for completeness).

Proposition 2. *Let $X_1, ..., X_N$ be N i.i.d. realizations of an IID-cluster with parameterized cardinality distribution p_ξ and feature density p_φ. If the prior $p(\xi, \varphi)$ =*

$p(\xi)\,p(\varphi)$, then the MAP estimate of (ξ,φ), is given by

$$\hat{\xi} = \mathrm{MAP}\left(p(\xi), p_{\xi}; |X_1|, ..., |X_N|\right), \tag{3.19}$$

$$\hat{\varphi} = \mathrm{MAP}\left(p(\varphi), p_{\varphi}; \biguplus_{n=1}^{N} X_n\right), \tag{3.20}$$

where $\biguplus$ denotes disjoint union.

Proof. Using Eq. (3.9), we have

$$p(\xi,\varphi)\prod_{n=1}^{N} f(X_n \mid \xi,\varphi) = p(\xi)\,p(\varphi)\prod_{n=1}^{N} p_{\xi}(|X_n|)\,|X_n|!\,U^{|X_n|}p_{\varphi}^{X_n} \tag{3.21}$$

$$= \prod_{n=1}^{N}|X_n|!\,U^{|X_n|}\prod_{n=1}^{N}p(\xi)\,p_{\xi}(|X_n|)\prod_{n=1}^{N}p(\varphi)\,p_{\varphi}^{X_n} \tag{3.22}$$

Hence, to maximize the posterior we simply maximize the second and last products in the above separately. This is achieved with Eq. (3.19) and Eq. (3.20). QED

Note that in MAP estimation, conjugate priors are usually used to ensure the MAP estimates have close-form solutions. The conjugate prior for *cardinality density parameter* ξ depends on the form of p_{ξ}. For example, for a Poisson likelihood parameterized by the rate ρ, a gamma distribution $\mathrm{Ga}(\cdot \mid \alpha,\beta)$ can be used as a conjugate prior which yields the posterior:

$$p(\rho \mid |X_1|, ..., |X_N|) \propto \mathrm{Ga}\left(\rho \mid \alpha + \sum_{n=1}^{N}|X_n|, \beta + N\right). \tag{3.23}$$

For a categorical likelihood parameterized by $\xi = (\xi_0, ..., \xi_M)$ belongs to the unit M-simplex, a Dirichlet distribution $\mathrm{Dir}(\cdot \mid \eta_0, ..., \eta_M)$ with $(\eta_0, ..., \eta_M)$ belongs to the unit M-simplex , can be used as a conjugate prior which yields the posterior (Minka, 2003):

$$p(\xi \mid |X_1|, ..., |X_N|) = \mathrm{Dir}\left(\xi \mid c_0 + \eta_0, ..., c_M + \eta_M\right), \tag{3.24}$$

where $c_m = \sum_{n=1}^{N}\delta_m[|X_n|]$.

The conjugate prior for *feature density parameter* φ depends on the form of p_φ. For example, a conjugate prior for mean μ of a Gaussian with known variance σ^2 is a Gaussian $\mathcal{N}(\cdot \mid \mu_0, \sigma_0^2)$ which yields the posterior (Murphy, 2007):

$$p(\mu \mid \uplus_{n=1}^{N} X_n) = \mathcal{N}\left(\mu \mid \left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1} \times \left(\frac{\mu_0}{\sigma_0^2} + \frac{n\overline{x}}{\sigma^2}\right), \left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}\right), \tag{3.25}$$

where $\overline{x} = \frac{1}{\sum_{n=1}^{N} |X_n|} \sum_{x \in X_1} \cdots \sum_{x \in X_n} x$.

As another example, a conjugate prior for variance σ^2 of a Gaussian with known mean μ is an inverse gamma distribution $\text{IG}(\cdot \mid \alpha, \beta)$ which yields the posterior (Jordan, 2010):

$$p(\sigma^2 \mid \uplus_{n=1}^{N} X_n) = \text{IG}(\sigma^2 \mid \alpha + \frac{\sum_{n=1}^{N} |X_n|}{2}, \beta + \frac{1}{2} \sum_{x \in X_1} \cdots \sum_{x \in X_n} (x - \mu)^2), \tag{3.26}$$

3.4 Numerical experiments

This section presents the classification experiments using one of the simplest point process models — the *Poisson point process model* (may briefly called *Poisson model*, see Section 2.4). The first experiment uses simulated data to illustrate the benefit of cardinality information when the features between the classes are similarly distributed. The second experiment shows the versatility of the framework by using real data from the Texture images dataset (Lazebnik et al., 2005). For simplicity, we use a *uniform class prior* in the classification phase.

In the following experiments, the ML technique (Section 3.3.5) is used to learn the data models instead of the MAP method (Section 3.3.6) since we do not have any prior information on the model parameters. We benchmark the classification performance of the Poisson model with the *naive Bayes (NB) model*, which shares the same assumption that the features themselves are i.i.d. While the NB model is not intended for PP data, it has been applied to MI learning, such as (Joachims, 1996; McCallum and Nigam, 1998; Maron, 1961; Csurka et al., 2004; Cadez et al., 2000), and can be regarded as a special case of the

IID-cluster model.[3]

3.4.1 Classification with simulated data

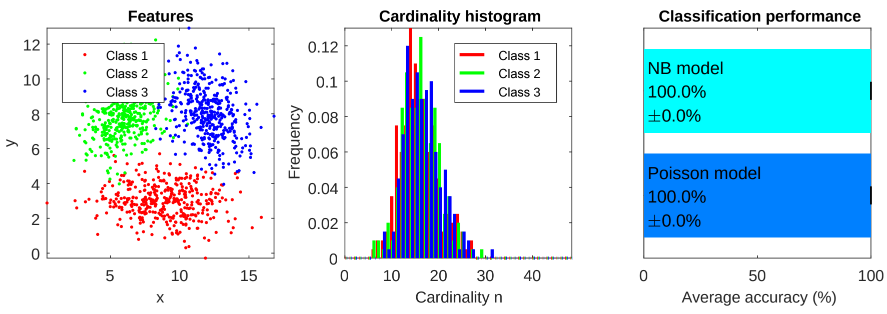
In this experiment, three datasets representing three diverse scenarios of PP data are considered. Each dataset consists of three clusters, and each cluster consists of 200 PPs generated from a Poisson model with a 2-D Gaussian intensity parameterized by (ρ, μ, Σ). As previously discussed in Section 2.4, a PP can be generated from a Poisson model (ρ, μ, Σ), by first sampling the number of points from the Poisson distribution with rate ρ, then sampling the corresponding number of points independently from the Gaussian with mean and covariance (μ, Σ). The parameters of the Poisson models used to generate data in this experiment are shown in Figure 3.1.

Three diverse scenarios of PP data are considered: in *dataset (i)*, the features of the PPs from each cluster are well separated from those of the other clusters, but their cardinalities significantly overlap (Figure 3.1a); in *dataset (ii)*, the cardinalities of the PPs from each cluster are well separated from those of the other clusters, but their features significantly overlap (Figure 3.1b); *dataset (iii)* is a mix of (i) and (ii) (Figure 3.1c).

In the training phase, ML is used to learn the parameters of the NB model and Poisson model from a training dataset consisting of 600 PPs (200 per class). Note that the feature densities of the Poisson model are indeed identical to those of the NB model.

In the test phase, 10 different test sets, each consisting of 300 PPs (100 per class), are used. The average classification accuracies are reported in Figure 3.1. Observe that in *dataset (i)*, both the NB and Poisson models perform extremely well. This is due to the fact that features of the PPs from different clusters are very well separated (Figure 3.1a), hence both the NB and Poisson classifiers, which can well exploit the feature information, deliver very good performance.

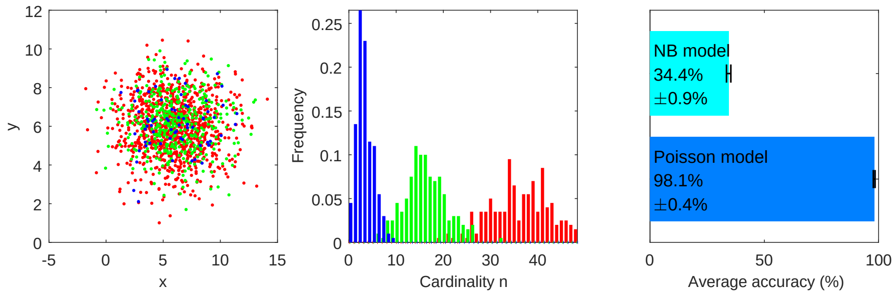
[3] We also benchmark with the distance-based classifiers, see Chapter 6.



Poisson model parameters

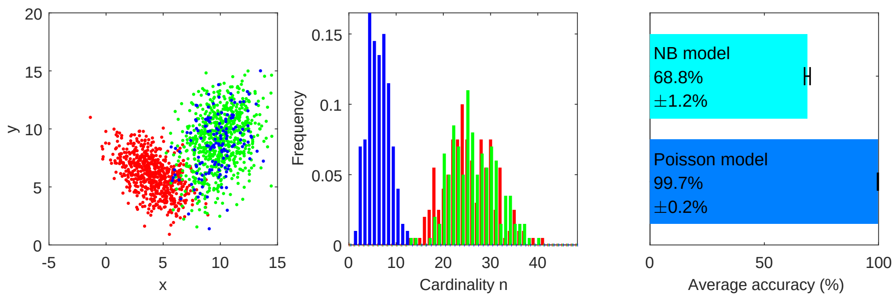
	ρ	μ	Σ
🟥	16	$\begin{bmatrix} 9 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 1 \end{bmatrix}$
🟩	16	$\begin{bmatrix} 6 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 2 & 0.6 \\ 0.6 & 2 \end{bmatrix}$
🟦	16	$\begin{bmatrix} 12 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 2 & -0.6 \\ -0.6 & 2 \end{bmatrix}$

(a) Dataset (i)



	ρ	μ	Σ
🟥	37	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$
🟩	16	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$
🟦	3	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$

(b) Dataset (ii)



	ρ	μ	Σ
🟥	26	$\begin{bmatrix} 4 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 3 & -1.6 \\ -1.6 & 3 \end{bmatrix}$
🟩	26	$\begin{bmatrix} 10 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 3 & 1.6 \\ 1.6 & 6 \end{bmatrix}$
🟦	6	$\begin{bmatrix} 10 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 3 & 1.6 \\ 1.6 & 6 \end{bmatrix}$

(c) Dataset (iii)

Figure 3.1: Simulated data and classification performance in three diverse scenarios of PP data. Dataset (i): well-separated in feature but overlapping in cardinality; dataset (ii): well-separated in cardinality but overlapping in feature; dataset (iii): a mix of (i) and (ii).

However, in *datasets (ii)* and *(iii)*, where features overlap, the Poisson model outperforms the NB since the Poisson model can exploit both the feature as well as the cardinality information in PPs. This helpful characteristic of the Poisson model is reflected in its density function Eq. (2.18) (or more generally, in the IID-cluster model density Eq. (2.17)). Compared to the NB likelihood Eq. (1.1), which includes only the feature density, the density function of the Poisson model (or IID-cluster model) consists of both feature density p_f and cardinality density p_c, hence ability to account for both feature and cardinality information of PP data.

3.4.2 Classification with Texture images dataset

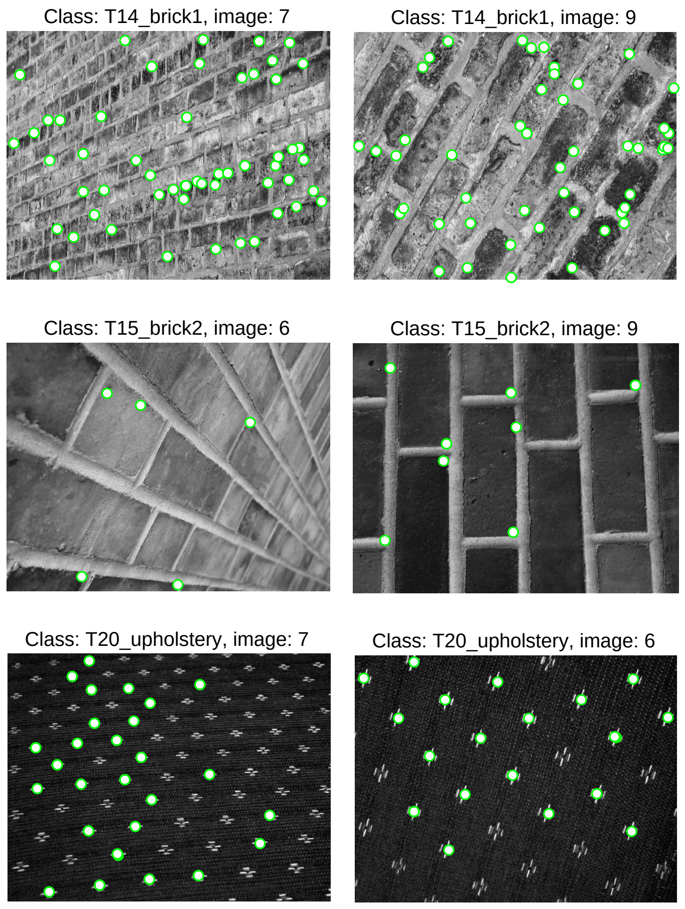

Class: T14_brick1, image: 7
Class: T14_brick1, image: 9
Class: T15_brick2, image: 6
Class: T15_brick2, image: 9
Class: T20_upholstery, image: 7
Class: T20_upholstery, image: 6

Figure 3.2: Example images from classes "T14_brick1", "T15_brick2", and "T20_upholstery" of the Texture dataset. Circles mark the detected SIFT keypoints.

This experiment involves clustering images from the classes "T14 brick1", "T15 brick2", and "T20 upholstery" of the *Texture images dataset* (Lazebnik et al., 2005). In this dataset, each class consists of 40 images, with some examples shown in Figure 3.2. Each image is compressed into a PP of 2-D features by first applying the scale-invariant feature transform (SIFT) algorithm (using the VLFeat library (Vedaldi and Fulkerson, 2008)) to produce a PP of 128-D SIFT features, which is then further compressed into a 2-D PP by principal component analysis (PCA). Figure 3.3 shows the superposition of the 2-D PPs from the three classes along with their cardinality histograms.

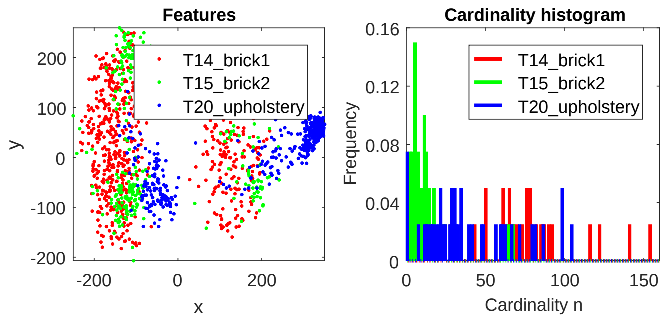
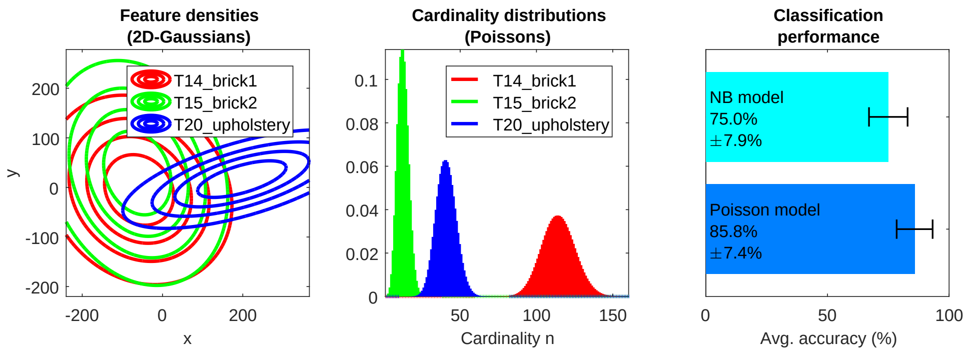


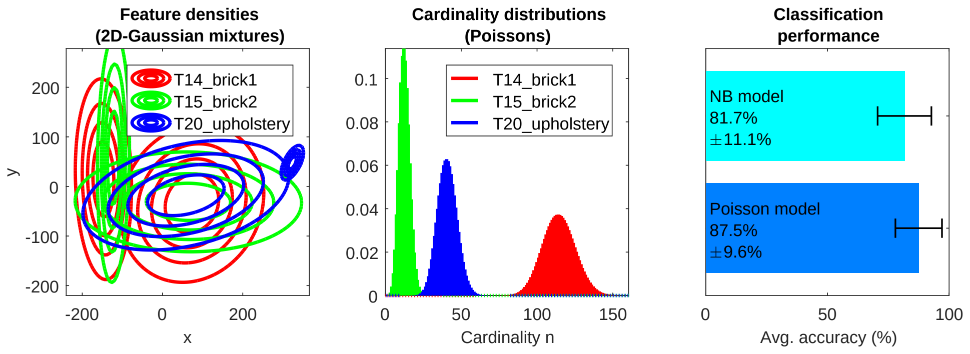
Figure 3.3: Extracted data from images of the Texture dataset. Left: 2-D features (after applying PCA to SIFT features). Right: Histogram of cardinalities of the extracted data.

In this experiment, we examine Poisson models with different feature densities, i.e., the Poisson model with single Gaussian feature density (Figure 3.4a), and the Poisson models with Gaussian mixture feature density (Figure 3.4b and Figure 3.4c). For all models, the 4-fold cross validation scheme is used. In the training phase, a training dataset consisting of 30 images per class is used to learn the model. In the test phase, test sets from the remaining images are used (10 images per class), and the average classification accuracies are reported.

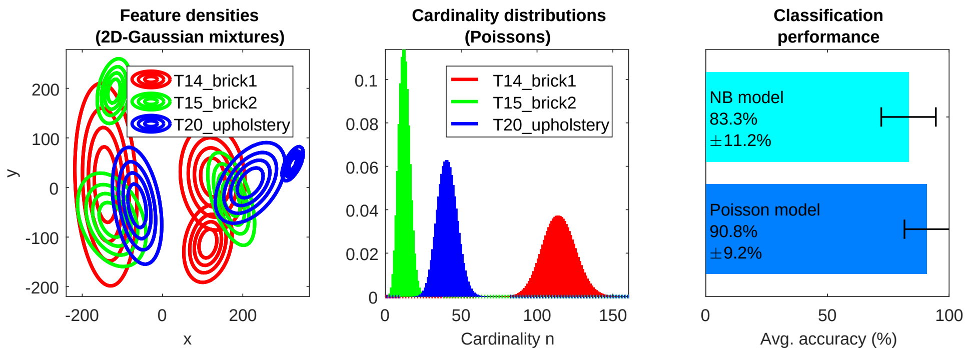
For the first model, the parameters of the Poisson model (with single Gaussian feature density) are learned using ML estimation (Section 3.3.5). The learned model is depicted in Figure 3.4a. Note that the feature density of the Poisson model is identical to that of the NB model. The average classification accuracies are also reported in Figure 3.4a. Observe that in this dataset, the features of PPs from different classes are not too overlapped, hence the NB can perform reas-



(a) Single Gaussian feature density



(b) Two-component Gaussian mixture feature density



(c) Three-component Gaussian mixture feature density

Figure 3.4: Learned distributions and classification performance on the Texture dataset. Left: feature distributions, Middle: cardinality distributions of Poisson models, Right: classification performance by NB and Poisson models (the error-bars indicate standard deviations of accuracies). Note that the feature densities of the Poisson model are indeed identical to those of the NB model.

onably well (with an average accuracy of 75%). The performance of the Poisson model is even better (86%), since in addition to feature information, the Poisson model can also exploit the cardinality information of the data, hence distinguishing the data even better.

For the Poisson models with Gaussian mixture feature density, the expectation-maximization (EM) algorithm is used to learn the parameters of the feature density (as discussed in Section 3.3.5). The learned models as well as the classification performance are depicted in Figure 3.4b (for 2-component Gaussian mixture) and Figure 3.4b (for 3-component Gaussian mixture). The feature densities of the Poisson models are also identical to those of the NB models. Observe that with these models, the performance of the NB and Poisson model are both improved (Figure 3.5) since the Gaussian mixture can better represent the features of PPs. The Poisson models again outperform the NB models for the same reason: the Poisson models can exploit both cardinality and feature information of the data, hence better distinguishing the data.

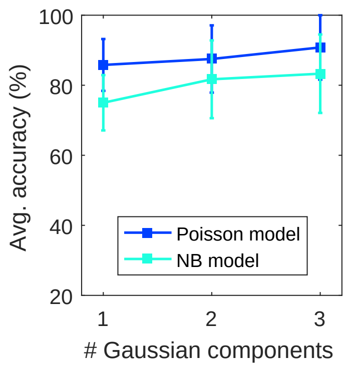


Figure 3.5: Classification performance on Texture dataset by NB and Poisson models with different number of Gaussian components in the feature density (the error-bars indicate standard deviations of accuracies).

It can be observed from Figure 3.5 that the performance of both NB and Poisson models increases with the number of Gaussian components, since the more components in the mixture, the better the data representation. To determine the optimal number of components, for low dimensional data, such as the compressed 2-D SIFT features in this experiment, we can simply visualize the data and manually decide the number of components. For high dimensional data, such as the 128-D SIFT features, techniques to determine the number of clusters in the data can be used, e.g., the cross-validation (Hill and Lewicki, 2006) or silhouette method (Rousseeuw, 1987) (see Section 5.4.1).

CHAPTER 4

MODEL-BASED NOVELTY DETECTION FOR POINT PATTERN DATA

Chapter's key points:

- Overview of PP novelty detection
- Issues of IID-cluster density in ranking PP data
- Novel ranking function for PP data
- Numerical experiments

In the previous chapter, we considered PP classification: a supervised learning problem with fully observed training data, i.e., all class labels are provided for learning. In this chapter, we work on novelty detection: a semi-supervised learning problem where only a part of the training data is labeled, more specifically, novelty detectors are trained with only normal data. After discovering undesirable properties of IID-cluster model density in ranking PP data, we improve it by proposing a **novel ranking function** for PP data based on this density. It is shown in numerical experiments that the proposed ranking function outperforms the naive Bayes likelihood as well as the IID-cluster model density in PP novelty detection.

4.1 Overview of PP novelty detection

As previously discussed, novelty detection is the semi-supervised task of identifying observations that are significantly different from the rest of the data (Markou and Singh, 2003; Pimentel et al., 2014). In novelty detection, there is no novel training data, i.e., *only normal data* is available for training. Hence it is not a special case of classification nor clustering (Chandola et al., 2009; Hodge and Austin, 2004), and is a fundamental problem in its own right.

Similar to classification, novelty detection involves a training phase and a detection phase. The training phase is the same as that for classification except that we are provided with only normal data, hence only the *normal class model* is learned. In the detection phase, test observations are *ranked* according to how well they fit the normal data model and those not well-fitted are deemed novel or anomalous (Chandola et al., 2009; Hodge and Austin, 2004). The preferred measure of goodness of fit is the **likelihoods** of the data.

Novelty detection is arguably the least attended PP learning task, since to the best of our knowledge, currently, there are *no PP novelty detection methods* in the literature. We argue that there are several reasons for this. First, PP novelty detection cannot simply extend from its counterpart for point data (see the example in Section 1.4). In addition, even though the tools for handling PP data exist (e.g., point process theory, set distances), they have not been well introduced to the machine learning community. Furthermore, novelty detection, even with point data, has received less attention compared to other learning tasks (such as classification and clustering), since the task of detecting outliers can also be done simply using the methods classification or clustering methods (Hodge and Austin, 2004; Chandola et al., 2009). However, undertaking novelty detection in a semi-supervised fashion, is still worth studying since there are cases where novel data cannot be observed sufficiently or not observed at all (e.g., malfunction data of a newly developed device, or suspicious usage of a newly deployed network service). In these cases, classification or clustering methods may be neither relevant nor well-performing for novelty detection task.

4.2 Issues of IID-cluster density in ranking PP data

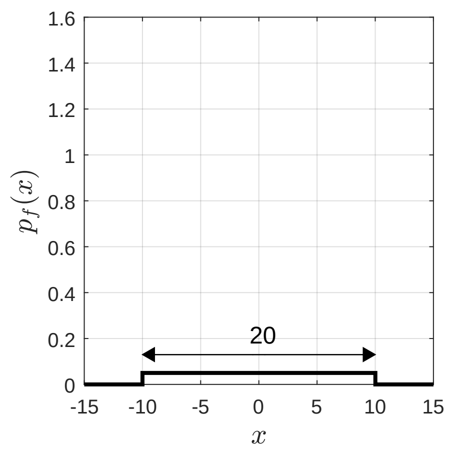
In Section 1.4, we showed that the NB likelihood of the constituent points cannot be a good ranking function for PP data. In particular, it suffers from incompatibility in the unit of measurement and does not appropriately account for the number of elements in PPs.

In this section, we point out that the IID-cluster point process density (used for PP classification in Chapter 3) is also not a proper likelihood for PP data in the novelty detection task, even though this density function is unit-less and incorporates both feature and cardinality information. In particular, we illustrate through concrete examples that the probability density of a PP does *not necessarily indicate how likely* the PP is. For this example, we reserve the term *likelihood* for the measure of how likely or probable an observation is.

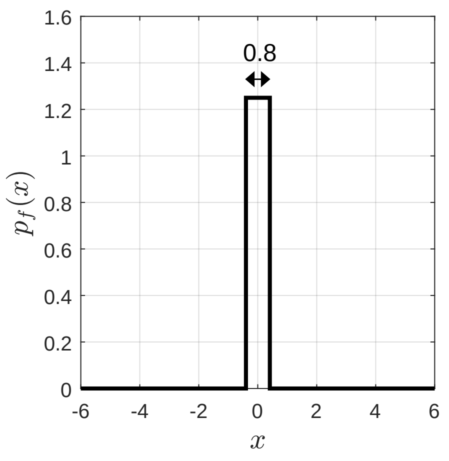
Consider two IID-cluster models with a common cardinality distribution but different uniform feature densities as shown in Figure 4.1. Since the feature density is uniform, PPs with the same cardinality are *equally likely*, and consequently their likelihoods should be *proportional to the cardinality distribution*.

If the probability density were an indication of how likely a PP is, then the plot of probability density against cardinality should resemble the cardinality distribution. However, this is not the case. Figure 4.2 indicates that for the IID-cluster with 'short' feature density, the probability density tends to decrease with increasing cardinality (Figure 4.2a). This phenomenon arises because the feature density given cardinality n is 0.05^n, which vanishes faster than the $n!$ growth (for $n \leq 20$). The converse is true for the IID-cluster with 'tall' feature density (Figure 4.2b). Thus, PPs with highest/least probability density are *not* necessarily the most/least probable.

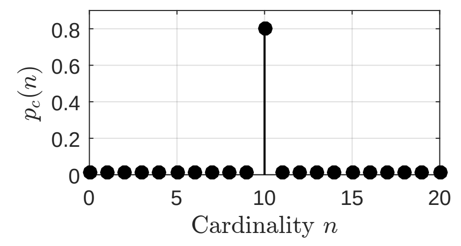
The same phenomenon also arises with Poisson models. Consider two Pois-



(a) ‘short’ uniform density

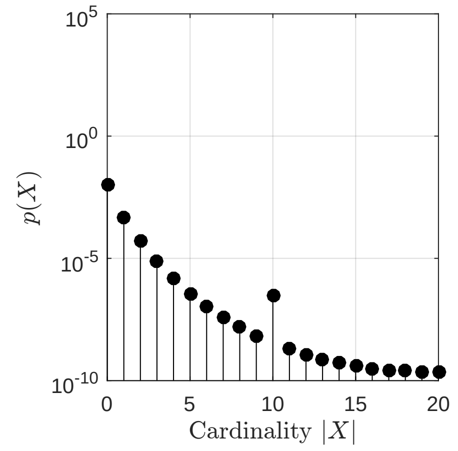


(b) ‘tall’ uniform density

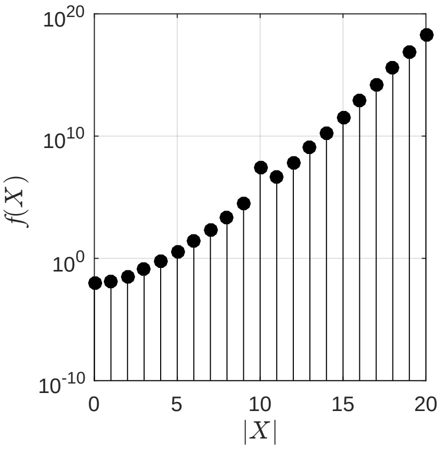


(c) Cardinality distribution on {0,...,20} with one mode at 10, the remaining cardinalities are equally likely with total mass 0.2

Figure 4.1: Feature and cardinality distributions of two IID-cluster point processes.

