Linnéuniversitetet
Kalmar Växjö

Master Thesis Project

Spot the Pain: Exploring the Application of Skeleton Pose Estimation for Automated Pain Assessment

Author: Angelica Hjelm Gardner
Supervisor: Welf Löwe
Examiner: Narges Khakpour
Reader: Francesco Flammini
Semester: VT 2022
Course Code: 5DV50E
Subject: Computer Science

Abstract

Automated pain assessment is emerging as an essential part of pain management in areas such as healthcare, rehabilitation, sports and fitness. These automated systems are based on machine learning applications and can provide reliable, objective and cost-effective benefits. To enable an automated approach, at least one channel of sensory input, known as modality, must be available to the system. So far, most studies of automated pain assessment have focused on facial expressions or physiological signals, and although body gestures are considered to be indicators of pain, not much attention has been paid to this modality. Using skeleton pose estimation, we can model body gestures and investigate how body movement information affects pain assessment performance in different approaches.

In this study, we explored approaches to pain assessment using skeleton pose estimation for three objectives: pain recognition, pain intensity estimation, and pain area classification. Because pain is a complex experience and is often expressed across multiple modalities, we analysed both unimodal approaches using only body data and bimodal approaches using skeleton pose estimation with facial expressions and head pose. In our experiments, we trained models based on two deep learning architectures: a hybrid CNN-BiLSTM and a recurrent CNN (RCNN), on a real-world dataset consisting of video recordings of people performing an overhead deep squat exercise. We also investigated bimodal fusion of body and face modalities in three different strategies: early fusion, late fusion and ensemble learning. Although our results are still preliminary, they show promising indications and possible future improvements. The best performance was obtained with ensemble for pain recognition (AUC 0.71), unimodal body CNN-BiLSTM for pain intensity estimation (AUC 0.75) and late fusion of body and face modalities using RCNN for pain area classification (AUC 0.75). Our experimental results demonstrate the feasibility of using skeleton pose estimation to represent body modality, the importance of incorporating body movements into automated pain assessment, and the exploration of the previously understudied assessment objective of localising pain areas in the body.

Keywords: Automated pain assessment; pain recognition; body movements; skeleton pose estimation; deep learning; neural networks.

Preface

This master's thesis was written during the aftermath of the unsettling time when society was noticeably affected by the COVID-19 pandemic. I am very grateful to the teachers, program coordinators and other staff at Linnaeus University for the adjustments and support they gave us during this difficult time, while still providing us with an exceptional and inspiring education.

There are several people involved in my degree project whom I would like to thank for their excellent guidance, support and feedback throughout this process: my supervisor, our degree course manager, the examiner and reader. Thank you all for your critical suggestions and encouraging comments. Especially my supervisor, Welf Löwe, who was always available and willing to answer any queries I had. While I discussed and formulated the research questions and experimental strategy with Prof. Welf Löwe, he ensured I was able to delve into this exploratory study and at the same time pointed me in the right direction whenever I needed it.

I would also like to gratefully acknowledge the support of Eduard Rall and other personnel at AIMO for kindly taking their time and effort to provide us with the data used for this study. There are several essential aspects of this project that would not have been possible without their help and support.

Finally, I would like to express my deep gratitude to my family. They have always motivated, encouraged and supported me with wise advice and kind words. My achievements would never have been possible without them. Above all, this is true for my husband Najib, who always stood by me and has been prouder of my accomplishments than I was.

Stay safe and I hope you enjoy reading this paper.

Angelica Hjelm Gardner
Stockholm, June 05, 2022

Contents

1 Introduction

The purpose of this study is to explore automated pain assessment with body movements as pain indicator using skeleton pose estimati on. The study is conducted through machine learning experiments on a real-world video dataset. The experimental objectives are pain recognition (detection of pain), pain intensity estimation (assessment of the pain level) and pain area classification (allocating the painful location in the body). Our experimental approaches are conducted either using only body data or in a combination with facial data to comprehensively investigate the influence of body movements on automated pain assessment solutions.

This section provides: **(1.1)** Brief background of automated pain assessment, **(1.2)** Relevance of the study, **(1.3)** Problem statement, **(1.4)** Outline of this work, **(1.5)** Contributions of the paper, **(1.6)** Target audience, and **(1.7)** Report structure.

1.1 Background

Pain assessment is an essential part of pain management in fields such as healthcare, rehabilitation, sports and fitness [1]. In pain assessment, the focus is on describing the pain, including the intensity, location or duration of the painful feeling. In a healthcare setting, pain is unlikely to be treated if it is not recognised, and an inappropriately assessed pain level can lead to suboptimal treatment. In a rehabilitation or fitness setting, it can be very beneficial to provide immediate, individualised feedback during physical activity, especially for those with musculoskeletal pain [2]. Although pain assessment is a general problem, it further affects patients with limited communication skills who are unable to articulate their experiences or whose self-report has low validity [3].

The experience of pain is complex, subjective and varies from person to person. Therefore, manual detection and measurement of pain is human-resource intensive and difficult to accomplish objectively [4]. This situation creates a need for automated systems where technical solutions can support individualised and patient-centred care in a robust and cost-effective manner. To enable such an automated approach, at least one pain indicator, called a modality, must be available as input to the system. Since the feeling of pain is both an unpleasant sensory and emotional experience, people express it in a variety of ways [3]. For this reason, researchers have focused on several modalities of interest in automatic pain assessment. The primary pain indicators are categorised into behavioural (e.g. facial expression, body gestures, paralinguistic vocalisation) and physiological indicators (e.g. brain activity, cardiovascular activity, skin conductance response) [1]. To date, much of the research has focused on the behavioural modality of facial expressions extracted from video recordings, sometimes in combination with head movements. Advances in wearable devices and electrode technology have also attracted research on the different physiological indicators of pain. In addition, studies increasingly aim to improve pain assessment performance by using more than one pain indicator simultaneously, this is known as a bi- or multimodal approach. An assessment system that processes information from one modality is a unimodal system; when multiple pain indicators are used, it is either a bimodal system (two modalities) or a multimodal system (three or more).

Despite the remarkable research on automated pain assessment, there are still few studies that incorporate the behavioural body modality perspective [3], [5]. To the best of our knowledge, previous studies of automated pain assessment that include body movements use multimodal approaches, and there are none that focus on unimodal body approaches. There may be several reasons for this gap. One reason could be that most of the publicly available pain databases only show the upper body or the head, neck and shoulder regions in videos [6]. However, perhaps the most relevant reason is that there is no formal standard for translating specific body gestures as expressions of pain. Compared to facial expressions, pain-related body behaviour is not as well established or as easily recognised when observed [7].

In the absence of a formal standard, research involving body movements differ in their approaches to translating body gestures into pain indicators. Some limit their anticipation of movement patterns to a particular type of pain, e.g. knee pain [8] or neck pain [9]. Others detect and process whole body regions from videos [10], and a few use a skeletal representation of body structure to model movements [11], [12]. Using such a skeleton pose estimation for body modeling is widely adapted in related research fields, e.g. emotion recognition [13]. The most common methods for body pose estimation are either a collection of body parts or a kinematic model [14]. The kinematic model is a collection of interconnected joints, similar to the human skeleton. In a video recording, the skeleton pose estimation creates a sequence of movements by forming a skeleton avatar based on the position and orientation of the body joints. Skeleton pose estimation has also been successful in research on automated movement assessment [15]. These aspects raise the question of whether the technique for estimating skeletal position to represent body movements in automated pain assessment deserves more attention.

Regardless of how the body movements were translated, the aim of earlier investigations was to detect pain (pain recognition) or to assess its level (pain intensity estimation). Werner et al. [3] suggested another interesting perspective that has not yet been explored: assigning the location of pain, i.e. the distinct area of pain in the body. Since it is possible to analyse the quality of movements using skeletal representations of the human body [16], it seems reasonable that skeleton pose estimation can be used to investigate all three of these aspects.

Most studies of automated pain assessment were limited to researching on databases recorded in controlled environments. One criticism of data collected in a laboratory setting is that people could easily be motivated to exploit a strategy by which their subconscious mind draws attention to painful feelings and reinforces pain-related responses [4]. As Nerella et al. [17] studied facial expression recognition on a real-world ICU (Intensive Care Unit) dataset, recognition performance was significantly worse than in previous reports for the same model. They showed that systems need to be trained on real-world data to achieve the ultimate goal of automated pain assessment in real environments. For our experiments, we use a real-world dataset of videos taken in uncontrolled environments, such as outdoors and at home. The video recordings show a person performing an overhead deep squat. We selected the data points so that about half of the subjects felt pain during the exercise and the rest did not. The subjects who feel pain do so in different parts of the body and with different pain intensity levels.

As far as we know, there is no exploratory research in the field of automated pain assessment that conducts experiments with skeleton pose estimation as a body pain indicator in a unimodal setup. Additionally, none of the previous studies of automated pain assessment tried to classify the pain area of the body in a multiclass approach. Consequently, exploring these aspects will be the focus of our study.

1.2 Motivations

Pain-related body gestures have not received much attention in studies of automated pain assessment. Although there is a research gap and a formal standard is lacking, other academic fields are actively investigating the orchestration of pain-related body gestures. In social sciences, pain-related body movements are described as a strong sign of pain [18]. This is even more true for musculoskeletal pain, where movement and physical activity are often the main trigger [19]. It is well known that body behaviour is both communicative and protective; body gestures involve both the expression of pain and provide non-verbal cues for communication [4], [7], [20], [21]. Furthermore, in emotion recognition, they study body movements for recognising our six basic emotions: Fear, Anger, Joy, Sadness, Disgust and Surprise [22], [23], [24], [25], [26]. From the results of Castellano et al. [27], movement gestures are suggested as an effective indicator of pain.

Of the few multimodal pain assessment studies that incorporate body movements, using a skeleton pose estimation is motivated by the typical pain behaviour people tend to adopt in response to a painful experience or anxiety about the painful sensation [11]. Pain-related movements are generally characterised by certain behavioural patterns, such as abrupt actions and limping. Protective behaviours such as hesitation, guarding and stiffness can also be observed [7].

Recognising the lack of research on body-based pain recognition, the EmoPain Challenge [5] was launched in 2020 as the first international competition aimed at comparing machine learning methods for pain assessment using human expressive behaviour. The challenge aimed to encourage incorporating body expressions for automated pain assessment and pain-related emotion recognition. The three tasks in this competition were: (i) Pain estimation from facial expressions, (ii) Pain recognition from multimodal movement, and (iii) Multimodal movement behaviour classification (i.e. classification of specific body movements as pain-related). We can observe that in this challenge, the focus was also on multimodal approaches or unimodal solutions using facial expressions.

For these reasons, our study serves a useful exploratory purpose. Even if our experimental trials would suggest that a unimodal body approach is not sufficient, it might still demonstrate whether the body modality influences bimodal settings and should, therefore, receive more attention in further bi- and multimodal studies.

1.3 Problem Statement

We conduct exploratory research using skeleton pose estimation to represent body modality in automated pain assessment with the objectives of pain recognition, pain intensity estimation and pain area classification. Experiments are first conducted in a unimodal approach where body information is the only pain indicator available to the system. The aim of running these experiments is to investigate basic performance from a unimodal body movement system. To more broadly understand what body movement data might indicate, we compare pain assessment performance from three different perspectives: merely the detection of pain, or can we also assess its intensity and even the body area where the pain is located? Second, experiments are also conducted with bimodal approaches using two pain indicators: body movements and facial expressions (with head pose). The aim of these experiments is to understand the role of body movements in a bi- or multimodal pain assessment system. We would like to answer the following research questions:

RQ1	What performance can we obtain when using skeleton pose estimation to represent body movements in unimodal pain assessment?
RQ2	Can unimodal pain assessment using skeleton pose estimation identify areas of pain in the human body?
RQ3	Does including body movement data improve pain assessment performance in a bimodal approach?

1.4 This Thesis Report

To answer the research questions, we conducted machine learning experiments, focusing on deep learning methods. Specifically, we selected two deep learning (DL) approaches that have shown promising results in recent pain assessment research: a hybrid CNN-LSTM architecture and a Recurrent CNN (RCNN). These models were explored in both unimodal and bimodal experiments. In the bimodal approaches, we fuse the body and face data in three ways: early fusion (feature-level combination), late fusion (decision-level combination) and ensemble learning (combining the predictions of multiple models).

The input data are video recordings showing a person's body and face while performing an overhead deep squat exercise. We extracted features that represent the selected pain indicators (body and face) using state-of-the-art tools. The human body skeleton was identified using a pose estimation model, and pain-related facial expressions were detected using a face recognition tool. The conducted experiments can be seen in Table 1.1.

Our results show promising performance on all three experimental objectives when using skeleton pose estimation, indicating the value of including body movements in automated pain assessment. Although the overall performance was good, it did not match the best-performing solutions from previous pain assessment studies. This is most likely because we used a real-world dataset and focused on exploration rather than optimising model generalisation capability and performance. We limited ourselves to exploring two deep learning architectures and three bimodal strategies. Although we selected these models and strategies based on previous studies, the successful application of DL techniques requires iterative decision making about architecture choices, increasing or decreasing model capacity, adding or removing regularisation features, or improving and optimising performance in other ways [28]. All of these processes are time-consuming to say the least, and given the exploratory nature of this study, we consider our limitations acceptable.

Approach	**Objective**	**Model architecture**
Unimodal (Body)	Pain recognition	CNN-LSTM
Unimodal (Body)	Pain recognition	RCNN
Unimodal (Body)	Pain intensity	CNN-LSTM
Unimodal (Body)	Pain intensity	RCNN
Unimodal (Body)	Pain area	CNN-LSTM
Unimodal (Body)	Pain area	RCNN
Unimodal (Face)	Pain recognition	CNN-LSTM
Unimodal (Face)	Pain recognition	RCNN
Unimodal (Face)	Pain intensity	CNN-LSTM
Unimodal (Face)	Pain intensity	RCNN
Unimodal (Face)	Pain area	CNN-LSTM
Unimodal (Face)	Pain area	RCNN
Bimodal (Early Fusion)	Pain recognition	CNN-LSTM
Bimodal (Early Fusion)	Pain recognition	RCNN
Bimodal (Early Fusion)	Pain intensity	CNN-LSTM
Bimodal (Early Fusion)	Pain intensity	RCNN
Bimodal (Early Fusion)	Pain area	CNN-LSTM
Bimodal (Early Fusion)	Pain area	RCNN
Bimodal (Late Fusion)	Pain recognition	CNN-LSTM
Bimodal (Late Fusion)	Pain recognition	RCNN
Bimodal (Late Fusion)	Pain intensity	CNN-LSTM
Bimodal (Late Fusion)	Pain intensity	RCNN
Bimodal (Late Fusion)	Pain area	CNN-LSTM
Bimodal (Late Fusion)	Pain area	RCNN
Ensemble	Pain recognition	all unimodal approaches
Ensemble	Pain intensity	all unimodal approaches
Ensemble	Pain area	all unimodal approaches

Table 1.1: The different experiments conducted in this study.

1.5 Contributions

Previous studies of automated pain assessment has focused on facial expressions or physiological signals as pain indicator, and only a few incorporate body movements into multimodal approaches. In those studies, the pain was detected either by pain recognition or by assessing its intensity, without determining the location of the pain. The contributions of this paper can be summarised as follows:

- The investigation of applications for the use of skeleton pose estimation in automated pain assessment, exploring both unimodal approaches and performance improvements in bimodal fusion. This study examines pain recognition and pain intensity assessment, as well as classification of body pain areas.

- The comparison of pain assessment performance from two different unimodal deep learning architectures with three different bimodal fusion strategies.

- The demonstration of automated pain assessment performance for face and body behavioural modalities on a real-world dataset.

1.6 Target groups

This study might be of interest mainly to an academic audience, but also to some extent to an industrial audience:

- The community of researchers focused on automated pain recognition and assessment. To date, the use of body movement as an input modality for these systems is unusual, although it has been suggested as future work [1], [27], [29]. In addition, some may be interested in the fact that this study used real-world data rather than the common public datasets, and how this may have affected predictive performance. They might also recognise the need to discuss the topic of standardising pain-related body gestures and present suitable substitute methods.

- Companies providing applications and services related to health, fitness, physiotherapy and rehabilitation. This exploratory study could be of interest for new service/product ideas as well as further development or improvement of existing ones.

- Clinical staff looking for information or indications of progress on automated pain assessment methods expected to benefit patients, especially those with limited communication skills.

1.7 Report Structure

The remainder of this report is structured as follows: Section 2 reviews existing automated pain assessment research and provides a technical background. Section 3 defines the dataset and feature extraction methods, presents the experimental design, inform about the deep learning models and fusion strategies used, and discuss reliability, validity and ethical considerations. Section 4 details how the experiments were conducted. Section 5 presents and evaluates the results as well as compares them with previous research. Finally, in Section 6 we state our conclusions and suggestions for future work.

2 Background

The standard practise in pain assessment has traditionally been self-report or visual inspection by an observer. Self-reporting requires patients to describe their pain using numerical, verbal or visual rating scales [30]. Such self-assessment has its limitations: it is subjective, can be influenced by mood, is hampered by language difficulties and comparisons are difficult [19]. It is also inaccessible to non-communicative patients [31]. An alternative to self-assessment is observer-based methods, where staff or caregivers use their intuitive sense, together with assessment tools, to estimate another person's pain. Due to the dualistic nature of pain, which is both a physical experience and a behavioural state, reliable use of a pain scale by an observer requires a considerable amount of prior training and experience. Observer-based assessments are also prone to error due to subjective bias and inconsistencies [3], and observers may overestimate pain levels [32]. In addition, certain healthcare circumstances may require multiple daily pain observations. This need for trained staff is resource-intensive and there is always a risk that episodes of pain may be missed or changes detected too late.

The introduction of automated methods for pain assessment allows for a computerised approach with potential benefits such as reliability, objectivity and continuous monitoring [4]. The possible applications for automated systems could extend beyond the capabilities of traditional approaches. Computerised approaches could be used as smart home monitoring to detect pain-related behaviours and get informed about how patients self-manage painful conditions at home [33], or to support automated rehabilitation systems that detects patient's quality of movement and assess recovery progress [34]. In healthcare, automated pain assessment could determine the impact of pain-modulating interventions, such as medication [35]. In anaesthesiological intensive care units, the benefit of an automated pain assessment lies in the avoidance of an over- or underuse of analgesics in patients with limited communication skills [36]. In addition, assessment of pain intensity could help predict long-term disability and quality of life outcomes [37]. In addition, there are commercial tools, such as smartphone apps, for application areas in geriatric and pediatric settings [38].

As a consequence of the critical aspects of these application areas, automated systems must meet certain requirements [1] including adapting and validating systems to diverse clinical populations, improving the truthfulness and robustness under non-laboratory conditions, and evaluating the acceptability of the technology. Acceptance of the use of automated AI systems and recognition of their ability to support diagnostics and therapy are generally high in surveys [36], [39]. From both a technical and clinical perspective, an optimal pain assessment system would be fully automated but allow for human interaction, able to continuously reflect information such as the presence and intensity of pain as well as its location, based on knowledge grounded in empirical literature, in line with well-established technological solutions, and following rigorous validation procedures to address issues such as bias and explainability [35]. Previous studies on automated pain assessment have mainly focused on exploring possible approaches and alternative AI solutions, while the actual implementation and practicality of these systems in the real-world are still largely unexplored.

To facilitate an automated approach to pain assessment, at least one channel of sensory input, known as a modality, must be made available to the system [3].

The primary modalities for pain recognition are categorised as behavioural (e.g. facial expression, body movement, paralinguistic vocalisation) and physiological (e.g. brain activity, cardiovascular activity, skin conductance response) [1]. Research on automated methods to support pain assessment has generated promising ideas and approaches through exploratory and confirmatory studies. The work to date can be viewed from two different angles, which we explore in the following subsections: **(2.1)** The input modalities used as pain indicators and **(2.2)** Their technical approach. When looking at these previous studies, one will be able to observe the limited inclusion of body movements already mentioned, even in multimodal approaches.

2.1 Pain Indicators

As mentioned, an automated pain assessment system requires at least one input modality as pain indicator. The modalities differ in terms of hardware. The majority of previous studies have analysed camera images or videos, focusing mainly on facial expressions [3]. There is also a notable line of research looking at contact-sensor approaches that use, for example, the electrical activity generated by skeletal muscles as a pain signal. Previous studies fall mainly into three categories [1]: **(2.1.1)** Pain assessment by facial expression, **(2.1.2)** Physiological signs for pain assessment, and **(2.1.3)** The multimodal assessment of pain.

2.1.1 Pain Assessment by Facial Expression

Facial expression scoring has been used in standard observer-based pain assessment tools, but manual coding of facial expressions is often too time-consuming for clinical practise [3]. For automated solutions, facial expressions are the most common research target for pain indicators of all behavioural modalities and they can be extracted from videos showing the facial region.

What distinguishes the studies on pain assessment through facial expressions from each other is their technical approach and research focus. Lui et al. [40] extracted facial expressions to estimate pain intensity, with the aim of creating a machine learning model that is interpretable and understandable in its decision making. Lee et al. [32] analysed facial keypoints of post- surgical patients by using images taken with a smartphone, illustrating the potential of efficient automated solutions. Vu et al. [30] process videos from two different datasets with the idea of reducing errors caused by the ML model memorising a dataset instead of learning actual pain-related facial patterns. Xu et al. [41] investigated correlations between different pain measurement methods. They used facial muscle movements to calculate a pain score and train a ML model that predicts multidimensional pain ratings. Zhou et al. [42] designed an architecture that can predict the pain intensity in each frame of a video by considering historical frames. Rezaei et al. [43] developed a system that estimates the pain intensity of older adults with dementia by comparing two images of the same person. Lucey et al. [29] studied situations where a patient is lying in bed and their face may be partially obscured due to the angle at which the patient is facing the camera. While video-based monitoring is considered a promising non-contact method for identifying moments of pain [44], it also has to cope with confounding factors and artefacts, such as changes in lighting, sudden movements and poor visibility of the face. In these situations, pain assessment can be hindered because information from certain facial features is missing, and the authors proposed technical solutions that could help improve performance. Pikulkaew et al. [31] and Bargshady et al. [45] developed more complex models that focus mainly on improving performance compared to the state-of-the-art. Rodriguez et al. [46] trains a model by cropping the facial region and providing raw images as input, unlike most approaches that extract facial features instead. Of the unimodal face approaches, only Xin et al. [47] report what they consider to be unsatisfactory performance. They mentioned that their approach analyses still face images and does not effectively use facial motion or temporal information, which could be the reason for the mediocre results.

