POSE: Project Proposal

POSE: Project Proposal **0**
1. Project Overview 3
1a. Title 3
1b. Objective 3
1c. Scope 3
1d. AI Techniques and Tools 4
1e. Stakeholders 4
1e(i). Project Team 4
1e(ii). End Users 4
1e(iii). Other Stakeholders 5
2. Computing Infrastructure 5
2a. Project Needs Assessment 5
2b. Hardware Requirements Planning 5
2c. Software Environment Planning 6
2d. Cloud Resources Planning 6
2e. Scalability, and Performance Planning 6
3. Security, Privacy, and Ethics (Trustworthiness) 7
3a. Problem Definition 7
3a(i). Goal 7
3a(ii). Strategy: Stakeholder Involvement 7
3b. Data Collection 7
3b(i). Goal 7
3b(ii). Strategy: Data Anonymization and Privacy Techniques 7
3c. AI Model Development 8
3c(i). Goal 8
3c(ii). Strategy: Explainability Tools 8
3c(iii). Technical Implementation: SHAP (SHapley Additive exPlanations) 8
3d. AI Deployment 8
3d(i). Goal 8
3d(ii). Strategy: On-device Inference for Privacy 8
3e. Monitoring and Maintenance 9
3e(i). Goal 9
3e(ii). Strategy: Performance monitoring 9
4. Human-Computer Interaction (HCI) 9

4a. Step 1: Defining HCI Requirements..... 9
4a(i). Understanding user requirements..... 9
4a(ii). Creating personas and scenarios..... 10
4a(iii). Conducting task analysis..... 10
4a(iv). Identifying accessibility requirements..... 10
4a(v). Outlining usability goals..... 10
5. Risk Management Strategy..... 11
5a. Stage 1: Problem Definition..... 11
5a(i). Key risks..... 11
5a(ii). Mitigation strategy: Stakeholder Engagement..... 11
5b. Stage 2: Data Collection..... 12
5b(i). Key risks..... 12
5b(ii). Technical mitigation strategy: Automated Data Cleaning..... 12
5c. Stage 3: AI Model Development..... 12
5c(i). Key risks..... 12
5b(ii). Technical mitigation strategy: Explainability tools (SHAP)..... 12
5d. Stage 4: AI Deployment..... 13
5d(i). Key risks..... 13
5b(ii). Mitigation strategy: Integration Testing..... 13
5e. Stage 5: Monitoring and Maintenance..... 13
5e(i). Key risks..... 13
5b(ii). Technical mitigation strategy: Performance monitoring with TFProfiler..... 13
5f. Residual Risk Assessment..... 13
5f(i). Algorithmic Bias..... 14
5f(ii). Model Drift..... 14
5f(iii). Data Privacy Issues..... 14
5f(iv). Edge Device Limitations..... 15
5f(v). Security Vulnerabilities..... 15
5f(vi). User Safety..... 15
5f(vii). Regulatory Compliance..... 16
5f(viii). Unintended Behavior..... 16
6. Data Collection Management and Report..... 16
6a. Area 1: Data Type..... 16
6a(i). Unstructured Data..... 16
6a(ii). Processed (Structured) Data..... 17
6b. Area 2: Data Collection Methods..... 18
6b(i). Sources of Data..... 18
6b(ii). Methodologies Applied..... 18

6b(iii). Ingestion for Training..... 19
6b(iv). Ingestion for Deployment..... 19
6c. Area 3: Data Collection Methods.....19
6c(i). Applicable Laws and Standards..... 19
6c(ii). Compliance Strategy and Results.....19
6d. Area 4: Data Ownership and Access Rights.....19
6d(i). Ownership and Access Control.....19
6d(ii). Lessons Learned.....20
6e. Area 5: Metadata Management.....20
6e(i). Metadata Content and Management System.....20
6f. Area 6: Data Versioning..... 20
6f(i). Version Control System and Strategy.....20
6g. Area 7: Data Preprocessing, Augmentation, and Synthesis.....21
6g(i). Preprocessing Techniques..... 21
6g(ii). Data Augmentation..... 21
6g(iii). Data Synthesis.....21
6h. Area 8: Data Management Risks and Mitigation.....21
6i. Area 9: Data Management Trustworthiness and Mitigation.....22
7. Model Development and Evaluation..... 22
7a. Model Development.....22
7a(i). Algorithm Selection..... 22
7a(ii). Feature Engineering and Selection.....22
7a(iii). Model Complexity and Architecture.....23
7b. Model Training..... 23
7b(i). Training Process.....23
7b(ii). Hyperparameter Tuning..... 24
7c. Model Evaluation.....24
7c(i). Performance Metrics..... 24
7c(ii). Cross-Validation.....25
7d. Implementing Trustworthiness and Risk Management in Model Development..... 25
7d(i). Risk Management Report.....25
7d(ii). Trustworthiness Report..... 26
7e. Apply HCI Principles in AI Model Development.....29
7e(i). Develop Interactive Prototypes..... 29
7e(ii). Design Transparent Interfaces.....30
7e(iii). Create Feedback Mechanisms.....30
8. Deployment and Testing Management Plan..... 30
8a. Deployment Environment Selection..... 30

8a(i). Documentation..... 30
8b. Deployment Strategy..... 30
8b(i). Documentation..... 30
8c. Security and Compliance in Deployment (Trustworthiness and Risk Management).. 30
8c(i). Application..... 30
9. Evaluation, Monitoring, and Maintenance Plan..... 30
9a. System Evaluation and Monitoring.....30
9a(i). Documentation..... 30
9b. Feedback Collection and Continuous Improvement.....31
9b(i). Documentation..... 31
9c. Maintenance and Compliance Audits..... 31

1. Project Overview

1a. Title

POSE: Pose Classification for Accessible User Interfaces

1b. Objective

POSE is an AI system designed to enable accessible, hands-free user interfaces through real-time pose classification. The system leverages a transfer-learned convolutional neural network to detect and classify specific poses – such as “hands raised” or “hands on hips” – from live camera input, with inference performed directly on an edge device. While a basic pose estimation model could infer poses by calculating joint ratios, POSE bypasses the need for this complex and potentially unreliable manual setup and yields a more generalizable result. The tool is designed for accessibility-focused product designers and interface developers aiming to create user experiences that prioritize autonomy and privacy. With POSE’s focus on edge deployment, it serves applications which value real-time, private interaction without reliance on cloud-based processing, offering a secure, efficient interface that meets accessibility needs in various practical scenarios.

1c. Scope

POSE is limited to detecting and analyzing specific body poses from live video taken at a designated camera angle, focusing on visual input to facilitate hands-free interaction. The system does not provide feedback on non-visual elements but instead interprets poses such as “hands raised” as an interaction cue. This functionality allows POSE to enhance accessibility in scenarios where conventional control methods may be impractical or inaccessible, such as enabling hands-free menu navigation in an application. Inference is performed locally on the

edge device's camera feed using a model trained on ethically sourced, labeled data representing a wide range of poses. Limitations include the computational efficiency of both the model and the edge device, potential biases in training data, and the need for a stable camera angle to ensure accurate pose classification.

1d. AI Techniques and Tools

POSE relies on transfer learning, a subset of deep learning, to adapt a pre-trained pose estimation model to the specific task of pose classification, enabling the system to classify user poses directly rather than requiring complex heuristics based on ratios between joint positions. LiteRT (formerly TensorFlow Lite) and TensorFlow are used for model training and inference tasks, ensuring compatibility with edge devices. OpenCV manages video input and image processing, while NumPy handles the necessary numerical operations. MoveNet serves as the pre-trained model for transfer learning, providing a robust base for pose estimation that is then extended to pose classification. Optimized for real-time, on-device performance, MoveNet offers two versions: Lightning for low-latency tasks and Thunder for enhanced accuracy, allowing POSE to meet both responsiveness and precision requirements on edge devices.

