Abstract

Pain assessment has remained largely unchanged for decades and is currently mainly based on self-reports, which are the current gold standard. Unfortunately, despite different versions for various applications, these have significant drawbacks. Self-reports are based solely on patients' subjective assessment of their pain and are therefore influenced by personal experience and are highly subjective, leading to uncertainty in ratings, difficult comparability, and significant variance between individuals, as well as discrepancies from the truth. In addition, accurate pain assessment remains time-consuming, while continuous measurement is almost impossible in clinical settings. Finally, the method is limited to people who are conscious and able to express their own level of pain.

To overcome the above-mentioned disadvantages of self-report, automated pain detection aims to provide a continuous, objective, and easy-to-acquire measure of pain in subjects. The basic idea is to train learning algorithms on sensory data related to pain intensity so that they can later make a prediction on unseen data. The successful implementation of such a model would make it possible to objectively, reliably, and continuously assess the intensity of pain in human subjects.

While the quest to create an automated pain monitoring system has led to various implementations of this idea, models have lacked several critical aspects for successful use in clinical settings. Although of paramount importance for medical applications, the performance of existing automated pain detection systems is currently inadequate. In addition, the different approaches to pre-processing, segmentation, feature extraction, classification and scoring that have been implemented in the past obscure the reasons for performance improvements of individual changes and make it challenging to identify suggestions for improvement through new approaches. Furthermore, the data acquisition, and therefore the systems, are based on recording devices with fixed settings, which makes their use in dynamic clinical practice almost impossible. Finally, the research community has focused on improving the learning algorithms for automatic pain detection, particularly in terms of

Chapter 1

Introduction

The recent development of Information and Communications Technology (ICT) affects our life in various aspects. While the extraction of knowledge from vast amounts of data in practical application becomes cheaper and more accessible, sectors of the economy, such as the automotive industry [1] among others, profit greatly thanks to the sensor data and automation of data analytics. As the number of affordable devices to collect sensor data, such as Inertial Measurement Units (IMUs) and smartphones, rose, the volume of available data for diverse use cases heightened as well. Numerous approaches to data analysis have appeared to extract useful information. One current trend to manage and gain knowledge from these newly obtained and extensive data streams is to apply Artificial Intelligence (AI). In particular, the idea of training computers and letting them *learn* specific tasks given a defined data set accelerated. Like the human learning process, Machine Learning (ML) algorithms aim to improve system performance based on experience. By fitting models on training data without incorporating explicit decision rules during the implementation process, these learning algorithms are able to generate predictions on new and unseen observations afterwards [2].

This trend was further fuelled by the recent success of Deep Learning (DL), which achieved outstanding performance results for various tasks. Here, so-called Artificial Neural Network (ANN) consisting of several layers with up to billions of parameters automatise the learning process. While the concepts of ANN were researched during the last century [3], their success emerged in the last years due to the availability of increased computational power and enormous data sets. Gaining momentum from challenges including millions of images, like the *ImageNet large scale visual recognition challenge* [4], DL tasks including image classification and object localisation and detection excelled primarily. The performance of these systems was incrementally improved

by the research community to the point of even outperforming humans in specific tasks [5].

With the goal of continuously improving patient care, innovations through DL and digitisation, in general, have also found their way into the healthcare sector. Today, a wide variety of electronic data processing has become part of everyday clinical practice. For example, Electrocardiograms (ECGs) are one of the standard monitoring devices to oversee cardiac activity in Intensive Care Unit (ICU) patients [6] and imaging techniques, such as Computer Tomography (CT) and Magnetic Resonance Imaging (MRI), are used for detection of cancer [7], stroke [8], inflammation [9], fractures [10], and various other disorders. In addition to technical devices as monitoring tools, intelligent systems have been developed to support clinical decisions, train inexperienced staff, and even enhance the knowledge of medical experts in the field. Various DL models have been researched and implemented, such as for the classification of skin cancer [11] based on colour images of the skin, of COVID-19 patients based on chest CT images [12] or of Parkinson's disease based on audio files [13].

While different medical applications, especially imaging techniques, could benefit from the digitalisation of the health sector, various areas still remain untouched and even outdated. Although accurate pain assessment is essential for various medical applications, its gold standard has remained unchanged over the past decades. Usually, pain is measured via verbal feedback from the patients utilising diverse questionnaires. While this approach may be sufficient in some cases, these questionnaires have significant drawbacks. On the one side, the questionnaires rely on subjects who can communicate their subjective level of pain. On the other hand, the measurement is highly subjective, as it originates from the persons themselves. Moreover, continuous monitoring of pain levels in people remains an obstacle.

