Research Article

Lithology classification of whole core CT scans using convolutional neural networks

Kurdistan Chawshin[1] · Carl Fredrik Berg[1] · Damiano Varagnolo[2] · Olivier Lopez[3]

Received: 3 February 2021 / Accepted: 11 May 2021
Published online: 28 May 2021

Abstract
X-ray computerized tomography (CT) images as digital representations of whole cores can provide valuable information on the composition and internal structure of cores extracted from wells. Incorporation of millimeter-scale core CT data into lithology classification workflows can result in high-resolution lithology description. In this study, we use 2D core CT scan image slices to train a convolutional neural network (CNN) whose purpose is to automatically predict the lithology of a well on the Norwegian continental shelf. The images are preprocessed prior to training, i.e., undesired artefacts are automatically flagged and removed from further analysis. The training data include expert-derived lithofacies classes obtained by manual core description. The trained classifier is used to predict lithofacies on a set of test images that are unseen by the classifier. The prediction results reveal that distinct classes are predicted with high recall (up to 92%). However, there are misclassification rates associated with similarities in gray-scale values and transport properties. To postprocess the acquired results, we identified and merged similar lithofacies classes through ad hoc analysis considering the degree of confusion from the prediction confusion matrix and aided by porosity–permeability cross-plot relationships. Based on this analysis, the lithofacies classes are merged into four rock classes. Another CNN classifier trained on the resulting rock classes generalize well, with higher pixel-wise precision when detecting thin layers and bed boundaries compared to the manual core description. Thus, the classifier provides additional and complementing information to the already existing rock type description.

Article Highlights

- A workflow for automatic lithofacies classification using whole core 2D image slices and CNN is introduced.
- The proposed classifier shows lithology-dependent accuracies.
- The prediction confusion matrix is exploited as a tool to identify lithofacies classes with similar transport properties and to automatically generate lithofacies hierarchies.

Keywords X-ray computerized tomography · Convolutional neural network · Classification · Lithofacies

✉ Kurdistan Chawshin, kurdistan.chawshin@ntnu.no | [1]Department of Geoscience and Petroleum, NTNU, S. P. Andersens veg 15, 7031 Trondheim, Norway. [2]Department of Engineering Cybernetics, NTNU, O. S. Bragstads Plass 2D, 7034 Trondheim, Norway. [3]Equinor ASA, Arkitekt Ebbells veg 10, 7053 Ranheim, Norway.

1 Introduction

Classifying lithofacies is an essential step toward characterizing reservoirs and better understanding their depositional environments. To predict reservoirs' saturation levels, and to perform subsequent effective reservoir modeling, it is crucial to correctly assess lithological properties such as grain size, grain shape, sorting and cementation. These lithological properties affect the petrophysical and transport properties of the reservoir rocks (e.g., porosity and permeability).

Conventional well-log interpretations performed for lithology classification typically overlook the heterogeneities below the log resolution. Currently, the whole cores extracted from wellbores are described through direct visual inspections by a team of geologists and/or petrophysicists. However, this process is time-consuming and the resulting facies classification can be affected by subjective interpretation.

The extraction of whole core data is currently requiring significant capital investment. Therefore, rapid and automated core classification and associated core analysis is seen as a key technology for enabling improved return on investments and to enhance the overall decision processes [36].

X-ray computerized tomography (CT) imaging is seen as one of the most effective nondestructive methods for inspecting whole cores at a submillimeter resolution, and the resulting digital image of the core is an aid toward the automation of the core classification process. CT images can indeed be incorporated in the classification workflow for a rapid lithology classification [10]. Whole core CT scanning has a long history in assisting the geologists to study extracted cores [39]. More precisely, 2D and 3D whole core CT scans provide high-resolution (submillimeter) information on the texture, composition and internal structure of the reservoir rocks. Moreover, whole core CT imagery may be performed in the early stages of the facies analysis process: these data can be employed before extrusion, when the core is still in an aluminum barrel [36].

From technical standpoints, each voxel in the CT images is represented by a gray-level value that indicates a certain level of X-ray attenuation. This grayscale value, and thereby the attenuation, is a function of the density and effective atomic number of the underlying material [36]. Since the first generation of the CT scanners, the scanning technique has gone through extensive refinements, and current CT images can predict 2D and 3D distribution of the chemical composition and density of the whole core [19]. This information, together with the fact that the whole core scans are stored digitally, aids laboratory analyses of the internal structure of the cores to be used in rock characterization and evaluation of plug drilling locations. Recent improvements in CT scanning and reconstruction algorithms, combined with developments in computing power and image analysis, have opened new possibilities for extracting even more information from whole cores, and thereby enhancing their value in operational settings and facilitating the automation of the core classification process.

The application of supervised and unsupervised machine learning algorithms has found significant use in many disciplines, including the petroleum industry. Recently, exploration and production companies have been extensively interested in the analysis of large data and automated solutions to reduce operational inefficiencies that slow down decision-making processes with associated losses of revenue [5].

Machine learning algorithms, especially artificial neural networks and support vector machines, have been successfully applied in several research studies to classify lithofacies and to estimate petrophysical properties using well log or core plug measurements [1, 2, 8, 10, 15, 18, 22, 25, 31, 33, 38, 41, 42, 49, 52].

In regard to image-based lithology classifications, several publications have utilized deep learning approaches to classify lithology based on the optical core photographs, borehole image logs, thin sections, and microtomographic images. De Lima et al. [12, 13] employed deep learning and transfer learning technique to classify core images of carbonate rocks. In another publication De Lima et al. [14] explored the use of deep convolutional networks to accelerate the microfacies classification based on rock thin sections. Valentin et al. [50] introduced a methodology for automatic lithofacies identification based on ultrasonic and microresistivity borehole images and a deep residual convolutional network. Baraboshkin et al. [6] compared the performance of several well-known neural network architectures (AlexNet, VGG, GoogLeNet, ResNet) to classify rock types based on the optical core images. Moreover, deep learning technique was utilized by Anjos et al. [4] to identify lithological patterns in carbonate rocks based on the microtomographic images.

In the majority of the aforementioned publications, either well log data or core analysis data have been used as inputs for the models learning phase. However, a recent trend is integrating both pieces of information together, potentially with also multiscale images. More specifically, Al-Obaidi et al. [3] used a combination of rock fabric properties extracted from image logs and well log-based petrophysical and compositional estimations to perform an automatic rock classification using a *k*-means based clustering method.

While artificial intelligence has been extensively employed for facies classification and petrophysical property estimations based on well log and core analysis data, there have been a few approaches that utilize CT images for facies classification and flow property estimations. These approaches employ information content of the CT images through the extraction of various features for clustering and classification purposes. Hall et al. [19] preprocessed the whole core CT images, extracted statistical features from processed images, and trained a Random Forest classifier to identify bioturbated core intervals. Odi and Nguyen [36] utilized physical features such as density, porosity and photoelectric effect, extracted from dual-energy CT scans, for supervised and unsupervised geological facies classifications. Moreover, the models were trained to learn the relationship between the CT extracted physical features and existing user-defined geological facies description.

Gonzalez et al. [17] considered a workflow for an automatic rock classification that combines conventional well logs, whole core CT images, optical core photographs, and routine core analysis (RCA) data. In this workflow, rock-fabric-related features are first extracted from whole core CT images and core photographs and then used to determine the rock classes by means of a clustering algorithm. Initially, the authors assumed several rock classes, and then they optimized this number by iteratively increasing the number of classes and minimizing a permeability-based cost function below a certain threshold. The obtained rock classes were finally used to train an artificial neural network to predict the classes from well log data. shin et al. [10] employed Support Vector Machines (SVM) to automatically classify lithofacies using the first order statistics and gray-level co-occurrence matrix (GLCM) features extracted from 2D cross-sectional whole core CT images. The authors used an SVM model to learn the relationship between the extracted features and expert-derived manual core descriptions.

In the mentioned publications, facies classification is performed using information content of the CT images in the form of various statistical and textural features. However, the CT images are not directly used as input for machine learning-based classifications.

In this study, we propose a workflow for automatic lithofacies classification that uses whole core CT image slices as input to train a CNN model. In the proposed approach, the need for manual feature extraction is eliminated as relevant features are learned by the network while it is being trained on a set of CT images. The obtained results reveal that the trained classifier is able to distinguish certain lithofacies classes with satisfying accuracy. However, lithofacies classes with similar texture and grayscale values are confused. In our workflow, the information acquired from prediction results is utilized to evaluate the misclassified lithofacies classes in terms of similarities in the transport properties. Further, as a post-classification processing step, hierarchical clustering analysis is performed to automatically cluster similar lithofacies classes using the prediction confusion matrix and then these results, together with porosity–permeability relationships, are used to group 20 lithofacies classes into 4 rock classes.

