VIETNAM NATIONAL UNIVERSITY - HO CHI MINH

UNIVERSITY OF SCIENCE

FINAL PROJECT PROPOSAL - TASK 2

HOPE-Net: A Graph-based Model for Hand-Object Pose Estimation

Subject: Graph mining

Class: 21KHDL

Teacher: Lê Ngọc Thành

Lê Nhựt Nam

Students: Doãn Anh Khoa - 21127076

Đoàn Việt Hưng - 21127289

Đinh Bảo Trân - 21127454

Lê Nguyễn Phương Uyên - 21127476

Ho Chi Minh City - 2024

Table of Contents

1) Task Assignment

Task Name	Team Member	Status
Introduction	Uyên	completed
Preliminaries and Backgrounds	Hưng, Uyên	completed
Related works	Hưng, Khoa, Trân	completed
Methodology	Khoa, Trân	completed
Experimental & Evaluation	Hưng, Khoa, Trân	completed
Conclusion	Uyên	completed
Slide	All member	completed

2) Introduction

2.1 Overview

- The authors present HOPE-Net, a lightweight model intended to mutually estimate in real-time the 2D and 3D postures of both hands and objects. The cutting-edge model HOPE-Net was created to take on the difficult task of estimating the poses of a hand and the object it is interacting with.
- The use of a graph-based model like HOPE-Net is innovative because it leverages the structural relationships between the hand and the object. By modeling the hand and the object as graphs, HOPE-Net can capture the dependencies between different parts of the hand and object, which leads to more accurate pose estimation.
- The authors highlight the model's potential applications in augmented reality, robotics, and action recognition, emphasizing its ability to handle occlusions and unpredictable camera motion.
- HOPE-Net's contributions to the field of pose estimation are significant. It advances the understanding of how to model and estimate complex hand-object interactions, setting the stage for further research in this area. Its graph-based approach may inspire new methodologies in related tasks, such as human pose estimation and scene understanding.

2.2 Motivation for Research

- The motivation of the paper is to address the limitations of existing pose estimation methods, which often struggle with occlusions, complex hand-object interactions, and varying environmental conditions. HOPE-Net aims to leverage graph-based neural networks to enhance the accuracy and robustness of pose estimation in these challenging scenarios.
- We choose this topic due to its relevance and potential for significant impact across multiple fields. Recent advancements in computer vision and machine learning have enabled more accurate pose

estimation, which is crucial for applications in augmented and virtual reality, robotics, human-computer interaction, and healthcare. This research aims to tackle challenges such as occlusions and dynamic scenes to enhance real-time processing capabilities. By addressing the scarcity of annotated data and leveraging synthetic data, the study contributes to both the academic community and practical applications. The interdisciplinary nature of this research promises to advance technology and improve user experiences in various contexts.

2.3 Scientific Significance and Real-World Applications

2.3.1 Theorical Meanings

HOPE-Net leverages advanced graph-based techniques to address the problem of estimating the pose of hands and objects from images. The theoretical underpinnings of HOPE-Net encompass several significant concepts in computer vision and machine learning:

- Graph Theory in Computer Vision: HOPE-Net utilizes graph theory to model the spatial relationships between keypoints of hands and objects. In this framework, keypoints are represented as nodes, and their connections are described as edges.
- Feature Concatenation and Conditioning: By concatenating image features with initial keypoint predictions, HOPE-Net conditions the graph convolution operations on visual information and the spatial context. This integration helps the model to refine keypoint predictions by leveraging both types of information, leading to more accurate pose estimations.
- Adaptive Graph U-Net: The Adaptive Graph U-Net in HOPE-Net converts 2D coordinates to 3D using graph convolution, pooling, and unpooling layers. It efficiently extracts and reconstructs hierarchical features, capturing coarse-level features and refining detailed ones, enabling effective multi-scale learning and 3D pose reconstruction from 2D inputs.

2.3.2 Practical Meanings

Addressing the problem of hand-object pose estimation can bring significant benefits in various fields. For example:

- VR/AR Applications: Enhance user experience by allowing more natural interactions with virtual objects.
- Robotics: Improve the ability to grasp and manipulate objects, enabling robots to perform more complex tasks.
- Security and Surveillance: Aid in identifying suspicious behaviors through the analysis of hand poses and the objects being held.

- Healthcare and Rehabilitation: Aids in physical therapy by analyzing and guiding patients' hand movements, helping in exercises and rehabilitation tasks.

3) Preliminaries and Backgrounds

3.1 Notation and Definitions

Notation/Terminology	Definitions
STNs	Spatial Transformer Networks (STNs) are neural networks designed to handle spatial transformations like scaling, rotation, and distortion
Renormalization Trick	The renormalization trick is a technique used to improve the performance of Graph Convolutional Networks (GCNs) when dealing with graphs of varying sizes and heterogeneous features
AUC	The Area Under the Curve (AUC) score is a performance metric used to evaluate the quality of a model's predictions, particularly in binary classification tasks. It represents the ability of the model to distinguish between positive and negative classes
MCP	(Metacarpoph0alangeal Joint): This is the joint between the metacarpal bone and the first phalanx of the finger
PIP	(Proximal Interphalangeal Joint): This is the joint between the two adjacent phalanges. Each finger has a PIP joint located between the MCP and DIP joints
DIP	(Distal Interphalangeal Joint): This is the joint between the phalanx closest to the fingertip
TIP	(Tip of the Finger): This is the end part of the finger, where the fingertip is located
MCP	(Metacarpophalangeal Joint) The joint at the base of the finger, where the finger meets the hand

3.2 Problem Statements

- **Hand-Object Pose Estimation**: The paper addresses the challenge of accurately estimating the poses of hands and the objects they manipulate in real-time. Estimating 3D poses of both the hand and object in interaction scenarios is crucial for applications in augmented reality (AR), virtual reality (VR), and human-computer interaction (HCI). However, this task is challenging due to the complex and highly articulated nature of the human hand and the occlusions that occur during object manipulation.

- **Occlusion and Interaction Complexity**: A significant problem in hand-object pose estimation is the occlusion of hand parts by the object and vice versa. This makes it difficult for conventional methods to accurately predict the pose. The interaction between the hand and object introduces further complexity, as the model needs to differentiate between the hand's joints and the object's contours.

- **Real-time Performance**: The need for real-time processing is a major challenge. Many applications require immediate feedback and interaction, which means that pose estimation models need to be both highly accurate and computationally efficient.

3.3 General Frameworks

- **ResNet (Residual Network)**: a deep learning architecture introduced in 2015 that utilizes shortcut connections to address the vanishing gradient problem, enabling effective training of very deep neural networks. Its residual blocks consist of convolutional layers with skip connections, allowing for efficient gradient flow and facilitating the development of networks with hundreds of layers, such as ResNet-50 and ResNet-152. This architecture has achieved state-of-the-art performance in image classification tasks and has significantly influenced advancements in computer vision.

- **Graph Convolutional Networks (GCNs)**: The HOPE-Net framework builds upon the concept of Graph Convolutional Networks (GCNs), which are used to model the spatial relationships between different joints of the hand and parts of the object. GCNs are advantageous because they can effectively capture the structural information of non-Euclidean data, like the human hand's skeletal structure. This helps in creating a more accurate model of the hand-object interaction.

- **Adaptive Graph U-Net**: The paper introduces an **Adaptive Graph U-Net** architecture, which is a variation of the traditional U-Net architecture, adapted for graph-based data. The U-Net is commonly used in segmentation tasks and is characterized by its "U" shape, which enables the model to capture both local and global features. The adaptive nature of the Graph U-Net allows it to dynamically adjust the graph structure during training, improving the model's robustness to occlusions and variations in hand-object interactions.

- **Multi-Stage Estimation**: The framework also utilizes a multi-stage estimation approach, where the 2D hand and object poses are first predicted and then lifted to 3D. This staged process allows the model to refine its predictions at each step, ultimately leading to more accurate and stable 3D pose estimations. This approach leverages the strengths of 2D image processing and combines them with 3D reconstruction techniques.

