Towards Probabilistic and Partially-Supervised Structural Health Monitoring

L. A. Bull

l.a.bull@sheffield.ac.uk

https://github.com/labull

A Thesis submitted to the University of Sheffield

for the degree of Doctor of Philosophy in the Faculty of Engineering

Department of Mechanical Engineering

University of Sheffield

February 24, 2020

Acknowledgements

First and foremost, thanks to my supervisors, Dr Nikolaos Dervilis and Prof. Keith Worden — both have made the PhD a truly enjoyable experience, while offering the best guidance, support and conversation.

I am also grateful for the support from the following friends within the DRG: particularly Tim Rogers, who has to deal with interruptions relating to sythesiser schematics, as well as work; Chandy Wickramarachchi, who has put up with these (often unrelated) conversations in the office, while happily sharing and repeatedly explaining her data; also Paul Gardner, who has been (forced) to join an uncountable number of coffee breaks for advice and chat; and Dr Elizabeth Cross, who has helpfully reminded me (more than once) that my PhD is in Engineering.

Thanks Dr Graeme Manson and Dr Rhys Pullin, for granting permission to use their experimental datasets.

Special thanks to my family and friends; and thank you Jess.

Dedicated to my parents

Abstract

One of the most significant challenges for signal processing in data-based structural health monitoring (SHM) is a lack of comprehensive data; in particular, recording labels to describe what each of the measured signals represent.

For example, consider an offshore wind-turbine, monitored by an SHM strategy. It is infeasible to artificially damage such a high-value asset to collect signals that might relate to the damaged structure *in situ*; additionally, signals that correspond to abnormal wave-loading, or unusually low-temperatures, could take several years to be recorded. Regular inspections of the turbine in operation, to describe (and *label*) what measured data represent, would also prove impracticable — conventionally, it is only possible to check various components (such as the turbine blades) following manual inspection; this involves travelling to a remote, offshore location, which is a high-cost procedure.

Therefore, the collection of labelled data is generally *limited* by some expense incurred when investigating the signals; this might include direct costs, or loss of income due to down-time. Conventionally, incomplete label information forces a dependence on unsupervised machine learning, limiting SHM strategies to damage (i.e. novelty) detection. However, while comprehensive and fully labelled data can be rare, it is often possible to provide labels for a limited subset of data, given a label *budget*. In this scenario, *partially-supervised* machine learning should become relevant. The associated algorithms offer an alternative approach to monitor measured data, as they can utilise *both* labelled and unlabelled signals, within a unifying training scheme.

In consequence, this work introduces (and adapts) partially-supervised algorithms for SHM; specifically, *semi-supervised* and *active* learning methods. Through applications to experimental data, semi-supervised learning is shown to utilise information in the unlabelled signals, alongside a limited set of labelled data, to further update a predictive-model. On the other hand, active learning improves the predictive performance by querying specific signals to investigate, which are assumed the most informative. Both discriminative and generative methods are investigated, leading towards a novel, probabilistic framework, to classify, investigate, and label signals for online SHM. The findings indicate that, through partially-supervised learning, the cost associated with labelling

data can be managed, as the information in a *selected* subset of labelled signals can be combined with larger sets of unlabelled data — increasing the potential scope and predictive performance for data-driven SHM.

Publications

Journal Publications

L. A. Bull, K. Worden, N. Dervilis. Towards semi-supervised and probabilistic classification in structural health monitoring. *Preprint Submitted to Mechanical Systems and Signal Processing*, 2019.

L. A. Bull, T. J. Rogers, C. Wickramarachchi, E. J. Cross, K. Worden, N. Dervilis. Probabilistic active learning: An online framework for structural health monitoring. *Mechanical Systems and Signal Processing*, 134 (pp. 106294), 2019.

L. A. Bull, K. Worden, R. Fuentes, G. Manson, E. J. Cross, N. Dervilis. Outlier ensembles: A robust method for damage detection and unsupervised feature extraction from high-dimensional data. *Journal of Sound and Vibration*, 453 (pp. 126-150), 2019.

L. A. Bull, K. Worden, G. Manson, N. Dervilis. Active learning for semi-supervised structural health monitoring. *Journal of Sound and Vibration*, 437 (pp. 373-388), 2018.

Conference Proceedings

L. A. Bull, K. Worden, N. Dervilis. Damage classification using labelled and unlabelled measurements. *Paper Presented at IWSHM*, 2019.

L. A. Bull, N. Dervilis, K. Worden. Experimental validation of the population-Form to represent nominally identical systems. *Paper Presented at IWSHM*, 2019.

L. A. Bull, T. J. Rogers, E. J. Cross, N. Dervilis, K. Worden. A Gaussian process form for population-based structural health monitoring. In conference proceedings *DAMAS*, 2019.

N. Dervilis, T. Zhang, *L. A. Bull*, E. J. Cross, T. J. Rogers, R. Fuentes, V. Dertimanis, I. Abdallah, E. Chatzi, K. Worden. A nonlinear robust outlier detection approach for SHM. *Paper Presented at IOMAC*, 2019.

L. A. Bull, K. Worden, T. J. Rogers, C. Wickramarachchi, E. J. Cross, T. McLeay, W. Leahy, N. Dervilis. A probabilistic framework for online structural health monitoring: Active learning from machining data streams. In conference proceedings *Recent Advances in Structural Dynamics*, 2019.

L. A. Bull, K. Worden, T. J. Rogers, E. J. Cross, N. Dervilis. Investigating engineering data by probabalistic measures. *Paper Presented at IMAC XXXVII*, 2019.

L. A. Bull, G. Manson, K. Worden, N. Dervilis. Active learning approaches to structural health monitoring. In *Special Topics in Structural Dynamics, Volume 5* (pp. 157-159). Springer, Cham, 2019.

Contents

1 Data-driven SHM **1**

1-1 Structural Health Monitoring 1

1-2 SHM in Practice 3

1-2.1 Methodologies 3

1-3 SHM as Pattern Recognition 4

1-3.1 A Probabilistic Approach 5

1-4 A Motivating Example: Conventional Learning in Data-driven SHM 7

1-4.1 Acoustic Emission Data 8

1-4.2 Feature Extraction and Dimension Reduction 8

1-4.3 Unsupervised Learning: Outlier Analysis for Damage Detection 11

1-4.4 Supervised Learning: Probabilistic Damage-Classification 14

1-5 Motivation: Outstanding Challenges in Data-driven SHM 19

1-5.1 Contribution 20

2 Towards Probabilistic and Partially-supervised SHM **21**

2-1 Probabilistic Classifiers 21

2-1.1 Generative Approach 21

2-1.2 Discriminative Approach 22

2-2 Partially-supervised Learning 24

2-2.1 Semi-supervised Learning 24

2-2.2 Active Learning 27

2-2.3 The Dangers of Partially-supervised Learning 30

2-3 Thesis Layout 31

3 Hierarchical Sampling for Active Learning **33**
3-1 Cluster-based Methods and Sampling Bias 33
3-1.1 Sampling Bias 35
3-2 A Cluster-based Framework for Guided Sampling 37
3-2.1 Clustering 38
3-2.2 An Overview of Guided Sampling and Label Propagation 39
3-2.3 Pruning & Node Properties 41
3-2.4 Admissible Clusters 43
3-2.5 The Select Procedure 44
3-2.6 Pruning Refinements 44
3-2.7 Label Propagation 45
3-2.8 The Algorithm 45
3-3 Experiments 47
3-3.1 Gnat Aircraft Data 47
3-3.2 Test procedure 50
3-3.3 Results & Discussion 51
3-4 Concluding Remarks 56

4 Probabilistic Active Learning for Online SHM **59**
4-1 Generative Mixture Models 59
4-2 A Probabilistic Model for Guided Sampling 60
4-2.1 A Bayesian Approach 61
4-2.2 Data query measures: uncertainty sampling 68
4-3 An Online SHM Framework 70
4-3.1 Guided Sampling 71
4-4 Experiments 73
4-4.1 Z24 bridge data 74
4-4.2 Machining data 78
4-4.3 Gnat Aircraft Data: Outlier Ensemble Features 81
4-5 Limitations 86
4-6 Concluding Remarks 88

5 Towards Probabilistic and Semi-Supervised Damage Classification **90**
5-1 Applications to SHM . . . 91
5-1.1 Related work . . . 91
5-1.2 Contribution . . . 92
5-2 Mixture Models for Semi-Supervised SHM . . . 92
5-2.1 Semi-Supervised updates: Expectation Maximisation . . . 95
5-3 Experiments . . . 98
5-3.1 Simulated Dataset . . . 99
5-3.2 Gnat Aircraft Data . . . 105
5-4 Concluding Remarks . . . 106

6 Towards a Combined Semi-Supervised and Active Learner **109**
6-1 Combined Online Framework . . . 109
6-1.1 Improved Uncertainty Sampling . . . 112
6-2 Experiments and Discussion . . . 113
6-3 Concluding Remarks . . . 119

7 Conclusions **122**
7-1 Summary . . . 123
7-2 Limitations & Future Work . . . 126
7-2.1 Future work . . . 127

1

DATA-DRIVEN SHM

Overview: As digital storage improves, and sensing devices proliferate, engineering systems have the potential to provide a variety of insightful data. This information has been utilised in various applications of structural dynamics, including: system identification [1, 2], model validation [3], control [4], and structural health monitoring (SHM) [5]. While datasets may be large, descriptions of what the measurements represent is regularly limited [6–9]. Considering SHM, this work explores novel methods for statistical pattern recognition, with limited information to describe the measured signals.

1-1. Structural Health Monitoring

'Structural health monitoring (SHM) refers to the process of implementing a damage detection strategy for aerospace, civil or mechanical engineering [systems]' [5]. Generally, a system is monitored over time though signal processing of measured data; these measurements are usually dynamic response [5], but alternative measures from temperature, image [10] or sound data have the potential to be used. Damage-sensitive features are extracted from the data, and the analysis of the features can be used to determine the current operational state of the system [11]. Ideally, feature-analysis should accommodate for benign variations in the operational conditions, including inevitable ageing or changes in the environment [12]. SHM strategies should be applied (and updated) online, in real time, during the operation of the monitored system [13].

(a) (b)

Figure 1.1: Applications of SHM: (a) The RAPTOR telescope system, investigated at the Los Alamos National Laboratory [16]. (b) A wind turbine, off the coast of Aberdeen, Scotland; image credit: TVP Film and Multimedia Ltd.

The development of SHM strategies for structural and mechanical systems should be considered an important aspect of engineering design, as automated diagnostics have the potential to detect and classify damage before more conventional (manual) inspection or testing[1]. Automated monitoring is particularly relevant to systems with limited access, as manual inspection can become problematic; this could refer to specific components that are difficult or impossible to inspect (e.g. the cutting tool within a turning machine [14]), or structures operating in remote locations (e.g. offshore wind turbines [15] or robotic telescopes [16], illustrated in Figure 1.1). Additionally, SHM is relevant to industries associated with high costs for maintenance or downtime, as well as those with a high risk to human safety. Key sectors include: aerospace, civil infrastructure, manufacturing, the automotive industry, and the power sector. In summary, the motivations for implementing SHM (*alongside* more traditional inspection and testing) are simple; automated monitoring has the potential to [17]:

- increase the safety of structures;
- increase the economic output by minimising downtime;
- reduce the cost of maintenence.

[1]Typically, conventional techniques involve Non-Destructive Testing (NDT), which assess the system *offline* and intermittently [5].

1-2. SHM in Practice

The typical SHM problem can be defined as hierarchy of levels, first suggested by Rytter [18]. This framework is generalised below [19]:

I *Detection*: an indication that damage might be present.
II *Location*: a prediction of the location of damage.
III *Classification*: a prediction of the type of damage.
IV *Assessment*: a prediction of the extent of damage.
V *Prediction*: a method for prognosis.

Generally, in practical applications, each level requires that the previous levels have been addressed. However, following recent trends in the literature [8, 20–22], the techniques suggested in this work look to combine the first three levels from the hierarchy in to one model; specifically, detection (I) and location/classification (II/III).

1-2.1. Methodologies

In simple terms, SHM follows two methodologies [5, 23].

The model-based approach: a physics-based model of the structure is built and then used to simulate data. This model is validated and updated using measured data. Some comparison between the model and measured data is then used to monitor the system.

The data-driven approach: the model of the system is not based on physical laws; instead, *machine learning* tools are applied to learn *patterns* within measured data. These patterns are then associated with different conditions of the system.

Both techniques have their advantages. When utilising a physics-based model, a variety of operational and environmental conditions have the potential to be simulated; however, the model must be regularly validated, in order to ensure reasonable predictions [3, 24]. Additionally, as computational models become more complex, incorporating uncertainty becomes increasingly difficult. On the other hand, when following the data-driven approach, complex behaviour can be learnt from the data without having to define a model from first physical principals. Additionally, uncertainty can be naturally incorporated within

probabilistic approaches to pattern-recognition [25]. Unfortunately, machine learning algorithms (generally) require large datasets to be recorded during system operation for reliable predictions. Specifically, for *unsupervised* methods in machine learning, large quantities of measured signals are required, while *supervised* techniques also require the measurements to be comprehensively labelled, to describe what each of the signals represent[2]. Furthermore, generic machine learning tools offer little insight into the underlying physics, i.e. they are *black box* models [25].

Analysis in SHM, and engineering in general, should require a combination of the model and data-driven approach, as valuable information can be derived from both methodologies. (As a result, existing research concerns the combination of both methodologies — a form of *grey-box* modelling [26] — as well as frameworks for combining measured and simulated data [3, 27].) The focus of this work, however, concerns the statistical modelling of measured signals; therefore, it is concerned with the data-driven aspects of SHM.

1-3. SHM as Pattern Recognition

'*Machine learning is a set of methods that can learn and detect patterns in data, and then use these uncovered patterns to predict future data, or perform other kinds of decision making*' [25] — ideally under uncertainty. Intuitively, machine learning theory can naturally address the problems stated in Rytter's hierarchy [5, 11]: patterns learnt from measured data can inform the current operating condition and diagnose faults, while considering the uncertainty of predictions [9]. In other words, machine learning algorithms should be able to discriminate between groups of measured signals that relate to the different operational and health conditions. For example: is the system operating under normal conditions, extreme temperatures, or, most critically, is the system damaged?

This work will refer to the data-driven SHM strategy illustrated in Figure 1.2. Specifically, SHM is viewed as a multi-class classification problem, which categorises measured data into groups, corresponding to the condition of the monitored system. The i^{th} input, denoted by $\mathbf{x}_i$, is defined by a D-dimensional vector of variables, which represents an *observation* of the system, s.t. $\mathbf{x}_i \in \mathbb{R}^D$.

[2]Supervised and unsupervised methods are introduced in detail in Section 1-4.

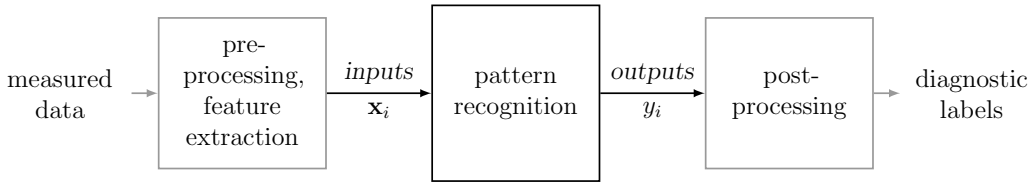


Figure 1.2: A framework for pattern recognition within SHM.

The variables can be direct measurements, or, more typically, features derived from the measured data by pre-processing and feature extraction. The data *labels* y_i, are used to specify the condition of the system, *directly* or *indirectly*. If indirectly, diagnostic labels can be inferred through some post-processing of the pattern recognition outputs y_i.

1-3.1. A Probabilistic Approach

Considering a probabilistic perspective, the expression $P(A)$ denotes the *probability* that event A is true. A probability requires that $0 \leq P(A) \leq 1$, such that $P(A) = 1$ implies that event A definitely *will* happen, while $P(A) = 0$ implies event A definitely *won't* happen [25].

In the context of SHM, the inputs $\mathbf{x}_i$ are (generally) assumed to be represented by some (continuous) random vector X, which can take any value within a given feature-space $\mathcal{X}$. The random vector is therefore associated with an appropriate probability density function (p.d.f.), denoted by lower-case p notation. The p.d.f. is such that the probability of X falling within the interval $a < X \leq b$ is,

$$P\left(a < X \leq b\right) = \int_a^b p\left(\mathbf{x}_i\right) \, d\mathbf{x}_i \quad \text{s.t.} \quad p\left(\mathbf{x}_i\right) \geq 0, \quad \int_{\mathcal{X}} p\left(\mathbf{x}_i\right) \, d\mathbf{x}_i = 1 \tag{1.1}$$

specifically, in this work, the observations are assumed to be sampled from some D-dimensional feature-space $\mathcal{X}$, s.t. $\mathbf{x}_i \in \mathcal{X} \in \mathbb{R}^D$.

For an increasingly narrow interval, the probability given some continuous random variable can be approximated as follows [25],

$$P\left(\mathbf{x}_i \leq X \leq \mathbf{x}_i + d\mathbf{x}_i\right) \approx p\left(\mathbf{x}_i\right) d\mathbf{x}_i \tag{1.2}$$

For discrete classification in SHM, the labels y_i are represented by a discrete random variable Y, which can take any value from the finite set $y_i \in \mathcal{Y} =$

$\{1, ..., K\}$; K is the number of classes which define the (observed) operational, environmental, and health conditions, while $\mathcal{Y}$ denotes the label-space. An appropriate probability mass function (p.m.f.) leads to,

$$P(Y = y_i) \quad \text{s.t.} \quad 0 \leq P(Y = y_i) \leq 1, \quad \sum_{y_i \in Y} P(Y = y_i) = 1 \tag{1.3}$$

From herein, probabilities such as $P(Y = y_i)$ are given as $P(y_i)$ for brevity; additionally, $p()$ notation refers to both p.d.fs and p.m.fs — the context should make this distinction clear.

A probabilistic perspective can naturally address the *ambiguous case*, in which measured signals cannot be categorised with certainty (given the data) [25]. Uncertainty is inevitable for all measured data in science and engineering applications, and in consequence, it should be modelled appropriately. Provided specific assumptions hold, probabilistic methods allow for predictions with well-defined uncertainty under Kolmogorov's axioms [28]; this is a significant advantage in risk-based applications, such as SHM [8, 9, 20, 29, 30]. For example, consider a *certain* prediction, which states an oil-rig is safe to use; this differs significantly to an *uncertain* prediction, leading to the same statement.

Fundamental probability theory

An overview of basic probability theory is provided; for further details, refer to text-books [25, 28, 31, 32]. Random variables have been (informally) introduced, so the basic operations/rules are provided.

The probability of a union of two events (i.e. the probability of A *or* B),

$$\begin{aligned} P(A \cup B) &= P(A) + P(B) - P(A \cap B) \\ &= P(A) + P(B) \quad \text{(if } A \text{ and } B \text{ are mutually exclusive)} \end{aligned} \tag{1.4}$$

The *joint* probability is the probability of A *and* B, which leads to the *product rule*,

$$P(A, B) = P(A \cap B) = P(A \,|\, B)\, P(B) \tag{1.5}$$

where $P(A \,|\, B)$ is the *conditional* probability of event A given that B has occurred; i.e. event A conditioned on B (1.8). A and B are interchangeable on the R.H.S.

Given a joint distribution, variables can be *marginalised out* by summing (or integrating) over all possible values for that variable,

$$P(A) = \sum_b P(A, B) = \sum_b P(A \,|\, B = b)\, P(B = b) \qquad \text{(discrete)} \tag{1.6}$$

$$\text{or} \quad p(A) = \int_{-\infty}^{\infty} p(A \,|\, B)\, p(B)\, dB \qquad \text{(continuous)}$$

(The same method applies when marginalising out A.)

For both discrete and continuous variables, the product rule can be applied multiple times to yield the *chain rule*,

$$p(X_{1:D}) = p(X_1)p(X_2 \,|\, X_1)p(X_3 \,|\, X_1, X_2) \dots p(X_D \,|\, X_{1:D-1}) \tag{1.7}$$

$$X_{1:D} \triangleq \{X_1, X_2 \dots, X_D\}$$

Also from the product rule (1.5), the *conditional probability density* is,

$$p(A \,|\, B) = \frac{p(A, B)}{p(B)} \tag{1.8}$$

Leading to Bayes' rule,

$$p(A \,|\, B) = \frac{p(B \,|\, A)\, p(A)}{p(B)} \tag{1.9}$$

Where $p(A)$ is the *prior*-distribution, $p(B \,|\, A)$ is the *likelihood*, $p(B)$ is the *marginal-likelihood*, and $p(A \,|\, B)$ is the *posterior*-distribution.

Finally, marginal (1.10) and conditional independence (1.11), denoted with $\perp$, imply that,

$$A \perp B \;\longleftrightarrow\; p(A, B) = p(A)\, p(B) \tag{1.10}$$

$$A \perp B \,|\, C \;\longleftrightarrow\; p(A, B \,|\, C) = p(A \,|\, C)\, p(B \,|\, C) \tag{1.11}$$

1-4. A Motivating Example: Conventional Learning in Data-driven SHM

As discussed, when categorising the measurements $\mathbf{x}_i$ from a system or structure, algorithms (or 'machines') can be applied to *learn* which diagnostic labels y_i are associated with certain patterns within the measured signals. Therefore, a

dataset must be available (in some form) in order to *train* the algorithm. The process of learning from a subset of training data can be defined in various ways; in the context of SHM, a visual example is provided, to motivate and demonstrate the research presented in this work.

1-4.1. Acoustic Emission Data

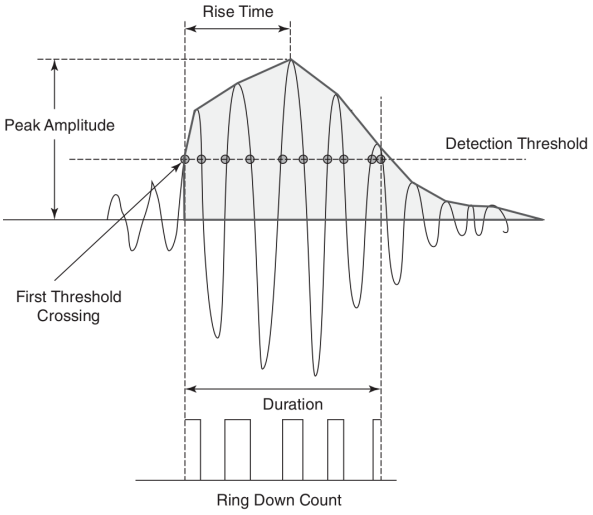
An acoustic emission (AE) dataset — collected by Rippengill *et. al* at Cardiff University [33] — is used to demonstrate statistical pattern recognition for SHM. Measurements were recorded during experiments in which the box-girder of a bridge was exposed to cyclic loading, from 0.1 to 85 kN [5]. Briefly, AE burst signals were extracted from the background noise of the measured data by setting a threshold based on the mean and six standard deviations; an example of a burst signal is shown in Figure 1.3a. A total of 91 AE burst signals were identified from the measured data (details of the test procedure can be found in [5, 33, 34]).

The object of this dataset is to distinguish between different AE sources, particularly those relating to crack growth, as this information should help to inform damage detection, classification and prognosis. There are various ways to implement machine learning in order to analyse the observed data; for example, time series analysis could learn a function (regression [25, 35]) in the time domain, to monitor the burst signals directly [34]. In this example, however, features are extracted from the burst signals, such that the feature-variables are sensitive to damage (as in [33]).

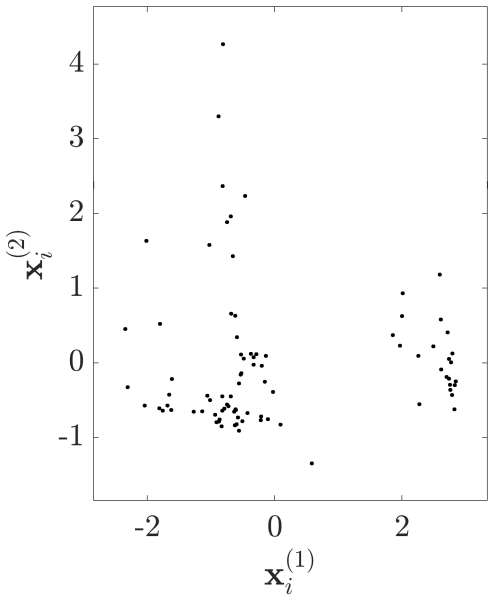
1-4.2. Feature Extraction and Dimension Reduction

Feature extraction involves the identification of *features* from the measured signals, which allow for one to distinguish between the damaged and undamaged states of the monitored system [36]. Ideally, the feature-set $\mathbf{x}_i$ should be a low-dimensional representation of the measured signals, which are sensitive to the condition of the system [5]. For the AE data, traditional AE features [5] are defined for each signal (illustrated in Figure 1.3a) as these variables should be sensitive to damage (i.e. crack-related events):

- *rise time*,



(a)



(b)

Figure 1.3: Acoustic emission data: (a) a burst signal with annotated features [5]; (b) 91 burst signals projected onto a two-dimensional feature space through PCA.

- *peak amplitude*,
- *duration*, and
- *ring down count*.

Therefore, each AE burst is represented by four-features; i.e. the observations are four-dimensional vectors. To visualise the data, and to aid discussion, *dimension reduction* is now applied, to represent the measurements in two dimensions, s.t.,

$$\{\mathbf{x}_i\}_{i=1}^{m}, \quad \mathbf{x}_i \in \mathbb{R}^2 \tag{1.12}$$

for m observations. Specifically, dimension reduction tools are a method for data compression, while retaining as much information as possible from the full feature-space. In this example, linear Principal Component Analysis (PCA) [25, 35] is applied; this is perhaps the most widely-used method for dimension reduction [37]. (PCA is used for visualisation throughout this work.) PCA is an orthogonal projection onto a lower-dimensional space, such that variation is maximised, dimension by dimension,

$$\mathbf{x}_i = \mathbf{W}^\top \hat{\mathbf{x}}_i \tag{1.13}$$

where $\hat{\mathbf{x}}_i$ denotes the observations in the original (full-dimensional) feature-space, and $\mathbf{x}_i$ denotes the observations in the principal subspace; $\mathbf{W}$ is an orthonormal projection matrix, defined by L linear basis-vectors. The optimal projection for maximum variation in L-dimensions is (provably [37]) obtained by setting the columns of $\mathbf{W}$ equal to the L eigenvectors with largest eigenvalues from the empirical covariance matrix, $\bar{\boldsymbol{\Sigma}} = \frac{1}{m-1}\sum_{i=1}^{m}(\mathbf{x}_i - \bar{\boldsymbol{\mu}})(\mathbf{x}_i - \bar{\boldsymbol{\mu}})^\top$, where $\bar{\boldsymbol{\mu}}$ is the sample mean, $\bar{\boldsymbol{\mu}} = \frac{1}{M}\sum_i \mathbf{x}_i$.

In this case, the data are projected from $D = 4$ to $D = 2$, so the first *two* eigenvectors (with largest eigenvalues) are used, i.e. $L = 2$. The corresponding feature-space $\mathcal{X}$ (the principal subspace) is plotted in Figure 1.3b. Machine learning tools are now applied to learn patterns in the data given this projection. Conventionally in SHM, patterns are learnt through *unsupervised* or *supervised* methods [5].

1-4.3. Unsupervised Learning: Outlier Analysis for Damage Detection

At this stage, descriptive labels for each of the AE signals are unavailable, as the physical process behind each signal has not been investigated; therefore, the dataset $\mathcal{D}_u$ is unlabelled,

$$\mathcal{D}_u = \{\tilde{\mathbf{x}}_i\}_{i=1}^{m}\,,\ \mathbf{x}_i \in \mathbb{R}^2 \tag{1.14}$$

$\tilde{\mathbf{x}}_i$ is used (throughout) to denote unlabelled observations. Here, *unsupervised learning* algorithms are suitable [38]; a variety of data-analysis and machine learning tools fall into this category. Some examples of methods include: dimensionality reduction, outlier analysis, and clustering [25]. These techniques aim to find patterns within a dataset from the information within the measured data alone; therefore, the learning process must *not* be informed by information from a label space $\mathcal{Y}$, as this information is not available [39].

The first level of Rytter's hierarchy, damage detection, is typically addressed though outlier analysis and novelty-detection algorithms [5, 40, 41]; therefore, novelty detection (or one-class classification [42]), is applied to demonstrate unsupervised SHM. (Examples of clustering and dimensionality-reduction are provided later in this work).

During novelty detection, the problem is to identify, from the measured data, if a machine or structure has deviated from the normal condition; that is, if the measured signals are novel [40]. In an engineering context, outliers can be suitably defined for *novelty detection* as:

> *'Data that deviate so much from other observations, as to arouse suspicions that they were generated by some different mechanism'* [43].

Therefore, outlying data should indicate a significant change in the underlying physics of that system, rather than benign fluctuations in measurement noise. Although this description is conceptually simple, detecting informative outliers from noisy engineering data is a non-trivial task.

The Mahalanobis distance

Statistical outlier analysis can be achieved by defining a parametric p.d.f, to characterise the random vector X. The parameters of the assumed p.d.f are estimated from the available (normal-condition) data, and a *discordancy test* can be used as a measure of novelty [40, 44].

Typically, the normal-condition data are assumed to be multivariate Gaussian-distributed,

$$\begin{aligned} p(\mathbf{x}_i) &= \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\,\boldsymbol{\Sigma}\,|^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu})^{\top} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right\} \end{aligned} \tag{1.15}$$

where the parameters are the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ (i.e. location and scatter). The parameters can be estimated in various ways, the most simple approach uses the sample mean, $\bar{\boldsymbol{\mu}} \triangleq \frac{1}{m}\sum_i \mathbf{x}_i$, and empirical covariance, $\bar{\boldsymbol{\Sigma}} \triangleq \frac{1}{m-1}\sum_{i=1}^{m} (\mathbf{x}_i - \bar{\boldsymbol{\mu}})(\mathbf{x}_i - \bar{\boldsymbol{\mu}})^{\top}$: corresponding to the maximum likelihood (ML) estimator — denoted with a bar script throughout. (More involved estimates are discussed in the proceeding chapters.)

To illustrate the method, a subset of the AE signals are assumed to represent the normal data, sampled from X. As such, this is *exclusive* outlier-analysis: while the labels remain unknown, the subset of normal-data are given. (Hence the alternative name for exclusive-analysis — one-class classifiers [42]). Specifically, the normal data $\mathcal{D}_u$ are the AE bursts due to *frictional processes* caused by the clamping arrangement of the test rig [34], shown by the green $\times\bullet$ markers in Figure 1.4. Any observations that are generated by an alternative mechanism (i.e. *crack related events* [34], $\bullet$ markers) can be considered as outliers; these data are visibly novel compared to the normal-data in the feature-space in Figure 1.4a.

The corresponding ML estimate of the p.d.f for the normal-condition data is also illustrated in Figure 1.4a; i.e. the sample mean and covariance $(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}})$, given the training subset ($\times$ markers). As the outlying data appear significantly different to the training data, a discordancy measure can be defined to quantify novelty. In this case, an appropriate metric is the Mahalanobis-squared-distance (MSD) [5, 40]. The MSD can be interpreted as a covariance-weighted squared-Euclidean-distance from the mean $\bar{\boldsymbol{\mu}}$ of the normal data — if the covariance is

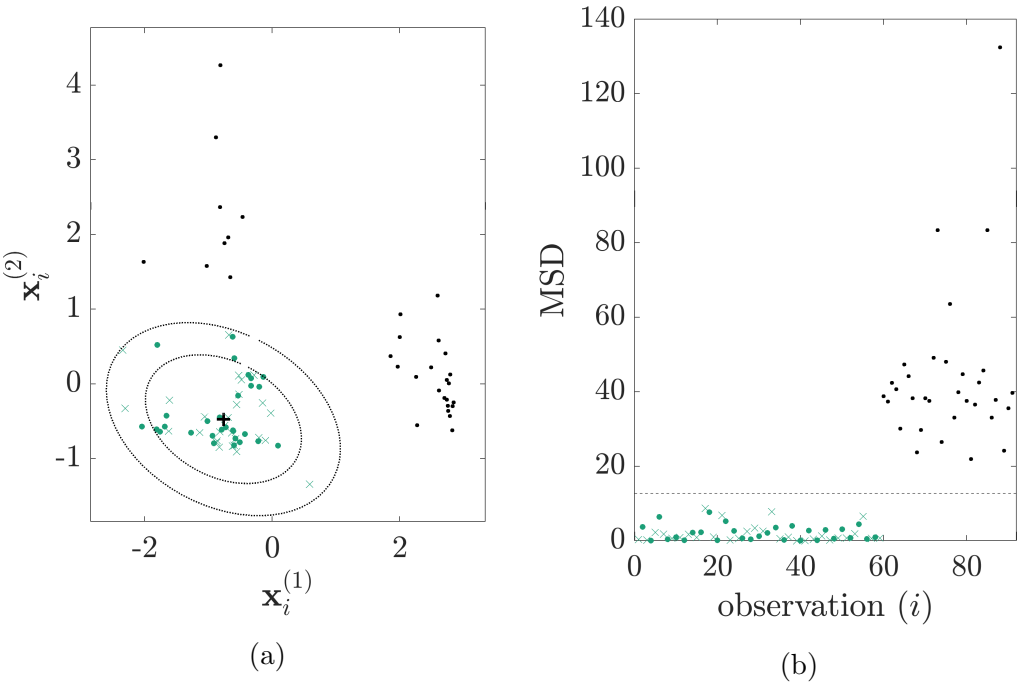


Figure 1.4: Exclusive (MSD) outlier analysis with the AE data; training and test sets are shown by $\times$ and $\bullet$ markers respectively: (a) Observations in the feature space; the ML estimate of $p(\mathbf{x}_i \,|\, \mathcal{D}_u)$ is shown by the sample mean (+) and covariance (dotted lines represent 2 and 3 sigma). (b) MSD discordancy measure for each of the burst signals.

equal to the identity they become synonymous [35],

$$MSD(\mathbf{x}_i^*) = (\mathbf{x}_i^* - \bar{\boldsymbol{\mu}})^\top \bar{\boldsymbol{\Sigma}}^{-1} (\mathbf{x}_i^* - \bar{\boldsymbol{\mu}}) \tag{1.16}$$

where $\mathbf{x}_i^*$ is the potentially outlying observation. In effect, the MSD quantifies the likelihood of *new* observations, given the data known to represent the normal-condition, $\mathcal{D}_u$; that is, $p(\mathbf{x}_i^* \,|\, \mathcal{D}_u)$ [35].

The corresponding MSD for each observation in the AE data is shown in Figure 1.4b. Considering that ML estimates of the parameters are used, the model risks *overtraining*; this implies that the model can overfit to the training data, leading to poor generalisation when applied to new data — the issues of overtraining are discussed in further detail in Section 4-2.1. Consequently, in this case, it is critical to ensure good generalisation through validation, or an alternative form of regularisation [35]. Typically in unsupervised SHM, a distinct set of normal-condition test-data (that were not used to estimate the parameters) are used as a *validation-set*, to ensure generalisation [40]; these data are shown by the green • markers in Figure 1.4. As expected, the normal condition data have low discordancy measures, suggesting these data are sampled from the same underlying distribution. On the other hand, crack-related signals show higher measures of discordancy, suggesting these are outliers, generated by some alternative and *novel* mechanism.

The performance of a novelty detector can be quantified using Type-I and Type-II errors given a distinct test-set (• markers) [40]. Specifically, Type-I errors — referred to as *false positives* (FP) — include observations that are classified as outliers when they are in fact inlying. On the other hand, Type-II errors — also called *false negatives* (FN) — include observations that are outlying, but fail to be rejected by the novelty detector. Conveniently, for the example in Figure 1.4b, there is zero-error for both Type-I and Type-II errors.

1-4.4. Supervised Learning: Probabilistic Damage-Classification

Moving up Rytter's hierarchy, damage location (II) and classification (III) are more problematic, as the corresponding algorithms require more information [9, 38]. (Critically, this information can be unavailable in SHM.) It is desirable, however, to classify measurements into multiple groups, which correspond to

the various system conditions, rather than simply undamaged or damaged.

Generally speaking, *supervised learning* is applied when information is available for multi-class damage-classification. These algorithms require fully-labelled training-data $\mathcal{D}_l$, such that each observation $\mathbf{x}_i$ is associated with a label $y_i \in \{1, ..., K\}$, for n collected data points,

$$\mathcal{D}_l = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \tag{1.17}$$

A supervised classifier approximates the mapping between the feature-space and the label-space, $f : \mathcal{X} \to \mathcal{Y}$. The classifier f is then used to predict the label associated with future measurements, and inform diagnostic decisions in the context of SHM.

