

AI-Based Solution for Exercise Posture Correction

Ngo Quoc Bao To Van Duc Pham Thien Nhat

Advisor: M.S.E. Le Dinh Huynh

December 11, 2022



FPT UNIVERSITY

1. Introduction
2. Related Work
 - Human Pose Estimation
 - Exercise Posture Correction
3. Proposed System
 - Pose Estimation Method
 - Pose Evaluation Method
4. Implementation
5. Experiment
6. Software Demonstration
7. Conclusion

Introduction

- People are living a sedentary life.
- Lack of physical activities can result in chronic conditions.
- Public health organizations recommend daily exercises as a countermeasure.

The Importance of Correct Forms

- Exercises have potential injury risks.
- Without proper form, users can put themselves at harm.
- The spine, shoulders, and knees are the most vulnerable.



Figure: Incorrect posture during a deadlift can put lots of pressure on the spine, leading to injuries.

The dilemma:

- People seek guidance from personal trainers.
- Not everyone can afford the services.

Our purposes:

- Build a computer vision-based exercise posture correction system.
- System must be robust, lightweight, and work in real-time.

Related Work

Motion Capture

- Requires the user to wear a suit with markers, fixtures, or sensors.
- Widely used in studios for its precision.
- Costly equipment and complex setup process.

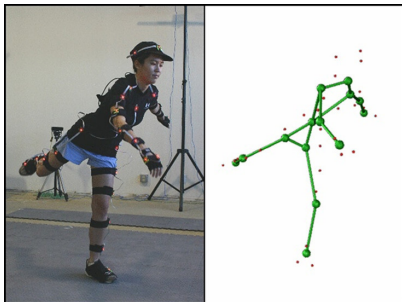
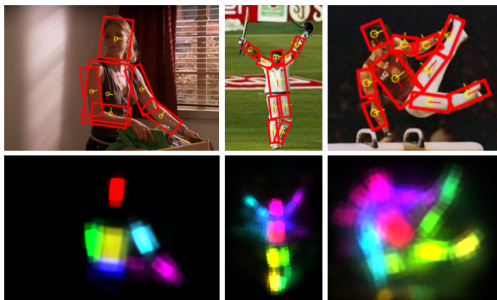


Figure: Pose estimation of a person with optical motion capture technology [1].

Pictorial Structures Framework



- Model spatial correlation of rigid body parts, then predict body joints location.
- Problem with depth ambiguity.

Figure: Pose estimation using PSR [2].

Flexible Model of Part Mixtures

- Captures contextual co-occurrence relations between parts.
- Lacks holistic reasoning.

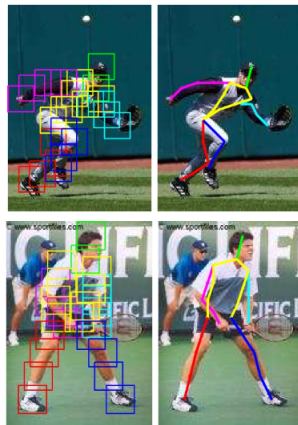


Figure: Pose estimation on the Parse dataset using FMP [3].

- CNN-based regression towards body joints.
- Use AlexNet as the backbone.

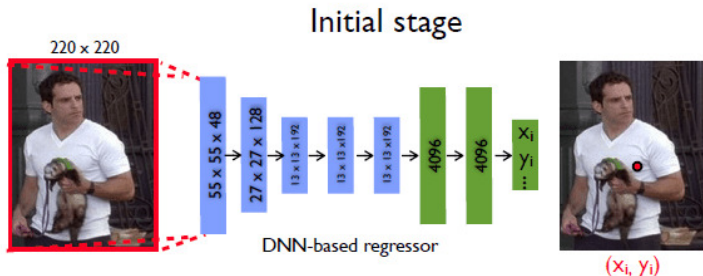


Figure: The architecture of DeepPose [4].