There are three main reasons for the popularity of using facial expressions as an input modality: it is easy to use [30], it contains informative and reliable cues to mental states, and research has developed coding systems that establish criteria for distinguishing facial features of pain from other emotional expressions [4]. The Facial Action Coding System (FACS) is an anatomically based coding system that classifies facial movements by appearance [48]. It is a common standard for systematically categorising the physical expression of emotions, and this coding system is used by the majority of research to date. FACS is described in more detail in section 3.3.1. Although facial expressions are considered sensitive and some are specific to pain, there are criticisms against the use of this modality. First, facial expressions may be easier to fake than other physiological pain responses, and second, expressions may vary, with individuals showing only some of these combinations [3]. If a person shows no facial response at all due to low pain intensity or expressiveness, the pain score could be zero even though the person feels pain. Typically, if facial indicators of pain are present, this is likely to be a significant implication and should be taken seriously [35]. However, pain produces multiple behavioural responses, both within and across modalities, which may hold additional valuable clues and allow for wider use. In healthcare and rehabilitation settings, body language is an important behavioural indicator of pain that would be overlooked in pain assessment by facial expression alone, particularly in patients with moderate to severe cognitive impairment and in patients who have difficulty communicating verbally. In addition, nonverbal and verbal pain vocalisations have also been shown to be clinically useful for pain assessment in young children and others with limited linguistic abilities [49].

In more recent analyses of facial modality, studies have found that a combination of facial expressions and head posture achieves higher predictive performance [50], [51], [52], [53]. A head pose can be described by the position and orientation of the head in 3-dimensional space. Werner et al. [50] observed certain subtle head movements associated with pain, dominated by rotating or raising the head. They compared the results obtained from facial expressions alone with those obtained by using data from facial expressions and head pose. Pain assessment performance increased when head pose information was included. This shows that even if gesture analysis would not be a good unimodal pain indicator, it can still serve as a complement in a multimodal approach.

2.1.2 Physiological Signs for Pain Assessment

Physiological signals can be used to assess pain because the body triggers significant autonomic changes when pain occurs. These physiological cues can be easily detected if the person is wearing some kind of biosensor. Lopez-Martinez and Picard [54] estimated pain intensity using autonomic signals from skin conductance. In another study by the same authors [55], they investigated pain recognition methods based on the use of multitask learning that accounts for subject-specific differences in skin conductance and heart rate features derived from wrist sensors. Pouromran et al. [56] investigated the estimation of pain intensity using electrodermal activity (EDA), electrocardiogram (ECG) and electromyography (EMG) signals. Thiam et al. [57] presented several novel approaches to pain intensity assessment, also based on EDA, ECG and EMG signals.

Automated pain assessment based on physiological signals has advantages: usually, it does not require the same large amounts of data as behavioural modalities [58] and may be more appropriate for measuring pain intensity. In a study from 2020, anaesthesiology ICU staff gave physiological signals the highest priority in unimodal approaches because the staff were traditionally familiar with this modality [36]. Nevertheless, sensors that detect physiological modalities are sensitive and may record unreliable signals [59] and therefore, they are generally used only in combinations of two or more signals or in multimodal approaches. Unlike facial expressions that can be recorded in a non-contact and unobtrusive way, physiological cues require sensors that are in contact with the body [44]. While we have seen advances in wearable technology, these devices are not easily accessible to everyone and can be uncomfortable or impractical for the person wearing them. Contact sensors also suffer from movement artifacts and potential loss of contact [49]. These disadvantages make it difficult to regularly collect pain-related data about a person over time, especially in uncontrolled environments, to detect the effects of physiotherapy or exercise and to offer timely treatment [4].

2.1.3 The Multimodal Assessment of Pain

Multimodal solutions to pain assessment combine two or more of the behavioural and/or physiological modalities, e.g. one or more physiological cues combined with facial expressions or an audio-visual combination. The main goal of multimodality is to improve performance by combining complementary data [60] with the idea that some information may only be present in one modality (e.g. audio stream) and not in the other (e.g. video frames). If pain assessment performance from the individual modalities is sufficiently good, their fusion usually leads to better results if the heterogeneous information sources complement each other. In addition to better performance, a multimodal system also allows for greater flexibility and availability. In real-world environments, one modality may not be available due to various factors, e.g. the face may be obscured by an object, as Lucey et al. [29] studied, or the facial area may be too small in a video for accurate measurement of facial expressions. A multimodal system might be able to compensate for this absence of one or even more modalities, and still provide a useful assessment. Moreover, people express pain in multimodal ways [4] and therefore, it is not surprising that including multiple data sources has shown to improve pain recognition [44].

It has been discovered that multimodality in practise can in some cases degrade the performance of the model [61]. The success of multimodal learning essentially depends on better quality and more meaningful representation of the data. In situations where multimodality lead to poorer performance, it should be investigated which modality is the bottleneck and focus on improving it.

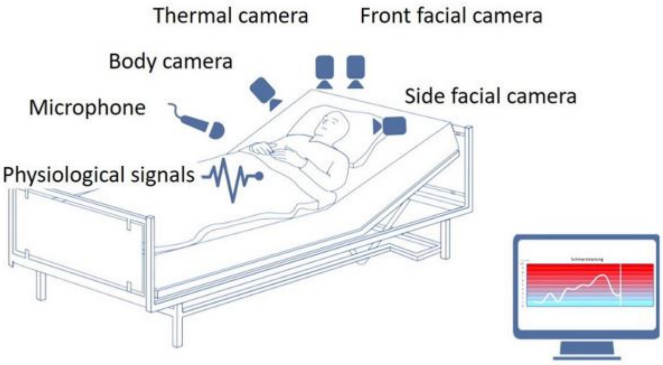


Figure 2.1: Potential components involved in pain monitoring. **Image source:** Adapted from [36].

Previous multimodal studies of automated pain assessment differ in what modalities they include, several possible modalities can be seen in Figure 2.1. Thiam et al. [58] conducted a comprehensive evaluation of different multimodal fusion strategies combining audio, video and physiological modalities. From the videos, they extracted facial expression and head pose, and used four different physiological signals (EMG, RSP, ECG and EDA) to perform both pain recognition (pain vs. no pain) and pain intensity estimation. Lopez-Martinez et al. [62] explored finding a solution to the variability of pain response between subjects. They extracted facial landmarks, head pose, eye gaze direction and facial muscle movements from videos to combine with two physiological signals. Then they clustered 85 subjects into profiles based on similarities in individual pain responses, and developed a model containing both shared parts for general pain-related patterns, and specific parts corresponding to the different profiles.

The majority of reviewed work used the common Facial Action Coding System (FACS) to obtain facial expressions as features for the face modality. However, Kächele et al. [60] extracted geometric-based and appearance-based features from facial landmarks instead. Their participants were allowed to move their head during the recordings so the facial region was not accurately recognised in every frame. They also included three physiological channels in their multimodal approach.

Some multimodal approaches involve the body modality. Salekin et al. [10] proposed a system that uses video (face and body) with audio (crying sound) and three physiological signals to assess postoperative pain in neonates (i.e. newborns in the first 28 days after birth). Werner et al. [63] combined several sensory modalities with facial expressions and head posture as well as audio signals to analyse para-linguistic responses. They mention that body movements are included in this multimodal approach, but it is not listed among their unimodal results. Olugbade et al. [11] analysed body movements through a skeleton representation in combination with muscle activity during physical exercise to identify pain levels in patients with chronic musculoskeletal pain. Other studies targeted a specific type of pain. Aung et al. [7] focused on detecting low back pain. They asked a group of experts to create behavioural categories that distinguished pain-related movement tendencies. Another study by Grip et al. [9] instead focused on neck pain by looking at different markers, such as the reaction time of each movement. Lai et al. [8] examined foot movements to detect knee pain and Olugbade et al. [12] recorded body movements of subjects wearing a suit with measurement units. The subjects had low back pain and performed a forward-moving exercise.

2.2 Technical Solutions

The other aspect to consider for automated pain assessment systems is the architecture of the technical solution. Previous studies have used one or a combination of two types: traditional machine learning and deep learning.

Machine learning (ML) is the ability of a computer to solve problems that require extracting patterns from data and make decisions based on those patterns [28]. A common general definition is that of Arthur Samuel [64]:

> *(Machine Learning is the) field of study that gives computers the ability to learn without being explicitly programmed.*

A ML algorithm is the method a system uses to perform its task. In traditional ML, these algorithms are often simple and depend heavily on the representation of the given data. Deep learning (DL) is a subfield of ML that structures algorithms into layers that create an Artificial Neural Network (ANN). ANNs learn and make decisions without the need to be given all instructions. DL algorithms are called "deep" because these models consist of multiple layers. The data is defined in an input layer and then a series of, so called, "hidden layers" that extract increasingly abstract features so the model can determine which concepts are useful for explaining the relationships in the data. Finally, there is an output layer for the prediction.

Many AI tasks can be solved using a traditional ML approach, but for more complex problems, it is impractical (sometimes even unattainable) to extract features that explain the data while also considering factors of variation [64]. Mapping data input to an output can be a complicated problem, e.g. considering all pain-related variations of body movements. Deep learning solves this by decomposing the complicated mapping into a series of nested, simpler mappings, each described by a different layer of the model [28].

Based on previous studies of automated pain assessment, the most widely used traditional ML approaches are variants of Support Vector Machines (SVM) and Random Forest [3]. Those are among the more powerful and versatile traditional models [64], and thus, it's understandable why they are popular in a considerable amount of the reviewed studies [2], [8], [11], [12], [29], [32], [53], [58], [60], [63], [65]. Others use a combination of traditional ML and DL, such as Lui et al. [40] that present a two-stage model with a deep neural network at the first stage, and a traditional model at the second stage. Lopez-Martinez and Picard [54] explored and compared traditional ML algorithms with two different DL models.

Yet, most recent pain assessment research focus on DL, especially to exploit temporal patterns [3]. A temporal pattern is a repeated segment of signals, e.g. a sequence of pain-related head poses [66]. DL methods offer automatic learning of both temporal dependencies (i.e. the effect of past behaviour on current behaviour) and automatic processing of temporal structures [28] such as pain-related trends in facial expressions and body movements.

The following subsections describe: **(2.2.1)** The deep learning algorithms used in previous studies of automated pain assessment, **(2.2.2)** Common steps for training, testing and evaluating deep learning models, and **(2.2.3)** Strategies for multi-modal combinations of deep neural networks.

2.2.1 Deep Learning Algorithms

Artificial Neural Networks (ANN) are at the very core of deep learning. These powerful and scalable algorithms were first introduced in neuroscience 1943 by McCullouch and Pitts [67] that presented a simplified computational model of how biological neurons might work together in animal brains to perform complex computations using propositional logic. Their contribution became the first ANN architecture and since then, there has been many more invented.

As described by Goodfellow et al. [28], deep learning (DL) models consist of structured layers that take in information from the previous layer and pass it on to the next layer. The layers in turn contain computational units. The basic computational unit in an ANN is the neuron, often referred to as a node or unit (see Figure 2.2). Each unit receives an input with an associated weight ***w***, the weight is assigned based on its relative importance over other inputs. The unit applies an activation function ***f*** to the weighted sum of its inputs, this function is called an Activation Function. The purpose of the activation function is to introduce non-linearity into the output of a unit. Each unit also receives a bias ***b*** associated with it. While the input weight decides how fast the activation function is triggered, the bias is used to delay the triggering of the activation function. There exist several popular activation functions; ReLU, sigmoid, tanh and softmax are names of a few common ones. Once the output of the unit is calculated, it is forwarded to the next layer of the model.

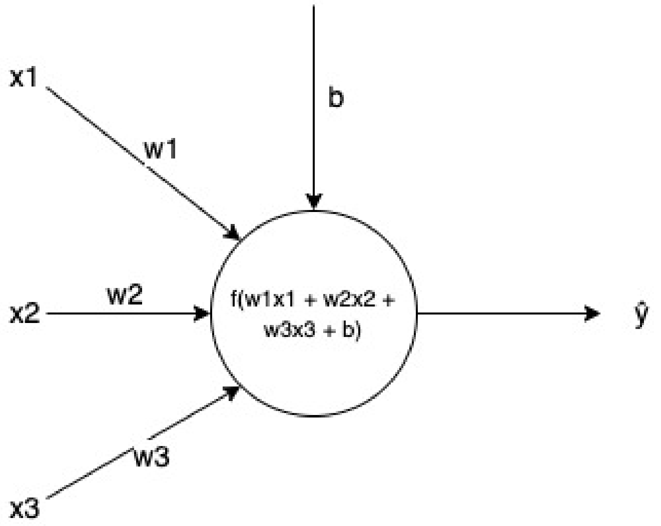


Figure 2.2: Each layer in an ANN consists of small computational units (neurons). The unit is a set of inputs, a set of weights, a bias and an activation function.

The most basic DL algorithm is a Fully-Connected Neural Network that consists of fully-connected layers which feeds all outputs from the previous layer to all its units, each unit providing one output to the next layer; making all units connected as can be seen in Figure 2.3. Most DL architectures end their stack with at least one or more fully-connected layers, even if the architecture's primary layer is another DL algorithm. These basic ANN are feedforward which means that the information moves in only one direction: forward, from the input layer, through the hidden layers and to the output layer. There are no cycles or loops in the network.

Initially, all input weights are randomly assigned and the network has no knowledge of the patterns in the data. For each sample in the training dataset, the ANN creates an output prediction. This prediction is compared to the ground truth using an optimisation process that requires a loss function to calculate the model error. Cross-entropy is a popular loss function that measures the difference between two probability distributions and quantifies the difference between the expected outcome and the outcome predicted by ANN. Any error is propagated back through the network to all its layers that adjust the weights accordingly using an optimisation function. Some common optimisation functions are Adam, RMSprop and Nadam. For an ANN to carry out this learning process, the backpropagation algorithm flows the error information backwards through the network, and the optimisation function calculates the direction of adjustment that should lead to better performance. When an ANN improves its performance in this way, it has learned from the patterns in the data samples and from its mistakes in prediction.

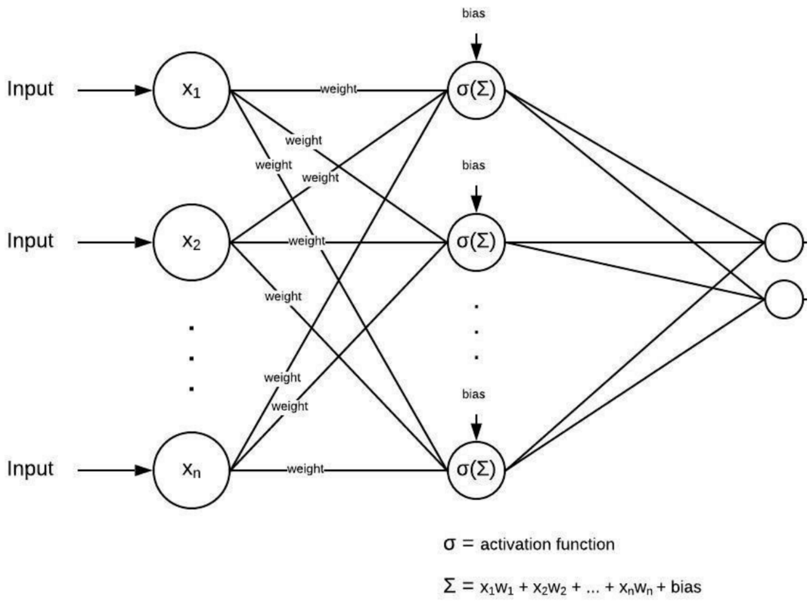


Figure 2.3: In Fully-Connected Networks, all units are connected.

In previous studies of automated pain assessment, a few developed novel DL architectures [41], [47], [55], [58]. However, most used two different types of common DL algorithms: Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

Convolutional Neural Networks (CNN) emerged from the study of the brain's visual cortex and is a popular DL algorithm used in many computer vision fields that deals with images and videos [64]. The most important building block of a CNN is the convolutional layer. A convolution is a mathematical operation that slides one function over another and measures the integral of their pointwise multiplication. Typical CNN architectures stack a few convolutional layers, followed by a pooling layer. Pooling layers are used to reduce the dimensions of the CNN output (i.e. the feature map). Hence, the pooling operation slides a filter over each channel of the map to summarize the features lying within the region covered by the filter. This way, the number of parameters to learn and the computation performed in the network are reduced [28]. There are some variations of the pooling process that can be selected, e.g. max-pooling returns the maximum value while average pooling returns the average of all the values. An example of a convolution and max-pooling operation can be seen in Figure 2.4. It is common to use a classic CNN architecture instead of developing a novel one. For instance, Vu et al. [30] propose a three-stage pain assessment framework that has the Inception-ResNet-v1 architecture [68] as backbone. Additionally, Pikulkaew et al. [31] used a model built on the ResNet-34 architecture [69]. However, most of the reviewed studies use VGGNet which is another classical CNN architecture that there exist several versions of, e.g. VGG-16 that has 16 layers and VGG-19 with 19 layers. This architecture has 2-3 convolutional layers and a pooling layer, then again 2-3 convolutional layers and a pooling layer, and so on, plus 2-3 final fully-connected layers before the output.

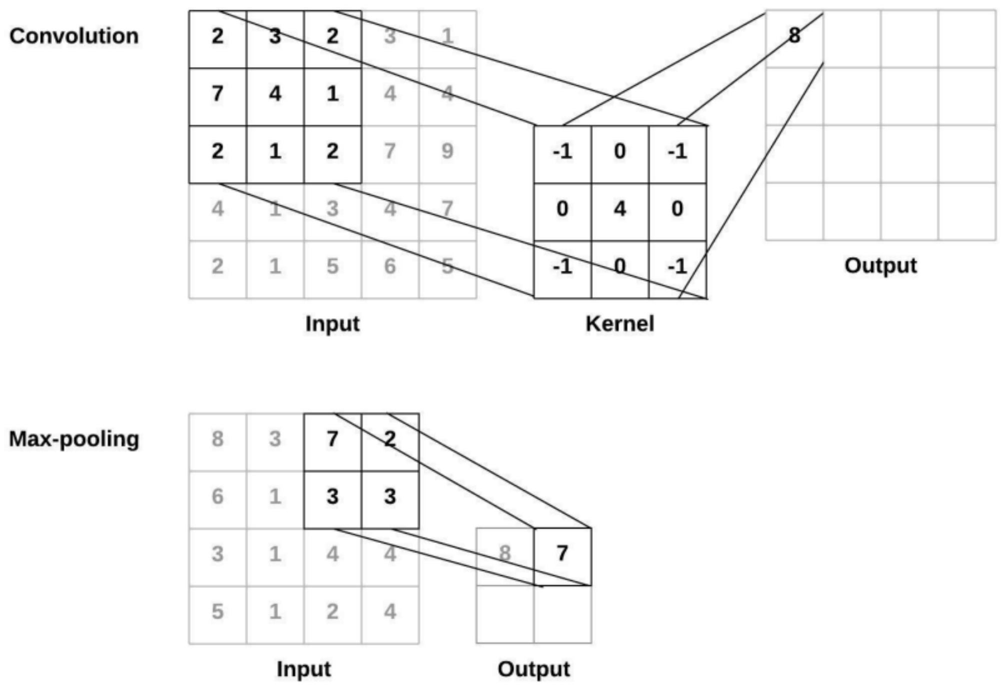


Figure 2.4: Example of a convolution process followed by a max-pooling operation where its output is the maximum value of the input.

Among the different challenges of automated pain assessment from video data is the ability to exploit both spatial and temporal information. Spatial information provides pain-related knowledge in a single video frame, such as the existence of a pain-related facial expression or body movement. On the other hand, temporal information exhibits the relationship between pain expressions revealed in consecutive video frames and thus provides valuable information about the behavioral state of a subject over time. CNNs can, in principle, consider both spatial and temporal information for dynamic pain patterns. However, to fully exploit temporal information, RNN is a more suitable DL algorithm.

Recurrent Neural Networks (RNN) are designed for sequence problems. Their recurrent connections add state (memory) to the network and therefore, allow it to naturally process temporal sequences by implicitly unfolding the networks according to the sequence length. The network needs to be provided with a time-steps number that represents how much memory we want the network to have. For instance, if we want the network to remember a full video sequence and this video sequence is 350 frames long, the time-steps number should be 350. Even though RNNs are promising in theory, they suffer from a technical problem of vanishing gradients which hampers learning of long data sequences. The gradients carry information used in the learning process so when the gradient becomes smaller and smaller, the parameter updates become insignificant which means no real learning is done. One type of RNN called Long Short-Term Memory (LSTM) is capable of overcoming this issue. To date, LSTM is the most common RNN type used in automated pain assessment studies. In LSTM, the cell state has three gates: input, forget and output gates (see Figure 2.5). The input gate controls what information should be stored in the state, the forget gate controls what information should be ignored or forgotten, and the output gate controls what information should be forwarded to the next state [28].

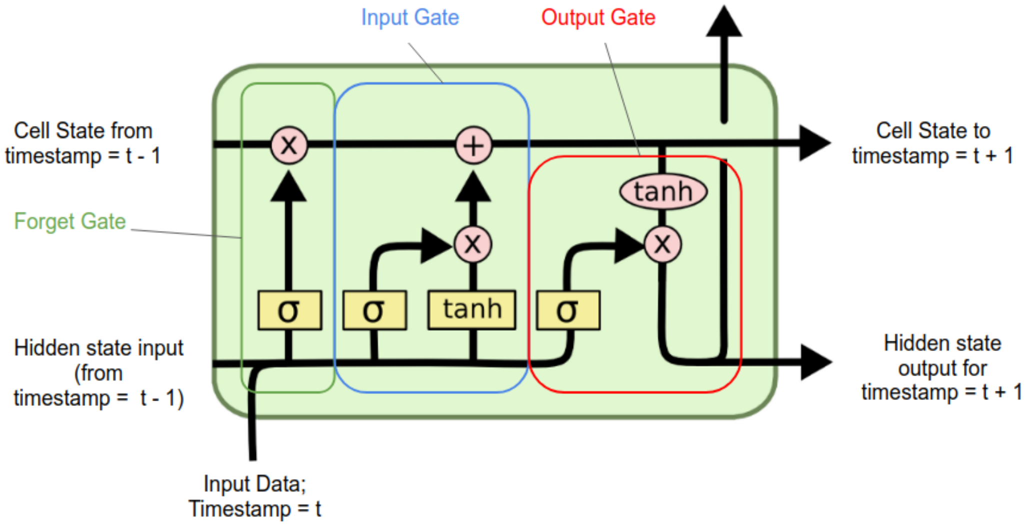

Input Gate
Output Gate
Cell State from timestamp = t - 1
Cell State to timestamp = t + 1
Forget Gate
tanh
X
+
X
X
σ
σ
tanh
σ
Hidden state input (from timestamp = t - 1)
Hidden state output for timestamp = t + 1
Input Data; Timestamp = t

Figure 2.5: A single LSTM cell with three its gates. **Image source:** [70]

Combining the advantages of CNN and RNN has been a prominent goal in recent automated pain assessment studies. Therefore, different structures of networks have been proposed to fuse convolutional and recurrent layers to capture both relevant structural and contextual information. Rodriguez et al. [46] was one of the first papers to use an approach that first learn facial features, then exploit temporal relationship between video frames. It outperformed the state-of-the-art at that time, and became an example of a hybrid CNN-LSTM model. Several following studies continued down the same path of developing hybrid pain assessment frameworks. For instance, Haque et al. [71] used a CNN model that is pre-trained on faces, called VGG-Face, then fine-tuned it against different pain modalities. Followed was an implementation of LSTM to estimate the temporal relations between the frames and perform sequence-level pain recognition. Salekin et al. [10] used three different unimodal hybrid CNN-LSTM networks (one for face, one for body and one for audio), and then combined their predictions at the end using late fusion. Rezaei et al. [43] developed a novel ANN architecture and compared its pain recognition performance to five other models from both traditional ML and DL.