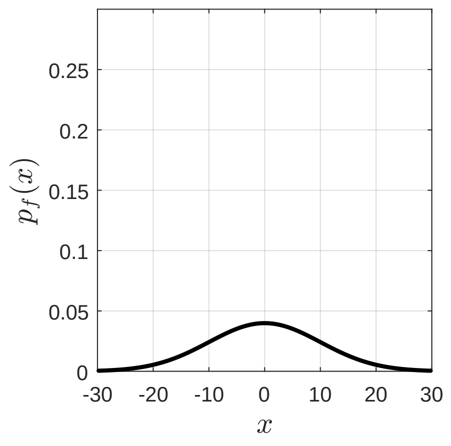


(a)

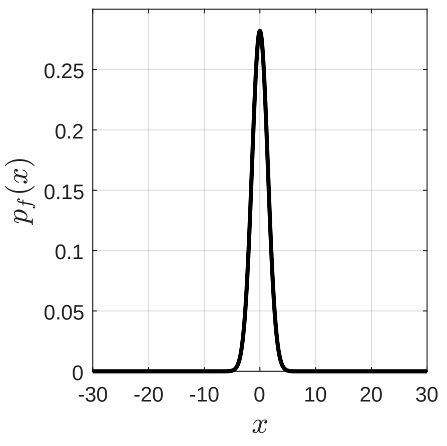


(b)

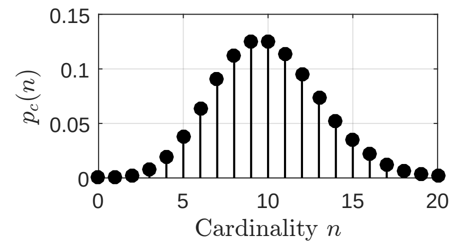
Figure 4.2: Probability density against cardinality: (a) ‘short’ feature density; (b) ‘tall’ feature density.



(a) 'short' Gaussian density with variance 100

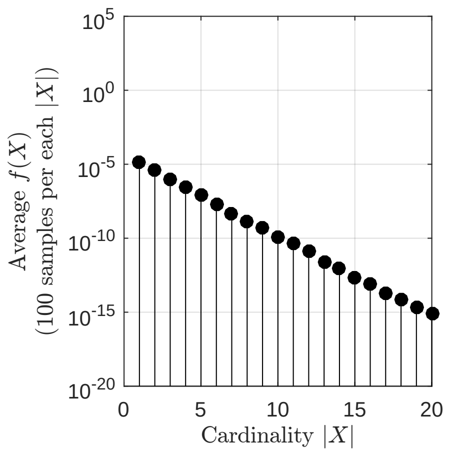


(b) 'tall' Gaussian density with variance 2

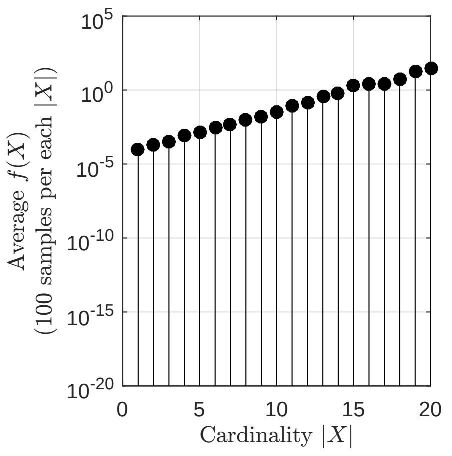


(c) Poisson distribution with rate 10

Figure 4.3: Feature and cardinality distributions of two Poisson models.



(a)



(b)

Figure 4.4: Average probability density against cardinality: (a) 'short' Gaussian feature density; (b) 'tall' Gaussian feature density. For each cardinality we sample 100 PPs and calculate the average density of these PPs.

son models with a common cardinality distribution but different Gaussian feature densities as shown in Figure 4.3. In this case, because the feature distributions (Gaussian) are not uniform, we calculate the average of the density for each cardinality (depicted in Figure 4.4). As previously discussed, we expect that the plot of probability density against cardinality should resemble the cardinality distribution. However, this again is not the case. For the Poisson model with 'short' Gaussian feature density, the probability density tends to decrease with increasing cardinality (Figure 4.4a) and the converse for the case of Poisson model with 'tall' Gaussian feature density (Figure 4.4b).

Reason of the aforementioned phenomenon

The aforementioned phenomenon arises from the *non-uniformity of the reference measure*. A measure μ is said to be *uniform* if for any measurable region A with $\mu(A) < \infty$, all points of A (except on set of measure zero) are *equi-probable* under the probability distribution $\mu/\mu(A)$.

One example is the **Lebesgue measure** vol on $\mathbb{R}^n$: given any bounded measurable region A, the probability distribution $vol(\cdot)/vol(A)$ is uniform on A, i.e., all realizations in A are equally likely. The probability density (as a Radon-Nikodym derivative) at a point X is the ratio of probability measure to reference measure at an infinitesimal neighbourhood of X, i.e.,

$$f(X) = \frac{P(dX)}{\mu(dX)}. \tag{4.1}$$

Hence, unless the reference measure is uniform, $f(X)$ is not a measure of how likely X is. This is also true even for *probability densities on the real line*. For example, the probability density of a zero-mean Gaussian distribution with standard variance 1 relative to the Lebesgue measure is the usual bell-shaped curve (Figure 4.5a), while its density relative to a zero-mean Gaussian distribution with variance 0.8 is a U-shaped curve where *the most probable point has the least probability density value* (Figure 4.5b).

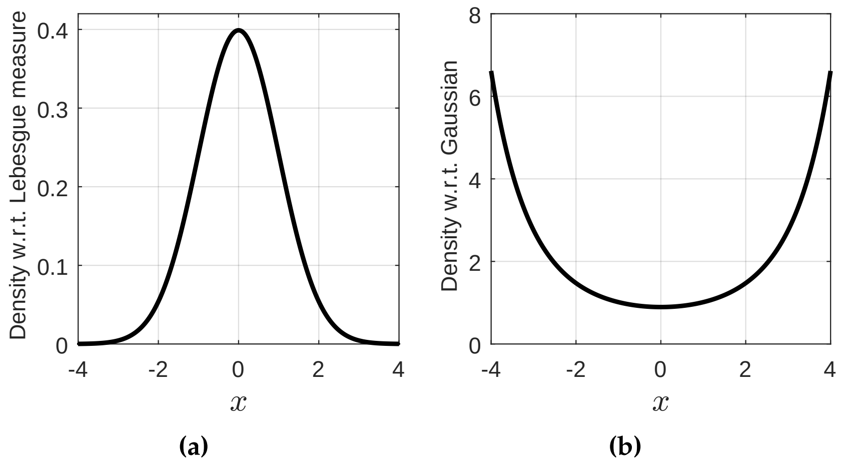


Figure 4.5: Density of a zero-mean unit-variance Gaussian w.r.t.: (a) Lebesgue measure; (b) zero-mean Gaussian with variance 0.8.

The reference measure μ for point processes defined by Eq. (2.11) is *not uniform* because for a bounded region $\mathcal{T} \subseteq \mathcal{F}(\mathcal{X})$, the probability distribution $\mu/\mu(\mathcal{T})$ is not necessarily uniform (unless all points of $\mathcal{T}$ have the same cardinality). Hence, probability densities of input PPs relative to μ are not indicative of how well they fit the normal data model.

Why is classification not affected by the aforementioned phenomenon?

In novelty detection we are interested in the likelihood of the input PP whereas in Bayesian classification (Chapter 3) we are interested in its likelihood ratio. The posterior class probability

$$p(y \mid X) = \frac{p(y)f(X \mid y)}{\int p(y)f(X \mid y)dy} \tag{4.2}$$

$$= \frac{p(y)P(dX \mid y)/\mu(dX)}{\int p(y)(P(dX \mid y)/\mu(dX))dy} \tag{4.3}$$

$$= \frac{p(y)P(dX \mid y)}{\int p(y)P(dX \mid y)dy} \tag{4.4}$$

is the ratio, at an infinitesimal neighbourhood dX, between the joint probability $P(dX, y)$, and the probability$P(dX)$, which is *invariant* to the choice of reference measure. In essence, the normalizing constant cancels out the influence of the reference measure, and hence, problems with the non-uniformity of the reference measure do not arise.

4.3 Ranking function for PP data

To the best of our knowledge, it is *not known* whether there exists a uniform reference measure on $\mathcal{F}(\mathcal{X})$ that dominates the point process probability distributions of interest (so that they admit densities). In this subsection, we propose a *suitable PP ranking function* for novelty detection by modifying the IID-cluster model density.

The IID-cluster probability density Eq. (2.13) is the product of the cardinality distribution $p_c(|X|)$, the cardinality-conditioned feature density $f_{|X|}(X)$, and a trans-dimensional weight $|X|!U^{|X|}$. Note that the cardinality distribution and the conditional joint feature density completely describes the point process. The conditional density $f_{|X|}(X)$ enables the ranking of PPs of the same cardinality, but cannot be used to rank across different cardinalities because it takes on *different units of measurement*. The weights $|X|!U^{|X|}$ reconcile for the differences in dimensionality and the unit of measurement between $f_{|X|}(X)$ of different cardinalities. However, the example in Section 4.2 demonstrates that weighting by $|X|!U^{|X|}$ leads to probability densities that are inconsistent with likelihoods.

In the generalization of the MAP estimator to PPs, Mahler (2007b) circumvented such inconsistency by replacing $|X|!U^{|X|}$ with $c^{|X|}$, where c is an arbitrary constant. More concisely, instead of maximizing the probability density $f(X)$, Mahler proposed to maximize $f(X)c^{|X|}/|X|!$. Since c is a free parameter, the generalized MAP estimate depends on the choice of c.

Inspired by Mahler's generalized MAP estimator, we replace the weight $|X|!U^{|X|}$ in the probability density by a general function of the cardinality $C(|X|)$, resulting in a ranking function of the form

$$r(X) = p_c(|X|)C(|X|)f_{|X|}(X). \tag{4.5}$$

The example in Section 4.2 demonstrates that, as a function of cardinality, the ranking should be *proportional to the cardinality distribution*, otherwise unlikely samples can assume high ranking values. In general, the ranking function is not solely dependent on the cardinality, but also varies with the features. Nonetheless, the example suggests that the ranking function, on average, should be proportional to the cardinality distribution. Hence, we impose the following consistency requirement: for a given cardinality n, the expected ranking value is proportional to the probability of cardinality n, i.e.,

$$\mathbb{E}_{X||X|=n}\left[r(X)\right] \propto p_c(n). \tag{4.6}$$

Propositional 3. *For a point process with probability density Eq. (2.13), a ranking function consistent with the cardinality distribution, i.e., satisfies Eq. (4.6), is given by*

$$r(X) \propto \frac{p_c(|X|)}{||f_{|X|}||_2^2} f_{|X|}(X) \tag{4.7}$$

where $||\cdot||_2$ denotes the L_2-norm.

Proof. Noting from Eq. (2.13) that $f(X \mid |X| = n) = n!U^n f_n(X)\delta_n[|X|]$, and using the integral Eq. (2.12) we have

$$\mathbb{E}_{X||X|=n}\left[f_n(X)\right] = \int f_n(X)\, f(X \mid |X| = n)\, \mu(dX) \tag{4.8}$$

$$= \frac{n!U^n}{n!U^n} \int (f_n(\{x_1,...,x_n\}))^2 d(x_1,...,x_n) \tag{4.9}$$

$$= ||f_n||_2^2. \tag{4.10}$$

Hence

$$\mathbb{E}_{X||X|=n}\left[r(X)\right] \propto \mathbb{E}_{X||X|=n}\left[\frac{p_c(|X|)}{||f_{|X|}||_2^2} f_{|X|}(X)\right] \tag{4.11}$$

$$= \frac{p_c(n)}{||f_n||_2^2}\mathbb{E}_{X||X|=n}\left[f_n(X)\right] \tag{4.12}$$

$$= p_c(n). \tag{4.13}$$

QED

Note that $||f_{|X|}||_2^2$ has units of $U^{-|X|}$, which is the same as the unit of $f(X)$, rendering the ranking function r unit-less, thereby avoiding the unit of measurement inconsistency described in Section 1.4.

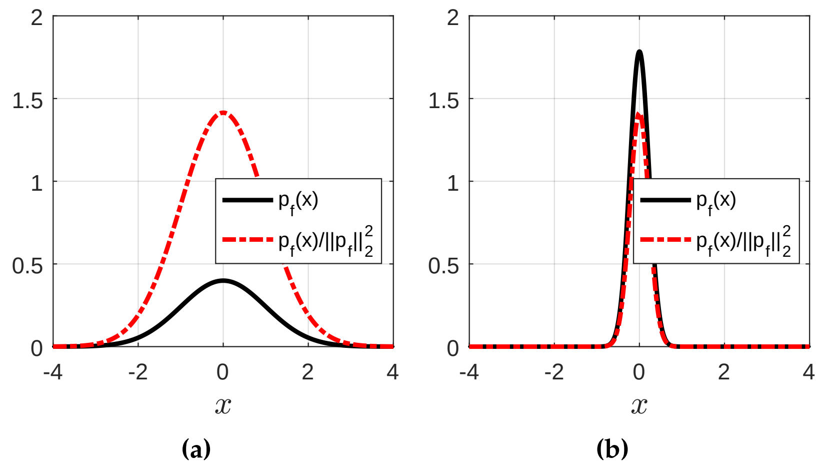


Figure 4.6: Probability density divided by energy: (a) 'short' Gaussian (mean = 0, variance = 1); (b) 'tall' Gaussian (mean = 0, variance = 0.05).

For an IID-cluster with feature density p_f, the ranking function reduces to

$$r(X) \propto p_c(|X|) \left(\frac{p_f}{||p_f||_2^2} \right)^X . \tag{4.14}$$

The feature density p_f, in the example of Section 4.2, is uniform and so $p_f/||p_f||_2^2 =$ 1 on its support. Hence, the ranking is equal to the cardinality distribution, as expected. Figure 4.6 illustrates the effect of dividing a non-uniform feature density (Gaussian) p_f, by its energy $||p_f||_2^2$: 'tall' densities become shorter and 'short' densities become taller[1], providing the right adjustment for multiplying together many large/small numbers.

Furthermore, Figure 4.7 and Figure 4.8 verify that the plot of the proposed ranking function against cardinality indeed resemble the cardinality distributions of the IID-cluster models (Figure 4.1c and Figure 4.3c).

[1] L_2-norm of Gaussian density function $||\mathcal{N}(x \mid \mu, \Sigma)||_2 = \sqrt{\mathcal{N}(\mu|\mu, 2\Sigma)}$ (Ho and Lee, 1964).

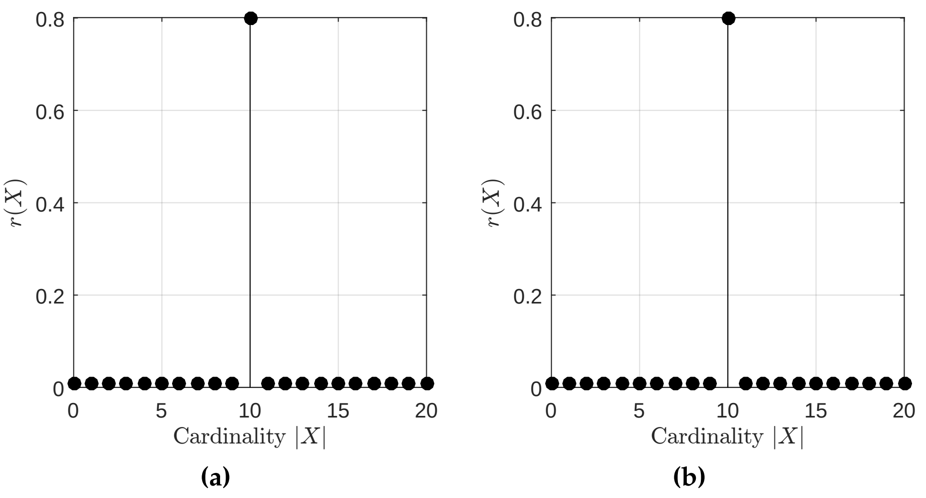


Figure 4.7: Ranking function against cardinality: (a) 'short' feature density; (b) 'tall' feature density.

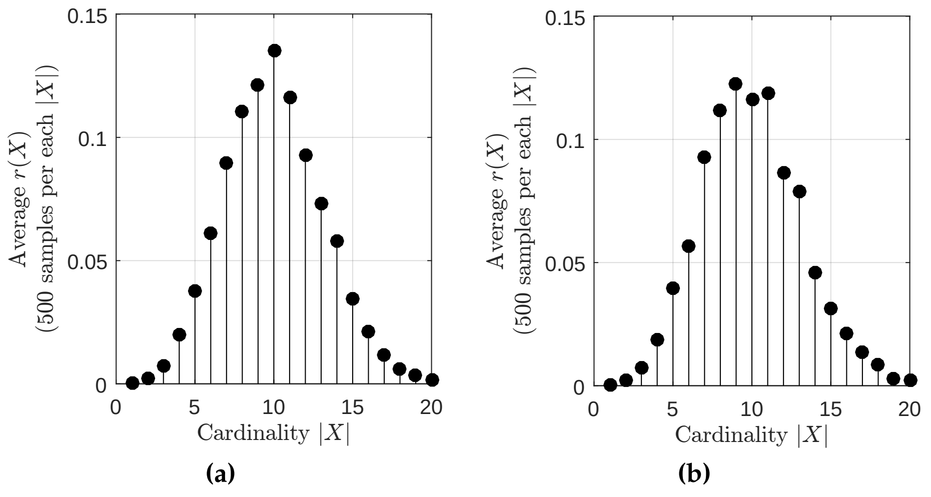


Figure 4.8: Average ranking function against cardinality: (a) 'short' Gaussian feature density; (b) 'tall' Gaussian feature density.

4.4 Numerical experiments

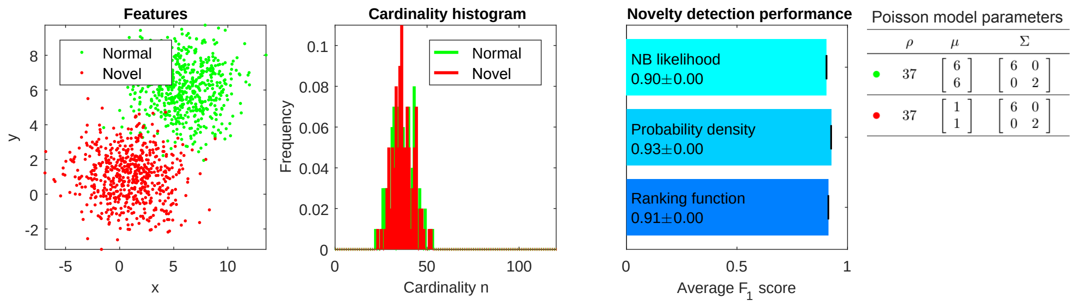
This section presents the novelty detection experiments on both simulated and real data using the *Poisson model*. The experiments illustrate the versatility of the framework and the effectiveness of the proposed ranking function relative to the NB likelihood and the standard Poisson model probability density. Like the classification experiments, ML method (Section 3.3.5) is used to learn the parameters of the NB and Poisson models in the training phase. For a fair comparison, in all experiments, the threshold is set the same as the 2$^{\text{nd}}$ 10-quantile of the ranking values of the *training (normal) data*.

4.4.1 Novelty detection with simulated data

In this experiment, we examine four scenarios of novel PP data using simulated datasets. Each dataset consists of two groups of data: normal and novel. Data from a group are generated from a Poisson model with a 2-D Gaussian intensity parameterized by (ρ, μ, Σ) as shown in Figure 4.9.

Normal data have cardinalities between 20 and 60 and are the same in all scenarios. For novel data, different types are considered: *scenario (i)* is an example of **feature novelty**, where novel observations are similar in cardinality to normal training data, but different in feature (Figure 4.9a); *scenario (ii)* is an example of **high-cardinality novelty**, where novel observations are similar in feature to normal training data, but have cardinalities ≥ 80 (Figure 4.9b); *scenario (iii)* is an example of **low-cardinality novelty**, where novel observations are similar in feature to normal training data, but have cardinalities ≤ 10 (Figure 4.9c); *scenario (iv)* is a **mix** of above novel types (Figure 4.9d).

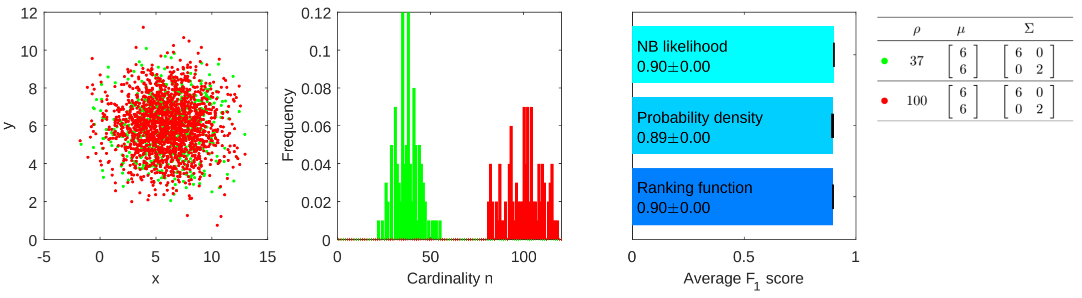
For training, we use the same 300 normal observations to learn the NB and Poisson model via ML estimation (Section 3.3.5). For testing, we use a test set containing 100 normal observations and 100 novelties of each type. In each scenario, the test is run 10 times with 10 different randomly sampled test sets and the averaged results are shown in Figure 4.9.



Poisson model parameters

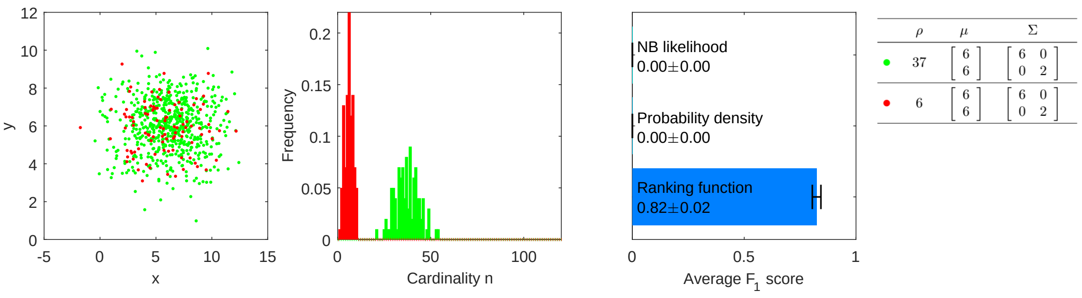
	ρ	μ	Σ
🟩	37	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	37	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$

(a) Scenario (i): feature novelty



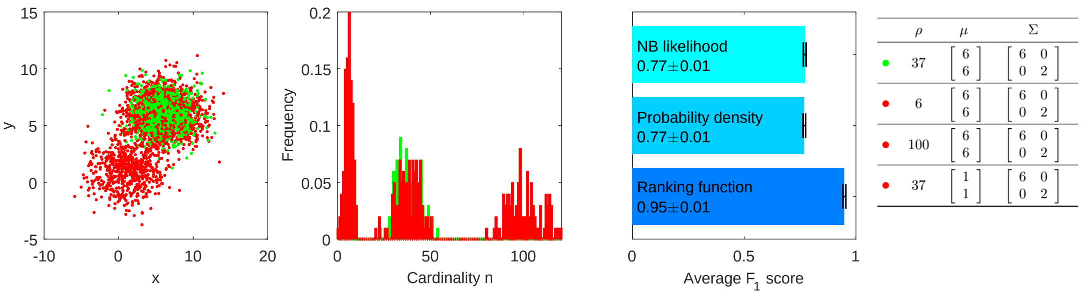
	ρ	μ	Σ
🟩	37	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	100	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$

(b) Scenario (ii): high-cardinality novelty



	ρ	μ	Σ
🟩	37	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	6	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$

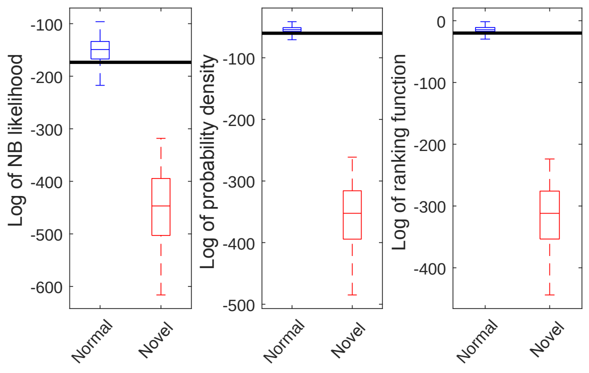
(c) Scenario (iii): low-cardinality novelty



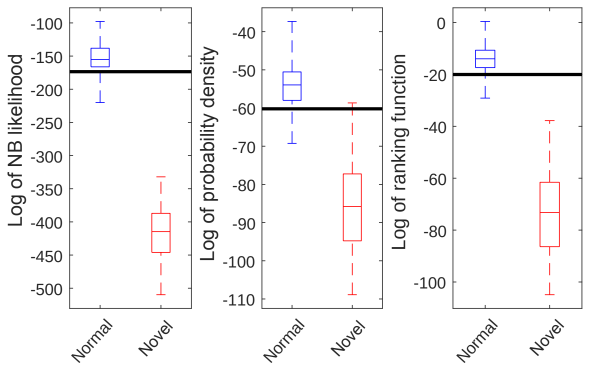
	ρ	μ	Σ
🟩	37	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	6	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	100	$\begin{bmatrix}6\\6\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$
🟥	37	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}6 & 0\\0 & 2\end{bmatrix}$

(d) Scenario (iv): a mix of novel types

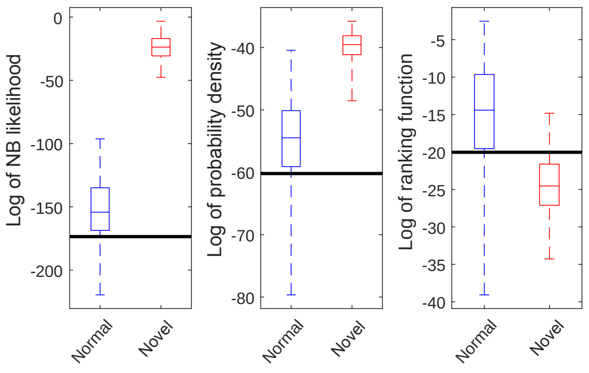
Figure 4.9: Simulated data and novelty detection performance in four diverse scenarios by NB likelihood, Poisson model probability density, and proposed ranking function. The error-bars (on the left subplots) represent standard deviations of the F_1 scores.



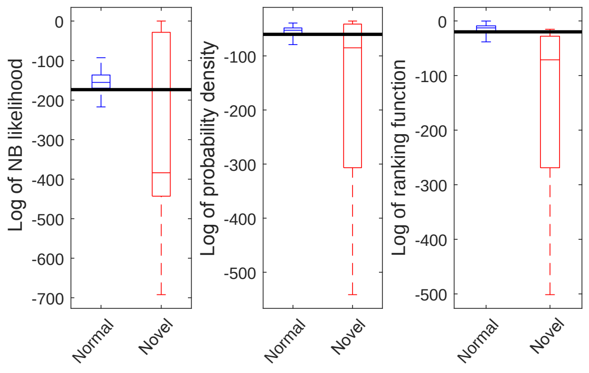
(a) Scenario (i): feature novelty



(b) Scenario (ii): high-cardinality novelty



(c) Scenario (iii): low-cardinality novelty



(d) Scenario (iv): a mix of novel types

Figure 4.10: Boxplots of log of NB likelihood, log of Poisson model probability density, and log of proposed ranking function for test data in one fold of the simulated dataset. Thick lines across the boxes indicate the chosen thresholds.

Observe that in *scenario (i)* where novelties are dissimilar to normal data in feature (Figure 4.9a), all NB likelihood, Poisson model probability density, and proposed ranking function perform well. In *scenario (ii)* where novelties have higher cardinalities compared to normal data (Figure 4.9b), all the likelihoods also perform well. However, note that the good performance of NB likelihood and Poisson model probability density in this case is by chance, since this scenario is in fact the *'short' feature density* case described in Section 4.2, where the probability density tends to decrease with increasing cardinality (see Figure 4.2a), hence the novelties (with higher cardinalities than normal data) have (by chance) smaller densities than the normal data (see Figure 4.10b)). *Scenario (iii)* (see Figure 4.9c), is also the *'short' feature density* case, however, in this case the novelties have less cardinalities compared to normal data, hence the Poisson model (as well as the NB model) gives larger densities to novelties than normal data (see Figure 4.10c), and leads to very poor performance (Figure 4.9c). On the other hand, the proposed ranking function performs well since it ranks data properly (see boxplots in Figure 4.10c). *Scenario (iv)* is a mix of aforementioned scenarios (see Figure 4.9d), hence the performance is the sum of the discussed scenarios.

4.4.2 Novelty detection with Texture images dataset

Using the Texture dataset from Section 3.4.2, we consider normal data taken from class "T14 brick1" and novel data taken from class "T20 upholstery". We use 4-fold cross validation: In each fold, the training data consist of 75% of images from the normal class (30 images), and the testing set consists of the remaining images from the normal class (10 images) and 25% of images from the novel class (10 images).

In this experiment, we examine Poisson models with different feature densities, i.e., the Poisson model with single Gaussian feature density, and the Poisson models with Gaussian mixture feature density.