1e. Stakeholders

1e(i). Project Team

The POSE project team consists of one member, Norman, who will take on multiple roles:

- **AI Engineer**: Designing and implementing the pose classification model using transfer learning, while optimizing for real-time edge device performance.
- **Data Scientist**: Collecting, labeling, and preparing datasets for training, and addressing potential data biases.
- **Software Developer**: Integrating TensorFlow Lite for model deployment, managing video input processing with OpenCV.
- **UI/UX Designer**: Developing an intuitive user interface to deliver real-time feedback on posture.
- **Project Manager**: Overseeing project timelines, milestones, and deliverables, ensuring all objectives are met.

1e(ii). End Users

The primary users of POSE include accessibility-focused product designers and interface developers who seek to create hands-free, user-friendly interaction experiences. These users can integrate POSE into a range of applications, such as hands-free navigation systems for individuals with mobility impairments or interactive displays in public spaces. By enabling gesture-based control through pose classification, POSE empowers designers and developers to

offer privacy-preserving, real-time interaction capabilities which do not rely on traditional input methods.

Users of applications built with POSE will interact with it through a setup involving a camera and a monitor for displaying feedback, all connected to an edge device. The camera captures the user's posture and movement in real time as input, while the edge device runs local inference on it. The pose is recognized as an interaction cue, and the interface updates accordingly.

1e(iii). Other Stakeholders

Other potential stakeholders in POSE include data providers, such as the MPII Human Pose dataset, which offers additional training data critical to the accuracy of the AI system's underlying pose classification model. Regulators are also key stakeholders, ensuring the project complies with data privacy laws like GDPR, particularly as the system processes video data on edge devices. Additionally, external partners (e.g. classmates, the professor, and any other interested parties within reason) may volunteer to play a role in testing and providing feedback on the system's usability and effectiveness.

2. Computing Infrastructure

2a. Project Needs Assessment

The AI system’s primary objective is **classification**, focusing on identifying specific body poses from live camera input in real-time. The input data types will consist of frames extracted from a video camera’s live feed, processed as individual **images** for pose recognition. The system will be deployed on an edge device with constraints related to power, network connectivity, and real-time performance.

Performance benchmarks include achieving **less than 50 ms latency per frame** to ensure real-time response, processing **at least 15–30 frames per second (FPS)** for smooth pose detection, and maintaining an **accuracy of over 85%** in classifying key poses like "crossed arms" or "hands raised," with minimal false positives.

Deployment constraints include running the system on an edge device with limited computational resources, **requiring lightweight models** optimized for performance and energy efficiency. The device must function effectively in low-power environments and **without constant internet access**, as the system will operate locally without cloud resources.

2b. Hardware Requirements Planning

For training (more precisely, fine-tuning on a dataset of labeled poses) the small pose classification model, **HiPerGator resources** will be used. Given that the model is based on

MoveNet, which is known for its lightweight design and ability to run efficiently even on edge devices, HiPerGator's computational resources will significantly reduce the time required for training.

For edge deployment, a **Raspberry Pi 5** will serve as the primary device. MoveNet is designed to run on devices with constrained resources, and its smaller model size (around 11 MB on Kaggle) ensures that it can perform real-time inference on the Raspberry Pi with minimal latency and power consumption.

2c. Software Environment Planning

Ubuntu will be used as the operating system due to its compatibility with LiteRT (TensorFlow Lite), the primary framework for training and deploying the pose classification model. The training pipeline will rely on Python with libraries such as NumPy for numerical operations, Pandas for data management, and OpenCV for handling image and video frames.

For edge deployment on the Raspberry Pi, Docker will be utilized to ensure a lightweight and consistent environment for running the optimized LiteRT model. A Python script will handle live video input, processing each frame for real-time pose classification. To enable remote monitoring, a Flask or FastAPI server will be set up, exposing a web portal to display the model's output.

Performance monitoring will be set up to track key metrics such as inference time, memory usage, and power consumption during real-time inference. Tools like TensorBoard can be used during the optimization phase to profile the model’s performance, and real-time logging on the Raspberry Pi can help monitor performance in deployment.

2d. Cloud Resources Planning

Access to HiPerGator is required for model training. In addition, Google Cloud Storage will be utilized to securely store the training data, ensuring consistent accessibility and data management throughout the training pipeline. No further cloud resources are needed, as model hosting, inference, and processing of user input data will be handled entirely on-device.

2e. Scalability, and Performance Planning

Given that the pose classification model will only handle one video input at a time on an edge device, focus will be placed on performance optimization over scalability to ensure that the model runs efficiently on devices with lower computational power than the Raspberry Pi 5.

Techniques such as model quantization, which reduces the precision of weights and activations, will be essential for decreasing the model size and making it more suitable for edge deployment.

Additionally, model pruning, which removes unnecessary neurons and connections, can further reduce computational overhead without sacrificing too much accuracy.

3. Security, Privacy, and Ethics (Trustworthiness)

3a. Problem Definition

3a(i). Goal

The goal is to ensure data security, fairness, and transparency across all user groups, with strict limits on data collection and retention. Potential risks include the misinterpretation of body language and improper use in surveillance that may infringe on individual rights. In addition, algorithmic bias could result in the model performing inconsistently across different demographic groups, such as race, body type, or gender, potentially causing unequal outcomes.

3a(ii). Strategy: Stakeholder Involvement

To gather diverse perspectives on the ethical impacts of the system, it is essential to involve the stakeholders (listed in a prior section above). This is to be accomplished through interviews and/or surveys aimed at identifying concerns and suggestions. By involving stakeholders early and continuously throughout the development process, the system can be designed with ethical considerations, promoting trust and responsibility.

3b. Data Collection

3b(i). Goal

The goal is to gather high-quality, representative, and unbiased data to train the pose classification model. Since the system will detect poses like "hands raised" or "arms crossed," it is crucial to collect data from a diverse range of participants with different body types, ages, genders, and ethnicities to minimize bias. This ensures the model can perform reliably across different demographic groups and in various real-world conditions. Additionally, the dataset must capture a wide range of poses from different angles and lighting environments to improve the robustness of the model in real-time applications.

3b(ii). Strategy: Data Anonymization and Privacy Techniques

Data minimization will be a key privacy strategy during data collection, ensuring that only essential pose-related information is captured. In other words, the dataset will specifically target the required poses like "hands raised" or "arms crossed," avoiding unnecessary data. Additionally, collected data will be stored only for the duration needed for model training and

will be securely deleted afterward, reducing long-term privacy risks while maintaining dataset efficiency and privacy.

3c. AI Model Development

3c(i). Goal

The goal is to develop a fair, interpretable, and robust pose classification model that performs reliably across diverse scenarios and user groups. Ensuring fairness across different demographic groups (age, body type, gender) and environments is critical to avoid biased predictions. The model should also be transparent, allowing stakeholders to understand how decisions are made and how the model functions under various conditions.

3c(ii). Strategy: Explainability Tools

To ensure the pose classification model is transparent and interpretable, explainability tools will be incorporated into the development process. These tools provide insights into how the model arrives at its predictions, making it easier to identify any potential biases or errors in decision-making. This improves stakeholder trust and ensures that the model's predictions are understandable and accountable.

3c(iii). Technical Implementation: SHAP (SHapley Additive exPlanations)

SHAP (SHapley Additive exPlanations) will be used to provide interpretability for the pose classification model by explaining how different body keypoints contribute to its predictions. SHAP integrates with TensorFlow, making it usable with POSE. The shap library includes visualization tools such as force plots, beeswarm plots, and waterfall plots, which help illustrate the importance of features for each prediction. By using the DeepExplainer, SHAP enables deep learning models to be interpreted in a granular and transparent way.

3d. AI Deployment

3d(i). Goal

The goal is to securely deploy the pose classification model on an edge device, ensuring reliable performance in real-world conditions. Since the model will be running locally on the device, the focus will be on optimizing its performance for real-time inference and maintaining security while minimizing external dependencies.

