Approaches towards Resource-saving and Explainability/Transparency of Deep-learning-based Image Classification in Industrial Applications

Constantin Rieder, Markus Germann, Samuel Mezger and Klaus Peter Scherer
Institute for Automation and Applied Informatics, Karlsruhe Institute of Technology, Hermann-von-Helmholtz-Platz 1, Eggenstein-Leopoldshafen, Germany

Keywords: Deep Learning, Information Systems, Intelligent Assistance, Transparency, Continuous Self Learning, Explainability, Neural Networks, Resource Saving Image Classification.

Abstract: In the present work a new approach for the concept-neutral access to information (in particular visual kind) is compiled. In contrast to language-neutral access, concept-neutral access does not require the need to know precise names or IDs of components. Language-neutral systems usually work with language-neutral metadata, such as IDs (unique terms) for components. Access to information is therefore significantly facilitated for the user in term-neutral access without required knowledge of such IDs. The AI models responsible for recognition transparently visualize the decisions and they evaluate the recognition with quality criteria to be developed (confidence). To the applicants' knowledge, this has not yet been used in an industrial setting. The use of performant models in a mobile, low-energy environment is also novel and not yet established in an industrial setting.

1 INTRODUCTION

Complex machines require a reliable and high-performance process in daily operation. The technical service of the respective manufacturer ensures sustained performance with worldwide networks. Malfunctions must be stopped quickly, and components have to be exchanged and replaced in a safe manner. For this purpose, manufacturers provide comprehensive technical information systems that describe spare parts management, diagnosis, maintenance and repair of machines and systems. Fast and unambiguous access to such information is therefore of crucial importance.

1.1 Initial Situation

Previous research projects have developed basic methods to analyze existing information and use ontologies to "semantify" the content. Often, a multimodal approach to semantic access to this information is implemented, using both voice assistants, audio recordings, and imaging to generate the query to the system. Several case studies have shown that multimodal access to the information generally means an increase in efficiency. However, the prerequisite here is that the user of the system clearly knows the terms (of the defective component or the malfunction). The corresponding technical terms are often unknown, so that service technicians cannot access the appropriate documentation, which leads to incorrect repairs and/or increased downtimes. With the application of semantic technologies and a set of requirements that should be taken into account, accessibility to resources in information systems can be drastically improved (Baumeister, 2016). As another important enhancement, Deep Learning technologies can simplify and support information access for the user. In this regard, the methods presented in this paper address the important aspects of transparency and resource conservation when using Deep Learning methods to support information access in information systems.

1.2 Motivation and Databases

The present research project aims at developing a concept-neutral access to information by means of artificial intelligence (AI) image understanding methods and at exploiting this technology industrially in semantic information systems. Two central aspects are emerging:

1. Need for resource-saving models: The use of information systems "in the field" is to take place on mobile devices in perspective. For this pur-

Rieder, C., Germann, M., Mezger, S. and Scherer, K.
Approaches towards Resource-saving and Explainability/Transparency of Deep-learning-based Image Classification in Industrial Applications.
DOI: 10.5220/0010575901640169
In *Proceedings of the 2nd International Conference on Deep Learning Theory and Applications (DeLTA 2021)*, pages 164-169
ISBN: 978-989-758-526-5

pose, the learned recognition models have to be resource-saving for memory and processor technology in order to be universally usable. Current architectures of image-processing AI models are typically memory and computationally expensive, which does not allow responsiveness on low-end devices.

2. Need for transparent recognition models, open for critical feedback: Especially for industrial use, in particular incorrectly recognized concepts are very unsatisfactory for the technicians. A criticizable recognition model can intuitively visualize how it has detected a concept (object class) on the current video image and which differentiating concepts would also be possible (with correspondingly lower recognition quality) (Kuwajima et al., 2019). It is difficult for the user to understand which input data influenced the decisions of a deep neural network, due to its black-box nature (Xie et al., 2020). Such transparency increases the user acceptance and can be utilized for the continuous self-learning of the model. Additionally it provides approaches for further actions such as the extension of the neural network with respect to complexity or the collection of further training data to improve the existing neural network (Kuwajima et al., 2019) (Fazi, 2020).

