Towards Understanding the Adaptation Space of AI-Assisted Data Protection for Video Analytics at the Edge

Clemens Lachner
Distributed Systems Group
TU Wien
Vienna, Austria
c.lachner@dsg.tuwien.ac.at

Zoltán Ádám Mann
Paluno
University of Duisburg-Essen
Essen, Germany
zoltan.mann@paluno.uni-due.de

Schahram Dustdar
Distributed Systems Group
TU Wien
Vienna, Austria
dustdar@dsg.tuwien.ac.at

***Abstract*—Edge computing facilitates the deployment of distributed AI applications, capable of processing video data in real time. AI-assisted video analytics can provide valuable information and benefits in various domains. Face recognition, object detection, or movement tracing are prominent examples enabled by this technology. However, such mechanisms also entail threats regarding privacy and security, for example if the video contains identifiable persons. Therefore, adequate data protection is an increasing concern in video analytics. AI-assisted data protection mechanisms, such as face blurring, can help, but are often computationally expensive. Additionally, the heterogeneous hardware of end devices and the time-varying load on edge services need to be considered. Therefore, such systems need to adapt to react to changes during their operation, ensuring that conflicting requirements on data protection, performance, and accuracy are addressed in the best possible way. Sound adaptation decisions require an understanding of the adaptation options and their impact on different quality attributes. In this paper, we identify factors that can be adapted in AI-assisted data protection for video analytics using the example of a face blurring pipeline. We measure the impact of these factors using a heterogeneous edge computing hardware testbed. The results show a large and complex adaptation space, with varied impacts on data protection, performance, and accuracy.**

***Index Terms*—edge computing, fog computing, artificial intelligence, data protection, anonymization, face blurring, Video Analytics Pipeline**

I. Introduction

Video feeds generated by distributed devices enable a variety of applications. For example, in smart cities, videos from cameras can be used for traffic monitoring, accident reporting, and law enforcement applications, among others [1]–[3]. In some cases, also user-generated content may be available. E.g., in the case of an accident, videos taken by passers-by with their smartphones may also aid the work of first responders.

Videos from public spaces may contain sensitive data associated with special security or privacy requirements [4], [5]. For example, people's faces or cars' license plates are personally identifiable information, and processing such information is subject to data protection regulations, such as the General Data Protection Regulation (GDPR) in the European Union [6].

Recent advances in artificial intelligence (AI), particularly in machine learning (ML), enable effective automatic video processing [7]–[10]. AI can also help in satisfying data protection requirements; e.g., faces and license plates can be automatically detected by ML-based object detection algorithms, and then anonymized by further video manipulation methods.

Usually, processing videos is very resource-intensive, but the end devices producing the videos are resource-constrained. Video processing may be offloaded to the cloud to benefit from the virtually unlimited computational capacity of the cloud. However, offloading to the cloud is associated with high latency and high network load. A better solution is to deploy devices – called edge nodes – with sufficient computational capability near the network edge. End devices can offload some video processing functionality to nearby edge nodes, thereby benefiting from low-latency access to computational capacity without overloading the core network. With the appropriate distribution of functionalities between end devices, edge nodes and the cloud, optimal performance can be achieved [11].

A challenge for such systems is the dynamic variability of their run-time environment [12]–[14]. For example, the number of smartphones offloading video processing to an edge node may change at run time. Also the capabilities of the connected smartphones and the properties of the videos to process can change over time. Thus, video processing systems at the edge must be self-adaptive to be able to react to changes at run time. Self-adaptation involves monitoring the system and its environment, analyzing whether changes threaten the satisfaction of the requirements, and the planning and execution of adaptations if needed to ensure the continued satisfaction of requirements [15]. For example, a smartphone may be able to perform face anonymization locally as long as the video contains only one face, but when the number of faces in the video increases, the processing may have to be offloaded to ensure proper operation. This requires an automatic run-time adaptation of the face-anonymization pipeline.