Probabilistic mixture models

Considering the AE data, a probabilistic example is provided. Following investigation of the signals, the 91 observations can be (approximately) split into three classes [34]:

- class 1 - frictional processes away from the crack (clamping in the experimental setup)
- class 2 - crack-related events (crack extension and crack-face rubbing)
- class 3 - crack-related events at a distance from the sensor (reflections)

The fully labelled data are illustrated in Figure 1.5a.

Continuing with a parametric and statistical approach, the random variable X is represented by a parametric p.d.f. However, the AE data are now *multi-class*, therefore, it is appropriate to model X with a mixture-model; in this case, a Gaussian Mixture Model (GMM) [20, 29, 30]. Through a GMM, the underlying distribution of the measured data $\mathbf{x}_i \in \mathcal{X}$, for *each class* k, is described by a Gaussian distribution,

$$p\left(\mathbf{x}_i \,|\, y_i = k\right) = \mathcal{N}\left(\mathbf{x}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \tag{1.18}$$

k is used to index the class group, such that $k \in \{1, ..., K\}$; therefore, $\boldsymbol{\mu}_k$ is the mean and $\boldsymbol{\Sigma}_k$ is the covariance of the data $\mathbf{x}_i$ with label $y_i = k$.

The discrete random variable Y, which describes the labels $y_i \in \{1, ..., K\}$, is assumed to be categorically distributed [31],

$$P(y_i) = \text{Cat}(y_i \,|\, \boldsymbol{\lambda}) \tag{1.19}$$

$\boldsymbol{\lambda}$ is vector of *mixing proportions*, which is a histogram over the label values, s.t. $\boldsymbol{\lambda} = \{\lambda_1, ..., \lambda_K\}$ and $P(y_i = k) = \lambda_k$.

As in the outlier example, the ML estimator can be used to approximate the parameter-set $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}\}$; however, to improve generalisation — and avoid validation procedures — a Bayesian approach can be adopted [25, 31, 32]. This involves considering the parameters to be random variables themselves, and incorporating *prior* belief in the distribution over their potential values, via Bayes' Rule (1.9). The Bayesian estimate leads to a *posterior*-distribution over the possible parameter values, rather than point estimates; in this example, the most probable value is selected for each parameter, corresponding to the maximum-*a-posteriori* (MAP) estimate[3]. (A detailed explanation of the Bayesian approach to statistical modelling is provided in Chapter 4.)

The resulting GMM of the AE data is visualised in the feature-space in Figure 1.5b: there are three observed classes ($K = 3$), therefore, there are three (Gaussian) base-distributions in the mixture-model. A random sample of 50% of the total data is used to train the algorithm ($\times$ markers, the set $\mathcal{D}_l$).

Having approximated the parameter-set, Bayes' rule (1.9) can be applied again, using (1.18) and (1.19), to define a generative classifier, which predicts the distribution over the class labels given an unseen signal $\mathbf{x}_i^*$ [25],

$$p(y_i^* = k \,|\, \mathbf{x}_i^*,\ \boldsymbol{\theta}) = \frac{p(\mathbf{x}_i^* \,|\, y_i^* = k,\ \boldsymbol{\theta})\, p(y_i^* = k \,|\, \boldsymbol{\theta})}{p(\mathbf{x}_i^* \,|\, \boldsymbol{\theta})} \tag{1.20a}$$

$$\boldsymbol{\theta} \triangleq \{\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\lambda}\} \tag{1.20b}$$

$$p(\mathbf{x}_i^* \,|\, \boldsymbol{\theta}) \triangleq \sum_{k=1}^{K} p(\mathbf{x}_i^* \,|\, y_i^* = k,\ \boldsymbol{\theta})\, p(y_i^* = k \,|\, \boldsymbol{\theta}) \tag{1.20c}$$

(Details behind this intuition are provided in Chapters 2, 4.) The predicted label is the most likely value of Y given the observation $\mathbf{x}_i^*$, i.e. the MAP

[3] Using MAP estimates, rather than the full posterior-distribution, implies that the models are *not* fully-Bayesian

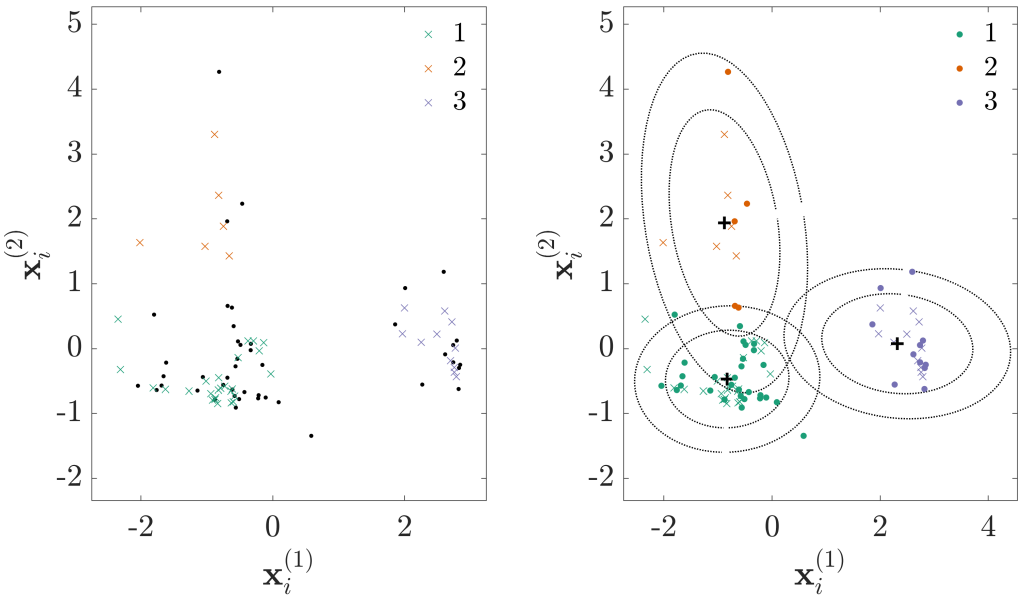


Figure 1.5: Multi-class classification of the AE data: (a) Observations in the feature space, $\times$ markers show the training set and $\bullet$ markers show the test-set. (b) Model predictions; the maximum *a posteriori* (MAP) estimate of the mean (+), covariance (dotted lines represent 2 and 3 sigma), and label predictions.

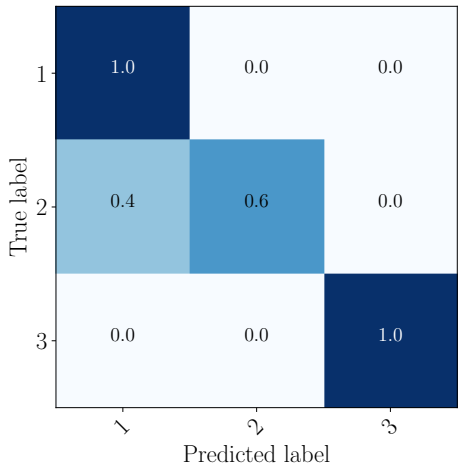


Figure 1.6: Confusion matrix for the predicted labels given the AE test data.

estimate from the posterior-distribution of the classifier (4.3),

$$\hat{y}_i^* = \text{argmax}_k \{p(y_i^* = k \,|\, \mathbf{x}_i^*)\} \tag{1.21}$$

Label predictions for the AE test data are shown in Figure 1.5b. Specifically, the test-data $\mathbf{x}_i^*$ ($\bullet$ markers) are the remaining 50% of the total data, that were not used to learn/train the parameters.

The performance of the classifier in Figure 1.5b can be assessed in various ways, all of which involve comparing the predicted labels $\hat{y}_i^*$ to the known (but hidden) *ground truth* labels. The most simple (and interpretable) metric is the classification accuracy; intuitively, this is the percentage of correctly classified signals, given the test-set[4]. For example, for the model in Figure 4.2b, the classification accuracy given the test data is 95.65%. To further investigate the predictive performance, a *confusion matrix* can be considered, plotted in Figure 1.6; the fractions of correctly classified data (for each class in $\mathcal{Y}$) are shown along the matrix diagonal, and the fractions of misclassified data are in the corresponding off-diagonal elements. Figure 1.6 illustrates that classes 1 and 3 have been correctly classified, while class 2 is confused with class 1 (not class 3, however): this is unsurprising, considering classes 1 and 2 are relatively mixed in the feature-space $\mathcal{X}$, shown in Figure 1.5b.

[4]Conversely, the classification *error* is the percentage of *misclassified* signals.

1-5. Motivation: Outstanding Challenges in Data-driven SHM

Referring back to Rytter's hierarchy, SHM should look to classify damage, following detection [18]. As such, it is desirable to categorise measured signals during operation, within a framework that considers multiple classes, relating to operational, environmental, and damage conditions [5, 20]. Training an appropriate multi-class classifier typically requires comprehensive and labelled measurements — as demonstrated with the AE data. This implies that measured data are available, corresponding to each of the expected conditions/classes, while the system has been regularly inspected, to provide descriptive labels.

A large body of the SHM literature presents the successful application of conventional supervised methods, e.g [9, 20, 29, 30], as these frameworks assume that sufficient sets of labelled data are available during training (either *a priori* or during operation). In certain applications, however, labelled data are initially unavailable (or limited), while further inspections of the system prove to be expensive [7]. For example, it is economically impractical to damage high-value systems *a priori*, in order to collect training-data that might relate to damage conditions. Environmental and Operational Variables (EOVs [12]) are also difficult to account for in the training data; these include signals that relate to temperature effects, variable loading, or variable boundary conditions. Finally, while measurements may be easy to collate in practice, comprehensive labelling is rare, as each label requires an inspection, often manually, and at a high-cost [13, 14]

Considering an offshore wind-turbine, it is infeasible to artificially damage this high-value asset to collect signals that might relate to the damaged structure *in situ*. Furthermore, the collection of EOV data *a priori* is problematic; for example, these signals might correspond to abnormal wave-loading or unusually low-temperatures — it could take several years before these measurements become available. In terms of labelled data, regular inspections of the turbine in operation can be impracticable: conventionally it is only possible to check various components (such as the turbine blades) following manual inspection; this involves travelling to a remote offshore location — a high-cost procedure.

Therefore, in the author's opinion, one of the most significant challenges

for data-based SHM is a lack of comprehensive data — more specifically, a lack of labels [6, 7, 14]. In certain applications, this missing information can force a dependence on unsupervised techniques during training, limiting SHM to damage detection. If an alternative approach to multi-class classification can provide accurate predictions given a *limited budget* of labelled data — while learning, adapting and updating online — such signal-processing methods should bring significant advances to SHM.

1-5.1. Contribution

This work suggests the use of *partially-supervised* machine learning tools, to work towards multi-class classification, given *limited labelled data*. Specifically, the suggested methods work towards the following[5].

1. The strategy should be **adaptive**, incorporating any new classes (novel data-groups) as they are discovered, during system operation.
2. Therefore, the algorithm should be capable of learning and updating **online**; that is, it should be computationally-efficient, to update/adapt during system operation.
3. Model predictions should enable **accurate diagnostics** (ideally under uncertainty) while using a **limited** number of **labelled data**.

[5] An outline of the thesis is provided at the end of Chapter 2.

2

TOWARDS PROBABILISTIC AND PARTIALLY-SUPERVISED SHM

Overview: The concepts of partially-supervised learning are introduced though visual examples, with reference to the technical sections of this work.

2-1. Probabilistic Classifiers

Before the introduction of partially-supervised learning, two perspectives of probabilistic classification are provided, as they are referenced throughout and used to (approximately) sub-categorise the associated methods.

2-1.1. Generative Approach

The first method for building a probabilistic classifier, introduced via the AE data, involves creating a joint-distributed model, of the form,

$$p(y_i, \mathbf{x}_i) = p(\mathbf{x}_i \mid y_i)\, p(y_i) \tag{2.1}$$

This is then conditioned (1.8) on the observed features $\mathbf{x}_i$, to provide the posterior-distribution over the class labels, i.e. Bayes' rule (1.9)[1],

$$p(y_i \,|\, \mathbf{x}_i) = \frac{p\left(\mathbf{x}_i \,|\, y_i\right) p\left(y_i\right)}{p(\mathbf{x}_i)} \tag{2.2}$$

This is a *generative classifier*, since it is possible to sample (generate) observed features for each class [25]: first, the class label y_i can be sampled from the prior-distribution $p(y_i)$, and then a feature-set $\mathbf{x}_i$ can be sampled from the likelihood, given the label, $p\left(\mathbf{x}_i \,|\, y_i\right)$. Generative methods for partially-supervised SHM are presented in Chapters 4, 5, 6.

Advantages: Prior-knowledge of the structure of the data in $\mathcal{X}$ can be naturally incorporated into generative models, via the likelihood $p(\mathbf{x}_i \,|\, y_i)$ [32]. For example, if measured signals are expected to present uni-model clusters in the feature-space, as with the AE data, the Gaussian-distribution (1.15) *might* be suitable to (at least) approximate the likelihood of the measurements given each class.

Disadvantages: The generative approach does not directly target the classification model $p(y_i \,|\, \mathbf{x}_i)$, since it is more focussed on density estimation, i.e. modelling $p(\mathbf{x}_i \,|\, y_i)$ [32]. If the underlying distribution of the data in the feature-space is complex (e.g. multi-modal, disjoint class-clusters), finding a suitable likelihood for $p(\mathbf{x}_i \,|\, y_i)$ can be problematic [32].

2-1.2. Discriminative Approach

An alternative approach models the conditional probability $p(y_i \,|\, \mathbf{x}_i)$ directly. This is a *discriminative classifier*, as it can discriminate between labels for a given observation, but it cannot generate examples [25] (there is no way to sample $\mathbf{x}_i$). Discriminative methods can be interpreted as modelling the *decision-boundary* between classes directly (visualised in Figure 2.1), rather than the underlying distribution of the data — as with the GMM in Figure 1.5b [32].

[1] Equivalent to the classifier (4.3) for the AE data.

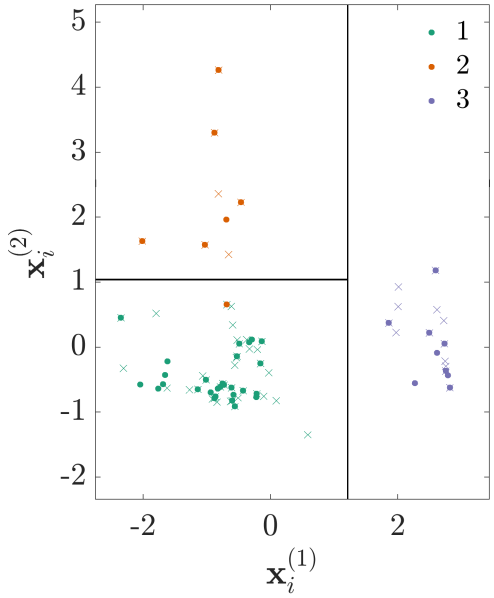


Figure 2.1: Discriminative classification of the AE data: decision-boundaries are shown by the solid black line, × markers show the training (and validation) set, and • markers show the test-set.

Advantages: Directly modelling $p(y_i \,|\, \mathbf{x}_i)$, rather than the class-conditionals, can be simpler — particularly if the *decision-boundary* between classes has a simple form, while the distribution of data is complex [32]. Intuitively, a compelling argument states that the classification problem should be modelled directly and simply, without attempting to solve a more complex (and general) problem as an intermediate step [45] — i.e. modelling $p(\mathbf{x}_i \,|\, y_i)$ [46]. This concept is visualised in Figure 2.1: here, the AE data-groups can be classified with simple, linear decision-boundaries[2].

Disadvantages: Discriminative methods are *black-box* classifiers (in the machine learning sense [25, 32]); as such, prior domain-knowledge of $\mathfrak{X}$ is difficult to include in the model. Furthermore, unlike the generative case (where the parameters of the class-conditionals are learnt independently) if a novel class is discovered, the whole model must be retrained [25]. This is significant for applications of machine learning where online training is required (which can

[2]This is a Decision Tree for classification, tree-based methods are introduced in Chapter 3.

be typical SHM). In this setting, streaming data imply that the multi-class problem may change, such that algorithm retraining is undesirable.

2-2. Partially-supervised Learning

While fully labelled data are infeasible in certain applications of SHM, it is often possible to include labels for a limited set (or *budget*) of measurements. Generally, the label budget is limited by some expense incurred when investigating the signals; this might include direct costs, associated with inspection, or loss of income due to down-time [14].

When working with limited labelled data (alongside unlabelled data), it is illogical to apply supervised learning, while ignoring the information in a (potentially large) set of unlabelled measurements. Similarly, it is unjustified to ignore the labelled data, which contains information relating to the underlying physics, to apply unsupervised algorithms. In this scenario, partially-supervised learning [39] becomes relevant to SHM; these algorithms offer an alternative approach to multi-class classification, as they utilise *both* labelled $\mathcal{D}_l$ (1.17) and unlabelled signals $\mathcal{D}_u$ (1.14), such that the training-set is,

$$\begin{aligned} \mathcal{D} &= \mathcal{D}_l \cup \mathcal{D}_u \\ &= \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \cup \{\tilde{\mathbf{x}}_i\}_{i=1}^{m} \end{aligned} \tag{2.3}$$

In other words, partially-supervised learners look to combine and exploit the information in labelled and unlabelled signals, within a unifying training scheme [39]. Two of the main approaches to the partially-supervised problem are *semi-supervised* [47] and *active* learning [48] — generally, this work concerns classifier-based variants of these algorithms.

2-2.1. Semi-supervised Learning

Semi-supervised learning utilises both the labelled and unlabelled data to inform the classification mapping, $f : \mathcal{X} \mapsto \mathcal{Y}$. Typically, a semi-supervised learner will use information in $\mathcal{D}_u$ to further update/constrain the classifier learnt from $\mathcal{D}_l$.

Unlabelled data can be incorporated in various ways. The most simple and intuitive approach, *self-labelling* [47, 49], trains a classifier using $\mathcal{D}_l$, and

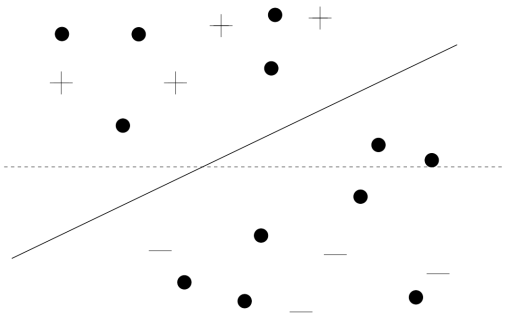


Figure 2.2: The maximum-margin decision-boundary for a two-class problem; +/− markers show positive/negative examples, while • markers show unlabelled instances. The dashed-line shows the decision-boundary given the labelled data only; the solid-line shows the boundary given both labelled *and* unlabelled instances. Image credit: [50].

then predicts the labels for the unlabelled signals $\tilde{\mathbf{x}}_i$. The classifier is then retrained using the labelled and unlabelled data. In the new training-set, some labels in $\mathcal{D}$ are the ground truth, from the supervised data, and the others are *pseudo-labels*, predicted by the classifier. Self-labelling is simple and can be applied to any supervised algorithm; however, the effectiveness is highly dependent on the method of implementation, and the supervised algorithm within it [47].

Discriminative methods

A more defined perspective considers *low-density-separation* [47]; this assumption implies that the decision-boundary of a classifier lies in low density regions of the feature-space; as such, the distances between the decision-boundary and its closest points in $\mathcal{X}$ are maximised. The use of a maximum-margin algorithm, such as the Support Vector Machine (SVM) [35], is most common in this setting; for example, the Transductive SVM (TSVM) [50] uses both the labelled data and the unlabelled data to maximise the margin of the classifier — through iterative self-labelling steps. Figure 2.2 visualises how unlabelled data can be used to maximise the margin about a linear decision boundary.

More recent developments in the literature include *graph-based* learners [51, 52]; these are discriminative methods [53], which involve building a graph where the nodes represent observed data (labelled and unlabelled), and the

edges represent the similarities between observations [54]. Here, the graph is used to represent the data on a manifold: a low-dimensional (nonlinear) embedding of the data, within the high-dimensional feature-space. As such, the *manifold* assumption is relevant here: '*the (high-dimensional) data lie (roughly) on a low-dimensional manifold*' [47]. Conveniently, the manifold assumption addresses the curse-of-dimensionality [32], which leads to an increasingly sparse feature-space in high dimensions; in this setting, statistical learning and density estimation (via generative methods) become problematic. Generally, graph-based methods inform semi-supervised learning through the smoothness assumption (for supervised learning) applied to the manifold: if two observations are close in a high-density region, they are likely to share the same label [47]. In view of this, the graph structure can be used to propagate labels from the labelled signals to the unlabelled instances.

Generative methods

Generative mixture models provide an alternative framework to incorporate unlabelled data [55, 56]. Specifically, generative-methods apply the cluster assumption: '*if points are in the same cluster, they are likely to be of the same class*[3]' [47]. As discussed, when following this approach to density estimation [32], a mixture of base-distributions are used to estimate the underlying distribution of the data, defined by $p(\mathbf{x}_i, y_i)$. Generative models can naturally account for labelled and unlabelled data, as the Expectation Maximisation (EM) algorithm (used to learn mixture models in the unsupervised case [25], explained in Chapter 5) can be modified, realtively simply, to incorporate labelled data [56, 57]. Furthermore, as knowledge of $\mathcal{X}$ can be incorporated by modelling it, *a priori* information can be included in many engineering applications, where knowledge of the data-stucture is available. However, if the assumptions of the generative model prove to be unreasonable (e.g unsuitable base-distributions), the structure imposed by the model can decrease the predictive accuracy. Figure 2.3 demonstrates how the GMM learnt with the AE data can be improved by considering the available unlabelled examples — this information is incorporated via the EM

[3]Note, this does not necessarily imply that each class is represented by a single, compact cluster in the feature-space; instead, it implies that observations from different classes are unlikely to appear in the same cluster [47].

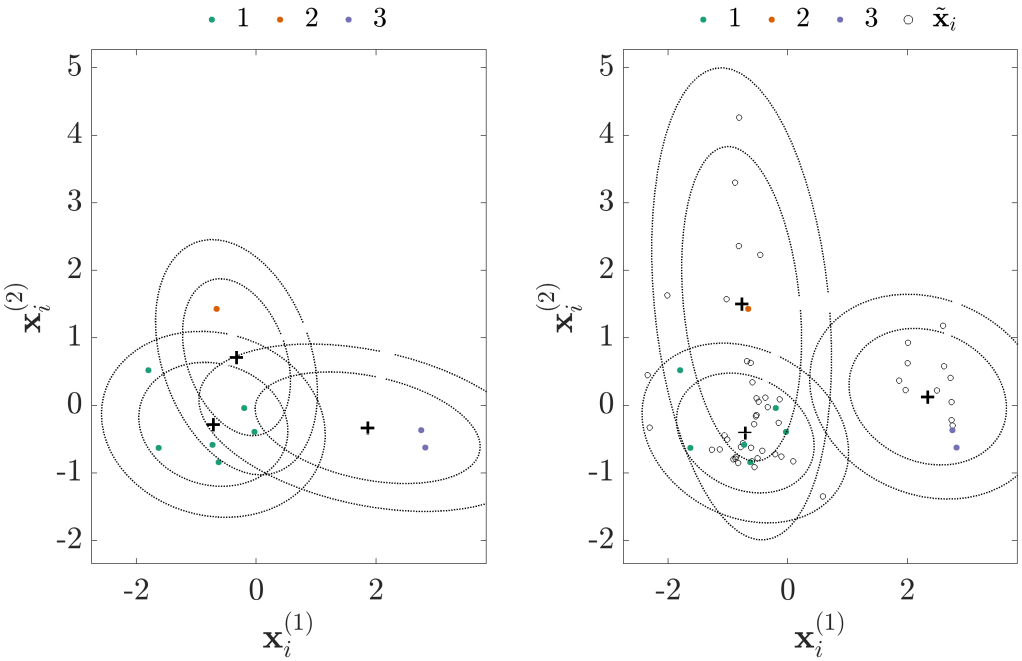


Figure 2.3: Semi-supervised GMM for the AE data: (a) supervised learning, given the labelled data only, • markers. (b) semi-supervised learning, given the labelled *and* unlabelled data, •/○ markers.

algorithm, introduced in Chapter 5.

2-2.2. Active Learning

The key hypothesis behind active learning states that an algorithm can provide improved performance, using fewer training labels, if it is allowed to select the data from which it learns [48]. Conventionally, training-data are selected by a random-sample, i.e. *passive learning*. As with semi-supervised techniques, the leaner utilises $\mathcal{D}_l$ and $\mathcal{D}_u$ — however, active techniques query/annotate the unlabelled data in $\mathcal{D}_u$ to extend the labelled dataset $\mathcal{D}_l$. Specifically, an active classifier looks to define an accurate mapping, $f : \mathcal{X} \mapsto \mathcal{Y}$, while keeping queries to a minimum [58]; the general (and simplified) steps are illustrated in Figure 2.4.

The key step for active algorithms is how to select the most informative

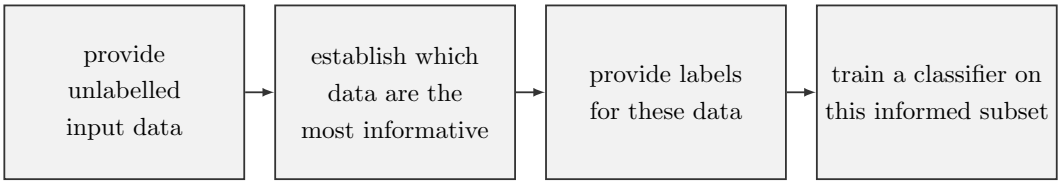


Figure 2.4: The general active learning heuristic.

signals to investigate [39, 59]. Generally, two frameworks can be used to direct queries [58–60], which are summarised below.

Classification-based

Several query regimes are based on supervised classification algorithms [59, 61], both discriminative and generative. Typical examples include query by committee and uncertainty sampling [48, 62].

Query by committee (QBC) approaches build an ensemble/committee of classifiers using a small, initial (random) sample of labelled data, leading to multiple predictions for unlabelled instances. Observations with the most conflicted label predictions are viewed as informative, thus, they are queried [59]. QBC methods can be conceptualised as a search through hypothesis space [48]. (The hypothesis space is used to describe the set of possible boundaries that a classifier can take, while the version space is a subset of these hypotheses, consistent with the labelled data seen so far [25] — as in Figure 2.5.) As more labels are observed by the learner, the set of plausible hypotheses will shrink, restricting the current version space [25]. Following QBC methods, observations who's labels explicitly shrink the version can be selected [60, 61] — in other words, data that lie in/near the shaded region of Figure 2.5. In order to implement a QBC query framework, it must be possible to: construct a committee of models that represent different regions of the version space; have some measure of disagreement among committee members, to direct queries [48].

Alternatively, *uncertainty-sampling* frameworks build a single classifier (either discriminative [61] or generative [48]) where signals corresponding to the *least confident* label predictions are queried. Uncertainty sampling is (perhaps) most interpretable applied to probabilistic algorithms, as the

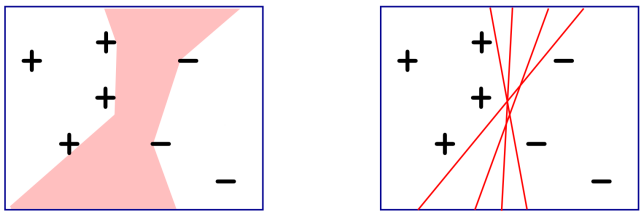


Figure 2.5: Left: version space for a binary linear classifier (shaded).
Right: some of the plausible hypotheses/classifiers (f) in the current version space. Image credit: [58].

posterior-probability over the class-labels $p(y_i \,|\, \mathbf{x}_i)$ can be used to quantify uncertainty/confidence. For example, consider a binary (two-class) problem: intuitively, uncertain samples could be instances whose posterior probability is nearest to 0.5 for both classes. This view can be extended to multiple (> 2) classes using the *Shannon entropy* [63] as a measure of uncertainty; for example — uncertain signals (based on high entropy) given the GMM of the AE data are illustrated in Figure 2.6. Conveniently, uncertainty sampling can be applied to *semi-supervised* mixture-models with little modification, combining *both* partially-supervised methodologies. Applications of this approach are presented in Chapters 4, 6.

Cluster-based

Alternatively, active-learning can exploit the (unsupervised) cluster structure in data to direct queries [60, 64, 65]. A typical example of cluster-based sampling, introduced by Dasgupta and Hsu [58], starts with a hierarchical-clustering of the unlabelled data, which divides the feature-space into many partitions. An informative set of training data is built by directing queries to areas of the feature-space that appear mixed in terms of labels (as sampling proceeds), while clusters that appear homogeneous are queried less. Conveniently, queried labels can be propagated to any remaining unlabelled instances in $\mathcal{D}_u$, using the cluster structure and a majority vote. As a result, (like generative models) this method can also become semi-supervised [60]. Figure 2.7 visualises hierarchical sampling and label propagation for clustered data. Following selection of the training-data, any conventional supervised classifier can be applied to learn the classification mapping $f : \mathcal{X} \mapsto \mathcal{Y}$.

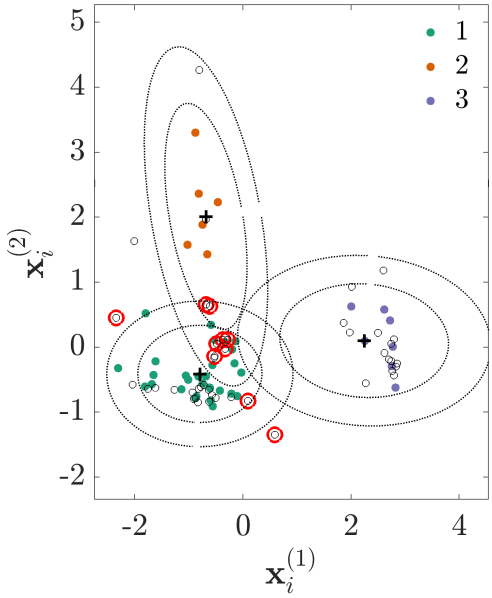


Figure 2.6: Uncertainty sampling (based on entropy) for the AE data: • markers show the training set, and ∘ markers show the unlabelled data — red circles indicate queries by the active learner.

Methods for cluster-based sampling are different to classifier-based frameworks, as the training-data are queried *before* learning any classifier. Therefore, the unsupervised clustering of the data must be defined independently (to direct queries), while the classification mapping is learnt afterwards. (Alternatively, the cluster-structure and a majority vote can be used for classification [65].) Cluster-based active learning is applied in Chapter 3, concerning hierarchical-sampling for active learning [60].

2-2.3. The Dangers of Partially-supervised Learning

While the intuition behind active and semi-supervised methods appears logical, care must be taken, as the performance of partially-supervised algorithms can prove to be worse than conventional (passive) learning [53, 55].

During active learning, if queries are too focussed on specific definitions of 'informative', the training-data can become poorly representative of the underlying distribution; this phenomenon is referred to as *sampling bias* [48, 66].

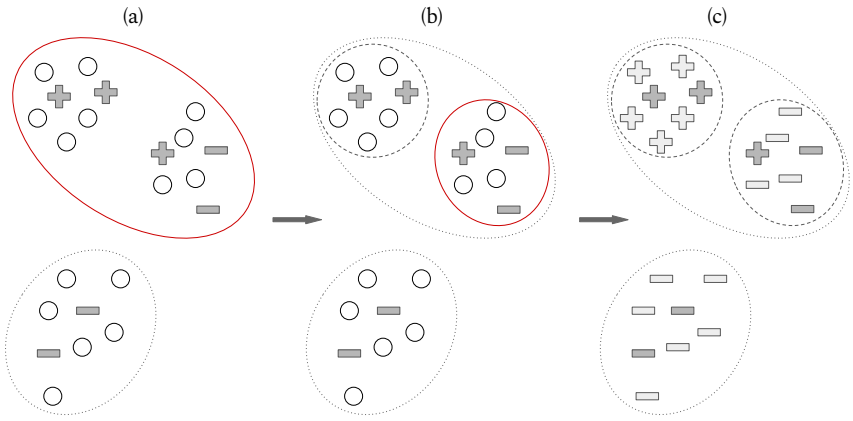


Figure 2.7: Unsupervised clusters at different resolutions, $+/-$ markers show positive/negative examples, while $\circ$ markers show unlabelled instances. (red) solid-outlines show clusters that appear mixed; as such, the associated data groups would be queried by the learner: (a) initial clusters, (b) clusters at an increased resolution, (c) label propagation (majority vote).

As such, queries should not focus too much on specific regions of the feature-space; for example, the version space around the decision boundary. To avoid the issues of sampling bias, variation can be achieved by combining different definitions of informative [66]. The issues of sampling bias are discussed (an visualised) in detail in the experimental Chapters 3, 4.

On the other hand, during semi-supervised learning, incorporating unlabelled signals has the potential to decrease the predictive performance, if the structure imposed by classifier proves inappropriate [47]. This can be particularly problematic for generative methods [55] — a caveat investigated in Chapters 5, 6.

2-3. Thesis Layout

A brief outline of each chapter is provided below. The work progresses while adapting algorithms for partially supervised SHM, considering each of the issues outlined in the *Contributions*, Section 1-5.1.

Ch. 3 *Hierarchical Sampling for Active Learning*: The application of cluster-based active learning to SHM data. Experiments demonstrate the advantages of partially-supervised learning, based on a nonparametric and discriminative method, which is trained offline.

Ch. 4 *Probabilistic Active Learning for Online SHM*: Introduces generative mixture models for probabilistic active-learning, via uncertainty-sampling. The suggested parametric algorithm (a GMM) can learn, update and adapt online, to classify streaming SHM data. A novel tool for unsupervised feature extraction from vibration data is also introduced.

Ch. 5 *Towards Probabilistic and Semi-supervised Damage Classification*: Extension of the GMM to utilise unlabelled signals through Expectation Maximisation; this is shown to improve the quality of the mixture model (while training off-line), and improve the diagnostic performance of the classifier.

Ch. 6 *Towards a Combined Semi-supervised and Active Learning Learner*: Combines active and semi-supervised learning methods for the GMM, introduced in Chapters 4 and 5, for an algorithm that can adapt and update online with streaming SHM data.

Ch. 7 *Conclusions*: Concluding remarks and future work.

3

HIERARCHICAL SAMPLING FOR ACTIVE LEARNING

Overview: Dasgupta's and Hsu's cluster-based active-learner (the DH algorithm) is applied to experimental SHM data from ground vibration tests of a Gnat aircraft. Results demonstrate the potential advantages of active and semi-supervised learning in SHM applications — in this case, based on a non-parametric and *discriminative* method. In this setting, the algorithm is trained *offline*, using unlabelled data that were collected *a priori*.

Firstly, the cluster-based approach is explained in detail, while considering the issues of sampling bias. The DH algorithm is then introduced for hierarchical sampling, and the algorithm is applied to SHM data. The advantages and limitations of this approach are discussed in the concluding remarks of the chapter.

3-1. Cluster-based Methods and Sampling Bias

At the risk of repetition, various cluster-based methods follow a similar framework, formalised by Dasgupta and Hsu [58]. In an ideal scenario, separable clusters will exist that are pure in terms of labels. Following definition by unsupervised learning, a few informative points $\tilde{\mathbf{x}}_i \in \mathcal{D}_u$ can be queried from each cluster to define a labelled set $\mathcal{D}_l$, and any remaining unlabelled points in $\mathcal{D}_u$ can than be assumed to have their most confident (majority) label [59, 60, 65],

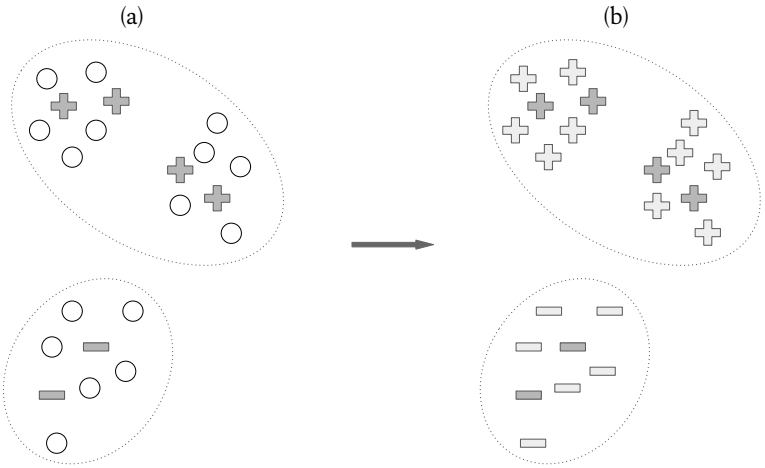


Figure 3.1: Ideal clusters (separable and pure): (a) clustering of query points $+/-$ and unlabelled instances $\circ$; (b) query points $\mathcal{D}_l$ (dark grey) and propagated labels $\mathcal{D}_u$ (light grey).

as in Figure 3.1. (Throughout this chapter, this approach is referred to as *label propagation*.) A supervised classifier can then be trained on the complete dataset $\mathcal{D}$, including queried and propagated labels, i.e. $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$. The label propagation steps are typical of *semi-supervised* learning [39], such that unlabelled instances in $\mathcal{D}_u$ are used to constrain the classifier by assuming their labels. Intuitively, the ability to naturally incorporate unlabelled data brings further benefits to cluster-based active learning, normally associated with semi-supervised algorithms [47].

