Project Report

Centennial Research Grant, University of Dhaka

Project Title:

IHABOT: Intelligent Hospital Assistance Robot to Fight Contagion by Reducing Doctor-Patient Interaction

Principal Investigator :
Dr. Sejuti Rahman
Assistant Professor
Department of Robotics and Mechatronics Engineering
University of Dhaka

Co- Investigator :
Sujan Sarker
Assistant Professor
Department of Robotics and Mechatronics Engineering
University of Dhaka

Abstract

During the COVID-19 pandemic, we realized the importance and necessity of automation in hospitals and healthcare facilities. Robotics has already established itself as a necessity in the medical field, ranging from automatic diagnostic systems to assisting nurses in health care facilities, thus reducing the tedious strain of physicians or nurses and increasing diagnostic accuracy. Our project utilizes the state-of-the-art advancements in vision-based action recognition, human robot interaction, artificial intelligence, and deep learning to build an autonomous, feature-rich hospital and clinic aid robot. The proposed Intelligent Hospital Assistance Robot (IHABOT) is equipped with a number of autonomous features. First of all, it can map its surroundings, determine the best route to take to its destination, and hence navigate by itself in a real-world environment. Second, it has a variety of sensors to collect physiological data from the patient, including temperature, systolic and diastolic blood pressure, oxygen saturation, and pulse rate. IHABOT uses artificial intelligence (AI) to automatically evaluate these physiological measures and detect patients who are deteriorating early, allowing for prompt treatment and a reduction in significant adverse events. Thirdly, in the absence of a doctor, this medical robot can keep track of and assess patients' performance on exercises throughout post-stroke therapy. The robot delivers a performance score that aids in both patient self-evaluation of performance as well as medical professionals' evaluation of patient development and prescription of required actions. Last but not the least, IHABOT diagnoses COVID-19 from radiography images and CT scans using a novel few-shot learning-based method. Often, the conventional diagnostic techniques with high accuracy have the setback of being expensive and sophisticated, requiring skilled individuals for specimen collection and screening, resulting in lower outreach. Therefore, medical robots like IHABOT, which do not require direct human intervention, can be used in hospitals for automated diagnosis and to lessen the likelihood of infection spreading through reduced human-to-human contact.

Acknowledgement

The investigators would like to specifically thank the Centennial Research Grant, University of Dhaka for funding this project. We also express our gratitude to the Department of Robotics and Mechatronics Engineering (RME), University of Dhaka, for providing us with the laboratory facilities.

Project Contributions

- List of Publications
  - Sejuti Rahman, Sujan Sarker, A K M Nadimul Haque, Monisha Mushtary Uttsha, Md Fokhrul Islam and Swakshar Deb, "AI-driven Stroke Rehabilitation Systems and Assessment: A Systematic Review," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, Under 2nd Round of Review (SJR rank: Q1, Impact factor: 3.802).
- Working Paper
  - Automated Diagnosis of COVID-19 from Radiography Images Using Few-shot Learning.

Contents

Chapter 1

Introduction

1.1 Importance of an Automated Medical Robot

Medical robots are defined as robots used in the health sciences or medical field. The use of robotics in healthcare and medicine has advanced significantly since it was first introduced in the operating room. In the 1980s, the first surgical assistance robots with robotic arms started to appear. Deep learning, computer vision, and AI have revolutionized medical robots over time, enhancing their capabilities in a range of additional healthcare settings. As a result, in a number of medical specialties, including surgery, radiotherapy, rehabilitation, laboratory diagnosis, prosthetics, and social care, robots are being employed to assist.

Robots operate uninterrupted, can conduct accurate movements even beyond the human range of motion, and can stay with patients for as long as is required. Additionally, they can automate routine or low-level operations while leaving high-level work to humans. Moreover, the World Health Organization estimates that there will be an additional 18 million healthcare worker shortages by 2030. Thus, the healthcare industry begins to investigate the possibilities of automated medical robots to support ongoing healthcare procedures.

As a result of the COVID-19 outbreak, we have come to realize the value and necessity of automation in hospitals and healthcare facilities. Robotics in the medical field has

already demonstrated that it is necessary, from automatic diagnostic systems to aid in healthcare facilities to lessen the tedious workload of nurses and improve diagnosis accuracy. Fig. 1.1 shows an application use-case of an autonomous assistance robot in a hospital setting. The goal of this project is to accumulate the features necessary to develop an autonomous, feature-rich assistance robot for hospitals and medical facilities.

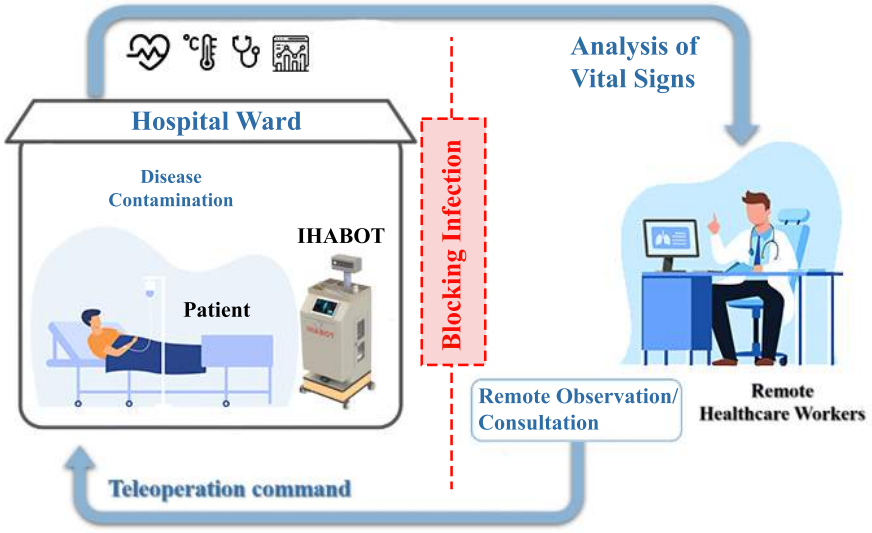


Figure 1.1: Application of an autonomous hospital assistance robot

1.2 Motivation of This Project

Since the first discovery of the Covid-19 virus in the human body, the environment has changed a lot and understood its frightening effects on global health, economics, government, and many other fields. Innovation has been and is still important, especially in the public health and medical sectors, as the very nature of the virus is still unknown to all. One thing is for sure, one of the most widely transmitted infections ever reported is this disease. Sadly, the frontline troops of this worldwide war, the doctors and the nurses were the most numerous to take the hard blow, and we saw a large number of casualties. According to WHO statistics, as of May 2021, COVID was responsible for at least 115,000 deaths among medical workers. The accelerated need for robotics and logistics development, which highlights the constant threats faced by healthcare workers

during the COVID epidemic, served as the driving force behind this project. In a hospital setting, healthcare workers frequently do a variety of physically demanding duties that raise their risk of contracting viruses like COVID 19. Robots can, however, perform the majority of these duties well while lowering the risk of contamination. Less human interaction is produced by enhancing the autonomous features of a healthcare assistance robot. The goal of this project is to create an autonomous hospital assistance robot that will guarantee a secure working environment for human medical personnel.

1.3 Challenges

Hospital assistance robots face a variety of design and development issues since they work in close proximity to patients in a delicate hospital setting. These challenges include safety, reliability, implementation costs, and maintenance. The major challenges encountered in developing an automated hospital assistance robot are listed below.

1.3.1 Safety

One of the main concerns when building a hospital assistance robot is safety. For medical robots, the conventional safety measures used for industrial robots and other robots are insufficient. For instance, safety standards for industrial robots recommend that the robots be contained in a cell with safety interlocks to avoid direct human contact. Hospital assistance robots, on the other hand, typically require direct contact with patients and medical staff. These robots concern human life and are utilized in medical settings for patients. As a result, the safety concerns for medical robots are more strict, serious, and important.

1.3.2 Robot Morphology

The exterior structure and design of the robot are referred to as its morphology. Defining the morphological structure of the robot is a challenging task and it depends on the

functionality of the robot. An anthropomorphic (human-like) appearance is preferred for the hospital assistance robot because it will be expected to work in a hospital environment and interact with patients. However, because it is necessary to service the patients, functional design considerations are needed to make space for medical equipment and medicine storage easier. Therefore, it is imperative to create a functional robot with anthropomorphic traits such as the ability to engage with natural language, perceive objects using computer vision, and understand human behavior, and so on.

1.3.3 User friendliness

The user-friendliness of a medical assistance robot is another significant design challenge. The robot's interfaces and modes of interaction should be made as user-friendly as possible for the healthcare staff. Additionally, the interface patients use to communicate with the robot should be fairly straightforward and shouldn't put them under too much stress.

1.3.4 Usability

Another significant problem is defining the functional components of a hospital assistance robot so that it can be used in a medical environment. The applications for which a robot is developed determine the functional capabilities the robot should have. A variety of difficulties must be overcome by the developers in order to make a robot useful in real-world situations. One important barrier for robotics, for instance, is real-world, real-time, robust sensing in human situations. Despite improvements in image resolution and solutions to camera recognition issues brought about by computer vision, algorithms still have trouble when objects or people move in front of the cameras, data is lost, sensors are covered up, or the environment is cluttered. These situations occur frequently in human social settings, and it can be challenging to understand, interact with, and learn from end users.

1.3.5 Cost

A robotic system's design, implementation, and maintenance costs make up its entire cost. Cost is a crucial element since it affects the end user's ability to utilize the service. It can be difficult to save costs because doing so might mean sacrificing the quality of the equipment, like sensors. High-quality sensors are needed in medical applications, though, because precise results are needed. As a result, maintaining high precision of the robotic system while optimizing cost is challenging.

1.3.6 Autonomy

It can be difficult to define the autonomy level of a hospital assistance robot. Less human involvement results from the robot's increased autonomy. However, human interaction is necessary in a hospital setting since patients could face emergencies. It also calls for human experts to assess any vital signs taken from the patients. In the case of any failure event, the human operator must also alter the robot's goal and plan accordingly.

1.3.7 Acceptability

A hospital assistance robot's acceptability is a key consideration in its design. The patients come from a variety of backgrounds and age groups, and they could be affected by social stigma. Thus, when engaging with the patients, the robot should be designed so as to not induce fear or discomfort. Realizing an optimum design is difficult since these elements can vary from person to person.

1.3.8 Reliability

The most important component of medical robots is reliability. A hospital assistance robot's duties include disease diagnosis, such as COVID identification from radiology pictures; task evaluation, such as evaluating patients' exercise levels; and vital sign analysis, such as blood pressure measurement. These diagnosis, assessment, and analysis

results need to line up with those of human practitioners and therapists. It is difficult to develop algorithms for this purpose that are highly accurate and precise.

1.3.9 Sustainability

The design of a hospital assistance robot is not a one-time endeavor; rather, it should be long-lasting. Designers should therefore consider how easily the systems may be updated. If any system component fails, it should be simple to replace it. It should be simple to upgrade to newer versions of the program and algorithm being utilized. All of these present difficult design challenges for a hospital assistance robot.

1.3.10 Adaptability

Robotics has made remarkable technological progress in recent years, yet many demonstrations outside of very constrained environments have failed. Since there is no "one-size-fits-all" answer to most problems, this is especially troublesome when building healthcare technology. Every individual, task, and care environment is unique, thus robots must be able to learn and adapt quickly.

1.3.11 Autonomous Navigation

An autonomous hospital assistance robot's navigation is a difficult challenge since it must deal with moving obstacles and changing environments.

1.4 Project Contribution

In this project we aim to develop an autonomous hospital assistance robot to deploy in a hospital setting that automates diagnosis, assessment, and analysis tasks while reducing patient-doctor interactions. The major contributions of our project is listed below.

- We developed a prototype intelligent medical assistance robot, called IHABOT, to automate human-level tasks in a hospital setting.
- We carried out the requirement analysis by thoroughly examining current works, and we designed the prototype in AutoCAD.
- We implemented algorithms in the prototype robot to automate the human medical staff level tasks: COVID diagnosis, exercise assessment, and vital analysis. Due to the unavailability of labeled data, we used few-shot learning for COVID diagnosis. For the exercise assessment purpose, the Graph Convolution Network (GCN) is utilized.
- We devise a real-time navigation algorithm to enable the robot to navigate autonomously in a hospital setting.
- We developed an android application that provides an interface for facilitating patient robot interaction.
- We implement a test-bed setting and analyzed the performance of the prototype.

1.5 Organization of the Report

The rest of the report is organized as follows. The state-of-the-art work in related fields are described in Chapter II. We detailed the proposed IHABOT in Chapter III. Automated navigation of the robot is described in Chapter IV. Chapter V details the automated Covid-19 detection from radiography images. Physical exercise monitoring, guidance, and evaluation are detailed in Chapter VI. Chapter VII presents the automated analysis of patient's vital signs. Finally we conclude our report in Chapter VIII.

Chapter 2

Literature Review

2.1 Intelligent Hospital Assistant Robots

Hospitals have been researched tremendously within Robotics field and revealed the complexity of functionalities at hospitals where several distinct artifacts (eg. [3], [4], [5]) are in play within naturalistic settings. Recent studies have displayed many capabilities of hospital robots for example collecting blood samples[6], moving patients[7], performing and supporting surgeries[8], transporting medical apparatus[9], providing companionship and mental support[10] etc. This reduces physically distributed tasks of the staffs.

In recent times, many industries are manufacturing robots to provide assistance to the nurses in clinics and hospitals. These robots are designed to operate under the direct control of the nurses. These will play the role of a teammate, aiding nurses by carrying out non-complex tasks etc. One example is Moxi[11] manufactured by Diligent Robotics, recovers and fetches materials to hospital rooms and nursing stations, takes out solied linen bags and delivers samples to laboratories. Moxi perceives the surrounding environment by sensors such as laser scanner and camera. In order to perform a simple fetch-an-object task, the robot needs to know the location of the warehouse and the locations of where the object is needed. Moxi navigates fully autonomously to the goal avoiding static and dynamic obstacles.

Another interesting concept of medical robot named YuMi[12] has been introduced by the robotics company ABB to work together with medical staff and lab workers. Yumi is a Dual-arm mobile robot which is equipped with seven DOF and a two-finger gripper. It is evaluated by Texas Medical Center Innovation in Huston where a variety of logistic tasks such as loading and unloading medical objects, grasping and controlling liquids, lifting and classifying test tubes etc is assigned to perform. Another mobile robot TUG[13] manufactured by Aethon does not have any robotic manipulator and thus it relies on the nurses and medical staff to load and unload equiments for delivering. Relay[14] is another example of a mobile robot built by Swisslog Healthcare that fetches medications and other important medical items. TUG and Relay are capable to navigate through hallways autonomously using sensors for environmental perception while avoiding static and moving obstacles.

However there are many commercially manufactured medical assistant robots whose main goal is to provide assistance to the patients. ROBEAR[15] is such a robot which is able to lift a patient from one bed to another or move a patient to a wheelchair or help them to stand up. It is developed by the RIKEN and Sumitomo Riko Company limited as an experimental nursing robot. Veemot[16] needle insertion robot developed by Veemot Systems automates the process to drawing blood and inserting intravenous therapy(IV) by correctly identifying the best vein with an accuracy of 83%. The social robot Pepper[17] built by Softbank Robotics also plays quite a few roles in healthcare field for example aiding medical staff, performing survey research on patient satisfaction, acting as receptionists in hospitals.

In a hospital setting, Das et al[18] developed a sitter robot that can keep track of patients' vital signs, engage them in conversation, and alert nurses to any unusual medical situations. The authors created a mobile app to supplement spoken orders provided to a robot using native interfaces like the camera and native voice recognition. The robot can speak with the patient during sitting sessions, help the patient with decision-making during pick-and-place tasks, and track their health over time thanks to the app. An Adaptive Robotic Nursing Assistant (ARNA)[19] is provided in Das' most recent study. ARNA is a versatile robot that assists nurses with routine chores including walking patients, getting items, and keeping tabs on patients' health. At the University

of Louisville in Kentucky's School of Nursing, ARNA was implemented and assessed in a hospital setting. The assessment of ARNA involved healthcare professionals. The results show an overall favorable response to the usage of a nursing robot based on a number of parameters including completion time, rate, and level of user satisfaction.

Additionally, ARNA was put through a cohort testing with 24 human participants, and the results of this initial user research show that the robot's crucial user sitter and walker qualities are both very useful and simple to operate. DeKonBot[20] is a recent prototype robot created by the Fraunhofer Institute for Manufacturing Engineering and Automation. It has a movable base and a robotic arm.Its primary function is to clean areas that might be polluted, including door knobs or elevator buttons. However, both research systems have only been tested in lab settings and are still in the development stage.

2.2 Automated Navigation

Numerous approaches have been put out for localizing, mapping, and planning a robot's course on its own. The positions and orientations of the robot are defined using two different forms of localization. Either global localization methods or localization methods with a focus on local communities are the subjects of research. In contrast to global localization, which determines a robot's position from an external frame like the stars or a stationary frame, local localization employs sensors (such as cameras, IMUs, tilt sensors, radars, and others) aboard a robot to localize its position in relation to its frame. Robots typically employ GPS, GNSS, and USAT sensors to determine their location around the globe. Methods for local localization are quick and responsive in terms of computing [21]. To estimate the position nowadays, localization uses the GPS/INS sensor fusion technique in conjunction with additional sensors [22]. Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun suggested an effective position estimation using the Monte Carlo Localization(MCL) approach [23]. Using a sonar sensor, Quoc-Nam Tang and Van Heip Dao proposed grid-based localization[24]. Although it operates in discrete representation, it has the disadvantage of being computationally inefficient

in space. The mapping algorithms that have been proposed so far can be generally categorized using the map representation and the underlying estimating method. One of them is the occupancy grid [25]. However, an entirely grid-based technique, such as the occupancy grid mapping, is computationally and memory intensive. The most often used estimate algorithms are Rao Blackwellized particle filters and Extended Kalman filters (EKFs) based on maximum likelihood methods. But their fundamental flaw is that they make a lot of assumptions about robot motion and sensor noise [26]. A technique for describing the spatial limitations among robot postures was put forth by Gutmann et al. [27] and involved building a network based on the history of sensor data. However, the enhancement of this network may not be feasible for real-time use. In his work, Murphy [28] developed the Rao-Blackwellized particle filters (RBPF), which each particle infers a potential robot trajectory and map, as an efficient method of resolving the SLAM problem. An effective and economical solution to the simultaneous localization and mapping (SLAM) problem was put out by Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard [29]. They made use of Rao-Blackwellized particle filters, but they offered a solution to the particle depletion issue by performing only certain re-sampling procedures. The proposed distribution's computation is quite similar to the FastSLAM-2 approach [30].

A approach of path planning that treats the surroundings like a potential field was put forth by Khatib [31]. In this hypothetical field, the target point draws the agent in while the barriers drive it away. But in this strategy, the agent occasionally enters local minima. Due to the extra care required to avoid local minima, this method is quite computationally expensive. In a grid context, Sturtevant et al. [32] and Gutmann et al. [33] proposed a graph-based path planning method. Using a graph technique like A* to extract the path from the graph requires more care than using a tree-based method, which is one drawback of the graph-based approach. A multi query planner Probabilistic Road Maps(PRM) technique with a learning phase was proposed by L. E. Kavraki [34]. The Query Phase merely needs to connect the start and target nodes and then look for a path, but the Learning Phase requires substantially more work to accomplish. The learning phase's graph, however, might be utilized for numerous ensuing inquiries. Because of this, PRM is referred to as a multi-query planner. In surroundings that are static

or barely changing, this is particularly advantageous. However, in some contexts the rate of change prevents the use of PRM's multi-query property. The added information and computational slowness of PRM are not welcome in such circumstances. Rapidly Exploring Random Tree(RRT) is a method for path planning in a high-dimensional configuration space that Steven M. LaValle [35] suggested. RRT is well known for its probabilistic completeness and is capable of efficiently covering the entire space. However, RRT is substantially faster than PRM at resolving a path planning issue. This is so because, unlike PRM, it takes into account the start and end nodes and restricts growth to the region immediately surrounding the existing graph. When it comes to large path planning issues (such as those with hundreds of dimensions), RRT is more effective than PRM in dynamic contexts. After analyzing all the methods we have discussed above, MCL for localization, an adaptive method of Rao-Blackwellized particle filters similar to FastSLAM for SLAM and RRT for path planning are selected for their computational friendly behavior and their real time efficiency. MCL for localization, an adaptive Rao-Blackwellized particle filter approach similar to FastSLAM for SLAM, and RRT for path planning are chosen for their computationally friendly behavior and real-time efficiency after assessing all the methods we have covered before.

2.3 AI-Driven Analysis of Vitals

The National Early Warning Score (NEWS) determines the degree of illness of a patient using six physiological findings and one observation. Many researchers tried to measure the impacts of this scoring system. In this paper [36], authors wanted to establish if there any effect of NEWS scoring system in reducing cardiorespiratory arrests. They find a significant relationship. They recommend NEWS as a part of daily nursing routines for all patients.

This research [37] found a nonvolatile effect of NEWS in predicting mortality, morbidity and diagnosing sepsis. They got 0.772 AUCROC in detecting morality by using NEWS. An important finding of this research is, the NEWS scoring system performs better than Systemic Inflammatory Response Syndrome (SIRS) scoring system in every matrix.

A possible mixture of data can be possible to gain tremendous efficiency in making medical decisions. Cristiano and Pasluosta [38] proposed a system named Internet of Health Things(IoHT) which takes vitals from different hospital wards, gathers them and applies the weighted early warning scoring system to evaluate the vitals. Along with NEWS parameters, level of pain and urine output are also being taken to evaluate the health risk of the patients.

2.4 Automated Covid-19 Detection from Radiography Images

The literature on deep learning-based computer vision algorithms for the diagnosis, prevention, and management of COVID-19 was addressed by Ulhaq et al. Along with a brief explanation of some of the current datasets, they also presented the deep learning algorithms relevant to infection control and treatment. They did, however, offer a pre-print (non-peer-reviewed) version of several papers, which restricts the work's acceptability. The authors of [14] talked about how AI and big data may help combat COVID-19. They discussed the existing SIR (Susceptible, Infected, and Removed) models and other deep models for identification, tracking, and outbreak prediction in addition to the existing deep learning architectures for COVID-19 detection and diagnosis. Various speech and text analysis techniques, cloud-based algorithms for infodemiology and infoveillance, deep learning algorithms for drug repurposing, outbreak prediction based on big data analysis, virus tracking, vaccine development, and drug discovery, among other techniques, were also included in the survey. However, they did not include any comparable quantitative analysis of the publications under consideration.

Medical imaging strategies for combating COVID-19 were evaluated by Shi et al. [13]. A number of contact-free image collecting methods, deep learning-based lung and lesion segmentation, X-ray and CT screens, COVID-19 severity analysis, and certain publically available datasets are all incorporated into the study. But they also did not offer any quantitative evaluation of the existing approaches. Their explanations of the current datasets are also not quite appropriate. Medical imaging strategies for

combating COVID-19 were evaluated by Shi et al. [13]. A number of contact-free image collecting methods, deep learning-based lung and lesion segmentation, X-ray and CT screens, COVID-19 severity analysis, and certain publically available datasets are all incorporated into the study. But they also did not offer any quantitative evaluation of the existing approaches. Their explanations of the current datasets are also not quite appropriate.

The SIR, SEIR (Susceptible, exposed, infected, and removed), and SIQR (Susceptible, infected, quarantined, and recovered) models were used by the authors to predict the pandemic mathematically in [15]. The authors of [17] provided a comparison list of publicly accessible datasets that included the picture data from COVID- 19 cases. However, they did not offer any new insight into earlier efforts or suggest any directions for further study in this area. Latif et al. [16] examined deep learning algorithms for risk assessment and patient prioritizing in addition to diagnosis, which is different from other research. Nguyen covered deep learning-driven medical imaging, Internet of Things-driven pandemic management strategies, and even Natural Language Processing (NLP)-based COVID-19 news analysis strategies in [18]. Like others, they did not, however, offer any comparative analysis. The majority of the articles under consideration attempted to cover a broad range of issues, but they lacked in-depth analyses in a particular area. None of the publications offer a quantitative examination of the works under discussion, which would be extremely beneficial to scholars. By concentrating on only one area—COVID-19 detection—our study seeks to get beyond these restrictions. Additionally, we offer a quantitative and comparative study of 315 distinct deep learning algorithms—something that no other survey articles have yet to achieve.

2.5 Physical Exercise Monitoring, Guidance, and Evaluation

Automated exercise assessment can be regarded as a classification task, categorizing a movement into correct or incorrect, or a regression task, predicting the score of a

movement. We continue our discussion by grouping the works in automated exercise assessment based on the task type and its significance.

Exercise Correctness Classification Exercises can be classified according to the accuracy with which they are performed. According to the literature, feature engineering-based algorithms have been the most frequently used. For instance, in [39], the authors used K nearest neighbor and SVM classifiers to identify compensatory motions in the pressure distribution, achieving F1 scores as high as 0.993. Jung et al. [40] on the other hand used model trees [41] and found modest results with an F-measure of 79.29 percent and a ROC of 0.91.

Lee, in [42], used a hybrid approach that combines a rule-based knowledge model and a predictive model for classifying the quality of motion as 0, 1, or 2. The findings indicated a good agreement level with the therapists' assessments. On this basis, Lee et al. [43] investigated several such hybrid models using a variety of classifiers, including Neural Networks (NNs), SVMs, and others and discovered that NNs produce an effective result. This was further demonstrated in [44], which combined reinforcement learning with a variety of classifiers. In [45], the authors developed an ensemble learning model composed of 18 classifiers, each trained on a random subspace. Using six categories, they found 92% accuracy for Brunnstrom and 82 percent for FMA scoring systems.

Another branch in the literature investigated the potential of deep learning for exercise quality classification. Zhi et al. [46] investigated the classification of compensatory motions in rehabilitation using both SVM and RNN classifiers where RNN did not perform as expected. Kaku et al. [47] also achieved unsatisfactory results with their CNN architecture paired with embedding modules, with an average accuracy of 70%. However, the work of [48] obtained great results, even in semi and uncontrolled environments. Zhu et al. [49] was also proven successful with their suggested multipath CNN, which was composed of a dynamic convolutional network called D-CNN and a state transition probability CNN called S-CNN, claiming a test accuracy level greater than 90%.

Exercise Quality Score Prediction Rather than anticipating discrete class labels as in classification approaches, the work in this section attempts to assign a continuous value

as an assessment score compared to that of a professional therapist. Typically, prior research in this field employs a distance function to assess the quality of performed and prescribed exercise [50, 51]. These approaches necessitate multiple pre-processing phases, impeding the system's end-to-end processing. In fact, hand-crafted feature based works are still prominent in the literature. The works of [52] and [40] found great results with their selection of features. Lee et al. [53] calculated the range of motion, their smoothness, and the occurrence of correct and erroneous movements. These features also worked well in score prediction, achieving a high level of agreement with the therapists. In another work, Liao et al. [54] presented a Spatio-temporal network capable of evaluating an exercise. They enhanced performance by combining temporal pyramids, multi-branch convolution, and recurrent layers. They used convolutions on tensors of joint data to disrupt the natural graph structure's fine spatial organization within the human body. Their approach yielded an average absolute deviation of only 0.02527.

Data Input Length Input length of the sensor data plays a vital role in assessing the exercises. In many cases, the patient might perform exercises at different speeds and the assessment score should not vary if they are performed correctly. However, most approaches keep the length of the exercise video fixed while training models, causing useful information to be left out. Some works such as [48], and [47] attempt to address this by choosing an input length that provides sufficient to capture movement data of all patients. But if a test sample contains useful information exceeding the predefined length, the model might fail to provide consistent results. One technique for dealing with this is dynamic time warping [55], which can handle temporal sequences of different lengths. This has been a widely used technique of past assessment tasks [51, 56, 57, 58] and even in recent works [59, 60]. A better solution was proposed by Zhu et al. [49] that trained a separate CNN network, D-CNN, to deal with dynamic input lengths. Recently, Deb et al. [61] proposed graph convolution based architecture for assessing the rehabilitation exercise to handle variable length and preserve local connectivity between each joint.