To enable adaptations at run time, appropriate adaptation

This paper was published in *IEEE 41st International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 7-12, 2021

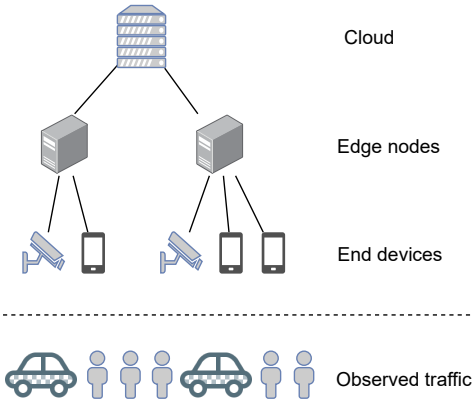

Fig. 1. Edge infrastructure in the considered use case

rules have to be defined at design time, specifying what adaptation to perform in which situation. Existing approaches to creating self-adaptive systems assume that the designer is able to define the appropriate adaptation rules [16]. For defining adaptation rules, it is crucial to understand i) what changes in the environment may happen, ii) what self-adaptations the system may perform, and iii) how those changes and self-adaptations impact the relevant system properties [17]. For video analytics at the edge, these questions are complicated, since there are many different types of possible environment changes and self-adaptations, with intricate implications on a variety of system properties. Environment changes may happen both at the infrastructure level and the application level, and also self-adaptations are possible on both levels.

This paper presents the first detailed study on the adaptation space of AI-assisted data protection for video analytics at the edge. We use the example of a ML-based face-anonymization pipeline which can be distributed between end devices and edge nodes. Our contributions are as follows:

- We identify relevant parameters of the environment that can change, possible self-adaptations that the system can perform, and key system metrics.
- We perform extensive measurements in a heterogeneous testbed to determine the effect of different environment changes and self-adaptations on the identified metrics.

Such results are needed to have a solid basis for designing the adaptation logic for AI-assisted data protection for video analytics at the edge. Our work is a first step in this direction and the results are potentially applicable to a wide field of other possible scenarios beyond face anonymization.

II. Motivating Scenario

We consider a use case of traffic monitoring in a smart city. Cameras are installed across the city to monitor the traffic on the streets. The video feed of each camera is streamed to a nearby edge node, as shown in Fig. 1. Edge nodes are computational resources deployed throughout the city, for example in road-side units (RSUs) or smart lampposts. The video feeds from the cameras are anonymized in the edge

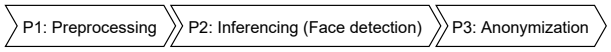

Fig. 2. Face-anonymization application in the considered use case

nodes before they are forwarded to the cloud. Traffic control experts use a cloud-based application to identify and analyze potential traffic incidents, such as traffic jams.

In addition to the statically deployed cameras, also citizens' smartphones may connect to a nearby edge node and stream their video feeds to the edge node. In this case, the anonymization of the video feed may happen either in the smartphone or in the edge node.

The anonymization of video feeds is performed by a Video Analytics Pipeline (VAP), schematically depicted in Fig. 2. This face-anonymization pipeline consists of three phases, where i) the video is pre-processed, ii) faces are detected in the video frames, and iii) faces are made unrecognizable.

During the operation of the system, many changes are possible. For example, assume that a smartphone is taking a video feed and anonymizing it using a face-blurring technique in real time. As more people join the scene, more faces need to be detected and blurred in the video, which increases the computational needs of the face-anonymization pipeline. If the computational capacity of the smartphone is not sufficient anymore, an adaptation is needed. One possibility is to offload one or more phases of the face-anonymization pipeline to a nearby edge node. Another possibility is to resort to a less resource-consuming anonymization technique (e.g., drawing a black rectangle over the faces instead of Gaussian blurring). Which of these potential adaptations is most appropriate has to be decided on the fly, depending on the given situation and on the implications of potential adaptations.

