# [[Trust-Aware Multi-View Learning for Hand Pose Estimation with Uncertainty]]


OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Adeep Sri Narayana
School of Computer Science
University of Galway

*Supervisor(s)*
Dr. Matthias Nickles

In partial fulfillment of the requirements for the degree of
*MSc in Computer Science (Artificial Intelligence)*

**DECLARATION** I, Adeep Sri Narayana, hereby declare that this thesis, titled "Trust-Aware Multi-View Learning for Hand Pose Estimation with Uncertainty ", and the work presented in it are entirely my own except where explicitly stated otherwise in the text, and that this work has not been previously submitted, in part or whole, to any university or institution for any degree, diploma, or other qualification.

Signature: 

# Abstract

Multi-view learning [1] has become a pivotal technique in machine learning [2], offering significant improvements in handling high dimensional and complex datasets by integrating information from multiple perspectives. By leveraging multiple view points, multi-view learning enhances model robustness and accuracy. The study is focused on a Large-scale Multiview 3D Hand Pose (MHP) Dataset [3] which provides a rich collection of hand poses from various angles. Such comprehensive datasets facilitate the development of sophisticated models capable of precise and real-time pose estimation.

However, trustworthiness, reliability, and uncertainty quantification of these models are critical, especially where precise hand pose estimation [4] is vital. To address these concerns, incorporating both trustability and estimation factor into multi-view learning frameworks is essential. This approach ensures model not only be accurate but also reliable, interpretable, and robustly flexible to different sources of uncertainty efficiently.

To build such a system, I propose to use a training strategy, where I combine the strengths of multiple models and dynamically adjust their contributions based on quality assessment factors such as image clarity, noise levels, occlusion [5], and more.

**Keywords:** Machine Learning, Multi-view Learning, Hand Pose Estimation, Occlusion

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

As the field of Artificial Intelligence [6] keeps on growing rapidly, there has been a significant impact on how we analyse complex and large datasets, particularly through Machine learning [2] and deep learning [7] techniques such as neural networks [8]. A key technique within these fields is multi-view learning [9], which enhances model robustness, accuracy and generalisability by integrating information from multiple perspectives. This approach is especially valuable in applications like computer vision [10], human-computer interaction, and augmented reality [11], where precise data analysis [12] is very crucial.

One such important application of multi-view learning [9] is hand pose estimation (HandPE) [4], which is essential in fields like sign language interpretation, virtual reality, and human-robot interaction. The process requires predicting the positions of various hand joints from multiple camera views. The Large-scale Multiview 3D Hand Pose Dataset [3] supports this approach by providing a comprehensive collection of hand pose data, including colour images, 2D joint projections, and 3D coordinates.

## 1.1 Motivation

While multi-view learning offers substantial benefits with respect to improved accuracy, ensuring the trustworthiness and reliability of these models remains a major challenge. In real-world applications, it is imperative that the models not only deliver accurate predictions but also provide insights into their confidence levels and be resilient to variations and noise in the data. This necessitates the development of frameworks that incorporate trustability [13] into multi-view learning.

The term "**trust**" can have multiple meanings depending on the context. In this approach, trust is quantified through various parameters (such as occlusion, noise, image clarity) that can contribute for confidence in the system to predict hand pose efficiently. Such parameters need to be combined into a trust quantifier that influences the training process, guiding the model to focus more on predictions it is less confident about. This trust-aware approach ensures that the model is not only accurate but also reliable and transparent, making it suitable for applications where uncertainty and data quality are significant concerns.

## 1.2 Objective and Research Questions

The main objective of this research is to develop a trust-aware multi-view learning framework that integrates modelling strategies which is good at efficiency and accuracy. By combining the strengths from different models, this framework aims to enhance multi-view hand pose estimation (HandPE) by quantifying and incorporating uncertainty into the model's outputs, thereby providing a measure of trustworthiness in the predictions. Additionally, the research seeks to enhance the model's robustness and reliability over handling noisy, ambiguous, or uncertain input data. A secondary goal is to ensure that the model remains accu-

rate and computationally efficient, making it suitable for deployment in resource-constrained environments so as to improve the applicability of HandPE models in real-world scenarios, particularly in environments where trust, reliability, and efficiency are critical.

- **RQ1:** Does incorporating trust-aware mechanisms improve the accuracy of multi-view hand pose estimation compared to state-of-the-art (SOTA) methods?
- **RQ2:** How does the use of trust-aware mechanisms affect the robustness of hand pose estimation models under varying levels of data quality and noise?
- **RQ3:** Can transfer learning enhance the performance of trust-aware multi-view hand pose estimation models compared to SOTA models?

# Chapter 2

# Background

Hand pose estimation [4] [3] is a crucial component in the field of computer vision, with remarkable implications for real-world usability in applications such as augmentedand virtual reality, sign language interpretation, and interactive systems. The goal is to determine the spatial positions of hand joints from images or videos, which can be particularly challenging due to the variable degree of freedom of hand movements, occlusions, and variability in hand shapes across different individuals.

## 2.1 Early Methods

### 2.1.1 Geometric and Kinematic Models

The methods discussed in Vision-based hand pose estimation [14] relied on geometric and kinematic models which were fundamentally rule-based systems. These early methods were particularly concerned with modeling the physical constraints and movements of hand joints, aiming to reconstruct the hand's posture from visual data .

#### 2.1.1.1 Geometric Models

Geometric models [14] attempted to reconstruct the hand's structure using geometric shapes like cylinders or spheres to represent the fingers and palms. By matching these geometric shapes to the image data, the system tried to infer the position and orientation of each part of the hand.

#### 2.1.1.2 Kinematic Chain Models

These leveraged the concept of kinematic chains [14], which are used in robotics to describe the interconnected, jointed structures. Applying this to HandPE involved creating a hierarchical model of joint dependencies within the hand, where the movement of one joint would necessarily affect the position of connected joints.

#### 2.1.1.3 Limitations

While these models were pioneering, they had significant limitations. They often required a perfectly segmented image (where the hand is cleanly separated from the background), and they struggled [14] with the complexities of real-world hand movements, such as self-occlusion (where fingers block each other), varying lighting conditions , and complex dynamic gestures.

## 2.2 Machine Learning Techniques

With the availability of more data and the new advancements in computational powerful machines, researchers began to explore statistical machine learning techniques [15] to improve hand pose estimation. These methods marked a significant shift from purely rule-based systems to data-driven approaches .

### 2.2.1 Feature-based Approaches

Early machine learning techniques for hand pose estimation often involved feature extraction [16] processes where features like edges, contours, or skin colour histograms were manually designed and extracted from the images. These features were then used as inputs to train classifiers or regressors.

### 2.2.2 Decision Trees and SVMs

ML Techniques like Decision Trees [17], Random Forests [18], and Support Vector Machines (SVMs) [19] were employed to classify the hands position based on the extracted features. These models offered more flexibility and better performance than the rigid geometric models, particularly in more controlled environments.

### 2.2.3 Limitations and Challenges

Despite their advantages, these methods still faced challenges. The performance [2] heavily depended on the quality of the feature extraction stage—poor features could significantly degrade the model's accuracy. Moreover, these methods typically required a substantial amount of labeled training data, which were difficult and expensive to collect for hand poses.

### 2.2.4 Transition to Hybrid Approaches

To overcome these limitations, hybrid approaches began to emerge, that combined traditional models with statistical learning elements. For example, integrating kinematic constraints [14] within a learning framework [15] helped improve the plausibility of the predicted hand poses.

## 2.3 Depth Sensors

[20] marked a pivotal shift in hand pose estimation. These sensors, such as the Microsoft Kinect [21], provided a depth map of the captured scene, which added a crucial third dimension to the visual data previously limited to 2D images. [22] shows how depth sensors have enhanced HandPE.

### 2.3.1 Depth Perception

Depth sensors [20] measure the distance of objects from the sensor, providing a direct insight into the spatial arrangement of the scene. This capability is particularly beneficial for distinguishing between the hand and the background, as well as accurately capturing the complex movements of fingers, which are often occluded or closely positioned.

### 2.3.2 Robust to Lighting Variations

Traditional cameras rely on visible light to capture images, making them highly sensitive to changes in lighting conditions. Bright lights can cause glare, while low light can make it difficult to see details. Depth sensors [20], on the other hand, typically use infrared light to measure distances, which allows them to function effectively regardless of the lighting in the environment. This makes depth sensors especially useful in varied lighting conditions, such as in dark rooms or outdoors under bright sunlight. This robustness ensures consistent performance across different settings, which is a significant advantage for applications like gesture recognition or object detection.