One of these comparison models was a CNN-LSTM hybrid and in their study, it performed rather poorly. However, the authors attributed this poor performance to over-parameterization which resulted in the model learning the training data too well and couldn't generalize to unseen samples. The CNN-LSTM hybrid had 30x more parameters than their proposed model and was thus overfitting the training set. Bargshady et al. [45] also proposed using the pretrained VGG-Face model but instead combine it with a bidirectional LSTM (BiLSTM). BiLSTM adds one more LSTM layer which reverses the direction of information flow. Thus, the input video will be presented forwards and backwards to two separate LSTM layers. Then, the outputs of both layers are connected. This means that for every point in a given video sequence, the BiLSTM has complete, sequential information about all points before and after it.

Zhou et al. [42] approached combining CNN and RNN in a different way. They propose a pain intensity estimation framework based on a Recurrent Convolutional Neural Network (RCNN). The key module of RCNN is the recurrent convolutional layer (RCL). Recurrent convolutional layers incorporates recurrent connections into each convolutional layer where the state of the RCL units evolve over time-steps [72]. By leveraging the RCNN architecture, the proposed pain assessment framework predicts pain intensity by considering a sufficiently large historical frames and thus, the framework encodes spatial information without losing temporal information of video sequences. See Figure 2.6 for a high-level difference between the feedforward CNN connections where information is only processed in one direction, and the RCNN connections where, similar to loops, each step also utilise knowledge from previous steps in the sequence.

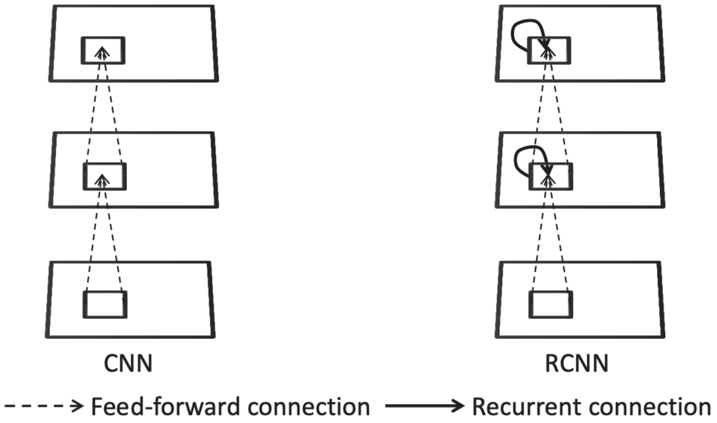


Figure 2.6: Difference between CNN (only feedforward connection) and RCNN (recurrent connection). **Image source:** Adapted from [72]

2.2.2 Training, Testing and Evaluating Deep Learning Models

When developing deep models, in addition to choosing an appropriate algorithm, it is also important to choose a suitable number of layers and fitting model configurations. This can be tricky and is largely dependent on the input data and problem domain. If we choose too few layers, the model might not learn well enough. But if we select too many layers, the model will start to memorise noise from the data samples instead of learning actual patterns. Additionally, most DL algorithms come with configurations called hyperparameters that affect the model's learning ability and final performance. The hyperparameter values are set before the training starts and if you do not optimise these values properly, it can lead to suboptimal performance. There are two approaches to selecting hyperparameter values: manual or automatic. Manual selection requires an understanding of what each hyperparameter does, how it works and how it allows the model to achieve good generalisation. Automatic selection, on the other hand, does not require the same understanding but is more computationally intensive. The most common automatic method is grid search, where we decide which hyperparameters we want to optimise and select a small finite set of values to examine. In grid search, we train one model for each combination of hyperparameters. The model that performs the best on the validation set is selected as the one with the most appropriate hyperparameter values. Typically, practitioners select grid search values on a logarithmic scale, e.g. a learning rate within the set {0.01, 0.001, 0.0001, 0.00001} and the number of hidden units within the set {50, 100, 200, 500, 1000}. Deep Learning models, including CNN and RNN, can have between a few and dozens of different hyperparameters. A brief description of five common hyperparameters are found in Table 2.2.

Hyperparameter	**Its effect**
Hidden units	Increasing the number of hidden units advances the capacity of the model. It also increases the time and memory costs for almost every model operation.
Learning rate	Works together with the optimisation algorithm by controlling how much we are adjusting the weights of our model with respect to the loss gradient. Both a too high or a too low learning rate will lead to a model with low capacity due to optimisation errors.
Convolution kernel size	Larger kernel sizes increases the number of parameters in the model and results in a narrower output dimension. Wider kernels require more memory for parameter storage and increase the model runtime.
Batch size	The number of training samples used in one "forward and backward" pass during model training. If we have 1000 training samples and the batch size=100, the algorithm trains itself with the first 100 training samples before learning from its error and adjusting its weights. Then it takes the second 100 samples and trains again, and so on. The size of a batch must be >= 1 and <= the number of samples in the training set.
Epochs	This hyperparameter is connected to the training process. When one epoch is finished, the ANN has seen all training data once. We train for multiple epochs and any value can be chosen. If we say that epochs=50, it means that the model goes through all the training data a total of 50 times.

Table 2.2: The effects of various hyperparameters on model capacity.

Another important consideration is the choice of performance metric. How will we measure the performance of the model and decide whether it is making progress? The most common metric is the accuracy of a system, but many applications or problem domains require additional metrics [28]. There are several types of performance metrics that are suitable for different problems. We will present the ones relevant to this study that has been used by the majority of previous studies in automated pain assessment [1].

There are four concepts that are important for the following performance metrics [73]. Below, these four concepts are presented in the context of binary classification where there are two classes that the model can predict: Pain or no pain.

- **True Positive (TP)**. The model predicts that someone is in pain and the person is actually experiencing pain. It is a true positive pain prediction.
- **True Negative (TN)**. The model predicts that someone does not feel pain and the person is not in pain. It is a true negative pain prediction.
- **False Positive (FP)**. The model predicts that someone is in pain but the person is not feeling pain. It is a false positive pain prediction.
- **False Negative (FN)**. The model predicts that someone does not feel pain but the person is experiencing pain. It is a false negative pain prediction.

Accuracy summarises the performance of a classification by dividing the correct predictions by the total number of predictions made by the model. It is the number of correctly predicted TP+TN relative to all data points, as shown in Equation 1:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Accuracy is a useful measure when the data set is balanced, i.e. there are almost as many videos with pain as those without [28]. In some cases where we have an imbalance, e.g. when there are not as many data samples with strong pain as with weak pain, Balanced Accuracy might be a necessary alternative. In Equation 2, ***M*** is the number of classes, ^{n}m is the number of data points belonging to class m, and ^{r}m is the number of accurately predicted data belonging to class m.

$$BalancedAccuracy = \frac{1}{M}\sum_{m=1}^{M}\frac{{}^{r}m}{{}^{n}m} \tag{2}$$

In some problem areas, it is more costly to make one kind of mistake than another: these mistakes are typically referred to as Type I and Type II errors. Type I errors refer to false positive (FP) predictions where a pain recognition system incorrectly classifies a person who is not experiencing pain as feeling pain. Conversely, a Type II error refers to the false negative (FN) predictions. FN occur when the model fails to recognise a painful experience by incorrectly classifying it as non-painful. In the event that one type of error is worse than the other, one way to solve this problem is to measure precision and recall [64]. Precision is the proportion of detected pain experiences that were correct, while recall is the proportion of pain events that were detected. A model that simply guess that all people are pain-free would achieve perfect precision, and a model that simply guess that all people are in pain would achieve perfect recall, so there is usually a trade-off between these two metrics [73].

Precision quantifies the number of correct pain predictions among all pain experiences predicted by the model, it calculates the accuracy of the true positives (TP) as seen in Equation 3:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Recall quantifies the number of correct pain predictions out of all the pain experiences the model could have recognised. Recall is also called the True Positive Rate (TPR) because we get the proportion of correct pain predictions:

$$Recall/TPR = \frac{TP}{TP + FN} \tag{4}$$

When using Precision and Recall, it is common to draw a PR (Precision-Recall) curve with Precision on the y-axis and Recall on the x-axis. A model that have no predictive skill would lie as a horizontal line at the bottom of this curve, and a model with perfect performance would give a curve as close as possible to the top right corner, as shown in Figure 2.7.

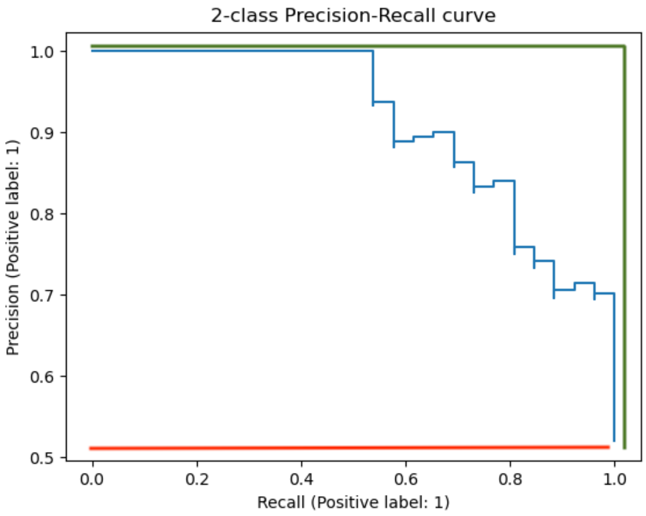


Figure 2.7: Example of a PR (precision-recall) curve. The red line represents a model with no skill, and the green line represents a model with perfect performance. **Image source:** Adapted from [74]

In cases where we want to summarise the performance of the model with a single number instead of a curve, we can convert Precision and Recall into F1-Score [28]. The F1-score balances Precision and Recall, and is often used when there is an imbalance in the class distribution. It is sometimes used as a statistical measure of accuracy and is determined as follows:

$$F1 = 2 * \left(\frac{Precision * Recall}{Precision + Recall} \right) \tag{5}$$

The previous performance metrics give us an indication of the performance of a model. However, when we want to compare models, it is common to use another measure: the Area under the ROC Curve (AUC) [73]. AUC summarises the trade-off between two equations: True Positive Rates (i.e. Recall) as was seen in Equation 4, and False Positive Rates, as seen in Equation 6. TPR describes the proportion of pain videos that the model correctly classified as painful experiences. We want the TPR to be as close to 1.0 as possible and if the TPR would have a lower value, it would mean that the model missed many painful experiences. FPR describes the proportion of pain-free videos that were misclassified as painful. We want the FPR to be as close to 0.0 as possible and if it has a higher value, it means that the model predicts many normal videos as showing pain patterns.

$$FPR = \frac{FP}{FP + TN} \tag{6}$$

Models predict probabilities where a threshold convert the probability to a prediction class. The default threshold is commonly 0.5, however, thresholds are problem-dependent and should be tuned [75]. Lowering the threshold will classify more videos as pain, thus increasing both False Positives (FP) and True Positives (TP). The ROC curve shows model performance across all possible thresholds, not just the one selected for a particular experiment, and the AUC gives an aggregated measure of all those possibilities [73]. By comparing AUC values and ROC curves between two models, we can quickly see if one of them has a better performance than the other where a higher AUC value indicates a better overall model performance. A perfect classifier has an AUC of 1.0, while a purely random classifier achieves about 0.5 [64]. In the ROC curve, a model with perfect performance is near the top left corner, as shown in Figure 2.8. The ROC curve and the PR (precision-recall) curve are quite similar. A rule of thumb is that the PR curve is preferred when the pain class is rare or when we are more concerned about misclassifying a painful experience that was not (FP) than missing painful experiences (FN). Otherwise, the ROC curve is recommended [28].

Confusion Matrix (CM) is another performance tool that is commonly used, but it is not a numerical measurement. The CM is a summary of the model predictions with the number of correct and incorrect predictions broken down by class. A CM provides information on how many errors the model makes and, more importantly, what type of errors [73]. When the Confusion Matrix is normalised, each class is represented as having 1.00 samples, i.e. the sum of each row is represented by 1.00, as seen in Figure 2.10. In a binary classification context, there will be four boxes representing true negatives (TN), false positives (FP), false negatives (FN) and true positives (TP). By inspecting the rows, we can analyse what errors the model seems to have more issues with, e.g. FP. In a multiclass classification context, the CM will have more boxes and we are able to analyse what classes the model was not good at distinguishing between. It is an NxN matrix where each row represents a class

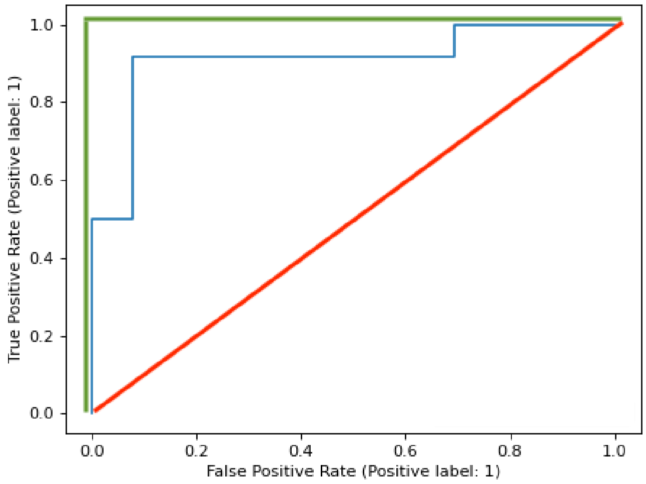


Figure 2.8: ROC curve where the red line represents a random classifier and the green line a model with perfect performance. **Image source:** Adapted from [76]

and for a model with near perfect performance, most of the predictions are on the diagonal, see Figure 2.9 for an example.

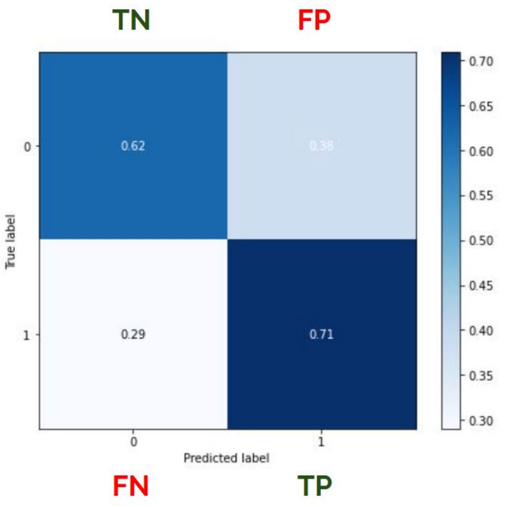
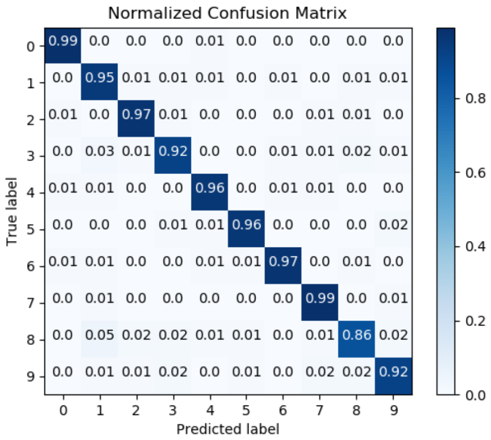


Figure 2.9: Example of a normalized Confusion Matrix for binary classification.

Finally, once we have the model architecture and performance metrics ready, we can start the training. Deep learning (DL) model training takes the process of how an ANN learns through backpropagation and places it in a larger context with additional aspects to consider. Before starting the training, the dataset is divided into separate parts for training and testing. It is important to split the training and test sets from the beginning so that the test set is never used until evaluation since independent training and test sets are necessary to obtain unbiased estimates of the generalisation capabilities of the model [28], [64], [75]. An unseen test set resembles a real-world situation where the model encounters new data that it has never seen before. Customarily, the training set is larger than the test set, with a



	0	1	2	3	4	5	6	7	8	9
0	0.99	0.0	0.0	0.0	0.01	0.0	0.0	0.0	0.0	0.0
1	0.0	0.95	0.01	0.01	0.01	0.0	0.01	0.0	0.01	0.01
2	0.01	0.0	0.97	0.01	0.0	0.0	0.0	0.01	0.01	0.0
3	0.0	0.03	0.01	0.92	0.0	0.0	0.01	0.01	0.02	0.01
4	0.01	0.01	0.0	0.0	0.96	0.0	0.01	0.01	0.0	0.0
5	0.0	0.0	0.0	0.01	0.01	0.96	0.0	0.0	0.0	0.02
6	0.01	0.01	0.0	0.0	0.01	0.01	0.97	0.0	0.01	0.0
7	0.0	0.01	0.0	0.0	0.0	0.0	0.0	0.99	0.0	0.01
8	0.0	0.05	0.02	0.02	0.01	0.01	0.0	0.01	0.86	0.02
9	0.0	0.01	0.01	0.02	0.0	0.01	0.0	0.02	0.02	0.92

Figure 2.10: A normalized Confusion Matrix for 10 classes. **Image source:** [77]

common ratio of 90:10 or 80:20 [64]. Another popular way to split the data is to split it into three separate sets rather than two: training, validation and test sets. The validation set is then used during the training process to validate model performance and tune its learning. The predictive performance we get with the validation set is not considered reliable enough because the model sees this data during training [28]. Therefore, we still need the test set to evaluate the final generalisation capabilities.

Splitting the data into three sets works together with a concept called Cross-Validation (CV). k-Fold CV is a resampling method in which the training set is split into one validation part and k-1 parts for training. Figure 2.11 shows an example where k=5. At the top of the figure, the full dataset is split into a training set and a test set. Further, the training set is split into five equal parts and we get five repetitions; using one of the parts for validation and the other parts for training. When all five repetitions are complete, the training is considered finished and we can test the performance of the model against the test data. k-Fold CV generally results in a less biased and less optimistic estimate of the model skill [73].

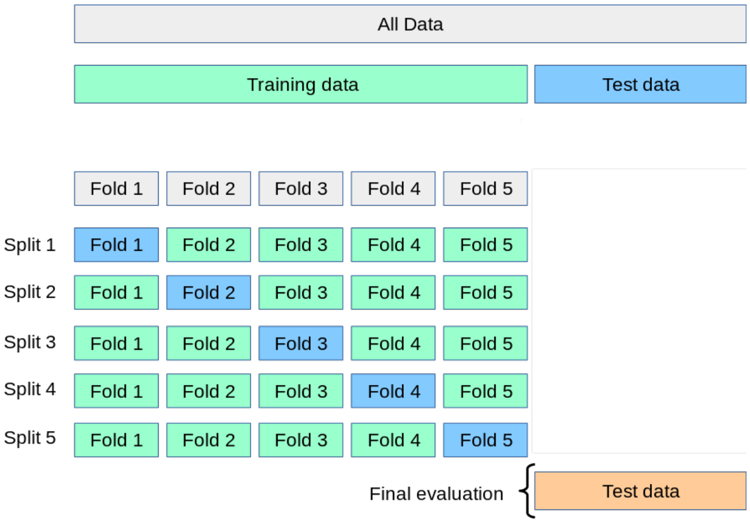


Figure 2.11: k-Fold CV where k = 5. **Image source:** Adapted from [78]

Previous studies on automated pain assessment have mostly proposed the use of a cross-validation procedure called Leave-One-Subject-Out (LOSO) [1]. In LOSO, one subject (person) is removed from the dataset to form a separate test set. This test set may contain one or more recordings, but they are all from the same person, and there are no recordings from that person in the training set. LOSO has the advantage of resembling a real-world situation where previously unknown people need to be assessed for pain.

2.2.3 Multimodal Deep Learning

Multimodal learning incorporates information from multiple sources [28]. As mentioned earlier, studies on automated pain assessment have increasingly focused on multimodal solutions that combine two or more of the behavioural and/or physiological pain modalities. From a technical perspective, multimodal research has taken several directions. One of these directions is fusion, which assumes that using the complementary aspects of heterogeneous data leads to more reliable models [79]. There are three techniques used for multimodal fusion of data: Early Fusion, Late Fusion and Intermediate/Hybrid Fusion [80].

Early fusion is perhaps the simplest alternative, it is also called data-fusion or feature-level fusion. It merges the data sources at the beginning of the process, before model training begins. The simplest form of early fusion is to concatenate the data features into a larger dataset, e.g. 50 features from one modality and 50 features from another modality result in a dataset with 100 features. A disadvantage would be if the resulting dataset becomes very high dimensional. The assumption behind early fusion is that the different data sources are independent. However, as we can understand, this assumption is not always true, as several modalities can have highly correlated features [81].

The most common alternative is late fusion, also known as decision-level fusion. Here, the data modalities are used independently by training separate unimodal models. This is suitable for cases where the different modalities may need tailored ways to process the inputs. After the unimodal models are trained, their prediction results are merged. There are several approaches to determine the optimal way in which these independently trained models are eventually merged. Bayes rules, max-fusion and average fusion are some examples of common rules.

Intermediate Fusion is a hybrid approach that attempts to combine the properties of the previous methods. It is the most flexible method that allows fusion at different stages of model training. For example, the different modalities can be simultaneously fused into a single shared representation layer at an early stage of the training process.

Ensemble learning is a concept related to multimodality. In ensemble learning, multiple machine learning models are trained to solve the same problem, similar to the late fusion approach, and these models are called ensemble members. After training, the members make predictions that are combined so that the ensemble can make a final classification based on the contributions of its members [28]. Ensembles can be as small as three, five or ten trained members, but can theoretically be as large as necessary. The intuition of ensemble learning is that members should be as accurate as possible, but also diverse enough to make statistically independent mistakes. In this way, the individual weaknesses of each member can be compensated for by the contributions of the other models [79].

Previous studies of automated pain assessment have achieved multimodality in all these different ways. Olugbade et al. [12] developed a bimodal approach using early fusion. Salekin et al. [10] developed a multimodal approach using late fusion. Most papers performed experiments with both early and late fusion [58], [59], [63], [71]. In addition, some use ensemble approaches instead [41], [45], [53].