2 DATASETS AND ACQUISITION

For the development of the image recognition component in the information system, test objects (concepts) were specified on which raw data is recorded and different recognition techniques can be tested. During data acquisition for the training corpus, the components of the test objects to be recognized were recorded from different angles and against different backgrounds using short film techniques. As test objects, in a first step a bicycle was chosen and after that a more complex structure car is regarded. Here, individual components such as ”saddle”, ”rear wheel” and in case of the car ”turn signal left” or ”steering wheel” and so on can be comprehensibly labeled and then recognized by solving a classification problem with neural networks.

The labels for the specific components of the bicycle test objects were set beforehand in a defined structure (see Figure 1). A structure template with an included hierarchical refinement was also defined for the car (see Figure 2). The recorded training data are associated with labels by sorting them into file folders with the corresponding name or label. The component structure of the car consists of about 100

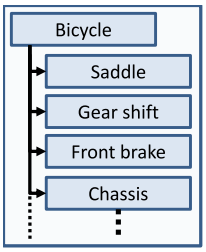

Figure 1: Defined bicycle concepts.

concepts and 100 corresponding folders. For example, all clips from the handbrake are sorted into the directory named ”Handbrake”. The directory structure then helps with the organisation of the clips and the directory names are used as class labels. A relatively large number of frames can now be extracted and labeled from the sorted clips. In order for the frames to be as versatile as possible, different angles and lighting conditions should be taken into account when filming.

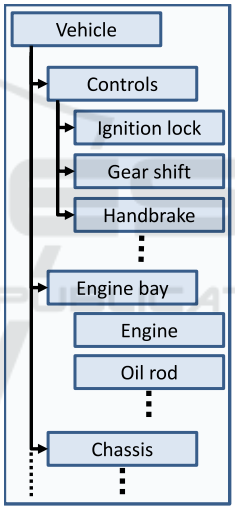

Figure 2: Defined vehicle concepts.

The concepts include various components of the vehicle from the interior and exterior. To refine the structure, these are divided into categories, such as control elements, body, engine compartment, electronics, etc.

3 CONCEPT AND MODELING

This section describes the concept of transparency and resource saving in more detail.

Deep Learning models are only as good as the data that was used for training. Bad or outlier data can lead to unsuitable models. If, as in this case, a model identifies technical components incorrectly or

not at all, we know that the decision is wrong and the model must be corrected. To find the reason for the wrong decision, we need to analyze the input data. But because there are too many of them, a manageable amount is selected to represent the input data set and to be provided to the user for control. The concept of resource saving refers to the autarkic use of the image recognition models. For this, certain requirements must be considered in order to implement the execution in a corresponding way.

3.1 Explainablility/Transparency

The transparency can be considered, among other things, on the basis of the following two criteria (called transparency criterion in the following), which a user should be able to assess:

- The reliability of the system (or the model) in finding the solution i.e. the confidence of the recognition: A typical question here is: "Are there other classes that come into question?" or which are the next best classes in terms of recognition. In addition, criteria for a capping are necessary.
- An explanation of why the chosen class was inferred by the system.

Examples of possible methods are random sampling, metrics such as difference measures, clustering, etc. Possible variants are a static pre-selection before delivery to the app or dynamic generation during the runtime of the app on the device (possibly using a currently captured image), heat maps and bounding boxes. Offering and selecting alternative possible outcomes could help improve prediction when, for example, gradient noise or high dimensionality of the data causes the model to overreact and many small pixel effects accumulate and interfere with the model output (Samek et al., 2021).

The structure for the transparency component of the system is illustrated in the following figure 3.

The top component (Frame Extractor), extracts frames from the video material and makes them available for the representative search. The representative search compares images and uses comparison algorithms for this. Therefore it uses the component for the comparison algorithms. This component provides algorithms from four different categories. The following histogram algorithms from the OpenCV (OpenCV, 2021a) library are used:

- Correlation: Computes the correlation between the two histograms.
- Chi Square: Applies the Chi-Squared distance to the histograms.