III. Adaptation Space for Data Protection

Optimizing the performance of a VAP operating at the edge is a major research topic. Various optimization objectives can be considered, including execution time, latency, and inferencing accuracy. Optimization is carried out during either design time or run time of an application, resulting in adaptations on the infrastructure or application level. Accuracy plays a crucial role for applications incorporating data protection mechanisms, e.g., anonymization, that have to adhere to specific privacy regulations like the GDPR. An application may have to sacrifice potential performance benefits from optimization strategies to achieve the necessary level of accuracy. The heterogeneous and dynamic environment in edge computing greatly increases the complexity of such optimization strategies. Therefore, we need a better understanding of the main levers for performance and data protection in each of the three phases of the VAP, as depicted in Fig. 2.

Table I displays the most relevant adaptations we identified and their impact on data protection and performance. For application adaptations, the relevant phase of the VAP (P1-P3) is indicated in the table. P0 is a special case of a pre-runtime phase. Infrastructure adaptations can relate to any

TABLE I
IMPACT OF ADAPTATIONS ON DATA PROTECTION AND PERFORMANCE

Adaptation	Phase	Data protection	Performance
Compiled Programming Language	P0	/	+
Greyscaling	P1	-	+++
Adequate Video Resolution	P1	+	-
Frame Skipping	P1	-	+
Adequate AI-Model	P2	+++	+
Batching	P2	/	++
AI-Inference Chaining	P2	++	- - -
Sophisticated Anonymization	P3	+	-
Enable Dedicated Hardware	*	/	+++
Overclocking	*	/	++
Migration	*	+/-	++
Offloading	*	-	++

phase, indicated by a *. The impact of an adaptation on data protection and performance is shown by + (positive impact), - (negative impact), or / (no significant impact).

The trigger for such adaptations can be a change on the application level or on the infrastructure level. Application-level triggers include changes in video content (e.g., the number of people in the video, the size of their faces in the video, the angle of faces to the camera, lightning conditions), changes in the number of video feeds to process, or changes in application requirements (e.g., different frame rate required depending on the purpose of the video feed). Infrastructure-level triggers include changes in the available hardware (e.g., mobile end devices connecting to or disconnecting from an edge node) and in the trustworthiness of hardware (e.g., a camera becoming compromised by a physical attack).

In the following, we describe the identified possible adaptations of each phase in more detail and reason about their impact based on the example of Section II.

A. Adaptations of the pre-processing phase (P1)

Video data needs to be prepared for processing by a face detection framework. This may involve encoding, decoding, or transcoding the video, greyscaling the frames, or resizing each frame. Greyscaling is needed since most object recognition frameworks (e.g., TensorFlow) take greyscaled images as input. Transforming the video can be a heavy computational task, depending on how the video is recorded (frame rate, resolution, codec, etc.). Manufacturers like Nvidia (NVEnc) integrate dedicated chips into their hardware to facilitate this task. Encoding, decoding or transcoding may have a significant impact on overall performance of a VAP, but do not affect data protection quality directly. While resizing and greyscaling are computationally not very expensive (in the sub-millisecond range per frame even on a Raspberry Pi4), adaptations concerning resizing an image could negatively affect data protection. This is because face detection may produce less accurate results on the resized video, leading to undetected and thus non-anonymized faces in the output video.

Adaptations changing the frame rate of an input video should take the context and nature of the input video into account. If (near-)real-time performance (and experience of a user) is the goal, the frame rate should not go below 24 FPS. However, if this is not possible due to computational limitations, skipping a given number of frames, i.e., providing only each k-th frame to the face detection step, provides a viable option for videos without abrupt changes. Frame skipping assumes that a face detected in frame n is in the same location in frames $n+1, \ldots, n+k-1$ as well. Thus, anonymization operates for each of these frames on the location where a face was detected in frame n. For videos with abrupt changes, this may degrade anonymization quality.

