Developing an Android Application for Indoor Control Inspections of Construction Progress

Dmitry Egorov

Supervisor
Aleksey D. Shadrikov
Senior Specialist
AI Track Curator at Samsung Innovation Campus

HSE University / 29 June 2024

Introduction

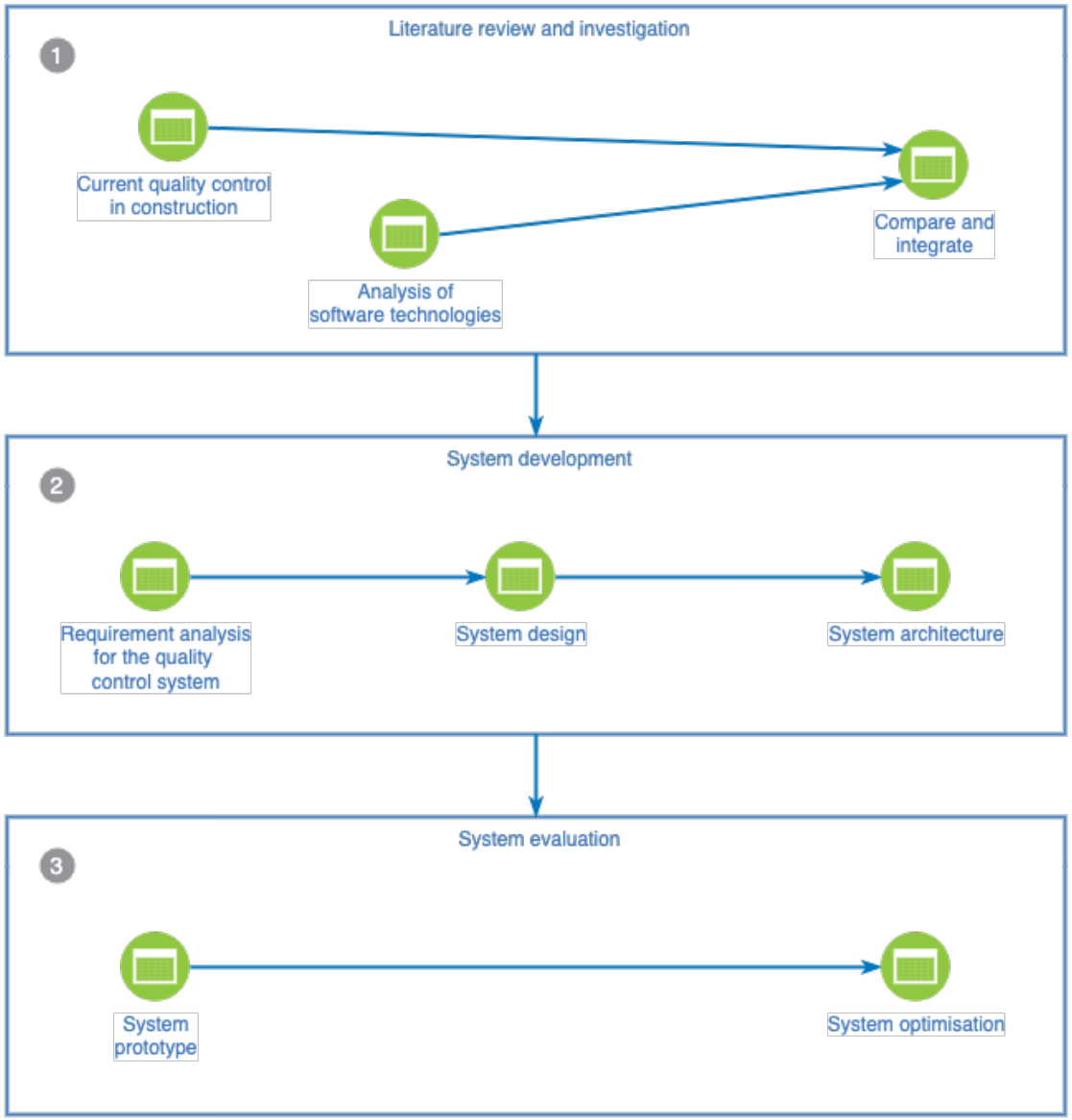
We propose customisable approaches to the system specifically for Android labs

- Traditional methods of indoor control inspections face challenges due to manual processes and data delays
- Main goal: to develop software that provides real-time inspections with on-device model inference
- Advantages: improved accuracy, reduced time on-site, and instant data synchronization.
- Prior research has explored construction progress monitoring using computer vision, AR, robotics, but with a focus on outdoor picture.