3d(ii). Strategy: On-device Inference for Privacy

The model will run locally on the edge device, which enhances privacy by keeping all data processing “on-site” without transmitting video data to external servers or the cloud. This

approach significantly reduces the risk of data breaches and unauthorized access, as no raw or processed data leaves the device. Additionally, on-device inference with a small model reinforces low latency and real-time performance, critical for the responsive interaction required by the pose classification system.

3e. Monitoring and Maintenance

3e(i). Goal

The goal is to continuously monitor the on-device performance of the AI model, detect any biases or performance degradation, and address model drift in real-time to ensure reliable operation in real-world conditions.

3e(ii). Strategy: Performance monitoring

Performance monitoring will be executed on-device to ensure that the model functions optimally in real-world conditions without external oversight. This approach allows continuous monitoring of key metrics like inference time, resource consumption, and frame rates, while ensuring privacy and security by keeping all data processing local.

3e(iii). Technical Implementation: TFProfiler

TFProfiler will be used to monitor the POSE model's performance on the edge device. It provides real-time metrics such as **inference time**, **FPS**, and **memory usage**, enabling a clear understanding of the model's behavior during execution. With profiling across different delegates (CPU, GPU, NNAPI), adjustments to optimize performance based on the hardware used can be made. Additionally, logs will be stored locally on the device, and alerts can be configured to trigger if performance drops below acceptable thresholds.

4. Human-Computer Interaction (HCI)

4a. Step 1: Defining HCI Requirements

4a(i). Understanding user requirements

Strategy: Semi-structured interviews

Conducting semi-structured interviews is a direct way to identify and comprehend the needs of my project's potential users. In particular, I would interview users to see if there is a demonstrable need for the application (i.e., an application which measures how consistently you pose properly during a speech, and perhaps enables you to practice certain poses). I would also

ask about the set of expectations a user might have for my application (e.g., features, mechanisms of interaction, etc.).

4a(ii). Creating personas and scenarios

Strategy: Role-playing exercises

As declared in the stakeholders section: *"The primary users of POSE include accessibility-focused product designers and interface developers…"* I would create a persona for a member of each of these groups, and predict how they would make use of the application. Role-playing exercises will help illustrate more nuanced use cases and baselines for using and *trusting* such an application. For example, there could be a student of theater who would want to use POSE to practice for a role, or a student of dance who would use it in preparation for a recital.

4a(iii). Conducting task analysis

Strategy: Hierarchical Task Analysis (HTA)

I intend to make the user interface for my application quite simple in order to minimize confusion and maximize accessibility, so it would be pragmatic to map the interaction flow out using hierarchical task analysis (HTA). Notably, this would allow me to identify and isolate potential (and/or proven) trouble spots and iterate on improving them. An example would be the user login and/or authentication flow (e.g., entering username & password, or signing in with SSO), which should be seamless and clear.

4a(iv). Identifying accessibility requirements

Strategy: Audit accessibility using Axe & incorporate accessibility features

A tool for systematically auditing accessibility is useful because it provides objective, traceable, and understandable metrics. Axe provides developer tools for several platforms, which will be very useful to evaluate both an administrator-facing "dashboard" view as well a user-facing view (e.g., mobile application with camera access that has ML model running live inference). In addition (and fundamentally), attention will be directed toward designing and implementing with accessibility in mind, so that features such as dark mode, screen reader compatibility, and more are integrated.

4a(v). Outlining usability goals

Strategy: Evaluate usability with Mixpanel and define clear goals

Similarly to the accessibility requirements, a systematic way to evaluate the application's usability will be a great reference while building the application. Mixpanel also offers a product

which integrates well with many other tools (e.g., Figma for design and prototyping) which will be useful when creating both administrator and user interfaces (as described above). Clear goals such as minimizing user error will be identified and pursued with practical means, such as building with interface components that contain input validation mechanisms.

5. Risk Management Strategy

5a. Stage 1: Problem Definition

5a(i). Key risks

- **Misalignment with Objective**s: There is a risk that the POSE project's goals might not fully align with the needs and expectations of its stakeholders. If the objectives are not adequately aligned, the system may fail to deliver value to end users and other key stakeholders.
- **Ethical Risks**: POSE could potentially be misused or lead to unintended societal impacts. For example, it might be used in surveillance contexts that infringe upon personal privacy. Also, the risk of reinforcing social biases, such as those related to body language interpretation, could affect system fairness.
- **Bias in Problem Framing**: The way the problem is defined could unintentionally embed bias into the solution. For example, framing the system solely around specific poses might overlook diverse use cases, body types, or disabilities such as missing limbs.
- **Undefined Success Metrics**: Without clear success metrics, there is a risk that the project may lack measurable benchmarks to assess the effectiveness of pose recognition. This could lead to ambiguous results or misguided project adjustments.
- **Stakeholder Exclusion**: If marginalized groups or key user representatives are excluded from the initial project framing, the system might fail to address their unique accessibility and usability needs; this is related to a bias in problem framing.
- **Legal and Regulatory Compliance**: Non-compliance with privacy and data protection laws, like GDPR, may expose the project to legal risks, especially since the system processes video data.

5a(ii). Mitigation strategy: Stakeholder Engagement

See section 3a(ii). This strategy involves conducting interviews, surveys, and feedback sessions with relevant stakeholders to ensure alignment with needs and address potential ethical and regulatory concerns.

5b. Stage 2: Data Collection

5b(i). Key risks

- **Data Quality**: Poor-quality data, such as noisy or incomplete samples, can lead to inaccurate or unreliable pose classifications, impacting the model's effectiveness.
- **Bias in Data**: Training data that lacks diversity in terms of body types, age, gender, and ethnicity could result in a model that performs inconsistently across different demographic groups, affecting fairness and accessibility.
- **Data Privacy**: Since the project involves image data collection, there is a risk of mishandling personal data, leading to privacy violations.
- **Data Representativeness**: Insufficiently representative data may limit the model's ability to generalize effectively to real-world environments, particularly under varying conditions such as lighting and camera angles.

5b(ii). Technical mitigation strategy: Automated Data Cleaning

Using libraries to preprocess image frames in order to enhance consistency in pose representations. Namely, pose-centric cropping can isolate and crop around the target pose areas. In principle, this targeted preprocessing would reduce noise from background elements, ensuring that only essential information contributes to the model, thereby improving accuracy.

5c. Stage 3: AI Model Development

5c(i). Key risks

- **Algorithmic Bias**: The model might develop biased patterns, which could lead to unfair or inaccurate predictions for certain demographic groups (e.g., body types or skin tones), compromising fairness and usability.
- **Explainability**: As the model complexity grows, it may become a "black box," making it challenging for stakeholders to understand how specific pose classifications are made.
- **Overfitting/Underfitting**: Overfitting on the training data could cause the model to perform poorly on new, unseen data, while underfitting might lead to a model that cannot capture the essential features of different poses, reducing its effectiveness.

5b(ii). Technical mitigation strategy: Explainability tools (SHAP)

See sections 3c(ii) and 3c(iii). SHAP (SHapley Additive exPlanations) will be employed to enhance the explainability of the pose classification model by illustrating the contribution of various body joint positions to its predictions.

5d. Stage 4: AI Deployment

5d(i). Key risks

- **Integration Issues**: Challenges in integrating the model with existing systems on the edge device (e.g., Raspberry Pi) could result in suboptimal performance or functionality issues, impacting user experience.
- **Security Breaches**: The deployment environment may be exposed to security threats, including unauthorized access to the device or data interception, which could compromise the privacy and security of user data.
- **Environmental Variability**: Variations in lighting, background, or camera angles during real-world use may affect the model's accuracy, leading to inconsistent pose recognition.

5b(ii). Mitigation strategy: Integration Testing

Perform extensive testing to ensure smooth integration of the AI model with the deployment environment, focusing on achieving stable and responsive real-time performance on the target edge device.