2 Methodology

In brief, we propose an automatic lithofacies classification workflow that uses whole core CT images and CNN and that is summarized in Fig. 1. The whole approach starts with preprocessing of 2D DICOM (Digital Imaging and Communication in Medicine) images. Lithofacies labels are then assigned to the processed images based on a user-defined geological core description. Lithofacies simply refers to a lithological subdivision that is distinguishable by its texture, grain size and the depositional environment. The labeled images are further augmented and used as inputs to train a CNN classifier. The trained classifier is then validated on a set of unseen images to predict lithofacies classes. Then, lithofacies classes that are deemed to be sufficiently similar are combined into rock classes (i.e., a combination of similar lithofacies classes form a rock class); in this step, the similarity indexes are computed starting from assessments of the transport properties (porosity and permeability) together with the degree of confusion in the confusion matrix resulting from the learning algorithm. Further, the classifier is coarsened with respect to

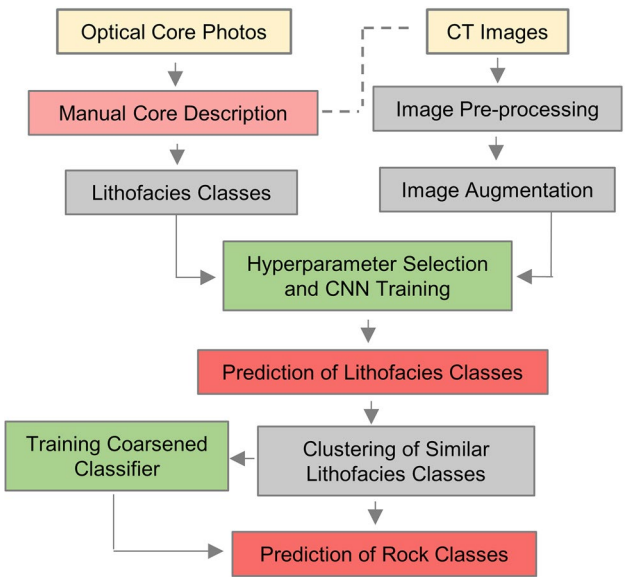

Fig. 1 Proposed workflow for lithofacies classification

the number of rock classes; in other words, the classifier is trained with a smaller number of rock classes. Finally, the coarsened classifier is employed to predict rock classes on a set of unseen images.

In the following subsections, the CNN algorithm and its general architecture will be explained in detail, followed by image preprocessing and image augmentation processes employed in this study.

2.1 Convolutional neural networks

Convolutional neural networks (CNN) have found significant applications in many sciences and industries. They have proven to be specifically effective in the fields of image recognition, voice recognition and classification. In general, neural networks draw the inspiration from the human brain. As mentioned before, this class of algorithms learns the relevant features directly from the input training data, so there is no need for manual feature extraction by a subject matter expert. Most of the modern CNN architectures consist of alternating convolutional and pooling layers followed by fully connected layers. The convolutional and pooling layers deal with feature extraction, while the fully connected layers map these extracted features into the final output. For an extensive discussion on CNN, we refer the interested reader to [53].

In the convolutional layers, a convolution operation is performed, i.e., a set of optimizable convolutional kernels are superposed in each position of the image represented by a 2D array of pixels. An element-wise multiplication between the elements of the kernel and the receptive field in the input image is performed, and the product results are summed up and stored in the corresponding position in the output feature map. Once the convolution operation is computed and stored for that specific location, the kernel is then moved either horizontally or vertically by an offset called *stride*. This process is repeated until the entire image is covered and the resulting feature map is completely populated. Convolutional layers are locally connected, whereas in the classic neural networks each neuron is fully connected to the neurons in the other layers.

To introduce nonlinearity, the outputs of the convolution operations pass through an activation function. The most common activation function is the rectified linear unit (ReLU); the advantage of using this specific function is that it allows fast and effective convergence during the training process. The feature map output of the convolutional layer records the exact position of the existing features in the input image. Therefore, minor spatial changes in the input image will yield a different feature map. To address this problem, a pooling layer is added after applying the nonlinear activation function (e.g., ReLU) to the feature map output of the convolution operation. A pooling operation is selected to be applied on each individual feature map. Two common pooling functions are average pooling and maximum pooling. The advantage of the added pooling layer is that the pooled feature map becomes invariant to local translations and spatial variations in the input image, e.g., edges, angles, feature positions, etc. [24].

The downsampled feature map outputs derived from the final pooling layer are then flattened into a 1D array of values that is connected to one or more fully connected layers that are referred to as dense layers. Here, input nodes are connected to output nodes by learnable weights [53]. The extracted features are eventually mapped into the final output of the network through the fully connected layers. Nonlinearities may also be introduced in the fully connected layers by adding an activation function (such as ReLU) following each fully connected layer.

Note that the activation function applied to the final fully connected layer is normally different than the other layers, and it is selected depending on the type of the task, i.e., classification and regression. A common activation function for multiclass classification is the so-called "softmax" function that returns the probability distribution of the predicted classes, i.e., it converts the output of the last layer into the predicted output class probabilities.

2.2 Information on the type of available data

The provided CT scan data consist of individual cross-sectional image slices from each core interval. Therefore, the number of image slices differ for each core, since depending on both the length of the core itself and the corresponding vertical image resolution (i.e., how many images are taken per meter of core). As an example, if the vertical image resolution is 0.4 millimeters and an individual core length is 1 meter, this results in more than 2000 individual image files for that 1 meter core interval. In our dataset, the image slices are stored in a 16-bit unsigned DICOM format, a standard format developed for medical images [34]. The DICOM images of individual cores have been then stacked together and stored as 3D raw images using the ImageJ software [43].

2.3 Image preprocessing

To prepare the images as inputs for our CNN training process, we need to discard undesired noncore regions. The images coming from certain zones can negatively affect the classification results, since they contain information that is nonrelated to the actual phenomena we want to model.

The first step we adopted is to remove border effects by cropping the 3D raw image slices into rectangular crops of size 256 × 256 pixels. A comparison of an example image before and after this cropping is shown in Fig. 2. After cropping, a global minimum and maximum intensity value, selected by observing the 3D histograms of all rectangular crops, is assigned to the images of the entire considered core intervals. Further, the intensity adjusted images are encoded in 8-bit format, i.e., 0– 255 gray-scale, and stored for further analysis.

Another preprocessing operation includes removing images with missing data associated with poor core recovery, induced fractures, or rush plugs. Note that the image slices with missing core intervals show low gray-level attenuation values (Fig. 3a).

We also note that the images dataset contains a number of other undesired artefacts related to core barrel couplings, drilling mud invasion, and cementation of high-density minerals such as pyrite and siderite (examples are shown in Fig. 3b and c). Also these zones need to be excluded from the training set.

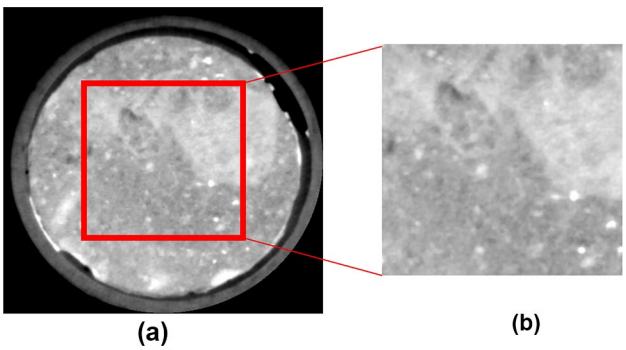

Fig. 2 Original DICOM image slices **a** are cropped (red square) into 256x256 squares **b** as a preprocessing step to prepare images to be used as inputs for the CNN training process

To flag and remove the above artefacts, we thus implemented dedicated type-dependent algorithms to the raw data. More precisely, to remove missing intervals we calculate the average attenuation μ_c in the center of the image using a centered square covering 40% of the total number of pixels. If the computed average attenuation is less than a predefined cutoff C_m, the image is flagged and removed:

$$f_m = \begin{cases} 1, \text{ if } \mu_c < C_m \\ 0, \text{ else} \end{cases} \quad (1)$$

where f_m is the flag for missing interval. The image is removed if f_m is equal to 1.

Intervals with high-density material appear very bright with relatively high gray-level attenuation readings. To identify these intervals, the average attenuation μ of the whole 2D image is computed and, if the average is greater than a predefined cutoff C_h, the image is flagged for removal:

$$f_h = \begin{cases} 1, \text{ if } \mu > C_h \\ 0, \text{ else} \end{cases} \quad (2)$$

where f_h is the high-density flag. The image is removed if f_h is equal to 1.