The active/guided sampling element of cluster-based techniques is defined by the sampling procedure; various methods have been proposed. Dasgupta and Hsu suggest a algorithm that favours instances from clusters that appear most mixed as querying progresses [60]. Alternatively, the density clustering algorithm, by Wang et al. [59], favours queries in regions populated by (relatively) dense groups of data. In this chapter, queries are directed to areas of the feature space that appear to be most mixed in terms of labels, as these clusters are assumed the most informative to both the cluster structure and final classification.

In reality, the ideal case shown in Figure 3.1 is rare. The relationship between labels and clusters could be insignificant, or there might be viable

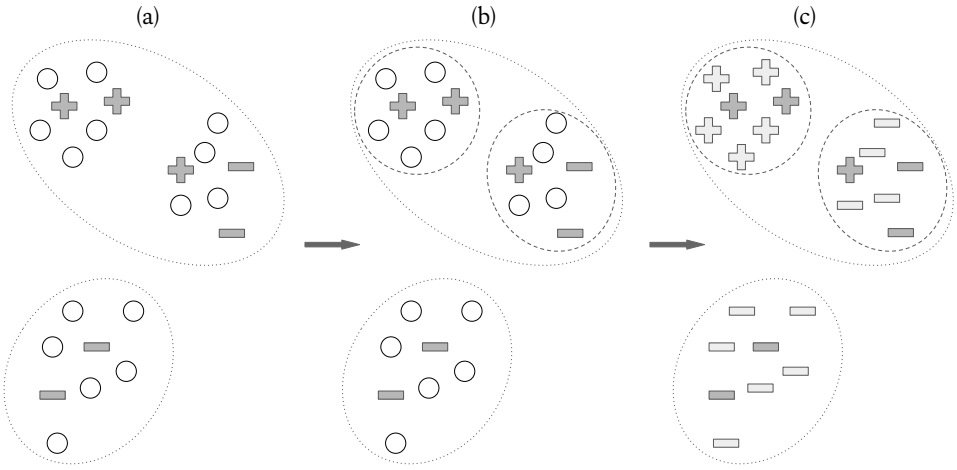


Figure 3.2: (a),(b) Identification of viable clusterings at different resolutions; (c) label propagation by majority vote.

(near pure) clusters but at many different resolutions [60] — as in Figure 3.2. For this reason, the performance of cluster-based methods depends critically on the quality of the clustering results [59, 66]; thus, the data clusters must be adaptive — actively changing as more information becomes available. Provided that there is some relationship between clustered groups of data and diagnostic labels, at whatever resolution, cluster-based active learning should exploit these patterns [58, 59].

3-1.1. Sampling Bias

As discussed in the introductory chapters, selecting specific observations can focus too much on certain regions of the feature-space (e.g. areas close to the decision boundary, or far away from cluster centres). This can neglect alternative regions that might be more representative of the underlying data distribution [61]. In consequence, while active learning has been shown to bring significant empirical advantages in the literature [48, 60, 65, 66], the author wishes to reiterate that there are times when selecting training data by a given measure (uncertainty or otherwise) can be worse than random sampling.

Specifically, the assumption of most classifiers, and data-based models in

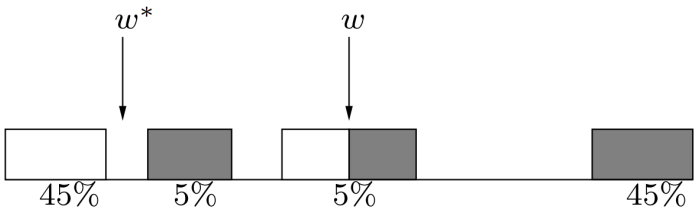


Figure 3.3: One-dimensional classification problem to demonstrate sampling bias. Image credit: [60].

general, is that the training data are representative of the underlying data distribution; this implies that samples are drawn independent-and-identically-distributed (i.i.d) from the underlying probability density [48]. While the underlying dataset might remain i.i.d, during active learning, the samples that define the training data are *guided*. Therefore, the data used to train the algorithm are inherently *not* i.i.d. As a result, care must be taken to ensure that the model does not become misrepresentative. For this reason, it is critical that any application of active learning to engineering data should consider: the type (complexity) of data that is being analysed, the quantity of data that is available, and the query budget.

To visualise sampling bias, consider the one-dimensional example in Figure 3.3, presented by in [60]. The data lie in four groups, and the classifier f_ω used to separate them is defined by some threshold value, $\omega \in \mathbb{R}$. The proportion of the dataset in each group is given by a percentage. Grey blocks have a $\{1\}$ label, and white blocks have a $\{0\}$ label. Most of the data lie in the two most external groups; therefore, a small, initial random sample has a high likelihood of coming from these. In this case, the initial classifier, denoted f_ω in (3.1), would lie somewhere between the two external groups shown in Figure 3.3.

$$f_\omega(x) = \begin{cases} 0 & x < \omega \\ 1 & x > \omega \end{cases} \tag{3.1}$$

As active learning proceeds, selecting uncertain observations, the classifier would most likely converge to ω, in the centre of Figure 3.3. However, the classifier ω has 5% error, while ω^* has only 2.5% error [60]. This occurs as

the most probable initial sample is poorly representative of the underlying distribution in the data [58]. It includes no observations in the second group from the left (5% grey block), and as a result, this group is overlooked; therefore, the learner is mistakenly confident that these data have a $\{0\}$ label [60]. In other words, this group *hides* behind the decision boundary ω due to a poorly placed initial classifier f. This example presents just one-dimension, in higher dimensions the problem can get worse, as there are more spaces for groups of data to hide [60].

To mitigate sample-bias with classifier-based methods, sampling can systematically include *representative* observations (i.e. those far away from the version space) as well as uncertain observations [66]. Several methods have been suggested; typical algorithms, such as the pre-clustering algorithm by Nguyen and Smeulders [65], or the QUIRE algorithm by Huang et al. [66], combine an unsupervised clustering with the classification algorithm. This leads to a hybrid framework, where a balance of uncertain observations (close to the decision boundary) and representative observations (near cluster centroids) are selected.

Alternatively, cluster-based frameworks [58, 60, 65] look to automatically mitigate sampling bias by querying across the entire cluster structure, even after a poorly representative initial sample. As discussed, the general cluster-based framework completely removes the classifier from the active learning steps; thus, the methods should prevent the learner from being constrained by an ill-informed hypothesis. In consequence, considering the issues of sampling bias, as well as the benefits associated with label prorogation, this chapter applies the DH algorithm as a cluster-based variation of active learning.

3-2. A Cluster-based Framework for Guided Sampling

The DH algorithm is an active learning tool proposed by Dasgupta and Hsu [60]. The method utilises a cluster-adaptive framework for guided sampling and label propagation, which is clearly defined in the original papers [58, 60]. Each stage of the algorithm is also explained here, with some slight differences in implementation — specifically, in Section 3-2.6.

3-2.1. Clustering

The DH learner starts with a hierarchical clustering of the input data (initially all the observations are unlabelled, i.e. $\mathcal{D}_u = \mathcal{D}$). In the experiments here, agglomerative clustering is used; this clustering algorithm works by sequentially joining groups of signals in the feature-space. Initially, it compares K groups, each containing one observation; i.e. $K = m$, as there are m observations in the unlabelled set $\mathcal{D}_u = \{\tilde{\mathbf{x}}_i\}_{i=1}^{m}$. At each step, the dissimilarity matrix d is assessed using (3.2) and (3.3) and the two most similar groups are merged, until there is a single cluster containing all the data, s.t. $K = 1$ [25].

Specifically, the dissimilarity between single data points is calculated using the Euclidean distance,

$$d(\mathbf{x}_i, \mathbf{x}'_i) = \sqrt{\sum_{j=1}^{D} (x_i^j - x_i'^j)^2} \tag{3.2}$$

(where superscript j is used to index the j^{th} feature from the vector $\mathbf{x}_i$), and the dissimilarity between groups of data is assessed with Ward's average linkage,

$$d_{r,s} = \sqrt{\frac{2m_r m_s}{m_r + m_s}} \times d(\bar{\mathbf{x}}_r, \bar{\mathbf{x}}_s) \tag{3.3}$$

where m_s and m_r are the number of data in groups r and s respectively, while $\bar{\mathbf{x}}_r$ and $\bar{\mathbf{x}}_s$ are the cluster centroids. Pseudocode for the agglomerative clustering algorithm is provided in Algorithm 1 [25].

The merging process can be represented with the use of a binary tree $\mathbf{T}$, called a dendrogram, illustrated in Figure 3.4. The initial groups (single observations) are represented by the leaves of the tree, at the bottom of the graph. Each time two groups are merged they are joined in the tree at a node u. The tree $\mathbf{T}$ can be defined as a set of nodes, $\mathbf{T} = \{u_i\}_{i=1}^{m+m-1}$ (including leaves); the height of branches represents the dissimilarity between two respective groups [25]. The root of the tree, at the top of the dendrogram, represents one group containing all the data.

If the tree is cut at any given height, a clustering is induced for a given number of groups K. For example, if the tree in Figure 3.4 was cut at height 2.5, this induces a clustering where $K = 2$, with groups: $\mathbf{\{}\{4,6\},\{2,5\}\mathbf{\}}$, $\mathbf{\{}1,3\mathbf{\}}$.

Algorithm 1: *Agglomerative clustering*

Input : Unlabelled data $\mathcal{D}_u = \{\tilde{\mathbf{x}}_i\}_{i=1}^m$
Output : Clustering structure $\mathbf{T}$

Compute dissimilarity matrix d between all observations in $\mathcal{D}_u$;
Initialise clusters as single observations: $\mathbf{T} = \{u_i, \ldots, u_m\}$,
  s.t. **for** $i \leftarrow 1 : m$ **do** $u_i \leftarrow \{i\}$;
Initialise set of clusters available for merging: $S \leftarrow \{1, ..., m\}$;
while *clusters are available to merge in* S **do**
  Pick the two most similar clusters to merge:
  $(j, k) \leftarrow \text{argmin}_{j,k \in S}(d_{j,k})$;
  Create new cluster $u_l \leftarrow u_j \cup u_k$;
  Mark j and k as unavailable: $S \leftarrow S \setminus \{j, k\}$;
  if $u_l \neq \{1, ..., m\}$ **then**
    Mark l as available, $S \leftarrow S \cup \{l\}$;
    Upate cluster structure, $\mathbf{T} \leftarrow \mathbf{T} \cup u_l$;
  end
  for $i \in S$ **do**
    Update dissimilarity matrix $d(i, l)$;
  end
end

3-2.2. An Overview of Guided Sampling and Label Propagation

To illustrate guided sampling and label propagation, one can return to the sampling bias example presented in [60], and shown in Figure 3.5. In this case, the dendrogram represents the *top few nodes* of a hierarchical clustering; therefore, each leaf defines a group of data, rather than singleton observations: proportions of the total data in each leaf are provided.

Following hierarchical clustering, the DH algorithm will work with a particular partition of the dataset at any given time, defined by a *pruning* $\mathbf{P}$ of the tree $\mathbf{T}$. A pruning of the tree is a subset of nodes that are *disjoint* and together *cover all the data*, i.e. $\mathbf{P} \subset \mathbf{T}$. Initially, the pruning is set as the root node from agglomerative clustering, a single group containing all the data,

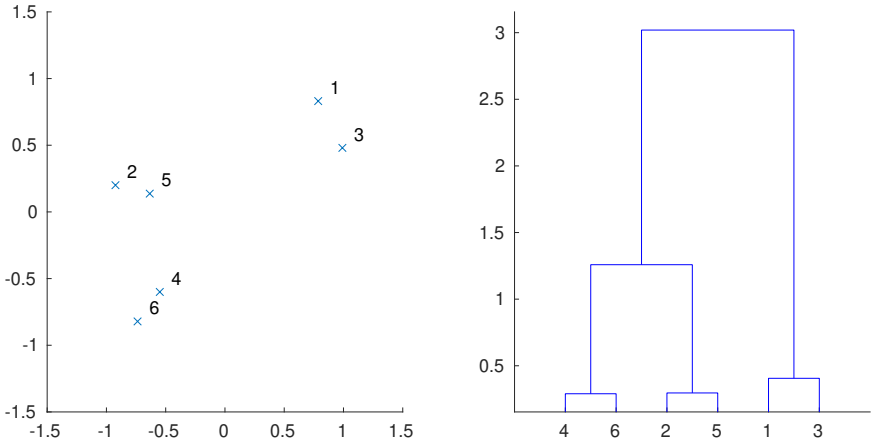


Figure 3.4: Dendrogram of hierarchical clustering, down to single observations $m = 6$.

i.e. $\mathbf{P} = \{1\}$. A small number of random points are drawn from this cluster and queried; these initial labels provide the first indication of the underlying distribution of the data, for all levels of the hierarchy. In this example, samples should reveal that the top node is very mixed, while nodes $\{2\}$ and $\{3\}$ are relatively homogeneous. Once this transpires, partition $\{1\}$ will be replaced with a pruning of $\mathbf{P} = \{2, 3\}$ [60]. The next set of observations will then be selected according to a querying strategy that favours the less pure node [60].

After further rounds of sampling, $\mathbf{P}$ would most likely be refined to $\{2, 4, 9\}$. At this stage, the benefits of cluster-based sampling become most obvious: considering the observations seen so far, it can be concluded that cluster $\{9\}$ is relatively pure, so fewer queries will be made from this group [60]; instead, future samples will be directed towards groups $\{2\}$ and $\{4\}$.

Guided sampling continues in this way, working down the dendrogram. Querying can be stopped at any stage — usually when the label budget runs out; when this is done, any remaining unlabelled data in $\mathcal{D}_u$ associated with each cluster in the final $\mathbf{P}$ are assigned their majority label, according to the queried data seen so far $\mathcal{D}_l$. In this way, the learner looks to label the entire dataset, $\mathcal{D}$, while keeping the number of erroneous (propagated) labels to a minimum [60].

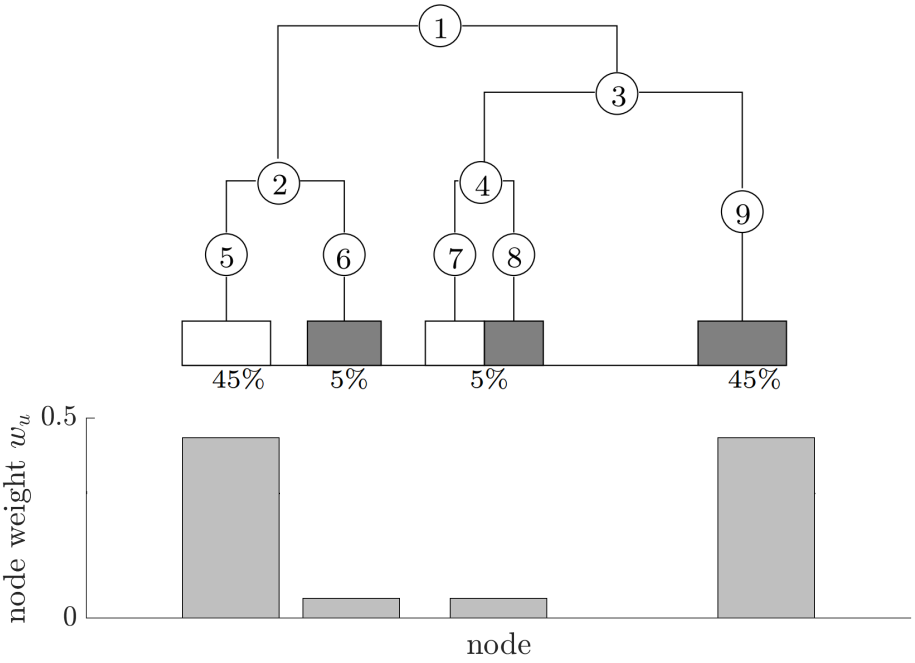


Figure 3.5: The top few levels of a hierarchical clustering. Clustered groups are shaded according to their majority label: (1) grey, (0) white. Image credit: [60]. The histogram visualised the weight of data in each leaf (i.e. node).

3-2.3. Pruning & Node Properties

For any node u in the tree $\mathbf{T}$, $\mathbf{T}_u$ denotes the subtree rooted at node u, as well as all the data contained in that node [60]. Therefore, a pruning of the tree $\mathbf{P} = \{v_1, ..., v_p\}$, is s.t. $\mathbf{T}_{v_i}$ are disjoint and together cover all the data [60]. Partial prunings are also considered when working with sub-trees; in this case, the associated leaves do not cover all the data.

The *weight* w_u of a node $u \in \mathbf{T}$ is the proportion of total data contained in the subtree of that node, where m_u is the number of data in $\mathbf{T}_u$.

$$w_u = \frac{m_u}{m} \tag{3.4}$$

The weight of a pruning $w(\mathbf{P})$ is the fraction of the total data contained in the pruning $\mathbf{P}$ [60]:

$$w(\mathbf{P}) = \sum_{v \in \mathbf{P}} w_v \tag{3.5}$$

For a complete pruning, $w(\mathbf{P}) = 1$, and for a partial pruning, $0 < w(\mathbf{P}) < 1$.

Following data queries

Having defined $\mathbf{T}$ from $\mathcal{D}_u$, the learner now starts to query data from nodes in the current pruning $\mathbf{P}$ to build the queried/labelled set $\mathcal{D}_l = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$.

For the K possible labels, i.e. $y = k \in \{1, \ldots, K\}$, the label *proportions* observed in each node u can be estimated,

$$p_{k,u} = \frac{n_{k,u}}{n_u} \tag{3.6}$$

where $n_{k,u}$ is the number of times $y_i = k$ from the queried data in u, while n_u is the total number of queries taken from node u. This is, effectively, a maximum likelihood estimate of the conditional probability distribution $p(y_i \,|\, \mathbf{x}_i)$, at each node, which represents a given area of the feature-space. Therefore, considering the definitions in Section 2-1, this is a *discriminative* approach, such that the conditional probability is *estimated* directly,

$$p(y_i = k \,|\, \mathbf{x}_i) \;\approx\; p(y_i = k \,|\, \mathbf{x}_i \in u) \;\approx\; p_{k,u} \tag{3.7}$$

Let the labelling of $\mathbf{P}$ be $\mathbf{L}$, such that the label assigned to node u is $L(u)$, where $L(u) \in \{1, 2, ..., K\}$. Intuitively, each cluster u is assigned its majority label, so $L(u) = \text{argmax}_k(p_{k,u})$. The approximate *error* induced when assigning all the data in cluster $\mathbf{T}_u$ with a label in $L(u)$ is given in (3.8) [60].

$$\epsilon_{L(u),u} = 1 - \max_k(p_{k,u}) \tag{3.8}$$

For a partial or complete pruning, the *error* introduced when assigning each cluster with its majority label is defined as [60]:

$$\epsilon(\mathbf{L}, \mathbf{P}) = \frac{1}{w(\mathbf{P})} \sum_{v \in \mathbf{P}} w_v \epsilon_{L(v),v} \tag{3.9}$$

Due to limited sampling, labels are only available in the queried nodes, and these queries are not necessarily indicative of the majority label. At a *given time*, $n_{k,u}(t)$ labels have been observed, and there has been $n_u(t)$ queries; so based on the labels seen so far, the current estimate for the label proportions is $p_{k,u}(t)$. The corresponding errors at this time are given by $\epsilon_{l,u}(t) = 1 - p_{k,u}(t)$ [60].

The *quality* of these estimates can be assessed using generalisation bounds. At any given time, the label proportion estimates can be assigned confidence intervals, denoted by superscripts $\{p_{k,u}^{LB}, p_{k,u}^{UB}\}$ [60]. The true value of $p_{k,u}$ is expected to lie within these bounds. Specifically, the confidence interval is defined using a variation of *Wald's interval* [60, 67],

$$\{p_{k,u}^{LB},\ p_{k,u}^{UB}\} = \{\max[p_{k,u}(t) - \delta_{k,u}(t),\ 0],\ \ \min[p_{k,u}(t) + \delta_{k,u}(t),\ 1]\} \tag{3.10}$$

for,

$$\delta_{k,u}(t) \approx \frac{1}{n_u(t)} + \sqrt{\frac{p_{k,u}(t)(1 - p_{k,u}(t))}{n_u(t)}} \tag{3.11}$$

3-2.4. Admissible Clusters

When pruning the tree it is useful to work down the dendrogram as far as possible [60]; in this way, clusters can be analysed at a higher resolution, so queries can be directed to specific areas of the feature space, and label propagation can be applied to more complex clusterings. To justify descending into lower levels of the hierarchy, however, the learner should first be confident about majority label estimates $L(u)$ for all nodes in the potential pruning.

Considering this, the *admissibility* $A_{k,u}(t)$ is defined to establish when and where the learner can be confident about a majority label estimate [60]:

$$A_{k,u}(t) = \text{True} \ \Leftrightarrow\ (1 - p_{k,u}^{LB}(t)) < \beta \cdot \min_{k' \neq k}(1 - p_{k',u}^{UB}(t)) \tag{3.12}$$

In words, for each cluster, a label is admissible if its (largest) expected error is at least β times less than the (smallest) expected error of any other label. For these experiments the hyper-parameter β is set to a value of 1.5, so (3.12) becomes,

$$A_{k,u}(t) = \text{True} \ \Leftrightarrow\ p_{k,u}^{LB}(t) > (1.5 p_{k',u}^{UB}(t) - 1) \ \ \forall \ \ k' \neq k \tag{3.13}$$

The set of admissible cluster-label (u, l) pairs is defined by $\mathcal{A}(t)$; at any given time there may be several labels associated with each node. The set $\mathcal{A}(t)$ is used throughout sampling to identify any new set of nodes that could make up a refined pruning — with increased homogeneity in each cluster.

Adjusted empirical error

The error estimates $\epsilon_{L(u),u}(t)$ can be inaccurate when a node has been inadequately sampled, as the learner has weak confidence about the label proportion estimates $p_{L(u),u}(t)$,

$$\epsilon_{L(u),u}(t) = 1 - p_{L(u),u}(t) \tag{3.14}$$

With this in mind, the admissibility can be used to adjust the empirical error and define a more conservative error-estimate in areas of sparse sampling [60],

$$\tilde{\epsilon}_{L(u),u}(t) = \begin{cases} 1 - p_{L(u),u}(t) & \text{if } \ (L(u),u) \in \mathcal{A}(t) \\ 1 & \text{if } \ (L(u),u) \notin \mathcal{A}(t) \end{cases} \tag{3.15}$$

In words, label proportion estimates are only valid when their cluster-label pairings are admissible. The *adjusted* empirical error is now,

$$\tilde{\epsilon}(\mathbf{L}, \mathbf{P}, t) = \frac{1}{w(\mathbf{P})} \sum_{v \in \mathbf{P}} w_v \tilde{\epsilon}_{L(v),v}(t) \tag{3.16}$$

3-2.5. The Select Procedure

The `select` procedure describes how the learner *actively* directs sampling in the current working partition ($\mathbf{P}$) of the tree. As suggested by Dasgupta and Hsu [60], the `select` procedure will favour nodes v that appear most mixed. Once a mixed node is chosen, a random sample is taken from the cluster that it represents, and the label is queried. Specifically, the select procedure is,

$$\texttt{Select } v \in \mathbf{P} \ \text{ with probability } \ \mathbb{P}(v) \propto w_v(1 - p^{LB}_{L(v),v}(t)) \tag{3.17}$$

In words, the likelihood of a node being queired is proportional to the (weighted) error associated with that node. This definition is used in the experiments; however, the procedure is flexible and can be modified according to the application.

3-2.6. Pruning Refinements

When refining the current pruning, $\mathbf{P} = \{v_i\}_{i=1}^{p}$, it is convenient to think of the process one node at a time. Therefore, for each node $v \in \mathbf{P}$, the best pruning and labelling of the associated subtree $\mathbf{T}_v$ is $(\mathbf{P}_v, \mathbf{L}_v)$. The following rule is used to define $(\mathbf{P}_v, \mathbf{L}_v)$, where $\mathbf{P}_v = \{v'_i\}_{i=1}^{p'}$:

- $(u, L(u)) \in \mathcal{A}(t)$ is defined for $v' \in \mathbf{P}_v$ and ancestors of $\mathbf{P}_v$ in $\mathbf{T}_v$.

For this implementation, while searching through $\mathbf{T}_v$ for the best pruning $\mathbf{P}_v$ (from the root node down), any new set of nodes must meet the above criteria. Additionally, any two child nodes $ch_u = \{u_{ch_1}, u_{ch_2}\}$ can only replace their parent node u if a reduction in the adjusted empirical error is observed,

$$\tilde{\epsilon}(\mathbf{L}, ch_u, t) < \tilde{\epsilon}_{L(u),u}(t) \quad \text{where} \quad \tilde{\epsilon}(\mathbf{L}, ch_u, t) = \frac{1}{w(u)} \sum_{i=1}^{2} w_{ch_i} \tilde{\epsilon}_{L(ch_i),ch_i}(t) \quad (3.18)$$

3-2.7. Label Propagation

An additional rule is added to this implementation, to prevent inconsistent performance at low query budgets ($n \ll m$). It states that label propagation to the unlabelled instances $\mathcal{D}_u$ only occurs if the number of clusters in the final admissible pruning is $\geq$ number of unique labels observed so far:

$$\text{Propagate label } L(v) \text{ in } \mathbf{T}_v \;\Leftrightarrow\; |\mathbf{P}| \geq K(t) \quad (3.19)$$

This is intuitive; for example, it is useless assuming labels for three admissible clusters across the whole data, when a total of seven classes have been observed.

3-2.8. The Algorithm

The pseudocode in Algorithm 2 summarises this implementation of the DH learner; code is also available at `https://github.com/labull?tab=repositories`.

Classification Following definition of the training-set by guided sampling, any supervised classifier can be trained using $\mathcal{D}, \mathbf{P}$ and $\mathbf{L}$. The classification algorithm is independent of the semi-supervised steps; therefore, it does not affect the *active* elements of the learner. Furthermore, as the 'no free lunch' theorem suggests [68], the performance of any algorithm is data-dependant. As a result, the choice of classifier is trivial when focussing on the partially-supervised characteristics (provided the same model is used throughout tests).

In fact, as suggested by Wang et al. [59], a classification algorithm is not necessary for cluster-based methods: future data can be classified according the final pruning $\mathbf{P}$ of the feature space and a majority vote — using the values

Algorithm 2: *Cluster-adaptive active learning*

Input : Agglomerative clustering $\mathbf{T}$, unlabelled data $\mathcal{D}_u$
Output : Pruning and labelling $\{\mathbf{P}, \mathbf{L}\}$,
semi-supervised dataset $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$

$\mathbf{P} \leftarrow \{\text{root}\}$; ▷ Initialise current pruning as the root node
$\mathbf{L} \leftarrow \{0\}$; ▷ Initialise arbitrary root label
$\mathcal{D}_l \leftarrow \{\}$; ▷ Initialise labelled data as empty set
#--- Guided Sampling ---#
for $t = 1 : B$ **do** ▷ Algorithm run budget B
  for $1 : b$ **do** ▷ Guided sampling, batch size b
    $v \leftarrow$ select($\mathbf{P}$); ▷ Select v from $\mathbf{P}$ according to (3.17)
    randomly sample $\tilde{\mathbf{x}}_i$ from $\mathbf{T}_v$; ▷ Adding to $\mathcal{D}_l$
    query $\tilde{y}_i$ and label $\tilde{\mathbf{x}}_i$, update $\mathcal{D}_l$ and $\mathcal{D}_u$; ▷ Labelled by engineer
    update $(n_u(t), p_{k,u}(t))$; ▷ For all nodes that contain $\tilde{\mathbf{x}}_i$
  end
  for *all nodes* $u \in \mathbf{T}$ **do** ▷ Compute admissibilities and errors
    update $(\mathcal{A},\ \tilde{\epsilon}_{L(u),u})$;
  end
  #--- Pruning Refinements ---#
  for *each* $v \in \mathbf{P}$ **do** ▷ Refine the pruning, node by node
    $(\mathbf{P}_v, \mathbf{L}_v) \leftarrow$ best pruning/labelling of $\mathbf{T}_v$; ▷ Re. Section 3-2.6
    $\mathbf{P} \leftarrow \mathbf{P}_v \cup (\mathbf{P} \setminus v)$; ▷ Update node v to refine $\mathbf{P}$
    $L(v) \leftarrow \mathbf{L}_v(v')$ for all $v' \in \mathbf{T}_v$; ▷ Update node labels $L(v)$
  end
end
#--- Label Propagation ---#
for *each cluster* $v \in \mathbf{P}$ **do** ▷ In the *final* pruning
  if $|\mathbf{P}| \geq K(t)$ **then** ▷ Additional rule (3.19) - compared to [60]
    propagate $L(v)$ to unlabelled data in $\mathbf{T}_v$; ▷ Label signals $\tilde{\mathbf{x}}_i$
  end
end

of $p_{k,u}$ as estimates of $p(y_i = k \,|\, \mathbf{x}_i)$. Nonetheless, a classification algorithm is applied in the experiments here, for direct comparison to conventional techniques. In consequence, the K-Nearest-Neighbour (KNN) algorithm is used as a basic nonparametric classifier, to predict the labels of test-data and provide a simple performance metric. The KNN classifier identifies the K *nearest* points in the training-set $\mathcal{D}$ to the test input $\mathbf{x}_i^*$ [25]; in this case, 15 neighbours are considered (s.t. $K = 15$), and the Euclidean-distance (3.2) is used. Given the K neighbouring points to the test input, the number of instances in each class is counted, and used to provide an empirical (maximum likelihood) estimate of the class conditional $p(y_i^* \,|\, \mathbf{x}_i^*)$; more specifically,

$$p(y_i^* = k \,|\, \mathbf{x}_i^*, \mathcal{D}, K) = \frac{1}{K} \sum_{i' \in I_K} \delta_{y_{i'},k} \tag{3.20}$$

where I_K are the set of indices for the KNNs to $\mathbf{x}_i^*$ in $\mathcal{D}$, and $\delta_{y'_i,k}$ is the Kronecker delta function — equal to unity when k is equal to the observed class label $y_{i'}$ in the set of KNNs. The predicted class label is then, $\hat{y}_i^* = \text{argmax}_k \{p(y_i^* = k \,|\, \mathbf{x}_i^*, \mathcal{D})\}$. This predicted label can be compared to the ground-truth from the test-set, to calculate the classification error e.

3-3. Experiments

3-3.1. Gnat Aircraft Data

The Gnat data are an experimental dataset, concerning the wing of a Gnat aircraft [69]. During ground vibration tests, the wing was excited using an electrodynamic shaker and band-limited white-noise. A network of sensors recorded the acceleration response at different points on the wing, shown in Figure 3.6b. The shaker was attached directly below P4 in Figure 3.6b, on the bottom surface of the wing [69]. During the experiments, artificial damage was introduced by sequentially removing one of nine inspection panels; the panels are shown in Figure 3.6a. (It is acknowledged that the removal of each panel represents a fairly large and significant fault.) The data represent a nine-class damage classification (location) problem; one class is associated with the removal of each panel. The network of sensors are split into groups A, B

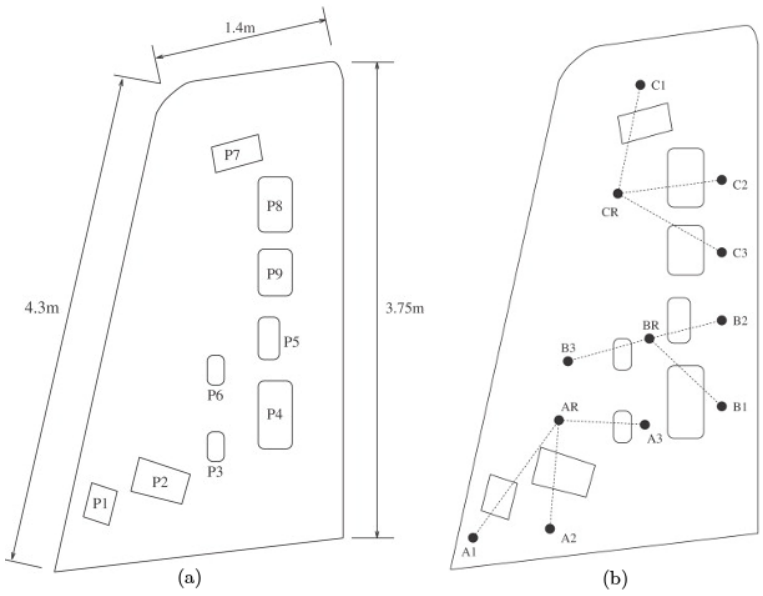


Figure 3.6: Wing schematics: (a) panel locations, (b) sensor layout.

and C; each group has one centrally-placed reference transducer (AR, BR, CR) and three response transducers (A/B/C1-3), labelled in Figure 3.6b.

It is expected that damage will manifest itself as alterations in the fundamental structural parameters; typically, a reduction in stiffness [5]. Changes in stiffness will alter the dynamic characteristics of the system; therefore, frequency domain observations can be used (as features) to (indirectly) monitor any physical changes that might relate to damage. In an attempt to represent SHM data in practice, only the response (output) data are used, to define observations in the frequency domain. As such, transmissibilities are used to monitor any changes that might relate to damage; specifically, this is a complex-valued function of frequency, which is the ratio of the response (transmitted) spectrum, to that of the reference spectrum. As such, there are nine transmissibilities — three for each group, represented by dotted lines in Figure 3.6b. The transmissibility is approximated via the discrete Fourier transform of the output acceleration time-series using a Welch estimator [70]. In all cases 1024 spectral lines were recorded, from 1024 to 2048Hz [69].

The are 1782 observations for each transmissibility — 198 for each damage condition. To reduce the dimensionality of the dataset, each transmissibility

is reduced to a single novelty index through a Mahalanobis-squared-distance (MSD) novelty detector [5, 69] — for details, refer to Section 1-4.3. To build the novelty detectors, regions of spectral lines from each transmissibility are selected with the aid of a Genetic Algorithm (GA). Briefly, the GA iterates though a population of MSD novelty detectors, learnt with different sets of spectral lines. The *fitness* of each set is assessed using the inverse classification error on a validation-set for a simple multilayer perception [35]. The 'fittest' sets are passed on to the next generation by combining their solutions. Mutation is also included by the occasional random switch of a feature. For a detailed discussion of the feature selection procedure, the reader is referred to [71].

It it should be mentioned that a validation set must be used to assess the fitness when applying a genetic algorithm for dimension reduction, and the availability of these sets can negate the need for active learning. However, if these data groups are small, they could be used as the initial sample for the DH learner. The investigation of further data could then be dictated by active learning; this is not particularly problematic when the partially-supervised method is learnt *offline*. Alternatively, effective and wholly unsupervised methods for feature extraction (with high-dimensional engineering data) would be ideal for partially-supervised learning, particularly in the online setting; a technique is proposed in the next chapter, Section 4-4.3.

In summary, the data represent a nine-class classification problem, concerning damage location. As such, the label space is $\mathcal{Y} = \{1, \dots, 9\}$ s.t. $y_i \in \mathcal{Y}$. The measured signals were converted to the frequency domain, to define nine transmissibilities; each transmissibility is then represented by a single novelty index, compressing the observation data to nine dimensions, thus $\mathbf{x}_i \in \mathbb{R}^9$. The dataset was designed to be wholly supervised; however, in these tests the labels are hidden, to demonstrate active learning. The data are projected through a linear transform via PCA (see Section 1-4.2), onto three dimensions for visualisation, as shown in Figure 3.7. Note, the experiments are *not* applied to this projection of the data, however, Figure 3.7 is still used to reference the separability of the data in the feature-space $\mathcal{X}$, as PCA highlights this variance.