B. Adaptations of the inference phase (P2)

The performance of the second phase of the pipeline is heavily dependent on the face-recognition framework itself as well as on the inference model used by the framework. In the context of AI-based inference tasks, performance is not only related to processing speed but also to the accuracy with which an object like a face is detected in an image. To detect faces in a video, a framework looks at every frame of the video, trying to infer if a face is present in the given frame. The accuracy is mostly dependent on factors like face-angle or lightning conditions in the picture [18]. A well-trained model embedded in modern frameworks like e.g., TensorFlow will generally allow for high inference speed with high accuracy. In some cases, inferencing may take significantly longer if more than one face is present in the frame. Such frameworks are not necessarily available and/or optimized for each system architecture like e.g., ARM, x86, x64. Differences in performance and accuracy can also occur due to the usage of legacy versions of such AI frameworks.

There are several possibilities for adaptations concerning inference that aim to improve accuracy and/or performance. The most common practice in AI-based VAPs is batching. Modern AI frameworks provide an API, allowing an application to send multiple frames in one request and process them in parallel. Adapting the batch size and frequency can be considered viable adaptations because they do not negatively affect data-protection quality, but may improve performance. Changing the pre-trained model for inferencing is another possibility. This adaptation becomes relevant if the analyzed video is prone to dynamic changes as it may increase accuracy and/or performance, but may also decrease data protection quality if accuracy drops due to the model change.

In a scenario as described in Section II, accuracy may also be improved if multiple inference steps are chained, e.g., first, persons are detected in a video frame, then on the resulting regions of the frame, face detection is performed. Such strategies heavily affect overall performance, but may be necessary to adhere to data protection regulations and policies. Another adaptation is switching between implementations in different programming languages. For example, Python is heavily used for AI applications in the research community, but being an interpreted language, applications written in Python generally perform worse than using a compiled language like C or C++.

C. Adaptations of the anonymization phase (P3)

The third phase involves the actual anonymization of detected faces in a video frame, i.e., a graphic overlay is drawn over the face in the image. A face-detection framework returns a bounding box (an area inside an image) that corresponds to, or covers the face detected inside a frame. This bounding box is then used to draw an overlay onto the image. Both performance and data protection quality depend on the desired graphic nature of the overlay. One possibility is blurring the face in an image by applying a Gaussian Blur Transformation function to the image part cropped using the previously described bounding box. Pixelating the face area can be considered an improvement in anonymization: the area is divided into an $n \times m$ image-tile matrix and each tile is transformed with e.g., Gaussian Blur. A simple alternative, with minimal performance impact, maybe a more blunt approach, where the cropped image part is just painted with a single color, resulting in e.g., a white circle drawn 'over' the face. However, similar to resizing and greyscaling described in Section III-A, the differences in execution times between these anonymization strategies are small compared to the inferencing time.

D. Adaptations of the infrastructure

The execution time of a task often heavily depends on the capabilities, e.g., CPU speed, architecture (ARM, x64, etc.), or type of hardware (CPU, GPU, TPU, etc.) available, of the device the task is running on. Executing all three phases on the same device may limit the overall performance of the VAP. A common approach is to decouple the phases of a VAP and execute them separately on different devices. Offloading tasks to the cloud or powerful edge nodes bears great potential to minimize execution time, but comes with the downside of increasing latency [19], [20]. Data locality requirements may hinder offloading, potentially forcing a task that operates on sensitive data, to run on specific premises [21].

Infrastructure adaptations may be useful in any of the three phases. Phase 2 can benefit the most from infrastructure adaptations in terms of performance, but may also suffer from downsides like additional energy costs and/or laborious human intervention. Infrastructure adaptations typically do not directly influence accuracy, hence they can increase performance without compromising data protection quality. Activating (pre-installed and available on demand) dedicated hardware like GPUs or Tensor Processing Units (TPUs) are viable infrastructural adaptations. In virtualized environments, migrating the inference component to a more powerful device can be a beneficial adaptation. However, data locality aspects and privacy policies need to be considered. Overclocking hardware may also be a possible adaptation that should be carried out carefully. Dedicated AI hardware like e.g., an Nvidia Jetson device, provides interfaces and scripts to change performance profiles (e.g., CPU voltage, fan speed, clock speed) on the fly. However, these adaptations may also increase energy consumption and potentially decrease hardware life expectancy. A further adaptation possibility is to execute the inference task on CPUs with different instruction sets. AI frameworks, like TensorFlow, support special instruction sets (Single Instruction Multiple Data) and other features that increase performance.

