Training machine learning on JPEG compressed images

M.Pistono[1,2], G.Coatrieux[1], J-C.Nunes[2] and M.Cozic[3]

1. IMT Atlantique Bretagne Pays De La Loire, LaTIM Inserm UMR1101, Brest 29238, France; 2. Universite de Rennes 1, LTSI Inserm UMR1099, Rennes 35042, France; 3. MEDECOM, Plougastel Daoules 29470, France;

Introduction

- **Machine learning** (ML) algoritms allows performing many complex tasks.
- ML work accordingly to two main stages: i) training during which ML model parameters are computed; ii) classification when the model is exploited.
- Training stage needs a **lot of data** and, in the case, of images, data volumes are important.
- ML algorithms work on raw images pixels without considering images are mostly stored in a compressed form. Thus, training complexity is handicapped by the image decompression process.
- There is an interest to see if it is possible to directly train ML models from compressed data.

Objectives

- Quantify and understand the influence of feeding ML model with partially JPEG decompressed images for various JPEG quality based on :

  1. **ML model accuracy.**
  2. **ML model training time.**
  3. **JPEG decompression time complexity**.

- Identify a **trade off** in-between loss of ML model accuracy and complexity gain or execution time when working with partially JPEG decompressed images.

JPEG partially decompressed datasets

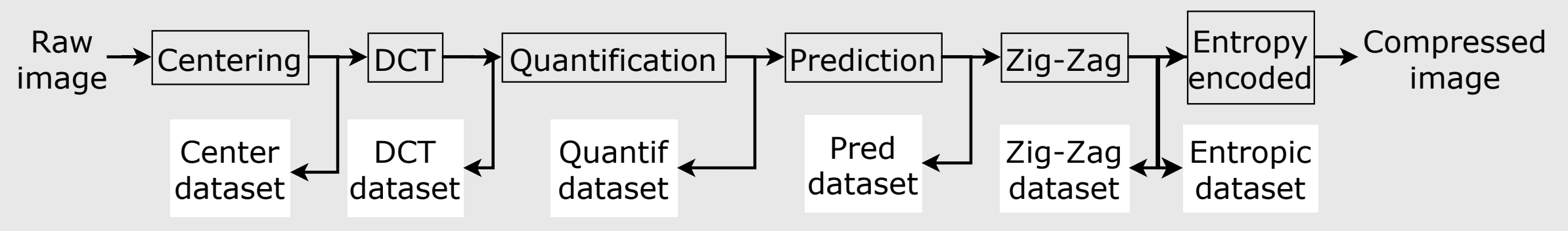


Figure 1: main JPEG steps for grayscale image

ML model were trained from data sets available at different steps of the JPEG compression phases for grayscale images. Taking an image database compressed accordingly to 5 different JPEG quality levels (100, 90, 80, 70, 60), 6 datasets were created per compressed image database:

- ***Center*** - after the pixel dynamic centering.
- ***DCT*** - image DCT coefficients of 8x8 pixel blocks.
- ***Quantiz*** - quantized DCT coefficients.
- ***Pred*** - AC DCT Coefficients along with predicted DC DCT coefficients.
- ***Zig-Zag*** - DCT coefficients reorganized accordingly to the well-known Zig-Zag transform (see Fig. 2).
- ***Entropic*** - DCT coefficients in their entropic encoded form and linearly organized as illustrated in Fig.3. The purpose of this dataset is to evaluate the impact of the Zig-Zag transformation on ML model accuracy.

In the following experiments, ML models are trained with only one of these datasets.