5e. Stage 5: Monitoring and Maintenance

5e(i). Key risks

- **Model Drift**: Over time, the model's performance may degrade due to changes in user behavior or environmental conditions, leading to reduced accuracy in pose classification.
- **Emerging Security Threats**: New vulnerabilities may arise post-deployment, posing risks to data privacy and the integrity of the edge device.
- **Performance Degradation**: The model's inference speed or accuracy may decrease over time, impacting real-time responsiveness and user satisfaction.

5b(ii). Technical mitigation strategy: Performance monitoring with TFProfiler

See sections 3e(ii) and 3e(iii). TFProfiler will monitor the POSE model's performance on the edge device, offering real-time metrics like inference time, FPS, and memory usage. This tool enables detailed insights into the model's behavior throughout execution.

5f. Residual Risk Assessment

	Impact			
	Acceptable	*Tolerable*	*Unacceptable*	*Intolerable*
	Little or No Effect	*Effects are Felt but Not Critical*	*Serious Impact to Course of Action and Outcome*	*Could Result in Disasters*

Likelihood	*Improbable*	*Risk Unlikely to Occur*		Model Drift	Security Vulnerabilities	Data Privacy Issues, User Safety, Regulatory Compliance
	Possible	*Risk Will Likely Occur*		Algorithmic Bias, Unintended Behavior, Edge Device Limitations		
	Probable	*RIsk Will Occur*				

5f(i). Algorithmic Bias

Mitigation actions include:

- **Diversify Training Data**: Expand the dataset to include a wide range of body types, abilities, ethnicities, and ages to ensure the model performs well across all user groups.
- **Continuous Monitoring**: Regularly monitor the model's performance across different demographics to detect and address any emerging biases.
- **User Feedback Mechanism**: Enable users to report misclassifications or biases, facilitating timely adjustments.

The risk is acceptable with monitoring and implementation of the strategies above.

5f(ii). Model Drift

Mitigation actions include:

- **Scheduled Retraining**: Establish a routine schedule for model retraining using new data to adapt to changes in user behavior and environment.
- **Performance Monitoring**: Implement tools to monitor the model's accuracy and performance metrics over time.
- **Feedback Integration**: Incorporate user feedback to identify areas where the model may be drifting and require adjustments.

The risk is acceptable with minimal action; drift is unlikely but should be monitored over time.

5f(iii). Data Privacy Issues

Mitigation actions include:

- **Data Minimization**: Collect only the data necessary for the functionality of the system, reducing the exposure of personal information.
- **Anonymization and Encryption**: Ensure all collected data is anonymized and securely encrypted both in transit and at rest.
- **Regulatory Compliance**: Comply with all relevant data protection laws and regulations, such as GDPR or local equivalents.

The risk is unacceptable, so all mitigation actions should be undertaken.

5f(iv). Edge Device Limitations

Mitigation actions include:

- **Model Optimization**: Use techniques like model pruning, quantization, and hardware acceleration to optimize the model for edge devices.
- **Compatibility Testing**: Test the system on a variety of edge devices (or emulators) to ensure consistent performance and identify limitations.
- **User Guidance**: Provide users with clear information about the minimum hardware requirements and optimal operating conditions.

The risk is acceptable with the monitoring and implementation of the strategies above.

5f(v). Security Vulnerabilities

The only mitigation actions within the scope of the project includes:

- **Security Protocols**: Implement robust security measures, including secure boot processes, encrypted storage, and authentication mechanisms.
    - This is reliant on the cloud systems used for storage of training data (Google Cloud), and for model training (HiPerGator).
- **Access Controls**: Limit user permissions and network access to only what is necessary for operation.

The risk is acceptable with the monitoring and implementation of the strategies above.

5f(vi). User Safety

Mitigation actions include:

- **Error Handling**: Design the system to handle errors gracefully without compromising user safety.
- **User Education**: Provide comprehensive instructions and warnings about the system's capabilities and limitations.

The risk is unacceptable, so all mitigation actions should be undertaken.

5f(vii). Regulatory Compliance

Mitigation actions include:

- **Documentation**: Maintain thorough documentation of compliance efforts, including policies, procedures, and training materials.
- **Stakeholder Engagement**: Communicate with regulatory bodies and stakeholders to stay informed about changes in legislation.
    - (While appropriate, this is beyond the scope of the project as an academic assignment.)

The risk is unacceptable, so all mitigation actions should be undertaken.

5f(viii). Unintended Behavior

Mitigation actions include:

- **Model Improvement**: Continuously refine the model through additional training, testing, and validation to enhance accuracy.
- **User Feedback Loop**: Implement mechanisms for users to report issues, enabling rapid response to unintended behaviors.
- **Clear Communication**: Provide users with detailed guidance on how to use the system effectively and what to expect from its performance.
- **Fail-Safe Defaults**: Design the system to default to safe states in cases of uncertainty or error.

The risk is acceptable with the monitoring and implementation of the strategies above.

6. Data Collection Management and Report

6a. Area 1: Data Type

POSE manages two types of data: Unstructured Data and Processed (Structured) Data.

6a(i). Unstructured Data

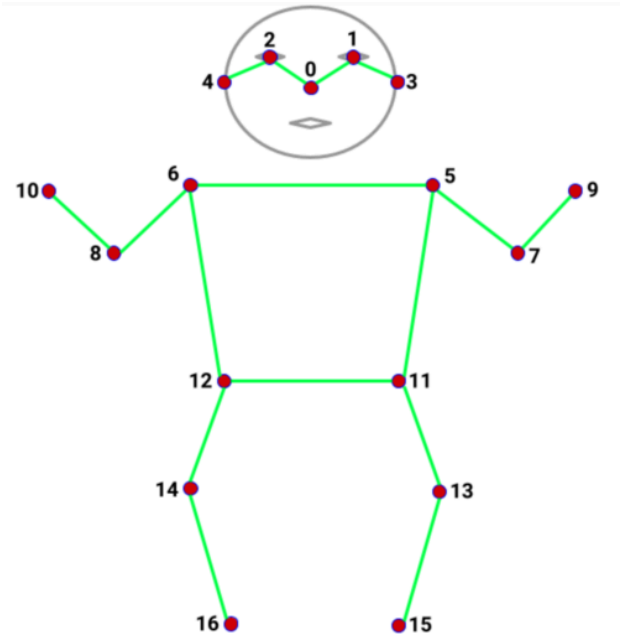
- **Type**: Images
- **Description**: This data consists of photos capturing various poses of individuals in real-world settings. Images depict poses such as “hands raised” or “arms crossed,” which are essential for training the model to recognize gestures linked to accessible user interactions.
- **Challenges**: Handling unstructured image data requires attention to consistency in resolution, lighting, and viewpoint. Variations across these dimensions can affect model training outcomes and the overall accuracy of pose classification. Additionally, ensuring

diverse representation in terms of body types, gender, age, and ethnic backgrounds is critical to prevent demographic biases.

- **Adjustments**: Standard preprocessing techniques, including resizing and cropping, are applied to images to ensure uniform input dimensions. Data is also attempted to be sourced from diverse places in order to capture a wide range of demographic features.

6a(ii). Processed (Structured) Data

- **Type**: Tabular Data
- **Description**: Structured data used for transfer learning consists of labeled joint positions, derived as output from the pose estimation model (MoveNet). This data provides a structured numerical representation of each detected pose, including x, y coordinates for key joints (e.g., wrist, elbow, shoulder).



"COCO Keypoints: Used in MoveNet and PoseNet"

0: nose
1: left_eye
2: right_eye
3: left_ear
4: right_ear
5: left_shoulder
6: right_shoulder
7: left_elbow
8: right_elbow
9: left_wrist
10: right_wrist

11: left_hip
12: right_hip
13: left_knee
14: right_knee
15: left_ankle
16: right_ankle

- **Data Granularity**: Each row in the structured data corresponds to an individual frame, with columns representing the coordinates of detected joints, labeled according to the corresponding pose class.
- **Challenges**: The structured nature of joint data requires alignment across frames, especially when using time-series data for gesture classification. Ensuring consistent labeling and handling missing data points (e.g., undetected joints due to occlusion or body part placement) are also significant challenges.
- **Adjustments**: Preprocessing occurs with a set accuracy threshold, meaning that images which yield joint estimates with low confidence are not used to train the pose classification model.