To address these issues, automated pain recognition systems emerged in the past years. Their main goal is to provide a reliable, continuous, and objective pain measurement based on unobtrusive collected data, either behavioural or physiological. To accomplish this task, varying ML approaches based on heterogeneous data sets were introduced by the research community in the past. While it could be shown that automated approaches can distinguish different pain levels in general, particularly between no and high pain, the established models focus on single modalities, persist in being incomparable, or lack the required accuracy and an explanation of the underlying decision process.

The purpose of this thesis is to address the aforementioned limitations of existing methods and improve the performance of automated pain classification. The focus is on the following research questions:

- Which modalities are essential for the task of automated pain classification?
  Narrowed to physiological sensor modalities, a comparison of common metrics is given and evaluated to find an optimal set of sensors for automated pain classification. Moreover, a minimal setup of wearable devices is examined to ensure an unobtrusive configuration that can be used in medical situations and daily living.

- Are DL techniques, similar to other ML tasks, actually better than traditional approaches in the area of automatic pain detection?
  DL achieved outstanding performance results for various classification problems. Their value for automated pain recognition still needs to be evaluated. Therefore, a fair comparison of classical ML and DL for the task of automated pain recognition is given.

- Which underlying features are important for the pain recognition task?
  ANNs, which have delivered amazing results, mostly consist of black-box models, so their decision-making process is neither comprehensible nor transparent. Therefore, a deeper understanding of the decision-making processes is not possible, which limits their use in medical contexts. To counteract this, methods of Explainable Artificial Intelligence (XAI) have been implemented to make the classification of the underlying models apparent.

The remainder of the Section is structured as follows: Fundamental concepts of time series data analysis and key characteristics of pain, such as its monitoring and physiological aspects, are given in Sections 1.1 and 1.2. Section 1.3 explains the motivation of the thesis in more detail. A brief overview of the contribution of this work is given in Section 1.4. Eventually, Section 1.5 summarises the structure of the thesis.

1.1 Time Series Analysis

Driven by digitalisation, the *Internet of Things* and the introduction of smart devices into our everyday lives, more and more time series data are available. These data sources and volumes are expected to continue to grow over the next few years, increasing the importance of high-quality analysis of these

data. Following this trend, the demand for competent Time Series Analysis (TSA) using statistical and machine learning techniques will grow as continuous monitoring and data collection become more common [14]. In particular, the introduction of cheap and small, yet research-grade, devices has enabled data collection to move out of the research lab and into people's everyday lives, where they have a natural perspective on the things they are trying to capture [15]. Computing devices worn on the body, so-called *wearables*, remain non-intrusive and blend into daily living, often generating additional amounts of time series data. However, without the fixed setup provided by studies and experiments, data collected in everyday life can be diverse, heterogeneous, and affected by artefacts introduced by motion, electrical interference, and many other confounding influences. This effect further strengthens the need for accurate and precise analysis of such data.

TSA is a specific way of analysing a sequence of data points collected over an interval of time. More precisely, the term TSA refers to the extraction of meaningful statistics, information, and characteristics from data points arranged in chronological order. Essential components of TSA are the detection of stationarity, trends, and seasonality, and it involves both exploratory and descriptive analysis, i.e. aggregating the characteristics of a given data set and analysing for patterns, trends, or relationships between variables [16]. Its deeper implementation helps to understand and decipher past relationships and to predict future trends. Representations of how variables change over time can be generated, and their temporal influence over time can be estimated. From this, additional knowledge is generated, and correlations between data and time series are better understood. To ensure consistency and provide reliable results, a large number of clean data points with limited noise are required. Larger data sets also ensure that the correlations and patterns identified truly reflect the data and are not skewed by outliers.