### 2.3.3 Real-time Tracking

Depth sensors enable real-time tracking of hands and fingers, an essential feature for interactive applications like gesture-based [23] control systems where latency needs to be minimal. The ability to capture and process depth information quickly ensures that the system can keep up with the fast and often complex movements of hands and fingers.

### 2.3.4 Accessibility and Consumer Use

Devices like Kinect[21] made depth-sensing technology accessible to the general public, motivating a wide range of consumer applications and driving further research and development in HandPE techniques.
Initial methods to hand pose estimation relied heavily on depth data, utilising techniques such as decision forests [24] or part-based models. These methods, while effective with depth sensors, do not perform well in scenarios only involving RGB cameras due to the lack of 3D data. With the advancement of deep learning [7], Convolutional Neural Networks (CNNs) [25] have become the backbone of most modern hand pose estimation systems, offering substantial improvements over traditional machine learning approaches.

## 2.4 Deep Learning and Convolutional Neural Networks

The adoption of deep learning, particularly CNNs[25] [16], has revolutionised many areas of computer vision, including hand pose estimation. CNNs are powerful tools for handling image data, learning hierarchical representations [26] that are effective for complex tasks like pose estimation.

### 2.4.1 Feature Extraction

CNNs automatically identify and learn the most relevant features from raw image data, unlike older approaches that relied manual feature design. This capability allows CNNs to focus on the most informative aspects of an image for HandPE [16], such as edges, textures, and specific patterns indicative of different hand parts.

### 2.4.2 Handling Variability

CNNs excel at managing variations within the data, such as different hand sizes, shapes, and complex gestures, because of to their ability to learn spatial features in hierarchical manner, allowing them to adapt to different hand configurations and movements. By capturing the spatial relationships between various parts of the hand, CNNs can robustly interpret a wide range of hand poses, even when faced with significant variability in the input data.

### 2.4.3 Data-Driven Learning

With sufficient training data [27], CNNs can achieve impressive accuracy by learning directly from the pixel values of images, adapting their parameters through backpropagation based on the loss between the predicted and actual hand poses, enabling CNNs to become increasingly precise in recognising and interpreting hand gestures over time.

### 2.4.4 End-to-End Learning

CNNs can be trained in an end-to-end manner, allowing them to take raw images as input and directly output the hand pose, simplifying the training pipeline and improving the efficiency of the learning process.

### 2.4.5 Transfer Learning

These include pre-trained model which has trained to large dataset in ImageNet [28]. Various model like AlexNet [29], VGGNet [30], GoogleNet [31] trained to these dataset are fine to for various hand pose estimation task. This approach is beneficial because it allows the model to start with a solid foundation, significantly reducing the time and computational resources needed to train the network from scratch. Additionally, transfer learning enhances the performance of the CNN, particularly when the available data for hand pose estimation is limited. There are a many model similar to these.

The transition to using CNNs in HandPE not only improved the accuracy and robustness of the estimations but also opened up new possibilities for real-time applications and complex scenarios that were previously challenging to handle with traditional machine learning techniques. The ongoing advancements in CNN architectures continue to push the boundaries in the space of HandPE.

## 2.5 Multi-view and Self-supervised Learning

Expanding on CNNs [25], the incorporation of multi-view data collection and self-supervised learning [32] methods offers further advancements:

### 2.5.1 Multi-view Data Collection

By capturing the hand from multiple cameras at different angles, the system can gather a more comprehensive set of data points for each pose. This helped [33] in dealing with issues like occlusions (where parts of the hand block each other) and varying lighting conditions, which typically challenge single-view systems.

### 2.5.2 Self-supervised Learning Framework

In cases where labeled data is insufficient or costly to obtain, self-supervised learning [34] strategies come into play. These methods involve creating pseudo labels from the exisiting data itself—often through a preliminary unsupervised process [35] and then using these labels to train the model. For instance, one might use the consistency of hand poses across multiple views as a training signal, encouraging the model to predict poses that are geometrically coherent when projected into different camera perspectives.

### 2.5.3 Leveraging Unlabeled Data

The approach [32] significantly reduces the dependency on large labeled datasets. By using unlabeled multi-view data, the model can improve its accuracy and robustness through exposure to a wider variety of hand poses and configurations, learning to reconcile discrepancies between its predictions and the pseudo labels.

### 2.5.4 Synthesis and Consistency

Multi-view setups [36] enable the synthesis of more reliable training labels and help enforce consistency across different views in the training data. This not only improves the model's performance but also its ability to hypothesise data to real-world applications.

While these techniques represent a move towards more autonomous, robust, and scalable systems, they do not have represent correct methods to estimate uncertainty which is important to make reliable systems.

## 2.6 Uncertainty Estimations

Uncertainty estimation is crucial for developing a reliable HandPM systems. There are various methods that can be used for such task:

- **Ensemble methods** [37] train multiple models with different subsets of the data. The different predictions made by these models provides a variance which provides an estimate of uncertainty.
- **Bayesian Neural Networks** [38] incorporate uncertainty directly into the model by treating the network weights as distribution than be fixed values. Variational Inference and other techniques can be used to make this computationally feasible.
- **Monte Carlo Dropout** [39] involves drop out approximations during training which help a model to generate distribution of its prediction.
- **Custom Loss Functions** [40] can be made which can directly estimation change and penalise less certain predictions.

## 2.7 Evaluation strategies

When evaluating HandPE models, especially in predicting keypoints, several important metrics are used to check how well the model performs. Root Mean Squared Error (RMSE) [41] is a metric that tells how close the predicted hand keypoints are to the actual positions by calculating the square root of the average squared differences. It highlights bigger mistakes, making it super useful for assessing overall accuracy. [42] used RMSE to see how well their model predicted 3D hand poses from just RGB images. Mean Absolute Error (MAE) looks at the average error without worrying about whether we're predicting too high or

too low, it checks how far off we are on average, [43] gives a clear picture of the average error in predictions. Mean Squared Error (MSE), which is related to RMSE, calculates the average of the squared differences, focusing even more on larger errors. [44] used MSE to ensure their predictions were consistent. Another important metric is Percentage of Correct Keypoints (PCK), which checks how many predicted keypoints are within a certain distance from the actual ones, making it perfect when we need to know if our predictions are generally in the right area. [45] used PCK to evaluate real-time hand pose predictions from images of hand in wild. Mean Joint Error (MJE) [46] measures the average distance between the predicted and actual joint positions, giving a direct look at how accurate the predictions are for each hand joint. All these metrics together ensuring that the predictions are accurate and reliable in various situations.

# Chapter 3

# Related Work

As hand pose estimation [4] has been a critical area of research and innovation, especially with the increasing demand for user interfaces, notably in the branch of virtual reality (VR) [47] and augmented reality (AR). As these technologies advances, the ability to accurately and efficiently tract and interpret hand movements is becoming increasingly essential for better user experience. This chapter focuses on the use of lightweight and other Convolutional Neural Networks (CNN) [25] models have proven to be highly effective in capturing the intricate details of hand movements while maintaining the computational efficiency required for real-time applications.

Moreover, the chapter explores the integration of incorporating uncertainty measures with trust metrics, making these systems more reliable, robust [48] even in scenarios with ambiguous or noisy input data.

## 3.1 Lightweight Convolution Neural Networks

Lightweight Convolutional Neural Networks (CNNs) [25] [49] have been developed in response to the growing demand for efficient and high-performance models, particularly in contexts with limited computational resources, such as mobile [50] and embedded devices. These models are meant to strike a delicate balance between minimising computing complexity and retaining high accuracy, making them appropriate for tasks such as hand posture estimation that require real-time performance.

The primary advantage of lightweight CNNs lies in their ability to perform well without requiring extensive hardware resources. By optimising the architecture to minimise the number of parameters and operations, these models can run efficiently on devices with limited processing power. This approach is highly crucial with respect to hand pose estimation, allowing for real-time processing by enabling the system to quickly and accurately track hand movements without lag.

### 3.1.1 Different Lightweight CNN models

#### 3.1.1.1 MobileNetV2

MobileNetV2 [51] is a standout example in this category and has been widely recognised for its effectiveness in balancing efficiency and performance. The model uses depthwise separable convolutions and inverted residuals, which significantly reduce the computational load while maintaining a high level of accuracy. These features make MobileNetV2 [52] particularly suitable for hand pose estimation, where the demands for both quick processing and precise outputs are high. The

model's ability to perform well on devices with limited resources has led to its adoption in numerous applications beyond hand pose estimation, further underscoring its versatility. However, MobileNetV2 is just one of several lightweight models designed with similar goals in mind.