In the intervals with core barrel couplings, the attenuation values in the middle of the images are lower than the attenuation values of the image edges (i.e., the edges are brighter, as shown in Fig. 3c). To detect intervals with core barrel couplings, the difference in average attenuation of the center and edges of the 2D image is calculated. As above, the center average attenuation μ_c is computed considering 40% of the total number of pixels using a centered square. To represent edge average attenuation μ_e, the outer 5% of the total number of pixels along the edges are considered. If the difference between center average attenuation and edge average

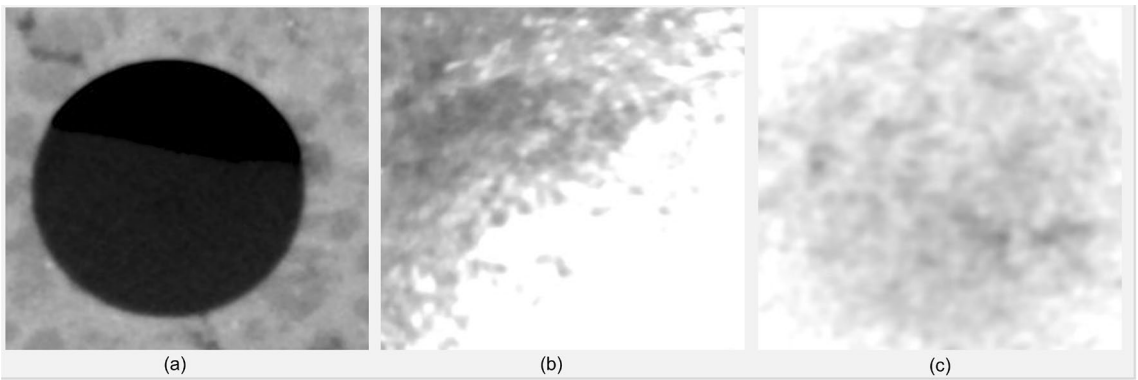

Fig. 3 2D image slices with: **a** missing CT values (due to rush plugs), **b** high-density material (cementation or drilling mud invasion), and **c** core barrel coupling

attenuation is greater than a predefined cutoff value C_b, the image interval is flagged and removed:

$$f_b = \begin{cases} 1, \text{ if } \mu_e - \mu_c > C_b \\ 0, \text{ else} \end{cases} \quad (3)$$

where f_b is the core barrel coupling flag. The image is removed if f_b is equal to 1. Note that the thresholds above have been computed using the global distribution of the minimum, mean and maximum intensity values observed in the dataset.

Finally, to reduce computational time associated with CNN training, the remaining 2D CT images are coarsened by a factor of four (i.e., the final image size is 64×64 pixels). Further, the images are rescaled, i.e., all pixel values are divided by 255, before being used as input for the CNN training.

2.4 Image augmentation

Generally, large amounts of training data are required to achieve a good performance in deep neural networks. Image augmentation is a strategy that is performed to boost the performance of the network through different kinds of modifications, e.g., random rotation, shifting, shearing and flipping, applied to the original images.

Image augmentation is applied during the training phase, so that the model can learn from more image examples. In our framework, we specifically considered rotation and horizontal flips of the original images. We thus implemented the "ImageDataGenerator" class in Python using the Keras API [11], a publicly available code that can be used for image augmentation purposes on the fly. The "ImageDataGenerator" class rotates the images randomly within a range of user-defined angles. Therefore, in case of squared images, it is very likely that for some specific rotation angles, the pixels will fall out of the image frame leaving some areas of the image with no pixels. There are a number of interpolation techniques such as nearest neighbor that can be used for those areas, but it can amend the key features resulting in dissimilar features counterproductive for training. To avoid this problem, the images were rotated outside Keras, while the horizontal flip was applied in Keras using "ImageDataGenerator" class on the fly during training the CNN classifier. The images were rotated by 90°, 180° and 270°. An example of the rotated and horizontally flipped images is shown in Figs. 4 and 5.

3 The dataset

3.1 Whole core CT scan images

This study uses whole core cross-sectional image slices from a well on the Norwegian continental shelf. The

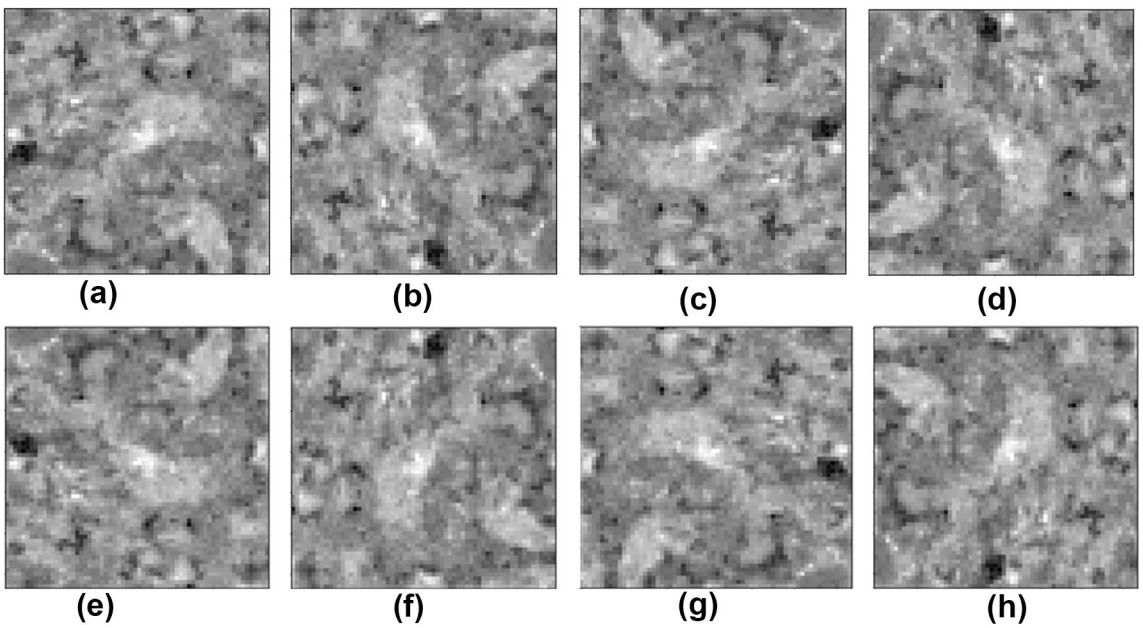

Fig. 4 An example of image augmentation applied on the CT images. **a** Original CT image, **b** original image rotated by 90 degrees, (**c**) original image rotated by 180 degrees, **d** original image rotated by 270 degrees, **e** original image horizontally flipped, **f** 90° rotated and horizontally flipped, **g** 180° rotated and horizontally flipped, **h** 270° rotated and horizontally flipped. Note that the images are coarsened by a factor of 4 with a final size of 64×64 pixels

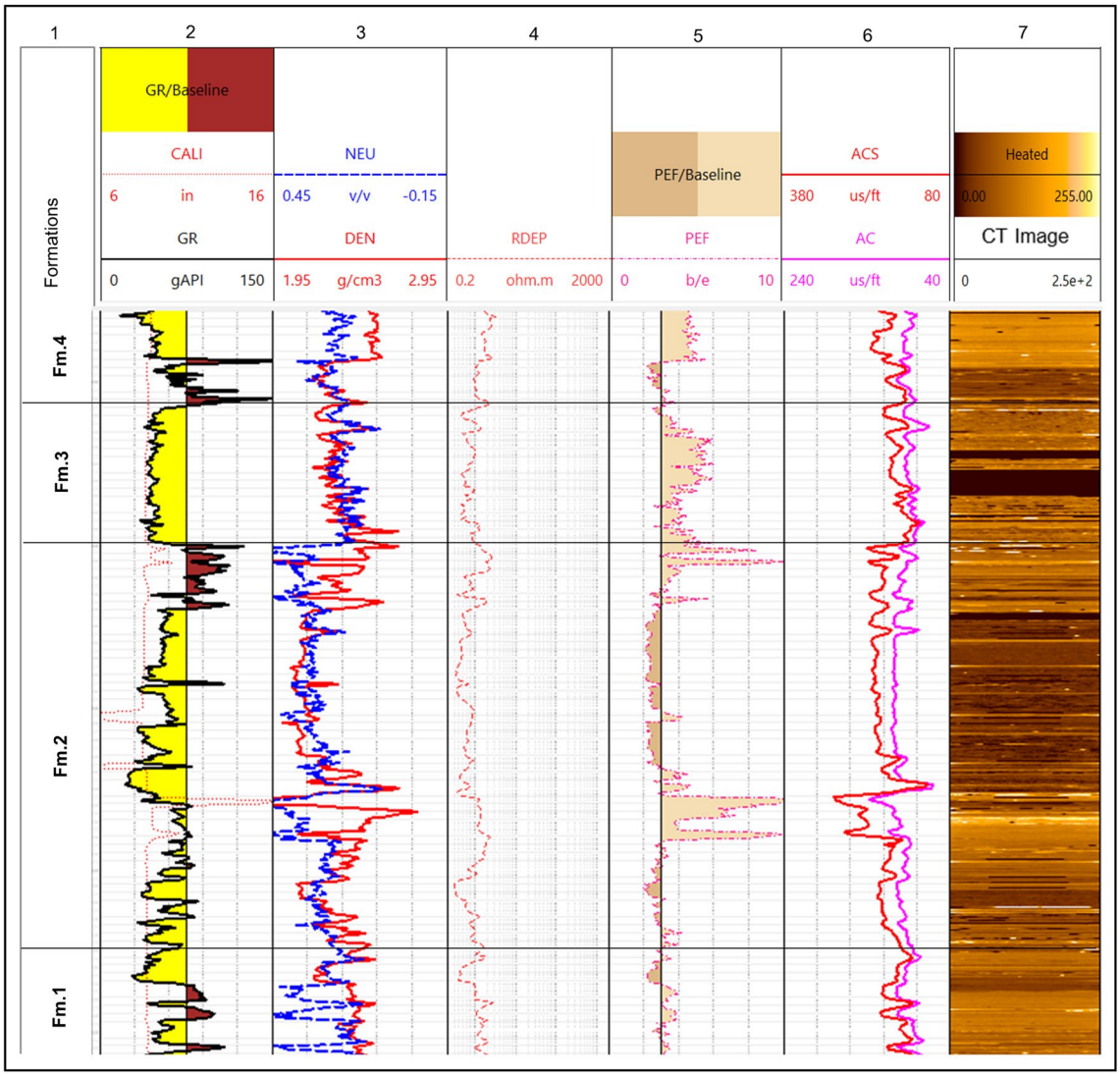

Fig. 5 Well log data and 2D cross section of the core CT image showing 142 meters of the studied well. Log tracks from left to right: track 1: Formations, track 2: Caliper (CALI) and Gamma ray (GR), track 3: Density (DEN) and Neutron (NEU), track 4: Deep resistivity (RDEP), track 5: Photoelectric factor (PEF), track 6: Compressional wave slowness (AC) and shear wave slowness (ACS), track 7: 2D cross section of whole core CT scan

