A Review of Meta-level Learning in the Context of Multi-component, Multi-level Evolving Prediction Systems

Abbas Raza Ali, Marcin Budka, Bogdan Gabrys

January 2015

A report written in the scope of the INFER project, a collaboration between Bournemouth University, Evonik Industries, and Research and Engineering Centre

Abstract

The exponential growth of volume, variety and velocity of data is raising the need for investigations of automated or semi-automated ways to extract useful patterns from the data. It requires deep expert knowledge and extensive computational resources to find the most appropriate mapping of learning methods for a given problem. It becomes a challenge in the presence of numerous configurations of learning algorithms on massive amounts of data. So there is a need for an intelligent recommendation engine that can advise what is the best learning algorithm for a dataset. The techniques that are commonly used by experts are based on a trial and error approach evaluating and comparing a number of possible solutions against each other, using their prior experience on a specific domain, etc. The trial and error approach combined with the expert's prior knowledge, though computationally and time expensive, have been often shown to work for stationary problems where the processing is usually performed off-line. However, this approach would not normally be feasible to apply on non-stationary problems where streams of data are continuously arriving. Furthermore, in a non-stationary environment the manual analysis of data and testing of various methods every time when there is a change in the underlying data distribution would be very difficult or simply infeasible. In that scenario and within an on-line predictive system, there are several tasks where Meta-learning can be used to effectively facilitate best recommendations including: 1) pre-processing steps, 2) learning algorithms or their combination, 3) adaptivity mechanisms and their parameters, 4) recurring concept extraction, and 5) concept drift detection. However, while conceptually very attractive and promising, the Meta-learning leads to several challenges with the appropriate representation of the problem at a meta-level being one of the key ones.

The goal of this review and our research is, therefore, to investigate Meta-learning in general and the associated challenges in the context of automating the building, deployment and adaptation of multi-level and multi-component predictive system that evolve over time.

Contents

Acknowledgements

The research leading to these results has received funding from the European Commission within the Marie Curie Industry and Academia Partnerships and Pathways (IAPP) programme under grant agreement n° 251617.

Chapter 1

Introduction

One of the major challenges in Machine Learning (ML) is to predict when one algorithm is more appropriate than another to solve a learning problem (Prudencio et al., 2011). Traditionally, estimating the performance of algorithms has involved intensive trial-and-error process which often demands massive execution time and memory together with the advise of experts that are not always easy to acquire (Giraud-Carrier et al., 2004). Meta-level Learning (MLL) has been identified as a potential solution to this problem Lemke et al., 2013a. It uses examples from various domains to produce a machine learning model, known as a Meta-learner, which is responsible for associating the characteristics of a problem with the most appropriate candidate algorithms found to have worked best on previously solved similar problems. The knowledge which is used by a Meta-learner is acquired from previously solved problems, where each problem is characterized by several features, known as Meta-features (MFs). MFs are combined with performance measures of learning algorithms to build a Meta-knowledge (MK) database. Learning at the base-level gathers experience within a specific problem, while MLL is concerned with accumulating experience over several learning problems (Giraud-Carrier 2008).

MLL started to appear in the machine learning literature in the 1980's and was referred to by different names like dynamic bias selection (Rendell et al., 1987), algorithm recommender (Brazdil et al., 2008), etc. Sometimes MLL is also used with a reference to ensemble methods (Duch et al., 2011) which can cause some confusion. So, in order to get a comprehensive view of exactly what MLL is, a number of definitions have been proposed in various studies. Vilalta and Drissi (2002a) and Vanschoren (2011) define MLL as the understanding of how learning itself can become flexible according to the domain or task and how it tends to adapt its behaviour to perform better. Giraud-Carrier (2008) describes it as the understanding of the interaction between mechanism of learning and concrete context in which that mechanism is applicable. Brazdil et al. (2008) view on MLL is that it is the study of methods that exploit Meta-knowledge to obtain efficient models and solutions by adapting the learning algorithms, while MK is a combination of characteristics and performance measures of Examples of Datasets (EoD). To some extent this definition is followed in this research as well.

Extracting MFs from a dataset plays a vital role in the MLL task. Several MF generation approaches are available to extract a variety of information from previously solved problems. The most commonly used approaches are descriptive (or simple), statistical, information theoretic, landmarking and model-based. The Descriptive, Statistical and Information-Theoretic (DSIT) features are easy to extract from the dataset as compared to the other approaches. Most of them have been proposed in the same period of time and are often used together in most of the studies. These approaches are used to assess similarity of a new dataset to previously analysed datasets (Bensusan et al., 2000). Landmarking is the most recent approach that tries to relate the performance of candidate algorithms to the performance obtained by simpler and computationally more efficient learners (Pfahringer et al., 2000). The Model-based approach attempts to capture the characteristics of a problem from the structural shape and size of a model induced from the dataset (Peng et al., 2002). The decision tree models are mostly used in this approach, where properties are extracted from the tree, such as tree depth, shape, nodes per feature, etc. (Giraud-Carrier, 2008).

The MF extraction approaches listed above are used in several implementations of decision-support systems for algorithm selection. One of the initial studies to address the practice of MLL was Machine Learning Toolbox (MLT) project by Graner et al. (1994). The project was a kind of expert system for algorithm selection which gathered user inputs through a set of questions about the data, the domain and user preferences. Although its knowledge-base was built through expert-driven knowledge engineering rather than via MLL, it still stood out as the first automatic tool that systematically relates application domain and dataset characteristics. In the same period, King et al. (1995) contributed with statistical and information theoretic measures based approach for classification tasks, known as Statistical and Logical learning (StatLog). A large number of MFs were used in StatLog together with a broad class of candidate models for the algorithm selection task. The project produced a thorough empirical analysis of various classifiers and learning models using different performance measures. StatLog was followed by various other implementations with some refinement in MF set, input datasets, Base-level Learning (BLL) and MLL algorithms. An EU funded research project Meta-Learning Assistant (METAL) had a key objective to facilitate a selection of the best suited classification algorithm for a data-mining task (Berrer et al., 2000). METAL introduced new relevant MFs and ranked various classifiers using statistical and information theoretic approaches. A ranking mechanism was also proposed by exploiting the ratio of accuracy and training time. An agent-based architecture for distributed Data Mining, Meta-learning Architecture (METALA), was proposed in (Botia et al., 2001). Its aim was the automatic selection of an algorithm that performs best from a pool of available algorithms by automatically carrying out experiments with each learner and task to induce a Meta-model for algorithm selection. The Intelligent Discovery Assistant (IDA) provided a knowledge discovery ontology that defined the existing techniques and their properties (Bernstein and Provost, 2001). IDA used three algorithmic steps of the knowledge discovery process, which included: 1) pre-processing, 2) data modelling, and 3) post-processing. It generated all valid processes and then a heuristic ranker could be applied to compute user-specified goals which were initially gathered as input. Later, Bernstein et al. (2005) research focused on extending Bernstein and Provost (2001) approach by leveraging the interaction between ontology to extract deep knowledge and case-based reasoning for MLL. One of the recent contributions to MLL practice was made by Mierswa et al. (2006) under Pattern Recognition Engineering (PaREn) project. A Landmarking operator was one of the outcomes of this project which was later embedded in RapidMiner. These systems are described in more detail in Section 2.4.4.

While there has been a lot of interest in MLL approaches and significant progress has been made, there are a number of outstanding issues which will be explained and some of which will be addressed. The main challenge of this work is a research on MLL strategies and approaches in the context of adaptive multi-level, multi-component predictive systems. This problem leads to several research challenges and questions which are discussed in detail in Chapter 3.

1.1 The review context and the INFER project summary

The research described in this report is closely related to and was conducted within the framework of the recently completed INFER[1] project. INFER stands for Computational Intelligence Platform for Evolving and Robust Predictive Systems and was a project funded by the European Commission within the Marie Curie Industry-Academia Partnerships & Pathways (IAPP) programme with a runtime from July 2010 until June 2014.

INFER project's research programme and partnership focused on pervasively adaptive software systems for the development of an open, modular software platform for predictive modelling applicable in different industries and a next generation of adaptive soft sensors for on-line prediction, monitoring and control in the process industry.

The main project goals were achieved by pursuing the following objectives within three overlapping research and partnership programme areas:

1. Area: Computational Intelligence – Objective 1: Research and development of advanced mechanisms for adaptation, increased robustness and complexity management of highly flexible, multi-component, multi-level evolving predictive systems.

2. Area: Software Engineering – Objective 2: Development of professionally coded INFER software platform for robust predictive systems building and intelligent data analysis.

3. Area: Process Industry / Control Engineering – Objective 3: Development of self-adapting and monitoring soft sensors for process industry.

When the project was starting in 2010, there were several freely accessible general purpose data mining and intelligent data analysis software packages and libraries on the market which could be used to develop predictive models, but one of their main drawbacks was that advanced knowledge of how to select and configure available algorithms was required. A number of commercial data mining/predictive modelling software packages were also available. These tools attempted to automate some steps of the modelling process (e.g. data pre-processing, handling of missing values or even model complexity selection) thus reducing required expertise of the user. Most of them were however either front-ends for a single data mining/machine learning technique or they were specialised tools designed specifically for use by a single industry. All these tools had one thing

[1] `http://infer.eu/`

in common – generated models were static and the lack of full adaptability implied the need for their periodic manual tuning or redesign.

The main innovation of the INFER project was therefore the creation and investigation of a novel type of environment in which the 'fittest' predictive model for whatever purpose would emerge – either autonomously or by user high-level goal-related assistance and feedback. In this environment, the development of predictive systems would be supported by a variety of automation mechanisms, which would take away as much of the model development burden from the user as possible. Once applied, the predictive system should be able to exploit any available feedback for its performance monitoring and adaptation.

There were (and still are) a lot of fundamental research questions related to the automation of data driven predictive models building, ensuring their robust behaviour and development of integrated adaptive/learning algorithms and approaches working on different time scales from real time adaptation to life long learning and optimisation. All of these questions provided the main thrust of advanced research conducted in the project and resulted in contributions to a large number (over 70) of high impact publications in top journals and international conferences. While all of the papers can be accessed via the project website (http://www.infer.eu) some of the key ones related to this review are listed below for easy access and reference. We split the publications using a set of distinct areas of interest and investigation and combine both the the older ones which led to the conception of the project in the first place and some which resulted from running the project. These are: i. complex adaptive systems and architectures (Gabrys et al., 2005; Ruta and Gabrys, 2007; Kadlec and Gabrys, 2009a; Zliobaite et al., 2012); ii. classifier and predictor ensembles (Ruta and Gabrys, 2002; Gabrys, 2002; Gabrys, 2004; Ruta and Gabrys, 2005; Gabrys and Ruta, 2006; Ruta and Gabrys, 2007; Riedel and Gabrys, 2007b; Budka and Gabrys, 2010b; Eastwood and Gabrys, 2012); iii. multi-level and multi-component predictors (Ruta and Gabrys, 2002; Riedel and Gabrys, 2005a; Riedel and Gabrys, 2005b; Riedel and Gabrys, 2007a; Riedel and Gabrys, 2009; Tsakonas and Gabrys, 2012; Lemke et al., 2013b; Tsakonas and Gabrys, 2013); iv. meta-learning (Lemke and Gabrys, 2010a; Lemke and Gabrys, 2010b; Lemke et al., 2013a, v. learning and adaptation in changing environments (Sahel et al., 2007; Kadlec et al., 2011; Tsakonas and Gabrys, 2011; Bakirov and Gabrys, 2013; Gama et al., 2014; Zliobaite and Gabrys, 2014); vi. representative data sampling and predictive model evaluation (Budka and Gabrys, 2010a; Budka et al., 2011; Budka and Gabrys, 2013); vii. adaptive soft sensors (Kadlec and Gabrys, 2008a; Kadlec and Gabrys, 2008b; Kadlec and Gabrys, 2008d; Kadlec et al., 2009; Kadlec and Gabrys, 2009b; Kadlec and Gabrys, 2009c; Kadlec and Gabrys, 2010; Kadlec and Gabrys, 2011; Kadlec et al., 2011; Budka et al., 2014) and viii. other application areas (Lemke and Gabrys, 2008; Lemke et al., 2009; Stahl et al., 2013; Salvador et al., 2014).

A variety of application areas and contexts have been used to illustrate the performance of developed approaches and/or to understand the mechanisms governing their behaviour. One of the key applications considered and tackled was that of adaptive soft sensors needed in the process industry.

The INFER software platform, developed with the creation of highly flexible, multi-component, multi-level evolving predictive systems in mind, supports parallel training, validation and execution of multiple predictive models, with each of them potentially being in a different state. Moreover, various optimization tasks can also be run in the background, taking advantage of idle computational resources. The predictive models running within the INFER platform are inherently adaptive. This means that they constantly evolve towards more optimal solutions as new data arrives. The importance of this feature stems from the fact, that real data is seldom stationary – it often undergoes various changes, which affect the relationships between inputs and outputs, rendering fixed predictive models unusable. A distinguishing feature of the INFER software is an intelligent automation of the predictive model building process, allowing non-experts to create well-performing and robust predictive systems with a minimal effort. At the same time, the system offers full flexibility for the expert users in terms of the choice, parameterisation and operation of the predictive methods as well as efficient integration of domain knowledge. While there is still a substantial development effort required before a viable commercial software product could be delivered the strong foundations have been created and it is our intention to build on them in the future.

More information on the INFER[2] project and its outcomes can be found following the link in the footnote.

The rest of the report is structured as follows. The next chapter covers the existing research in MLL area, including some important components of an MLL system. Those components include: 1. the sources of existing and ways of automatic generation of datasets, 2. Meta-feature generation and selection using various approaches, and 3. base-level learning algorithms performance measures, such as accuracy, execution time, etc. This is followed by sections discussing existing Meta-learning systems in the context of their applicability to the supervised and unsupervised algorithms. The last section of Chapter 2 illustrates the adaptive mechanisms aspect in detail. Based on the conclusions and recommendations extracted from the literature review, Chapter 3 describes research challenges of this work in the context of multi-component and multi-level adaptive systems. And finally the summary is provided in Chapter 4.

[2] `http://infer.eu/`

Chapter 2

Existing Research

A lot research has been conducted on automating Machine Learning (ML) algorithm selection in the last three decades. The focus of many of those studies is on various components of Meta-level Learning (MLL). Because of our particular interest in MLL, the scope of this literature review is confined to areas that are directly related to the MLL research. The high-level overview of the components which are discussed in this chapter is shown in Figure 2.1. The first section is discussing ways of gathering real-world datasets and techniques to create synthetic datasets which are known as Examples of Datasets (EoD). These EoD are used to generate Meta-features (MFs) and associated performance measures which are discussed in Sections 2.2 and 2.3 respectively. MF are combined with performance measures to build Meta-knowledge (MK) dataset which becomes the input of MLL. The last section illustrates adaptive mechanisms in the context of MLL which are an important aspect of our research focus.

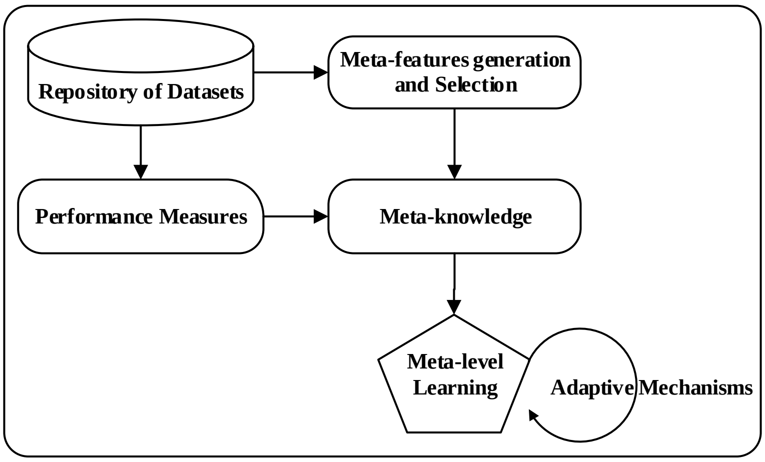

Figure 2.1: Scope of existing research review

2.1 Repository of Datasets

A repository of datasets representing various problems is one of the key components of the entire MLL system. As Vanschoren (2011) states, there is no lack of experiments being done, but the datasets and information obtained often remains in the *people's heads and labs*. This section explores the sources of real-world datasets that are used in the existing studies to build MK database. However, real-world datasets are usually hard to obtain but artificially generated datasets would be a possible solution of this problem. In the following subsections, studies that are dealing with the real-world data, those which elaborate the techniques to generate artificial datasets, and published resources are discussed.

2.1.1 Real-world Datasets

The real-world datasets can be difficult to find and gather in the desired format. An effort has been made to extract useful sources of data from various studies. Table 2.1 presents datasets that are used in different researches for MLL purpose. Most of them are gathered from UCI Machine Learning Repository (UCI) (Bache and Lichman, 2013).

Table 2.1: Real-world datasets used in various studies

Research Work	**Datasets**	**Sources**	**Dataset Filters**
King et al. (1995)	12	Satellite image, Hand written digits, Karhunen-Loeve digits, Vehicle silhouettes, Segment data, Credit risk, Belgian data, Shuttle control, Diabetes, Heart disease, German credit, Head injury (King, 1995)	
Lindner and Studer (1999)	80	UCI and DaimlerChrysler	
Sohn (1999)	19	Satellite image, Hand written digits, Karhunen-Loeve digits, Vehicle silhouettes, Segment data, Credit risk, Belgian data, Shuttle control, Diabetes, Heart disease, German credit, Head injury (King, 1995) and 7 other datasets used in StatLog project	Three datasets of StatLog having cost information involved in misclassification
Berrer et al. (2000)	58	Meta-Learning Assistant (METAL) project datasets	38 datasets with no missing values
Soares et al. (2001)	45	UCI and DaimlerChrysler	Dataset with more than 1000 instances
Bernstein and Provost (2001)	15	Balance Scale, Breast Cancer, Heart disease, Heart disease - compressed glyph visualization, German Credit Data, Diabetes, Vehicle silhouettes, Horse colic, Ionosphere, Vowel, Sonar, Anneal, Australian credit data, Sick, Segment data (Bache and Lichman, 2013)	
Todorovski et al. (2002)	65	UCI and METAL project datasets	38 datasets with no missing values
Brazdil et al. (2003)	53	UCI and DaimlerChrysler	Datasets with more than 100 instances

Bernstein et al. (2005)	23	Balance Scale, Heart disease, Heart disease, Heart disease - compressed glyph visualization, German Credit Data, Diabetes, Vehicle silhouettes, Ionosphere, Vowel, Anneal, Australian credit data, Sick, Segment data, Robot Moves, DNA, Gene, Adult 10, Hypothyroid, Waveform, Page blocks, Optical digits, Insurance, Letter, Adult (Bache and Lichman, 2013)	
Peng et al. (2002)	47	UCI	
Kopf and Iglezakis (2002)	78	UCI	Dataset with less than 1066 instances and the number of attributes ranged from 4 to 69
Prudencio and Ludermir (2004)	I: 99 Time-series (TS) and II: 645	I: Time-series Data Library[1] and II: M3 competition[2]	I: Stationary data and II: Yearly data
Prudencio and Ludermir (2008)	50	WEKA project[3]	On average datasets contain 4,392 instances and 14 features
Wang et al. (2009)	46 and 5	Time Series Data-mining Archive[4] and Time Series Data Library[5]	
Kadlec and Gabrys (2009a)	3	Thermal oxidiser, Industry drier, and Catalyst activation datasets of process industry	On-line prediction datasets
Lemke and Gabrys (2010a)	2 consisting of 111 TS	NN3[6] - Monthly business with 52-126 observations and NN5[6]- daily cash machine withdrawals with 735 observations in each series	NN5 including some missing values
Abdelmessih et al. (2010)	90	UCI	Datasets with more than 100 instances
Duch et al. (2011)	5 and 2	Leukemia, Heart, Wisconsin, Spam, and Ionosphere are real-world datasets gathered from UCI and two synthetic datasets parity and monks	
Rossi et al. (2014)	2	Travel Time Prediction (TTP) consists of 24,975 instances and Electricity Demand Prediction (EDP) consists of 27,888 instances	

Warden (2011) wrote a concise handbook that covers the most useful sources of publicly available datasets. A lot of new sources of free and publicly available data that have emerged over the last few years are discussed. Apart from discussing data-sources, methods to get datasets in bulk from those sources are also discussed in detail. Table 2.2 presents most of the sources from the author's book.