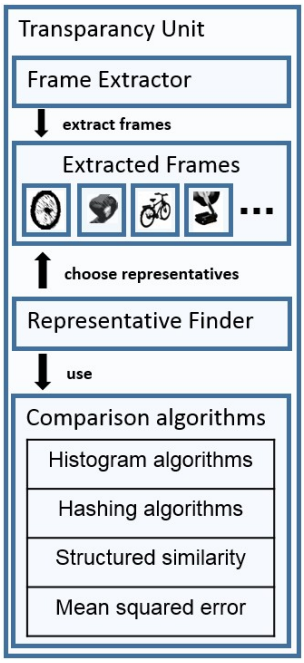

Figure 3: Architectural sketch for transparency unit.

- Intersection: Calculates the intersection between two histograms.
- Relative Entropy: Bhattacharyya distance, used to measure the "overlap" between the two histograms.

Histograms represent the distribution of colors in an image. The histogram can be displayed as a graph that gives a good impression of the intensity distribution (pixel values). Assuming an RGB color space (8 Bit-Encoding), the pixel values range from 0 to 255. For other color spaces, the pixel range looks correspondingly different. The following image hashing algorithms from OpenCV(OpenCV, 2021b) and ImageHash (ImageHash, 2021) are used:

- Color Hash
- Average Hash
- Wavelet Hash
- Perceptual
- Difference

The color hash is an image hash based on color moments. The color hash algorithm analyzes the color distribution and black-gray components without position information. The image hash algorithms (Average, Perceptual, Difference, Wavelet) analyze the image structure for luminance without color information(ImageHash, 2021).

Structural similarity index (SSIM) is another method for measuring the similarity between two images. It is often used in television and cinema. It

represents a so-called perception-based model and is calculated over different parts of the image (Brunet et al., 2012). Another measure of image quality and established method for comparing images is the mean square error (MSE) method (Gonzalez and Woods, 2018). It estimates the absolute error.

3.2 Resource Saving

In the future, information systems will be used ”in the field” on mobile devices. For this, the learned recognition models must be resource-saving for memory and processor technology in order to be deployable over a wide area. Current architectures of image-processing AI models are typically extremely memory- and computationally-intensive, making the use on microdevices impractical. From this, this work is characterized in the configuration and realization of image classification with the following features:

- Usage-related Features
  - Lowest possible memory consumption
  - Battery-saving processor performance
  - High performance (highest possible accuracy)
- Acquisition Technical Features
  - Efficient training of the network with as little sample data as possible (short videos)
  - How performance degrades with less sample data due to degradation studies

For the realization and execution of the experiments, a Raspberry Pi 4 was chosen as the hardware, because it is a small computer with the size of a credit card and matches our requirements. Unlike other mobile devices of this size, it is capable of running a full desktop operating system. By running Linux, it proves to be extremely flexible and reliable in use. Therefore, it forms a suitable experimental platform under the aspect of mobility and resource conservation.

4 REALISATION

It is not intended to train or build new neural networks on the Raspberry Pi. Therefore, instead of the complete TensorFlow package, it is sufficient to install only the significantly smaller interpreter and so TensorFlow Lite (TensorFlow, 2021) has been installed on the Raspberry Pi hardware platform.

4.1 Experimental Setup

For the first experimental setup, MobileNetV2 was used and trained on 13 classes. It is a convolutional neural network architecture that uses, among other things, depthwise separable convolutions. The result is a streamlined and lightweight deep neural network that can perform well on mobile devices (Sandler et al., 2019). This above mentioned neural network served as a base reference, i.e. as the unoptimized model, and was then converted or quantized in further steps with TensorFlow Lite into different formats with the goal of further reducing and streamlining it, so that the following smaller versions of the base model or setups were created:

- TF Lite Model
- Quantized Flat16 Model
- Quantized INT8 Model
- Quantized Dataset

4.2 Results

To compare the models (see figure 4) special software was developed to process the entire test data set through the networks and to evaluate the predictions.

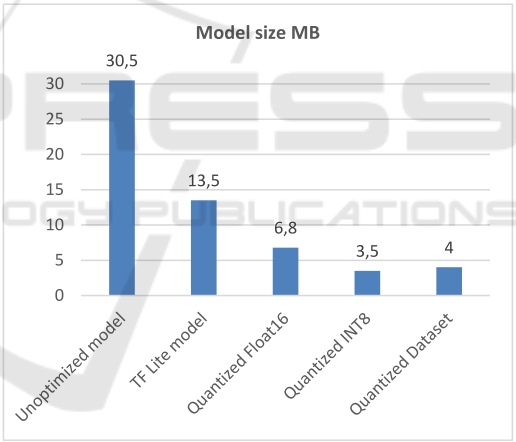

Figure 4: Shrinking models.