#### 3.1.1.2 ShuffleNet

ShuffleNet [53] introduces the concepts of pointwise group convolutions and channel shuffling. By processing channels in groups and then shuffling them, ShuffleNet reduces the number of computations required while still ensuring that information is thoroughly mixed across the network. This architecture excels in scenarios where computational efficiency is a priority, making it suitable for hand pose estimation, where it can deliver fast and accurate results.

#### 3.1.1.3 GhostNet

GhostNet [54] takes a unique method, concentrating on decreasing redundancy in feature maps. Instead of utilising expensive convolution procedures to generate all feature maps, GhostNet first constructs a small collection of intrinsic feature maps before producing additional "ghost" feature maps using simple linear operations. This approach dramatically reduces the amount of parameters and computations, allowing GhostNet [55]to run effectively even on low-power devices. Despite its lightweight design, GhostNet maintains competitive accuracy, making it a viable model for handPE applications.

In summary, lightweight CNNs like MobileNetV2, ShuffleNet, and GhostNet bring a lot to the table when it comes to hand pose estimation, especially in set-

tings where resources are limited. These models show that you can get real-time performance and high accuracy without needing bulky, power-hungry architectures.

## 3.2 Uncertainty in Hand Pose Estimation

Uncertainty plays a vital role in HandPE, however the input data can often be imperfect such as blurry, noisy, or partially occluded images. These imperfections introduce uncertainty into the model's predictions. To estimate these uncertainty the model should be able to measure its predictions. [56] introduces an approach that combines Bayesian modeling with Active Learning to improve HandPE. By adapting the DeepPrior architecture [57] into a Bayesian Neural Network (BNN) [39] brings a probabilistic approach to the networks. BNNs have different types of priors that are used to guide the model's learning process based on varying assumptions.

- **Gaussian priors** [57] assume that the model's parameters follow a normal distribution, typically centered around zero, which helps to prevent overfitting, especially when the data is noisy or sparse.
- **Laplace priors** [58] encourage sparsity by making most parameters zero, focusing the model's attention on only the most significant features or joints. This is particularly useful in hand pose estimation where only a few key aspects of the pose might be important. [58] shows how a sparse model can still capture the essential details of hand movement.
- **Uniform priors** are used when there is little prior knowledge about the parameter values, assuming all possibilities within a range are equally likely.

This type of prior is useful in the early stages of training or when the data is very diverse, allowing the model to learn without any preconceptions. [59] demonstrated the effectiveness of this approach in their work on latent 2.5D heatmap regression, where the model was able to adapt to a wide range of hand poses without being biased by strong prior assumptions.

## 3.3 Trustworthiness

Trust can have many different meaning with respect to the task and context it is applied. Trust [60] has very broad perspective especially within the realm of Artificial Intelligence.
In multiview systems, where multiple cameras or sensors are used to capture and analyse a scene from different angles, trust becomes a multifaceted concept. Trust in these systems is not just about the accuracy of individual views but also about how these views interact and complement each other to provide a coherent and reliable understanding of the scene.

### 3.3.1 Trusted Multi-View Learning

The work reported [13] enhances the models robustness on multiple views by dynamically assessing the quality of each view for different samples. The proposed method integrates multiple views at an evidence level using the Dirichlet distribution [13], combined with the Dempster-Shafer theory [61] to model class probabilities and provide accurate uncertainty estimations. The implementation involves a unified learning framework that integrates multi-view information at an evidence level, promoting both classification reliability and robustness. This

approach ensures trusted decision-making by adapting to the varying quality of views, essential for safety-critical applications like medical diagnosis and autonomous driving. Extensive experiments validated the model's superior accuracy, reliability, and robustness, significantly contributing to the field of multi-view classification.

### 3.3.2 Trust with Opinion Aggregation

[62] introduces a method for trusted multi-view learning by aggregating opinions from multiple data sources . This approach uses evidence theory to represent the uncertainty of opinions and measures the consistency across different views. The method integrates opinions at an evidence level, reducing overall uncertainty and improving the reliability of multi-view learning results. The implementation provides non-negative outputs, which are shown as evidence, by substituting an activation layer for the conventional softmax layer. The opinions are then aggregated using evidence accumulation, which increases the reliability of the learning results. The framework is validated through extensive experiments, demonstrating its effectiveness in accuracy, reliability, and robustness across various multi-view datasets.

### 3.3.3 Trust-Aware Estimation

In context to our research with respect to HandPE, trust will be measured on accurate model performance in scenarios where conditions are not perfect such blurry images, when there's a lot of noise or when part of the hand is hidden. This is where trust metrics will help us understand and improve the reliability of

the model under various conditions.

#### 3.3.3.1 Image Clarity

The clarity of an image directly affects how well the model can detect and estimate hand poses. When an image is clear, the model can make accurate predictions. However, if the image is blurry—maybe due to motion or low resolution—the model's performance can suffer. The work [63] focuses on creating models that are aware of image clarity. These models adjust their confidence based on the quality of the input, meaning they might give a less certain prediction if the image isn't clear enough.

#### 3.3.3.2 Noise Estimation

Noise refers to random variations in the image data, which can come from environmental factors or limitations of the camera. Noise can introduce errors in hand pose estimation. Bayesian approaches [64] are particularly effective at dealing with noise because they don't just give a single prediction—they provide a range of possible predictions along with a measure of confidence. This makes the model's output more reliable, even when the input image is noisy.

#### 3.3.3.3 Occlusion Detection

Occlusion [5] occurs when parts of the hand are hidden from the camera's view, making it harder for the model to estimate the pose accurately. Handling occlusions is a significant challenge in HandPE. Recent studies have explored different ways to detect and manage occlusions. Probabilistic models [39] are also used to

account for these missing parts, helping to improve the overall accuracy.

While significant progress have been made in each of these areas, there lies a **research gap** in integrating a lightweight models like MobileNetV2 [51] with Bayesian Neural Networks (BNNs) [39] for hand pose estimation. While MobileNetV2 is known for its speed and efficiency, it lacks robust uncertainty estimation, which is crucial for applications requiring high reliability. BNNs, on the other hand, provide uncertainty metrics but are computationally expensive. Combining these approaches can develop a model that balances efficiency with robust uncertainty estimation.

# Chapter 4

# Data

## 4.1 Dataset Overview

[3] introduces a comprehensive dataset designed for training and evaluating HandPE models. This dataset includes colour images of hands captured from multiple viewpoints, along with detailed annotations of hand joints in both 2D and 3D. The Dataset [3] provides a comprehensive collection of hand pose data captured from multiple perspectives, it is specifically designed to support research in hand pose estimation. The dataset is accessible through the following link: `https://www.rovit.ua.es/dataset/mhpdataset/`

### 4.1.1 Data Composition

- **Images** in the dataset are stored in '.jpg' file format.
- **Total Size:** The dataset contains 148,000 samples, with each sample rep-

resenting a distinct hand pose captured from multiple angles.

- **Views:** The dataset provides four different camera views for each hand pose, capturing the hand from various angles to ensure comprehensive coverage of the hand's position and orientation, This multi-view setup is crucial for accurate hand pose estimation, allowing models to learn from various perspectives.
- **Annotations:** Each sample includes 42 annotated key-points corresponding to various finger joints and parts of the hand, enabling precise localization of the hand's pose. The positions are annotated in '.txt' file format.
- For storing the camera calibration parameters, '.pkl' [1] file format was used.

### 4.1.2 Dataset Structure and Content

The dataset is structured into multiple sequences, where each sequence consists of a set of frames. Each frame contains the following types of ground truth data:

- **3D Points of Hand Joints:** For each frame, 3D coordinates of hand joints are provided as captured by the Leap Motion Controller.
- **Colour Images:** Each frame includes four colour images of the hand, captured simultaneously from different perspectives using four different cameras. These images are provided at a resolution of 640x480 pixels.
- **2D Joint Projections:** The 3D joint coordinates are projected onto the 2D image plane of each camera, providing corresponding 2D points for each joint in every colour image.

[1] pkl is a file format created by Python's pickle module, which serializes and de-serializes Python objects.

- **Bounding Boxes:** Bounding boxes are generated for each hand image by projecting the 3D points onto the camera coordinate frame and extracting the maximum and minimum values for the X and Y coordinates.