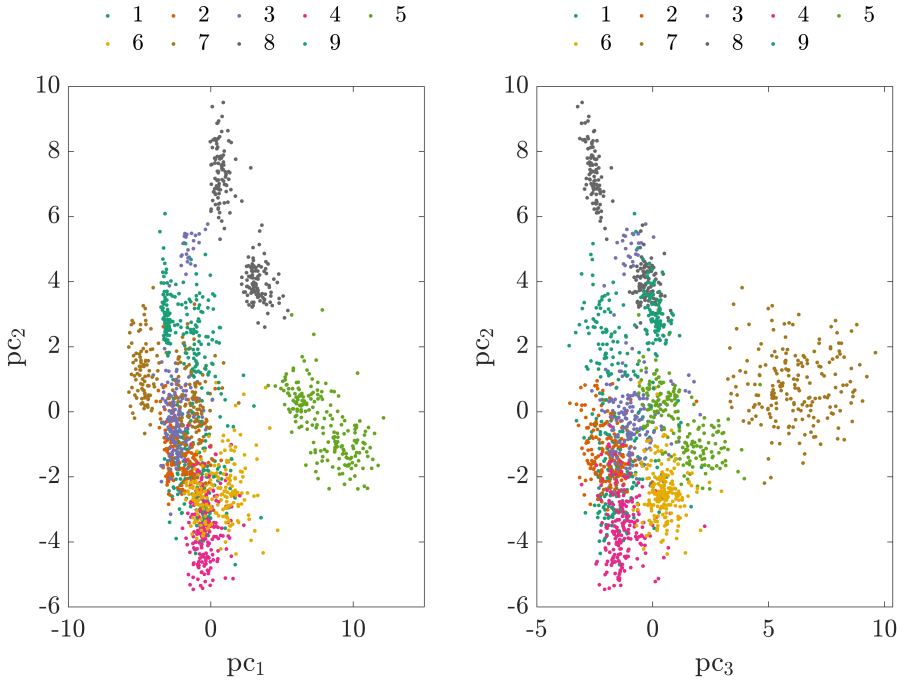


Figure 3.7: Visualisation of the Gnat data, first three principal components.

3-3.2. Test procedure

DH active learning will be compared to two *passive learning* benchmark methods: random sample training and standard supervised learning. For each experiment, the observation data and hidden labels are split into a test set $\mathcal{D}_{test}$ (33%) and a *potential* training set $\tilde{\mathcal{D}}$ (66%) using random indices.

1. *Standard supervised learning:* conventional *passive* learning in engineering applications. All the available training data are used to train the classifier, $\mathcal{D} = \mathcal{D}_l = \tilde{\mathcal{D}}$. As a result, this method is the most expensive (in terms of labels); therefore, the achieved accuracy should be considered the target performance.
2. *Random sample training:* another form of passive learning [61, 62], which takes a random sample of n data from the potential training set, then queries the labels: $\mathcal{D} = \mathcal{D}_l \subset \tilde{\mathcal{D}}$. The classifier is trained using this labelled subset alone.

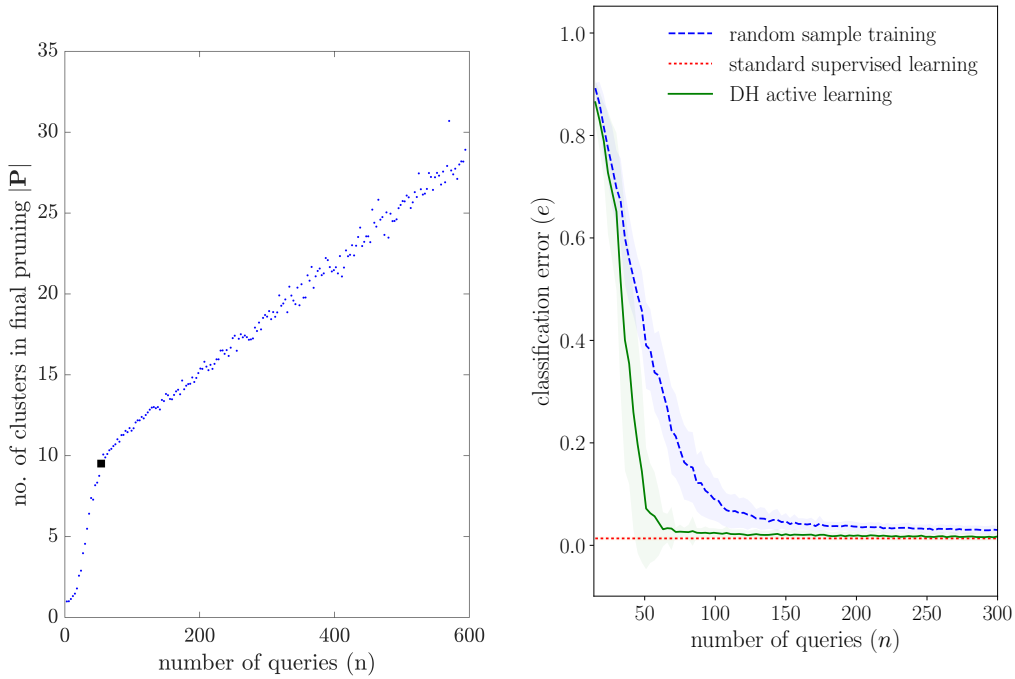
3. *DH active learning:* $\tilde{\mathcal{D}}$ is presented as a pool of unlabelled instances. Following Algorithm 2, guided sampling *actively* selects n of the most informative data, according to the *select* procedure; such that the labelled set is $\mathcal{D}_l = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$. When the budget runs out, the labels are propagated to the remaining unlabelled data $\mathcal{D}_u$ in $\tilde{\mathcal{D}}$, throughout the admissible cluster structure. A classifier is trained using this dataset, where $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$.

For standard supervised learning $\tilde{\mathcal{D}}$ and $\mathcal{D}_{test}$ are resampled 100 times, and the classifier is trained/validated 10 times; the predictive performance of the model is then evaluated using the test-set (i.e. 1000 runs in total). For methods 2 and 3 the same procedure applies while increasing the sample budget n for the labelled data, such that $n = \{15, 18, 21, ..., 594\}$:

3-3.3. Results & Discussion

The first admissible pruning and labelling of $\mathbf{T}$ (leading to label propagation) was generally found after 54 queries. According to the rules set out in Section 3-2.6, this occurs when the number of clusters in the refined pruning $\mathbf{P}$ is greater than or equal to the number of labels seen so far, $K(t)$. Intuitively, this should (usually) occur when $|\mathbf{P}| \geq 9$ — this threshold is shown by the highlighted point in Figure 3.8a. Interestingly, after this point, the number of clusters in the final pruning grows almost linearly with n; suggesting the additional rule (3.19) works well to define when label propagation is suitable/stable.

The classification error e is plotted against an increasing query budget n — shown in Figure 3.8b. Each curve has a shaded region representing one standard deviation about the mean. Results show that using the DH learner provides a significant increase in classification performance, particularly for lower query budgets. As to be expected, there is a notable increase in the classification performance as label propagation becomes admissible, $n \gtrapprox 54$. At this stage, just 3.0% of the hidden labels are used, and the average error on the test-set is 6.26%. This is compared to the supervised learning error, 1.35%, which requires *all* the hidden labels. In other words, at $n = 54$, the DH active learner achieves 95.0% of the performance of the supervised learning benchmark, while using just 3.0% of the labels; this is a significant achievement for engineering



(a) (b)

Figure 3.8: (a) Average number of clusters in the final pruning $|\mathbf{P}|$ for an increasing query budget n; ■ indicates the point at which label propagation becomes admissible, $(n, |\mathbf{P}|) = (54, 9.52)$. (b) Classification error e for an increasing query budget n. Plots are provided for the DH learner and both benchmark methods.

applications. At the same query budget, random sample training reaches 62.7% of the performance of supervised learning; this reduction in relative performance (32.3%) further highlights the advantages brought about by cluster-adaptive partially-supervised learning.

Following 102 queries, the DH learner achieves 98.9% of the wholly supervised benchmark performance, while using only 5.7% of the hidden labels. Here random sample training achieves 92.4% of supervised learning performance, for the same label budget n.

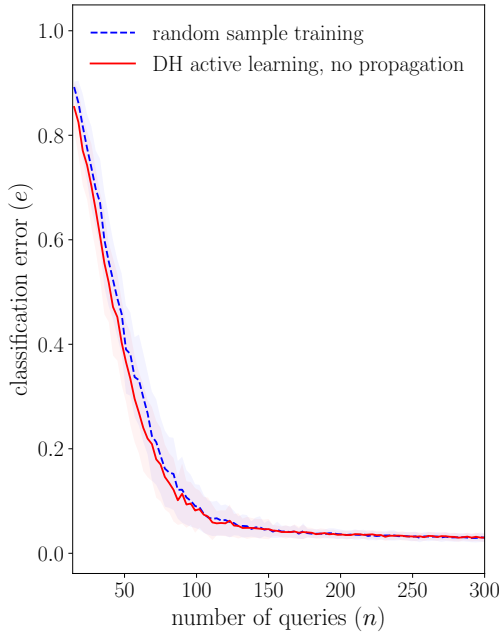


Figure 3.9: Classification error e for an increasing query budget n. Plots are provided for classifiers trained using guided sampling (the DH learner *without* label propagation) vs. random sample training.

To highlight any advantages from the learner actively directing queries (guided sampling), the classification error (without label propagation) is compared to random sample training in Figure 3.9. Ideally, a classifier trained using a subset selected via guided sampling would outperform one trained by a plain random sample. However, Figure 3.9 fails to illustrate a significant advantage.

As a result, it is safe to deduce that improvements provided by the DH learner, in these specific experiments, are a result of cluster-adaptive label propagation. In order to increase the influence of guided sampling, the `select` procedure (Equation 3.17) could be adapted for applications to engineering data. However, it is acknowledged in the original paper [60] that guided sampling will only provide a significant benefit when the hierarchical clustering has some large and fairly pure clusters near the top of the tree. (These will quickly be identified, and very few queries will subsequently be made in those regions [60].) It is clear from Figure 3.7 these data do not present the ideal case; although, some relatively pure, separate groups are still shown in the data projections (classes 5 and 7).

To investigate this further, the averaged confusion matrix for supervised learning experiments is provided in Figure 3.10. This is shown in an attempt to highlight classes that are mixed, as these are assumed the most confused. With successful guided sampling, querying should be higher in the confused, mixed groups, while reduced in homogeneous, separable groups. Specifically, classes 9, 6, 3 should receive a high number of queries, while classes 8, 7, 5, 4 are queried less.

Averaged sample counts across each class are provided in Figure 3.11. There is not a great deal of specificity for guided sampling, however, the `select` procedure does successfully direct queries to some extent: in particular, classes 5 and 7 are sampled significantly less than other groups; this makes sense, as they are among the least confused in Figure 3.10, additionally, they define clear, separable clusters in Figure 3.7. Class 2 also has a low query fraction, which is justified considering its ranking in the confusion matrix.

For the remaining classes, guided sampling is more ambiguous. This is understandable, considering how mixed these classes are — see Figure 3.7. Class 8, however, is observed to be relatively separate in the data projections, and it is the least confused; despite this, it is frequently queried by the learner. It is likely that the clustering results are poorly representative of the underlying distribution of the data in class 8, for high levels of the hierarchy. As a result, guided sampling is less influential for this class. The same principle leads to higher queries in classes 1 and 4 than might seem necessary, although, this is less surprising, as these clusters are visibly mixed in the data projections. To

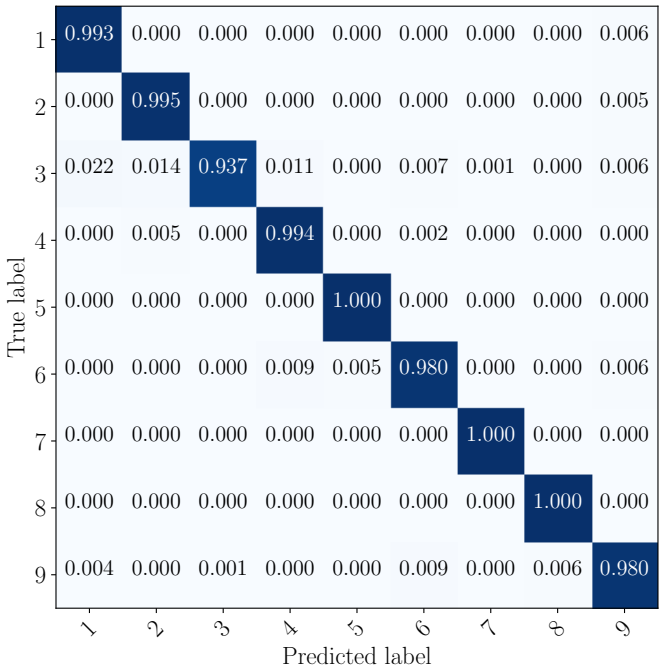


Figure 3.10: Averaged confusion matrix.

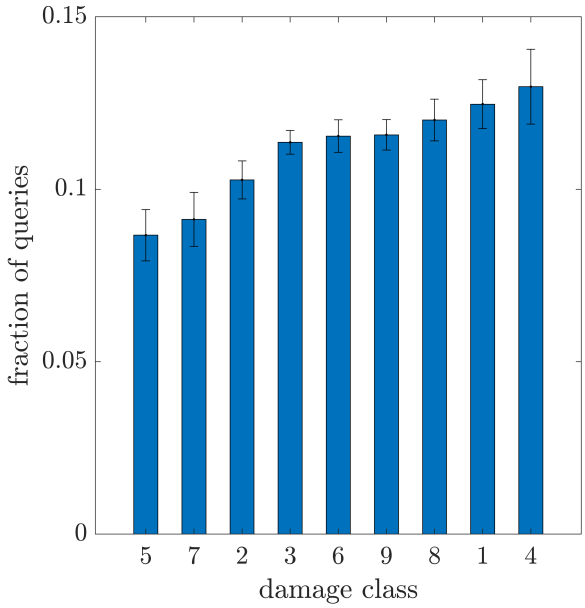


Figure 3.11: Average faction of (n) queries per class.

improve guided sampling for these groups, the initial clustering could be defined in an alternative manner. Experiments with alternative linkage functions and distance metrics (other than Ward's average linkage and Euclidean distance) might pose a solution; however, the issue is very application specific. In the-best case scenario, the input data will define more separable and pure clusters.

3-4. Concluding Remarks

To introduce partially-supervised methods to data-based engineering, a cluster-based algorithm has been applied to data from aircraft experiments. Dasgupta's and Hsu's (DH) algorithm is applied [60], which starts with a hierarchical clustering of the unlabelled data, dividing the feature-space into many partitions. An informative training set is built by directing queries to areas of the feature-space that appear mixed in terms of labels, while clusters that appear homogeneous are queried less. When appropriate, queried labels can be propagated to any remaining unlabelled instances, using the cluster structure and a majority vote — a process typically associated with semi-supervised learning. Any standard supervised classifier can then be learnt from the resulting labelled dataset.

Experiments successfully demonstrate that cluster-adaptive active learning has the potential to significantly reduce labelling costs, by utilising both labelled and unlabelled data in a partially-supervised framework. The DH algorithm provides a significant increase in performance over passive training with a random sample of the same budget n; furthermore, the classification performance is significantly improved when compared to the supervised learning benchmark, which requires all the data to be labelled. Notably, following label propagation ($n \gtrapprox 54$), the DH active learner achieves 95.5% of supervised learning performance, while using just 3.0% of the labels.

In the experiments here, active learning is successful as a result of cluster-adaptive label propagation — a process enabled by the hierarchical framework of the algorithm. Although guided sampling is directing queries to some extent, this procedure alone is not influential enough to directly affect the classification performance. Alternative select procedures might increase the influence of guided sampling, although in real terms, the success of this mechanism is very data specific. If relatively pure, separable clusters existed in high levels of the hierarchy, guided sampling should be more influential.

Moving Forward

The algorithm is well suited to engineering applications: it utilises unlabelled data, and, importantly, the damage classes do not need to be defined *a priori*. As a result, new labels can be included as they are discovered. The algorithm is limited in some respects, however, as a large set of measured signals must be available *a priori*, to build the tree structure. In consequence, the DH learner is *less* suitable in more challenging applications of *online* SHM, where measurements are also unavailable *a priori*. In this online case (Section 1-5.1), the partially-supervised algorithm itself must train, update and adapt during system operation, which can be problematic for discriminative (e.g tree-based) methods, as discussed in Section 2-1.

To address this, future work should consider modifications to accept a stream of online measurements, such that the model of the underlying data structure is updated online, during system operation. Additionally, it would be desirable to use probabilistic methods to provide well-defined uncertainties, which can be associated with the propagated labels; as such, probabilistic models should

allow the select procedure and label propagation to be controlled in a statistical manner.

4

PROBABILISTIC ACTIVE LEARNING FOR ONLINE SHM

Overview: A novel, probabilistic framework for the classification, investigation and labelling of data is suggested as an online strategy for Structural Health Monitoring (SHM). The proposed parametric algorithm (a Gaussian Mixture Model) can learn, update and adapt online, to classify streaming SHM data. The model of the data allows for the definition of a multi-class classifier, to aid both damage detection and identification, while using a limited number of the most informative labelled data. The algorithm is applied to three datasets in the online setting; the Z24 bridge data, a machining (acoustic emission) dataset, and measurements from ground vibration aircraft tests. In the experiments, active learning is shown to improve the online classification performance for damage detection and classification. A novel tool for unsupervised feature extraction from vibration data is also introduced.

4-1. Generative Mixture Models

Considering the conclusions of Chapter 3, in the context of SHM it is desirable to work towards a classifier that can update and adapt online, such that any new groups of data are incorporated into the model as they are discovered. Further-

more, as discussed in Section 1-5.1, working towards probabilistic predictions is desirable in risk-based applications, while keeping the number of labelled data to a minimum. In consequence, it should be clear that the application of *generative* methods (introduced in Sections 2-1, 1-3.1) can offer a natural way to address these issues:

- the model of the data is relatively simple to retrain upon discovering new groups of data (the class can simply be added to the existing mixture model);
- additionally, any unlabelled data can be incorporated into well-defined probabilistic models, relatively simply. (The extension to include unlabelled data is presented in Chapter 6.)

Furthermore, when working with engineering datasets, *assuming* a parametric mixture model (for density estimation) can also be useful, given prior knowledge of the structure of the data for that application. For example, SHM signals recorded from a mechanical system or structure should remain relatively consistent for a given operating, environmental, or health condition — synonymous with the consistent underlying physics[1] [5].

4-2. A Probabilistic Model for Guided Sampling

A probabilistic approach is suggested as the foundation for an active framework with engineering data. This approach is built around a supervised probabilistic mixture model, which is learnt from a small initial (random) sample of labelled measured data. As with existing models in the literature [25, 31, 57], the measured data, $\mathbf{x}_i$, are assumed to be sampled from a parametric mixture model; specifically, a Gaussian Mixture Model (GMM) [25, 32]. Therefore, referring back to the theory introduced in Section 1-4.4, the underlying distribution of the measured data $\mathbf{x}_i \in \mathcal{X}$, for each class k, is described by a Gaussian distribution,

$$p\left(\mathbf{x}_i \,|\, y_i = k\right) = \mathcal{N}\left(\mathbf{x}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \tag{4.1}$$

where k is used to index the class group, s.t. $k \in \{1, ..., K\}$; therefore, $\boldsymbol{\mu}_k$ is the mean and $\boldsymbol{\Sigma}_k$ is the covariance of the data $\mathbf{x}_i$ with label k (i.e. there

[1] In turn, this justifies the cluster-assumption for semi-supervised mixture models, Section 2-2.1.

are K Gaussian base-distributions). If the Gaussian distribution proves too restrictive in describing the data for each component (e.g. the class clusters are multi-modal), an alternative base-distribution should be selected. The examples in this work, however, are *appropriately* described by a GMM for active learning.

Again, the discrete random variable, $y_i \in \{1, ..., K\}$, which describes the labels is assumed to be categorically distributed [31],

$$P\left(y_i\right) = \text{Cat}(y_i \,|\, \boldsymbol{\lambda}) \tag{4.2}$$

$\boldsymbol{\lambda}$ is vector of *mixing proportions*, which is a histogram over the label values, s.t. $\boldsymbol{\lambda} = \{\lambda_1, ..., \lambda_K\}$ and $P\left(y_i = k\right) = \lambda_k$. Bayes' rule is applied using (5.1) and (5.2) to define the generative classifier, used to predict the class associated with an unseen signal, $\mathbf{x}_i^*$ [25],

$$p(y_i^* = k \,|\, \mathbf{x}_i^*,\ \boldsymbol{\theta}) = \frac{p\left(\mathbf{x}_i^* \,|\, y_i^* = k,\ \boldsymbol{\theta}\right) p\left(y_i^* = k \,|\, \boldsymbol{\theta}\right)}{p(\mathbf{x}_i^* \,|\, \boldsymbol{\theta})} \tag{4.3a}$$

$$\boldsymbol{\theta} \triangleq \{\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\lambda}\} \tag{4.3b}$$

$$p(\mathbf{x}_i^* \,|\, \boldsymbol{\theta}) \triangleq \sum_{k=1}^{K} p\left(\mathbf{x}_i^* \,|\, y_i^* = k,\ \boldsymbol{\theta}\right) p\left(y_i^* = k \,|\, \boldsymbol{\theta}\right) \tag{4.3c}$$

4-2.1. A Bayesian Approach

The most straight-forward estimate of the model parameters $\boldsymbol{\theta}$, is the maximum likelihood (ML) estimate given the available data $\mathcal{D}_l$. In this case, $\boldsymbol{\theta}$ corresponds to the sample mean and covariance, and the sample mixing parameters. While a maximum likelihood approach is intuitive, it can be poorly representative of the underlying distribution of the data when the sample size n, is small [25]. For example, consider a class of data which relates to one of the permitted operating conditions of a system; these data might represent the normal operation of a bridge during cold temperatures. Although an engineer might expect this behavior to occur frequently during winter, it may have been observed infrequently in the current dataset $\mathcal{D}_l$. In this case, the maximum likelihood estimate would predict an unreasonably low probability (i.e. mixing proportion) for that class, as the parameters have been defined such that only the available

data are the most likely. In other words, the model has *overfit* the training data; this can lead to poor generalisation when predicting new data.

To prevent over-training and generalisation issues, various methods can be applied to regularise or validate a maximum likelihood model [35]. Alternatively, a Bayesian approach can address the issue of overtraining; this can be interpreted as a form of self-regularisation. In this case, the parameters of the model, $\boldsymbol{\theta}$, are also considered to be random variables, and prior knowledge is incorporated to provide a more robust estimate of the model.

Bayesian parameter estimates

Considering the distribution of the measured data over the feature-space $\mathcal{X}$, a prior is placed over the mean and covariance parameters for each class, $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$. A natural choice of prior, which is conjugate to the Gaussian distribution (leading to analytically tractable solutions [31, 32]) is the Normal-inverse-Wishart (NIW) distribution [25],

$$p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathrm{NIW}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{S}_0) \tag{4.4}$$

The hyperparameters of the mixture model $(\mathbf{m}_0, \kappa_0, \nu_0, \mathbf{S}_0)$ can be interpreted as follows: $\mathbf{m}_0$ is the prior mean for the location of each class $\boldsymbol{\mu}_k$, and κ_0 determines the strength of the prior [25]; $\mathbf{S}_0$ is (proportional to) the prior mean of the covariance, $\boldsymbol{\Sigma}_k$, and ν_0 determines the strength of that prior [25]. These hyperparemeters are defined such that the prior belief states that each class is represented by a zero-mean and unit-variance Gaussian distribution. (Specifically, $p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathrm{NIW}(\mathbf{0}, 1, D, \mathbb{I})$, where $\mathbb{I}$ is the identity matrix $[D \times D]$, and $\mathbf{0}$ is a D-dimensional vector of zeros.) In other words, the prior assumes that the input data are normalised in the feature-space, and as such, the measured data are normalised within the online heuristic, to support this belief.

Considering the distribution over the label-space $\mathcal{Y}$, a Dirichlet prior (Dir) is placed over the mixing proportions [31], $\boldsymbol{\lambda}$,

$$p(\boldsymbol{\lambda}) = \mathrm{Dir}(\boldsymbol{\lambda} \,|\, \boldsymbol{\alpha}) \propto \prod_{k=1}^{K} {\lambda_k}^{\alpha_k - 1} \tag{4.5}$$

$$\boldsymbol{\alpha} \triangleq \{\alpha_1, \ldots, \alpha_k\} \tag{4.6}$$

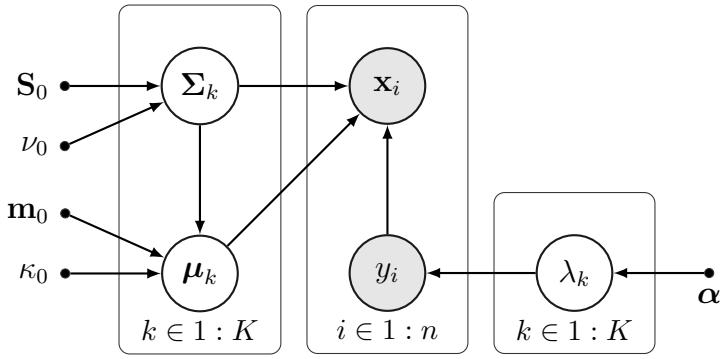


Figure 4.1: Graphical model for the GMM $p(\mathbf{x}_i, y_i, \boldsymbol{\theta})$ over the *labelled* data $\mathcal{D}_l$. As the dataset is supervised, both $\mathbf{x}_i$ and y_i are observed variables. (Shaded and white nodes are the observed and latent variables respectively; arrows represent conditional dependencies; dots represent constants (i.e. hyperparameters).)

Again, this is a natural choice of prior, as the Dirichlet distribution is conjugate to the categorical distribution [31]. The second prior introduces the hyperparameters, $\boldsymbol{\alpha} = \{\alpha_1, ..., \alpha_K\}$, which can be used to incorporate any prior belief of the probability (or weighting) of each class. In this application, each class is assumed to be *equally* weighted, s.t. $\alpha_k = n/K$, $\forall k$. This prior is used as it represents a general case; if (application specific) prior-knowledge of the class weights is available, it should certainly be included. The generative statistical model, $p(\mathbf{x}_i, y_i, \boldsymbol{\theta})$, has now been defined. The graphical model corresponding to the problem (including dependences) is shown in Figure 4.1, including any hyperparameters.

The set of labelled data, $\mathcal{D}_l$, is used to establish the initial number of classes, K. These data can then be used to calculate the Bayesian estimates of the model parameters. Note, in the context of SHM, the *initial* measured signals are regularly assumed to represent a single class, i.e. $K = 1$. These measurements should, hopefully, relate to the normal-operating-condition only. As conjugate prior distributions have been assumed, the posterior distribution over the parameter estimates can be found analytically; these are calculated for each class, $k \in \mathcal{Y}$. Firstly, the posterior distribution of $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is NIW, with

updated parameters (denoted by subscript n) [25, 31],

$$p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathcal{D}_l) = \text{NIW}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathbf{m}_n, \kappa_n, \nu_n, \mathbf{S}_n) \tag{4.7a}$$

$$\mathbf{m}_n = \frac{\kappa_0}{\kappa_0 + n_k}\mathbf{m}_0 + \frac{n_k}{k_0 + n_k}\bar{\mathbf{x}}_k \tag{4.7b}$$

$$n_k \triangleq \sum_{i=1}^{n} \delta_{k,y_i} \tag{4.7c}$$

$$\bar{\mathbf{x}}_k \triangleq \frac{\sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i}{n_k} \tag{4.7d}$$

$$\kappa_n = k_0 + n_k \tag{4.7e}$$

$$\nu_n = \nu_0 + n_k \tag{4.7f}$$

$$\mathbf{S}_n = \mathbf{S}_0 + \mathbf{S}_k + \kappa_0 \mathbf{m}_0 \mathbf{m}_0^\top - \kappa_n \mathbf{m}_n \mathbf{m}_n^\top \tag{4.7g}$$

$$\mathbf{S}_k \triangleq \sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i \mathbf{x}_i^\top \tag{4.7h}$$

again, δ_{k,y_i} is the Kronecker delta function, equal to 1 when k is equal to the observed class y_i, for the corresponding observation $\mathbf{x}_i$. The bar notation $\bar{\mathbf{x}}_k$ is the empirical mean (ML estimate) of the data in group k; the number of observations in that group is n_k; finally, $\mathbf{S}_k$ is the uncentered sum-of-squares matrix for the data in class k (5.5h). The Bayesian estimates of $\boldsymbol{\mu}_k$ (5.5b) and $\boldsymbol{\Sigma}_k$ (5.5g) are interpretable: the posterior mean $\mathbf{m}_n$ is a complex combination of the prior and the maximum-likelihood estimate; the posterior scatter matrix $\mathbf{S}_n$ is the prior scatter matrix, plus the empirical scatter matrix, plus an additional term associated with uncertainty in the mean [25].

Similarly, the posterior for the parameters of the categorical distribution over Y is Dirichlet [31],

$$\begin{aligned} p(\boldsymbol{\lambda}|\mathcal{D}_l) &\propto \text{Dir}(\boldsymbol{\lambda} \,|\, \{\alpha_1 + n_1, \ldots, \alpha_K + n_K\}) \\ &= \prod_{y=1}^{K} \lambda_y{}^{n_y + \alpha_y - 1} \end{aligned} \tag{4.8}$$

Intuitively, the posterior is obtained by adding the pseudo-counts from the prior α_k to the empirical counts, $n_k = \sum_{i=1}^{n} \delta_{k,y_i}$.

In order to make class predictions for the unlabelled data, $\tilde{\mathbf{x}}_i \in \mathcal{D}_u$, the posterior predictive distributions associated with the labels, Y, and the observations, X, can be found analytically. This is done by marginalising out the

parameters from the model [25, 31]. For unlabelled measurements, $\tilde{\mathbf{x}}_i \in X$, the posterior predictive distribution is a Student-t distribution [25],

$$p(\tilde{\mathbf{x}}_i \,|\, y_i = k, \mathcal{D}_l) = \int\int p(\tilde{\mathbf{x}}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, y_i = k, \mathcal{D}_l)\; d\boldsymbol{\mu}_k d\boldsymbol{\Sigma}_k \tag{4.9}$$

$$= \mathcal{T}\left(\tilde{\mathbf{x}}_i \,|\, \mathbf{m}_n,\; \frac{\kappa_n + 1}{\kappa_n(\nu_n - D + 1)}\mathbf{S}_n,\; \nu_n - D + 1\right) \tag{4.10}$$

$$= \mathcal{T}\left(\tilde{\mathbf{x}}_i \,|\, \mathbf{m}', \mathbf{S}', \nu'\right) \tag{4.11}$$

$$= \frac{\Gamma(\nu'/2 + D/2)}{\Gamma(\nu'/2)} \frac{\mathbf{S}'^{\,-1/2}}{\nu'^{\,D/2}\pi^{D/2}} \times \ldots$$

$$\ldots \left[1 + \frac{1}{\nu'}(\tilde{\mathbf{x}}_i - \mathbf{m}')^\top \mathbf{S}'^{\,-1}(\tilde{\mathbf{x}}_i - \mathbf{m}')\right]^{-\left(\frac{\nu'+D}{2}\right)} \tag{4.12}$$

$$\Gamma(a) \triangleq \int_0^\infty u^{a-1} e^{-u} du \tag{4.13}$$

The first two terms $\mathbf{m}', \mathbf{S}'$ in (4.11) define the mean and scale parameters respectively, and the third term ν' is the number of *degrees of freedom*. The Student-t distribution is suitable, as it has heavier tails than the Gaussian distribution, to account for the fact that the parameters are estimated from a finite set. However, as more data become available, and the degrees of freedom increase ($n_k \to \infty$, thus $\nu' \to \infty$), the Student-t tends towards the Gaussian distribution [25].

Likewise, the posterior predictive distribution associated with the labels, Y, is,

$$p(\tilde{y}_i = k \,|\, \mathcal{D}_l) = \int p(\tilde{y}_i \,|\, \boldsymbol{\lambda}) p(\boldsymbol{\lambda} \,|\, \mathcal{D}_l)\; d\boldsymbol{\lambda} \tag{4.14}$$

$$= \frac{n_k + \alpha_k}{n + \alpha_0} \tag{4.15}$$

where $\alpha_0 = \sum_{k=1}^K \alpha_k$ [25].

As in the previous examples, by utilising Bayes' rule and the posterior predictive distributions in (4.11) and (4.14), a generative classifier can be defined [25]. This is used to predict the label distribution, $p(\tilde{y}_i \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l)$, for the

unlabelled data, $\tilde{\mathbf{x}}_i \in \mathcal{D}_u$,

$$p(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) = \frac{p(\tilde{\mathbf{x}}_i \,|\, \tilde{y}_i = k, \mathcal{D}_l)\; p(\tilde{y}_i = k \,|\, \mathcal{D}_l)}{p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l)} \tag{4.16}$$

When predicting the label of future data, the maximum *a posteriori* estimate of the class labels is used to assess classification performance. This is the value in Y with the highest probability given the observation $\tilde{\mathbf{x}}_i$ [25], denoted by $\hat{y}_i$,

$$\hat{y}_i = \underset{k \in Y}{\operatorname{argmax}} \left\{ p(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) \right\} \tag{4.17}$$

As in (4.3), the marginal likelihood in (4.16), which normalises the predictive distribution over Y, is determined by the following integral; this is a discrete sum for a discrete random variable,

$$p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l) = \int p(\tilde{\mathbf{x}}_i \,|\, \tilde{y}_i = k, \mathcal{D}_l)\; p(\tilde{y}_i = k \,|\, \mathcal{D}_l)\; dy \tag{4.18a}$$

$$\equiv \sum_{k=1}^{K} P(\tilde{\mathbf{x}}_i \,|\, \tilde{y}_i = k, \mathcal{D}_l)\; P(\tilde{y}_i = k \,|\, \mathcal{D}_l) \tag{4.18b}$$

In summary, a generative classifier has been defined via a *supervised* Gaussian mixture model, with Bayesian estimates of the model parameters. As such, each class of data is represented by a Student-t distribution in the feature-space, which tends to a Gaussian distribution as more data (in that class) become available. The model is illustrated in the feature-space in the next section; additionally, code for the classifier is available at `https://github.com/labull?tab=repositories`.

A visual example: acoustic emission data

In order to visualise the mixture model — beyond the graphical representation in Figure 4.1 — the parameters are learnt for the acoustic emission (AE) dataset, introduced in Section 1-4.1. In summary, these data represent a two-dimensional, three-class classification problem, s.t. $\mathbf{x}_i \in \mathbb{R}^2$ and $y_i \in \mathcal{Y} = \{1, 2, 3\}$. Each observation, $\mathbf{x}_i$, represents the first two principal components of the features extracted from AE burst signals, collected during experiments concerning the box girder of a bridge [33]. The signals are generated by various AE sources, specifically [34]:

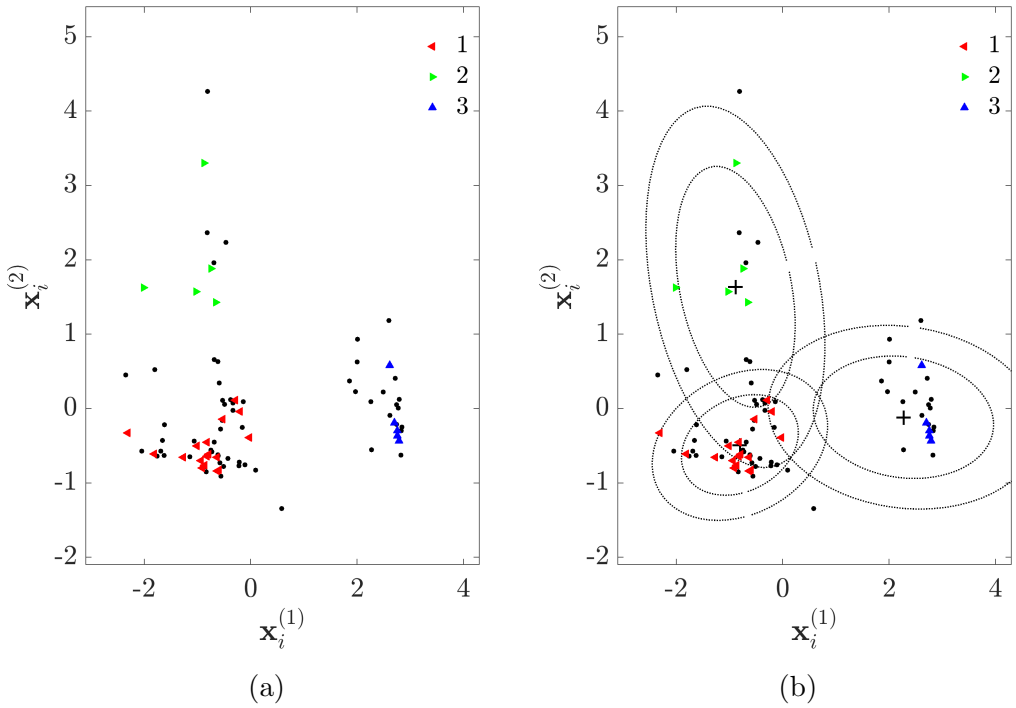


Figure 4.2: Multi-class classification of the AE data. (a) Observations in the feature-space, $\mathcal{X}$, illustrating the labelled set $\mathcal{D}_l$ (colour markers) and the unlabelled data $\mathcal{D}_u$ (black markers). (b) The generative mixture model $p(\mathbf{x}_i, y_i, \boldsymbol{\theta})$; maximum *a posteriori* (MAP) estimate of the mean (+) and covariance (dotted lines represent two and three sigma).