TABLE II
DEVICES USED FOR THE EXPERIMENTS

Name	CPU	GPU	RAM
Desktop PC	AMD Ryzen 5600X@3.7GHz	not used	32GB
Laptop	Intel i7-7820HQ@2.9GHz	not used	16GB
Intel NUC	Intel i5-7260U@1.5GHz	not used	16GB
Raspberry Pi4	ARM Cortex-A72@1.5GHz	not used	4GB
Nvidia Jetson TX2	ARM Cortex-A57@2GHz	Nvidia Pascal, 256 cores	8GB

IV. EVALUATION

We implemented the face-anonymization pipeline described in Section II using Python 3.6, OpenCV 3.4 and TensorFlow 1.14. Parameters can be used to choose between different versions of each phase of the pipeline. We performed a series of experiments to evaluate the impact of different factors on the performance of the face-anonymization pipeline. We performed the experiments on a heterogeneous testbed consisting of several different types of devices, shown in Table II. The source code and the results are available online[1].

Overall, the empirical findings reinforced the results of the theoretical analysis of Section III. In the following, we present some of the quantitative findings from the experiments.

Fig. 3 shows the impact of the used hardware on the VAP's performance. The fastest device (desktop PC) offers roughly 30 times higher performance than the slowest one (Raspberry Pi). Thus, offloading computations to a more powerful device offers huge potential for improving performance.

Fig. 4 shows the impact of the number of faces in the video on the VAP's performance. The results for the NUC are shown; the results for the other devices are similar. Processing a video with many faces leads to a performance loss of about 5% compared to a video containing a single face. This is an example of the impact of an environmental factor. The system has no influence on the number of faces in the video, but the system may have to adapt to react to changes in the number of faces in the video, to counteract the performance loss.

Fig. 5 shows the impact of different anonymization methods on the performance of the face-anonymization pipeline. The performance difference between the fastest and slowest anonymization method is about 2.8%. The difference is small because the anonymization method only affects the performance of the last phase of the face-anonymization pipeline. The execution time of the face-anonymization pipeline is dominated by the second phase (face detection), hence accelerating the third phase has only limited effect.

Fig. 6 shows the impact of frame skipping on the performance of the face-anonymization pipeline. If face detection is performed only for every 5th frame, this leads to roughly a 5 times performance increase for the whole face-anonymization

[1]https://github.com/clemenslachner/EdgeAIAdaptations

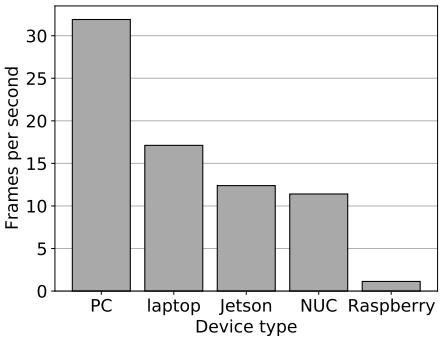

Fig. 3. Throughput of the face-anonymization VAP run on different devices

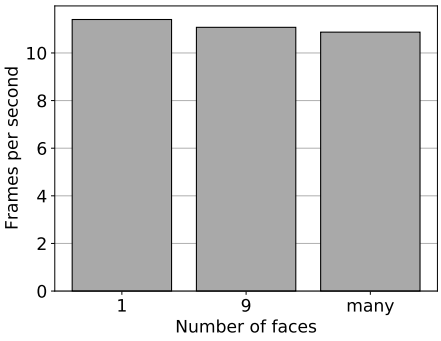

Fig. 4. Throughput of the VAP on the NUC, depending on the number of faces in the video

