53rd CIRP Conference on Manufacturing Systems

Distributed Incremental Class Learning for Live Image Recognition

Benjamin Maschlera\*, Simon Kamma, Michael Weyricha

aInstitut of Industrial Automation and Software Engineering, University of Stuttgart, Pfaffenwaldring 47, 70569 Stuttgart, Germany

\* Corresponding author. Tel.: +49 711 685 67295; fax: +49 711 685 67302. *E-mail address:* benjamin.maschler@ias.uni-stuttgart.de

Abstract

In this paper, a novel light-weight incremental class learning algorithm for live image recognition is presented. It features a dual memory method and is capable of learning new, formerly unknown classes as well as conducting its learning across multiple instances at multiple locations simultaneously without storing any images. In addition to tests on the ImageNet dataset, a prototype based upon a Raspberry Pi and a webcam is described and used for further evaluation. It can be shown that the proposed algorithm allows for the performant execution of image classification tasks while learning new classes at several sites at once, which enables its application to a variety of industry use cases.

© 2019 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the 53rd CIRP Conference on Manufacturing Systems

*Keywords:* Artificial Intelligence; Artificial Neural Networks; Deep Learning; Distributed Learning; Incremental Class Learning; Live Image Recognition; Transfer Learning

1. Introduction (Rest der Seite, Ms)

Here introduce the paper, and put a nome­nclature if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, bold and 10 pt. Here follows further instructions for authors.

1. Related Work (max. 1 Seite ggf. Etwas mehr, Ms / Ka)

Here introduce the paper, and put a nome­nclature if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, bold and 10 pt. Here follows further instructions for authors.

Anlehnung an 1. Teil des Konzeptions-Dokuments: Recherche der unterschiedlichen Modul-Möglichkeiten (Ka)

Nothing I know (Ka)

Continual / Transfer / Federated Learning (Ms)

The Distributed Incremental Class Learning Algorithm bases on the Dual-Memory Method and consist of two modules. A slow-learning module (orientated on the Hippocampus) and a fast-learning module (orientated on the Neocortex). The slow-learning module, named Module A within this work, is a pre-trained Feature-Extraction Architecture. This module can be fix or further adjusted while lifetime with small update steps. In this work this module is fixed, so learning rate is 0. This module extracts features and forward them to the second module, Module B. Module B is an incremental Classificator, who can be trained “on-the-fly”. A graphical overview of the Algorithm is given in Figure 1.

Figure 1: Schematic Overview of the Methodology for Distributed Incremental Class Learning Algorithm

1. Methodology (max. 1 Seite eher weniger, Ka)

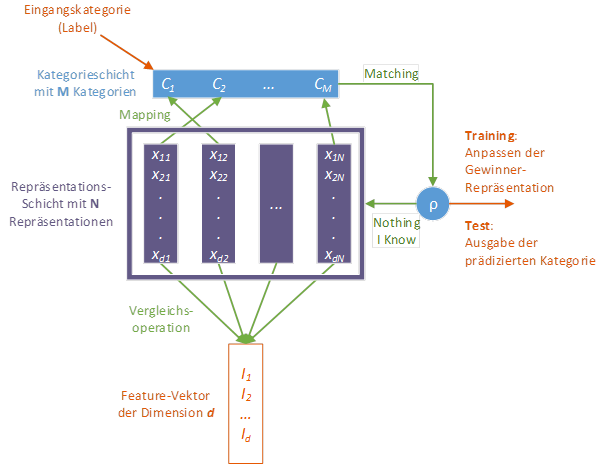
Feature Extraction is nowadays a task which can be effectively done by Deep Neural Networks (DNN). Depending on the application, different DNNs can be used for this module. On the one hand, the module should extract good and relevant features from the input data. But on the other hand, the DNNs should also run in real-time applications on (mobile) edge devices, e.g. on a Raspberry Pi. Therefore, the memory consumption and computational complexity of the DNN is also a relevant criterion. For these reasons, MobileNet-V2 is used as Feature Extraction module. It is designed for mobile application with low memory consumption and operations per input image. The available and pre-trained MobileNet-V2 Architecture within Tensorflow 2.0 is used. Pre-Training took place on ImageNet. The last fully connected layers of MobileNet-V2 are removed and the features will be the input to Module B.

Figure 2: Structure of the FuzzyARTMAP-Network

Module B is an incremental Classificator, which should generally meet the following requirements:

* Trainable on a Data Stream, where samples of the different classes appear randomly in time and order
* All the time there must be a working Classificator for the already seen and trained Classes
* Computational Complexity and Memory Consumption should be limited or grow slowly with respect to the number of known classes