Time series data consist of multiple quantitative observations measured at different points in time. The *discrete-time data* refer to a single entity or subject. In contrast, data from several individuals at one point in time are called *cross-sectional* and are distinguished from *continuous data* from a single entity but at several points in time, which are called *time series*. Typically, successive data points of time series are equally spaced in time, but this is not always the case, depending on the application. *Panel data*, sometimes referred to as *longitudinal data*, are data that cover not only a single time series but also several units over various points in time [17]. Moreover, *univariate* time series are those where only a single changing variable is given, for example, data of one sensor modality. In contrast, *multivariate* time series are those in which several variables change over time, for example, multiple

sensor modalities at the same time. However, this definition remains imprecise in many aspects. More formally, time series data can be introduced in two separate ways. On the one hand, it can be seen as a mapping from the time domain $\mathcal{T}$ to that of the real numbers [16] and is defined by

$$X : \mathcal{T} \to \mathbb{R}^k \quad , \tag{1.1}$$

where $\mathcal{T} \subseteq \mathbb{R}$ and $k \in \mathbb{N}$. On the other hand, time series can be seen as a stochastic process

$$\{X_t\}_{t \in \mathcal{T}} \quad , \tag{1.2}$$

where X_t represents the value of the random variable X at time point t. It is further distinguished whether $\mathcal{T}$ forms a set of real numbers or integers. The first is called a continuous-time stochastic process, whereas the second represents a stochastic process in discrete time.

There are several tools available for TSA. An increasingly popular method is building machine learning algorithms, which are described in more detail in Section 1.1.1, and in particular the use of DL methods (Section 1.1.2).

1.1.1 Machine Learning

As the name suggests, *machine learning* is an approach to improving system performance based on teaching machines to learn from experience using computer techniques. While human knowledge and expertise are based on assimilated memory and past findings, computers rely on data. Using such data, ML aims to build mathematical algorithms to acquire and train models. During the training process, these learning models are fed with exemplary data and trained for a specific task, for example, by statistically analysing the data. Subsequently, predictions can be made for new and unseen observations [2] to create further insights into new records.

For the successful implementation of *learning algorithms*, several common steps are followed. The fundamental prerequisite is a comprehensive data set. Here, researchers are dependent on publicly available data sets or have to acquire their own data. Data sets usually consist of a limited number of records, so-called *instances* or *samples*, which describe an event or an object with several *attributes*. In order to further prepare the available samples, a preprocessing is carried out, which includes tasks such as data cleaning and noise reduction. If the data are only available as a continuous time series, a segmentation step can be used to cut out separate samples. Meaningful characteristics of the instances, called *features*, are then extracted. The features are then fed into a model that is adapted to a specific task, also called

learning or *training*. While being trained on the given *training data set*, the objective of the learning algorithms is to perform well on later unseen data. This ability, also referred to as *generalisation*, is facilitated by training data that reflect the characteristics of the entire sample space and a model that is not over-trained on the training set. Such *overfitting* of learning algorithms indicates memorisation of the training sample without acquiring knowledge of the underlying concepts and relationships. To evaluate the performance, the model is tested by making *predictions* (also referred to as *inference*) on an independent *testing set*. The underlying rules for the given task are called the *facts* or *ground truth*, which the model tries to approximate [2]. Usually, the *training set* and *testing set* are obtained by splitting the initial given data set at the beginning. The steps described are part of the Pattern Recognition Chain (PRC), which summarises the essential core stages to realise ML models. Formally, a data set $\mathcal{D}$ with $N_\mathcal{D} \in \mathbb{N}^*$ samples is defined as

$$\mathcal{D} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N_\mathcal{D})}\right\} \quad . \tag{1.3}$$

Here, each instance $\mathbf{x}$ represents an input vector in a $N_\mathbf{x}$-dimensional sample space $\mathcal{X}$ as

$$\mathbf{x} = \left[x_1, x_2, \ldots, x_{N_\mathbf{x}}\right]^\intercal \in \mathcal{X} \quad , \tag{1.4}$$

where $N_\mathbf{x} \in \mathbb{N}^*$. Usually, an *outcome* information related to every sample is given as well in order to train an effective prediction model. The associated outcome of a given instance is called a *label* and defined as a vector $\mathbf{y}$ of several outcome values y as

$$\mathbf{y} = \left[y_1, y_2, \ldots, y_{N_\mathbf{y}}\right]^\intercal \in \mathcal{Y} \quad , \tag{1.5}$$

where $N_\mathbf{y} \in \mathbb{N}^*$, $\mathcal{Y}$ is the set of all labels (*label space*), and since there is precisely one label for each sample, it is usually the case that $N_\mathbf{x} = N_\mathbf{y}$. The resulting data label pairs can be written as $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$, where $\mathbf{y}^{(i)}$ is the label of sample $\mathbf{x}^{(i)}$.