Table 2.1: Existing Rehabilitation Assessment Works (SDP = Structural Data Preservation, CC = Correctness Classification, SP = Score Prediction, FE = Feature Engineered, DM = Deep Model, Acc = Accuracy, MAD = Mean Absolute Deviation)

Study	System Task	Learning Type	Algorithm	Fixed Input length?	Region of Interest	SDP	No. of Subjects (Stroke:Healthy)	Performance
[39]	CC	FE	KNN, SVM	Y	Upper extremity	N	8 (8:0)	F1 scores: 0.993 (Binary), 0.981 (Multi-class)
[40]	CC, SP		Model trees				5 (5:0)	F1 score: 79.29%, RMSE: 0.32
[42]	CC		Multimodal Modal		Elbow joint		26 (15:11)	F1 score: 0.8565
[43]			Hybrid model(ML + Rule based)				26 (15:11)	F1 score: 0.8279
[44]			RL+Classifier				26 (15:11)	F1 score: 0.8119 (RL+NN)

[45]		Ensemble sub-space discrimi-nant analysis		Hands		50 exercise samples[1]	Acc: 92%
[46]		SVM, RNN		Upper limbs		19 (9:10)	F1 scores: 0.82 (SVM), 0.27 (RNN)
[47]		CNN		Upper extremity		48(48:0)	Acc: 70%
[48]	DM	Rehab-Net (CNN)		Arm		D1 - 4 (4:0), D2 - 10 (10:0) [2]	Acc: 97.89% (D1), 88.87% (D2)
[49]		Multipath CNN	N	Upper extremity		WHARF [62], 49 (0:49) [2]	Acc: 90.63%
[50]	FE	DTW Distances with KNN	N	Upper limb	N	29 (21:8)	Acc. 82.1
[51]		DTW distances		Upper extremity		5 (0:5)	Acc: 91.9% (Posture), 93.75% (Trajectory)

[52]	SP	FE	SVR	Y	Shoulder and elbow		24 (24:0)	RMSE: 2.1273
[54]			CNN, LSTM, HNN		Full body	Y	UI-PRMD [63]	Abs. deviation: 0.00656 (CNN)
[53]	CC, SP		Binary classifier with threshold modelling		Elbow joint	N	26 (15:11)	F1 score: 0.8436
[61]	SP	DM	STGCN	N	Full body	Y	KIMORE [64] and UI-PRMD [63]	MAD: 0.57 (KIMORE), 0.011 (UI-PRMD)
[58]	CC	FE	DTW distances	N	Full body		5 (0:5)	Acc: 93.46%

[1] Population details not given [2] Multiple datasets used

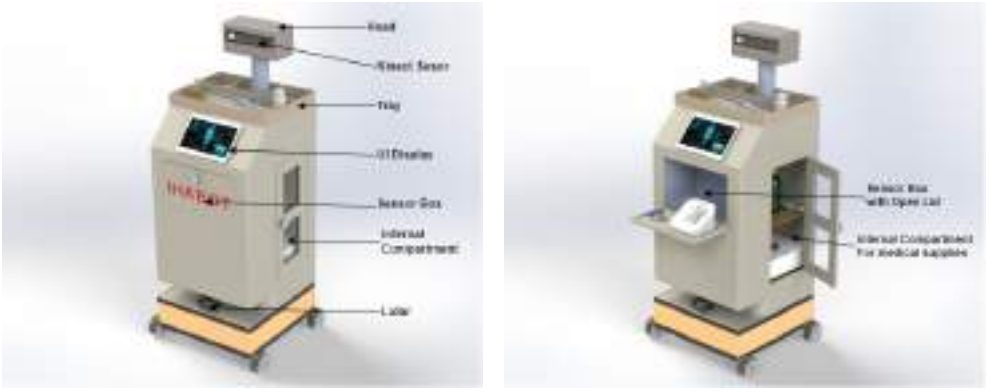
Chapter 3

Proposed IHABOT

Progressively socially assistive robots are being popular and considered more efficient in dangerous and iterative jobs. Healthcare sector can benefit itself by handling patients with a high level of safety at clinical settings using robots. Handling patients with contagious disease always comes with high risk for nurses. Healthcare robots can be a great help here. Moreover, during manual patient handling, the predominant risk of staff injury is excessive back and shoulder loading. An Intelligent Hospital Assistant Robot(IHABOT) is highly desired to enhance the efficacy and quality of care that nurses and their paraprofessional staff can provide. Such an assistant could improve a nurse's working conditions by off-loading some of his or her most physically demanding duties, thereby reducing the potential for self-injury or injury to the patient even without taking the risk of contagious patients.

3.1 Design of IHABOT

With inexpensive parts, IHABOT can perform a range of diagnosis and patient monitoring related tasks. The capacity to perform a variety of functions is occasionally conflicting with low-cost components. A challenging hardware design made it possible by considering the requirements through an iterative approach.



(a) (b)

Figure 3.1: IHABOT hardware setup with Lid and Internal Compartment: (a) Closed; (b) Open;

3.1.1 Robot Platform and Sensor Setup

IHABOT is based on a four-wheel differential drive system with four 30 rpm encoder motors mounted on four corners, considered to be the best possible stable platform. The robot has a rectangular footprint with a width of 48 cm and a length of 55 cm to accommodate all the built-in system components with a total height of 130cm. Robot body is made from pvc plastic that is highly durable and less denser than other metal sheets available. Converging the center of gravity towards its center of the footprint was one of the critical considerations while designing. A lidar is incorporated at the lower front part of the robot to navigate and map throughout the unknown environment avoiding low lying objects. A Kinect camera from Microsoft is installed in the robot's head for exercise assessment. A tab from Huawei is equipped on the chest area of the robot at a height of 90cm for better usability from hospital bed and even from standing position.

3.1.2 Sensor Box Setup for vitals measurement

Sensor box is installed at appropriate height for better ergonomic user experience measuring vitals form sitting position.There are three vital measurement devices for blood pressure, oxygen saturation, heartbeat rate and body temperature measurement.

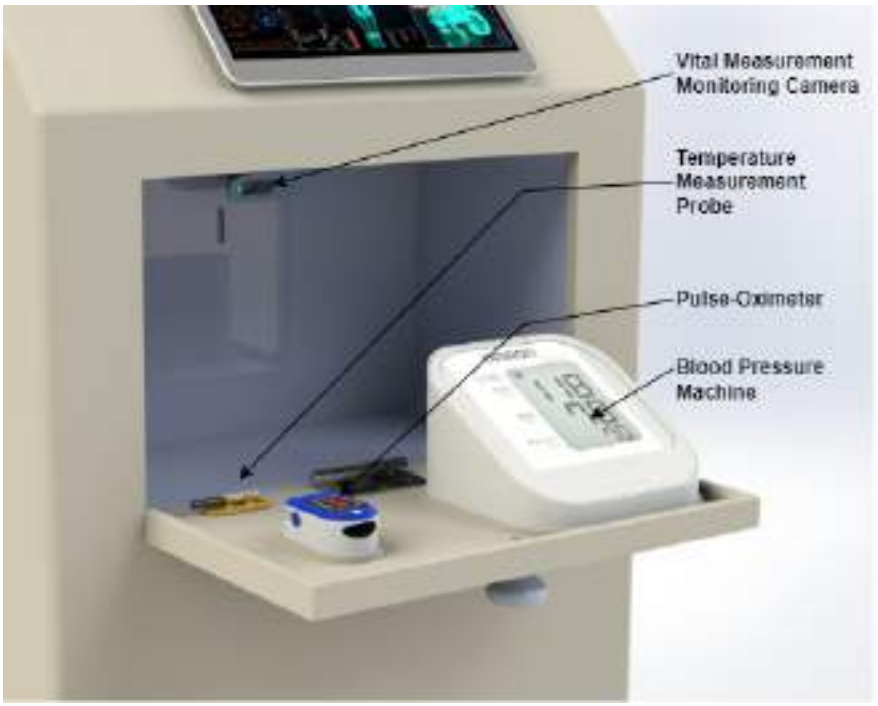


Figure 3.2: Sensor Box Setup for Vitals Measurement.

Blood Pressure Machine For the purpose of measuring patient blood pressure, a blood pressure monitor from "AccuMax" is installed on the sensor box. Additionally, it measures heart rate. Patients have to wear the arm cuff and press the ON button to measure vitals.

Pulse Oximeter Sensor box is also equipped with a pulse oximeter for the patient's oxygen saturation. With its heart rate measurement, system can reassure the heart rate comparing with blood pressure machine heart rate.

Temperature Probe To gauge body temperature, a temperature sensor is also accessible. To assess body temperature, patient must affix it inside armpit.

3.1.3 Medical Equipment Containers

A tray for regularly used medical supplies, including basic medications, hand sanitizer, and even stethoscopes for doctors, is located on top of the robot. It has a total are

of 36cm x 23cm with 5cm fencing. Three subsections have been added for better organization. At the left side of the robot there is another closed compartment for infrequently used medical supplies. Its overall volumes measure 50cm by 50 cm by 56 cm. Thus IHABOT will travel through elevators and corridors 24 hours a day bringing medications, test samples, and other essential goods, increasing throughput and efficiency without exerting much effort.

3.2 Features of IHABOT

Considering the fact of contagious Acute Respiratory Distress Syndrome(ARDS) such as COVID-19, IHABOT's features are evaluated frequently to maximize its efficiency.

1. Patient Doctor intermediary: IHABOT has both local and teleportation mode so that it can be the best intermediary system between doctor and patient.

2. Check Vitals: This robot consists of three vitals measuring devices - Blood pressure machine to measure systolic and diastolic blood pressure, Pulse Oximeter to measure oxygen saturation of the blood, Digital thermometer to measure body temperature. Here, both Blood Pressure machine and Pulse Oximeter can measure heart rate. What gives us flexibility to overcome any environment disturbance to get an accurate measurement.

3. Auto diagnosis of common disease: Most of the common diseases show symptoms such as cough and fever. Measuring vitals such as blood pressure, body temperature, heart rate and oxygen saturation those diseases can be primarily diagnosed at the initial stages before any critical situation arises.

4. Send Diagnosis Report: After preliminary diagnosis of a patient, if anything is considered critical IHABOT will send a report to the associated doctor about his/her patient. It will make sure the patient is not neglected with such pre-indication for being contagious.

5. Medication Reminders: It is very common to forget about the medication times that most nurses perform in a routine roundup. Since IHABOT is considered a replacement for nurses in case of indirect patient handling, it comes with medication reminders for the robot. When the time comes, the robot will guide itself to the patients to remind them about their medication details with auditory response.

6. Diagnose ARDS Based diseases from radiography image: When the lungs are suffering from acute respiratory distress syndrome(ARDS), the body's essential organs are unable to receive enough oxygen. It is common in Pneumonia, COVID-19 along with other lung disease. IHABOT has a module where providing X-ray images of patients, doctors can easily diagnose the issue.

7. Robust Navigation: It is very crucial for a medical assistant robot to have the best navigation system through the hallway and use elevators of the hospital. IHABOT uses the Robot Operating System(ROS) framework and best applicable path planning algorithm to map the environment and navigate itself described in later chapters.

8. Exercise evaluation :Using the Kinect Sensor mounted on the head of the robot, patient exercise is evaluated.

9. Dynamic Self learning and object recognition:

3.3 System Design

Behavior Coordination is one of the main challenges in any robotic system that interacts with humans. This behavioral system of the IHABOT robots was developed and implemented in a multi-stage development process because IHABOT's purpose explicitly required an autonomous system functioning for at least a couple of days since it does not have a self charging system.

Its control system is divided into two main parts- Navigation and Diagnosis with user control. Each module has its main controller as Raspberry Pi(RPi) 4B. Navigation and

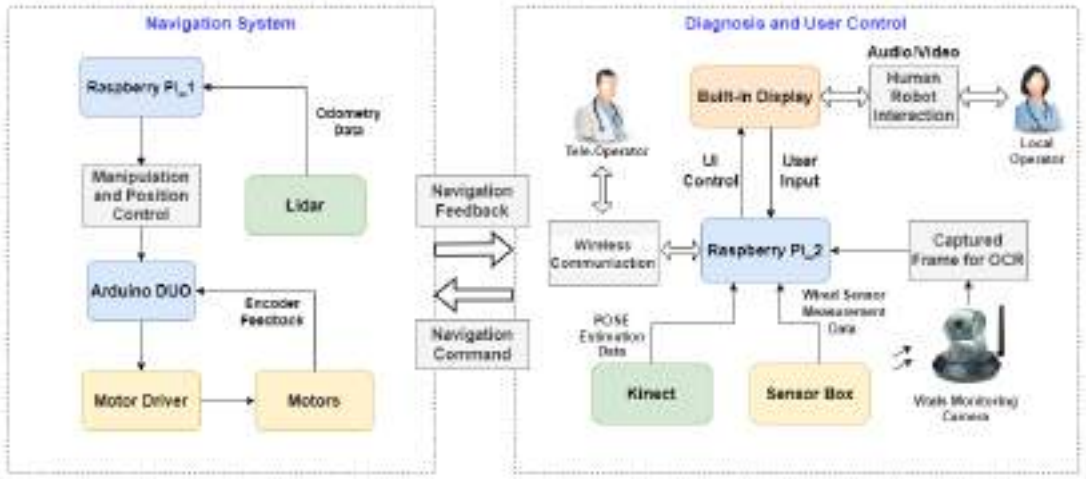


Figure 3.3: System Design of IHABOT.

mapping task is assigned to the first RPi. Using Robot operating system framework odometry data is being collected from the lidar continuously. Initially the hospital environment is mapped. Then using that generated map robot can navigate itself avoiding the obstacles. Both pairs of right and left side motors' encoder data is being averaged separately to avoid slip of wheel while rotation is performed. All the navigational simulation is being fed back to the teleoperation device where only the doctors have access to. Navigation feedback is also sent to the other module to perform diagnosis and human interface tasks smoothly.

2nd RPi is purposed for diagnosing feature control and all other human robot interaction tasks. It is wirelessly connected to a tab mounted on the chest area of the robot. A Kinect sensor is also incorporated to evaluate patients' exercise while other modules are not being used.

When IHABOT reaches the patients, its display will instruct them on how to measure their vitals. Robot will first request that the patient open the front lid. Patients will then be asked to take the temperature probe and hold it for some amount of time until guided to put back. Temperature probe is the only wired sensor that is directly connected to the controller. Afterwards, the robot will navigate patients to put their index finger inside

the attached pulse oximeter to measure their oxygen saturation. Finally patients will be required to put on the arm cuff to measure blood pressure and close the lid. Detecting if the lid is closed, Internal vitals monitoring camera will capture the display of those vital measuring devices and follow through the OCR of that frame to store vitals' data for later diagnosis tasks.

3.4 IHABOT-User Interface

3.5 Conclusion

It is said that human structure is the best example for better stability a mobile robot designer can be motivated from. They don't necessarily need to imitate human maneuvering, moreover the design can be customized according to the application needs for efficient use of resources. We designed IHABOT considering all those parameters.
What were the main design considerations along with functionality and behavioral coordination is presented in this section. Since the IHABOT's main objective was to build an affordable hospital assistant robot that can be used to eliminate direct interaction with the patient, we believe, this purpose is served at a traded off level with multimodal functionality incorporating some important diagnosis systems.
IHABOT is only focused on low level of impaired patients who can measure vitals themselves with proper guidelines reducing the contagious diseases spreading to healthcare professionals. In future, we are also planning to implement the self charging system with self sterilizing features.

Chapter 4

Automated Navigation

One of IHABOT's primary responsibilities is being able to locate the patient correctly in order to gather data from the patient's body's numerous sensors and transmit that information back to a distant server so that a medical expert may continuously monitor the patient's condition. IHABOT must complete its navigation task using Localization stacks, Perception stacks, and Motion planning stacks. It should be able to accurately perceive its position in a given environment using the localization stacks. If it doesn't know where it is, it won't be able to collect the necessary data from patients. Therefore, localization is a crucial step for IHABOT. A map is necessary for the localization stack to localize itself. It will use the map to plan its route to a particular location. Without a map, it won't be able to plan its travel for a certain mission. We've covered two crucial topics: correctly localizing itself and creating a solid map for it to operate efficiently. If any one of IHABOT's functions is not properly maintained, the system's entire goal will be jeopardized. The next challenge is determining the quickest path to its destination, such as a patient's location. It will be able to assess its surroundings and plot a short-cut around any impediments, both stationary and moving. Therefore, we must focus on three key issues in order to enable proper self-navigation:

1. Localization
2. Perception

3. Motion planning or Path planning

We'll go into great depth about how we overcame those challenges and given IHABOT the ability to navigate independently in this section. Robot Operating System (ROS) has been utilized for the implementation.

4.1 Localization

Finding a robot's stance in a pre-mapped environment is called localization. In order to track the robot's position and orientation, we construct a probabilistic technique to filter noisy sensor measurements. The Adaptive Monte Carlo Localization (AMCL) method is utilized to localize our robot in a known map. An expanded form of MCL is AMCL. Let's first examine how MCL functions.

4.1.1 What is Monte Carlo Localization (MCL)

The most widely used localization technique in robotics is the Monte Carlo Localization algorithm, or MCL. The robot is lost in a particular two-dimensional map and has no idea where it is. The robot is attempting to determine its pose by resolving the global localization problem because the beginning state is unknown.

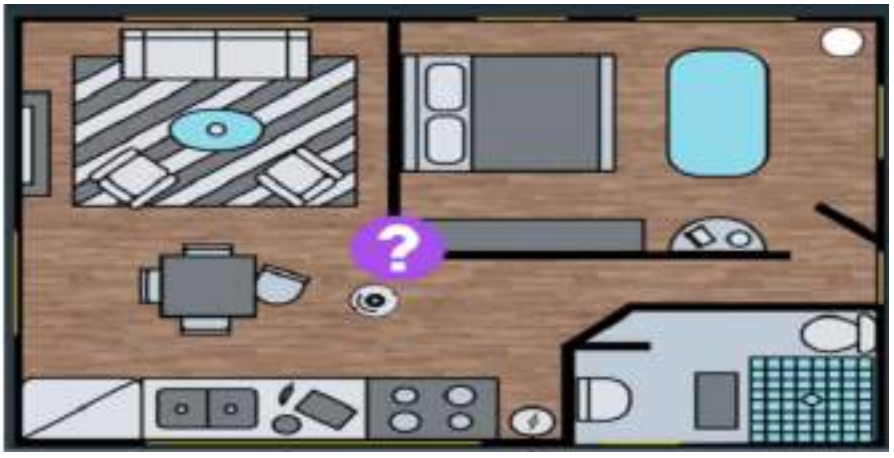


Figure 4.1: A robot is unknown about its environment

The robot has sensors built inside it that allow it to detect barriers like objects and walls and finally pinpoint its location.

Figure 4.2: Robot is measuring its position using its onboard sensor

Here, the x, y, and orientation vectors of the current robot pose are all expressed in terms of the global coordinate frame.

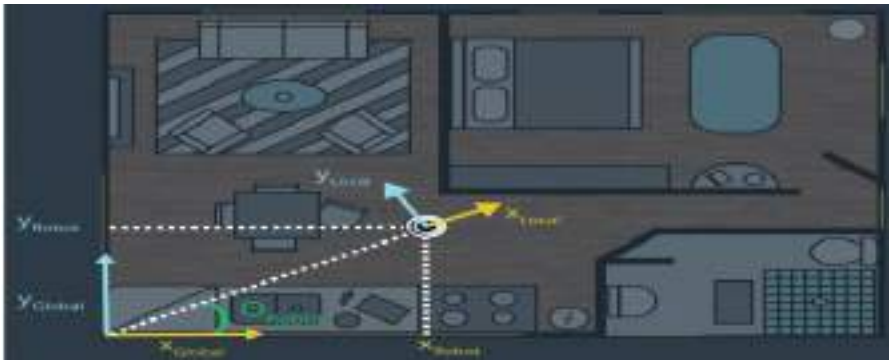


Figure 4.3: Robot's current true pose in map

The Monte Carlo localization approach distributes particles uniformly and randomly over the entire map at first. These particles are just depicted in simulation and do not actually exist. Similar to the robot, each red circle denotes a single particle, and each particle has an x coordinate, y coordinate, and orientation θ. Therefore, each of these particles indicates a potential location for the robot. Particles are given a weight in addition to the tree-dimensional vector. The discrepancy between a particle's expected attitude and the robot's actual pose is the particle's weight. A particle's weight determines its significance, and larger particles tend to be more accurate. Large-weight particles have a higher chance of surviving the re-sampling procedure. For instance, P2 will have a higher probability of surviving than P1 since it is heavier and hence has a

larger weight. Particles with a lot of weight are more likely to survive the re-sampling procedure whereas the others are more likely to perish.

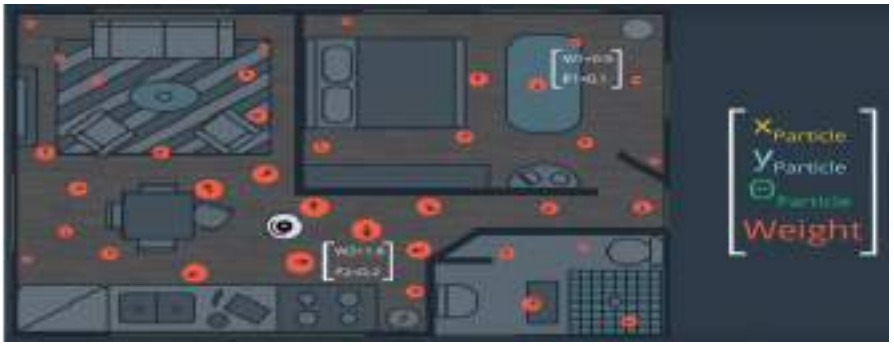


Figure 4.4: Initialization of particles for localizing robot's pose

Particles will finally converge and estimate the robot's pose after numerous runs of the Monte Carlo localization method and after various levels of resampling.

Figure 4.5: Concentrated particles according to robot's true pose

Based on sensory data, the robust Monte Carlo localization algorithm calculates the posterior distribution of a robot's position and orientation. A recursive Bayes filter is what this procedure is called. Roboticists can infer the state of a dynamical system from sensor measurements by using a Bayes filtering strategy.

The following definitions are crucial to understand when localizing mobile robots:

- Dynamical system: The mobile robot and its environment
- State: The robot's pose, including its position and orientation

- Measurements: Perception data(e.g. laser scanners) and odometry data(e.g. rotary encoders)

With the help of measurements, Bayes filtering aims to estimate a probability density over the state space. The belief is represented by the probability density, also referred to as posterior, and is written as follows:

$$Bel(X_t) = P(X_t|Z_{1...t}) \tag{4.1}$$

where:

X_t : State at time t

$Z_{1...t}$: Measurements from time 1 up to time t

Algorithm 1 Monte Carlo localization(MCL) Algorithm

1: procedure MCL(x_{t-1}, u_t, z_t)
2: Initialize:
$X_t \leftarrow \phi$
3: for m=1 to M do
4: $x_t^{[m]} \leftarrow$ MotionUpdate($u_t, x_{t-1}^{[m]}$)
5: $w_t^{[m]} \leftarrow$ SensorUpdate($z_t, x_t^{[m]}$)
6: $X_t \leftarrow X_t + <x_t^{[m]}, w_t^{[m]}>$
7: end for
8: for m=1 to M do
9: draw $x_t^{[m]}$ with probability $\propto w_t^{[m]}$
10: $X_t \leftarrow X_t + x_t^{[m]}$
11: end for
12: return X_t
13: end procedure

where:

x_{t-1} : Previous Belief

u_t : Actuation Command

z_t : Sensors Measurements

The two main portions of the Monte Carlo Localization Algorithm are each represented by a pair of for loops. The motion and sensor update is covered in the first section, and the re-sampling procedure is covered in the second. The MCL seeks to identify the robot's stance represented by the belief given a map of the surroundings. The algorithm inputs the previous belief, the actuation command, and the sensor readings at each iteration.

Initially, m particles are generated at random to create the belief. The hypothetical state is then calculated each time the robot travels in the first for loop. The weight of the particle is then calculated using the most recent sensor readings.

The prior state is now add to both motion and measurement.

The MCL's second part is where a re-sampling procedure takes place.Here, only the particles with a high likelihood of surviving are drawn again in the following iteration.

The algorithm then outputs the modified belief, and a new cycle of iterations begins by implementing the subsequent motion based on the updated sensor readings.

4.1.2 Adaptive Monte Carlo Localization (AMCL)

For a robot moving in two dimensions, AMCL is a probabilistic localization system. It applies the adaptive (or KLD-sampling) Monte Carlo localization method, which tracks a robot's pose against a known map using a particle filter. An extremely large number of particles covering the entire state space are used to initialize particle filters. A robot can have a multi-modal posterior distribution when additional data are collected from different types of sensors, such as sonar or laser sensors, and the forecast of the robot's position is updated in accordance with the measurements. A Kalman Filter approximates

the posterior distribution to be a Gaussian, which is a significant departure from this. The particles converge to a single value in state space after several rounds.

The steps followed in a Particle Filter are:

- Re-sampling: Draw a random sample from the sample set using replacement, following the (discrete) distribution established by the important weights. This example serves as an illustration of the notion.
- Sampling: To select a sample from the distribution that captures the dynamics of the system, use the control information and prior belief. The density determined by the product of the distribution and an instance of the prior belief is now represented by the current belief. The proposal distribution used in the following phase is based on this density.
- Importance sampling: Samples are weighted based on importance and sample X's likelihood given measurement Z.

These three procedures generate a sample from the posterior belief with each iteration. The samples' significance weights are adjusted after n iterations so that they add up to 1.

Maintaining the random distribution of particles over the state space is a major challenge for particle filters, which becomes impossible for large dimensional problems. These factors make an adaptive particle filter far superior to a simple particle filter in terms of convergence speed and computational efficiency.

The main concept is to limit the inaccuracy caused by the particle filter's sample-based representation. The real posterior is thought to be represented by a discrete, piecewise constant distribution, such as a discrete density tree or a multidimensional histogram, in order to calculate this bound. We can choose the number of samples for such a representation so that the difference between the true posterior and the maximum likelihood estimate (MLE) based on the samples does not exceed a predetermined threshold. The number of particles required is proportional to the inverse of this threshold, as is ultimately determined.

Starting with a map of the environment, we may either manually localize the robot by setting it to a certain point, in which case we are using adaptive particle filters, or we could very well have the robot start from with no initial estimate of its position. We now create fresh samples that forecast the robot's position following the motion command as it advances.

Sensor readings are incorporated by re-weighting these samples and normalizing the weights. Generally it is good to add a few random uniformly distributed samples as it helps the robot recover itself in cases where it has lost track of its position. In those cases, without these random samples, the robot will keep on re-sampling from an incorrect distribution and will never recover. The reason why it takes the filter multiple sensor readings to converge is that within a map, we might have dis-ambiguities due to symmetry in the map, which is what gives us a multi-modal posterior belief.

4.2 Mapping and SLAM

A key component of robot navigation is mapping. A robot needs to be aware of its location in a given environment at a given time. SLAM is used to map a scene and localize a robot at the same time. To create a map, we utilized gmapping, a ros-based package. To learn grid maps from laser range data, one can use the very effective Rao-Blackwellized particle filter GMapping. The Grid-based FastSLAM technique is used by the gmapping ROS package to map the surroundings and determine the robot's path. The following algorithms are utilized by the Grid-based FastSLAM algorithm:

- Monte Carlo Localization (MCL)
- Occupancy Grid Mapping

As was demonstrated in the last section, how MCL functions. Let's examine the inner workings of occupancy grid mapping.

Figure 4.6: Construction of Measurement Cone using sonar sensor

4.2.1 Occupancy Grid Mapping

The binary base filter is used by the occupancy grid mapping technique to calculate the occupancy value of each cell. The algorithm's initial parameters include the postures, measurements, and prior cell occupancy values. The program now iterates through every grid cell, checking each one to see if the rangefinder sensors are currently picking it up. Let's look at an example to help us better comprehend this circumstance. A mobile robot is being seen here detecting its surroundings. White and black are used to emphasize the cells that are within the measurement cone. The algorithm will take into account those white and black cells as cells falling under the perceptual field of the measurements while iterating through all the cells. The algorithm will now compute the new belief for the cells that are covered by the measurement cones using the binary base filter technique. The inverse sensor model is combined with the prior belief to calculate the cell's new state. The probability of the posterior map, given the measurements and poses, and after deducting the initial belief, which is the map's starting state in its log odd form, is represented by the inverse sensor model. The poses and measurements are inputs into the inverse sensor model algorithm for each cell.

Algorithm 2 Occupancy Grid Mapping Algorithm

```
1: procedure OGD({l_{t-1,i}}, x_t, z_t)
2:     for all cells m_i do
3:         if m_i in perceptual field of z_t then
4:             l_{t,i} ← l_{t-1,i} + Inverse_Sensor_Model(m_i, x_t, z_t) − l_0
5:         else
6:             l_{t,i} ← l_{t-1,i}
7:         end if
8:     end for
9: end procedure
```

where:

l_{t-1} : Previous Belief

z_t : Sensors Measurements

The occupancy value of the cells outside of the measurement cone does not change. The algorithm now returns the modified cell occupancy values, and another iteration cycle begins.

4.2.1.1 Inverse Sensor Model

We observed the inverse sensor model in the occupancy grid mapping algorithm, which is the likelihood of the map providing the measurements and poses in its log odds representation. Let's retrieve the likelihood that the map will provide the data from the log odds Representation to show that.

$$Inverse\_Sensor\_Model(m_i, x_t, z_t) = \log \frac{p(m_i|z_t, x_t)}{1 - p(m_i|z_t, x_t)}$$

We can now see how the map's probability is limited to values between zero and one. Mobile robots using range finder sensors can measure barriers in a conical field of vision

and instantly detect them. Robots can only detect a small part of these cells at the cone's edges, making the chances of detection weaker. It is difficult and complex to compute the state of the cells at the cone's boundary due to poor probability. An inverse sensor model is presented to streamline this issue and estimate the condition of the cells inside, outside, and adjacent to the measurement cone. With the help of this technique, you may determine if a cell is empty, occupied, or unknown.

4.2.2 Grid-based FastSLAM Techniques

The grid-based FastSLAM algorithm was created by modifying the FastSLAM algorithm for grid maps. The particle filter technique is used by the grid-based FastSLAM algorithm to represent the world using grid maps.