[1] `http://datamarket.com/data/list/?q=provider:tsdl`

[2] `http://forecasters.org/resources/time-series-data/m3-competition`

[3] Machine Learning Group at University of Waikato `http://www.cs.waikato.ac.nz/ml/weka`

[4] `http://www.cs.ucr.edu/~eamonn/time_series_data`

[5] `http://datamarket.com/data/list/?q=provider:tsdl`

[6] Neural Network forecasting competition

Table 2.2: List of publicly available Data Repositories

Source	**Description**	**Datasets**	**Industry**
AnalcatData	Datasets that are used by Jeffrey S. Simonoff in his book *Analyzing Categorical Data*, published in July 2003	83	Cross-industry
Amazon Web Services	A centralized repository of public datasets		Astronomy, Biology, Chemistry, Climate, Economics, Geographic and Mathematics
Bioassay data	Virtual screening of bioassay (active/inactive compounds) data by Amanda Schierz	21	Life Sciences
Canada Open Data	Canadian government and geospatial data		Government & Geospatial
Datacatalogs	List of the most comprehensive open data catalogs		
data.gov.uk	Data of UK central government departments, other public sector bodies and local authorities	9616	Government and Public Sector
data.london.gov.uk	Data of UK central government departments, other public sector bodies and local authorities	563	Government and Public sector
Data.gov/Education	Educational high-value datasets	70,897	Cross-industry
ELENA	Non-stationary streaming data of flight arrival and departure details for all the commercial flights within the USA	13 features and 116 million instances	Aviation
KDD Cup	Annual Data Mining and Knowledge Discovery competition datasets		cross-industry
National Government Statistical Web Sites			
Open Data Census US Census Bureau	Assesses the state of open data around the world		Government and Public sector
OpenData from Socrata	Freely available datasets	10,000	Business, Education, Government, Social and Entertainment
Open Source Sports	Many sports databases, including Baseball, Football, Basketball and Hockey		Entertainment
UCI	A collection of databases, domain theories, and data generators that are used by the ML community for the empirical analysis of learning algorithms	199	Physical Sciences, Computer Science & Engineering, Social Sciences, Business and Game
Yahoo Sandbox datasets	Language, graph, ratings, advertising and marketing, competition, computing systems and image datasets		Cross-industry

2.1.2 Synthetic Datasets

MFs are used as predictors in an MLL system. Typically, many MFs are extracted from a dataset, thereby leading to a high-dimensional sparsely populated feature space which has always been a challenge for learning algorithms. Hence, to overcome this problem sufficient number of datasets is required which may not be possible only from the repositories of the real-world datasets as they can be hard to obtain. So, artificially generated datasets might help in solving this issue. Rendell and Cho (1990) work on systematic artificial data generation is considered as one of the initial efforts in this regard.

Bensusan and Giraud-Carrier (2000) used 320 artificially generated boolean datasets with 5 to 12 features in each one. These artificial datasets were benchmarked on 16 UCI and DaimlerChrysler real-world datasets. Similarly Pfahringer et al. (2000) generated 222 datasets, each containing 20 numeric and nominal features having 1K to 10K instances classified between 2 to 5 classes. Additionally 18 real-world UCI problems were used to evaluate the proposed approach.

Soares (2009) proposed a method to generate a large number of datasets by transforming the existing datasets, known as *datasetoids*. An artificial dataset was generated against each symbolic attribute of a given dataset, obtained by switching the selected attribute with the target variable. This method was used on 64 datasets gathered from the UCI repository and it generated a total of 983 *datasetoids*. At the end potential anomalies related to the artificial datasets were also discussed as well as their proposed solutions were presented. Those identified anomalies were: 1) the new target variable having missing values, 2) the target variable being very skewed, and/or 3) the corresponding target variable being completely unrelated to the remaining features. One very simple solution proposed for these problems was to simply discard the *datasetoids* which showed any of the above mentioned properties. This method produced promising results, therefore enabling the generation of new datasets which could solve the scarce datasets problems.

Wang et al. (2009) used both synthetic and real-world Time Series (TS) from diverse domains for MLL based forecasting method selection study. The details of real-world datasets are given in Table 2.1 while remaining synthetic datasets were generated using statistical simulation to facilitate the detailed analysis of forecasting association with data characteristics. A total of 264 artificial datasets were generated to exhibit a number of different characteristics including, for instance, perfect and strong trend, perfect seasonalityor certain type and level of noise. The data was transformed into a sample of 1000 instances for each of the original TSs while it was unchanged for the number of data-points smaller than 1000.

Soares et al. (2009) generated 160 artificial datasets to obtain a wide range of cluster structures. There were two methods used to generate datasets: 1) a standard cluster model using Gaussian multi-variate normal distributions, and 2) Ellipsoid cluster generator. There were three parameters selected for both techniques including: i) the number of clusters which were the same for both cases (2, 4, 8, 16), ii) dimensions (2, 20 for Gaussian, and 50, 100 for Ellipsoid), and iii) the size of each cluster for both techniques were the same (uniformity in [10, 100] for 2 and 4 clusters case and [5, 50] for 8 and 16 clusters case). For each of the 8 combinations of cluster number and dimension, 10 different instances were generated, giving 80 datasets in each method.

Duch et al. (2011) used two artificially generated datasets out of a total of seven whereas the remaining five were the real-world problems. One artificially generated dataset had binary features, named as *Parity*, whereas the other one with nominal features was known as *Monks*. These support features are computed using Quality of Projected Clusters (QPC) projection.

Reif et al. (2012a) presented a novel data generator approach for numerical features and classification datasets that could be used as input dataset for MLL which represented an entirely different approach from Soares (2009). The proposed system was able to generate datasets using genetic programming with customized parameters. In the proposed setting MLL could be supported in two different ways: 1) the MFs space could be filled in a more controlled way and the discovered ”empty areas” could be populated rather than generating random datasets, and 2) thoroughly investigating MFs based on their descriptive power which could be useful for certain MLL problems and generating datasets with MFs allowed more controlled experiments that might lead to a significant utilization of particular MFs. Since the dataset was generating multiple MFs therefore this task was treated as multi-objective optimization problem. The proposed system was able to incorporate a variable set of arbitrary MFs. The user was able to build a custom set of MFs simply by providing the functions that compute the MFs.

2.1.3 Datasets from Published Research

Another source of EoD are the published ML studies. As ML has been one of the most active research areas in the last few decades where several experimentations have been conducted, these experiments become a very useful way of gathering EoD representing various domains. The additional benefit that usually comes with most of the datasets used in existing ML benchmarking and experimental studies is the relative ranking and predictive performance data for the evaluated ML algorithms. It is of particular interest as the ML algorithms performance data is used and needed as as a target variable in the context of an MLL system. It is very time, memory and processor consuming task to compute performance measures for massive amount of datasets and numerous configurations of learning algorithms.

Usually, presumably due to space limitations on publications, researches publish only the final results with minimal details. However, in the context of MLL, relying on such minimal information leads to several problems, for example, in most of the instances researches only report the best algorithm, usually report limited number and detail of experimentations, mostly skip detailed configurations of the algorithms, etc. Vanschoren et al. (2014) introduced a novel platform for ML research known as OpenML. ML researchers can share datasets, algorithms, their configurations, and experiment setups on this platform which other researchers can use to compare results. OpenML framework is one of the possible solutions for most of the mentioned concerns which resolves two key challenges of MLL systems: i) gathering a large number of datasets from different domains, and ii) performances of the tested ML algorithms on these datasets.

2.1.4 Discussion and Summary

An ML system relies on a good training dataset to build a reliable and well performing predictive model. Similarly, at the Meta-level, the MK dataset is used as a training-set of MLL, and the quality of this MK dataset is dependent on sufficient number and quality of EoD from different domains. These EoD are used to generate MFs which act as predictors and the estimated predictive performance evaluated ML algorithms for these EoD are used as the target variable in the MK dataset. However gathering sufficient number of real-world datasets is quite difficult. The real-world datasets which are used in various studies for experimentations are listed in Table 2.1. Most of the studies gathered datasets from the UCI with different filtering options and the remaining few studies gathered datasets from different data-mining competitions. In most cases the number of EoD that are used to build MK has been very low. However, as identified and shown in Table 2.2 there are various sources from which a relatively large (and quickly growing) number of real-world datasets representing different domains could be beneficially used in the future though they have not been used so far.

Some MLL researches resolved the problem of the number and quality of available datasets by building their MK datasets using artificially generated EoD. They have adopted two different approaches to generate these synthetic datasets: 1) by transforming real-world datasets; and 2) by utilizing statistical and genetic programming approaches. Bensusan and Giraud-Carrier (2000), Pfahringer et al. (2000), Soares (2009) and Wang et al. (2009) proposed different feature transformation approaches to generate different combinations of datasets from the limited number of real-world datasets. The statistical and genetic programming approaches were proposed by Soares et al. (2009) and Duch et al. (2011) for MLL systems. In some of the approaches statistical functions with a threshold (or cut-off) values are used to generate data while others used optimization techniques. Reif et al. (2012a) proposed an intelligent technique which does not generate random data, but fill the MFs in a more controlled way by discovering and populating the empty areas within the real-world datasets.

Combining all the proposed approaches iteratively could offer a potential solution to the dataset scarcity; i.e., initially gathering the existing available real-world problems, then transforming those datasets by generating several others and finally applying various other techniques to generate artificial datasets independently (see Figure 2.2). Although this solution could be useful if the purpose would be only gathering a large number of EoD, in the context of the MLL research the predictive performance data for numerous learning algorithms and their configurations is needed and not normally readily available. Considering all three necessary components of an MLL system, gathering datasets from published experimental evaluations and benchmarking of ML algorithms would seem to be more attractive, however, there are a lot of challenges with such data related to reporting only the best learning algorithms, publishing limited information of experimentations, availability of datasets used in the research, lack of detailed configurations of evaluated learning algorithms, etc. OpenML platform has attempted to address most of these issues focusing on the consistency and completeness of the gathered information but as it is in a preliminary stage of development it currently lacks sufficiently large number of problems from different domains and sufficiently robust and comprehensive number of machine learning algorithms tested for each of the datasets to be very useful in its current form.

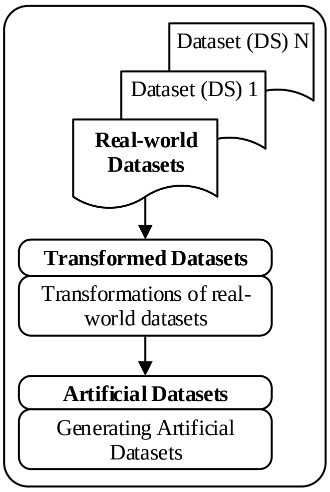

Figure 2.2: Phase-wise collection of Examples of Datasets

2.2 Meta-features Generation and Selection

One of the primary applications of Meta-level Learning (MLL) is to recommend the best learning algorithm or to rank various ML algorithms for a new problem without the need for executing and evaluating these learning algorithms on the problem at hand. The role of such systems is to identify previously solved similar problems, and with the assumption that the previously found best algorithms will also work best on the new problem, make appropriate recommendations. As directly comparing large and complex datasets is normally infeasible, the similarity between different problems/datasets is carried out using a number of so called Meta-features (MFs) offering a simplified representation of the problems/datasets. There are three most commonly used MF generation approaches which allow to induce a mapping between the characteristics of a problem and the best performing learning algorithms for the problem. These approaches are discussed in the following sections.

2.2.1 Descriptive, Statistical and Information-Theoretic Approach

The Descriptive, Statistical and Information-Theoretic (DSIT) approach is the simplest and the most commonly used MF generation approach that extracts a number of DSIT based MF values directly from a dataset representing an ML problem. The DSIT based MFs and the related MLL approaches primarily based on such MFs are reviewed below.

Rendell et al. (1987) proposed Variable-bias Management System (VBMS) that was one of the earliest efforts towards data characterization. Only two descriptive MFs, namely: the number of training instances and the number of features, were used to select the best among three symbolic learning algorithms. Later Rendell and Cho (1990) enhanced the existing system by adding useful MFs of complexity based on shape, size and structure. Statistical and Logical learning (StatLog) project by King et al. (1995) further extended VBMS MFs by considering a larger number of dataset characteristics. A problem was described in the context of its descriptive and statistical properties. Several characteristics of a problem spanning from simple (descriptive) to more complex (statistical) ones were generated and later used by various studies. These characteristics were used to investigate why certain algorithms perform better on a particular problem as well as to produce thorough empirical analysis of the learning algorithms.

Sohn (1999) initially used most of the datasets and MFs that were used in StatLog project which were later on enhanced with information-theoretic MFs. Furthermore, three new descriptive features were added by transforming the existing MFs, for example in the form of ratios. These MFs were used to rank several classification algorithms with considerably better performance as compared to the previous studies. It was also claimed that the classification error and execution-time are important response variables to choose a suitable classification algorithm for a problem.

In the same year Lindner and Studer (1999) proposed an extensive list of DSIT based MFs of a problem under the name of Dataset Characterization Tool (DCT). The authors distinguished three categories of dataset characteristics, namely simple, statistical and information-theory based measures. The descriptive MFs have been used to extract general characteristics of the dataset, whereas statistical characteristics were mainly extracted from numeric attributes, while information-theoretic based measures from nominal attributes. A Case-based Reasoning (CBR) approach to select the most suitable algorithm for a given problem was also proposed.

Reif et al. (2012b) presented a novel approach of generating informative MFs by simply averaging over all attributes of the source datasets. They proposed a two-fold approach. In the first fold DSIT based MFs are generated using the previously introduced traditional approach. The second fold is used to describe the differences over datasets that are not accessible using the typically used mean of MFs that have been computed in the first fold. This approach preserves more information of such MFs while producing a feature vector with a fixed size. An additional level of MFs selection is proposed to automatically select the most useful MFs out of the initially generated ones. All MFs that are used in the above studies are shown in Figure 2.3.

2.2.2 Landmarking Approach

Another technique of MF generation is Landmarking which characterizes a dataset using the performance of a set of simple learners. Its main goal is to identify areas in the input space where each of the simple learners can be regarded as an expert (Vilalta and Drissi, 2002a).

The basic idea behind Landmarking is to use the estimated performance of a learning algorithm on a given task for discovering additional information about its nature. A landmark learner or landmarker is defined as the learning mechanism whose performance is used to describe a problem (Bensusan and Giraud-Carrier, 2000). Landmarkers posses a key property that their execution time is always shorter than the Base-learner's time, otherwise this approach would bring no benefit. In the remaining parts of this section, various studies dealing with Landmarking approaches are discussed in detail.

One of the earliest studies on Landmarking was conducted by (Bensusan and Giraud-Carrier, 2000). This approach is claimed to be simpler, more intuitive and effective than the DSIT measures. A set of 7 landmarkers were trained on 10 different sets of equal size. Each dataset was then described by a vector of MFs (see Landmarkers branch of Figure 2.3), which are the error rates of the 7 landmarkers, and labelled by the target learners (see Table 2.3) which produce the highest accuracy. Several experimentations have been performed to compare landmarking approach with DSIT. In the first experiment Landmarking was compared with 6 information-theoretic DCT features of Lindner and Studer (1999) (see information-theoretic MFs section of Figure 2.3). In most of the cases of this experiment landmarking outperformed the DSIT based approach. In another experiment the ability of landmarking to describe a problem and discriminate between two areas of expertise are highlighted. In most of the cases C5.0 Adaptive Boosting (C5.0 boost) (Quinlan, 1998) landmarker performed best. The last experiment benchmarked 16 real-world datasets from the UCI Machine Learning Repository (UCI) (Bache and Lichman, 2013) and the DaimlerChrysler where again the landmarking approach resulted in the best overall performance.

Pfahringer et al. (2000) also evaluated a landmarking approach while comparing it with the DSIT MF generation approach - DCT. They performed three types of experiments, namely: 1)

Artificial rule list and sets generation; 2) Selecting learning models; and 3) Comparing the landmarking with the information-theoretic approach. These experiments were almost the same as performed by Bensusan and Giraud-Carrier (2000), and the target learners (see Table 2.3) were the same as used in Meta-Learning Assistant (METAL) project. In the first experiment the set of landmarkers consisted of a Linear Discriminant Analysis (LDA), Naive Bayes and C5.0 Decision Tree (C5.0 tree) learners, while the base-learners performance relative to each other was predicted using C5.0 boost, LDA, and Rule Learner (Ripper). In addition to 3 landmarkers, 5 descriptive MFs (shown in the descriptive approach in Figure 2.3) have also been extracted from 216 datasets. The Ripper was found to be the top performer in this experimentation. For selecting the best learning model experiment, authors tried to investigate the capability of landmarking in deciding whether a learner involving multiple learning algorithms performs better than the other candidate algorithms. Here only C4.5 Decision Tree algorithm (C4.5) was used as a Meta-learner trained with 222 artificial boolean datasets and tested with 18 UCI problems (Bache and Lichman, 2013). Even though the landmarking accuracy was higher it did not have a significant effect on the overall performance of a system whose ultimate goal is to accurately select the best learning model. In the last experiment, the landmarking approach was compared with the DSIT and also the combination of both approaches. 320 artificially generated binary datasets were produced where the combined approach performed best for all 10 Meta-learners followed by the landmarking with significant difference as compared to DCT approach.

Soares et al. (2001) sample-based landmarkers used estimates of the performance of algorithms on a small sample of the data and then had been used as the predictors of the performance of those algorithms on the entire dataset. Additionally, a relative landmarker addressed the inability of the earlier landmarker to assess relative performance of algorithms. This sampling-based relative landmarking approach was later compared with the DSIT DCT MFs (Lindner and Studer, 1999) as done by most of the landmarking studies. The ten algorithms, listed in Table 2.3, wer used on 45 datasets, with more than 1000 instances, mostly gathered from the UCI (Bache and Lichman, 2013) and the DaimlerChrysler repositories. These datasets have been ranked by the *Nearest-Neighbour* using Adjusted Ratio of Ratios (ARR) measure. To observe the performance of the ranking method, the authors varied the value of k from 1 to 25. In comparison with other studies reported in the literature, the sample-based relative landmarking approach showed improvements in the algorithm ranking task as compared with the traditional DCT measures.

Kopf and Iglezakis (2002) proposed a new approach for assessing the quality of case bases constructed using landmarking and DCT based MFs. The meta-learner was based on case-base reasoning approach using the quality assessed cases. Tasks were described by their similarity, consistency,incoherency, uniqueness and minimality. A brief overview of necessary requirements for the implementation of the case-based properties has also been provided in their study. A comprehensive experimentation was performed to compare variants of DCT DSIT approach, landmarking and their combinations. MFs were constructed for the experiments from the UCI datasets (see Table 2.1) which contained up to 25% missing values. Error rates for ten different classification algorithms from the METAL project were determined for different subsets of data characteristics mentioned in Table 2.3 and restricted to three Base-learners that are shown in Figure 2.3. The empirical results show the proposed approach in combination with DSIT, and landmarking approaches as a promising one though not significantly different from previous meta-learning studies.

Abdelmessih et al. (2010) presented an overview of a landmarking operator and its evaluation. This landmarking operator was developed as part of an open-source RapidMiner data-mining tool. As mentioned repeatedly in the above studies, landmarkers selection is a critical process and the two basic criteria to select a landmarker were suggested in this study to be: 1) a landmarker has to be simple and require minimum execution (processing) time; and 2) it has to be simpler than the target learner(s). Following these conditions, RapidMiner provided the landmarkers shown in Figure 2.3 and the target algorithms, for which the accuracy was predicted (see Table 2.3). For the evaluation of these landmarkers, 90 datasets from the UCI (Bache and Lichman, 2013) and other sources were collected with at least 100 samples in each. By following the existing studies, the landmarking operator has been compared with the DSIT MFs of StatLog (King et al., 1995) and DCT (Lindner and Studer, 1999), where landmarking resulted in *5.1-8.3%* overall performance improvement in all cases.

Table 2.3: Target Learners used in various studies

Target Learners	**Bensusan and Giraud-Carrier (2000)** **Pfahringer et al. (2000)** **Soares et al. (2001)** **Kopf and Iglezakis (2002)** **Giraud-Carrier (2005)**	**Abdelmessih et al. (2010)**
C5.0 tree	✔	✔
C5.0 Rule Induction (C5.0 rules)	✔	
C5.0 boost	✔	
Naive Bayes classifier (NB)	✔	✔
Instance-based Learning (IBL)	✔	
Multi-layer Perceptron (MLP)	✔	✔
Radial-basis Function (RBF)	✔	
LDA	✔	
Ripper	✔	
Linear Discriminant Trees (Ltree)	✔	
k-Nearest Neighbour (k-NN)		✔
Random Forests (RF)		✔
One Rule Learner (OneR)		✔
Support Vector Machines (SVM)		✔
Total Target Learners	10	7

2.2.3 Model-based Approach

Model-based MF generation is another effort towards task characterization in MLL domain. In this approach the dataset is represented in a data structure that can incorporate the complexity and performance of the induced hypothesis. Later the representation can serve as a basis to explain the reasons behind the performance of the learning algorithm Giraud-Carrier (2008). Several research works utilizing the Model-based approach are discussed below.

Bensusan et al. (2000) study was an initial effort towards model-based approach. The authors proposed to capture the information directly from the induced decision trees for characterizing the learning complexity. Figure 2.3 lists the 10 descriptors computed from induced decision trees. Using these MFs, a task representation and algorithm to store and compare two different tree structures has been explained in detail with examples. The authors also elaborated on the motivation of using the induced decision trees directly rather than the predefined properties used in decision tree based MFs that made explicit properties implicit in the tree structure. Finally, higher-order MLL approach was generalized by proposing data structures to characterize other algorithms. A tree like structure was used for Decision Trees (DT) in this work, sets were proposed for *rule sets* and graphs for Neural Networks (NNs).

Peng et al. (2002) effort was towards improving the dataset characterization by capturing structural shape and size of the decision tree induced from the dataset. For that purpose 15 features were proposed, known as *DecT* and shown in Figure 2.3, which do not overlap with Bensusan et al. (2000). These measures were used to rank 10 learning algorithms in various experiments. In the first experiment DCT (Lindner and Studer, 1999) DSIT MFs and 5 landmarkers (Worst Nodes Learner, Average Nodes Learner, NB, and LDA) were compared with DecT. The results proved the performance enhancement of the proposed approach. In another experiment DecT measures were compared with the same DCT measures and landmarkers to rank the learning algorithms based on the accuracy and time where again DecT performed better. The last experiment was performed to select MFs by reducing the number of features to 25, 15 and 8 respectively. The k-Nearest Neighbour algorithm, with various values of *k* between 1 to 40, was used to select k datasets for ranking the performance of learning algorithms. The results suggested that the proposed feature selection did not significantly influence the performance of either DecT or even DCT. Overall, DecT outperformed the other approaches.