Figure 4.1: This is a figure from [3] showing 3D points of each knuckle in the finger, finger tips, palm position and normal palm.

| Joint ID | Finger | Bone | Joint ID | Finger | Bone |
|---|---|---|---|---|---|
| 1 | Index | Distal | 11 | Pinky | Intermediate |
| 2 | Index | Metacarpal | 12 | Pinky | Proximal |
| 3 | Index | Intermediate | 13 | Ring | Distal |
| 4 | Index | Proximal | 14 | Ring | Metacarpal |
| 5 | Middle | Distal | 15 | Ring | Intermediate |
| 6 | Middle | Metacarpal | 16 | Ring | Proximal |
| 7 | Middle | Intermediate | 17 | Thumb | Distal |
| 8 | Middle | Proximal | 18 | Thumb | Metacarpal |
| 9 | Pinky | Distal | 19 | Thumb | Intermediate |
| 10 | Pinky | Metacarpal | 20 | Thumb | Proximal |

Table 4.1: [3] lists the unique Joint IDs corresponding to various fingers and bones in a hand.

# Chapter 5

# Methodology

## 5.1 Dataset Preparation

The dataset underwent several preprocessing steps to ensure compatibility with the model architecture:

### 5.1.1 Annotation Processing

The first step in processing involves reading and saving joint annotations. The `saveAnnotation` function takes the path to save the joint annotations and the 2D positions of the keypoints and writes these annotations to a text file. The joints are labeled according to a predefined order:

$$\text{Joints} = \{\text{"F4\_KNU1\_A"}, \text{"F4\_KNU1\_B"}, \ldots, \text{"PALM\_POSITION"}\}$$

The positions are saved in a format where each line corresponds to a joint and its 2D coordinates $(x, y)$.

### 5.1.2 Natural Sorting and Recursive File Search

- The `natural_sort` function is used to sort file names containing numeric values in a natural order.
- The `recursive_glob` function is employed to search and list files in directories recursively, after filtering them based on a given pattern.

### 5.1.3 Camera Calibration and 3D to 2D Projection

The intrinsic camera parameters [65] (focal lengths $F_x$, $F_y$ and principal point coordinates $C_x$, $C_y$) are defined as:

$$\text{Camera Matrix} = \begin{bmatrix} F_x & 0 & C_x \\ 0 & F_y & C_y \\ 0 & 0 & 1 \end{bmatrix}$$

Additionally, distortion coefficients [65] are provided to correct lens distortions:

$$\text{Distortion Coefficients} = [k_1, k_2, p_1, p_2, k_3]$$

For each session, the rotation and translation vectors are loaded from the calibration data to project the 3D joint positions into the 2D image plane using

the `cv2.projectPoints` [65] function. The 3D joint positions are transformed into the camera coordinate system, and bounding boxes are computed based on the 2D projections. An offset based on the mean depth value is applied to the bounding box dimensions to accommodate for possible variations in hand position.

### 5.1.4 Keypoint Visualisation

The `renderPose` function is used to overlay the detected keypoints and skeletal connections on the image. The function draws lines between connected joints and colours each joint according to its label. Additionally, bounding boxes are visualised using the `plot_image_with_bboxes` function, which overlays the bounding box on the image for easy inspection.

### 5.1.5 Data Augmentation

To enhance the robustness of the model, data augmentation techniques are applied to the images and keypoints. The `transform` function employs the `albumentations`[1] [66] library to perform a series of transformations:

- **Cropping**: The image is cropped based on the bounding box coordinates.
- **Resizing**: The cropped image is resized to a fixed size (224x224 pixels).
- **Rotation and Shifting**: Random rotations and shifts are applied to simulate different orientations.

[1]To know more details visit: `https://pypi.org/project/albumentations/0.0.10/`

- **Colour Jittering**: Brightness, contrast, and saturation of the image are adjusted randomly.

These transformations help in simulating a wide range of hand poses and lighting conditions, ensuring that the model receives a holistic view of the hand pose from multiple perspectives.

### 5.1.6 Dataset Handling and Splitting

The dataset is encapsulated in a `HandPose` class, which implements the necessary PyTorch Dataset methods [67] for easy loading and transformation of the data. The dataset is split into training and validation subsets using the `IdxGenerator` class, which ensures reproducibility by utilising a fixed random seed. The training set consists of 80% of the data, while the remaining 20% is reserved for validation.

### 5.1.7 Error Handling and Data Cleaning

The data is inspected for errors in keypoints or bounding boxes. The function `error_kpts` checks if any keypoint lies outside the specified bounding box. If such errors are detected, the corresponding entries are removed from the dataset to ensure the integrity of the training data. This step is crucial to avoid introducing noise into the model during training.

### 5.1.8 Model Training and Validation

The dataset is then prepared by creating PyTorch `DataLoader` objects [67] for both training and validation sets. The `DataLoader` shuffles the data and batches it for efficient training of the model. A typical batch size of 128 is used, and the images are normalised before being passed into the model. By thoroughly preprocessing the data, the pipeline prepares the dataset for efficient and effective training, ensuring that the model receives well-structured and meaningful inputs.

## 5.2 Model Architecture:

We have created a hybrid model, where the core model architecture is based on **MobileNetV2** [51] and modified this core model by integrating **Bayesian Neural Networks** (BNNs) [39] for quantifying uncertainties.

### 5.2.1 Model Initialisation

The original MobileNetV2 [51] which includes the depth-wise separable convolutions was retained. We replaced the final fully connected layer with Bayesian linear layers [39], which allow the model a distribution of possible outcomes.

#### 5.2.1.1 Bayesian Layer Implementation:

In our BNN implementation, each weight ( w ) in the Bayesian layers is treated as a distribution rather than a fixed value. The Bayesian layer outputs a probability distribution for each keypoint, reflecting the model's confidence in its predic-

tions. The **Kullback–Leibler (KL) divergence** [68] $D_{\mathrm{KL}}$ is used to measure how much the learned distribution of weights $q(w \mid X)$ diverges from a prior distribution $p(w)$ for the input $X$ represented as:

$$[\boldsymbol{D_{KL}}\left(\boldsymbol{q(w \mid X)} \parallel \boldsymbol{p(w)}\right)]$$

This encourages the model to stay close to the prior, ensuring that predictions remain robust and reliable, even in the presence of noisy or ambiguous data.

### 5.2.2 Trust-Aware Mechanism

The trust-aware mechanism integrates various metrics related to the input data quality, model prediction quality, and consistency across different views or time steps. These metrics include:

- **Image Clarity:** Measured using the variance of the Laplacian [69], this metric assesses how sharp or blurred the image is, which can directly impact prediction accuracy.
- **Noise Estimation:** By calculating the variance of the noise [64] in the image, we can determine how much random variation is present, which may degrade model performance.
- **Occlusion Detection:** This metric identifies whether keypoints are occluded [5] or not visible, which can significantly affect prediction accuracy.
- **Historical Performance:** The model's past performance, measured in terms of Mean Squared Error (MSE) [70] and Mean Absolute Error (MAE) [71], provides insight into how well the model has performed on similar data in the past.

- **Prediction Confidence:** The variance in the output of the Bayesian layer gives an estimate of the model's confidence in its predictions.
- **Cross-View Consistency:** If multiple views of the same object are available, this metric checks the consistency of predictions across different views.
- The **trust score** is computed as a weighted combination of these metrics:

$$\begin{aligned}\textbf{Trust Score} = w_1 \cdot \text{MSE} + w_2 \cdot \text{MAE} + w_3 \cdot \text{PCK} \\ + w_4 \cdot \text{MJE} + w_5 \cdot \text{Confidence} + w_6 \cdot \text{Consistency}\end{aligned}$$

  where each $w_i$ represents the weight assigned to the corresponding metric.

- **Weighted MSE Loss** used during training is a weighted mean squared error (MSE) loss [72], where the weights are derived from the trust scores. This ensures that the model focuses more on predictions with higher trustworthiness. The weighted MSE loss $L$ is calculated as:

$$L = \frac{1}{N} \sum_{i=1}^{N} w_i \cdot (\hat{y}_i - y_i)^2$$

  where $N$ is the total number of data point prediction, $w_i$ is the trust score for the $i$-th prediction, $\hat{y}_i$ is the predicted value, and $y_i$ is the ground truth value.

- Adam Optimizer is chosen with a learning rate of $1 \times 10^{-3}$.