3 Method

To investigate using skeleton pose estimation in automated pain assessment, we conducted machine learning experiments that systematically test the predictive performance of deep learning models trained with both unimodal and bimodal approaches. In this study, we extract the body skeleton pose and facial expressions from the same video to use as, what we consider to be, uni- and bimodal approaches. One could argue that using only video data is not purely bi- or multimodal, and that it would be better to include physiological cues and/or speech. At the same time, it is convenient to recover multiple data representations from video data alone, especially in circumstances where patients do not accept contact-sensors or video sound is absent. This is particularly interesting as there are studies suggesting that facial expressions may be able to replace some physiological cues [44] and that movements may be able to replace some verbal communication [82]. In the following subsections we define the: **(3.1)** Structure of the dataset, **(3.2)** Experimental approach, **(3.3)** Feature extraction phase, **(3.4)** Deep learning model architectures, **(3.5)** Bimodal fusion strategies, **(3.6)** Reliability and validity concerns and **(3.7)** Ethical considerations.

3.1 Dataset Structure

The dataset used in this study comes from a private real-world collection provided by AIMO[1] and the research project LOUISA[2], that interactively analyses the human musculoskeletal system and any pain using a 2D camera on a smartphone and several other modalities including sound and skin conductance. The dataset used in the present experiment consists of 1059 video recordings from 807 participants. Participants are early adulthood and middle-aged adults. Each video recording shows a person performing an overhead deep squat. The overhead deep squat is suggested by the National Academy of Sports Medicine (NASM) as a useful indicator of movement quality. This exercise challenges the mobility of all the major joints in the kinetic chain used to describe human movement. Therefore, the overhead deep squat is a commonly used key exercise for movement assessment [83] and could be an appropriate exercise to provide clues to pain-related movements.

All participants answered a self-assessment questionnaire about pain in connection with the recording of the exercise. About half of the subjects are pain-free, the other half have pain in one or more areas of the body (pain area) and in varying intensity (pain level). The dataset contains a total of 41 body areas, e.g. the right back shoulder blade, the left front hip, the right back gluteus and the left lower back. It also contains a numerical pain intensity rating between 1-10 (mild to severe).

The videos are recorded at a fixed resolution of 480x720 pixels and the recordings are between 5-7 seconds long. Both the face and the body are visible in the video sequences, so it is possible to extract data on body movement, facial expression and head pose from a single video. The definition of 'movement' is a change in position over time and should therefore be described as a series of successive video frame sequences. The video frame rates (i.e. recorded frames per second) vary depending on the capacity of the recording device used. Therefore, we had to trim the video sequences so that they were all the same length and we chose a fixed length

[1] https://www.aimo-fit.com/

[2] https://www.interaktive-technologien.de/projekte/louisa

of 350. For shorter videos, we append a copy of the last frame until the recording reaches 350 frames, and for longer videos we remove the frames after 350.

The dataset is divided into a training set and a test set. Even though most previous studies of automated pain assessment suggested using the LOSO approach, leaving out only one person is not a good scenario for this study. We want to test all pain intensities and all pain areas at least once, and there is not one person in the dataset that will suffice with so many different video recordings. Instead, we split the subjects 90:10 so that we have 80 subjects in the test set and 727 subjects in the training set. The videos of these 80 participants are included in the test set, while the videos of the remaining participants remain in the training set. In this way, we have the advantage that we can test never-before-seen subjects, but still test all the prediction targets at least once.

3.2 Scientific Approach

Systematic experimentation is an essential part of answering applied machine learning (ML) questions. Since the results of a single experiment are probabilistic and subject to variance, repeated trials are required to determine both the expected outcome and its range of variation [84]. In such experiments, we have complete control over the environment and conduct repeated trials to gain insights from the results. The independent variables are held constant and modified one at a time to determine their effects on the dependent variable. In this study, the independent variables are the input modalities, model architectures, model hyperparameters, bimodal fusion strategies and experimental objectives. The dependent variable is the estimated predictive performance on unseen data. As this is an exploratory study, we have chosen a practical methodology inspired by the suggestions of the "Build-and-fix" approach described by A. Ng [85]. An overview of our method can be seen in Figure 3.12, it is divided into four phases:

1. **Experiment Setup:** In this phase, we build the technical environment to prepare for conducting repeatable experiments. The project is structured after the Cookiecutter Data Science template[3], where files and directories are arranged to make the experiments straightforward and manageable. The experiments are conducted in Jupyter Notebooks[4]. Jupyter Notebooks are web-based interactive platforms that are used to create and share documents which contains live code, equations, visualizations, and text. In addition, experiment objectives are set, deep learning models and fusion strategies are selected, and the dataset is preprocessed by feature extraction.

    (a) Our experimental objectives are: Pain recognition, pain intensity estimation and pain area classification. Pain recognition can be understood as a binary classification problem, where the goal is to determine whether a person is in pain or not. For this objective, we divide our dataset samples into two categories: those who have pain and those who do not. Pain intensity estimation can be understood either as the determination of a numerical intensity or as a multiclass classification of several pain levels. For this objective, we divide the dataset samples into three different pain levels depending on what the participants self-reported as their

[3] https://github.com/drivendata/cookiecutter-data-science

[4] https://jupyter.org/

pain intensity rating. The pain area objective can also be understood as multiclass classification. Therefore, we divide the dataset samples into four different body surface areas, depending on what the participants themselves indicated as a painful area in their body.

   (b) Deep learning models and bimodal fusion strategies are selected based on a combination of theoretical advantages, performance presented in previous studies, and ease of implementation. The models and strategies should be theoretically appropriate for the problem domain, have been used in previous research with promising performance, and have straightforward implementation so it does not take too much time from experimentation. The DL models are described in section 3.4, while the fusion strategies are described in section 3.5.

   (c) Our dataset is a real-world video collection of people performing an overhead deep squat exercise. We perform feature extraction to transform this raw data into numerical features that can be processed by the models. For the body modality, we extract features that estimate the skeletal position, and for the face modality, we extract features that represent facial expressions and head pose.

2. **Baseline Experiments:** Once the experimental environment is ready, we implement a baseline model to establish a basic performance as quickly as possible. In machine learning, a baseline is a simple model that gives reasonable results without taking a lot of time to build [64]. It serves as a comparison when we attempt to improve performance, and it can tell us whether changing the model architecture and optimising the hyperparameters adds value. It is acceptable if the baseline result is poor, it may simply indicate that the algorithm has room for improvement. It is recommended to choose a baseline based on the structure of the data. In this study, the input is a video sequence, which means that an RNN, e.g. a simple LSTM, would be suitable [28].

3. **Model Optimization:** The next phase focus on improving pain assessment performance using different techniques. Optimising model performance is one of the more challenging aspects of implementing ML solutions [64]. Since this study is primarily exploratory, the scope of optimisation is not exhaustive. Rather, we focus on:

   (a) Optimising hyperparameters with automatic grid search. We optimised: the number of hidden units, activation function, learning rate optimisation function, dropout rate and size of the convolution kernel.

   (b) Generating data variants through data augmentation. Data augmentation increases the size of the dataset using techniques such as randomly rotating existing videos so it appears to the model as if it is new recordings. This has the advantage of a larger dataset, can address class imbalance, and forces the model to be tolerant of variations in position, orientation and size of the target [64].

   (c) Improving the generalisation ability of the model through regularisation techniques that improve performance on unseen data.

4. **Model Experiments:** In the final phase, we perform experiments with the optimised models, all conducted experiments was seen in Table 1.1. It is the primarily the results from this final phase that we take into account during the evaluation. We selected the following set of performance metrics: {accuracy, balanced accuracy, AUC, precision, recall, F1 score} and we also use the confusion matrix tool.

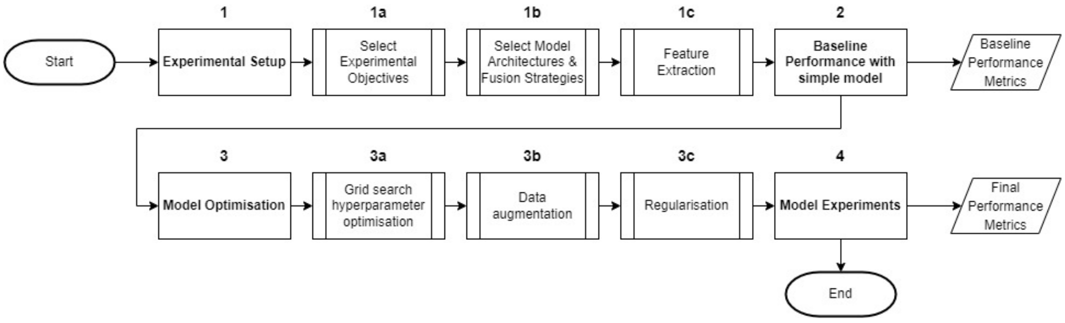

1
1a
1b
1c
2
Start
Experimental Setup
Select Experimental Objectives
Select Model Architectures & Fusion Strategies
Feature Extraction
Baseline Performance with simple model
Baseline Performance Metrics
3
3a
3b
3c
4
Model Optimisation
Grid search hyperparameter optimisation
Data augmentation
Regularisation
Model Experiments
Final Performance Metrics
End

Figure 3.12: An overview of the method used in this study with four phases.

3.3 Feature Extraction

The feature extraction process occurred in phase 1c in our method (Figure 3.12). In video-based pain assessment, feature extraction is part of the processing pipeline. Feature extraction refers to the process of converting raw data (the videos) into numerical features that can be processed by the ML model. In this study, feature extraction only needs to be done once before beginning the experiments. We selected two open-source toolkits: one for pose estimation (body features) and another for facial expression and head pose extraction (facial features). The two toolkits used are PoseNet[5] pose estimation model that detects the spatial positions of key body joints (keypoints) to form a body skeleton avatar, and OpenFace[6] face recognition toolkit that estimates the facial action units (AUs) and head pose.

3.3.1 Facial Action Units and Head Pose

For the representation of face modality, we use the Facial Action Coding System (FACS) to extract facial features and calculate a pain score based on these extracted facial expressions. FACS is an anatomically based coding system that classifies facial movements by appearance [48]. It is a common standard used to systematically categorise the physical expression of emotions. FACS defines Action Units (AU) which represent a contraction or relaxation of one or more muscles. AUs are frame-level metrics that indicate the presence or absence of facial movement and calculate its intensity. They are a consistent and reliable method for representing pain expression [41]. Pain-related AUs can be seen in Table 3.3.

[5] https://github.com/tensorflow/tfjs-models/tree/master/pose-detection

[6] https://github.com/TadasBaltrusaitis/OpenFace

AU4	Brow Lowering	AU12	Lip Corner Puller
AU6	Cheek Raising	AU20	Lip Stretch
AU7	Eyelid Tightening	AU25	Lips Parting
AU9	Nose Wrinkling	AU26	Jaw Dropping
AU10	Upper Lip Raising	AU43	Eyes Closed

Table 3.3: Pain-related Action Units description. **Source:** FACS [86].

Based on FACS and AU, a pain metric called Prkachin and Solomon Pain Intensity (PSPI) was developed. Prkachin and Solomon [87] argued that pain expression has a distinct configuration compared to other emotional expressions and defined a single number that measures pain as a combination of AU intensities:

$$PSPI = AU4 + max(AU6, AU7) + max(AU9, AU10) + AU43 \tag{7}$$

Although the OpenFace toolkit we use is one of the best-performing open-source facial toolkits, it is only trained to predict 9 of the 10 pain-related face action units. The action unit (AU) that OpenFace cannot recognise is AU43 - Eyes Closed. Rezaei et al. [43] suggest that action unit 45 (AU45) 'blinking' can be used instead of AU43. However, we will not take this chance since we believe it is very likely that participants will blink during these video recordings, regardless of whether they are in pain or not. Consequently, we do not want to confuse the model with a person not having pain but still showing the presence of a facial action unit, which in this case is not associated with pain, but resembles it.

The OpenFace toolkit also estimates head posture. Head pose is represented by three rotation angles (pitch, yaw and roll) and three coordinates (x,y,z) for the position of the head in a frame are also extracted using the same tool, as shown in Figure 3.13. The rotation is in world coordinates, where the camera is the origin and the position of the head is given in relation to the camera.

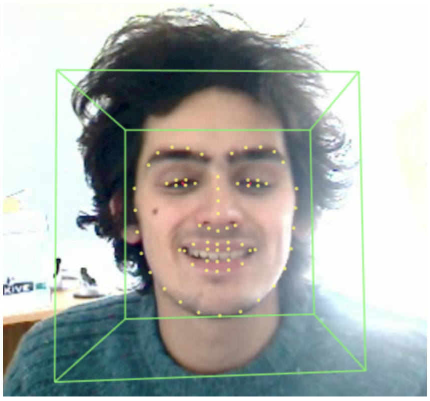


Figure 3.13: Facial landmarks and head pose extracted by OpenFace. **Image source:** Adapted from [88].

3.3.2 Skeleton Pose Estimation

For the representation of body modality, we use skeleton pose estimation. The human body is a complex structure with joints and limbs, and we can model it by extracting keypoints that reflect the body poses [89]. Most methods use a kinematic model consisting of a set of joint positions and limb orientations to represent the human body structure [21]. This kinematic model creates a skeleton avatar that can be used both in isolation to estimate a pose and in a sequential order where multiple poses form a movement. The kinematic model has limitations in terms of texture and shape information, but offers advantages in flexible movement representation.

Our chosen pose estimation toolkit estimates the human pose from a video by locating individual, specific keypoints and predicting their relative displacements, which allows grouping keypoints into a pose, as shown in Figure 3.14.

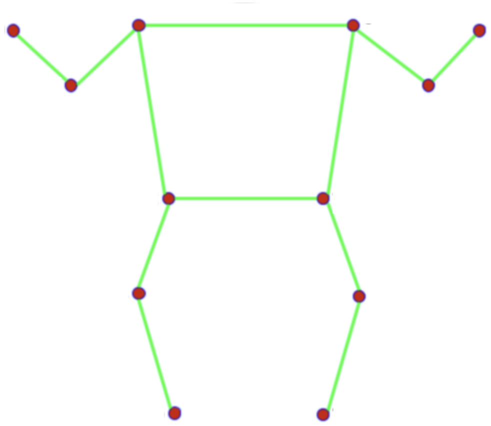


Figure 3.14: Skeleton keypoints extracted by PoseNet: left shoulder, right shoulder, left elbow, right elbow, left hand, right hand, left hip, right hip, left knee, right knee, left foot and right foot. **Image source:** Adapted from [90]

3.4 Deep Learning Models

In this study, we experimented with three different models, see our method in Figure 3.12: one baseline (phase 2) and two deeper architectures (phase 3-4). All three models were trained to predict the three experimental objectives: pain detection, pain intensity estimation and pain area classification.

We implemented a simple LSTM model as the baseline, suggested by Goodfellow et al. [28]. The authors proposed reasonable choices for model configurations which we adopted: the optimisation algorithm ***Adam***, 50 units per layer, batch size=32 and epochs=50. They also suggested to use early stopping universally, which we have added to all experiments. Early stopping means that we stop the training early if we notice that the model is not improving anymore. Since LSTMs traditionally use the activation function ***tanh***, we decided to keep this default. The baseline model has an identical architecture for all approaches and its hidden layers can be seen in Table 3.4. Experiments with this simple baseline model are conducted in method phase 2 (Figure 3.12).

Layer Type	**Configuration**
RNN	LSTM, units=50, activation=tanh
Fully-Connected	Dense, units=50

Table 3.4: A simple LSTM baseline model.

In the following phase, we implemented two deep learning models that were used for the rest of the study. The purpose of these models was to implement more complex and capable networks that would seemingly increase performance in pain assessment. After considering theoretical advantages of different DL architectures and recent pain assessment studies, we decided to use a hybrid CNN-BiLSTM and a Recurrent CNN (RCNN). We select hyperparameter values for these two models through hyperparameter optimisation (phase 3a in Figure 3.12) and conduct experiments with them in phase 4.

The CNN-LSTM architecture has been proposed in several studies for pain assessment. Many of these studies used the VGG-Face model, which was pre-trained for face recognition and then fine-tuned for pain assessment. VGG-Face is based on the VGG-16 architecture, which has also been used in other studies. Since we already extract facial features and use them as input instead of raw images, we saw no need to use VGG-Face. Nevertheless, we still adopt the VGG-16 architecture. We made an adjustment to the convolutional layers in the VGG-16 architecture by changing from 2D to 1D. The second dimension was not needed because we do not provide images directly as input. VGG-16 is a simple CNN architecture with convolutional layers that use small kernel-size filters. The network starts with a depth of 64 units and gradually increases by a factor of 2 until it reaches 512. As we started training this model, we quickly realised that it was over-parameterised for our problem, a situation similar to Rezaei et al. [43]. VGG-16 contains 5 convolution stacks with gradually increasing units, and ends up with two fully-connected layers of 4096 units each. Over-parameterisation means that we have more model parameters than necessary and thus train a model that is too complex, resulting in poor performance. Therefore, we removed the first stack and the last two stacks from this architecture. We also reduced the number of units in the fully-connected layers. Finally, we added two LSTM layers to create the hybrid architecture. Following Bargshady et al. [45], we use a bidirectional LSTM. The bidirectional approach is useful because we do not know at what point during a video sequence that the person feels pain. When the LSTM layers are bidirectional, they have information about all points in the video, before and after. The architecture of the our CNN-BiLSTM hidden layers can be seen in Table 3.5. Hyperparameter optimisation for this model showed that the ***tanh*** activation function and ***Nadam*** optimisation function gave the best performance.

Our Recurrent CNN (RCNN) implementation is based on the paper by Liang and Hu [72] that Zhou et al. [42] also used for pain assessment. Just as we adjusted the convolutional layers to 1D in the hybrid CNN-BiLSTM model, we made the same adjustment here for the same reason. We also needed to reduce the number of stacks in this model as it was too complex for our problem. The architecture of the RCNN hidden layers can be seen in Table 3.6. The first layer in the model is a standard feed-forward convolutional layer without recurrent connections. Follow-

Layer Type	**Configuration**
CNN	Conv1D, units=128, kernel_size=3, activation='tanh'
CNN	Conv1D, units=128, kernel_size=3, activation='tanh'
Pooling	Maxpooling, pool_size=2, strides=2
CNN	Conv1D, units=256, kernel_size=3, activation='tanh'
CNN	Conv1D, units=256, kernel_size=3, activation='tanh'
Pooling	Maxpooling, pool_size=2, strides=2
RNN	LSTM, units=350
RNN	LSTM, units=350
Fully-Connected	Dense, units=256, activation='tanh'
Fully-Connected	Dense, units=512, activation='tanh'

Table 3.5: VGG-16 and Bidirectional LSTM hybrid model.

ing, there are several stacks placing one recurrent convolutional layer followed by a normal convolution, and so on. Maxpooling is used at the end of the last stack. Additionally, we added two fully-connected layers at the end of the architecture which were not there in the original implementation.

Hyperparameter values were selected through grid search optimisation, and it showed that the ***tanh*** activation function and ***Nadam*** optimisation function gave the best performance for this model as well.

Layer Type	**Configuration**
CNN	Conv1D, units=128, kernel_size=1, padding='same', activation='tanh'
RCL	PReLU
CNN	Conv1D, units=256, kernel_size=3, padding='same', kernel_initializer='he_normal', activation='tanh'
RCL	PReLU
Pooling	MaxPooling1D, pool_size=2, strides=2
Fully-Connected	Dense, units=256, activation='tanh'
Fully-Connected	Dense, units=512, activation='tanh'

Table 3.6: Recurrent CNN (RCNN) model.

3.5 Bimodal Fusion Strategies

The deep learning models presented in the previous section form the basis for our experiments. They are used exclusively in the unimodal approaches and with some adaptations in the bimodal approaches. For our experiments, we have chosen three bimodal fusion strategies: an early fusion approach, a late fusion approach and an ensemble approach.

Early fusion is a feature-level approach where we extract features for the different modalities and then merge them into one dataset. We then train the same model architectures as for the unimodal approaches (CNN-BiLSTM and RCNN), but this time the model is given a larger dataset that contains both body and face modality information. This description can be seen in Figure 3.15.

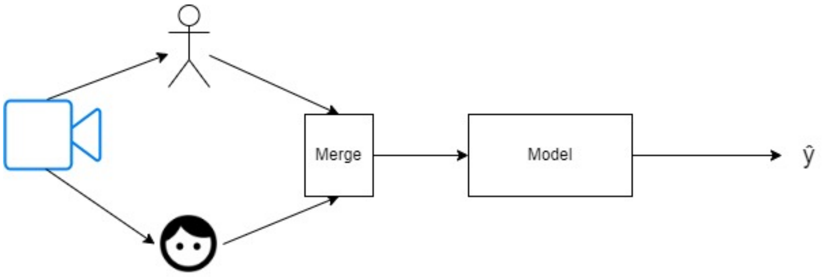


Figure 3.15: Early fusion approach where we combine the input features into a larger dataset before model training.

In contrast, late fusion is a decision-level approach where we extract the features and train separate and independent unimodal models, then concatenate what they have learned at the end of the architecture. This concatenation layer takes the inputs from each model and provides a single output that is the concatenation of the information that the models have learned from their respective modality. The late fusion concatenation approach is illustrated in Figure 3.16.

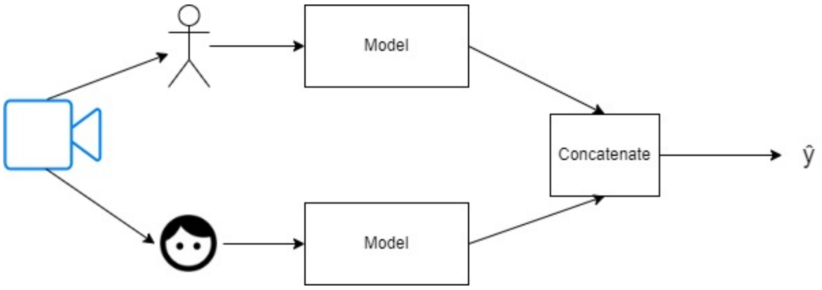


Figure 3.16: Late fusion approach where we concatenate the last layer of two unimodal models to get a joined prediction.

The last fusion strategy we have chosen is an ensemble learning approach and we selected the weighted average ensemble technique proposed by J. Borges [91]. In this approach, unimodal models are trained and added as members for the ensemble. When the ensemble receives a video to make a pain assessment prediction, it weights the contribution of each ensemble member based on its performance in the validation set. The members who performed well in the validation set are given a higher weight and therefore contribute more to the prediction. The weights are then combined to calculate a prediction that is the output of the ensemble. The search to optimise the weights is performed using a randomised search based on the Dirichlet distribution. The ensemble approach is shown in Figure 3.17. Our ensembles will consist of four members: Body CNN-BiLSTM, Body RCNN, Face CNN-BiLSTM, Face RCNN. We have chosen the weighted ensemble approach because it gives us information about the weight given to each member. In this way, we can investigate how important the body modality is for the predictions and highlight its potential importance in a bimodal setting.