Neuro-cognitive inspired mechanism was proposed by Duch et al. (2011) to analyse learning based transformations that generate useful hidden features for MLL. The types of transformations include restricted random projections, optimization using projection pursuit methods, similarity and general kernel-based features, conditionally defined features, and features derived from partial successes of various learning algorithms. The binary features were extracted from DT and rule-based algorithms, continuous features were discovered by projection pursuit, linear SVM and simple projections. NB was used to calculate posterior probabilities along these lines while k-NN and kernel methods were used to find similarity-based features. The proposed approach also evaluated and illustrated Multi-dimensional Scaling (MDS) mappings and Principal Component Analysis (PCA), Independent Component Analysis (ICA), Quality of Projected Clusters (QPC), SVM projections in the original, one-, and two-dimensional space. Various real-world and synthetic datasets (details can be found in Table 2.1) were used for visualization and to analyse the kind of structures they create. The classification accuracies for each dataset were predicted using five classifiers including NB, k-NN, Separability Split Value Tree (SSV), Linear and Gaussian kernel SVM in the original, one- and two-dimensional spaces. The results showed an overall significant improvement almost in all five algorithms as compared to the existing approach also proposed by the authors.

2.2.4 Discussion and Summary

There are three common MF generation approaches proposed in the reviewed publications for MLL: 1) DSIT, 2) Landmarking, and 3) Model-based. The DSIT MFs approach was introduced at the early stage of MLL development where Rendell et al. (1987) proposed two descriptive features for VBMS. Later on Rendell and Cho (1990) added more descriptive features to the original list. The statistical MFs were introduced by King et al. (1995), and Sohn (1999) proposed information-theoretic features combined with some existing descriptives to represent a problem at a Meta-level. Finally, Lindner and Studer (1999) proposed an extensive list of DSIT MFs, known as DCT. The DCT measures became a benchmarked approach to represent a problem using the DSIT approach. These measures were later used in several studies for experimentation, e.g. Berrer et al. (2000), Giraud-Carrier (2005), etc., and compared with other MF approaches.

Landmarking and Model-based approaches are more recent ones and have been outperforming the DSIT in almost all the comparative studies. The earliest study on landmarking was conducted by Bensusan and Giraud-Carrier (2000) where the approach was claimed to be simpler, more intuitive and efficient than DSIT. The proposed approach was compared with and outperformed information-theoretic measures of DCT with a significant difference. Though one common deficiency that is observed in several MLL studies is the use of smaller number of Examples of Datasets (EoD) for experimentations which raised a question on the significance of the reported results. Pfahringer et al. (2000) used a different set of landmarkers but the same target learners as Bensusan and Giraud-Carrier (2000). This work can be considered to offer improvements to the previous one in two aspects: 1) huge number of synthetic datasets were used; and 2) some descriptive MFs were combined with the landmarkers. This approach was also compared with DCT features where landmarking showed significant improvement in the results. Similarly Soares et al. (2001), Kopf and Iglezakis (2002) and Abdelmessih et al. (2010) used different sets of target learners, landmarkers, number of dataset examples, and compared their approaches with a different set of DSIT measures. All of them reported improved results of the landmarking approach over the DSIT.

[7] Tabular representation of the visualization can be seen in Appendix A)

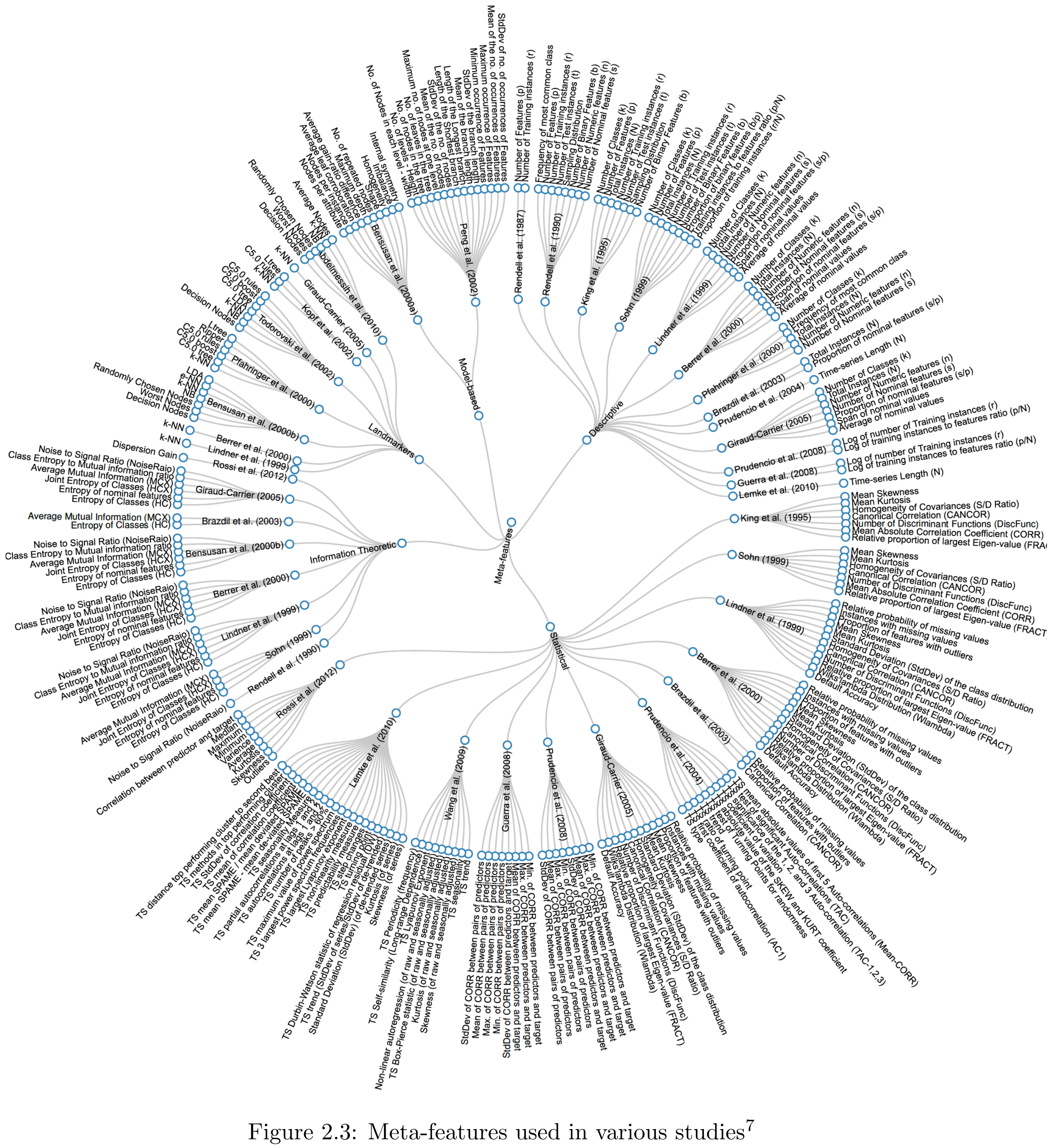

Figure 2.3: Meta-features used in various studies[7]

Bensusan et al. (2000) approach to characterizing the learning complexity by directly inducing MFs from the model is the earliest work towards model-based approach. In this work 10 descriptors (MFs) were computed from the induced decision trees which can be seen in Figure 2.3. Peng et al. (2002) effort was towards improving this characterization by focusing on structural shape and size of the decision tree induced from the datasets. The other dimension of this work was to compare the proposed model-based approach with DCT, DSIT and landmarking measures. Various experimentations were performed with variations of MFs and landmarkers where the model-based approach consistently performed better. A problem with these Meta-level problem representations is that they can not easily accommodate non-stationary environments. Most of the effort has been dedicated to the stationary environment, even though there are some recent studies addressing MFs for a dynamically changing environment, i.e. Rossi et al. (2014), but these are not mature enough to represent the entire domain. Although Rossi et al. (2014) used traditional MF that are used to characterize stationary data, only those MFs were computed that characterize individual variables. Moreover, there are separate features computed for training and selection windows. Their reliability is highly dependant on the number and quality of examples, thus the larger the number of examples in a window, the potentially higher the reliability of the problem representation at the Meta-level. However, in a rapidly changing environment there is often a very limited number of examples between consecutive concept changes. Hence there is an unaddressed need for novel MFs and approaches which can cope with small data samples.

From the above studies it can be observed that combining significant MFs from different feature generation approaches might be useful as shown in Figure 2.4.

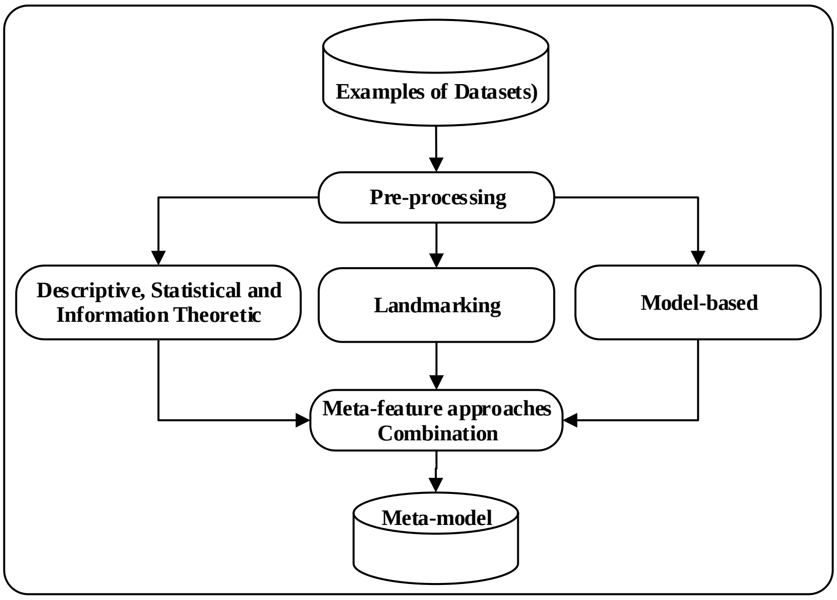
Examples of Datasets)
Pre-processing
Descriptive, Statistical and Information Theoretic
Landmarking
Model-based
Meta-feature approaches Combination
Meta-model

Figure 2.4: Combining Significant Meta-features from various approaches

2.3 Base-level Learning

In the context of Meta-level Learning (MLL), Base-level Learning (BLL) algorithms are used to build predictive models on input datasets and for MLL purposes are used to compute a set of performance measures, i.e, accuracy, execution-time, etc. These performance measures are combined together with their respective Meta-features (MFs) in the Meta-knowledge (MK) database. A Meta-learner uses these performances as a target variable. The remaining sections discuss several studies concerned with the roles and characteristics of individual and combined BLL algorithms utilised within the MLL context.

Brazdil et al. (2003) proposed an MLL based approach to rank candidate algorithms where k-Nearest Neighbour (k-NN) was used to identify the datasets that were most similar to the query dataset. The pool of candidate algorithms contained an ensemble method, namely C5.0 Adaptive Boosting (C5.0 boost), which performed well for 19 out of 53 datasets in the presence of 9 other algorithms. The performance of ensemble methods were ranked with individual learning algorithms. In general, several researches used C5.0 boost ensemble method with other individual algorithms and found it to be the top performing method.

The applicability of MLL on a Time-series (TS) task was demonstrated by Lemke and Gabrys (2010a). Several individual and combination of forecasting algorithms were used to investigate which model works best in which situation. In the experiments 5 forecasting combination methods were used including 1) *simple average* where all available forecasts are averaged, 2) *simple average with trimming* which do not take the worst performing 20% models into account, 3) *variance-based method* where weights for a linear combination of forecasts are determined using past forecasting performance, 4) *out-performance method* which determines weights based on the number of times a method performed best in the past, and 5) *variance-based pooling* which first groups past forecast performance into 2-3 clusters and then takes their average to obtain the final forecast. The results of these experiments showed that the forecast combination methods perform better than individual models which are listed in Table 2.4. Further discussion of this work can be found in Chapter 2.4.4.

Menahem et al. (2011) proposed a new MLL based ensemble scheme for one-class problems know as TUPSO. The TUPSO combined one-class Base-classifiers via a Meta-classifier to produce a single prediction. The BLL component generates predictions of classifiers which are used to extract aggregated MFs as well as one-class accuracy and f-score estimates. The one-class performance evaluator computed each Base-classifier on only positively labelled instances using 4 algorithms including: 1) global density estimation, 2) peer group analysis, 3) Support Vector Machines (SVM), and 4) attribute distribution function approximation (ADIFA) on 53 distinct datasets (details can be seen in Table 2.1). There are 15 aggregated MFs computed from the predictions of Base-classifiers that are clustered into four groups: 1) summation-based (votes, predictions, weighted predictions, power and log of weighted predictions), 2) variance-based (votes, predictions, and weighted), 3) histogram-based, and 4) representation-length. In an empirical evaluation an ensemble method, Fixed-rule, produced worse classification accuracy when compared to MLL based ensembles - TUPSO.

Table 2.4: Base-level learning strategies used in different studies

Research Work	**Sampling Strategy**	**Base-learners**	**Performance Measure**
King et al. (1995)	9-fold Cross-Validation (CV) for datasets with less than 2500 instances	k-NN, Radial-basis Function (RBF), Density Estimation, Classification and Regression Trees (CART), Inductive CART (IND-CART), Back-propagation, NewID, C4.5 Decision Tree algorithm (C4.5), CN2 Induction Algorithm (CN2), Quadratic Classifier (Quadra), Cal5, AC^2, Smooth Multiple Additive Regression Technique (SMART), Logistic Regression, Fisher's Linear Discriminant (FLD), ITrule, Causal Structure for Inductive Learning (CASTLE), Naive Bayes classifier (NB)	Misclassification error, Run-time speed
Bensusan and Giraud-Carrier (2000)	stratified 10-fold CV	NB, Multi-layer Perceptron (MLP), RBF, C5.0 Decision Tree (C5.0 tree), C5.0 Rule Induction (C5.0 rules), C5.0 boost, Instance-based Learning (IBL), Linear Discriminant Analysis (LDA), Rule Learner (Ripper), Linear Discriminant Trees (Ltree)	
Pfahringer et al. (2000)	10-fold CV	NB, MLP, RBF, C5.0 tree, C5.0 rules, C5.0 boost, IBL, LDA, Ripper, Ltree	Mean Absolute Error (MAE)
Soares et al. (2001)		NB, MLP, RBF, C5.0 tree, C5.0 rules, C5.0 boost, IBL, LDA, Ripper, Ltree	
Peng et al. (2002)	10-fold CV	C5.0 tree, C5.0 rules, C5.0 boost, Ltree, LDA, NB, IBL, MLP, RBF, Ripper	Mean Squared Error (MSE), Run-time speed
Todorovski et al. (2002)	10-fold CV	C5.0 tree, C5.0 rules, C5.0 boost, Ltree, Ripper, NB, k-NN [8], LDA	MSE and Spearman's Rank Correlation Coefficient (SRCC)
Kopf and Iglezakis (2002)	10-fold CV	NB, MLP, RBF, C5.0 tree, C5.0 rules, C5.0 boost, IBL, LDA, Ripper, Ltree	
Brazdil et al. (2003)	10-fold CV	C5.0 tree, C5.0 rules, C5.0 boost, Ltree, IBL, Ripper, LDA, NB, MLP, RBF	Adjusted Ratio of Ratios (ARR)
Prudencio and Ludermir (2004)	I: Train and test and II: train, test and validate	I: J.48 and II: MLP	MAE
Giraud-Carrier (2005)	10-fold CV	NB, MLP, RBF, C5.0 tree, C5.0 rules, C5.0 boost, IBL, LDA, Ripper, Ltree	
Guerra et al. (2008)	10-fold CV	MLP[8]	Normalized MSE
Wang et al. (2009)	80% Training and 20% testing partition	Exponential Smoothing (ES), Auto-regressive Integrated Moving Average (ARIMA), Random Walk (RW), Neural Network (NN)	
Kadlec and Gabrys (2009a)	Leave-one-out CV	Multiple Linear Regression (MLR), MLP, RBF, Lazy-learning	MSE and SRCC

[8] k=1

[8] hidden nodes = 1, 2, 3, 8, 16, 32

Lemke and Gabrys (2010a)	10-fold CV	ARIMA, Structural model, Iterated (single exponential smoothing, Taylor smoothing, theta, NN, elman NN), Direct (regression, theta Moving Average (MA), single exponential smoothing, Taylor smoothing, NN)	Symmetric Mean Absolute Percentage Error (SMAPE)
Abdelmessih et al. (2010)	10-fold CV	NB, k-NN, MLP, C5.0 tree, Random Forests (RF), One Rule Learner (OneR), SVM	Root Mean Squared Error (RMSE)
Rossi et al. (2012)	Training and testing	RF, SVM, CART, Projection Pursuit Regression (PPR)	Normalized MSE
Rossi et al. (2014)	Training and testing	RF, SVM, CART, PPR, Multivariate Adaptive Regression Splines (MARS)	Normalized MSE

2.3.1 Discussion and Summary

The MK database usually consists of MFs and performance measures (target) of different learning algorithms which are predicted accuracies for Examples of Datasets (EoD). These predictive values are computed, in the context of MLL, through BLL. The predictive accuracies of learning algorithms are used a basis for identifying the best algorithm from the pool of methods, their ranking, and/or combination. Another level of complexity is introduced by the different parametrizations of the algorithms which were overlooked by several studies where only default configurations were considered. Furthermore, most of them selected only the best algorithm from the pool to minimize the representation complexity of MK dataset, therefore very few of them stored information about the ranking and relative performance of evalutaed BLLs. Table 2.4 shows different learning strategies, Base-learners and performance measures that various MLL studies used at the Base-level. It can be observed that the 10-fold cross validation strategy, MAE accuracy measure and few learning algorithms have become a norm to use at the Base-level. The same Base-level learning strategies are used in some MLL studies for TS with different ARIMA and exponential smoothing algorithms. Another common deficiency that can be observed from various studies is related to the granularity of information that is being stored in MK database.

Table 2.5 summarises and groups the reviewed studies according to the four dominant performance measures which were used as the target variable for an MLL system.

Table 2.5: Different Performance Measures that are used in MLL studies

Performance Measure(s)	**Description**	**Research Work**
Best learning algorithm	The performance measure only contains of the classification accuracy of best learning algorithm for each single dataset	Utgoff (1984); Graner et al. (1994); King et al. (1995); Bensusan et al. (2000)
Ranking of learning algorithms	To predict a ranked list of learning algorithms in a pool which are sorted based on a performance measure, e.g. classification accuracy, run-time, etc.	King et al. (1995); Brazdil et al. (2003); Vilalta et al. (2004)

Quantitative Prediction Reif (2012)	To directly predict the performance of the target learning algorithm in an appropriate unit, i.e., by training separate regression model for each target algorithm	Gama and Brazdil (1995); Sohn (1999); Kopf and Iglezakis (2002); Bensusan and Kalousis (2001); Reif (2012)
Predicting Parameters	The MLL target variable could be one parameter value or a set of values	Soares et al. (2004); Soares and Brazdil (2006); Kadlec and Gabrys (2009a); Lemke and Gabrys (2010a)

2.4 Meta-learning

The Meta-knowledge (MK) induced for the Meta-level Learning (MLL) purposes provides a means for making informed decisions in relation to which algorithms are likely to perform best/well for a given problem (Giraud-Carrier, 2008). This section presents the history of the most promising decision-support systems for algorithm selection, followed by a review of the applicability of MLL to the supervised and unsupervised learning algorithms.

2.4.1 Existing Systems

Based on the reviewed literature, Utgoff (1984) can be considered as the earliest effort towards developing MLL systems where a system named Shift To A Better Bias (STABB) was proposed. It was a demonstration that a learner's bias could be adjusted dynamically. Later this work became an initial point of reference and was enhanced in several studies. One of them was Variable-bias Management System (VBMS) by Rendell et al. (1987), where a relatively simple MLL system was proposed. VBMS selected the best among three symbolic learning algorithms as a function of only two dataset characteristics, namely, the number of training instances and the number of features. As mentioned in one of the previous sections, this was then further improved in Rendell and Cho (1990).

Machine Learning Toolbox (MLT) project by Graner et al. (1994) was one of the initial attempts to address the applications of MLL. MLT produced a toolbox consisting of 10 symbolic learning algorithms for classification. The part of MLT project that provides assistance with the algorithm selection is known as a Consultant. The Consultant was based on a stand-alone expert system which maintained a knowledge-base that considered the experiences acquired from the evaluation of learning algorithms. Considerable insight into many important Machine Learning (ML) issues was gained which had been translated into rules that formed the basis of Consultant-2. Consultant-2 was also an expert system for algorithm selection which gathered user inputs through a set of questions about the data, the domain and user preferences. Based on the user response relevant rules led to either additional questions or, eventually, a classification algorithm recommendation. Although its knowledge base had been built through an expert-driven knowledge engineering rather than via MLL it still stands out as the first automatic tool that systematically related application domain and dataset characteristics to the most suitable classification algorithms. Additionally Consultant-3 provided advice and help on the combination of learning algorithms. It is also able to perform self-experimentation to determine the effectiveness of an algorithm on a learning problem.

In Statistical and Logical learning (StatLog) project King et al. (1995) presented the results of comprehensive experiments on classification algorithms. The project was an extension of VBMS by considering a larger number of Meta-features (MFs), together with a broad class of candidate models for algorithms selection. Its aim was to compare several symbolic learning algorithms on twelve large real-world classification tasks. Some MLL algorithms were used for model selection task where statistical measures, e.g., skewness, kurtosis and covariance, that produced higher accuracy were reported. Additionally, a thorough empirical analysis of 16 classifiers on 12 large real-world datasets and learning models using accuracy and execution time measures of performance were produced. There is no single algorithm that performed best in the experimentation phase. Symbolic algorithms resulted in the best performance for datasets with extreme distributions, i.e., where a distribution was far from normal (i.e., specifically with skew > 1 and kurtosis > 7), and the worst in the scenarios where the data was evenly distributed. In contrast, the Nearest Neighbour algorithm was found to be accurate for datasets containing evenly distributed in terms of scale and importance of the features.