Both the top-1 (see figure 5) and top-5 (see figure 6) hit rates were evaluated, and it was measured how long it takes the Raspberry Pi to perform the inference (see figure 7).

The model implemented with Keras has about 30 MB in SavedModel format and achieves a total accuracy of 97%. However, this model cannot be loaded or executed with the TFLite interpreter on the Raspberry Pi.

Running the TensorFlow Lite Converter in the default setting, the model is converted to the flatbuffer format, without changing the numerical values. The weights are stored in the FLOAT32 data type, i.e. as a floating point number with 32 bits. Therefore, the accuracy hardly changes compared to the original

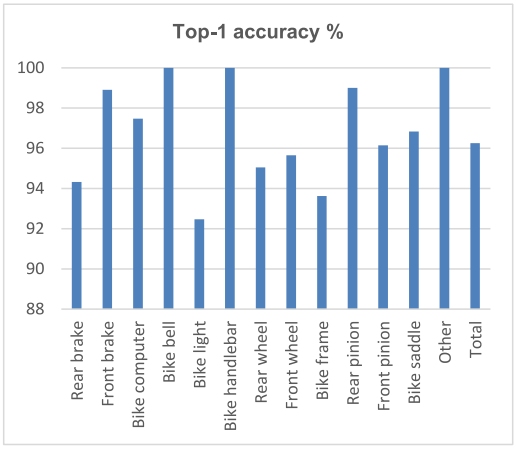

Figure 5: Top-1 accuracy for bike components.

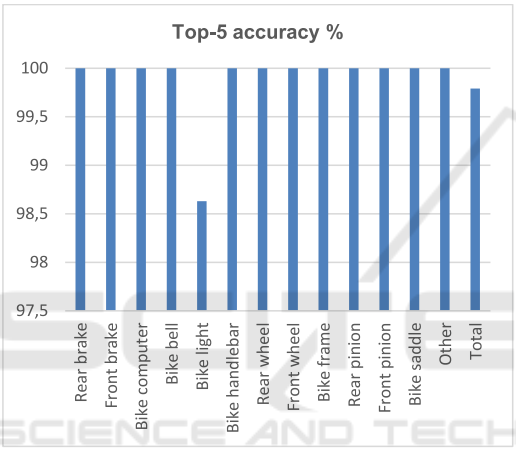

Figure 6: Top-5 Accuracy for bike components.

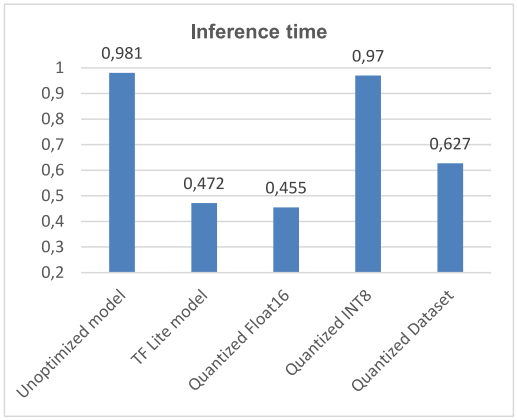

Figure 7: Total inference time in seconds.

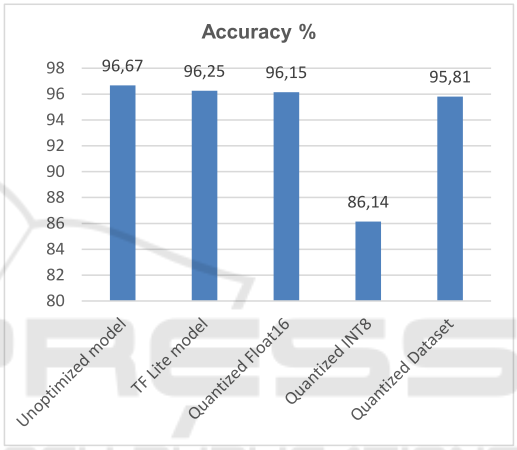

Figure 8: Total accuracy.

model. In order to recognize the component located on an image with this model, the Raspberry Pi needs 472 ms (see figure 7).