For the first model, the parameters of the Poisson model with single Gaus-

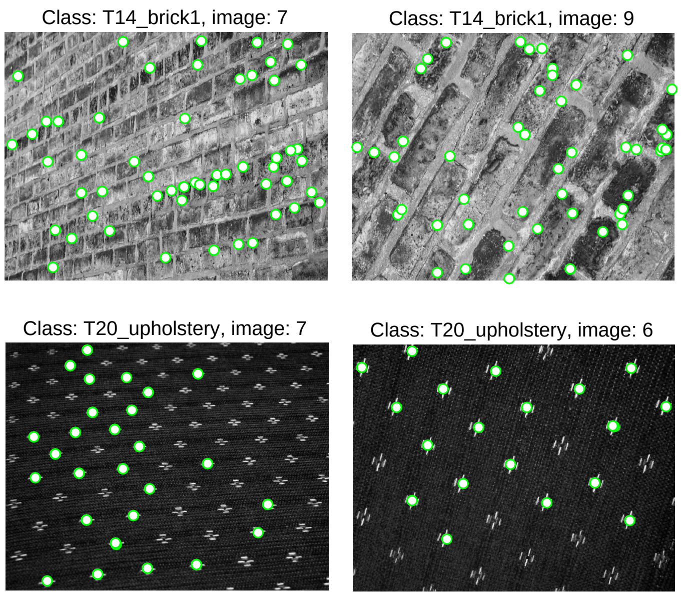

Class: T14_brick1, image: 7
Class: T14_brick1, image: 9
Class: T20_upholstery, image: 7
Class: T20_upholstery, image: 6

Figure 4.11: Example images from classes "T14_brick1", and "T20_upholstery" of the Texture dataset. Circles mark the detected SIFT keypoints. (This figure is identical to a part of Figure 3.2, but reproduced here for reading convenience.)

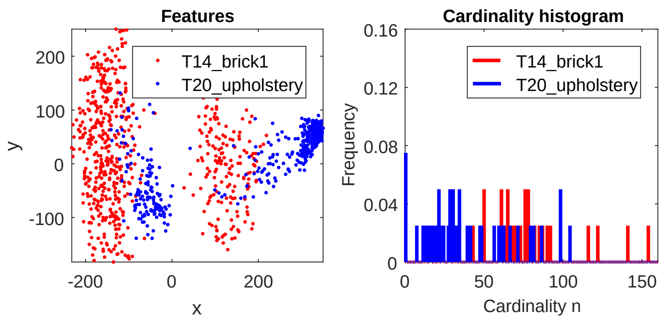
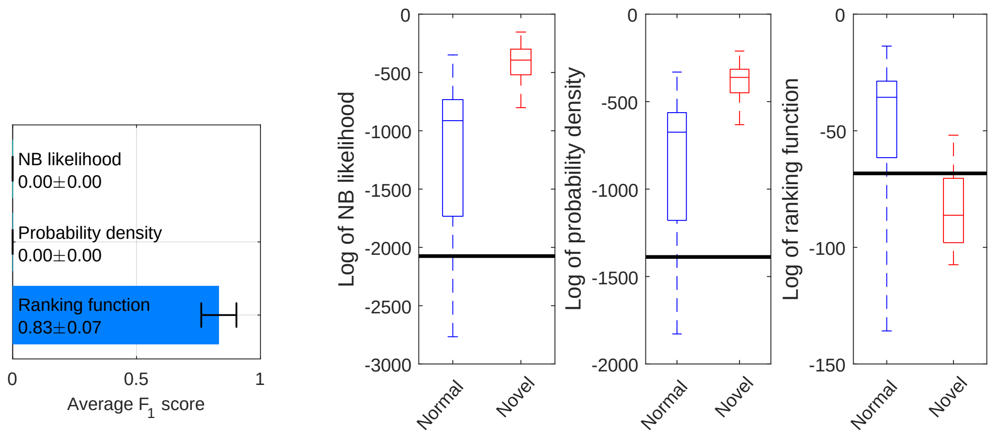
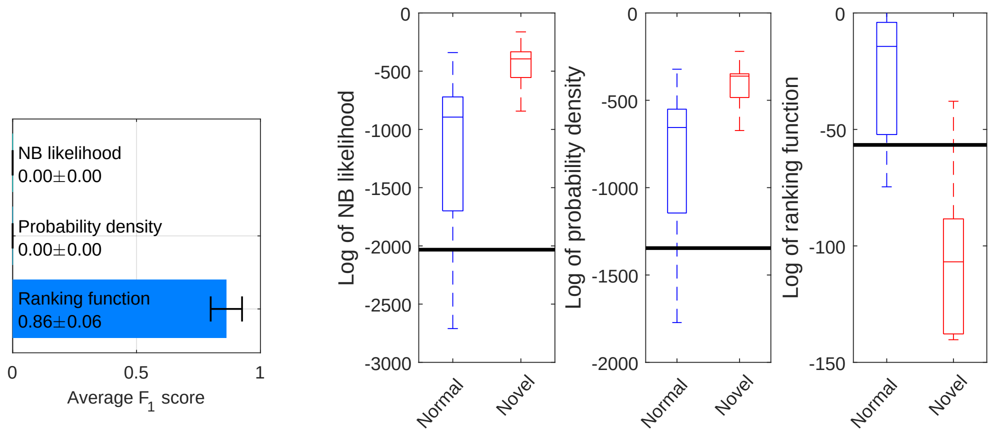


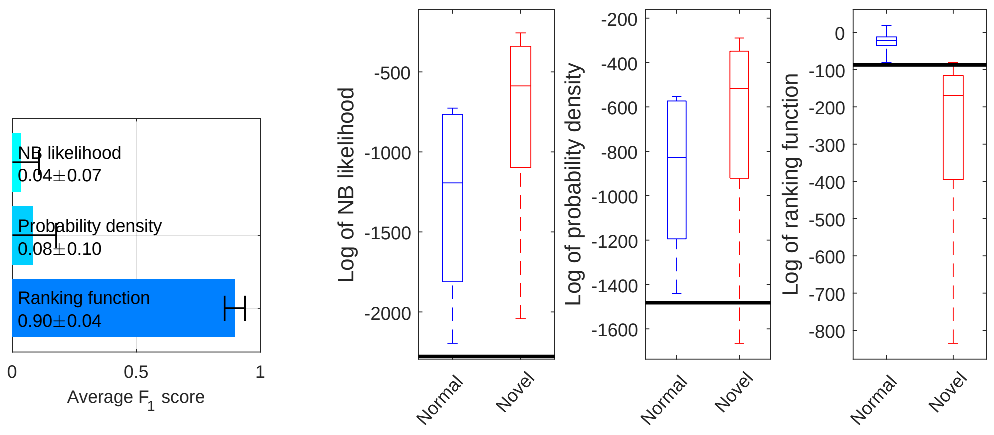
Figure 4.12: Extracted data from images of the Texture dataset. Left: 2-D features (after applying PCA to SIFT features). Right: Histogram of cardinalities of the extracted data. (This figure is identical to a part of Figure 3.3, but reproduced here for reading convenience.)



(a) Single Gaussian feature density



(b) Two-component Gaussian mixture feature density



(c) Three-component Gaussian mixture feature density

Figure 4.13: Left: Novelty detection performance on Texture data by 3 ranking functions: NB likelihood, Poisson model density, and the proposed ranking function (error-bars indicate standard deviations of F1-scores). Right: Boxplots of log of NB likelihood, log of Poisson model probability density, and log of proposed ranking function for test data in one fold of the Texture dataset (thick lines across the boxes indicate the chosen thresholds).

sian feature density are learned using ML estimation (see Section 3.3.5). Note that the learned models (for both NB and Poisson models) are indeed identical to that of class "T14 brick1" in Figure 3.4a. The average F_1 scores of novelty detection are reported in Figure 4.13a. It can be observed from the boxplots in Figure 4.13a that the NB likelihood and the probability density rank novel PPs even higher than normal data, hence they cannot detect any novelty (average F_1 scores of 0). On the other hand, the proposed ranking function provides a suitable ranking for PPs (see the boxplot in Figure 4.13a), hence achieving good performance (average F_1 scores of around 0.83). Note that this case is similar to *scenario (iii): low-cardinality novelty* in the simulated experiment in Section 4.4.1.

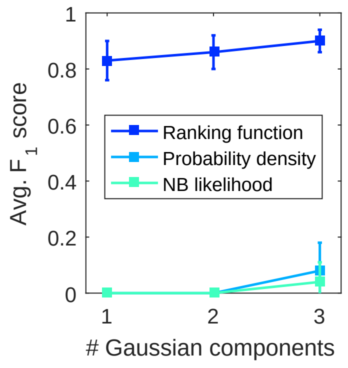


Figure 4.14: Novelty detection performance on Texture dataset by NB and Poisson models with different number of Gaussian components in the feature density (the error-bars indicate standard deviations of F_1 scores).

For the Poisson models with Gaussian mixture feature density, the EM algorithm is used to learn the parameters of the feature density (as discussed in Section 3.3.5). The learned models (for both NB and Poisson models) are similar to those of class "T14 brick1" in Figure 3.4b and Figure 3.4c. The average F_1 scores of novelty detection are reported in Figure 4.13b and Figure 4.13c. Compared to the first model, these models fit the data better, however, the NB likelihood and the probability density still fail in ranking the given PPs (see boxplots in Figure 4.13b and Figure 4.13c), hence still delivering very poor performance (average F_1 scores of $0 - 0.1$). On the other hand, the proposed ranking function provides a consistent ranking for PPs (see boxplots in Figure 4.13b and Figure 4.13c), hence achieving good performance (average F_1 scores of 0.86

– 0.90) and even better than that of the first model (average F_1 scores of 0.83) since these models better suit the data.

CHAPTER 5

MODEL-BASED CLUSTERING FOR POINT PATTERN DATA

Chapter's key points:

- Existing methods for PP clustering
- Mixture of IID-cluster point processes
- EM clustering with IID-cluster mixture
- Clustering with an unknown number of clusters
- Numerical experiments

In the previous chapters, the problems of learning PP data with fully observed labels (classification) and partly missing labels (novelty detection) were studied. In this chapter, we present PP clustering — a PP learning problem with complete label missing data. The proposed method uses the finite **mixture of IID-cluster point processes** as the generative model for PP clusters. Then, the **expectation-maximization (EM)** technique is deployed to learn the mixture model and cluster the PP data.

5.1 Existing methods for PP clustering

As previously discussed, in general, the aim of clustering is to partition the dataset into groups so that *members in a group are similar* to each other (Murphy, 2012). A clustering or partitioning of a given set of observations $\{X_1, ..., X_N\}$ is often represented by the (latent) cluster assignment $y_{1:N}$, where y_n denotes the cluster label for X_n. Clustering is an unsupervised learning problem since the class (or cluster) labels are not given (Jain et al., 1999; Russell and Norvig, 2003).

In PP learning, there exist two clustering algorithms: the Bag-level Multiple Instance Clustering (*BAMIC*) algorithm (Zhang and Zhou, 2009), and the Maximum Margin Multiple Instance Clustering (*M^3IC*) algorithm (Zhang et al., 2009). BAMIC adapts the k-medoids algorithm with the Hausdorff distance as a measure of dissimilarity between PPs (Zhang and Zhou, 2009).[1] On the other hand, in $\mathrm{M^3IC}$, the clustering is posed as a non-convex optimization problem which is then relaxed and solved via a combination of the Constrained Concave-Convex Procedure and Cutting Plane methods (Zhang et al., 2009). While these non-model-based algorithms are simple to use, they lack the ability to exploit statistical trends in the data, not to mention computational problems with high dimensional inputs and large datasets (Murphy, 2012).

5.2 Model-based PP clustering

In this section, we propose a model-based approach to the clustering problem for PP data. *Mixture modeling* is the most common probabilistic approach to clustering, where the aim is to estimate the cluster assignment $y_{1:N}$ via likelihood or posterior inference (Murphy, 2012). The point process formalism enables extension of mixture models to PP data, in particular, the finite **mixture of point processes**.

A finite *mixture model* assumes K underlying clusters labeled 1 to K, with prior probabilities $\pi_1, ..., \pi_K$, and is characterized by the parameters $\theta_1, ..., \theta_K$ in some space Θ. Let $f(X_n \,|\, \theta_k) \triangleq f(X_n \,|\, y_n = k, \theta_{1:K})$ denote the likelihood of X_n given that cluster k generates an observation. Then

$$f(X_{1:N}, y_{1:n} | \pi_{1:K}, \theta_{1:K}) = \prod_{n=1}^{N} \pi_{y_n} f(X_n | \theta_{z_n}), \tag{5.1}$$

Marginalizing the joint distribution Eq. (5.1) over the cluster assignment $y_{1:N}$

[1] The distance-based technique for PP clustering will be discussed more in Chapter 6.

gives the data likelihood function

$$f(X_{1:N}|\pi_{1:K},\theta_{1:K}) = \prod_{n=1}^{N}\sum_{k=1}^{K}\pi_k\, f(X_n|\theta_k). \tag{5.2}$$

Thus, in a finite mixture model, the likelihood of an observation is a *mixture of K probability densities*. The *posterior probability of cluster label $y_n = k$* (i.e., the probability that, given $\pi_{1:K},\theta_{1:K}$ and X_n, cluster k generates X_n) is

$$p\,(y_n = k|X_n,\pi_{1:K},\theta_{1:K}) = \frac{\pi_k\, f(X_n|\theta_k)}{\sum_{\ell=1}^{K}\pi_\ell\, f(X_n|\theta_\ell)}. \tag{5.3}$$

Under a mixture model formulation, clustering can be treated as an *incomplete data problem* since only the $X_{1:N}$ of the complete data $\{(X_n, y_n)\}_{n=1}^{N}$ is observed and the cluster assignment $y_{1:N}$ is unknown or missing. We seek $y_{1:N}$ and the *mixture model parameter* $\psi \triangleq (\pi_{1:K},\theta_{1:K})$ that best explain the observed data $X_{1:N}$ according to a given criterion such as the maximum likelihood (ML) method. ML is intractable in general and often requires the **expectation-maximization (EM)** technique (Dempster et al., 1977; Little and Rubin, 2002) to find approximate solutions.

5.3 EM clustering via IID-cluster mixture model

The EM technique maximizes the data likelihood Eq. (5.2) by generating a sequence of *iterates* $\{\psi^{(i)}\}_{i=0}^{\infty}$ using the following two main steps (Dempster et al., 1977; Little and Rubin, 2002):

- *E-step:* Compute the expectation $Q(\psi \mid \psi^{(i-1)})$ defined as

$$Q(\psi|\psi^{(i-1)}) \triangleq E_{y_{1:N}|X_{1:N},\psi^{(i-1)}}\left[\log f(X_{1:N}, y_{1:N}|\psi)\right]. \tag{5.4}$$

- *M-step:* Find $\psi^{(i)} = \underset{\psi}{\operatorname{argmax}}\, Q(\psi|\psi^{(i-1)})$.

It has been proven that the data likelihood $\ell(\psi^{(i)} \mid X_{1:N}) \triangleq f(X_{1:N}\,|\,\psi^{(i)})$ computed by Eq. (5.2) increases after each EM iteration and consequently converges to a local maximum (Dempster et al., 1977; Little and Rubin, 2002). In practice, the iteration is terminated at a user defined number N_{iter} or when increment in $\ell(\psi^{(i)} \mid X_{1:N})$ falls below a given threshold. The **optimal cluster label** estimate is the mode of the cluster label posterior Eq. (5.3).

In the E-step, following the arguments of the standard EM from (Bilmes, 1998) with the assumption that observations $X_{1:N}$ are independent and identically distributed (i.i.d.), the expectation $Q(\psi|\psi^{(i-1)})$ has the form:

$$Q(\psi|\psi^{(i-1)}) = \sum_{k=1}^{K}\sum_{n=1}^{N}\log(\pi_k f(X_n|\theta_k))p(y_n = k|X_n, \psi^{(i-1)}). \tag{5.5}$$

where $p(y_n = k|X_n, \psi^{(i-1)})$ is computed by Eq. (5.3).

Then, the M-step can be accomplished by *separately* maximizing $Q(\pi_{1:K}, \theta_{1:K}|\psi^{(i-1)})$ over $\theta_1, ..., \theta_K$ and $\pi_{1:K}$. Using the Lagrange multiplier with constraint $\sum_{k=1}^{K}\pi_k = 1$, yields the optimal component weights:

$$\pi_k^{(i)} = \frac{1}{N}\sum_{n=1}^{N} p(y_n = k|X_n, \psi^{(i-1)}). \tag{5.6}$$

The optimal component parameters θ_k depend on the specific form of $f(X_n|\theta_k)$, and is intractable in general.

Fortunately, close form solutions can be derived for special cases of the **IID-cluster point process mixture model**, i.e., a finite mixture point process model with

$$f(X|\theta_k) = p_{\xi_k}(|X|)\,|X|!\,U^{|X|}p_{\varphi_k}^{X}, \tag{5.7}$$

where $\theta_k = (\xi_k, \varphi_k)$ with ξ_k and φ_k denoting the parameters of the *cardinality distribution* and *feature distribution*, respectively. By substituting Eq. (5.7) into

Eq. (5.5), we obtain

$$\begin{aligned}Q(\psi|\psi^{(i-1)}) = &\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\log\left(\pi_k\right) p(y_n = k|X_n,\psi^{(i-1)})\right]\\ &+\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\log\left(|X_n|!\,U^{|X_n|}\right) p(y_n = k|X_n,\psi^{(i-1)})\right]\\ &+\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\log\left(p_{\xi_k}(|X_n|)\right) p(y_n = k|X_n,\psi^{(i-1)})\right]\\ &+\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\log\left(p_{\varphi_k}^{X_n}\right) p(y_n = k|X_n,\psi^{(i-1)})\right]\end{aligned} \tag{5.8}$$

i.e., $Q(\psi\,|\,\psi^{(i-1)})$ is *fully decomposed*, hence it can be maximized *separately* over the parameters of cardinality and feature distributions, i.e., $\xi_{1:K}$ and $\varphi_{1:K}$ respectively (similar to Proposition 1 in Section 3.3.5).

In addition, it can be observed from Eq. (5.8) that both $\log\left(p_{\xi_k}(|X_n|)\right)$ and $\log\left(p_{\varphi_k}^{X_n}\right)$ are accompanied by weight $p(y_n = k\,|\,X_n,\psi^{(i-1)})$, therefore, the values of the cardinality and feature distributions' parameters ξ_k and φ_k, which maximize $Q(\psi\,|\,\psi^{(i-1)})$, are equivalent to their *MLEs with data weighted by* $p(y_n = k\,|\,X_n,\psi^{(i-1)})$.

Using the aforementioned reasoning, we can derive the iteration update in M-step for parameters of several frequently used point processes, by incorporating the weight $p(y_n = k|X_n,\psi^{(i-1)})$ into their MLEs (shown in Section 3.3.5):

- For a **categorical cardinality distribution** with maximum cardinality M, where $\xi_k = (\xi_{k,0},...,\xi_{k,M})$ belongs to the unit M-simplex, the iteration update is

$$\xi_{k,m}^{(i)} = \frac{\sum_{n=1}^{N}\delta_m\left[|X_n|\right]p(y_n = k\,|\,X_n,\psi^{(i-1)})}{\sum_{\ell=0}^{M}\sum_{n=1}^{N}\delta_\ell[|X_n|]\,p(y_n = k\,|\,X_n,\psi^{(i-1)})}. \tag{5.9}$$

  where $0 \le m \le M$, $\delta_i[j]$ is the Kronecker delta. Note that a Laplace prior can be used to address over-fitting (see Section 3.3.5).
- For a **Poisson cardinality distribution**, where $\xi_k > 0$ is the mean cardinal-

ity, the iteration update is

$$\xi_k^{(i)} = \frac{\sum_{n=1}^{N} |X_n| p(y_n = k | X_n, \psi^{(i-1)})}{\sum_{n=1}^{N} p(y_n = k | X_n, \psi^{(i-1)})}. \tag{5.10}$$

- For a **categorical feature distribution** with maximum feature M, where $\varphi_k = (\varphi_{k,0}, ..., \varphi_{k,M})$ belongs to the unit M-simplex, the iteration update is

$$\varphi_{k,m}^{(i)} = \frac{\sum_{n=1}^{N} \sum_{x \in X_n} \delta_m [x] p(y_n = k | X_n, \psi^{(i-1)})}{\sum_{\ell=0}^{M} \sum_{n=1}^{N} \sum_{x \in X_n} \delta_\ell [x] p(y_n = k | X_n, \psi^{(i-1)})}. \tag{5.11}$$

  where $0 \le m \le M$, $\delta_i[j]$ is the Kronecker delta. Note that a Laplace prior can be used to address over-fitting (see Section 3.3.5).
- For a **Gaussian feature distribution**, where $\varphi_k = (\mu_k, \Sigma_k)$ is the mean and covariance pair, the iteration update is

$$\mu_k^{(i)} = \frac{\sum_{n=1}^{N} p(y_n = k | X_n, \psi^{(i-1)}) \sum_{x \in X_n} x}{\sum_{n=1}^{N} |X_n| p(y_n = k | X_n, \psi^{(i-1)})}, \tag{5.12}$$

$$\Sigma_k^{(i)} = \frac{\sum_{n=1}^{N} p(y_n = k | X_n, \psi^{(i-1)}) \sum_{x \in X_n} A_k^{(i)}(x)}{\sum_{n=1}^{N} |X_n| p(y_n = k | X_n, \psi^{(i-1)})}, \tag{5.13}$$

  where $A_k^{(i)}(x) = (x - \mu_k^{(i)})(x - \mu_k^{(i)})^{\mathrm{T}}$.
- For a **Gaussian mixture feature distribution**, where φ_k is the Gaussian mixture parameter, $\varphi_k^{(i)}$ can be determined by applying the standard EM algorithm on the weighted data (see e.g., (Gebru et al., 2016)) with the weights for $x \in X_n$ given by $p(y_n = k | X_n, \psi^{(i-1)})$ (computed by Eq. (5.3)).

To verify the formulas of the iteration updates in Eq. (5.9)-Eq. (5.13), in the following part, we take the derivatives of $Q(\psi | \psi^{(i-1)})$ with respect to the corresponding parameters and equate to zero.

Categorical cardinality distribution

A categorical cardinality distribution with maximum cardinality M parameterized by $\xi_k = (\xi_{k,0}, ..., \xi_{k,M})$ belongs to the unit M-simplex, has probability of the form:

$$p_{\xi_k}(n) = \prod_{m=0}^{M} \xi_{k,m}^{\delta_m[n]}, \tag{5.14}$$

where $\delta_i[j]$ is the Kronecker delta.

Finding ξ_k that maximizes $Q(\psi|\psi^{(i-1)})$ (Eq. (5.8)) with the constraint $\sum_{m=0}^{M} \xi_{k,m} = 1$ can be done using the Lagrange multiplier method. We first form the Lagrange function with the third term of $Q(\psi\,|\,\psi^{(i-1)})$ in Eq. (5.8) and the constraint:

$$\mathcal{L}(\xi_k, \lambda) = \sum_{k=1}^{K}\sum_{n=1}^{N} \log\left(p_{\xi_k}(|X_n|)\right) p(y_n = k|X_n, \psi^{(i-1)}) + \lambda\left(\sum_{m=0}^{M} \xi_{k,m} - 1\right), \tag{5.15}$$

where λ is the Lagrange multiplier.

By equating to zero the partial derivatives of Eq. (5.15) with respect to $\xi_{k,0}, \ldots, \xi_{k,M}$ and λ, we obtain the update of $\xi_{k,m}$ for ith iteration (the same as Eq. (5.9)):

$$\xi_{k,m}^{(i)} = \frac{\sum_{n=1}^{N} \delta_m\left[|X_n|\right] p(y_n = k\,|\,X_n, \psi^{(i-1)})}{\sum_{\ell=0}^{M}\sum_{n=1}^{N} \delta_\ell[|X_n|]\, p(y_n = k\,|\,X_n, \psi^{(i-1)})}. \tag{5.16}$$

where $0 \le m \le M$.

Poisson cardinality distribution

A Poisson cardinality distribution with the mean cardinality $\xi_k > 0$ has density of the form:

$$p_{\xi_k}(n) = \frac{1}{n!}\,(\xi_k)^n\, e^{-\xi_k}. \tag{5.17}$$

By equating to zero the derivative of $Q(\psi\,|\,\psi^{(i-1)})$ (Eq. (5.8)) with respect to ξ_k, we obtain the update of ξ_k for ith iteration (the same as Eq. (5.10)):

$$\xi_k^{(i)} = \frac{\sum_{n=1}^{N} |X_n|\, p(y_n = k\,|\,X_n, \psi^{(i-1)})}{\sum_{n=1}^{N} p(y_n = k\,|\,X_n, \psi^{(i-1)})}. \tag{5.18}$$

Categorical feature distribution

A categorical feature distribution with maximum feature M parameterized by $\varphi_k = (\varphi_{k,0}, \ldots, \varphi_{k,M})$ belongs to the unit M-simplex, has probability of the form:

$$p_{\varphi_k}(x) = \prod_{m=0}^{M} \varphi_{k,m}^{\delta_m[x]}, \tag{5.19}$$

where $\delta_i[j]$ is the Kronecker delta.

Finding φ_k that maximizes $Q(\psi|\psi^{(i-1)})$ (Eq. (5.8)) with the constraint $\sum_{m=0}^{M} \varphi_{k,m} = 1$ can be done using the Lagrange multiplier method. We first form the Lagrange function with the last term of $Q(\psi \mid \psi^{(i-1)})$ in Eq. (5.8) and the constraint:

$$\mathcal{L}(\varphi_k, \lambda) = \sum_{k=1}^{K} \sum_{n=1}^{N} \log\left(p_{\varphi_k}^{X_n}\right) p(y_n = k | X_n, \psi^{(i-1)}) + \lambda\left(\sum_{m=0}^{M} \varphi_{k,m} - 1\right), \tag{5.20}$$

where λ is the Lagrange multiplier.

By equating to zero the partial derivatives of Eq. (5.20) with respect to $\varphi_{k,0}, \ldots, \varphi_{k,M}$ and λ, we obtain the update of $\varphi_{k,m}$ for i^{th} iteration (the same as Eq. (5.11)):

$$\varphi_{k,m}^{(i)} = \frac{\sum_{n=1}^{N} \sum_{x \in X_n} \delta_m[x] p(y_n = k | X_n, \psi^{(i-1)})}{\sum_{\ell=0}^{M} \sum_{n=1}^{N} \sum_{x \in X_n} \delta_\ell[x] p(y_n = k | X_n, \psi^{(i-1)})}. \tag{5.21}$$

where $0 \le m \le M$.

Gaussian feature distribution

A D-dimensional Gaussian feature distribution with the mean and covariance $\varphi_k = (\mu_k, \Sigma_k)$, where Σ is positive-definite, has density of the form:

$$p_{\varphi_k}(x) = (2\pi)^{-\frac{D}{2}} |\Sigma_k|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_k)^{\mathrm{T}} \Sigma_k^{-1}(x-\mu_k)}, \tag{5.22}$$

By equating to zero the derivatives[2] of $Q(\psi \mid \psi^{(i-1)})$ (Eq. (5.8)) with respect

[2] May refer to (Petersen et al., 2012) for manipulation on matrices.

to μ_k and Σ_k, we obtain the update of μ_k and Σ_k for i^{th} iteration (the same as Eq. (5.12) and Eq. (5.13)):

$$\mu_k^{(i)} = \frac{\sum_{n=1}^{N} p(y_n = k | X_n, \psi^{(i-1)}) \sum_{x \in X_n} x}{\sum_{n=1}^{N} |X_n| p(y_n = k | X_n, \psi^{(i-1)})}, \tag{5.23}$$

$$\Sigma_k^{(i)} = \frac{\sum_{n=1}^{N} p(y_n = k | X_n, \psi^{(i-1)}) \sum_{x \in X_n} A_k^{(i)}(x)}{\sum_{n=1}^{N} |X_n| p(y_n = k | X_n, \psi^{(i-1)})}, \tag{5.24}$$

where $A_k^{(i)}(x) = (x - \mu_k^{(i)})(x - \mu_k^{(i)})^{\mathrm{T}}$.

To conclude, the complete EM clustering for PP data using the IID-cluster point process mixture model is given in Algorithm 1.

Algorithm 1: Clustering for PP data using IID-cluster point process mixture model.

```
Input: PP dataset $\{X_1, \dots, X_N\}$,
       number of clusters $K$,
       number of iterations $N_{\text{iter}}$ .
Output: cluster assignment $y_1, \dots, y_N$.
initialize $\psi^{(0)} = \left\{ \left( \pi_k^{(0)}, \xi_k^{(0)}, \varphi_k^{(0)} \right)_{k=1}^{K} \right\}$;
for $i = 1$ to $N_{\text{iter}}$ do
    for $k = 1$ to $K$ do
        for $n = 1$ to $N$ do
            compute posterior $p(y_n = k \mid X_n, \psi^{(i-1)})$ using Eq. (5.3);
        end
        estimate component weight $\pi_k^{(i)}$ using Eq. (5.6);
        estimate cardinality distribution parameters $\xi_k^{(i)}$ (Section 5.3);
        estimate feature distribution parameters $\varphi_k^{(i)}$ (Section 5.3);
    end
end
for $n = 1$ to $N$ do
    return cluster assignment $y_n = \operatorname{argmax}_{k \in \{1,\dots,K\}} p(y_n = k \mid X_n, \psi^{(N_{\text{iter}})})$;
end
```

5.4 Clustering with an unknown number of clusters

The proposed method in the previous section is a solution for the PP clustering problem with a known number of clusters. In the case of an unknown number of clusters, the two following approaches can be applied.[3] One approach makes use of general techniques for determining the number of clusters in the dataset, such as *cross valuation* and *silhouette* methods. The other approach poses the problem of determining the number of clusters as the *model selection* problem in the Bayesian framework, which can be solved using, e.g., MAP estimation methods.