The Meta-Learning Assistant (METAL) project was developed to facilitate selection of the best suited classification algorithm for a data-mining task (Berrer et al., 2000). It guides the user in two ways: 1) in discovering new and relevant MFs; and 2) in a selection or ranking of classifiers using an MLL process. The main deliverable of this project was the Data Mining Advisor (DMA), a Web-based MLL system for the automatic selection of classification learning algorithms (Giraud-Carrier, 2005). The DMA returned a list of ten algorithms that were ranked according to how well they met the stated goals in terms of accuracy and training time. It implemented ranking mechanisms by exploiting the ratio of accuracy and training time. The choice of an algorithm ranking, rather than selecting the best-in-class, was motivated by a desire to give as much information as possible and as a consequence a number of algorithms could be subsequently executed on the dataset.

The Meta-learning Architecture (METALA), developed by Botia et al. (2001), was an agent-based architecture for distributed Data Mining, supported by MLL. The system supported an arbitrary number of algorithms and tasks, and automatically selected an algorithm that appeared best from the pool of available algorithms. Like in the case of DMA, each task was characterized by Descriptive, Statistical and Information-Theoretic (DSIT) features relevant to its usage, including the type of input data it required, the type of model it induced, and how well it handled noise. It had been designed to automatically carry out experiments with each learner and task, and induce a Meta-model for an algorithm selection. As new tasks and learning algorithms were added to the system, corresponding experiments were performed and the Meta-model was updated.

The Intelligent Discovery Assistant (IDA) provided a Knowledge Discovery (KD) ontology that defined the existing techniques and their properties (Bernstein and Provost, 2001). It supported three algorithmic steps of the KD process, including preprocessing, data modelling and post-processing. The approach used in this system was the systematic enumeration of valid data-mining processes so that potentially fruitful options were not overlooked, and effective ranking of these valid processes based on user-defined preferences e.g., prediction accuracy, execution speed, etc. IDA systematically searched for an operation whose pre-conditions have been met and whose indicators were consistent with the user-defined preferences. Similarly, its post-conditions searched for an operation and the process terminated once the goal had been reached. Once all valid KD processes had been generated, a heuristic ranker was applied to return user-specified goals. Bernstein et al. (2005) research had focused on extending the IDA approach by leveraging the interaction between ontologies to extract deep knowledge and case-based reasoning for MLL. The system also used procedural information in the form of rules fired by an expert system. The case-base was built around 53 features to describe cases and the ontology came from human experts.

Mierswa et al. (2006) developed a landmarking operator in RapidMiner as part of Pattern Recognition Engineering (PaREn) project, which was an open source system for data mining. This operator extracted landmarking features from a given dataset by applying seven fast computable classifiers on it (shown in Figure 2.3).

Table 2.6: Existing Meta-learning Systems

Research Work	**Name**	**Approach**	**Contributions**	**Limitations**
Utgoff (1984)	STABB	Statistical	Initial effort towards MLL	Limited to altering only one kind of learner's bias with fixed order of choices
Rendell et al. (1987)	VBMS	Descriptive	Biases are dynamically located and adjusted according to problem characteristics and prior experience	VBMS is a relatively simple MLL system that learns to select the best among three symbolic learning algorithms as a function of only two dataset characteristics
Rendell and Cho (1990)	Empirical Learning as a Function of Concept Character	DSIT	Complex MFs based on shape, size and concentration, and artificial data generation is used	These complex MFs are expensive to compute
Graner et al. (1994)	MLT	Rule-based	An expert system for algorithm selection by gathering user input through questions and trigger relevant rules while the knowledge-base was built through expert-driven knowledge engineering	Its knowledge base was built through expert-driven knowledge engineering rather than MLL
King et al. (1995)	StatLog	Statistical	A thorough empirical analysis of learning algorithms and models is produced by comparing several symbolic learning algorithms on twelve real-world classification tasks	For a given dataset, algorithms were characterized only as applicable or non-applicable, i.e., they do not provide a way to rank the algorithms; furthermore, that characterization was based on a simple comparison of accuracies regardless of any statistical significance test
Berrer et al. (2000) and Giraud-Carrier (2005)	METAL - DMA	DSIT and Landmarking	Discovers new and relevant MFs and algorithm ranking in terms of accuracy and execution time	The outcome of the prediction model is only the best classifier for the new dataset. It does not support multi-operator workflows

Botia et al. (2001)	METALA	Model-based	Agent-based architecture for distributed data-mining, automatically carry out experiments and induce a Meta-model for algorithm selection, it provides architectural mechanisms necessary to scale the DMA	DMA's MFs are used to represent a problem, no contribution to introduce new features
Bernstein and Provost (2001)	IDA	Model-based	Its goal is to rank pre-processing, modelling and post-processing steps that are both valid and consistent with the user-defined preferences	The data should be already pre-processed considerably by the user for IDA to model it and evaluating the resulting models
Bernstein et al. (2005)	IDA - An Ontology-based Approach	Model-based	Extending IDA approach by leveraging the interaction between ontology for deep knowledge and Case-Based Reasoning for MLL	The case-based is built on fixed 53 features and the system is still in the early stages of implementation
Mierswa et al. (2006)	PaREn	Landmarking	A Landmarking operator for MLL developed in RapidMiner	Very limited Examples of Datasets (EoD) (from UCI Machine Learning Repository (UCI)) are used to build MK
eLICO (2012)	e-Laboratory for Inter-disciplinary Collaborative Research (e-LICO)	Model-based	An e-Laboratory for inter-disciplinary collaborative research in data-mining and data-intensive science	Meta-learning component is using RapidMiner's landmarking system which is built on only 90 UCI datasets

e-LICO was a project for data-mining and data-intensive science (eLICO, 2012). This project comprised of three layers: 1) e-Science, 2) Application, and 3) Data-mining. The e-Science and data-mining layers formed a generic environment that was adapted to different scientific domains by customizing the application layer. The architecture of e-LICO project was shown in Figure 2.5.

The e-Science layer was built on an open-source e-science infrastructure that supported content creation through collaboration at multiple scales in dynamic virtual communities. The Taverna[9], open-source data-mining and predictive analysis solution (RapidAnalytics) and RapidMiner (Mierswa et al., 2006) components had been used to design and enact data-analysis workflows. The system also provided a variety of general-purpose and application-specific services and a broad tool-kit in designing and sharing such workflows with data-miners all over the word using *myExperiment* portal. The IDA (Bernstein and Provost, 2001) exposed MLL capabilities by automatically creating processes tailored for the specification of input data and a modelling task. The RapidMiner's DMA component helped to design processes by recommending operators that fitted well with the existing operators in a process. The data-mining layer provided comprehensive multimedia data-mining

[9] A suite of tools used to design and execute scientific workflows and experimentation. http://www.taverna.org.uk

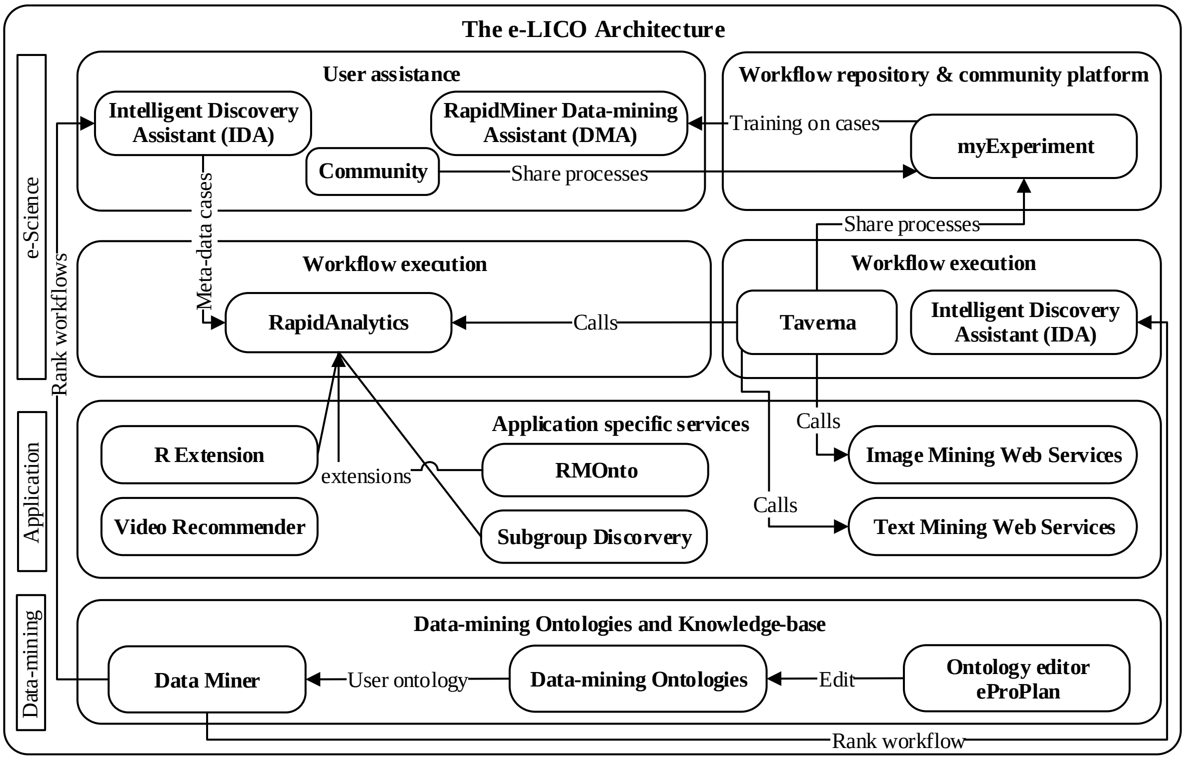

Figure 2.5: e-LICO project architecture

tools that were augmented with preprocessing and learning algorithms developed specifically to meet challenges of data-intensive, knowledge rich sciences. The knowledge-driven data-mining assistant relied on a data-mining ontology and knowledge-base to propose ranked workflows for a given task. The application layer initially came as an empty shell which had to be built by the domain user from different components of the system. At the application layer, e-LICO was showcased in two application domains: 1) a systems biology, and 2) a video recommendation task.

2.4.2 Regression and Classification Problems

This section covers and discusses different aspects of MLL that is used for regression and classification tasks in different systems.

Todorovski et al. (2002) addressed a novel approach of predictive clustering trees to rank classification algorithms using dataset properties. The approach was to illustrate ML algorithms ranking where the relative performance of the algorithms had to be predicted from a given dataset's MFs. For that purpose the performance of eight Base-level algorithms, mentioned in Table 2.4, has been measured on 65 classification tasks gathered from the UCI repository and the METAL project. Furthermore, DSIT dataset characteristics from StatLog and Dataset Characterization Tool (DCT) were combined to create an MK dataset consisting of 33 MFs. The properties of individual attributes were aggregated using average, minimum or maximum functions. The landmarking approach was used in this study with 7 simple and fast learners, shown in Figure 2.3, to investigate the ranking task performance. The proposed dataset characterization approach with clustering tree outperformed with a significant margin the DCT and the histogram approach which used a grained aggregation of DCT properties.

Vilalta and Drissi (2002a) presented four approaches to MLL consisting of learning from Base-learners; 1) Stacked generalization, 2) Boosting, 3) Landmarking, and 4) Meta-decision trees. The information collected from the performance of Base-level Learning (BLL) algorithms was incorporated into the MLL process. Stacked generalization was considered a form of MLL where each set of Base-learners were trained on a dataset and the original feature representation was then extended with the predictions of the Base-learners. These predictions were received by successive layers as inputs and the output was passed on to the next layer. A single (Meta-)learner at the topmost layer computed the final prediction. Boosting was another approach that was considered as a form of MLL. It generated a set of Base-learners by generating variants of the training set using sampling with replacement technique under a weighted distribution. This distribution is modified for every new variant by assigning more weights to the incorrectly classified examples using the most recent hypothesis. Boosting took the predictions of each hypothesis over the original training set to progressively improve the classification of those examples for which the last hypothesis failed.

In the last proposed approach, the Base-learners consisted of a combination of several inductive models induced from Meta-decision trees. A decision tree was built where each internal node represented a MF that predicted a class probability for a given example by a set of models whereas the leaf nodes corresponded to a predictive model. Given a new example, the Meta-decision tree selected the most suitable model to predict the target value. Todorovski and Dvzeroski, 2003 used the same approach for MLL discussed in this section.

An instance-based learning algorithm, k-Nearest Neighbour (k-NN), was used to identify the datasets that were most similar to the one at hand by Brazdil et al. (2003). The candidate Base-learning algorithms were not ranked but selected based on a multi-criteria aggregated measure that took accuracy and time into account. The proposed methodology had been evaluated using various experiments and analysis at the Base- and Meta-level learning. The Meta-data used in this study was obtained from METAL project which contained estimates of accuracy and time for 10 algorithms (listed in Table 2.4) on 53 datasets, using 10-fold Cross-Validation (CV). The k-NN algorithm was used at the Meta-level to select the best candidate algorithm for a new dataset. For two values of the number of neighbours, 1 and 5, the k-NN showed a significant improvement in the results, particularly with k=1, as compared to the trial-and-error approach.

Two MLL approaches were investigated to select models for Time-series (TS) forecasting by Prudencio and Ludermir (2004) in different case studies. In the first case study, a single BLL algorithm was used to select models to forecast stationary TS. The base-level and meta-level learning algorithms and configurations are given in Table 2.4 and Table 2.7 for both case studies while details of datasets and MFs are listed in Table 2.1 and Figure 2.3 respectively. In another case study a more recent and sophisticated approach - NOEMON (Kadlec and Gabrys, 2009a) was used to rank three models of the M3-Competition. In both case studies the experiments revealed significant results by taking into account the quality of algorithm selection and forecasting algorithm performance aspects of the selected models.

Active MLL method, in combination with *Uncertainty Sampling* and outlier detection, had been proposed by Prudencio and Ludermir (2008) to support the selection of informative and anomaly-free Meta-examples for MLL. Some experiments were performed in a case study where Multi-layer Perceptron (MLP) was used to predict the accuracies of 50 regression problems at the Base-level learning (the details can be seen in Table 2.1) and k-NN[10] at the Meta-level. The MFs used in the case study consisted of 10 simple and statistical measures which can be seen in Figure 2.3. The results of experiments revealed that the proposed approach was significantly better than the previous work on Active MLL. Also the *Uncertainty Sampling* method increased the performance when the outliers were eliminated from the MK which affected 5% of the data.

Guerra et al. (2008) used Support Vector Machines (SVM), with different kernel functions, as a Meta-regresor to predict the performance of a candidate algorithm, MLP, based on descriptive and statistical features of the learning tasks. For experimentation purposes the input datasets and MFs used in this study were the same as those in the Prudencio and Ludermir (2008) work. The MLP was used as a base-learner to compute the normalized Mean Squared Error (MSE) which was averaged over 10 training runs. Table 2.4 contains details of the learning strategy which were used at the base-level. At the meta-level, SVM with different kernel functions (listed in Table 2.7) were applied to predict the normalized MSE and Mean Absolute Correlation Coefficient (CORR) between the predicted and the actual target values of the MLP. Later the performance of the Meta-regressor (SVM) was compared with three different benchmarked regression algorithms which were used in the previous work including Linear Regression, k-NN[11] and M5 algorithm (Decision Trees (DT) Quinlan (1992)). The experiments revealed that the SVM with Radial-basis Function (RBF) kernel (particularly with γ=0.1) obtained better performance as a Meta-regressor when compared to the mentioned benchmark algorithms.

Kadlec and Gabrys (2009a) proposed a generic architecture for the development of on-line evolving predictive systems. The architecture defined an environment that links four classes techniques from the ML area: 1) ensemble methods, 2) local learning, 3) meta-level learning, and 4) adaptability and also the interaction between them. The Meta-level learning is discussed in this section whereas adaptability aspects of this paper are discussed in Sections 2.5.1 respectively.

The Meta-level Learning module of Kadlec and Gabrys (2009a) architecture was responsible for high-level learning, control and decision making. Meta-level was the most complex but least diverse top layer of the architecture. In this study a Meta-learner was defined as building a high-level global knowledge of the models which were incrementally grown by applying the eveloving architecture to various tasks. The main goal of Meta-level layer was to optimise the predictions in terms of the global performance function which was achieved by 1) controlling the population at lower levels to cover unexplored parts of the input space, 2) looking for relations between algorithm configurations of the paths and the achieved performance, and 3) adapting the combinations in order to reflect the current state of the data. In general this layer was used to learn the dependency between the pool of learning algorithms and the performance at various levels. Several experiments

[10] k = 1, 3, 5, 7, 9 and 11 nearest neighbours

[11] k=1

had been performed using three real-world datasets from the process industry where adaptive and static techniques were compared. The automated data pre-processing and model selection took a lot of the model development effort away from the user.

An empirical study on rule induction based forecasting method selection for univariate TS was conducted by Wang et al. (2009). The study aimed to identify characteristics of a univariate TS and evaluated the performance of four popular forecasting methods (listed in Table 2.4) using a large collection of datasets listed in Table 2.1. These two components are integrated in an MLL framework which automatically discovers the relations between forecasting methods and data characteristics (shown in Figure 2.3). Furthermore, C4.5 decision tree learning technique was used to generate quantitative rules of MFs and categorical rules were constructed using an unsupervised clustering approach.

Lemke and Gabrys (2010a) investigated applicability of MLL for TS prediction and identified an extensive set of MFs that were used to describe the nature of TS. The feature pool consisted of general statistical, frequency spectrum, autocorrelation, and behaviour of forecasting methods (diversity) measures (see Figure 2.4). These measures were extracted for two sets of datasets from popular TS competitions, see Table 2.1 for details, and the target was to predict the next 18 observations for NN3[12] and 56 for NN5[12]. Using these datasets empirical experiments had been performed that had provided the basis for further MLL analysis. Extensive list of simple (seasonal), complex (Auto-regressive Integrated Moving Average (ARIMA)), structural and computational intelligence (Feed-forward Neural Network (NN)), and forecast combination methods were used for experimentation which can be seen in Table 2.4. From the pool of individual algorithms NN and Moving Average (MA) performed quite well for NN3 series while for NN5 the Symmetric Mean Absolute Percentage Error (SMAPE) in general was quite high where a combination method *variance-based pooling* out-performed all the individual and combination algorithms. At the end three experiments were performed to explore MFs using decision trees, comparing various MLL approaches (details are given in Table 2.7), and simulating NN5 on *zoomed ranking* method and on its combination. The conclusion of this study was that the ranking-based combination of forecasting methods clearly outperformed the individual methods in all experiments.

2.4.3 Clustering

This section discusses the use of MLL in the context of unsupervised learning.

De-Souto et al. (2008) presented a novel framework that applied an MLL approach to clustering algorithms, which was one of the initial efforts towards unsupervised algorithms. The proposed architecture was very similar to the MLL approach used to rank regression and classification algorithms. It extracted features of input examples from available datasets and associated them to the performance of the candidate algorithms in clustering that data to construct MK database. The MK database was used as an input dataset for the Meta-level learning and generated a Meta-model which was used in the selection or ranking of the candidate algorithms at a test mode.

[12] Neural Network forecasting competition, http://www.neural-forecasting-competition.com

Some implementation issues were also addressed which included: 1) the selection of datasets; 2) the selection of candidate clustering algorithms; and 3) the selection of the set of MFs that can better represent the problem at the Meta-level. In order to evaluate the framework, a case study using cancer gene expression microarray datasets was conducted. Seven candidate algorithms, listed in Table 2.7, and eight descriptive and statistical MFs were extracted, namely, *log10* of the number of examples and a ratio of the totalnumber of examples divided by the total number of features, a multi-variant normality, a percentage of outliers, a percentage of missing values, the skewness of Hotelling T^2-test, a Chip - type of microarray, and a percentage of features that were kept after applying the selection filter. Also, a regression SVM algorithm was used as the Meta-learner. The results were compared with the default ranking, where the average performance was suggested for all datasets. The mean and standard deviation of the Spearman's Rank Correlation Coefficient (SRCC) correlation for both rankings generated by the proposed approach was found to be significantly higher than the default one.

Soares et al. (2009) employed the De-Souto et al. (2008) framework in the ranking task of candidate clustering algorithms in a range of artificial clustering problems with two different sets of MFs. The first set had five MFs that were calculated using univariate statistics: quartiles, skewness and kurtosis, in order to summarize the multivariate nature of the datasets. This set included Coefficient of Variation (CoV), CoV of second and third quartiles, CoV of skewness and kurtosis while the other set had the same first four MFs as presented in De-Souto et al. (2008). In this paper three new candidate clustering algorithms were applied on each learning task that are listed in Table 2.7 and two Meta-learners were used, i.e., Support Vector Regression (SVR) and MLP. The methodology was evaluated using 160 artificially generated datasets, whose details are discussed in Section 2.1.4. Both Meta-learners were applied to the two sets of MFs separately and then compared with the default ranking method. The rankings predicted by the SVR and MLP methods were found to be significantly higher correlated than the default ranking. However, there was no significant difference between the correlation values of MLP and SVR methods for both Meta-datasets. Finally the authors had also highlighted the selection of MFs in the context of unsupervised MLL as an important issue that could be subjected to further analysis.

