 ISO/IEC JTC 1/SC 29/AG 03 N0057

**ISO/IEC JTC 1/SC 29/AG 03  
MPEG Liaison and Communication   
Convenorship: KR**

**Document type:** Output Document

**Title:** White Paper on Neural Network Coding

**Status:** Approved

**Date of document:** 2022-01-23

**Source:** ISO/IEC JTC 1/SC 29/AG 03

**Expected action:** None

**Action due date:** None

**No. of pages:** 6 (with cover page)

**Email of Convenor:** kyuheonkim@khu.ac.kr

**Committee URL:** https://isotc.iso.org/livelink/livelink/open/jtc1sc29ag3

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/AG 03 MPEG VIDEO CODING**

**ISO/IEC JTC 1/SC 29/AG 03 N0057**

**January 2022, Online**

|  |  |
| --- | --- |
| **Title** | **White Paper on Neural Network Coding** |
| **Source** | **AG 03, MPEG Liaison and communication** |
| **Status** | **Approved** |
| **Serial Number** | **21272** |

**Neural Network Coding (NNC) –**

**Efficient Storage and Inference of Neural Networks for Multimedia Applications**

Artificial neural networks have been adopted for a broad range of tasks in almost every technical field, such as medical applications, transportation, network optimization, big data analysis, surveillance, speech, audio, image and video classification, image and video compression, and many more. Their recent success is based on the feasibility of processing much larger and complex neural networks (deep neural networks, DNNs) than in the past, and the availability of large-scale training data sets. An additional factor for the exponential growth is the appearance of new use cases, such as federated learning with continuous communication between many devices. Accordingly, this requires the highest compression in order to minimize the overall communication traffic, and reduce the size of networks for inference. Thus a standard for neural network coding (NNC) has been defined in ISO/IEC 15938-17 “Compression of Neural Networks for Multimedia Description and Analysis”.

Examples of specific applications targeted by the standard include (image) classification, image/video compression or federated training. In many (image) classification applications, trained models (and possibly updates) need to be deployed to a large number of target devices, such as mobile phones or smart cameras. In image/video compression, models adjusted to specific content characteristics may need to be sent frequently to the decoder. In federated training scenarios, updated models need to be exchanged frequently between nodes. All these use cases benefit from smaller serialized representations of trained models, and many of the target use cases also benefit from more energy efficient inference with models of reduced complexity.

# Scope of the specification and relation to exchange formats

As the coding of neural networks is likely to have a hardware dependent and hardware independent component, the NNC standard is designed as a toolbox of compression technologies. Some of these technologies require specific representations in an exchange format (i.e., sparse representations, adaptive quantization), and thus a normative specification for representing outputs of these technologies is defined. For other technologies (e.g. pruning), that do not require a certain representation, NNC also specifies generic metadata. Therefore, NNC is independent of a particular neural network exchange format. However, interoperability solutions with the most common formats are described in the annexes of the specification, including PyTorch[[1]](#footnote-1), TensorFlow[[2]](#footnote-2), Open Neural Network Exchange (ONNX)[[3]](#footnote-3), and the Neural Network Exchange Format (NNEF)[[4]](#footnote-4). Interoperability with these formats is specified in two ways: Either, NNC is used independently by compressing all parameter tensors of a neural network and including the respective NN structure or connection graph into the bitstream, or NNC is used within an external framework by also coding neural network parameters tensor-wise, while all structure data is handled by the external framework.

# Features of the standard

Achieving compact representations of trained neural networks addresses two main goals: (i) providing efficiency when the NN is stored or transmitted and (ii) allowing for resource-efficient inference. The importance of these goals depends on the specific use case. For example, for frequent updates in a federated training scenario involving nodes in a cloud infrastructure the efficient transmission is most important, while for infrequent deployments of a trained NN to an embedded device supporting efficient inference is crucial. Methods addressing the second goal also support the first one, and need to output a representation that can be used directly for inference (at least on specific target platforms), while methods addressing the first one will have additional coding steps requiring decoding at the receiving end.