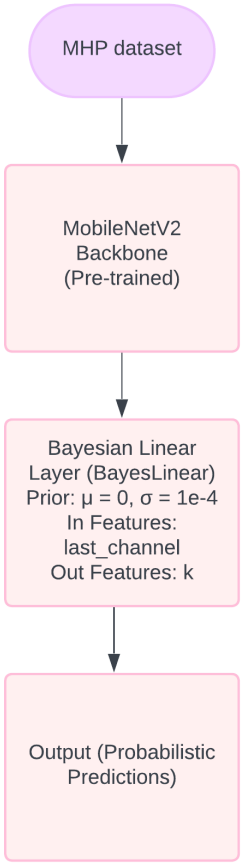


Figure 5.1: Diagram represent probabilistic neural network architecture utilising a pre-trained MobileNetV2 backbone and a Bayesian Linear Layer to process the MHP dataset, producing probabilistic predictions that account for model uncertainty.

### 5.2.3 Metrics for Evaluation

During training and validation, several metrics are logged which include RMSE, MAE, PCK and MJE respectively. Finally producing a **Trust Aware Bayesian MobileNet** (TABM) model[1].

[1]Follow the link to the Github: `https://github.com/Adeepsn/Thesis`

## 5.3 Tools and Technology

Python [73] was the primary language used throughout the research. It is widely utilised for its extensive libraries that favoured in the domains of machine learning, deep learning and neural networks.

### 5.3.1 Deep Learning Framework

- PyTorch [67] is the core deep learning framework utilised for coding the DL pipeline. It allows for flexibility in model building and training. PyTorch's support for GPU (Graphics processing unit) acceleration makes it suitable for training large-scale models like those required for hand pose estimation.
- PyTorch Lightning [67] is a high-level interface for PyTorch,
- TorchBNN [74] is a specialised library within PyTorch for implementing Bayesian Neural Networks [75].

### 5.3.2 Computer Vision and Image Processing

- OpenCV [65] is used extensively for reading, processing, and displaying images.
- Albumentations [66] is a fast image augmentation library that enhances model robustness by applying transformations.

### 5.3.3 Data Handling and Scientific Computing

- NumPy [76] is fundamental for numerical operations in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. Pandas [77] is used for handling structured data in the form of DataFrames.

- PyTorch's DataLoader [67] is used to manage the data pipeline, efficiently loading data in batches during training. This ensures that the model is fed with data consistently without overwhelming the system memory, particularly important when dealing with large datasets like those in this project.

### 5.3.4 Visualisation

- Matplotlib [78] is used for creating visualisations. It is employed to plot images, keypoints, and model predictions, helping in visual debugging and result presentation.

- Seaborn [78] is particularly useful for visualising the distribution of errors and other metrics in the model evaluation phase.

### 5.3.5 Execution Environment

- Google Colab: All the code executions was run on Google Colab Pro, A100 GPU and L4 GPU were used.

# Chapter 6

# Experiments

## 6.1 Experimentation Phase Analysis

### 6.1.1 Standard MobileNet (No BNNs)

In this experiment, the MobileNet model [51] was trained for 10 epochs on the same dataset. The goal was to first establish a baseline model in terms of accuracy and computational efficiency as our primary goal is to make a lightweight model. This model was selected as the baseline model due to its effectiveness for the task and beneficial where computational resources are constrained. But the results were suboptimal, with a high RMSE. The lack of uncertainty quantification made this approach less reliable on its own.

### 6.1.2 Alternative CNN Models Architectures

Following the suboptimal results from the standard MobileNet, more different architectures were explored. These included deeper CNN architectures such as **ResNet** [79], **GhostNet** [54], **EfficientNet** [80], known for their strong performance in various computer vision tasks. Few other lightweight models were also explored. The aim was to determine and inspect how these models could improve the accuracy on hand pose estimation by leveraging different architectural designs to capture finer details in the data.

#### 6.1.2.1 Limitation

While some model architectures offered some improvements in feature extraction, others introduced significant computational overhead. The models were more resource-intensive, requiring longer training times and greater memory usage, which diminished their feasibility for hand pose estimation. These models faced issues with respect to lack of uncertainty quantification.

### 6.1.3 BNNs with Different Priors

The final experiment involved integrating Bayesian Neural Networks (BNNs) into the baseline MobileNet architecture. BNNs are distinguished by their ability to quantify uncertainty in predictions by treating the model's weights as distributions rather than fixed values. Different priors were tried and experimented. **Uniform Prior** [81] led the model to make weaker predictions, resulting in low accuracy. **Laplace prior** [82] performed better than uniform but often resulted in mixed prediction. **Gaussian prior** [83] produced the most efficient results,

balancing accuracy and uncertainty effectively.

# Chapter 7

# Results

## 7.1 Overall Model Performance

The Trust-Aware Bayesian MobileNetV2[1] (TABM) model was evaluated on a MHP dataset [3] consisting of 74,027 training samples and 8,226 testing samples. The model's performance metrics across the entire test set are as follows:

- Total RMSE [41]: 0.0592
- Total MAE [70]: 0.0397
- Total MSE [41]: 0.0035
- Total PCK [84]: 0.9365
- Total MJE [71]: 0.0626

[1]Follow the link to the Github: `https://github.com/Adeepsn/Thesis`

These metrics indicate that the model performs with a high degree of accuracy, particularly in predicting the correct keypoints with a high PCK value. The RMSE and MAE values are relatively low, demonstrating the model's ability to minimise both the squared and absolute differences between the predicted and actual keypoints.

### 7.1.1 Finger Specific Metric

From the results produced in Table 7.1, the model demonstrate consistency in its error metrics across different fingers.

| Metric | Value |
|---|---|
| **Total RMSE** | 0.0593 |
| **Total MAE** | 0.0398 |
| **Total MSE** | 0.0035 |
| **Total PCK** | 0.9366 |
| **Total MJE** | 0.0627 |
| **Index Finger RMSE** | 0.0494 |
| **Index Finger MAE** | 0.0343 |
| **Index Finger MSE** | 0.0024 |
| **Index Finger PCK** | 0.4355 |
| **Index Finger MJE** | 0.0537 |
| **Middle Finger RMSE** | 0.0772 |
| **Middle Finger MAE** | 0.0522 |
| **Middle Finger MSE** | 0.0060 |
| **Middle Finger PCK** | 0.0265 |
| **Middle Finger MJE** | 0.0784 |
| **Pinky Finger RMSE** | 0.0432 |
| **Pinky Finger MAE** | 0.0313 |
| **Pinky Finger MSE** | 0.0019 |
| **Pinky Finger PCK** | 0.5020 |
| **Pinky Finger MJE** | 0.0494 |
| **Ring Finger RMSE** | 0.0747 |
| **Ring Finger MAE** | 0.0518 |
| **Ring Finger MSE** | 0.0058 |
| **Ring Finger PCK** | 0.0840 |
| **Ring Finger MJE** | 0.0814 |
| **Thumb Finger RMSE** | 0.0471 |
| **Thumb Finger MAE** | 0.0329 |
| **Thumb Finger MSE** | 0.0022 |
| **Thumb Finger PCK** | 0.3227 |
| **Thumb Finger MJE** | 0.0519 |
| **Center Finger RMSE** | 0.0521 |
| **Center Finger MAE** | 0.0386 |
| **Center Finger MSE** | 0.0027 |
| **Center Finger PCK** | 0.0566 |
| **Center Finger MJE** | 0.0682 |

Table 7.1: Trust-Aware Bayesian MobileNet Results

While the variability in PCK score highlights areas where the model could further increase upon refinement. The consistency in RMSE, MAE, and MJE across keypoints reflects the model's adaptability to different anatomical features of the hand. The overall analysis reveal the model is generally robust and consistent.