2.4.4 Discussion and Summary

There have been several MLL systems developed since the beginning of this area. Almost all the systems are developed for algorithm recommendations for the classification and regression tasks. Three main MF generation approaches were used in these systems which are listed in Table 2.6, where DSIT approach is found to be the most widely used. A landmarking based algorithm recommendation system is available as part of the RapidMiner, a commonly used open-source data-mining software. It was part of PaREn project and the landmarking functionality is available as an operator in the software. One of the most recent and large-scale projects related to MLL was e-LICO, the purpose of which was to solve data-mining and data-intensive problems. This project used MLL for algorithm recommendation by leveraging the existing systems, i.e., IDA and

RapidMiner's DMA component proposed by (Bernstein and Provost, 2001). Limitations of those systems are discussed in Table 2.6.

Apart from the existing software systems and tools there have been several studies where MLL was used specifically for regresion, forecasting, classification or clustering tasks. Several MF based problem representations have been proposed for the regression and classification tasks. Most of the comparisons in those studies focused on different MF approaches, selection of candidate algorithms and different sets of Meta-Learners. The problem representation using MFs has received the most attention, with landmarking and model-based approaches frequently compared with DCT DSIT features, and outperforming the DSIT approach in all reported studies with a significant difference. Not much effort has been dedicated to the model-based approach in the last few years as the landmarking with additional DSIT features have been considered as an overall better approach. The landmarking has also been proposed to solve problems other than algorithm recommendations, e.g., Kadlec and Gabrys (2009a) used landmarking approach for a recurrent concept extraction. Various studies investigated the applicability of MLL for TS problems including Prudencio and Ludermir (2004), Wang et al. (2009), and Lemke and Gabrys (2010a). Prudencio and Ludermir (2004) proposed descriptive and statistical features to represent a TS task to rank various seasonal and ARIMA models. Later on Lemke and Gabrys (2010a) used an extensive list of MF covering statistical, frequency spectrum, autocorrelation, and diversity measures for a TS prediction task. The pool of TS algorithms contained seasonal, ARIMA, structure and computational intelligence, and forecasting combination methods. The features used in this study to represent TS task at the Meta-level were better as compared to the previous studies.

There have been few studies which applied the MLL to clustering algorithms. De-Souto et al. (2008) effort was the initial step in investigating the knowledge representation for unsupervised problems. Landmarking was used to rank several unsupervised candidate algorithms, as listed in Table 2.7, combined with eight descriptive and statistical MFs which were used to represent unsupervised problems at the Meta-level. Most of them were the same as used in several regression and classification problem representations. Soares et al. (2009) employed De-Souto et al. (2008) framework by enhancing the list of landmarkers and proposed two different MF representations of an unsupervised task. One of the MFs list consisted of features proposed by De-Souto et al. (2008). The results showed an improvement of the proposed approach over the default base-line, but no significant difference was observed between the two different representations of the unsupervised problems. Finally, the authors had also highlighted the selection of MFs in the context of unsupervised MLL as an important issue that could be subjected to further analysis. All the existing MLL studies discussed in this section have only considered and were applied within stationary environments. Additionally these systems have the same issue which were discussed in the previous sections that the MK dataset did not have sufficient number of Meta-examples (MEs).

Table 2.7: Meta-level learning strategy used in various studies

Research Work	**Learning Strategy**	**Meta-learners**	**Performance**
Sohn (1999)	DSIT approach	Disc, QDisc, LoGID, k-NN, Backpropagation, Learning Vector Quantization (LVQ), Kohonen, RBF, Inductive CART (INDCART), C4.5 Decision Tree algorithm (C4.5), Bayesian Trees	Disc algorithm ranked as top performing algorithm
Lindner and Studer (1999)	Numeric, Symbolic and Mixed features characterization	Naive Bayes classifier (NB), MLP, RBF, CN2 Induction Algorithm (CN2), Iterative Dichotomiser 3 (ID3), MC4, T2, Winnow, Oblique Classifier-1 (OC1), One Rule Learner (OneR), Rule Learner (Ripper), Instance-based Learning (IBL)[13], C5.0 Decision Tree (C5.0 tree), Naive Bayes/Decision-Tree (NBT), Lazy Decision Trees (LazyDT), Parallel Exemplar-Based Learning System (PEBLS)	Numeric and mixed features characterization performed better
Bensusan and Giraud-Carrier (2000)	Landmarking approach compared with Information-Theoretic characterization	NB, k-NN[14], Elite-Nearest Neighbour (e-NN), Decision Nodes Learner (Decision Nodes), Worst Nodes Learner, Randomly Chosen Nodes Learner (Randomly Chosen Nodes), Linear Discriminant Analysis (LDA)	Landmarking (C5.0 Rule Induction (C5.0 rules)) approach outperformed Information-Theoretic
Pfahringer et al. (2000)	Landmarking approach compared with DSIT characterization	C5.0 tree, Ripper, Linear Discriminant Trees (Ltree)	Landmarking (C5.0 Adaptive Boosting (C5.0 boost)) performed better than others
Peng et al. (2002)	Model-based approach compared with Landmarking and DSIT characterization	k-NN	Model-based approach outperformed the remaining two
Prudencio and Ludermir (2004)	Descriptive and Statistical approach	I: Simple Exponential Smoothing (ES) and Time-delay NN and II: Random Walk (RW), Holt's linear ES (HL), Auto-regressive (AR), NOEMON	I: Simple ES and II: NOEMON performed better
De-Souto et al. (2008)	Landmarking approach to rank unsupervised learning algorithms	Single Linkage (SL), Complete Linkage (CL), Average Linkage (AL), k-Means (k-M), Mixture Models (M), Spectral Clustering (SP), Shared Nearest Neighbours (SNN)	The proposed approach outperformed the default ranking
Guerra et al. (2008)	Descriptive and Statistical approach	SVM with linear, quadratic, and RBF (γ=0.1, 0.05, 0.01) functions	Normalized MSE and CORR between predicted and target values

[13] 0-4

[14] k=1

Soares et al. (2009)	Landmarking approach to rank unsupervised learning algorithms	SL, CL, AL, k-M, M, SNN, Farthest First (FF), DB-Scan (DBS), X-Means (XM)	The proposed approach outperformed the default ranking
Wang et al. (2009)	Statistical approach on TS	ES, ARIMA, RW, NN	
Lemke and Gabrys (2010a)	Statistical approach on TS	NN, DT, SVM, Zoomed ranking (best method and combination)	The proposed approach showed superiority over simple model selection approaches
Abdelmessih et al. (2010)	Landmarking approach compared with Descriptive, DSIT characterization	NB, k-NN, MLP, OneR, Random Forests (RF)	Landmarking approach (k-NN) outperformed others
Rossi et al. (2012)	DSIT	RF	MetaStream outperformed default and ensemble approaches
Rossi et al. (2014)	DSIT	RF, NB, k-NN	MetaStream outperformed default and ensemble approaches

2.5 Adaptive Mechanisms

The Machine Learning (ML) and heuristic search algorithms require tuning of their parameters for a good performance. It can be achieved through off-line sensitivity analysis by testing different parameters to determine their best value in a stationary environment (Sikora, 2008). However, the optimal set of values for the parameters keep changing over time in non-stationary environments because of the change in the underlying data distribution where off-line sensitivity analysis becomes ineffective. In a dynamically changing environments domain Meta-level Learning (MLL) mechanism is considered to be one of the most effective techniques to learn the optimal set of parameters (Sikora, 2008). The rest of this section discusses various techniques of acquiring and exploiting Meta-knowledge (MK) in non-stationary environments, that have been proposed in the context of the existing predictive systems.

One of the earliest efforts employing an MLL based approach to achieve adaptivity in a non-stationary environment was presented by Widmer (1997). MLL was applied in time-varying environments for the purpose of selecting the most appropriate learning algorithm. For a traditional two-level learning model different types of attributes were defined at the Base- and Meta-level. The predictive attributes were used to induce models at the Base-level on raw examples from datasets if there existed a significant correlation between the predictors and the observed class distribution. On the other hand contextual attributes were employed to identify the current concept associated with the data and systematic changes in their values which indicated a concept drift. These attributes were identified using an MLL approach which was proposed in Widmer (1997). This allowed a learning algorithm to select the examples that had the same context as the training data and newly arrived examples. These conceptual clues helped in adapting the systems faster by filtering the historical instances used for training that had the same context as the newly arrived instances. The proposed technique was evaluated by comparing two operational systems at the Meta-level that differed in the underlying learning algorithm as well as their way of processing contextual information including METAL(B) that used a Bayesian classifier and METAL(IB) that was based on an instance-based learning. The instance-based learner was used in four variants which included: 1) context relevant instance selection; 2) instance weighting; 3) feature weighting; and 4) combination of instance and feature weighting. The general conclusion of numerous experiments that were performed using real-world and synthetic datasets was that MLL produced quite significant improvement over the existing approaches for changing environments. Additionally, from the results it could be observed that the METAL(B) approach proved to be effective in domains (datasets) with high noise rates and several irrelevant attributes whereas the instance-based approach showed higher accuracy for the remaining domains.

Klinkenberg (2005) proposed an MLL framework for automatically selecting the most promising algorithm and its parametrization at each step in time where the data was arriving in batches. For each batch a set of Meta-features (MFs) (as listed in Table 2.9) were extracted directly from the raw data which was used in the Base-level Learning (BLL) to create a Meta-example. A number of Meta-examples were used to induce a Meta-learner whenever a new batch became available, which in turn, helped in predicting the best learning algorithm and the best set of instances at a given time point. The MFs used in this work were more relevant to the problem under analysis. Furthermore, this work also investigated the aspects used to speed-up the algorithm selection process using the proposed MLL approach without losing the gained reduction in the error rate. The proposed drifting concept approaches, i.e., adaptive time window and batch selection strategy, were evaluated by comparing them with three non-adaptive mechanisms: 1) full memory; 2) no memory; and 3) fixed size window. The experiments were performed using two real-world problems: 1) information filtering of unstructured business news data; and 2) predicting business cycle from economics domain. For business news dataset both adaptive techniques outperformed trivial non-adaptive approaches. Two evaluations were performed for the business cycle dataset where the data was split into 5 and 15 equally sized batches where the fixed size window approach performed slightly better than the adaptive techniques.

Sikora (2008) proposed an MLL mechanism to learn the optimal parameters while the learning algorithm was trying to learn its target concept in a non-stationary environment. MLL was used to tune a temperature (τ) parameter of the Softmax Reinforcement Learning (RL) algorithm using a Boltzmann distribution. Moreover, the time-weighted method had been used where the action value estimates were the sample average of prior rewards. The Softmax algorithm became a random search for a higher τ value, whereas for a low value it approached a greedy search. The effectiveness of the proposed MLL algorithm was evaluated by dynamically learning the optimal value of τ using two case-studies: 1) k-Armed bandit - the classic RL problem, and 2) bidding strategy - stylized e-procurement problem. In the k-Armed bandit problem the variable k was defined as actions available to an agent and each action returned a reward from a different distribution. In this work (k=) 10 actions (1,...,10) were available to an agent where each action returned a reward using a Normal distribution. The effectiveness of MLL in a non-stationary environment was tested by rotating the reward distributions among the 10 actions. The algorithm was tested with three different temperature parameter values of 5, 50 and 500 for both stationary and dynamic environments. For the stationary environment the performance of τ=5 approached the best action with a maximum average reward. As the environment became more and more dynamic these awards kept falling. In contrast, the performance of the MLL algorithm returned better rewards in both environments as well as responded faster to the changes in the environment. The bidding problem was analysed as a 2 player symmetric game (2 homogeneous sellers) with n actions, where n was the variable cost (price) range split into equally sized bands. One of the sellers was modelled using the Softmax RL algorithm while the other one was supposed to be using different learning algorithms, i.e., ϵ-greedy - a genetic algorithm proposed by Goldberg (1989). The same three values of τ were used for both stationary and dynamic environments, where the stationary environment produced best result for the lowest value of temperature. However, no single value of temperature did best in the dynamic environment, while MLL algorithm approached the best reward for both environments. Furthermore it was observed from the experiments that the best value of τ was achieved from MLL approach in all the scenarios.

Kadlec and Gabrys (2009a) architecture supports a life-long learning by providing several adaptation mechanisms across computational path level (preprocessing methods followed by individual base-level algorithms), path combination level (combination of base-level algorithms) and a Meta-level hierarchical structure. There were four adaptation loops defined across various levels of hierarchy including self-adaptation capability of the computational and combination layer, where as the remaining two loops connected Meta-level layer to the lower layers. These feedback loops helped the proposed architecture to keep validity of the models in changing environments. It could be achieved by switching particular modules to the incremental mode. The computational path level adaptation loop consisted of the predictions feedback which were compared to the actual (target) values. Whereas at the path combination level the combinations were represented in the same way as in the computational path, which was a benefit of this representation that and meant that similar adaptation mechanisms could be applied at different levels. In the case of weighted combinations, the contribution of particular computation paths were dynamically changed to the final prediction by modifying the weights. A Meta-level adaptation had influence on the dynamic behaviour of the entire architecture. At this level the performance measures were gathered from all levels of the architecture together with the global performance. It allowed to analyse the performance achieved across various levels and also to estimate the influence of the changes at different states of the model. Several experiments demonstrated that the variety of adaptation mechanisms applied at different levels may have a significant effect on the performance of the models. One of the key contribution of the proposed architecture, was the opening of a large space for future research that could focus on the interaction between different techniques, dynamic behaviour, implementation of novel adaptation techniques and meta-level methods.

A comprehensive framework, design problems, taxonomy of adaptive learning, and different areas of learning under concept drift were presented by Zliobaite (2010). The proposed framework was used to analyse the problem of training set formation where two areas, i.e., 1) incremental learning; and 2) causes of concept drift were discussed. The incremental learning explained the difference between concept drift and periodic seasonality with examples while the causes of concept drift were elaborated on using Bayesian decision theory, where three causes were highlighted that might change over time. There were four design sub-problems and techniques addressed within the framework that needed to be solved: 1) future assumptions about source and target instances; 2) structural change types or configuration patterns of data over time; 3) identified four key learner adaptivity areas, and 4) model selection which was further categorized into two different groups. The taxonomy of concept drift learners was categorized as an evolving learner where four methods wer proposed and the methods that determined how the models or instances were to be changed at a given time were grouped separately under a triggering concept. At the end three major research areas were outlined: 1) time context; 2) transfer learning by gaining knowledge from similar type of past problems; and 3) models which have properties of adaptation incorporated into learners. Also several dimensions which are relevant to the applications implementing concept drift were defined. Figure 2.6 presents all the key areas and available solutions of learning under concept drift.

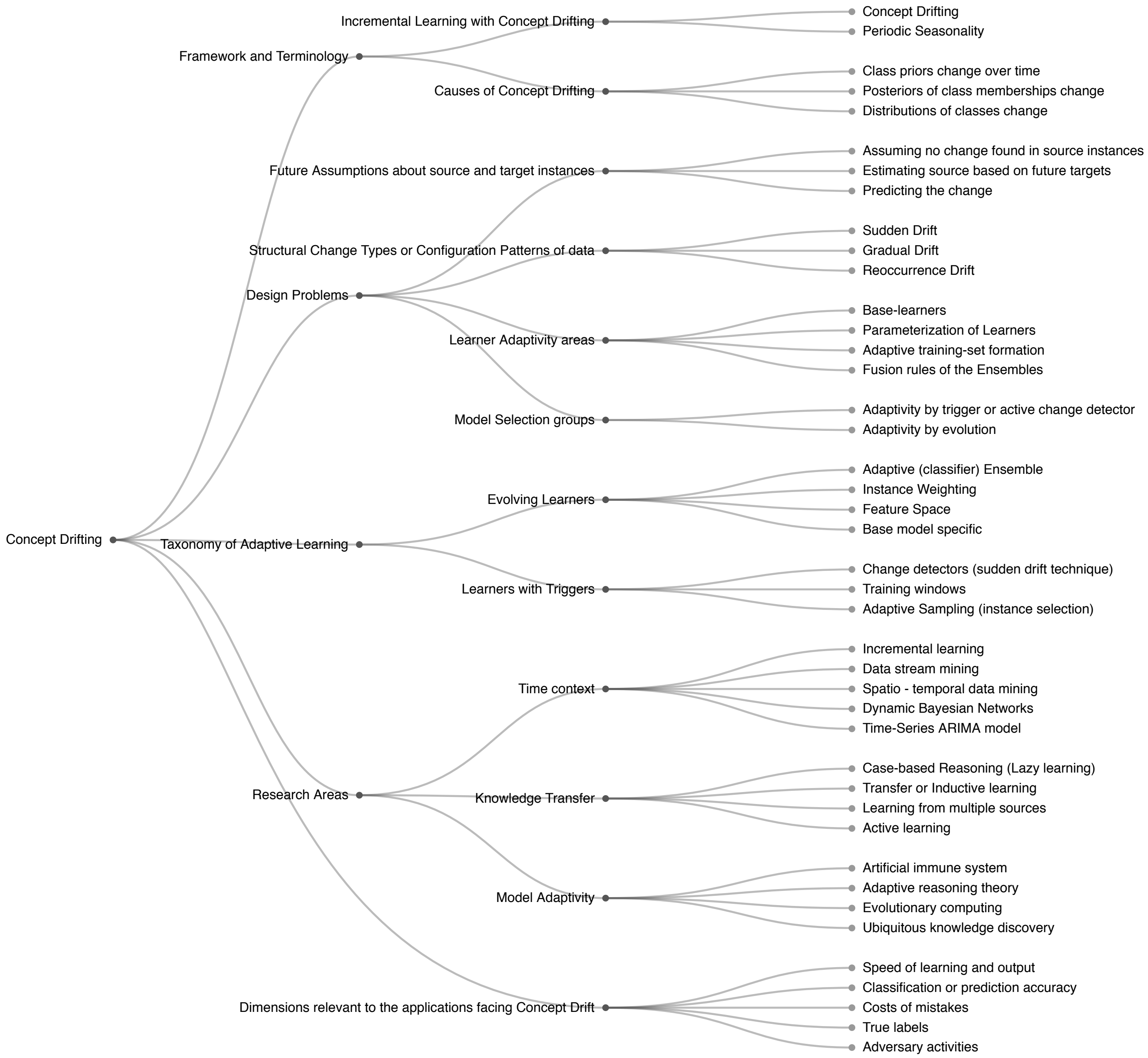
Concept Drifting
Framework and Terminology
Incremental Learning with Concept Drifting
Concept Drifting
Periodic Seasonality
Causes of Concept Drifting
Class priors change over time
Posteriors of class memberships change
Distributions of classes change
Design Problems
Future Assumptions about source and target instances
Assuming no change found in source instances
Estimating source based on future targets
Predicting the change
Structural Change Types or Configuration Patterns of data
Sudden Drift
Gradual Drift
Reoccurrence Drift
Learner Adaptivity areas
Base-learners
Parameterization of Learners
Adaptive training-set formation
Fusion rules of the Ensembles
Model Selection groups
Adaptivity by trigger or active change detector
Adaptivity by evolution
Taxonomy of Adaptive Learning
Evolving Learners
Adaptive (classifier) Ensemble
Instance Weighting
Feature Space
Base model specific
Learners with Triggers
Change detectors (sudden drift technique)
Training windows
Adaptive Sampling (instance selection)
Research Areas
Time context
Incremental learning
Data stream mining
Spatio - temporal data mining
Dynamic Bayesian Networks
Time-Series ARIMA model
Knowledge Transfer
Case-based Reasoning (Lazy learning)
Transfer or Inductive learning
Learning from multiple sources
Active learning
Model Adaptivity
Artificial immune system
Adaptive reasoning theory
Evolutionary computing
Ubiquitous knowledge discovery
Dimensions relevant to the applications facing Concept Drift
Speed of learning and output
Classification or prediction accuracy
Costs of mistakes
True labels
Adversary activities

Figure 2.6: Learning under Concept Drifting (Zliobaite, 2010)

An MLL approach for periodic and automatic algorithm selection for time-changing data, named Meta-Stream, was presented by Rossi et al. (2012). A Meta-classifier was periodically applied to predict the best learning algorithm for a new unlabelled chunk of data. General DSIT MFs of Travel Time Prediction (TTP) problem were extracted from the historical and new data (see Figure 2.3) and mapped together with their predictive performance computed from different models to induce the Meta-classifier. Experiments were performed to compare the performance of the MetaStream to the default trial-and-error approach for both static and dynamically updating strategies at the Meta- and Base-levels. Moreover, the Base-level MetaStream and Default results were compared with the dynamic Ensemble approach. The learning strategy adopted at the Base-level can be seen in Table 2.4, also the training window (ω) of 1000 instances with a step size (λ) of

1 was used at this level. The Meta-level learning strategy was presented in Table 2.7. The Meta-examples (MEs) labelled as tie were investigated separately by keeping and discarding them from the training and test sets. The empirical results showed that the MetaStream outperformed the baseline and ensemble approaches with a significant margin in most of the cases for both stationary and dynamic environments. In general, the two pairs of algorithms, e.g., Random Forests (RF)-Classification and Regression Trees (CART) and Support Vector Machines (SVM)-CART were found to be the best algorithms for TTP problem. Finally, the authors also realized that the MFs should be related to the non-stationary data problem rather than characteristics which were extracted for the traditional MLL problems.