5.4.1 Determining the number of clusters

Determining the number of clusters K in a dataset is an important problem, not only because most clustering algorithms require K as input (Kaufman and Rousseeuw, 1990), but also because the quality of partitioning can be significantly affected by this input parameter (Han et al., 2012). Determining the correct K, however, is one of the most challenging aspects of clustering (Jain, 2010) since the right number of clusters in a dataset is usually ambiguous (Han et al., 2012). In this section, we briefly introduce some frequently used methods for determining the number of clusters K, which can be used together with the proposed PP clustering to address the clustering problem with an unknown number of clusters.

Cross validation is usually used in the evaluation process *for classification* (as used in the experiments in Chapter 3), however, it can also be used to determine the number of clusters in a dataset (Han et al., 2012). In (model-based) classification, cross validation is deployed as follows: a given dataset is partitioned into m parts; $m - 1$ parts are used to learn the data models, and the remaining part (called the test part) is used to test the learned models (with the classification task). This learn and test process is repeated m times (with a different test part each time), and the average classification performance is reported.

[3] Techniques discussed here assist the proposed clustering method in this chapter to deal with the unknown number of clusters scenarios. In Chapter 6, we propose another clustering method for PP data, based on set distances, which can automatically infer the number of clusters.

We can apply this cross validation technique in determining the number of clusters *for (model-based) clustering* (Hill and Lewicki, 2006), such as the proposed PP clustering method. In particular, with a certain number of clusters K, the clustering algorithm is run on $m - 1$ parts of the dataset to learn the underlying models of the data. Then, we compute the likelihoods of the observations in the test part given their clusters, i.e., $f(X_n \mid \theta_k)$ (see Section 5.2) (Hill and Lewicki, 2006). This process is repeated m times (with a different test part each time) and the average likelihood ℓ_{avg} is obtained.

There can be a trend that a larger number of clusters K yields a larger average likelihood ℓ_{avg}, since increasing K can reduce the size of the clusters, therefore allowing the clusters' distributions to better represent observations belonging to the clusters. However, this trend may be mitigated after the right number of clusters is reached (as illustrated in Figure 5.1), because splitting a well-represented cluster only gives a minor improvement in data representation. Hence, one can choose the number of clusters at the turning point in the curve of average likelihood ℓ_{avg} with respect to the number of clusters (Figure 5.1).

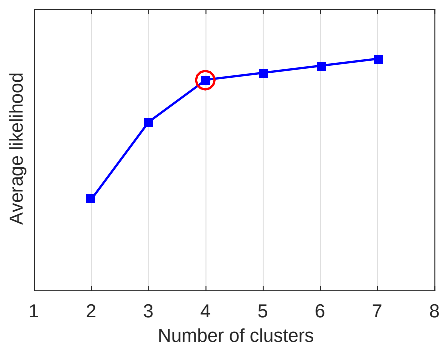


Figure 5.1: The average likelihood of observations with respect to the number of clusters. The circle marks the turning point, at which the number of clusters can be chosen.

Another method for determining the number of clusters in the dataset is **silhouette** (first described by Rousseeuw (1987)). In this method, the data are first partitioned into a certain number of clusters K, using any technique, such as the proposed PP clustering method. Then, for each datum X_n, two quantities are computed: $a(n)$ is the average dissimilarity of X_n to other observations in X_n's cluster, and $b(n)$ is the smallest average dissimilarity of X_n to clusters other than X_n's cluster, where the average dissimilarity of datum X_n to cluster C is defined as the average dissimilarity of X_n to observations in C.

In the case of PPs, dissimilarity can be measured by a set distance (such as the Hausdorff, Wasserstein, and OSPA distances described in Section 2.5). More specifically, let A be the cluster to which X_n belongs, then $a(n)$ can be computed as:

$$a(n) = \frac{1}{|A| - 1} \sum_{X \in A} d(X, X_n), \tag{5.25}$$

where d denotes a set distance. To compute $b(n)$, we first compute $c(n, C)$: the average dissimilarity of X_n to observations in a cluster $C \neq A$

$$c(n, C) = \frac{1}{|C|} \sum_{X \in C} d(X, X_n), \tag{5.26}$$

then $b(n)$ is obtained as follows

$$b(n) = \min_{C \neq A} c(n, C). \tag{5.27}$$

Note that $a(n)$ can be interpreted as how fit X_n is in its assigned cluster, in the sense that the smaller $a(n)$ is, the closer X_n is to other observations in its assigned cluster. On the other hand, the cluster with the smallest average dissimilarity $b(n)$ can be considered as the next best fit cluster for X_n (Rousseeuw, 1987).

Given $a(n)$ and $b(n)$, the *silhouette* $s(n)$ for X_n is defined as:

$$s(n) \triangleq \frac{b(n) - a(n)}{\max\{a(n), b(n)\}}, \tag{5.28}$$

Observe from its definition that $-1 \leq s(n) \leq 1$. $s(n)$ approximates 1 if $a(n) \ll b(n)$ which implies that X_n is *appropriately assigned* to cluster A, since a smaller $a(n)$ indicates a more fitting X_n in A, and a larger $b(n)$ indicates a less likely

X_n is assigned to other clusters. In contrast, $s(n)$ close to -1 implies that X_n is *misclassified* to A by the same reasoning (Rousseeuw, 1987).

Hence, the average silhouette $s(n)$ over all observations in the dataset can be used as an indicator of how appropriately the data have been partitioned into K clusters. Then, one way to choose the number of clusters K is to select K for which the average silhouette is largest (Rousseeuw, 1987). One can also re-scale the data to increase the likelihood of the silhouette being maximized at the right number of clusters using the methods proposed by de Amorim and Hennig (2015).

For other methods to determine the number of clusters in the dataset, refer to materials such as (Kaufman and Rousseeuw, 1990; Witten and Frank, 2005; Han et al., 2012; Murphy, 2012).

5.4.2 Bayesian approach

In a model-based clustering approach where data are represented by a mixture model, such as the proposed PP clustering in Section 5.2, the number of clusters K is identical to the number of mixture components. Under the Bayesian framework, determining the optimal value for K can be treated as a *model selection* problem (Murphy, 2012). Such a value for K can be computed as the *MAP estimate* of the posterior

$$p(K \mid X_{1:N}) \propto p(K)\, p(X_{1:N} \mid K), \tag{5.29}$$

where $p(K)$ is the *prior distribution* representing our knowledge/assumption on K before seeing any data. In the case that an uniform prior is used (e.g., when no assumption on K is made), the MAP estimate is equivalent to the maximum of the *(marginal) likelihood* $p(X_{1:N} \mid K)$ (Murphy, 2012).

In general, evaluating the likelihood $p(X_{1:N} \mid K)$ is rather difficult, hence, in practice, some approximations, such as *Bayesian information criterion* (Schwarz et al., 1978; Fraley and Raftery, 2007), are frequently used. Alternatively, stochastic sampling methods, such as *reversible jump MCMC* (Green, 1995), can be applied. However, this method is computationally expensive and difficult to implement (Murphy, 2012).

A more promising approach is to perform *Gibbs sampling on the Dirichlet process mixture model*, which can be simpler in implementation while still allowing an infinite number of mixture components (Murphy, 2012). A main requirement for Gibbs samplers is that the conditional probabilities in the sampling iterations have to be easily computed and sampled, hence the data model should be designed so that such conditionals can be obtained. This can be considered as a future research topic for PP clustering.[4]

Remark: The Bayesian solution for PP clustering (if proposed) can also be adapted for *semi-supervised learning*, where only labeled training data for certain clusters are available and the objective is to compute the posterior of the missing labels. This approach can also address the novelty detection problem (Chapter 4) without having to rank the input observations, albeit at greater computational cost. We can consider these as future research topics for PP learning.

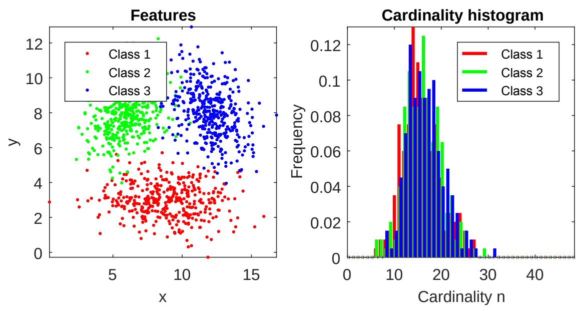
5.5 Numerical experiments

In this section, we evaluate the proposed EM clustering with both simulated and real data. Our experiments assume *mixture of Poisson models* (for the first two experiments) and *mixture of IID-cluster models* (for the last experiment). The relevant performance indicators are: purity, normalized mutual information (NMI), Rand index, and F_1 score (see Section 2.3.5).[5]

5.5.1 Clustering with simulated data

In this experiment, we use the same simulated dataset described in Section 3.4.1 (but without data labels). Each dataset is represented by a mixture of three

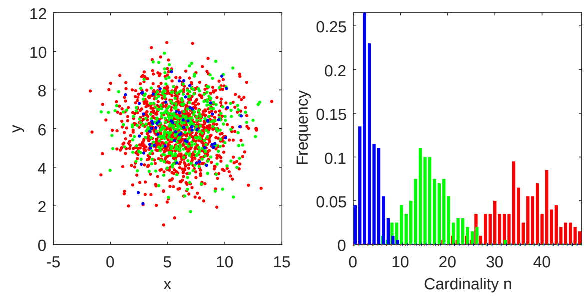
[4] May see some preliminary result in (Vo et al., 2017a).
[5] A comparison and discussion of the performance of the proposed model-based method versus the distance-based methods (including BAMIC) will be given in the next chapter.



Poisson model parameters

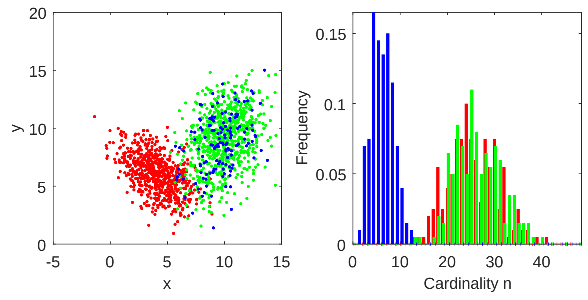
	ρ	μ	Σ
🔴	16	$\begin{bmatrix} 9 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 1 \end{bmatrix}$
🟢	16	$\begin{bmatrix} 6 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 2 & 0.6 \\ 0.6 & 2 \end{bmatrix}$
🔵	16	$\begin{bmatrix} 12 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 2 & -0.6 \\ -0.6 & 2 \end{bmatrix}$

(a) Dataset (i)



	ρ	μ	Σ
🔴	37	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$
🟢	16	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$
🔵	3	$\begin{bmatrix} 6 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$

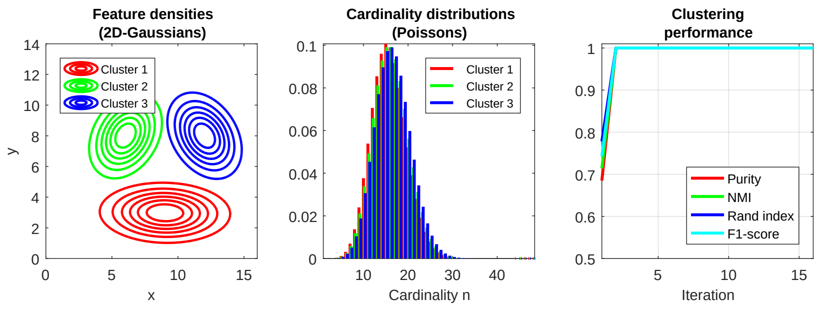
(b) Dataset (ii)



	ρ	μ	Σ
🔴	26	$\begin{bmatrix} 4 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 3 & -1.6 \\ -1.6 & 3 \end{bmatrix}$
🟢	26	$\begin{bmatrix} 10 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 3 & 1.6 \\ 1.6 & 6 \end{bmatrix}$
🔵	6	$\begin{bmatrix} 10 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 3 & 1.6 \\ 1.6 & 6 \end{bmatrix}$

(c) Dataset (iii)

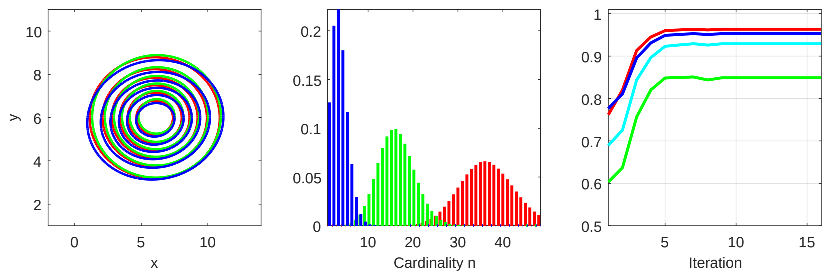
Figure 5.2: Simulated data of three diverse scenarios of PP data. Dataset (i): well-separated in feature but overlapping in cardinality; dataset (ii): well-separated in cardinality but overlapping in feature; dataset (iii): a mix of (i) and (ii). (This figure is identical to a part of Figure 3.1, but reproduced here for reading convenience.)



Estimated parameters

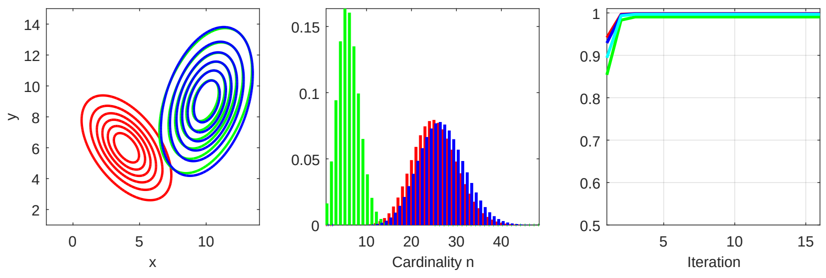
	π	ρ	μ	Σ
●	0.33	15.7	$\begin{bmatrix} 9.0 \\ 3.0 \end{bmatrix}$	$\begin{bmatrix} 6.3 & -0.1 \\ -0.1 & 1.0 \end{bmatrix}$
●	0.33	16.0	$\begin{bmatrix} 6.1 \\ 8.0 \end{bmatrix}$	$\begin{bmatrix} 2.0 & 0.6 \\ 0.6 & 2.0 \end{bmatrix}$
●	0.33	16.3	$\begin{bmatrix} 12.0 \\ 8.0 \end{bmatrix}$	$\begin{bmatrix} 2.0 & -0.7 \\ -0.7 & 2.0 \end{bmatrix}$

(a) Dataset (i)



	π	ρ	μ	Σ
●	0.33	36.5	$\begin{bmatrix} 6.0 \\ 6.0 \end{bmatrix}$	$\begin{bmatrix} 6.1 & 0.0 \\ 0.0 & 2.0 \end{bmatrix}$
●	0.33	16.1	$\begin{bmatrix} 6.1 \\ 6.1 \end{bmatrix}$	$\begin{bmatrix} 6.0 & 0.0 \\ 0.0 & 2.1 \end{bmatrix}$
●	0.33	3.2	$\begin{bmatrix} 6.1 \\ 5.9 \end{bmatrix}$	$\begin{bmatrix} 6.7 & 0.2 \\ 0.2 & 2.0 \end{bmatrix}$

(b) Dataset (ii)



	π	ρ	μ	Σ
●	0.33	25.3	$\begin{bmatrix} 4.0 \\ 6.0 \end{bmatrix}$	$\begin{bmatrix} 2.9 & -1.5 \\ -1.5 & 3.0 \end{bmatrix}$
●	0.33	5.9	$\begin{bmatrix} 10.0 \\ 9.0 \end{bmatrix}$	$\begin{bmatrix} 3.1 & 1.8 \\ 1.8 & 5.7 \end{bmatrix}$
●	0.33	26.3	$\begin{bmatrix} 10.1 \\ 9.0 \end{bmatrix}$	$\begin{bmatrix} 3.1 & 1.6 \\ 1.6 & 6.0 \end{bmatrix}$

(c) Dataset (iii)

Figure 5.3: The learned Poisson model distributions and EM clustering performance for three diverse scenarios of PP data. Dataset (i): well-separated in feature but overlapping in cardinality; dataset (ii): well-separated in cardinality but overlapping in feature; dataset (iii): a mix of (i) and (ii).

Poisson models with 2-D Gaussian feature distributions. These mixtures are learned by the EM algorithm with the number of clusters $K = 3$ (Section 5.3).

The learned distributions and the clustering performance is shown in Figure 5.3. The proposed EM is effective in this dataset: it results in the learned distributions being very close to the true distributions (to see that, compare the estimated parameters in Figure 5.3 with the true parameters in Figure 3.1). In addition, since the used models fit the data well, the clustering performance is very good in all three cases, even with the difficult case in *dataset (ii)* where the feature densities from all clusters completely overlap (see Figure 5.3).

5.5.2 Clustering with Texture images dataset

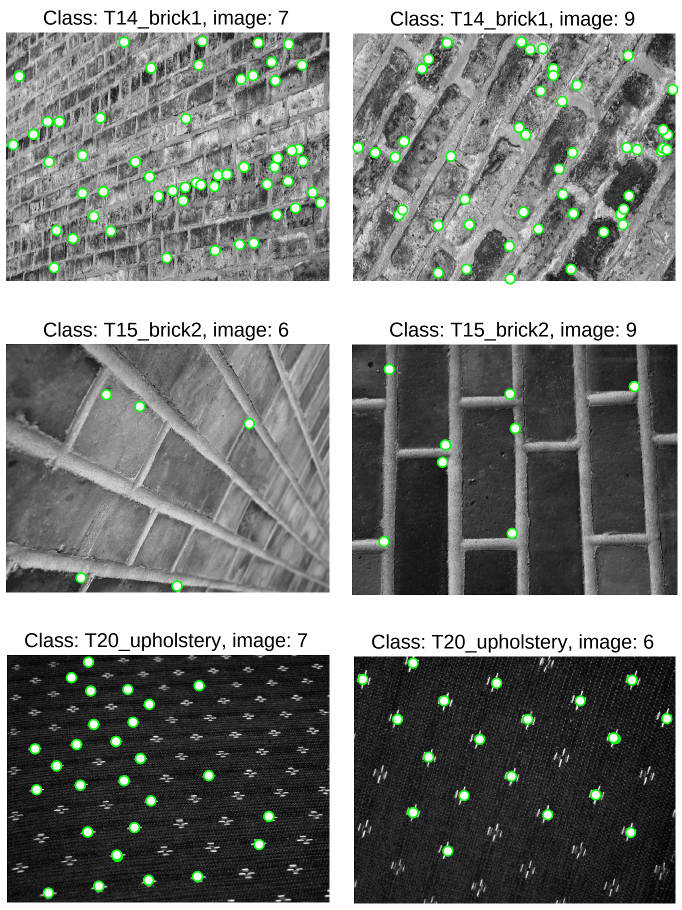

Class: T14_brick1, image: 7
Class: T14_brick1, image: 9
Class: T15_brick2, image: 6
Class: T15_brick2, image: 9
Class: T20_upholstery, image: 7
Class: T20_upholstery, image: 6

Figure 5.4: Example images from classes "T14_brick1", "T15_brick2", and "T20_upholstery" of the Texture dataset. Circles mark the detected SIFT keypoints. (This figure is identical to Figure 3.2, but reproduced here for reading convenience.)

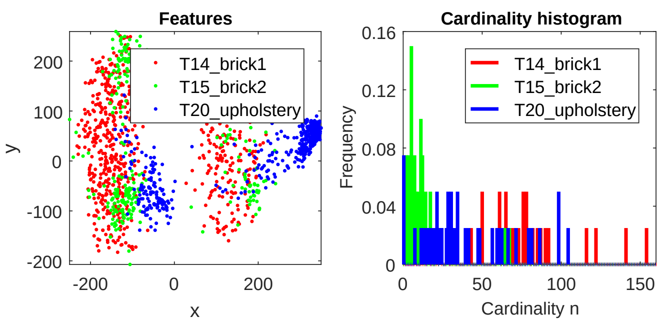


Figure 5.5: Extracted data from images of the Texture dataset. Left: 2-D features (after applying PCA to SIFT features). Right: Histogram of cardinalities of the extracted data. (This figure is identical to Figure 3.3, but reproduced here for reading convenience.)

This experiment uses images from three classes "T14 brick1", "T15 brick2", and "T20 upholstery" of the Texture dataset described in Section 3.4.2, but without class labels. We examine Poisson models with different feature densities, i.e., the Poisson model with single Gaussian feature density, and the Poisson models with Gaussian mixture feature density. For all Poisson models, the EM algorithm (proposed in Section 5.3) with the number of clusters $K = 3$ is used to learn the model and cluster the data.

For the first Poisson model with single Gaussian feature density, the learned distributions and the clustering performance are shown in Figure 5.7a. It can be observed that the clustering performance is not so good with this model, the reason being that the singe Gaussian feature distribution is *too simple* to fit this data well.

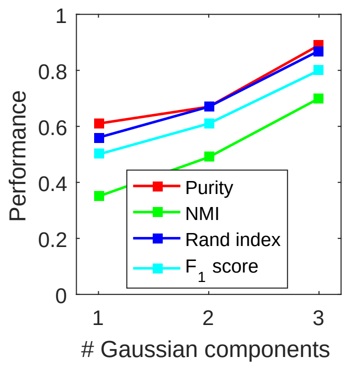
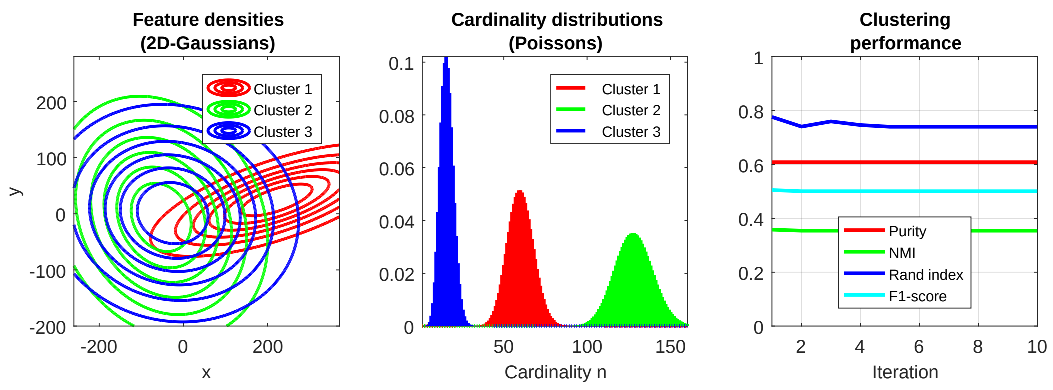
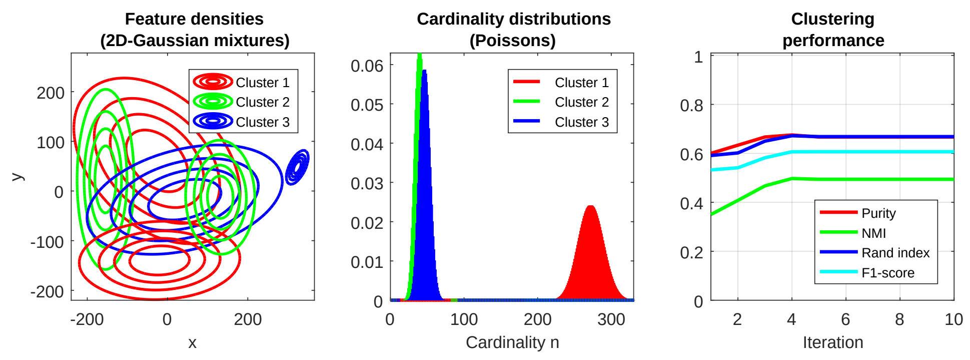


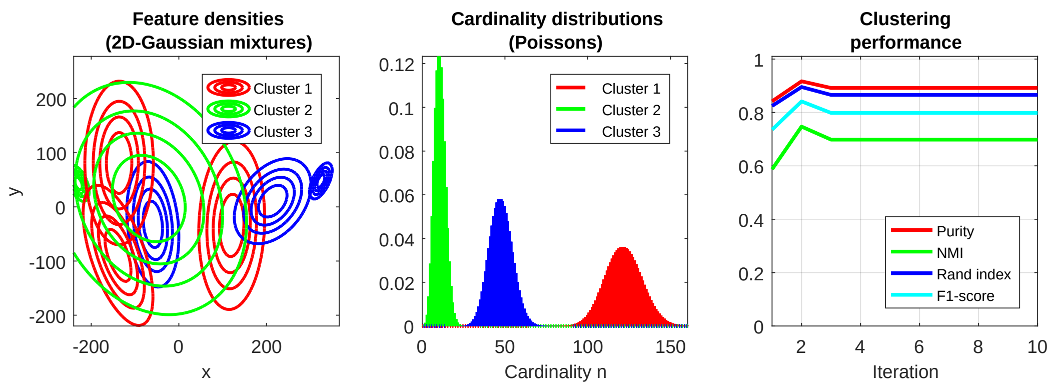
Figure 5.6: Clustering performance on Texture dataset by Poisson models with different number of Gaussian components in the feature density.



(a) Single Gaussian feature density



(b) Two-component Gaussian mixture feature density



(c) Three-component Gaussian mixture feature density

Figure 5.7: Learned Poisson models and the clustering performance on the Texture dataset.

For the Poisson models with Gaussian mixture feature density, since there is no close-form solution for the M-step in the proposed EM algorithm , we deploy an additional EM algorithm to learn the feature density (as discussed in Section 5.3). In particular, we first re-sample the given data by duplicating PPs in such a way that the weight of a PP X_n approximates $p(y_n = k \mid X_n, \psi^{(i-1)})$ (computed by Eq. (5.3)). Then, the disjoint union is applied on these PPs to form a point dataset consisting of all constituent points of the PPs. Finally, a standard EM is executed on the obtained point dataset to learn the Gaussian mixture feature density.

The learned distributions and the clustering performance by these models are shown in Figure 5.7b and Figure 5.7c. Observe that the clustering performance is significantly improved (especially when three-component Gaussian mixture feature density is used) since these models better fit the data. This also illustrates the importance of choosing a suitable data model in the model-based learning technique.

5.5.3 Finding proximity groups with Reality Mining dataset

In this experiment, we use the *Reality Mining dataset* (Eagle and Pentland, 2006). This dataset was collected at the MIT Media Lab using mobile phones from about 100 participants over 9 months in 2004. The participants were recruited on a voluntary basis from the members of the Media Lab and the students of the Sloan Business School, which is adjacent to the Media Lab. The dataset includes multiple types of data recorded from the phones such as call logs, cell tower IDs, application usage, phone status, and Bluetooth scans. To demonstrate the feasibility of our approach on discrete data, we focus on *Bluetooth scan logs*.

During the data collection, every 5 minutes, the phone scans for surrounding phones which are Bluetooth enabled and records the IDs[6] of the detected ones. We assume that each participant in this experiment owns only one phone, thus the phone ID can also be considered as the ID of its carrier/owner. Adding

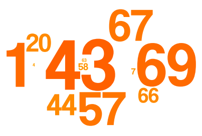
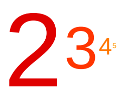
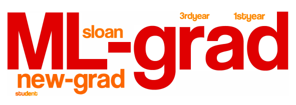
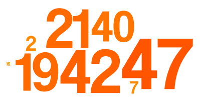
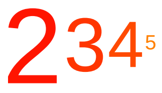
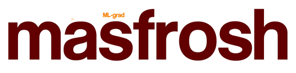
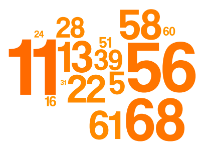
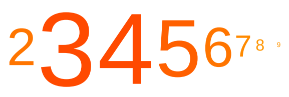
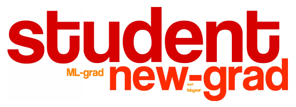
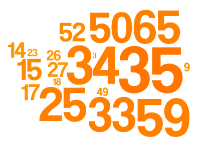
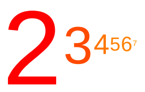
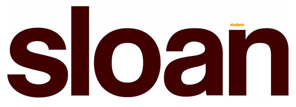
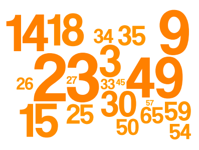
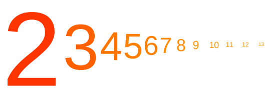
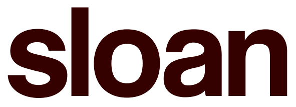
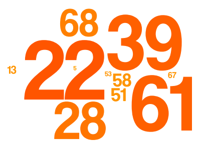
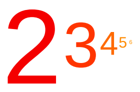
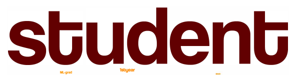
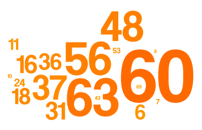
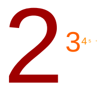
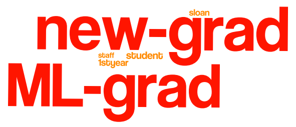
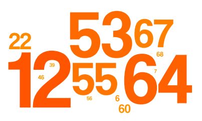
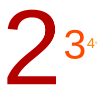
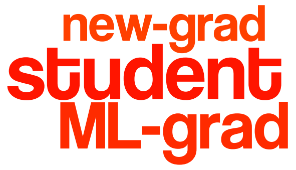
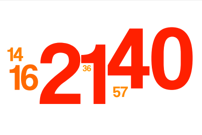
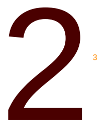
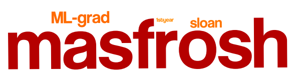
[6] In this experiment, we map each phone to a unique ID in the range from 1 to N, where N is the number of considered phones.

the scan owner's phone ID, each Bluetooth scan is a PP of phone IDs (that are co-located at the scanning time).