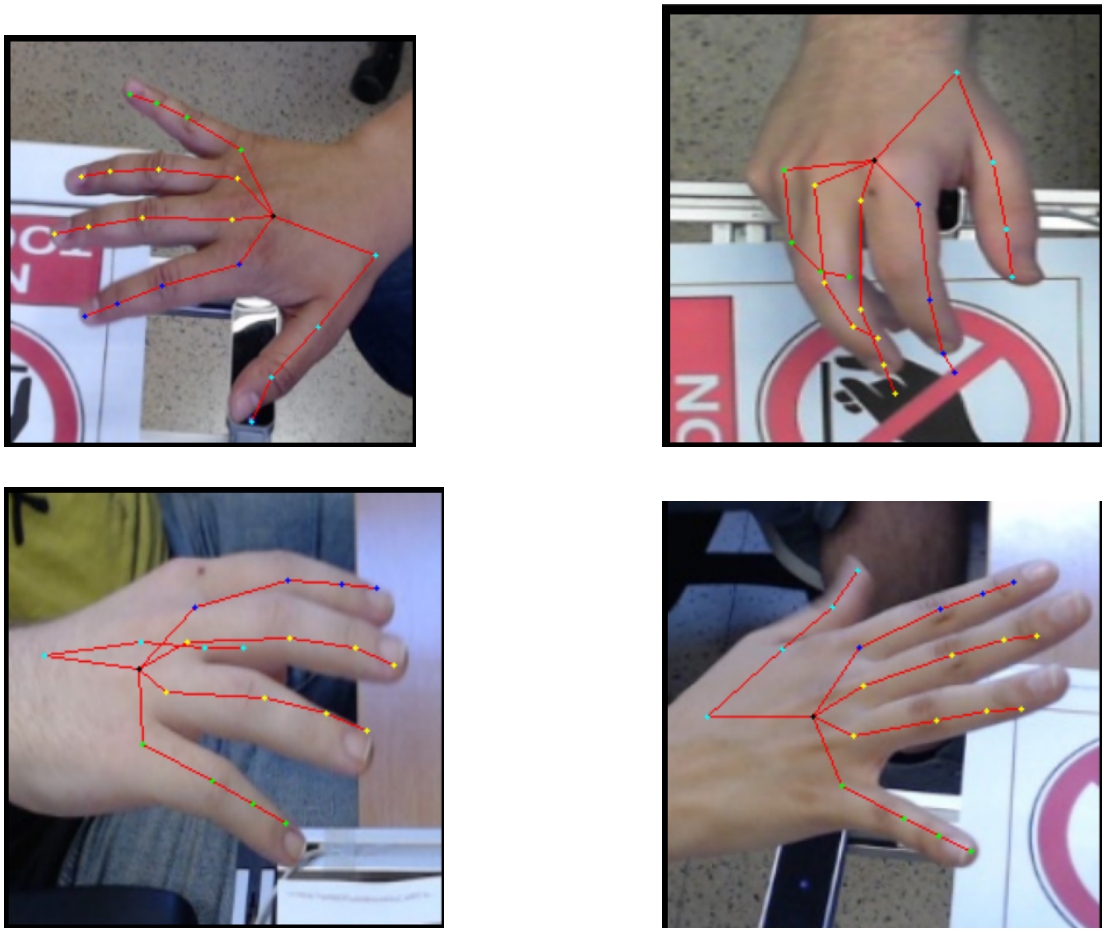

Figure 7.1: Results of the TABM model on random samples of image from the MHP dataset

## 7.2 Comparative Analysis

### 7.2.1 Baseline model

The table 7.2 compares the performance of the Trust-Aware Bayesian MobileNet (TABM) Model and the Standard MobileNet in terms of their RMSE after training on different epochs.

| **Metric** | **TABM Model (3 Epochs)** | **Std MobileNet (10 Epochs)** |
|---|---|---|
| **Total RMSE** | 0.0593 | 0.101 |
| **Total MAE** | 0.0398 | 0.0931 |
| **Total PCK** | 0.9366 | 0.6521 |

Table 7.2: Comparison of Trust-Aware Bayesian MobileNet (TABM) and Standard MobileNet

- In our comparison, the TABM model, even after just 3 epochs of training, significantly outperformed the standard MobileNet model that was trained for 10 epochs. Specifically, the TABM model achieved a much lower RMSE (0.0593 compared to 0.101) and MAE (0.0398 compared to 0.0931), indicating that it made more accurate predictions with fewer errors.

- Moreover, the TABM model's PCK score was much higher at 0.9366, compared to the standard MobileNet's 0.6521. This means the TABM model was far better at accurately predicting keypoints, consistently placing them within the correct locations.

- Overall, these results show that the TABM model is not only more accurate but also more efficient, requiring less training time to achieve better performance. This makes it a highly effective model for tasks where precise and reliable keypoint prediction is crucial.

### 7.2.2 Performance on Different Lightweight Models

Table 7.3 TABM model also outperforms the other lightweight models across all key performance metrics, particularly in RMSE, MAE, and PCK with the significantly low MJE suggests the model's reliable predictions with more accuracy.

| Models | Mob_v3Small | Regnet | ShuffleNet |
|---|---|---|---|
| **RMSE** | 0.1975 | 0.1620 | 0.1015 |
| **MAE** | 0.1518 | 0.0993 | 0.0602 |
| **MSE** | 0.0390 | 0.0262 | 0.0103 |
| **PCK** | 0.2953 | 0.6669 | 0.8407 |
| **MJE** | 0.2379 | 0.1628 | 0.0968 |

Table 7.3: Performance metrics from various lightweight models

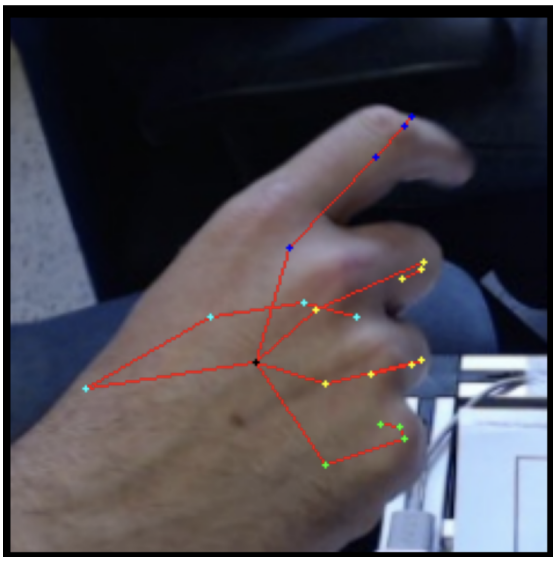

Figure 7.2: The image clearly shows the TABM models ability to precisely detect key-points when the thumb finger is occluded

Thus the result from Table 7.3 and the observation from Figure 7.2 justifies **RQ2** positively. As the trust aware mechanisms and uncertainty approximation helps to estimate key points correctly showing evidences the of TABM model's

robustness in case of high occlusion. Maintaining highest PCK by the TABM model provides reliability and consistency in keypoint prediction.

### 7.2.3 Performance on Different CNN Models

Table 7.4 shows TABM model outperforming popular CNN architectures like ResNet [79], GhostNet [54] and EfficientNet [80].

| **Models** | **ResNet18** | **GhostNet** | **EfficientNet** |
|---|---|---|---|
| **RMSE** | 0.1260 | 0.1114 | 0.2037 |
| **MAE** | 0.0918 | 0.0812 | 0.1677 |
| **MSE** | 0.0158 | 0.0124 | 0.0413 |
| **PCK** | 0.6460 | 0.7105 | 0.8211 |
| **MJE** | 0.1446 | 0.1281 | 0.1238 |

Table 7.4: Performance metrics from Different CNN models

On evaluating results from Table 7.2, 7.3, 7.4, TABM model shows that leveraging transfer learning from a pre-trained MobilnetV2 backbone, when integrated with a trust-aware mechanism and Bayesian uncertainty quantification does increase performance when compared to models like **EfficientNet** [80] and **GhostNet** [54] which are considered state of the art in low resource environment and computational efficiency. Even though the scope to this research was limited to MHP dataset [3]. It does provide a hypothesis of validating **RQ3** correctly.

Based on training and experimenting several model, Our TABM model outperformed several others, showcasing its potential strength in handling the data effectively. This positive result suggests that the approach has merit. However, since our experiments were limited to just one dataset, we cannot conclusively answer **RQ1**, hence is states negative. While the model's success on this spe-

cific dataset is encouraging, it does not provide sufficient evidence to confirm its general effectiveness across diverse data scenarios.

# Chapter 8

# Conclusion

In this thesis, we he investigated a research gap by integrating MobileNet with Bayesian Neural Networks (BNNs) to enhance the performance of hand pose estimation. The model was rigorously evaluated using the MHP (Multi-Hand Pose) dataset, which, while not widely recognised, provides a rich and challenging test-bed due to its diverse range of hand poses, varying lighting conditions, and the presence of multiple hands in complex scenarios.

## 8.1 Key Findings

The results from our experiments indicate that the proposed TABM model exhibits strong performance across a variety of metrics. This consistent performance across various challenging cases within the MHP dataset demonstrates the robustness of the model. The ability to detect keypoints despite the dataset's inherent challenges such as occlusions and variations in hand pose further supports this

claim.