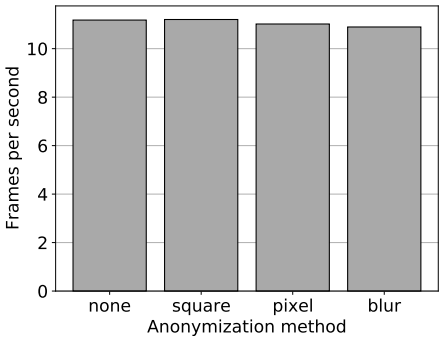

Fig. 5. Throughput of the VAP on the NUC, with different anonymization methods

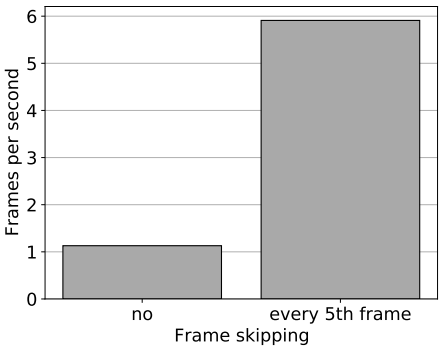

Fig. 6. Effect of frame skipping on the throughput of the VAP on the Raspberry Pi

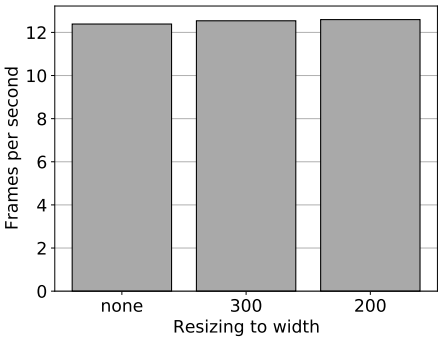

Fig. 7. Effect of resizing on the throughput of the VAP on the Jetson board

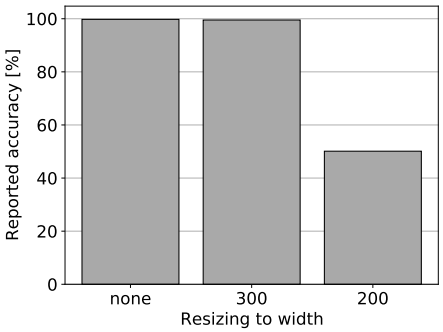

Fig. 8. Effect of resizing on the accuracy of the VAP on the Jetson board

pipeline. Hence, activating frame skipping or changing the number of skipped frames is a very effective adaptation.

Fig. 7-8 show the impact of resizing the frames on the measured performance and the accuracy reported by the AI framework. Scaling the video to smaller sizes leads to a slight increase in performance, but may lead to a dramatic loss in accuracy. This underlines the importance of understanding the effect of different adaptations on key metrics.

V. Related Work

Privacy and security are critical aspects of video analytics systems. Recent research identified edge computing as a key enabler for privacy-sensitive systems dealing with real-time video processing [22], [23]. Today's hardware capabilities potentially enable real-time video processing at the edge where, typically, data originates. In the context of privacy in video-based media spaces, Boyle et al. [24] proposed a framework – a descriptive theory – that defines how one can think of privacy while analyzing media spaces and their expected or actual use. The framework explains three normative controls: solitude, confidentiality and autonomy, yielding a vocabulary related to the subtle meaning of *privacy*. A more technical introduction to video surveillance is given by Senior in [25]. The paper briefly summarizes the elements in an automatic video surveillance system, including architectures, followed by the steps in video analysis, from preprocessing to object detection, tracking, classification and behaviour analysis. Our implementation builds on the high-level architecture described in that paper, and adds AI-based video processing capabilities. Chattopadhyay et al. demonstrate how the practical problem of privacy invasion can be successfully addressed through DSP hardware in terms of smallness in size and cost optimization [26]. This is particularly useful for edge computing, where computational resources may be scarce. Much research focuses on encryption and anonymization of image and video data. Other, more application-specific approaches, often involve preprocessing of video streams to anonymize or obscure specific parts of a frame. An example is the work of Schiff et al. [27] that proposes *Respectful Cameras*, i.e., cameras that respect the privacy preferences of individuals. Their real-time approach preserves the ability to monitor activity while obscuring individual identities. This is achieved by identifying colored markers such as hats or vests, which are automatically tracked by their system. The identities of people wearing, e.g., a colored vest, are obscured by adding a solid overlay over the face in every frame.