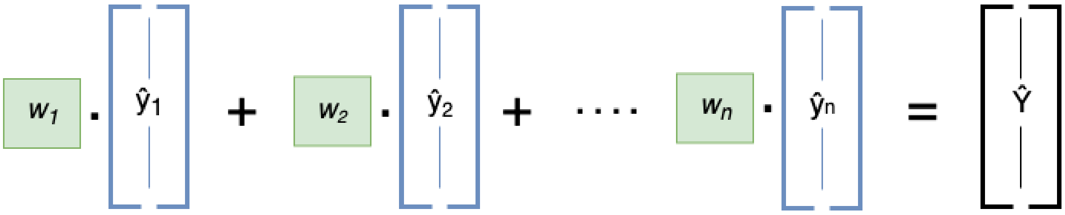


Figure 3.17: The Weighted Ensemble technique gives a weight to each of its member's prediction before combining them. **Image source:** DeepStack [91].

3.6 Reliability and Validity

Randomness is an important aspect of machine learning which makes the reproducibility of results challenging. On a practical level, this means we may have difficulty reproducing the results on different experimental runs of a model, even if we run the same script with the same data. Trying to address this reliability issue requires full visibility into the source code of the model, hyperparameter values and other details of the environment setup. We have provided this information in our report and in the public source code repository[7]. Nevertheless, what is still missing to make the experimental implementation transparent is the used dataset, which is a private video collection and therefore not publicly accessible. The other aspect of tackling the reliability issue is to control for non-determinism, which we can do by setting a random seed in our pipeline. The "seed" is a starting point for the sequence and guarantees that if we start with the same seed, we will get the same sequence of numbers. The random seed must be set in the environment, in the experimentation session, for each pseudo-random generator and for all model layers that introduce randomness, this is described in section 4. Some variance in the performance of deep learning algorithms should be expected when conducting multiple experiments. We measured this variance when performing cross-validation splits and for the most part, the variance in performance was low (< 0.1) during both training and validation. Low variance indicates that performance outcomes tend to be close to each other and ideally, the variance should not change too much from one iteration to the next.

The validity of this study depends on the quality of the data and the development and evaluation of our machine learning models. We will address relevant concerns related to ecological, internal and external validity.

Ecological validity considers how accurately our pain assessment performance reflects characteristics and behaviours in a real-world context. In this matter, we look at how the dataset was recorded and how we evaluate the models. As we are using a real-world dataset, this should minimise the risk that our performance is not transferable to a real-world setting. There is concern that pain behaviour may differ between real pain and simulated pain that occurred under artificial conditions, such as when recording a dataset in a controlled environment where participants are exposed to experimentally induced pain [49]. These concerns are not relevant in our study because participants had real pain sensations. In addition, our training and test sets do not contain the same participants and we assume that this resembles a real situation. Finally, working with sequences (full video recordings) has a higher ecological validity than working with single video frames, because in a clinical

[7] https://github.com/angelicagardner/skeleton-pose-estimation-for-pain-assessment

setting, the assessment of pain is done by observing the person over a longer period of time [43].

Internal validity concerns the extent to which our results represents the truth in the population we are studying. If we can support that our study has internal validity, we can conclude that adding body modality to automated pain assessment would improve performance for pain recognition, pain intensity estimation and pain area classification. The internal validity is highly influenced by the quality of our data. Since we are using a real-world dataset, lightning and other environmental conditions will affect the performance of the pain assessment. Two state-of-the-art toolkits were used to extract data features, but they are not necessarily the best and we should keep in mind this influence of data feature representation on performance. When Sapiński et al. [92] investigated emotion recognition using skeletal poses, they reported better results when the pose estimation tool had a lower error rate. This was the case for almost all datasets and models they explored. These results suggest that even a minor displacement in the skeletal pose can affect the recognition performance. Furthermore, Nerella et al. [17] analysed the performance of the OpenFace toolkit on a real-world dataset and reported lower performance than expected. As far as we know, the use of better feature extraction toolkits could even improve the pain assessment performance by providing more accurate feature representations. Additionally, we used techniques such as cross-validation, more than a single measure of performance (metric) and an independent test set, all of which are recommended to achieve confident and correct estimations of model performance [93]. To avoid data leaks, the training, validation and test sets contain only full videos, preventing separate frames from the same video ending up in different subsets. Mixing video frames would lead to a scenario where the model is validated or tested against data already seen because the frames are from the same video, making the results misleading but this is not an issue in our study. On the contrary, limited explainability of deep learning models makes it complicated to understand how AI models make decisions or to observe the cause and effect within a system that led it to make a particular decision. Since we do not know how the neurons in a deep learning model work together to arrive at its final decision, explainable AI methods could help characterise the transparency of the model and its expected impact to build confidence - but this explainability is missing from our study.

External validity concerns the extent to which our findings can be generalised to other contexts and whether the relationship we found between pain assessment performance and body modality persists across different individuals, settings and treatments. As we do not use the same participants for the training and test sets, this should help to ensure that our models learn general pain patterns rather than becoming accustomed to participants, and can therefore be generalised to other individuals. However, there is the problem of bias. Participants were not recorded with the purpose of this study in mind, so participant bias is eliminated. Nonetheless, our models were trained and tested on a dataset restricted to individuals of European descent in early adulthood and middle-age, with video recorded in Western industrial society. Since we provide the models with extracted body and facial features, the background environment from the videos would not have any impact. Both men and women participated in the study, but we have no explicit knowledge of the ratio. Therefore, we cannot be sure that the performance we obtained is broadly representative of the assessment of pain. For example, there is indication that face recognition algorithms perform differently in seniors with dementia [37] and that little to no response to pain is usually evident in critically ill infants [35]. Studies of pain in children do not use the Facial Action Coding System (FACS), instead there are two other systems that are similar to it [10]. Consequently, these groups cannot be generalised to our findings. This essentially underlines the importance of collecting a broad and representative sample, but in relation to a group of subjects and perhaps the type of pain. It also highlights the need for caution against over-generalising the results that automated pain assessment shows [94]. In our study, we excluded personal data unrelated to pain, such as age and gender, for privacy reasons, but consideration should be given to whether such information is necessary for trustworthy AI and for confidence in the results. To date, studies of automated pain assessment have not considered how pain behaviour may vary between genders, people from different racial and ethnic backgrounds, or context [49]. Careful expansion of our dataset to provide a broader and more representative sample across these dimensions is necessary to establish confidence in the quality of our performance and to manage predictable and unpredictable risks in using pain assessment in the real-world. Furthermore, we used three different targets for pain assessment, all of which were similarly complicated. Locating pain in the body performed slightly worse, suggesting that this setting may require additional thought. However, our datasets for pain intensity and pain area were very imbalanced and this should be a prioritised correction to focus on. Participants in our experiments perform a deep overhead squat and it is well known that this exercise is influenced by pain [83]. Nevertheless, we believe that pain assessment using skeleton pose estimation needs to be investigated for other types of exercises and movements in order to transfer the results to other situations.

We should clarify what we mean by the theoretical construct 'performance'. We describe machine learning models as "well-performing" or as having 'outperformed' another model. These terms are widely used in machine learning research and typically refer to accurate results, i.e. how correct a model was in its predictions, or a model that has both an accurate result and acceptable time complexity, i.e. the measure of how fast or slow an algorithm performs for the size of the data input. In this study, we do not consider time or space complexity, so any construct associated with 'performance' refers only to accurate predictions.

Finally, we should mention that exploratory research like ours has the disadvantage of producing an incomplete result due to its experimental focus. Therefore, it should be remembered that our performance results do not support a ready-to-deploy solution, but rather encourage the inclusion of body movements in automated pain assessment.

3.7 Ethical considerations

In this study, a private real-world dataset was used for the experiments and each participant has given their informed consent for the scientific use of their data. We only use the actual video recording and self-assessment pain answers, without unnecessary identifiers such as name, ID number or email. In addition, we took necessary privacy precautions through secure, established methods for sharing and storing the video data. We deleted all the data after this study was completed.

Other ethical considerations include the impact of automated pain assessment systems on patient well-being and care. Effective pain management is an important but sensitive indicator of the quality of patient care. Although automated systems could achieve satisfactory performance in pain assessment, their introduction into routine clinical practise has been limited due to concerns about understanding their behaviour. Especially in high-risk settings such as healthcare, understanding the decisions and limitations of various types of pain assessment models is critical to the acceptance of the technology and in that regard, there could be limitations to the trustworthiness of pain assessment results. For example, bias and fairness in a proposed system are limited to the data used and concepts such as explainability and interpretability would most likely help to improve predictive performance and assess which pain patterns the model is looking for. However, these concepts related to trustworthy AI are not readily available in more complex AI configurations such as deep learning and have not been considered in this study.

4 Implementation

In this section, we describe the implementation of the methodology introduced in the last section. Here, we present the: **(4.1)** Technical requirements, **(4.2)** Dataset setup, **(4.3)** Experimental objectives, **(4.4)** Feature extraction phase and **(4.5)** Model implementations.

The associated source code can be found in our public repository[8]. Our experiments were conducted in Jupyter Notebooks located in the **/notebooks** folder. The notebooks have descriptive names, e.g. baseline_binary.ipynb, baseline_intensity.ipynb and baseline_area.ipynb for baseline experiments. Additionally, we have attempted to modularised and organised the source code since Jupyter Notebooks can become very long otherwise. So instead, methods for data preprocessing, loading the datasets, creating the deep learning models, etc. are located in the **/src** folder as separate Python files, and are then imported into each notebook.

4.1 Technical Requirements

Machine learning experiments were conducted in Python using TensorFlow version 2.8.0 and the Keras Functional API. The models were trained using an Nvidia GeForce RTX 3090 GPU. The additional packages used in the experiments were: NumPy and Pandas data manipulation tools, Matplotlib and Scikit-plot for visualisations and the Scikit-learn library for machine learning.

To ensure reproducibility, we set a random seed in the Jupyter Notebooks. The chosen seed value can preferably be defined in a single variable so that it can be reused throughout the pipeline, let us take ***seed_value=42*** as an example. The value 42 has no special meaning, but it is commonly used in machine learning experiments. In practise, however, different seeds should be tested. Below are three suggestions for where we use the random seed:

- Dropout and similar layers that introduce randomness, e.g. Dropout(0.1, seed=seed_value)
- Python uses a random seed for its hashing algorithm so we can set the ‘PYTHONHASHSEED‘ environment variable at a fixed value:

  ```
  import os
  os.environ['PYTHONHASHSEED']=str(seed_value)
  ```

- Set the Python, NumPy and TensorFlow pseudo-random generator at a fixed value:

  ```
  import random
  import numpy as np
  import tensorflow as tf
  random.seed(seed_value)
  np.random.seed(seed_value)
  tf.set_random_seed(seed_value)
  ```

[8] https://github.com/angelicagardner/skeleton-pose-estimation-for-pain-assessment

4.2 Dataset setup

Since the dataset used for our experiments is not publicly available, we mention our setup so if another dataset was to be used, folder and file paths could be the same.

In the root folder, we have a **data** folder with two subfolders: **data/raw** and **data/processed**. In the **data/raw** folder, we have a .csv file with one row per video recording that contains the ground truth such as the participant id, video file name, pain level and pain area. The folder also contains subfolders, one per video recording, that includes the original video as well as the .csv files for skeleton pose and facial expression after feature extraction was done.

In the **data/processed** folder, we have two subfolders: **data/processed/train** and **data/processed/test**. After the feature extraction (phase 1c in our method, Figure 3.12), we divided the dataset into training and test sets to assure participants in the test set is never seen by the model during training. In turn, both **data/processed/train** and **data/processed/test** folders will contain separate folders for skeleton data and facial data that includes all .csv files with the ready datasets. We used separate folders for skeleton data and facial data to easily split between the modalities when using unimodal approaches.

4.3 Experimental Objectives

Following is a description of phase 1a implementation in our method approach (Figure 3.12) and the source code for this phase is found in the file: **/src/data/load_dataset.py**. In our experiments, we pursued three goals: Pain recognition, pain intensity estimation and pain area classification. Pain recognition is easily divided into two classes: Pain and no pain. The other two goals, however, required more effort.

For pain intensity estimation, we examined the ground-truth in the dataset for pain levels. The dataset contains numerical pain levels between 1-10 (mild to severe pain). Using a common numerical rating scale for pain intensity, we were able to divide these levels into three categories: 1-3 (mild pain), 4-7 (moderate pain) and 8-10 (severe pain).

For the classification of pain areas, we grouped the original 40+ body areas into four larger areas for easier recognition, as seen in 4.18. This strategy may sound somewhat limiting. However, it is easier to train a model with fewer classes and since this study has an exploratory purpose, experimenting was the main focus. For future work, it would be easy to extend the classes to more body areas.

4.4 Feature Extraction

We used the toolkits PoseNet and OpenFace for feature extraction. To install these models, we followed the documentation of each toolkit which lists both software requirements and how to download the models. OpenFace has description for both Unix[9] and Windows[10] installations. PoseNet, in turn, is used with JavaScript and is installed via script tags or via npm[11].

In some of our videos, the subject is standing too far away which leads to the facial area being too small for OpenFace to estimate facial action units. We solved this problem by first locating, cropping and scaling the face area before providing

[9] https://github.com/TadasBaltrusaitis/OpenFace/wiki/Unix-Installation

[10] https://github.com/TadasBaltrusaitis/OpenFace/wiki/Windows-Installation

[11] https://github.com/tensorflow/tfjs-models/tree/master/pose-detection/src/posenet#installation

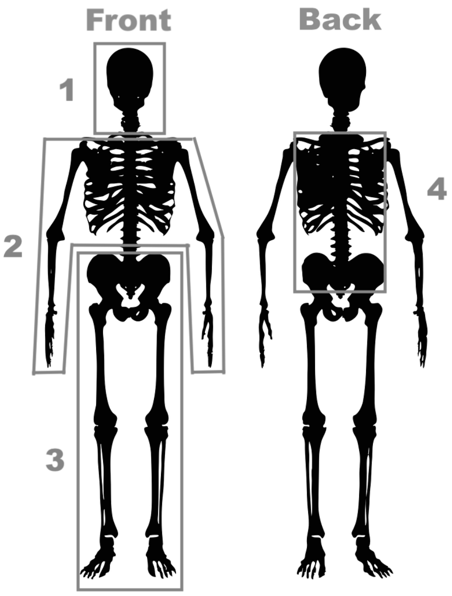


Figure 4.18: Body pain areas: (1) head and neck, (2) upper body, (3) lower body,and (4) back region.

it to OpenFace for recognition. For this task, we used the OpenCV computer vision library and the source code is found in the file **/src/data/resize_videos.py**.

The source code for the feature extraction phase is found in the file **/src/data/make_dataset.py** (implementation of phase 1c in Figure 3.12) and the resulting dataset for each modality is:

- **Skeleton pose estimation:** 35 features, consisting of confidence scores and the x- and y-positions of 12 keypoints.

- **Facial action units and head pose:** 25 features, 18 of them represent the presence of 9 pain-related facial action units (0 absent, 1 present) and the intensity of each action unit (from 0 to 5). 6 other features make up the 3D location of the head in relation to the camera and the head rotation in world coordinates with camera being the origin. The last feature is the calculated PSPI pain score.

4.5 Model Implementations

We implemented the deep learning models using the Keras Functional API which can handle complex models with the functionality we needed for our bimodal late fusion approach. When developing a model with this API, the model layers are grouped into an object with training and inference functions using the Keras Model class. Since we have three different experimental objectives, there are some slight changes to the models depending on what they are supposed to predict. The input layer and the hidden layers stay the same regardless of the experimental objective. However, the output layer needs to be adapted to the purpose of the prediction. For pain recognition (binary classification), the output layer will have 1 unit and the activation function ***sigmoid***, Dense(units=1, activation='sigmoid'). Here, we use binary cross-entropy as loss function. For the objectives of pain intensity estimation and pain area classification, we need to use an activation function and a loss function that are suitable for multiclass classification, and these are the activation function ***softmax*** and the loss function categorical cross-entropy. The number of units in the output layer is the same as the number of pain intensity or pain area classes, i.e. units=3 or units=4 respectively.

The hyperparameter optimisation phase in our method (3a in Figure 3.12) was implemented using automatic grid search. We focused on optimising six model configurations, as seen in Listing 1:

```
units = [64, 128, 256, 512]
activations = ['relu', 'tanh', 'sigmoid']
kernel_size = [3, 5]
learning_rate = [0.001, 0.0001, 0.00001]
optimizer = ['Adam', 'RMSprop', 'Nadam']
dropout = [0.0, 0.1, 0.2, 0.3]
```

Listing 1: Values tested during hyperparameter optimisation

After the hyperparameter optimisation finished, we manually transferred the best values to our Python model class files.

In addition, we applied data augmentation and regularisation techniques with the aim of improving the model performances through more data samples and better generalisation capabilities. Data augmentation (phase 3b in our method) on sequential data was indeed a challenge. Ultimately, we used two techniques from Iwana and Uchida [95]: Jitter and Rotation. The authors created a library for augmenting time series data[12] in Python that works with Keras, which we have used. Jitter is a way to simulate additive noise, and rotation mirrors the axis. In our public repository, we find this library at the location **/src/lib/time_series_augmentation**.

To increase model generalisation capacity, the following phase in our method was 3c Regularisation (Figure 3.12). To regularise the models, we used two techniques: Dropout and Batch Normalisation. Dropout is an excellent regulariser that is easy to implement and compatible with many models. Batch normalisation is a technique for standardising inputs to a neural network, but it has a regularisation effect. Sometimes, batch normalisation is reducing the generalisation error to such an extent that dropout can be omitted. However, we have found that using Dropout with a small value of 0.1 still has a positive effect on the performance of our models.

Finally, we trained the models in preparation for running our last experiments (method phase 4). During model training, we used a batch size of 32 and the training runs for up to 100 epochs. The learning is done in a way that minimises the loss in the validation set and we follow an early stopping strategy where we terminate the training early if the model no longer improves. In case we had to stop the training early, the model's best weights were restored. We set the k-fold cross-validation (CV) to ***k=3*** since we noticed that training performance started to decrease with higher values of ***k***. Since we have a training set of 820 videos, a k-fold CV with k=3 will use 547 videos for training and 273 videos for validation in each fold. A batch size of 32 means that the training set is divided into 23 batches, each batch containing 32 videos. For one epoch, the model goes through 23 batches of 32 videos each, i.e. 23 updates of the model after learning from 32 video samples. At

[12] https://github.com/uchidalab/time_series_augmentation/

100 epochs, the model has gone through the entire training set 100 times per fold, and is validated 10 times throughout the training session.

There is an imbalance between classes in the experimental objectives of pain intensity estimation and pain area classification where a some classes contain very few samples. Furthermore, the dataset samples without pain (the ’no pain’ class) exceeds all pain states. Therefore, we decided not to include a "no pain" class for either pain intensity estimation or pain area classification as we did not want to cause more imbalance. Accordingly, all data samples where the participant felt no pain were also excluded from these objectives. Additionally, imbalanced datasets need to be handled in some way. For sequential data, it is common to use weighted losses in training. The Scikit-learn library has a function ***compute_class_weight*** that estimates weights for each class in the dataset. For instance, in predicting the pain intensity estimate, the class distribution is as follows:

```
Mild Pain:       219
Moderate Pain:   158
Severe Pain:     3
```

This class distribution is before we apply data augmentation on the minority classes. However, we can see that there are only three original video samples of severe pain in the dataset. Classes with few samples do not contribute significantly to the model error and tend to be ignored during training [96]. A simple yet effective way to solve this problem is to assign an appropriate weight to each class. A larger weighting value increases the importance of that specific class during training, which will hopefully improve the detection performance of minority classes.

Similarly, we attempt to correct the imbalanced pain area classes:

```
Lower Body:      245
Back Region:     73
Upper Body:      57
Head and Neck:   5
```

We provide the deep learning models in separate Python source code files for easy import into the notebooks. Here, we plot them using neural network model graphs to provide an overview of their layer architectures. The CNN-BiLSTM model is found in the file **/src/models/CNNLSTM.py** and the model architecture is seen in Figure 4.19. In this architecture, we see the Convolutional, Maxpooling and Bidirectional LSTM layers, as described earlier. There are also Batch Normalization and Dropout layers for regularization capabilities, Fully-Connected (Dense) layers, and a TimeDistributed wrapper around some layers to allow for applying temporal slices of an input.

The late fusion version of the CNN-BiLSTM model has two unimodal paths where one will receive the body data as input, and the other receives the face data as input. These models are concatenated at the last layer before output, as seen in Figure 4.22. This model is located in the file **/src/models/CNNLSTMfusioned.py**.

The Recurrent CNN model (RCNN) is found in the file **/src/models/RCNN.py**. In Figure 4.21, we can see the stacks of recurrent connections as well as batch normalization and dropout layer for regularization purposes. We have also concluded this architecture with Fully-Connected (Dense) layers.

Equivalently, the late fusion version of the RCNN has two unimodal paths that are concatenated at the last layer before output. The model architecture is located in the file **/src/models/RCNNfusioned.py** and can be seen in Figure 4.22.

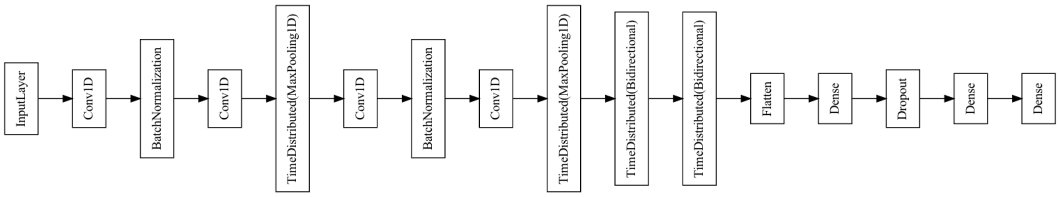

InputLayer
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
TimeDistributed(Bidirectional)
TimeDistributed(Bidirectional)
Flatten
Dense
Dropout
Dense
Dense

Figure 4.19: The final CNN-BiLSTM model with the input layer farthest to the left.