- **End-to-End Learning**: HOPE-Net is designed as an end-to-end learning system, meaning that the model is trained to directly predict the 3D poses from raw input data without the need for intermediate steps or manual feature extraction. This approach is facilitated by deep learning, which allows the model to learn complex mappings from input images to output poses through backpropagation and large-scale data.

3.4 Challenges and Limitations

- **Occlusion:** Hands move quickly as they interact with the world, and handle an object. When the hand and object occlude each other from nearly any given point of view, accurately identifying keypoints becomes difficult. Occluded points may be incorrectly identified or missed.

- **Hand-Pose and Object Shapes:** The shape of an object usually constrains the types of hand poses that can be used to handle it. There is a significant variability in the shapes, sizes, and appearances of both hands and objects. This diversity requires models to be highly adaptable and capable of generalizing across different instances.

- **Dynamic and Real-Time Processing:** Estimating poses in real-time is essential for applications like augmented reality and robotics. Achieving high accuracy without compromising on speed and computational efficiency is a major challenge.

- **Lighting and Environmental Conditions:** Varying lighting conditions and environmental backgrounds can adversely affect the performance of pose estimation models. Robustness to these factors is crucial for reliable performance in diverse settings.

- **Data Scarcity and Annotation:** Acquiring annotated datasets for hand-object interactions is difficult and expensive. The scarcity of high-quality, annotated data hinders the training and evaluation of models.

4) Related Works

Our work is related to two main lines of research: joint hand-object pose prediction models and graph convolutional networks for understanding graph-based data.

Hand-Object Pose Estimation. Due to the strong relationship between hand pose and the shape of a manipulated object, several papers have studied joint estimation of both hand and object pose.

- *Oikonomidis et al. [7]* used hand-object interaction as context to better estimate the 2D hand pose from multiview images. The authors propose an optimization problem that jointly estimates the 26-DOF hand pose and the pose of the manipulated object using markerless multi-camera input. Extensive experiments with both simulated and real-world data validate the effectiveness of the approach.
  - Strength: Joint Hand-Object Model - The approach effectively models the interaction between the hand and the object, considering occlusions and physical constraints. This leads to more accurate hand pose estimation compared to methods that treat the hand in isolation.
  - Weakness: Computational Overhead - The method incurs additional computational costs due to the need to evaluate hand-object interpenetration constraints, making it slower compared to some other methods, especially when dealing with multiple views.
- *Oberweger et al. [6]* proposed an iterative approach by using Spatial Transformer Networks (STNs) to separately focus on the manipulated object and the hand to predict their corresponding poses. Later they estimated the hand and object depth images and fused them using an inverse STN.
  - Strengths: The framework can achieve higher accuracy by iteratively refining predictions than single-pass methods. The feedback mechanism helps the model handle occlusions better by gradually improving the estimations and leveraging partial visibility in different iterations.
  - Weaknesses: The iterative nature of the method can be computationally expensive, making it less suitable for real-time applications without significant optimization. The accuracy of the initial predictions can affect the overall performance, as large initial errors may not be fully corrected through iterations.

Graph Convolution Networks. Graph convolution networks allow learning high-level representations of the relationships between the nodes of graph-based data.

- *Zhao et al. [11]* proposed a SemGCN. This novel neural network architecture operates on regression tasks with graph-structured data, for capturing both local and global relationships among human body joints for 2D and 3D human pose estimation.
  - Strengths: By explicitly modeling the semantic relationships between body joints, SemGCN achieves better performance than traditional methods that treat joints independently. It reduces the number of parameters compared to other deep learning models, thus enhancing throughput and maintaining high accuracy. The graph-based approach allows the model to generalize well across different datasets and can be easily adapted to various types of input data.

  - Weaknesses: While the graph-based approach is powerful, it introduces additional complexity in terms of model design and implementation. The performance of SemGCN is highly dependent on the accuracy of the initial 2D pose detection. The model requires a large amount of annotated data for training, which can be a limitation in scenarios where such data is not readily available.

- *Li et al. [5]* used Actional-Structural Graph Convolutional Network (AS-GCN) for skeleton-based action recognition. AS-GCN enhances this by incorporating an A-link inference module to capture action-specific latent dependencies and extending skeleton graphs to represent higher-order dependencies.

  - Strengths: Rich Dependency Capture: AS-GCN captures both local and global dependencies among joints, improving action recognition accuracy. Future Pose Prediction: The additional prediction head enhances recognition performance by capturing detailed action patterns. State-of-the-Art Performance: AS-GCN outperforms existing methods on large-scale datasets like NTU-RGB+D and Kinetics.
  - Weaknesses: The model's complexity may lead to higher computational and storage requirements. The need for pretraining the A-link inference module and extensive training can be time-consuming.

Research foundation of this paper:

- *Gao et al. [1]* introduced the Graph U-Net structure with their proposed pooling and unpooling layers. Propose encoder-decoder architectures like Graph U-Nets with their proposed pooling and unpooling layers.

  - Strengths:
    * Graph U-Nets Architecture: Can encode and decode high-level features while maintaining local spatial information => Promising performance on pixel-wise prediction tasks.
    * gPool Layer: Selects nodes to form a smaller graph based on their projection values on a trainable vector.
    * gUnpool Layer: Restores the graph to its original structure using the position information from the corresponding gPool layer.
  - Weaknesses:
    * There is no locality information among nodes in graphs. The partition operation is not applicable on graphs, which cannot directly apply those pooling operations to graphs.
    * The global pooling operation will reduce all nodes to one single node, which leads to the restricted flexibility of networks.

    * The connectivity of selected nodes is inconsistent.

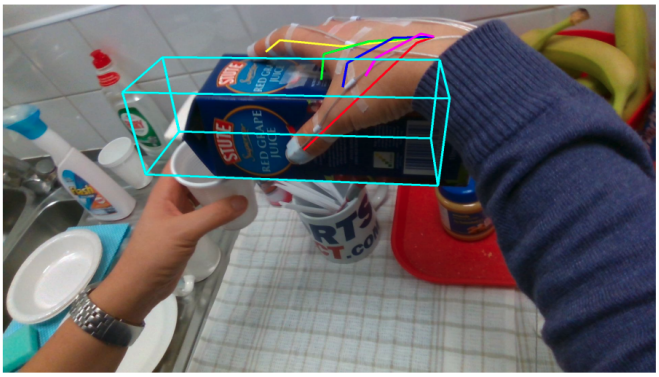
- However pooling method did not work well on graphs with low numbers of edges, such as skeletons or object meshes. *Ranjan et al. [8]* used fixed pooling and *Hanocka et al. [2]* used edge pooling to prevent holes in the mesh after pooling. This paper proposes a new Graph U-Net architecture with different graph convolution, pooling, and unpooling. We use an adaptive adjacency matrix for our graph convolutional layer and new trainable pooling and unpooling layers.