6b. Area 2: Data Collection Methods

6b(i). Sources of Data

- **Public Photo Sharing Platforms**: Images were sourced from Pexels, Unsplash, and Pixabay, platforms offering generous free licenses for both commercial and non-commercial use. These services provide high-quality images with diverse subjects, which enhances the dataset's demographic representation and variety in pose scenarios.
- **Personal Camera Input**: Additional images plan to be captured using a laptop webcam to expand the dataset. This will include images of the project lead and volunteers who provide informed consent posing in various key postures to ensure adequate sample sizes for each target pose.

6b(ii). Methodologies Applied

- **Manual Collection and Curation**: Each image from the public photo sharing platforms was chosen and reviewed to align with the model's requirements. Specific instructions and care plans to be taken to ensure the results of the personal camera input are valid and usable. This approach allows for control over the diversity and pose variety in the dataset, ensuring the inclusion of a wide range of demographic groups and specific postures that are important for accurate model training.

6b(iii). Ingestion for Training

Data ingestion for training is managed by integrating Google Cloud Storage with TensorFlow's DataLoader pipeline. Initially, the **google.cloud.storage** library is used to download images from the designated Google Cloud Storage bucket to a local directory. Once downloaded, these images are ingested using TensorFlow, which enables efficient batch processing and data augmentation as needed.

6b(iv). Ingestion for Deployment

In deployment, the model will ingest data through direct camera input on the edge device, capturing real-time video frames for immediate pose classification. Each frame is processed locally on the device using LiteRT (TensorFlow Lite), enabling efficient, low-latency inference without relying on network connectivity.

6c. Area 3: Data Collection Methods

6c(i). Applicable Laws and Standards

Since the dataset includes images of individuals, considerations for privacy and data protection were addressed in line with GDPR guidelines for handling personal data. Given that images were sourced from platforms (Pexels, Unsplash, and Pixabay) that provide free licenses permitting commercial and non-commercial use, each platform's licensing terms were carefully reviewed to ensure lawful usage. Additionally, informed consent plans to be obtained from personal acquaintances participating in supplemental photo sessions to further support ethical compliance.

6c(ii). Compliance Strategy and Results

The security of the dataset is paramount and taken care of by being hosted and securely managed within Google Cloud Storage, which offers compliance with ISO/IEC 27001 information security standards. Anonymization is ensured given that the images from public photo-sharing platforms do not contain names or other personally identifying information of individuals, and this practice extends to photos taken of acquaintances for additional poses. As a result, no significant challenges arose with ensuring data security and anonymization.

6d. Area 4: Data Ownership and Access Rights

6d(i). Ownership and Access Control

Data ownership is clearly defined to align with both project needs and compliance requirements. Images sourced from public platforms remain under the ownership of those platforms or their respective contributors. However, the POSE project team has the legal right to use these images as per the platforms' free-use licenses. Similarly, any personal photos contributed by

acquaintances are used with explicit consent and remain the property of the individuals, with access granted for training purposes exclusively within this project.

To ensure secure and controlled access, all data is stored in Google Cloud Storage, where access is restricted to authorized personnel involved in the POSE project. Permissions are carefully managed through Google Cloud IAM (Identity and Access Management), with access logs maintained to audit usage and ensure accountability.

6d(ii). Lessons Learned

Throughout the data collection and storage process, implementing clear access controls has proven critical for safeguarding data privacy. The use of Google Cloud's IAM and regular access audits has strengthened data security and highlighted best practices for managing permissions.

6e. Area 5: Metadata Management

6e(i). Metadata Content and Management System

Key metadata attributes include data source, timestamp, pose labels, file format, and image resolution. Each image file, whether from public platforms or personal contributions, is tagged with relevant metadata, providing transparency and traceability throughout the dataset.

To efficiently handle metadata, Google Cloud Storage's metadata labeling features are used, allowing for structured tracking and retrieval of data. This approach minimizes errors and supports organized access across project stages, ensuring consistency and clarity as the data moves through preprocessing and training. The challenge of receiving inconsistent metadata given the use of multiple sources was anticipated and handled efficiently during the manual data sourcing/ingestion process.

6f. Area 6: Data Versioning

6f(i). Version Control System and Strategy

Data versioning is supported through object versioning in Google Cloud Storage. When object versioning is enabled on a bucket, previous versions of objects are preserved, allowing for data recovery and historical tracking without reorganizing folder structures. Each version is assigned a unique identifier, ensuring transparency and traceability of changes at the object level.

6g. Area 7: Data Preprocessing, Augmentation, and Synthesis

6g(i). Preprocessing Techniques

Data preprocessing centers on pose-centric cropping to highlight the human subject in each image. This technique refines the dataset by focusing on the central figure, but handling varied image resolutions presents a challenge. To standardize cropping, preprocessing assumes the subject's central position, though accuracy sometimes requires manual adjustment. Another key preprocessing step addresses low-confidence detections; joints with confidence scores below a defined threshold are excluded, filtering outliers and enhancing data quality for training the MoveNet model's smaller variants.

6g(ii). Data Augmentation

Although data augmentation was not initially employed, it is planned to boost sample diversity and improve model robustness. Anticipated augmentation techniques will include rotation, flipping, and slight scaling, which add variability while preserving essential pose details. These transformations will avoid visibility-reducing adjustments, such as blur or opacity, to maintain pose clarity. The increased dataset size and diversity from these transformations are expected to significantly improve the model's generalizability across real-world scenarios.

6g(iii). Data Synthesis

No synthetic data generation is used, as direct collection and planned augmentation are deemed sufficient for producing the needed dataset diversity.

6h. Area 8: Data Management Risks and Mitigation

- **Privacy Breaches**: Since images contain identifiable human subjects, there is a risk of privacy breaches. Mitigation involves using images solely from publicly licensed platforms and obtaining explicit consent from volunteers. Anonymization is enforced by ensuring no names or personal identifiers are included. This strategy has effectively minimized privacy risks.
- **Data Corruption**: Data integrity is essential, especially given reliance on high-quality images for accurate pose classification. Mitigation includes integrity checks in Google Cloud Storage, which verifies file integrity upon upload and has proven effective.
- **Bias in Data Representation**: A lack of demographic diversity could introduce bias, impacting model performance. Mitigation includes curated image selection from diverse sources and demographics. While this has enhanced representation, the manual data curation yielded a small dataset; augmentation is planned to increase the dataset size while still using representative data.

- **Data Loss**: Unintended data loss during storage or transfer could impact project continuity. Mitigation employs versioning in Google Cloud Storage, enabling recovery of previous file versions, which has been very effective.

6i. Area 9: Data Management Trustworthiness and Mitigation

- **Data Minimization**: Only essential pose-related data is collected which aids in minimizing privacy risks and improving data processing efficiency. However, this has also resulted in a small dataset; planned data augmentation will address this concern while still maintaining trustworthiness.
- **Data Integrity and Verification**: To maintain data trustworthiness, integrity checks are applied within Google Cloud Storage during uploads, with versioning enabled for easy recovery of previous data states if needed.
- **Transparency in Data Sources**: Using licensed platforms (Pexels, Unsplash, Pixabay) and securing consent for personal images builds transparency. This strategy has effectively supported lawful, ethical sourcing, and maintaining thorough documentation of all updates will enhance data provenance clarity.

7. Model Development and Evaluation

7a. Model Development

7a(i). Algorithm Selection

The project implements a pose classification model trained via transfer learning on MoveNet (see model card here), a pre-trained convolutional neural network (CNN) which predicts positions of joints for an image of a person.

The pose classifier will use the 17 joint positions outputted by MoveNet as structured data for training to ultimately classify poses such as "hands raised" or "arms crossed".