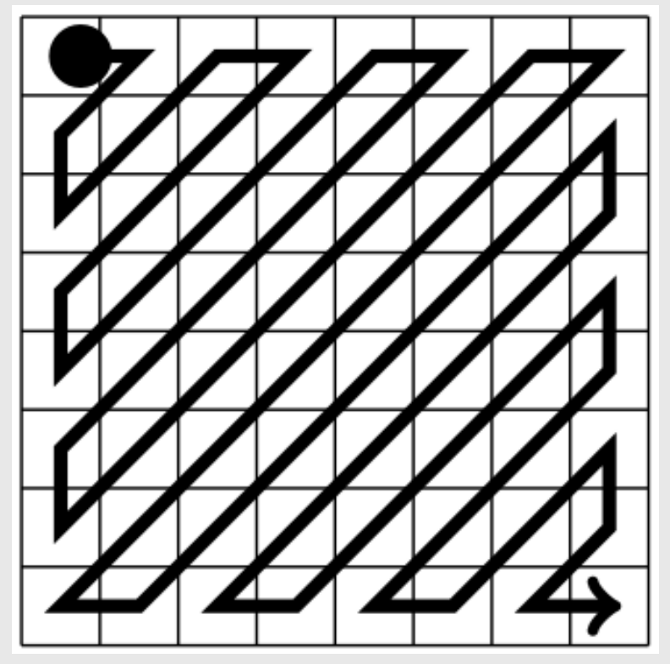


Figure 2: Zig-Zag dataset reorganisation

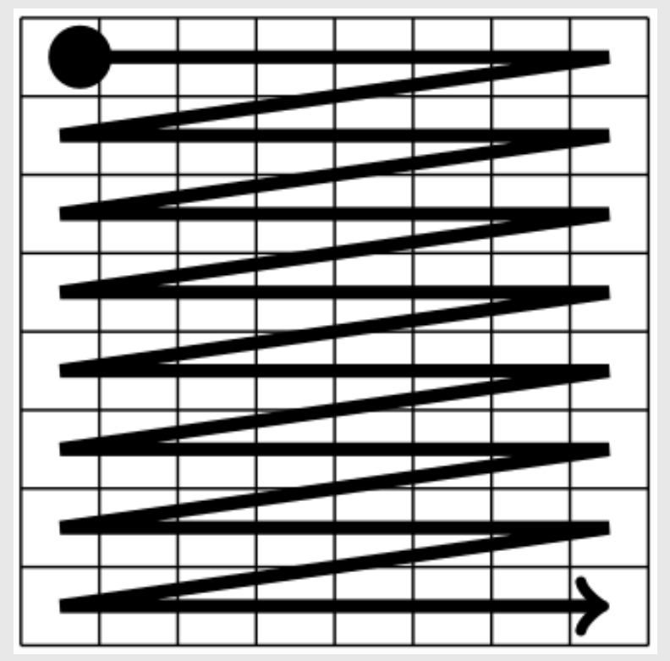


Figure 3: Entropic dataset reorganisation

Experimented Machine Learning models for classification

1.Neural Network (NN)

- A ML model is composed of a set of neurons (Fig.4). Layers in between the NN input and output layers are referred to as hidden layers. In the case one NN has several hidden layers it is considered as a deep neural network.
- One neuron is defined by : a weight vector w and a bias b and takes as input a vector x to compute : $y = \sigma(w^t x + b)$, where $\sigma(\cdot)$ is the activation function (e.g. ReLU, logistic).
- The training stage purpose is to find neurons' weights.