Modern Approaches to HPE

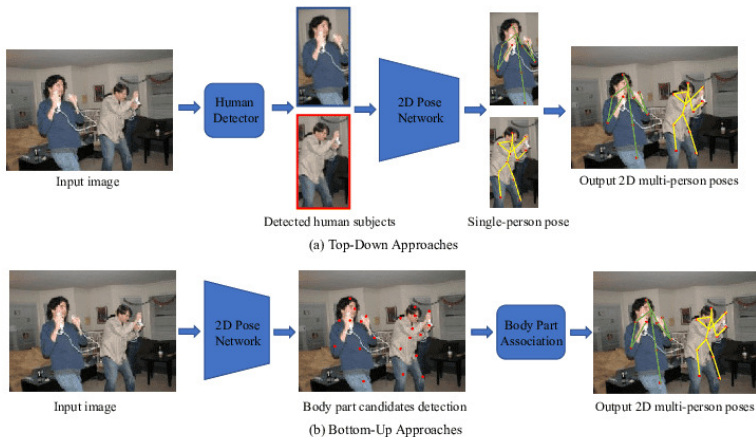


Figure: Two common approaches to human pose estimation [5].

- Follows the Bottom-up approach.
- Utilize Heatmaps and Part Affinity Fields.
- Great scalability.

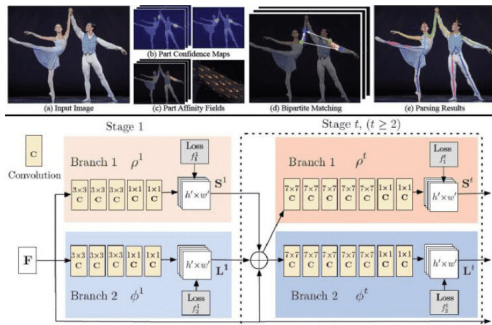


Figure: Proposed architecture of OpenPose [6].

Mask R-CNN

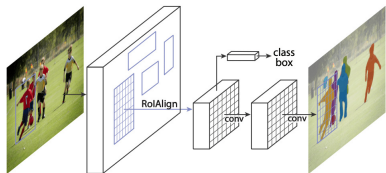


Figure: Mask R-CNN Framework [7].

- Resemble Top-down approaches.
- Estimate pose using human detection and body parts detection.
- Performs both tasks parallelly.

- Follows the top-down approach, but more CNN-oriented.
- Can maintain semantically strong & spatially precise high-resolution representations.

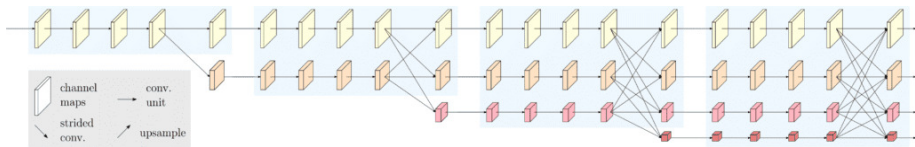


Figure: Proposed architecture of HRNet [8].

- Uses a more efficient ShuffleBlock for the backbone of HRNet.
- Achieved a good accuracy-complexity trade-off.

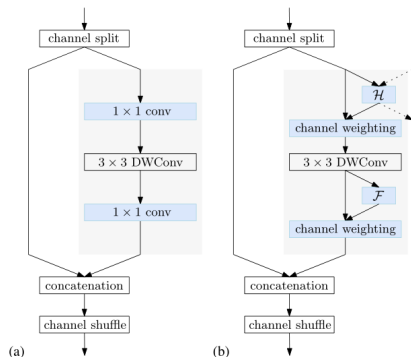


Figure: Building block of Lite-HRNet [9].
(a) The shuffle block. (b) Lite-HRNet conditional channel weighting block.

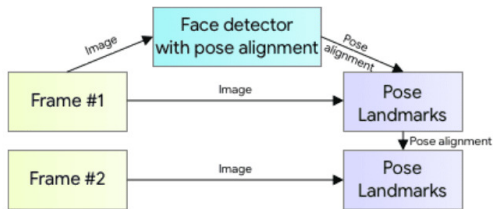


Figure: The inference pipeline proposed in BlazePose [10].

- Follows the bottom-up approach.
- Utilizes a detector-tracker inference pipeline.
- Detector runs until a person is detected, then the tracker take over.
- Limitation: can only be used for single person.

Uses the encoder-decoder architecture with two encoders:

- Encoder 1 predict heat maps for all joints, used when training.
- Encoder 2 regress into joints coordinates, used when inferring.