Moreover, the model's reliability was evidenced by the minimal variance observed in performance metrics across repeated trials. This consistency suggests that the model is not only accurate but also dependable, delivering stable predictions regardless of the specific input variations within the dataset.

### 8.1.1 Significance of the Study

This research presents a model that balances efficiency with probabilistic reasoning, an approach that is particularly relevant for hand estimation on resource-constrained devices. The integration of Bayesian principles allows the model to quantify uncertainty in predictions, a feature that enhances its utility in scenarios where inputs may be noisy or ambiguous.

This study demonstrates the possibility to develop models that are both robust, trusted and reliable. This finding could encourage further exploration and broader adoption for future hand pose estimation research.

### 8.1.2 Limitations

Despite the promising results, it is important to acknowledge the limitations of this study. The primary limitation is the reliance on a single dataset, which constrains the generalisability of the findings. While the MHP dataset is well-suited for evaluating the model's performance in hand pose estimation, it does not encompass the full spectrum of potential real-world variations, such as different cultural hand signs, variations in hand sizes, or extreme lighting conditions.

Additionally, the computational complexity introduced by the Bayesian components, while justified by the enhanced uncertainty handling, may pose challenges for deployment on extremely resource-limited devices. Future optimisations could focus on reducing this overhead without sacrificing the model's robustness.

### 8.1.3 Future Work

To address the limitations and further validate the findings, future research should involve testing the proposed model on a broader range of datasets, including those that cover a wider variety of hand poses and environmental conditions. Expanding the evaluation to include datasets like RHD [85] (Rendered Hand Pose) or FreiHAND [86] could provide additional insights into the model's generalisability and robustness across different domains.

Moreover, exploring the integration of other lightweight architectures, such as ShuffleNet and GhostNet, with Bayesian Neural Networks could lead to further improvements in both efficiency and performance. These models could be compared to the TABM to determine the optimal balance between computational cost and predictive accuracy.

Furthermore, the application of this model in real-time systems, such as augmented reality interfaces or sign language recognition tools, can be explored. Such applications would not only validate the practical utility of the model but also highlight areas where further improvements might be necessary.

In conclusion, the proposed TABM model represents a step forward in the field of hand pose estimation, offering a robust and reliable solution for challeng-

ing real-world scenarios. While the findings are promising, further research is essential to confirm the model's broader applicability and to explore its potential in diverse and dynamic environments. This work lays a foundation for future advancements, particularly in the integration of lightweight architectures with probabilistic reasoning for enhanced performance.

# References


[1] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv preprint arXiv:1304.5634*, 2013. ii

[2] E. Alpaydin, *Machine learning*. MIT press, 2021. ii, 1, 6

[3] F. Gomez-Donoso, S. Orts-Escolano, and M. Cazorla, "Large-scale Multiview 3D Hand Pose Dataset," Jul. 2017, arXiv:1707.03742 [cs]. [Online]. Available: http://arxiv.org/abs/1707.03742 ii, vii, viii, 1, 4, 22, 24, 38, 43

[4] R. Li, Z. Liu, and J. Tan, "3d hand pose estimation: Cameras, methods, and datasets," *Pattern Recognition*, vol. 93, pp. 251–272, 2019. ii, 1, 4, 14

[5] Q. Ye and T.-K. Kim, "Occlusion-aware hand pose estimation using hierarchical mixture density network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–817. ii, 20, 30

[6] E. B. Hunt, *Artificial intelligence*. Academic Press, 2014. 1

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015. 1, 8

[8] A. Dongare, R. Kharde, A. D. Kachare *et al.*, "Introduction to artificial

neural network," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 2, no. 1, pp. 189–194, 2012. 1

[9] H. Ju, L. Yunhui, and Y. Ming, "Multi-camera calibration method based on minimizing the difference of reprojection error vectors," *Journal of Systems Engineering and Electronics*, vol. 29, no. 4, pp. 844–853, 2018. 1

[10] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022. 1

[11] J. Carmigniani and B. Furht, "Augmented reality: an overview," *Handbook of augmented reality*, pp. 3–46, 2011. 1

[12] S. Brandt and S. Brandt, *Data analysis*. Springer, 1998. 1

[13] Z. Han, C. Zhang, H. Fu, and J. T. Zhou, "Trusted Multi-View Classification," Feb. 2021, arXiv:2102.02051 [cs]. [Online]. Available: http://arxiv.org/abs/2102.02051 2, 18

[14] A. Erol, G. Bebis, M. Nicolescu, R. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, pp. 52–73, Oct. 2007. 4, 5, 6

[15] S. Gollapudi, *Practical machine learning*. Packt Publishing Ltd, 2016. 5, 6

[16] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks," *ACM Transactions on Graphics*, vol. 33, no. 5, pp. 1–10, Sep. 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2629500 6, 8, 9

[17] Y. Chen, W. Gao, and J. Ma, "Hand gesture recognition based on decision tree," *Institute of Computing Technology, Chinese Academy of Sciences, Beijing*, 2000. 6

[18] D. Tang, H. Jin Chang, A. Tejani, and T.-K. Kim, "Latent regression forest: Structured estimation of 3d articulated hand posture," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 3786–3793. 6

[19] C. R. MIHALACHE and B. APOSTOL, "A study on classifiers accuracy for hand pose recognition," *BULETINUL INSTITUTULUI POLITEHNIC DIN IASI, Bul. Inst. Polit. Iasi*, vol. 59, pp. 69–80, 2013. 6

[20] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," *Consumer Depth Cameras for Computer Vision: Research Topics and Applications*, pp. 119–137, 2013. 7

[21] F. Pedersoli, S. Benini, N. Adami, and R. Leonardi, "Xkin: an open source framework for hand pose and gesture recognition using kinect," *The Visual Computer*, vol. 30, pp. 1107–1122, 2014. 7, 8

[22] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-Time Human Pose Recognition in Parts from Single Depth Images." 7

[23] Y. Zhou, G. Jiang, and Y. Lin, "A novel finger and hand pose estimation technique for real-time hand gesture recognition," *Pattern Recognition*, vol. 49, pp. 102–114, 2016. 8

[24] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012. 8

[25] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "3d convolutional neural networks for efficient and robust hand pose estimation from single depth images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1991–2000. 8, 10, 14, 15

[26] D. Tang, Q. Ye, S. Yuan, J. Taylor, P. Kohli, C. Keskin, T.-K. Kim, and J. Shotton, "Opening the black box: Hierarchical sampling optimization for hand pose estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2161–2175, 2018. 8

[27] E. Dibra, T. Wolf, C. Oztireli, and M. Gross, "How to refine 3d hand pose estimation from unlabelled depth data," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 135–144. 9

[28] 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. 10

[29] A. A. Barbhuiya, R. K. Karsh, and S. Dutta, "Alexnet-cnn based feature extraction and classification of multiclass asl hand gestures," in *Proceeding of Fifth International Conference on Microelectronics, Computing and Communication Systems: MCCS 2020*. Springer, 2021, pp. 77–89. 10

[30] S. A. Salman, A. Zakir, and H. Takahashi, "Sdfposegraphnet: spatial deep feature pose graph network for 2d hand pose estimation," *Sensors*, vol. 23, no. 22, p. 9088, 2023. 10

[31] M. R. Al Koutayni, V. Rybalkin, J. Malik, A. Elhayek, C. Weis, G. Reis, N. Wehn, and D. Stricker, "Real-time energy efficient hand pose estimation: A case study," *Sensors*, vol. 20, no. 10, p. 2828, 2020. 10

[32] C. Wan, T. Probst, L. V. Gool, and A. Yao, "Self-supervised 3d hand pose estimation through training by fitting," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 10 853–10 862. 10, 11

[33] J. Zhao, X. Xijiong, X. Xu, and S. Sun, "Multi-view Learning Overview: Recent Progress and New Challenges," *Information Fusion*, vol. 38, Feb. 2017. 10

[34] A. Spurr, A. Dahiya, X. Wang, X. Zhang, and O. Hilliges, "Self-supervised 3d hand pose estimation from monocular rgb via contrastive learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 11 230–11 239. 11

[35] X. Zheng, C. Wen, Z. Xue, P. Ren, and J. Wang, "HaMuCo: Hand Pose Estimation via Multiview Collaborative Self-Supervised Learning," Aug. 2023, arXiv:2302.00988 [cs]. [Online]. Available: http://arxiv.org/abs/2302.00988 11