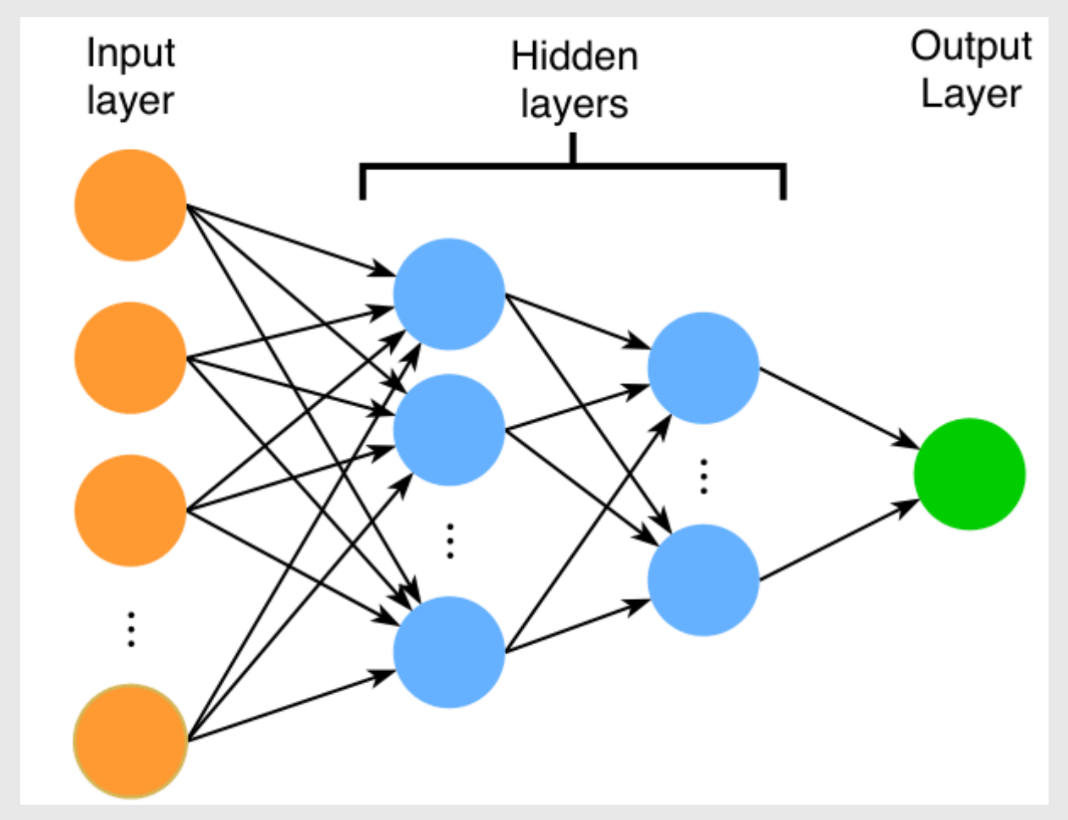


Figure 4: Exemple of NN structure

Experimented NN models and their parametrization -

- The Fu and Guimaraes [1] model - a single hidden layer NN of 1024 neurons with the sigmoid function, fully connected to the input layer, regularized by dropout and followed by a softmax layer.
- A double hidden layer model – a NN of 750 and 200 fully connected neurons activated by the sigmoid function, respectively, with a dropout fixed to $p = 0.6$ for the first layer which then feeds a softmax layer.
- Each model was trained for 200 epochs with a batch size of 128 using the adadelta optimizer.

2.Convolutional Neural Network (CNN)

CNN are deepneural networks the first layers of which are composed of convolutional layers (CL) sometimes associated with pooling layers (PL) used to reduce space dimensions, see Fig.5. One CL layer is defined by the order of its filter and a the filter shift in pixels at each step (stride). Common PL layers are mean or maxpooling.