Several authors proposed using adaptations to cope with dynamic changes of edge computing systems. Breitbach et al. combine different data placement and task scheduling policies to adaptively react to changes in the system context [28]. Gand et al. introduce a fuzzy controller for self-adaptive container orchestration for edge devices [29]. Samir and Pahl propose using adaptations for self-healing of edge cluster systems [30]. Wang and Xie develop an algorithm for the adaptive choice

of parameters in mobile augmented reality systems [31].

Previous approaches rely on carefully selected parameters with known impact on the metrics of interest. As we have seen, identifying the parameters and their impact for AI-assisted data protection for video analytics at the edge is a non-trivial task, which has not been solved yet. Our work thus paves the way towards effective adaptation algorithms for AI-assisted privacy-preserving video analytics at the edge.

VI. Conclusions and future work

This paper presented preliminary results of the first systematic study of the adaptation space of AI-assisted data protection for video analytics at the edge. Using a face-anonymization pipeline as running example, we identified several possible adaptations and analyzed their impact on performance and data protection. We also implemented and empirically evaluated several of the considered adaptations. The results show a wealth of different adaptation options on both the infrastructure and application level. The results also show that the impact of these adaptations varies significantly.

In the future, we intend to extend our research to further types of edge AI analytics applications and devices, as well as with further relevant metrics. In addition, we plan to transform the results to performance models that can be used to automatically reason about possible adaptations at run time.

Acknowledgement. This work was partially supported by the European Union's Horizon 2020 research and innovation programme under grant 871525 (FogProtect). We thank Sebastian Kaindl, Christoph Doppelhammer, and Johannes Braitenthaller for their great support in the implementation of our evaluation VAP.

References

[1] C. E. Catlett, P. H. Beckman, R. Sankaran, and K. K. Galvin, "Array of things: A scientific research instrument in the public way: Platform design and early lessons learned," in *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering*. ACM, 2017, pp. 26–33.

[2] F. Nunes, "Aveiro, Portugal: Making a digital city," *Journal of Urban Technology*, vol. 12, no. 1, pp. 49–70, 2005.

[3] A. Cenedese, A. Zanella, L. Vangelista, and M. Zorzi, "Padova smart city: An urban Internet of Things experimentation," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*. IEEE, 2014, pp. 1–6.

[4] F. Muhammad, W. Anjum, and K. S. Mazhar, "A critical analysis on the security concerns of Internet of Things (IoT)," *International Journal of Computer Applications (0975 8887)*, vol. 111, no. 7, 2015.

[5] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the Internet of Things: a review," in *2012 International Conference on Computer Science and Electronics Engineering*, vol. 3. IEEE, 2012, pp. 648–651.

[6] D. Ayed, E. Jaho, C. Lachner, Z. Á. Mann, R. Seidl, and M. Surridge, "FogProtect: Protecting sensitive data in the computing continuum," in *Advances in Service-Oriented and Cloud Computing: International Workshops of ESOCC 2020, Revised Selected Papers*, 2021, pp. 179–184.

[7] F. Dornaika and J. Ahlberg, "Fast and reliable active appearance model search for 3-D face tracking," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 4, pp. 1838–1853, 2004.

[8] R. Feraund, O. J. Bernier, J.-E. Viallet, and M. Collobert, "A fast and accurate face detector based on neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 42–53, 2001.

[9] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.* IEEE, 2005, pp. 255–260.