- class 1 - frictional processes other than crack-related events (clamping in the experimental setup),
- class 2 - crack-related events (crack extension and crack-face rubbing),
- class 3 - crack-related events, at a distance from the sensor (i.e. AE reflection signals with a relatively long rise-time).

A small subset of labelled data (i.e. $\mathcal{D}_l$) is illustrated in Figure 4.2a, along with a larger set of unlabelled data, $\mathcal{D}_u$. The mixture model is then learnt using the labelled dataset, and label predictions are made for the unlabelled data. The maximum *a posteriori* (MAP) estimate of the parameters of the mixture model are shown in Figure 4.2b.

Various probabilistic measures can now be used to estimate which of the measurements in $\mathcal{D}_u$ are the most informative when labelled. These observations

can be queried, and the cause can be investigated by the engineer to provide descriptive labels. Following the investigation and labelling of any queried data, $\mathcal{D}_l$ now includes the new observations. Therefore, the model is retrained and then further data can be queried; this process iterates until a label budget is reached, or applied sequentially to streaming data (online). This sampling and training framework is typical of *classifier-based* active learning [48, 66, 72]. Details of the application-specific heuristic are provided in the following sections.

4-2.2. Data query measures: uncertainty sampling

In the active learning literature, reviewed in Section 2-2.2, there are numerous approaches to define which of the unlabelled data are the most informative [48, 60, 65, 66]. Generally speaking, if labelled, these data provide the largest increase in the classification performance. However, as previously discussed, considering sampling bias, if queries are too focussed on a specific definition of 'informative', the training-set built by the algorithm can be poorly representative of the underlying distribution of the data. To combat sampling bias, the query framework should not focus too much on specific regions of the feature-space; here, this is avoided by combining different definitions of 'informative' [66]. Usually, these measures correspond to *representative* or *uncertain* observations, according to the current estimate/model of the underlying data distribution [65, 66]. In this work, two probabilistic measures are utilised to direct queries; the typical data queried by these measures are illustrated with the AE data in Figure 4.3.

Firstly, the *entropy* of the posterior-predictive-distribution over the labels, $p(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l)$, can be interpreted as a measure of uncertainty [63]; specifically, the entropy of the outcome $k \in Y$, is defined as the average Shannon information content [63],

$$H(\tilde{y}_i) = -\sum_{k=1}^{K} P(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) \log P(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) \tag{4.19}$$

As a result, selecting data from $\mathcal{D}_u$ with a large entropy can be considered uncertainty sampling; that is, extending the training set by selecting data from the unlabelled pool $\mathbf{x}_i \in \mathcal{D}_u$ with the most 'mixed' or 'conflicted' label predictions. This criterion will almost always query observations at the boundaries

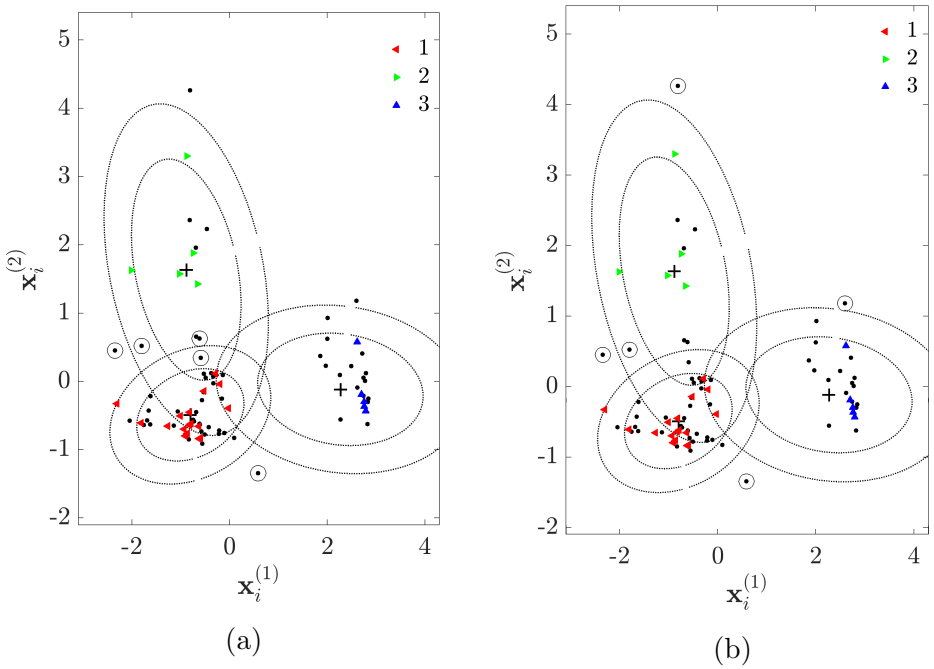


Figure 4.3: Queries over the mixture model for the AE data. The labelled set $\mathcal{D}_l$ is shown by the colour markers, and the unlabelled data, $\mathcal{D}_u$, are shown by black markers. The queried data from $\mathcal{D}_u$ are circled; in (a) these data have the *largest entropy*; in (b) the data have the *lowest likelihood* given the current model.

between two or more classes; to demonstrate this, queries directed by a large entropy are illustrated in Figure 4.3a. Note, conversely, prioritising low entropy can select measurements near the centre of the data-groups associated with each cluster, i.e. the *representative* examples.

Alternatively, observations in $\mathcal{D}_u$ with the *lowest likelihood* given the current model estimate can be queried, $p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l)$. This refers to the marginal likelihood (4.18) from the Bayes classifier, defined in (4.16), i.e.

$$p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l) = \sum_{k=1}^{K} p(\tilde{\mathbf{x}} \,|\, \tilde{y}_i = k, \mathcal{D}_l)\; p(\tilde{y}_i = k \,|\, \mathcal{D}_l) \tag{4.20}$$

This can be interpreted as the likelihood of a new observation, having marginalised out the effects of the parameters, $\boldsymbol{\theta}$, in (4.14) and (4.9), and the labels, y_i, in (4.18). Again, querying data with a low-likelihood can be seen as uncertainty sampling; however, in this case, the corresponding label distribution is not

necessarily 'mixed'. Therefore, the queried data can appear in the cluster extremities that are *not* at the boundary between two or more classes. In other words, these outlying measurements are not necessarily uncertain *in terms of the labels*. Considering these properties, low-likelihood data become suitable for querying drifting data streams, typical to online SHM, where the novel data are unlikely to appear between the boundaries of existing classes. Instead, new classes of data are likely to appear as extreme values under the current mixture model, as illustrated in Figure 4.3b.

The author wishes to reiterate: selecting training data by a given measure (uncertainty or otherwise) can be worse than random sampling. Specifically, the assumption of most classifiers, is that the training data are representative of the underlying data distribution; this implies that the samples are drawn i.i.d from the underlying probability density [48]. While the underlying data might remain i.i.d, the samples that define the training set are *guided*; therefore, the data used to learn the algorithm are inherently *not* i.i.d for an active learner. Therefore, care must be taken to ensure that the model does not become misrepresentative: it is critical that any application of active learning to engineering data should consider the type (complexity) of data that is being analysed, the quantity of data that is available, and the query budget. As shown in the experiments in Section 4-4, the benefits of active learning can vary from dataset to dataset.

4-3. An Online SHM Framework

To apply active learning to streaming data for online SHM, a framework for querying data and retraining the model must be formalised. There are various ways to approach this problem in the machine learning literature; for example, query by committee methods [48, 73] (Section 2-2.2) learn multiple classifiers which can be applied to *drifting data streams*. Disagreement amongst the classifiers is used to direct queries to aid uncertainty sampling [73]. In this work, however, the framework is built around a single model. The suggested algorithm is online, despite completely retraining the model (brute-force updates) for each new set of data. Specifically, brute-force learning is possible, as the model is quick to compute, since the parameters are defined through conjugate updates. Furthermore, if desired, the algorithm can be modified to perform cheaper

'online' updates of the parameters, mitigating the need to completely 'retrain' [74].

4-3.1. Guided Sampling

In the experiments, the data arrive in batches of size B, and the learner is permitted a limited number of queries per batch, q_b. The number of queries per batch defines the overall sample budget; this can be predefined according to the application and the costs associated with labelling. The initial distribution of data $p(\mathbf{x}_i, y_i = 1 \,|\, \mathcal{D}_l)$ is learnt from the first batch, which is assumed to be wholly labelled as class 1; that is, the normal operating condition. This assumption is reasonable in the context of SHM, as the system should be operating correctly for a large portion of the initial measured data. As a result, this model initialises as a one-class classifier [42]. If a new class of data is discovered following queries, the model updates accordingly; as such, the number of classes K does *not* need to be defined *a priori*.

The suggested active learner assumes the most informative data are defined through uncertainty sampling, using *entropy* (4.19) and *marginal likelihood* measures (4.20). Although this risks sampling bias, as only uncertain samples are targeted, these measurements are assumed to provide the largest increase in classification performance for the experiments in this work (as is common practice in the active learning literature [48]). To address sampling bias to some extent, *both* high-entropy and low-likelihood are considered as measures of *uncertainty*. As discussed, this implies that queries occur in the cluster extremities, *as well as* the boundaries between existing classes. Therefore, sampling a variety of uncertain data in this way should provided an informative training-set, $\mathcal{D}_l$, from the unlabelled streaming data, $\mathcal{D}_u$.

As each new batch of measured data arrives, the model makes a prediction for the unlabelled data $\mathcal{D}_u$, based on the labelled data seen so far in $\mathcal{D}_l$. Note, the dataset $\mathcal{D}_u$ includes the *new* batch, as well as unlabelled data from *previous* batches. The learner then queries q_b measurements from $\mathcal{D}_u$, s.t. $q_b/2$ records are queried according to high-entropy, and $q_b/2$ are queried with the lowest likelihood. This query regime effectively introduces two hyperparameters: one which determines how many of the data will be labelled, and one which

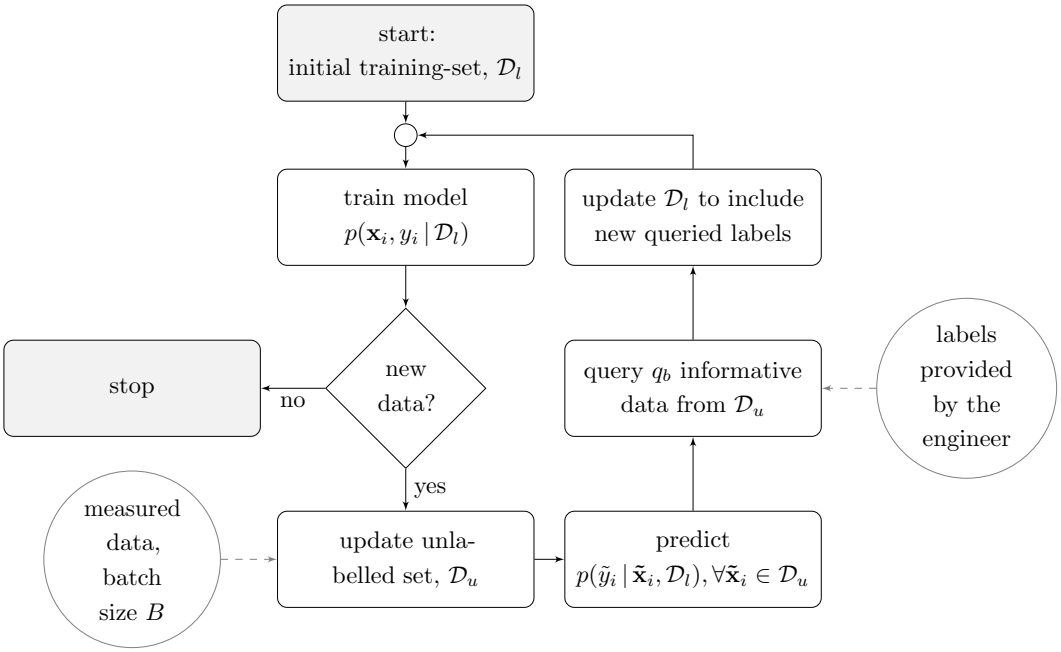


Figure 4.4: Flow chart to illustrate the online active learning process.

determines what fraction of high-entropy and low-likelihood data should be queried. In this work, an equal number of each measure is queried for simplicity. The sample budget, q_b, is the independent variable in the experiments; therefore, the proportion of each query measure is kept consistent. The investigation of various sampling regimes is being considered for future work. The online heuristic is illustrated in Figure 4.4.

Test procedure

In order to assess the diagnostic performance of the learner, the full dataset is split in half, using every other sample. This provides a distinct 'moving' test set, $\mathcal{D}_{test} = \{\mathbf{x}_i^*, y_i^*\}$. The model can then be used to predict the labels for the test data, $\hat{y}_i$ (4.17), and these can be compared to the actual labels, y_i^*, to determine an online performance metric. The *macro* f_1 score is used, which is a weighted balance of precision (P) and recall (R). Precision and recall can be defined in terms of numbers of true positives (TP), false positives (FP) and

false negatives (FN) for each class, $k \in Y$ [25],

$$P_k = \frac{TP_k}{TP_k + FP_k} \tag{4.21a}$$

$$R_k = \frac{TP_k}{TP_k + FN_k} \tag{4.21b}$$

The macro f_1 score is then defined by [25],

$$f_{1,k} = \frac{2P_k R_k}{P_k + R_k} \tag{4.22a}$$

$$f_{1\ macro} = \frac{1}{K} \sum_{k \in Y} f_{1,k} \tag{4.22b}$$

The macro-averaged f_1 metric is used, as this weights the score for each class equally, irrespective of the proportion of the data in each class. This is suitable in the context of online SHM, as newly-discovered groups of data are assumed to be equally important to the classification, despite infrequent observations; i.e. the new data might relate to damage.

4-4. Experiments

The new heuristic is applied here to three datasets to demonstrate the advantages of active learning for online SHM. To highlight the effects of uncertainty sampling, the method is compared to the *same* classifier learnt using data sampled at *random* from each batch, i.e. standard *passive learning*. As such, for the passive learning benchmark, q_b data are sampled randomly from $\mathcal{D}_u$ at each iteration (rather than selecting uncertain data with entropy and likelihood measures).

It is important to note — if the active learner queries any *past* data (this is particularly likely with entropy) this may have limitations in practice, as labelling engineering data in hindsight may not be possible, particularly when manual inspection is involved. Intuitively, the structure (or damage) will have changed since that data record. However, in the experiments here, labelling past data is considered to be feasible, as labelling in hindsight can be possible using engineering judgement and other sources of measured data. For example, consider that it is possible to assume that previous outlying data are the result

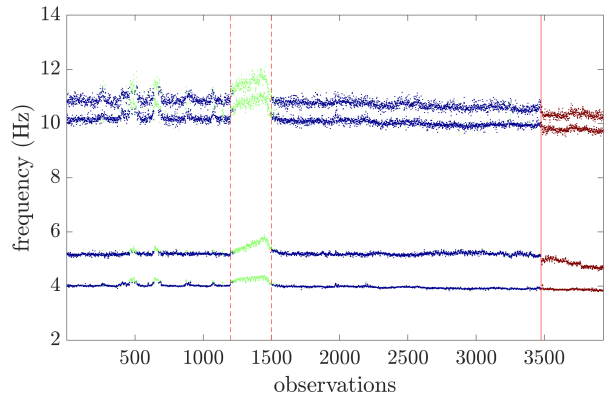
of cold temperature effects, following inspection of temperature plots (as is done with the Z24 data in the next section). The practical limitation of labelling of past data is highlighted, however, as it is an important consideration when applying the framework.

4-4.1. Z24 bridge data

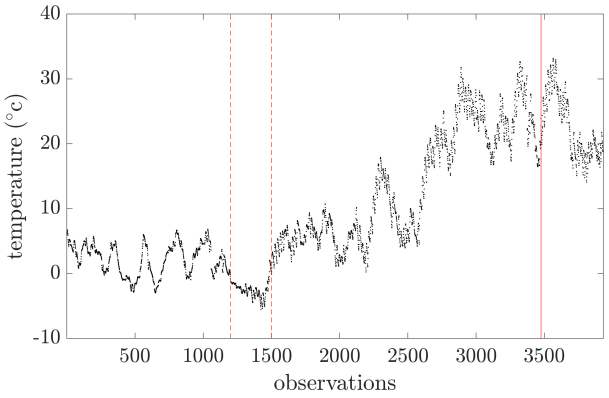
The Z24 bridge was a concrete highway bridge in Switzerland, connecting Koppigen and Utzenstorf. In the late 1990s, before its demolition, it was used for experimental SHM purposes under the SIMCES project [75]. Over a twelve-month time period, a series of sensors were used to capture dynamic response measurements, in order to extract the first four natural frequencies of the structure. Environmental measurements were also recorded, including air temperature, deck temperature, humidity and wind speed [76]. This is a relatively large dataset, with 3932 observations in total. During the benchmark project, different types of damage were artificially introduced towards the end of the monitoring year, starting from observation 3476 [41]. The natural frequencies, as well as deck temperature, are shown in Figure 4.5. Visible fluctuations in the natural frequencies can be observed in Figure 4.5a, for $1200 \leq n \leq 1500$, while there is little variation following the introduction of damage at observation 3476. The early fluctuations appear to relate to periods of very low temperature in the bridge deck, which can be observed in the temperature plot, Figure 4.5b. It is believed that the asphalt layer in the deck experienced very low temperatures during this time, leading to increased structural stiffness.

To define a classification problem for the active learning experiments, the four natural frequencies are selected as the observation data, s.t. $\mathbf{x}_i \in \mathbb{R}^4$. Firstly, the damage data are assumed to represent their own class, from observation 3476. Outlying observations within the remaining dataset were then determined using the robust Minimum Covariance Determinant (MCD) algorithm [41, 77]. These outlying data are illustrated in Figure 4.5a; as discussed, they appear to relate to cold temperatures effects. A three-class classification problem can now be defined, s.t. $y_i \in \{1, 2, 3\}$:

- class 1: normal condition data;



(a)



(b)

Figure 4.5: Z24 bridge data: (a) time history of natural frequencies, (b) time history of average deck temperature.

- class 2: outlying data due to environmental effects;
- class 3: damage.

In this application, it is clearly undesirable for an engineer to investigate the structure following each data acquisition from the bridge. Therefore, if active learning can provide an improved classification performance, compared to passive learning (random sampling) with the same sample budget, this demonstrates the relevance of active methods.

Results

Plots are provided for an increasing label budget per iteration. As discussed, the dataset is split in half, to define the training set and test set; i.e. each set contains 1966 observations for the Z24 data. Both sets increase at the same rate, and the f_1 score is assessed using the test set. The queries per batch are kept constant with $q_b = 2$, while the batch size is increased, s.t. $B \in \{8, 16, 24, 48\}$. These values correspond to query ratios of 1:4, 1:8, 1:12 and 1:24, for labelled to unlabelled data respectively. Active learning (uncertainty sampling) and the passive learning benchmark (random sampling) are applied 50 times for each query-budget ratio. The results are provided in Figure 4.6; error bars illustrate the one-sigma (σ) deviation.

Active learning for guided sampling successfully directs queries for an increased classification performance with these data. For all query budgets, there is a clear increase in the f_1 score when uncertainty sampling is used to build the training-set, $\mathcal{D}_l$. At times, sampling bias appears to negatively effect the f_1 score metric; specifically, in the early stages of monitoring, when 1:12 data are queried in Figure 4.6c. In general, however, the increase in the classification performance appears to outweigh the risk for this application.

As expected, there are drops in the classification performance as new classes are discovered by the learner; however, these are less exaggerated when an active framework is used. (The drops in performance occur as the macro-averaged f_1 score weights each class equally.)

Another advantage for active learning is consistent model predictions; this occurs because data selection follows a deterministic process. In other words, the active learner will always select the same observations, if identical data are presented in the same order. As a result, the f_1 scores are consistent, because the variability associated with the 'informativeness' of a random sample is eliminated. For lower query budgets (Figures 4.6c and 4.6d) while active learning increases the performance, it appears the classifier does not have enough information to build a reliable model of the data; thus, the f_1 scores are particularly low for both active and passive learning. To combat this issue, the query regime must be adapted (to sample the novel classes sooner), or the model should be updated to deal with this lack of information; these ideas are

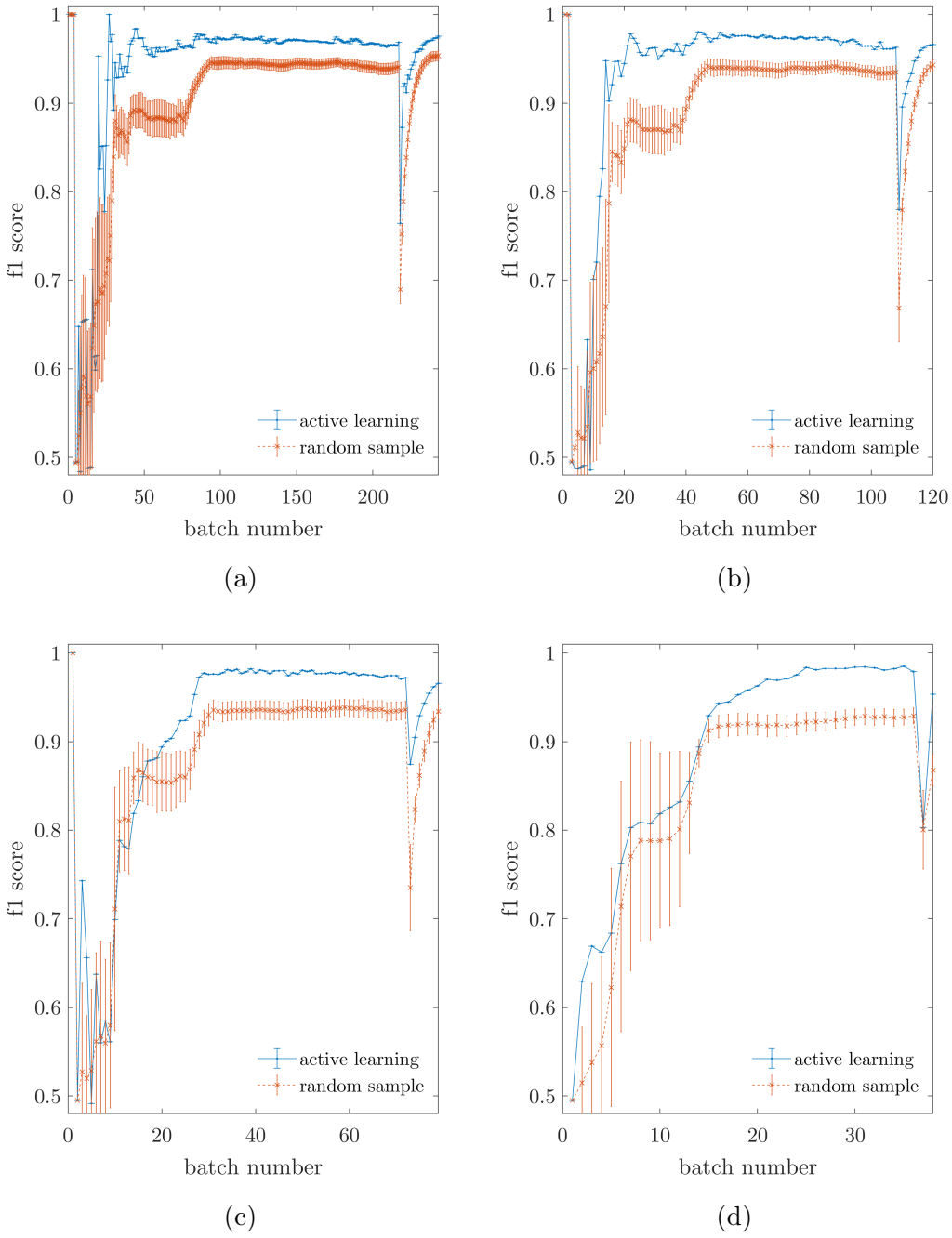
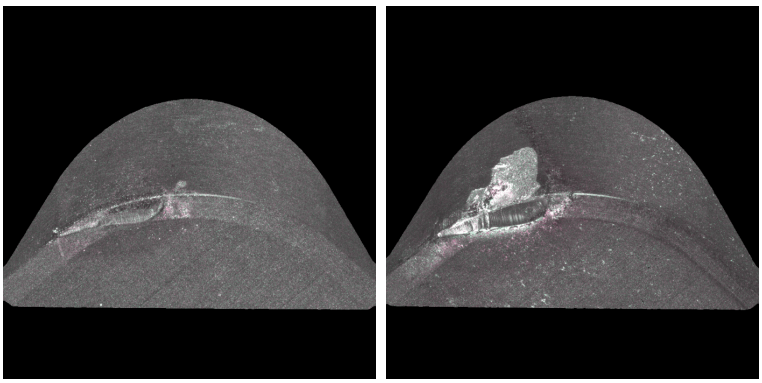


Figure 4.6: Online classification performance (f_1 score) for the Z24 data, for query budgets (as ratios): (a) 1:4; (b) 1:8; (c) 1:12, (d) 1:24.

discussed in the conclusions.

4-4.2. Machining data

The machining data are an acoustic emission dataset, collected by Wickramarachchi *et. al.*, during experiments concerning a turning operation, used to manufacture metallic components [14]. During normal operation, the cutting tool deteriorates, leading to tool wear, see Figure 4.7. Tool wear is undesirable, as it produces a poor surface finish for the machined component, which can lead to the onset of crack propagation, reducing the time in service for the manufactured product [78]. Consequently, it is critical to monitor wear of the tool; however, the current procedure requires the machining operation to be stopped, to allow for manual inspection. As a result, these inspections are infeasible in practice, due to cost and time implications [14], thus, the high-value cutting tools may be discarded prematurely when used in industry. For the experimental dataset used in this work, inspection of the tool is carried out using a 3D microscope, the resulting images are illustrated in Figure 4.7.



(a) (b)

Figure 4.7: Tool wear following inspection: (a) minor tool wear, (b) catastrophic failure of the tool.

Significant cost savings can be achieved if a model is capable of tool wear predictions while using a minimal number of tool inspections. In order to build a model to predict the current state of wear, acoustic emission (AE) measurements were taken during a typical machining operation, until catastrophic failure of the tool — see Figure 4.7b. Measurements were made by placing an AE sensor

on the machine turret; these data were recorded in the time domain, and then converted into the frequency domain. Following various signal processing steps, the measured data have 129 dimensions, with 1729 observations. For further details, see [14] — in this work, the measured data were collected using a similar experimental procedure; however, these tests concern the collection of data for a different machining operation. The data are compressed through a random projection; this method for dimension reduction is frequently used in the compressive sensing literature [79], and it is applied to online SHM in [8]. Using this approach, a random matrix is generated and normalised (replacing the $\mathbf{W}$ matrix in PCA (1.13)) and used to project the data on to 20 dimensions in an online manner, as each new batch of data arrives. 20 features were chosen, as this produced a relatively challenging feature-space for the classification problem. Therefore, the measured data are defined s.t. $\mathbf{x}_i \in \mathbb{R}^{20}$. As the annotation of these measurements is expensive, the tool was inspected at 10 regular intervals during the experiments. This corresponds to nine different classes (ranges) of tool wear, and one class after tool failure, s.t. $y_i \in \{1, ..., 10\}$. Table 5.2 summarises the dataset as a classification problem; this view of the data does not take advantage of the fact that damage can only increase.

By using AE measurements, such as the dataset presented in this work, it is desirable to accurately monitor tool wear online, while keeping the number of tool investigations (to annotate the measured data) to a minimum. Considering this aim, the active learner is applied to the machining data sequentially, as if it were online. As with all the experiments, the class labels, y_i, are hidden from the algorithm, and only measurements queried by the learner are provided with labels. Therefore, this framework implies that the engineer only needs to investigate the system when the learner queries.

Results

In these tests, the batch size is increased s.t. $B \in \{8, 16, 24\}$, corresponding to query ratios of 1:4, 1:8, and 1:12, for labelled to unlabelled data respectively. Again, the sample budget per batch is $q_b = 2$, and active/passive learning methods are applied 50 times. Plots are provided in Figure 4.8. Active learning brings consistent improvements to the classification performance (i.e. the predictive model in (1.20)) with the machining data, although, these advantages

Table 4.1: Machining AE data classes

Class label (y_i)	Observations (i)	Description
1	1 - 173	wear 1
2	174 - 346	wear 2
3	347 - 519	wear 3
4	520 - 692	wear 4
5	693 - 865	wear 5
6	866 - 1038	wear 6
7	1039 - 1211	wear 7
8	1212 - 1383	wear 8
9	1384 - 1555	wear 9
10	1556 - 1729	tool failure

are less significant: note the reduced axis range for the f_1 score. It is believed this occurs because the data are relatively separable in the feature-space, thus, the use of active learning is less effective. Intuitively, a multi-class classification problem that is less mixed in the feature-space should benefit less from active learning. Nevertheless, uncertainty sampling provides an increase in the classification performance at low query budgets; particularly when 1 in 12 data are labelled, see Figure 4.8c. Figure 4.8a shows that active learning can still be utilised at high query budgets for these data, as the variability of the prediction is reduced, such that the performance of the active learner is comparable to the upper bound (1σ) of the expected performance for random sampling, see Figures 4.8a and 4.8b. Furthermore, for all query budgets, the active learner appears to be more resilient to significant drops in the classification performance, particularly when new classes are introduced. This effect is most likely due to low-likelihood queries successfully targeting data relating to new classes, thus identifying them (and incorporating them into the model) sooner than random sampling. The variation in the classification performance for active learning is the result of the random projections for each repeat (before

averaging), and not the active learning heuristic, which still builds the training-set deterministically. Likewise, the variation in the passive learning performance is also influenced by the random-projection, as well as the random sampling.

4-4.3. Gnat Aircraft Data: Outlier Ensemble Features

The Gnat dataset was introduced in Section 3-3.1, concerning *damage* location using signals recorded during aircraft ground vibration tests. A network of sensors measured time-series (acceleration) data from the wing, and the measurements are then converted into the frequency domain, such that nine transmissibilities are used to monitor the condition of the system. In the previous experiments, dimension reduction was achieved *offline* by reducing each transmissibility into a single novelty index, where feature selection is guided by a genetic algorithm [71]. In the experiments here, however, dimension reduction is unsupervised, as labels are initially unavailable; furthermore, it should be implemented *online*, such that the method could be applied to streaming data. As a result, generally, the genetic algorithm features are unsuitable, so a novel method for *unsupervised* dimension reduction is introduced.

Data summary

In the online setting, these data represent a 10-class problem; one class is associated with the normal condition (including repairs) and one class for each state of damage (nine in total). There are 2500 observations in the dataset; 700 one-shot measurements for the normal condition and 200 for each damage condition [71]. The data are ordered such that they represent the true sequence of experiments [69]; therefore, each set of damaged tests is followed by a normal condition test. This is done to simulate an online SHM environment, where damage is followed by 'maintenance' procedures (panel replacement), bringing the structure back to the normal operating condition. Table 4.2 summarises the ordered dataset.

As in the original papers [69, 71], these data are compressed to nine-dimensions using nine Mahalanobis-squared-distance (MSD) novelty detectors [5], one learnt from each transmissibility. To achieve this in an unsupervised setting, feature-bagging with outlier ensembles [80] is used to provide robust

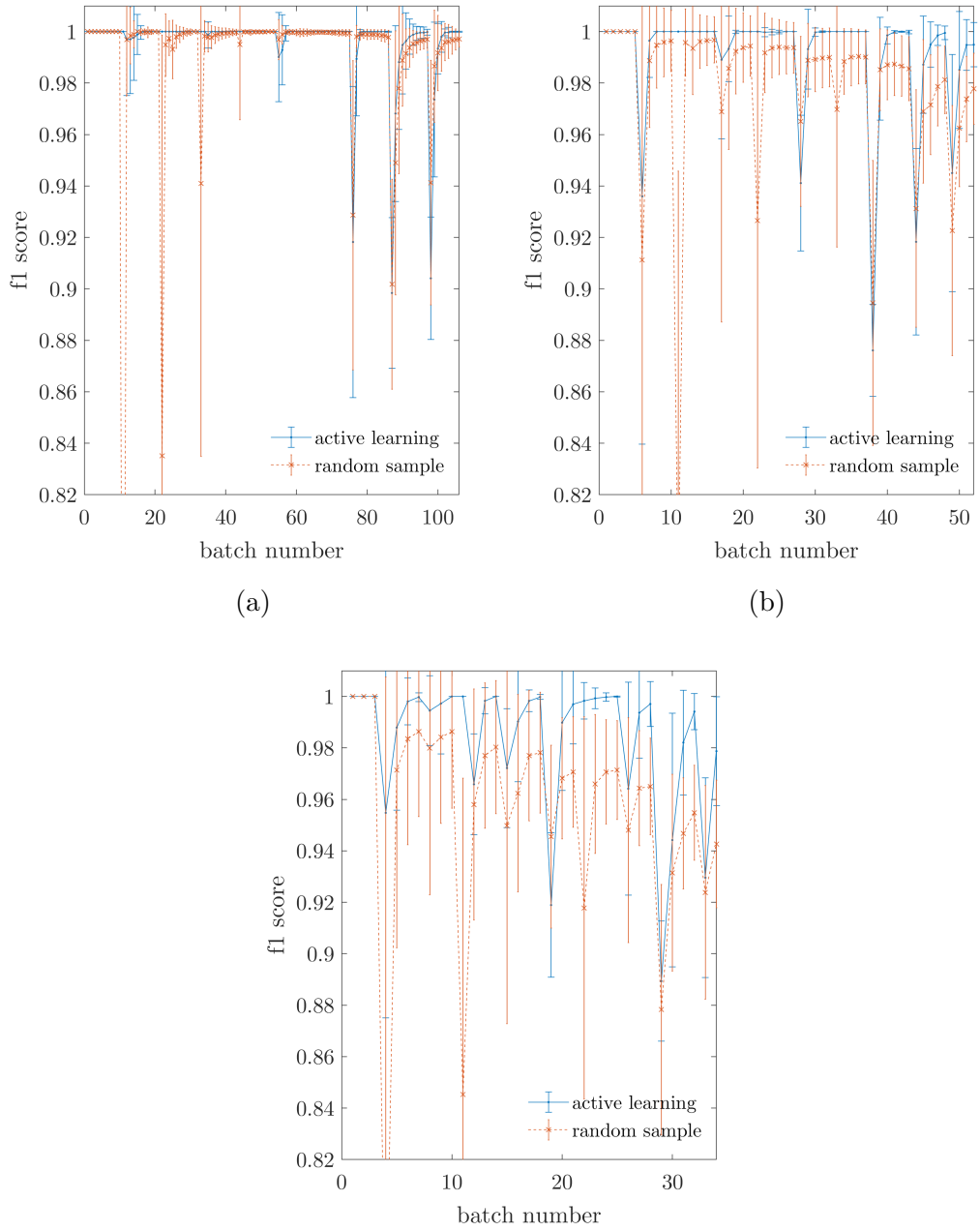


Figure 4.8: Online classification performance (f_1 score) for the machining AE data, for query budgets (as ratios): (a) 1:4; (b) 1:8; (c) 1:12.

Table 4.2: Gnat data classes

Class label (y_i)	Observations (i)	Description
1	1 - 100	normal
2	101 - 200	damage 1 (panel 1)
3	201 - 300	damage 2 (panel 2)
4	301 - 400	damage 3 (panel 2)
1	401 - 500	normal
2	501 - 600	damage 1 (panel 1)
3	601 - 700	damage 2 (panel 2)
4	701 - 800	damage 3 (panel 2)
1	801 - 900	normal
5	901 - 1000	damage 4 (panel 4)
6	1001 - 1100	damage 5 (panel 5)
7	1101 - 1200	damage 6 (panel 6)
1	1201 - 1300	normal
5	1301 - 1400	damage 4 (panel 4)
6	1401 - 1500	damage 5 (panel 5)
7	1501 - 1600	damage 6 (panel 6)
1	1601 - 1700	normal
8	1701 - 1800	damage 7 (panel 7)
9	1801 - 1900	damage 8 (panel 8)
10	1901 - 2000	damage 9 (panel 9)
1	2001 - 2100	normal
8	2101 - 2200	damage 7 (panel 7)
9	2201 - 2300	damage 8 (panel 8)
10	2301 - 2400	damage 9 (panel 9)
1	2401 - 2500	normal

discordancy measures, while avoiding (supervised) feature selection in the frequency domain.

Outlier ensembles: feature bagging

Ensemble analysis is regularly applied in the machine learning literature to reduce the dependence of model prediction on a specific realisation of the data [81, 82]. In general terms, an ensemble refers to a weighted combination of M *diverse* base predictors, $\hat{f}_{m'}$ [25], defining an ensemble output $\hat{f}_E$,

$$\hat{f}_E(\boldsymbol{x}_i) = \sum_{m'=1}^{M} w_{m'} \hat{f}_{m'}(\boldsymbol{x}_i), \tag{4.23}$$

The base predictor $f_{m'}$, refers to a machine learning model; typically, a supervised classifier is used [25, 81]. For *outlier ensembles*, however, the base predictor is an unsupervised novelty detector. Ensemble analysis can greatly increase the robustness of pattern recognition models [81], as the combined predictions are more immune to benign variations in the data that relate to noise, rather than novelty.