Rossi et al. (2014) extended their original work (Rossi et al., 2012) in two main directions: 1) instead of selecting only a single algorithm, a combination of multiple regressors could be selected, when the average of the predictions performed better than the individual; and 2) more comprehensive experimental evaluation was performed by adding another real-world problem - *Electricity Demand Prediction (EDP)* (see Table 2.1). Furthermore the list of MFs extracted from the data was also enhanced in this work, as listed in Table 2.8. The characteristics were extracted separately from the training and evaluation windows because the training window had target information available from where supervised characteristics could be extracted, i.e., information about the relationship between the predictive and target variables. The pool of Base- and Meta-level algorithms with their configurations are listed in Table 2.4 and Table 2.7 respectively. The experimental results showed that for TTP dataset the pair of regressors, regardless of the presence of tie resolution strategy, outperformed the default and ensemble based approaches. However, in case of EDP, the MetaStream clearly outperformed the default, but was worse than the ensemble which could lead to a conclusion that the observations made for pairs of regressors were also valid for multi-regressors. Moreover, slightly higher error rate was recorded for RF Meta-learner of the MetaStream than the default but was lower than the ensemble approach for the TTP dataset, whereas for the EDP dataset the MetaStream outperformed the default but was worse than the ensemble. These results showed that the MetaStream was able to select the best algorithm more accurately than the baseline trial-and-error and ensemble-based approaches in a time-changing environment.

Table 2.8: Meta-features used in MetaStream to characterize the data

Meta-features	**Training window**	**Selection window**
Average, Variance, Minimum, Maximum and Median of continuous features	✔	✔
Average, Variance, Minimum, Maximum and Median of the target	✔	
Correlation between numeric features		✔
Correlation of numeric attributes to the target	✔	
Possibility of existence of outliers in numeric features		✔
Possibility of existence of outliers in the target	✔	
Dispersion gain	✔	
Skewness of numeric features		✔
Kurtosis of numeric features		✔

2.5.1 Discussion and Summary

This section covered the adaptability mechanisms of a number of existing systems using MLL approaches. In these studies the main focus was put on the applicability of MLL particularly in the context of non-stationary environments. MLL can be beneficial in such a case by minimizing the processing time that is consumed to periodically train the model, extracting recurring concepts, automatically detecting concept drift and estimating dynamic adaptive window size, which in turn can generate accurate predictions in dynamic environments. However, applying MLL to support an adaptive mechanism is a recent and emerging area. As a result most of the research use the same MFs for a time-varying environment as for the stationary environments. If MLL is introduced in a system then the overall performance of such a system becomes dependent on an appropriate representation of the problem at the Meta-level in the form of extracted, informative MFs. The drawback of using a set of MFs which are usually used in a stationary environment is that the entire target dataset should be available at once when MLL is applied to find the best algorithm for that dataset. This is not normally the case for streaming data and unavailability of target variables makes calculation of some useful MFs impossible.

Widmer (1997)'s work on applying MLL for non-stationary environments is considered to be the earliest effort. It addressed two key areas in the context of dynamic environments: 1) dynamic tracking of changes; and 2) extraction of recurring concepts. The problem representation in Widmer (1997) was quite general as very few predictive and contextual MFs were extracted. However, neither of the two proposed MLL approaches performed better then the default for several domains. Klinkenberg (2005) used different BLL algorithms which were automatically selected at the Meta-level. Additionally the Meta-level approach for adaptive time window and recurring concept extraction for the target concept were part of the research. The research was one of the initial efforts to represent an adaptivity problem with the relevant MFs rather than using general features which were usually productive for the stationary environment. Although these features (as listed in Table 2.9) were not sufficiently expressive to represent a non-stationary environment at the Meta-level, they were still better than general features (used to represent stationary problems) as evidenced by the experiments which showed a significant improvement.

Sikora (2008) proposed a reinforcement learning approach to address the automatic algorithm recommendation problem using MLL in a non-stationary environment. The focus of the research was to find the optimal value of the Softmax algorithm's parameter τ where it would recommend the best algorithm for the target concept at the Meta-level. The same deficiency was observed in this work that the non-stationary problem representation was not addressed in sufficient detail and focus was only on the algorithm recommendation using MFs which were proposed for static data. Kadlec and Gabrys (2009a) proposed a life-long learning architecture that provided several adaptation mechanisms across a pool of candidate learning algorithms and their combinations. The dynamic behaviour of the entire architecture was analysed at the Meta-level where the global performances as well as information from both pools could be analysed to estimate the influence of the changes at different levels of the model. The decrease in prediction ability of a local model below a certain level was considered as a new concept which led to building a new receptive field. The landmarking approach was quite simple and effective to detect concept drift, and based on that, periodically train a new local predictor. The effectiveness of MLL for the two mentioned areas was supported by improved results recorded from two case-studies.

Rossi et al. (2012) approach was quite similar to Klinkenberg (2005) where periodic algorithm selection for a time-changing data was proposed. Similarly to various other studies the authors computed the Descriptive, Statistical and Information-Theoretic (DSIT) MFs. Even though the Meta-level approach performed better than the Base-level, there was no comparison shown with the other MLL systems from where it could be concluded that even the general representation of the problem could work for a non-stationary environment. The problem representation using general MFs was a drawback of this effort which was subsequently attempted to rectify in Rossi et al. (2014). The authors computed separate MFs for historical and incoming data. As the target variable was not available in the incoming data the unsupervised features were computed for the data available in the evaluation window. The performance of the proposed approach was better than the BLL and worse than an ensemble based approach but despite this it was considered to be a good effort towards representing a time-varying problem at the Meta-level. In almost all the studies that are discussed in this section MLL outperformed the BLL methods. However, a common drawback has been observed in the problem representation area at the Meta-level for time-varying data. Most of the work used general MFs whereas only some tried to focus on this area by proposing some features for the non-stationary data.

Table 2.9: Adaptive mechanisms used in previous studies

Research Work	**Adaptivity mechanisms addressed**	**Meta-features/Parameters**
Widmer (1997)	Recurring concept extraction	window size=100 and significance level=0.01
Klinkenberg (2005)	Recurring concept extraction, adaptive time window, periodic algorithm selection	No. of batches used for training at the previous batch No. of non-interrupted most recent training batches Most successful learner on the previous batch Most successful learner overall on all batches seen so far
Kadlec and Gabrys (2009a)	Concept drift detection and Periodic algorithm selection	Landmarking
Rossi et al. (2012)	Periodic algorithm selection	ML: ω=1000, λ=1, η=0 MLL: ω=300, γ=25, λ=1, η= 0
Rossi et al. (2014)	Periodic algorithm selection (with more relevant representation of the non-stationary problem)	TTP dataset: ML: ω=1000, λ=1, η=2 MLL: ω=300, γ=24, λ=1, η=0 EDP dataset: ML: ω=672, λ=336, η=0 MLL: ω=300, γ=25, λ=1, η=0

Chapter 3

Research Challenges

The goal of Meta-level Learning (MLL) is to analyse and recommend the best methods and techniques for a problem on the basis of previously solved problems and without or with minimal intervention of human experts (Duch et al., 2011). The existing approach of analysing the problem and selecting the best learning algorithm is to apply a wide range of algorithms, with many possible parametrizations, on a problem simultaneously and then select an algorithm from a ranked list based on performance estimates like accuracy, execution-time, etc. Also choosing the best algorithm for a specific problem in an ever increasing number of models and their almost infinite configurations is a challenging task. Even with sophisticated and parallel learning algorithms, the computational power in terms of the execution-time, memory, and the overall human effort are still one of the biggest limitations. Every task leads to new challenges and demands dedicated effort for detailed analysis and modelling.

The main theme of this work is research on MLL strategies and approaches in the context of adaptive multi-level, multi-component predictive systems for time-varying environments. In these systems there are multiple areas where MLL can be used to efficiently recommend the most appropriate methods and techniques. Therefore three areas of an evolving predictive systems dealing with streaming data have been identified where the applicability of MLL can be an effective and efficient approach. These are listed below:

1. Learning A Path Recommendation:
   A learning path includes pre-processing steps, learning algorithms or their combination and adaptivity mechanism parameters. These three components are interlinked with each other where MLL recommends the learning algorithm or their combinations preceded by optimised pre-processing steps from a pool of available methods. The adaptivity mechanism parameters are the additional parameters which are linked with the algorithm's configuration. Figure 3.1 shows the complex learning path recommender.

   i. Pre-processing Steps Recommendation:
   MLL can be applied to find the most appropriate combination of pre-processing steps. Since in time-varying environment trying various pre-processing methods and techniques to find the best combination for a concept will make the entire system ineffective. Instead

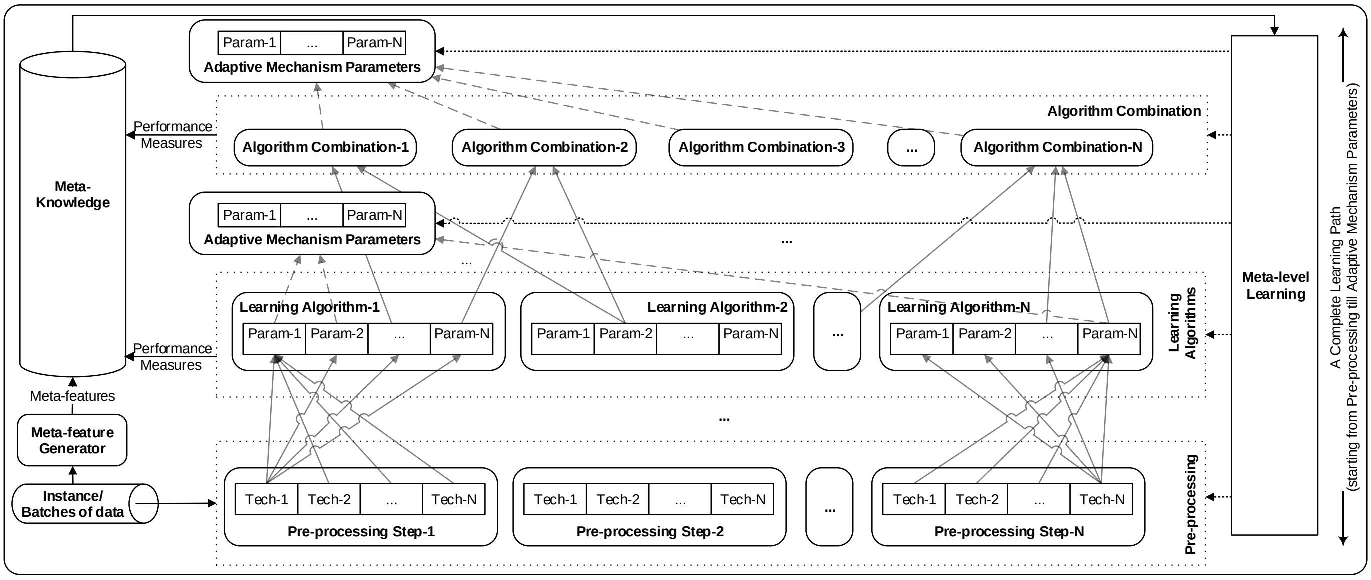

Figure 3.1: Learning Path Recommendation

of spending time on testing various methods on every concept drift detection MLL can help to instantly recommend the best pre-processing steps from the methods under observations.

ii. Algorithm or Combination Recommendation:
Finding the optimal algorithm for a dataset is a traditional application of MLL (Giraud-Carrier, 2008). Automatic discovery of the optimal algorithm can be beneficial for both stationary and particularly non-stationary environments where it can help in minimizing the processing time which is usually spent on the rigorous testing of various learning algorithms with their different parametrizations. MLL can recommend the best learning algorithm, its parametrization, and their combination instantly from the pool of available learners.

iii. Adaptivity Mechanism Parameters:
The adaptive mechanism with static parameters, i.e., training and evaluation window size, step size, and delay, would be ineffective for the dynamic environments where the underlying distribution of incoming data keeps changing. These parameters can be bound with learning algorithm configuration. The most appropriate set of adaptivity parameters can be extracted at the Meta-level based on the best learning algorithm selected for the current concept.

2. Recurring Concepts Extraction:
In a non-stationary environment the underlying distribution of the incoming data keeps changing which in turn can make even the most recent historical concept ineffective to retrain the model for the current concept. Using MLL the historical batches (concepts) of data could be extracted from the Meta-knowledge (MK) which in turn can be used as a training-set for the current data. This process can be named as *Reverse Knowledge Extraction* where Meta-features

(MFs) of the current concept can be used to extract the Meta-examples (MEs) of relevant concepts from MK datasets. These MEs could ultimately lead to extracting the model whose underlying distribution follows the concept which is currently under observation. This model can be retrained to incorporate a new concept in the existing model.

3. Concept Drift Detection:
   In an adaptive mechanism retraining of a model is usually triggered by a change detection process. MLL can help in automatically identifying a drift to maximize the efficiency of the system. MLL can help to automatically detect the concept drift and trigger the algorithm retraining process instantly. For instance, the MFs of incoming data can be computed as well as cumulated on arrival of every batch and simultaneously compared with the set of MEs, from MK dataset, whose learning algorithm (used as a target variable in the MK) is used to score the current batches of data. The concept drift can be detected at the Meta-level if the ME of the current concept does not match with the cluster of MEs whose learning algorithm is currently selected.

The scope of this research is limited to the representation of MK in non-stationary environments which falls under the algorithm or the combination recommendation tasks. The applicability of MLL in these areas leads to several research questions which are listed below.

1. Gathering examples of datasets to build a static Meta-knowledge database:
   i. The time-changing environments require dynamic MK databases which must be updated with the MFs of different batches of data having different distribution. A dynamic MK database keeps on growing with the ME of new concepts. Apart from the dynamically growing database which will gradually build-up, a static MK database may be required at least for the initial phase of the system. When do the benefits of a static database outweigh the costs of maintaining it? Furthermore what are the alternative techniques of utilizing MLL without having prior knowledge particularly for the initial phase of the system.
   ii. Building-up a static MK database would raise another research challenge of what strategy should be adopted to generate synthetic MEs, i.e., either by directly transforming the existing MEs which are generated by limited real-world datasets or by generating artificial examples of datasets?

2. A Base-level Learning strategy to compute performance measures of Meta-examples:
   i. Base-level Learning (BLL) is used to build predictive models using examples of datasets to compute a set of performance measures which are mapped with their respective MEs. What strategy would be adopted to select the best learning algorithm and its parametrization for an ME at the Base-level, i.e., level of granularity of algorithm parametrization, algorithm ranking or combination, model validation and performance measures?

3. Feature generation and selection to represent a problem at the Meta-level:

   i. Would the traditional MF generation approaches, which are usually specialized for the algorithm recommendation task, be adequate to represent three new proposed areas of the system at the Meta-level or based on the complexity of the new problems a different representation would be required?

   ii. In a non-stationary environment the target variable would not be available at the time of algorithm selection at the Meta-level. It will restrict computing some important MFs, e.g., correlation between the target and the predictive variables. What would be the impact of the absence of these significant features on the performance of MLL and in a later stage how the MK database could be updated when the target variable will be known?

4. Representation and storage of dynamically growing complex Meta-Knowledge database:

   i. What level of granularity would be required for the appropriate representation of a problem? For instance, the target variable of the MEs would be only the best learning algorithm, ranking, algorithm parametrization or their combination?

   ii. What type of performance measures will be stored in the MK database for three different areas, e.g., accuracy, run-time speed? For instance, the run-time speed measure might be useful particularly for a non-stationary environment which would help to identify both an accurate as well as efficient learning algorithm.

5. Meta-level Learning strategy for algorithm recommendation:

   i. What strategies and algorithms would be used at the Meta-level to efficiently search the target objectives of the mentioned three areas from the MK database?

   ii. If MLL process recommends a different learning algorithm and its parametrization for the target concept then what would be the strategy of replacing the current algorithm and how this change would impact the overall performance of the system?

Chapter 4

Summary

This literature review and identification of key research challenges have been focused on the detailed study of existing Meta-level Learning (MLL) concepts and systems for both stationary and non-stationary environments. We are particularly interested in fully automating the process of building, deployment and maintenance of potentially complex multi-component, multi-level evolving predictive systems operating in continously changing environments, as described in some of our previous publications and those resulting from the INFER project.

The review of the existing research has been structured into the coverage of five key components of an MLL system: (i) Available real and synthetic datasets for modelling at the Meta-level; (ii) Meta-features generation and selection approaches; (iii) Base-level learners as an input to the Meta-learning; (iv) Meta-learning; (v) Meta-learning based adaptive mechanisms for non-stationary environments.

There are various methods to gather Examples of Datasets (EoD) discussed though all of them have some limitations. Similarly several Meta-feature generation techniques are reviewed from previous work though the majority of them have been introduced in the context of and are suitable for a stationary MLL system. Hence the applicability and effectiveness of such Meta-features for non-stationary environments remains an open research question. A consistently and systematically evaluated performance of base-models on EoDs forms a critical part of a reliable input data (i.e. label or target variable) for the MLL. Collecting such performance data is the most time and processor intensive task especially if numerous configurations and parametrisations of base-learners are to be adequately taken into account. Such reliable collection of previously solved problems with thorough benchmarking of base-learners suitable for MLL do not currently exist and remain an open challenge.

A number of previously proposed MLL systems have been discussed in detail which included the application of MLL to both supervised and unsupervised learning problems. The development and evolution of the MLL field in the last three decades has been discussed and various systems have been compared with the previous ones. However, there are very few systems that have been targeted towards and can deal with non-stationary problems which is our main area of interest. It is only in the last five years that non-stationary MLL have been receiving some interest. The primary focus has been on the problem representation of a streaming data at the Meta-level.

There are multiple roles for Meta-learning in the scope of INFER project and the developed automated and autonomous predictive modelling system and approaches working in continuously changing environments which we are intending to explore in our continuing research in this area.

Appendix A

Meta-features

Table A.1: Meta-features used in various studies

Meta-Features	**Rendell et al. (1987)** **Rendell and Cho (1990)**	**King et al. (1995)**	**Sohn (1999)**	**Lindner and Studer (1999)** **Berrer et al. (2000)** **Giraud-Carrier (2005)**	**Bensusan et al. (2000)**	**Bensusan and Giraud-Carrier (2000)**	**Pfahringer et al. (2000)**	**Todorovski et al. (2002)**	**Peng et al. (2002)**	**Kopf and Iglezakis (2002)**	**Brazdil et al. (2003)**	**Prudencio and Ludermir (2004)**	**Prudencio and Ludermir (2008)** **Guerra et al. (2008)**	**Wang et al. (2009)**	**Lemke and Gabrys (2010a)**	**Abdelmessih et al. (2010)**	**Rossi et al. (2012)**	**Feurer et al. (2014)**
Descriptive Meta-features																		
Number of Classes (k)		✔	✔	✔			✔											✔
Frequency of most common class	✔						✔											
Number of Features (p)	✔[1]	✔	✔															✔
Total Instances (N)		✔	✔	✔			✔				✔	✔			✔			✔
Dataset Dimentionality																		✔
Number of Training instances (r)	✔[1]	✔	✔										✔[2]					
Number of Test instances (t)	✔	✔	✔															
Sampling Distribution	✔	✔																
Number of Binary Features (b)	✔	✔	✔															
Number of Numeric features (n)	✔			✔			✔											✔
Number of Nominal features (s)	✔			✔			✔											✔
Proportion of binary features (b/p)			✔															
Proportion of nominal features (s/p)				✔							✔							✔
Span of nominal values				✔														
Average of nominal values				✔														✔
Training instances to features ratio (N/p)			✔										✔[2]					
Proportion of training instances (r/N)			✔															
Statistical Meta-features																		
Relative probability of missing values				✔							✔							✔
Instances with missing values				✔														✔
Proportion of features with outliers				✔							✔						✔	
Mean Skewness (SKEW)		✔	✔	✔										✔	✔[3]		✔	
Mean Kurtosis (KURT)		✔	✔	✔										✔	✔[3]		✔	
Average																	✔	
Variance																	✔	
Minimum																	✔	
Maximum																	✔	
Median																	✔	
Correlation between predictor and target																	✔	
Standard Deviation (StdDev) of the class distribution				✔											✔[4]			✔
Homogeneity of Covariances (S/D Ratio)		✔	✔	✔														

[1] only these two features are used in Rendell et al. (1987), they are also part of Rendell and Cho (1990)

[2] Log

[3] of series

[4] of de-trended series

Canonical Correlation (CANCOR)		✔	✔	✔							✔							
Number of Discriminant Functions (DiscFunc)				✔														
Mean Absolute Correlation Coefficient (CORR)		✔	✔															
Relative proportion of largest Eigenvalue (FRACT)		✔	✔	✔														
Wilks'lambda Distribution (Wlambda)				✔														
Default Accuracy				✔														
coefficient of variation (COEF-VAR)												✔						
absolute value of the SKEW and KURT coefficient												✔						✔
Time-series (TS) mean absolute values of first 5 auto-correlations (Mean-CORR)												✔						
TS test of significant auto-correlations (TAC)												✔						
TS significance of the 1, 2, and 3 Auto-correlation (TAC-1,2,3)												✔						
TS test of Turning Points for randomness												✔						
TS first coefficient of auto-correlation (AC1)												✔						
TS type												✔						
TS trend												✔		✔	✔[5]			
TS turning point												✔[6]			✔			
TS Durbin-Watson statistic of regression residual (DW)															✔			
TS step changes															✔			
TS predictability measure															✔			
TS non-linearity measure															✔			
TS largest Lyapunov exponent														✔	✔			
TS 3 largest power spectrum frequencies															✔			
TS maximum value of power spectrum															✔			✔
TS number of peaks > 60%															✔			
TS auto-correlations at lags 1 and 2															✔			
TS partial auto-correlations at lags 1 and 2															✔			
TS seasonality Measure														✔	✔			
TS mean Symmetric Mean Absolute Percentage Error (SMAPE) - mean deviated SMAPE															✔			
TS mean SMAPE / mean deviated SMAPE															✔			
TS mean of correlation coefficient															✔			
TS StdDev of correlation coefficient															✔			
TS methods in top performing cluster															✔			
TS distance top performing cluster to second best															✔			
TS Serial CORR Box-Pierce statistic														✔[7]				
TS Non-linear autoregressive structure														✔[8]				
TS Self-similarity (Long-range Dependence														✔				
TS Periodicity (frequency)														✔				
Min. of CORR between predictors and target													✔					
Max. of CORR between predictors and target													✔					
Mean of CORR between predictors and target													✔					
StdDev of absolute value of CORR between predictors and target													✔					
Min. of CORR between pairs of predictors													✔					
Max. of CORR between pairs of predictors													✔					
Mean of CORR between pairs of predictor													✔					
StdDev of absolute value of CORR between pairs of predictors													✔					
Information Theoretic Meta-features																		
Entropy of Classes (HC)			✔	✔		✔					✔							
Entropy of nominal features			✔	✔		✔												
Joint Entropy of Classes (HCX)			✔	✔		✔												
Average Mutual Information between Class and Nominal Features (MCX)			✔	✔		✔					✔							
Class Entropy to Mutual information ratio				✔		✔												✔
Noise to Signal Ratio (NoiseRaio)	✔			✔		✔												
Dispersion Gain																	✔	