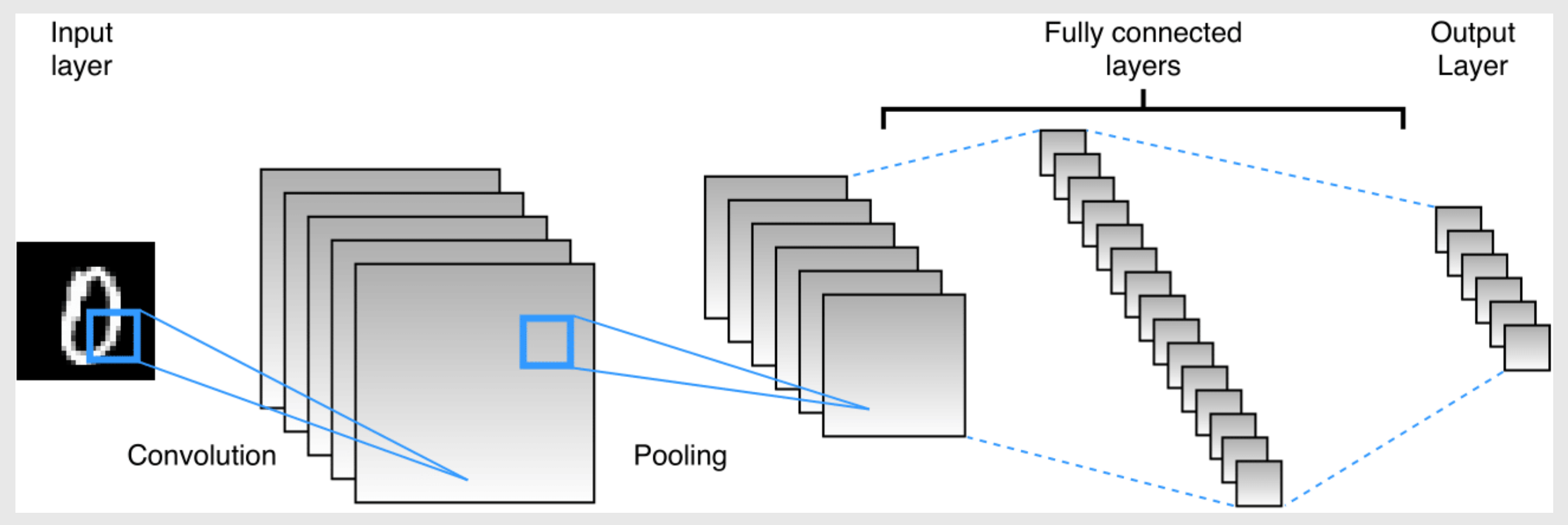


Figure 5: Exemple of CNN structure

Experimented NN models and their parametrization -

- The Ulicny and Dahyot [2] (U&D) model with or without batch normalization (BN) after each layer - see Tab.1 for parameters' values.
- The Keras CNN model - parameterization is given in Tab.2.
- Models were trained for 300 epochs with a batch size of 256, using SGD optimizer with a learning rate of 0.1, and a momentum of 0.85.

Ulicny and Dahyot CNN architecture

Layer	Conv1	Conv2		Conv3	MaxPool1	Conv4	MaxPool2		Dense	
Dimension	64	64	Dropout1	64		128		Dropout2	512	Dropout3
Kernel	4×4	3×3	$p = 0.25$	3×3	3×3	3×3	3×3	$p = 0.25$		$p = 0.5$
Stride	2×2	1×1		1×1	2×2	1×1	2×2			

Table 1: Ulicny and Dahyot architecture

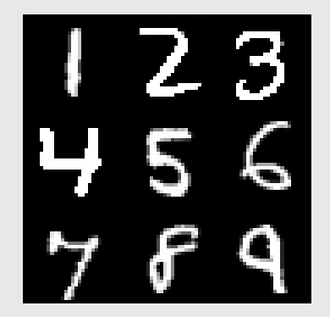
Keras CNN architecture

Layer	Conv1	Conv2	MaxPool1		Conv3	Conv4	MaxPool2		Dense	
Dimension	32	32		Dropout1	64	64		Dropout2	512	Dropout3
Kernel	3×3	3×3	2×2	$p = 0.25$	3×3	3×3	2×2	$p = 0.5$		$p = 0.5$