7a(ii). Feature Engineering and Selection

MoveNet - by nature of its underlying CNN architecture - learns features from images automatically. In other words, it captures patterns directly from the training data (which has been preprocessed with cropping/resizing and augmentation). For example, during training MoveNet may learn patterns may include edges or shapes like the outline of a head or the contour of an arm.

MoveNet outputs joint positions as keypoints; these are termed "landmarks" and represent the initial features for the pose classifier. These landmarks are then normalized in order to be invariant to scale and orientation. Then, they are processed into pose embeddings (processed features) for the pose classifier to use.

7a(iii). Model Complexity and Architecture

Pose Estimation:
Step 0: MoveNet performs pose estimation on raw images and outputs a set of keypoints with joint coordinates and confidence scores. These are then normalized to be invariant to scale and orientation.

This "initial step" obtains the input data for the pose classifier and will be combined into a single model with pipelines.

Pose Classification:
Step 1: *InputLayer* - receives preprocessed and normalized input data (joint coordinates & confidence scores, a.k.a. "landmarks").
Step 2: *LandmarksToEmbeddingLayer* - input data is converted into pose embeddings.
Step 3: *DenseLayer* - 128 neurons, relu6 activation function; introduces nonlinearity into the model and allows it to learn more complex patterns.
Step 4: *DropoutLayer* - rate of 0.5, randomly setting 50% of its inputs to 0 at each update to prevent overfitting.
Step 5: *DenseLayer* - 64 neurons, relu6 activation function; further processing data.
Step 6: *DropoutLayer* - rate of 0.5, similar to dropout layer above, used for further regularization.
Step 7: *DenseLayer* - 8 neurons (number of classes), softmax activation converts output into probabilities with each neuron representing a class; highest probability is selected as predicted pose, such as "hands up" or "arms crossed".

7b. Model Training

7b(i). Training Process

Prior to training, data augmentation techniques are applied to the training dataset. They include slight rotations, scaling transformations, and horizontal flips. These modifications simulate real-world variability in pose data, helping the model recognize poses accurately even when presented with novel variations.

The training process itself includes:

- **Large Batch Sizes and Epochs**: Large batch sizes (64 to 128) to speed up convergence while ensuring stability in gradient updates. The use of 100 to 200 epochs ensures the model is sufficiently exposed to varied pose samples, enabling it to generalize across different scenarios and users.
- **Adam Optimization**: The Adam optimizer is chosen for its adaptive learning rate and efficient handling of sparse gradients. An initial learning rate of 0.001 is applied, which balances quick convergence while minimizing the risk of overshooting the optimal point.

- **Dropout Layers for Regularization**: Two dropout layers, each with a rate of 0.5, are placed after the dense layers. This regularization technique reduces overfitting by randomly setting 50% of the inputs to zero during training, compelling the model to learn distributed representations and depend less on any single neuron.

7b(ii). Hyperparameter Tuning

Hyperparameter tuning (of learning rate and joint confidence threshold) will be used alongside cross-validation so that a final best-performing model can be identified and used.

- **Learning Rate**: This hyperparameter determines how much the model's weights are updated during each iteration of training. A grid search approach will be used to explore a range of learning rates (e.g., 0.0001 to 0.01). This tuning ensures the model can efficiently converge without encountering issues like overshooting the optimal point or converging too slowly.
- **Joint Confidence Threshold**: This hyperparameter determines the minimum confidence level required for keypoints to be considered valid. This threshold will be adjusted to balance the trade-off between accuracy and noise tolerance. For example, a higher threshold may reduce false positives by excluding uncertain joint detections, while a lower threshold could increase recall by allowing more keypoints to be included. A range (e.g., 0.5 to 0.9) will be tested to find the optimal threshold that maximizes classification accuracy.

7c. Model Evaluation

7c(i). Performance Metrics

Performance metrics can be considered in relation to two elements: pose **estimation** and pose **classification**.

A pre-trained pose estimation model (MoveNet) is used to train the pose classification model. MoveNet's developers have already released metrics and quantitative analyses of their model based on keypoint "mean average precision (mAP)" (see model card here).

Given that no expansion is made on MoveNet's pose estimation mechanisms and that there is no reliable nor scientific way, in the scope of this project, to obtain a ground truth with which its performance may be evaluated, the existing performance metrics obtained by its developers will be reported.

For the pose classification model, its performance will be directly evaluated in terms of accuracy using precision and recall.

- **Precision** measures the proportion of correctly predicted positive poses out of all poses predicted as positive. This will show how accurate the positive predictions are.

- **Recall** measures the proportion of actual positive poses that were correctly identified. This will show how well the model captures all true positives.

7c(ii). Cross-Validation

K-fold cross validation will be implemented during the training of the pose classification model.

The data will be split into k subsets (folds); for each of the k iterations, k - 1 folds will be used for training and 1 fold for validation. This will ensure every data point gets a chance to be in a training set and validation set.

Once all k iterations are completed, performance metrics will be obtained for each model that was trained. That data will then be averaged in order to get a more robust estimate of the model's performance.

7d. Implementing Trustworthiness and Risk Management in Model Development

7d(i). Risk Management Report

Automated Data Cleaning was the chosen technical mitigation strategy (*see section 5b(ii)*).

This involves pose-centric cropping in order to isolate and crop around the target pose areas in an attempt to reduce noise from background elements in service of accuracy.

Image before preprocessing (left), image after preprocessing (right)

The original image is 4480 x 6720 pixels in dimension and has a lot of unnecessary vertical space. In contrast, the preprocessed image is just 224 x 224 pixels in dimension, and centers the person in the image. Notably, it maintains all of the key joints for the pose (“crossed arms”) to still be recognizable - namely the torso, shoulders, and arms.

Training the model with the cleaned data yielded much better results than training with raw image data. The model trained with unprocessed images yielded an accuracy (on the training set) of 0.44, while the model trained with processed images yielded an accuracy of 0.77.

7d(ii). Trustworthiness Report

SHAP (SHapley Additive exPlanations) was chosen as the technical mitigation strategy (*see section 3c(iii)*). It attempts to explain how different body keypoints influence model predictions. The shap library provides visualization tools such as force plots, beeswarm plots, and waterfall plots to illustrate feature importance for individual predictions. KernelExplainer was used due to difficulties with getting the DeepExplainer to work. That said, it should be possible to investigate