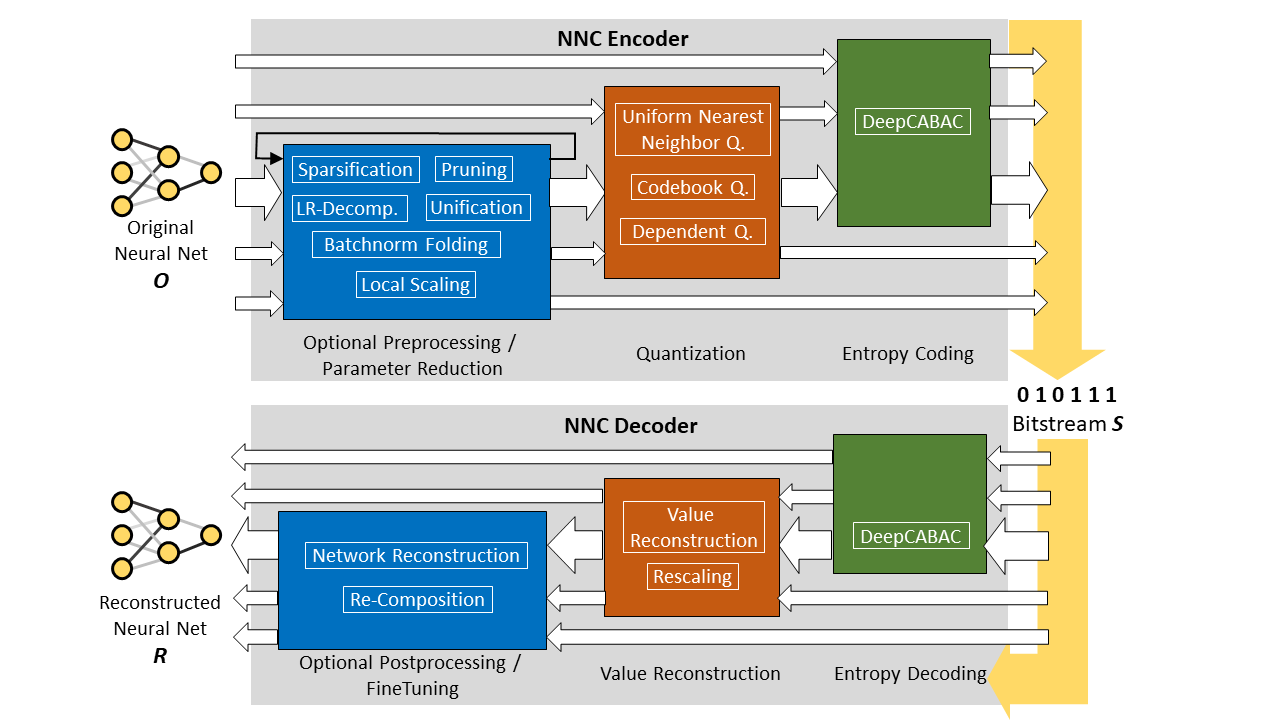


Fig. 1 NNC Overview.

NNC contains the following processing steps with associated coding tools:

**Parameter reduction methods** process a model to obtain a more compact representation. Examples of such methods include *parameter sparsification, parameter pruning, weight unification,* and *decomposition methods.*

*Sparsification* processes parameters or groups of parameters to produce a sparse representation of the model, e.g., by replacing some weight values with zeros. The sparsification may generate additional metadata (e.g. masks). The sparsification can be structured or unstructured. NNC includes methods for unstructured sparsification with compressibility loss, structured sparsification using micro-structured sparsification.

*Unification* processes the parameters to produce groups of similar parameters. Unification does not eliminate or constrain the weights to be zero, but it lowers the entropy of model parameters by making them similar to each other. NNC includes a method for weight unification.

*Pruning* reduces the number of parameters by eliminating parameters or groups of parameters. The procedure results in a dense representation which has less parameters in comparison to the original model, e.g., by removing some redundant convolution filters from the layers. NNC includes a method for combined pruning and sparsification.