studied well penetrates four main formations denoted as Fm.1, Fm.2, Fm.3 and Fm.4 in Fig. 5: Formation 1 consists of very fine-grained argillaceous sandstones and cemented sandstones, Formation 2 constitutes successive layers of mudstones and fine-grained sandstones, Formation 3 consists of granule-rich medium-grained sandstones and spiculites (a biogenic rock composed of sponge silica spicules), and Formation 4 comprises mud and calcite rich marlstones. As mentioned in Sect. 2.3, the images were provided in 16-bit unsigned DICOM slices with a vertical resolution of approximately 0.45 millimeters. The individual DICOM images were stacked and stored as 3D raw images and then cut into rectangular crops. In addition, a global minimum and maximum intensity value was assigned to all images before they were encoded in 8-bit format. The images with undesired artefacts were removed as described above, and the remaining images were coarsened by a factor of four to reduce computational time.

3.2 The lithofacies from the employed core description

We exploit information obtained from a manual core-based lithology description, performed by a geologist, as groundtruth to create the training lithofacies classes. The CNN classifier was then trained to learn the relationship between the image features extracted by the convolution process, and the corresponding lithofacies classes. For the

Table 1 Lithofacies classes and their associated fractions derived from core-based lithology descriptions (225524 images from 142 meter of core)

Lithofacies labels	Description	Fraction
Marl	Mud/clay rich marl	0.0214
CalMarl	Marl with caliche cementation	0.0157
SpiculiteSS	Medium-grained spiculitic sandstone	0.0438
Mudstone	Dark gray mudstone with plain parallel bedding, mottled mudstone	0.1181
WCemBelSS	Well-cemented medium-grained sandstone with Belemnite fossils	0.0035
GraMSSDispC	Granule-rich medium-grained sandstone with dispersed carbonate cementation	0.103
PCemGraMSS	Poorly cemented granule-rich medium-grained sandstone	0.032
WCemMSS	Well-cemented medium-grained sandstone	0.025
MudsHighDens	Mudstone with high density minerals (pyrite)	0.005
ArgFineSS	Argillaceous fine-grained sandstone	0.0726
RippleFineSS	Fine-grained sandstone with ripple cross-lamination	0.0809
MassFineSS	Massive fine-grained sandstone	0.099
CrossFineSS	Fine-grained sandstone with cross-stratified lamination	0.093
MudFineSS	Muddy fine-grained sandstone	0.0397
BioFineSS	Bioturbated fine-grained sandstone	0.0121
WCemFineSS	Well-cemented fine-grained sandstone	0.013
ContMud	Continental mudstone	0.0906
MassVeryFineSS	Massive very fine-grained green sandstone	0.0385
CemVeryFineSS	Cemented very fine-grained green sandstone	0.0667
VeryFineSSHorizontal	Very fine-grained sandstone with horizontal lamination	0.0264

sake of completeness, we report that our dataset presents 20 lithofacies classes derived through the manual core description mentioned above (the abbreviated classes together with a short description is found in Table 1). The three most abundant lithofacies are mudstone (marine and continental), granule-rich medium-grained sandstone with dispersed cementation, and fine-grained sandstone with different textures/laminations (ripple, cross-stratified and massive); these are interbedded with other sparser lithofacies.

4 Training phase

In this section, the training phase will be explained in detail. The section starts with the strategy used to separate train and test samples followed by training steps and hyperparameter optimization processes.

4.1 Division of the dataset in training vs. test data

A standard data analysis paradigm is to train a machine learning model on a set of data considered as the groundtruth and then evaluate its statistical performance on another set of unseen instances, again considered as correctly labeled in the manual labeling process. Considering the statistical distribution of the images in our dataset, we assessed that a suitable training vs. test sets splitting ratio is 80% for training and 20% for testing. To maintain continuous intervals and at the same time balancing the frequency of the lithofacies within each set, the train and test sets were selected manually. The reason for not selecting train and test sets randomly is that the images are slowly varying, so a random selection would give similar data points in both sets. Approximately 20% of the train set was employed as validation set, which is used to evaluate the performance of the model during training (see Sect. 4.2.3).

For completeness, the distribution of different lithofacies classes in the resulting train and test sets is presented in Fig. 6, from which we can see similar class distributions in both sets.

4.2 Details on the CNN training process

The CNN training is a process by which the kernels weights in the convolutional layers, the weights in the fully connected layers, and their associated biases are adjusted in such a way that the difference between the predicted labels and the given labels (i.e., the groundtruth) is minimized. Training is commonly performed by a forward- and back-propagation process throughout the entire network using a gradient descent optimization algorithm and a loss function. The loss function computes the difference between the output predictions, computed through forward propagation, and the actual label. The network

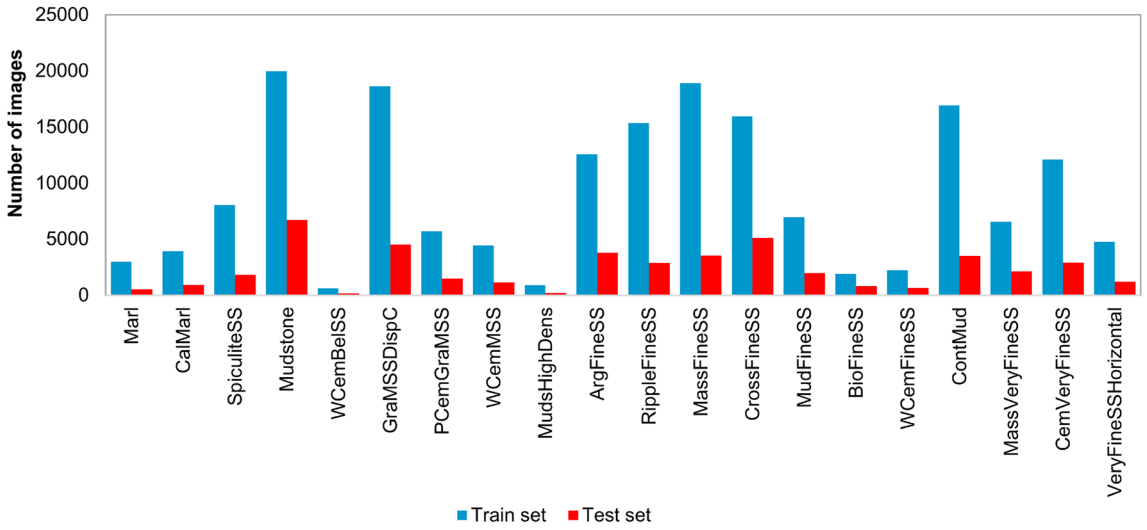

Fig. 6 Distribution of different lithofacies classes in the train (blue) and test (red) sets

performance is evaluated using the loss function. Cross-entropy is typically used as the loss function for multi-class classification tasks, whereas the mean squared error is typically used for prediction of continuous values, i.e., regression analysis [53]. In the current study, we are dealing with a multiclass classification task. Therefore, we used cross-entropy to determine the loss function of the CNN model as given by [9]:

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\left(p(\hat{y}_i)\right), \tag{4}$$

where y_i and $\hat{y}_i$ are, respectively, the true and predicted labels of the i^{th} sample, p is the probability, while N is the total number of training samples.

As mentioned above, the learnable parameters are updated iteratively using a gradient descent optimization algorithm that seeks to minimize cross-entropy losses. Basically, the partial derivative of the loss function with respect to each learnable parameter is first calculated; once the whole loss function gradient is computed, the learnable parameters are updated using [53]:

$$w^{+} = w - \alpha\frac{dL}{dw} \tag{5}$$

where w refers to each learnable parameter with w^{+} being the updated value, α stands for learning rate, and L is the loss function. The learning rate is an important hyperparameter that determines how fast the learnable parameter (e.g., weight) should move in the direction of the gradient. Note that finding the optimal learning rate during training is crucial for neural networks, since the training process may not converge when using a too high learning rate (in this case, indeed, the optimizer overshoots the minimum and lands in a zone of the parameters space that leads to worse loss values).

To avoid this issue, it is common to employ various types of optimizers so to search the optimum weight and kernel parameters using a pool of different gradient descents strategies, among which then choose the best one. Examples of the different types of descent methods are stochastic, batch and mini-batch gradient descents. These methods vary in terms of the number of samples used to compute the error between the actual and predicted labels.

In our study, we evaluated the performance of the RMSProp [20] and Adam [29] optimizers to optimize the weights. The obtained results revealed that Adam outperformed the RMSProp. Therefore, we eventually optimized the weights using the Adam optimizer together with a mini-batch gradient descent method. Note that this is the most common variation of gradient descent used in deep learning; to give some intuitions, mini-batch gradient descent splits the training data into small batches and calculates the error per batch before updating the learnable parameters.