Table 2: Keras CNN architecture

Experimental results

1. NN experiments on MNIST



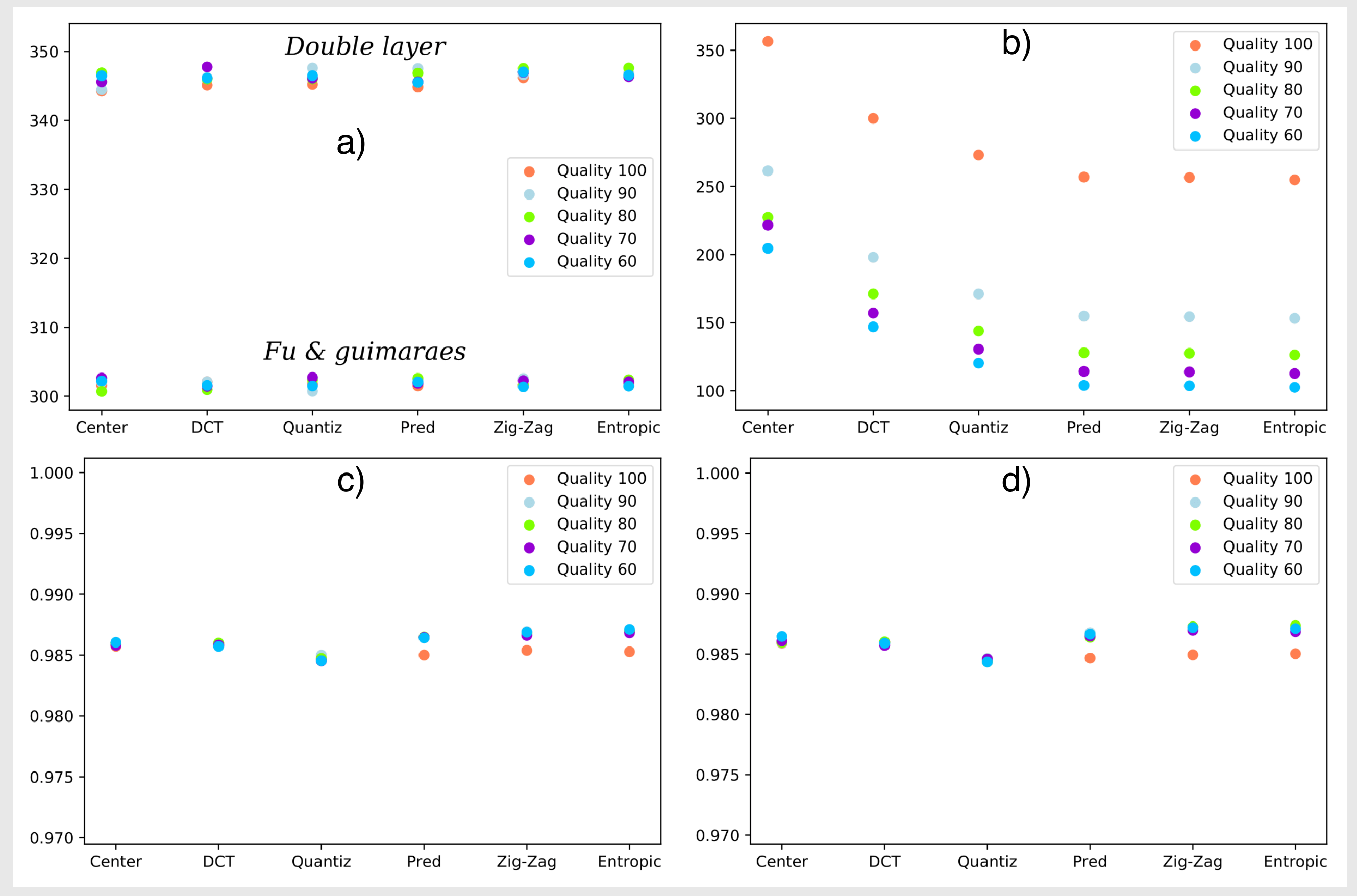
MNIST collection: 70 000, 28×28 images of handwritten digits with 60 000 training images and 10 000 test images.

Result analyzis

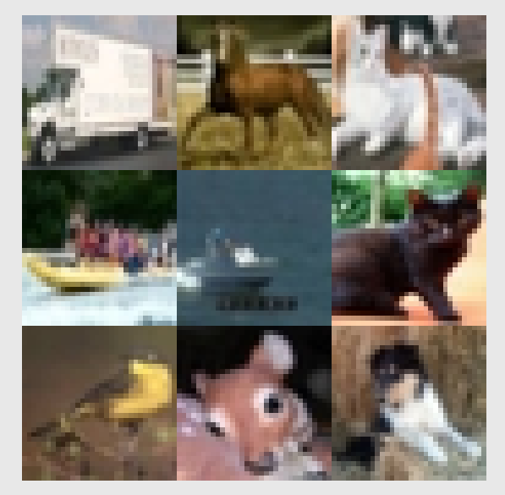
- Partial decompressed JPEG dataset and JPEG quality factor have low impact on NN accuracy (Fig.6 c and d) and training time (Fig.6 a).
- Nearly 50% decompression time can be saved working with Entropic dataset (Fig.6 b).