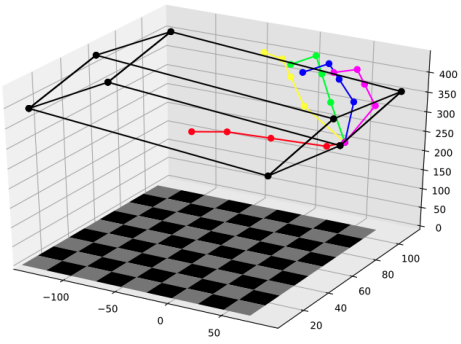
5) Methodology

5.1 Input and Output

- Input
  - RGB image dataset
  - True 2D coordinates of hand joints and object vertices
  - True 3D coordinates of hand joints and object vertices

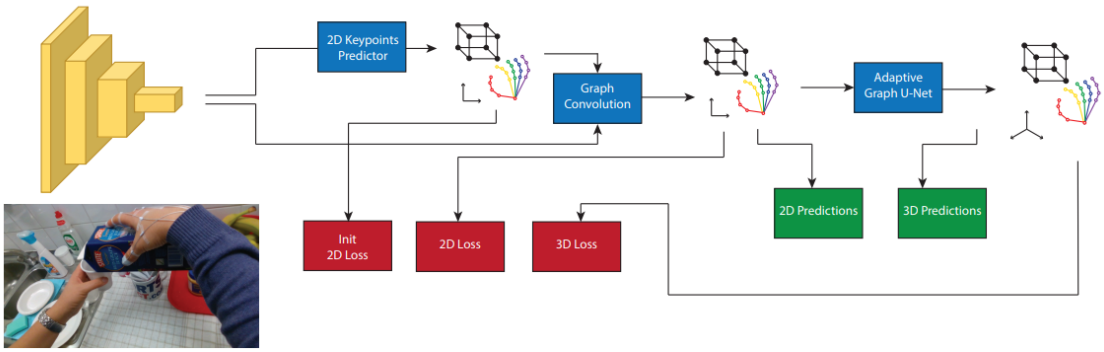
- Output
  - A set of trained weights
  - Predicted 2D coordinates of hand joints and object vertices
  - Predicted 3D coordinates of hand joints and object vertices
  - Additionally, the program exports video to visualize the results of HOPE-Net model based on true and predicted 3D coordinates of hand joints and objects





5.2 The architecture of HOPE-Net

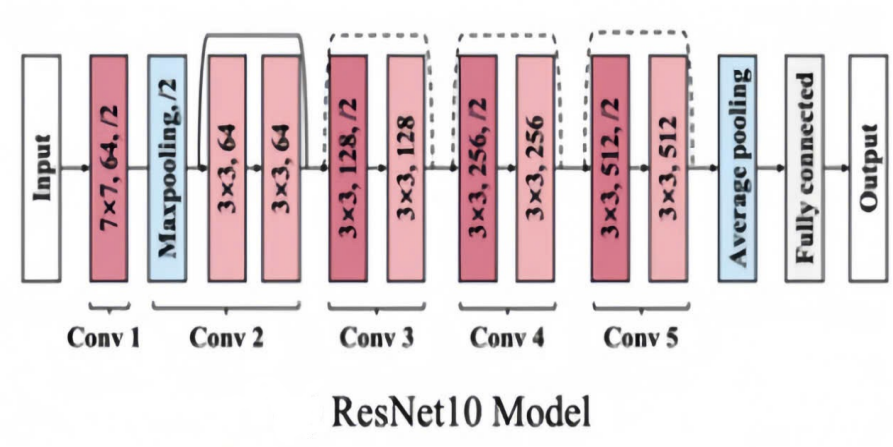
- The architecture includes: Convolutional neural network, Graph convolution, and Adaptive Graph U-Net
  - The model starts with a convolutional neural network(ResNet10) as the image encoder and for predicting the initial 2D coordinates of the hand and object key points (hand joints and tight object bounding box corners),
  - The coordinates concatenated with the image features are used as the features of the input graph of a simple graph convolution (a 3-layered graph convolution) to use the power of neighbor features to refine the predicted 2D predictions.
  - Finally, the 2D coordinates predicted in the previous step are passed to a Graph U-Net architecture(Adaptive Graph U-Net) using a series of graph convolutions, poolings, and unpoolings to find the 3D coordinates of the hand joints and object vertices.



5.3 Image Encoder and Graph Convolution

- This paper uses use a lightweight residual neural network (ResNet10) that is a simplified and smaller version of the Residual Neural Network architecture. ResNet10 has fewer layers and parameters

compared to larger versions like ResNet50 or ResNet101, making it faster and less computationally expensive. RestNet10 processes and transforms input images into a numerical representation (often called an embedding) that the network can more easily work with. The image encoder typically includes layers like convolutions and pooling.

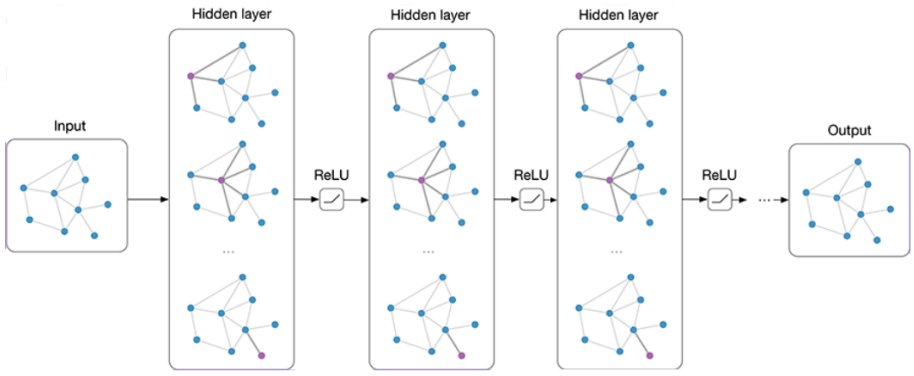


- ResNet, especially in its lighter variants, is computationally efficient, making it suitable for real-time applications like pose estimation. The residual connections in ResNet enable the network to learn robust features that are essential for accurately capturing complex hand-object interactions. Often, ResNet is pre-trained on large image datasets (like ImageNet), providing a strong starting point for learning relevant features in the specific context of hand-object pose estimation.

- The core of ResNet architectures is the residual blocks, where the network learns to refine features by adding (or skipping) connections between layers. After the final residual block, the network typically has a large number of feature maps, each representing different learned aspects of the image. For instance, in ResNet architectures, this can be 2048 feature maps (each corresponding to one filter in the last convolutional layer). The Global Average Pooling layer then averages each of these 2048 feature maps across their spatial dimensions (height and width). This operation reduces the spatial dimensions (e.g., 7x7 or 14x14) down to a single value per feature map, resulting in a 2048D vector. Therefore, The image encoder produces a 2048D feature vector for each input image.

- Then using a fully connected layer to produce initial predictions of the 2D coordinates of the keypoints (hand joints and corners of the object's tight bounding box). Inspired by the architecture of the paper [4], we concatenate these features with the initial 2D predictions of each keypoint, yielding **a graph with 2050 features** (2048 image features plus initial estimates of x and y) for each node.

- A 3-layer adaptive graph convolution network is applied to this graph to use adjacency information of the graph with a graph with 2050 features and modify the 2D coordinates of the keypoints.



- Concatenating the image features with the predicted x and y coordinates of each keypoint forces the graph convolutional network to adjust the 2D coordinates based on both the image features and the initial 2D predictions. The adjusted 2D coordinates for the hand and object keypoints are then input into our adaptive Graph U-Net. This network employs adaptive graph convolution, pooling, and unpooling techniques to transform the 2D coordinates into 3D coordinates.