[5] StdDev of series / StdDev of de-trended series

[6] ratio

[7] of raw and trend/seasonally adjusted

[8] of raw and trend/seasonally adjusted

Landmarkers																		
Decision Nodes Learner (Decision Nodes)						✔		✔								✔		
Worst Nodes Learner (Worst Nodes)						✔										✔		
Randomly Chosen Nodes Learner (Randomly Chosen Nodes)						✔										✔		✔
Naive Bayes classifier (NB)						✔		✔								✔		✔
k-Nearest Neighbour (k-NN)				✔[9]		✔[10]	✔	✔[10]		✔						✔[10]		✔
Elite-Nearest Neighbour (e-NN)						✔												
Linear Discriminant Analysis (LDA)						✔		✔								✔		✔
C5.0 Decision Tree (C5.0 tree)								✔										✔
C5.0 Adaptive Boosting (C5.0 boost)							✔	✔										
C5.0 Rule Induction (C5.0 rules)							✔	✔		✔								
Rule Learner (Ripper)							✔											
Linear Discriminant Trees (Ltree)							✔			✔								
Average Nodes Learner (Average Nodes)																✔		
Model-based Meta-features																		
Nodes per attribute					✔													
Nodes per instance					✔													
Average leaf corroboration					✔													
Average gain-ratio difference					✔													
Maximum depth					✔													
No. of repeated nodes					✔													
Shape					✔													
Homogeneity					✔													
Imbalance					✔													
Internal symmetry					✔													
No. of Nodes in each level - width									✔									
No. of levels - Height									✔									
No. of nodes in the tree									✔									
No. of leaves in the tree									✔									
Maximum no. of nodes at one level									✔									
Mean of the no. of nodes									✔									
StdDev of the no. of nodes									✔									
Length of the Shortest branch									✔									
Length of the Longest branch									✔									
Mean of the branch length									✔									
StdDev of the branch length									✔									
Minimum occurrence of Features									✔									
Maximum occurrence of Features									✔									
Mean of the no. of occurrences of Features									✔									
StdDev of no. of occurrences of Features									✔									
Weight sum of dataset																		
Minimum weight sum of dataset																		
Average weight sum of dataset																		
StdDev weight sum of dataset																		
No. neighbours for dataset																		
Minimum No. neighbours for dataset																		
Maximum No. neighbours for dataset																		
Average No. neighbours for dataset																		
StdDev of No. neighbours for dataset																		
Principal Component Analysis (PCA) 95%																		✔
PCA skewness																		✔
PCA kurtosis																		✔
Total Meta-features	9	13	19	25	10	14	8	7	15	3	7	11	10	9	23	7	10	22

[9] k = 3 used only in Giraud-Carrier (2005)

[10] k = 1

Glossary of Terms

A

AL Average Linkage. 36, 37

ARIMA Auto-regressive Integrated Moving Average. 23, 24, 33, 35, 37

ARR Adjusted Ratio of Ratios. 16, 23

Average Nodes Average Nodes Learner. 54

B

b Number of Binary Features. 52

BLL Base-level Learning. 2, 22, 24, 31, 38, 43, 44, 48

C

C4.5 C4.5 Decision Tree algorithm. 16, 23, 36

C5.0 boost C5.0 Adaptive Boosting. 15–17, 22, 23, 36, 54

C5.0 rules C5.0 Rule Induction. 17, 23, 36, 54

C5.0 tree C5.0 Decision Tree. 16, 17, 23, 24, 36, 54

CANCOR Canonical Correlation. 53

CART Classification and Regression Trees. 23, 24, 42

CASTLE Causal Structure for Inductive Learning. 23

CBR Case-based Reasoning. 15

CL Complete Linkage. 36, 37

CN2 CN2 Induction Algorithm. 23, 36

CORR Mean Absolute Correlation Coefficient. 32, 36, 53

CoV Coefficient of Variation. 34

CV Cross-Validation. 23, 24, 31

D

DBS DB-Scan. 37

DCT Dataset Characterization Tool. 14–20, 30, 31, 35

Decision Nodes Decision Nodes Learner. 36, 54

DiscFunc Number of Discriminant Functions. 53

DMA Data Mining Advisor. 27–29, 35

DSIT Descriptive, Statistical and Information-Theoretic. 2, 14–20, 27, 28, 30, 34–37, 44

DT Decision Trees. 18, 32, 37

DW Durbin-Watson statistic of regression residual. 53

E

e-LICO e-Laboratory for Interdisciplinary Collaborative Research. 29, 30, 34

e-NN Elite-Nearest Neighbour. 36, 54

EoD Examples of Datasets. 1, 6, 11, 12, 19, 24, 29, 50

ES Exponential Smoothing. 23, 36, 37

F

FF Farthest First. 37

FLD Fisher's Linear Discriminant. 23

FRACT Relative proportion of largest Eigenvalue. 53

H

HC Entropy of Classes. 53

HCX Joint Entropy of Classes. 53

I

IBL Instance-based Learning. 17, 23, 36

ICA Independent Component Analysis. 19

ID3 Iterative Dichotomiser 3. 36

IDA Intelligent Discovery Assistant. 2, 27, 29, 34

INDCART Inductive CART. 23, 36

K

k Number of Classes. 52

KD Knowledge Discovery. 27

k-M k-Means. 36, 37

k-NN k-Nearest Neighbour. 17–19, 22–24, 31, 32, 36, 37, 54

KURT Kurtosis. 52, 53

L

LazyDT Lazy Decision Trees. 36

LDA Linear Discriminant Analysis. 16–18, 23, 36, 54

Ltree Linear Discriminant Trees. 17, 23, 36, 54

LVQ Learning Vector Quantization. 36

M

M Mixture Models. 36, 37

MA Moving Average. 24, 33

MAE Mean Absolute Error. 23, 24

MARS Multivariate Adaptive Regression Splines. 24

MCX Average Mutual Information between Class and Nominal Features. 53

MDS Multi-dimensional Scaling. 19

ME Meta-example. 35, 42, 48, 49

METAL Meta-Learning Assistant. 2, 7, 16, 27, 28, 30, 31

METALA Meta-learning Architecture. 2, 27, 29

MF Meta-feature. 1, 2, 6, 10–12, 14–22, 24, 26–35, 38, 39, 41–44, 47–49

MK Meta-knowledge. 1, 6, 7, 12, 22, 24, 26, 29, 30, 32, 33, 35, 38, 47–49

ML Machine Learning. 1, 6, 9, 11, 12, 26, 30, 32, 38, 44

MLL Meta-level Learning. 1–3, 5–7, 10–12, 14, 18, 19, 22, 24–35, 38, 39, 41–44, 46–50

MLP Multi-layer Perceptron. 17, 23, 24, 32, 34, 36, 37

MLR Multiple Linear Regression. 23

MLT Machine Learning Toolbox. 2, 26, 28

MSE Mean Squared Error. 23, 24, 32, 36

N

N Total Instances. 52

n Number of Numeric features. 52

NB Naive Bayes classifier. 17–19, 23, 24, 36, 37, 54

NBT Naive Bayes/Decision-Tree. 36

NN Neural Network. 18, 23, 24, 33, 36, 37

NoiseRaio Noise to Signal Ratio. 53

O

OC1 Oblique Classifier-1. 36

OneR One Rule Learner. 17, 24, 36, 37

P

p Number of Features. 52

PaREn Pattern Recognition Engineering. 3, 28, 29, 34

PCA Principal Component Analysis. 19, 54

PEBLS Parallel Exemplar-Based Learning System. 36

PPR Projection Pursuit Regression. 24

Q

QPC Quality of Projected Clusters. 11, 19

Quadra Quadratic Classifier. 23

R

r Number of Training instances. 52

Randomly Chosen Nodes Randomly Chosen Nodes Learner. 36, 54

RapidAnalytics open-source data-mining and predictive analysis solution. 29

RBF Radial-basis Function. 17, 23, 32, 36

RF Random Forests. 17, 24, 37, 42

Ripper Rule Learner. 16, 17, 23, 36, 54

RL Reinforcement Learning. 39

RMSE Root Mean Squared Error. 24

RW Random Walk. 23, 36, 37

S

s Number of Nominal features. 52

S/D Ratio Homogeneity of Covariances. 52

SKEW Skewness. 52, 53

SL Single Linkage. 36, 37

SMAPE Symmetric Mean Absolute Percentage Error. 24, 33, 53

SMART Smooth Multiple Additive Regression Technique. 23

SNN Shared Nearest Neighbours. 36, 37

SP Spectral Clustering. 36

SRCC Spearman's Rank Correlation Coefficient. 23, 34

STABB Shift To A Better Bias. 26, 28

StatLog Statistical and Logical learning. 2, 14, 17, 26, 28, 30

StdDev Standard Deviation. 52–54

SVM Support Vector Machines. 17–19, 22, 24, 32, 34, 36, 37, 42

SVR Support Vector Regression. 34

T

t Number of Test instances. 52

TS Time-series. 8, 10, 22, 24, 31, 33, 35, 37, 53

U

UCI UCI Machine Learning Repository. 7–10, 12, 15–17, 29, 30

V

VBMS Variable-bias Management System. 14, 19, 26, 28

W

Wlambda Wilks'lambda Distribution. 53

Worst Nodes Worst Nodes Learner. 54

X

XM X-Means. 37

References

Abdelmessih, Sarah D., Faisal Shafait, Matthias Reif, and Markus Goldstein (2010). “Landmarking for Meta-Learning using RapidMiner”. In: *RapidMiner Community Meeting and Conference.* Online.

Aha, David W., Dennis Kibler, and Marc K. Albert (1991). “Instance-based Learning Algorithms”. In: *Machine Learning*, pp. 37–66.

Al-Jubouri, Bassma and Bogdan Gabrys (2014). “Multicriteria approaches for predictive model generation: a comparative experimental study”. In: *Computational Intelligence in Multi-Criteria Decision-Making (MCDM), 2014 IEEE Symposium on.* IEEE, pp. 64–71.

Alpaydin, Ethem (2010). *Introduction to Machine Learning.* 2nd. The MIT Press.

Apeh, Edward, Bogdan Gabrys, and Amanda Schierz (2014). “Customer profile classification: To adapt classifiers or to relabel customer profiles?” In: *Neurocomputing* 132, pp. 3–13.

Auer, Peter, Nicolo Cesa-Bianchi, and Paul Fischer (2002). “Finite-time analysis of the multiarmed bandit problem”. In: *Machine learning* 47 (2).

Bache, Kevin and Moshe Lichman (2013). *UCI Machine Learning Repository.* URL: `http://archive.ics.uci.edu/ml`.

Bakirov, Rashid and Bogdan Gabrys (2013). “Investigation of expert addition criteria for dynamically changing online ensemble classifiers with multiple adaptive mechanisms”. In: *IFIP International Conference on Artificial Intelligence Applications and Innovations.* Springer, Berlin, Heidelberg, pp. 646–656.

Bensusan, Hilan and Christophe G. Giraud-Carrier (2000). “Discovering Task Neighbourhoods Through Landmark Learning Performances”. In: *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD).* London, UK, UK: Springer-Verlag, pp. 325–330.

Bensusan, Hilan, Christophe G. Giraud-Carrier, and Claire J. Kennedy (June 2000). “A Higher-order Approach to Meta-learning”. In: *Proceedings of the ECML workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, 109—117.

Bensusan, Hilan and Alexandros Kalousis (2001). “Estimating the Predictive Accuracy of a Classifier”. In: *Proceedings of the 12th European Conference on Machine Learning (EMCL).* London, UK: Springer-Verlag, pp. 25–36.

Bergstra, J., R. Bardenet, Y. Bengio, and B. Keegl (2011). “Practical Network Blocks Design with Q-Learning”. In: *Proceedings of the 25th International Conference on Advances in Neural Information Processing Systems.*

Bernstein, Abraham and Foster Provost (2001). “An Intelligent Assistant for the Knowledge Discovery Process”. In: *Proceedings of the International Joint Conferences on Artificial Intelligence*

(IJCAI) Workshop on Wrappers for Performance Enhancement in KDD. Seattle, Washington, USA.

Bernstein, Abraham, Foster Provost, and Shawndra Hill (Apr. 2005). “Toward Intelligent Assistance for a Data Mining Process: An Ontology-Based Approach for Cost-Sensitive Classification”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.4, pp. 503–518.

Berrer, Helmut, Iain Paterson, and Jorg Keller (2000). “Evaluation of Machine-Learning Algorithm Ranking Advisors”. In: *Proceedings of the PKDD-2000 Workshop on DataMining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions.*

Bishop, C. and P. E. Hart (1995). “Neural Networks for Pattern Recognition”. In:

Bossard, Lukas, Matthieu Guillaumin, and Luc J. Van Gool (2014). “Food-101 - Mining Discriminative Components with Random Forests.” In: *ECCV (6)*. Vol. 8694. Lecture Notes in Computer Science. Springer, pp. 446–461.

Botia, Juan A., Antonio F. Gomez-Skarmeta, Mercedes Valdes, and Antonio Padilla (2001). “METALA: A Meta-learning Architecture”. In: *Proceedings of the International Conference, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*. London, UK: Springer-Verlag, pp. 688–698.

Box, George and Gwilym Jenkins (1970). “Time Series Analysis”. In:

Brazdil, Pavel, Christophe Giraud-Carrier, Carlos Soares, and Ricardo Vilalta (2008). *Metalearning: Applications to Data Mining.* 1st ed. Springer Publishing Company, Incorporated.

Brazdil, Pavel B., Carlos Soares, and Joaquim Pinto Da Costa (Mar. 2003). “Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results”. In: *Journal of Machine Learning* 50.3, pp. 251–277.

Budka, Marcin, Mark Eastwood, Bogdan Gabrys, Petr Kadlec, Manuel Martin Salvador, Stephanie Schwan, Athanasios Tsakonas, and Indrė Žliobaitė (2014). “From sensor readings to predictions: On the process of developing practical soft sensors”. In: *International Symposium on Intelligent Data Analysis*. Springer, Cham, pp. 49–60.

Budka, Marcin and Bogdan Gabrys (2010a). “Correntropy-based density-preserving data sampling as an alternative to standard cross-validation”. In: *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, pp. 1–8.

Budka, Marcin and Bogdan Gabrys (2010b). “Ridge regression ensemble for toxicity prediction”. In: *Procedia Computer Science* 1.1, pp. 193–201.

Budka, Marcin and Bogdan Gabrys (2013). “Density-Preserving Sampling: Robust and Efficient Alternative to Cross-Validation for Error Estimation”. In: *IEEE Transactions on Neural Networks and Learning Systems* 24.1, pp. 22–34.

Budka, Marcin, Bogdan Gabrys, and Katarzyna Musial (2011). “On accuracy of PDF divergence estimators and their applicability to representative data sampling”. In: *Entropy* 13.7, pp. 1229–1266.

Chawla, Nitesh V., Kevin Bowyer, Lawrence Hall, and Philip Kegelmeyer (2002). “SMOTE: Synthetic Minority Over-sampling Technique”. In: *Journal of Artificial Intelligence Research* 16, pp. 321–357.

Cressie, N. A. C. (1993). “Statistics for spatial data”. In:

Crone, Sven (2006). *NN3 Forecasting Competition [Online].* URL: `http://www.neural-forecasting-competition.com/NN3`.

Crone, Sven (2008). *NN5 Forecasting Competition [Online].* URL: `http://www.neural-forecasting-competition.com/NN5`.

Crone, Sven (2010). *NN-GC1 Forecasting Competition [Online].* URL: `http://www.neural-forecasting-competition.com`.

De-Souto, Marcilio, Ricardo Bastos Cavalcante Prudencio, Rodrigo Soares, and et al. (2008). "Ranking and selecting clustering algorithms using a meta-learning approach". In: *IEEE International Joint Conference on Neural Networks*, pp. 3729–3735.

Deng, Jia, Wei Dong, Richard Socher, Li-jia Li, and et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *In CVPR.*

Duch, Wlodzislaw, Tomasz Maszczyk, and Marek Grochowski (2011). "Optimal Support Features for Meta-Learning". In: *Meta-Learning in Computational Intelligence.* Vol. 358. Studies in Computational Intelligence. Springer, pp. 317–358.

Duda, R. O. and P. E. Hart (1973). "Pattern Classification and Scene Analysis". In:

Eastwood, Mark and Bogdan Gabrys (2012). "Generalised bottom-up pruning: A model level combination of decision trees". In: *Expert Systems with Applications* 39.10, pp. 9150–9158.

eLICO (2012). *An e-Laboratory for Interdisciplinary Collaborative Research in Data Mining and Data-Intensive Sciences.* URL: `http://www.e-lico.eu`.

Fei-Fei, Li, Rob Fergus, and Pietro Perona (Apr. 2007). "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories". In: *Comput. Vis. Image Underst.* 106.1, pp. 59–70. ISSN: 1077-3142.

Feurer, Matthias, Jost Tobias, and Frank Hutter (2014). "Using meta-learning to initialize bayesian optimization of hyperparameters". In: *Proceedings of the International Conference on Meta-learning and Algorithm Selection (MLAS)*, pp. 3–10.

Frank, Eibe and Ian H. Witten (1998). "Generating Accurate Rule Sets Without Global Optimization". In: *Fifteenth International Conference on Machine Learning*, pp. 144–151.

Gabrys, B, K Leiviska, and J Strackeljan (2005). *Do Smart Adaptive Systems Exist?: Best Practice for Selection and Combination of Intelligent Methods.*

Gabrys, Bogdan (2002). "Combining neuro-fuzzy classifiers for improved generalisation and reliability". In: *Proc. the Int. Joint Conference on Neural Networks (IJCNN'2002), Honolulu, USA.* IEEE, pp. 2410–2415.

Gabrys, Bogdan (2004). "Learning hybrid neuro-fuzzy classifier models from data: to combine or not to combine?" In: *Fuzzy Sets and Systems* 147.1, pp. 39–56.

Gabrys, Bogdan and Andrzej Bargiela (1999). "Neural networks based decision support in presence of uncertainties". In: *Journal of Water Resources Planning and Management* 125.5, pp. 272–280.

Gabrys, Bogdan and Dymitr Ruta (2006). "Genetic algorithms in classifier fusion". In: *Applied soft computing* 6.4, pp. 337–347.

Gama, J. and P. Brazdil (1995). "Characterization of Classification Algorithms". In:

Gama, João, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia (2014). "A Survey on Concept Drift Adaptation". In: *ACM Comput. Surv.* 46.4.

Gama, Joao, Pedro Medas, Gladys Castillo, and Pedro Rodrigues (Sept. 2004). “Learning with Drift Detection”. In: *Advances in Artificial Intelligence* 3171.10, pp. 286–295.

Genuer, Robin, Jean-Michel Poggi, and Christine Tuleau-Malot (2010). “Variable Selection Using Random Forests”. In: *Pattern Recogn. Lett.* 31.14, pp. 2225–2236.

Giraud-Carrier, Christophe (2005). “The Data Mining Advisor: Meta-learning at the Service of Practitioners”. In: *Proceedings of the Fourth International Conference on Machine Learning and Applications (ICMLA).* Washington, DC, USA: IEEE Computer Society, pp. 113–119.

Giraud-Carrier, Christophe (2008). “Meta-learning - A Tutorial”. In: *Proceedings of the Seventh International Conference on Machine Learning and Applications (ICMLA).* San Diego, CA, USA.

Giraud-Carrier, Christophe, Ricardo Vilalta, and Pavel Brazdil (Mar. 2004). “Introduction to the Special Issue on Meta-Learning”. In: *Journal of Machine Learning* 54.3, pp. 187–193.

Gittins, John (1979). “Bandit processes and dynamic allocation indices”. In: *Series B (Methodological)* abs/1607.00215.

Glorot, Xavier and Yoshua Bengio (2010). “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, PMLR.*

Goldberg, David E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* 1st. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0201157675.

Grabczewski, Krzysztof and Norbert Jankowski (2007). “Versatile and Efficient Meta-Learning Architecture: Knowledge Representation and Management in Computational Intelligence”. In: *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 51–58.

Graner, Nicolas, Sunil Sharma, Derek H. Sleeman, and et al. (1994). “The Machine Learning Toolbox Consultant”. In: *International Journal on Artificial Intelligence Tools* 2.3, pp. 307–328.

Grochowski, M, W Duch, and et al. (2008). “Projection Pursuit Constructive Neural Networks-Based on Quality of Projected Clusters”. In: *Lecture Notes in Computer Science*, pp. 754–762.

Guerra, Silvio B., Ricardo B. Prudencio, and Teresa Ludermir (Sept. 2008). “Predicting the Performance of Learning Algorithms Using Support Vector Machines as Meta-regressors”. In: *Proceedings of the 18th international conference on Artificial Neural Networks (ICANN).* Berlin, Heidelberg: Springer-Verlag, pp. 523–532.