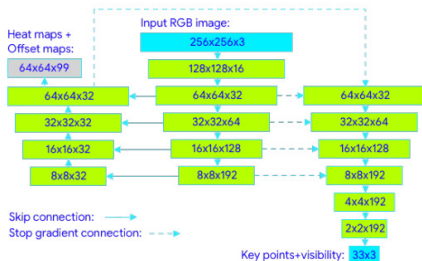


Figure: The proposed Pose estimation network of BlazePose [10].

- Pipeline consists of two main components:
 - Keypoints extraction: OpenPose.
 - Pose evaluation: Geometric/ML.
- Lacks real-time capabilities.
- Requires a complete video sequence.

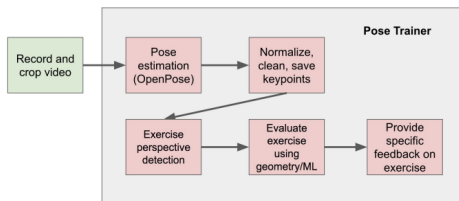


Figure: PoseTrainer's pipeline [11].

- Data-driven.
- Dynamic time warp (DTW) is used to match two sequences.
- Limitation: Can only predict binary labels.

- Uses joint angles to judge the posture.
- Can provide specific feedbacks.

Ohri *et al.* proposed a similar pipeline [12]:

- Replaced OpenPose with BlazePose.
- Recognized two categories of exercises:
 - Low-intensity: Evaluate using predefined angles.
 - High-intensity: Evaluate using DTW.
- Limited in the ability to correct user postures on-the-fly.

Proposed System

Proposed System Overview

Our system consists of two main stages: pose estimation and pose evaluation.

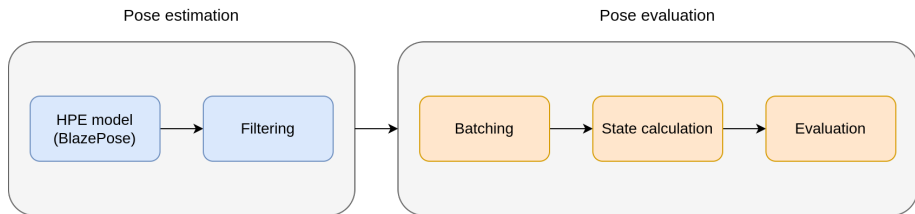


Figure: Proposed system overview.

Pose Estimation Model Selection

BlazePose:

- Lightweight CNN architecture.
- Designed specifically for mobile purposes.
- Perform single-person pose estimation.
- Shipped in Google's Mediapipe.

Model	FPS	AR Dataset PCK@0.2	Yoga Dataset PCK@0.2
OpenPose (body only)	0.4	87.8	83.4
BlazePose Full	10	84.1	84.5
BlazePose Lite	31	79.6	77.6

Table: BlazePose vs OpenPose [10].

Pose Filtering

There are cases where the model can not identify the pose correctly:

- Keypoints are wrong and clump up into a cloud-shaped figure.
- Several keypoints are missing although the body is visible.

Partial solution: employ a preliminary pose-checking procedure.



Figure: Erroneous pose estimation.

- Our proposal: group every N adjacent frames into batches, further process on batch units.
- Two important factors play a role in determining N :
 - Model capability: accuracy, noise, etc.
 - Human movement: what counts as a *singular* movement?

Selecting N

- We select the minimum N that satisfies the constraints.
- N corresponds to time $t \in [0.3, 0.4]$ is practically sound.
- N affects how we register movements.

State Calculation

- **State**: the phase of the exercise being performed.
- Enables the flexibility of the system:
 - In evaluation.
 - Within state calculation itself.
- Can distinguish same pose, different phases.
- Can be as simple as measuring joints displacement. Let $P_i = (x_i, y_i)$, $1 \leq i \leq M$ be the coordinates of M keypoints constructing a pose P ; P^k , $1 \leq k \leq N$ be the poses within a batch. The displacement of an arbitrary keypoint i is:

$$d_i = P_i^N - P_i^1$$

- Use sets of geometric guidelines for each exercise.
- Can give specific guidelines to correct postures.
- Integrates well with other parts of the system.