ML approaches can be divided into distinct categories, each with its own subdivisions. Generally, a distinction is made between *supervised* and *unsupervised learning*. In supervised learning, the given data sample is mapped to its fact. The ground truth to be achieved must already be available during training as a label that specifies the desired model output for each sample. The trained learning algorithm represents a function f realising one possible mapping from $\mathcal{X}$ to $\mathcal{Y}$ in the form of

$$f : \mathcal{X} \mapsto \mathcal{Y} \tag{1.6}$$

for a given labelled data set, which is revised as

$$\mathcal{D} = \left\{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(N_\mathcal{D})}, \mathbf{y}^{(N_\mathcal{D})})\right\} \quad . \tag{1.7}$$

The retrieved mapping function can be applied to any observation of $\mathcal{X}$ and generate a predicted label $\hat{\mathbf{y}}^{(i)}$ for the sample $\mathbf{x}^{(i)}$ as

$$f\left(\mathbf{x}^{(i)}\right) = \hat{\mathbf{y}}^{(i)} \quad . \tag{1.8}$$

A mismatch or *error* between predicted label $\hat{\mathbf{y}}^{(i)}$ and actual ground truth $\mathbf{y}^{(i)}$ is measured by an *error* or *loss function* $\mathcal{L}(\cdot)$ and highly dependent on the applied learning algorithm, its configuration, and parameters θ. The main objective of supervised learning is to optimise this error and minimise the mismatch of predicted and actual labels during training by estimating ideal parameters $\hat{\theta}$ as

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmin}} \, \mathcal{L}\left(f_\theta(\mathcal{X}), \mathcal{Y}\right) \quad , \tag{1.9}$$

where θ depicts the set of parameters specific to f and Θ is the set of all possible parameter configurations. If the desired fact of each instance is a discrete label, it is called a *classification* problem; if it is continuous, it is referred to as *regression* problem ($y \in \mathbb{R}$) [2]. Classification problems are further distinguished by the number of distinct types of labels, called *classes*. There are *binary classification* problems when there are only two classes, where one label is marked as *positive* and the other one as *negative*, and *multi-class classification* problems when there are several classes. To create a categorical output in classification problems, a distribution of integers is associated with the different labels. For a binary classification, the classes are encoded as a single one or zero for the positive or negative class, respectively ($y \in \{0, 1\}$). Typically, in multi-class classification, the number of elements in the label vector $N_\mathbf{y}$ is set equal to the number of classes available N_C so that each position in the vector represents a class, where zero indicates that the class is absent and a one encodes the presence of the class. The applied ML models for classification tasks are also referred to as *classifiers*. In addition, a higher dimensionality or a single output can be given for the target in regression tasks as well. In unsupervised learning, on the other hand, there is no label information, and a transformation is learned for the patterns in hand without a given target. For example, in *clustering*, the main goal is to categorise the given samples based on their similarities. To do this, observations are grouped into groups or *clusters* in such a way that more similar instances are grouped together. Often, a number of clusters are defined prior to the execution of the algorithm, which has a high impact on the resulting outcome.

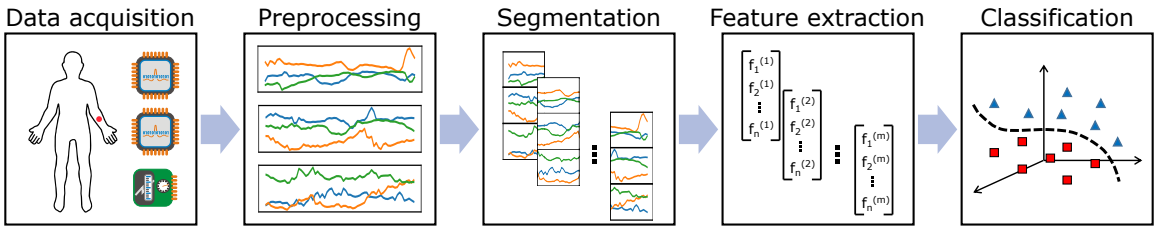

Figure 1.1: PRC with individual steps that should be optimised in parallel to achieve the best performance for the given ML task [20].