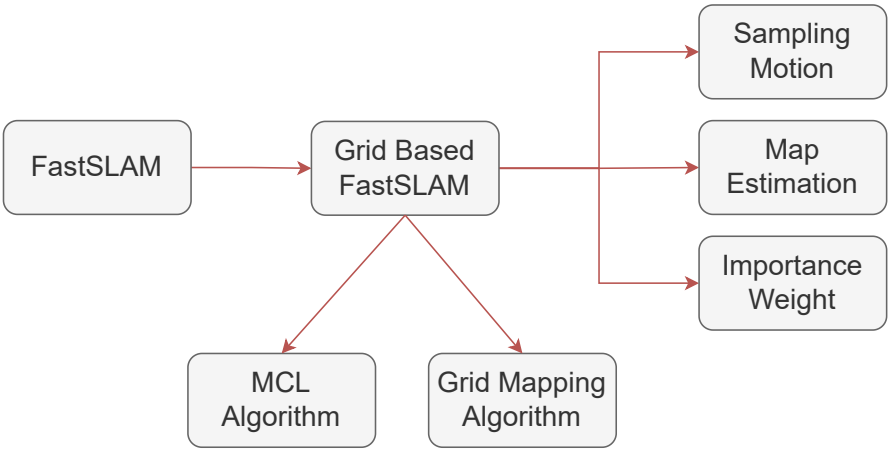


Figure 4.7: Composition of FastSLAM

To adapt FastSLAM to grid mapping, we need three different techniques:

- Sampling Motion-$P(z_t|x_{t-1}[k], u_t)$: Estimates the current pose given the k-th particle previous pose and the current controls u, with the MCL algorithm.
- Map Estimation-$P(m_t|z_t, x_t[k], m_{t-1}[k])$: Estimates the current map given the current measurements, the current k-th particle pose, and the previous k-th particle map, with the Occupancy Grid Mapping algorithm.

- Importance Weight-$P(z_t|x_t[k], m_t[k])$: Estimates the current likelihood of the measurement given the current k-th particle pose and the current k-th particle map, with the MCL algorithm.

The Grid-based FastSLAM algorithm's core components are the sampling motion, map estimation, and importance weight approaches. They are put into practice to provide measurements and controls while also estimating the map and robot trajectory.

Here is the Grid-based FastSLAM algorithm, which resembles the MCL technique quite closely and includes some more information about map estimate.

Algorithm 3 Grid-Based FastSLAM Algorithm

1: procedure FastSLAM(x_{t-1}, u_t, z_t)
2: Initialize:
$X_t \leftarrow \phi$
3: for k=1 to M do
4: $x_t^{[k]} \leftarrow$ Sample_Motion_Model($u_t, x_{t-1}^{[k]}$)
5: $w_t^{[k]} \leftarrow$ Measurement_Model_Map($z_t, x_t^{[k]}, m_{t-1}^{[k]}$)
6: $w_t^{[k]} \leftarrow$ Updated_Occupancy_Grid($z_t, x_t^{[k]}, m_{t-1}^{[k]}$)
7: $X_t \leftarrow X_t + <x_t^{[k]}, m_t^{[k]}, w_t^{[k]}>$
8: end for
9: for k=1 to M do
10: draw i with probability $\propto w_t^{[i]}$
11: $X_t \leftarrow X_t + <x_t^{[i]}, m_t^{[i]}>$
12: end for
13: return X_t
14: end procedure

In actuality, the MCL algorithm and the Occupancy Grid Mapping algorithm were combined to create this algorithm. The Grid-based FastSLAM method is divided into two portions, which are each represented by two for loops, just as the MCL algorithm. The motion, sensor, and map updates are covered in the first section, and the re-sampling procedure is covered in the second.

Step 1: Previous Belief

Step 2: Sampling Motion

Step 3: Importance Weight

Step 4: Map Estimation

Step 5: Re-sampling

Step 6: New Belief

4.3 Path Planning

So far, our attention has been on the issues of localization—where a robot is in relation to a map—and mapping—how to create a map of the environment. Both of these are essential components for robotic mobility. The decision-making features of mobile robotics will now be the focus of our discussion. Navigation and route planning. Path planning is the process of figuring out the route that will get the robot to its destination given a map and a goal location. Once a route has been established, the robot must follow it to its destination, where it may run into unanticipated obstacles that obstruct its intended course. The robot can alter its initial path using obstacle avoidance strategies based on real-time sensor readings. There are three methods for planning a path. The first strategy, known as discrete (or combinatorial) path planning, is the simplest of the three. Using the foundation of discrete planning as a framework, the other two approaches—sample-based path planning and probabilistic path planning—will create more broadly applicable path planning solutions.

4.3.1 Discrete Planning

In discrete planning, the robot's workspace is explicitly discretized into a connected graph, and the best path is determined using a graph-search method. This process discretizes the entire workspace, making it incredibly thorough and precise (the precision

can actually be expressly changed by altering how fine you choose to discretize the space). Consequently, discrete planning may be prohibitively expensive in terms of computation for big path planning tasks. Although discrete path planning is precise and elegant, it works best with low-dimensional issues. Sample-based path planning is a better strategy for high-dimensional situations.

4.3.2 Sample-Based Planning

Sample-based path planning explores the workspace to build a graph piecemeal. Sample-based planning employs a number of samples to create a discrete representation of the workspace rather than discretizing every area of the workspace. Due to the relatively limited number of samples utilized, the resulting graph is not as exact as one made using discrete planning, but it is constructed considerably more quickly. In some cases, it is preferable to develop a feasible path fast rather than waiting hours or even days to construct the ideal path, even though the path produced using sample-based planning may not be the greatest option.

4.3.3 Probabilistic Path Planning

Probabilistic route planning takes into consideration the unpredictability of the robot's movements, unlike the prior two approaches, which approached the path planning problem in a generic way without considering who or what might be carrying out the activities. In some contexts, it might not offer much of a benefit, but in others, such those with sensitive or high-risk locations, it might be very beneficial.

4.3.4 Sample-Based and Probabilistic Path Planning

To choose the best route from a robot position to a specified goal in a known environment, we employed sample-based and probabilistic path planning in our study. Therefore, in this section, we will briefly cover sample-based and probabilistic path planning. In contrast to combinatorial path planning, sample-based path planning does not attempt to

systematically discretize the full configuration space. Instead, it builds a representation of the space by randomly (or semi-randomly) sampling the configuration space. Due to the relatively limited number of samples utilized, the resulting graph is not as exact as one made using combinatorial planning, but it is constructed much more quickly. The likelihood of discovering a path, if there is one, approaches 1 as time goes on and the number of samples gets closer to infinity, making this strategy probabilistically complete. Although this method has significant drawbacks, it is particularly effective in high-dimensional spaces. A homogeneous sampling of a space is unlikely to cover tiny or constrained spaces, as the passage seen in the image below. A significant amount of samples must occupy the passage because it is the sole path from start to objective. If not, the algorithm will return "no solution found" for a problem that obviously has a solution. There are numerous sample-based planning strategies, each with distinct advantages and disadvantages.

The following sections will teach you about

- Probabilistic Roadmap Method
- Rapidly Exploring Random Tree Method

4.3.4.1 Probabilistic Roadmap Method(PRM)

The probabilistic roadmap, or PRM for short, is one popular sample-based path planning technique. PRM builds a graph to represent the free space by randomly sampling the workspace. It accomplishes this without having to build or discretize the C space. To determine if a randomly produced node is in the freespace or is in collision with an obstruction, PRM just needs a collision check function.

Let's increase the number of random samples in this workspace and examine how PRM functions. The learning phase is the process of creating a graph. Seeing as how PRM adds random configurations to the graph by sampling them, this is what it accomplishes. By creating a fresh random configuration represented by a graph node and determining if it collides with any existing nodes, it does this.

Figure 4.8: Generated node is in occupied space

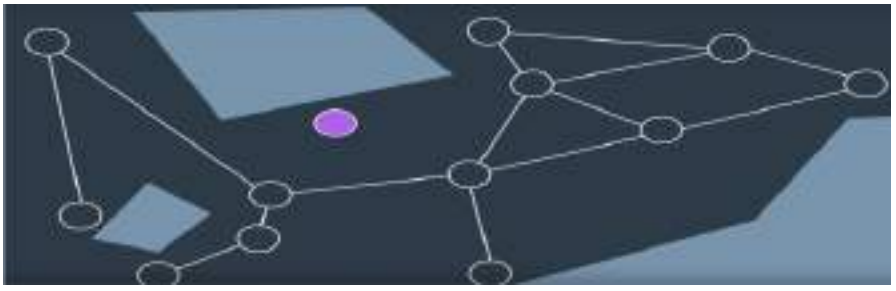


Figure 4.9: Sampling nodes in free space

If it differs from the node you can see above, PRM will attempt to connect it to its neighbors. There are several methods for doing this. PRM can either search for the node's K nearest neighbors or any number of neighbors within a predetermined radius of the node. Once the neighbors have been chosen, PRM will test whether it can successfully provide each of them an edge.

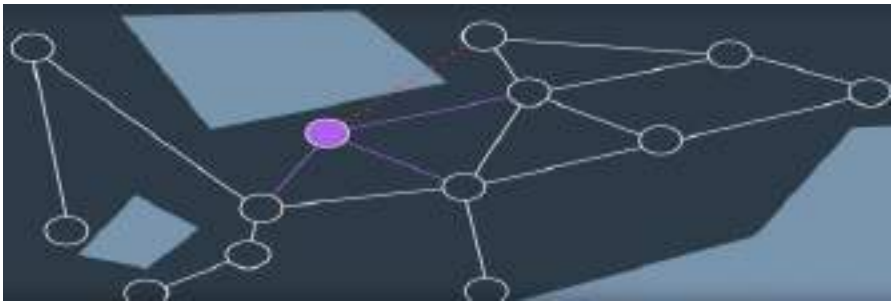


Figure 4.10: Checking if constructed edge is in occupied space or not

As you can see in the image above, three of the edges are free to be added while one is colliding with a barrier. The technique can now be repeated for a different randomly generated node as this node has been added to the graph. Since this procedure is repeated for each neighbor of each new node, the local planner must act swiftly to discover a connection connecting two nodes or return that such a link does not exist. Drawing a straight line between two nodes and then checking to see if any portion of it collides

with an obstruction is one quick technique to achieve this.You may check to see if any samples are colliding with each other by placing a number of them on the line with equal spacing. You have two options: either take an incremental approach, testing the sample at the midpoint first, or a binary method, checking it at the edge first. In this instance, that immediately returns a collision.

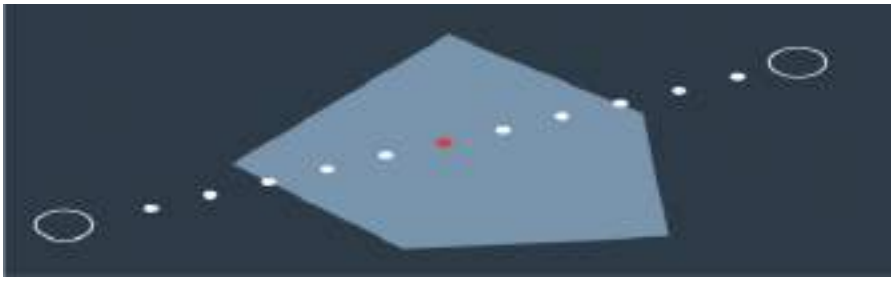


Figure 4.11: Checking if constructed edge is collisioned with obstacle

If not, you could keep segmenting the edge and check each midway sample for a collision; if none were detected across all samples, the edge could then be added to the graph.

Figure 4.12: Construction of edge in free space

The process of creating new nodes and linking them to the graph continues until a predetermined condition is satisfied, such as when a predetermined number of nodes or edges have been generated or when a predetermined period of time has passed. The learning period is now complete. The query phase of PRM then starts. where a path from the start to the goal is found using the resulting graph. The first step is to connect each of them to the graph. PRM achieves this by locating the nodes that are closest to the start and goal and attempting to connect them using the local planner. If this procedure is successful, the path from the beginning to the end can be discovered using a search algorithm like A*.

The resulting path may not be optimal, but it proves that moving from start to goal is feasible.

Figure 4.13: Robot's motion path using PRM

Algorithm 4 The pseudocode for the PRM learning phase

```
procedure PRM
    Initialize:
    an empty graph
    for n iterations do
        Generate a random configuration
        if the configuration is collision free then
            add the configuration to the graph
            for each of the k neighbors do
                find a collision-free path between the neighbor and original configuration
                if edge is collision-free then
                    add the edge to the graph
                end if
            end for
        end if
    end for
end procedure
```

Rapidly Exploring Random Tree Method (RRT) The randomly explored random tree approach, or RRT for short, is another frequently applied sample-based path design technique. Since RRT is a single query planner, it is different from PRM. You may remember that PRM spent the majority of its learning phase creating a representation of the entire workspace. Although this required a lot of calculation, the resulting graph can be utilized for several queries. For each unique query, RRT creates a new graph while taking into account the start and target positions, obviating the necessity for a comprehensive graph. As a result, the graph becomes significantly smaller, more directed, and has a shorter computing time. Some situations change too quickly, and the RRT approach works well in these environments. PRM is perfect for static environments where you can reuse the graph. We are path planning in the same setting as before with the same start and objective setups. Let's see how the RRT approach appears. However,

these will be expressly taken into account from the beginning rather than being added throughout the learning period. Then we begin to construct a model of the workspace; the PRM constructs a graph, whereas the RRT constructs a tree, a sort of graph in which each node has a single parent. The lack of lateral connections between seemingly neighboring nodes is less of an issue when using a single query planner because you are just concerned with getting from start to goal.

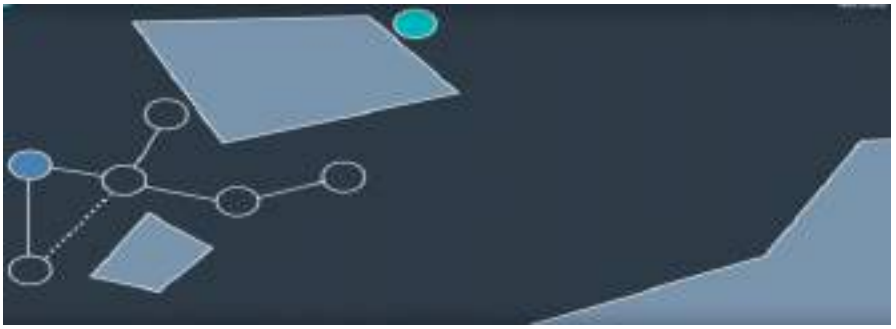


Figure 4.14: Tree building procedure in RRT algorithm

What then is the algorithm?
A node will be generated using RRT at random, and if it is within a certain delta distance of its nearest neighbor, it can be connected directly. Naturally, if the local planner finds the edge to be free of collisions.

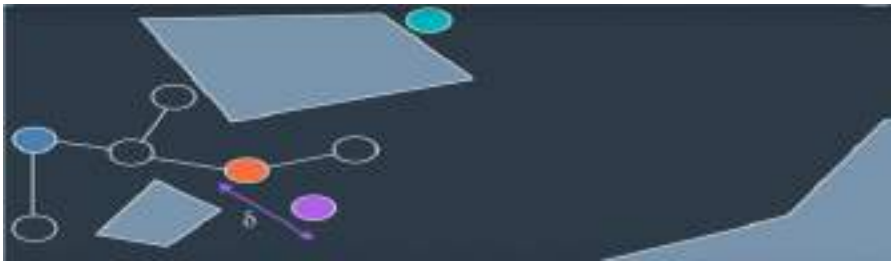


Figure 4.15: Generating collision free the closest neighbor node

The likelihood of the edge between a freshly formed node and its nearest neighbor being collision-free is less likely if the node is remote from all other nodes. In this scenario, RRT will generate a new node in the same direction but a distance delta away rather than connecting to current node.

Next, if there are no collisions along this edge, the node is added to the tree. Nodes can be produced by randomly selecting points from the search space, which would prefer

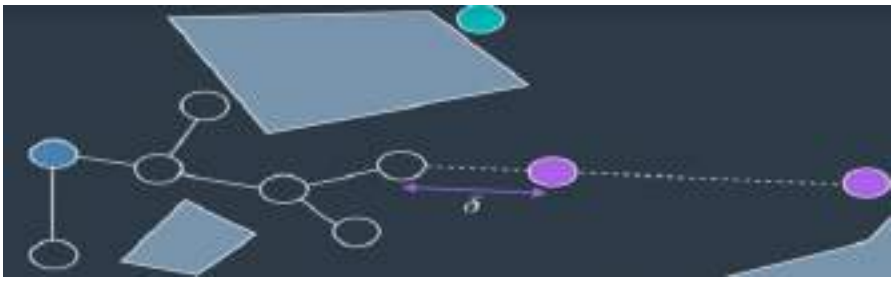


Figure 4.16: Generating more collision free the closest neighbor nodes

large, unknown areas, or alternatively, some greediness can be introduced by raising the likelihood of sampling close to the target, which would slant new samples in the direction of the goal. RRT is a single query planner, hence it is frequently advantageous to incorporate very small biasing. Growing two trees is one of the RRT method's variations both from the beginning and the end. In order to create an edge between the tree with the most recently added node and the other tree, RRT grows each tree in turn.

Figure 4.17: Final Robot's motion path

It ultimately succeeds. RRT is aware that there is a direct route from the beginning to the end.

This workspace is a greatly simplified illustration and not a true picture of the issue that PRM or RRT would be expected to address. In multidimensional spaces, PRM and RRT exhibit outstanding performance for robots with multiple degrees of freedom. In fact, they have proved successful in resolving issues that more established path planning algorithms are unable to.

Algorithm 5 The pseudocode for the RRT learning phase

1: procedure RRT($q_{init}, K, \Delta q$)
2: Initialize:
$G.init(q_{init})$
3: for k = 1 to K do
4: $q_{rand} \leftarrow RAND\_CONF()$
5: $q_{near} \leftarrow NEAREST\_VERTEX(q_{rand}, G)$
6: $q_{new} \leftarrow NEW\_CONF(q_{near}, q_{rand}, \Delta q)$
7: $G.add_vertex(q_{new})$
8: $G.add_edge(q_{near}, q_{new})$
9: end for
10: return G
11: end procedure

where:

q_{init} : Initial configuration

K : number of vertices in RRT

Δq : incremental distance

4.3.4.2 RRT & Non-holonomic Systems

We won't go into great depth here, but the RRT approach and the PRM method both provide planning for non-holonomic systems. This is because, just as it already takes into account how far a new node is from an existing tree, the RRT approach may also take into account additional limitations (such a car's turning radius at a specific speed) when adding nodes to a graph.

- Path planning for non-holonomic systems can use RRT

- The efficiency of the procedure will decrease if δ is set to a large value since the local planner is more likely to run into collisions along a longer path

4.4 Result

4.4.1 Environment

This a simple L shape environment where we test our IHABOT robot. As it is seen in 4.18

Figure 4.18: Test Environment

4.4.2 Mapping & Localization

The robot created a map in the environment using the gmapping approach with the help of ROS package and starting to localization in the mapped environment.

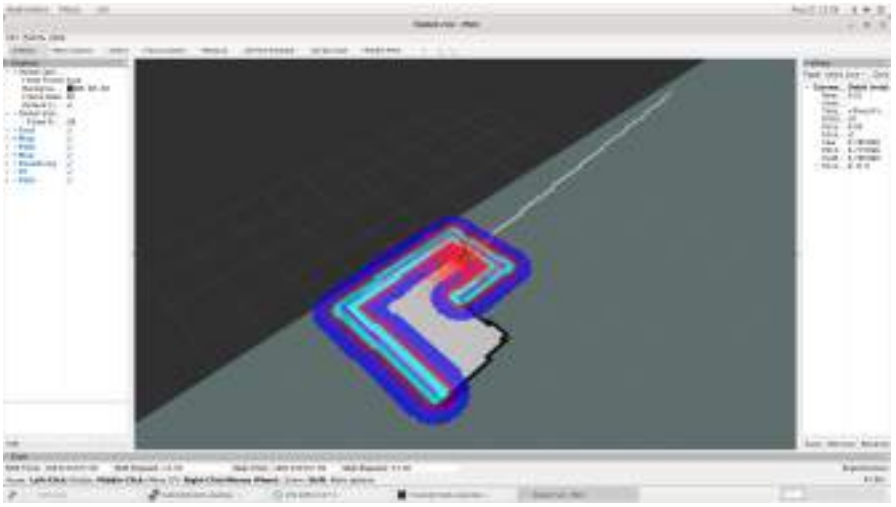


Figure 4.19: Robot Localizing in the Environment

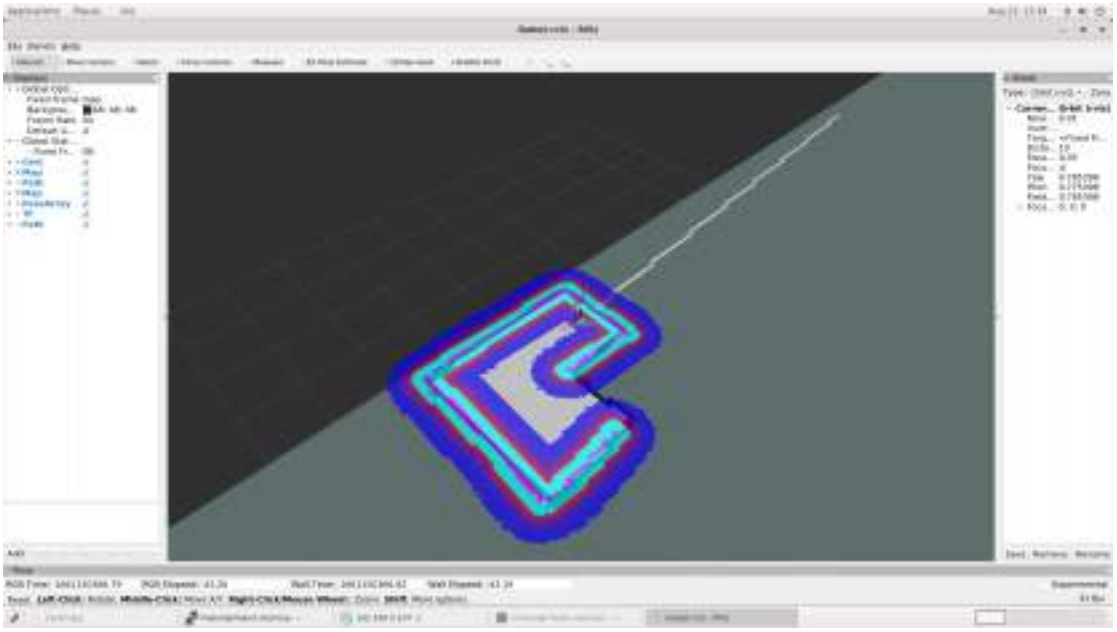


Figure 4.20: After Localization

4.4.3 Path Planning

Using the RRT algorithm, IHABOT finds it optimal path in the environment. In the 4.21 the starting position & goal position is defined. And in the 4.22, the robot is on the move to goal from the source.

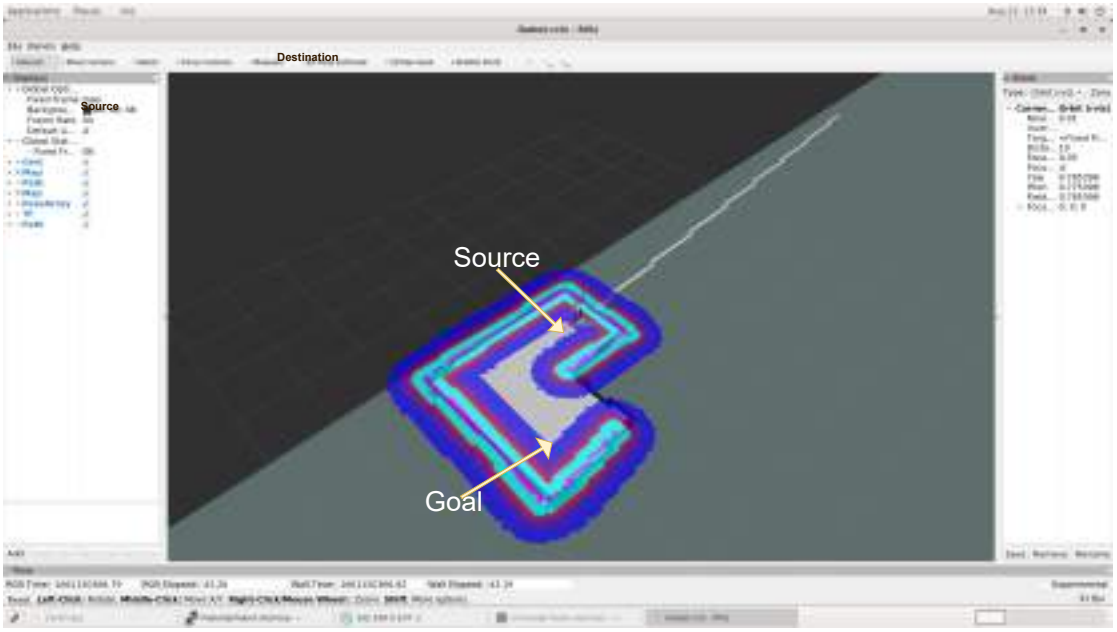


Figure 4.21: Source & Destination Selection

After selecting the source and goal, IHABOT finds an optimal path (Source → A → B → C → Goal). Which we can see from figure 4.22.

Now if an obstacle is placed in the middle of its path, IHABOT can pass through the obstacle. We can see the result in the figure 4.23. Instead going from B point to C point (because an obstacle is placed in the B point), it creates a different path and passes through B' → C' point to the goal position.

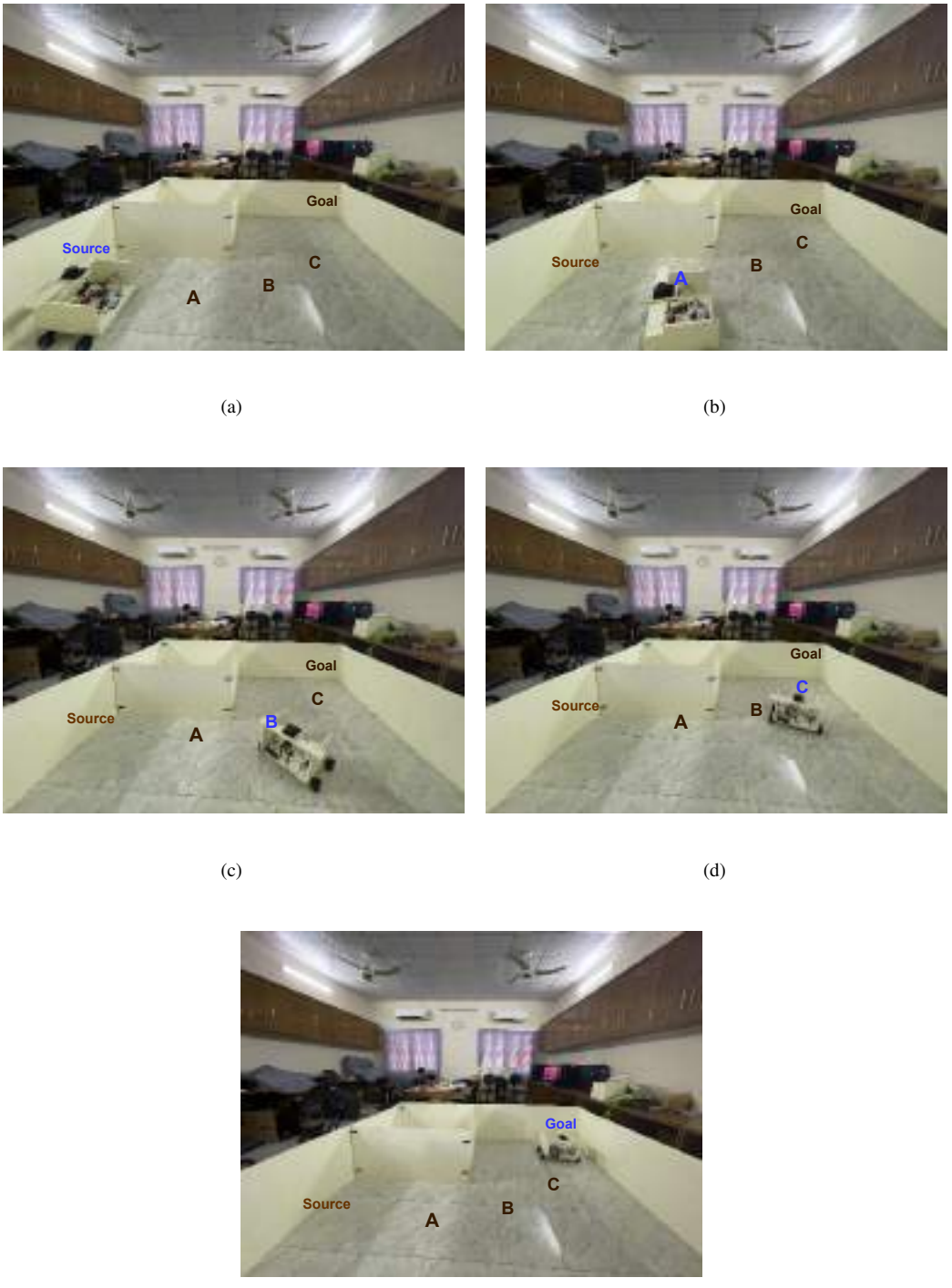


Figure 4.22: IHABOT is moving towards from source to destination by building an optimal path: (a) Source; (b) First Step; (c) Second Step; (d) Third Step; (e) Goal

(a) (b)

Figure 4.23: IHABOT is moving towards from source to destination where an obstacle is situated in its optimal path: (a) Crossing alternating of B point; (b) Crossing alternating of C point

Chapter 5

Automated Covid-19 Detection from Radiography Images

5.1 Introduction

Human can learn new ideas with very little and direct supervision. For example, a child can generalize the idea of “Zebra” from a single picture. In contrast, deep learning models need thousands of training examples to learn the concept of "Zebra". So, This motivates us to work with a new setting in computer vision “Few-shot” learning, which consists of learning from a very few labelled examples. Few-shot Learning (FSL) is a sub-filed of machine learning. So before giving the definition of FSL, let us recall how machine learning is defined in the literature.