Hem, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2014). “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *Computing Research Repository (CoRR)* abs/1406.4729.

Hinton, Geff, N. Srivastava, and Swersky K. (2014). *Overview of mini-batch gradient descent lecture of Neural Networks for Machine Learning course.* URL: `http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture\_slides\_lec6.pdf`.

Hochreiter, Sepp and Jurgen Schmidhuber (1997). “Long short-term memory”. In: *Neural Computation*, pp. 1735–1780.

Hutter, F., H. Hoos, and K Leyton-Brown (2011). “Sequential model-based optimization for general algorithm configuration”. In: *Proceedings of the Fifth International Conference on Learning and Intelligent Optimization (LION).*

Hyndman, Rob J. and Yeasmin Kh (2008). “Automatic time series forecasting: The forecast package for R”. In: *Journal of Statistical Software.*

Jankowski, Norbert and Krzysztof Grabczewski (2011). “Universal Meta-Learning Architecture and Algorithms”. In: *Meta-Learning in Computational Intelligence.* Ed. by Norbert Jankowski, Wlodzislaw Duch, and Krzysztof Grabczewski. Vol. 358. Studies in Computational Intelligence. Springer, pp. 1–76.

Kadlec, Petr and Bogdan Gabrys (2008a). “Adaptive local learning soft sensor for inferential control support”. In: *Computational Intelligence for Modelling Control & Automation, 2008 International Conference on.* IEEE, pp. 243–248.

Kadlec, Petr and Bogdan Gabrys (2008b). “Gating Artificial Neural Network Based Soft Sensor”. In: *New Challenges in Applied Intelligence Technologies.* Springer, Berlin, Heidelberg, pp. 193–202.

Kadlec, Petr and Bogdan Gabrys (2008c). “Learnt topology gating artificial neural networks”. In: *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on.* IEEE, pp. 2604–2611.

Kadlec, Petr and Bogdan Gabrys (2008d). “Soft sensor based on adaptive local learning”. In: *International Conference on Neural Information Processing.* Springer, Berlin, Heidelberg, pp. 1172–1179.

Kadlec, Petr and Bogdan Gabrys (2009a). “Architecture for development of adaptive on-line prediction models”. In: *Memetic Computing* 1.4, p. 241.

Kadlec, Petr and Bogdan Gabrys (2009b). “Evolving on-line prediction model dealing with industrial data sets”. In: *Evolving and Self-Developing Intelligent Systems, 2009. ESDIS’09. IEEE Workshop on.* IEEE, pp. 24–31.

Kadlec, Petr and Bogdan Gabrys (2009c). “Soft sensors: where are we and what are the current and future challenges?” In: *IFAC Proceedings Volumes* 42.19, pp. 572–577.

Kadlec, Petr and Bogdan Gabrys (2010). “Adaptive on-line prediction soft sensing without historical data”. In: *Neural Networks (IJCNN), the 2010 International Joint Conference on.* IEEE, pp. 1–8.

Kadlec, Petr and Bogdan Gabrys (2011). “Local learning-based adaptive soft sensor for catalyst activation prediction”. In: *AIChE Journal* 57.5, pp. 1288–1301.

Kadlec, Petr, Bogdan Gabrys, and Sibylle Strandt (2009). “Data-driven soft sensors in the process industry”. In: *Computers & chemical engineering* 33.4, pp. 795–814.

Kadlec, Petr, Ratko Grbić, and Bogdan Gabrys (2011). “Review of adaptation mechanisms for data-driven soft sensors”. In: *Computers & Chemical Engineering* 35.1, pp. 1–24.

Kalousis A., Hilario M. (2001). “Model selection via meta-learning: a comparative study”. In: *Internation Journal on Artificial Intelligence Tools* 10.4, pp. 525–554.

King, Ross (1995). *Statlog Project Data Set.* URL: `http://mlr.cs.umass.edu/ml/datasets/Statlog+Project`.

King, Ross, C Feng, and Alistair Sutherland (1995). “StatLog: Comparison of Classification Algorithms on Large Real-World Problems”. In: *Journal of Applied Artificial Intelligence* 9.3, pp. 289–334.

Klinkenberg, Ralf (Oct. 2005). “Meta-Learning, Model Selection, and Example Selection in Machine Learning Domains with Concept Drift”. In: *Annual workshop of the special interest group* *on machine learning, knowledge discovery, and data mining of the German Computer Science Society (GI).* Saarbrucken, Germany.

Komer, Brent, James Bergstra, and Chris Eliasmith (2014). “Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn”. In: *ICML workshop on AutoML.*

Kopf, Christian and Ioannis Iglezakis (Aug. 2002). “Combination of Task Description Strategies and Case Base Properties for Meta-learning”. In: *Proceedings of the 2nd International Workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-learning (IDDM).* Helsinki, Finland, pp. 65–76.

Kordik, Pavel and Jan Cerny (Dec. 2014). “Building predictive models in two stages with meta-learning templates optimized by genetic programming”. In: *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning, CIEL*, pp. 27–34.

Kovarik, Oleg and Richard Malek (2012). *Meta-learning and Meta-optimization.* Tech. rep. Prague, Czech Republic: Technical Report, Czech Technical University.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS).* Vol. 1. USA: Curran Associates Inc., pp. 1097–1105.

LeCun, Yann, Corinna Cortes, and Christopher J. C. Burges (1999). “The MNIST Dataset Of Handwritten Digits”. In:

Lemke, Christiane, Marcin Budka, and Bogdan Gabrys (2013a). “Metalearning: a survey of trends and technologies”. In: *Artificial Intelligence Review.* URL: `https://doi.org/10.1007/s10462-013-9406-y`.

Lemke, Christiane and Bogdan Gabrys (2008). “Do we need experts for time series forecasting?” In: *Proc. of the 16th European Symposium on Artificial Neural Networks (ESANN’2008), Bruges, Belgium.* ESANN.

Lemke, Christiane and Bogdan Gabrys (June 2010a). “Meta-learning for time series forecasting and forecast combination”. In: *Neurocomputing* 73.10-12, pp. 2006–2016.

Lemke, Christiane and Bogdan Gabrys (July 2010b). “Meta-learning for time series forecasting in the NN GC1 competition”. In: *Fuzzy Systems (FUZZ)*, pp. 1–5.

Lemke, Christiane, Silvia Riedel, and Bogdan Gabrys (2009). “Dynamic combination of forecasts generated by diversification procedures applied to forecasting of airline cancellations”. In: *Proc. of the IEEE Symposium Series on Computational Intelligence 2009.* IEEE.

Lemke, Christiane, Silvia Riedel, and Bogdan Gabrys (2013b). “Evolving forecast combination structures for airline revenue management”. In: *Journal of Revenue and Pricing Management* 12.3, pp. 221–234.

Li, S. Z. (1995). “Markov Random Field Modeling in Computer Vision”. In:

Lin, Tsung-Yi, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, and et al. (2014). “Microsoft COCO: Common Objects in Context”. In: *Computing Research Repository (CoRR)* abs/1405.0312.

Lindner, Guido and Rudi Studer (1999). “AST: Support for Algorithm Selection with a CBR Approach”. In: *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery (PKDD).* London, UK, UK: Springer-Verlag, pp. 418–423.

Maszczyk, Tomasz, Marek Grochowski, and Wlodzislaw Duch, eds. (2010). *Advances in Machine Learning II*. Vol. 263. Studies in Computational Intelligence. Springer.

Menahem, Eitan, Lior Rokach, and Yuval Elovici (2011). "Combining One-Class Classifiers via Meta-Learning". In: *Computing Research Repository (CoRR)* abs/1112.5246.

Mierswa, Ingo, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler (Aug. 2006). "YALE: Rapid Prototyping for Complex Data Mining Tasks". In: *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. Ed. by Lyle Ungar, Mark Craven, Dimitrios Gunopulos, and Tina Eliassi-Rad. New York, NY, USA: ACM, pp. 935–940.

Morik, Katharina and Martin Scholz (2003). "The MiningMart Approach to Knowledge Discovery in Databases". In: *In Ning Zhong and Jiming Liu, editors, Intelligent Technologies for Information Analysis*. Springer, pp. 47–65.

Movielens (1998). *MovieLens Data Sets*. URL: `http://grouplens.org/node/12`.

Murtagh, Fionn and Pierre Legendre (2014). "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?" In: *Journal of Classification* 31.3, pp. 274–295.

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified linear units improve restricted Boltzmann machines". In: *International Conference of Machine Learning (ICML)*.

Nilsback, Maria-Elena and Andrew Zisserman (2008). "Automated Flower Classification over a Large Number of Classes". In: *2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing*, pp. 722–729.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python ". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Peng, Yonghong, Peter A. Flach, Carlos Soares, and Pavel Brazdil (2002). "Improved Dataset Characterisation for Meta-learning". In: *Proceedings of the 5th International Conference on Discovery Science (DS)*. London, UK: Springer-Verlag, pp. 141–152.

Petris, Giovanni and Sonia Petrone (2011). "State Space Models in R". In: *Journal of Statistical Software* 41 (4).

Pfahringer, Bernhard, Hilan Bensusan, and Christophe Giraud-Carrier (2000). *Meta-learning by landmarking various learning algorithms*. Tech. rep. Bristol, UK: University of Bristol.

Prudencio, Ricardo, Marcilio deSouto, and Teresa Ludermir (2011). *Selecting Machine Learning Algorithms Using the Ranking Meta-Learning Approach*. Meta-Learning in Computational Intelligence, Studies in Computational Intelligence. Springer Berlin Heidelberg.

Prudencio, Ricardo B. C. and Teresa B. Ludermir (2004). "Meta-learning approaches to selecting time series models". In: *Journal of Neurocomputing* 61, pp. 121–137.

Prudencio, Ricardo B. C. and Teresa B. Ludermir (June 2008). "Active Meta-Learning with Uncertainty Sampling and Outlier Detection". In: *IEEE International Joint Conference on Neural Networks*, pp. 346–351.

Quinlan, John R. (1992). "Learning With Continuous Classes". In: *AI'92 (Adams and Sterling, Eds)*. Singapore: World Scientific, pp. 343–348.

Quinlan, John R. (1998). *C5.0: An Informal Tutorial*. URL: `http://www.rulequest.com/see5-unix.html`.

R, Core Development Team (2010). *R: A Language and Environment for Statistical Computing.* Vienna, Austria. URL: `http://www.r-project.org/`.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing.* Vienna, Austria. URL: `http://www.R-project.org/`.

Reif, Matthias (Feb. 2012). "A Comprehensive Dataset for Evaluating Approaches of various Meta-Learning Tasks". In: *First International Conference on Pattern Recognition Applications and Methods*. Vilamura, Algarce, Portugal.

Reif, Matthias, Faisal Shafait, and Andreas Dengel (2012a). "Dataset Generation for Meta-Learning". In: *KI-2012: Poster and Demo Track*. Saarbrucken, pp. 69–73.

Reif, Matthias, Faisal Shafait, and Andreas Dengel (2012b). "Meta2-Features: Providing Meta-Learners More Information". In: *KI-2012: Poster and Demo Track*. Saarbrucken, pp. 74–77.

Rendell, Larry and Howard Cho (Sept. 1990). "Empirical Learning as a Function of Concept Character". In: *Journal of Machine Learning* 5.3, pp. 267–298. ISSN: 0885-6125.

Rendell, Larry, Raj Sheshu, and David Tcheng (1987). "Layered concept-learning and dynamically variable bias management". In: *Proceedings of the 10th international joint conference on Artificial intelligence (IJCAI)*. Vol. 1. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 308–314.

Riedel, Silvia and Bogdan Gabrys (2005a). "Evolving multilevel forecast combination models-an experimental study". In: *Proc. of the Nature-Inspired Smart Information Systems Symposium (NiSIS'2005), 4 - 5 October 2005, Albufeira, Portugal.* NiSIS.

Riedel, Silvia and Bogdan Gabrys (2005b). "Hierarchical multilevel approaches of forecast combination". In: *Operations Research Proceedings 2004*. Springer, Berlin, Heidelberg, pp. 479–486.

Riedel, Silvia and Bogdan Gabrys (2007a). "Combination of multi level forecasts". In: *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology* 49.2, pp. 265–280.

Riedel, Silvia and Bogdan Gabrys (2007b). "Dynamic pooling for the combination of forecasts generated using multi level learning". In: *Neural Networks, 2007. IJCNN 2007. International Joint Conference on.* IEEE, pp. 454–459.

Riedel, Silvia and Bogdan Gabrys (2009). "Pooling for combination of multilevel forecasts". In: *IEEE Transactions on Knowledge and Data Engineering* 21.12, pp. 1753–1766.

Rijn, J. N. van, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, and et al. (2013). "OpenML: a collaborative science platform". In: *in Proceedings of ECML/PKDD'13.*

Rossi, Andre Luis Debiaso, Andre Carlos Ponce de Leon Ferreira de Carvalho, and et al. (2014). "MetaStream: A meta-learning based method for periodic algorithm selection in time-changing data". In: *Journal of Neurocomputing* 127, 52–64.

Rossi, Andre Luis Debiaso, Andre Carlos Ponce de Leon Ferreira de Carvalho, and Carlos Soares (2012). "Meta-Learning for Periodic Algorithm Selection in Time-Changing Data". In: *Brazilian Symposium on Neural Networks*, pp. 7–12.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, and et al. (2014). *ImageNet Large Scale Visual Recognition Challenge.* URL: `http://arxiv.org/abs/1409.0575`.

Ruta, Dymitr and Bogdan Gabrys (2002). "A theoretical analysis of the limits of majority voting errors for multiple classifier systems". In: *Pattern Analysis and Applications* 5.4, pp. 333–350.

Ruta, Dymitr and Bogdan Gabrys (2005). “Classifier selection for majority voting”. In: *Information fusion* 6.1, pp. 63–81.

Ruta, Dymitr and Bogdan Gabrys (2007). “Neural network ensembles for time series prediction”. In: *Neural Networks, 2007. IJCNN 2007. International Joint Conference on.* IEEE, pp. 1204–1209.

Ruta, Dymitr, Bogdan Gabrys, and Christiane Lemke (2011). “A generic multilevel architecture for time series prediction”. In: *IEEE Transactions on Knowledge and Data Engineering* 23.3, pp. 350–359.

Sahel, Zoheir, Abdelhamid Bouchachia, Bogdan Gabrys, and Paul Rogers (2007). “Adaptive mechanisms for classification problems with drifting data”. In: *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems.* Springer, Berlin, Heidelberg, pp. 419–426.

Salvador, Manuel Martin, Bogdan Gabrys, and Indrė Žliobaitė (2014). “Online detection of shutdown periods in chemical plants: A case study”. In: *Procedia Computer Science* 35, pp. 580–588.

Shah, Chandra (1997). “Model selection in univariate time series forecasting using discriminant analysis”. In: *International Journal of Forecasting* 13.4, pp. 489–500.

Sikora, Riyaz T. (Dec. 2008). “Meta-learning optimal parameter values in non-stationary environments”. In: *Journal of Knowledge-Based Systems* 21.8, pp. 800–806.

Smith-Miles, Kate (Dec. 2009). “Cross-disciplinary perspectives on meta-learning for algorithm selection”. In: *ACM Computing Surveys* 41.1, pp. 1–25.

Soares, Carlos (Apr. 2009). “UCI++: Improved Support for Algorithm Selection Using Datasetoids”. In: *Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD).* Berlin, Heidelberg: Springer-Verlag, pp. 499–506.

Soares, Carlos and Pavel B. Brazdil (2006). “Selecting Parameters of SVM Using Meta-learning and Kernel Matrix-based Meta-features”. In: *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC).* New York, NY, USA: ACM, pp. 564–568.

Soares, Carlos, Petr Kuba, and Peter Flach (2004). “A meta-learning method to select the kernel width in support vector regression”. In: *Mach. Learning*, pp. 195–209.

Soares, Carlos, Johann Petrak, and Pavel Brazdil (2001). “Sampling-Based Relative Landmarks: Systematically Test-Driving Algorithms Before Choosing”. In: *Proceedings of the 10th Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Knowledge Extraction, Multi-agent Systems, Logic Programming and Constraint Solving.* London, UK: Springer-Verlag, pp. 88–95.

Soares, Rodrigo G., Teresa B. Ludermir, and Francisco A. Carvalho (Sept. 2009). “An Analysis of Meta-learning Techniques for Ranking Clustering Algorithms Applied to Artificial Data”. In: *Proceedings of the 19th International Conference on Artificial Neural Networks (ICANN).* Berlin, Heidelberg: Springer-Verlag, pp. 131–140.

Sohn, So Y. (1999). “Meta analysis of classification algorithms for pattern recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.11, pp. 1137–1144.

Stahl, Frederic, Bogdan Gabrys, Mohamed Medhat Gaber, and Monika Berendsen (2013). “An overview of interactive visual data mining techniques for knowledge discovery”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 3.4, pp. 239–256.

Sutskever, Ilya, James Martens, George Dahl, and Geoffrey E. Hinton (2013). “Practical Network Blocks Design with Q-Learning”. In: *International Conference of Machine Learning (ICML).*

Thornton, Chris, Frank Hutter, Holger Hoos, and Kevin Leyton-Brown (2013). “Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms”. In: *ACM International Conference on Knowledge Discovery and Data Mining*, pp. 847–855.

Todorovski, Ljupco, Hendrik Blockeel, and Saso Dzeroski (2002). “Ranking with Predictive Clustering Trees”. In: *Proceedings of the 13th European Conference on Machine Learning (ECML).* London, UK: Springer-Verlag, pp. 444–455.

Todorovski, Ljupvco and Savso Dvzeroski (Mar. 2003). “Combining Classifiers with Meta Decision Trees”. In: *Journal of Machine Learning* 50.3, pp. 223–249.

Tsakonas, Athanasios and Bogdan Gabrys (2011). “Evolving Takagi-Sugeno-Kang fuzzy systems using multi-population grammar guided genetic programming”. In: *Proc. of the International Conference on Evolutionary Computation Theory and Applications (ECTA'11).* INSTICC.

Tsakonas, Athanasios and Bogdan Gabrys (2012). “GRADIENT: Grammar-driven genetic programming framework for building multi-component, hierarchical predictive systems”. In: *Expert Systems with Applications* 39.18, pp. 13253–13266.

Tsakonas, Athanasios and Bogdan Gabrys (2013). “A fuzzy evolutionary framework for combining ensembles”. In: *Applied Soft Computing* 13.4, pp. 1800–1812.

Tusell, Fernando (2011). “Kalman Filtering in R”. In: *Journal of Statistical Software* 39 (2).

Utgoff, Paul Everett (1984). “Shift of bias for inductive concept learning”. PhD thesis. New Brunswick, NJ, USA: Rutgers University.

Vanschoren, Joaquin (2011). “Meta-Learning Architectures: Collecting, Organizing and Exploiting Meta-Knowledge”. In: *Meta-Learning in Computational Intelligence.* Ed. by Norbert Jankowski, WlMeta-learningodzislaw Duch, and Krzysztof Grabczewski. Vol. 358. Studies in Computational Intelligence. Springer, pp. 117–155.

Vanschoren, Joaquin, Jan N. Rijn, Bernd Bischl, and Luis Torgo (June 2014). “OpenML: Networked Science in Machine Learning”. In: *SIGKDD Explor. Newsl.* 15.2, pp. 49–60.

Vilalta, Ricardo and Youssef Drissi (Oct. 2002a). “A perspective view and survey of meta-learning”. In: *Artificial Intelligence Review* 18.2, pp. 77–95.

Vilalta, Ricardo and Youssef Drissi (2002b). “A Perspective View and Survey of Meta-learning”. In: *Artif. Intell. Rev.* 18.2, pp. 77–95. ISSN: 0269-2821.

Vilalta, Ricardo, Christophe Giraud-Carrier, and Pavel Brazdil (2010). *Meta-Learning - Concepts and Techniques.* Data Mining and Knowledge Discovery Handbook. US: Springer.

Vilalta, Ricardo, Christophe Giraud-carrier, Pavel Brazdil, and Carlos Soares (2004). “Using Meta-Learning to Support Data Mining”. In:

Wang, Xiaozhe, Kate Smith-Miles, and Rob Hyndman (June 2009). “Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series”. In: *Journal of Neurocomputing* 72.10-12, pp. 2581–2594.

Warden, Pete (2011). *Data Source Handbook - A Guide to Public Data.* OŔeilly Media.

Widmer, Gerhard (June 1997). “Tracking Context Changes through Meta-Learning”. In: *Journal of Machine Learning* 27.3, pp. 259–286.

Williams, Ronald J. (1992). “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Machine Learning* (9), pp. 41–49.

Wolpert, David (2001). “The supervised learning no-free-lunch Theorems”. In: *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications*, pp. 25–42.

Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson (2014). “How Transferable Are Features in Deep Neural Networks?” In: *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*. Vol. 2. Cambridge, MA, USA: MIT Press, pp. 3320–3328.

Zliobaite, Indre (2010). “Learning under Concept Drift: An Overview”. In: *Computing Research Repository (CoRR)* abs/1010.4784.

Zliobaite, Indre, Albert Bifet, Mohamed Gaber, Bogdan Gabrys, Joao Gama, Leandro Minku, and Katarzyna Musial (2012). “Next challenges for adaptive learning systems”. In: *ACM SIGKDD Explorations Newsletter* 14.1, pp. 48–55.

Zliobaite, Indre and Bogdan Gabrys (2014). “Adaptive preprocessing for streaming data”. In: *IEEE Transactions on Knowledge and Data Engineering* 26.2, pp. 309–321.