Implementation

The figure demonstrates the high-level architecture of our software.

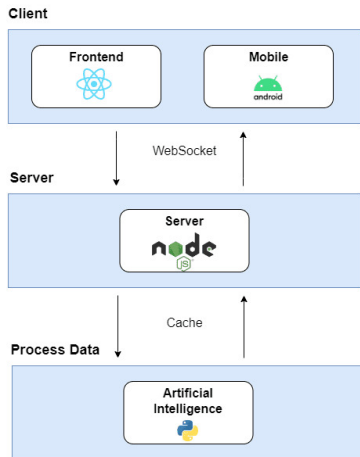


Figure: Architecture overview.

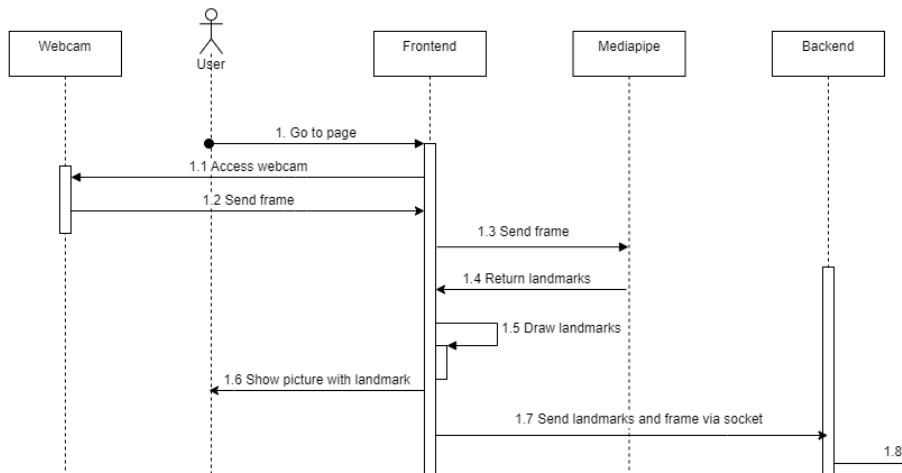


Figure: Frame processing diagram for webcam and Android.

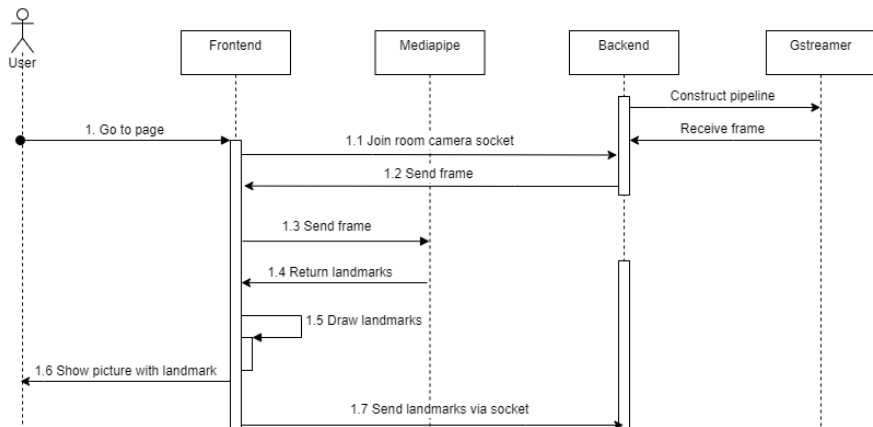


Figure: Frame processing diagram for other camera inputs.

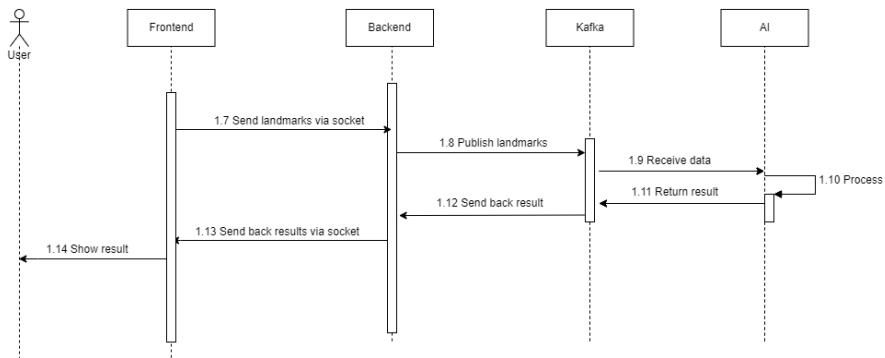


Figure: Landmark communication diagram.