Definition 1.1.1 (Machine Learning) [65]. A computer program is said to learn from experience E with respect to some classes of task T and performance measure P if its performance can improve with E on T measured by P.

Table 5.1: Machine learning example based on definition 1.1.1

Task T	Experience E	Performance P
image classification	large scale image example for each class	classification accuracy

FSL is a special case of machine learning, which tries to obtain good learning performance using limited supervised information from the training set D_{train},which consists of instance of inputs $x_i's$ along with their corresponding output $y_i's$. Formally, we define FSL in Definition 2.2.

Definition 1.1.2. "Few-Shot Learning (FSL) is a type of machine learning problems (specified by E, T and P), where E contains only a limited number of examples with supervised information for the target T". [66] Now table 1.2 has one extra column under

Table 5.2: FSL example based on definition 1.1.2

task T	Experience E		Performance P
	supervised information	prior knowledge	
image classification	a few labeled images for each class of the target T	raw images of other classes or pre-trained models	classification accuracy

“experience E" which is marked as “prior knowledge" compared to table 1.1. As E contains only a few examples with supervised information directly, it is normal that usual supervised learning algorithms fail on FSL problems. Therefore, "FSL methods make the learning of target T feasible by combining the available supervised information in E with some prior knowledge, which is any information the learner has about the unknown function before seeing the examples" [66].

Remark 1. Few shot learning has variation regrading the supervised information is provided. In case of only one example with supervised information in E, FSL is called one shot learning (OSL). In extreme case, when experience E does not contain any example with supervised information, FSL becomes zero shot learning (ZSL).

Few-shot learning (FSL) is commonly studied using C-way-K-shot setting. The aim is to differentiate between C classes with K samples of each. For example a classification problem might be to classify between C=10 classes with only K=5 samples from each class. Our modern classification approaches is dependent on millions of parameters and can not generalize sufficiently. So, in this setting the model can not be trained using conventional methods. If the data is inadequate to constrain the task, then one possible solution will be to share experience from other similar tasks. Few-shot learning follows this approach. So, it can be characterized as a meta-learning problem. To this end, few-shot learning focuses to recognise novel categories from a few labelled samples. The availability of very few samples challenges ’the generalization’ and ’fine-tuning’ process in deep learning [67]. Data augmentation and regularisation techniques can

improve overfitting in such scheme, but they do not solve the issue. On the other hand, the approaches on few-shot classification separate training into meta-learning phase. In meta-learning, the transferrable knowledge is passed to the network by good initial conditions [67], embedding modules [68] [69], and different optimisation strategies [70]. Then the target learning tasks are learned by fine-tuning with the transferred optimisation strategy and evaluated in a forward pass [68], [69], [71], [72] without updating network's parameters.

Detecting diseases and solving clinical problems by analyzing clinical images are known as medical image analysis. It's motivation is to help radiologist and clinicians to make diagnostics and treatments more efficient. The computer aided diagnosis (CAD) and computer aided detection (CADx) is need to be effective in terms of performance, since it is directly involved in the process of diagnosis and treatment. Therefore it is desirable to expect high value from the performance parameters such as accuracy, F-measure, Precision, Recall, Sensitivity and Specificity in medical image analysis.

From last few years, many biomedical health problems and diseases (e.g. brain tumor detection, skin cancer detection, breast cancer detection) are using Artificial Intelligence (AI) based solutions [73], [74], [75], [76]. One of the important power Deep learning techniques have is it can reveal image features, which are obscure in the original images. Specially, Convolutional Neural Network (CNN) has been proven very useful in extracting features and learning from images, which makes it widely adopted by the research community [77]. Deep learning techniques on chest X-Rays images are getting popularity with the availability of the deep CNNs and the promising results it has shown in different applications in biomedical field with the help of transfer learning because of data unavailability [73]. Also in case of COVID-19 data availability remain as the biggest problem. So instead of using different pre-trained CNN models, few-shot Learning can be useful as it enables a deep model to learn information from only a handful of labeled examples per category. Recently, Few-shot learning has been applied on chest x-ray images to classify some lung diseases where it performed promisingly well [78]. So, Few-shot learning has great potential to detect COVID-19 as well, which has not been explored yet.

5.1.1 Objectives

The objectives of this project are as follows:

- To implement Few-shot Learning in chest x-ray image for classification to detect COVID-19.
- To explore the ability of "Few-shot Learning" compare to "Deep Learning" in chest x-ray image classification.
- To find out the performance of different methods on chest x-ray images with some few-shot (1-shot, 5-shot etc.) settings.

5.2 Related Works

In computer assisted diagnosis, Deep Learning serves as a popular framework for diagnosis of diseases. So, researchers tend to diagnose COVID-19 with Deep learning from radiography images. We find two types of radiography images of COVID-19, i.g. X-ray and CT scan of lungs, publicly available online. Some classic architectures were used to diagnose COVID-19. And also some literature used their own customized Convolutional Neural Network (CNN) models/architectures. Both classic and customized CNN models achieved good results.

In [79], authors used four CNN models i.e. DenseNet-201, ResNet-18, AlexNet, SqueezNet to do multi-class classification (COVID-19, Normal, Viral Pneumonia). Chest X-ray of 213 COVID-19 as well as 1341 Normal and 1345 Viral Pneumonia were used to form their own customized dataset[80] which is available in Kaggle. With data augmentation and transfer learning, they got comparable results getting 98.3% accuracy with SqueezNet. In [81], authors used ResNet-50 naming their model COVID-ResNet. They got 100% Precision and Recall/Sensitivity for COVID-19 classification. They argued their accuracy (96.23%) to be higher than COVID-Net (93.3%) [82] on same dataset (COVIDx [83]). In [84], [85], [86], authors used VGG-19, MobileNet V-2, NasNetLarge respectively. It is worth mentioning that their model gained comparable

results. Specially, in [85], authors trained their MobileNet V-2 end to end from scratch and achieved accuracy of 99.18% in binary classification (COVID-19, Non-COVID-19). In [87], authors proposed a Capsule Networks, called COVID-CAPS rather than a conventional CNN to deal with a smaller dataset. Moreover, COVID-CAPS was reported to achieve an accuracy of 95.7%, sensitivity of 90%, and specificity of 95.8%. In [88] have worked on a very small database of 105 COVID-19, 80 Normal and 11 SARS X-ray images to detect COVID-19 Xray images using modified pre-trained CNN model named DeTraC (Decompose, Transfer and Compose) to project the high dimension feature space into a lower one. This would help to produce more homogeneous classes, lessen the memory requirements and achieved accuracy, sensitivity and specificity of 95.12%, 97.91% and 91.87% respectively. On the other hand, in [89], authors introduced a deep CNN, called COVID-Net for the detection of COVID-19 cases from around 14k chest X-ray images, however the achieved accuracy was 93.3%. In [90], author used pre-trained and fine-tuned SqueezeNet network with Bayesian optimization to classify COVID-19 images, which showed promising result on a small dataset. This approach should be evaluated on a large COVID and Non-COVID dataset.

Annotated Radiography chest X-ray images of COVID-19 are rare as COVID-19 is a novel disease. Lack of data may not provide generalization to the model rather the model got overfitted and may not give good prediction. So, Few-shot learning can be a solution here which is discussed in next sections.

5.2.1 Object detection with Few-shot Learning

In the object detection/classification scenario, researcher developed several Few-shot learning algorithms. In [91], authors gave a comparative analysis of some Few-shot learning methods. They evaluated four models i.e. ProtoNet [69], MatchingNet [68], MAML (Model Agnostic Meta Learning) [67], RelationNet [92] with standard 1-shot and 5-shot settings on mini-ImageNet (a subset of ImageNet [93]) and CUB[94]. However, Few-shot learning is categorized in some methods based on approaches. "Learning to Initialize" is one of the approaches where the model aims to choose a set of parameters that can be fine-tuned easily for another task via a few gradient descent steps and limited

number of data. MAML falls under this category. Another approach is "Learning to Compare" based method. Moreover, in the classical deep learning framework, the models learn "how to classify" from training data and evaluate the results using test data. In the metric-learning framework, the model learn "how to learn to classify" given a set of training tasks and evaluate using a set of test tasks. In this context, we use one set of classification task to solve other novel task.

"Initialization based methods" aim to solve few-shot learning task by “Learning to Fine-tune”. This approach focuses to learn fine-tune parameters of the network so that it can learn with a limited number of labeled train examples and a small number of gradient descent updates for novel classes [95], [96]. An example of this approach is MAML [67]. It provides a good initialization of a set of parameters and achieve fast adaption with few number of data. Another approach in this class focuses on "Learning an Optimizer". A good example of this approach is LSTM based meta-learner [70]. The idea is that the optimization steps will be constrained to produce model’s parameter that generalize well when trained with only few example data. Although these "initialization based methods" are capable of achieving fast adaption with a limited number of data, these methods have limitations of handling domain shifting between base and novel classes.

"Distance or metric learning based methods" tend to solve few-shot classification task by “Learning to Compare”. The idea is that if a classifier can find the similarity of two images, it can classify a novel input image with the labeled images [97]. During training, metric learning based methods make their prediction constrained on distance or metric with few labeled data to build a sophisticated comparison model. Some examples of distance metrics include Cosine similarity [68], Euclidean distance [69], CNN-based Relation module [92], Ridge Regression [71], and Graph Neural Network [98]. A representation of these methods are given in Figure ??.

5.2.2 Model Agnostic Meta Learning (MAML)

Meta learning is a process in which a model is made to "learning to learn" [99]. MAML follows the same approach of meta learning. Rather than learning the update function or learning rule, MAML learns the parameters of the model in a fully differentiable way

[67]. The term agnostic is used because it is not made for only one particular case, it can be used for classification, regression and reinforcement learning [67]. In few-shot learning scenario, the model's parameters are updated with support set from training data by few number of gradient descent steps. Support set may have one or very few number of images e.g. five from each class, thus the model can learn in 1-shot or 5-shot settings. Then the query/test set is used to evaluate the model's performance. To prove the model be efficient, authors evaluate MAML with Omniglot [100] and miniImageNet for 1-shot and 5-shot setting. Comparing to their mentioned models, MAML got state-of-the-art accuracy. However, the model is simple, elegant and neat, but in contrast it may need higher order derivatives which make the training time consuming. Again the model has advantage of using various scenarios.

5.2.3 Prototypical Network (ProtoNet)

In [69], authors proposed "Prototypical Networks" (ProtoNet) for few-shot classification task, where the classifier will generalize to new classes that are unseen in the training set, given only a limited number of examples of each novel class. Moreover, ProtoNet learns a distance metric (Euclidean distance) in which classification is performed by computing distances to representative prototypes of each class. The idea is that each class is represented by the mean of its support examples in a representative space learned by the Neural Network (NN). The NN is trained to perform well in the few-shot setting by implementing episodic training, specifically. To evaluate the model's performance, authors used mini-ImageNet and Omniglot dataset with both 1-shot and 5-shot setting. Compared to other approaches for few-shot classification task, ProtoNet represents a simpler inductive bias that proves to be beneficial in the few-shot learning domain, and also achieves excellent results. In [69], authors also analyse showing that a simple design decision can provide substantial improvements over recent methods involving complex architectural choices and meta-learning. The performance of ProtoNet can be improved by carefully choosing the distance metric, and by modifying the episodic learning process. Authors furthur generalized ProtoNet to zero-shot learning and achieve

state-of-the-art results on the CU-Birds dataset. However, the simplicity and efficiency of "Prototypical Networks" makes it a promising method for few-shot classification.

5.2.4 Matching Network (MatchingNet)

In [68], authors employed the idea from metric learning based on deep imaging features and from recent developments of augmenting neural networks with external memories, naming their model Matching Network (MatchingNet). This model learns a network that maps a small labelled support set and an unlabelled query set to its label. It need fine-tuning to adapt to novel class. Cosine similarity is used as the matching function. Authors evaluated MatchingNet in one-shot learning problems for vision using Omniglot and ImageNet. This method improves one-shot accuracy from 87.6% to 93.2% on ImageNet and from 88.0% to 93.8% on Omniglot compared to other competing approaches. Some key insights is noticed in this model. Firstly, one-shot learning proves to be easier when the network is trained to do one-shot classification. Secondly, in the NN a non-parametric structure makes it easier to remember and adapt to new example sets in the same tasks. A drawback of this model is also encountered that as the support set grows big in size, the computation for each gradient update becomes more costly.

5.2.5 Relation Network (RelationNet)

Recently a conceptually simple, efficient, and general framework is proposed called Relation Network (RelationNet) for few-shot learning[92]. In RelationNet, the classifier will learn to recognise novel classes by giving only few samples from each class. This model is trained end-to-end from scratch. During training, it learns to learn a Convolutional based deep distance metric to do comparison among a small number of training images within episodes in which each episode is designed to mimic the few-shot setting. When the RelationNet is trained it is able to do classification on query images of new classes by computing relation scores without further fine-tuning the network. This framework can be extended to zero-shot learning. Extensive experiments have been done on five benchmarks. This model achieved a good result for 1-shot and 5-shot

setting for both mini-ImageNet and Omniglot dataset. Authors claimed this model to be a unified and effective approach for few-shot as well as zero-shot learning.

5.2.6 Siamese Neural Network(SiameseNet)

Siamese convolutional neural network [97] clone the same neural network architecture and learn a distance metric on top of these representations. This model is incredibly powerful for FSL, specially at one shot classification. Figure ?? represents siamese network in which there are two exact neural network. Both network pass 2 images and they learn a distance function between the two vector representations produced by the same exact neural network. So, when they add a convolutional neural network on it, the process takes an image through a series of convolutions until they eventually reach a feature vector. And then they are going to use a metric to compare the feature vectors between different images. They train the model on pairs of same images and different images. In omniglot dataset, this model does a pretty good job being closer to the human lavel accuracy. But again it is only 105×105 gray scale images. So there is not that much complexity to the image dataset.

Table 5.3: Overview of Some COVID-19 Detection Approaches Using Few-shot Learning.

Study	Proposed Model	Training Method	Image Type	Class	Dataset	FSL Setting	Performance
Aradhay et al.	GRN + PRN	Supervised	X-Ray	COVID-19, Bac. Pneumonia, Vir. Pneumonia Normal	[101, 102]	1 Shot (2, 3, 4) Class	ACC: C2:100% C3:85.23% C2:74.05%
Arifin et al.	SSD MobileNet V1 SSD MobileNet V2	Supervised	X-Ray	COVID-19, Vir. Pneumonia, Normal	[80, 101, 102, 103]	1 Shot 3 Class	ACC: SSD MobileNet V1:92.48% SSD MobileNet V2:94.00%

Jadon et al.	Siamese Nete-orks (TL)	Supervised	X-ray	COVID-19, Vir. Pneumo-nia, Normal	[80, 104]	1 Shot 3 Class	ACC:96.4% PRE:0.965 REC:0.962 F1:0.959
Miao et al.	UMLF-COVID	Unsupervised	X-Ray	COVID-19, Normal, Vir. Pneumo-nia Bac. Pneumo-nia Other Pneu-monia	[101]	(1, 5, 10) Shot (3, 4) Class	ACC: C3/S10:0.93 C4/S10:0.90
Szucs et al.	DVMN		X-Ray	15 Diseases	[101, 104]	(1, 2, 5) Shot (2, 4) Class	Avg. ACC:0.8544 Avg. F1:0.7710

Voulodimos et al.	Few-shot Driven U-Net	Supervised	CT	COVID-19 Positive, COVID-19 Negative	[101]		AUC: 0.968
Fayjie et al.	FSS for MIS	Supervised	CT	COVID-19, Non-COVID-19	[105]	5 Shot 2 Class	DSC:89.74%
Basset et al.	FSS-2019-nCOV	Semi-supervised	CT	COVID-19, Infected, Lesion-free	[103, 106, 107, 108]		DSC: CT-1:0.798 CT-2:0.632
Chen et al.	Self-Supervised COVID-19 Diagnosis Method	Self-supervised	CT	COVID-19, Non-COVID-19	[82]	(1, 2, 3, 4, 5, 6) Shot 2 Class	C2/S6: ACC:0.855 AUC:0.945

Zheng et al.	Unsupervised Meta Learning with Self-Knowledge Distillation	Unsupervised	CT, X-ray	COVID-19, SARS MERS, H1N1, Bac. Pneumonia	[80]	1 Shot 5 Class	ACC: CT:0.986 X-ray:0.995

Table 5.4: Overview of Some COVID-19 Detection Approaches Using Few-shot Learning.

Dataset Name	Image Type	COVID-19 Images	Other Images	Used By
COVID-19 Image Data Collection	X-Ray	468	Normal: 79, Bac. Pneumonia: 79, Vir. Pneumonia: 79	
COVID-19 Radiography Database	X-ray	219	Normal: 1341, Vir. Pneumonia: 1345	
COVID-19 CT segmentation dataset	CT	CT-1: 110 (60 patients), CT-2: 373	CT-2: 353 infected, 285 lesion-free	
COVID-CT	CT	349 (216 patients)	463 non-COVID-19	
Chest XRayAI	X-Ray		612	
BIMCV-COVID19+	X-Ray, CT			
Chest X-Ray Images (Pneumonia)	X-Ray		Normal: 51,490, CNV: 37,256, DME: 11,599, Drusen: 8,768	
COVID-19 CT Lung and Infection Segmentation Dataset	CT	70	Lung Cancer: 113, PE: 402	

5.3 Methodology

5.3.1 Problem Setup

We solve our problem in few-shot learning scenario. So, we will split our dataset in three parts: a train set, a support set, and a test set. In few-shot settings, the support set and the test set must share the same label space. In contrast, the train set has its own label space which is different from support/test set [92]. If the support set contains K labelled examples for each of C unique classes, the few-shot problem is called C-way K-shot. The classifier is trained with the support set to assign a class label $\hat{y}$ to each example image $\hat{x}$. During training, meta-learning approach is followed to extract the transferable the knowledge from the train set and apply these knowledge to support set and the test set. An episode-based training method is followed where each episode mimic the few-shot setting, as proposed in [68]. During each training, an episode is formed by randomly selecting C classes with K labelled samples from the train set. This is the sample set $S = (x_i, y_i)_{i=1}^{m} (m = K \times C)$ [92], and the fraction of the remainder of those C classes' samples will use as the query set $Q = (x_j, y_j)_{j=1}^{n}$ [92]. This sample/query set is designed to mimic the support/test set. Further, the classifier can be fine-tuned using the support set.

5.3.2 Few-shot Model

5.3.2.1 Learning by Distance metric

The first step is to train the model with train data split which can be considered as base classes. To extract the features, we use various backbones. Once the backbone is trained, then we validate the model with validation split which are novel classes. Then we can test it with test split which are also novel classes. Few-shot model (see Figure 5.1) has two module: embedding module f_φ and distance metric module d_ϕ. For one-shot setting, from each C class image $x_i \in R^D$ from sample set S and image $x_j \in R^D$ from query set Q are fed through embedding module f_φ (f being the embedding module function

and φ being the trained architecture parameters). For K-shot setting ($K > 1$), feature maps from K images are summed over element-wise to form a single class feature map of M-dimensional with $c_k \in R^M$. Backbone produce feature maps $f_\varphi(x_i) : R^D \rightarrow R^M$ and $f_\varphi(x_j) : R^D \rightarrow R^M$. These class feature maps $f_\varphi(x_i)$ and $f_\varphi(x_j)$ are concatenated with operator $C(f_\varphi(x_i), f_\varphi(x_j))$ (C being concatenation operation function). Then the concatenated features are fed into the distance metric module d_ϕ which produces scalar results for each class, referring to distance scores d that represents the similarity between support inputs and query inputs. For example, in C-way K-shot setting the distance score d is as follows [92]:

$$d = d_\phi(C(f_\varphi(x_i), f_\varphi(x_j))); i = 1, 2, 3, ...C \tag{5.1}$$

The model gives distance scores as output which is a continues value $d : R^M \times R^M \rightarrow [0, +\infty)$. In the case of Relation Network[92], authors intend to use mean square error (MSE) loss as objective function to train the model. MSE loss function minimizes loss by regressing the relation score d to the ground truth where matched pairs have similarity 1 and the mismatched pair have similarity 0. The objective function can be expressed as follows:

$$\varphi \leftarrow_\varphi \sum_{i=1}^{m} \sum_{j=1}^{n} (d - 1 \times (y_i == y_j))^2 \tag{5.2}$$

For the other cases, given the distances $d : R^M \times R^M \rightarrow [0, +\infty)$, model produces a distribution over classes for a query point x based on a softmax over distances in the embedding space. Learning procedure advances by minimizing the negative log-probability $J(\phi) = -\log(p_\phi(y = k|x))$ of the true class k via ADAM optimizer. Whereas,

$$p_\phi(y = k|x) = \frac{\exp(-d(f_\phi(x_i, c_k))}{\sum_k \exp(-d(f_\phi(x_i, c_k))} \tag{5.3}$$

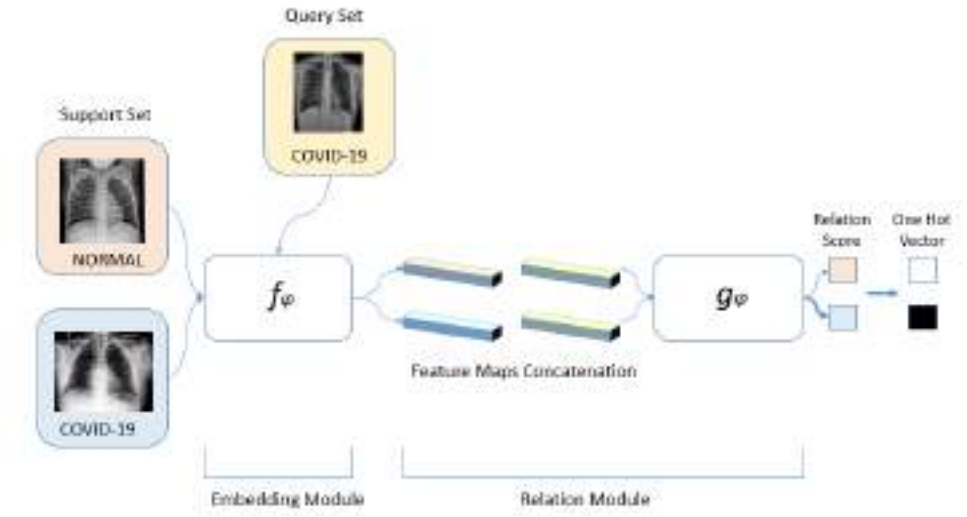


Figure 5.1: Relation Network architecture for a 2-way 1-shot problem with one query image.

5.3.2.2 Initialization by parameters

The Model Agnostic Meta Learning [67] is an initialization based meta-learning algorithm. In this work, each support set is used to learn the initial model parameters using only a few gradient updates. The learnt model is conditioned on the support set, as different support sets have different gradient updates. It is to be noted that when the query set samples are predicted by the learnt model in the meta-training stage, the loss of the query set is used to update the parameters of the initial model, not the learnt one. As mentioned above, we consider a model represented by a paramterized function f_φ with parameters φ. When learning for a new task $\mathcal{T}_i$, the parameters of the model φ become φ'. In this method, the updated parameter vector φ_i is calculated using one gradient descent updates on task $\mathcal{T}_i$. For instance, when using one gradient update, following equation is used.

$$\varphi_i' = \varphi - \alpha \nabla_\varphi \mathcal{L}_{\mathcal{T}_i}(f_\varphi) \tag{5.4}$$

The step size α is fixed as a hyperparameter. The parameters of the model are optimized by the performance of f_φ with respect to φ across tasks sampled from $p(\mathcal{T})$. The

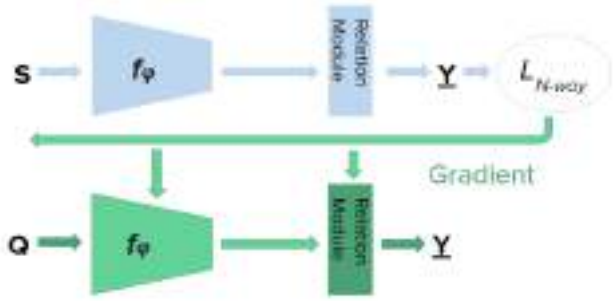


Figure 5.2: Relation Network + MAML architecture for few-shot classification problem with support images (S) and query image (Q).

objective function is as follows:

$$\min_{\varphi} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi - \alpha \nabla_\varphi \mathcal{L}_{\mathcal{T}_i}(f_\varphi)}) \tag{5.5}$$

It is to be noted that the optimization is calculated over the parameters φ, however the objective function is computed using the updated model parameters φ'. This method aims to optimize the parameters of the model such that one gradient steps on a new task $\mathcal{T}_\rangle$ will produce effective performance on that task. The optimization across tasks is performed via Adaptive Moment Estimation (ADAM) as mentioned earlier, the parameters φ are updated as follows:(Here β is the meta step.)

$$\varphi_i' = \varphi - \beta \nabla_\varphi \mathcal{L}_{\mathcal{T}_i} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\varphi'}) \tag{5.6}$$

Combining Relation Network and MAML In this work, we intend to combine Relation Network[92] and Model Agnostic Meta-learning (MAML)[67]. So, feature maps are extracted from images by feature encoder for both support set and query set. Then relation scores are calculated by equation 5.1. MSE loss is calculated. During optimization the parameters are updated with equation 5.6. The objective function for this method is equation 5.5.

Distance Correlation Metric Distance correlation is an extent of strength between non-linear random variables. The range of Distance correlation is from 0 to 1, where 0 infers the non-relation behavior between variables X & Y and 1 infers that there

is relation between X & Y. Distance correlation does not mean the point to point correlation between the distances themselves, in contrast it is a correlation between the scalar products that the double centered matrices are made of. Let take two random variables X & Y, where $(X_k, Y_k), k = 1, 2, ..., n$. At first, a n by n distance matrices is computed $(a_{j,k)}$ and $(b_{j,k})$ (equations 5.7 & 5.8) that contains all pairwise distances. Then double centered distances are calculated (equations 5.9 & 5.10). Since any sort of covariance is the cross-product of moments and distances are not moments, thus it is needed to compute them into moments. To do this, first the deviations from the mean is calculated. Lastly, the arithmetic average of the products is computed A and B to get the squared distance variance (equation 5.11). The distance variance simply represents the distance covariance of two variables which are identical and it is the square root of the following variable:

$$a_{j,k} = \|X_j - Y_k\|, j, k = 1, 2, 3, ..., n \tag{5.7}$$

$$b_{j,k} = \|Y_j - Y_k\|, j, k = 1, 2, 3, ..., n \tag{5.8}$$

$$A_{j,k} = a_{j,k} - a_{j,.} - a_{.,k} + a_{.,.} \tag{5.9}$$

$$Bj, k = b_{j,k} - b_{j,.} - b_{.,k} + b_{.,.} \tag{5.10}$$

$$dCov_n^2(X, Y) = \frac{1}{n^2} \sum_{j=1}^{n} \sum_{k=1}^{m} A_{j,k} B_{j,k} \tag{5.11}$$

$$dVar_n^2(X, Y) = dCov_n^2(X, Y) \tag{5.12}$$

Formally, the formula for distance correlation can be written:

$$dCorr = \frac{dCov(X, Y)}{\sqrt{dVar(X) dVar(Y)}} \tag{5.13}$$

Mahalanobis Distance Metric Mahalanobis distance is a powerful multivariate distance metric which can calculate the distance between a point or vector and a sample distribution. It has many applications in classification on imbalanced datasets, one-class

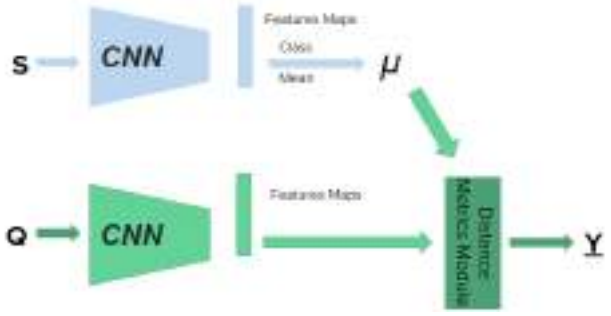


Figure 5.3: CNN architecture with distance metric function module for few-shot classification problem with support images (S) and query image (Q). Distance metrics used for our proposed methods are Distance Correlation and Mahalanobis Distance

classification, and multivariate anomaly detection and many more. Considering its various useful applications, this metric is seldom discussed in Machine Learning. So, in this context, mahalanobis distance metric is introduced and discovered its performance in few-shot learning.