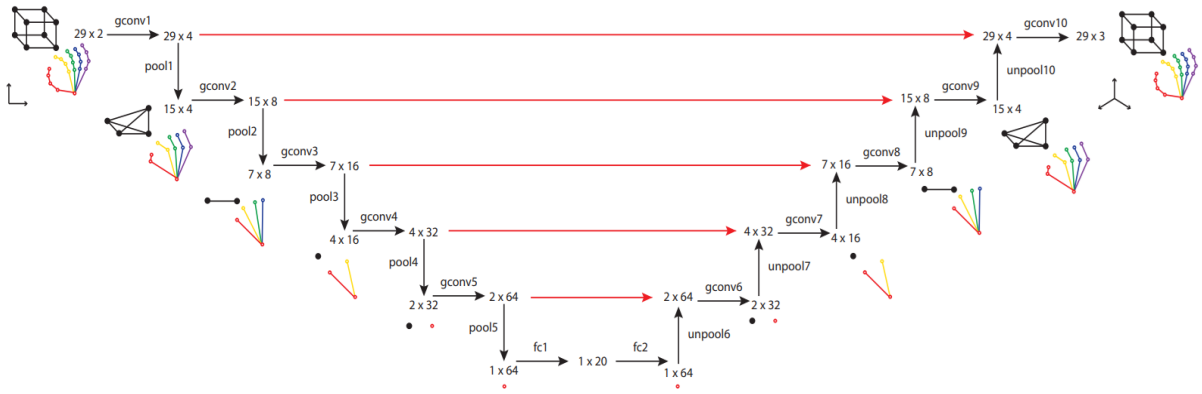
5.4 Adaptive Graph U-Net

- In the duration of studying the previous architectures, these authors in the paper figured out that
  - The Graph U-Net Model (*Gao et al. [1]*) used a sigmoid function in the pooling layer (gPool), which could lead to the problem of vanishing gradients. As a result, the selected nodes for pooling might not be updated, reducing the learning effectiveness of the network.
  - In addition, The gPool in Gao et al.'s model removes vertices and all their connected edges without any procedure to reconnect the remaining vertices. While this might not cause issues in dense graphs, it can be problematic for sparse graphs (e.g., mesh or hand/body skeleton graphs) where removing a node and its edges could split the graph into several isolated subgraphs, thus destroying connectivity—an essential feature in a graph convolutional neural network.
  - The model architecture with Fixed Pooling (*Ranjan et al. [8]*): Fixed pooling methods typically reduce the graph's size by a fixed ratio (e.g., 2x reduction). When applying fixed pooling, nodes or edges that contain crucial information might be aggregated or removed without considering their significance in the overall graph structure. This can degrade the performance of the model, especially in cases where maintaining detailed local structures is critical.
  - The model architecture with Edge Pooling (*Hanocka et al. [2]*): While edge pooling focuses on local connectivity by contracting edges, it might not adequately preserve the global structure of the graph. The emphasis on local structure might lead to an imbalance where the overall global properties of the graph are not well-maintained, potentially hindering the network's ability to learn useful representations at a higher level.

- In this network, the authors simplify the input graph by applying new trainable pooling in the encoding part, and in the decoding part, they add those nodes again with our graph unpooling layers. They care about simplifying the graph to obtain global features of the hand and object but also try to preserve local features via skip connections and feature concatenation from the encoder to the decoder layers.

- To address the vanishing gradients problem, the new model uses a fully connected layer during pooling instead of a sigmoid function. This ensures that the nodes selected for pooling are properly updated during training.

- They update the adjacency matrix in the graph convolution layers by using the adjacency information as a kernel applied to the graph. This approach allows the network to automatically find and maintain the connectivity of the nodes after each pooling layer, avoiding the issue of disconnected subgraphs, preserving the integrity of the graph, and improving the network's ability to learn features from sparse graphs.



- In each pooling layer, we roughly cut the number of nodes in half, while in each unpooling layer, we double the number of nodes in the graph. The red arrows in the image are the skip layer features passed to the decoder to be concatenated with the unpooled features.

5.4.1 Graph Convolution

- The core part of a graph convolutional network is the implementation of the graph convolution operation. The authors implemented a graph convolution layer based on the Renormalization Trick mentioned in [3].

- Hand joints can have many connections to object vertices. Renormalization Trick helps to reduce the influence of points with more connections, ensuring that all parts of the hand and object are learned more fairly. It also helps the model generalize better from data points with different structures, helping it learn the complex relationships between hands and objects more accurately.

- The output features of a graph convolution layer for an input graph with N nodes, k input features, and l output features for each node is computed as,

$$Y = \sigma(\tilde{A}XW),$$

where

  - σ is the activation function,
  - $W \in \mathcal{R}^{k \times l}$ is the trainable weights matrix,
  - $X \in \mathcal{R}^{N \times k}$ is the matrix of input features,
  - $\tilde{A} \in \mathcal{R}^{N \times N}$ is the row-normalized adjacency matrix of the graph,

$$\tilde{A} = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}},$$

where

  - $\hat{A} = A+I$, $\hat{A}$ simply defines the extent to which each node uses other nodes' features. Therefore, So $\tilde{A}X$ is the new feature matrix in which each node's features are the averaged features of the node itself and its adjacent nodes.
  - $\hat{D}$ is the diagonal node degree matrix.

- Previously, the authors experimented with an approach using a fixed adjacency matrix that was predefined based on the kinematic structure of the hand skeleton and the object bounding box. This matrix represents how nodes (joints or features) are connected fixedly. Then, they found that the static adjacency matrix does not adapt to the data during training, and the relationships between joints or features are fixed and predefined, which might not capture all the relevant relationships needed for the task. For instance, while the matrix may capture physical connections well, it might miss out on strong relationships that are not explicitly encoded, such as interactions between fingertips.

- For Adaptive Graph U-Net, instead of using a fixed adjacency matrix, they allow the network to learn an adjacency matrix. This learned matrix is more flexible and is often referred to as an "affinity matrix" because it can represent weighted connections between nodes. Unlike a traditional adjacency matrix that is binary (0 or 1), the affinity matrix can have continuous values that represent the strength or weight of connections between nodes. This means that nodes can be connected with varying degrees of strength to many other nodes, reflecting more complex relationships.

- In an adaptive graph convolution operation, both the adjacency matrix (A) and the weights matrix (W) are updated during the backpropagation step. By allowing the adjacency matrix to be learned,

the model can capture subtle and complex relationships between nodes that are not explicitly defined in the original kinematic structure.

- The paper uses ReLU as the activation function for the graph convolution layers. Their observation indicates that the network trains faster and generalizes better the network applied Batch or Group Normalization Batch Normalization or Group Normalization.

5.4.2 Graph Pooling

- As mentioned earlier, there are some problems when using the sigmoid function due to its weaknesses. The use of sigmoid made the network not update the randomly initialized selected pooled nodes during the entire training phase and lost the advantage of the trainable pooling layer.

- To solve the problem, the paper uses a fully connected layer for the graph pooling layer as a kernel, and applies it to the transpose of the feature matrix. Compared to the previous pooling layers, the authors found that this trainable pooling layer updated very well during training. Also due to using an adaptive graph convolution, this pooling does not separate the graph into subgraphs.

5.4.3 Graph Unpooling

- The unpooling layer used in their Graph U-Net is also different from the previous unpooling layers. This paper adds matrices that are pooled nodes to the graph with initially empty features and uses the subsequent graph convolution to update these matrices from the encoding phase to the decoding phase throughout training.

- Similar to the trainable pooling layer, they use a fully connected layer for an unpooling layer and apply it on the transpose matrix of the features to obtain the desired number of output nodes, and then transpose the matrix again.

5.5 Loss Function and Training the Model

- Loss function for training the model has three parts. We first calculate the loss for the initial 2D coordinates predicted by ResNet (L_{init2D}). We then add this loss to that calculated from the predicted 2D and 3D coordinates (L_{2D} and L_{3D}),

$$L = \alpha L_{\text{init2D}} + \beta L_{2D} + L_{3D},$$

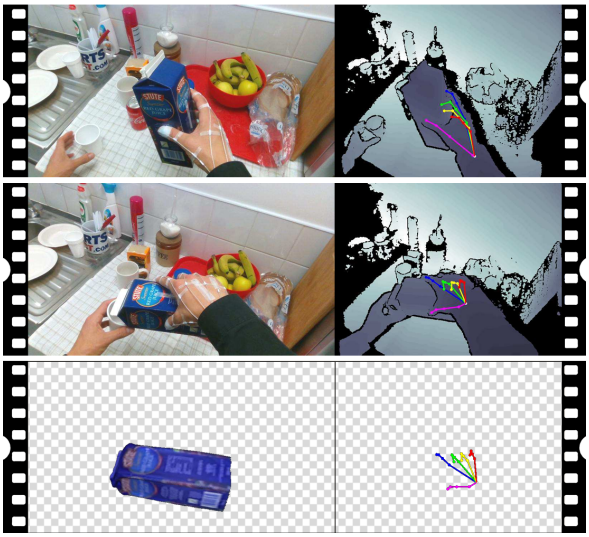
where we set α and β to 0.1 to bring the 2D error (in pixels) and 3D error (in millimeters) into a similar range. For each of the loss functions, we used Mean Squared Error.