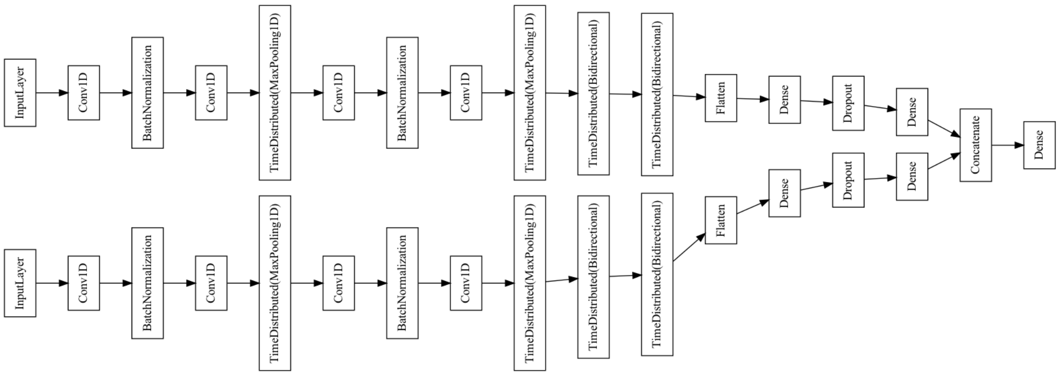

InputLayer
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
TimeDistributed(Bidirectional)
TimeDistributed(Bidirectional)
Flatten
Dense
Dropout
Dense
InputLayer
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
Conv1D
BatchNormalization
Conv1D
TimeDistributed(MaxPooling1D)
TimeDistributed(Bidirectional)
TimeDistributed(Bidirectional)
Flatten
Dense
Dropout
Dense
Concatenate
Dense

Figure 4.20: The CNN-BiLSTM late fusion model using concatenation of two models into a single output.

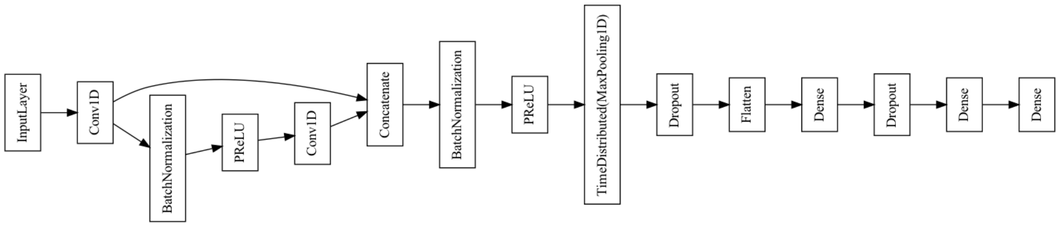


Figure 4.21: The RCNN model with the input layer farthest to the left.

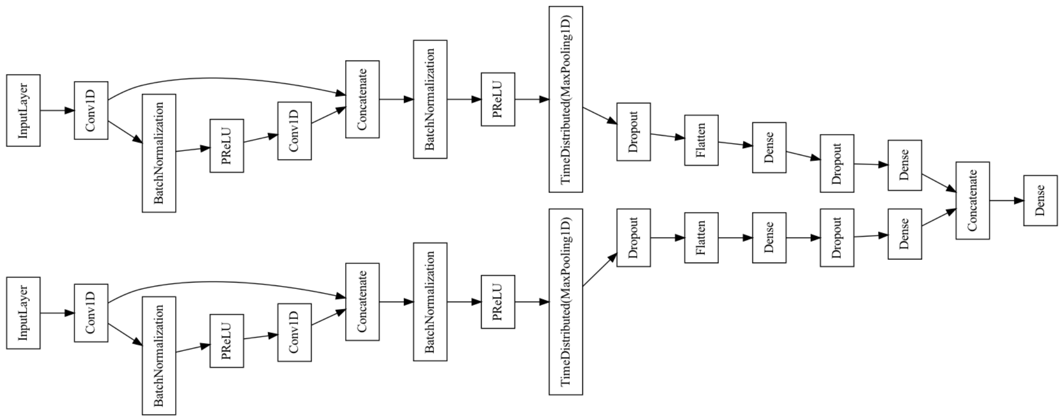


Figure 4.22: The RCNN late fusion model using concatenation of the two models into a single output.

Our bimodal Early Fusion approach does not require a separate model implementation since it is using the same DL models as the unimodal approaches. Instead, we have a function for loading and concatenating the early fusioned dataset which is returned with 60 features (35+25).

Finally, we used a Python framework called DeepStack[13] to create the Weighted Average Ensemble. This framework has a wrapper class for Keras models that adds them as ensemble members and the code for creating this class, adding its member and estimating the member weights can be seen in Listing 2:

```
dirichletEnsemble = DirichletEnsemble()
dirichletEnsemble.add_member(cnnlstm_body_member)
dirichletEnsemble.add_member(rcnn_body_member)
dirichletEnsemble.add_member(cnnlstm_face_member)
dirichletEnsemble.add_member(rcnn_face_member)
dirichletEnsemble.fit()
```

Listing 2: Code for creating the ensemble, adding members and estimating member weights.

The ensemble is fit on the training and validation data to calculate each member's performance on the validation set. Based on these performances, it will assign weights to be used for member predictions on new samples. The DeepStack framework is available via pip (the Python package manager), however it only supports TensorFlow version 1. We are using TensorFlow version 2, Keras now comes packaged together with TensorFlow and is imported as **tensorflow.keras.** Therefore, we had to download the DeepStack framework source code and update a few keras import statements for it to work with our environment. In our public repository, the updated DeepStack source code that supports TensorFlow version 2 can be located at **/src/lib/DeepStack**.

[13] https://github.com/jcborges/DeepStack

5 Evaluation

In this section, we present model performances from unimodal and bimodal experiments in subsections for the three experimental objectives: **(5.1)** Pain recognition, **(5.2)** Pain intensity estimation and **(5.3)** Pain area classification. In each subsection, we discuss model performances for our selected metrics: accuracy (or balanced accuracy), AUC, precision, recall and F1 score. We also review some interesting results from previous studies on automated pain assessment to examine the extent to which they agree with or differ from our results. Finally, we have a subsection specifically for **(5.4)** Ensemble performances, where we consider body movement contributions.

First and foremost we consider the baseline performance from our simple LSTM model in our method phase 2 (Figure 3.12). Both the unimodal body and late fusion bimodal approaches obtained interestingly good baseline performance. Baseline models commonly lack complexity and have little predictive power, however in this case, the simple LSTM model seems to have been a decent choice. As can be seen in Appendix A, the unimodal body approaches achieved AUC values of 0.63, 0.78 and 0.64 for pain recognition, pain intensity estimation and pain area classification, respectively. The bimodal late fusion approaches obtained AUC values 0.69, 0.7 and 0.65. As for unimodal face and bimodal early fusion, they did obtained a similarly good performance for pain recognition, however, did not reach a good performance for the other two objectives. The unimodal face and early fusion approaches achieved 0.31 and 0.4 for pain intensity estimation and 0.27 and 0.39 for pain area classification.

In the reviewed studies of automated pain assessment, Bargshady et al. [45] performed best with their Ensemble based on multiple CNN-BiLSTM models trained on facial expressions only (AUC 0.9367), followed by Rodriguez et al. [46] VGG-LSTM model also trained on facial expressions (AUC 0.933), and Salekin et al. [10] with a late Fusion CNN-LSTM trained on face, body, audio and physiological signals (AUC 0.9). Most of the other reviewed studies had similar performance to ours, and some even had lower performance. In our study, Ensemble was the best approach for pain recognition (AUC 0.71), Unimodal Body CNN-BiLSTM for estimating pain intensity (AUC 0.75), and Late Fusion RCNN for pain area classification (AUC 0.75). As we can see, our results do not reach the best performance from previous studies nor does it achieve >80% accuracy which was the accepted pain detection truthfulness mentioned for healthcare staff participants in a study by Walter et al [36]. Nevertheless, the focus of our study was not to fully optimise and increase model performance, but rather to keep an exploratory purpose.

Furthermore, Walter et al. [65] was the only pain assessment study we reviewed that reported seeing no advantage of multimodal approaches over unimodal approaches. They combined facial expressions and head posture from videos with three physiological signals. In their results, the unimodal and multimodal approaches were very close in pain detection performance and differed only slightly. When reading their study, we suspected that the reason might be that their technical solution was not powerful enough, as most other studies show how multimodal approaches outperform all unimodal approaches. However, in our results, we also find that the performance of pain prediction is very similar between unimodal and bimodal approaches for all three objectives. This could be due to several reasons, such as the choice of an inappropriate model architecture for the problem domain, or that the features in the two modalities are highly correlated and do not contain enough complementary information. Nevertheless, we still saw an advantage of bi- and multimodality. During our experimentation, a scenario occurred where the face was too small to accurately measure facial expressions in several videos. This did not have a major impact as we instead preprocessed the videos to crop and resize the facial area before using the facial extraction tool, which in turn facilitated the detection and extraction of facial action units (AUs). However, the problem still existed in 30 of the video samples where preprocessing did not help. Thus, in these cases, it might be beneficial to use a bi- or multimodal approach so the system can still give a reasonable prediction when one modality is missing.

Additionally, Peeters et al. [97] demonstrated a low correspondence between facial expressions and pain-related body stiffness, suggesting that pain-related facial and body expressions often occur separately. In our study, this would suggest that bimodal fusion strategies for our modalities (face and body) would preferably occur at later stages, i.e. late fusion or ensemble learning. We could indeed see some tendency that late fusion approaches performed better than early fusion, however, these results were still subtle.

5.1 Pain Recognition

Pain recognition was theoretically the easiest objective of all three to achieve good model performance because it was approached as a binary classification task, i.e. to determine whether a person felt pain or not. With this division of the dataset, the distribution between the two classes was more balanced than for the other objectives. Therefore, it was interesting to note how challenging performance improvement was. Although we used two different modalities, two different architectures, three different bimodal strategies and tried to optimise model performance by hyperparameter optimisation, the improvements obtained were mostly not significant and the performance was not much better than the simple baseline. This shows that model improvements are not always strictly additive.

The model results for pain recognition can be found in the Table 5.7. The Early Fusion CNN-BiLSTM received fairly low performance: 43.28% accuracy and 0.52 AUC. From its low recall (0.39) but high precision (0.88), we can understand that the model seems to be mostly guessing that all people are pain-free. Since its AUC value is also low and close to random (0.52), we can understand that the model performance is not going to improve even if we fit it to another threshold. Other models that faced a similar scenario was Body RCNN, Face CNN-BiLSTM, Early Fusion RCNN and Late Fusion RCNN. By investigating the performance metrics, it seems like CNN-BiLSTM was overall a better model architecture choice for pain recognition with the best-performing model being the Late Fusion CNN-BiLSTM with performances: 83.96% accuracy, 0.68 AUC, 0.89 precision and 0.93 recall.

Model	Accuracy	AUC	Precision	Recall	F1
Body CNN-BiLSTM	69.34%	**0.7**	0.92	0.71	0.92
Body RCNN	43.43%	0.6	0.92	0.37	0.92
Face CNN-BiLSTM	52.61%	0.64	0.92	0.49	**0.93**
Face RCNN	62.31%	0.62	0.91	0.63	**0.93**
Early Fusion CNN-BiLSTM	43.28%	0.52	0.88	0.39	**0.93**
Early Fusion RCNN	51.12%	**0.7**	0.93	0.47	**0.93**
Late Fusion CNN-BiLSTM	**83.96%**	0.68	0.89	**0.93**	**0.93**
Late Fusion RCNN	53.36%	0.63	**0.96**	0.48	**0.93**

Table 5.7: Model results for all performance metrics in pain recognition. Best results are marked in bold.

For model comparison, we primarily use AUC values as can be seen in Figure 5.23. The CNN-BiLSTM models are coloured blue and the RCNN models are coloured orange. In the figure, we can see the unimodal approaches on the left and the bimodal fusion approaches on the right. Model performances for pain recognition vary between AUC 0.52-0.7. Here we can easily see that apart from the early fusion approach, the CNN-BiLSTM architecture outperforms the RCNN approach for pain recognition. The highest AUC value of 0.7 was achieved by Unimodal Body CNN-BiLSTM and Early Fusion RCNN. However, in Table 5.7 we can see that the Unimodal Body CNN-BiLSTM achieved much better accuracy and precision/recall than Early Fusion. It demonstrates that another threshold would have been better suited for the Early Fusion model, and this situation is a good example of why it might be necessary to examine multiple performance metrics for machine learning models.

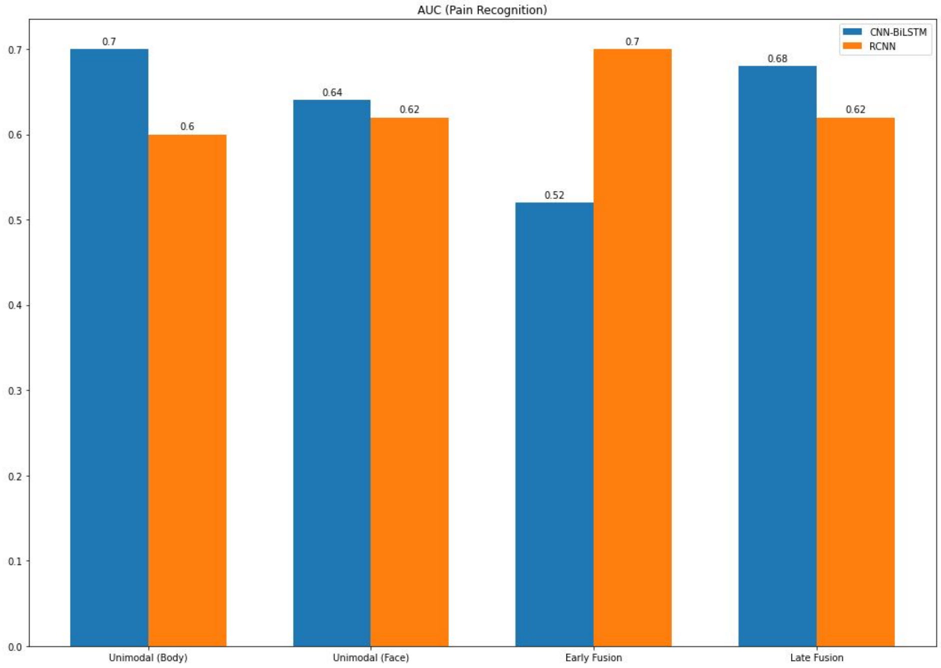


Figure 5.23: AUC for Pain Recognition.

When improving performance is a challenge it could indicate that the problem is difficult to solve. Therefore, we looked for promising signs of models learning pain patterns from the data during training. In Figure 5.24, our unimodal body RCNN received between 0.81-0.87 during training, but only 0.7-0.74 during validation and even lower 0.6 during testing. This is a common scenario to be expected in ML experiments and it suggests that the model simply needs to get better at generalising to unseen samples.

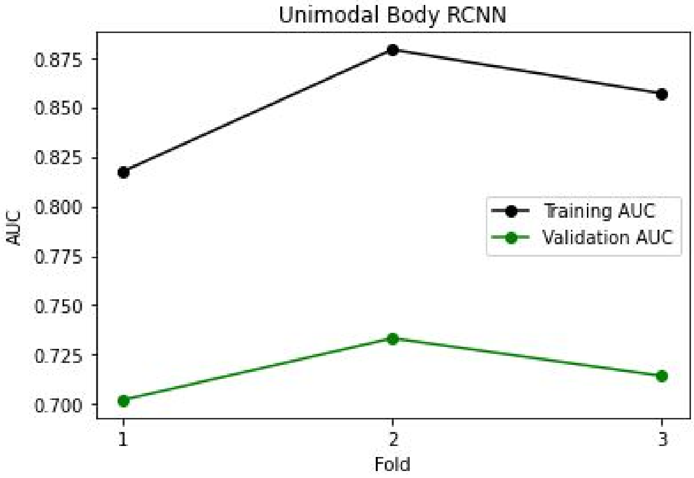


Figure 5.24: Training and validation performance for pain recognition with the unimodal body approach when k-Fold CV k=3.

We also looked at the loss curve of the same model to see that the prediction errors are gradually decreasing (Figure 5.25).

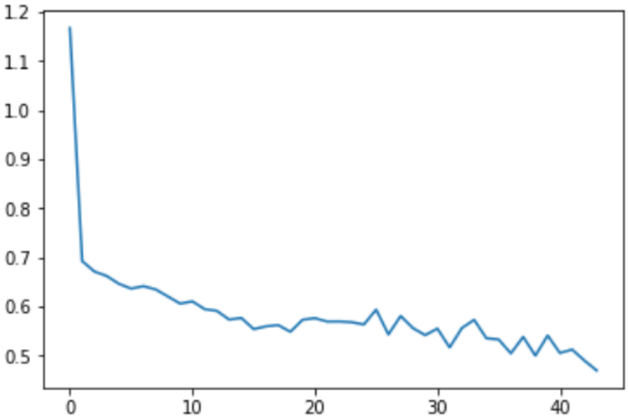


Figure 5.25: Unimodal body RCNN loss curve.

Another good indication of promising pain recognition performance is to look at the confusion matrix. Figure 5.26 shows how the unimodal body CNN-BiLSTM made its predictions and where it made misclassifications. The darker the blue shade, the more predictions were made in that category. We can see that most predictions were made when a person felt pain and the model classified this as a painful experience (TP - true positive). The second largest prediction category was when a person felt no pain and the model classified this as no pain (TN - true negative). Although this result can still be improved, it looks like the model is learning to distinguish pain patterns.

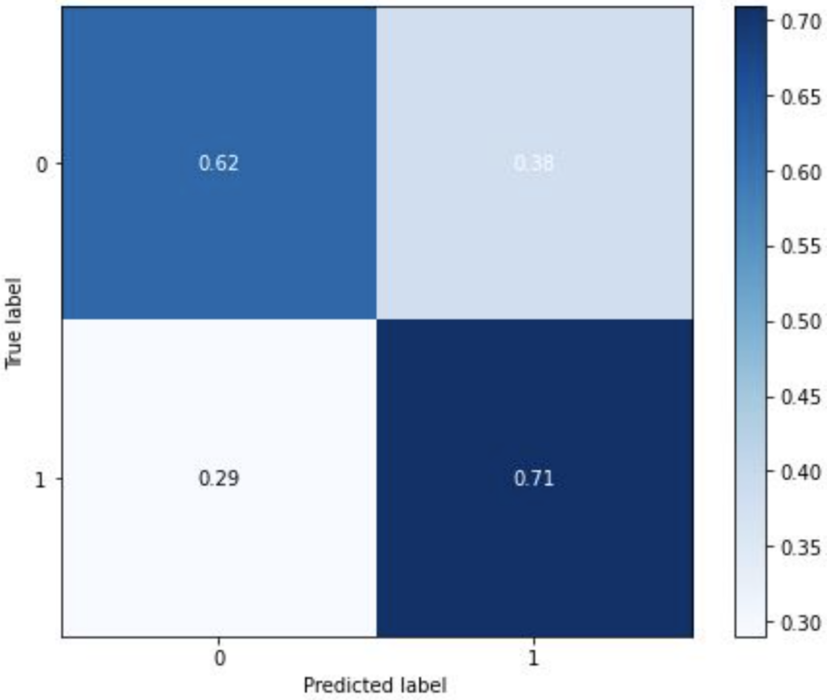


Figure 5.26: Confusion Matrix of Unimodal Body CNN-BiLSTM.

Consequently, we consider it likely that the models need better generalisation capabilities on unseen data. We base this assumption on the promising training results and on the confusion matrices showing model predictions. Typically, better model generalisation is achieved by applying some of the regularisation techniques we performed in method phase 3c (Figure 3.12), although they were limited in scope. When the generalisation techniques do not work, as in our case, it usually means that more data needs to be collected. This reasonates with us because although we have a reasonable amount of videos, pain experiences are very subjective and most people have individual responses to some degree. This might require more data for the model to learn not only the patterns but also the variation between how individuals show them.

5.2 Pain Intensity Estimation

Pain intensity estimation was approached as a multiclass classification task with three pain levels: mild, moderate and severe pain. Model performances for this objective can be seen in Table 5.8 where we can acknowledge the moderately high values of AUC but apparent lower values for the other metrics. These results suggest that another threshold should have been chosen for the pain intensity estimation to retrieve better model performances.

Model	**Balanced Accuracy**	**AUC**	**Precision**	**Recall**	**F1**
Body CNN-BiLSTM	30.30%	**0.75**	**0.54**	**0.52**	**0.29**
Body RCNN	24.01%	0.71	0.44	0.43	0.24
Face CNN-BiLSTM	36.36%	0.66	0.35	0.34	0.25
Face RCNN	19.91%	0.57	0.20	0.21	0.11
Early Fusion CNN-BiLSTM	38.10%	0.65	0.38	0.39	0.26
Early Fusion RCNN	34.20%	0.65	0.33	0.32	0.24
Late Fusion CNN-BiLSTM	**41.56%**	0.70	0.41	0.41	0.27
Late Fusion RCNN	22.94%	0.58	0.22	0.20	0.14

Table 5.8: Model results for all performance metrics in pain intensity estimation. Best results are marked in bold.

From these results, the best model for pain intensity was the Unimodal Body CNN-BiLSTM. This result is noteworthy because it shows the value of body movement data in automated pain assessment, but we were still expecting a better performance from facial data. Since we used a real-world dataset, the faces in our videos were occasionally too small for the OpenFace facial toolkit to capture facial expressions. OpenFace is trained on videos recorded in a controlled environment, and therefore had difficulty recognising facial action units (AUs) when our videos did not match the same conditions. We believe that to some extent this may explain the low performance of our results when using the face modality.

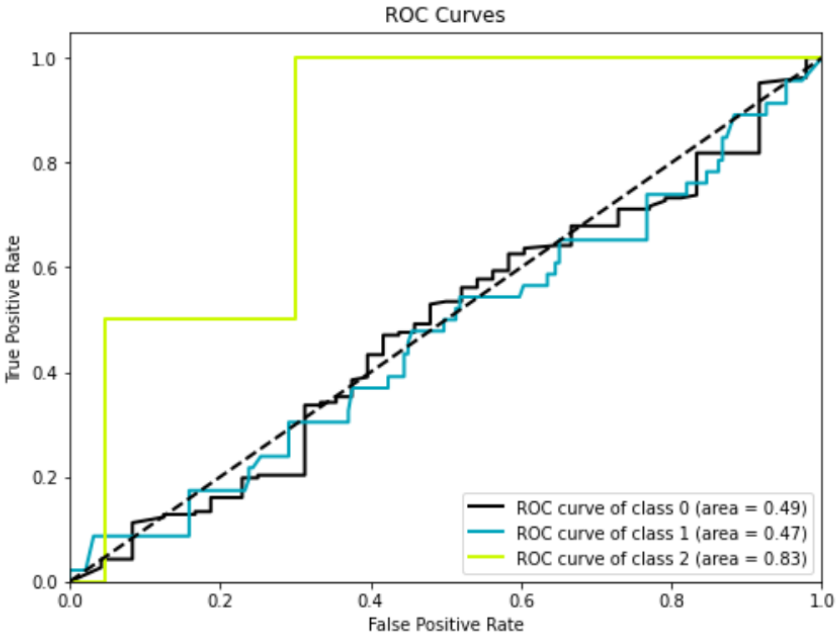


Figure 5.27: ROC curve for the Unimodal Body RCNN. A model with excellent performance would reach the top left corner.