Deployment

1. Docker.



Versatile

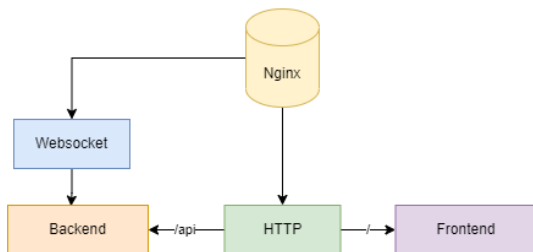
Anyone on the team
can use and run the
services



Cloud

Easy deployment to
cloud service like
Azure, AWS

2. Nginx.



3. Azure:

- 2 vCPUs.
- 4GB RAM.
- 32GB Disk.






B2S Standard	
2	vCPUs
4	GB
	4 Data disks
	3200 Max IOPS
	8 GB Local SSD
	Premium disk support
	Load balancing

Figure: Azure B-Series B2S specifications.

Benchmark

- CPU: 15-25% depends on number of connected users
- Ram: stable at 2.85Gb
- Network: 5Mb/min -> 85Kb/s per user
- Disk: 100Kb/s per user

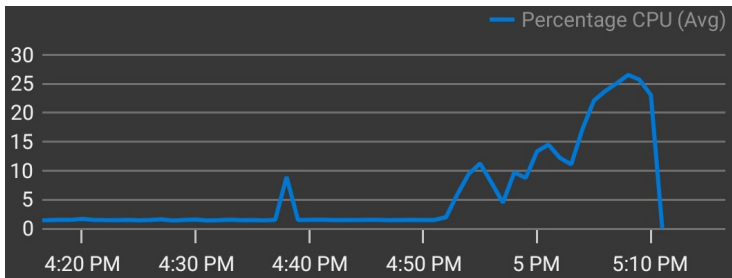


Figure: CPU usage when users are connected.

- BlazePose from Google's Mediapipe.
- Central AI system is implemented in Python as a public package.
- The backend is responsible for pose estimation.

The software and package source code is available at <https://github.com/tokudayo/AIFA>.

Experiment

We fill in details for three exercises: shoulder press, deadlift, and hammer curl.

Example

For the shoulder press, the state calculation procedure:

- Measure wrists' and elbows' displacements.
- Change in direction of motions means *top/bottom* position.

Some guidelines for the evaluation procedure:

- Body must stay straight all the time.
- Forearm must stay parallel to the body.
- Arms should be fully extended at the *top* position.
- Elbows should reach chest level at the *bottom* position.
- *Negative* motion should be expanded across 1.5-3 seconds.

Experimental Setup

We evaluate the performance of the system and benchmark its speed on high-medium-end and low-medium-end machines:

Specification	System A	System B
OS	Ubuntu 22.04	Windows 10
CPU	Ryzen 7 4800H	Intel i7-8565U
RAM	16GB	8GB
Python version	3.9.10	3.8.5

Table: System specifications.

Dataset

- We create a small dataset due to the lack of benchmark data.
- Dataset consists of online videos and our recorded videos.
- Each repetition is counted as one example.
- Ground truth based on whether they conform to the guidelines.



Figure: Our collected dataset.

- We denote *wrong posture* the positive class.
- Metrics we used to measure the system's performance:
 - Accuracy:

$$\text{Acc.} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision:

$$\text{Prec.} = \frac{TP}{TP + FP}$$

- Recall:

$$\text{Recall} = \frac{TP}{TP + FN}$$

The table below summarizes our results.

Exercise	Examples	Accuracy (%)	Precision	Recall
Shoulder press	22	95.5	0.91	1
Deadlift	42	90.4	0.95	0.86
Hammer curl	20	95	1	0.9

Table: Accuracy, precision, and recall of our system on three exercises.

Most common errors are detected by our system, with the exception of several types of tilting.