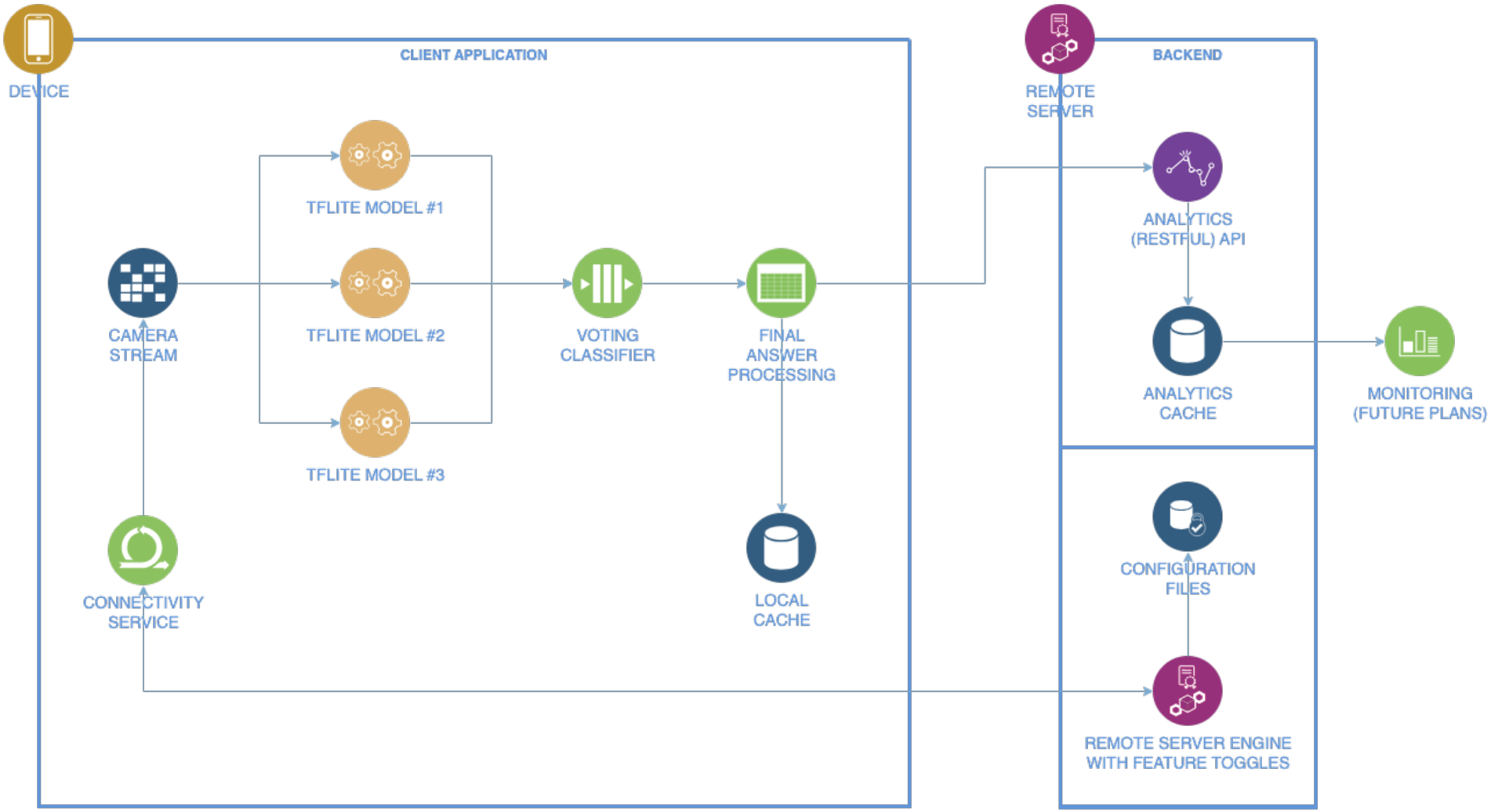
Objectives are:

1. Developing a client-server system architecture for communication and data transfer.
2. Implementing an Android application with adapted interface and inspection features (object recognition).
3. Integrating efficient on-device object recognition models and implementing additional post-processing tasks.
4. Evaluating the system under several mocks.