*Decomposition* performs a matrix decomposition operation to change the structure of the weights of a model. This standard includes a method for low rank/low displacement rank for convolutional and fully connected layers.

Along with the reduction methods mentioned above, this standard includes decomposition methods that are introduced and tested as part of a parameter quantization technique. Examples of such methods are batchnorm folding and local scaling adaptation.

The parameter reduction methods could be combined or applied in sequence to produce a compact model.

**Parameter quantization methods** reduce the precision of the representation of parameters. If supported by the inference engine, the quantized representation can be used for higher-efficient inference. NNC includes methods for uniform quantization, codebook-based quantization, dependent scalar quantization, and iterative QP optimization.

**Entropy coding methods** encode the results of parameter quantization methods. This standard includes DeepCABAC as entropy encoding method

More details about the methods are described in an overview paper [1].

# Performance and reference software

The NNC standard provides a compression efficiency of up to 97% for transparent coding use cases, i.e. without degrading the classification and inference capability of the respective NN. This is reflected by the obtained evaluation results, where compression efficiency in terms of compressed bitrate vs. original NN bitrate is analyzed. Here, performance metrics for relevant use cases in multimedia for the original as well as decoded and reconstructed network are evaluated, such as constant top-1 and top-5 classification accuracy for image classification. In addition, much higher coding gains can be obtained if the classification accuracy is allowed to drop.

The NNC compression performance is evaluated for a verification dataset. The dataset includes three models (VGG16, ResNet50, MobileNetV2) for image classification, one model (DCase) for audio classification and an image autoencoder (UC12B). The experiments were carried out, using the standard reference software NCTM (Neural Network Compression Test Model, version 6.0).

Table 1: NNC Transparent Coding Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | *cr* in % | Top-1 / Top-5 acc. reconstr. | Top-1 / Top-5  acc. original | Orig. size (bytes) |
| VGG16 | 2.98 | 70.51 / 89.54 | 70.93 / 89.85 | 553.43 M |
| ResNet50 | 6.54 | 74.42 / 91.80 | 74.98 / 92.15 | 102.55 M |
| MobileNetV2 | 12.18 | 71.13 / 90.06 | 71.47 / 90.27 | 14.16 M |
| DCase | 4.12 | 58.15 / 92.35 | 58.27 / 91.85 | 467.26 k |
| Model | ***cr* in %** | **PSNR / SSIM**  **reconstructed** | **PSNR / SSIM**  **original** | **Orig. size (bytes)** |
| auto-encoder for image compression | 17.34 | 29.98 / 0.954 | 30.13 / 0.956 | 304.72 k |

The coding results for the NN test set are provided in Table 1, showing transparent results at working points with similar accuracy of original and reconstructed pre-trained NNs for Top-1 / Top-5 classification accuracies and Peak Signal-to-Noise Ratio (PSNR) / Structural Similarity Index Measure (SSIM) for image compression. The NNC compression capability is given as compression ratio *cr* in Table 1: As an example for VGG16 with *cr* = 2.98%, NNC is able to compress the neural network from ~553 Mbyte to a bitstream size of ~16.5 Mbyte without performance degradation. More details about the evaluation procedure and further results can be found in [1].

For assessing coding contributions of individual tools, an exemplary analysis is given in Fig. 2 for VGG16 and MobileNetV2. Regarding overall coding pipelines, low rank decomposition reduces the overall number of parameters and provides data reduction as well. In contrast, sparsification and unification are not included, as these tools only reduce the number of unique values, but do not reduce the data size themselves. Therefore, Fig. 2 shows results for the (pre-trained) original and low rank decomposition pipelines. For ease of results comparison, the working points for Fig. 2 are selected such that the models achieve top-1/top-5 accuracies of 70.5%/89.5% for VGG16 and around 68.6%/88.8% for MobileNetV2 only varying in a small range. Note that VGG16 does not contain batch norm layers, hence BN does not add coding gains for this NN.