Quantization is an optimization method that achieves a smaller memory size in exchange for less precise numerical values. Depending on the processor and data type, it can also provide shorter latency. Quantizing to FLOAT16, or 16-bit floating point numbers, can cut the model size in half without seriously affecting the accuracy of the predictions. In addition, quantization reduced the time to compute the output by about 20 ms. If the accuracy of the weights is reduced to only 8 bits, so that the model is only 3.5 MB in size, a significant decrease in the accuracy rate can be observed (see figure 8). This obviously results in the loss of important information that helps the artificial neural network to classify the images correctly. In order to further reduce the size of the model without losing too much accuracy, a representative data set can be used to distinguish more precisely between relevant and unimportant information. The range of values that the variables assume when pixel information is passed through is examined. Using this highly effective method, it was possible to reduce the size of the model to only 4 MB, while maintaining a hit accuracy that is no more than one percent lower than that of the original model.

5 CONCLUSION

In many application areas, such as in technical service locally in the field, these resource-efficient and explainable AI models have proven to be extremely useful and efficient in the first approach, because

- The necessary expertise on the machine is provided locally.
- According to the architecture, the system has a high degree of self-sufficiency.
- The data and computational load is local.
- Energy resources are available in a limited way and the system is capable of explanation.

Neural network architectures have been designed, developed, and implemented in this work to be efficient and resource-efficient, and to assist the service technician with image understanding procedures on self-sufficient and transportable devices. The second important innovation is criticality and transparency of the system. With the option to view further detection results and adjust them if necessary. This leads to improved user acceptance and creates the basis for continuous improvements.

6 OUTLOOK

This methodology of imaging AI methods under the aspect of resource conservation and transparency is also already relevant for the computer-aided development of new materials (Virtual Materials Design) and is planned in this application as a prototype. These tools support the developer to evaluate experimental results and to identify structure-property relationships.

ACKNOWLEDGEMENTS

The work presented in this article is supported and financed by Zentrales Innovationsprogramm Mittelstand (ZIM) of the German Federal Ministry of Economic Affairs and Energy. The authors would like to thank the project management organisation AiF in Berlin for their cooperation, organisation and budgeting.

REFERENCES

Baumeister, J. (2016). Requirements of affective information systems in the industrial domain. In Ezquerro, M. T. H., Nalepa, G. J., and Méndez, J. T. P., editors, *AfCAI*, volume 1794 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Brunet, D., Vrscay, E. R., and Wang, Z. (2012). On the mathematical properties of the structural similarity index. *IEEE Transactions on Image Processing*, 21(4):1488–1499.

Fazi, M. B. (2020). Beyond human: Deep learning, explainability and representation. *Theory, Culture & Society*.

Gonzalez, R. C. and Woods, R. E. (2018). *Digital image processing*. Pearson, New York, NY, fourth edition edition. Hier auch später erschienene, unveränderte Nachdrucke ; Literaturverzeichnis: Seite 1143-1155 ; "This edition of Digital Image Processing is a major revision of the book." - Preface.

ImageHash (2021). ImageHash 4.2.0: an image hashing library written in python. https://pypi.org/project/ImageHash/. Last checked on Mar 17, 2021.

Kuwajima, H., Tanaka, M., and Okutomi, M. (2019). Improving transparency of deep neural inference process. *CoRR*, abs/1903.05501.

OpenCV (2021a). OpenCV: histogram comparison. https://docs.opencv.org/3.4/d8/dc8/tutorial_histogram_comparison.html. Last checked on Mar 17, 2021.

OpenCV (2021b). OpenCV: module of implementations of different image hashing algorithms. https://docs.opencv.org/3.4/d4/d93/group__img__hash.html. Last checked on Mar 18, 2021.

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Muller, K.-R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2019). Mobilenetv2: Inverted residuals and linear bottlenecks.

TensorFlow (2021). TensorFlow Lite: deploy machine learning models on mobile and iot devices. https://www.tensorflow.org/lite. Last checked on Mar 17, 2021.

Xie, N., Ras, G., van Gerven, M., and Doran, D. (2020). Explainable deep learning: A field guide for the uninitiated.