[10] A. Hampapur, S. Borger, L. Brown, C. Carlson, J. Connell, M. Lu, A. Senior, V. Reddy, C. Shu, and Y.-L. Tian, "S3: The IBM smart surveillance system: From transactional systems to observational systems," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 4. IEEE, 2007, pp. IV–1385.

[11] Z. Á. Mann, "Optimization problems in fog and edge computing," in *Fog and edge computing: Principles and paradigms*, R. Buyya and S. N. Srirama, Eds. Wiley, 2019, pp. 103–121.

[12] A. Brogi, S. Forti, and A. Ibrahim, "Predictive analysis to support fog application deployment," *Fog and edge computing: principles and paradigms*, pp. 191–222, 2019.

[13] A. Brogi, S. Forti, C. Guerrero, and I. Lera, "How to place your apps in the fog: State of the art and open challenges," *Software: Practice and Experience*, vol. 50, no. 5, pp. 719–740, 2020.

[14] O. Vermesan and P. Friess, *Internet of Things: Converging technologies for smart environments and integrated ecosystems*. River Publishers, 2013.

[15] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[16] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 4, no. 2, pp. 1–42, 2009.

[17] S. Schoenen, Z. Á. Mann, and A. Metzger, "Using risk patterns to identify violations of data protection policies in cloud systems," in *Service-Oriented Computing – ICSOC 2017 Workshops*. Springer, 2018, pp. 296–307.

[18] T. Marciniak, A. Chmielewska, R. Weychan, M. Parzych, and A. Dabrowski, "Influence of low resolution of images on reliability of face detection and recognition," *Multimedia Tools and Applications*, vol. 74, no. 12, pp. 4329–4349, 2015.

[19] X. Ran, H. Chen, Z. Liu, and J. Chen, "Delivering deep learning to mobile devices via offloading," in *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*, 2017, pp. 42–47.

[20] P. Liu, B. Qi, and S. Banerjee, "EdgeEye: An edge service framework for real-time intelligent video analytics," in *Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking*, 2018, pp. 1–6.

[21] Z. Á. Mann, A. Metzger, J. Prade, and R. Seidl, "Optimized application deployment in the fog," in *International Conference on Service-Oriented Computing*. Springer, 2019, pp. 283–298.

[22] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos, "Edge analytics in the Internet of Things," *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 24–31, 2015.

[23] M. Grambow, J. Hasenburg, and D. Bermbach, "Public video surveillance: Using the fog to increase privacy," in *Proceedings of the 5th Workshop on Middleware and Applications for the Internet of Things*, 2018, pp. 11–14.

[24] M. Boyle, C. Neustaedter, and S. Greenberg, "Privacy factors in video-based media spaces," in *Media Space 20+ Years of Mediated Life*. Springer, 2009, pp. 97–122.

[25] A. Senior, *Protecting privacy in video surveillance*. Springer, 2009, vol. 1.

[26] A. Chattopadhyay and T. E. Boult, "PrivacyCam: A privacy preserving camera using uCLinux on the Blackfin DSP," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[27] J. Schiff, M. Meingast, D. K. Mulligan, S. Sastry, and K. Goldberg, "Respectful cameras: Detecting visual markers in real-time to address privacy concerns," in *Protecting Privacy in Video Surveillance*. Springer, 2009, pp. 65–89.

[28] M. Breitbach, D. Schäfer, J. Edinger, and C. Becker, "Context-aware data and task placement in edge computing environments," in *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2019, pp. 1–10.

[29] F. Gand, I. Fronza, N. El Ioini, H. R. Barzegar, S. Azimi, and C. Pahl, "A fuzzy controller for self-adaptive lightweight edge container orchestration," in *10th International Conference on Cloud Computing and Services Science (CLOSER)*, 2020, pp. 79–90.

[30] A. Samir and C. Pahl, "Self-adaptive healing for containerized cluster architectures with Hidden Markov Models," in *2019 Fourth International*

Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2019, pp. 68–73.

[31] H. Wang and J. Xie, "User preference based energy-aware mobile AR system with edge computing," in *IEEE INFOCOM – IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1379–1388.