These PPs are clustered to find *proximity groups* of participants using the proposed EM algorithm with the number of components $K = 9$ (preferred by (Nguyen et al., 2013)). In this experiment, each group is represented by an IID-cluster point process with categorical feature and categorical cardinality distribution. The learned point processes are plotted in Figure 5.8, where the feature and cardinality distributions (categoricals) are visualized as tag-clouds of participant IDs and cardinalities, respectively. Note that the feature distribution of a group indicates the appearance frequency of participants in the group, whereas the cardinality distribution can be interpreted as the pattern of the number of participants gathering.

Note that the dataset does not include group labels of observations, thus we cannot evaluate the clustering result quantitatively. However, since the dataset also provides the affiliation labels of participants, we can justify how appropriate the clustering is in terms of the affiliation structure. In particular, the participants are either members of the MIT Media Lab or students of the Sloan Business School. The members of the MIT Media Lab are further split into subgroups: *staff*, *student*, *ML-grad*, *new-grad*, and *masfrosh*. Most of them work or study at the Media Lab on a full time basis, except the masfrosh members, who are first-year master students, only attend some classes at the Media Lab. The Sloan students (denoted as the *sloan* group) only attend classes at the Sloan Business School.

To visualize the affiliation structure of the clusters, we count the number of participants belonging to each affiliation in each cluster, then use these counts as weights to plot the tag clouds of the affiliation labels (the third column in Figure 5.8). Observe that the found affiliation structure of clusters discovered by our approach is consistent with those published in the pervasive computing domain, such as (Nguyen et al., 2013, 2016). However, unlike those studies

No.	Feature distribution (Participant ID cloud)	Cardinality distribution (Cardinality cloud)	Clustering result (Affiliation cloud)
1			
2			
3			
4			
5			
6			
7			
8			
9			

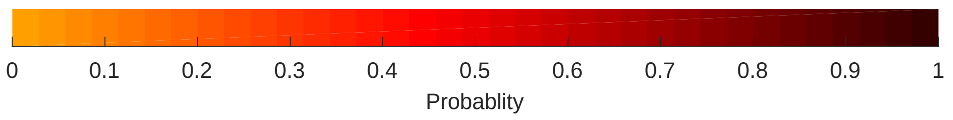


Figure 5.8: Proximity patterns learned from Reality Mining dataset using the proposed PP clustering method. The font size and color (with the color map plotted at the bottom) indicate the frequency/weight of objects.

that can only discover the members of each proximity group (i.e., the feature distributions), our approach can also discover the cardinality (i.e., the size of observations), which can be used to better explain the groups.

For example, clusters 4 and 5 represent the proximity groups of Sloan students, since they consist of almost only Sloan members (Figure 5.8). This is reasonable since the Sloan students usually meet each other and rarely meet members of other affiliations. However, note that these two groups are different in both feature and cardinality distributions (Figure 5.8), which implies two sub-groups in the Sloan students: students 25, 33, 34, 35, 50, 59, 65 usually meet each other in small groups of 2 or 3 individuals (cluster 4), while students 9, 14, 15, 18, 23, 49 may meet each other in groups of more individuals (cluster 5). This may indicate different gathering habits of different groups of students in this school.

Another interesting pair of clusters is 2 and 9, which mainly consist of masfrosh students. Although these two clusters both consist of masfrosh students, their cardinalities are again different, and the same justification as given above can be made: students 19, 21, 40, 42, 47 usually meet each other in groups of 2 to 4 individuals (cluster 2), while students 21, 40 very usually only meet each other (cluster 5), indicating that they (students 21 and 40) may be close friends.

CHAPTER 6

DISTANCE-BASED LEARNING FOR POINT PATTERN DATA

Chapter's key points:

- Set distances and PP learning
- Distance-based PP clustering
- Distance-based PP classification
- Distance-based PP novelty detection
- Comparison with model-based approaches

In the three previous chapters, we presented model-based approach for PP learning problems. These approaches suit cases where the data follow (or are assumed to follow) some underlying statistical model. However, when this is not the case, the proposed methods are not relevant or perform poorly. In this chapter, we introduce the alternative approaches for PP learning using the set distances (described in Section 2.5). We start this chapter with a general discussion on the properties of set distances and the implications in the context of design choices for PP learning algorithms (Section 6.1). In next sections, the distance-based PP learning methods are proposed: clustering in Section 6.2, classification in Section 6.3, and novelty detection in Section 6.4. The chapter concludes with a comparison and discussion of distance-based methods versus model-based methods (Section 6.5).

6.1 Set distances and PP learning

As previously mentioned in Section 2.5, a distance is a fundamental measure of dissimilarity between two objects. Hence, the notion of distance or metric is important to learning approaches without models (Jain, 2010; Amores, 2013; Pimentel et al., 2014). In the next sections, we discuss the properties of set distances and the implications in the context of design choices for PP learning algorithms. The choice of set distance in PP learning directly influences the performance and hence it is important to select distances that are compatible with the applications.

6.1.1 Hausdorff distance

Hausdorff distance (defined in Section 2.5) has been successfully applied in applications dealing with PP data, such as detecting objects from binary images (Huttenlocher et al., 1993a; Rucklidge, 1995), or measuring the dissimilarities between 3-D surfaces — sets of coordinates of points (Cignoni et al., 1998). In PP learning, it has been applied in classification (Amores, 2013) and clustering (Zhang and Zhou, 2009).

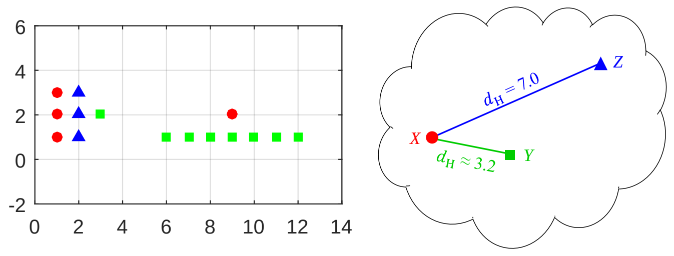


Figure 6.1: Cardinality difference and outliers in the Hausdorff distance. Left: Sets X (red ●), Y (green ■), and Z (blue ▲) in $\mathbb{R}^2$. Right: Abstract impression of the Hausdorff distances between the finite sets X, Y, and Z.

The Hausdorff distance could produce some undesirable effects for many PP learning applications since it may group together PPs that are intuitively dissimilar while separating PPs that are similar. Specifically:

- The Hausdorff distance is relatively *insensitive to dissimilarities* in cardinality (Schuhmacher et al., 2008). Consequently, it can group together PPs with large differences in cardinality (e.g., X and Y in Figure 6.1). This can be undesirable in many applications since the cardinalities of the PPs are important in PP learning.

- The Hausdorff distance *penalizes outliers heavily* — elements in one set which are far from every element of the other set (Schuhmacher et al., 2008). Consequently, it tends to separate similar sets that differ only in a few outliers (e.g., X and Z in Figure 6.1). This is undesirable in applications where the observed PPs of underlying groups are contaminated by outliers due to spurious noise. Nonetheless, there are applications where it is desirable to separate PPs with outliers from those without.

The **Chamfer "distance"** (Gavrila and Philomin, 1999):

$$d_{\mathtt{C}}(X,Y) = \frac{1}{|X|} \sum_{x \in X} \min_{y \in Y} \underline{d}(x,y), \tag{6.1}$$

is a variation of the Hausdorff construction, but it does not satisfy the metric axioms. Since it is very similar to the Hausdorff distance, it also suffers from issues similar to those discussed above.

6.1.2 Wasserstein distance

The Wasserstein distance (defined in Section 2.5) partially addresses cardinality insensitivity and reduces the undesirable penalty on the outliers of the Hausdorff distance (Schuhmacher et al., 2008). See, for example, in Figure 6.1, the Wasserstein distances $d_{\mathtt{W}}^{(2)}(X,Y) \approx 5.9 > d_{\mathtt{W}}^{(2)}(X,Z) \approx 3.6$ as expected. However, it still has a number of drawbacks.

- It is still possible for the Wasserstein distance to group together dissimilar sets while separating similar sets, as illustrated in Figure 6.2. Intuitively, X and Z are very similar whereas X and Y are quite dissimilar, but the Wasserstein distance disagrees, i.e., $d_{\mathtt{W}}^{(2)}(X,Y) \approx 1.6 < d_{\mathtt{W}}^{(2)}(X,Z) \approx 2.3$. The large Wasserstein distance between X and Z is due to the moving of earth from the bottom blue pile in Figure 6.3b over long distances (the two longest blue arrows to red piles in Figure 6.3b). Note that the elements of Z are not so balanced around the elements of X, and thus require the pile to be moved over long distances. On the other hand, the elements of Y are more balanced around the elements of X thereby requiring less work (see Figure 6.3a) and hence a smaller resulting distance. In general, the Wasserstein distance depends on *how well balanced* the numbers of points of X are distributed among the points of Y.

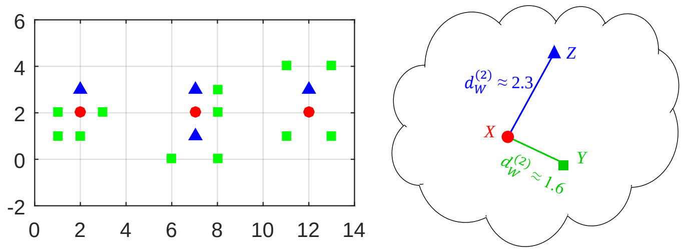
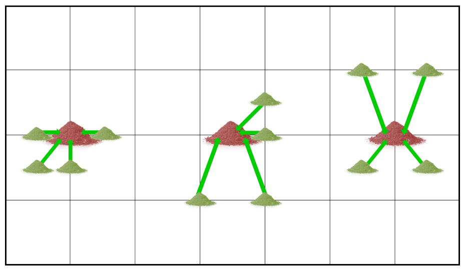
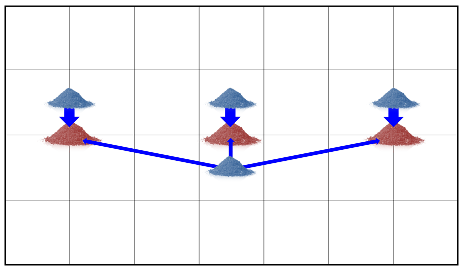


Figure 6.2: Left: Sets X (red ●), Y (green ▪), and Z (blue ▲) in $\mathbb{R}^2$. Right: Abstract impression of the Wasserstein distances between the finite sets X, Y, and Z.



(a) To compute $d_W^{(p)}(X,Y)$: move Y's earth piles (green) to form X's earth piles (red).



(b) To compute $d_W^{(p)}(X,Z)$: move Z's earth piles (blue) to form X's earth piles (red).

Figure 6.3: Earth mover's interpretation of the Wasserstein distance. Sets X, Y, and Z in Figure 6.2 are considered as collections of earth piles. The blue/green arrows represent the amounts and directions of the transportations of the blue/green earth piles. (This figure is identical to Figure 2.6, but reproduced here for reading convenience.)

- Both the Wasserstein and Hausdorff distances are *not defined if one of the sets is empty*. However, in PP data, empty PPs are not unusual. For example, in WiFi log data where each datum (a log record) is a set of WiFi access point IDs around the scanning device at a given time, there are instances when there are no WiFi access points leading to empty observations. In image data where each image is represented by a set of features describing some objects of interest, images without any object of interest are represented by empty PPs.

6.1.3 OSPA distance

The OSPA distance (defined in Section 2.5) is a metric with several salient properties that can address some of the undesirable effects of the Hausdorff and Wasserstein distances (Schuhmacher et al., 2008).

- The OSPA distance penalizes relative differences in cardinality in an impartial way by introducing an additive component on top of the average distance in the optimal sub-pattern assignment. The first term in Eq. (2.23) (see Section 2.5) is the dissimilarity in feature while the second term is the dissimilarity in cardinality.
- The OSPA distance is defined for any two PPs. It is equal to c (i.e., maximal) if only one of the two PPs is empty, and zero if both PPs are empty.
- *The outlier penalty can be controlled via parameter* p. The larger the p, the heavier the penalty on outliers. Note that the role of p in OSPA is similar to that for the Wasserstein distance, however, it is mitigated due to the cutoff c. In practice, it is common to use $p = 2$.[1]
- *The cutoff parameter* c *controls the trade-offs between feature dissimilarity and cardinality difference* (see Figure 6.4 for illustration). Indeed, c determines the penalty for cardinality difference and is also the largest allowable base distance between constituent elements of any two sets. As a general guide: 1) to *emphasize feature dissimilarity*, c should be as *small* as the typical base distance between constituent elements of the PPs in the given dataset; 2) conversely, to *emphasize cardinality difference*, c should be *larger than* the maximum base distance in the given dataset; 3) for a *balanced emphasis* on cardinality and feature, a *moderate* value of c in between the two aforementioned values should be chosen.

Figure 6.4 shows four PPs X, Y, Z and O, where: Y has elements that are

[1] For the rest of this chapter, we use $p = 2$, unless stated.

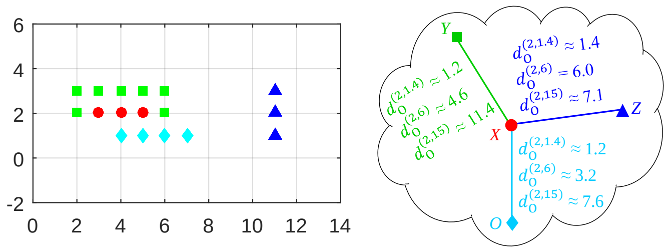


Figure 6.4: Trade-offs between feature dissimilarity and cardinality difference in the OSPA distance. Left: Sets X (red ●), Y (green ■), Z (blue ▲), and O (cyan ♦) in $\mathbb{R}^2$. Right: Abstract impression of the OSPA distances between the sets X, Y, Z, and O.

closest to the individual elements of X, but has a larger cardinality; Z has elements far away from the elements of X, but has the same cardinality; while O is visually most similar to X. In this scenario, the typical base distance between the elements of the PPs is about 1.4 and the maximum base distance is about 9.5. Choosing a small cutoff $c = 1.4$ yields $d_O^{(2,1.4)}(X, Y) < d_O^{(2,1.4)}(X, Z)$, indicating an emphasis of feature dissimilarity over cardinality difference. Choosing a large cutoff $c = 15$ yields $d_O^{(2,15)}(X, Y) > d_O^{(2,15)}(X, Z)$, indicating an emphasis of cardinality difference over feature dissimilarity. Choosing a moderate cutoff $c = 6$, makes O closest to X, indicating a balanced emphasis on both feature dissimilarity and cardinality difference.

6.2 Distance-based PP clustering

In the distance-based clustering context, the overall goal is to partition a given dataset $\mathcal{D} = \{X_1, ..., X_N\} \subseteq \mathbb{X}$ into disjoint clusters which minimize the sum of the distances between observations and their cluster centers, while penalizing the trivial partition $\mathcal{P} = \{\{X_1\}, ..., \{X_N\}\}$ (i.e., each observation is a cluster) which yields a zero sum of distances. More concisely, let $\boldsymbol{\mu} : \mathcal{D} \to \mathbb{X}$ be a mapping that assigns a cluster center to each observation in $\mathcal{D}$, i.e., $\boldsymbol{\mu}(X)$ is the center of the cluster to which X belongs, then the clustering problem can be stated as

$$\min_{\boldsymbol{\mu}} \sum_{X \in \mathcal{D}} d\left(X, \boldsymbol{\mu}(X)\right) + \gamma(X)\delta_X[\mu\left(X\right)], \tag{6.2}$$

$$\text{subject to } \boldsymbol{\mu}\left(C\right) = C, \forall C \in \boldsymbol{\mu}(\mathcal{D}), \tag{6.3}$$

where $\delta_A[B] = 1$ if $A = B$, and is 0 otherwise, $\gamma : \mathcal{D} \to [0, \infty)$ is a user chosen penalty function that imposes a penalty for the selection of an observation X as its own cluster centre, and hence penalizes the identity map $\boldsymbol{\mu} : X \mapsto X$ as a solution.

Remark: The mapping $\boldsymbol{\mu}$ provides a partitioning $\mathcal{P} = \left\{\mathcal{P}_1, ..., \mathcal{P}_{|\boldsymbol{\mu}(\mathcal{D})|}\right\}$ of the dataset $\mathcal{D}$, where $\mathcal{P}_k = \{X \in \mathcal{D} : \boldsymbol{\mu}\left(X\right) = C_k\}$ is the k^{th} cluster and $\boldsymbol{\mu}\left(\mathcal{D}\right) = \left\{C_1, ..., C_{|\boldsymbol{\mu}(\mathcal{D})|}\right\}$ is the set of cluster centers or centroids. The constraints ensure that if an observation $C \in \mathcal{D}$ is a cluster centre, then C must belong to the cluster with centre C. The user defined penalty $\gamma(X)$ can also be interpreted in terms of the preference for datum X to be a centroid: the smaller the $\gamma(X)$, the more we prefer X to be a centroid.

Note that the cluster center $\boldsymbol{\mu}\left(X\right)$ of a datum X can be either defined as the mean (or more generally the *Fréchet mean*) of the observations in its group (e.g., k-means) or chosen among observations in the dataset, i.e., $\boldsymbol{\mu} : \mathcal{D} \to \mathcal{D}$ (e.g., k-medoids). In general, the Fréchet mean of a collection of observations is computationally intractable (Baum et al., 2015) and a better strategy is to select the centroids from the dataset. Such centroids, also known as *exemplars* (Frey and Dueck, 2007), can be efficiently computed as well as serving as real prototypes for the data.

6.2.1 Existing distance-based method for PP clustering

To the best of our knowledge, the Bag-level Multiple Instance Clustering (BAMIC) (Zhang and Zhou, 2009) is the only distance-based clustering algorithm for PPs. BAMIC adapts the k-medoids algorithm with the Hausdorff distance as a measure of dissimilarity between PPs (Zhang and Zhou, 2009).

The Hausdorff distance, used by BAMIC, has several undesirable properties as discussed in Section 6.1.1. Moreover, since BAMIC is based on the k-medoids algorithm, it requires the number of clusters as input, which is not always available in practice. Determining the correct number of clusters is one of the most challenging aspects of clustering (Jain, 2010). While it is possible to apply techniques such as cross validation or silhouette to determine the number of clusters (Section 5.4.1), this process incurs substantial computational cost. In addition, it is mathematically more principled to jointly determine the number of clusters and their centers.

6.2.2 AP clustering with set distances

Overview

In this work, we propose to use the **affinity propagation (AP)** algorithm (Frey and Dueck, 2007) for the described clustering problem. The AP algorithm has been widely used in several applications due to its ability to automatically infer the number of clusters and its fast execution time. Some recent examples are image clustering (Dueck and Frey, 2007), protein interaction graph partitioning (Vlasblom and Wodak, 2009), text clustering (Guan et al., 2011), community detection (Nguyen et al., 2013), and location detection (Nguyen et al., 2014).

In particular, we propose a versatile PP clustering algorithm using the **AP algorithm** (Frey and Dueck, 2007) **with the OSPA distance** (defined in Section 2.5.4) as a dissimilarity measure. For the sake of performance comparison, we also include the Hausdorff and Wasserstein distances (Section 2.5.2 and Section 2.5.3) as baselines. Using message passing, AP provides good approximate solutions to the problem Eq. (6.2)-Eq. (6.3) (Dueck and Frey, 2007; Frey and Dueck, 2007), thereby determining the number of clusters automatically from the data. Compared to k-medoids (used in BAMIC), AP can find clusters faster with considerably lower error (Frey and Dueck, 2007) and does not require random initialization of cluster centers (since AP first considers all observations as exemplars). In addition, the OSPA distance does not suffer from the undesirable effects as the Hausdorff distance used in BAMIC, as well as being more flexible, as discussed in Section 6.1.3.

Algorithm

The AP algorithm uses the similarity values between all pairs of observations in the data set $\mathcal{D} = \{X_1, ..., X_N\}$ and the user-defined exemplar **preferences**, as input and returns the 'best' set of exemplars. The similarity values of interest in this work are the negatives of the OSPA distances between the PPs in $\mathcal{D}$. The preference value for a datum X_n is the *negative of the penalty*, i.e., $-\gamma(X_n)$, the larger its preference, the more likely that X_n is an exemplar. In AP, the exemplar for an observation X_n (which could be X_n itself or another observation) is represented by a variable c_n, where $c_n = k$ means that X_k is the exemplar for X_n.

Note that a configuration $(c_1, \ldots, c_N)$ provides an equivalent representation of the decision variable $\boldsymbol{\mu} : \mathcal{D} \to \mathcal{D}$ in problem Eq. (6.2)-Eq. (6.3) by defining $\boldsymbol{\mu}(X_n) = X_k$ iff $c_n = k$. Treating each c_n as a random variable, a factor graph with nodes $c_1, \ldots, c_N$ can be constructed by encoding into the functional potentials the similarities between pairs of observations, the preferences for each observation, as well as the constraints that ensure valid cluster configurations. Constraint Eq. (6.3) means that in a valid configuration $(c_1, \ldots, c_N)$, $c_{c_n} = c_n$, i.e., if X_k is an exemplar for any observation, then the exemplar of X_k is X_k. This constraint can be enforced by setting the potential of any configuration $(c_1, \ldots, c_N)$ with $c_{c_n} \neq c_n$ to $-\infty$. Ideally, performing max-sum message-passing yields a configuration that maximizes the sum of all potentials in this factor graph, and hence a solution to the clustering problem Eq. (6.2)-Eq. (6.3). AP is an efficient approximate max-sum message-passing algorithm using a protocol originally derived from loopy propagation on factor graphs (Frey and Dueck, 2007).[2] Further details on the AP algorithm can be found in (Frey and Dueck,

[2] An equivalent binary graphical model representation for AP was later proposed in (Givoni and Frey, 2009). Instead of creating a latent node for each individual observation as in (Frey and Dueck, 2007), a binary node $b_{n,k}$ is created for each pair (X_n, X_k) and $b_{n,k} = 1$ if X_k is an exemplar for X_n. Message-passing on this new factor graph representation yields the same

2007; Dueck and Frey, 2007). In the following, we discuss specific details for the clustering of PP data, summarized in Algorithm 2.

Algorithm 2: Clustering of PP data using set distances. See text for values of $\gamma(X_n)$.

```
Input: PP dataset $\mathcal{D} = \{X_1, ..., X_N\}$,
       stopping threshold $\theta$.
Output: cluster assignment $c_1, ..., c_N$.
for $n, k \in \{1, ..., N\}$ do
    initialize messages $r(n,k) = 0$; $a(n,k) = 0$;
    compute similarity $s(n,k) = -d_0(X_n, X_k)$;
    assign preference $s(n,n) = -\gamma(X_n)$;
end
repeat
    for $n, k \in \{1, ..., N\}$ do
        Update responsibility
         $r(n,k) = s(n,k) - \max_{k' \neq k}\{a(n,k') + s(n,k')\}$;          (6.4)
        Update availability
         $a(n,k) = \min\{0, r(k,k)\} + \sum_{n' \notin \{n,k\}} \max\{0, r(n',k)\}$;   (6.5)
        Update self-availability $a(k,k) = \sum_{n' \neq k} \max\{0, r(n',k)\}$;   (6.6)
    end
until change of any $r(\cdot,\cdot)$ or $a(\cdot,\cdot) < \theta$;
for $n \in \{1, ..., N\}$ do
    return cluster assignment $c_n = \underset{k}{\operatorname{argmax}} \left(r(n,k) + a(n,k)\right)$;
end
```

The algorithm starts by computing all pairwise *similarities* input for AP: $s(n,k) = -d_0(X_n, X_k)$, and *preferences* $s(k,k) = -\gamma(X_k)$. A common practice is to give all observations the same preference, e.g., the median of the similarities (which results in a moderate number of clusters) or the minimum of the similarities (which results in a small number of clusters) (Frey and Dueck, 2007).

The AP algorithm passes two types of messages. The **responsibility** $r(n,k)$, defined in Eq. (6.4) indicating how well X_n trusts X_k as its exemplar, is sent from observation X_n to its candidate exemplar X_k. Then, the **availability** $a(n,k)$, defined in Eq. (6.5) reflecting the accumulated evidence for X_k to be an exemplar for X_n, is sent from a candidate exemplar X_k to X_n. Note from Eq. (6.4)-

solution.

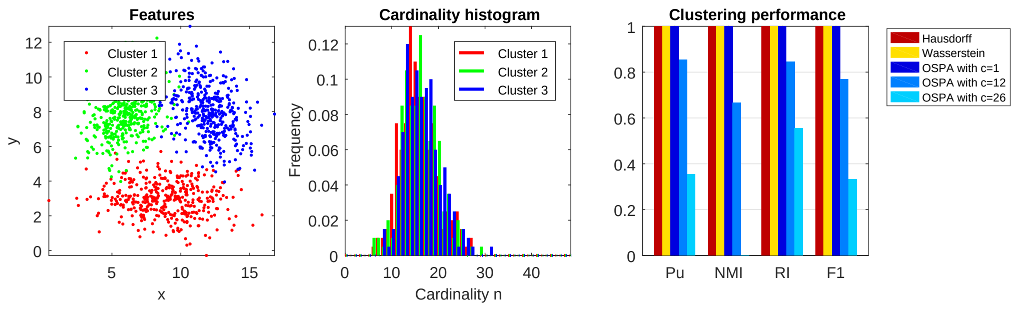
Eq. (6.5) that the responsibility $r\left(n,k\right)$ is calculated from the availability values that X_n receives from its potential exemplar, whereas the availability $a\left(n,k\right)$ is updated using the 'support' from observations that consider X_k as their *candidate exemplar*. Note from Eq. (6.5) that when an observation is assigned to exemplars other than itself, its availability falls below zero. Such negative availabilities in turn decrease the effect of input similarities $s\left(n,k'\right)$ in Eq. (6.4), thereby eliminating the corresponding PPs from the set of potential exemplars.

The loopy propagation is usually terminated when changes in the messages fall below a threshold (e.g., used in Algorithm 2), or when the cluster assignments stay constant for some iterations, or when the number of iterations reaches a given value (Frey and Dueck, 2007). The cluster label c_n is the value of k that maximizes the sum $r(n,k)+a(n,k)$ (Frey and Dueck, 2007).

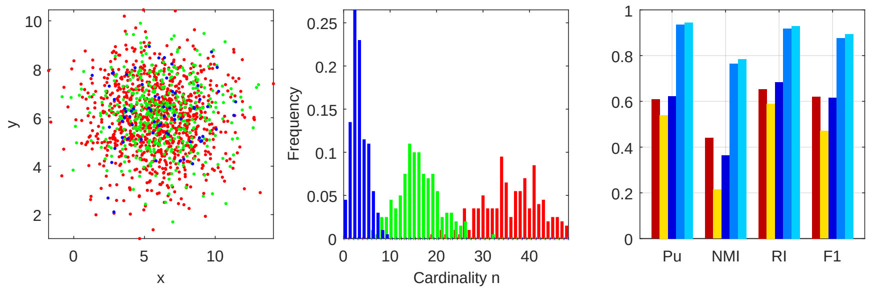
6.2.3 Experiments

In this section, we evaluate the performance of the proposed AP-based clustering algorithm on both simulated and real PP data. In particular, we compare the clustering performance amongst the Hausdorff, Wasserstein and OSPA distances. Note that BAMIC (which uses the k-medoids algorithm instead of AP) can be treated as AP clustering with the Hausdorff distance.[3]

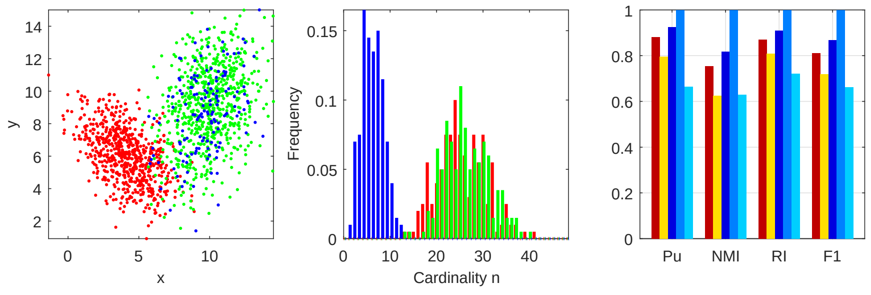
Since the result of the AP algorithm depends on the choice of exemplar preferences, we first empirically select the exemplar preference that yields the best performance in terms of the number of clusters for each distance, and then benchmark the best case performance of one distance against the others. The relevant performance indicators are: purity (Pu), normalized mutual information (NMI), rand index (RI), and F_1 score (F1) (see Section 2.3.5).