In our study, the final optimal approach was to consider a batch size of 32 images and a CNN classifier training process of 70 epochs (where an epoch is a period in which all the training samples have been presented at least once to the network).

4.2.1 Hyperparameter selection

Generally, there exist two types of parameters in the machine learning algorithms. As mentioned in Section 4.2,

the kernel weights in the convolutional layers, the weights in the fully connected layers, and their associated biases are learnable, and thus optimized during the training process. The second type of parameters, referred to as hyperparameters, determine the structure of the cost function that is minimized, and need to be set by the user. These hyperparameters include the learning rate, the number of convolutional layers, the number of kernels in the convolutional layers, and the number of neurons in the fully connected layers. It is quite straightforward to realize that the performance of a machine learning model is highly dependent on the right choice of both the parameters and the hyperparameters. The process of adjusting the hyperparameters is called hyperparameter tuning.

As previously explained, the here proposed CNN classifier was developed in Keras using the Tensorflow backend. In our case we solve the hyperparameter tuning problem using the Keras tuner library [30, 37, 40]. This library enables to define a search space that includes the considered hyperparameters and an opportune tuner that will automate the solution of this tuning process. More precisely, the task of the tuner is to evaluate a certain number of hyperparameter combinations in a model that is explicitly set-up for hypertuning, i.e., a *hypermodel*. The considered hyperparameters in this study are presented in Table 2. Four tuners are available is Keras, including `RandomSearch`, `Hyperband`, `BayesianOptimization`, and `Sklearn`. For more information on the differences among these approaches, we direct the interested reader to [7, 23, 32, 47].

In this study, we utilize the Hyperband algorithm [32], a relatively new method for tuning the iterative algorithms. Basically, the strategy behind this approach is to try a large number of random configurations using adaptive resource allocation and an early stopping rule to quickly converge to a high-performance model. More specifically, the random configurations are run for a specific number of epochs (i.e., one or two) per configuration, and then the top-performing model configurations based on the previous results are trained for longer runs. Finally, the algorithm returns a best configuration trained to the assigned maximum number of epochs. The optimized classifier architecture, obtained by this hyperparameter selection processes, is presented in Fig. 7, and described in detail in the next section.

4.2.2 Classifier architecture

The proposed CNN classifier architecture is shown in Fig. 7. Its input and output layers consist of 2D image slices and lithofacies classes that have been derived from the available core descriptions. The classifier employs four distinct convolutional layers, indicated as "`Conv1`," "`Conv2`," "`Conv3`," and "`Conv4`," with 240, 48, 48 and 240

Table 2 Potential hyperparameters and the potential search space used in this work during the hyperparameter selection

Training hyperparameters	Parameter space
Number of convolutional layers	(1, 2, 3, **4**)
Number of convolutional kernels (filters)	(16, **48**, 80, 112, 144, 176, 208, **240**)
Kernel size	(3, **5**)
Learning rate	(0.01, 0.001, **0.0001**)
Number of neurons in the fully connected layer	(32, 64, 96, 128, 160, 192, 224, **256**)
Dropout rate	(0, **0**.2, 0.4, 0.6)

The final optimal values are shown in bold. Note that two numbers are bold for convolutional kernels since two convolutional layers have 48 kernels each, while the other two have 240 kernels each (Fig. 7)

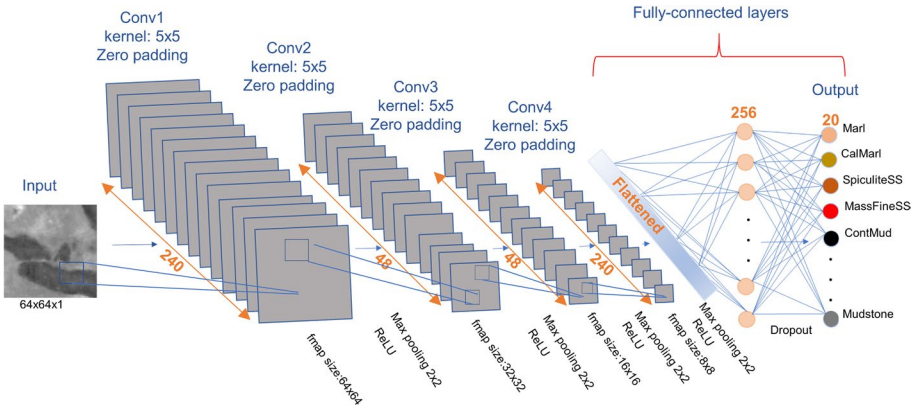

Fig. 7 Proposed CNN architecture for lithofacies classification

convolutional kernels, respectively. Note that here we employ a kernel size of 5×5; this specific dimension was indeed resulting as optimal from the hyperparameter tuning process, and has been used in all our convolutional layers.

In order to preserve the original image size, we moreover applied a zero padding technique in each convolutional layer; i.e., we added a layer of pixels with values of zero around the image edges. The convolution operation in each layer is in our scheme then performed using a stride of 1, and the resulted feature maps are passed through a ReLU activation function to introduce nonlinearity. In our context the stride is, basically, the number of pixel shifts when the kernels are moved throughout the input image. After applying the ReLU function, the feature maps are sent to the subsequent pooling layer, where they are downsampled using a max pooling layer with pooling window size of 2×2 and a stride of 1.

The pooled feature maps of the last convolutional layer are flattened into a one-dimensional vector that is connected to the output layer in the fully connected layer. The proposed network contains one hidden layer with 256 neurons. As mentioned before, the number of neurons in the hidden layer is a hyperparameter that was optimized during hyperparameter tuning. A ReLU function is also applied to the hidden layer followed by the dropout layer. Dropout is a regularization technique, where randomly selected neurons are discarded during training (i.e., they are temporarily removed from the network together with their incoming and outgoing connections). The dropped-out neurons are not employed in the backpropagation phase [21, 48]. A dropout rate of 0.2 was applied in the proposed network meaning that one in 5 of the neurons in the hidden layer will be randomly ignored from each update iteration. As mentioned in Table 2, the dropout rate is a hyperparameter that, as the others, is optimized during the hyperparameter tuning phase. This regularization scheme is meant to prevent overfitting, and can be interpreted as an attempt to optimize the bias-variance tradeoff of the overall estimator. For more details about the statistical interpretations of regularization see [44].

Another common regularization technique in deep learning is *batch normalization*. In batch normalization, the output of a convolutional layer is normalized before being used in the next one. This technique is known to have also a regularization effect, and it is empirically known to typically speedup the network training, plus make it less sensitive to the initialization point [26]. We note that, however, this is not guaranteed in general settings—and indeed, in the current study, more accurate results were obtained without using batch normalization. The last layer in Fig. 7 is the output layer with 20 nodes corresponding to the 20 lithofacies labels. The proposed architecture provides 1'628'612 trainable parameters.

4.2.3 Classifier evaluation

As mentioned previously, 20% of the training images were utilized as the validation set. The cross-entropy loss and accuracy were considered as training metrics to evaluate the performance of the CNN classifier during training. Figure 8 shows how the accuracy and cross-entropy change over time during the training process. As one can see from the plots, the classification accuracy increases with increasing number of epochs in both the training and validation sets. However, the cross-entropy loss decreases with increasing number of epochs. The training metrics start to converge at around 70 epochs.

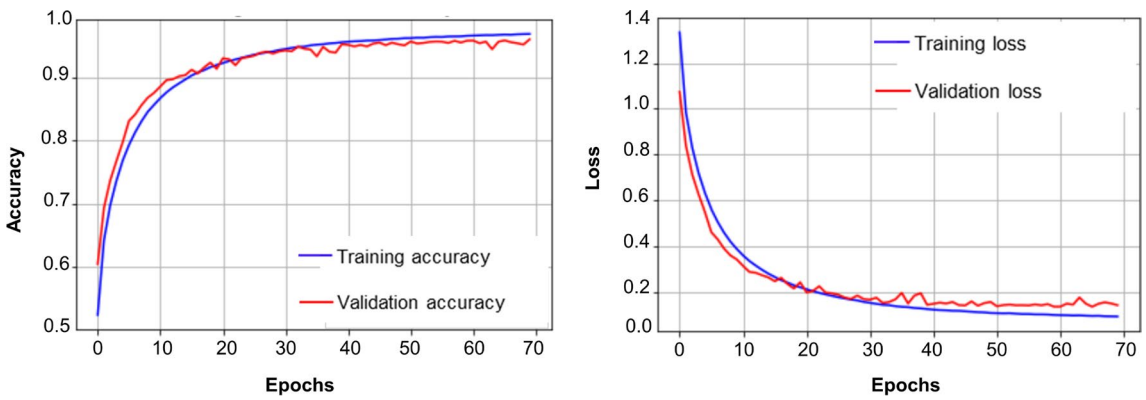

Fig. 8 Model performance on the training and validation set. The plot to the left shows the accuracy results by increasing the number of epochs, whereas the plot to the right shows loss results by increasing the number of epochs

5 Results

The lithofacies classification results acquired by using the trained classifier on a set of unseen images will be presented and discussed in the following sections.