6) Experimental & Evaluation

6.1 Dataset

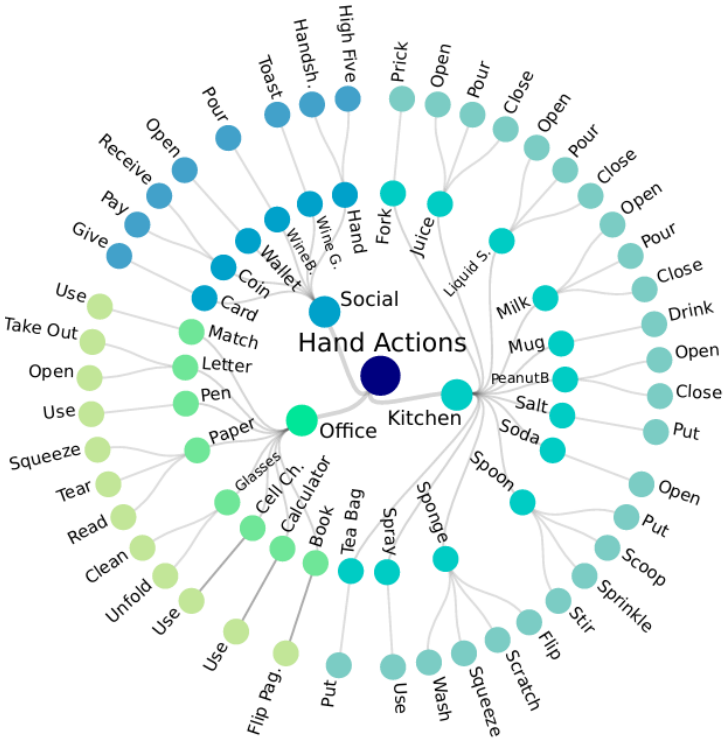
6.1.1 First-Person Hand Action Dataset

- **Context and Content**
  - This dataset features videos of hand actions performed from a first-person perspective, meaning the camera is worn by the user, giving an egocentric view.
  - The actions involve manipulating everyday objects such as milk cartons, juice bottles, liquid soap, and salt.
  - The dataset captures a variety of actions including opening, closing, pouring, and placing these objects.



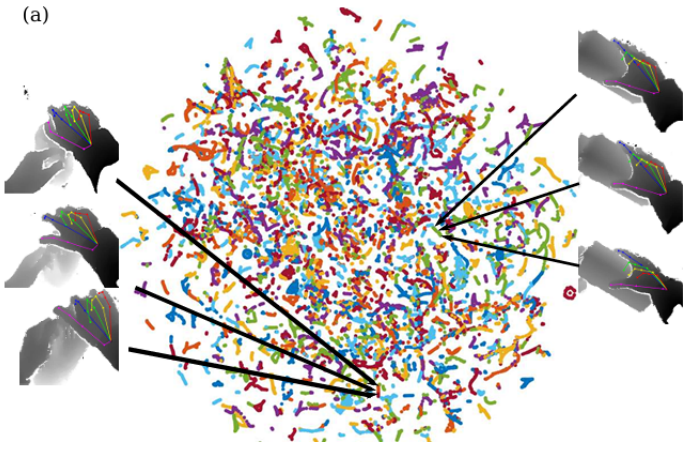
- **Visualization**

  A graph illustrating the action associated with each object



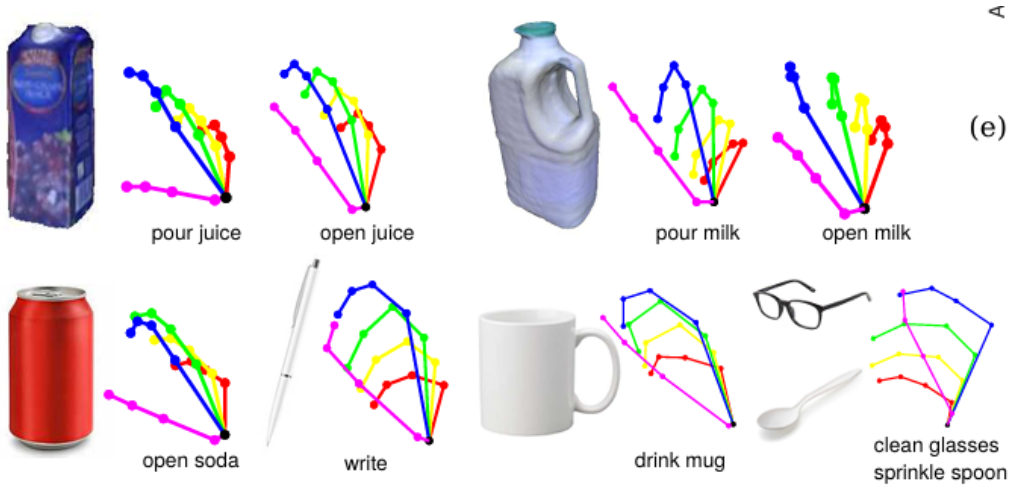
  - Objects with Multiple Actions:
    - Certain objects are used in various contexts and thus involve multiple hand actions.
    - For example: spoon (stirring, scooping, serving), sponge (scrubbing, cleaning, wiping), liquid soap (dispensing, lathering, rinsing),...
  - Objects with Single Action:
    - Some objects are linked to only one specific hand action.
    - For example: calculator (pressing buttons), pen: (writing or drawing), cell charger (plugging in and unplugging).

- **t-SNE visualization of Hand Pose Embedding**



- Colored Dots: Each dot represents a unique hand pose.
- Trajectories: Each trajectory shows a sequence of actions, illustrating how poses transition over time within an action sequence.

- **Correlation Between Objects, Grasps, and Actions**
  - A diagram depicts the relationships between objects, types of grasps, and actions.



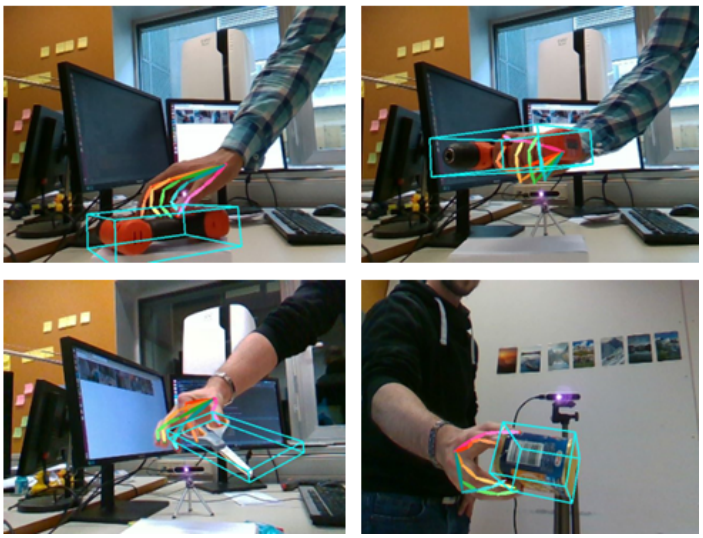
  - *Average Poses*: The poses shown represent the average pose across all action sequences for a given class.
  - *Objects and Grasps*: An object can be associated with multiple grasps depending on the action (e.g., 'juice carton' and 'milk bottle' might involve different types of grasps).
  - *Grasps and Actions*: A single grasp can be linked to various actions (e.g., a 'lateral grasp' might be used in both 'sprinkle' and 'clean glasses').