(a) Dataset (i)



(b) Dataset (ii)



(c) Dataset (iii)

Figure 6.5: Simulated data and clustering performance in three diverse scenarios of PP data. Dataset (i): well-separated in feature but overlapping in cardinality; dataset (ii): well-separated in cardinality but overlapping in feature; dataset (iii): a mix of (i) and (ii). (A part of this figure is identical to Figure 3.1, but reproduced here for reading convenience.)

Clustering with simulated data

In this experiment, we use the same simulated dataset described in Section 3.4.1. Three different cutoff values for the OSPA distance are examined: $c = 1$ (small); $c = 12$ (moderate); and $c = 26$ (large). Note that $c = 1$ is a typical value of the intra-PP base distance (i.e., base distance between the features within the PPs in the dataset), $c = 26$ is an estimate of the maximum intra-PP base distance, and $c = 12$ is a moderate value of the intra-PP base distance.

In *dataset (i)* (Figure 6.5a), the Hausdorff, Wasserstein and OSPA distances with small and moderate cutoffs show good performance. The OSPA distance with a large cutoff tends to emphasize the cardinality dissimilarities (which are negligible in this scenario) over feature dissimilarities (see Section 6.1.3) leading to poor clustering performance.

In *dataset (ii)* (Figure 6.5b), where cardinality difference is the main discriminative information, the Hausdorff and Wasserstein distances perform poorly since they are unable to capture cardinality dissimilarities between the PPs. The OSPA distance with a small cutoff tends to emphasize feature dissimilarities (which are negligible in this scenario) over cardinality dissimilarities (see Section 6.1.3) leading to poor performance. On the other hand the OSPA distance with moderate and large cutoffs perform better since they can appropriately capture cardinality dissimilarities.

In *dataset (iii)* (Figure 6.5c), the results again confirm the discussions above. The OSPA distance with moderate cutoff provides a balanced emphasis on both feature and cardinality dissimilarities (see Section 6.1.3), yielding the best performance.

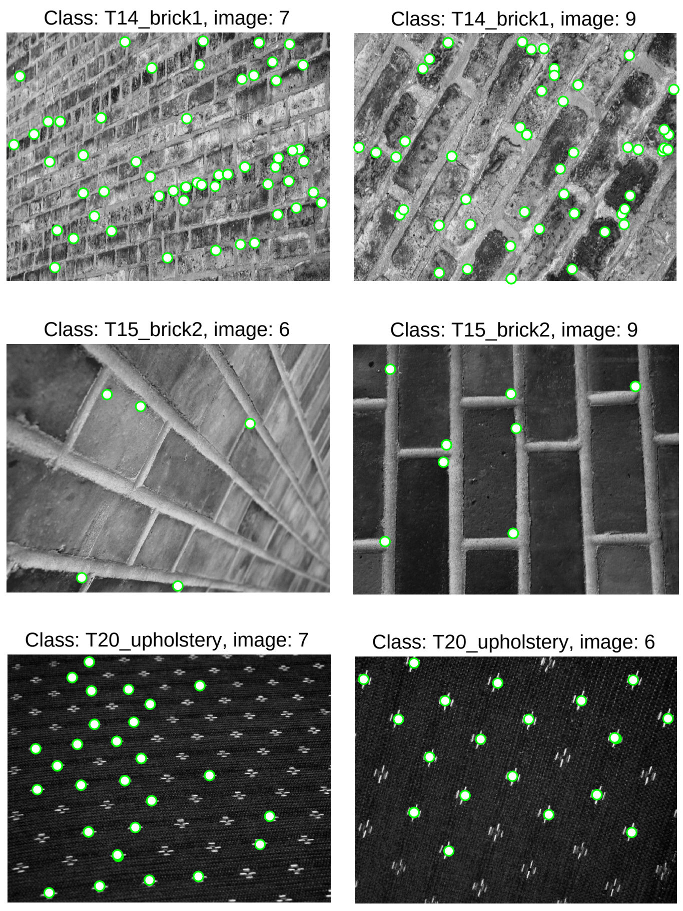

Class: T14_brick1, image: 7
Class: T14_brick1, image: 9
Class: T15_brick2, image: 6
Class: T15_brick2, image: 9
Class: T20_upholstery, image: 7
Class: T20_upholstery, image: 6

Figure 6.6: Example images from classes "T14_brick1", "T15_brick2", and "T20_upholstery" of the Texture dataset. Circles mark the detected SIFT keypoints. (This figure is identical to Figure 3.2, but reproduced here for reading convenience.)

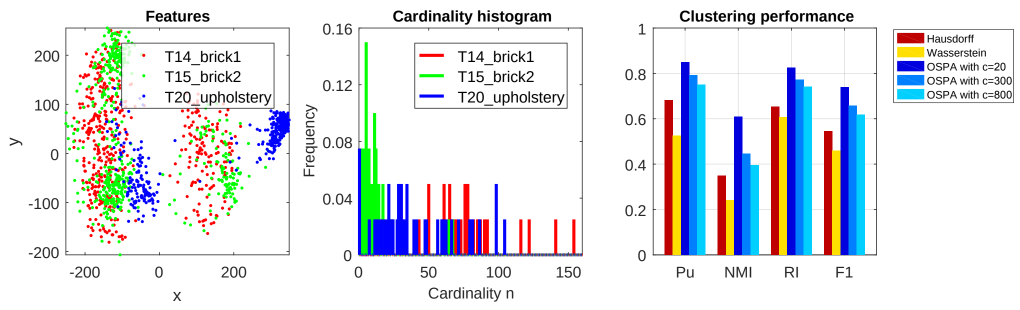


Figure 6.7: PP data extracted from images of classes "T14 brick1", "T15 brick2", and "T20 upholstery" of the Texture dataset, and clustering performance for various distances. (A part of this figure is identical to Figure 3.3, but reproduced here for reading convenience.)

Clustering with Texture dataset

In this experiment, we cluster the Texture images dataset described in Section 3.4.2. The clustering performance is summarized in Figure 6.7. It can be seen that the OSPA distances outperform the Hausdorff and Wasserstein distances, since it can properly incorporate both feature and cardinality information. The poor performance of the Hausdorff and Wasserstein distances is due to the significant overlap in the features and their inability to measure cardinality dissimilarities in the data.

6.3 Distance-based PP classification

The classification problem can be approached with or without knowledge of the underlying data model (Cover and Hart, 1967). In Chapter 3, we introduced a method for PP classification exploiting the underlying data model. In this section, we focus on another approach: the **k-nearest neighbors (k-NN)** classifier (Cover and Hart, 1967; Keller et al., 1985), which does not require prior knowledge of the data model.

6.3.1 Existing distance-based method for PP classification

Classifiers with distance-based techniques (such as k-nearest neighbour (k-NN) algorithm) using set distances such as *Hausdorff* (Huttenlocher et al., 1993b), *Chamfer* (Gavrila and Philomin, 1999), and *Earth Mover's* (Zhang et al., 2007; Rubner et al., 1998) are the only known distance-based methods for PP classification (Amores, 2013). These distances, however, do not perform well in some common cases of PP classification, such as when classes significantly overlap in cardinality, or when the PPs contain outliers.

6.3.2 k-NN classification with OSPA

The general k-NN classifier has two phases: training and classifying. Contrary to eager learning algorithms in which a model is learned from training data

[3] Note that the M^3IC algorithm (Zhang et al., 2009) (see Section 5.1) does not belong to the model-based or the distance-based approach, hence is out of the scope of this work. In future broader research, it may be included as another baseline.

in the training phase, the k-NN algorithm delays most of its computational effort until the classifying (or test) phase. In the training phase, the only task is storing class labels of the training observations. In the test phase, when a new observation is passed to query its label, the algorithm determines its k *nearest observations*, with respect to some distance, in the training set. The queried observation is then assigned the most popular label among its k nearest observations.

In this work, we propose to use the *OSPA distance* (Section 2.5.4 and Section 6.1.3) which is more versatile and better at capturing feature and cardinality dissimilarities between PPs (compared to aforementioned set distances), hence the OSPA distance would be more effective with the k-NN algorithm for PP classification.

Unlike existing k-NN classification which only stores the class labels in the training phase, our proposed approach exploits training data to *learn a suitable dissimilarity measure*. Since the fully observed training data can be used to assess whether the set distance agrees with the notion of similarity/dissimilarity of the application under consideration, in principle, a suitable distance can be learned. A simple approach is to perform cross-validation on the training data for a range of distances and select the best. Intuitively, a suitable distance entails small dissimilarities between observations in the same class, but large dissimilarities between observations from different classes. Hence, for a given training dataset, we seek a distance (or its parameterization) that minimizes the ratio of inter-class dissimilarity to intra-class dissimilarity. In general, learning an arbitrary distance from training data is numerically intractable. However, it is possible to learn low dimensional parameters such as the cutoff parameter in the OSPA distance.

The OSPA distance provides the capability to adapt the weighing between feature dissimilarity and cardinality dissimilarity via the *cutoff parameter* c (see Section 6.1.3). While the right balance between feature and cardinality dissimilarities varies from one application to another, it can be learned from the fully observed training data via cross-validation. However, cross-validation is not suitable for small training datasets. In the following, we describe an alternative approach that also accommodates small datasets.

Let $\bar{d}_0^{(p,c)}(X, C)$ denote the average OSPA distance with cutoff c, from a PP X to its k nearest neighbours in a collection C (of PPs), and let C_ℓ denote the class of PP observations with class label ℓ in the training set. Then, the *intra-class dissimilarity* for C_ℓ is defined by $\hat{D}^{(p,c)}(C_\ell) = \max_{X \in C_\ell} d_0^{(p,c)}(X, C_\ell)$ while its *inter-class dissimilarity* is defined by $\check{D}^{(p,c)}(C_\ell) = \min_{j \neq \ell} \min_{X \in C_\ell} d_0^{(p,c)}(X, C_j)$. To enforce small intra-class dissimilarity and large inter-class dissimilarity, we seek cutoff parameters that minimize the worst-case (over the training data set) ratio of intra-class dissimilarity to inter-class dissimilarity

$$\alpha(c) = \max_\ell \left(\frac{\hat{D}_0^{(p,c)}(C_\ell)}{\check{D}_0^{(p,c)}(C_\ell)} \right). \tag{6.7}$$

The operations max, min in the definition of α can be replaced by the average or a combination thereof.

6.3.3 Experiments

In the following experiments, we benchmark the classification performance of the OSPA distance against the Hausdorff[4] and Wasserstein[5] distances on both simulated and real data. Since the performance depends on the choice of k (the number of nearest neighbours), we ran our experiments for each $k \in \{1, ..., 10\}$ and benchmark the best case performance of one distance against the others.

Classification with simulated data

This experiment also examines the classification performance on the three diverse scenarios from the simulated datasets of Section 3.4.1. Using a 10-fold cross validation, the average classification performance is summarized in Fig-

[4] and hence the Chamfer "distance", see Section 6.1.1.
[5] and hence the Earth Mover's distance, see Section 2.5.3.

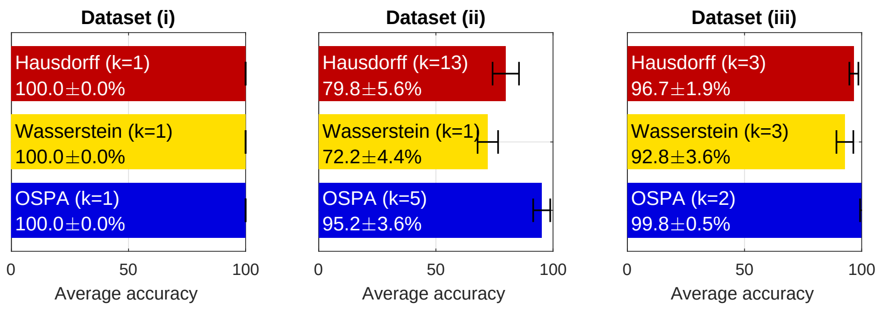


Figure 6.8: Classification performance on simulated data for various distances (k is the number of nearest neighbours). The error-bars represent the standard deviations of the accuracies.

ure 6.8.

Observe that in *dataset (i)*, where the features of the PPs from one cluster are well separated from those of the other clusters, all distances perform well. In *dataset (ii)* and a part of *dataset (iii)*, where the features of the PPs from one cluster overlap with those of the other clusters, the OSPA distance outperforms the Hausdorff and Wasserstein since it can better capture the cardinality dissimilarities in the data.

Classification with Texture data

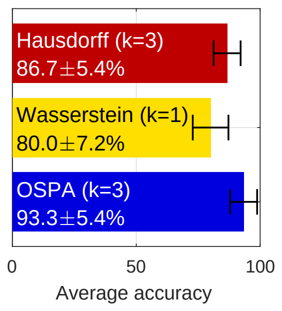


Figure 6.9: Classification performance on Texture data for various distances (k is the number of nearest neighbours). The error-bars represent the standard deviations of the accuracies.

This experiment examines the classification of the extracted PP data from the Texture image dataset described in Section 3.4.2. Using a 4-fold cross validation, the average performance is summarized in Figure 6.9. It can be observed that in this dataset, the OSPA distance again performs best, since it can provide a good balance between feature and cardinality dissimilarities.

6.4 Distance-based PP novelty detection

As previously discussed, the novelty detection problem for PP data has not been studied in the literature (see Section 4.1). Like classification, novelty detection can be approached with or without knowledge of the underlying data model. In Chapter 4, we introduced a method for PP novelty detection using the point process model. In this section, we focus on another approach without using a data model: the *nearest neighbor* method, the most common approach to novelty detection (Pimentel et al., 2014).

6.4.1 Novelty detection with set distances

In general, the **nearest neighbor** method is based on the assumption that normal observations are closer to the training (normal) data than novelties (Hautamäki et al., 2004). This approach requires a suitable notion of distance between observations (Pimentel et al., 2014). This section introduces a solution to the novelty detection problem for PP data by incorporating set distances into the nearest neighbour algorithm.

If the distance (e.g., Hausdorff, Wasserstein or OSPA) between the candidate PP and its *nearest normal neighbour* (NNN) is greater than a given threshold, then the candidate is deemed as novel, otherwise it is normal. A suitable threshold can be chosen experimentally. One suitable threshold (as chosen for experiments in this work) is the 95th-percentile of the inter-class distances (between normal training observations and their NNNs). However, we stress that no single threshold is guaranteed to work well for all cases.

Similar to classification with OSPA (Section 6.3.2), training data can be used to determine a suitable balance between feature dissimilarity and cardinality dissimilarity with the cutoff parameter c. However, there is no inter-class dissimilarity, and hence minimizing the intra-class dissimilarity for normal data yields the trivial solution $c = 0$. To determine a suitable balance, consider the *cardinality dissimilarity* $d^{(p)}_{\texttt{card}}(X, Y) = \frac{1}{n}(n - m)$ and *feature dissimilarity*

$d_{\mathtt{feat}}^{(p)}(X,Y) = \frac{1}{n}\min_{\pi\in\Pi_n}\sum_{i=1}^{m}\underline{d}\left(x_i,y_{\pi(i)}\right)^p$ between all pairs of observations X,Y in the normal training set (assuming the cardinality m of Y is not greater than the cardinality n of X, otherwise we compute $d_{\mathtt{card}}^{(p)}(Y,X)$ and $d_{\mathtt{feat}}^{(p)}(Y,X)$).

Note that for $d_{\mathtt{feat}}^{(p)}$ we use the base distance $\underline{d}$ to capture the absolute feature dissimilarity rather than the capped feature dissimilarity from base distance $\underline{d}^{(c)}$ (Section 2.5.4). To decide whether a test PP datum T is novel, we need to determine its cardinality dissmilarity and feature dissimilarity relative to the normal data. The relative cardinality dissmilarity and feature dissimilarity of T (with respect to the normal data) can be defined as $d_{\mathtt{card}}^{(p)}(T,T^*))/m_{\mathtt{card}}^{(p)}$ and $d_{\mathtt{feat}}^{(p)}(T,T^*)/m_{\mathtt{feat}}^{(p)}$, where T^* is T's NNN, $m_{\mathtt{card}}^{(p)}$ and $m_{\mathtt{feat}}^{(p)}$ are large values (e.g., maximum or 95th-percentile) of $d_{\mathtt{card}}^{(p)}(Y,X)$ and $d_{\mathtt{feat}}^{(p)}(Y,X)$) in the normal data set, respectively. It can be observed that summing the relative dissmilarities and scaling by $m_{\mathtt{feat}}^{(p)}$ gives the uncapped OSPA "distance"

$$(d_{0}^{(p)}(T,X(T)))^p = \frac{m_{\mathtt{feat}}^{(p)}}{m_{\mathtt{card}}^{(p)}}d_{\mathtt{card}}^{(p)}((T,T^*) + d_{\mathtt{feat}}^{(p)}(T,T^*) \qquad (6.8)$$

Hence, a suitable cutoff parameter is $c = \left(m_{\mathtt{feat}}^{(p)}/m_{\mathtt{card}}^{(p)}\right)^{1/p}$.

6.4.2 Experiments

In this section, we examine the novelty detection performance of the Hausdorff, Wasserstein and OSPA distances on both simulated and real data.

Novelty detection with simulated data

In this experiment, we consider class 2 from each simulated data set in Section 3.4.1 as normal data, and classes 1 and 3 as novel data. This allows us to study three diverse scenarios: *dataset (i)* is an example of **feature novelty**, where novel observations are similar in cardinality to normal training data, but dissimilar in feature (Figure 3.1a); *dataset (ii)* is an example of **cardinality novelty**, where novel observations are similar in feature with normal training data, but dissimilar in cardinality (Figure 3.1b); *dataset (iii)* is a **mix** of feature and cardinality novelty (Figure 3.1c).

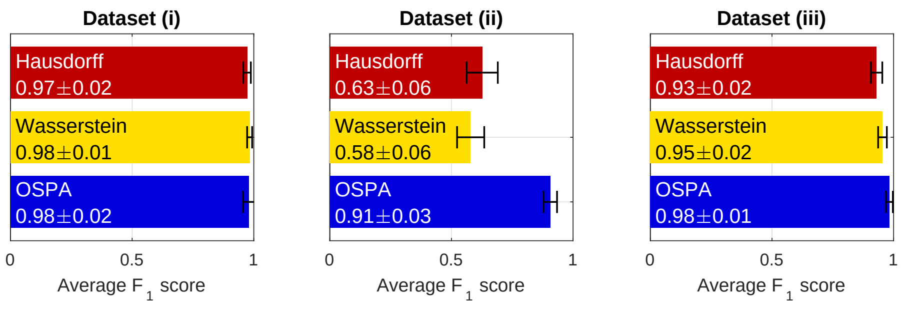
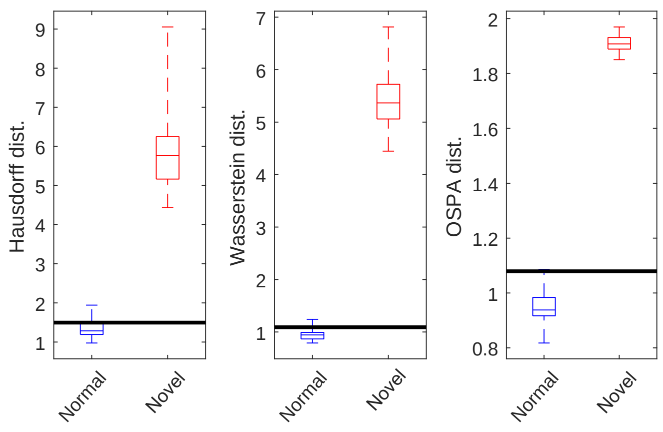


Figure 6.10: Novelty detection performance on simulated data for various distances. The error-bars represent the standard deviations of the F_1 scores.

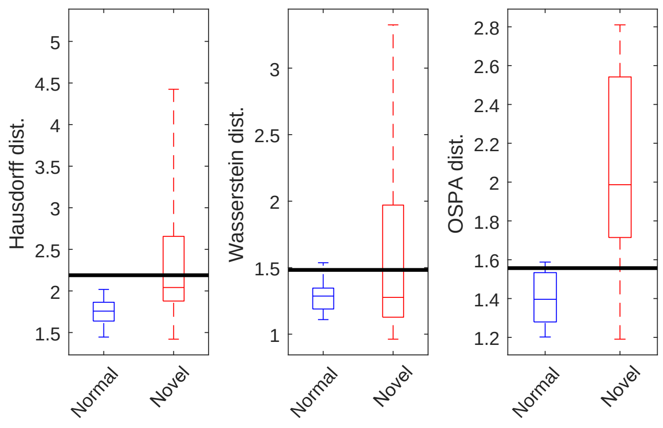
Using a 10-fold cross validation, the average performance is summarized in Figure 6.10. Figure 6.11 shows boxplots of the distances between the test PPs and their NNNs in the training set. It can be seen that in *dataset (i)*, where novel and normal data are dissimilar in feature, all distances perform well. In *dataset (ii)* and a part of *dataset (iii)*, where novel and normal data overlap in feature, the OSPA distance outperforms the Hausdorff and Wasserstein since it can appropriately penalize the cardinality dissimilarity between normal and novel data.

Novelty detection with Texture data

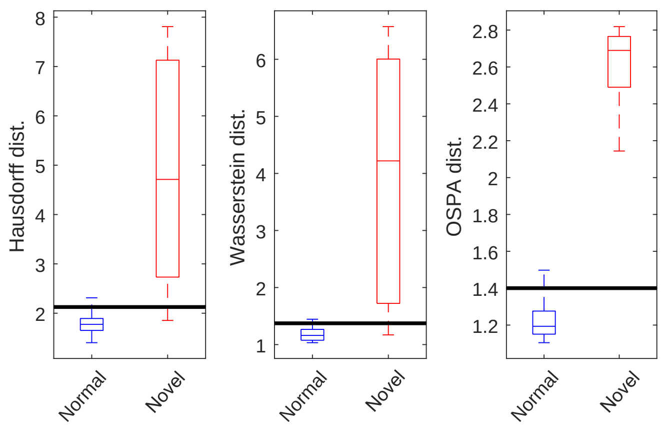
Using the Texture dataset from Section 3.4.2, we consider normal data taken from class "T14 brick1" and novel data taken from class "T20 upholstery". We use a 4-fold cross validation scheme. In each fold, the training data consist of 75% of images from the normal class (30 images), the testing set consists of the remaining images from normal class (10 images) and 25% of images from the novel class (10 images). For this dataset, all distances perform well (see Figure 6.12 and Figure 6.13) since normal and novel data are sufficiently distinguished in feature (see the feature plot in Figure 4.12).



(a) Dataset (i)



(b) Dataset (ii)



(c) Dataset (iii)

Figure 6.11: Boxplots of distances between test data and their NNNs for one data fold of the simulated dataset. Thick lines across the boxes indicate the chosen thresholds.

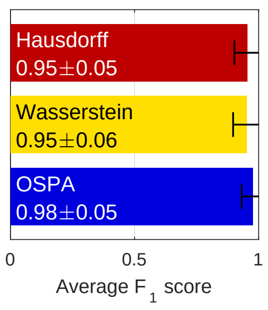


Figure 6.12: Novelty detection performance on Texture data for various distances. Error-bars indicate standard deviations of F1-scores.

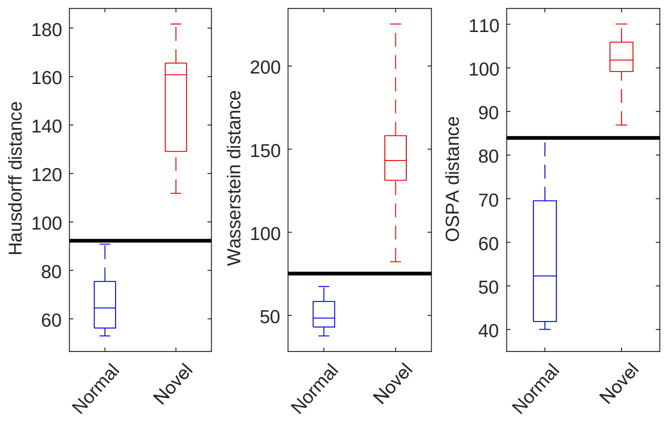


Figure 6.13: Boxplots of distances between test data and their NNNs for one data fold of the Texture dataset. Thick lines across the boxes indicate the chosen thresholds.

6.5 Comparison with model-based methods

This section compares the performance of the distance-based methods proposed in this chapter with the equivalent model-based methods proposed in Chapter 3, Chapter 4, and Chapter 5. In *classification*, the distance-based approach using k-NN classifier with OSPA distance (Section 6.3) and the model-based approach using IID-cluster models (Chapter 3) both show similar performance. The good performance of these approaches on both simulated and real data (Texture images) indicates that both IID-cluster models and the OSPA distance are versatile enough for handling various scenarios of PP data in classification.

In *novelty detection*, although the proposed ranking function outperforms other model-based methods, such as NB likelihood and Poisson model density (Chapter 4), it is still not as good as the distance-based method based on the nearest neighbour technique with the OSPA distance (Section 6.4). The reason for this could be due to the ranking function which is based on the IID-cluster density derived with the nonuniform reference measure (see Chapter 4). This may indicate that problem of ranking PPs using data likelihood is still unsolved completely, hence requiring more effort in future research.

In *clustering*, for the simulated datasets, the performance of the distance-based approach using the AP algorithm with set distance (Section 6.2) is similar to that of the model-based approach using the mixture of Poisson models (Chapter 5). For the real dataset (Texture images), the model-based approach (with Poisson model with Gaussian mixture feature distribution) outperforms the distance-based approach, since the model-based methods have a greater degree of freedom to better describe the data, e.g., to describe the non-spherical shape of the features in this case.

CHAPTER 7
CONCLUSION

Chapter's key points:

- Summary and discussion on the work
- Some directions for future work

7.1 Summary and discussions

Point pattern data, also known as bags or multiple instance data, are abundant in nature and many applications (Chapter 1). However, fundamental PP learning problems, in particular classification, novelty detection, and clustering, have received limited attention. In this work, we attempted to solve these learning problems using two approaches: one with knowledge of the underlying data model (model-based approach), and one without (distance-based approach). The proposed methods are listed in Table 7.1.

TASK	**MODEL-BASED APPROACH**	**DISTANCE-BASED APPROACH**
Classification	MLE of IID-cluster point process	k-NN with OSPA distance
Novelty detection	Ranking function based on IID-cluster point process density	NN-based novelty detection with OSPA distance
Clustering	EM with mixture of IID-cluster point processes	AP with OSPA distance

Table 7.1: Proposed methods for PP learning problems.

Model-based approach

Central to the model-based approaches is the notion of generative models from which the data are (or assumed to be) sampled. In the context of PP learning, point process models can be used. In particular, for *classification*, we propose ML and MAP estimation methods to learn the IID-cluster models. The learned models are then used to classify new data using a Bayes classifier (Chapter 3). For *novelty detection*, we introduce a novel ranking function based on the IID-cluster model density. Data are ranked according to this function and PPs with a ranking lower than some threshold will be deemed novel (Chapter 4). For *clustering*, we propose to use a finite mixture of IID-cluster point processes as the generative model for PP clusters. Then, the EM technique is deployed to learn the mixture model and cluster the data (Chapter 5).

The greatest advantage of model-based approaches for PP learning is that they can exploit the statistical patterns in PP data. This not only can result in good performance in many applications (as shown in the experiments in Chapter 3, Chapter 4, and Chapter 5), but also assists in gaining some insights into the data (see e.g., the experiment of finding proximity groups in Section 5.5.3). In addition, the model learned from the data can also be viewed as a characterization of the data and can be applied in tasks such as data stream clustering and classification (Nguyen et al., 2015).

Distance-based approach

The model-based approaches, however, have a main drawback that, in practice the true underlying model of the data may be unknown, or may be too complex to learn (Markou and Singh, 2003). In such scenarios, non-model-based approaches should be better choices. In this work, we focus on the distance-based approach.

In the distance-based approach, the notion of distance measure plays an important role. In the context of PP data, set distances are suitable candidates for distance measure. The choice of set distances in PP learning directly influences performance, hence it is important to select distances that are compatible with the applications. In this work, we propose to use the OSPA distance due to its salient properties for PP learning tasks (Section 6.1.3).

In particular, we apply the AP algorithm with the OSPA distance for the *clustering* task (Section 6.2). The performance comparison shows that the OSPA distance (with a suitable cutoff parameter) outperforms the other set distances, such as the Hausdorff and Wasserstein distances. This is due to the properties of OSPA that can address the undesirable effects of the Hausdorff and Wasserstein distances (Section 6.1.3). In *classification* and *novelty detection*, we use the nearest neighbour approach together with the OSPA distance (Section 6.3 and Section 6.4). We also propose to use training data to learn the OSPA's cutoff parameter using the inter-class and intra-class dissimilarities (for classification, Section 6.3), and the cardinality and feature dissimilarities (for novelty detection, Section 6.4). The performance of OSPA in these tasks also outperforms the Hausdorff and Wasserstein distances, and in some cases, outperforms the model-based methods. However, we stress that there is no single distance that works for all applications. In practice, to determine which distance (and its parameters) is better suited to which application, it is important to assess whether the distance agrees with the notion of similarity/dissimilarity specific to that application.