5.1 Lithofacies prediction

To evaluate the performance of the trained CNN classifier on unseen data, the model was used to predict lithofacies in another part of the well, previously denoted as the test set. For consistency, the test images are passed through the same processes of image preprocessing and rescaling before being actually classified. The corresponding prediction accuracy metrics and confusion matrix calculated by cross-classifying the lithofacies classes from core description (classification groundtruth) and CNN prediction are summarized in Table 3 and Fig. 9. Here, accuracy is defined as the sum of true positives divided by total number of samples in the test set (i.e., probability of correct classification). Precision is quantified as the sum of true positives divided by the sum of true positives and false positives across all the lithofacies classes in the test set. In other words, precision represents the probability that the predicted lithofacies class, given the classification results for individual images, actually belongs to that class. Recall is calculated as the sum of the true positives divided by the sum of true positives and false negatives across all the lithofacies classes. Precision and recall results are combined into a single measurement, i.e., the f1-score, through the following formula:

$$\text{f1-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} . \quad (6)$$

The confusion matrix provides information on the similarity of the lithofacies classes in the confusion space. If we consider each row in the confusion matrix as a vector representing a particular class, the i^{th} coordinate in that row vector shows the degree of misclassification of the considered class with the i^{th} class [16]. In other words, the diagonal values of the confusion matrix represent the recall, while the off-diagonal values correspond to the degree of misclassifications. All the row vectors in the computed confusion matrix are normalized to one. Therefore, in the case of 100% accuracy, the i^{th} coordinate of the i^{th} row vector will be 1, while all the off-diagonal coordinates will be 0.

Looking at the confusion matrix in Fig. 9, we observe that the proposed classifier is able to predict some of the lithofacies classes with recall values above 0.7. More specifically, granule-rich medium-grained sandstones with dispersed calcite cementation record the highest recall (0.92), followed by very fine-grained sandstones with horizontal lamination (0.75), massive fine-grained sandstones (0.72) and poorly cemented granule-rich medium-grained sandstones (0.71). However, the classifier misclassifies the other lithofacies into another class or a set of classes with different degrees of confusion.

In particular, the classifier misclassifies very fine-grained lithofacies classes, i.e., marl, marl with caliche cementation, mudstone, mudstone with high density minerals, muddy fine-grained sandstone, cemented very fine-grained green sandstone, massive very fine-grained green sandstone, and continental mudstone. Examples of these misclassified lithofacies classes are illustrated in Fig. 10, from which we can see that these lithofacies classes actually show similar texture and grain sizes, therefore similar gray-scale values, with no distinct features. This explains the difficulties that the classifier encounters in doing its designed task. As lithofacies with similar grayscale and textural properties are expected to exhibit similar transport properties, porosity and permeability data from core analysis measurements were used to investigate the transport properties of the classified lithofacies. Figure 11 shows the porosity–permeability cross-plot for core plug samples from the same core data as in our CT images, where different colors correspond to the different lithofacies that have

Table 3 Prediction accuracy metrics on the test set using the trained CNN classifier. Support shows the number of predicted samples for each class

Lithofacies labels	Precision	Recall	F1-score	Support
Marl	0.23	0.39	0.29	542
CalMarl	0.27	0.52	0.36	918
SpiculiteSS	0.50	0.64	0.56	1835
Mudstone	0.53	0.61	0.56	6684
WCemBelSS	0.19	0.16	0.17	160
GraMSSDispC	0.83	0.92	0.87	4498
PCemGraMSS	0.84	0.71	0.77	1491
WCemMSS	0.82	0.65	0.72	1161
MudsHighDens	0.26	0.50	0.34	187
ArgFineSS	0.48	0.52	0.50	3774
RippleFineSS	0.36	0.53	0.43	2879
MassFineSS	0.86	0.72	0.78	3522
CrossFineSS	0.68	0.44	0.53	5096
MudFineSS	0.30	0.28	0.29	1979
BioFineSS	0.79	0.31	0.44	824
WCemFineSS	0.72	0.64	0.68	653
ContMud	0.44	0.33	0.38	3489
MassVeryFineSS	0.36	0.28	0.32	2121
CemVeryFineSS	0.54	0.52	0.53	2906
VeryFineSSHorizontal	0.86	0.75	0.80	1192

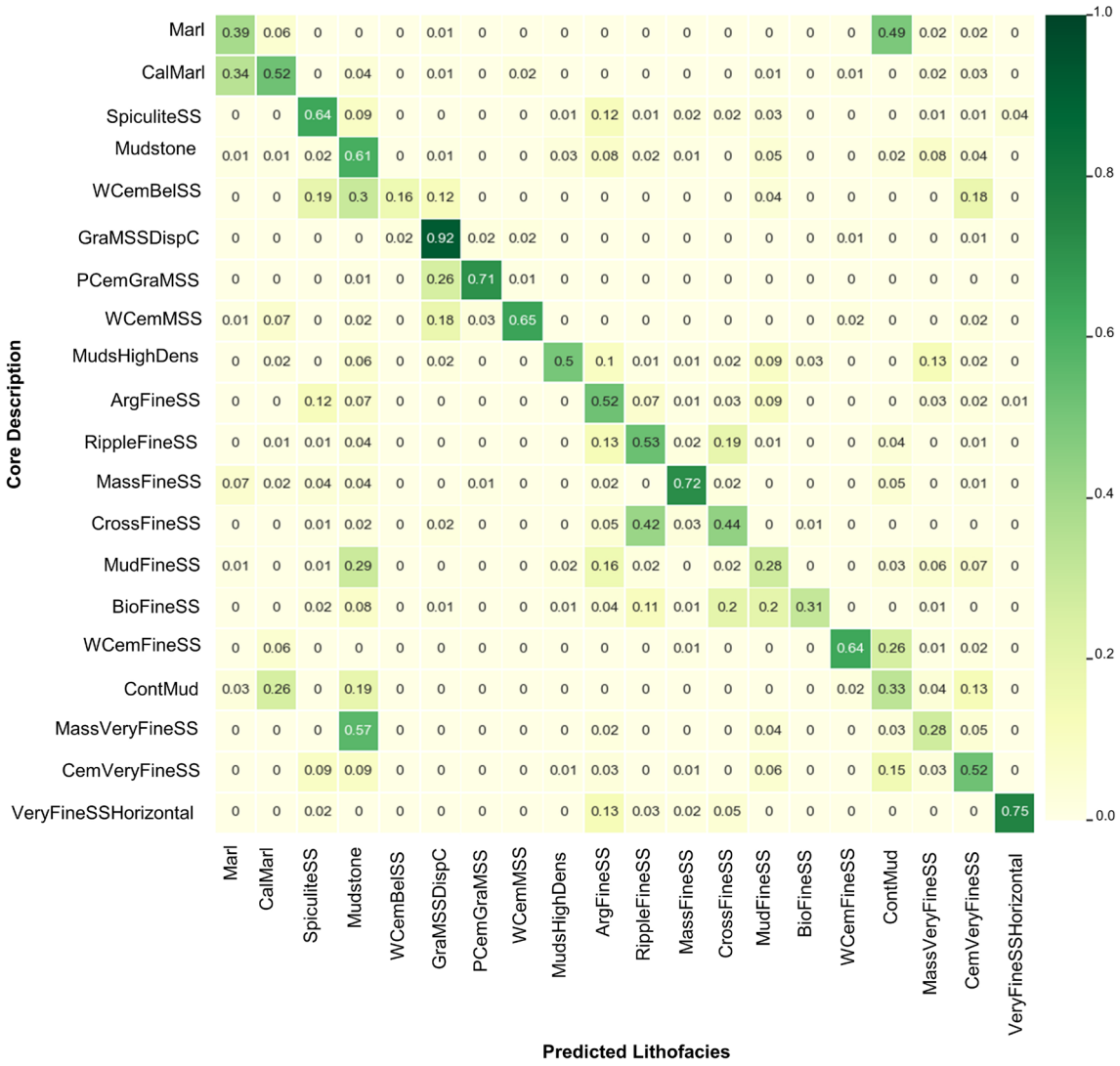

Fig. 9 Confusion matrix for the test set prediction

been derived from the manual core description. Here, we can see that the aforementioned misclassified lithofacies fall into the same region with porosity and permeability values less than 0.20 and 10 mD, respectively (marked by red ellipsoid in Fig. 11).

Likewise, fine-grained sandstones with horizontal lamination, ripple cross-lamination and cross-stratified lamination are not classified with satisfying performance. The porosity–permeability cross-plot shows that these lithofacies, together with massive fine-grained sandstones, fall in the same region in the porosity–permeability cross-plot. More specifically, they exhibit porosity values above 0.28 and permeability values ranging from 100 mD to approximately 30 Darcy (represented by the blue ellipsoid).

Granule-rich medium-grained sandstone samples (P-CemGraMSS, GraMSSDispC) spread out in the regions with permeability values ranging from 30 mD up to 50 Darcy (the green ellipsoid). However, most of the samples belonging to these classes exhibit porosity and permeability values above 0.20 and 1 Darcy, respectively. The prediction results shown in Fig. 9 indicate that poorly cemented granule-rich sandstone (PCemGraMSS) lithofacies are mainly misclassified as granule-rich sandstone with dispersed calcite cementation. The spiculite sandstone samples exhibit porosity values ranging from 0.20 to 0.28 and permeability values from 1 mD to 20 mD. The spiculite lithofacies is mostly misclassified as argillaceous fine-grained sandstone, showing similar porosity values. However, some of the measurements belonging to the argillaceous fine-grained lithofacies class exhibit higher porosity and permeability values, similar to the other fine-grained sandstones, and fall in the blue ellipsoid in Fig. 11.