⇨ Feeding NN with partially decompressed images seems not increasing training time, not diminishing accuracy and allows a computation gain by not fully decompressing images.

Figure 6: MNIST a) NN training time; b) decompression time; c) Fu & Guimaraes accuracy; d) double layer accuracy



2. CNN experiments on CIFAR-10



CIFAR-10 collection: 60 000 32×32 color images organized in 10 classes (airplanes, birds, cars, ...). 50 000 images constitute the training set and the others the test set.

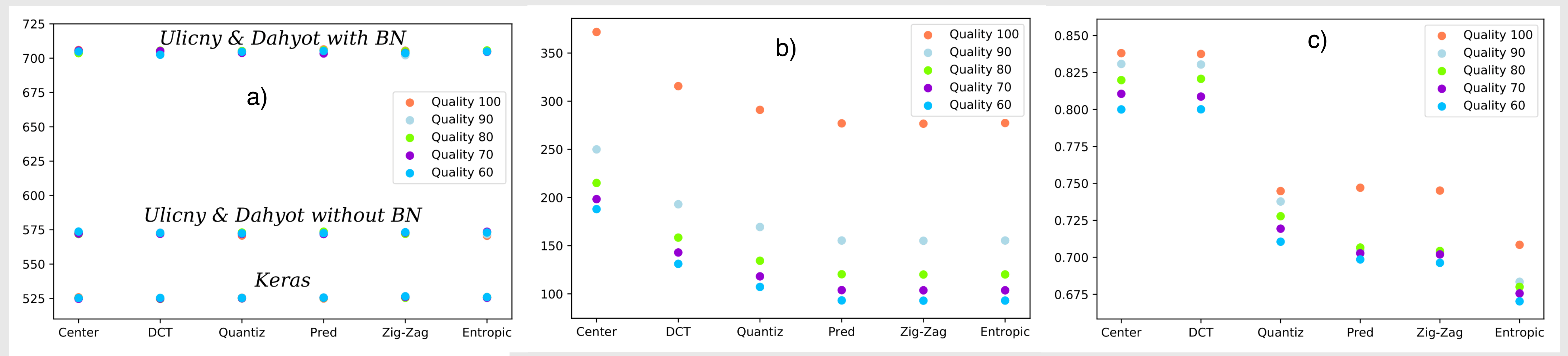
In this experiment, images are grayscale converted.