But there are also tasks such as dimensionality reduction (e.g. *principal component analysis* [18]) or algorithms such as *Gaussian mixture models* [19] that are categorised as unsupervised learning. There are even approaches in between, called *semi-supervised learning*, which work on data sets with incomplete or missing label spaces. The mixture of supervised and unsupervised approaches aims to complete corrupted or absent labels or even create a label space automatically without human annotation. However, the distinction between supervised, semi-supervised and unsupervised learning can sometimes be blurred as these categories overlap, and some methods cannot be clearly classified. Although the acquisition of labels is often a challenge, most of the time ML tasks are defined as supervised because of their better performance compared to other tasks.

Figure 1.1 visualises the independent steps of the PRC for a classification task using time series data of several sensors. While several measures are performed individually, all of them need to be optimised in parallel to find the optimal model for the given task.

Although it can lead to outstanding results for various tasks, ML faces recurring problems that need to be overcome, where the issues may be dependent on the given task to solve. As mentioned above, a comprehensive and extensive data set is required to learn the desired associations correctly. It is essential that the classes or relationships to be recognised are represented in the data set and sufficient numbers of samples substitute each statement. Besides the quantity of data points, the quality of the samples and the labels also play a crucial role. The task of establishing valid ground truths is often tedious and is further complicated by the time-consuming manual labelling of numerous samples by experts or the quantification of specifications that are difficult to grasp, such as a sensation of pain, even though the task may seem obvious or straightforward to humans. Nevertheless, correct labelling is crucial so ML models can learn the underlying concepts adequately. In addition, varying preprocessing steps were introduced in the past to overcome distortion of the learning algorithms introduced by artefacts and outliers in

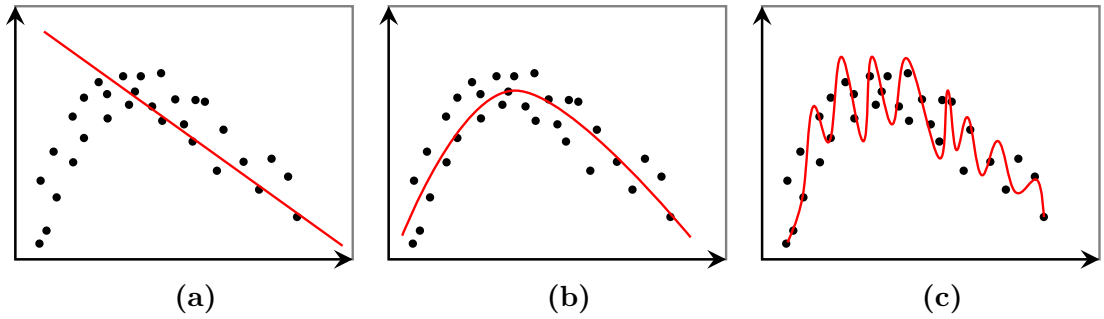

Figure 1.2: Scheme of three different approaches to fit a curve through the same data points. While a linear fit **(a)** cannot adequately represent the data at hand, it suffers from underfitting, a highly adjusted and customised fit **(c)** suffers from overfitting, it does not replicate the underlying structure of the data, although it maps many samples correctly. In contrast, a quadratic function **(b)** reflects the composition of the data and generalises applicable to unseen data points as well.

the available data or an uneven distribution of samples per class, also referred to as *unbalanced* data. During the creation and training of the learning algorithm, its *capacity*, i.e. its ability to adapt to the given training set, must also be monitored. Although accurate learning and a close fit to the training data set are important, the models created should also perform well on the test data set. This generalisation capability is highly dependent on the capacity of the chosen learning model. If it is too high, memorisation of the training data set is possible, resulting in deficient performance on unseen data points that differ from the samples in the training data set (overfitting). In contrast, if it is too low, the model suffers from *underfitting* and is not able to obtain a sufficiently low error value on the training set [21]. Figure 1.2 visualises the underlying problem of overfitting and underfitting, presented for the task of fitting a curve for data samples that seem to follow a quadratic distribution. While a linear fit cannot adequately represent the data at hand, a highly adjusted and customised fit cannot replicate the underlying structure of the data, although it maps many samples correctly. In contrast, a quadratic function reflects the underlying structure of the data and generalises applicable to unseen data points as well.