- Including BlazePose:
 - System A achieved 35 FPS at 720×405 .
 - System B achieved 25 FPS at 720×405 .
- Excluding BlazePose:
 - System A achieved 1350 FPS.
 - System B achieved 600 FPS.

- Inherits the advantages of single-frame and sequence processing methods.
- Ability to give real-time, live evaluation and correction.
- Less prone to noises compared to single-frame methods.
- Robustness from taking temporal information into account.
- Have the possibility to extend to other types of activities other than exercising.

Limitations

- Limited in picking up hard-to-see errors when viewing on a flat surface.
- Requiring a clear, direct view.
- Limited to single-person inference.
- Occasional estimation errors may affect the system.

Software Demonstration

Conclusion

Conclusion

We present a lightweight end-to-end system for real-time exercise posture correction with several advantages over existing HPE-based methods.

We demonstrate the practicality of our work by releasing a package in Python, as well as building applications for web users and Android devices.

- [1] Adam G. Kirk, James F. O'Brien, and David A. Forsyth. "Skeletal Parameter Estimation from Optical Motion Capture Data". In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*. July 2005, pp. 782–788. URL: <http://graphics.cs.berkeley.edu/papers/Kirk-SPE-2005-06/>.
- [2] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. "Pictorial structures revisited: People detection and articulated pose estimation". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. June 2009, pp. 1014–1021. DOI: [10.1109/CVPR.2009.5206754](https://doi.org/10.1109/CVPR.2009.5206754).
- [3] Yi Yang and Deva Ramanan. "Articulated pose estimation with flexible mixtures-of-parts". In: *CVPR 2011*. June 2011, pp. 1385–1392. DOI: [10.1109/CVPR.2011.5995741](https://doi.org/10.1109/CVPR.2011.5995741).

References II

- [4] Alexander Toshev and Christian Szegedy. “DeepPose: Human Pose Estimation via Deep Neural Networks”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. June 2014, pp. 1653–1660. DOI: 10.1109/CVPR.2014.214.
- [5] Qi Dang, Jianqin Yin, Bin Wang, et al. “Deep learning based 2D human pose estimation: A survey”. In: *Tsinghua Science and Technology* 24.6 (Dec. 2019), pp. 663–676. ISSN: 1007-0214. DOI: 10.26599/TST.2018.9010100.
- [6] Zhe Cao, Gines Hidalgo, Tomas Simon, et al. *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. 2018. DOI: 10.48550/ARXIV.1812.08008. URL: <https://arxiv.org/abs/1812.08008>.
- [7] Kaiming He, Georgia Gkioxari, Piotr Dollár, et al. *Mask R-CNN*. 2017. DOI: 10.48550/ARXIV.1703.06870. URL: <https://arxiv.org/abs/1703.06870>.

- [8] Jingdong Wang, Ke Sun, Tianheng Cheng, et al. *Deep High-Resolution Representation Learning for Visual Recognition*. 2019. DOI: 10.48550/ARXIV.1908.07919. URL: <https://arxiv.org/abs/1908.07919>.
- [9] Changqian Yu, Bin Xiao, Changxin Gao, et al. *Lite-HRNet: A Lightweight High-Resolution Network*. 2021. DOI: 10.48550/ARXIV.2104.06403. URL: <https://arxiv.org/abs/2104.06403>.
- [10] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, et al. *BlazePose: On-device Real-time Body Pose tracking*. 2020. DOI: 10.48550/ARXIV.2006.10204. URL: <https://arxiv.org/abs/2006.10204>.
- [11] Steven Chen and Richard R. Yang. *Pose Trainer: Correcting Exercise Posture using Pose Estimation*. DOI: 10.48550/ARXIV.2006.11718. URL: <https://arxiv.org/abs/2006.11718>.

- [12] Ashish Ohri, Shashank Agrawal, and Garima S. Chaudhary. “On-device Realtime Pose Estimation & Correction”. In: *International Journal of Advances in Engineering and Management* 3 (2021). DOI: 10.35629/5252-030716911696. URL: https://ijaem.net/issue_dcp/0n%5C%20device%5C%20Realtime%5C%20Pose%5C%20Estimation%5C%20and%5C%20Correction.pdf.

Thanks