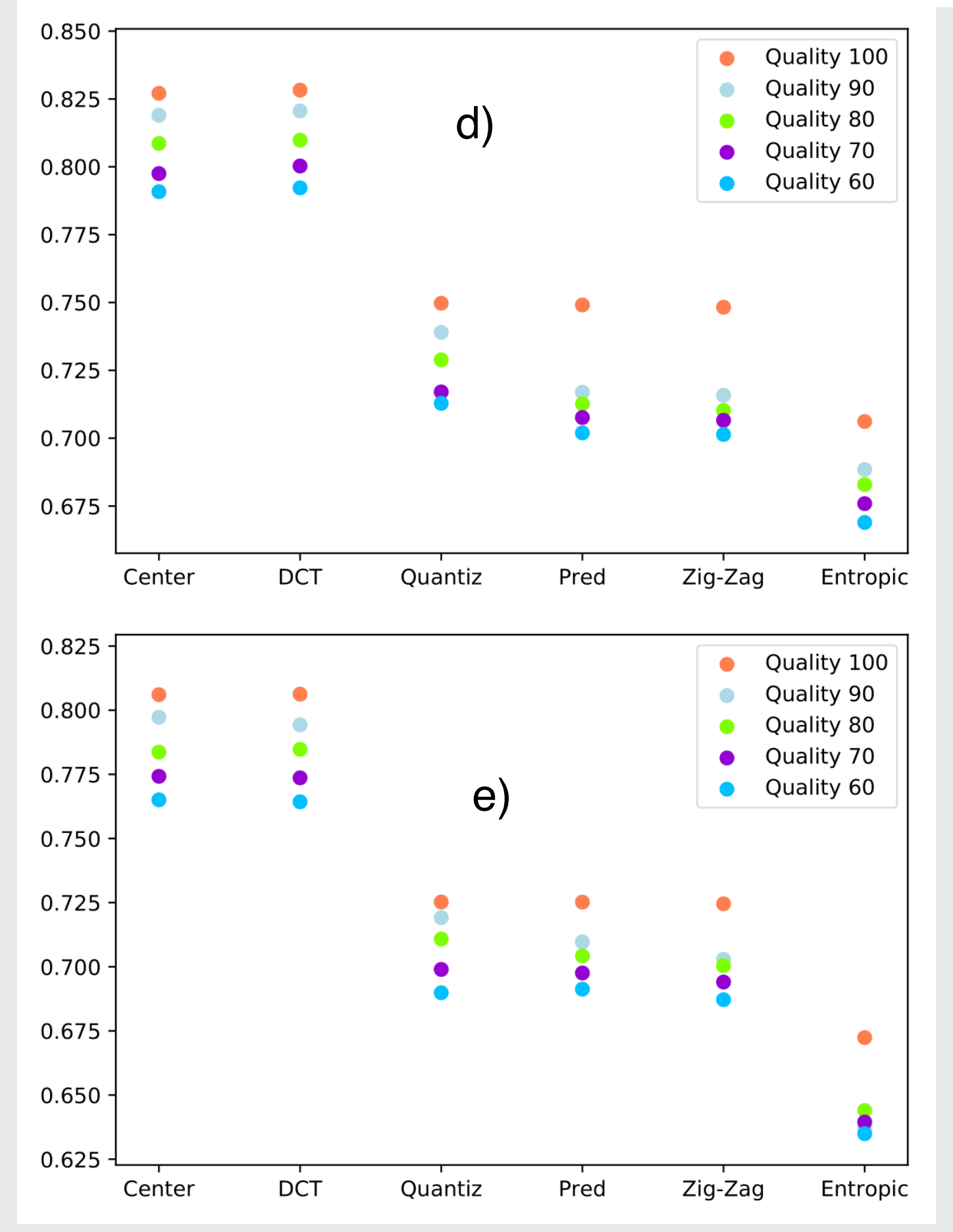
Result analyzis

- JPEG DCT, block reorganization and quality significantly impact CNN accuracy (Fig.7 c, d and e).
- CNN convolutionnal layers may not be suited in the DCT frequency space.
- Organization of entropic dataset needs to fit to the convolution kernel size.
- CNN training time is not dependant of the input dataset (Fig.7 a).
- Decompression time (Fig.7 b) can be saved at the price of an accuracy loss.
- CIFAR-10 image content complexity, along with the decrease of the JPEG quality factor also impact accuracy.

⇨ New CNN architectures have to be studied.

Figure 7: CIFAR-10 - a) CNN training time; b) decompression time; c) U&D accuracy; d) U&D accuracy with BN; e) Keras CNN model accuracy





Conclusion

- **NN models do not suffer of accuracy loss whatever the partially decompressed data set.**
- By training NN model on only de-Zig-Zag transformed or Entropic decoded data, one can gain up to 49% of computation time on image decompression.
- **For CNN models a compromise has to be found in-between decompression computation time and model's accuracy.**
- Feeding CNN models with data issued from *Pred dataset* allows saving 50% of the decompression computation time, at the price of an accuracy loss of 7.5% on Keras model.
- **Specific model architecture have to be investigated** so as to take into account the: JPEG partitioning in 8 × 8 blocks, DCT transform (frequency domain) and Zig-Zag transform (1D space).
- These preliminary results have to be confirmed with experiments with more image data sets and more complex classification.
- Future works will investigate if model training can be more efficient only using a subset of partially decompressed data.

References: [1] D. Fu and G. Guimaraes. Using compression to speed up image classification in artificialneural networks, 2016.
[2] Matej Ulicny and Rozenn Dahyot. On using cnn with dct based image data. InProceedings of the 19th Irish Machine Vision and Image Processing conference, 2017.