Importantly, successful ensemble analysis requires a *diverse* set of base-predictors [35, 83]; roughly speaking, there are two main approaches to introduce variability [81, 83]. Firstly, the base predictor can be varied across members in the ensemble (i.e. changing hyperparameters, or the algorithm itself); alternatively, *for the same model*, variability can be introduced through bootstrap samples from the dataset — i.e. sampling with replacement.

Conveniently, bootstrap sampling methods can be used to address applications of outlier analysis to high-dimensional data [84]. Specifically, the useful behaviour of measurements in high-dimensional space is often described by a subset of dimensions, which are difficult to discover in practical settings [69, 71, 81]. The use of bootstrap-sampled features (feature bagging), introduced by Lazarevic and Kumar [80], has been shown to provide a novel, successful framework for outlier analysis in high-dimensional feature spaces [85, 86]. The resulting ensemble can provide a robust measure of novelty, as the combined outputs reduce the effect of any noisy/misrepresentative features. As a result, feature bagging can provide a more general, robust approach to feature selection, reducing the uncertainty associated with this inherently difficult process [81].

Applications to the Gnat data An ensemble of M *diverse* MSD novelty detectors (1.16) is defined using random (bootstrap sampled) subsets of features $\mathbf{x}'_i$ from each transmissibility. As such, the ensemble refers to a combination of M *diverse* base-predictors, which define an ensemble output MSD_E,

$$MSD_E(\mathbf{x}_i) = \frac{1}{M}\sum_{m'=1}^{M} MSD_{m'}(\mathbf{x}'_i) \tag{4.24}$$

i.e. each m'^{th} member is an MSD novelty detector (1.16), trained using a different subset of features, with ML empirical parameters ($\bar{\boldsymbol{\mu}}_{m'}$ and $\bar{\boldsymbol{\Sigma}}_{m'}$ from (1.16)). The novelty indices from each member in the ensemble are combined through averaging to provide a single *robust* novelty index, MSD_E, from high-dimensional data [81, 82]. In this way, an outlier ensemble is built for each transmissibilty, compressing the dataset to nine dimensions in an unsupervised manner (such that only the normal condition data are used). Interestingly, the features found via unsupervised outlier ensembles provide a similar predictive performance (when used to train a classifier) compared to the supervised features, found offline, via the genetic algorithm — outlined in Section 3-3.1 [84].

As a result, online features for the Gnat data now represent a 10-class classification problem in nine dimensions; one class defines the normal operating condition and nine for the damaged states, s.t. $y_i \in \{1, ..., 10\}$ and $\mathbf{x}_i \in \mathbb{R}^9$.

Active learning results

For the Gnat data, the batch size is varied over $B \in \{8, 10, 16, 20, 24\}$ (while $q_b = 2$) to show a range of active learning effects. This corresponds to query ratios of 1:4, 1:5, 1:8, 1:10 and 1:12, for labelled to unlabelled data. As before, the results in Figure 4.9 show improvements when uncertainty sampling is used; particularly for high query budgets, shown in Figures 4.9a, 4.9b and 4.9c. With the Gnat data, however, improvements appear to become less significant as the query budget decreases. This implies that active learning fails to provide significant improvements as the learner is allowed to query less. To investigate this further, the framework is run for a 1:12 query budget; the results are shown in Figure 4.10, and demonstrate a clear example of sampling bias. In this case, the performance of active learning is worse than standard passive

learning (random sampling); as discussed, this phenomenon is well established as a critical issue when applying active learning [48, 60, 65].

It is hypothesised that the performance of active learning deteriorates at low query budgets because the Gnat data represent a particularly difficult classification problem, with 10 classes in a mixed feature-space. While the complexity of the classification means that active learning can bring significant advantages at high query budgets (Figures 4.9a, 4.9b and 4.9c), once the number of queries falls below a critical point ($\sim$ 1:10), the data become misrepresentative of the underlying distribution; in consequence, there is not enough information in the model to successfully direct queries in a way that benefits the classification. These results are important, as they imply that while active learning is useful for complex online classification, if the sample budget is too low, it can have a detrimental effect on the performance. As a result, is it critical that a method is defined to establish when (and how much) querying is required; this idea is being considered for future work.

4-5. Limitations

While the proposed active learning model works well for these data, the fact that this is a *parametric-statistical* model should be considered; in other words, assumptions are made about the distribution of the measured data. If the classes of data form disjoint (multimodal) clusters in the feature-space, this active framework might still bring advantages compared to random sample training for the same classifier; however, it is unlikely that the performance of either method would compare to that of nonparametric classifiers. (Nonparametric refers to the method used to describe the data distribution.) Some examples of such algorithms include: Gaussian process classification, relevance vector machines, or support vector machines [25]. Importantly, it is desirable to build an active learner around probabilistic measures in engineering (as in this work) as these models provide uncertainties with the associated predictions; however, a more general framework might be achieved by using a nonparametric approach, which does not make assumptions regarding the distribution of the data in $\mathcal{X}$ — such as the framework suggested in [8].

Critically, a method must be defined to determine when and how much data

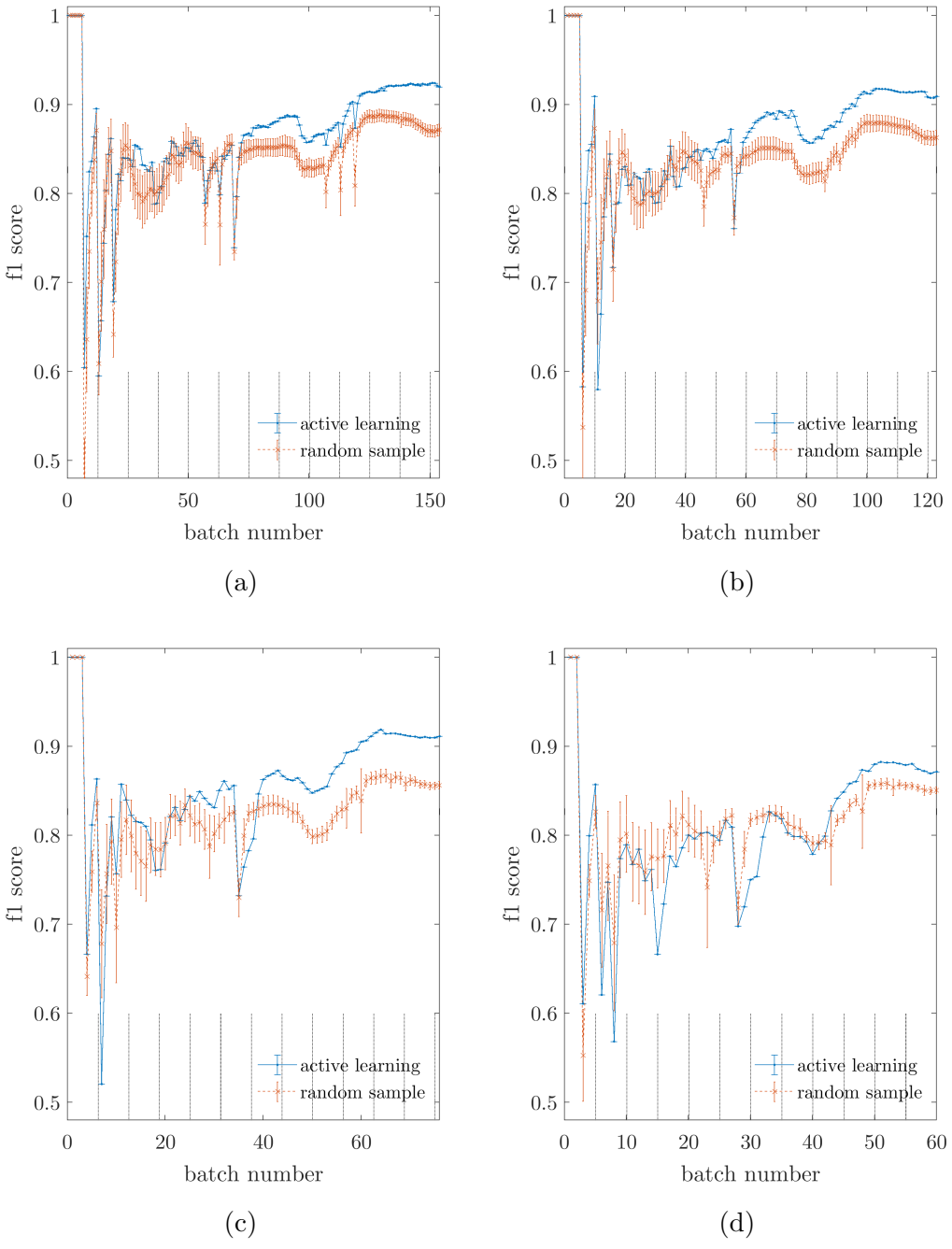


Figure 4.9: Online classification performance (f_1 score) for the Gnat data, for query budgets (as ratios): (a) 1:4; (b) 1:5; (c) 1:8, (d) 1:10. Dotted vertical lines indicate the introduction of new class data – according to Table 4.2.

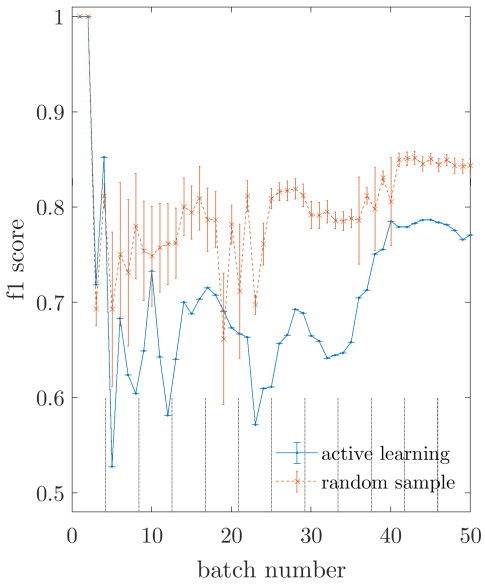


Figure 4.10: Online classification performance (f_1 score) for the Gnat data, for a query budget of 1:12. The results show significant sampling bias, which is detrimental to the classification performance. Dotted vertical lines indicate the introduction of new class data – according to Table 4.2.

to query in the online setting for active learning in SHM. In this work, a fixed number of measurements were queried with each batch of data; however, the algorithm might perform better if data are sampled only when necessary. In this way, the algorithm could choose when and which data to query, based on properties of the probabilistic model. Additionally, the automation of when to query should protect against too few data being sampled, which has been shown to lead to sampling bias with the Gnat data. Finally, the sampling regime could determine which type of data to query (i.e. high entropy, low-likelihood, or another measure), providing further automation to the SHM strategy.

4-6. Concluding Remarks

This chapter has defined a probabilistic approach to guide data queries in a novel strategy for *online* structural health monitoring. The model is initialised as a one-class classifier (novelty detection) and adapts online as new classes are

discovered — becoming a probabilistic multi-class classifier. In the experiments, the framework is applied to three datasets: the Z24 bridge data, a machining (acoustic emission) dataset, and a vibration-based dataset from a Gnat aircraft. The active learning algorithm is applied to the measurements as if they were online, recorded live from the systems in operation.

Generally, the results show a clear increase in the online diagnostic performance of the probabilistic classifier, when active learning is used to build the training-set through uncertainty sampling; this is compared to standard passive learning, where the same number of observations are investigated at random. Furthermore, the variability of the classification performance is significantly reduced when active learning is utilised. It is important to note that there are issues concerning sampling bias at low query budgets, particularly for the Gnat data. However, the definition of a probabilistic method to determine *when* to query (i.e. the optimal query budget) should be investigated for future work. Finally, in order to further utilise the information in the unlabelled data $\mathcal{D}_u$, the generative mixture model should be extended, to become *semi-supervised*.

5

TOWARDS PROBABILISTIC AND SEMI-SUPERVISED DAMAGE CLASSIFICATION

Overview: This chapter looks to investigate semi-supervised learning for the Gaussian mixture model introduced in Chapter 4, such that the model is informed by *both* labelled and unlabelled signals. The generative statistical model is introduced in the *offline* setting, and it is shown to improve the classification performance, compared to supervised learning, with simulated and experimental SHM data, while requiring no further inspections of the system. These results indicate that, through semi-supervised mixture-models in SHM, the cost associated with labelling data could be managed, as the information in a small set of labelled signals can be *combined* with larger sets of unlabelled data.

The theory behind semi-supervised updates for the Gaussian Mixture Models is introduced for damage-classification, via. the Expectation Maximisation (EM) algorithm. The semi-supervised learner is applied to simulated and experimental data, followed by a discussion on extending semi-supervised updates to the online framework in Chapter 4.

5-1. Applications to SHM

To reiterate, semi-supervised methods can bring significant advantages to SHM. In contrast to the previous chapter, where the unlabelled data $\mathcal{D}_u$ are only utilised to extend the labelled set, leading to the model $p(\mathbf{x}_i, y_i, \boldsymbol{\theta} \,|\, \mathcal{D}_l)$, a semi-supervised learner also uses the unlabelled data to inform the model, s.t. $p(\mathbf{x}_i, y_i, \boldsymbol{\theta} \,|\, \mathcal{D})$ where $\mathcal{D} = \mathcal{D}_u \cup \mathcal{D}_l$.

For example, returning to the hypothetical offshore wind-turbine; it is only possible to provide labels describing the condition of various components (such as the turbine blades) following manual inspection; this involves travelling to a remote offshore location, which is a high-cost procedure. By utilising semi-supervised tools, the cost associated with labelling data can be managed, as the information in a small set of labelled data can be combined with larger sets of unlabelled data ($\mathcal{D} = \mathcal{D}_u \cup \mathcal{D}_l$), recorded from the monitored system.

5-1.1. Related work

Semi-supervised methods have been applied to SHM in previous work. In the context of bridge monitoring, Chen *et al.* introduce a graph-based approach for label propagation [54, 87] — see Section 2-2.1 for the principals behind graph-based learners. Specifically, the objective-function of a multi-resolution classifier [88, 89] is modified, such that the weighting parameters are optimised over the labelled and the unlabelled data; additionally the graph-based classifier [54] within the heuristic is semi-supervised. The Shannon entropy [63] is used to approximate an uncertainty associated with the confidence vector over the predicted labels for the unlabelled data; this information is included in the cost function, which learns the weights of the multi-resolution classifier, as well as the filter-coefficients within each graph-based classifier [54].

Further work concerns the application of K-means [22] and fuzzy-C-means [21] for semi-supervised SHM. (Fuzzy-C-means [51] is an adaptation of K-means clustering [25, 35], such that each signal can belong to more than one cluster, according to membership weights.) Firstly, Huang *et al.* [21] use fuzzy-C-means within an online SHM strategy; the proposed method becomes partially-supervised during a *label-matching step*, where the unsupervised clusters are compared to known classes from the supervised data. Bouzenad *et al.* [22]

define a similar online heuristic using K-means; in this case, new clusters are created when a distance-based threshold is broken within the unsupervised algorithm. These heuristics can be considered as *clustering with constraints* [47]; an alternative view of semi-supervised learning, where partial-supervision is introduced through constraints on an *unsupervised* algorithm.

5-1.2. Contribution

This chapter suggests an alternative perspective, through *generative-mixture-models* for probabilistic and semi-supervised damage classification — with a view to extending the active learner proposed in Chapter 4. Provided certain assumptions hold, under Kolmogorov's axioms [28], generative methods allow for predictions with well-defined uncertainty — a significant advantage in risk-based applications. Additionally, in an engineering context, prior knowledge of the structure of the measured data is often available (e.g. drifting data streams or uni-modal clusters in the feature-space). As discussed, this *a priori* knowledge is easy to include within a generative framework, through the model definition.

5-2. Mixture Models for Semi-Supervised SHM

Generative models can naturally account for labelled and unlabelled data, as the Expectation Maximisation (EM) algorithm (used to learn mixture models in the unsupervised case [25, 90]) can be modified to incorporate labelled data [56, 57]. In agreement with the online framework proposed in Chapter 4, a Gaussian Mixture Model (GMM) is used to model the underlying distribution of the data.

The first step in the semi-supervised GMM follows conventional supervised-learning, identical to the active learner in Section 4-2. Here, Bayesian estimates of $\boldsymbol{\theta}$ are defined by treating each parameter as a random variable, and placing prior distributions over the possible outcomes. For reference, the graphical model is provided again in Figure 5.1, and equations for the supervised GMM are reprinted.

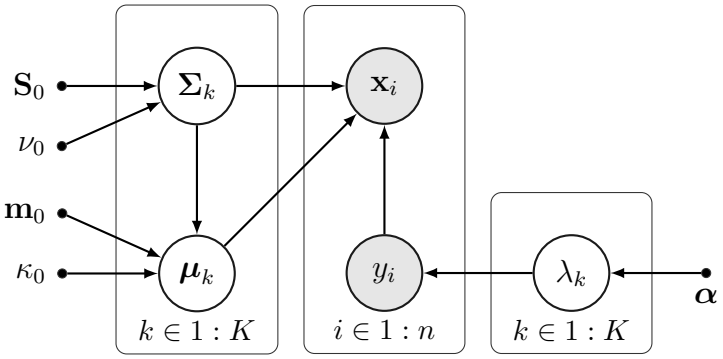


Figure 5.1: Graphical model for the supervised GMM $p(\mathbf{x}_i, y_i, \boldsymbol{\theta})$ over the *labelled* data $\mathcal{D}_l$.

The feature-space likelihood, for $\mathbf{x}_i \in \mathcal{X}$,

$$p\left(\mathbf{x}_i \,|\, y_i = k\right) = \mathcal{N}\left(\mathbf{x}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \tag{5.1}$$

Label-space likelihood, for $y_i \in \mathcal{Y}$,

$$P\left(y_i\right) = \mathrm{Cat}(y_i \,|\, \boldsymbol{\lambda}) \tag{5.2}$$

Priors over the parameter estimates $\boldsymbol{\Sigma}$, $\boldsymbol{\mu}$, and $\boldsymbol{\lambda}$,

$$p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathrm{NIW}\left(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{S}_0\right) \tag{5.3}$$

$$p(\boldsymbol{\lambda}) = \mathrm{Dir}(\boldsymbol{\lambda} \,|\, \boldsymbol{\alpha}) \tag{5.4}$$

With hyperparameters $p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathrm{NIW}(\mathbf{0}, 1, D, \mathbb{I})$, and $p(\boldsymbol{\lambda}) = \mathrm{Dir}(\boldsymbol{\lambda} \,|\, \boldsymbol{\alpha})$, where $\alpha_k = n/K$, $\forall k$. That is, the priors encode the belief that the measured data are expected to be unit-variance and zero-mean (i.e. the feature-space is normalised), while each class in the mixture model is equally likely.

Posterior distributions over the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, given the labelled data $\mathcal{D}_l$,

$$p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathcal{D}_l) = NIW(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathbf{m}_n, \kappa_n, \nu_n, \mathbf{S}_n) \tag{5.5a}$$

$$\mathbf{m}_n = \frac{\kappa_0}{\kappa_0 + n_k}\mathbf{m}_0 + \frac{n_k}{k_0 + n_k}\bar{\mathbf{x}}_k \tag{5.5b}$$

$$n_k \triangleq \sum_{i=1}^{n} \delta_{k,y_i} \tag{5.5c}$$

$$\bar{\mathbf{x}}_k \triangleq \frac{\sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i}{n_k} \tag{5.5d}$$

$$\kappa_n = k_0 + n_k \tag{5.5e}$$

$$\nu_n = \nu_0 + n_k \tag{5.5f}$$

$$\mathbf{S}_n = \mathbf{S}_0 + \mathbf{S}_k + \kappa_0 \mathbf{m}_0 \mathbf{m}_0^\top - \kappa_n \mathbf{m}_n \mathbf{m}_n^\top \tag{5.5g}$$

$$\mathbf{S}_k \triangleq \sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i \mathbf{x}_i^\top \tag{5.5h}$$

The posterior distribution over $\boldsymbol{\lambda}$ given the labelled data,

$$p\left(\boldsymbol{\lambda} \,|\, \mathcal{D}_l\right) \propto \mathrm{Dir}(\boldsymbol{\lambda} \,|\, \{\alpha_1 + n_1, \ldots, \alpha_K + n_K\}) \tag{5.6}$$

Posterior predictive distributions (marginalising out the parameters), given the labelled data, $\mathcal{D}_l$,

$$p\left(\mathbf{x}_i^* \,|\, y_i^* = k, \mathcal{D}_l\right) = \mathcal{T}\left(\mathbf{x}_i^* \,|\, \mathbf{m}_n,\ \frac{\kappa_n + 1}{\kappa_n\left(\nu_n - D + 1\right)}\mathbf{S}_n,\ \nu_n - D + 1\right) \tag{5.7}$$

$$P\left(y_i^* = k \,|\, \mathcal{D}_l\right) \propto \frac{\alpha_k + n_k}{\sum_{k=1}^{K} \alpha_k + n} \tag{5.8}$$

It is useful to define the maximum *a posteriori* (MAP) estimate of the parameters, denoted $\hat{\boldsymbol{\theta}}$, corresponding to the mode of the posterior distributions defined in (5.5) and (5.6) [25]; i.e. $p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \,|\, \mathcal{D}_l) p\left(\boldsymbol{\lambda} \,|\, \mathcal{D}_l\right)$,

$$\hat{\boldsymbol{\theta}} \mid \mathcal{D}_l \;=\; \left\{\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}, \hat{\boldsymbol{\lambda}}\right\} = \mathrm{argmax}_{\boldsymbol{\theta}}\left\{p(\boldsymbol{\theta} \,|\, \mathcal{D}_l)\right\} \;\therefore \tag{5.9a}$$

$$\hat{\boldsymbol{\mu}}_k = \mathbf{m}_n \tag{5.9b}$$

$$\hat{\boldsymbol{\Sigma}}_k = \frac{\mathbf{S}_n}{\nu_n + D + 2} \tag{5.9c}$$

$$\hat{\lambda}_k = \frac{\alpha_k + n_k - 1}{\sum_{k=1}^{K} \alpha_k + n - K} \tag{5.9d}$$

At this stage, the parameters that define the likelihoods over $\mathcal{X}$ (5.1) and $\mathcal{Y}$ (5.2) have been learnt given information in the *labelled data only*.

5-2.1. Semi-Supervised updates: Expectation Maximisation

The distribution over the parameters $\boldsymbol{\theta}$ is now updated using the unlabelled data $\mathcal{D}_u$. For the unlabelled observations, the label y_i can be considered a latent variable, which is denoted $\tilde{y}_i$ (as throughout); in this situation, the maximum *a posteriori* (MAP) estimates (5.9) are more challenging to compute [25]. The EM algorithm [90] is one method that solves this issue. The appropriate implementation of semi-supervised EM [53, 57] is similar to the unsupervised case, however, the log-likelihood of the model (and therefore the E/M-steps) are modified, such that the log-likelihood is maximised over both the labelled and the unlabelled data.

Specifically, the learning problem is defined to approach the MAP estimate of the parameters $\boldsymbol{\theta}$ given the labelled and unlabelled subsets, which is,

$$\begin{aligned} \hat{\boldsymbol{\theta}} \mid \mathcal{D} &= \operatorname{argmax}_{\boldsymbol{\theta}} \left\{ \frac{p(\mathcal{D} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathcal{D})} \right\} \\ &= \operatorname{argmax}_{\boldsymbol{\theta}} \left\{ \frac{p(\mathcal{D}_u \mid \boldsymbol{\theta}) p(\mathcal{D}_l \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathcal{D}_u, \mathcal{D}_l)} \right\} \qquad (5.10) \\ \mathcal{D} &\triangleq \mathcal{D}_u \cup \mathcal{D}_l \qquad (5.11) \end{aligned}$$

As such, it is assumed that $\mathcal{D}_u$ and $\mathcal{D}_l$ are conditionally independent. In this case, the assumption proves appropriate, as the training data are random samples from the underlying distribution: implicitly, random-sampling selects representative data that are independent and identically distributed (i.i.d) [32]. For numerical stability, the MAP estimate is implemented as a maximisation of the expected joint log-likelihood of (5.10) across the complete dataset [47],

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta} \mid \mathcal{D}) &= \mathcal{L}(\boldsymbol{\theta} \mid \mathcal{D}_u, \mathcal{D}_l) \\ &\propto \sum_{i=1}^{m} \log \sum_{k=1}^{K} p\left(\tilde{\mathbf{x}}_i \mid \tilde{y}_i = k, \boldsymbol{\theta}\right) p(\tilde{y}_i = k \mid \boldsymbol{\theta}) \ \ldots \\ &+ \sum_{i=1}^{n} \log \left[p\left(\mathbf{x}_i \mid y_i = k, \boldsymbol{\theta}\right) p(y_i = k \mid \boldsymbol{\theta}) \right] + \log p(\boldsymbol{\theta}) \qquad (5.12) \end{aligned}$$

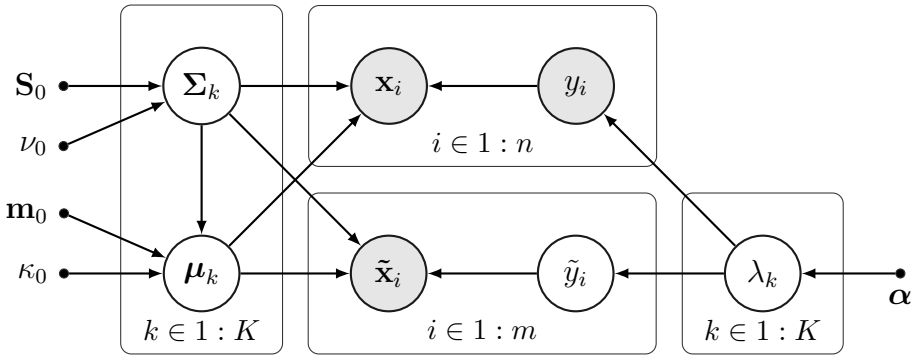


Figure 5.2: Graphical model of the GMM over both the *labelled* data $\mathcal{D}_l$ and the *unlabelled* data $\mathcal{D}_u$. For the unsupervised set, $\tilde{\mathbf{x}}_i$ is the only observed variable, while $\tilde{y}_i$ is a latent variable.

(The constant terms have been dropped for convenience.) As there exists a label y_i for each $x_i \in \mathcal{D}_l$, y_i is an observed variable for the term in (5.12) associated with the labelled data. However, in $\mathcal{D}_u$ the labels are unknown; therefore, the latent variable $\tilde{y}_i$ is marginalised out from the likelihood — this appears as a sum over k in (5.12). The model dependencies, including the observed and latent variables for each set, are illustrated in Figure 5.2.

In the EM algorithm, during each E-step, the *unlabelled* observations are classified using the current estimate of the model parameters and the classifier defined by (4.3). The M-step corresponds to finding the $\hat{\boldsymbol{\theta}}$ [1], given the *predicted labels* for unlabelled cases as well as the labelled data.

E-step Initially, during the E-step, the responsibility matrix $\mathbf{r}$ is computed for the unlabelled data; this is the posterior distribution from the classifier defined in (4.3), thus, it is an $n \times K$ matrix,

$$r_{ik} = p(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \boldsymbol{\theta}) = \frac{p\left(\tilde{\mathbf{x}}_i \,|\, \tilde{y}_i = k, \boldsymbol{\theta}\right) p\left(\tilde{y}_i = k \,|\, \boldsymbol{\theta}\right)}{p(\tilde{\mathbf{x}}_i \,|\, \boldsymbol{\theta})}, \; \forall\, \tilde{\mathbf{x}}_i \in \mathcal{D}_u \;\; \forall k \in \mathcal{Y} \tag{5.13}$$

The *effective counts* per class in $\mathcal{D}_u$ is the weighted number of points assigned to class k — this is the sum of the k^{th} column in the responsibility matrix, $r_k = \sum_{i=1}^{m} r_{ik}$ [25]. For the $\mathcal{D}_l$, however, the ground truth of $p(y_i = k \mid \mathbf{x}_i)$ is

[1] Note, the *initial* estimate of $\hat{\boldsymbol{\theta}}$ is estimated from the labelled data only, and equations (5.5), (5.6), (5.9).

given by the training labels y_i; therefore, the posterior distribution is known for the labelled points, which are discrete delta functions in the known class label [32],

$$p(y_i = k \,|\, \mathbf{x}_i) = \delta_{k,y_i}, \quad \forall\, (\mathbf{x}_i, y_i) \in \mathcal{D}_l \tag{5.14}$$

again, δ_{k,y_i} is the Kronecker delta, which equals 1 when k is the observed label y_i. In summary, the total (effective) counts per class over the complete (labelled and unlabelled) dataset are,

$$N_k = n_k + r_k \tag{5.15a}$$

$$N = |\mathcal{D}_l| + |\mathcal{D}_u| = n + m \tag{5.15b}$$

M-step In each M-step, the equations used to update $\hat{\boldsymbol{\theta}}$ involve modifications to the supervised case, as defined in equations (5.5), (5.6), (5.9). Firstly, the vector of mixing proportions $\hat{\boldsymbol{\lambda}}$, for each element is,

$$\hat{\lambda}_k = \frac{\alpha_k + N_k - 1}{\sum_{k=1}^{K} \alpha_k + N - K} \tag{5.16}$$

The mean and covariance estimates are found by modifying (5.5), to give the parameters,

$$\mathbf{m}_n = \frac{\kappa_0}{\kappa_0 + N_k}\mathbf{m}_0 + \frac{N_k}{k_0 + N_k}\bar{\mathbf{x}}_k \tag{5.17a}$$

$$\bar{\mathbf{x}}_k \triangleq \frac{\sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i + \sum_{i=1}^{m} r_{ik}\tilde{\mathbf{x}}_i}{N_k} \tag{5.17b}$$

$$\kappa_n = k_0 + N_k \tag{5.17c}$$

$$\nu_n = \nu_0 + N_k \tag{5.17d}$$

$$\mathbf{S}_n = \mathbf{S}_0 + \mathbf{S}_k + \kappa_0 \mathbf{m}_0 \mathbf{m}_0^\top - \kappa_n \mathbf{m}_n \mathbf{m}_n^\top \tag{5.17e}$$

$$\mathbf{S}_k \triangleq \sum_{i=1}^{n} \delta_{k,y_i}\, \mathbf{x}_i \mathbf{x}_i^\top + \sum_{i=1}^{m} r_{ik}\, \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top \tag{5.17f}$$

leading to the same equations for MAP estimation,

$$\hat{\boldsymbol{\mu}}_k = \mathbf{m}_n \tag{5.18a}$$

$$\hat{\boldsymbol{\Sigma}}_k = \frac{\mathbf{S}_n}{\nu_n + D + 2} \tag{5.18b}$$

The semi-supervised updates turn out to be interpretable. The MAP estimates are similar to the supervised case in (5.5); however, information in $\mathcal{D}_u$ contributes to the counts (N and N_k), as well as the mean $\bar{\mathbf{x}}_k$ and scatter $\mathbf{S}_k$ estimates.

EM learning The EM algorithm iterates between steps, leading to a hill-climbing search, which finds a *local* maximum in the parameter space. EM is sensitive to the initial estimate of $\hat{\boldsymbol{\theta}}$; to deal with this, the algorithm is normally initialised (randomly) many times. In this application, however, the starting point can be informed by the labelled data; as such, the initial guess is the MAP estimate given the labelled data, calculated with (5.5) and (5.6). This additional information mitigates the need to re-initialise the algorithm. Learning proceeds to iterate between E-steps ((5.13) and (5.14)) and M-steps ((5.17) and (5.18)), until the log-likelihood of the model (5.12), converges [90]. Semi-supervised EM is summarised in Algorithm 3.

Algorithm 3: *Semi-supervised EM for a Gaussian Mixture Model*

Input : Labelled data $\mathcal{D}_l$, unlabelled data $\mathcal{D}_u$
Output : Semi-supervised MAP estimates of $\hat{\boldsymbol{\theta}} = \left\{\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}\right\}$

Initilise $\hat{\boldsymbol{\theta}}$ using the labelled data, $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \mid \mathcal{D}_l)\}$. Supervised GMM equations (5.5), (5.6) and (5.9);
while *the joint log-likelihood* $\mathcal{L}(\boldsymbol{\theta} \mid \mathcal{D})$ (5.12) *improves* **do**
    E-step: use the current model $p(\mathbf{x}_i, y_i, \hat{\boldsymbol{\theta}})$ to estimate class-membership for the unlabelled data $\mathcal{D}_u$ (5.13);
    M-step: update the MAP estimate of $\hat{\boldsymbol{\theta}}$ given the component membership for *all* observations $\hat{\boldsymbol{\theta}} := \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \mid \mathcal{D}_l \cup \mathcal{D}_u)\}$. Semi-supervised GMM equations (5.16), (5.17) and (5.18);
end

Following semi-supervised EM, the updated MAP estimates $\hat{\boldsymbol{\theta}}$ define the predictive classifier (4.3); this is used to predict the distribution over the class-labels for new observations $p(y_i^* \mid \mathbf{x}_i^*)$.

5-3. Experiments

Probabilistic and semi-supervised damage classification is applied to a simulated example and the offline Gnat data. The simulated data demonstrate and visualise the model, while the experimental data present a more realistic and

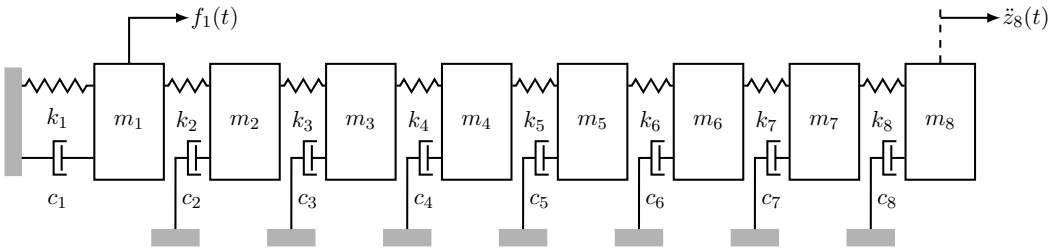


Figure 5.3: The simulated 8-DOF system

Table 5.1: 8DOF system parameters

m_1	:	0.5993 kg
$\{m_2, ..., m_8\}$	:	0.4194 kg
k_1	:	10^{-6} kN/m
$\{k_2, ..., k_3\}$	:	56.7 kN/m
$\{c_1, ..., c_8\}$	:	$0.03 \times c_c$ Ns/m

practical application.

5-3.1. Simulated Dataset

The simulated data represent measurements from an eight-degree-of-freedom (8-DOF) system. The system is defined to represent an experimental rig designed at the Los Alamos National Laboratory (LANL) [5]. A schematic of the 8-DOF system is shown in Figure 5.3[2]. The input forcing on mass i at time t is $f_i(t)$, and $z_i(t)$ is the system response (output) of mass i at time t.

The system parameters are summarised in Table 5.1. The values for critical damping c_c are defined using the decoupled equations of motion. The system is set with approximately 3% of critical damping. The spring constant k_1 is set to near zero, as this corresponds to a rigid-body mode of the experimental rig. The forcing, $f_1(t)$, is a white-noise excitation applied to mass 1, while the response, $\ddot{\mathbf{z}}(t)$, is simulated for all masses. Additive Gaussian noise is applied to the outputs, such that the signal-to-noise ratio (relative to variance) is 40dB.

It is expected that damage will manifest itself as alterations in the funda-

[2]Note: there is repeated notation for the physical parameters m and k, however, the context and use of indices $(1-8)$ should make this clear.

mental structural parameters; in this case, a reduction in stiffness [5]. Changes in stiffness will alter the dynamic characteristics of the system; therefore, frequency domain observations can be used to (indirectly) monitor any physical changes that might relate to damage. In an attempt to represent SHM data, only the system outputs $\ddot{\mathbf{z}}(t)$ are used to define observations in the frequency domain. As such, the transmissibility between masses one and eight $T_{8,1}(\omega)$ is used as a frequency domain observation; i.e. the ratio of the spectrum of the output at mass eight, $\ddot{z}_8(t)$, to the spectrum of the output at mass one, $\ddot{z}_1(t)$[3]. The transmissibility is approximated via the discrete Fourier transform of the output time series. A Hanning window is applied to each signal, sampled at 400.45Hz for 8 seconds. The transmissibilities are truncated, such that there are 1040 bins in the frequency domain, ranging from 0 - 130 Hz.

In terms of the SHM strategy, each transmissibility is an observation of the system; a transmissibility is generated every 8s from the time-series data, and these data are used for monitoring. For demonstration, it is useful to compress the transmissibility data (1040-dimensions) onto two dimensions using Principal Component Analysis (PCA) (see Section 1-4.2), to visualise the model[4]. As a result of PCA, observations $\mathbf{x}_i$ are two-dimensional, such that $\mathbf{x}_i \in \mathbb{R}^2$.