Compared to the model-based methods, the distance-based methods have several advantages. First, they do not require prior knowledge of the data model, hence they are suitable for a wide range of applications where data models are unknown or too complex to learn. Moreover, learning parameters for set distances (in particular the cutoff of OSPA distance) is far simpler than learning model parameters (such as parameters of the IID-cluster model).

However, we again emphasize that there is no single approach that works well for all applications. For example, the distance-based method can outperform the model-based method in novelty detection for Texture data (see Section 6.4.2 and Section 4.4.2), whereas in clustering for Texture data, the distance-based method performs worse (see Section 6.2.3 and Section 5.5.2). In addition, computing set distances which require the optimal assignment of elements, such as Wasserstein and OSPA distances, can be rather expensive for PPs with high cardinality.

7.2 Future directions

Since PP learning is a developing field of study with a limited number of studies, there are still problems which are unsolved or whose existing solutions can be improved. We suggest some potential research directions for PP learning as follows:

- *Model-based PP clustering with an unknown number of clusters.* Since it is usually the case in practice that the number of clusters are unknown, such a clustering method is worth studying. Although we already have a distance-based method which can address this problem (i.e., AP clustering with set distances shown in Section 6.2), the existing model-based method (proposed in Chapter 5) can only handle the scenario of a known number of clusters. This generalization of the proposed clustering method in Chapter 5 inherits the nice properties of model-based methods, such as the ability to exploit statistical patterns in the data. As previously discussed in Section 5.4.1, a promising approach to this problem can be to use an infinite mixture model with Dirichlet process, which can be computed effectively by a Gibbs sampler.[1]
- *General semi-supervised PP learning.* In semi-supervised learning, which is an area of emerging interest in machine learning (Chapelle et al., 2006), only a part of the training data is labeled. In this work, we have dealt with a special case of this type of learning: the novelty detection problem. As discussed in Section 5.4.1, the Bayesian solution for PP clustering (if proposed) can also be adapted for general semi-supervised learning, by

[1] May see some preliminary result in (Vo et al., 2017a).

computing the posterior of the missing labels. In addition, this approach can also address the novelty detection problem without having to rank the input observations (as the method proposed in Chapter 4), albeit at greater computational cost.

- *More computationally efficient set distances.* As discussed in Chapter 6, the notion of distance plays an important role in distance-based learning approach, since it significantly affects not only the quality of the outcome, but also the computational cost of the learning methods. Of the set distances examined in this work, the OSPA distance has salient properties that can address some of the undesirable effects of the other distances, hence delivering better performance in many applications. However, computing OSPA is rather expensive, especially for PPs with large cardinality, since it requires the optimal assignment of PPs' elements. Hence, in order to apply the proposed distance-based learning method to big data, it is necessary to develop novel set distances which have similar salient properties to those of the OSPA but are more efficient in computation.

BIBLIOGRAPHY

Abou-Assaleh, T., Cercone, N., Keselj, V., and Sweidan, R. (2004). N-gram-based detection of new malicious code. In *Proc. 28th Annual Int. Comput. Softw. Appl. Conf. (COMPSAC)*, volume 2, pages 41–42. IEEE. (Cited on page 11)

Adomavicius, G. and Tuzhilin, A. (2001). Using data mining methods to build customer profiles. *Comput.*, 34(2):74–82. (Cited on page 9)

Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749. (Cited on page 2)

Aeberhard, S., Coomans, D., and de Vel, O. (1994). The performance of statistical pattern recognition methods in high dimensional settings. In *IEEE Signal Process. Workshop Higher Order Statistics, Ceasarea*, volume 4, pages 14–6. (Cited on page 9)

Agichtein, E., Brill, E., and Dumais, S. (2006). Improving web search ranking by incorporating user behavior information. In *Proc. 29th Annual Int. ACM SIGIR Conf. Research and development in information retrieval*, pages 19–26. ACM. (Cited on page 2)

Alpaydin, E. (2004). *Introduction to machine learning*. MIT press. (Cited on pages 2, 25, and 26)

Amores, J. (2013). Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105. (Cited on pages 7, 13, 44, 50, 114, and 127)

Andrews, S., Tsochantaridis, I., and Hofmann, T. (2003). Support vector machines for multiple-instance learning. *Advances in neural information processing systems*, pages 577–584. (Cited on page 50)

Arimond, G. and Elfessi, A. (2001). A clustering method for categorical data in tourism market segmentation research. *J. Travel Research*, 39(4):391–397. (Cited on page 33)

Aucouturier, J.-J., Defreville, B., and Pachet, F. (2007). The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music. *J. Acoust. Soc. Am.*, 122(2):881–891. (Cited on page 12)

Babu, G. J. and Feigelson, E. D. (1996). *Astrostatistics*, volume 3. CRC Press. (Cited on page 7)

Baccelli, F. and Blaszczyszyn, B. (2010). *Stochastic Geometry and Wireless Networks: Volume 1: THEORY*. Now Publishers Inc. (Cited on page 38)

Baddeley, A., Bárány, I., and Schneider, R. (2007). Spatial point processes and their applications. *Stochastic Geometry: Lectures given at the CIME Summer School held in Martina Franca, Italy, September 13–18, 2004*, pages 1–75. (Cited on pages 40, 41, 42, 44, 52, 53, and 54)

Baddeley, A. and Lieshout, M. V. (1993). Stochastic geometry models in high-level vision. *J. Appl. Statistics*, 20(5-6):231–256. (Cited on page 38)

Baddeley, A. and Turner, R. (2000). Practical maximum pseudolikelihood for spatial point patterns. *Australian & New Zealand J. Statistics*, 42(3):283–322. (Cited on page 54)

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*. (Cited on page 4)

Barbieri, R., Matten, E. C., Alabi, A. A., and Brown, E. N. (2005). A point-process model of human heartbeat intervals: new definitions of heart rate and heart rate variability. *American J. Physiology-Heart and Circulatory Physiology*, 288(1):H424–H435. (Cited on page 38)

Battistelli, G., Chisci, L., Fantacci, C., Farina, A., and Graziano, A. (2013). Consensus cphd filter for distributed multitarget tracking. *IEEE J. Selected Topics in Signal Processing*, 7(3):508–520. (Cited on page 39)

Battistelli, G., Chisci, L., Morrocchi, S., Papi, F., Benavoli, A., Di Lallo, A., Farina, A., and Graziano, A. (2008). Traffic intensity estimation via phd filtering. In *European Radar Conf.*, pages 340–343. IEEE. (Cited on page 39)

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical anal. of probabilistic functions of markov chains. *Ann. Mathematical Statistics*, 41(1):164–171. (Cited on page 31)

Baum, M., Balasingam, B., Willett, P., and Hanebeck, U. D. (2015). Ospa barycenters for clustering set-valued data. In *18th Int. Conf. Inf. Fusion (Fusion)*, pages 1375–1381. IEEE. (Cited on page 119)

Beard, M., Reuter, S., Granström, K., Vo, B.-T., Vo, B.-N., and Scheel, A. (2016). Multiple extended target tracking with labeled random finite sets. *IEEE Trans. Signal Processing*, 64(7):1638–1653. (Cited on page 39)

Beard, M., Vo, B.-T., and Vo, B.-N. (2013). Multitarget filtering with unknown clutter density using a bootstrap gm-cphd filter. *IEEE Signal Processing Letters*, 20(4):323–326. (Cited on page 39)

Beard, M., Vo, B.-T., and Vo, B.-N. (2015a). Bayesian multi-target tracking with merged measurements using labelled random finite sets. *IEEE Trans. Signal Processing*, 63(6):1433–1447. (Cited on page 39)

Beard, M., Vo, B.-T., Vo, B.-N., and Arulampalam, S. (2015b). Void probabilities and cauchy-schwarz divergence for generalized labeled multi-bernoulli models. *arXiv preprint arXiv:1510.05532*. (Cited on page 39)

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522. (Cited on page 24)

Besag, J. (1975). Statistical analysis of non-lattice data. *The statistician*, pages 179–195. (Cited on page 54)

Besag, J. (1977). Some methods of statistical analysis for spatial data. *Bulletin of the Int. Statistical Institute*, 47(2):77–92. (Cited on pages 54 and 55)

Bilmes, J. A. (1998). A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *Int. Comput. Sci. Institute*, 4(510). (Cited on page 92)

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press. (Cited on page 26)

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer. (Cited on pages 5, 17, 24, 25, 26, 27, 28, 49, and 58)

Bolton, R. J. and Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Sci.*, pages 235–249. (Cited on pages 4 and 29)

Botterill, T., Mills, S., and Green, R. (2008). Speeded-up bag-of-words algorithm for robot localisation through scene recognition. In *2008 23rd Int. Conf. Image and Vision Comput. New Zealand*, pages 1–6. IEEE. (Cited on page 11)

Bowsher, C. G. (2007). Modelling security market events in continuous time: Intensity based, multivariate point process models. *J. Econometrics*, 141(2):876–912. (Cited on page 38)

Boyan, J., Freitag, D., and Joachims, T. (1996). A machine learning architecture for optimizing web search engines. In *AAAI Workshop on Internet Based Information Systems*, pages 1–8. (Cited on page 2)

Brijs, T., Swinnen, G., Vanhoof, K., and Wets, G. (1999). Using association rules for product assortment decisions: A case study. In *Proc. 5th ACM SIGKDD Int. Conf. knowledge discovery and data mining*, pages 254–260. ACM. (Cited on page 9)

Bunescu, R. C. and Mooney, R. J. (2007). Multiple instance learning for sparse positive bags. In *Proc. 24th Int. Conf. Machine learning*, pages 105–112. ACM. (Cited on page 50)

Burbidge, R., Trotter, M., Buxton, B., and Holden, S. (2001). Drug design by machine learning: support vector machines for pharmaceutical data analysis. *Computers & chemistry*, 26(1):5–14. (Cited on page 4)

Byers, S. D. and Raftery, A. E. (2002). Bayesian estimation and segmentation of spatial point processes using voronoi tilings. (Cited on page 56)

Cadez, I. V., Gaffney, S., and Smyth, P. (2000). A general probabilistic framework for clustering individuals and objects. In *Proc. 6th ACM SIGKDD Int. Conf. knowledge discovery and data mining*, pages 140–149. (Cited on pages 9, 17, and 61)

Carvalho, C. M., Chang, J., Lucas, J. E., Nevins, J. R., Wang, Q., and West, M. (2012). High-dimensional sparse factor modeling: Appl. in gene expression genomics. *J. the American Statistical Association*. (Cited on page 13)

Cavnar, W. B., Trenkle, J. M., et al. (1994). N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175. (Cited on pages 9 and 11)

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput. Surveys (CSUR)*, 41(3):15. (Cited on pages 28, 30, and 70)

Chapelle, O., Scholkopf, B., and Zien, A. (2006). *Semi-supervised learning*. MIT Press. (Cited on pages 6, 29, and 140)

Chechik, G., Ie, E., Rehn, M., Bengio, S., and Lyon, D. (2008). Large-scale content-based audio retrieval from text queries. In *Proc. 1st ACM Int. Conf. Multimedia Information Retrieval*, pages 105–112. ACM. (Cited on page 12)

Chen, D., Sain, S. L., and Guo, K. (2012). Data mining for the online retail industry: A case study of rfm model-based customer segmentation using data mining. *J. Database Marketing & Customer Strategy Management*, 19(3):197–208. (Cited on page 9)

Cheng, C.-H. and Chen, Y.-S. (2009). Classifying the segmentation of customer value via rfm model and rs theory. *Expert systems with applications*, 36(3):4176–4184. (Cited on page 4)

Cheplygina, V., Tax, D. M., and Loog, M. (2015). On classification with bags, groups and sets. *Pattern Recognition Letters*, 59:11–17. (Cited on pages 13, 14, 15, and 50)

Chickering, D. M. and Heckerman, D. (1999). Fast learning from sparse data. In *Proc. 15th Conf. Uncertainty in artificial intelligence*, pages 109–115. Morgan Kaufmann Publishers Inc. (Cited on page 12)

Cignoni, P., Rocchini, C., and Scopigno, R. (1998). Metro: measuring error on simplified surfaces. In *Comput. Graphics Forum*, volume 17, pages 167–174. Wiley Online Library. (Cited on page 114)

City-of-Hobart (2016). Significant tree point. https://www.data.gov.au/dataset/significant-tree-point. Accessed November 2016. (Cited on page 7)

Collins, S. R., Miller, K. M., Maas, N. L., Roguev, A., Fillingham, J., Chu, C. S., Schuldiner, M., Gebbia, M., Recht, J., Shales, M., et al. (2007). Functional dissection of protein complexes involved in yeast chromosome biology using a genetic interaction map. *Nature*, 446(7137):806–810. (Cited on page 33)

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537. (Cited on page 24)

Coon, D. and Mitterer, J. O. (2007). *Introduction to psychology: Gateways to mind and behavior*. Thomson Wadsworth. (Cited on page 2)

Cormack, G. V., Gómez Hidalgo, J. M., and Sánz, E. P. (2007). Spam filtering for short messages. In *Proc. sixteenth ACM Conf. Conf. Inf. and Knowl. management*, pages 313–320. ACM. (Cited on page 11)

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory*, 13(1):21–27. (Cited on pages 20, 27, and 127)

Cressie, N. and Wikle, C. K. (2011). *Statistics for spatio-temporal data*. John Wiley & Sons. (Cited on page 7)

Croft, W. B., Metzler, D., and Strohman, T. (2010). *Search engines: Information retrieval in practice*, volume 283. Addison-Wesley Reading. (Cited on page 24)

Cruz, J. A. and Wishart, D. S. (2006). Applications of machine learning in cancer prediction and prognosis. *Cancer informatics*, 2. (Cited on page 4)

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop statistical learning in computer vision, ECCV*. (Cited on pages 7, 11, 17, and 61)

Daley, D. J. and Vere-Jones, D. (1988). *An introduction to the theory of point processes*, volume 2. Springer. (Cited on pages 19, 37, 38, and 43)

Daley, D. J. and Vere-Jones, D. (2003). *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*. Springer. (Cited on page 13)

Dasgupta, D. and Majumdar, N. S. (2002). Anomaly detection in multidimensional data using negative selection algorithm. In *Proc. 2002 Congress on Evolutionary Computation (CEC)*, volume 2, pages 1039–1044. IEEE. (Cited on page 30)

de Amorim, R. C. and Hennig, C. (2015). Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information Sciences*, 324:126–145. (Cited on page 101)

Delany, S. J., Cunningham, P., and Coyle, L. (2005). An assessment of case-based reasoning for spam filtering. *Artificial Intelligence Review*, 24(3-4):359–378. (Cited on page 11)

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *J. Royal Statistical Soc. Series B (Methodological)*, pages 1–38. (Cited on pages 31, 35, 58, 91, and 92)

Deusch, H., Reuter, S., and Dietmayer, K. (2015). The labeled multi-bernoulli slam filter. *IEEE Signal Processing Letters*, 22(10):1561–1565. (Cited on page 39)

Dhillon, I. S. and Modha, D. S. (2001). Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1-2):143–175. (Cited on page 33)

Diehl, C. P. and Hampshire, J. B. (2002). Real-time object classification and novelty detection for collaborative video surveillance. In *Proc. 2002 Int. Joint Conf. Neural Networks (IJCNN)*, volume 3, pages 2620–2625. IEEE. (Cited on page 29)

Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1):31–71. (Cited on pages 7, 13, 15, 24, and 50)

Diggle, P. J. (1983). *Statistical analysis of spatial point patterns*. Academic Press. (Cited on page 7)

Diggle, P. J. (2014). *Statistical Analysis of spatial and spatio-temporal point patterns*. CRC Press. (Cited on pages 7 and 13)

Dong, L. (2006). *A comparison of multi-instance learning algorithms*. PhD thesis, The University of Waikato. (Cited on page 50)

Dueck, D. and Frey, B. J. (2007). Non-metric affinity propagation for unsupervised image categorization. In *11th Int. Conf. Comput. Vision (ICCV)*, pages 1–8. IEEE. (Cited on pages 35, 120, and 122)

Eagle, N. and Pentland, A. S. (2006). Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268. (Cited on page 108)

Ellinger-Ziegelbauer, H., Gmuender, H., Bandenburg, A., and Ahr, H. J. (2008). Prediction of a carcinogenic potential of rat hepatocarcinogens using toxicogenomics analysis of short-term in vivo studies. *Mutation Research/Fundamental and Molecular Mechanisms of Mutagenesis*, 637(1):23–39. (Cited on page 24)

Elliot, P., Wakefield, J. C., Best, N. G., Briggs, D., et al. (2000). *Spatial epidemiology: methods and applications.* Oxford Univ. Press. (Cited on page 7)

Engle, R. F. and Russell, J. R. (1998). Autoregressive conditional duration: a new model for irregularly spaced transaction data. *Econometrica*, pages 1127–1162. (Cited on page 38)

Erman, J., Arlitt, M., and Mahanti, A. (2006). Traffic classification using clustering algorithms. In *Proc. 2006 SIGCOMM workshop on Mining network data*, pages 281–286. ACM. (Cited on page 33)

Esponda, F., Forrest, S., and Helman, P. (2004). A formal framework for positive and negative detection schemes. *IEEE Trans. Syst., Man, Cybern., Part B (Cybernetics)*, 34(1):357–373. (Cited on page 30)

Estivill-Castro, V. (2002). Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75. (Cited on page 33)

Fantacci, C., Vo, B.-N., Vo, B.-T., Battistelli, G., and Chisci, L. (2015). Consensus labeled random finite set filtering for distributed multi-object tracking. *arXiv preprint arXiv:1501.01579*. (Cited on page 39)

Färber, I., Günnemann, S., Kriegel, H.-P., Kröger, P., Müller, E., Schubert, E., Seidl, T., and Zimek, A. (2010). On using class-labels in evaluation of clusterings. In *MultiClust: 1st international workshop on discovering, summarizing and using multiple clusterings held in conjunction with KDD*. (Cited on page 35)

Fei-Fei, L. and Perona, P. (2005). A Bayesian hierarchical model for learning natural scene categories. In *IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognition (CVPR), 2005*, volume 2, pages 524–531. IEEE. (Cited on page 11)

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., et al. (2010). Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79. (Cited on page 4)

Fiksel, T. (1988). Estimation of interaction potentials of gibbsian point processes. *Statistics*, 19(1):77–86. (Cited on page 54)

Filannino, M. (2011). Dbworld e-mails data set. http://archive.ics.uci.edu/ml/datasets/DBWorld+e-mails. Accessed October 2016. (Cited on page 11)

Forrest, S., Perelson, A. S., Allen, L., and Cherukuri, R. (1994). Self-nonself discrimination in a computer. In *Proc. IEEE Computer Society Symposium on Research in Security and Privacy*, pages 202–212. (Cited on page 30)

Foulds, J. and Frank, E. (2010). A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(01):1–25. (Cited on pages 7, 13, 14, and 15)

Fraley, C. and Raftery, A. E. (2007). Bayesian regularization for normal mixture estimation and model-based clustering. *J. classification*, 24(2):155–181. (Cited on page 101)

Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *Sci.*, 315(5814):972–976. (Cited on pages 21, 35, 119, 120, 121, 122, and 123)

Fürnkranz, J. (1998). A study using n-gram features for text categorization. *Austrian Research Institute for Artifical Intelligence*, 3(1998):1–10. (Cited on page 9)

Garstka, J. and Peters, G. (2015). Fast and robust keypoint detection in unstructured 3-d point clouds. In *Informatics in Control, Automation and Robotics (ICINCO), 2015 12th Int. Conf.*, volume 2, pages 131–140. IEEE. (Cited on page 9)

Gärtner, T., Flach, P. A., Kowalczyk, A., and Smola, A. J. (2002). Multi-instance kernels. In *ICML*, volume 2, pages 179–186. (Cited on page 50)

Gavrila, D. M. and Philomin, V. (1999). Real-time object detection for "smart" vehicles.

In *Proc. 7th Int. Conf. Comput. Vision, 1999*, volume 1, pages 87–93. (Cited on pages 20, 50, 115, and 127)

Gebru, I. D., Alameda-Pineda, X., Forbes, F., and Horaud, R. (2016). Em algorithms for weighted-data clustering with application to audio-visual scene analysis. *IEEE transactions on pattern analysis and machine intelligence*, 38(12):2402–2415. (Cited on page 94)

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Comput. Vision and Pattern Recognition (CVPR), 2012 IEEE Conf.*, pages 3354–3361. IEEE. (Cited on page 9)

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian data analysis*, volume 2. Chapman & Hall/CRC. (Cited on page 56)

Geyer, C. J. et al. (1999). Likelihood inference for spatial point processes. *Stochastic geometry: likelihood and computation*, 80:79–140. (Cited on page 41)

Geyer, C. J. and Møller, J. (1994). Simulation procedures and likelihood inference for spatial point processes. *Scandinavian J. statistics*, pages 359–373. (Cited on pages 53 and 56)

Givoni, I. E. and Frey, B. J. (2009). A binary variable model for affinity propagation. *Neural computation*, 21(6):1589–1600. (Cited on page 121)

González, F. A. and Dasgupta, D. (2003). Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4(4):383–403. (Cited on page 30)

Google (2016). Google books ngrams. https://aws.amazon.com/datasets/google-books-ngrams. Accessed October 2016. (Cited on page 11)

Google (2017). How search works. https://www.google.com.au/insidesearch/howsearchworks/thestory. Accessed January 2017. (Cited on page 2)

Gostar, A. K., Hoseinnezhad, R., and Bab-Hadiashar, A. (2013). Robust multi-bernoulli sensor selection for multi-target tracking in sensor networks. *IEEE Signal Processing Letters*, 20(12):1167–1170. (Cited on page 39)

Gostar, A. K., Hoseinnezhad, R., and Bab-Hadiashar, A. (2014). Sensor control for multi-object tracking using labeled multi-bernoulli filter. In *17th Int. Conf. Information Fusion (FUSION)*, pages 1–8. IEEE. (Cited on page 39)

Gostar, A. K., Hoseinnezhad, R., and Bab-Hadiashar, A. (2015). Multi-bernoulli sensor control via minimization of expected estimation errors. *IEEE Trans. Aerospace and Electronic Systems*, 51(3):1762–1773. (Cited on page 39)

Gostar, A. K., Hoseinnezhad, R., and Bab-Hadiashar, A. (2016). Multi-bernoulli sensor-selection for multi-target tracking with unknown clutter and detection profiles. *Signal Processing*, 119:28–42. (Cited on page 39)

Graepel, T., Candela, J. Q., Borchert, T., and Herbrich, R. (2010). Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *Proc. 27th Int. Conf. machine learning (ICML-10)*, pages 13–20. (Cited on page 4)

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proc. 23rd Int. Conf. Machine learning*, pages 369–376. ACM. (Cited on page 24)

Green, P. J. (1995). Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, pages 711–732. (Cited on page 102)

Gross, R. (2014). *Psychology: The Science of Mind and Behaviour - 6th Edition*. Hodder Education. (Cited on page 2)

Guan, R., Shi, X., Marchese, M., Yang, C., and Liang, Y. (2011). Text clustering with seeds affinity propagation. *IEEE Trans. Knowl. Data Eng.*, 23(4):627–637. (Cited on page 120)

Guha, S., Rastogi, R., and Shim, K. (1999). Rock: A robust clustering algorithm for categorical attributes. In *Proc. 15th Int. Conf. Data Eng., 1999*, pages 512–521. IEEE. (Cited on page 9)

Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. A. (2008). *Feature extraction: foundations and applications*, volume 207. Springer. (Cited on page 6)

Guzella, T. S. and Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7):10206–10222. (Cited on page 4)

Han, J., Pei, J., and Kamber, M. (2012). *Data mining: concepts and techniques*. Elsevier. (Cited on pages 24, 25, 98, and 101)

Hautamäki, V., Kärkkäinen, I., and Fränti, P. (2004). Outlier detection using k-nearest neighbour graph. In *ICPR*, pages 430–433. (Cited on pages 31 and 131)

Hill, T. and Lewicki, P. (2006). *Statistics: methods and applications: a comprehensive reference for science, industry, and data mining*. StatSoft, Inc. (Cited on pages 68 and 99)

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Magazine*, 29(6):82–97. (Cited on page 4)

Ho, Y. and Lee, R. (1964). A bayesian approach to problems in stochastic estimation and control. *IEEE Trans. Automatic Control*, 9(4):333–339. (Cited on page 78)

Hoang, H. G., Vo, B.-N., Vo, B.-T., and Mahler, R. (2015). The cauchy–schwarz divergence for poisson point processes. *IEEE Trans. Information Theory*, 61(8):4475–4485. (Cited on page 39)

Hoang, H. G. and Vo, B. T. (2014). Sensor management for multi-target tracking via multi-bernoulli filtering. *Automatica*, 50(4):1135–1142. (Cited on page 39)

Hodge, V. J. and Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126. (Cited on pages 28, 29, 30, and 70)

Hoffman, J. R. and Mahler, R. P. (2004). Multitarget miss distance via optimal assignment. *IEEE Trans. Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(3):327–336. (Cited on pages 45 and 46)

Hoseinnezhad, R., Vo, B.-N., and Vo, B.-T. (2013). Visual tracking in background subtracted image sequences via multi-bernoulli filtering. *IEEE Trans. Signal Processing*, 61(2):392–397. (Cited on page 39)

Hoseinnezhad, R., Vo, B.-N., Vo, B.-T., and Suter, D. (2012). Visual tracking of numerous targets via multi-bernoulli filtering of image data. *Pattern Recognition*, 45(10):3625–3635. (Cited on page 39)

Huttenlocher, D. P., Klanderman, G. A., and Rucklidge, W. J. (1993a). Comparing images using the hausdorff distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(9):850–863. (Cited on pages 45 and 114)

Huttenlocher, D. P., Noh, J. J., and Rucklidge, W. J. (1993b). Tracking non-rigid objects in complex scenes. In *Proc. 4th Int. Conf. Comput. Vision, 1993*, pages 93–101. IEEE. (Cited on pages 20, 45, 50, and 127)

Ioannou, Y., Taati, B., Harrap, R., and Greenspan, M. (2012). Difference of normals as a multi-scale operator in unorganized point clouds. In *2012 2nd Int. Conf. 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 501–508. IEEE. (Cited on page 7)

Ionescu, R. T., Popescu, M., and Grozea, C. (2013). Local learning to improve bag of visual words model for facial expression recognition. In *Workshop challenges in representation learning, ICML*. (Cited on page 12)

ISPRS (2006). Isprs test on extracting dems from point clouds. http://www.itc.nl/isprswgIII-3/filtertest. Accessed October 2016. (Cited on page 7)

Iváncsy, R. and Vajk, I. (2006). Frequent pattern mining in web log data. *Acta Polytechnica Hungarica*, 3(1):77–90. (Cited on page 9)

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666. (Cited on pages 35, 44, 98, 114, and 120)

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Comput. surveys (CSUR)*, 31(3):264–323. (Cited on pages 33, 34, and 90)

Javed, O. and Shah, M. (2002). Tracking and object classification for automated surveillance. In *Comput. Vision – ECCV 2002*, pages 343–357. Springer. (Cited on page 24)

Jensen, J. L. and Møller, J. (1991). Pseudolikelihood for exponential family models of spatial point processes. *The Annals of Applied Probability*, pages 445–461. (Cited on page 55)

Jiang, M., Yi, W., Hoseinnezhad, R., and Kong, L. (2016). Adaptive vo-vo filter for maneuvering targets with time-varying dynamics. In *19th Int. Conf. Information Fusion (FUSION)*, pages 666–672. IEEE. (Cited on page 39)

Jin, Q., Schulam, P. F., Rawat, S., Burger, S., Ding, D., and Metze, F. (2012). Event-based video retrieval using audio. In *Proc. of INTERSPEECH*, page 2085. (Cited on page 12)

Jing, L., Ng, M. K., and Huang, J. Z. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Trans. Knowl. Data Eng.*, 19(8):1026–1041. (Cited on pages 12 and 13)

Joachims, T. (1996). A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, DTIC Document. (Cited on pages 6, 9, 11, 12, 17, and 61)

Jordan, M. (2010). The conjugate prior for the normal distribution. *Lecture notes on Stat260: Bayesian Modeling and Inference*. (Cited on page 61)

Jyothsna, V., Prasad, V. R., and Prasad, K. M. (2011). A review of anomaly based intrusion detection systems. *Int. J. Computer Applications*, 28(7):26–35. (Cited on page 29)

Kaufman, L. and Rousseeuw, P. (1987). *Clustering by means of medoids*. Elsevier/North-Holland. (Cited on page 35)

Kaufman, L. and Rousseeuw, P. J. (1990). *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons. (Cited on pages 34, 35, 98, and 101)

Keller, J. M., Gray, M. R., and Givens, J. A. (1985). A fuzzy k-nearest neighbor algorithm. *IEEE Trans. Systems, Man and Cybernetics*, (4):580–585. (Cited on pages 27 and 127)

Kešelj, V., Peng, F., Cercone, N., and Thomas, C. (2003). N-gram-based author profiles for authorship attribution. In *Proc. Conf. pacific association for computational linguistics, PACLING*, volume 3, pages 255–264. (Cited on page 11)