Here, we see that even though most of the core measurements can be separated by the identified clusters in

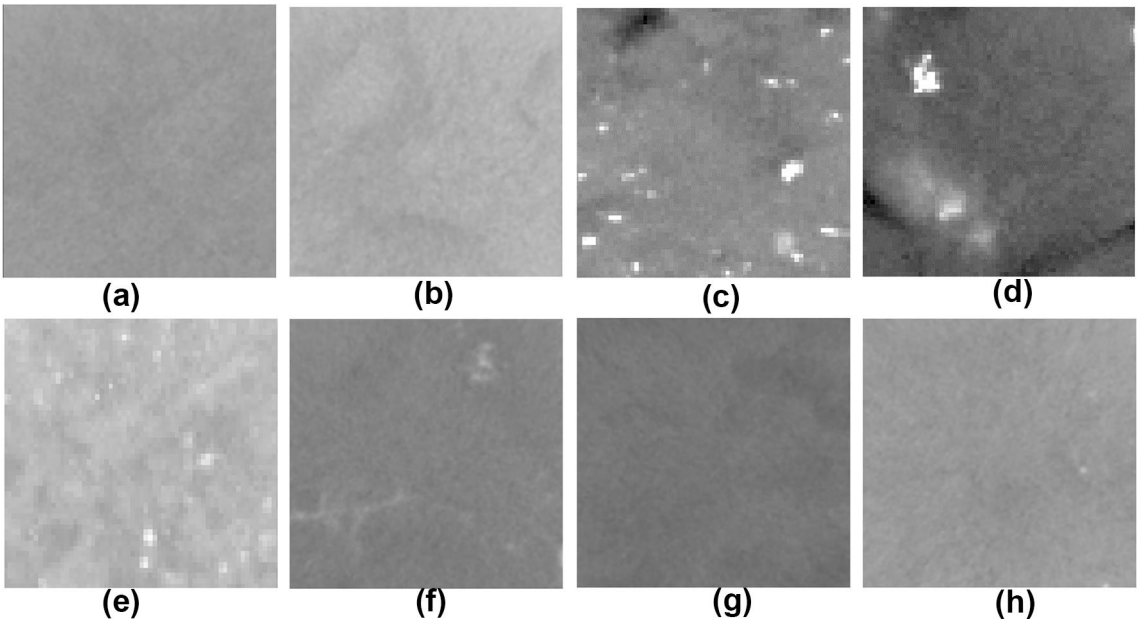

Fig. 10 Examples of very fine-grained lithofacies classes with similar textures and grain sizes with no distinct features. This type of images confuse the trained classifier and result in misclassifications and model deficiencies. **a** Marl, **b** Marl with caliche cementation, **c** Mudstone, **d** Mudstone with high-density minerals, **e** Cemented very fine-grained sandstone, **f** Massive very fine-grained green sandstone, **g** Muddy fine-grained sandstone, **h** Continental mudstone. The size of images is 64 × 64 pixels

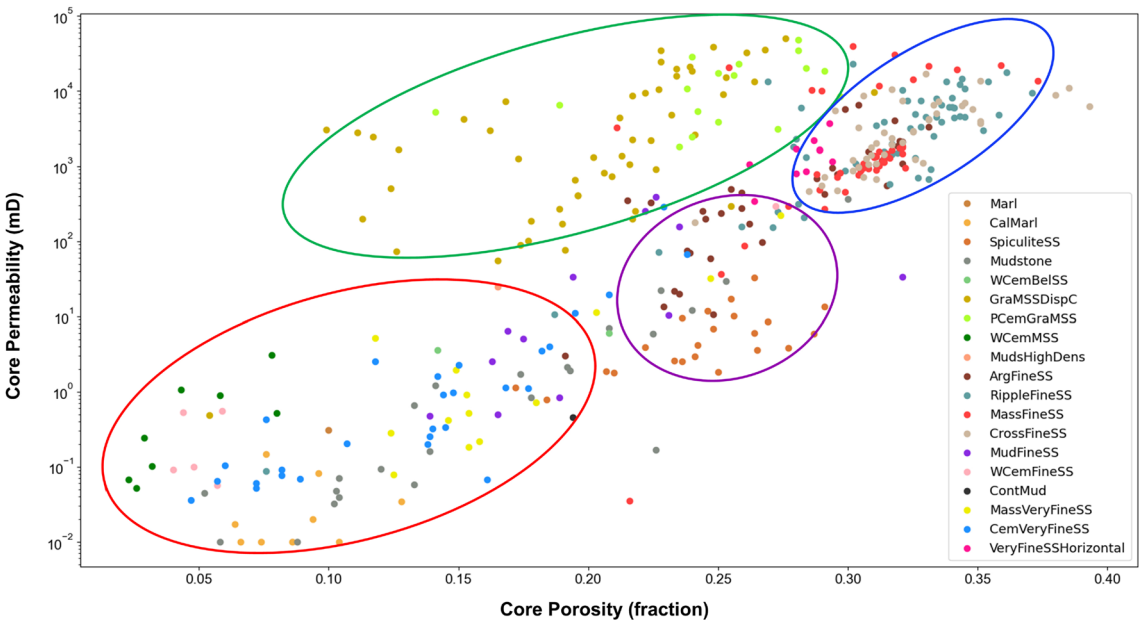

Fig. 11 Porosity–permeability cross-plot from available core measurements for the studied well. The lithofacies derived from core description are shown in different colors. The misclassified lithofacies exhibit similar porosity–permeability relationship marked by ellipsoids with different colors

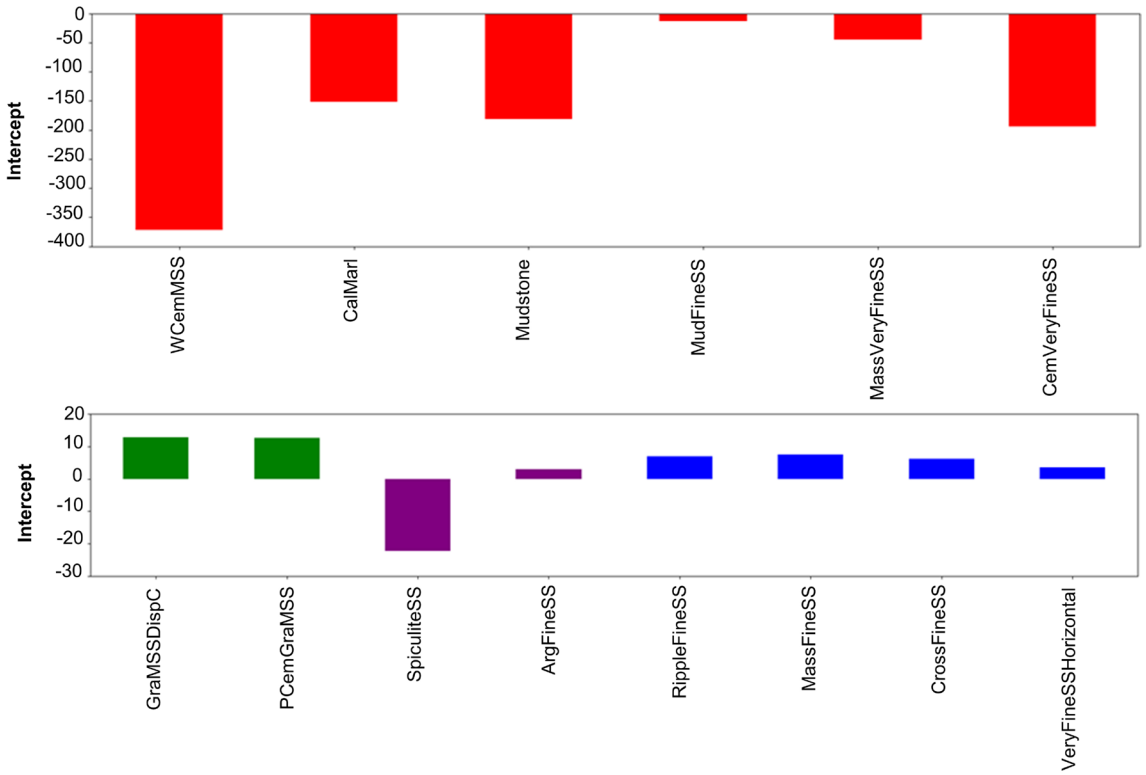

Fig. 12 Intercept values computed from the porosity–permeability cross-plot for misclassified lithofacies. Similar lithofacies, misclassified by the classifier, are presented with similar colors corresponding to the lithofacies clusters identified in Fig. 11

Fig. 11, some of the measurements exhibit a wide range of porosity and permeability values falling into more than one cluster, e.g., mudstone, cemented very fine-grained sandstone and argillaceous fine-grained sandstone samples.