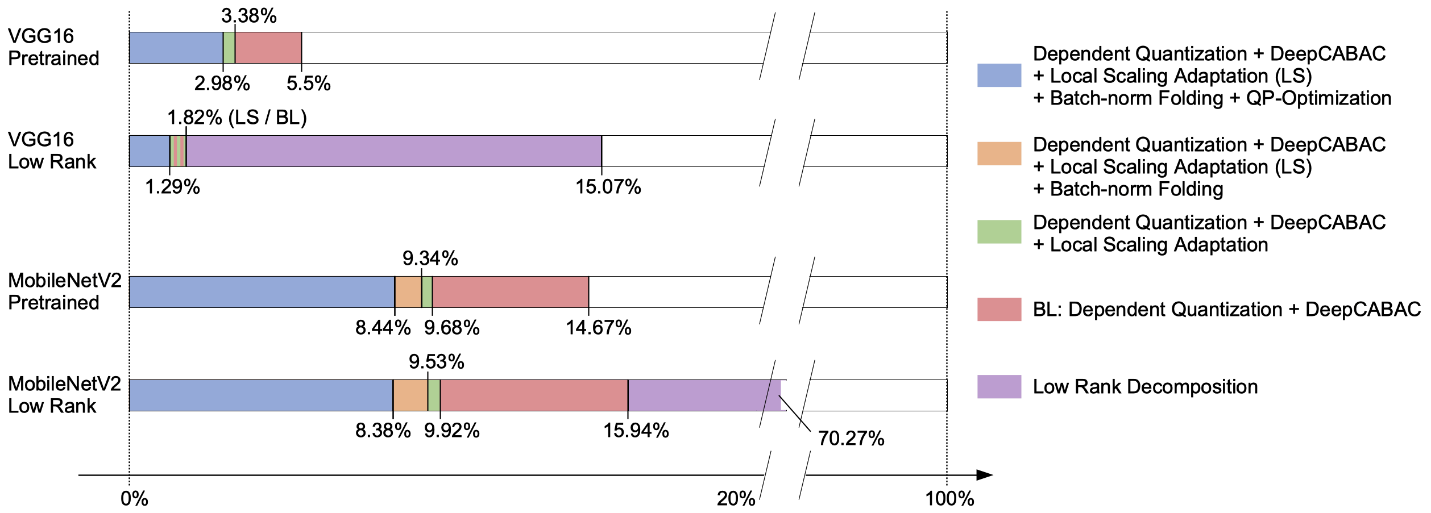


Fig. 2 Successive (lossy) coding results in terms of compression ratio for individual coding tools for pretrained original and low rank decomposition pipelines.

A complete implementation of the standard is provided as reference software in ISO/IEC 15938-18 (to be finalized in fall 2022).

# Outlook: Coding of incremental neural network updates

A second edition of the NNC standard, which focuses on incremental updates of neural networks is under preparation. Incremental coding deals with a base neural network (i.e. an instance of a trained neural network for the particular use case) and an updated neural network, which represents an incremental update with respect to the base neural network. The updated neural network is typically the result of one of the following operations (this list is considered non-exhaustive):

* The base neural network is retrained with other data or parameters.
* The base neural network and the updated neural network are compressed versions of the same network with different compression ratio.
* The updated neural network is the result of applying transfer learning, starting from the base neural network.
* The updated neural network uses (part of) the base neural network in its structure, possibly with retraining (parts of) the base neural network.

# References

[1] H. Kirchhoffer et al., "Overview of the Neural Network Compression and Representation (NNR) Standard," in IEEE Transactions on Circuits and Systems for Video Technology, doi: 10.1109/TCSVT.2021.3095970.

1. https://pytorch.org/ [↑](#footnote-ref-1)
2. https://www.tensorflow.org/ [↑](#footnote-ref-2)
3. https://onnx.ai/ [↑](#footnote-ref-3)
4. https://www.khronos.org/nnef [↑](#footnote-ref-4)