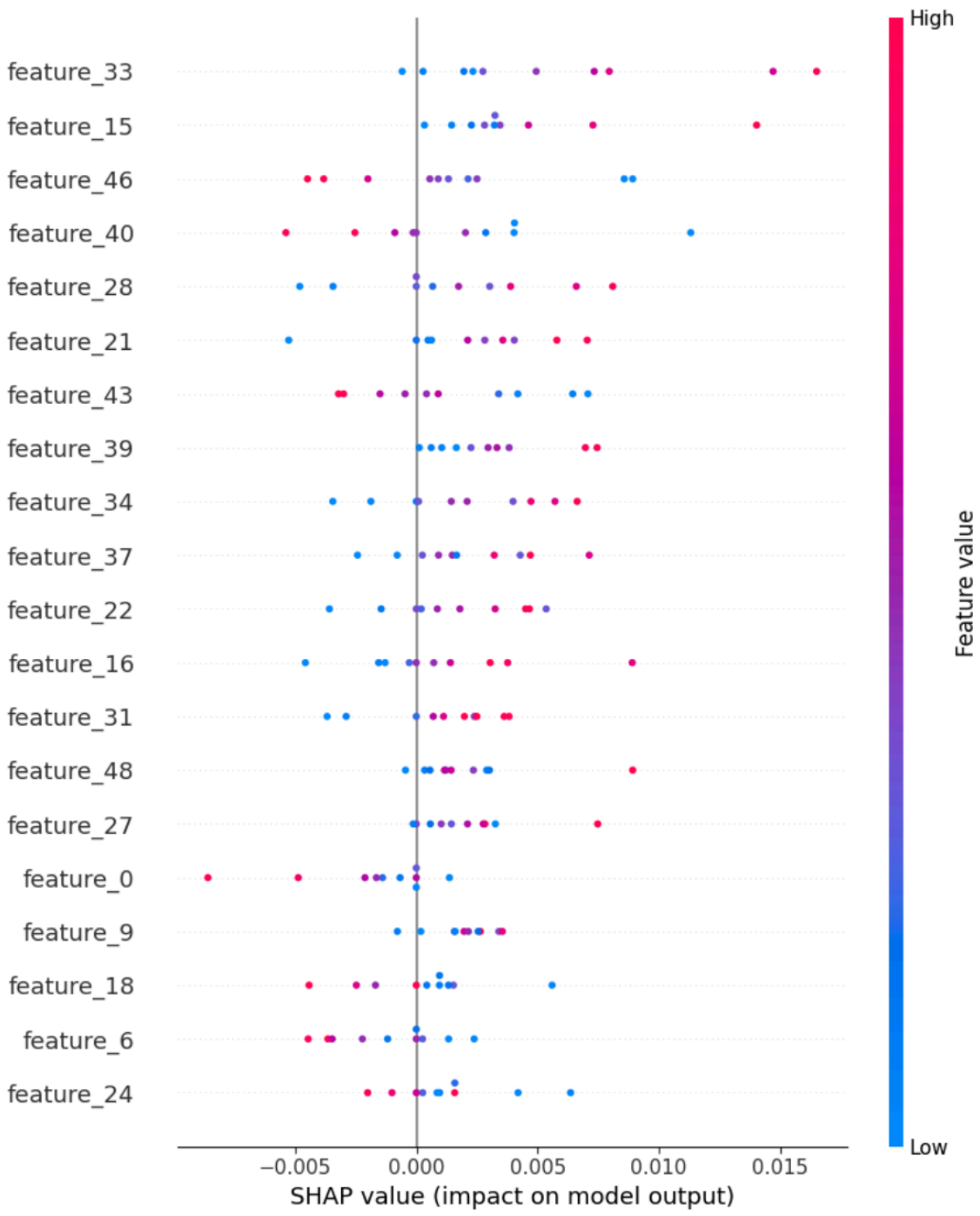
and resolve these challenges, since DeepExplainer offers more specialized capabilities for deep learning models.

The SHAP decision plot and summary plot (*see below*) collectively provide a detailed view of feature importance and their contributions to the model's predictions. Important features, such as feature_33, feature_15, and feature_46, are shown as significant across both plots, indicating that they play a major role in influencing model output.

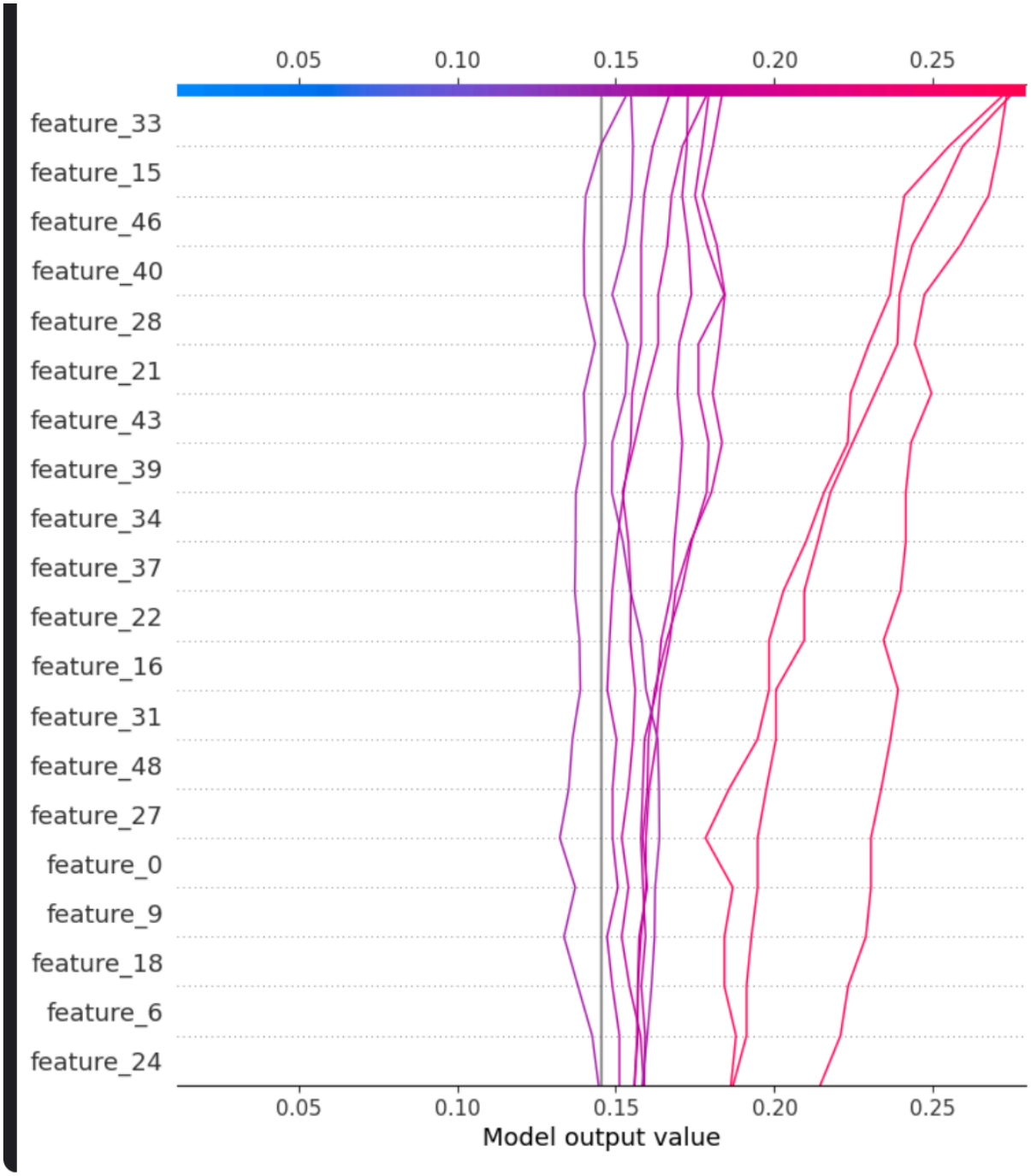
- The summary plot displays the overall distribution and impact of features across multiple samples, revealing variability in their importance and the direction of influence.
- The decision plot highlights how individual features cumulatively impact predictions for specific samples, showcasing both positive and negative contributions.

The color gradient used in both plots—where red represents high feature values and blue represents low values—helps identify whether high or low feature values contribute positively or negatively to the model's output.

To enhance interpretability, the next steps involve mapping these generic feature names (e.g., feature_33, feature_15) to more meaningful labels, such as specific body keypoint coordinates or derived pose metrics. A deeper investigation into the most influential features identified in these plots can reveal their relationship with the model's predictions and the underlying data.



SHAP Summary Plot



SHAP Decision Plot

7e. Apply HCI Principles in AI Model Development

7e(i). Develop Interactive Prototypes

Gradio will be used to create an accessible web interface that can be used to directly interact with the model. Specifically, Gradio has the option to implement "Streaming components" (source here) which use a live camera feed as input, which is followed by a prediction which is displayed

on the screen. This output will be labeled with the pose that is predicted along with a confidence score for the sake of transparency. The output will be immediate, which will allow the user to see the model "in action" in real-time.

7e(ii). Design Transparent Interfaces

Gradio will be used to incorporate a transparent user interface as well as feedback opportunities for users. Specifically, the output displayed by the model's inference (using live camera input) will not only include the class (e.g., "hands raised") but also a confidence score (e.g., 0.92) so that the user can gauge how certain the model is of its output.

7e(iii). Create Feedback Mechanisms

Components such as flags (source here) will be implemented which will allow the user to describe and report incorrect model inference, a UI bug/glitch, and/or provide other feedback.