Linear damage is simulated as reductions in the spring constant k_5; the normal condition is when k_5 is at 100%, and a damage class is associated with each reduction in stiffness: there are five damage classes. Generally, a continuous parameter problem should not be framed as classification; however, discrete-steps are considered suitable to define a multi-class problem for this example. The data define a six-class problem, with 500 observations in each group; the data are summarised in Table 5.2, and the feature-space is shown in Figure 5.4.

Model visualisation: supervised learning vs. semi-supervised

The dataset is split (at random) into a training-set (2/3 of the total data, $\mathcal{D}$) and a test-set (1/3 of the total data, $\mathcal{D}_{test} = \{\mathbf{x}_i^*, y_i^*\}$). Of the training-data $\mathcal{D}$,

[3] If many transmissibilities were used, the damage identification task would be trivial, since, for a chain-like system, the transmissibility is itself a high-accuracy (deterministic) localisation criterion [5].

[4] The algorithm is applied to more realistic engineering data in the next experiment.

Table 5.2: Simulated data

Class label (y_i)	Observation index (i)	% k_5
1	1 - 500	100%
2	501 - 1000	97%
3	1001 - 1500	93%
4	1501 - 2000	88%
5	2001 - 2500	82%
6	2501 - 3000	70%

10% are labelled (the subset $\mathcal{D}_l$), while 90% remain unlabelled (the subset $\mathcal{D}_u$). The training subsets are shown in the feature-space in Figure 5.4.

Figure 5.4 plots the GMM for the supervised and semi-supervised case. In both plots, the prior is included to visualise its influence on the base distributions of the mixture model. Specifically, with few data available for training, the prior should have a large influence on the posterior distributions in order to regularise the model; this is intuitive, as the parameters defined in (5.5b) and (5.5g) are a convex combination of the prior and the maximum-likelihood (empirical) estimate.

Figure 5.4a shows the GMM given the labelled data only, i.e. $p(\mathbf{x}_i, y_i \,|\, \hat{\boldsymbol{\theta}})$ where $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \,|\, \mathcal{D}_l)\}$. Here, the training data are a small subset, and, as a result, the prior has a large influence on base-distribution estimates. The influence of the prior is strong, as there is not enough information to appropriately model data, while avoiding overtraining. On the other hand, Figure 5.4b shows the mixture model can better represent the data distribution when unlabelled instances are used to inform the MAP estimates, such that $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \,|\, \mathcal{D}_l, \mathcal{D}_u)\}$. Here, the base-distributions better represent each class, and the influence of the prior is reduced, while the model remains self-regularised and robust.

It should be clear that the model is representative, as the density is well approximated by a GMM. If the data have multi-model class components, or the classes cannot (at least approximately) be represented by a Gaussian distribution, semi-supervised learning via a *Gaussian* mixture model will break down. In this case, an alternative base-distribution must be selected.

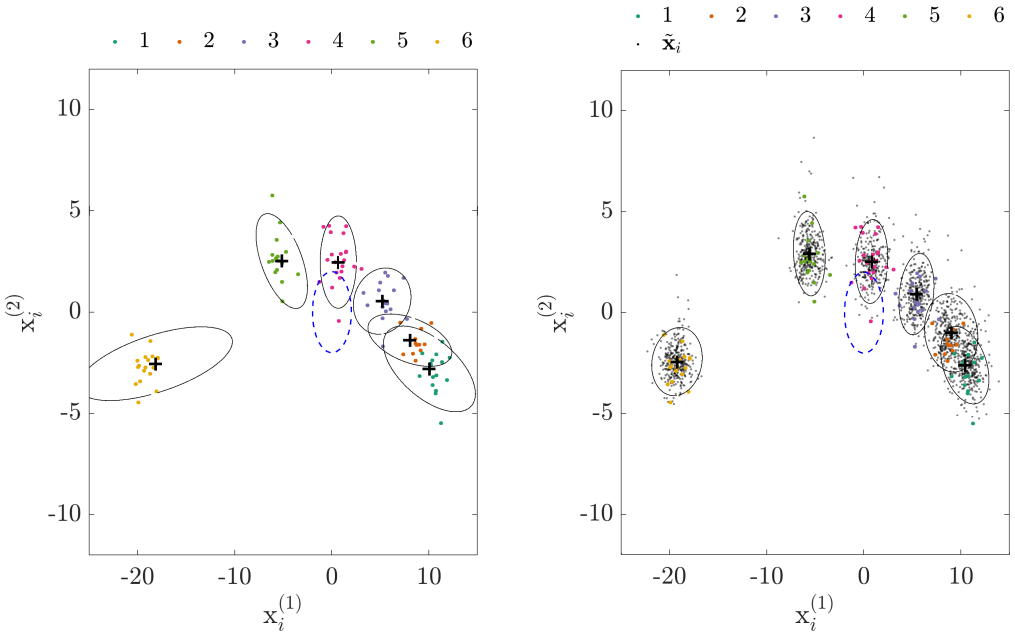


Figure 5.4: The GMM: (a) supervised learning, i.e. $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \,|\, \mathcal{D}_l)\}$ (b) semi-supervised learning, i.e. $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \{p(\boldsymbol{\theta} \,|\, \mathcal{D}_l, \mathcal{D}_u)\}$. Ellipses represent the MAP of the covariance (two-sigma), + markers represent the MAP of the mean, and the blue ellipse represents the prior.

Classification test-procedure

The performance of the model (for classification) is assessed for an increasing number of labelled to unlabelled data. The proportion of labelled data in the training-set is increased in 5% increments, from 20% – 100%. For each proportion of labelled to unlabelled data, the GMM is initially learnt given the labelled data only. Equation (4.3) is then used to classify the test-data, such that the predicted labels are the MAP of the posterior-distributions, $\hat{y}_i^* = \text{argmax}_k \{p(y_i^* = k \,|\, \mathbf{x}_i^*, \mathcal{D}_l)\}$. At this stage, the classification performance provides a benchmark for standard supervised (*passive*) learning.

The model is then updated via semi-supervised EM, given the labelled *and* unlabelled data. Label predictions are now the MAP estimates conditioned on the whole dataset, $\hat{y}_i^* = \text{argmax}_k \{p(y_i^* = k \,|\, \mathbf{x}_i^*, \mathcal{D}_l, \mathcal{D}_u)\}$. The classification

performance is re-assessed for the semi-supervised model.

As in Chapter 4, macro-averaged f_1 score (4.22) is used to assess the classification performance. For interpretability in the context of SHM, the (balanced) misclassification error e (from type-I errors for each class) is also used as a performance metric,

$$\begin{aligned} e_k &= \frac{FP_k}{FP_k + TP_k} \\ e &= \frac{1}{K} \sum_{k \in Y} e_k \end{aligned} \tag{5.19}$$

Results

Figures 5.5 and 5.6 show the classification performance (f_1 score and error) for supervised and semi-supervised learning, while increasing the proportion of labelled data to unlabelled data; the curves represent the average over 50 repeats. Semi-supervised learning consistently improves the classification performance, particularly for low proportions of labelled observations. Notably, at 2.49% labelled data, there is a 0.0380 improvement in the f_1 score, corresponding to a 3.87% reduction in the classification error — this is a significant improvement for SHM applications.

For very low proportions of labelled data ($< 0.995\%$), semi-supervised learning can decrease the classification performance — shown by a negative gain in f_1 score (or error reduction) in Figures 5.5 and 5.6. It hypothesised that the performance drops for large quantities of unlabelled data ($m \gg n$), because the natural weighting in the log-likelihood leads to the labelled instances being effectively ignored [47, 57]. To accommodate for much larger sets of unlabelled data ($m \gg n$), a re-weighted version of the joint-likelihood has been suggested [47, 56]; the investigation of this approach is suggested for future work.

Intuitively, as the proportion of labelled data reaches 100% ($m \ll n$), improvements through semi-supervised learning reduce, as there is less information gain from smaller sets of unlabelled signals. Considering the chosen method for density estimation, and the structure of the simulated data, these results are to be expected: as discussed, the underlying density is well-approximated by the chosen mixture model (a GMM in this case, Figure 5.4b). The validity of this

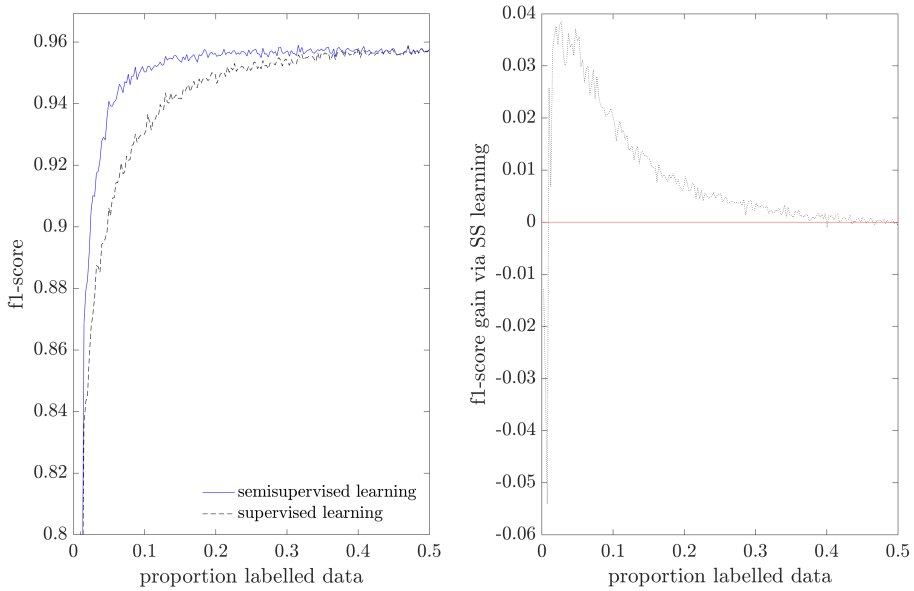


Figure 5.5: Classification performance assessed by the f_1 score for the supervised GMM vs. the semi-supervised GMM. Left: classification performance for an increasing proportion of labelled data. Right: the gain in f_1 score through semi-supervised updates, the red highlights zero-gain.

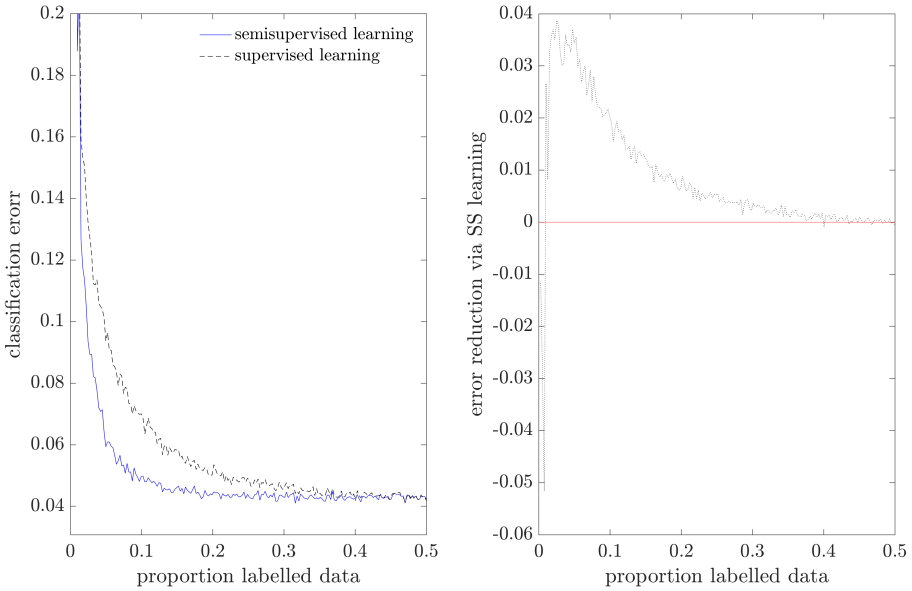


Figure 5.6: Classification error (e) for the supervised GMM vs. the semi-supervised GMM. Left: classification error for an increasing proportion of labelled data. Right: error reduction through semi-supervised updates, the red line highlights zero-error-reduction.

assumption is critical when using generative mixture models for semi-supervised learning.

5-3.2. Gnat Aircraft Data

The offline features for the Gnat data, introduced in Section 3-3.1, are used in this application, as the algorithm is *trained* in the offline setting. As such, the data represent a nine-class classification problem, concerning damage location; therefore, the label space is $y_i \in \{1, \dots, 9\}$. The measured signals are converted to the frequency domain, to define nine transmissibilities; the are 1782 observations for each transmissibility — 198 for each damage condition. Each transmissibility is then represented by a single novelty index, compressing the observation data to nine dimensions, thus $\mathbf{x}_i \in \mathbb{R}^9$.

Results

The same classification test-procedure (applied to the simulated data) is now applied to the Gnat data; results are shown in Figures 5.7 and 5.8. Again, semi-supervised updates through EM consistently improve the f_1 score and reduce the classification error, while, in this application, the data represent more practical SHM data. As with the simulated example, for very low proportions of labelled data $< 1.26\%$ ($m \gg n$), semi-supervised model updates decrease the predictive performance, as the effect of the unlabelled data appear to outweigh the labelled instances in the likelihood cost function. The general improvements through the semi-supervised GMM indicate that the experimental data can be (at least approximately) represented with a mixture of Gaussians; the maximum increase in the f_1 score is 0.0405, corresponding to a 3.83% reduction in the classification error for 2.94% labelled data.

For both tests, it is believed that semi-supervised improvements should increase if the data are approximated by some more flexible likelihood, i.e. $p(\mathbf{x}_i \,|\, \boldsymbol{\theta})$. A nonparametric representation, or a discriminative approach, would be a natural way to achieve this.

5-4. Concluding Remarks

An alternative method for semi-supervised learning has been introduced to Structural Health Monitoring (SHM). The probabilistic approach utilises Expectation Maximisation (EM) over a generative mixture model, to improve the performance of damage classification under well-defined uncertainty — a significant advantage in risk-based applications. In the proposed method, a Gaussian Mixture Model (GMM) is used to describe the underlying distribution of data from a simulated example and measured data from aircraft experiments (ground tests). The classification accuracy (based on the GMM) is shown to improve significantly when the likelihood is maximised over the labelled *and* unlabelled data (semi-supervised learning), rather than the labelled data alone (supervised learning). More specifically, semi-supervised updates lead to 3.87% and 3.83% reductions in the classification error for the simulated and experimental datasets respectively. These improvements correspond to

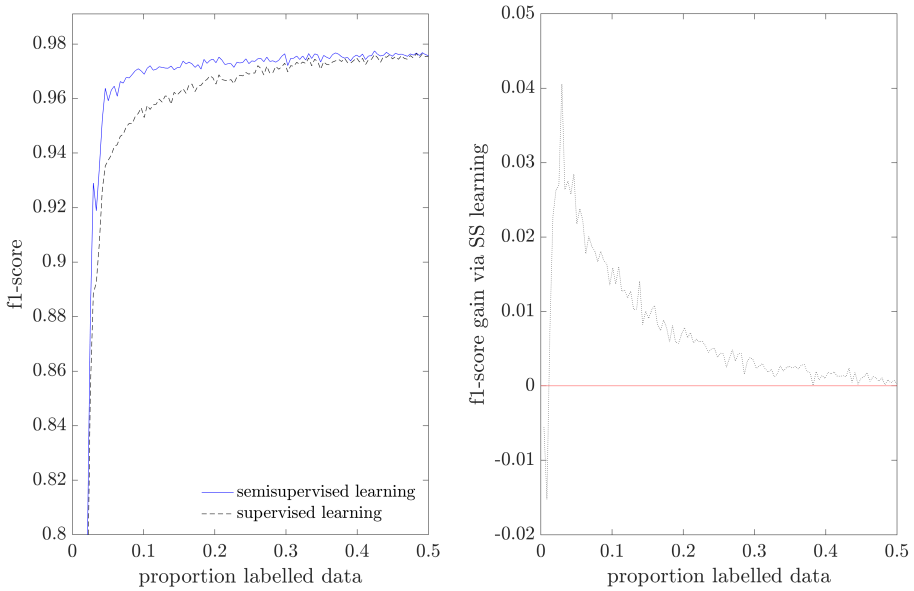


Figure 5.7: Classification performance assessed by the f_1 score for the supervised GMM vs. the semi-supervised GMM. Left: classification performance for an increasing proportion of labelled data. Right: the gain in f_1 score through semi-supervised updates, the red line highlights zero-gain.

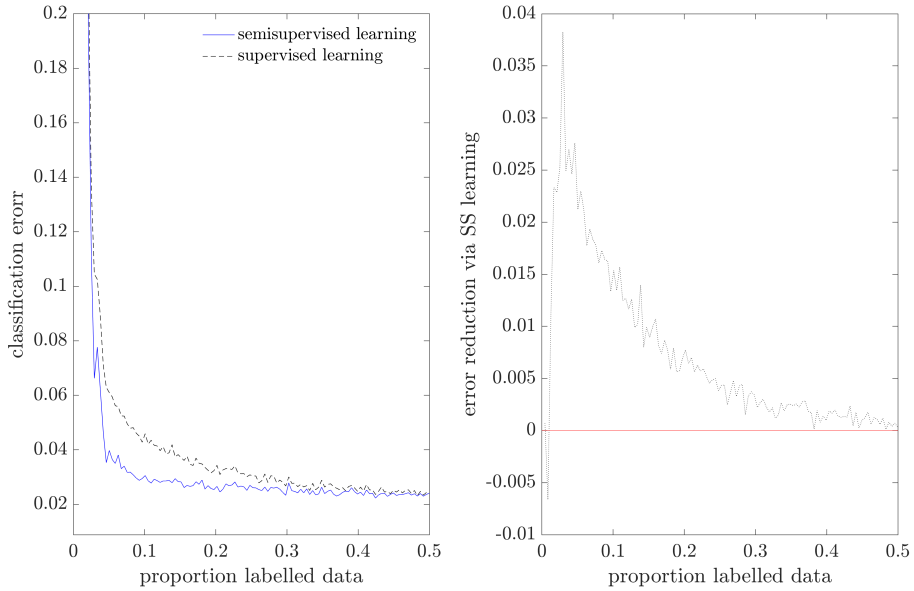


Figure 5.8: Classification error (e) for the supervised GMM vs. the semi-supervised GMM. Left: classification error for an increasing proportion of labelled data. Right: error reduction through semi-supervised updates, the red highlights zero-error-reduction.

labelling just 2.49% of the measurements for the simulated data, and 2.94% of the measurements for the experimental data — low proportions of labelled data bring significant advantages to SHM, as investigating the structure to label the measured signals can be a high-cost procedure.

While the proposed method is successful, care must be taken to ensure that the assumed (parametric) mixture model (a GMM in this case) appropriately models the underlying distribution of data. If the imposed structure is inappropriate, the inclusion of unlabelled data will decrease the model quality. Considering this limitation, future work should apply the proposed semi-supervised methodology to nonparametric mixture models, in order to describe (more complex) underlying distributions of SHM data. Most importantly, the proposed semi-supervised methodology should be incorporated within an online framework, such as the active learning framework proposed in Chapter 4.

6

TOWARDS A COMBINED SEMI-SUPERVISED AND ACTIVE LEARNER

Overview: The active and semi-supervised methodologies, introduced in Chapters 4, 5, are combined here to define a partially-supervised, probabilistic algorithm. The suggested framework adapts and updates online when applied to streaming SHM data, while using limited labels. An improved method for guided sampling within the active GMM is also introduced. Experiments demonstrate the algorithm applied to the Gnat, machining, and Z24 data. The framework is shown to increase the predictive performance of the online, multi-class classifier — provided that the assumptions of the mixture model are not violated.

6-1. Combined Online Framework

The online and probabilistic SHM framework (introduced in Chapter 4) is extended here to include the signals that remain unlabelled following queries. To achieve this, the adaptive GMM now includes semi-supervised updates via EM — i.e. the theory introduced in Chapter 5. The result of combining these methods into an online framework is summarised in Figure 6.1. The process operates online, using signals as if they were recorded live from the system in

operation. As in Chapter 4, the algorithm is active, such that uncertain data are queried to define $\mathcal{D}_l$. In this case, however, the model utilises the remaining unlabelled signals in $\mathcal{D}_u$, which have not been investigated.

For streaming SHM data, the measurements are assumed to arrive in batches of B measured signals. Therefore, the number of queries per batch q_n defines the overall sample budget for the active learner, i.e. $q_n \times$ the total number of batches. The mixture model initialises as a one class classifier, such that the first batch of signals are assumed to represent the normal-condition only, $p(\mathbf{x}_i, y_i = 1 \,|\, \mathcal{D}_l)$. If a new class of data is discovered, the model updates accordingly; therefore, as in Chapter 4, the number of classes K does *not* need to be defined *a priori*.

To summarise Figure 6.1: As a new batch of unlabelled data arrives, the parameters of the model are estimated via standard supervised learning, given the labelled data (queried so far) and equations (5.5) and (5.6), i.e. $p(\boldsymbol{\theta} \,|\, \mathcal{D}_l)$. The supervised model then predicts the label for the unlabelled signals $p(\tilde{y}_i \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l)$ — using (5.7), (5.8) and (4.3) — and q_b (uncertain) measurements from $\mathcal{D}_u$ are queried. As usual, uncertain data are assumed to be the most informative, further discussed in Section 6-1.1. The queried signals are investigated by the engineer, to provide labels y_i, and the data are added to the labelled set $\mathcal{D}_l$.

Figure 6.1 shows the additional semi-supervised step highlighted in green. Rather than predicting labels for the test-set immediately, the parameters are updated via semi-supervised EM, to find the MAP estimate of the model given the labelled *and* unlabelled data $p(\boldsymbol{\theta} \,|\, \mathcal{D}_l, \mathcal{D}_u)$. The online active learner now incorporates information in the stream of *unlabelled* signals, as well as the queried data; therefore, the predictive distributions are now semi-supervised, $p(y_i^* \,|\, \mathbf{x}_i^*, \mathcal{D}_l, \mathcal{D}_u)$.

The online density estimation is set-up in the same way as the offline case in Chapter 4, therefore, the graphical model (Figure 5.2) and corresponding equations remain the same. *Importantly*, the unlabelled set considered during EM must include signals from the new batch, *as well as* previous batches; otherwise, the hyperparameters of the prior-distributions (defined in Section 5-2) do not make sense and the model breaks down. These data must considered, because a significant proportion of the signals are likely to belong to the new class: if these observations are ignored within the EM, the prior will associate an unreasonably large mixing proportion with the new component.

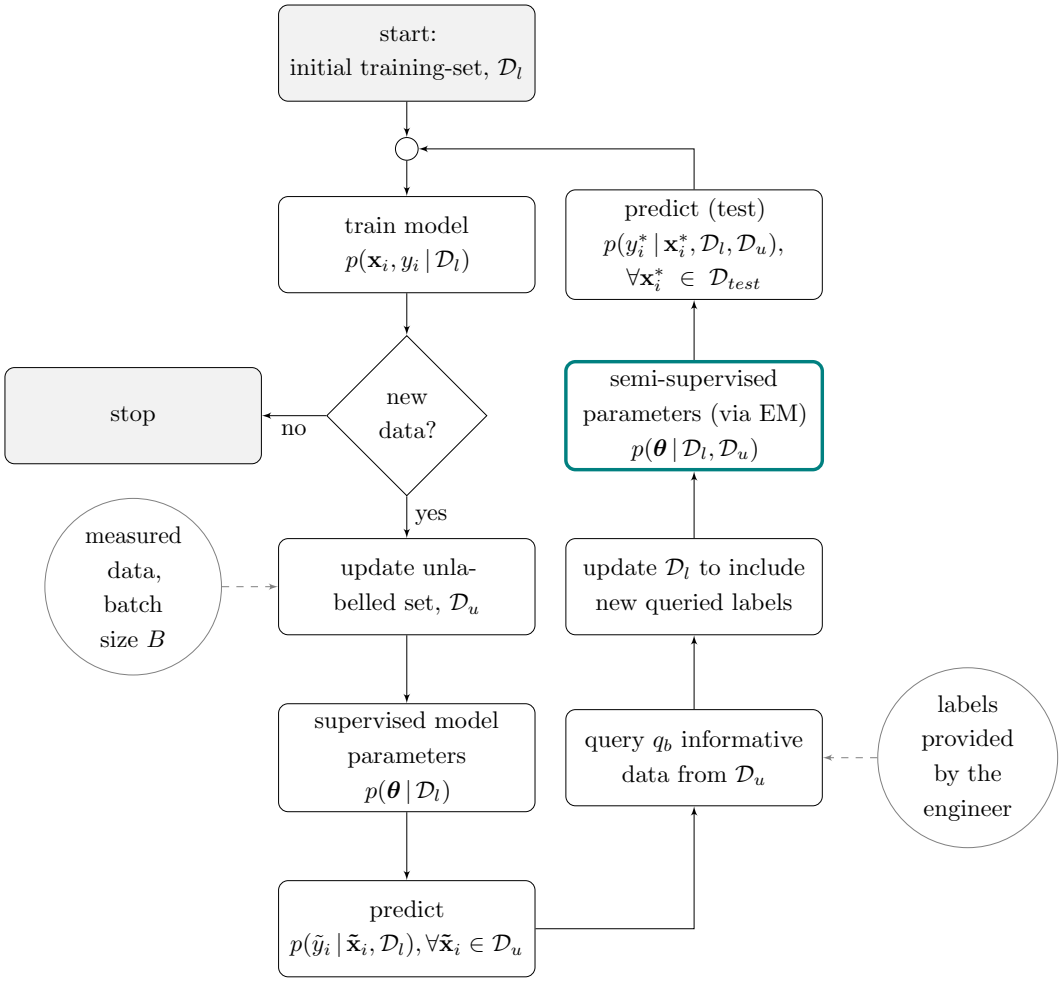


Figure 6.1: Flow chart for a online partially-supervised learner that is both semi-supervised and active. The green box highlights the additional (semi-supervised) step, compared to the online heuristic in Figure 4.4.

For the data used in these tests, initial experiments suggest that the (negative) effects of sampling bias increase if data are queried *after* the EM updates. As a result, in the proposed method, queries occur *before* the model becomes semi-supervised — this order is shown in Figure 6.1. It is hypothesised that sampling bias gets worse (if data are selected after EM), as queries should consider uncertainties given the labelled data only, i.e. $p(\tilde{y}_i \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l)$. Therefore, the unlabelled data (at this stage) appear to be unhelpful, as guided sampling looks to improve the information content of the *supervised set* $\mathcal{D}_l$.

6-1.1. Improved Uncertainty Sampling

Following the concluding remarks from Chapter 4, an alternative query method is proposed to try and reduce the influence of sampling bias; this attempts to introduce further variation in the training-set, while remaining focussed on uncertain examples. Similar to the query regime in Chapter 4, signals with high Shannon entropy,

$$H(\tilde{y}_i) = -\sum_{k=1}^{K} P(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) \log P(\tilde{y}_i = k \,|\, \tilde{\mathbf{x}}_i, \mathcal{D}_l) \tag{6.1}$$

and low likelihood,

$$p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l) = \sum_{k=1}^{K} p(\tilde{\mathbf{x}} \,|\, \tilde{y}_i = k, \mathcal{D}_l)\ p(\tilde{y}_i = k \,|\, \mathcal{D}_l) \tag{6.2}$$

associated with label predictions are considered the most informative. In this implementation, q_n signals are sampled from the *new batch* of unlabelled measurements with probability *proportional to* the (normalised) uncertainty measures — as opposed to the signals with the most extreme values of high-entropy (6.1) and low likelihood (6.2), as in Chapter 4. Formally, $q_n/2$ signals corresponding to each uncertainty measure are sampled, with probabilities such

that

$$\texttt{Select } \tilde{\mathbf{x}}_i \in \mathcal{D}_u \quad \text{with probability} \quad \mathbb{P}(\tilde{\mathbf{x}}_i) \propto \bar{H}(\tilde{y}_i) \tag{6.3}$$

$$\bar{H}(\tilde{y}_i) \triangleq \frac{H(\tilde{y}_i)}{\sum_i^m H(\tilde{y}_i)} \tag{6.4}$$

$$\texttt{Select } \tilde{\mathbf{x}}_i \in \mathcal{D}_u \quad \text{with probability} \quad \mathbb{P}(\tilde{\mathbf{x}}_i) \propto \frac{1}{\bar{p}(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l)} \tag{6.5}$$

$$\bar{p}(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l) \triangleq \frac{p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l)}{\sum_i^m p(\tilde{\mathbf{x}}_i \,|\, \mathcal{D}_l)} \tag{6.6}$$

As a result, similar to queries within the DH learner (Chapter 3), sampling has a finite probability of selecting any observation in $\mathcal{D}_u$ — uncertain or otherwise. Therefore, while sampling should favour uncertain signals, any observation could be queried — this should help protect against sampling bias.

6-2. Experiments and Discussion

In each application, the dataset is split in half (using every other observation) to define a distinct test-set $\mathcal{D}_{test}$ and a training-set $\mathcal{D}$, which arrive in batches (at the same rate) to represent streaming data. The test-data are used to assess the predictive performance online via the f_1 score (4.22) following model updates. The algorithm is limited to various query budgets, quoted as percentages (and ratios) of the total training-data. For each budget, four variations of the online framework are applied for comparison:

- Random sampling (**RS**) — the *passive learning* benchmark. Follows the framework presented in Figure 4.4 where q_n signals are selected from each batch at *random*.
- Semi-supervised learning (**RSEM**) — following random sampling, the parameters of the model are updated to consider the remaining unlabelled signals via EM, before predicting labels for the test-set.
- Active learning (**AL**) — Follows the framework presented in Figure 4.4, where q_n signals are selected using the uncertainty measures defined in Section 6-1.1 (high entropy and low likelihood). The unlabelled signals are *not* considered.

- Combined semi-supervised and active learning (**ALEM**) — extends the active learner (AL) to consider the remaining unlabelled signals via. EM, before predicting labels for the test-set.

All variations of the online algorithm are applied 50 times for each query budget. The plots represent the mean of the online predictive-performance (f_1 score) given the test-set.

Gnat Data

Firstly, the combined framework is applied to the genetic algorithm features from the Gnat data — the feature-set was extracted in [71] and introduced in Section 3-3.1. This projection is chosen as it is shown to benefit from semi-supervised updates in Chapter 5; as a result, the combined tests should highlight the effects of uncertainty sampling.

The data are ordered such that each damage-state proceeds the next $\{1, \dots, 9\}$, to imitate streaming SHM signals. It is acknowledged, however, that discrete, sequential data-groups do not represent streaming data in practice. In summary, this is a nine-class dataset, in a nine-dimensional feature-space, s.t. $\mathbf{x}_i \in \mathbb{R}^9$ and $y_i \in \{1, 2, \dots, 9\}$. To improve the separability of the data-groups, the data is projected (within the online framework) into the *full* principal-component space using (1.13).

The results are presented in Figure 6.2. With these data, the general f_1 scores are as follows: random sampling (RS) (i.e. passive learning) leads to the lowest online classification performance; this is improved by active learning (AL); however, semi-supervised updates (RSEM) (generally) provide a larger increase in performance; finally, the two-partially supervised methods combined (ALEM) lead to the best classifications. Theoretically, these results make sense: Section 5-3.2 shows that the model of these data improves when considering the unlabelled signals within a GMM; furthermore, uncertainty sampling (in this case) appears to further increase the predictive performance, agreeing with the experiments in Chapter 4. Most interestingly, compared to active learning, semi-supervised updates (via EM) lead to more significant increases in the f_1 score.

Considering these results, it appears logical to apply semi-supervised learn-

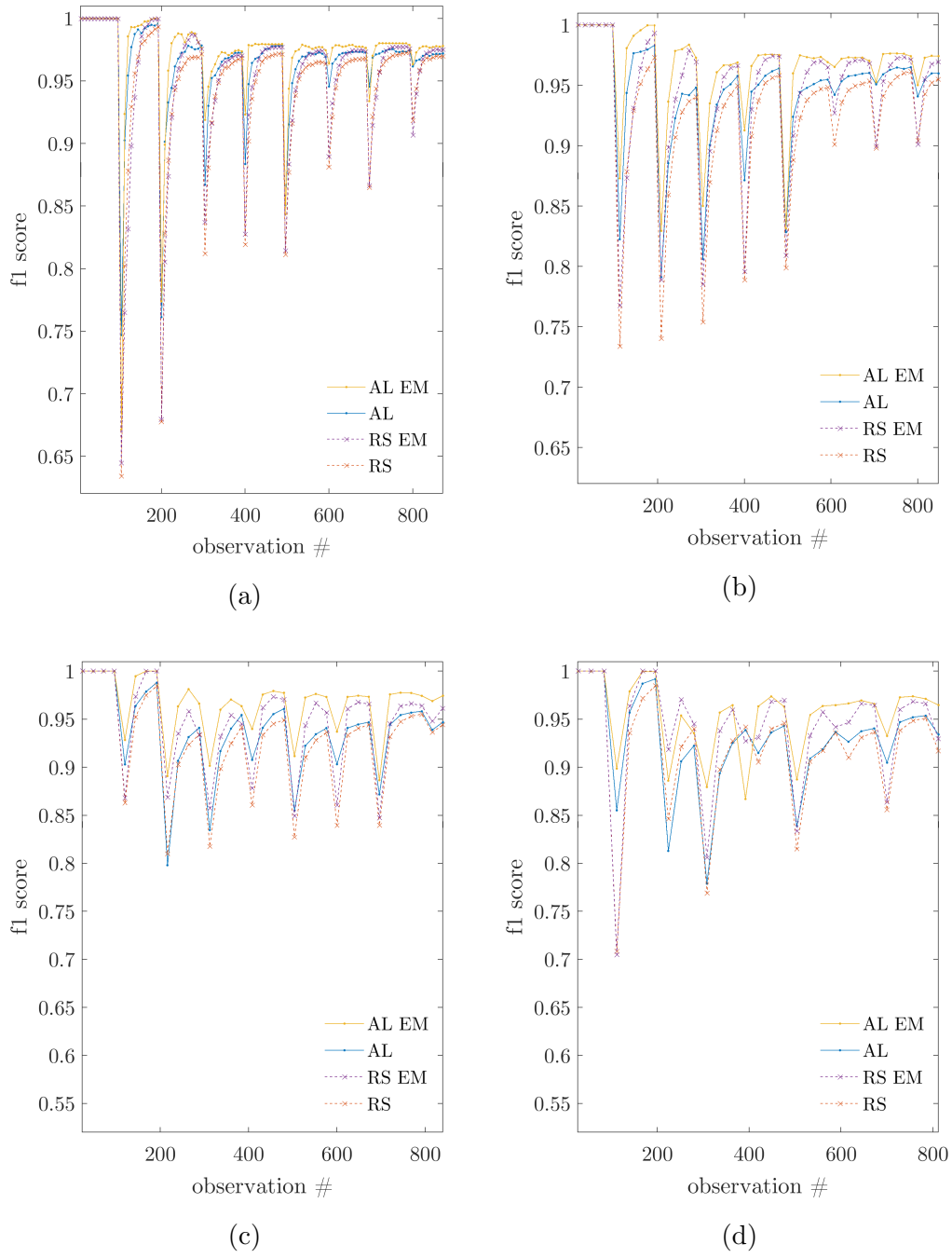


Figure 6.2: Online classification performance (f_1 score) for the Gnat data, for query budgets (as percentages and ratios of $\mathcal{D}$): (a) 25% (1:4); (b) 12.5% (1:8); (c) 8.33% (1:12), (d) 7% (1:14).

ing alone, as major increases in the performance follow EM updates, rather than uncertainty sampling. Furthermore, as a semi-supervised learner does not require guided sampling, the risks of sampling bias can be mitigated. Unsurprisingly, however, this effect is application-specific, and different behaviour is demonstrated in the following applications.

Machining Data

The machining data were introduced in Section 4-4.2; this is an acoustic emission dataset, recorded from a turning machine in operation during tool-wear tests [14]. The features are the same online variables extracted in Section 4-4.2 (via random projection); however, as with the Gnat data, the variables are also projected into the full principal-component space (1.13) to maximise variance. (The feature-space is $\mathbf{x}_i \in \mathbb{R}^{15}$ and the label-space, $y_i = \{1, \dots, 10\}$.) The dataset was shown to benefit from active learning in Chapter 4; therefore, tests here should highlight the advantages of EM updates within the online framework.

The results are presented in Figure 6.3. Similar trends can be observed; however, in this case, the most significant increase in the f_1 score appears to follow uncertainty sampling, rather than EM updates. Passive learning via random sampling (RS), and semi-supervised learning (without active learning, RSEM) leads to the lowest f_1 scores, particularly for lower query budgets (Figures 6.3c, 6.3d). As expected, active learning (AL) and semi-supervised active learning (ALEM) generally outperform the other two methods. EM updates often improve the predictive performance, most significantly at higher query budgets, Figures 6.3a, 6.3b.