- **Graph Model Integration**
  - Pose Transformation: Each object in a frame is translated and rotated according to the provided 6D annotations to align the 3D object mesh to the correct pose.
  - Bounding Box Calculation: A tight oriented bounding box is computed using Principal Component Analysis (PCA) on the vertex coordinates of the 3D mesh.
  - Graph Representation: The eight 3D coordinates of the corners of this bounding box are used as nodes in a graph, which forms the basis of the pose estimation model.

6.1.2 HO-3D Dataset

- **Context and Content**

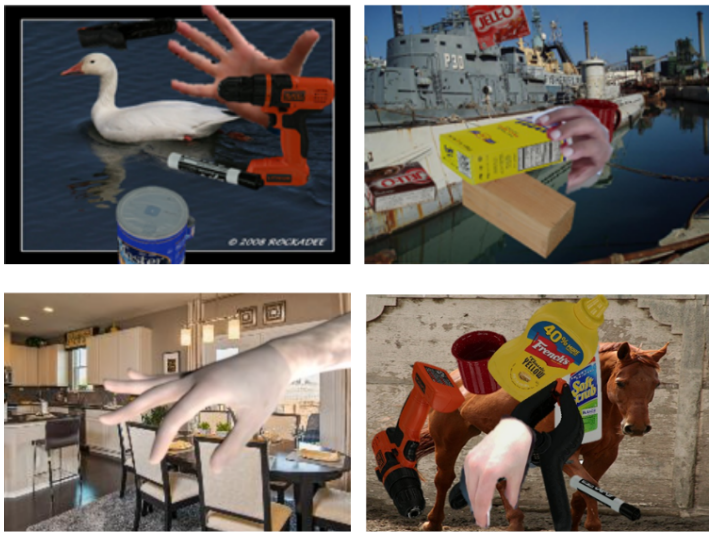
- HO-3D differs from the First-Person Hand Action Dataset as it provides a third-person view, meaning the camera is placed separately from the subject, capturing the interactions from a distance.
- The dataset includes a wide range of hand-object interactions, where hands and objects are generally smaller and appear further from the camera.



- **Annotations**
  - In the evaluation set, only the wrist coordinates are annotated, meaning the detailed positions of individual fingers and the full hand are not provided.
  - This limited annotation can restrict the level of detail captured in models trained on this dataset.

6.1.3 ObMan Dataset

- **Context and Content**
  - ObMan is a synthetic dataset, meaning the images are computer-generated rather than captured from real-world interactions.
  - The dataset focuses on hand-object interactions, created by rendering hand meshes interacting with objects selected from ShapeNet, a large-scale 3D object repository.



- **Usage and Findings**
  - Pre-training on ObMan's synthetic data helps the model learn general hand-object interaction patterns.
  - Fine-tuning the model with real-world images improves its performance, addressing the gap between synthetic and real data.

6.1.4 Comparison and Observation of Hand Action Datasets

1. First-Person Hand Action Dataset

Name	Quantity
Total Frames	21,501
Training Frames	11,019
Evaluation Frames	10,482

- Annotations: Detailed 6D pose annotations (3D translation and rotation for each object).
- Observations:
  - *Focus*: This dataset emphasizes detailed 6D pose annotations, which are crucial for tasks requiring precise object localization and orientation in 3D space.
  - *Size*: Moderate in size compared to the other two datasets, which may limit its coverage of various hand actions and object interactions.
  - *Use Case*: Ideal for training models that need high-precision pose estimation and understanding of object interactions in a first-person perspective.

2. HO-3D Dataset

Name	Quantity
Total Frames	77,558
Training Frames	66,034
Evaluation Frames	11,524

- Subjects: 10 people interacting with 10 distinct objects.
- Observations:
  - *Focus*: Contains a significant number of frames and a diverse set of subjects and objects. This diversity can help in training models that generalize well across different human-object interactions.
  - *Size*: Larger than the First-Person Hand Action Dataset, providing a more extensive set of interactions and potentially improving model robustness.
  - *Use Case*: Suitable for tasks that require diverse examples of human-object interactions, offering a balance between size and diversity.

3. ObMan Dataset

Name	Quantity
Total Frames	154,298
Training Frames	141,550
Validation Frames	6,463
Evaluation Frames	6,285

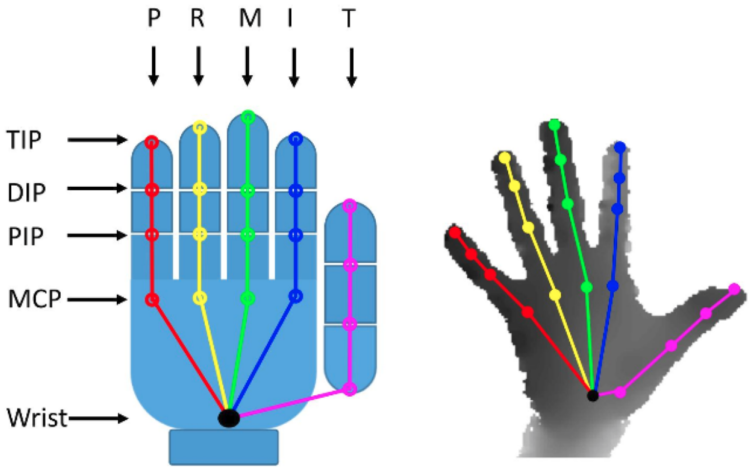
- Focus: Large-scale dataset with extensive training frames but smaller validation and evaluation subsets.
- **Observations**:
  - *Size*: The largest dataset among the three, providing a substantial amount of data for training. However, the smaller validation and evaluation subsets may limit the ability to thoroughly test model performance.
  - *Generalization Issue*: Models trained on ObMan alone struggle with generalization to real-world images, suggesting that while the dataset is large, it may not capture the variability present in real-world scenarios effectively.
  - *Use Case*: Useful for training models in a controlled environment but may need augmentation with real-world data to improve generalization.

Summary

- **First-Person Hand Action Dataset** is best for detailed pose estimation tasks but may lack the scale and diversity of the other datasets.
- **HO-3D Dataset** offers a good balance of size and diversity, making it suitable for training models that need to generalize across different human-object interactions.
- **ObMan Dataset** provides the largest volume of training data but may not generalize well to real-world scenarios due to its synthetic nature.

Common Feature Across Datasets: Hand model

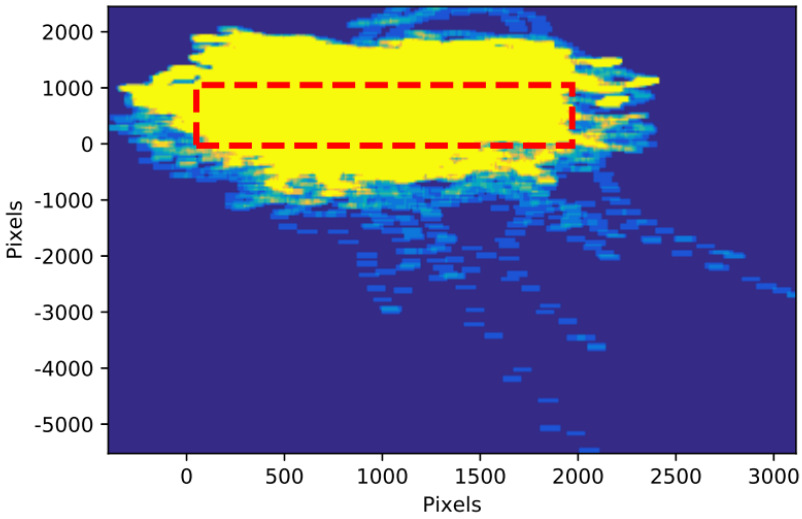
- All three datasets use a 21-joint hand model, where each hand is represented by one joint at the wrist and four joints per finger.
- This uniformity in the hand model allows for consistency when training and evaluating pose estimation methods across different datasets.