From previous work on pain assessment, there are two studies that suggest pain intensity estimation does not work well for lower levels of pain [5], [53]. Egede et al. [5] demonstrated baseline performance where detection of low pain was significantly worse than classification of no pain and high pain. Werner et al. [53] also suggested that multimodality, in particular, does not work well for low pain intensity. In our study, we did not generally see the same pattern. The models seemed to have more difficulty in predicting the "severe" pain intensity class, which we believe is more related to the problem of an imbalanced dataset than to the actual pain intensities. However, one case where this pattern was visible is in the unimodal body approaches. As can be seen from the ROC curve of the Unimodal Body RCNN in Figure 5.27, it performed poorly in detecting classes 0 and 1, i.e. mild and moderate intensity, but better in class 2 (severe pain). Could this indicate that pain-related body movements are more clearly detected when pain is more intense?

For pain intensity estimation, model performances ranges between AUC 0.57-0.75 and it does not seem to be as large difference between performances for the CNN-BiLSTM and RCNN architectures with this objective, even though CNN-BiLSTM still demonstrates as the better architecture choice (Figure 5.28).

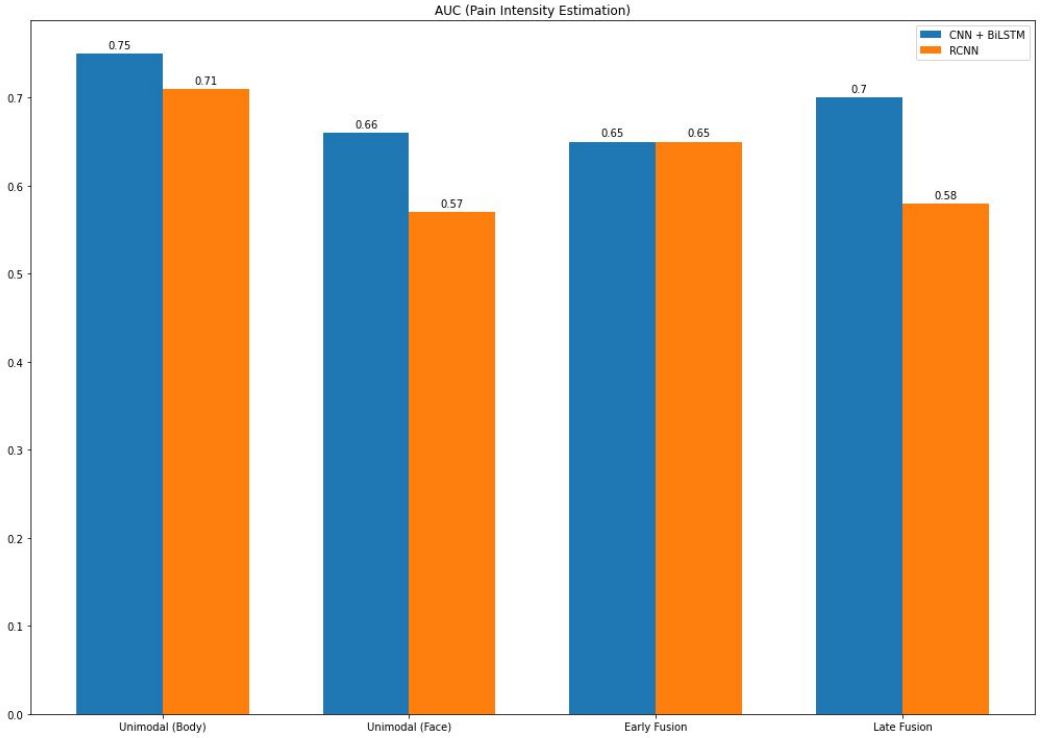


Figure 5.28: AUC for Pain Intensity Estimation

5.3 Pain Area Classification

We approached pain area classification as a multiclass classification task, predicting pain areas from four classes: Lower body, back region, upper body, head and neck region. Model performances for this objective can be seen in Table 5.9. As we are not aware of any pain assessment study that focus on locating pain areas in a multiclass setting, it is interesting to note that this objective was not particularly more difficult to assess than pain recognition or pain intensity estimation. However, just like with pain intensity estimation, our default threshold of 0.5 does not seem to have been the best choice for this objective if we compare AUC values with the other performance metrics. This would mean that for a more suitable threshold, model performances would be more satisfactory.

Model	Balanced Accuracy	AUC	Precision	Recall	F1
Body CNN-BiLSTM	23.75%	0.67	0.33	0.01	0.23
Body RCNN	24.00%	0.63	0.34	0.26	0.21
Face CNN-BiLSTM	28.14%	0.61	0.29	0.16	0.21
Face RCNN	28.57%	0.64	0.30	0.16	0.22
Early Fusion CNN-BiLSTM	27.27%	0.62	0.26	0.13	0.19
Early Fusion RCNN	16.88%	0.60	0.20	0.07	0.07
Late Fusion CNN-BiLSTM	35.93%	0.67	0.35	**0.29**	**0.24**
Late Fusion RCNN	**50.65%**	**0.75**	**0.55**	0.20	0.19

Table 5.9: Model results for all performance metrics in pain area classification. Best results are marked in bold.

When comparing model performances for pain area classification in Figure 5.29, AUC results vary between 0.6-0.75 and the best performing model is Late Fusion RCNN.

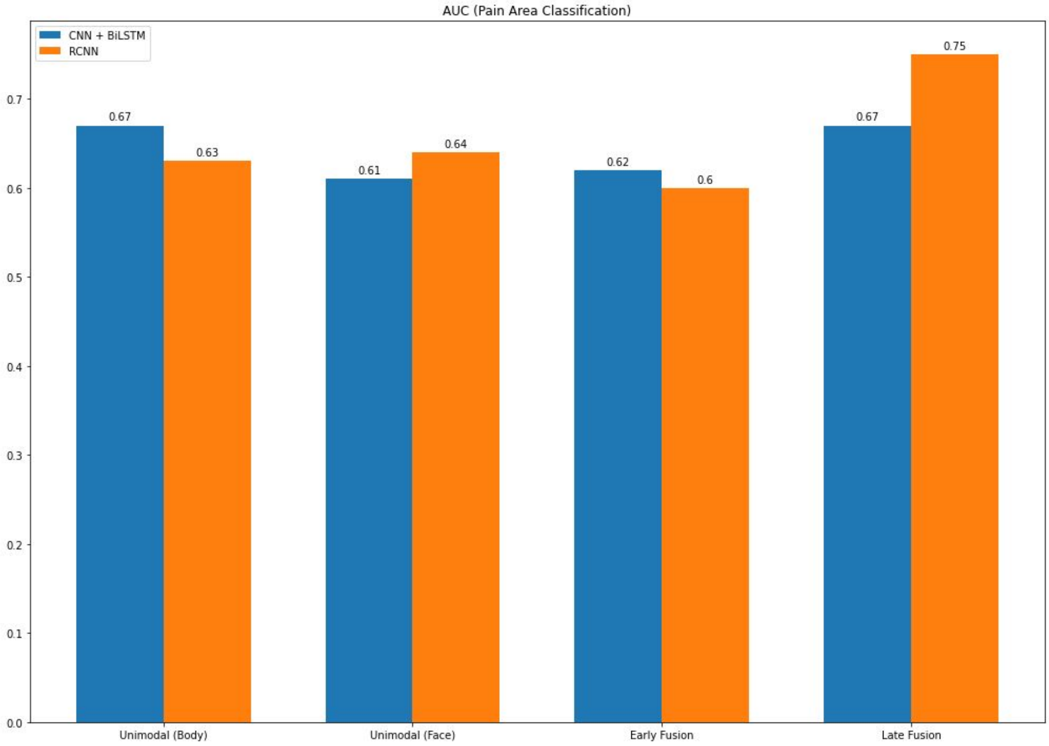


Figure 5.29: AUC for Pain Area Classification

Inspecting the confusion matrix of the Late Fusion RCNN for body area classification (Figure 5.30), we notice a few interesting aspects. First, the model could not predict any samples from class 1 correctly (Head and neck region). This is most probably an error related to the imbalance in the dataset and too few samples from this region in the training set. Second, although the model gets the majority of the other class predictions correct, it confuses class 0 (back region) with class 3 (upper body). This could be because back region and upper body might be areas demonstrating similar pain patterns in an overhead deep squat exercise, or it could be that participants had pain in both of these areas. Considering the latter aspect, we presume another interesting aspect would have been to explore classification of pain areas with multiple outputs, i.e. multioutput classification. A machine learning model that is shown samples where a person has pain in multiple body areas will not learn optimally if the participant shows indications of pain in multiple areas but at the same time, the model can only output one class (area). This could be an interesting perspective to analyse.

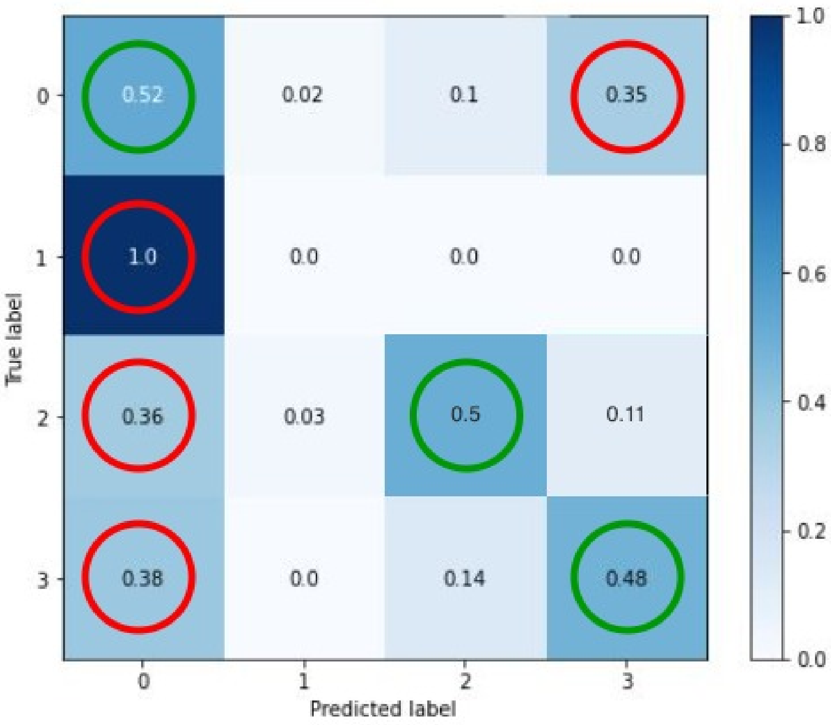


Figure 5.30: Confusion Matrix for Late Fusion RCNN for pain area classification. The darker shades should be located in the diagonal for good model performance.

Furthermore, even though the Late Fusion RCNN was the best-performing model for pain area classification, this model has a higher precision than recall which might not be well suited for all problem domains. Machine learning experiments commonly aim to maximise both precision and recall values even though they usually come with a tradeoff, but in certain scenarios, one might be more important than the other. For instance, in a rehabilitation setting where missing a painful experience could be worse than manually processing a few incorrect (false) pain predictions, a higher recall (the true positive rate, TPR) would be more important than precision. But the opposite could also be true for some other scenario. For example in a healthcare setting where it would be more important to not prescribe unnecessary pain medication, and we would therefore want to minimise the false positives (FP) by getting a higher precision score.

By looking at the Precision-Recall curve for the Late Fusion RCNN, we can notice if this model had trouble classifying certain pain areas. In Figure 5.31, the black line represents the back region, the blue line represents the head and neck region, the green line represents the lower body, and the yellow line represents the upper body. The blue line located at the bottom of the PR-curve demonstrates that this model has no capability to classify pain area if it is located in the head and neck region, as we could also see from the confusion matrix (Figure 5.30). The model shows the best performance for classifying lower body pain (green line). The model mixup between back region and upper body region likely contribute to the lower PR-curve performance for these two body areas.

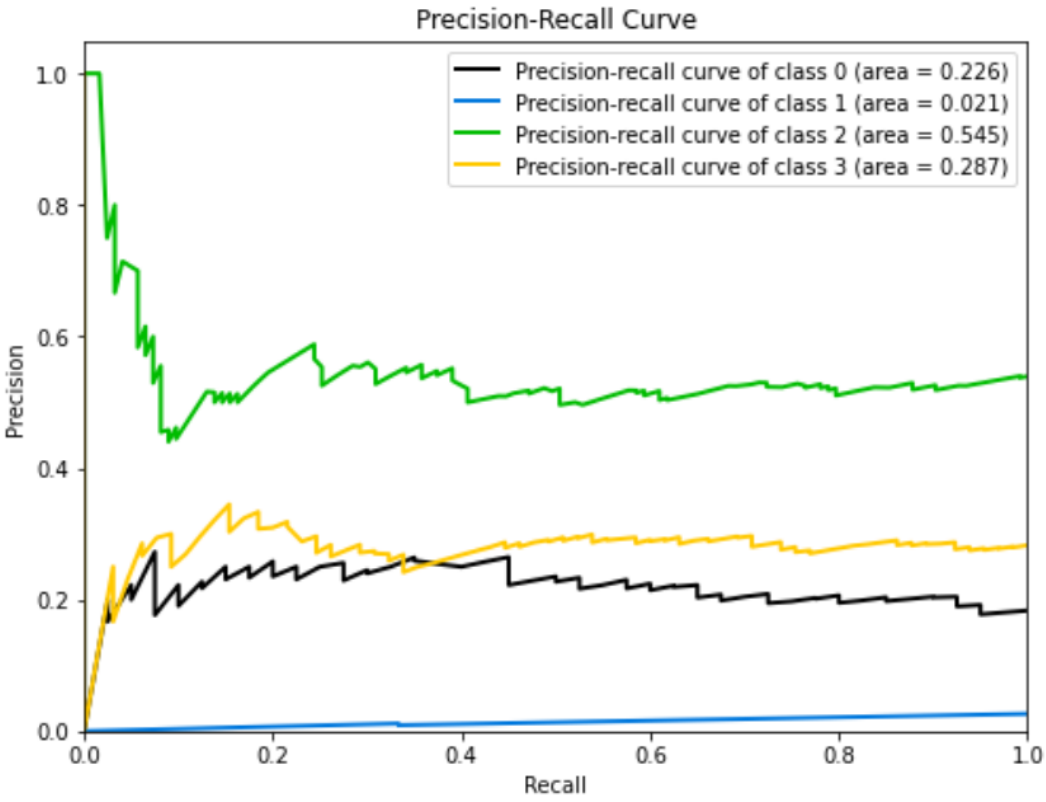


Figure 5.31: RP-curve for the Late Fusion RCNN for pain area classification. A model with excellent performance would reach the top right corner.

5.4 Ensembles

Finally, we look at the ensemble performances summarised in Table 5.10. We had three Ensembles, all with the same Weighted Average ensemble approach, and they contain the same type of members: Unimodal Body CNN-BiLSTM, Unimodal Face CNN-BiLSTM, Unimodal Body RCNN, Unimodal Face RCNN. What differs the ensembles is the objective they were trained towards: pain recognition, pain intensity estimation or pain area classification.

Ensemble performance was mostly unimpressive in our study except for the pain recognition ensemble. It obtained full score for all metrics except AUC, which indicates that this model performed well for our selected threshold, but would not perform the same for all other possible thresholds.

Metric	Pain Recognition	Pain Intensity	Pain Area
Accuracy	**100.00%**	53.68%	39.39%
AUC	**0.71**	0.61	0.54
Precision	**1.00**	0.40	0.31
Recall	**1.00**	0.38	0.30
F-1 Score	**1.00**	0.30	0.26

Table 5.10: Performance metrics for the three different Ensembles.

When evaluating the ensemble performances, we looked at the weighting of the different ensemble members. We are particularly interested in how the unimodal body approaches contributed to the ensemble predictions.

For pain recognition, it seems the ensemble favoured both the body and face CNN-BiLSTM approaches. The weights of the members were distributed as:

```
Body CNN-BiLSTM (weight: 0.6587)
Face CNN-BiLSTM (weight: 0.2500)
Body RCNN (weight: 0.0899)
Face RCNN (weight: 0.0014)
```

For pain intensity estimation, the unimodal face CNN-BiLSTM was given the primary weight:

```
Body CNN-BiLSTM (weight: 0.0037)
Face CNN-BiLSTM (weight: 0.8544)
Body RCNN (weight: 0.1301)
Face RCNN (weight: 0.0117)
```

For pain area classification, it was again the unimodal face CNN-BiLSTM model that received the most weight for the predictions:

```
Body CNN-BiLSTM (weight: 0.0928)
Face CNN-BiLSTM (weight: 0.8981)
Body RCNN (weight: 0.0073)
Face RCNN (weight: 0.0019)
```

Comparing the ensemble performances and what members they have prioritised, we can observe that when the ensemble divided is weights more evenly between the two modalities, it achieved a better performance than when it only favoured its facial member. This could be seen as an indication of the body modality’s contribution to bi- and multimodal approaches.

6 Conclusions and Future Work

In this study, we investigated the application of skeleton pose estimation for automated pain assessment. Experiments were conducted using deep learning models in unimodal and bimodal approaches with body movements and facial expressions. Body movements were represented by skeleton pose estimation and facial expressions by facial action units (AUs) and head pose. The experiments had three objectives: Pain detection, pain intensity estimation and pain area classification. In this way, our study highlighted two of the future goals for the development of automated pain assessment: exploring the importance of pain-related body movements and assigning the location of pain in the body.

Our study exemplified how challenging the task of pain assessment using skeleton pose estimation could be. It was difficult to increase assessment performance beyond baseline, even though we experimented with complex architectures, hybrid CNN-BiLSTM and recurrent CNN (RCNN), as well as three different bimodal strategies. Nevertheless, our results look encouraging, with a unimodal body approach as the best model for pain intensity estimation (AUC value of 0.75) and promising unimodal body performance in pain recognition (0.7) and pain area classification (0.67). These results indicate the performance we can achieve in pain assessment based on body movements **(RQ1)**. The experimental objective of pain area classification was not particularly more challenging compared to the other two objectives (pain recognition and intensity) that has already been studied by previous research. When classifying pain area, we found that the inclusion of body modality provided an assessment opportunity for this localisation **(RQ2)**. Although pain area classification was slightly more difficult than the other two objectives, the models performed decently, especially the late fusion of body and face data with an AUC of 0.75. We observed that the models had an issue with predicting head and neck pain, which we relate to an imbalance in our pain area dataset. We attempted to address this problem by increasing the sample size through data augmentation and using class weights during the training process. However, this did not seem to have the hoped-for effect and other methods of handling imbalanced datasets should be tested. A similar imbalance also occurred between pain levels when estimating pain intensity. The difference between AUC scores and other performance metrics for these two objectives of pain intensity and pain area suggests that the models show promising performance but our selected threshold of 0.5 was not optimal. For pain intensity estimation, the best-performing model reached an AUC value of 0.75 but balance accuracy of 30.30%, and for pain level classification, the best-performing model reached an AUC value of 0.75 but balanced accuracy of 50.65%.

In reviewing the ensembles and member contributions, we found that ensemble performance improved when the body modality was given significant importance and priority when making predictions. This was the case for the pain recognition ensemble, which gave ≈0.65 weight to a unimodal body member and ≈0.25 weight to a unimodal face member, reaching an AUC of 0.71. Compared to the pain intensity and pain area ensembles that gave the most weight to unimodal facial approaches, they merely achieved an AUC of 0.61 and 0.54. These results suggest that the inclusion of body movement improves performance and should be considered as an additional source of information in bi- and multimodal pain assessment approaches **(RQ3)**.

Our exploratory results could provide a basis for future research on pain assessment using skeleton pose estimation and encourage investigation of the research objective of pain area classification. They can also be seen as confirmation of previous pain assessment studies that used skeletal representations of body movements. However, since our performance does not reach the best results of previous studies (AUC 0.9367) nor the acceptable level for clinical use (> 80%), we see our results as promising but with a need for model generalisation improvement.

In the future, we would like to explore the classification of pain areas with multiple outputs (multioutput classification), as some participants may have pain in multiple body areas. Other combinations of multimodal approaches should also be investigated, e.g. using physiological signals together with body movements and facial expressions, or videos with audio. In such cases, the dataset would need to include more than the video modality we have used. Training and testing on different datasets would also be a good way to evaluate the generalisability of our results. Until generalisability between different databases is investigated, it is not known whether our conclusions would be applicable in other situations. Given the importance of data input features, explainable AI could help to understand feature importance and why models make certain predictions. In this way, we might perceive how to improve data representation for the different modalities, especially body movements that lack a formal pain-related standard. There are several approaches to explainability, for instance building a surrogate model or through techniques such as LIME and layer-wise relevance propagation. Furthermore, our study was limited in terms of exploring model architectures and fusion strategies and these aspects could be further investigated, as well as performing more comprehensive model optimisation. It would also be interesting to explore when a person only feels pain at a certain stage of the video and if it matters for the model predictions how long the duration of the painful feeling lasted?

References

[1] P. Werner, D. Lopez-Martinez, S. Walter, A. Al-Hamadi, S. Gruss, and R. Picard, “Automatic recognition methods supporting pain assessment: A survey,” *IEEE Transactions on Affective Computing*, pp. 1–1, 2019.

[2] T. A. Olugbade, N. Bianchi-Berthouze, N. Marquardt, and A. C. de C. Williams, “Human observer and automatic assessment of movement related self-efficacy in chronic pain: From exercise to functional activity,” *IEEE Transactions on Affective Computing*, vol. 11, no. 2, pp. 214–229, 2020.

[3] T. Hassan, D. Seuß, J. Wollenberg, K. Weitz, M. Kunz, S. Lautenbacher, J.-U. Garbas, and U. Schmid, “Automatic detection of pain from facial expressions: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 6, pp. 1815–1831, 2021.

[4] S. Cao, D. Fu, X. Yang, P. Barros, S. Wermter, X. Liu, and H. Wu, “How can ai recognize pain and express empathy,” 2021.

[5] J. O. Egede, S. Song, T. A. Olugbade, C. Wang, A. Williams, H. Meng, M. Aung, N. D. Lane, M. Valstar, and N. Bianchi-Berthouze, “Emopain challenge 2020: Multimodal pain evaluation from facial and bodily expressions,” 2020. [Online]. Available: https://arxiv.org/abs/2001.07739

[6] P. Werner, “A list of pain recognition databases that are publicly available for research.” 2021. [Online]. Available: https://github.com/philippwerner/pain-database-list

[7] M. S. H. Aung, S. Kaltwang, B. Romera-Paredes, B. Martinez, A. Singh, M. Cella, M. Valstar, H. Meng, A. Kemp, M. Shafizadeh, A. C. Elkins, N. Kanakam, A. de Rothschild, N. Tyler, P. J. Watson, A. C. d. C. Williams, M. Pantic, and N. Bianchi-Berthouze, “The automatic detection of chronic pain-related expression: Requirements, challenges and the multimodal emopain dataset,” *IEEE Transactions on Affective Computing*, vol. 7, no. 4, pp. 435–451, 2016.