The Mahalanobis distance between two sample variables (vectors) x and y with respect to the training data x_i is computed by equation 5.14. In this equation, mean vector μ and covariance matrics S from the training samples $x_i; i = 1, 2, 3, ..., n$ are computed by equations 5.15 and 5.16.

$$d = \sqrt{(x - y)S^{-1}(x - y)^T} \tag{5.14}$$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{5.15}$$

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \tag{5.16}$$

Here, the term $(x\check{}y)$ in equation 5.14 is refering to the distance of the vector from each other. Now, consider y as mean vector of a sample distribution, it clearly indicates the distance from the mean vector. Then multiplying by the inverse covariance matrix means multivariate equivalent of the regular standardization process namely, $z = \frac{(x\check{}\mu)}{\sigma}$, which is, $z = \frac{sample\check{}mean}{variance}$. In addition, the usefulness of dividing by the covariance is significant. When the variables are strongly correlated, the covariance is high. Thus dividing by a high covariance significantly normalize the distance. However, when the variables are not correlated, the covariance is small and the distance is not reduced much. Consecutively, it manages both the scaling and the relation of the two variables.

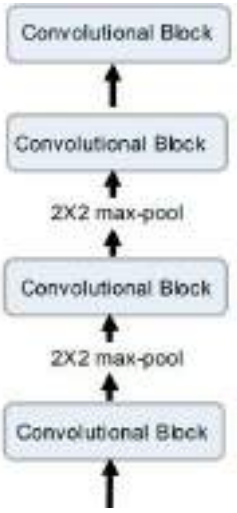


Figure 5.4: Convolution Neural network which has 4 layer of convolution block followed by max pooling operation on first two layers.

Figure 5.5: Convolution Block, each consists of convolution operation within 3×3 window, batch normalization, non-linearity function ReLU.

5.3.2.3 Network Architecture

Feature extraction of images is a crucial part. The performance of the model is largely dependent on how the features are being extracted. Convolution Neural Network (CNN) has been proven to be the best for feature extraction. To this task we use 4 layer CNN and Wide Residual Network, which have been used by authors of the few-shot learning classification [92], [67], [91], [68], [69]. This architecture has been introduced below.

Conv-4 Conv-4 has four convolutional blocks (see Figure 5.4). Each convolution block has a 64 unit 3×3 convolution layer, a batch normalisation layer and a ReLU non-linearity function. In embedding module, first two blocks also contain a 2×2 max-pooling layer but the later two convolution blocks have no max-pooling layer.

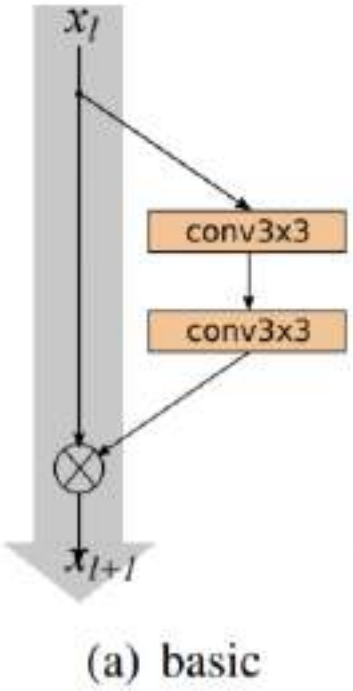


Figure 5.6: Basic residual block used in wide residual network.

Wide Residual Networks (WRN)[1] It is an advance version of Residual Networks (ResNet) [109]. Instead of becoming deeper, this time the architecture gets wider for WRN. It becomes effective because of two reasons: (1) The number of layers have been reduced and (2) Training time is shorter now. Conventionally, WRN-d-k indicate that WRN has the depth of d with widening factor k. In WRN[1] paper, author use B(M) to denote residual block structures, here M is a list with the window sizes (square spatial kernels) of the convolution layers in a block. For instance, B(3,3) refers to a residual block with 3×3 and 1×1 convolution layers. For our specific classification problem, we use WRN-50-2 which outperforms ResNet-152[109] in ImageNet and the training time is significantly faster because of having 3 times fewer layers.

5.3.2.4 Data set

We address our problem as classification in few shot learning scenario which is a meta learning task. So, a dataset is necessary to do our experiment. There are many datasets available which follows the standard train-validation-test split for supervised learning

group name	output size	block type = $B(3,3)$
conv1	32×32	$[3\times3, 16]$
conv2	32×32	$\begin{bmatrix} 3\times3, 16\times k \\ 3\times3, 16\times k \end{bmatrix} \times N$
conv3	16×16	$\begin{bmatrix} 3\times3, 32\times k \\ 3\times3, 32\times k \end{bmatrix} \times N$
conv4	8×8	$\begin{bmatrix} 3\times3, 64\times k \\ 3\times3, 64\times k \end{bmatrix} \times N$
avg-pool	1×1	$[8 \times 8]$

Figure 5.7: Structure of WRN. Convolutions groups are shown in brackets where N denotes the number of blocks in a group, down sampling is performed by first layers of groups conv3 and conv4. In this specific example shown, the network uses a basic block B(3,3). Final classifier group is omitted for clearance.

task. But none of them follows the train-validation-test split for meta learning task. So, we have to make a dataset of our own as proposed in [68]. For this purpose, we have used CheXpert[110] and COVID-19 dataset[80]. CheXpert[110] data set is huge and consists of 14 classes with 224,316 images of 65,240 patients. Since we have a limited memory to do our task, we made a sample dataset for our experiment. We have used 4,050 images from five classes - Atelectasis, Cardiomegaly, Consolidation, Edema, and Pleural Effusion. On the other hand, we have used 2,828 images of classes, Normal, Pneumonia, and COVID-19, from COVID-19 dataset[80]. The number of images for each class is given in Table 6.1. The train-validation-test split used in our experiment is given in Figure 5.8.

It is noticeable that Atelectasis, Cardiomegaly, Consolidation and Pneumonia are in train split. These disease have already occurred in the past. And those diseases have high clinical importance and relevance in practical cases. In Chexpert[110], there are plenty of chest x-ray images available for Atelectasis, Cardiomegaly and Consolidation. Pneumonia is a common disease and in COVID-19 Dataset[80] we get enough chest x-ray images to do our task. To validate the models are learning well, a validation set of data is made with Edema and Pleural Effusion which are also taken from Chexpert[110]. After learning from train set and validating by validation set, the models are tested on test set. Our test data consists of COVID and Normal chest x-ray images which are taken

Table 5.5: Dataset Distribution of Different Classes

Data Taken	Class Names	Number of Images
CheXpert[110]	Atelectasis	810
	Cardiomegaly	810
	Consolidation	810
	Edema	810
	PleuralEffusion	810
COVID-19 Dataset[80]	COVID-19	614
	Normal	1127
	Pneumonia	1087

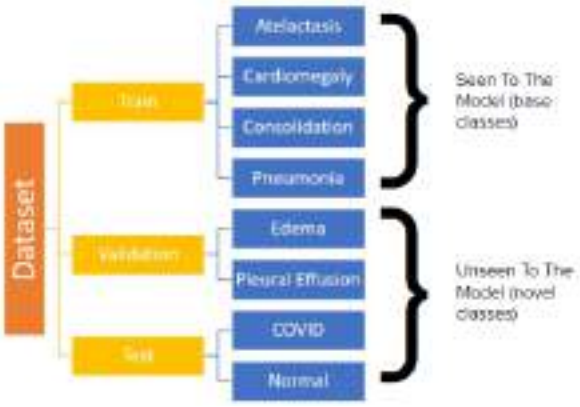


Figure 5.8: Dataset Split For Meta Learning task. Setting is done specifically for our task. COVID-19 is our main concern, So, we put COVID-19 in test set.

from COVID-19 Dataset[80]. The reason behind using COVID in test set is that it is fairly a new disease. Our main goal is to find out the distinguishable imaging features of COVID-19 from the model which is trained on the disease we already have. To fulfill the need of having 2 classes in test set we add Normal from COVID-19 Dataset[80]. We can randomly shuffle the classes to split the dataset differently[91]. But for our specific task, we prefer COVID-19 to be in test split.

5.3.2.5 Implementation Details

We do our implementation on models by following these tasks 1) Meta-training with the train and validation set 2) Meta-test with the test set to see the performance to classify COVID-19.

Figure 5.9: Example images from train split classes. Atelectasis (left), Cardiomegaly (left-middle), Consolidation (right-middle), and Pneumonia (right). Bounding boxes indicate where the lesion are.

Figure 5.10: Example images from validation split classes. Edema (left) and Pleural Effusion (left-middle). Bounding boxes indicate where the lesion are.

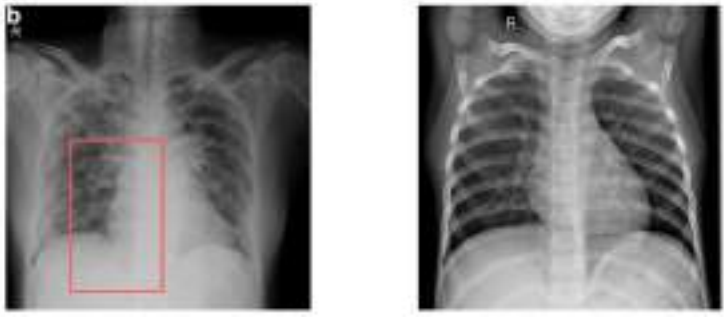


Figure 5.11: Example images from test split classes. COVID-19 (left) and Normal (left-middle). Bounding boxes indicate where the lesion are.

In the meta-training stage, we train 1,000 episodes for every few shot tasks. In each episode, we sample 2 classes classification for N-shot (N is 1, 5, 10, 15, 20, 25, 30 in both meta-training and meta-testing stages). Each episode consists of batch size 2*(N+10). For each class, we pick N labeled instances as our support set and 10 instances for the query set in a N-shot task. In the fine-tuning or meta-testing stage for all methods, we average the results over 100 experiments. All methods are trained from scratch and use the Adam optimizer with initial learning rate 10^{-3}. A scheduler is applied with gamma

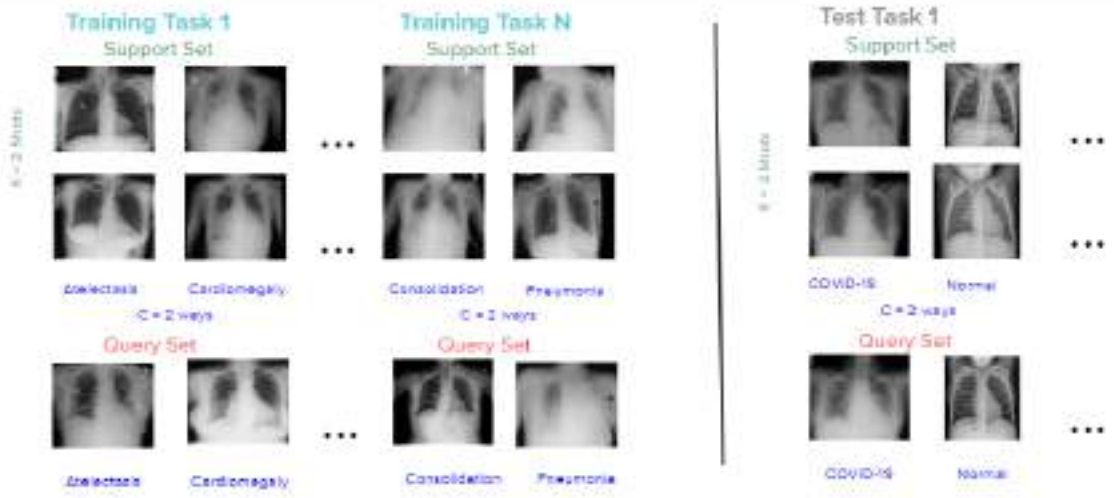


Figure 5.12: Support set and Query set for each training and testing task. In this simple diagram, 2-way 2-shots setting is used.

equal to 0.1 and step size 100, means that at every 100 step learning rate will be reduced by power of 0.1. In pre-processing step, histogram equalization is applied to enhance the contrast of the images. We resize images to 224*224 to feed into feature encoder and apply normalization and cropping in both the meta-training or meta-testing stage. We get distance scores from each network. Distance scores resembles the distance of query images from each class distribution. The higher the score, the more an image related to the class. We use the validation set to select the training episodes with the best accuracy. At last, the distance score is one hot encoded to finalize the result. At every 50 episode, the model is validated with the validation set. Validation is done for 10 times/episodes. Model is saved when validation gives higher accuracy. Models include MatchingNet, PrototypicalNet, RelationNet, MAML, RelationNet+MAML, and different Convolution Neural Network (CNN) with distance correlation metrics and mahalanobis distance metrics.

- 2-way K-shot (K = (1, 5) in general for all methods, (10, 15, 20, 25, 30) in Relation Network only), means that from 2 classes K number of images will be selected randomly, which forms support set 5.12
- Episodic Training is conducted as mentioned by Chen et al. [91].
- Each episode mimics 2 way K shot.

Hyper Parameters	*Value*
Episode	1000
Validation episode	10
Learning rate	0.001
Scheduler	gamma = 0.1 (step size = 100)

Figure 5.13: Hyper parameters used in this few-shot classification problem. These parameters are used by Chen et al. [91] in various methods described here.

- Each episode has Support set and Query set of 2-way K-shots.
- Support set has 2*K images.
- Query set has 2*10 images.
- Batch size for each task will be 2*(K+10).

We conduct this experiments on some common Methods indicated in earlier paragraph. The most common settings in few-shot classification are 1-shot and 5-shot classification, means that 1 or 5 labeled samples are available from each class. We mention earlier that a four-layer convolution backbone (Conv-4) and Wide Residual Network (WRN) [109] will be used with an input size of 224×224 as in Chen et al. (2017) [91] and perform 2 way classification for only novel (Unseen) classes during meta-testing stage.

5.4 Experimental Results

In this section, the performance of the model is discussed. From the experiment, we get training accuracies, losses and validation accuracies per episode. The trend of accuracy and loss indicate the performance of the model. Also t-SNE for every shot is plotted to visualize the discriminant features in 2D plane. GradCAM visualization is also showed for correctly classified and mis-classified cases with explanation.

5.4.0.1 Accuracy and Loss

We compare our results with two performance measures: (1) Accuracy and (2) Loss over the episodes to validate the correctness of our implementation on dataset in Table 6.1. Accuracy is simply the measurement of correctly classified samples over total number of samples. Loss is the measurement of penalty for bad predictions. High accuracy means there is low loss, thus the models prediction is good. Conventionally, accuracy is measured in percentage.

During meta-training, every episode is recorded. In figure 5.14. 5.15, accuracy is plotted in every 100 episode starting from 1. From the graph, we observe that CNN with correlation distance learns relatively well. Compare to other models, baseline learns less. The methods get their stable accuracy in the range from 200 to 300 episode, giving a consistent magnitude till 1000 episode. After every 100 episode, models are set in evaluation mode and checked the accuracy with validation dataset which is unseen by the models. From figure 5.16 and 5.17, validation accuracy also follows the same trend as train accuracy, where CNN with correlation outperforms all the models. Finally, in table ??, average test accuracy over 100 episodes on test split is reported. From that table, CNN with correlation has the highest accuracy. CNN with correlation outperform the other models for its robustness and sophisticated design. Another approach, namely CNN with mahalanobis distance, achieves comparable accuracy on test dataset. RN+MAML also acquires good accuracy. Other methods achieves nearly the same numbers as reported by Chen et al. [91].

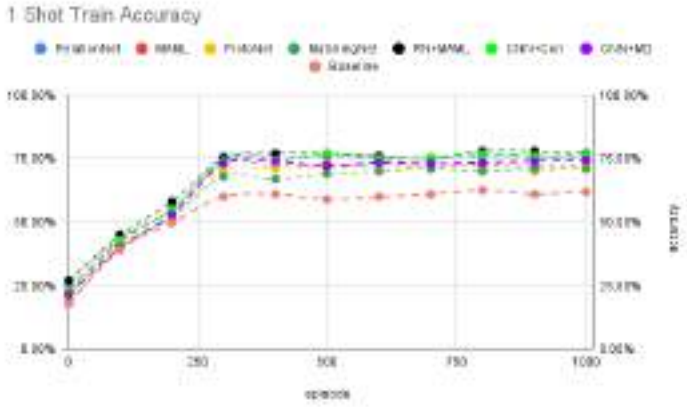


Figure 5.14: Training Accuracy over 1000 episodes. The model tends to get stable slowly after 200 episode.

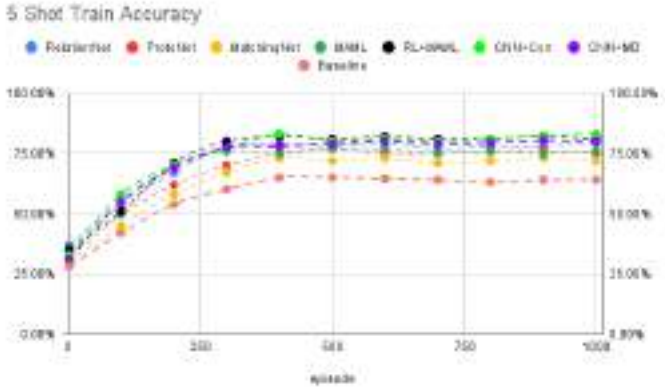


Figure 5.15: Training Accuracy over 1000 episodes. The model tends to get stable slowly after 200 episode.

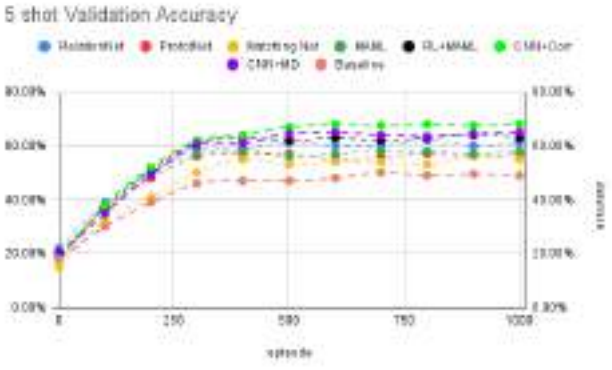


Figure 5.17: Validation Accuracy over 1000 episodes. The model tends to get stable slowly after 200 episode.

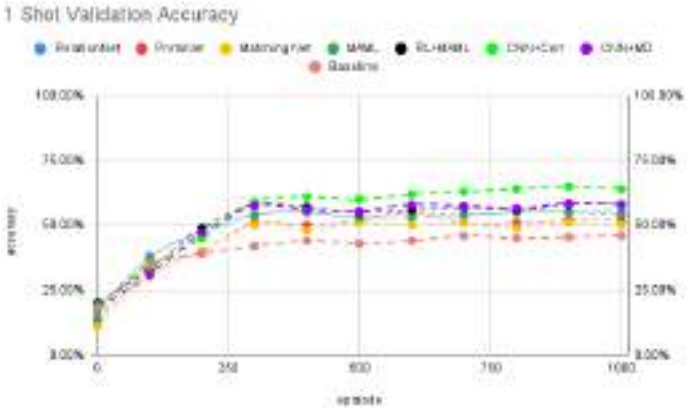


Figure 5.16: Validation Accuracy over 1000 episodes. The model tends to get stable slowly after 200 episode.

Methods	Backbone	Classifier	1 shot accuracy	5 shot accuracy
Baseline	Conv-4	Linear	46.00%	49.10%
MatchingNet(2016)	Wide Residual Network_50_2	Cosine Distance	51.20%	55.12%
ProtoNet(2017)	Wide Residual Network_50_2	Euclidean Distance	52.38%	57.48%
MAML(2017)	Conv-4	Linear Classifier	54.10%	58.61%
RelationNet(2017)	Wide Residual Network_50_2	Relation Module	55.24%	61.45%
RN+MAML (ours)	Conv-4	Relation Module	57.67%	63.34%
CNN+Mahalanobis (ours)	Wide Residual Network_50_2	Mahalanobis Distance	59%	66%
CNN+Correlation (ours)	Wide Residual Network_50_2	Correlation Distance	**61.00%**	**68.00%**

Figure 5.18: Average Test Accuracy over 100 episodes.

5.4.0.2 Ways and Shots vs Accuracy

It has been theorise that increasing number of shots can increase the performance of the model. The reason behind this, increasing number of shots means model is seeing more general distribution of the dataset. So, it becomes easier for the model to predict the class for query images. In figure 5.19, it has been demonstrated that accuracy increases as shots increases. For this purpose, CNN with correlation distance has been used which is the outperformed model in table 5.18. In contrast, if the number of ways (classes) increases accuracy will decrease accordingly. It is reported in figure 5.20. The reason

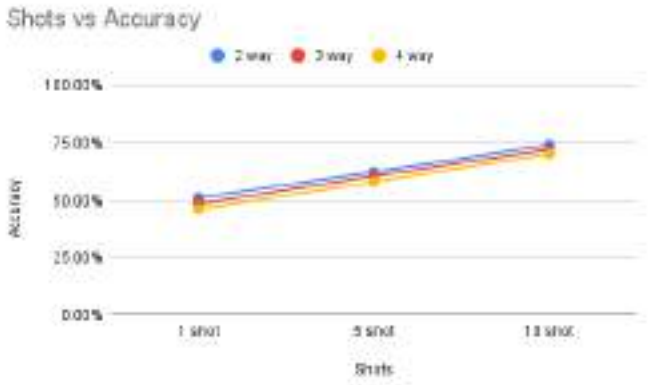


Figure 5.19: Shots vs Accuracy. Increasing shots increases the number of images seen by the model which increases accuracy.

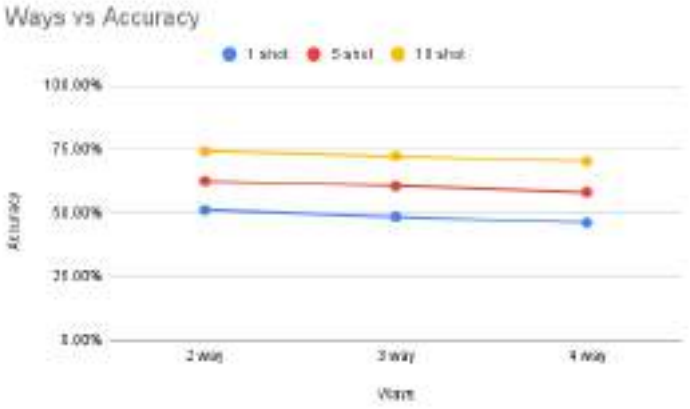


Figure 5.20: Ways vs Accuracy. Increasing ways increases the number of classes seen by the model which confuses the model affecting the predictions, thus decreases accuracy.

is that increasing the ways add more classes in distribution, so model get confused. Prediction is affected because of this. Thus the model gives low accuracy.

5.4.0.3 Performance of Relation Network for increasing number of shots

In meta-training stage, for every shot, RelationNet tends to learn slowly. After 900 episodes the model seems to have a stable accuracy. It is trained up to 1,000 episodes. In the official RelationNet[92] implementation, 500,000 training episodes have been performed with 600 validation episodes. As we have limited resources (we are using Google Colaboratory with limited access to GPU), we have to select the number of episodes manually. Moreover, the model has randomness in accuracy. So we get

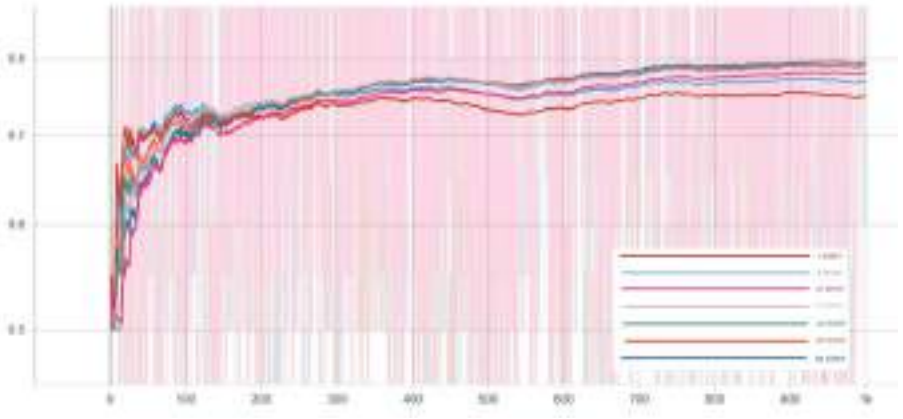


Figure 5.21: Training Accuracy over 1000 episodes. The model tends to get stable slowly after 900 episode.

Figure 5.22: Training loss over 1000 episodes. The model tends to get stable slowly after 900 episode.

a fluctuating accuracy curve for every shot. The accuracy curve is smoothed with exponential moving average to visualize the trend. It gives more weight to the previously smoothed value so that we get a smooth curve. As we increase the shots, the model gives better accuracy trend. At 30 shot, we get the highest accuracy trend. and lowest training loss. At every 50 step, validation step is done to see the performance of the model. The validation accuracy curve also has randomness(In Fig 5.23). From the figure 5.23, it is clear that the model does not perform well on validation set. Clearly train accuracy does not match the validation accuracy. This means that the model overfits. However, the model should give a consistent validation accuracy over step. But we get randomness of accuracy on every step. All in all relation net does not perform well on validation. So we do tested it for 30 shots. As expected the result is close to the validation.

Several reasons have been encountered for the bad performance of the model. First

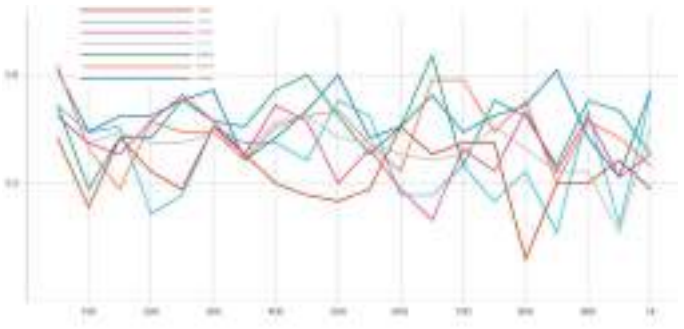


Figure 5.23: Validation Accuracy over 1000 episodes. The model is validated in every 50 episodes.

of all, the chest x-ray images are not pre-processed before applying the model. Many images have very low contrast. So, the infectious regions can not be detected correctly by the model. For this reason, the extracted features have been affected. Many images of different classes has the same imaging property. So, the model gets confused. Secondly, Relation net uses only four convolution blocks. And it has linear cnn architecture. It may not be enough to extract features for chest x-ray. Thirdly, we use a sample dataset which has a limited number of data. So the model does not learn enough to distinguish between different classes. Further, t distributed Stochastic Neighborhood Embedding t-SNE analysis will give some insights in later section 5.4.1.

5.4.1 t-SNE with PCA

t distributed Stochastic Neighborhood Embedding (t-SNE)[111] is a popular dimensionality reduction technique. It maps the high dimensional features into a lower dimension to visualize the characteristic of the data. The goal is to take features of a high-dimensional space and find a reasonable representation of those features in a lower-dimensional space, typically the 2D plane. The algorithm is non-linear and adapts to the underlying data, performing different transformations on different regions. Similarly, Principal Component Analysis (PCA) is also a dimensionality reduction technique. In [111], author recommended to use PCA before t-SNE to reduce the computation time when the number of features is very high.

As in our case, the embedding module produces features. We map the features from the embedding module to a 2D dimensional plane with t-SNE. For each N-shot (N = 1, 5),

Figure 5.24: t-SNE plot on seen classes. (0) Atelectasis, (1) Cardiomegaly, (2) Consolidation and (3) Pneumonia

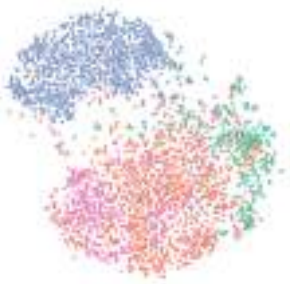


Figure 5.25: t-SNE plot on seen classes. (0) COVID-19, (1) Edema, (2) Normal and (3) Pleural Effusion

CNN (with correlation distance) produces 2048 features as it takes image size 224*224 with bath size 16. The feature size for each train and validation dataset is (M, 2048), M = the total number of images in dataset. By PCA with 50 number of components, the dimension reduces to (M, 50) features. Then t-SNE is performed with 2 number of components and perplexity of 30 to visualize features. This gives us (M, 2) dimensional features which is mapped in 2D plane. The t-SNE plot for both seen and unseen classes in 5 shot is represented in Fig 5.24 and 5.25 respectively.