Kim, D. Y., Vo, B.-N., and Vo, B.-T. (2016). Online visual multi-object tracking via labeled random finite set filtering. *arXiv preprint arXiv:1611.06011*. (Cited on page 39)

Kim, D. Y., Vo, B.-T., and Vo, B.-N. (2013). Data fusion in 3d vision using a rgb-d data via switching observation model and its application to people tracking. In *Int. Conf. Control, Automation and Information Sciences (ICCAIS)*, pages 91–96. IEEE. (Cited on page 39)

Knuth, D. (1998). *The Art of Computer Programming. Vol. 2 Seminumerical Algorithms*. Addison Wesley. (Cited on page 7)

Kononenko, I. (2001). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109. (Cited on page 4)

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97. (Cited on page 47)

Kuhn, H. W. (1956). Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3(4):253–258. (Cited on page 47)

Kuo, R. J., Chen, C., and Hwang, Y. (2001). An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. *Fuzzy sets and systems*, 118(1):21–45. (Cited on page 4)

Larlus, D., Verbeek, J., and Jurie, F. (2010). Category level object segmentation by combining bag-of-words models with dirichlet processes and random fields. *Int. J. Comput. Vision*, 88(2):238–253. (Cited on page 12)

Lavoué, G. (2011). Bag of words and local spectral descriptor for 3d partial shape retrieval. In *Eurographics Workshop 3D object retrieval*. Citeseer. (Cited on page 11)

Lazebnik, S., Schmid, C., and Ponce, J. (2005). A sparse texture representation using local affine regions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1265–1278. (Cited on pages 61 and 65)

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551. (Cited on page 4)

Lee, K., Caverlee, J., and Webb, S. (2010). Uncovering social spammers: social honeypots+ machine learning. In *Proc. 33rd Int. ACM SIGIR Conf. Research and development* *in information retrieval*, pages 435–442. ACM. (Cited on page 4)

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *Machine learning: ECML-98*, pages 4–15. Springer. (Cited on page 24)

Li, T., Mei, T., Kweon, I.-S., and Hua, X.-S. (2011). Contextual bag-of-words for visual categorization. *IEEE Trans. Circuits and Systems for Video Technology*, 21(4):381–392. (Cited on page 11)

Li, W.-J., Wang, K., Stolfo, S. J., and Herzog, B. (2005). Fileprints: Identifying file types by n-gram analysis. In *Proc. 6th Annual IEEE SMC Inf. Assurance Workshop*, pages 64–71. IEEE. (Cited on page 11)

Li, Z., Imai, J.-i., and Kaneko, M. (2009). Facial-component-based bag of words and phog descriptor for facial expression recognition. In *SMC*, pages 1353–1358. (Cited on page 12)

Lin, C.-Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram cooccurrence statistics. In *Proc. 2003 Conf. of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics. (Cited on page 11)

Linsen, L. (2001). *Point cloud representation*. Univ., Fak. für Informatik, Bibliothek. (Cited on page 7)

Little, R. J. and Rubin, D. B. (2002). *Statistical analysis with missing data*. John Wiley & Sons. (Cited on pages 58, 91, and 92)

Liu, D.-R. and Shih, Y.-Y. (2005). Hybrid approaches to product recommendation based on customer lifetime value and purchase preferences. *J. Systems and Softw.*, 77(2):181–191. (Cited on page 9)

Liu, Y., Zhao, W.-L., Ngo, C.-W., Xu, C.-S., and Lu, H.-Q. (2010). Coherent bag-of audio words model for efficient large-scale video copy detection. In *Proc. ACM Int. Conf. Image and Video Retrieval*, pages 89–96. ACM. (Cited on page 12)

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. vision*, 60(2):91–110. (Cited on page 11)

Lund, O., Nielsen, M., Kesmir, C., Petersen, A. G., Lundegaard, C., Worning, P., Sylvester-Hvid, C., Lamberth, K., Røder, G., Justesen, S., et al. (2004). Definition of supertypes for hla molecules using clustering of specificity matrices. *Immunogenetics*, 55(12):797–810. (Cited on page 33)

Lundquist, C., Hammarstrand, L., and Gustafsson, F. (2011). Road intensity based mapping using radar measurements with a probability hypothesis density filter. *IEEE Trans. Signal Processing*, 59(4):1397–1408. (Cited on page 39)

Ma, L., Smith, D., and Milner, B. P. (2003). Context awareness using environmental noise classification. In *INTERSPEECH*. (Cited on page 12)

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. (Cited on page 34)

Maggio, E., Taj, M., and Cavallaro, A. (2008). Efficient multitarget visual tracking using random finite sets. *IEEE Trans. Circuits and Systems for Video Technology*, 18(8):1016–

1027. (Cited on page 39)

Mahler, R. (2003a). Multi-target Bayes filtering via first-order multi-target moments. *IEEE Trans. Aerospace & Electronic Systems*, 39(4):1152–1178. (Cited on page 41)

Mahler, R. (2007a). Phd filters of higher order in target number. *IEEE Trans. Aerosp. Electron. Syst.*, 43(4). (Cited on page 38)

Mahler, R. and Vo, B.-T. (2014). An improved cphd filter for unknown clutter backgrounds. In *SPIE Defense + Security*, pages 90910B–90910B. International Society for Optics and Photonics. (Cited on page 39)

Mahler, R. P. (2003b). Multisensor-multitarget sensor management: A unified bayesian approach. In *AeroSense 2003*, pages 222–233. Int. Society for Optics and Photonics. (Cited on page 39)

Mahler, R. P. (2003c). Multitarget Bayes filtering via first-order multitarget moments. *IEEE Trans. Aerosp. Electron. Syst*, 39(4):1152–1178. (Cited on page 38)

Mahler, R. P. (2007b). *Statistical multisource-multitarget information fusion*. Artech House, Inc. (Cited on pages 41 and 76)

Mahler, R. P. (2014). *Advances in statistical multisource-multitarget information fusion*. Artech House, Inc. (Cited on page 38)

Mahler, R. P., Vo, B.-T., and Vo, B.-N. (2011). Cphd filtering with unknown clutter rate and detection profile. *IEEE Trans. on Signal Processing*, 59(8):3497–3513. (Cited on page 39)

Mandel, M. I. and Ellis, D. (2005). Song-level features and support vector machines for music classification. In *ISMIR*, volume 2005, pages 594–599. (Cited on page 12)

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*, volume 1. Cambridge univ. press Cambridge. (Cited on pages 4, 28, 32, 35, 36, and 37)

Markou, M. and Singh, S. (2003). Novelty detection: a review – part 1: statistical approaches. *Signal Process.*, 83(12):2481–2497. (Cited on pages 5, 16, 17, 28, 31, 70, and 138)

Markou, M. and Singh, S. (2006). A neural network-based novelty detector for image sequence analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1664–1677. (Cited on page 29)

Maron, M. E. (1961). Automatic indexing: an experimental inquiry. *JACM*, 8(3):404–417. (Cited on pages 9, 17, and 61)

Maron, O. and Lozano-Pérez, T. (1998). A framework for multiple-instance learning. *Advances in neural information processing systems*, pages 570–576. (Cited on page 50)

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451. (Cited on page 33)

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive Bayes text classification. In *AAAI-98 Workshop learning for text categorization*, volume 752, pages 41–48. (Cited on pages 9, 11, 17, and 61)

Meissner, D., Reuter, S., and Dietmayer, K. (2013). Road user tracking at intersections using a multiple-model phd filter. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 377–382. IEEE. (Cited on page 39)

Mesterharm, C. and Pazzani, M. J. (2011). Farm ads data set. http://archive.ics.uci.edu/ml/datasets/Farm+Ads. Accessed October 2016. (Cited on page 11)

Mian, A., Bennamoun, M., and Owens, R. (2010). On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. *Int. J. Comput. Vision*, 89(2-3):348–361. (Cited on page 9)

Minka, T. (2003). Bayesian inference, entropy, and the multinomial distribution. *Online tutorial*. (Cited on page 60)

Mishra, N., Schreiber, R., Stanton, I., and Tarjan, R. E. (2007). Clustering social networks. In *Algorithms and Models for the Web-Graph*, pages 56–67. Springer. (Cited on page 33)

Mitchell, T. M. (1997). *Machine learning*. McGraw Hill. (Cited on page 2)

Moller, J. and Waagepetersen, R. P. (2003). *Statistical inference and simulation for spatial point processes*. CRC Press. (Cited on pages 6, 7, 13, 19, 37, 41, 42, 44, 52, 53, and 56)

Mullane, J., Vo, B.-N., Adams, M. D., and Vo, B.-T. (2011). A random-finite-set approach to bayesian slam. *IEEE TranS. Robotics*, 27(2):268–282. (Cited on pages 38 and 39)

Murata, T. and Saito, K. (2006). Extracting users' interests from web log data. In *Proc. 2006 IEEE/WIC/ACM Int. Conf. Web Intell.*, pages 343–346. IEEE Comput. Soc. (Cited on page 9)

Murphy, K. P. (2007). Conjugate bayesian analysis of the gaussian distribution. *def*. (Cited on page 61)

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press. (Cited on pages 2, 5, 24, 25, 27, 33, 34, 49, 51, 55, 56, 89, 90, 101, and 102)

Nanopoulos, A. and Manolopoulos, Y. (2000). Finding generalized path patterns for web log data mining. In *Current Issues in Databases and Inf. Systems*, pages 215–228. Springer. (Cited on page 9)

Neto, H. V. and Nehmzow, U. (2007). Real-time automated visual inspection using mobile robots. *J. Intelligent and Robotic Systems*, 49(3):293–307. (Cited on page 29)

Ngai, E. W., Xiu, L., and Chau, D. C. (2009). Application of data mining techniques in customer relationship management: A literature review and classification. *Expert systems with applications*, 36(2):2592–2602. (Cited on page 4)

Ngo, C.-W., Jiang, Y.-G., Wei, X.-Y., Zhao, W., Liu, Y., Wang, J., Zhu, S., and Chang, S.-F. (2009). Vireo/dvmm at trecvid 2009: High-level feature extraction, automatic video search, and content-based copy detection. *Proc. TRECVID2009*, pages 415–432. (Cited on page 12)

Nguyen, H.-L., Woon, Y.-K., and Ng, W.-K. (2015). A survey on data stream clustering and classification. *Knowl. and Inf. systems*, 45(3):535–569. (Cited on page 138)

Nguyen, T., Nguyen, V., Salim, F. D., and Phung, D. (2016). Secc: Simultaneous extraction of context and community from pervasive signals. In *IEEE Int. Conf. Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE. (Cited on page 109)

Nguyen, T., Phung, D., Gupta, S., and Venkatesh, S. (2013). Extraction of latent patterns and contexts from social honest signals using hierarchical Dirichlet processes. In *IEEE Int. Conf. Pervasive Computing and Communications (PerCom)*, pages 47–55. IEEE. (Cited on pages 109 and 120)

Nguyen, T.-B., Nguyen, T., Luo, W., Venkatesh, S., and Phung, D. (2014). Unsupervised inference of significant locations from wifi data for understanding human dynamics. In *Proc. 13th Int. Conf. Mobile and Ubiquitous Multimedia*, pages 232–235. ACM. (Cited on page 120)

Nguyen, V., Martinelli, A., Tomatis, N., and Siegwart, R. (2005). A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics. In *IEEE/RSJ Int. Conf. Intell. Robots and Systems, 2005 (IROS 2005)*, pages 1929–1934. IEEE. (Cited on page 33)

Nüchter, A. and Lingemann, K. (2016). Robotic 3d scan repository. http://kos.informatik.uni-osnabrueck.de/3Dscans. Accessed October 2016. (Cited on page 7)

Nwe, T. L., Foo, S. W., and De Silva, L. C. (2003). Speech emotion recognition using hidden markov models. *Speech communication*, 41(4):603–623. (Cited on page 24)

Ogata, Y. and Tanemura, M. (1984). Likelihood analysis of spatial point patterns. *J. Royal Statistical Society. Series B (Methodological)*, pages 496–518. (Cited on page 53)

Ohser, J. and Mücklich, F. (2000). *Statistical analysis of microstructures in materials science*. Wiley. (Cited on page 7)

Pancoast, S. and Akbacak, M. (2012). Bag-of-audio-words approach for multimedia event classification. In *Interspeech*, pages 2105–2108. (Cited on page 12)

Pancoast, S. and Akbacak, M. (2013). N-gram extension for bag-of-audio-words. In *2013 IEEE Int. Conf. Acoustics, Speech and Signal Process.*, pages 778–782. IEEE. (Cited on page 12)

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proc. ACL-02 Conf. Empirical methods in natural language Process.-Volume 10*, pages 79–86. Association for Computational Linguistics. (Cited on page 24)

Papi, F. and Kim, D. Y. (2015). A particle multi-target tracker for superpositional measurements using labeled random finite sets. *IEEE Trans. Signal Processing*, 63(16):4348–4358. (Cited on page 39)

Papi, F., Vo, B.-N., Vo, B.-T., Fantacci, C., and Beard, M. (2015). Generalized labeled multi-bernoulli approximation of multi-object densities. *IEEE Trans. Signal Processing*, 63(20):5487–5497. (Cited on page 39)

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proc. 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics. (Cited on page 11)

Pavlov, D. Y. and Pennock, D. M. (2002). A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *Advances in neural Inf. Process. systems*, pages 1441–1448. (Cited on page 13)

Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The* *adaptive web*, pages 325–341. Springer. (Cited on page 2)

Perrin, G., Descombes, X., and Zerubia, J. (2005). A marked point process model for tree crown extraction in plantations. In *IEEE Int. Conf. Image Processing*, volume 1, pages I – 661–4. IEEE. (Cited on page 38)

Petersen, K. B., Pedersen, M. S., et al. (2012). The matrix cookbook. *Technical University of Denmark*. (Cited on page 96)

Pham, D. L., Xu, C., and Prince, J. L. (2000). Current methods in medical image segmentation 1. *Annual review of biomedical engineering*, 2(1):315–337. (Cited on page 24)

Pimentel, M. A., Clifton, D. A., Clifton, L., and Tarassenko, L. (2014). A review of novelty detection. *Signal Process.*, 99:215–249. (Cited on pages 28, 30, 31, 44, 70, 114, and 131)

Plinge, A., Grzeszick, R., and Fink, G. A. (2014). A bag-of-features approach to acoustic event detection. In *2014 IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, pages 3704–3708. IEEE. (Cited on page 12)

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes: The art of scientific Comput. (3rd edition*. Cambridge Univ. Press. (Cited on pages 28 and 33)

Punchihewa, Y., Papi, F., and Hoseinnezhad, R. (2014). Multiple target tracking in video data using labeled random finite set. In *Int. Conf. Control, Automation and Information Sciences (ICCAIS)*, pages 13–18. IEEE. (Cited on page 39)

Punchihewa, Y., Vo, B.-N., and Vo, B.-T. (2016). A generalized labeled multi-bernoulli filter for maneuvering targets. In *19th Int. Conf. Information Fusion (FUSION)*, pages 980–986. IEEE. (Cited on page 39)

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier. (Cited on page 25)

Quinn, J. A. and Williams, C. K. (2007). Known unknowns: Novelty detection in condition monitoring. In *Iberian Conf. Pattern Recognition and Image Analysis*, pages 1–6. Springer. (Cited on page 29)

Rabiner, L. R. and Juang, B.-H. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, 3(1):4–16. (Cited on page 31)

Rathnayake, T., Gostar, A. K., Hoseinnezhad, R., and Bab-Hadiashar, A. (2015). Labeled multi-bernoulli track-before-detect for multi-target tracking in video. In *18th Int. Conf. Information Fusion (Fusion)*, pages 1353–1358. IEEE. (Cited on page 39)

Rezatofighi, S. H., Gould, S., Vo, B. T., Vo, B.-N., Mele, K., and Hartley, R. (2015). Multi-target tracking with time-varying clutter rate and detection profile: Application to time-lapse cell microscopy sequences. *IEEE trans. medical imaging*, 34(6):1336–1348. (Cited on page 39)

Rifkin, R. and Klautau, A. (2004). In defense of one-vs-all classification. *J. Machine Learning Research*, 5:101–141. (Cited on pages 26 and 50)

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge university press. (Cited on page 26)

Ristic, B. and Vo, B.-N. (2010). Sensor control for multi-object state-space estimation using random finite sets. *Automatica*, 46(11):1812–1818. (Cited on page 39)

Ristic, B., Vo, B.-N., and Clark, D. (2011). A note on the reward function for phd filters with sensor control. *IEEE Trans. Aerospace and Electronic Systems*, 47(2):1521–1529. (Cited on page 39)

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. computational and applied mathematics*, 20:53–65. (Cited on pages 68, 99, 100, and 101)

Rubner, Y., Tomasi, C., and Guibas, L. J. (1998). A metric for distributions with applications to image databases. In *6th IEEE Int. Conf. Comput. Vision*, pages 59–66. (Cited on pages 46, 50, and 127)

Ruch, P., Baud, R., and Geissbühler, A. (2002). Evaluating and reducing the effect of data corruption when applying bag of words approaches to medical records. *Int. J. Medical Informatics*, 67(1):75–83. (Cited on page 11)

Rucklidge, W. J. (1995). Locating objects using the hausdorff distance. In *Proc. 5th Int. Conf. Comput. Vision, 1995*, pages 457–464. IEEE. (Cited on pages 45 and 114)

Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A modern approach*. Prentice Hall. (Cited on pages 5, 17, 27, 33, and 90)

Rusu, R. B. (2010). Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, 24(4):345–348. (Cited on page 9)

Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *IEEE Int. Conf. Robotics and Automation (ICRA), 2011*, pages 1–4. IEEE. (Cited on page 7)

Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., and Beetz, M. (2008). Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941. (Cited on page 9)

Rygielski, C., Wang, J.-C., and Yen, D. C. (2002). Data mining techniques for customer relationship management. *Technology in Soc.*, 24(4):483–502. (Cited on page 9)

Saad, Y. (2003). *Iterative methods for sparse linear systems*. SIAM. (Cited on page 56)

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229. (Cited on page 1)

Santos, J. M. and Embrechts, M. (2009). On the use of the adjusted rand index as a metric for evaluating supervised classification. In *Artificial neural networks–ICANN 2009*, pages 175–184. Springer. (Cited on page 33)

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient ransac for point-cloud shape detection. In *Comput. graphics forum*, volume 26, pages 214–226. Wiley Online Library. (Cited on page 9)

Schuhmacher, D., Vo, B.-T., and Vo, B.-N. (2008). A consistent metric for performance evaluation of multi-object filters. *IEEE Trans. Signal Process.*, 56(8):3447–3457. (Cited on pages 20, 45, 46, 48, 114, 115, and 117)

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464. (Cited on page 101)

Shekhar, R. and Jawahar, C. (2012). Word image retrieval using bag of visual words. In *Document Anal. Systems (DAS), 2012 10th IAPR Int. Workshop*, pages 297–301. IEEE. (Cited on page 11)

Simon, H. A. (1983). Why should machines learn? In Michalski, R., Carbonell, J., and Mitchell, T., editors, *Machine learning: An artificial intelligence approach*, pages 25–37. Springer. (Cited on pages 1 and 2)

Sitek, A., Huesman, R. H., and Gullberg, G. T. (2006). Tomographic reconstruction using an adaptive tetrahedral mesh defined by a point cloud. *IEEE Trans. Medical Imaging*, 25(9):1172–1179. (Cited on page 9)

Skomoroch, P. N. (2009). Wikipedia page traffic statistics. https://aws.amazon.com/datasets/wikipedia-page-traffic-statistics. Accessed October 2016. (Cited on page 9)

Stanford Graphics Laboratory (2014). Stanford 3d scanning repository. http://graphics.stanford.edu/data/3Dscanrep. Accessed October 2016. (Cited on pages 7 and 10)

Stoyan, D., Kendall, W. S., and Mecke, J. (1995). *Stochastic geometry and its applications*. John Wiley & Sons. (Cited on pages 37, 38, 42, 43, and 44)

Stoyan, D. and Penttinen, A. (2000). Recent applications of point process methods in forestry statistics. *Statistical Sci.*, 15(1):61–78. (Cited on page 38)

Sun, H., Sun, X., Wang, H., Li, Y., and Li, X. (2012). Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model. *IEEE GeoSci. and Remote Sensing Letters*, 9(1):109–113. (Cited on page 11)

Surace, C. and Worden, K. (2010). Novelty detection in a changing environment: a negative selection approach. *Mechanical Systems and Signal Processing*, 24(4):1114–1128. (Cited on page 29)

Takacs, R. (1986). Estimator for the pair–potential of a gibbsian point process. *Statistics: A J. Theoretical and Applied Statistics*, 17(3):429–433. (Cited on page 54)

Tarassenko, L., Hayton, P., Cerneaz, N., and Brady, M. (1995). Novelty detection for the identification of masses in mammograms. (Cited on pages 29 and 31)

Tax, D. M. and Duin, R. P. (1999). Support vector domain description. *Pattern recognition letters*, 20(11):1191–1199. (Cited on pages 30 and 31)

Tax, D. M. and Duin, R. P. (2001). Uniform object generation for optimizing one-class classifiers. *J. Machine Learning Research*, 2(Dec):155–173. (Cited on page 31)

Taylor, D. W. and Corne, D. W. (2003). An investigation of the negative selection algorithm for fault detection in refrigeration systems. In *Int. Conf. Artificial Immune Systems*, pages 34–45. Springer. (Cited on page 30)

Tirilly, P., Claveau, V., and Gros, P. (2008). Language modeling for bag-of-visual words image categorization. In *Proc. 2008 Int. Conf. Content-based image and video retrieval*, pages 249–258. ACM. (Cited on page 11)

Toldo, R., Castellani, U., and Fusiello, A. (2009). A bag of words approach for 3d object categorization. In *Int. Conf. Comput. Vision/Comput. Graphics Collaboration Techniques and Appl.*, pages 116–127. Springer. (Cited on page 12)

Tong, C. H., Gingras, D., Larose, K., Barfoot, T., and Dupuis, É. (2013). The canadian planetary emulation terrain 3d mapping dataset. *Int. Journal Robotics Research*. (Cited on page 7)

Truccolo, W., Eden, U. T., Fellows, M. R., Donoghue, J. P., and Brown, E. N. (2005).

A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of neurophysiology*, 93(2):1074–1089. (Cited on page 38)

Tryon, R. C. (1939). *Cluster analysis: correlation profile and orthometric (factor) analysis for the isolation of unities in mind and personality*. Edwards brothers. (Cited on page 33)

Tsoumakas, G. and Katakis, I. (2006). Multi-label classification: An overview. *Int. J. Data Warehousing and Mining*, 3(3). (Cited on page 24)

Uijlings, J. R., Smeulders, A. W., and Scha, R. J. (2009). Real-time bag of words, approximately. In *Proc. ACM Int. Conf. Image and Video Retrieval*, page 6. ACM. (Cited on page 11)

van Lieshout, M. (2000). *Markov Point Processes and their Applications*. Imperial College Press. (Cited on pages 40 and 41)

Vapnik, V. (1995). *The nature of statistical learning theory*. Springer. (Cited on pages 26 and 27)

Vedaldi, A. and Fulkerson, B. (2008). Vlfeat: An open and portable library of comput. vision algorithms. http://www.vlfeat.org/. (Cited on page 65)

Vlasblom, J. and Wodak, S. J. (2009). Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. *BMC bioinformatics*, 10(1):99. (Cited on page 120)

Vo, B.-N. and Ma, W.-K. (2006). The Gaussian mixture probability hypothesis density filter. *Signal Process., IEEE Trans.*, 54(11):4091–4104. (Cited on page 39)

Vo, B.-N., Phung, D., Tran, Q. N., and Vo, B.-T. (2017a). Model-based multiple instance learning. *arXiv:1703.02155*. (Cited on pages 102 and 140)

Vo, B.-N., Singh, S., and Doucet, A. (2005). Sequential monte carlo methods for multitarget filtering with random finite sets. *Aerosp. Electron. Syst., IEEE Trans.*, 41(4):1224–1245. (Cited on pages 39 and 41)

Vo, B. N., Vo, B.-T., and Hoang, H. (2017b). An efficient implementation of the generalized labeled multi-bernoulli filter. *IEEE Trans. Signal Processing*, 65(8):1975–1987. (Cited on page 39)

Vo, B.-N., Vo, B.-T., and Mahler, R. P. (2012). Closed-form solutions to forward-backward smoothing. *IEEE Trans. Signal Processing*, 60(1):2–17. (Cited on page 39)

Vo, B.-N., Vo, B.-T., Pham, N.-T., and Suter, D. (2010). Joint detection and estimation of multiple objects from image observations. *IEEE Trans. Signal Process.*, 58(10):5129–5141. (Cited on page 39)

Vo, B.-N., Vo, B.-T., and Phung, D. (2014). Labeled random finite sets and the bayes multi-target tracking filter. *IEEE Trans. Signal Process.*, 62(24):6554–6567. (Cited on page 39)

Vo, B.-T. and Vo, B.-N. (2013). Labeled random finite sets and multi-object conjugate priors. *IEEE Trans Signal Process.*, 61(13):3460–3475. (Cited on page 39)

Vo, B.-T., Vo, B.-N., and Cantoni, A. (2007). Analytic implementations of the cardinalized probability hypothesis density filter. *Signal Process., IEEE Trans.*, 55(7):3553–3567. (Cited on page 39)

Vo, B.-T., Vo, B.-N., and Cantoni, A. (2009). The cardinality balanced multi-target multi-bernoulli filter and its implementations. *IEEE Trans. Signal Process.*, 57(2):409–423. (Cited on page 39)

Wang, J. and Zucker, J.-D. (2000). Solving multiple-instance problem: A lazy learning approach. (Cited on page 13)

Wasserman, L. (2004). *All of statistics: a concise course in statistical inference*. Springer. (Cited on pages 55 and 56)

West, K. and Cox, S. (2004). Features and classifiers for the automatic classification of musical audio signals. In *ISMIR*. Citeseer. (Cited on page 12)

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann. (Cited on pages 2, 24, 25, and 101)

Woo, H., Kang, E., Wang, S., and Lee, K. H. (2002). A new segmentation method for point cloud data. *Int. Journal Machine Tools and Manufacture*, 42(2):167–178. (Cited on page 9)

Wu, L., Hoi, S. C., and Yu, N. (2010). Semantics-preserving bag-of-words models and applications. *IEEE Trans. Image Process.*, 19(7):1908–1920. (Cited on page 12)

Xu, L., Krzyzak, A., and Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. Syst., Man, Cybern.*, 22(3):418–435. (Cited on page 24)

Xu, R. and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Trans. Neural Networks*, 16(3):645–678. (Cited on page 34)

Yang, M.-S., Hu, Y.-J., Lin, K. C.-R., and Lin, C. C.-L. (2002a). Segmentation techniques for tissue differentiation in mri of ophthalmology using fuzzy clustering algorithms. *Magnetic Resonance Imaging*, 20(2):173–179. (Cited on page 33)

Yang, Y., Guan, X., and You, J. (2002b). Clope: a fast and effective clustering algorithm for transactional data. In *Proc. 8th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 682–687. (Cited on page 9)

Yoganarasimhan, H. (2016). Search personalization using machine learning. *Available at SSRN 2590020*. (Cited on page 2)

Zahay, D., Peltier, J., Schultz, D. E., and Griffin, A. (2004). The role of transactional versus relational data in imc programs: Bringing customer data together. *J. advertising research*, 44(01):3–18. (Cited on page 9)

Zeng, Z., Zhang, S., Li, H., Liang, W., and Zheng, H. (2009). A novel approach to musical genre classification using probabilistic latent semantic analysis model. In *2009 IEEE Int. Conf. Multimedia and Expo*, pages 486–489. IEEE. (Cited on page 12)

Zhang, D., Wang, F., Si, L., and Li, T. (2009). M3ic: Maximum margin multiple instance clustering. In *IJCAI*, volume 9, pages 1339–1344. (Cited on pages 90 and 127)

Zhang, J., Marszałek, M., Lazebnik, S., and Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. Comput. Vision*, 73(2):213–238. (Cited on pages 20, 50, and 127)

Zhang, M.-L. and Zhou, Z.-H. (2009). Multi-instance clustering with applications to multi-instance prediction. *Appl. Intell.*, 31(1):47–68. (Cited on pages 90, 114, and 119)

Zhang, M. Y., Russell, J. R., and Tsay, R. S. (2001a). A nonlinear autoregressive conditional duration model with applications to financial transaction data. *J. Econometrics*, 104(1):179–207. (Cited on page 38)

Zhang, Q., Goldman, S. A., et al. (2001b). Em-dd: An improved multiple-instance learning technique. In *NIPS*, volume 1, pages 1073–1080. (Cited on page 50)

Zhang, X. (2011). Adaptive control and reconfiguration of mobile wireless sensor networks for dynamic multi-target tracking. *IEEE Trans. Automatic Control*, 56(10):2429–2444. (Cited on page 39)

ZhiLiu (2011a). Amazon commerce reviews. http://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set. Accessed October 2016. (Cited on page 11)

ZhiLiu (2011b). Reuter-50-50 data set. http://archive.ics.uci.edu/ml/datasets/Reuter_50_50. Accessed October 2016. (Cited on page 11)

Zhong, Y. (2009). Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *Comput. Vision Workshops (ICCV Workshops), 2009 IEEE 12th Int. Conf.*, pages 689–696. IEEE. (Cited on page 9)

Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.

INDEX