[8] D. T. H. Lai, P. Levinger, R. K. Begg, W. L. Gilleard, and M. Palaniswami, “Automatic recognition of gait patterns exhibiting patellofemoral pain syndrome using a support vector machine approach,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 5, pp. 810–817, 2009.

[9] H. Grip, F. Ohberg, U. Wiklund, Y. Sterner, J. Karlsson, and B. Gerdle, “Classification of neck movement patterns related to whiplash-associated disorders using neural networks,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 7, no. 4, pp. 412–418, 2003.

[10] M. S. Salekin, G. Zamzmi, D. Goldgof, R. Kasturi, T. Ho, and Y. Sun, “Multimodal spatio-temporal deep learning approach for neonatal postoperative pain assessment,” *Computers in Biology and Medicine*, vol. 129, p. 104150, Feb 2021. [Online]. Available: http://dx.doi.org/10.1016/j.compbiomed.2020.104150

[11] T. A. Olugbade, N. Bianchi-Berthouze, N. Marquardt, and A. C. Williams, "Pain level recognition using kinematics and muscle activity for physical rehabilitation in chronic pain," in *2015 International Conference on Affective Computing and Intelligent Interaction (ACII)*, 2015, pp. 243–249.

[12] T. Olugbade, M. Aung, N. Bianchi-Berthouze, N. Marquardt, and A. Williams, "Bi-modal detection of painful reaching for chronic pain rehabilitation systems," 11 2014, pp. 455–458.

[13] Z. Yang, A. Kay, Y. Li, W. Cross, and J. Luo, "Pose-based body language recognition for emotion and psychiatric symptom interpretation," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 294–301.

[14] F. Noroozi, C. A. Corneanu, D. Kamińska, T. Sapiński, S. Escalera, and G. Anbarjafari, "Survey on emotional body gesture recognition," *IEEE Transactions on Affective Computing*, vol. 12, no. 2, pp. 505–523, 2021.

[15] T. Hakim, "A comprehensive review of skeleton-based movement assessment methods," 07 2020.

[16] A. Paiement, L. Tao, S. Hannuna, M. Camplani, D. Damen, and M. Mirmehdi, "Online quality assessment of human movement from skeleton data," *BMVC 2014 - Proceedings of the British Machine Vision Conference 2014*, 01 2014.

[17] S. Nerella, A. Bihorac, P. Tighe, and P. Rashidi, "Facial action unit detection on icu data for pain assessment," 2020. [Online]. Available: https://arxiv.org/abs/2005.02121

[18] L. I. Strand, K. F. Gundrosen, R. K. Lein, M. Laekeman, F. Lobbezoo, R. Defrin, and B. S. Husebo, "Body movements as pain indicators in older people with cognitive impairment: A systematic review," *European Journal of Pain*, vol. 23, no. 4, pp. 669–685, 2018.

[19] D. Corbett, C. Simon, T. Manini, S. George, J. Riley, and R. Fillingim, "Movement-evoked pain: transforming the way we understand and measure pain," *PAIN*, p. 1, 10 2018.

[20] L.-C. Hydén and M. Peolsson, "Pain gestures: The orchestration of speech and body gestures," *Health: An Interdisciplinary Journal for the Social Study of Health, Illness and Medicine*, vol. 6, no. 3, pp. 325–345, 2002.

[21] C. Zheng, W. Wu, C. Chen, T. Yang, S. Zhu, J. Shen, N. Kehtarnavaz, and M. Shah, "Deep learning-based human pose estimation: A survey," 2020. [Online]. Available: https://arxiv.org/abs/2012.13392

[22] C. Beyan, S. Karumuri, G. Volpe, A. Camurri, and R. Niewiadomski, "Modeling multiple temporal scales of full-body movements for emotion classification," *IEEE Transactions on Affective Computing*, pp. 1–1, 2021.

[23] S. Chen, Y. Tian, Q. Liu, and D. Metaxas, "Recognizing expressions from face and body gesture by temporal normalized motion and appearance features," 07 2011, pp. 7 – 12.

[24] N. Dael, M. Goudbeek, and K. Scherer, "Perceived gesture dynamics in nonverbal expression of emotion," *Perception*, vol. 42, pp. 642–57, 06 2013.

[25] A. Crenn, A. Meyer, H. Konik, R. A. Khan, and S. Bouakaz, "Generic body expression recognition based on synthesis of realistic neutral motion," *IEEE Access*, vol. 8, pp. 207 758–207 767, 2020.

[26] F. Pollick, H. Paterson, A. Bruderlin, and A. Sanford, "Perceiving affect from arm movement," *Cognition*, vol. 82, 01 2002.

[27] G. Castellano, L. Kessous, and G. Caridakis, "Emotion recognition through multiple modalities: Face, body gesture, speech," in *Affect and Emotion in Human-Computer Interaction*, 2008.

[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016.

[29] P. Lucey, J. F. Cohn, I. Matthews, S. Lucey, S. Sridharan, J. Howlett, and K. M. Prkachin, "Automatically detecting pain in video through facial action units," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 3, pp. 664–674, 2011.

[30] M. T. Vu, M. Beurton-Aimar, P.-y. Dezaunay, and M. C. Eslous, "Automated pain estimation based on facial action units from multi-databases," in *2021 Joint 10th International Conference on Informatics, Electronics Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision Pattern Recognition (icIVPR)*, 2021, pp. 1–8.

[31] K. Pikulkaew, W. Boonchieng, E. Boonchieng, and V. Chouvatut, "2d facial expression and movement of motion for pain identification with deep learning methods," *IEEE Access*, vol. 9, pp. 109 903–109 914, 2021.

[32] M. Lee, L. Kennedy, A. Girgensohn, L. Wilcox, J. S. E. Lee, C. W. Tan, and B. L. Sng, "Pain intensity estimation from mobile video using 2d and 3d facial keypoints," 2020.

[33] R. Fritz, M. Wilson, G. Dermody, M. Schmitter-Edgecombe, and D. Cook, "Automated smart home assessment to support pain management: Multiple methods analysis," *Journal of Medical Internet Research*, vol. 22, p. e23943, 11 2020.

[34] A. W. K. Lam, D. Varona-Marin, Y. Li, M. Fergenbaum, and D. Kulić, "Automated rehabilitation system: Movement measurement and feedback for patients and physiotherapists in the rehabilitation clinic," *Human–Computer Interaction*, vol. 31, no. 3-4, pp. 294–334, 2016. [Online]. Available: https://doi.org/10.1080/07370024.2015.1093419

[35] Z. Hammal and J. F. Cohn, "Towards multimodal pain assessment for research and clinical use," in *Proceedings of the 2014 Workshop on Roadmapping the Future of Multimodal Interaction Research Including Business Opportunities and Challenges*, ser. RFMIR '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 13–17. [Online]. Available: https://doi.org/10.1145/2666253.2666257

[36] S. Walter, S. Gruss, S. Frisch, J. Liter, L. Jerg-Bretzke, B. Zujalovic, and E. Barth, "“what about automated pain recognition for routine clinical use?” a survey of physicians and nursing staff on expectations, requirements, and acceptance," *Frontiers in Medicine*, vol. 7, 2020.

[37] K. Prkachin and Z. Hammal, "Automated assessment of pain: Prospects, progress, and a path forward," in *Companion Publication of the 2021 International Conference on Multimodal Interaction*, ser. ICMI '21 Companion. New York, NY, USA: Association for Computing Machinery, 2021, p. 54–57. [Online]. Available: https://doi.org/10.1145/3461615.3485671

[38] M. Atee, K. Hoti, and J. Hughes, "A technical note on the painchek™ system: A web portal and mobile medical device for assessing pain in people with dementia," 06 2018.

[39] Y. Xiang, L. Zhao, Z. Liu, X. Wu, J. Chen, E. Long, D. Lin, Y. Zhu, C. Chen, Z. Lin *et al.*, "Implementation of artificial intelligence in medicine: Status analysis and development suggestions," *Artificial Intelligence in Medicine*, vol. 102, p. 101780, 2020.

[40] D. Liu, F. Peng, A. Shea, Ognjen, Rudovic, and R. Picard, "Deepfacelift: Interpretable personalized models for automatic estimation of self-reported pain," 2017. [Online]. Available: https://arxiv.org/abs/1708.04670

[41] X. Xu and V. R. d. Sa, "Exploring multidimensional measurements for pain evaluation using facial action units," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, 2020, pp. 786–792.

[42] J. Zhou, X. Hong, F. Su, and G. Zhao, "Recurrent convolutional neural network regression for continuous pain intensity estimation in video," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 1535–1543.

[43] S. Rezaei, A. Moturu, S. Zhao, K. M. Prkachin, T. Hadjistavropoulos, and B. Taati, "Unobtrusive pain monitoring in older adults with dementia using pairwise and contrastive training," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 5, pp. 1450–1462, 2021.

[44] R. Yang, Z. Guan, Z. Yu, X. Feng, J. Peng, and G. Zhao, "Non-contact pain recognition from video sequences with remote physiological measurements prediction," 2021.

[45] G. Bargshady, X. Zhou, R. C. Deo, J. Soar, F. Whittaker, and H. Wang, "Ensemble neural network approach detecting pain intensity from facial expressions," *Artificial Intelligence in Medicine*, vol. 109, p. 101954, 2020.

[46] P. Rodriguez, G. Cucurull, J. Gonzàlez, J. M. Gonfaus, K. Nasrollahi, T. B. Moeslund, and F. X. Roca, "Deep pain: Exploiting long short-term memory networks for facial expression classification," *IEEE Transactions on Cybernetics*, pp. 1–11, 2017.

[47] X. Xin, X. Lin, S. Yang, and X. Zheng, "Pain intensity estimation based on a spatial transformation and attention cnn," *PLOS ONE*, vol. 15, no. 8, p. e0232412, 2020.

[48] P. Ekman and W. V. Friesen, "Facial action coding system: Manual," 1978.

[49] K. M. Prkachin and Z. Hammal, "Computer mediated automatic detection of pain-related behavior: Prospect, progress, perils," *Frontiers in Pain Research*, vol. 2, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/fpain.2021.788606

[50] P. Werner, A. Al-Hamadi, K. Limbrecht-Ecklundt, S. Walter, and H. Traue, "Head movements and postures as pain behavior," *PLOS ONE*, vol. 13, p. e0192767, 02 2018.

[51] P. Werner, A. Al-Hamadi, K. Limbrecht-Ecklundt, S. Walter, S. Gruss, and H. C. Traue, "Automatic pain assessment with facial activity descriptors," *IEEE Transactions on Affective Computing*, vol. 8, no. 3, pp. 286–299, 2017.

[52] P. Werner, A. Al-Hamadi, R. Niese, S. Walter, S. Gruss, and H. Traue, "Towards pain monitoring: Facial expression, head pose, a new database, an automatic system and remaining challenges," 09 2013.

[53] P. Werner, A. Al-Hamadi, R. Niese, S. Walter, S. Gruss, and H. C. Traue, "Automatic pain recognition from video and biomedical signals," in *2014 22nd International Conference on Pattern Recognition*, 2014, pp. 4582–4587.

[54] D. Lopez-Martinez and R. Picard, "Continuous pain intensity estimation from autonomic signals with recurrent neural networks," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 5624–5627.

[55] ——, "Multi-task neural networks for personalized pain recognition from physiological signals," 2017.

[56] F. Pouromran, S. Radhakrishnan, and S. Kamarthi, "Exploration of physiological sensors, features, and machine learning models for pain intensity estimation," *PLOS ONE*, vol. 16, no. 7, p. e0254108, 2021.

[57] P. Thiam, H. Hihn, D. A. Braun, H. A. Kestler, and F. Schwenker, "Multi-modal pain intensity assessment based on physiological signals: A deep learning perspective," *Frontiers in Physiology*, vol. 12, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/fphys.2021.720464

[58] P. Thiam, V. Kessler, M. Amirian, P. Bellmann, G. Layher, Y. Zhang, M. Velana, S. Gruss, S. Walter, H. C. Traue, D. Schork, J. Kim, E. André, H. Neumann, and F. Schwenker, "Multi-modal pain intensity recognition based on the senseemotion database," *IEEE Transactions on Affective Computing*, vol. 12, no. 3, pp. 743–760, 2021.

[59] P. Thiam and F. Schwenker, "Multi-modal data fusion for pain intensity assessment and classification," in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2017, pp. 1–6.

[60] M. Kächele, P. Thiam, M. Amirian, P. Werner, S. Walter, F. Schwenker, and G. Palm, "Multimodal data fusion for person-independent, continuous estimation of pain intensity," 09 2015.

[61] Y. Huang, C. Du, Z. Xue, X. Chen, H. Zhao, and L. Huang, "What makes multi-modal learning better than single (provably)," 2021. [Online]. Available: https://arxiv.org/abs/2106.04538

[62] D. Lopez-Martinez, O. Rudovic, and R. Picard, "Physiological and behavioral profiling for nociceptive pain estimation using personalized multitask learning," 2017.

[63] P. Werner, A. Al-Hamadi, S. Gruss, and S. Walter, "Twofold-multimodal pain recognition with the x-ite pain database," in *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*, 2019, pp. 290–296.

[64] A. Geron, *Hands-on machine learning with Scikit-Learn and TensorFlow*, 2nd ed. O'Reilly Media, Inc., 2019.

[65] S. Walter, S. Gruss, H. Traue, P. Werner, A. Al-Hamadi, M. Kächele, F. Schwenker, A. Andrade, and G. Moreira, "Data fusion for automated pain recognition," in *2015 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, 2015, pp. 261–264.

[66] P. Hong and T. Huang, "Learning to extract temporal signal patterns from temporal signal sequence," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 2, 2000, pp. 648–651 vol.2.

[67] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biology*, vol. 52, pp. 99–115, 1990.

[68] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," 2016. [Online]. Available: https://arxiv.org/abs/1602.07261

[69] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[70] R. T.J.J., "Lstms explained: A complete, technically accurate, conceptual guide with keras," 2020. [Online]. Available: https://medium.com/analytics-vidhya/lstms-explained-a-complete-technically-accurate-conceptual-guide-with-keras-2a650327e8f2

[71] M. A. Haque, R. B. Bautista, F. Noroozi, K. Kulkarni, C. B. Laursen, R. Irani, M. Bellantonio, S. Escalera, G. Anbarjafari, K. Nasrollahi, O. K. Andersen, E. G. Spaich, and T. B. Moeslund, "Deep multimodal pain recognition: A database and comparison of spatio-temporal visual modalities," in *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 2018, pp. 250–257.

[72] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3367–3375.

[73] G. S. Handelman, H. K. Kok, R. V. Chandra, A. H. Razavi, S. Huang, M. Brooks, M. J. Lee, and H. Asadi, "Peering into the black box of artificial intelligence: Evaluation metrics of machine learning methods," *American Journal of Roentgenology*, vol. 212, no. 1, pp. 38–43, 2019, pMID: 30332290. [Online]. Available: https://doi.org/10.2214/AJR.18.20224

[74] [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[75] G. Developers, "Machine learning crash course with tensorflow apis," https://developers.google.com/machine-learning/crash-course, accessed 02/05/22.

[76] [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.plot_roc_curve.html

[77] [Online]. Available: https://scikit-plot.readthedocs.io/en/stable/metrics.html

[78] [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html

[79] M. P. Ponti Jr., "Combining classifiers: From the creation of ensembles to the decision fusion," in *2011 24th SIBGRAPI Conference on Graphics, Patterns, and Images Tutorials*, 2011, pp. 1–10.

[80] K. Gadzicki, R. Khamsehashari, and C. Zetzsche, "Early vs late fusion in multimodal convolutional neural networks," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, 2020, pp. 1–6.

[81] K. Liu, Y. Li, N. Xu, and P. Natarajan, "Learn to combine modalities in multimodal deep learning," 2018.

[82] S. Piana, A. Staglianò, F. Odone, A. Verri, and A. Camurri, "Real-time automatic emotion recognition from body gestures," 2014.

[83] C. Bishop, M. Edwards, and A. Turner, "Screening movement dysfunctions using the overhead squat," *Professional Strength and Conditioning Journal*, 10 2016.

[84] C. Drummond, "Machine learning as an experimental science (revisited)," in *AAAI workshop on evaluation methods for machine learning*, 2006, pp. 1–5.

[85] A. Ng, *Advice for applying Machine Learning*. Stanford University. [Online]. Available: https://cs229.stanford.edu/materials/ML-advice.pdf

[86] 2022. [Online]. Available: https://www.cs.cmu.edu/~face/facs.htm

[87] K. M. Prkachin and P. E. Solomon, "The structure, reliability and validity of pain expression: Evidence from patients with shoulder pain," *Pain*, vol. 139, no. 2, p. 267–274, 2008.

[88] 2022. [Online]. Available: https://github.com/TadasBaltrusaitis/OpenFace#functionality

[89] R. Josyula and S. Ostadabbas, "A review on human pose estimation," 2021.

[90] 2022. [Online]. Available: https://github.com/tensorflow/tfjs-models/tree/master/pose-detection#example-code-and-demos

[91] J. Borges, "Deepstack: Ensembles for deep learning." [Online]. Available: https://github.com/jcborges/DeepStack

[92] T. Sapiński, D. Kamińska, A. Pelikant, and G. Anbarjafari, "Emotion recognition from skeletal movements," *Entropy*, vol. 21, p. 646, 06 2019.

[93] M. Littmann, K. Selig, L. Cohen-Lavi, Y. Frank, P. Hönigschmid, E. Kataka, A. Mösch, K. Qian, A. Ron, S. Schmid, A. Sorbie, L. Szlak, A. Wiener, N. Ben-Tal, M. Niv, D. Razansky, B. Schuller, D. Ankerst, T. Hertz, and B. Rost, "Validity of machine learning in biology and medicine increased through collaborations across fields of expertise," *Nature Machine Intelligence*, vol. 2, 01 2020.

[94] B. Taati, S. Zhao, A. B. Ashraf, A. Asgarian, M. E. Browne, K. M. Prkachin, A. Mihailidis, and T. Hadjistavropoulos, "Algorithmic bias in clinical populations—evaluating and improving facial analysis technology in older adults with dementia," *IEEE Access*, vol. 7, pp. 25 527–25 534, 2019.

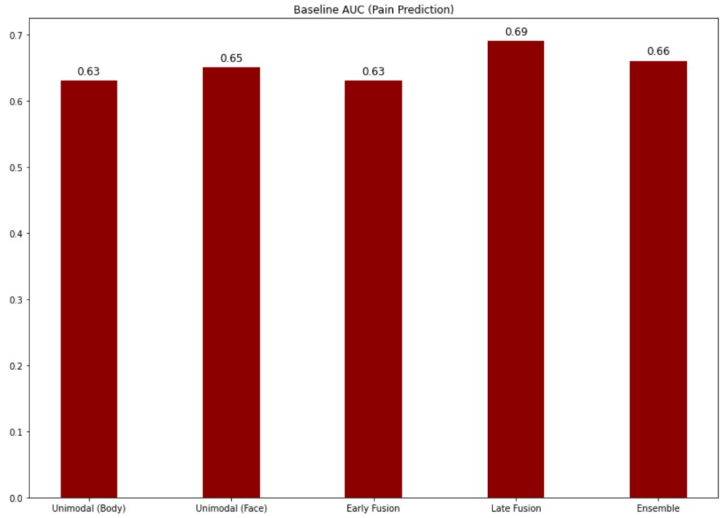
[95] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *PLOS ONE*, vol. 16, no. 7, p. e0254841, jul 2021. [Online]. Available: https://doi.org/10.1371%2Fjournal.pone.0254841

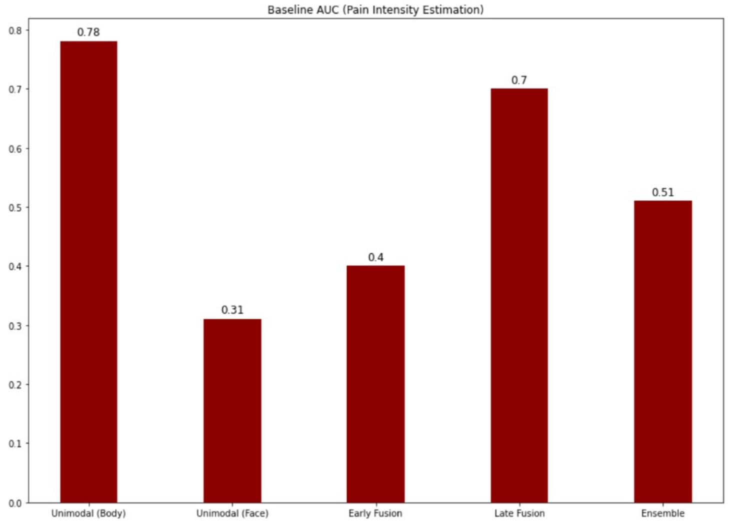
[96] T. H. Phan and K. Yamamoto, "Resolving class imbalance in object detection with weighted cross entropy losses," 2020.

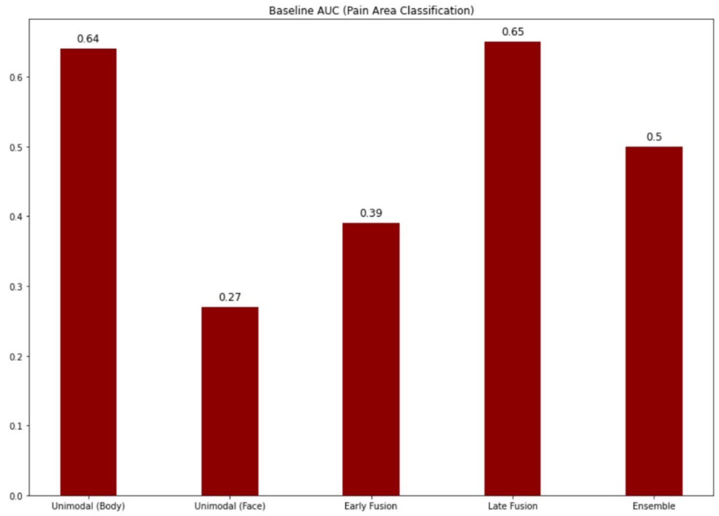
[97] P. A. Peeters and J. W. Vlaeyen, "Feeling more pain, yet showing less: The influence of social threat on pain," *The Journal of Pain*, vol. 12, no. 12, pp. 1255–1261, 2011.

A Baseline AUC Results

Baseline AUC performances of the two unimodal and two bimodal fusion approaches.


Baseline AUC (Pain Prediction)
Unimodal (Body)
Unimodal (Face)
Early Fusion
Late Fusion
Ensemble


Baseline AUC (Pain Intensity Estimation)
Unimodal (Body)
Unimodal (Face)
Early Fusion
Late Fusion
Ensemble


Baseline AUC (Pain Area Classification)
Unimodal (Body)
Unimodal (Face)
Early Fusion
Late Fusion
Ensemble