Methodology



High-level system architecture overview with data flows



On-device *vs* server-based inference comparison

Aspect	**Our Solution (On-Device Inference)**	**Server-Based Inference**	**Conventional Approaches**
Recognition	Automated via on-device ML model	Automated via ML model on server	Manual inspection and identification
Latency	Low	Moderate	High
Offline Capability	Fully functional	Not available	Fully functional (manual processes)
Model Update	Requires model sync	Easy to update	Not applicable
Inspection Accuracy	High with antifraud module	High with antifraud module	Variable
Report Generation	Automated report generation	Automated report generation	Manual report preparation

Soft Voting Classifier

activity_main.xml MainActivity.kt efficientnet_lite4_int8.tflite ×

Model

Name	EfficientNet-lite image classifier (quantized)
Description	Identify the most prominent object in the image from a set of 1,000 categories such as trees, animals, food, vehicles, person etc.
Version	v1
Author	TensorFlow
License	Apache License. Version 2.0 http://www.apache.org/licenses/LICENSE-2.0.

Tensors

Inputs

Name	Type	Description	Shape	Min / Max
image	Image <uint8>	Input image to be classified. The expected image is 300 x 300, with three channels (red, blue, and green) per pixel. Each element in the tensor is a value between min and max, where (per-channel) min is [0] and max is [255].	[1, 300, 300, 3]	[0] / [255]

Outputs

Name	Type	Description	Shape	Min / Max
probability	Feature <uint8>	Probabilities of the 1000 labels respectively.	[1, 1000]	[0] / [1]

Android Application Development

Table: List of frameworks for the app

Library	Description
CameraX	Interacting with the camera and showing preview
NavigationKTX	Navigating inside the app and switching between fragments
DataStore	Saving data locally
TFLite	Image classification models inference
	MobileNet v3, EfficientNet-Lite4, and DenseNet
Diskord	Connecting to Discord server via REST
JUnit	Launching unit-tests for the View Model (MVVM architecture)

Server for retrieving the necessary configuration settings and feature flags, see FeatureService Kotlin class
Sealed class (with descendants such as window, door and radiator) for abstract object that we detect
pre-trained on ImageNet–1K (ILSVRC-2012-CLS dataset)

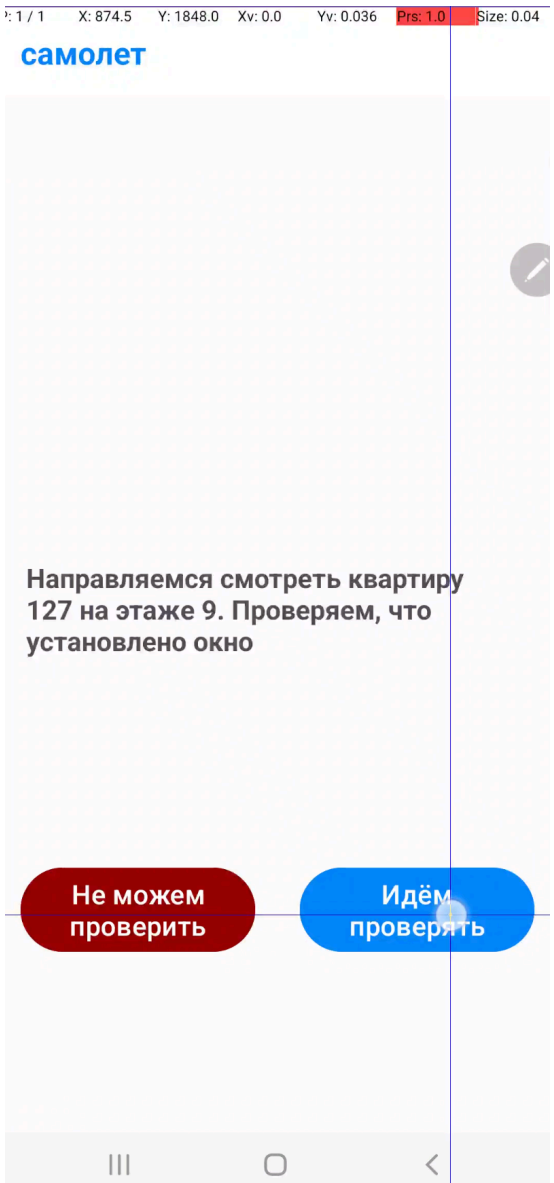
Degree of credibility

Modern mobile devices are equipped with a variety of sensors that can provide some information about the device's environment

- GPS sensor
- Accelerometer
- Gyroscope
- Magnetic field sensor
- Proximity sensor

For example, the GPS location does not match the expected construction site coordinates $\rightarrow$ a warning is triggered

UI


: 1 / 1 X: 874.5 Y: 1848.0 Xv: 0.0 Yv: 0.036 Prs: 1.0 Size: 0.04
самолет
Направляемся смотреть квартиру 127 на этаже 9. Проверяем, что установлено окно
Не можем проверить
Идём проверять