With the machining data, however, semi-supervised updates can lead to inferior predictions, most notably when a new class is discovered (corresponding to drops in the f_1 score). It is believed that semi-supervised updates fail at this time, as EM will associate an inappropriate amount of unlabelled signals from *previous* batches with a new class when it is discovered. In fact, considering the model setup, this is not unreasonable — the priors (5.3) have been specified such that all classes are equally weighted across the *whole* dataset — this was done represent a general case. To prevent EM failing upon discovering a new class, the prior (5.3) could be adjusted, to associate less data with the new component; however, the influence of the new component must not be removed

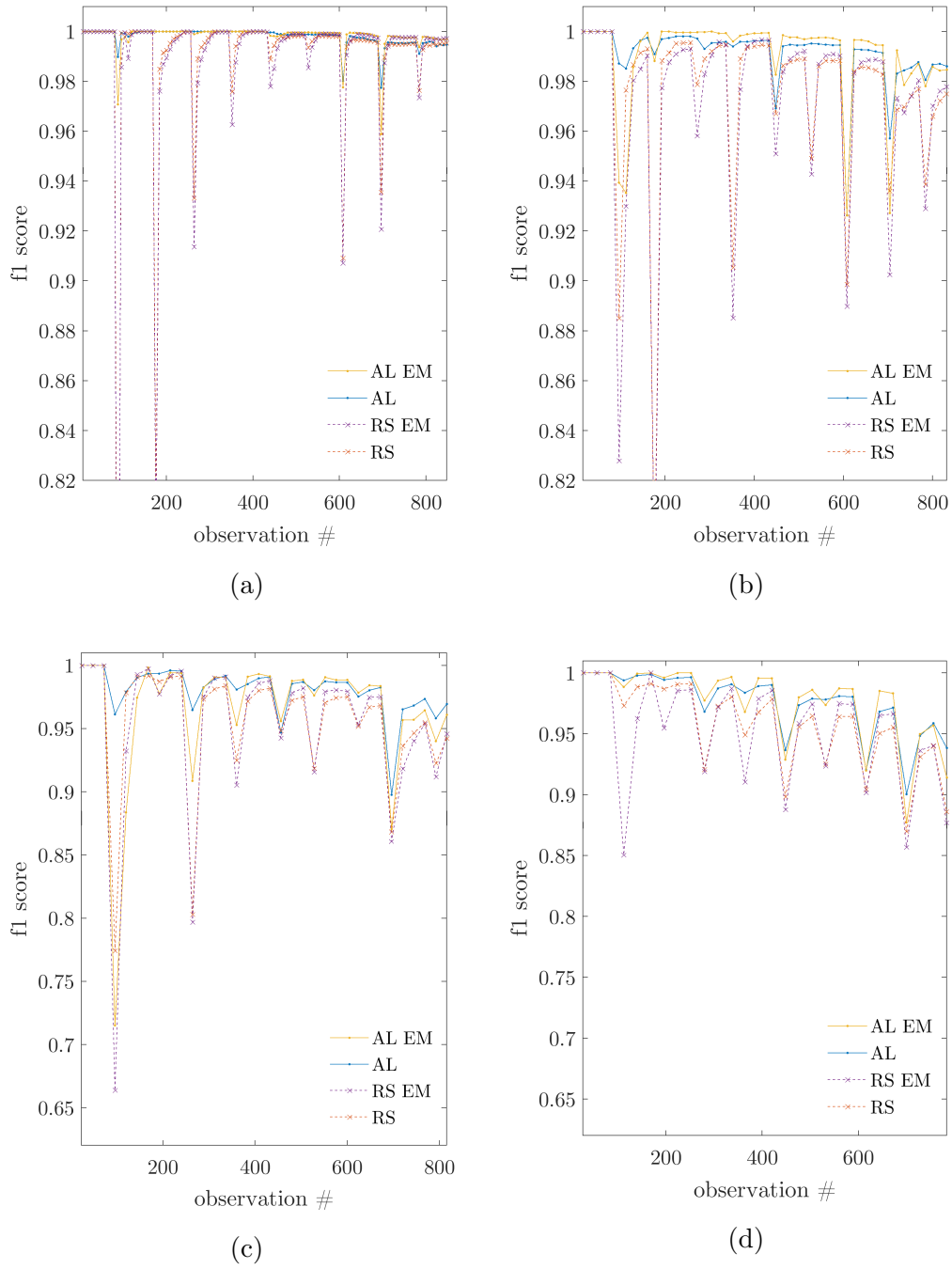


Figure 6.3: Online classification performance (f_1 score) for the machining data, for query budgets (as percentages and ratios of $\mathcal{D}$): (a) 25% (1:4); (b) 12.5% (1:8); (c) 8.33% (1:12), (d) 7% (1:14).

from the model when predicting *new* signals (which are likely to belong to the new group). This could be addressed by considering different priors for the novel and previous subsets of unlabelled data (as new signals are more likely to be associated with the newly discovered class).

Considering the Gnat and machining data applications, it seems that, generally, incorporating unlabelled signals within the online framework will improve the predictive performance of the GMM — with some potential tuning of the hyperparameters. Unfortunately, these examples do not represent a general case for all the data presented in this work: in the online setting, assuming a semi-supervised GMM appears too restrictive in certain feature-spaces, while active learning can still improve the performance (demonstrated with the Z24 data below).

Z24 Data

The Z24 data were introduced in Section 4-4.1. In summary, the data are labelled to represent a three-class classification problem: the first four natural frequencies of the bridge define the feature-space, s.t. $\mathbf{x}_i \in \mathbb{R}^4$. The label-space is s.t. $y_i \in \{1, 2, 3\}$,

- class 1: normal condition data,
- class 2: outlying data due to environmental effects,
- class 3: damage.

In the online setting, the data are shown to benefit from uncertainty sampling in Chapter 4; therefore, experiments here should highlight the effects of EM updates.

The results are presented in Figure 6.4. In agreement with Chapter 4, straight active learning (AL) improves the online f_1 score for all query budgets; however, the introduction of semi-supervised learning (RSEM and ALEM) reduces the classification performance throughout — a particularly bad example is presented here. In fact, incorporating unlabelled signals within the online GMM generally leads to a predictive performance that is inferior to passive learning via random sampling (RS). In this case, while the joint-likelihood given the labelled and unlabelled data (5.1) is increasing (a definition for EM updates [90]), the likelihood appears to be *negatively* correlated with the predictive

performance. This effect indicates that the density estimation (a GMM in this case) becomes inappropriate to model the underlying distribution of the data when considering the unlabelled instances.

In consequence, in agreement with Chapter 5, the results highlight that careful implementation is required for semi-supervised learning with mixture models, particularly with streaming data. While semi-supervised learning avoids sampling bias, it seems that the assumptions of the model become increasingly restrictive.

6-3. Concluding Remarks

In this chapter, experiments indicate that combining both active and semi-supervised learning can improve the predictive performance of a multi-class classifier for online SHM. The combined, partially-supervised algorithm is shown to increase the online f_1 score for the Gnat and machining data, while using a limited budget of labelled signals; however, the parametric assumptions (relating to the underlying distribution of the data) appear to become increasingly restrictive when unlabelled signals are used to constrain the classifier. Specifically, as demonstrated with the Z24 data, when the joint-log-likelihood of the model is maximised — given both the labelled and unlabelled signals — the online predictive performance can become worse than conventional (passive) learning. This reduced performance indicates that the joint-log-likelihood is *negatively* correlated with the f_1 score; therefore, the GMM becomes unrepresentative of the underlying distribution *when the information in the unlabelled signals is considered*.

In consequence, if the approximate form of the data distribution is unknown, a more general likelihood (i.e. base-distribution) *must* be used to describe the mixture model, as parametric likelihoods become increasingly restrictive when combining semi-supervised and active methodologies. The influence of the likelihood function is hardly surprising: if the data (labelled and unlabelled) do not represent a mixture of Gaussians in the feature-space, they cannot be modelled with a GMM. It is important to note, however, that the assumptions appear to become more restrictive when y_i is included as a latent variable for the unlabelled data. In consequence, the primary focus of future work should

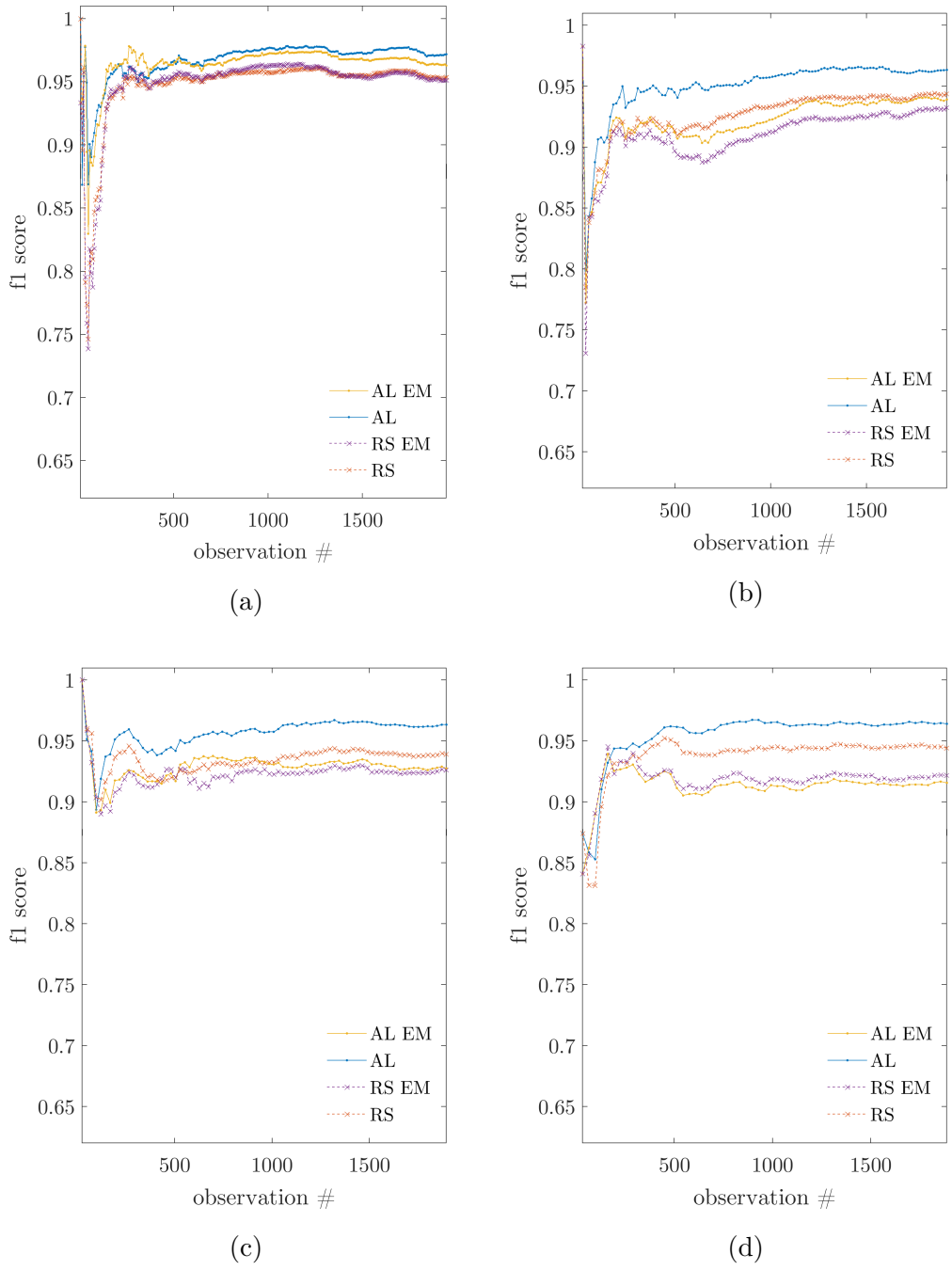


Figure 6.4: Online classification performance (f_1 score) for the Z24 data, for query budgets (as percentages and ratios of $\mathcal{D}$): (a) 25% (1:4); (b) 12.5% (1:8); (c) 8.33% (1:12), (d) 6.25% (1:16).

concern non-parametric methods for semi-supervised classification in online SHM. This might involve the use of discriminative classifiers, or non-parametric, generative mixture-models — these ideas are discussed in Chapter 7.

7

CONCLUSIONS

Overview: In the context of signal processing for Structural Health Monitoring (SHM), this work adapts methods for probabilistic and partially-supervised pattern recognition; specifically, **semi-supervised** and **active learning**. Typically in SHM, signals can be recorded from a system in operation; however, information to describe what measurements represent can be unavailable, particularly *a priori*. When following a data-based approach, this lack of information prevents the application of conventional supervised-learning algorithms, forcing a dependence on outlier analysis or damage detection in many practical applications.

In consequence, this research presents probabilistic machine learning tools to address multi-class classification in SHM, *when information to label the measured signals is limited*. Specifically, this research works towards the following:

1. An SHM strategy should be **adaptive**, incorporating any new classes (novel data-groups) as they are discovered, during system operation.
2. Therefore, the algorithm should be capable of learning and updating **online**; that is, it should be computationally-efficient, to update/adapt during system operation.
3. Model predictions should enable **accurate diagnostics** (ideally under uncertainty) while using a **limited** number of **labelled data**.

7-1. Summary

Chapter 3 : Hierarchical Sampling for Active Learning

To introduce the potential advantages of *both* semi-supervised and active learning for SHM, the DH algorithm [60] is applied to data from ground vibration tests concerning a Gnat aircraft [69]. The DH learner is a discriminative algorithm, which groups the measured signals into clusters of similar observations within the feature-space. Through active learning, a *limited* (but informative) set of signals are labelled, by querying observations in clusters that present uncertain classifications. When appropriate, semi-supervised learning is enforced, which associates the queried labels to any remaining unlabelled signals using the cluster-structure.

The experiments demonstrate that partially-supervised learning has the potential to significantly reduce the costs associated with labelling signals in SHM. There is a significant increase in the classification performance compared to conventional (passive) learning using the same budget of labelled data. Furthermore, the damage-classes to not need to be defined *a priori*, such that new groups of data can be add to the framework as they are discovered.

The algorithm is successful, however, it is limited in several respects for SHM. While *labels* for the measurements are not required *a priori*, a large set of observations is needed to build an informative cluster structure — this guides the partially-supervised aspects of the algorithm. As a result, the DH learner is less suitable for online applications with *streaming data*, where measured signals are also unavailable *a priori*; instead, the signals arrive incrementally during system operation. Additionally, as the DH learner follows a discriminative approach, updating/adapting the predictive model becomes problematic in the online setting.

Chapter 4 : Probabilistic Active Learning for Online SHM

Considering issues for online implementation, *generative mixture models* are adopted to work towards a partially-supervised, online framework. Conveniently, generative mixture models do not need to be (completely) retrained when a new class is discovered — a new component is simply added to the mixture.

Furthermore, unlabelled data can be naturally included within a probabilistic model under *well-defined uncertainty* (provided certain assumptions hold) — a significant advantage in risk-based applications.

Firstly, a mixture of Gaussians (GMM) used to define a probabilistic *active learner* for SHM — this framework is then extended to become semi-supervised in Chapter 6. The model initialises as a one component mixture, and adapts as new classes are discovered, leading to a generative, multi-class classifier. The training-set is extended by selecting signals from the data-stream that are uncertain in terms of (low) likelihood and (high) entropy (associated with the label predictions) — these data are assumed the most informative when updating the GMM. The framework is applied to three datasets — the Gnat data, the Z24 bridge data, and an acoustic emission dataset from machining experiments. In all cases, the data are presented as if they were recorded live from the systems in operation.

Results demonstrate that active learning can lead to significant increases in the *online* diagnostic-performance of a probabilistic multi-class classifier: the use of uncertainty sampling (based on entropy and likelihood[1] appears to select more informative training data than conventional passive learning (i.e. random sampling), and the variability in the classification performance is reduced. However, the experiments also demonstrate *sampling bias*[2]; that is, if queries become too focussed on specific regions of the feature-space (in this case, uncertain regions), the performance of active learning can become worse than passive learning; a clear example is shown for the Gnat aircraft data for (very) low query budgets. To address sampling bias, less restrictive methods for uncertainty sampling should be defined, to introduce variation in the training set — such as the method proposed in Chapter 6.

[1] In fact, the use of information metrics appears to offer various interesting options for future work in engineering, including experimental design, model updating, and system identification.

[2] This breaks the assumption of i.i.d training data, discussed in Section 7-2.

Chapter 5 : Towards Probabilistic and Semi-Supervised Damage Classification

Before extending the *online* GMM to become semi-supervised, the inclusion of unlabelled signals is introduced for the *offline* case. Specifically, following standard supervised learning, the information in the unlabelled measurements is incorporated via Expectation Maximisation (EM); this maximises the (MAP) joint-likelihood of the model given *both* the labelled and unlabelled data. In other words, semi-supervised EM extends the conventional *unsupervised* EM algorithm to consider the available labelled data.

The results indicate that, through semi-supervised mixture models, the cost associated with labelling data can be managed in SHM, as information in a small set of labelled data can be successfully combined with larger sets of unlabelled signals. The predictive-performance is shown to significantly increase when the underlying model of the data considers the information available in the unlabelled signals, rather than the labelled subset alone (standard supervised learning).

While the algorithm succeeds, care must be taken to ensure that the parametric mixture model (a GMM in this case) is appropriate. If the underlying distribution of the data cannot be approximated by a GMM, the structure imposed can lead to inferior predictions when the unlabelled data are considered — this issue becomes more apparent in the experiments of Chapter 6. Furthermore, like the DH learner, here the algorithm is only demonstrated in the offline setting; however, EM updates should combine naturally with the online, active-learning framework proposed in Chapter 4.

Chapter 6 : Towards a Combined Semi-Supervised and Active Learner

In the final experiments, active and semi-supervised methodologies are combined to define a partially-supervised generative mixture model, for *online* SHM. Again, the algorithm is applied to data that represent streaming signals, recorded from systems in operation. The *active* learner queries the most informative signals from the streaming data (uncertainty sampling), while *semi-supervised* updates (via EM) are added to include information in the remaining unlabelled

instances. To reduce the effects of sampling bias, measurements are sampled with a likelihood that is *proportional to* the uncertainty measures (low likelihood or high entropy) — rather than selecting the *most* uncertain examples of each case. In this way, all observations in the unlabelled set have a finite probability of being queried; therefore, variation within the training set is introduced.

For the simulated and machining datasets, experiments indicate that the (online) diagnostic performance improves with the combined classifier: generally, both active and semi-supervised steps improve the model predictions within the online framework. However, as suggested in Chapter 5, the assumptions of the parametric mixture model (GMM) appear to become increasingly restrictive *when unlabelled signals are considered* in the online case — an example of this is shown for applications to the Z24 bridge data. Therefore, if it is not possible to define an *appropriate* parametric mixture model given *a priori* domain knowledge, a more general likelihood function *must* be used, to approximate the underlying distribution of data, particularly for the online case; this issue is the primary focus for future work.

7-2. Limitations & Future Work

Through partially-supervised machine learning, this research successfully works towards multi-class classification in SHM, where the measured data are initially unavailable, and information to annotate the signals is limited. Referring again to the *contributions* in Section 1-5.1, the combined partially-supervised algorithm (presented in Chapter 6) addresses the following:

1. The algorithm is **adaptive**, such that novel data-groups can be included in the mixture model as they are discovered, and the number of classes do not need to be defined *a priori*.
2. Due to conjugate updates, the algorithm is capable of learning and updating **online**.
3. Provided the assumptions of the mixture model are appropriate, the classifier is capable of labelling predictions under well-defined uncertainty, while using a **limited** number of **labelled data** through active and semi-supervised methods.

Importantly, however, the assumptions of the mixture model must be considered; specifically:

- The components in the mixture model (i.e. each class of data) can be appropriately modelled by the selected base-distribution.
- The data used to train the algorithm are independent and identically distributed (i.i.d).

The first assumption appears to (mostly) affect semi-supervised learning. As partially-supervised mixture models have been *introduced* via the GMM, the data presented in this work are (intentionally) selected as they can be *approximated* by a mixture of Gaussians. Importantly, the 'true' distribution for these datasets is in fact unknown, and they are certainly non-Gaussian. Nonetheless, mixture models offer useful methods for density-estimation given prior knowledge of the expected feature-space. It should be considered, however, that including information from the unlabelled signals appears to lead to the model breaking down more rapidly than the supervised case.

The second assumption directly affects active learning. For an active learner (uncertainty sampling or otherwise), while the underlying data might be i.i.d, the queries are not, as the samples are *directed* by the algorithm given information from previous samples. Intuitively, a model is likely to become unrepresentative if it is trained given uncertain data only — these data are not a good reflection of the general underlying distribution. Despite issues, active learning has been shown to bring empirical improvements to classification performance, supported by this work; but, clearly, measures must be put in place to prevent more extreme cases of sampling bias — in this case, the model will become unrepresentative, even if the model selection is (somehow) perfect.

7-2.1. Future work

Model complexity

Considering the limitations introduced by parametric base-distributions, there are several options for future work. All of these involve increasing the complexity of the model (or decision boundary) within the generative framework, or moving to discriminative classifiers.

An obvious option is to increase the complexity of the base-distributions within the mixture model. This could be achieved by approximating each class with its own GMM, such that each group is represented by a multi-modal mixture of Gaussians (with a finite number of components). Alternatively, the base-distributions can be described with an *infinite* mixture of Guassians, i.e. a Dirichlet Process [91, 92]; in this way, each class is estimated by a GMM in which the number of components do not need to be defined *a priori* (the number of components, K, becomes an additional latent variable). The result is a *parameter free* method for density estimation, that can represent increasingly complex and multi-model distributions for each class. Such methods lead to intractable integrals for the marginal-likelihood; therefore, approximate inference [93, 94] must be implemented to estimate the posterior-distributions.

By implementing the Dirichlet Process as an *unsupervised* algorithm (such as the work in [8]), rather than a supervised mixture model, labelled data could be incorporated through modifications of the approximate inference. In this way, a parameter-free, partially-supervised mixture model could be implemented as *restraints on a clustering algorithm*; this approach allows for increasingly complex feature-spaces to be approximated, as only important clusters need to be labelled, and the label-set does not (necessarily) define the exact number of components.

A more significant change considers *discriminative* classifiers. Importantly, this does not eliminate probabilistic (or Bayesian) models; for example, Tipping's Relevance Vector Machine [95] could be adapted for partially-supervised learning in SHM; the RVM is a Bayesian treatment of the Support Vector Machine (SVM), which can address complex feature-spaces by modelling the decision-boundary directly. Additionally, algorithms for Gaussian process classification [96, 97] present further probabilistic, discriminative options. Also, the caveat that discriminative methods must be completely retrained is less relevant when approximate inference is required for (more complicated) generative classifiers.

Sampling Bias

The issues of non-i.i.d data, caused by sampling bias, could be further reduced through additional definitions of *informative* (as well as those suggested in this work), to extend the training data. Variation in the query regime should

help select observations that are informative while remaining representative. Alternatively, the query regime could adopt a framework that is based on change-point detection algorithms [98], as well as measures of uncertainty. Like uncertainty sampling, change-point detection can be based on probabilistic models, however, the associated methods can be used to establish *when* to query, rather than *which* observations. In this way, the active learner should be less susceptible to sampling bias, and the framework might naturally define an appropriate query budget for a given application.

References

[1] L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, 1987.

[2] E. N. Chatzi and A. W. Smyth. The unscented Kalman filter and particle filter methods for nonlinear structural system identification with non-collocated heterogeneous sensing. *Structural Control and Health Monitoring: The Official Journal of the International Association for Structural Control and Monitoring and of the European Association for the Control of Structures*, 16(1):99–123, 2009.

[3] J. E. Mottershead and M. Friswell. Model updating in structural dynamics: a survey. *Journal of Sound and Vibration*, 167(2):347–375, 1993.

[4] R. N. Clark. *Control System Dynamics*. Cambridge University Press, 1996.

[5] C. R. Farrar and K. Worden. *Structural Health Monitoring: a Machine Learning Perspective*. John Wiley & Sons, 2012.

[6] L. Bull, K. Worden, G. Manson, and N. Dervilis. Active learning for semi-supervised structural health monitoring. *Journal of Sound and Vibration*, 437:373–388, 2018.

[7] N. Dervilis, E. Papatheou, I. Antoniadou, E. Cross, and K. Worden. On the usage of active learning for SHM. In *Proceedings of ISMA2016*. Sheffield, 2016.

[8] T. Rogers, K. Worden, R. Fuentes, N. Dervilis, U. Tygesen, and E. Cross. A Bayesian non-parametric clustering approach for semi-supervised structural health monitoring. *Mechanical Systems and Signal Processing*, 119:100 – 119, 2019.

[9] N. Mechbal, J. S. Uribe, and M. Rébillat. A probabilistic multi-class classifier for structural health monitoring. *Mechanical Systems and Signal Processing*, 60:106–123, 2015.

[10] R. Zaurin and F. Necati Catbas. Structural health monitoring using video stream, influence lines, and statistical analysis. *Structural Health Monitoring*, 10(3):309–332, 2011.

[11] K. Worden and G. Manson. The application of machine learning to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):515–537, 2006.

[12] H. Sohn. Effects of environmental and operational variability on structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):539–560, 2006.

[13] L. Bull, T. Rogers, C. Wickramarachchi, E. Cross, K. Worden, and N. Dervilis. Probabilistic active learning: an online framework for structural health monitoring. *Preprint Submitted to Mechanical Systems and Signal Processing*, 2003.

[14] C. Wickramarachchi, T. McLeay, S. Ayvar-Soberanis, W. Leahy, and E. Cross. Tool wear inspection of polycrystalline cubic boron nitride inserts. In *Special Topics in Structural Dynamics, Volume 5*, pages 259–266. Springer, 2019.

[15] N. Dervilis, M. Choi, S. Taylor, R. Barthorpe, G. Park, C. Farrar, and K. Worden. On damage diagnosis for a wind turbine blade using pattern recognition. *Journal of sound and vibration*, 333(6):1833–1850, 2014.

[16] C. J. Stull, S. G. Taylor, J. Wren, D. L. Mascareñas, and C. R. Farrar. Real-time condition assessment of RAPTOR telescope systems. *Journal of Structural Engineering*, 139(10):1763–1770, 2012.

[17] E. Cross. *On Structural Health Monitoring in Changing Environmental and Operational Conditions*. PhD thesis, University of Sheffield, 2012.

[18] A. Rytter. *Vibrational Based Inspection of Civil Engineering Structures*. PhD thesis, Dept. of Building Technology and Structural Engineering, Aalborg University, 1993.

[19] K. Worden and J. M. Dulieu-Barton. An overview of intelligent fault detection in systems and structures. *Structural Health Monitoring*, 3(1):85–98, 2004.

[20] L. Qiu, S. Yuan, H. Mei, and F. Fang. An improved Gaussian mixture model for damage propagation monitoring of an aircraft wing spar under changing structural boundary conditions. *Sensors*, 16(3):291, 2016.

[21] Y. Huang, L. Gong, S. Wang, and L. Li. A fuzzy based semi-supervised method for fault diagnosis and performance evaluation. In *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1647–1651, July 2014.

[22] A. E. Bouzenad, M. El Mountassir, S. Yaacoubi, F. Dahmene, M. Koabaz, L. Buchheit, and W. Ke. A semi-supervised based k-means algorithm for optimal guided waves structural health monitoring: A case study. *Inventions*, 4(1), 2019.

[23] R. J. Barthorpe. *On Model-and Data-based Approaches to Structural Health Monitoring*. PhD thesis, University of Sheffield, 2010.

[24] E. N. Chatzi, B. Hiriyur, H. Waisman, and A. W. Smyth. Experimental application and enhancement of the xfem–ga algorithm for the detection of flaws in structures. *Computers & Structures*, 89(7-8):556–570, 2011.

[25] K. P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT press, 2012.

[26] U. Tygesen, K. Worden, T. Rogers, G. Manson, and E. Cross. State-of-the-art and future directions for predictive modelling of offshore structure dynamics using machine learning. In *Dynamics of Civil Structures, Volume 2*, pages 223–233. Springer, 2019.

[27] M. Friswell and J. E. Mottershead. *Finite Element Model Updating in Structural Dynamics*, volume 38. Springer Science & Business Media, 2013.

[28] A. Papoulis. *Probabilities, Random Variables, and Stochastic Processes*. McGraw-Hill, 2965.

[29] A. Farhidzadeh, S. Salamone, and P. Singla. A probabilistic approach for damage identification and crack mode classification in reinforced concrete structures. *Journal of Intelligent Material Systems and Structures*, 24(14):1722–1735, 2013.

[30] S. Rippengill, K. Worden, K. M. Holford, and R. Pullin. Automatic classification of acoustic emission patterns. *Strain*, 39(1):31–41, 2003.

[31] A. Gelman, H. S. Stern, J. B. Carlin, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2013.

[32] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[33] S. Rippengill, K. Worden, K. M. Holford, and R. Pullin. Automatic classification of acoustic emission patterns. *Strain*, 39:31–41, 2003.

[34] G. Manson, K. Worden, K. Holford, and R. Pullin. Visualisation and dimension reduction of acoustic emission data for damage detection. *Journal of Intelligent Material Systems and Structures*, 12:529–536, 2001.

[35] B. C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

[36] H. Sohn, C. R. Farrar, F. M. Hemez, D. D. Shunk, D. W. Stinemates, B. R. Nadler, and J. J. Czarnecki. A review of structural health monitoring literature: 1996–2001. *Los Alamos National Laboratory, USA*, 2003.

[37] S. Rogers and M. Girolami. *A First Course in Machine Learning*. CRC Press, 2016.

[38] K. Worden, C. R. Farrar, G. Manson, and G. Park. The fundamental axioms of structural health monitoring. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2082):1639–1664, 2007.

[39] F. Schwenker and E. Trentin. Pattern classification and clustering: a review of partially supervised learning approaches. *Pattern Recognition Letters*, 37(1):4–14, 2014.

[40] K. Worden, G. Manson, and N. R. Fieller. Damage detection using outlier analysis. *Journal of Sound and Vibration*, 229(3):647–667, 2000.

[41] N. Dervilis, E. Cross, R. Barthorpe, and K. Worden. Robust methods of inclusive outlier analysis for structural health monitoring. *Journal of Sound and Vibration*, 333(20): 5181–5195, 2014.

[42] M. M. Moya, M. W. Koch, and L. D. Hostetler. One-class classifier networks for target recognition applications. In *Proceedings World Congress on Neural Networks*, pages 359–367, 1993.

[43] D. M. Hawkins. *Identification of Outliers*. Springer, 1980.

[44] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley & sons, 2005.

[45] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[46] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems*, pages 841–848, 2002.

[47] O. Chapelle, B. Scholkopf, and A. Zien. *Semi-Supervised Learning*. MIT press, 2006.

[48] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[49] L. Bull, K. Worden, G. Manson, and N. Dervilis. Active learning for semi-supervised structural health monitoring. *Journal of Sound and Vibration*, 437:373–388, 2018.

[50] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-612-2.

[51] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, Boston, fourth edition, 2009.

[52] Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

[53] X. J. Zhu. *Semi-supervised Learning Literature Survey*. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[54] S. Chen, F. Cerda, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovacevic. Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring. *IEEE Transactions on Signal Processing*, 62(11): 2879–2893, June 2014.

[55] F. G. Cozman, I. Cohen, and M. C. Cirelo. Semi-supervised learning of mixture models. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 99–106, 2003.

[56] K. Nigam, A. McCallum, S. Thrun, T. Mitchell, *et al.* Learning to classify text from labeled and unlabeled documents. *AAAI/IAAI*, 792:6, 1998.

[57] A. K. McCallumzy and K. Nigamy. Employing EM and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning (ICML)*, pages 359–367. Citeseer, 1998.

[58] S. Dasgupta. Two faces of active learning. *Theoretical Computer Science*, 412(19): 1767–1781, 2011.

[59] M. Wang, F. Min, Z.-H. Zhang, and Y.-X. Wu. Active learning through density clustering. *Expert Systems with Applications*, 85:305–317, 2017.

[60] S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 208–215. ACM, 2008.

[61] J. Kremer, K. Steenstrup Pedersen, and C. Igel. Active learning with support vector machines. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4 (4):313–326, 2014.

[62] M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic. Active learning: an empirical study of common baselines. *Data Mining and Knowledge Discovery*, 31(2): 287–313, 2017.

[63] D. J. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[64] T. Hofmann and J. M. Buhmann. Active data clustering. In *Advances in Neural Information Processing Systems*, pages 528–534, 1998.

[65] H. T. Nguyen and A. Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 79. ACM, 2004.

[66] S. J. Huang, R. Jin, and Z. H. Zhou. Active learning by querying informative and representative examples. In *Advances in Neural Information Processing Systems*, pages 892–900, 2010.

[67] A. DasGupta. *Probability for Statistics and Machine Learning: Fundamentals and Advanced Topics*. Springer Science & Business Media, 2011.

[68] D. H. Wolpert and W. G. Macready. No free lunch theorems for search. Technical report, Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[69] G. Manson, K. Worden, and D. Allman. Experimental validation of a structural health monitoring methodology: Part III. damage location on an aircraft wing. *Journal of Sound and Vibration*, 259(2):365–385, 2003.

[70] P. Welch. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, 1967.

[71] K. Worden, G. Manson, G. Hilson, and S. Pierce. Genetic optimisation of a neural damage locator. *Journal of Sound and Vibration*, 309(3):529–544, 2008.

[72] L. Bull, G. Manson, K. Worden, and Dervilis. Active learning approaches to structural health monitoring. In N. Dervilis, editor, *Special Topics in Structural Dynamics, Volume 5*, pages 157–159. Springer International Publishing, 2019.

[73] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from data streams. *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 757–762, 2007.

[74] M. Song and H. Wang. Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. In *Intelligent Computing: Theory and Applications III*, volume 5803, pages 174–184. International Society for Optics and Photonics, 2005.

[75] G. D. Roeck. The state-of-the-art of damage detection by vibration monitoring: the SIMCES experience. *Structural Control and Health Monitoring*, 10(2):127–134, 2003.

[76] B. Peeters and G. De Roeck. One-year monitoring of the Z24-bridge: environmental effects versus damage events. *Earthquake Engineering & Structural Dynamics*, 30(2): 149–171, 2001.

[77] P. J. Rousseeuw and K. V. Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

[78] N. Ghosh, Y. Ravi, A. Patra, S. Mukhopadhyay, S. Paul, A. Mohanty, and A. Chattopadhyay. Estimation of tool wear during CNC milling using neural network-based sensor fusion. *Mechanical Systems and Signal Processing*, 21(1):466–479, 2007.

[79] Y. C. Eldar and G. Kutyniok. *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.

[80] A. Lazarevic and V. Kumar. Feature bagging for outlier detection. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 157–166. ACM, 2005.

[81] C. C. Aggarwal. Outlier ensembles: position paper. *ACM SIGKDD Explorations Newsletter*, 14(2):49–58, 2013.

[82] A. Zimek, M. Gaudet, R. J. Campello, and J. Sander. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 428–436. ACM, 2013.

[83] A. Zimek, R. J. Campello, and J. Sander. Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *ACM Sigkdd Explorations Newsletter*, 15(1):11–22, 2014.

[84] L. Bull, K. Worden, R. Fuentes, G. Manson, E. Cross, and N. Dervilis. Outlier ensembles: A robust method for damage detection and unsupervised feature extraction from high-dimensional data. *Journal of Sound and Vibration*, 453:126 – 150, 2019.

[85] C. C. Aggarwal and S. Sathe. *Outlier Ensembles: an Introduction*. Springer, 2017.

[86] H. V. Nguyen, H. H. Ang, and V. Gopalkrishnan. Mining outliers with ensemble of heterogeneous detectors on random subspaces. In *International Conference on Database Systems for Advanced Applications*, pages 368–383. Springer, 2010.

[87] S. Chen, F. Cerda, J. Guo, J. B. Harley, Q. Shi, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovacevic. Multiresolution classification with semi-supervised learning for indirect bridge structural health monitoring. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3412–3416, May 2013.

[88] A. Chebira, Y. Barbotin, C. Jackson, T. Merryman, G. Srinivasa, R. F. Murphy, and J. Kovavcevic. A multiresolution approach to automated classification of protein subcellular location images. *BMC Bioinformatics*, 8(1):210, Jun 2007.

[89] S. Mallat. *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, 3rd edition, 2008.

[90] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[91] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10(1-3):19–41, 2000.

[92] C. E. Antoniak. Mixtures of dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, pages 1152–1174, 1974.

[93] D. M. Blei, M. I. Jordan, *et al.* Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[94] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[95] M. E. Tipping. The relevance vector machine. In *Advances in Neural Information Processing Systems*, pages 652–658, 2000.

[96] H. Nickisch and C. E. Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

[97] J. Hensman, A. Matthews, and Z. Ghahramani. Scalable variational Gaussian process classification. 2015.

[98] R. P. Adams and D. J. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.