In practice, combining insights and data from these datasets could potentially yield models that are both well-trained on detailed actions and capable of generalizing to real-world conditions.

6.2 Implementation

6.2.1 Experimental



- Due to the nature of first-person video, hands often exit the frame, with roughly half of the frames in the First-Person Hand Action dataset having at least one keypoint outside the view.
- Thus, detection-based models are less effective for this dataset. Instead, a regression-based model is used to obtain initial 2D coordinates.
- To prevent overfitting, a lightweight ResNet is employed, which generalizes better and operates in near real-time.
- The official training and evaluation splits for both datasets are used, and pretraining is done on the ObMan dataset.
- **HOPE-Net Training Strategy**
  - HOPE-Net has different parameters and complexity, so the image encoder and graph components are trained separately.
  - The 2D to 3D conversion network is trained independently as it does not rely on the annotations. To improve robustness, 2D points are augmented with Gaussian noise ($\mu = 0$, $\sigma = 10$).

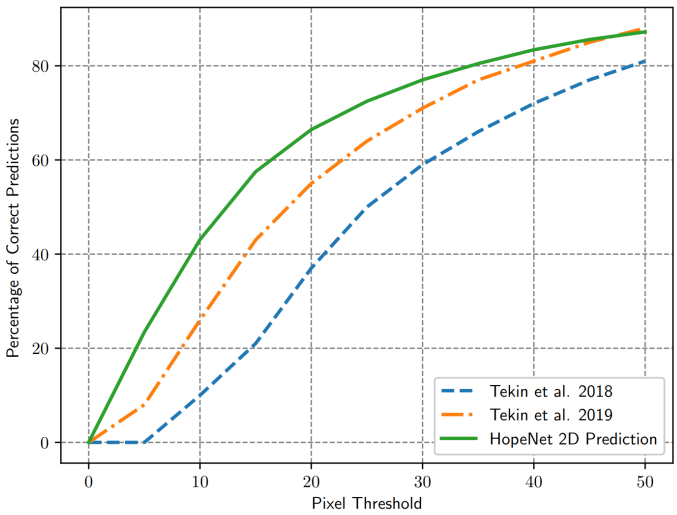
6.2.2 Training Details

- All images are resized to 224×224 pixels and processed with PyTorch.
- For both the FPHA and HO-3D datasets, ResNet is trained with an initial learning rate of 0.001, reducing it by 0.9 every 100 steps for 5000 epochs.

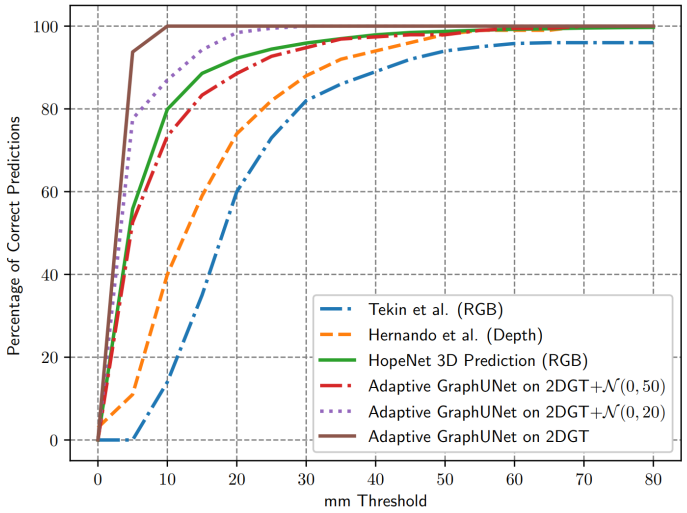
- The graph convolutional network is trained for 10,000 epochs with an initial learning rate of 0.001, reducing to 0.1 every 4000 steps.
- Finally, the model is trained end-to-end for another 5000 epochs.

6.2.3 Evaluation Metrics

- The models are evaluated using the percentage of correct pose (PCP) for both 2D and 3D coordinates.
- In this metric, a pose is considered correct if the average distance to the ground truth pose is less than a specified threshold.

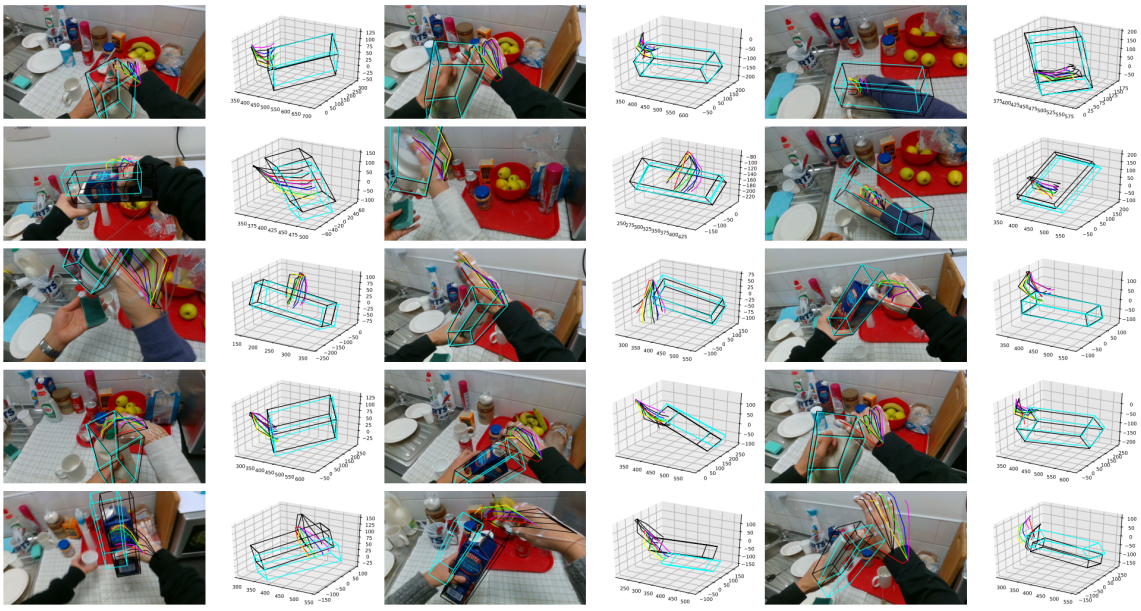


- We can see the performance of the model in hand and object pose estimation across two datasets. On the First-Person Hand Action dataset, HOPE-Net's 2D object pose estimates surpass the state-of-the-art model by Tekin et al. [10] and [9], even without an object locator and without temporal constraints.

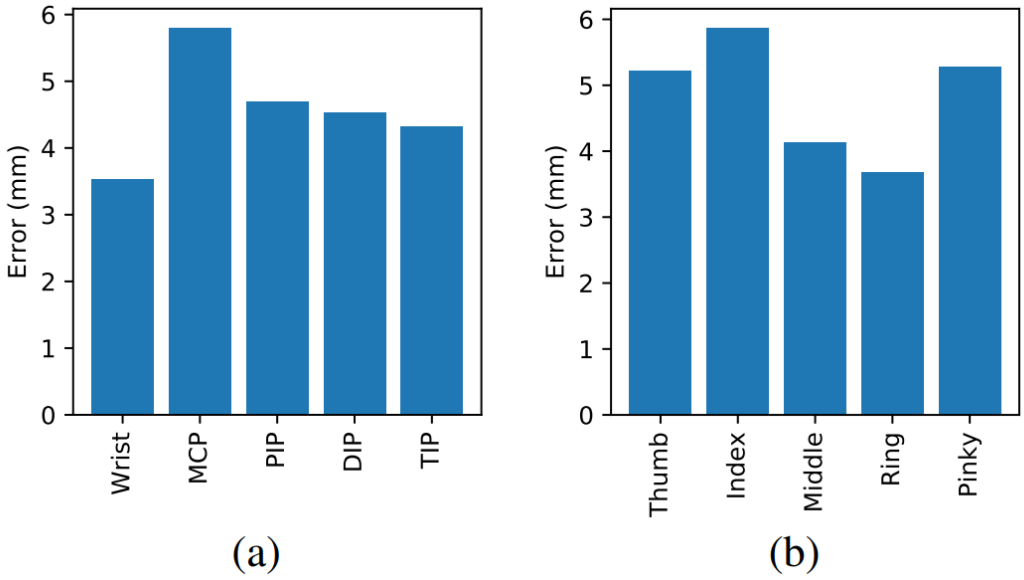


- The model also demonstrates superior performance in 3D pose estimation compared to Tekin et al.'s RGB-based model and Hernando et al.'s depth-based model, despite not using object localization or temporal information.