Results and Discussion

- Our comprehensive solution for for indoor control inspections of construction progress
- A server mock and a client application
- On-device ML for real-time inference directly on the Android phone or tablet
- The minimum supported version is Android Oreo (API 26)
- The ML system is implemented using TensorFlow Lite, employing lightweight models optimised for on-device inference
- Standard indoor lighting conditions, though variations in lighting are common in actual construction sites
- The application requires Internet connectivity, at least during app launch

Our work contributes to the field by providing a practical solution for construction progress monitoring and highlights improvements over existing methods in terms of credibility and data transfer delays

Future directions for the Android part: extend support for compose layouts

Code Availability Statement.
https://github.com/DmitrijEgorow/AndroidFramingScanner

Thank you!

[1] Y. Zou, A. Kiviniemi, and S. W. Jones, "A review of risk management through BIM and BIM-related technologies", Safety science, vol. 97, pp. 88–98, 2017.

[2] X. Wang, P. E. Love, M. J. Kim, C.-S. Park, C.-P. Sing, and L. Hou, "A conceptual framework for integrating building information modeling with augmented reality", Automation in construction, vol. 34, pp. 37–44, 2013.

[3] J. Wang, W. Sun, W. Shou, X. Wang, C. Wu, H.-Y. Chong, Y. Liu, and C. Sun, "Integrating BIM and LiDAR for real-time construction quality control", Journal of Intelligent Robotic Systems, vol. 79, pp. 417–432, 2015.

[4] M. Golparvar-Fard, F. Pena-Mora, and S. Savarese, "Automated progress monitoring using unordered daily construction photographs and IFC-based building information models", Journal of Computing in Civil Engineering, vol. 29, no. 1, p. 04 014 025, 2015.

[5] S. Halder, K. Afsari, J. Serdakowski, S. DeVito, M. Ensafi, and W. Thabet, "Real- time and remote construction progress monitoring with a quadruped robot using augmented reality", Buildings, vol. 12, no. 11, p. 2027, 2022.

[6] C. Zhang and D. Arditi, "Automated progress control using laser scanning technology", Automation in construction, vol. 36, pp. 108–116, 2013.

[7] M. Golparvar-Fard, F. Peña-Mora, and S. Savarese, "Integrated sequential as-built and as-planned representation with d 4 ar tools in support of decision-making tasks in the aec/fm industry", Journal of Construction Engineering and Management, vol. 137, no. 12, pp. 1099–1116, 2011.

Thank you!

[8] M. Foroughi Sabzevar, M. Gheisari, and J. Lo, "Development and assessment of a sensor-based orientation and positioning approach for decreasing variation in camera viewpoints and image transformations at construction sites", Applied Sciences, vol. 10, no. 7, p. 2305, 2020.

[9] S. Zollmann, C. Hoppe, S. Kluckner, C. Poglitsch, H. Bischof, and G. Reitmayr, "Augmented reality for construction site monitoring and documentation", Proceedings of the IEEE, vol. 102, no. 2, pp. 137–154, 2014.

[10] Y. Zhou, H. Luo, and Y. Yang, "Implementation of augmented reality for segment displacement inspection during tunneling construction", Automation in Construction, vol. 82, pp. 112–121, 2017.

[11] C.-S. Park, D.-Y. Lee, O.-S. Kwon, and X. Wang, "A framework for proactive construction defect management using BIM, augmented reality and ontology-based data collection template", Automation in construction, vol. 33, pp. 61–71, 2013.

[12] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al., "Searching for mobilenetv3", in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 1314–1324.

[13] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger, "Densely connected convolutional networks", arXiv preprint arXiv:1608.06993, 2016.

[14] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks", in International conference on machine learning, PMLR, 2019.

Soft Voting Classifier

Model

Name	MobileNetV3 image classifier
Description	Identify the most prominent object in the image from a set of 1,001 categories such as trees, animals, food, vehicles, person etc.
Version	v1
Author	TensorFlow
License	Apache License. Version 2.0 http://www.apache.org/licenses/LICENSE-2.0.

Tensors

Inputs

Name	Type	Description	Shape	Min / Max
image	Image <float32>	Input image to be classified. The expected image is 224 x 224, with three channels (red, blue, and green) per pixel. Each element in the tensor is a value between min and max, where (per-channel) min is [0.0] and max is [1.0].	[1, 224, 224, 3]	[0] / [1]

Outputs

Name	Type	Description	Shape	Min / Max
logit	Feature <float32>	Logits vector of the 1001 labels respectively. Apply softmax to the logits vector to get probabilities if needed.	[1, 1001]	[] / []