These are general requirements for an incremental Classificator. Additionally the Algorithm within this work should not save any training data. Based on these Requirements a FuzzyARTMAP-Network was chosen. A big advantage is the property of the FuzzyARTMAP to solve the plasticity-stability dilemma due to his architecture. The FuzzyARTMAP-Network was adjusted slightly and the principal structure can be seen in Figure 2. The input data, here the extracted features from Module A, is a *d*-dimensional vector. In the so-called representation layer are *N* representations, which are each realized with a *d*-dimensional vector, too. Every representation performs a comparison operation. In this work this operation is realized with the cosine similarity. After this step, the Mapping from the representations to the corresponding classes takes place in the Category Layer. A Class can have more than one representation in this network. After this mapping the Categories have so called Matching-Values, which can be compared to an adaptive threshold (Vigilance Parameter). If the “winner”-representation has a matching value above the threshold, the winning class will be the prediction of the network while inference. In Training the winning representation will further be adjusted and generalized with the input data. If the threshold is not reached, the network declares “Nothing I Know”. In training phase, a new representation is created with the input vector and the training label is used to perform a correct mapping to the Category Layer. While Inference, the network forces the user to give a label for the input data, or it can just randomly give the input data a class label, e.g. 8 if 7 classes are known already.

Nothing I Know detects an unknown class with the help of an adaptive threshold. It is calculated based on the matching-values in the network and the number of category nodes. With a factor, the “clearness” of the winning node is adjusted. The final threshold is given by the following formula:

Depending on the application, the threshold could also be a fixed value for.

1. Experiments (1 Seite eher ein bisschen mehr, Ka / Ms)

Here introduce the paper, and put a nome­nclature if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, bold and 10 pt. Here follows further instructions for authors.

**Zwischenüberschrift: Experimental Setup (Ka)**

Parameter, Hardware

**Zwischenüberschrift: Incremental Learning Performance**

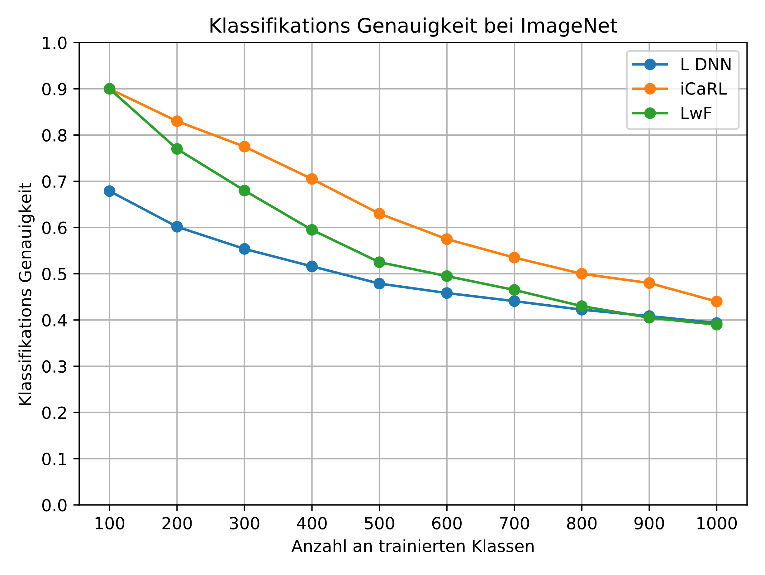
For an evaluation of the proposed algorithm different test on ImageNet are performed. As a reference, iCaRL and Learning without Forgetting (LwF) are used. iCaRL and LwF use a ResNet-Architecture as Feature Extraction Module. They perform 100 epochs per incremental step. iCaRL saves 20.000 training images to achieve these results. DICL perform 1 epoch per incremental step, so every training image is seen once. Additionally due to time restriction 10 training images per class are used, instead of up to 1.300, depending on the class. The results for the three Algorithms with 10 incremental steps are shown in Figure 3.

Figure 3: Results for ImageNet, 10 incremental steps

Kleinschrittigeres Lernen (Ka)

Einfluss der Konsolidierung (Ka)

**Zwischenüberschrift: Distributed Learning Performance**

Verteiltes Lernen auf ImageNET-10 mit 2, 5 oder 10 Geräten (Ka)

**Zwischenüberschrift: Bessere Performance mit ResNet (Ms)**

Anlehnung an Evaluationsdokumente

Weitere Experimente im Verlauf des Herbstes (Ms)

1. Prototype (max. 0,5 Seiten, Ms)

Here introduce the paper, and put a nome­nclature if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, bold and 10 pt. Here follows further instructions for authors.

Beschreibung des Raspberry-Pi-basierten Prototyps

Bild des Prototypen

Ggf. Abbildung der GUI aus Ausarbeitung

1. Conclusion and Transfer (max. 0,5 Seiten, Ms)

Here introduce the paper, and put a nome­nclature if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, bold and 10 pt. Here follows further instructions for authors.

Modularität: Einfache Anpassbarkeit auf Anforderungen (bessere Hardware erlaubt bessere Ergebnisse)

Transfer im Sinne eines Ausblicks auf potentielle Anwendungsfelder?

References

[1] Van der Geer J, Hanraads JAJ, Lupton RA. The art of writing a scientific article. J Sci Commun 2000;163:51-9.

[2] Strunk Jr W, White EB. The elements of style. 3rd ed. New York: Macmillan; 1979.

[3] Mettam GR, Adams LB. How to prepare an electronic version of your article. In: Jones BS, Smith RZ, editors. Introduction to the electronic age. New York: E-Publishing Inc; 1999. p. 281-304.