From the t-SNE plots, it is clear that CNN with Correlation Distance has learnt to distinguish the features for 5 shot. Pneumonia (3) class is distinguished very clearly. However, the features of Atelectasis (0), Cardiomegaly (1) and Consolidation (2) are almost distinguishable in t-SNE plots. In the intersection of these three disease some features are still undifferentiable by the model. Thus the model does not give good accuracy. However, for validation dataset, the model is getting unseen images from unseen/novel classes. As shown in t-SNE plots (Fig 5.25, the model discriminate the features of Normal class (2) from other classes, like COVID-19 (0) Edema (1) and Pleural Effusion(3). But Edema (1) and Pleural Effusion (3) are not distinguishable, which

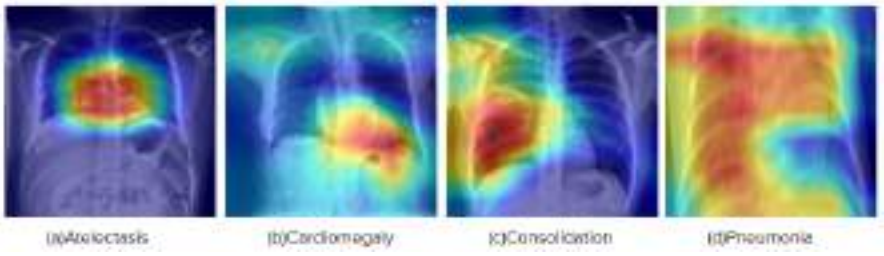
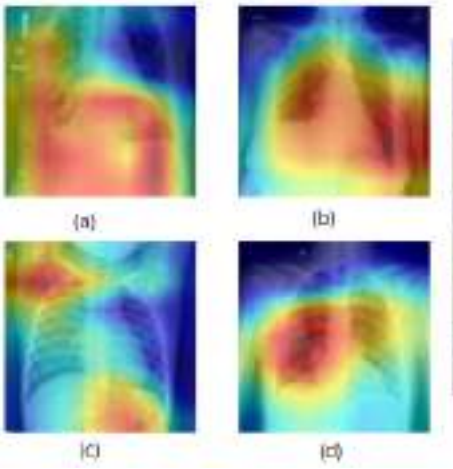


Figure 5.26: Grad CAM visualization of seen classes, correctly classified.



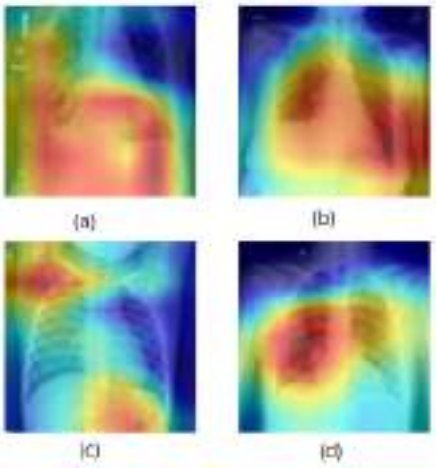
Label	Classified as	Actual Class
(a)	COVID-19	Edema
(b)	COVID-19	Pleural Effusion
(c)	Edema	Normal
(d)	Pleural Effusion	COVID-19

Figure 5.27: Grad CAM visualization of unseen classes, correctly classified.

gives bad prediction for those classes hampering the accuracy of the model.

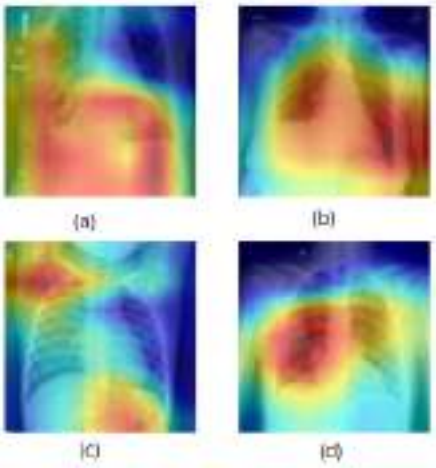
5.4.2 Chest X-ray Visualization with Grad CAM [2]

Gradient-weighted Class Activation Mapping (Grad-CAM) [2] is a useful visualization tool. It uses the gradients of any target class that flows into the final convolution layer to generate a coarse localization map highlighting the important parts in the image for predicting the class. So, we can use it to evaluate our model. As CNN with correlation distance gets higher accuracy, we evaluate this model to visualize some correctly classified 5.26, 5.27 and miss classified cases 5.28, 5.29.



Label	Classified as	Actual Class
(a)	COVID-19	Edema
(b)	COVID-19	Pleural Effusion
(c)	Edema	Normal
(d)	Pleural Effusion	COVID-19

Figure 5.28: Grad CAM visualization of seen classes, miss classified.



Label	Classified as	Actual Class
(a)	COVID-19	Edema
(b)	COVID-19	Pleural Effusion
(c)	Edema	Normal
(d)	Pleural Effusion	COVID-19

Figure 5.29: Grad CAM visualization of unseen classes, miss classified.

5.5 Conclusion

In this work, we investigate few-shot learning for classification on our custom chest x-ray. The dataset was formed from various sources which are reliable and annotated by expert radiologist. Some simple and sophisticated networks were re-implemented on this dataset and the performances were reported. Further three new methods have been proposed such as combination of Relation Network and Model Agnostic Meta Learning, CNN with correlation distance and CNN with Mahalanobis distance. Three models gets competitive performance on this dataset. Specifically, CNN+Correlation gets higher accuracy due to its deeper backbone network and discriminative distance metric. To evaluate the performance of the state of the art model, t-SNE and Grad CAM

visualization were conducted. From the visualization, it was clear that CNN+Correlation can distinguish classes from each other effectively. In future, this method can be developed to do more generalized benchmark dataset such as ImagNet, CUB or CIFAR. Due to scientists' and researchers' constant conscientious effort to unbolt new methods for the effectuation of COVID-19 detection with the power of deep learning, the research in this domain has made significant progress in a short time. However, due to the absence of a benchmark dataset with enough data of COVID-19, researchers have reported performances based on different datasets and evaluation metrics, rendering the idea of comparative analysis, far-fetched. So We will present our customized dataset and hopefully our result will demonstrate few shot learning as viable solution for early diagnosis of COVID-19. We hope that this work will facilitate future enthusiasts who will work in this domain. We believe that our efforts will have a positive impact on the fight against COVID-19 during this pandemic situation.

Chapter 6

Physical Exercise Monitoring, Guidance, and Evaluation

[1]

6.1 Method

6.1.1 Problem Formulation

Suppose, ith RGBD video of an arbitrary exercise $\mathbf{e}$ is represented by $\boldsymbol{V_i} = \langle \mathbf{X}_t | 1 \dots n_i \rangle$ where, $\mathbf{X}_t$ and n_i denote frame t and number of frames, respectively. We consider n_i to be variable across videos. Each video has a ground-truth score annotation $y_i \in [0, 1]$ representing the assessment/quality of the performed exercise. A higher y_i score indicates better performance by the user. The training dataset includes a set of tuples $\{(\boldsymbol{V_i}, y_i) : i \in [0, \mathcal{T}]\}$ where $\mathcal{T}$ represents the total number of training videos associated with the exercise $\mathbf{e}$. We train an end-to-end θ_e parameterized model, $\mathcal{F}$, for exercise $\mathbf{e}$ that can predict a continuous score, $\hat{y}_j$ close to the ground-truth assessment score, y_j^*

[1]Part of this chapter has already been published in Swakshar Deb, M. Fokhrul Islam, Shafin Rahman, and Sejuti Rahman, "Graph Convolutional Networks for Assessment of Physical Rehabilitation Exercises," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 30, pp. 410-419, 2022. (SJR rank: Q1, Impact factor: 3.802).

for a given test video V_j. We formulae $\hat{y}_j$ as follows:

$$\hat{y}_j = \mathcal{F}(V_j; \theta_{\mathbf{e}}), \quad s.t. \quad \hat{y}_j \approx y_j^* \tag{6.1}$$

We assume that each RGBD frame presents a single prominent human subject. A well-known way to model human is to use skeleton data of different joint co-ordinates [53, 54]. In this paper, we adopt skeleton-based human modeling because it is easy to obtain. Advanced RGBD cameras like Kinects can provide automatic skeleton joint co-ordinates in real-time [63, 112]. Other possible choices include the BlazePose [113] and VideoPose3D [114] algorithms. According to studies [115, 116], Kinectv2 [117] and Vicon are reliable sources of skeleton data for motion-based tasks because the output is closely aligned with the ground truth system (i.e., the stereophotogrammetric system).

Suppose, there are N number of joints and each joint has C dimensional co-ordinate vector that depend on the pose estimation algorithm (2D, 3D or 6D [114, 118, 119]. The dimension of tth frame and ith video become $\mathbf{X}_t \in \mathbb{R}^{N \times C}$ and $\boldsymbol{V}_i \in \mathbb{R}^{\mathcal{T} \times N \times C}$, respectively. Different body joints play essential roles while assessing the qualify of the performed exercise. In our experiments, we notice that such roles vary from one exercise to other. Suppose, $\mathbf{M}_j \in \mathbb{R}^{\mathcal{T} \times N \times N}$ is self-attention map of all joints representing their roles/importance for jth video. By analyzing $\mathbf{M}_j$ of many users (patients and experts), we can get meaningful insight about final predicted score. Moreover, it can provide guidance to the user about where to emphasize to achieve better performance. With this motivation, given an RGBD video $\boldsymbol{V_j}$ as input, our overall goals are: (a) Exercise assessment: to predict the score $\hat{y}_j$, (b) Role of body-joints: to calculate self-attention map, $\mathbf{M}_j$.

6.1.2 Solution Framework

We investigate Graph Convolutional Networks (GCNs) [120] as our core solution framework for assessing physical rehabilitation exercises. GCN is a natural choice because GCN represents human skeleton data as graphs to extract features from the topological

structure of neighborhood body-joints. Our final design incorporates a recent spatio-temporal GCN (STGCN) to extract both spatial and temporal feature together as a unified network. Due to the nature of rehabilitation exercise data, our extension addresses the following challenges. (1) Variable-length processing: Unlike related problems with sequential data (action/motion classification in general), rehabilitation exercise data exhibit significant variations within variable-length data. A notable reason could be the exercise actors are largely diverse people, from expert therapists to patients with different disabilities and diseases. Moreover, collected rehabilitation data may originate from constrained lab/gymnasium environments, indoor/outdoor, and home-based settings. Furthermore, based on the therapist's prescription, rehabilitation exercises may need to perform with a variable number of repetitions. As a result, different users task a different amount of time to perform the same exercise with the same number of repetitions. (2) Explicit guidance: Existing exercise assessment approaches converts this task as a classification [121, 122] or regression [53, 54] problem. However, they do not provide any explicit guidance about where (body-joints) to emphasize or attend to improve the assessment quality.

We first describe the preliminaries of a vanilla STGCN adapted to address rehabilitation exercise assessment task. Then, we show that it can predict a baseline assessment score, but it cannot fully address the challenges mentioned earlier.

Exercise Assessment with Vanilla STGCN: We illustrate the network in Fig. 6.1(a) and (b). We represent human body-joints at each video frame as graph, $G = (\mathcal{A}, \mathcal{V}, \mathcal{E})$ where, $\mathcal{V}$ is set of vertex (body-joints), $\mathcal{E}$ is a set of edges (connection of body-joints) and $\mathcal{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix of the graph G. For spatial configuration partitioning, we dismantle $\mathcal{A}$ into several matrixes $\mathcal{A}_k$, where $\mathcal{A} + \mathbf{I} = \sum_k \mathcal{A}_k$, representing the adjacency matrix of hop k. The initial conditions are $\mathcal{A}_0 = \mathbf{I}$ and $\mathcal{A}_1 = \mathcal{A}$. To extract dependencies from body-joints, we forward the skeleton data of a video, $\boldsymbol{V} \in \mathbb{R}^{\mathcal{T} \times N \times C}$ to STGCN block. Initially, we perform temporal convolution over the input skeleton sequences using kernel Γ^u. Then, we concatenate the input and the temporal features to

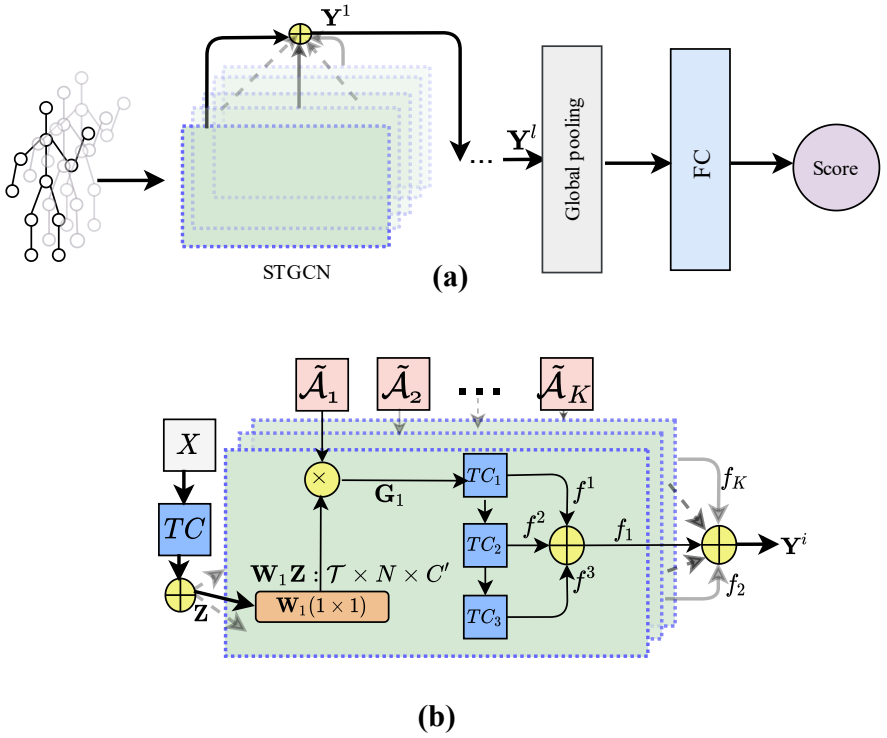


Figure 6.1: 'TC' and ⊕ denote temporal convolution, concatenation and element-wise multiplication, respectively. GCN based end-to-end models using (a) Vanilla STGCN. (b) Illustrate the detailed components of the green STGCN block of (a)

extract spatial feature. All he initial processing can be written as:

$$\mathbf{Z} = \mathbf{X} \oplus (\Gamma^{u} \otimes \boldsymbol{V}) \tag{6.2}$$

Here, $\mathbf{Z} \in \mathbb{R}^{\mathcal{T} \times N \times (C + C^{u})}$ is processed video representation, where, C^{μ} is the number of filter used in temporal convolution, $\otimes$ and $\oplus$ denote the temporal convolution and concatenation respectively. To extract spatial features from the topological structure of human skeleton, we perform graph convolution on $\mathbf{Z}$ and for the kth hop adjacency matrix, $\mathcal{A}_k$ using the update rule of GCN [120] is:

$$\mathbf{G}_k = \sigma(\tilde{\mathcal{A}}_k \mathbf{Z} \mathbf{W}_k) \tag{6.3}$$

where, $\tilde{\mathcal{A}}_k = D_k^{-\frac{1}{2}}(\mathcal{A}_k + I)D_k^{-\frac{1}{2}}$, D is a diagonal degree matrix, $\mathbf{W}_k$ is the learnable weight matrix and σ is an nonlinear activation function. This equation performs linear transformation on the feature space and then aggregate the neighbour information using the normalize adjacency matrix. Then, we implement three Temporal Convolutional

Layers (TCNs) with same padding and kernel Γ_1^l, Γ_2^l and Γ_3^l correspondingly to extract different level of temporal features. To recognize movement patterns at different levels of abstraction, we concatenate both higher and lower level features. The operations involve: $f^1 = \Gamma_1^l \otimes \mathbf{G}_k$, $f^2 = \Gamma_2^l \otimes f^1$, $f^3 = \Gamma_3^l \otimes f^2$ and $f_k = f^1 \oplus f^2 \oplus f^3$, where f_k represents the spatio-temporal features extracted from the kth hop. Finally, we concatenate outputs from each hop.

$$\mathbf{Y} = f_1 \oplus f_2 \oplus \ldots \oplus f_k \tag{6.4}$$

The output of the STGC block, $\mathbf{Y} \in \mathbb{R}^{\mathcal{T} \times N \times C_{out}}$, where, $C_{out} = \sum_{i=1}^{3} kC_i^l$ is a 3D tensor. In experiment, we stack multiple STGCN blocks to extract more complex features. Next, we apply a global average pooling to the output of the last STGCN, $\mathbf{Y}^l$, and calculate the feature vector $Y_{pool} \in \mathbb{R}^{C_{out}}$ is further processed by a series of FC layers to predict a continuous assessment score.

Issues with Vanilla STGCN: The extracted spatio-temporal features, $\mathbf{Y}_{pool}$ can provide a baseline assessment score, but there is scope for improvement. In Fig. 6.1(a), one can notice that the global pooling layer before the FC layers ignores sequential dependencies resided among the spatio-temporal features across frames/body movements. Thus, users performing the same exercise at different pace (slow or fast) extract different spatio-temporal features. This issue becomes more critical in regression-based learning. Furthermore, different exercises pay particular emphasis to specific joints. But, vanilla STGCN treats all body-joints equally. It cannot provide the role/importance of joints in predicting the assessment score.

6.1.3 Our Extension

In this subsection, we extend the vanilla STGCN for the rehabilitation exercise assessment task to address the issues mentioned above. We include two components to address variable-length processing and calculating the role of body-joints to guide users explicitly.

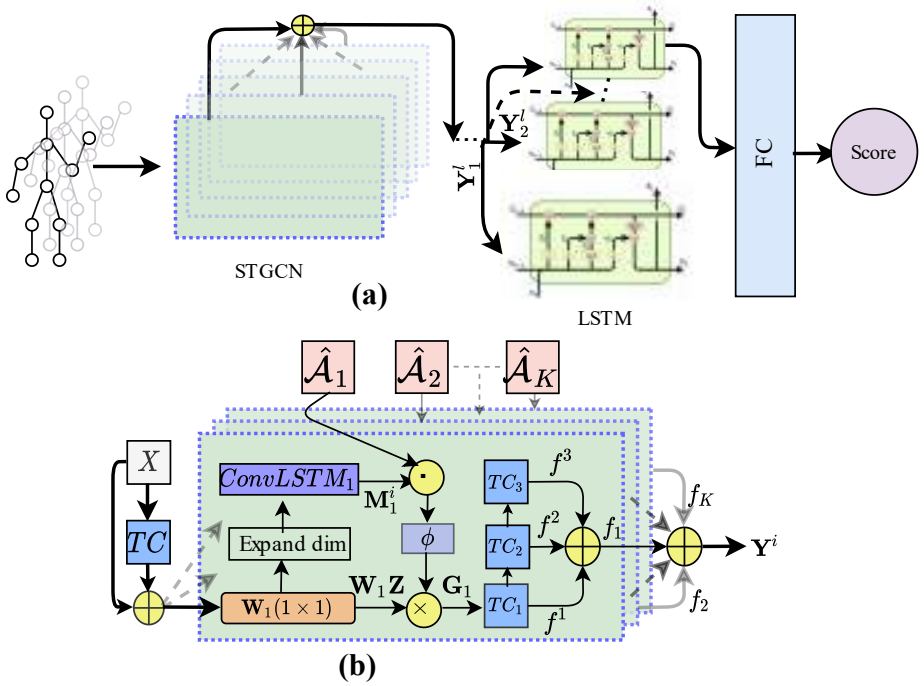


Figure 6.2: 'TC', ⊕ and ⊙ denote temporal convolution, concatenation and element-wise multiplication, respectively. (a) Extended STGCN end-to-end models for rehab exercise assessment. (b) Illustrate the detailed components of the green STGCN block of (a).

Variable-length Processing: We employ an LSTM instead of global pooling layers (see Fig. 6.2(a) and (b)). It has some benefits over pooling. (1) LSTM captures sequential dependencies presented in spatio-temporal feature vectors. It is inherently designed to extract discriminative features that have accumulated over time. This subtle information plays an important role in predicting the correctness score of an exercise. Smoothness, for example, is an important criterion for scoring a given exercise. We must examine the temporal features (velocity, acceleration) overall conjugative time frames to discern the smoothness of a movement (see Table 6.5 for experimental evidence). Using a pooling layer instead of an LSTM may fail to extract the smoothness information. (2) Pooling layers take average or maximum response for predefined window size. It may lose some subtle features necessary for predicting correctness score. Instead, LSTM searches for more meaningful regions from these extracted features. We reshape, $\mathbf{Y}^l$ to $\mathbf{Y}_r \in \mathbb{R}^{\mathcal{T}\times NC_{out}}$ where $\mathbf{Y}^l$ is the output from the last STGCN. The input to the LSTM is $\mathbf{Y}_r^t \in \mathbb{R}^{NC_{out}}$, the tth sequence of Y_r. Considering the characteristics of the rehabilitation exercise data and regression-based problem instead of classification, LSTM fits better instead of global pooling. LSTM considers the variation of spatio-temporal features

over the temporal dimension, whereas global pooling breaks the sequential nature of data. Therefore, it helps variable-length processing and allows users to perform the same exercise at different paces.

Role of Body-joints: The role of each joint is different in each exercise. For example, in the squatting exercise, (Ex 5), the ankle, knee, spine, and shoulder play an important role, whereas, in the lifting arms exercise (Ex 1), the elbow, spine, thumb, and wrist joint are more significant than the rest of the joints. We want to emphasize that capturing this joint role is crucial in determining exercise quality (Table 6.5). However, vanilla STGCN (in Eq 6.3) treats all body joints equally. This joint role should change depending on temporal and spatial context while assessing rehabilitation exercise. It motivates us to implement the self-attention module where we treat each joint differently depending on its role in a given exercise. To calculate the role of individual body-joints depending on the both temporal and spatial context, we replace the $\tilde{\mathcal{A}}_k$ with a trainable self-attention map, ($\mathbf{M}_k \in \mathbb{R}^{\mathcal{T}\times N\times N}$) as follows:

$$\mathbf{G}_k = \sigma(\phi(\hat{\mathcal{A}}_k \odot \mathbf{M}_k)\mathbf{Z}\mathbf{W}_k) \tag{6.5}$$

Where, $\odot$ represents the Hadamard product, $\hat{\mathcal{A}}_k = \mathcal{A}_k + I$ and $\phi(\cdot)$ is the normalizing factor. Fig. 6.2(b) illustrates the attention-guided STGC block. We improve STGCN by modifying the adjacency matrix dynamically with a self-attention map calculated from ConvLSTM layers. Let, $\mathbf{Q}_k = ZW_k \in \mathbb{R}^{\mathcal{T}\times N\times C'}$ expanded to $\mathbf{Q}_k$ to $\mathbb{R}^{\mathcal{T}\times N\times 1\times C'}$, where, C' = no of convolutional filters, is the input to the ConvLSTM. The operations at time t are as follows:

$$i_k^t = \sigma(W_i * \mathbf{Q}_k^t + U_i * h_k^{t-1} + b_i) \tag{6.6}$$

$$f_k^t = \sigma(W_f * \mathbf{Q}_k^t + U_f * h_k^{t-1} + b_f) \tag{6.7}$$

$$o_k^t = \sigma(W_o * \mathbf{Q}_k^t + U_o * h_k^{t-1} + b_o) \tag{6.8}$$

$$g_k^t = \tanh(W_c * \mathbf{Q}_k^t + U_c * h_k^{t-1} + b_c) \tag{6.9}$$

$$c_k^t = f_k^t \odot c_k^{t-1} + i_k^t \odot g_k^t \tag{6.10}$$

$$h_k^t = o_k^t \odot \tanh(c_k^t) \tag{6.11}$$

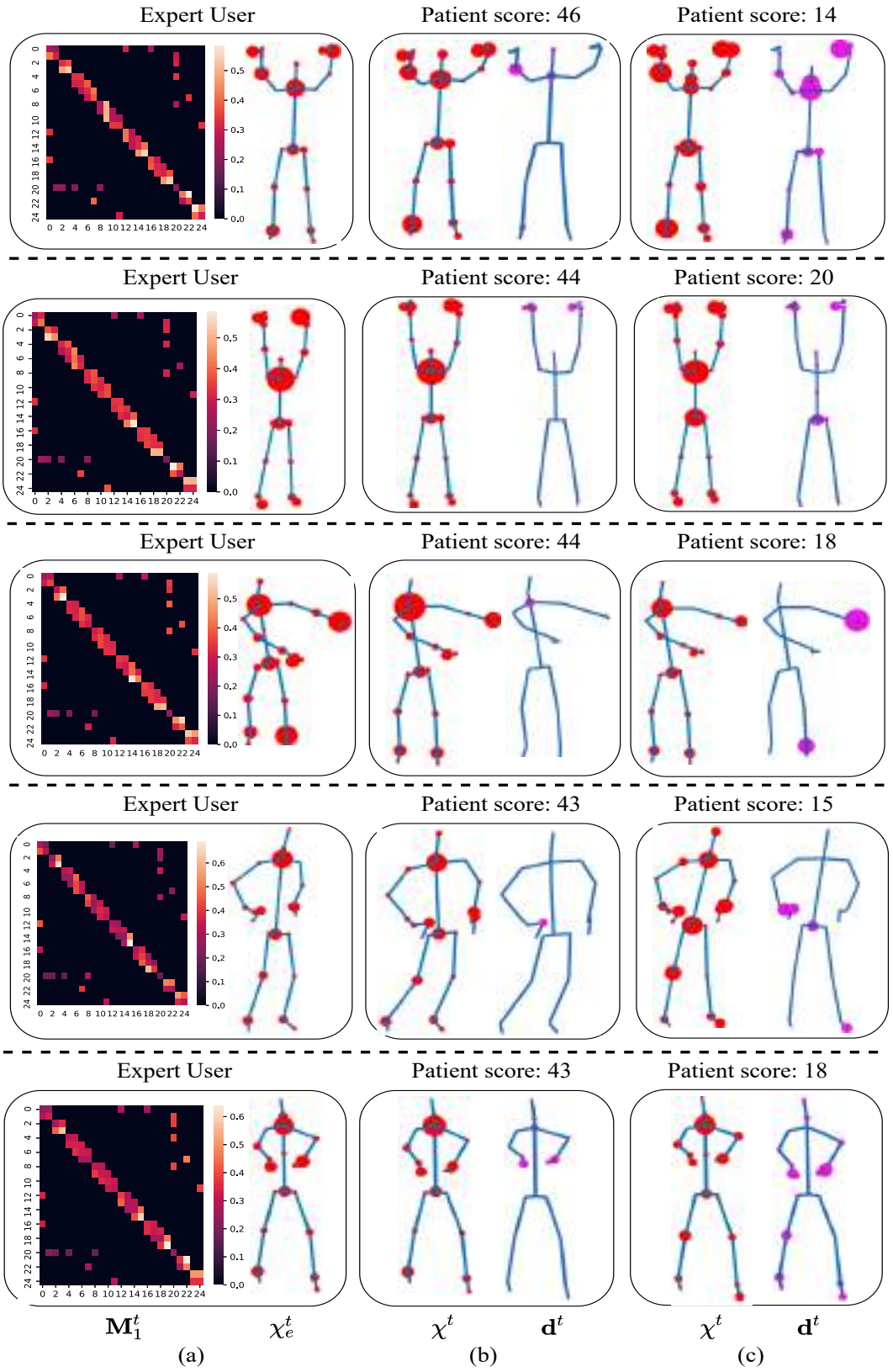


Figure 6.3: A detailed illustration of the role of different body-joints (red circles) while performing five KIMORE exercises by a) expert users, b) & c): patients with higher & lower scores, respectively. The larger circle represents the higher role of that joint. In (b) & (c): left skeletal Fig. shows role of joints (χ^t) of the patient where right shows the deviation from that (χ_e^t) of the expert user. This deviation map (d^t) can guide a patient to get better assessment score by fixing his joint roles.

where, $*$ is the convolution operation, σ represents the sigmoid function, W_i, W_f, W_o, W_c, $U_i, U_f, U_o, U_c \in \mathbb{R}^{1\times1}$ represent the conv. kernels, b_i, b_f, b_o, b_c are the bias parameter. The number of kernels is the same as the number of joints in the skeleton graph. The final output from convLSTM does not encapsulate any structural information. We inject the graph structure by elementwise multiplication with the adjacency matrix (Eq. 6.5). Then, we apply a normalizing factor $\phi(\cdot)$. $\mathbf{M_k} = h_k \in \mathbb{R}^{\mathcal{T}\times N\times N}$, is the self-attention map where each row indicate the attention weights for a body-joints with its neighbors. We can calculate the role of the body-joints in assessing rehabilitation exercises. Fig. 6.3(a) shows a self-attention map where we highlight the role of body joints. The greater attention value demonstrates the higher emphasis/role on that joint. Some joints may get the same importance but contribute to both high and low scores across trials. It means that those joints have little influence on the overall score because subjects have moved them ideally, and there is nothing much to improve scores by focusing on those joints. On the other hand, some joints are more prominent in determining low scores. Such joints are the ones on which the patient should focus. We can calculate such roles of joints for both expert and inexpert (possibly patient) users. We notice that when a patient gets a low assessment score (0-20), his/her joint-role pattern varies significantly in comparison to the expert's pattern (35-40). The joint role, $\chi \in \mathbb{R}^{\mathcal{T}\times N}$ is computed by column-wise summation over $\mathbf{M}_1 \odot A_1$, where M_1 denotes the first hop attention map. In Fig. 6.3(b) and (c), we calculate $d^t = \chi_e^t - \chi^t$ to find this pattern variation at time t. χ_e is calculated by averaging the joint role of the expert therapists. Visualizing d^t, a user can know where to emphasize to achieve a better score in future trails.

We train our proposed network using Huber loss because of less sensitivity to outliers [123]. We simply forward a test sequence of skeleton data (body-joints) to calculate a continuous assessment score during inference. We employ the Huber loss function which is defined as follows:

$$L(y_i - \hat{y}_i) = \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2; & |y_i - \hat{y}_i| \leq \delta \\ \delta(|y_i - \hat{y}_i| - \frac{\delta}{2}); & otherwise \end{cases} \tag{6.12}$$

where y is the true label and $\hat{y}$ is the predicted label. This Huber loss uses the properties

of both the Mean squared loss and the Mean absolute deviation loss depending on a parameter δ. When the error (i.e., the difference between actual and the estimated value) is less than a small value δ, it acts as the Mean squared loss. When the error is greater than δ, it approaches the mean absolute loss, which is less sensitive to the outliers. The skeleton data captured by various pose estimation algorithms (Kinectv2, Vicon, BlazePose, VideoPose3D) may contain incorporate incomplete, noisy or redundant information of joint positions [124, 125]. This incomplete or noisy skeleton data can deteriorate the performance, especially when some prominent joints are disturbed. They act as outliers in the sample space. The Huber loss is less sensitive to those outliers.