To estimate the extent of possible underfitting and overfitting, learning algorithms are usually evaluated on different data sets than those on which they were previously trained, as testing on the same data would lead to overly optimistic performance results [22]. Testing the ML models on new unseen data is believed to give a reasonable estimate of its performance in real-life situations. As data is limited in most scenarios, one of the most straightforward approaches is to split the available data set into training and testing

data. As this can lead to random splits that benefit the model's evaluation performance, several Cross Validation (CV) schemes have been introduced. Here, portions of the available data are used to test and train a learning algorithm in different iterations to achieve an overall performance by averaging the accuracies of the individual runs. Different CV-schemes specify in different ways how to exclude a test set, also called *hold-out*, usually with a fixed size for each run. Schemes involving all possible training sets of a given size are referred to as *exhaustive data splitting*, while those that omit some splits are called *partial data splitting*. Generally, CV schemes allow performing several runs of evaluation even when the number of data samples for the given task may be limited, thus reporting more valid performance metrics that have been subject to more testing. Although the methods can be applied independently of the underlying learning algorithm, the strengthened assessment comes at the cost of increased computation time for evaluation. One of the most used approaches is the exhaustive k-fold CV, where the available data is split into k splits, also called *folds*. Each is excluded once as a test set, while the rest is used as training data during the k repeated evaluation runs, each time using a different fold as a test set [23]. While this technique may be appropriate for independent objects, its use is questionable for person-specific data. As models can learn person-specific features rather than the given task, for example, a disease detection, k-fold CV results can be misleading [24] as the models recognise people from training in the test phase (*subject-dependent* split). Therefore, splits are preferred where subjects only appear in the test or training data set (*subject-independent* split). One possible implementation is the exhaustive Leave-one-subject-out (LOSO) CV, which is a variant of the k-fold CV, but folds consists of a single subject. It evaluates the given model on new subjects, where each person is left out and tested upon once [25].

As mentioned earlier, key characteristics are often manually generated from the given observation and fed to the learning algorithms during training rather than the raw data itself. This step of extracting so-called Hand-Crafted Features (HCFs) from the given instances is used to create optimal representations for the considered problem to solve. Ideally, the extracted features present information about the samples themselves that describe complementary content from specific aspects. While the features should fully describe the data points and their key characteristics, they significantly reduce the initial amount of input data, allowing for efficient learning. Meaningful HCFs are especially important to overcome the so-called *semantic gap* between the low-level representation of observations (for example, the pixels of an image) and the high-level abstraction (like the human perception

of the mentioned image) often associated with its ground truth [26]. Quality feature vectors decode high-level properties that are highly relevant to the task at hand, with as few individual features as possible. To do so, traditional feature extraction heavily relied on domain knowledge. Experts usually communicated their reasoning and insights to be implemented as algorithms. Conventional extraction of HCFs is therefore time-consuming and dependent on prior knowledge of the field, highlighting the need for other solutions. One way to avoid this complicated step is given by other methods of AI, especially DL. These concepts are introduced in the next chapter.

1.1.2 Deep Learning

AI has many different subfields and manifestations. One of them is DL, which deals with the development of large neural network models built to make accurate, data-driven decisions by training countless numbers of neuronal parameters to generalise on carrying out a particular task. Large data sets are required for the implementation and correct training. The networks learn immediately from the raw data fed into them using a divide-and-conquer strategy. The basis of the processing is formed by countless small computational units called neurons, which learn simple functions. Complex tasks can then be achieved by combining the individual elements by stacking them together into layers that form complex networks consisting of millions of parameters. These Neural Networks (NNs) have been applied to several fields and achieved outstanding performance, sometimes even outperforming humans for tasks such as image classification. The success of any ML task depends on knowing what to monitor and how to measure it, which makes the task of feature generation and selection such a crucial step in building learning algorithms. Traditional ML approaches rely on domain expertise to solve the challenge of finding meaningful characteristics. Although HCFs can lead to promising results, their identification and feature design, in general, remains a time-intensive activity. DL tries to overcome this issue by attempting to learn essential characteristics for the task at hand automatically. Features are learned directly from low-level raw data, and outputs are generated by a highly complex non-linear mapping that depends on millions of network parameters. To do this, DL relies on massive amounts of example data. But when large data sets are available, the approach often builds highly accurate models for complex domains and has shown great success in the past in tasks such as image classification, speech recognition, and many others [27].