[36] J. N. Kundu, S. Seth, V. Jampani, M. Rakesh, R. V. Babu, and A. Chakraborty, "Self-supervised 3d human pose estimation via part guided novel image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6152–6162. 11

[37] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017. 12

[38] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International conference on machine learning*. PMLR, 2015, pp. 1613–1622. 12

[39] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059. 12, 17, 20, 21, 29

[40] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision," *Advances in neural information processing systems*, vol. 30, 2017. 12

[41] T. O. Hodson, "Root mean square error (rmse) or mean absolute error (mae): When to use them or not," *Geoscientific Model Development Discussions*, vol. 2022, pp. 1–10, 2022. 12, 38

[42] G. Moon, S.-I. Yu, H. Wen, T. Shiratori, and K. M. Lee, "Interhand2. 6m: A dataset and baseline for 3d interacting hand pose estimation from a single rgb image," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer, 2020, pp. 548–564. 12

[43] Y. Cai, "Vision-based 3d human and hand pose analysis," 2021. 13

[44] H. Rhodin, J. Spörri, I. Katircioglu, V. Constantin, F. Meyer, E. Müller, M. Salzmann, and P. Fua, "Learning monocular 3d human pose estimation from multi-view images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8437–8446. 13

[45] A. Boukhayma, R. d. Bem, and P. H. Torr, "3d hand shape and pose from images in the wild," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 10 843–10 852. 13

[46] A. Armagan, G. Garcia-Hernando, S. Baek, S. Hampali, M. Rad, Z. Zhang, S. Xie, M. Chen, B. Zhang, F. Xiong *et al.*, "Measuring generalisation to unseen viewpoints, articulations, shapes and objects for 3d hand pose estimation under hand-object interaction," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*. Springer, 2020, pp. 85–101. 13

[47] O. G. Guleryuz and C. Kaeser-Chen, "Fast lifting for 3d hand pose estimation in ar/vr applications," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 106–110. 14

[48] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3593–3601. 14

[49] Z. Zhang, J. Tang, and G. Wu, "Lightweight human pose estimation under resource-limited scenes," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 2170–2174. 15

[50] J. Zhang, D. Zhang, X. Xu, F. Jia, Y. Liu, X. Liu, J. Ren, and Y. Zhang, "Mobipose: Real-time multi-person pose estimation on mobile devices," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 136–149. 15

[51] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017. 15, 21, 29, 35

[52] A. Canepa, E. Ragusa, C. Gianoglio, P. Gastaldo, and R. Zunino, "A pose-based hand image classification method pretrained mobilenetv2 for embedded target devices," in *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*. IEEE, 2021, pp. 1–6. 15

[53] M. Vasileiadis, C.-S. Bouganis, G. Stavropoulos, and D. Tzovaras, "Optimis-

ing 3d-cnn design towards human pose estimation on low power devices." in *BMVC*, 2019, p. 42. 16

[54] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1580–1589. 16, 36, 43

[55] X. Li, Y. Guo, W. Pan, H. Liu, and B. Xu, "Human pose estimation based on lightweight multi-scale coordinate attention," *Applied Sciences*, vol. 13, no. 6, p. 3614, 2023. 16

[56] R. Caramalau, B. Bhattarai, and T.-K. Kim, "Active learning for bayesian 3d hand pose estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3419–3428. 17

[57] M. Oberweger and V. Lepetit, "Deepprior++: Improving fast and accurate 3d hand pose estimation," in *Proceedings of the IEEE international conference on computer vision Workshops*, 2017, pp. 585–594. 17

[58] C. Wan, T. Probst, L. Van Gool, and A. Yao, "Dense 3d regression for hand pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5147–5156. 17

[59] U. Iqbal, P. Molchanov, T. B. J. Gall, and J. Kautz, "Hand pose estimation via latent 2.5 d heatmap regression," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 118–134. 18

[60] B. Leichtmann, C. Humer, A. Hinterreiter, M. Streit, and M. Mara, "Effects of explainable artificial intelligence on trust and human behavior in a high-risk decision task," *Computers in Human Behavior*, vol. 139, p. 107539, 2023. 18

[61] A. Bera, D. Bhattacharjee, and M. Nasipuri, "Fusion-based hand geometry recognition using dempster–shafer theory," *International journal of pattern recognition and artificial intelligence*, vol. 29, no. 05, p. 1556005, 2015. 18

[62] W. Liu, X. Yue, Y. Chen, and T. Denœux, "Trusted Multi-View Deep Learning with Opinion Aggregation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 7585–7593, Jun. 2022. 19

[63] H. Liang, J. Yuan, J. Lee, L. Ge, and D. Thalmann, "Optimized clarity global hand pose estimation with arbitrary postures," *IEEE Transactions on Cybernetics*, vol. 49, no. 2, pp. 527–541, 2017. 20

[64] K. Shridhar, F. Laumann, and M. Liwicki, "A comprehensive guide to bayesian convolutional neural network with variational inference," *arXiv preprint arXiv:1901.02731*, 2019. 20, 30

[65] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000. 26, 27, 33

[66] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *Information*, vol. 11, no. 2, p. 125, 2020. 27, 33

[67] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019. 28, 29, 33, 34

[68] C. I. Papadopoulos, *On the Kullback-Leibler information measure and statistical inference*. Wayne State University, 1971. 30

[69] R. Bansal, G. Raj, and T. Choudhury, "Blur image detection using laplacian operator and open-cv," in *2016 International Conference System Modeling & Advancement in Research Trends (SMART)*. IEEE, 2016, pp. 63–67. 30

[70] M. S. Error, "Mean squared error," *MA: Springer US*, pp. 653–653, 2010. 30, 38

[71] M. A. Error, "Mean absolute error," *Retrieved September*, vol. 19, p. 2016, 2016. 30, 38

[72] T. Kim, J. Oh, N. Kim, S. Cho, and S.-Y. Yun, "Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation," *arXiv preprint arXiv:2105.08919*, 2021. 31

[73] M. F. Sanner *et al.*, "Python: a programming language for software integration and development," *J Mol Graph Model*, vol. 17, no. 1, pp. 57–61, 1999. 33

[74] M. Mowbray, H. Kay, S. Kay, P. C. Caetano, A. Hicks, C. Mendoza, A. Lane, P. Martin, and D. Zhang, "Probabilistic machine learning based soft-sensors for product quality prediction in batch processes," *Chemometrics and Intelligent Laboratory Systems*, vol. 228, p. 104616, 2022. 33

[75] J. Lampinen and A. Vehtari, "Bayesian approach for neural networks—review and case studies," *Neural networks*, vol. 14, no. 3, pp. 257–274, 2001. 33

[76] C. R. Harris, K. J. Millman, S. J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020. 34

[77] W. McKinney *et al.*, "Pandas, python data analysis library," *URL http://pandas. pydata. org*, pp. 3–15, 2015. 34

[78] E. Bisong and E. Bisong, "Matplotlib and seaborn," *Building machine learning and deep learning models on google cloud platform: A comprehensive guide for beginners*, pp. 151–165, 2019. 34

[79] B. Koonce and B. E. Koonce, *Convolutional neural networks with swift for tensorflow: Image recognition and dataset categorization.* Springer, 2021. 36, 43

[80] B. Koonce and B. Koonce, "Efficientnet," *Convolutional neural networks with swift for Tensorflow: image recognition and dataset categorization*, pp. 109–123, 2021. 36, 43

[81] S. Van Dongen, "Prior specification in bayesian statistics: three cautionary tales," *Journal of theoretical biology*, vol. 242, no. 1, pp. 90–100, 2006. 36

[82] A. Kaban, "On bayesian classification with laplace priors," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1271–1282, 2007. 36

[83] C. K. Williams and D. Barber, "Bayesian classification with gaussian processes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 12, pp. 1342–1351, 1998. 36

[84] Kili Technology, "Human Pose Estimation: The Ultimate Beginner's Guide (2023 Edition)," 2023, accessed: 2024-08-23. [Online]. Available: https://kili-technology.com/data-labeling/machine-learning/human-pose-estimation-ultimate-beginners-guide-2023-edition 38

[85] S. Lee, H. Park, D. U. Kim, J. Kim, M. Boboev, and S. Baek, "Image-free domain generalization via clip for 3d hand pose estimation," in *Proceedings*

*of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 2934–2944. 47

[86] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox, "Freihand: A dataset for markerless capture of hand pose and shape from single rgb images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 813–822. 47