Inference. In inference, we utilize the same architecture as shown in Fig. 6.2. For a test video $\mathbf{X} \in \mathbb{R}^{\mathcal{T} \times N \times C_{in}}$ where $\mathcal{T}$ can be of any length, first we construct the skeleton graph. The skeleton data can be extracted from blazepose [113] or any motion-capture devices. For STGC block we use skeleton graph as the input. Inside each STGC block, first we perform the temporal convolution operation and concatenate with the original input namely $\mathbf{Z} \in \mathbb{R}^{\mathcal{T} \times N \times (C + C^{\gamma})}$. Second, inside each graph convolutional network we perform attention guided graph convolution operation over $\mathbf{Z}$. Finally there are three temporal convolutional layers concatenated together. There can be multiple stacked STGC blocks and after the last STGC block there is multiple LSTM layers followed by a dense layer to predict the final score.

6.2 Experiment

6.2.1 Setup

Dataset: We experiment on two rehabilitation exercise datasets. (1) Kinematic assessment of Movement for remote monitoring of physical Rehabilitation (KIMORE) [112] dataset includes RGBD videos and the ground-truth score annotations of five types of exercises. It has two groups, control groups (expert and non-expert) and a group with pain and postural disorder (Parkinson, back-pain, stroke). The control group includes 44 healthy subjects, from which 12 subjects were physiotherapists and experts in the

Table 6.1: Summary of the both KIMORE and UI-PRMD datasets

Feature	UI-PRMD dataset [63]	KIMORE dataset [112]
Reference	Vakanski et al. (2018)	Capecci et al. (2019)
Year	2018	2019
Sensor	Kinect v2 + Vicon	Kinect v2
Modality	Skeleton data	RGB-D and skeleton data
No. of Subjects	10	78
No. of Exercises	10	5
Score range	0 - 1	0 - 50

Table 6.2: Computational cost for KIMORE dataset

Stage	# of Videos	# of Parameters	Execution Time	Avg. time per video
Train	460	0.722 million	50.13 minutes	–
Test	116	0.722 million	2.64 seconds	22.7 milliseconds

rehabilitation of back pain and postural disorders, and the remaining 32 were non-expert healthy subjects. The pain and postural disorder group contain 34 subjects suffering from chronic motor disabilities. (2) UI-PRMD [63] dataset contains ten rehabilitation exercises collected from 10 healthy subjects using Kinect and Vicon sensors. Each subject performed ten repetitions of the same activity. The data includes positions and angles of full-body joints. A summary of these two datasets is given in Table 6.1.

Computational cost: In Table 6.2, we report training/testing time for the KIMORE dataset. The timing suggests that our proposed method can provide results in real-time, considering real-time skeleton data generation is available. Note that the computational cost is measured using a single Tesla K80 GPU.

Evaluation Process: Similar to [53, 54, 126], we evaluate our model using mean absolute deviation (MAD), mean absolute percentage Error (MAPE) and root mean square error (RMSE) scores. The lower the score, the more accurate is the predicted score. Mean Absolute deviation (MAD) is an average of the absolute deviation between true values

and predicted values. MAD, being a scale-dependent measure, cannot be applied to compare methods that are applied to data with different scales. On the other hand, Mean Absolute Percentage Error (MAPE), the percentage equivalent of MAD, is a scale-independent metric. However, MAPE tends to infinity or becomes undefined if the ground truth value equals 0 for any sample in the data. Another widely used evaluation metric is Root Mean Squared Error (RMSE) which is defined as the square root of the squared error (no absolute deviation). RMSE penalizes large errors due to the squared term. It is also a scale-dependent metric like MAD. The equations of MAD, MAPE and RMSE are given as follows:

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|y - \hat{y}|$$

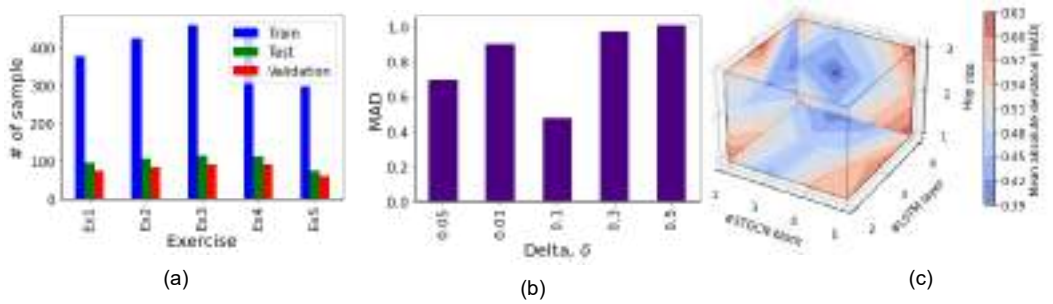
$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y - \hat{y}}{y}| \times 100$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2}$$

Here, n is the sample size and y and $\hat{y}$ are the ground-truth and predicted value respectively.

Implementation Details[2]: We train the model using Adam optimizer for 1500 epochs with a learning rate of 0.0001. The batch size is 3 and 10 for UI-PRMD and KIMORE dataset, respectively. We use cross-validated hop size, $k = 2$. The output space dimensionality of LSTM layers is $80, 40, 40, 80$ followed by a fully connected layer with linear activation. According to the validation set, we choose the best model to evaluate the model performance on the test set. We apply a dropout mechanism with a dropout probability of 0.25. Similar to [54, 120], we also report the 10-run result to fairly evaluate the performance of our model against existing works. We perform both training and testing ten times. After each run, we store the performance metrics (MAD, RMSE, MAPE) and finally take the average of stored results, ensuring the reliability of our results. We implement our model using tensorflow2.0. Similar to [127], we apply the Resnet mechanism on each STGC block.

[2]Code and evaluation are available at: `https://github.com/fokhruli/STGCN-rehab`



(a) (b) (c)

Figure 6.4: (a) Train-test-val split on KIMORE dataset. (b) Performance of our proposed model on different values of δ. The optimal δ value is 0.1. (c) Hyper-parameter validation on KIMORE exercise 5 (in MAD). The cross ($\times$) indicates the optimum point.

Hyper-parameter Validation: We conduct hyperparameter tuning on a separate validation set. We set the validation split = 0.2 from the total training set inside the `fit()` method of tensorflow 2.0. The train-test-valadiation split is shown in Fig. 6.4 (a). We validate the model for hop size k within $\{1, 2, 3\}$, the number of STGCN blocks within $\{1, 2, 3, 4, 5\}$, the number of stacked LSTM layers within $\{2, 3, 4, 5\}$, the learning rate within $\{10^{-3}, 10^{-4}, 10^{-5}\}$. When the model suffers from high variance, we can decrease the hop size (k), STGCN, and LSTM layers, reducing model complexity. In contrast, we can increase the values of those hyper-parameters to lessen the bias problem. Based on those validation experiments, we use $k = 2$, three STGC blocks, four LSTM layers and learning rate 10^{-4} for our final model. After finding the optimal model, we conduct experiment on δ sensitivity in Fig. 6.4(b). Fig. 6.4(c) shows the validation results (MAD) of our method using different numbers of hop size, STGCN blocks, and stacked LSTM layers for KIMORE exercise 5. It shows that with the increase of STGCN block number from 2 to 3, the performance is boosted, while further increasing the block number leads to no significant improvement. As we increase hop size from 1 to 2, the performance is improved, but for hop size 3 the overall performance deteriorates since there is an influence of unnecessary joints disrupting the informative features for larger neighborhood size. Moreover, increasing the stacked LSTM layers, the model's overall performance continues to improve since LSTM is better suited for extracting meaningful features from sequential dependencies. As we increase the number of LSTM, the performance increases significantly, but after 4 LSTM layers, there is a slight improvement. Considering the computational cost, we select LSTM layers to be 4.

Table 6.3: Results of ten exercises (Ex) on the UI-PRMD dataset using the evaluation metric MAD (lower values indicate better results).

Ex	Ours	Song et al.[124]	Zhang et al.[128]	Liao et al. [54]	Li et al. [129]	Shahroudy et al. [130]	Du et al.[131]
Ex1	0.009	0.011	0.022	0.011	0.011	0.018	0.030
Ex2	0.006	0.006	0.008	0.028	0.029	0.044	0.077
Ex3	0.013	0.010	0.016	0.039	0.056	0.081	0.137
Ex4	0.006	0.014	0.016	0.012	0.014	0.024	0.036
Ex5	0.008	0.013	0.008	0.019	0.017	0.032	0.064
Ex6	0.006	0.009	0.008	0.018	0.019	0.034	0.047
Ex7	0.011	0.017	0.021	0.038	0.027	0.049	0.193
Ex8	0.016	0.017	0.025	0.023	0.025	0.051	0.073
Ex9	0.008	0.008	0.027	0.023	0.027	0.043	0.065
Ex10	0.031	0.038	0.066	0.042	0.047	0.077	0.160

6.2.2 Overall Result

We report our results on UI-PRMD and KIMORE datasets in Table 6.3 and 6.4, respectively. Following the work [54], the table mentions MAD performance on each of ten different exercises included in UI-PRMD dataset. Similarly, we present MAD, RMS, and MAPE results on five exercises of the KIMORE dataset. Liao et al.[54] proposed a temporal pyramid network to process the multiple-scale version of the movement repetitions. The initial hierarchical layers in the model employ for learning spatial dependencies in human movements and are followed by a series of LSTM recurrent layers for modeling temporal correlations in learned representations. Still, they ignored the human skeleton's topology information, failing to extract the expressive power residing in spatial features. On the other hand, Yan et al.[127] built a spatio-temporal graph neural network (STGCN) while neglecting the sequential nature of the spatio-temporal features because of using global average pooling. Moreover, Song et al. [124] also proposed a GCN to explore discriminative features that spread over all skeleton joints using multi-stream information (occlusion, jittering, etc.) and focus on the spatial context to

Table 6.4: Results of five exercises (Ex) on KIMORE using the evaluation metrics MAD, RMS, and MAPE (lower values indicate better results).

Metric	Ex	Ours	Song et al.[124]	Zhang et al.[128]	Liao et al.[54]	Yan et al.[127]	Li et al.[129]	Du et al.[131]
MAD	Ex1	0.799	0.977	1.757	1.141	0.889	1.378	1.271
	Ex2	0.774	1.282	3.139	1.528	2.096	1.877	2.199
	Ex3	0.369	1.105	1.737	0.845	0.604	1.452	1.123
	Ex4	0.347	0.715	1.202	0.468	0.842	0.675	0.880
	Ex5	0.621	1.536	1.853	0.847	1.2184	1.662	1.864
RMSE	Ex1	2.024	2.165	2.916	2.534	2.017	2.344	2.440
	Ex2	2.120	3.345	4.140	3.738	3.262	2.823	4.297
	Ex3	0.556	1.929	2.615	1.561	0.799	2.004	1.925
	Ex4	0.644	2.018	1.836	0.792	1.331	1.078	1.676
	Ex5	1.181	3.198	2.916	1.914	1.951	2.575	3.158
MAPE	Ex1	1.926	2.605	5.054	2.589	2.339	3.491	3.228
	Ex2	1.272	3.296	10.436	3.976	6.136	5.298	6.001
	Ex3	0.728	2.968	5.774	2.023	1.727	4.188	3.421
	Ex4	0.824	2.152	3.901	2.333	2.325	1.976	2.584
	Ex5	1.591	4.959	6.531	2.312	3.802	5.752	5.620

learn the joint attention. Zhang et al. [128] incorporated high-level semantics of joints (joint type and frame index) into the network with the help of joint- and frame-level modules to hierarchically exploit the joint relationship. However, they fail to capture strong sequential dependencies between consecutive frames through several spatial and temporal maxpooling layers. Du et al.[131] did use temporal information but ignored the spatial information. Li et al.[129] ignored both the topological structure and the sequential nature of human body-joint features. Our model successfully outperforms existing methods in the rehabilitation exercise assessment task on both datasets. This success becomes possible because of the spatio temporal graph network with learned anisotropic filters by the self-attention mechanism that separately considers spatial and

Table 6.5: Ablation study on Ex5 (Exercise 5) of KIMORE dataset. We incrementally add more components to compare the performance (lower values indicate better result).

Is-Stacked	Aggregation Style	Has-TCN Concatenated	Has-self attention	MAD	RMSE	MAPE
No	Global Pool	No	No	2.585	3.795	8.920
Yes	Global Pool	No	No	1.472	2.560	4.878
Yes	Global Pool	Yes	No	1.365	2.184	4.320
Yes	LSTM	Yes	No	0.767	1.484	2.340
Yes	LSTM	Yes	Yes	0.478	0.981	1.516

Table 6.6: Results for evaluating Exercise 5 of the KIMORE dataset with MAD, RMS, and MAPE while collecting joint/skeleton data using various pose estimation algorithms.

Metric	Algorithm	Ours	Liao et al. [54]	Yan et al. [127]	Li et al. [129]	Du et al. [131]
MAD	BlazePose [113]	0.971	4.043	3.709	4.548	6.309
	VideoPose3D [114]	1.855	2.554	3.084	3.546	4.669
	Kinectv2 [117]	0.621	0.847	1.218	1.663	1.864
RMSE	BlazePose [113]	1.993	5.991	5.657	7.194	8.681
	VideoPose3D [114]	3.822	3.908	4.943	5.202	6.012
	Kinectv2 [117]	1.180	1.914	1.951	2.575	3.158
MAPE	BlazePose [113]	3.081	15.618	15.917	20.897	25.816
	VideoPose3D [114]	6.810	8.102	10.790	11.964	14.750
	Kinectv2 [117]	1.591	2.312	3.802	5.752	5.620

temporal directions. Moreover, our model fully utilizes the benefit of deep learning techniques, ensuring end-to-end learning. For [124, 127, 128, 129, 131], we replaced the last softmax layer with a fully-connected layer with linear activations. Other than that, we closely followed the proposed implementation as described by the authors in the respective papers.

Ablation Study: In Table 6.5, we report different variations of GCN assessing rehabilitation exercises. We include or replace different components to estimate the contribution of them. Here, we experiment with and without stacked STGCN, spatio-temporal feature aggregation style (global pool/LSTM), the inclusion of concatenated TCN outputs, and self-attention (ConvLSTM based joint role) components. First, we compare a plain (without stacked) GCN vs. stacked GCN using global pooling as feature aggregation and without using any other parts. We notice performance improved in the stacked case because of extracting more complex features. On top of the stacked version, we add TCN feature concatenations mentioned in Sec. 6.1.2. It helps to improve performance because of considering temporal information. Results further improve while replacing the global pool with LSTM (see Sec. 6.1.3), since it augments more spatio-temporal information. Finally, we include the self-attention mechanism using ConvLSTM, which is our final recommendation. We achieve the best results in this configuration because of dynamically calculating the role of different joints while assessing exercises.

Feature Visualization:

After training our final model with variable-length data, we also test our model with fixed-length input. We create four fixed-length versions (100, 150, 170 and 200 frames) of the same test data. These fixed-length versions represent rehabilitation exercises performed at different (slow/fast) pace and varying repetitions. We extract the latent feature representation after the LSTM layer for both variable-length and fixed-length test data as input. Fig. 6.5 shows those features using the 2D t-SNE plot of several exercises from the KIMORE dataset. One can notice that our model provides similar feature representations for different input lengths. It tells that our model can successfully assess physical rehabilitation exercises no matter how many repetitions or how slowly users perform the movement.

Effect of joints in different exercises: By analyzing the natural topological structure of the human body and extracting spatial information using attention-guided graph convolution, we treat each joint differently based on spatial and temporal context. It produces anisotropic filters that are more powerful than isotropic filters used in vanilla

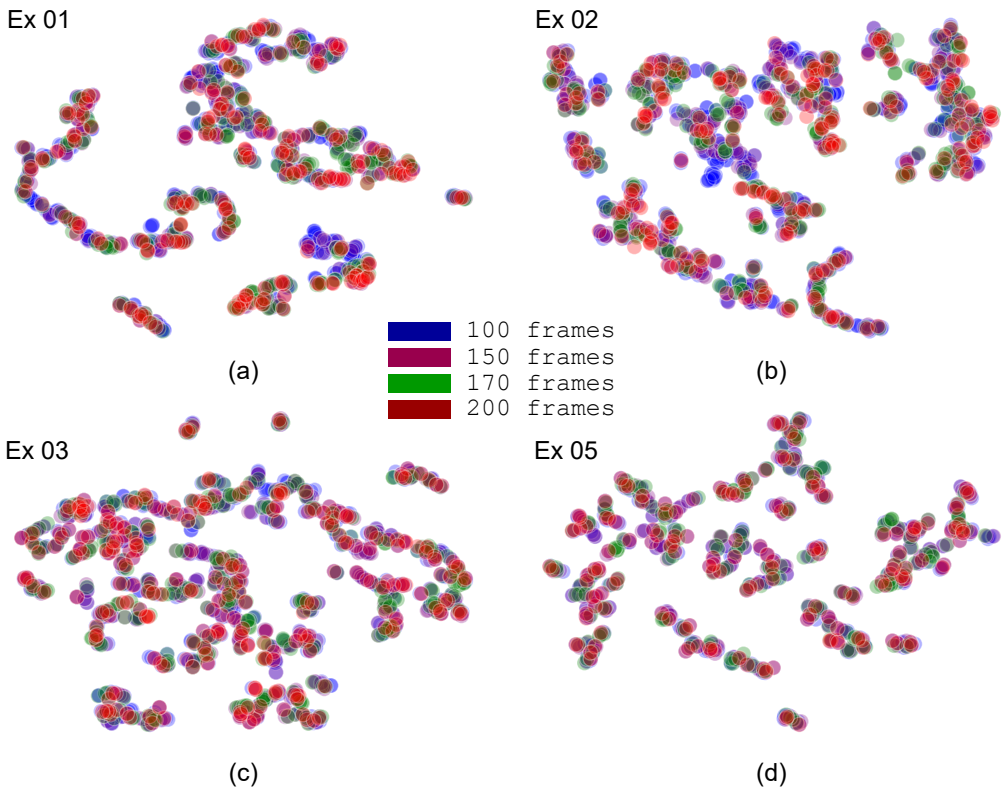


Figure 6.5: t-SNE [132] visualization of features for two exercise videos (Ex2 and Ex3) from KIMORE dataset by representing exercises in different lengths. Our model extracts similar features from different lengths of the same video.

STGCN. In Fig. 6.6, we show the effect of different joints (from expert users) on individual exercises belonging to the KIMORE dataset.

Result using RGB camera: The KIMORE and UI-PRMD datasets collect joint positions using Microsoft Kinectv2 [117] and Vicon sensors. However, rather than solely based on RGBD sensors, we also experiment with economically available RGB videos. We use pre-trained pose estimation algorithms to extract the 3D pose of a human being. The KIMORE dataset [112] provides RGB videos of patients performing rehabilitation exercises. We validate the performance of our model with two other pose estimator methods, namely BlazePose [113] and VideoPose3D [114] trained on MS Coco [133] and Human3.6M [134] datasets, respectively. Both ways use RGB information to estimate the 3D pose of a movement. In Table 6.6, we compare the performance of our model with different pose estimation algorithms. Since the Microsoft Kinectv2 sensor uses RGBD information to detect human poses, it outperforms the other two pose estimation algorithms based on RGB cameras.

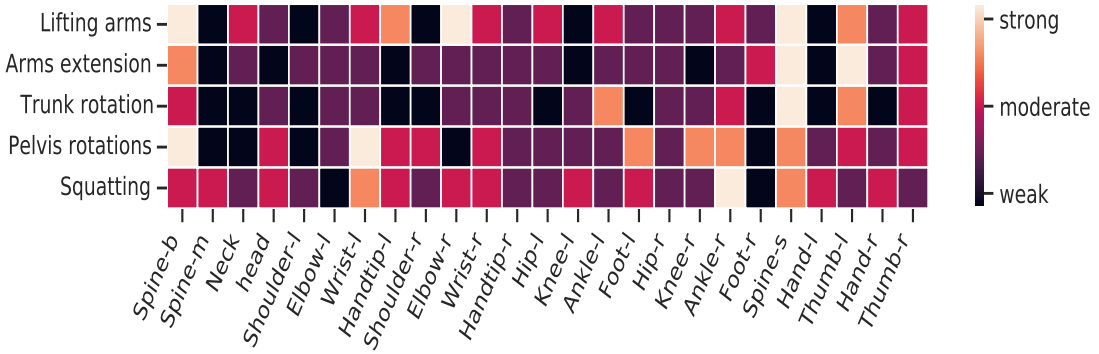


Figure 6.6: A visualization of how the role of different joints varies with different exercises as determined by the attention-value computed by our method. For example, in the lifting arm exercise (Ex 1), the elbow, spine, thumb, and wrist play a more important role than the rest of the joints. Similarly, in pelvis rotation (Ex 4), spine base and wrist contribute more than other joints. These findings closely align with the KIMORE dataset paper that discusses the variation of role of joints in individual exercises as suggested by medical professionals.

6.3 Experimental Data Collection

In order to analyse the performance of our proposed algorithm in the context of real time prediction, we collect the dataset using Microsoft Kinect v2 as a RGB-D vision sensor. The Microsoft for Windows v2 sensor uses a novel Time-of-Flight (ToF) technology while the previous sensor (Kinect v1) belongs to the category of Structured Light (SL) cameras. Compared with cameras based on SL technology, ToF cameras have a longer range and the images appear to be more accurate without holes in the depth map [62]. The depth map reflects the round-trip time of flight for single laser pulses. Compared with the previous version, Kinect v2 provides a higher depth map resolution (512×424 vs 320×240), allowing thin objects to be recognized and solving some ambiguity problems. Moreover, Kinect v2 is an inexpensive, unobtrusive and easy to set up sensor that can be used both in home and clinical environments to monitor subjects during physical rehabilitation. The depth features allow the recognition of different subjects and different body parts in the field of view, while the increase in resolution permits the identification of the 3D points of 25 distinct body parts at 30fps. Compared with the previous version, a bio-correction allows each joint to be mapped consistently with an anatomic reference.

We considered the Squatting rehabilitation exercise that are used widely in Bangladesh.

The dataset is illustrated in Fig 7.4, where each subject demonstrating the exercise in correct and incorrect manner.

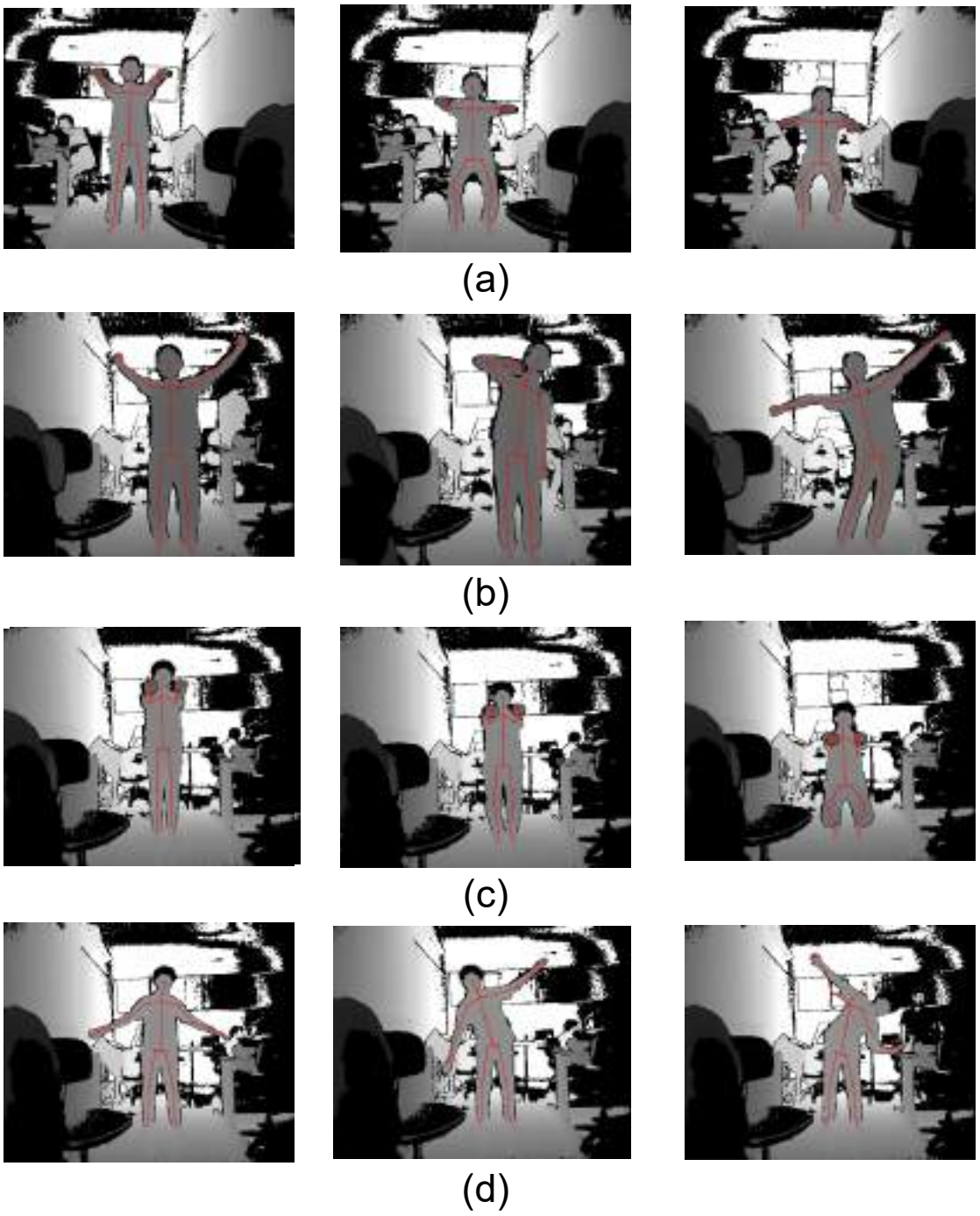


Figure 6.7: Different subject performing exercise in home set up. (a) subject 1 performing right exercise (b) subject 1 performing wrong exercise (c) subject 2 performing right exercise (d) subject 2 performing wrong exercise

Table 6.7: This table shows Score of 5 different subjects in both correct and way on Squatting exercise (exercise 5). Data was collected by kinect sensor for several trials and evaluated by our proposed method(Model trained on KIMORE dataset).

Subject	Correct			Incorrect		
	MAD	RMSE	MAPE	MAD	RMSE	MAPE
Subject 1	0.55	1.25	1.52	0.75	1.82	2.25
Subject 2	0.65	1.30	1.65	0.35	0.74	0.86
Subject 3	0.49	0.95	1.33	0.72	1.63	2.13
Subject 4	0.52	1.12	2.05	0.46	0.86	1.23
Subject 5	0.61	1.07	1.82	0.34	0.72	0.83

Table 6.7 represents the real time prediction of our proposed rehabilitation model of the Squatting exercise (exercise 5), where each perform three trial consisting of both correct and incorrect movement pattern. We evaluate the total score out of 50, where the higher values indicate better exercise quality.

Chapter 7

Automated Analysis of Vital Signs

Now-a-days Artificial intelligence is vastly being used to analyze large amounts of clinical and primary vitals in health care. It has shown that AI can be trustworthy not only analyzing primitive vitals but also making critical decisions. We are living in an age where we have a glut of data. It is impossible for humans to process this quantity of data. On the other hand the AI is more than capable of handling this data and transforming it to usable outputs or responses. To evaluate the data there are some scoring methods such as quick Sequential Organ Failure Assessment (qSOFA), Systemic Inflammatory Response Syndrome (SIRS) etc. As we are dealing with more usable and feasible scoring system, we choose National Early Warning Score (NEWS) scoring systems that includes some vital parameters which can be measured by patient himself without needing helps of others.

NEWS is a well validated track-and-trigger early warning score system that is used to identify and respond to patients at risk of deteriorating. It is based on a simple scoring system in which a score is allocated to physiological measurements already undertaken when patients present to, or are being monitored in health care settings. Six parameters are used to evaluate the critical level of the patient.

A score is allocated to each physiological parameter, the magnitude of the score reflecting how extreme the parameter varies from the norm. This score is then aggregated, and uplifted for people requiring oxygen.

We inherited the scoring mechanism in our system. Based on the parameters they are measuring we fixes some similar vitals those are given as such:

1. Oxygen saturations - a measure of how much hemoglobin is currently bound to oxygen compared to how much hemoglobin remains unbound
2. Temperature - the body temperature
3. Systolic blood pressure - measures the force the heart exerts on the walls of the arteries each time it beats
4. Diastolic blood pressure- measures the force the heart exerts on the walls of the arteries in between beats
5. Pulse rate- a measurement of the heart rate, or the number of times the heart beats per minute

7.1 Vitals Processing

The main challenges were to measure the vitals accurately and process with germane methods. To measure the vitals we choose convenient instruments so that the patients don't have to face any difficulties.