8. Deployment and Testing Management Plan

8a. Deployment Environment Selection

8a(i). Documentation

The deployment environment consists of two primary configurations: an **edge-based** deployment utilizing a Raspberry Pi 5 with a Raspberry Pi Camera Module, and a Gradio-based **web interface** for feedback collection.

The edge deployment ensures that all processing occurs directly on the device, aligning with the project's emphasis on privacy by avoiding the need to send data to external servers. By keeping operations local, the system is also independent of internet connectivity, ensuring reliable performance even in offline environments. The Raspberry Pi 5 was selected due to its affordability, portability, and compatibility with the LiteRT (formerly TensorFlow Lite), which efficiently supports the lightweight pose estimation model (MoveNet) and the classification model. While edge-based deployment typically poses challenges related to limited computing power, this constraint is mitigated here by leveraging small and optimized models designed for real-time inference on resource-constrained devices. As such, edge deployment meets the project's operational goals of privacy, offline functionality, and scalability for hands-free interaction systems.

The minimal web deployment via the Gradio interface is primarily designed for user feedback collection. While Gradio does not store or transmit user data, it processes inputs temporarily for the purpose of demonstrating the model's inference capabilities. This web interface highlights the flexibility of the system for interactive feedback but requires additional considerations for secure hosting and network access to ensure compliance with privacy standards.

8b. Deployment Strategy

8b(i). Documentation

The deployment strategy employs a combination of Docker and TensorFlow Serving for containerization and model deployment, ensuring consistent runtime environments and simplified scaling to additional devices. Docker Compose is utilized to orchestrate the deployment, streamlining the setup and management of multiple services.

Reliability across both edge and web configurations is maintained through integrated monitoring with Grafana and Prometheus. These tools track metrics such as inference latency, frame processing rate, and resource utilization, enabling real-time detection and resolution of performance issues. System-level configurations and fallback mechanisms further enhance operational robustness.

The edge deployment configuration ensures that all data processing occurs locally, on-device. This configuration prioritizes privacy, as no raw or processed images are transmitted or stored externally. The system is also independent of internet connectivity, ensuring reliable performance even in offline environments. To address the challenges of limited computing power on edge devices, lightweight and optimized versions of the pose estimation and classification models are deployed. These models leverage LiteRT for real-time inference, achieving the required performance for hands-free interaction systems. Scalability for edge deployment could be achieved by designing the system to accommodate additional Raspberry Pi devices, which can be monitored synchronously using Grafana and Prometheus. Each device would function independently, enabling decentralized processing while maintaining cohesive system performance in larger deployments.

The web deployment configuration enables feedback collection and remote demonstration capabilities of the developed pose classification model. While not intended for continuous real-world operation, it provides an interactive platform for users to engage with the model and submit feedback directly. The interface operates within a secure Docker environment and relies on temporary data processing, with no data being stored or transmitted beyond the local runtime. While scalable for limited user interaction, the Gradio deployment introduces a reliance on network connectivity for operation. Consequently, this setup is suitable for testing, gathering user feedback, and demoing.

8c. Security and Compliance in Deployment (Trustworthiness and Risk Management)

8c(i). Application

Building upon strategies outlined in earlier sections, deployment prioritizes privacy, data minimization, and adherence to GDPR standards:

- **Device and Network Security**: For edge devices, secure boot processes and encrypted storage are recommended. For the Gradio interface, securing network traffic via HTTPS and restricting access to authorized users are key priorities. These strategies minimize the attack surface for data breaches.
- **Device-Level Protections**: Secure boot processes and encrypted storage mechanisms are recommended for the Raspberry Pi to safeguard against unauthorized access. Additionally, regular updates to the device's firmware and operating system will address emerging vulnerabilities.
- **Web Interface Implications**: The Gradio interface operates within a secure Docker environment. While data processed through the interface is temporary and discarded immediately, secure hosting and adherence to GDPR-compliant practices are essential, especially for public-facing implementations.
- **Monitoring Access Controls**: Administrative access to system health dashboards is restricted through authentication measures in Grafana and Prometheus. Access logs are maintained to ensure accountability and detect unauthorized access attempts.
- **Data Minimization**: As established, only essential pose-related data is used, with no storage of personal or identifiable information. All video frames captured by the Raspberry Pi Camera Module are processed in real-time and discarded immediately after inference.
- **Compliance**: The deployment complies with GDPR standards by adhering to data encryption and anonymization guidelines. No personal identifiable information (PII) is collected during deployment, further reinforcing privacy compliance. Compliance audits would be conducted periodically with reference to access logs and monitoring mechanisms to ensure alignment with regulatory updates and security best practices.

9. Evaluation, Monitoring, and Maintenance Plan

9a. System Evaluation and Monitoring

9a(i). Documentation

POSE is monitored using Prometheus and Grafana, which collectively provide a comprehensive overview of system health and operational metrics. Prometheus collects and aggregates real-time metrics, while Grafana visualizes the data, enabling the identification of trends and potential issues quickly and alerting if a threshold is exceeded.

The monitored metrics include:

- **Inference Latency**: Measures the time taken for the model to process each video frame, ensuring real-time responsiveness.
- **Frame Processing Rate**: Tracks the number of frames processed per second, verifying that the system maintains the required throughput for smooth pose classification.
- **GPU/CPU Utilization**: Monitors resource consumption to detect potential performance bottlenecks or inefficiencies.

Sudden increases in inference latency or drops in frame processing rates are prioritized as the primary indicators of system/model drift or performance degradation. These changes often signal issues such as environmental variability or hardware inefficiencies, which require immediate attention.

For more advanced implementations beyond the demo stage, comparing input data distributions over time to the characteristics of the training dataset can provide deeper insights into drift caused by environmental or input condition changes. While not implemented for this project, this method could complement the system's current monitoring approach in future iterations.

When drift or performance issues are detected, the mitigation strategy involves retraining the model using updated datasets that better reflect current input conditions, ensuring continued reliability and accuracy in real-world applications.

9b. Feedback Collection and Continuous Improvement

9b(i). Documentation

Feedback from users is collected through a built-in basic feedback mechanism implemented via Gradio's interactive user interface. Users can report issues, such as misclassifications or major latency, directly through the interface.

The feedback collected is systematically analyzed to inform system improvements. For instance, if multiple users report consistent misclassification under specific conditions - such as the model favoring the right hand over the left for recognizing a "raised hand" pose - this insight would trigger a review of the training data and model behavior. The model would then be retrained with updated, balanced data to address the issue and subsequently re-released as an improved version.

It is up to the individual or organization implementing POSE to define the threshold for determining whether a collection of user feedback is statistically significant enough to justify system changes. Additionally, they must allocate resources such as time, budget, and workforce for retraining and re-releasing the model based on the scope of the required updates. These

decisions ensure that improvements are both impactful and feasible within the operational context.

9c. Maintenance and Compliance Audits

Building on the strategies outlined in the Deployment and Testing Management Plan and Security, Privacy, and Ethics (Trustworthiness) sections (*see section 8 and section 3 respectively*), the system integrates monitoring, maintenance, and compliance:

- **Maintenance Strategies**: Both the edge deployment and Gradio interface are monitored using Docker-based orchestration tools, with Grafana and Prometheus tracking key metrics such as inference latency, frame processing rates, and resource utilization. Real-time diagnostics enable the timely detection of performance degradation or hardware issues (*See section 8b(i)*).
- **Compliance:** Compliance with GDPR and related standards is upheld by processing data locally (edge devices) or temporarily (Gradio interface), with no storage or transmission of sensitive data. Regular audits verify encryption practices, access controls, and adherence to privacy guidelines (*See section 8c(i)*).
- **Risk Mitigation**: Key risks such as drift, performance degradation, and emerging security vulnerabilities are addressed through:
    - Real-time monitoring tools such as Grafana Alerting (*see section 9a(i) for metrics tracked and drift detection methods*).
    - Secure deployment practices, including encrypted storage and secure boot (*see section 8c(i)*).
    - Retraining workflows triggered by significant performance changes (*see section 9b(i) for user feedback integration*).