6.2.4 Robustness Evaluation



- Evaluating the model's robustness to Gaussian noise added to 2D coordinates and found that the graph model effectively mitigates this noise.
- Performance on the HO-3D dataset, which includes third-person views, shows HOPE-Net achieves an AUC score of 0.712 for 2D pose and 0.967 for 3D pose estimation, outperforming other models. Note that in HO-3D, only the wrist is annotated, not the full hand.
- The below charts illustrate differences in errors between joints and fingers:



- According to Chart (a), the highest error level is observed in the MCP class. This is the class most obscured and hardest to recognize when the hand is in motion.
- According to Chart (b), the highest error levels are associated with the thumb, index finger, and little finger:
  - For the thumb and index finger, which have the widest and most varied range of motion, more data is required for accurate recognition and action prediction.
  - For the pinky finger, it is the most obscured finger, making it more challenging to recognize.

6.2.5 Comparison with Other Models

Architecture	Average Error (mm)
Fully Connected	185.18
Adaptive Graph Convolution	68.93
Adaptive Graph U-Net	**6.81**

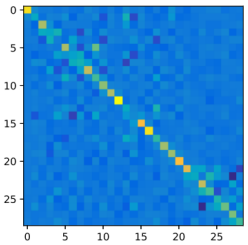
- To demonstrate the effectiveness of the U-Net structure, it was compared to two other models: one using three Fully Connected Layers and another using three Graph Convolutional Layers without pooling and unpooling.
- The Adaptive Graph U-Net model outperforms other models in estimating 3D poses with the lowest error.
- The importance of each graph convolutional model in the 3D output was analyzed, with each model trained to convert 2D coordinates of the hand and object.

6.2.6 Adaptive Graph U-Net Performance

Pooling method	Average Error (mm)
gPool [5]	153.28
Fixed Pooling Layers	7.41
Trainable Pooling	**6.81**

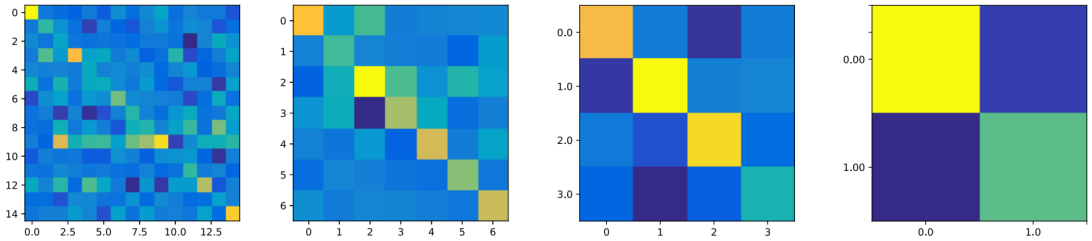
Trainable Pooling is the most effective pooling method, yielding the best results among the methods compared.

6.2.7 Adjacency Matrix Analysis & Importance of Adjacency Matrix Initialization



A_0

- The A_0 matrix shows that the corners of the object bounding box (row and column indices 21 through 29) are strongly dependent on each other.
- This indicates that these points have a close relationship during processing. There is also a relatively strong connection between the fingertips in the A_0 matrix, indicating that these points are of greater importance or have a close relationship within the graph structure.



A_1 A_2 A_3 A_4

- As we move from the A_1 matrix to the A_4 matrix, these matrices become simpler, with fewer strong connections, suggesting that complex relationships are learned in the initial layers and gradually become more generalized in the later layers.
- The adjacency matrices learned from the adaptive graph convolution model successfully capture important relationships between data points, especially for the points at the corners of the bounding box and the fingertips.

- The initialization of the adjacency matrix plays a crucial role in shaping the model's learning process and its final performance.

Initial Adjacency Matrix	Average Error (mm)
Zeros ($\mathbf{0}_{n\times n}$)	92805.02
Random Initialization	94.42
Ones ($\mathbf{1}_{n\times n}$)	63.25
Skeleton	12.91
Identity ($\mathbf{I}_{n\times n}$)	**6.81**

- While a well-suited initialization can lead to better and faster learning, a poorly chosen one might hinder the model's ability to capture the necessary relationships, leading to higher error rates and less effective results.

- Therefore, careful consideration of adjacency matrix initialization is essential in models like the Adaptive Graph U-Net, where the relationships between nodes are fundamental to the model's success.

6.2.8 Efficiency of HOPE-Net

- HOPE-Net is composed of a lightweight feature extractor (ResNet10) and two graph convolutional neural networks that are over ten times faster than even the most basic image convolutional neural networks.

- The core inference process of this model can be executed in real-time on an Nvidia Titan Xp. With this GPU, it takes only 0.005 seconds to perform both 2D and 3D inference on a single frame.

7) Conclusion

- In this paper, the authors presented a model for estimating hand-object 2D and 3D poses from a single image, utilizing an image encoder followed by a cascade of two graph convolutional neural networks. Our approach surpasses the state-of-the-art while also operating in real-time.

- However, our method does have certain limitations. When trained on the FPHA and HO-3D datasets, the model performs well for objects that are similar in size or shape to those encountered during training but may not generalize effectively across all object categories.

  – For instance, objects with non-convex geometries or lacking a well-defined 3D bounding box could pose a challenge for our technique. To enhance accuracy for real-world applications, a larger dataset that includes a broader variety of shapes and environments would be beneficial.

- Some applications in future work:
  - Future research could explore integrating temporal information into our graph-based model to improve pose estimation and move toward action detection.
  - Additionally, graph classification methods could be incorporated into the framework to infer semantic categories, enabling applications such as sign language recognition or gesture interpretation.
  - Beyond hand pose estimation, the Adaptive Graph U-Net proposed in this work has the potential to be applied to other areas, including graph completion, protein classification, mesh classification, and body pose estimation.

References

[1] Hongyang Gao and Shuiwang Ji. "Graph U-Nets". In: *International Conference on Learning Representations (ICLR)*. 2019.

[2] Rana Hanocka et al. "MeshCNN: A Network with an Edge". In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), p. 90.

[3] Thomas N Kipf and Max Welling. "Semi-supervised classification with graph convolutional networks". In: *International Conference on Learning Representations (ICLR)*. 2017.

[4] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. "Convolutional Mesh Regression for Single-Image Human Shape Reconstruction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[5] Maosen Li et al. "Actional-Structural Graph Convolutional Networks for Skeleton-Based Action Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[6] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. "Generalized Feedback Loop for Joint Hand-Object Pose Estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[7] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A. Argyros. "Full DOF Tracking of a Hand Interacting with an Object by Modeling Occlusions and Physical Constraints". In: *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2088–2095.

[8] Anurag Ranjan et al. "Generating 3D Faces Using Convolutional Mesh Autoencoders". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 704–720.

[9] Bugra Tekin, Federica Bogo, and Marc Pollefeys. "H+o: Unified Egocentric Recognition of 3D Hand-Object Poses and Interactions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 1–2, 6.

[10] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. "Real-Time Seamless Single Shot 6D Object Pose Prediction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 292–301.

[11] Long Zhao et al. "Semantic Graph Convolutional Networks for 3D Human Pose Regression". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 3425–3435.