7.1.1 Measurement processes of the Vitals:

The system has different measurement instruments for taking the vital readings.

Vital	Measuring Instrument
Blood pressure (Both systolic and diastolic)	?
Oxygen saturation and pulse rate	?
Temperature	?

Both pressure(?) meter and oxygen(?) meter have seven segment digital display. When user measures the vitals, the display shows the values. So to read the values from the display the system uses Optical Character Recognition(OCR).

As seven segment font is not very similar to normal digit font, we faced challenges to find the appropriate algorithm. However, we noticed PaddleOCR is a good fit to our system. It can accurately extract text from the display.

PaddleOCR: PaddleOCR is an ocr framework or toolkit which provides multilingual practical OCR tools that help the users to apply and train different models in a few lines of code. PaddleOCR offers a series of high-quality pretrained models. This contains three types of models to make OCR highly accurate and close to the commercial products. It provides Text detection, Text direction classifier and Text recognition. PaddleOCR offers various models in its toolkit, including the flagship PP-OCR and the latest algorithms such as SRN, NRTR and more.

These are some results of taking readings from the displays of meters

(a) (b)

Figure 7.1: Extracting text from blood pressure meter using PaddleOCR

7.1.2 Utilizing vitals

There are some critical ranges of value for each of the vitals, based on a practical dataset, we trained a machine learning model to get fit with the appropriate limits of the parameters. The model got maximum accuracy in training as well as testing. By using

this model we can accurately predict whether the patient is in critical condition or not. The predictions are in between 4 values. 0 : Normal 1 : Mild 2 : Severe 3 : Critical

If the values of the vitals are in expected range then the prediction will give 0 (Normal). Similarly, when the measurements of the vitals crosses the required boundary, the model gives 3 (Critical) as output.

(a) (b)

(c)

Figure 7.2: Ranges of each vital

7.2 Training with vitals dataset

In order to make an accurate model based on our vital parameter's dataset we have chosen Random Forest Classifier. The reason behind selecting the random forest classifier is, this algorithm is considered as a highly accurate and robust method because of the number of decision trees participating in the process. Our dataset is very sensitive to the specific ranges of every parameter that is why there was a high chance for overfitting in

training. However, this selected algorithm does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.

Random Forest Classifier: Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that

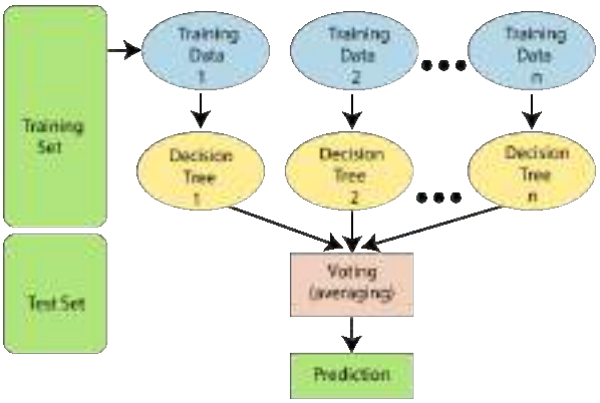


Figure 7.3: Random Forest Classifier

contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

7.2.1 Training Dataset:

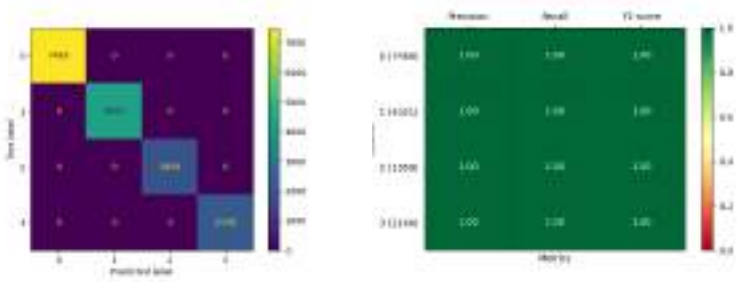
The dataset we used for training consists of 80k data samples with 5 features. In training, we used 70% of the full dataset. And performed testing on the rest of the data.

TEMPF	PULSE	BPSYS	BPDIAS	POPCT	SCORE
99.1	90	129	75	99	0
97.5	71	167	82	98	0
98.4	89	118	76	98	0
97.6	85	124	95	98	0
98.2	80	156	92	98	0
98.6	97	100	70	100	3
98.2	177	122	84	97	3
98.6	81	105	65	100	1
98	75	130	86	98	0
98.2	106	121	91	94	2
97.9	97	124	83	98	1
98.7	90	179	76	100	0
98.6	84	122	79	99	0

Figure 7.4: Dataset

7.2.2 Training Result:

As the target column is directly related to the specific ranges of each features that is why the model has gained maximum performances in training.



(a) (b)

Figure 7.5: Training Results: (a) Confusion Matrix; (b) Classification Report;

And similarly it got highest scores in all of the evaluation metrics such as F1-score, Recall and Precision.

Chapter 8

Conclusions

In this project, an autonomous assistance robot called IHABOT is developed as a prototype for a hospital setting. The robot's main functions, such as COVID-19 diagnosis from radiography images, the patient's physical exercise evaluation, and vital sign analysis, are able to automate tasks that would normally require human intervention. Moreover, a navigation algorithm is also devised that enables IHABOT to navigate autonomously. IHABOT's performance in the constrained laboratory setup proves that it has the ability to assist in hospital environments while lowering the risk of virus contamination during doctor-patient interactions. Again, the results of COVID diagnosis and exercise evaluation show that IHABOT can achieve human-level performance if trained with proper machine learning models. For instance, the patient's exercise evaluation results by the IHABOT and those of the expert therapist are very compatible. However, the prototype's evaluation was restricted to limitations in a lab setting among a small number of user groups. Future research will be interesting when the IHABOT is assessed in a real hospital environment with a broad population of patients. IHABOT's autonomy is also restricted to a few certain tasks. We intend to increase its level of autonomy in the future by utilizing cutting-edge machine learning methods.

Bibliography

[1] S. Zagoruyko and N. Komodakis, "Wide residual networks," CoRR, vol. abs/1605.07146, 2016. [Online]. Available: http://arxiv.org/abs/1605.07146

[2] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," CoRR, vol. abs/1610.02391, 2016. [Online]. Available: http://arxiv.org/abs/1610.02391

[3] J. E. Bardram, "Activity-based computing for medical work in hospitals," ACM Transactions on Computer-Human Interaction (TOCHI), vol. 16, no. 2, pp. 1–36, 2009.

[4] C. Heath and P. Luff, "Documents and professional practice: "bad" organisational reasons for "good" clinical records," in Proceedings of the 1996 ACM conference on Computer supported cooperative work, 1996, pp. 354–363.

[5] G. Symon, K. Long, and J. Ellis, "The coordination of work activities: cooperation and conflict in a hospital context," Computer Supported Cooperative Work (CSCW), vol. 5, no. 1, pp. 1–31, 1996.

[6] A. Zivanovic and B. L. Davies, "A robotic system for blood sampling," IEEE Transactions on Information Technology in Biomedicine, vol. 4, no. 1, pp. 8–14, 2000.

[7] T. L. Chen and C. C. Kemp, "Lead me by the hand: Evaluation of a direct physical interface for nursing assistant robots," in 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI). IEEE, 2010, pp. 367–374.

[8] C. Passerotti and C. A. Peters, "Robotic-assisted laparoscopy applied to reconstructive surgeries in children," World journal of urology, vol. 24, no. 2, pp. 193–197, 2006.

[9] B. Mutlu and J. Forlizzi, "Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction," in 2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI). IEEE, 2008, pp. 287–294.

[10] K. Goris, J. Saldien, and D. Lefeber, "Probo, a testbed for human robot interaction," in 2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI). IEEE, 2009, pp. 253–254.

[11] S. Tripathy, R. Kabir, S. Arafat, and S. K. Saxena, "Futuristic technologies for advanced detection, prevention, and control of covid-19," in Diagnostic Strategies for COVID-19 and other Coronaviruses. Springer, 2020, pp. 161–173.

[12] G. Fragapane, H.-H. Hvolby, F. Sgarbossa, and J. O. Strandhagen, "Autonomous mobile robots in hospital logistics," in IFIP International Conference on Advances in Production Management Systems. Springer, 2020, pp. 672–679.

[13] V. Mishra, "Artificial intelligence: the beginning of a new era in pharmacy profession," Asian Journal of Pharmaceutics (AJP), vol. 12, no. 02, 2018.

[14] Z. H. Khan, A. Siddique, and C. W. Lee, "Robotics utilization for healthcare digitization in global covid-19 management," International journal of environmental research and public health, vol. 17, no. 11, p. 3819, 2020.

[15] J. Wilkinson, "The strong robot with the gentle touch," 2017.

[16] T. S. Dahl and M. N. Kamel Boulos, "Robots in health and social care: A complementary technology to home care and telehealthcare?" Robotics, vol. 3, no. 1, pp. 1–21, 2013.

[17] M. Kyrarini, F. Lygerakis, A. Rajavenkatanarayanan, C. Sevastopoulos, H. R. Nambiappan, K. K. Chaitanya, A. R. Babu, J. Mathew, and F. Makedon, "A survey of robots in healthcare," Technologies, vol. 9, no. 1, p. 8, 2021.

[18] S. K. Das, A. Sahu, and D. O. Popa, "Mobile app for human-interaction with sitter robots," in Smart Biomedical and Physiological Sensor Technology XIV, vol. 10216. SPIE, 2017, p. 85.

[19] S. K. Das, "Adaptive physical human-robot interaction (phri) with a robotic nursing assistant." 2019.

[20] D. Hu and S. Li, "Recognizing object surface materials to adapt robotic disinfection in infrastructure facilities," Computer-Aided Civil and Infrastructure Engineering, 2022.

[21] B. Barshan and H. F. Durrant-Whyte, "Inertial navigation systems for mobile robots," IEEE transactions on robotics and automation, vol. 11, no. 3, pp. 328–342, 1995.

[22] J. Kim, Y. Kim, and S. Kim, "An accurate localization for mobile robot using extended kalman filter and sensor fusion," in 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). IEEE, 2008, pp. 2928–2933.

[23] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," AAAI/IAAI, vol. 1999, no. 343-349, pp. 2–2, 1999.

[24] J.-H. Lim and C.-U. Kang, "Grid-based localization of a mobile robot using sonar sensors," KSME international journal, vol. 16, no. 3, pp. 302–309, 2002.

[25] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit et al., "Fastslam: A factored solution to the simultaneous localization and mapping problem," Aaai/iaai, vol. 593598, 2002.

[26] U. Frese and G. Hirzinger, "Simultaneous localization and mapping-a discussion," in Proceedings of the IJCAI Workshop on Reasoning with Uncertainty in Robotics. Seattle, 2001, pp. 17–26.

[27] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in Proceedings 1999 IEEE International Symposium on Computational

Intelligence in Robotics and Automation. CIRA'99 (Cat. No. 99EX375). IEEE, 1999, pp. 318–325.

[28] K. P. Murphy, "Bayesian map learning in dynamic environments," Advances in neural information processing systems, vol. 12, 1999.

[29] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in Proceedings of the 2005 IEEE international conference on robotics and automation. IEEE, 2005, pp. 2432–2437.

[30] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit et al., "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in IJCAI, vol. 3, 2003, pp. 1151–1156.

[31] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in Autonomous robot vehicles. Springer, 1986, pp. 396–404.

[32] N. R. Sturtevant, V. Bulitko, and Y. Björnsson, "On learning in agent-centered search." in AAMAS, 2010, pp. 333–340.

[33] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "Real-time path planning for humanoid robot navigation," in IJCAI, 2005, pp. 1232–1237.

[34] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," IEEE transactions on Robotics and Automation, vol. 12, no. 4, pp. 566–580, 1996.

[35] S. M. LaValle et al., "Rapidly-exploring random trees: A new tool for path planning," 1998.

[36] Mahran, G. S. Khalaf, M. M. Mehany, Tolba, Asmaa, and Atiaa, "The role of national early warning score in detecting and decreasing cardiorespiratory arrest amongst patients with acute coronary syndrome," American Journal of Nursing Research, vol. 8, no. 3, pp. 406–411, 2020. [Online]. Available: http://pubs.sciepub.com/ajnr/8/3/11

[37] A. F. Oduncu, G. S. Kıyan, and S. Yalçınlı, "Comparison of qsofa, sirs, and news scoring systems for diagnosis, mortality, and morbidity of sepsis in emergency department," The American Journal of Emergency Medicine, vol. 48, pp. 54–59, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0735675721002916

[38] C. A. da Costa, C. F. Pasluosta, B. Eskofier, D. B. da Silva, and R. da Rosa Righi, "Internet of health things: Toward intelligent vital signs monitoring in hospital wards," Artificial Intelligence in Medicine, vol. 89, pp. 61–69, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0933365717301367

[39] S. Cai, G. Li, X. Zhang, S. Huang, H. Zheng, K. Ma, and L. Xie, "Detecting compensatory movements of stroke survivors using pressure distribution data and machine learning algorithms," Journal of neuroengineering and rehabilitation, vol. 16, no. 1, pp. 1–11, 2019.

[40] H.-T. Jung, J. Park, J. Jeong, T. Ryu, Y. Kim, and S. I. Lee, "A wearable monitoring system for at-home stroke rehabilitation exercises: A preliminary study," in BHI. IEEE, 2018, pp. 13–16.

[41] E. Frank, Y. Wang, S. Inglis, G. Holmes, and I. H. Witten, "Using model trees for classification," Machine Learning, vol. 32, no. 1, pp. 63–76, 1998.

[42] M. H. Lee, "An Intelligent Decision Support System for Stroke Rehabilitation Assessment," in The 21st International ACM SIGACCESS Conference on Computers and Accessibility, 2019, pp. 694–696.

[43] M. H. Lee, D. P. Siewiorek, A. Smailagic, A. Bernardino, and S. Bermúdez i Badia, "An exploratory study on techniques for quantitative assessment of stroke rehabilitation exercises," in Proceedings of the 28th ACM UMAP, 2020, pp. 303–307.

[44] ——, "Co-Design and Evaluation of an Intelligent Decision Support System for Stroke Rehabilitation Assessment," Proceedings of the ACM on Human-Computer Interaction, vol. 4, no. CSCW2, pp. 1–27, 2020.

[45] "An Automatic Rehabilitation Assessment System for Hand Function Based on Leap Motion and Ensemble Learning, author=Li, Chenguang and Cheng, Long and Yang, Hongjun and Zou, Yongxiang and Huang, Fubiao," Cybernetics and Systems, pp. 1–23, 2020.

[46] Y. X. Zhi, M. Lukasik, M. H. Li, E. Dolatabadi, R. H. Wang, and B. Taati, "Automatic detection of compensation during robotic stroke rehabilitation therapy," JTEHM, vol. 6, pp. 1–7, 2017.

[47] A. Kaku, A. Parnandi, A. Venkatesan, N. Pandit, H. Schambra, and C. Fernandez-Granda, "Towards data-driven stroke rehabilitation via wearable sensors and deep learning," in Machine Learning for Healthcare Conference. PMLR, 2020, pp. 143–171.

[48] M. Panwar, D. Biswas, H. Bajaj, M. Jöbges, R. Turk, K. Maharatna, and A. Acharyya, "Rehab-net: Deep learning framework for arm movement classification using wearable sensors for stroke rehabilitation," IEEE TBE, vol. 66, no. 11, pp. 3026–3037, 2019.

[49] Z.-A. Zhu, Y.-C. Lu, C.-H. You, and C.-K. Chiang, "Deep learning for sensor-based rehabilitation exercise recognition and evaluation," Sensors, vol. 19, no. 4, p. 887, 2019.

[50] Z. Zhang, Q. Fang, and X. Gu, "Objective assessment of upper-limb mobility for poststroke rehabilitation," IEEE TBE, vol. 63, no. 4, pp. 859–868, 2015.

[51] D. Antón, A. Goni, and A. Illarramendi, "Exercise recognition for Kinect-based telerehabilitation," Methods of Information in Medicine, vol. 54, no. 02, pp. 145–155, 2015.

[52] J. Wang, L. Yu, J. Wang, L. Guo, X. Gu, and Q. Fang, "Automated Fugl-Meyer assessment using SVR model," in IEEE ISBB, 2014, pp. 1–4.

[53] M. H. Lee, D. P. Siewiorek, A. Smailagic, A. Bernardino, and S. B. i. Badia, "Learning to assess the quality of stroke rehabilitation exercises," in Proceedings of the 24th IUI, 2019, pp. 218–228.

[54] Y. Liao, A. Vakanski, and M. Xian, "A deep learning framework for assessing physical rehabilitation exercises," IEEE TNSRE, vol. 28, no. 2, pp. 468–477, 2020.

[55] M. Müller, "Dynamic time warping," Information Retrieval for Music and Motion, pp. 69–84, 2007.

[56] S.-W. Kim, S. Park, and W. W. Chu, "An index-based approach for similarity search supporting time warping in large sequence databases," in Proceedings 17th International Conference on Data Engineering. IEEE, 2001, pp. 607–614.

[57] C.-J. Su, C.-Y. Chiang, and J.-Y. Huang, "Kinect-enabled home-based rehabilitation system using Dynamic Time Warping and fuzzy logic," Applied Soft Computing, vol. 22, pp. 652–666, 2014.

[58] A. Yurtman and B. Barshan, "Automated evaluation of physical therapy exercises using multi-template dynamic time warping on wearable sensor signals," CMPB, vol. 117, no. 2, pp. 189–207, 2014.

[59] S. Schez-Sobrino, D. N. Monekosso, P. Remagnino, D. Vallejo, and C. Glez-Morcillo, "Automatic recognition of physical exercises performed by stroke survivors to improve remote rehabilitation," in MAPR. IEEE, 2019, pp. 1–6.

[60] S. Schez-Sobrino, D. Vallejo, D. N. Monekosso, C. Glez-Morcillo, and P. Remagnino, "A distributed gamified system based on automatic assessment of physical exercises to promote remote physical rehabilitation," IEEE Access, vol. 8, pp. 91 424–91 434, 2020.

[61] S. Deb, M. F. Islam, S. Rahman, and S. Rahman, "Graph convolutional networks for assessment of physical rehabilitation exercises," IEEE TNSRE, vol. 30, pp. 410–419, 2022.

[62] B. Bruno, F. Mastrogiovanni, A. Sgorbissa, T. Vernazza, and R. Zaccaria, "Analysis of human behavior recognition algorithms based on acceleration data," in ICRA. IEEE, 2013, pp. 1602–1607.

[63] A. Vakanski, H.-p. Jun, D. Paul, and R. Baker, "A data set of human body movements for physical rehabilitation exercises," Data, vol. 3, no. 1, p. 2, 2018.

[64] M. Capecci, M. G. Ceravolo, F. Ferracuti, S. Iarlori, A. Monteriù, L. Romeo, and F. Verdini, "The KIMORE Dataset: KInematic Assessment of MOvement and Clinical Scores for Remote Monitoring of Physical REhabilitation," IEEE TNSRE, vol. 27, no. 7, pp. 1436–1448, July 2019.

[65] T. M. Mitchell, Machine Learning. New York: McGraw-Hill, 1997.

[66] Y. Wang and Q. Yao, "Few-shot learning: A survey," CoRR, vol. abs/1904.05046, 2019. [Online]. Available: http://arxiv.org/abs/1904.05046

[67] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," arXiv preprint arXiv:1703.03400, 2017.

[68] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra et al., "Matching networks for one shot learning," in Advances in neural information processing systems, 2016, pp. 3630–3638.

[69] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in Advances in neural information processing systems, 2017, pp. 4077–4087.

[70] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[71] L. Bertinetto, J. F. Henriques, P. H. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," arXiv preprint arXiv:1805.08136, 2018.

[72] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in International conference on machine learning, 2016, pp. 1842–1850.

[73] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan, "Medical image analysis using convolutional neural networks: a review," Journal of medical systems, vol. 42, no. 11, p. 226, 2018.

[74] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," Annual Review of Biomedical Engineering, vol. 19, no. 1,

pp. 221–248, 2017, pMID: 28301734. [Online]. Available: https://doi.org/10.1146/annurev-bioeng-071516-044442

[75] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep learning applications in medical image analysis," Ieee Access, vol. 6, pp. 9375–9389, 2017.

[76] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," Medical image analysis, vol. 42, pp. 60–88, 2017.

[77] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," Neural Information Processing Systems, vol. 25, 01 2012.

[78] D. Rajan, J. J. Thiagarajan, A. Karargyris, and S. Kashyap, "Self-training with improved regularization for few-shot chest x-ray classification," arXiv preprint arXiv:2005.02231, 2020.

[79] M. E. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. Al-Emadi et al., "Can ai help in screening viral and covid-19 pneumonia?" arXiv preprint arXiv:2003.13145, 2020.

[80] T. Rahman, "Covid-19 radiography dataset," March 2020, accessed 2 June 2020. [Online]. Available: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

[81] M. Farooq and A. Hafeez, "Covid-resnet: A deep learning framework for screening of covid19 from radiographs," arXiv preprint arXiv:2003.14395, 2020.

[82] L. Wang and A. Wong, "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest radiography images," 2020. [Online]. Available: https://github.com/lindawangg/COVID-Net

[83] L. Wang, "Covidx dataset," https://github.com/lindawangg/COVID-Net/blob/master/docs/COVIDx.md, 2020, online; last accessed July-13,2020.

[84] I. D. Apostolopoulos and T. A. Mpesiana, "Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks," Physical and Engineering Sciences in Medicine, p. 1, 2020.

[85] I. D. Apostolopoulos, S. I. Aznaouridis, and M. A. Tzani, "Extracting possibly representative covid-19 biomarkers from x-ray images with deep learning approach and image data related to pulmonary diseases," Journal of Medical and Biological Engineering, p. 1, 2020.

[86] N. S. Punn and S. Agarwal, "Automated diagnosis of covid-19 with limited posteroanterior chest x-ray images using fine-tuned deep neural networks," arXiv preprint arXiv:2004.11676, 2020.

[87] P. Afshar, S. Heidarian, F. Naderkhani, A. Oikonomou, K. Plataniotis, and A. Mohammadi, "Covid-caps: A capsule network-based framework for identification of covid-19 cases from x-ray images," arXiv preprint arXiv:2004.02696, 04 2020.

[88] A. Abbas, M. Abdelsamea, and M. Gaber, "Classification of covid-19 in chest x-ray images using detrac deep convolutional neural network," arXiv preprint arXiv:2003.13815, 03 2020.

[89] S. Wang, B. Kang, J. Ma, X. Zeng, M. Xiao, J. Guo, M. Cai, J. Yang, Y. Li, X. Meng, and B. Xu, "A deep learning algorithm using ct images to screen for corona virus disease (covid-19)," MedRxiv, 02 2020.

[90] F. Uçar and D. Korkmaz, "Covidiagnosis-net: Deep bayes-squeezenet based diagnosis of the coronavirus disease 2019 (covid-19) from x-ray images," Medical Hypotheses, vol. 140, p. 109761, 04 2020.

[91] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. Wang, and J.-B. Huang, "A closer look at few-shot classification," in International Conference on Learning Representations, 2019.

[92] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1199–1208.

[93] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009, pp. 248–255.

[94] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[95] C. Finn, K. Xu, and S. Levine, "Probabilistic model-agnostic meta-learning," in Advances in Neural Information Processing Systems, 2018, pp. 9516–9527.

[96] Y. Bengio, S. Bengio, and J. Cloutier, Learning a synaptic learning rule. Citeseer, 1990.

[97] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in ICML deep learning workshop, vol. 2. Lille, 2015.

[98] V. G. Satorras and J. Bruna, "Few-shot learning with graph neural networks," ArXiv, vol. abs/1711.04043, 2018.

[99] J. Schmidhuber, "Learning to control fast-weight memories: An alternative to dynamic recurrent networks," Neural Computation, vol. 4, no. 1, pp. 131–139, 1992.

[100] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in Proceedings of the annual meeting of the cognitive science society, vol. 33, no. 33, 2011.

[101] J. P. Cohen, "Covid-19 image data collection," https://github.com/ieee8023/covid-chestxray-dataset, March 2020, online; last accessed July-13,2020.

[102] D. Kermany, K. Zhang, and M. Goldbaum, "Labeled optical coherence tomography (oct) and chest x-ray images for classification," Mendeley data, vol. 2, 2018.

[103] SIRM, "Covid-19 database," 2020, accessed 3 June 2020. [Online]. Available: https://www.sirm.org/category/senza-categoria/covid-19/

[104] RSNA, "Rsna pneumonia detection challenge," 2018, accessed 7 June 2020. [Online]. Available: https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/

[105] S. Q. Ali Mohammad Alqudah, "Augmented covid-19 x-ray images dataset," March 2020, accessed 3 June 2020. [Online]. Available: http://dx.doi.org/10.17632/2fxz4px6d8.4

[106] Radiopaedia, 2020, accessed 7 June 2020. [Online]. Available: https://radiopaedia.org/

[107] Threadreader, "Chest xray," 2020, accessed 7 June, 2020. [Online]. Available: https://threadreaderapp.com/thread/1243928581983670272.html

[108] P. Mooney, "Chest x-ray images (pneumonia)," Mar 2018, accessed 2 June 2020. [Online]. Available: https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

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

[110] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya et al., "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," arXiv preprint arXiv:1711.05225, 2017.

[111] L. van der Maaten and G. Hinton, "Viualizing data using t-sne," Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 11 2008.

[112] M. Capecci, M. G. Ceravolo, F. Ferracuti, S. Iarlori, A. Monteriù, L. Romeo, and F. Verdini, "The KIMORE dataset: KInematic assessment of MOvement and clinical scores for remote monitoring of physical Rehabilitation," IEEE TNSRE, vol. 27, no. 7, pp. 1436–1448, 2019.

[113] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device Real-time Body Pose tracking," CVPR Workshop on Computer Vision for Augmented and Virtual Reality, 2020.

[114] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, "3d human pose estimation in video with temporal convolutions and semi-supervised training," in IEEE CVPR, 2019, pp. 7753–7762.

[115] M. Capecci, M. G. Ceravolo, F. Ferracuti, S. Iarlori, S. Longhi, L. Romeo, S. N. Russi, and F. Verdini, "Accuracy evaluation of the kinect v2 sensor during dynamic movements in a rehabilitation scenario," in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2016, pp. 5409–5412.

[116] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "A study of vicon system positioning performance," Sensors, vol. 17, no. 7, p. 1591, 2017.

[117] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman et al., "Efficient human pose estimation from single depth images," IEEE TPAMI, vol. 35, no. 12, pp. 2821–2840, 2012.

[118] R. Khirodkar, V. Chari, A. Agrawal, and A. Tyagi, "Multi-instance pose networks: Rethinking top-down pose estimation," in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2021, pp. 3122–3131.

[119] M. A. Fisch and R. Clark, "Orientation keypoints for 6d human pose estimation," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.

[120] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv:1609.02907, 2016.

[121] Z. Zhang, Q. Fang, L. Wang, and P. Barrett, "Template matching based motion classification for unsupervised post-stroke rehabilitation," in IEEE International Symposium on Bioelectronics and Bioinformatics (ISBB), 2011, pp. 199–202.

[122] I. Ar and Y. S. Akgul, "A computerized recognition system for the home-based physiotherapy exercises using an rgbd camera," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 22, no. 6, pp. 1160–1171, 2014.

[123] J. Cavazza and V. Murino, "Active regression with adaptive huber loss," arXiv preprint arXiv:1606.01568, 2016.

[124] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, "Richly activated graph convolutional network for robust skeleton-based action recognition," IEEE TCSVT, vol. 31, no. 5, pp. 1915–1925, 2020.

[125] J. Wang, S. Jin, W. Liu, W. Liu, C. Qian, and P. Luo, "When human pose estimation meets robustness: Adversarial algorithms and benchmarks," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 11 855–11 864.

[126] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 3634–3640, 2018.

[127] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in AAAI, vol. 32, no. 1, 2018.

[128] P. Zhang, C. Lan, W. Zeng, J. Xing, J. Xue, and N. Zheng, "Semantics-guided neural networks for efficient skeleton-based human action recognition," in IEEE CVPR, 2020, pp. 1112–1121.

[129] C. Li, Q. Zhong, D. Xie, and S. Pu, "Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation," IJCAI, pp. 786—792, 2018.

[130] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3D human activity analysis," in IEEE CVPR, 2016, pp. 1010–1019.

[131] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in CVPR, 2015, pp. 1110–1118.

[132] L. Van Der Maaten, "Accelerating t-sne using tree-based algorithms," The Journal of Machine Learning Research, vol. 15, no. 1, pp. 3221–3245, 2014.

[133] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in European conference on computer vision. Springer, 2014, pp. 740–755.

[134] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," IEEE transactions on pattern analysis and machine intelligence, vol. 36, no. 7, pp. 1325–1339, 2013.

Appendix A: List of Acronyms

LSTM	Long short term memory
SLAM	Simultaneous localization and mapping
LIDAR	Light Detection And Ranging
CNN	Convolutional Neural Network
SSD	Single Shot Multibox Detector
OCR	Optical Character Recognition
RNN	Recurrent Neural Network
SMS	Short Message Service
PWM	Pulse Width Modulation
GPU	Graphics Processing Units
API	Application Programming Interface
REST	Representational State Transfer
GUI	Graphical User Interface
PTRC	Physical Therapy and Rehabilitation Center