In order to investigate the acquired classification results more quantitatively, we fit a log-linear regression line to map the porosity–permeability relationships of different lithofacies with more than five measurements. The computed intercepts can be used as an indication of similarity in transport properties between different lithofacies. The resulted intercept values are presented in Fig. 12; from this we can clearly infer that most of the lithofacies with similar transport properties tend to group into similar sets of intercept values. However, argillaceous fine-grained sandstone samples exhibit similar intercept values to the samples in the blue ellipsoid, which is expected due to the presence of argillaceous samples with higher range of porosity and permeability falling into the blue ellipsoid in Fig. 11. The colors in Fig. 12 correspond to the lithofacies clusters identified by ellipsoids in Fig. 11. Considering the similarities in transport properties of the misclassified lithofacies, it is not unreasonable to expect classification confusion amongst these classes.

In addition to the aforementioned similarities in texture and grayscale values, there are other issues that can create uncertainties and affect the training process and generalization capability of the trained classifier. One issue is related to the dipping and interchanging lithofacies. As an example, fine-grained argillaceous sandstones, ripple cross-laminated and cross-stratified sandstones interchange within the studied intervals creating difficulties in assigning a clear boundary during core description. Moreover, in the intervals with dipping lithofacies it is not easy to define a horizontal bed boundary. Another important point is related to the groundtruth labels derived from manual core description. These labels are assigned by visual inspection of the whole cores (or core photos), and they do not have pixel-wise resolution creating inconsistencies during training phase.

Figure 13 shows a section of the predicted test set together with the 2D whole core CT image and expert-derived core description. The classifier is able to predict the granule-rich (PCemGraMSS and GraMSSDispC) and well-cemented medium-grained sandstone lithofacies with fair accuracy. However, mudstone and fine-grained sandstone lithofacies (ripple cross-laminated and cross-stratified) are confused with other similar lithofacies.

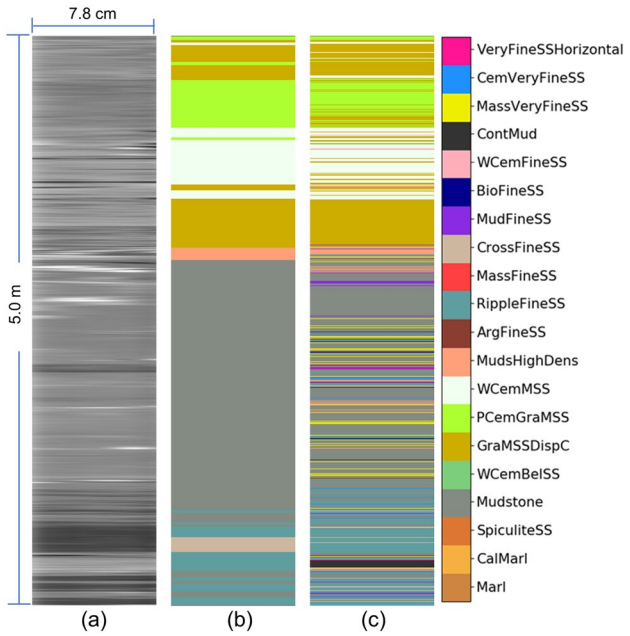

Fig. 13 Lithofacies prediction results from a section of the test set. **a** 2D whole core CT image, **b** Lithofacies classes from manual core description, **c** Lithofacies prediction using the trained CNN classifier

6 Post-classification processing

In the previous section, we mentioned that the confusion matrix can provide invaluable information about the similarities and relationships between different lithofacies classes, and then we showed that the confused lithofacies classes exhibit similar porosity–permeability trends. In fact, in Fig. 11 we see that the misclassified lithofacies group into four different rock classes based on their porosity and permeability relationships. This guides us to consider if the lithofacies classification task can be coarsened with respect to the number of lithofacies classes. For this end inspired by Godbole [16], we use the information acquired from the confusion matrix to generate lithofacies hierarchies based on the degree of confusion for the different lithofacies classes.

6.1 Automatic generation of lithofacies hierarchies

Hierarchical clustering is a method in clustering analysis that aims at building a hierarchy of clusters based on a predefined similarity metric. Generally, two approaches are considered in performing hierarchical clustering analysis, i.e., *agglomerative* and *divisive clustering* [27]. Agglomerative clustering, also called the "bottom-up" approach, starts with each element in a singleton cluster and pairs of clusters being merged successively until a specific stopping criterion is satisfied. The divisive, also called the "top-down" approach, starts instead with all the elements in a single cluster; splitting is then performed recursively by moving down in the hierarchy. In hierarchical clustering similar clusters are grouped successively using a similarity metric, which is often a distance measure defined on the feature space [27]. The most common similarity metrics are Euclidean distance, Mahalanobis distance and Kullback-Leibler distance measure. There are different methods to measure distance between clusters; among these, the single-linkage [46], the complete-linkage [28], and the minimum variance (Ward) [51] methods are the most popular ones. More specifically, the single linkage (or nearest neighbor) clustering method looks for pairs of elements from two clusters that have minimum distance. In other words this approach basically considers recursively the closest pairs of elements from two clusters to measure the distance. In the complete linkage method, instead, the distance between two clusters is computed as the distance between the farthest elements of the two clusters. In both cases, the clusters with minimum distance measure are merged to form a larger cluster. The single-link algorithm is simple to implement, but it is known to suffer from chaining effects [35] that produce elongated clusters and long chains. By contrast, the complete link algorithm forces consistent diameter and spherical clusters. The Ward's clustering method is then a special case of an objective function approach that looks for aggregate deviations of the elements. In fact, this method pretends to merge two clusters, and then estimates a centroid for the resulting cluster and calculates the sum of the squared deviations of all the elements from the new centroid. This algorithm then picks the merge with minimum within cluster variance or the merge with smallest deviation from the new centroid. The output of the hierarchical clustering is presented in a dendrogram representing the nested clustering of the elements and their similarity levels.

In this study, we perform hierarchical clustering using the empirical confusion matrix from the classifier as the

Table 4 Confusion matrix of four classes. Here, we consider four classes for simplicity

	GraMSS-DispC	PCem-GraMSS	RippleFi-neSS	CrossFineSS
GraMSS-DispC	0.92	0.02	0	0
PCem-GraMSS	0.26	0.71	0	0
RippleFi-neSS	0	0	0.53	0.19
CrossFineSS	0.02	0	0.42	0.44

Table 5 Similarity matrix computed using the confusion matrix in Table 4

	GraMSS-DispC	PCem-GraMSS	RippleFineSS	CrossFineSS
GraMSS-DispC	0	1.35	1.66	1.78
PCem-GraMSS	–	0	1.69	1.81
RippleFineSS	–	–	0	0.38
CrossFineSS	–	–	–	0

quantitative measure of distance between the various lithofacies. This corresponds to use an Euclidean distance as the inter-class similarity metric between lithofacies class vectors in the confusion space. More precisely, the Euclidean distance is, in our work, calculated by summing up the absolute differences in the coordinate values of two class vectors. To exemplify the process, consider the confusion matrix in Table 4, where for simplicity we show only the results relative to four classes. Each class is represented by a vector in the confusion space, i.e., $\overrightarrow{GraMSSDispC} = \{0.92, 0.02, 0, 0\}$ represents GraMSSDispC lithofacies class. The Euclidean distances mentioned above are then calculated by summing up the absolute differences in the coordinate values of the class pairs. In this way it is possible to compute an upper triangular similarity matrix as the one shown in Table 5. This, in particular, clearly shows that RippleFineSS and CrossFineSS classes are the most similar ones among the set of classes considered in this sub-confusion matrix used to exemplify the process.

The computations and considerations in the example above are then performed and observed in the original complete confusion matrix; the resulting similarity matrix is then used as the input for the hierarchical agglomerative clustering step.

The dendrogram resulting from this clustering step is presented in Fig. 14 and shows the overall result of clustering similar lithofacies classes together. It is worth mentioning that various clustering methods result in different dendrogram structures. In this work, we started by performing hierarchical agglomerative clustering using all the three methods mentioned above, i.e., single-linkage, complete-linkage and Ward's method; we then observed that, among these approaches, the Ward's method returned the clustering structure that is the most coherent in terms of keeping lithofacies with similar transport properties together.

We also note that the vertical axis in a dendrogram is used as a reference distance that shows the similarity of the lithofacies classes. This means that the plot shows not only how different the classes are, but also the order by which lithofacies clustering occurs. We note that the obtained dendrogram clearly reflects the semantic similarity of the lithofacies classes in the confusion space. Indeed, for example, the plot shows that mudstone and massive very fine-grained green sandstone (MassVeryFineSS) classes are grouped before any other lithofacies classes; this is in line with the fact that these facies are, from a lithological perspective, the most similar ones within the set of classes we considered. The second most similar lithofacies classes are ripple cross-laminated (RippleFineSS) and cross-stratified fine-grained sandstone (CrossFineSS). As it should be, they form in the obtained dendrogram the second cluster in the hierarchy. The third cluster instead forms by merging the muddy fine-grained sandstones and well-cemented sandstones with Belemnite fossils. Then, this newly formed cluster is merged with the first cluster at a higher level of similarity distance. Moreover, argillaceous fine-grained sandstone (ArgFineSS) class clusters with spiculite sandstone. These lithofacies classes show a high degree of confusion with each other in the confusion space, as confirmed by Fig. 9.

As we explore the dendrogram upward, the similarity of lithofacies classes that are clustering together decreases. We indeed can note that the hierarchical clustering derived from similarity of lithofacies classes in the

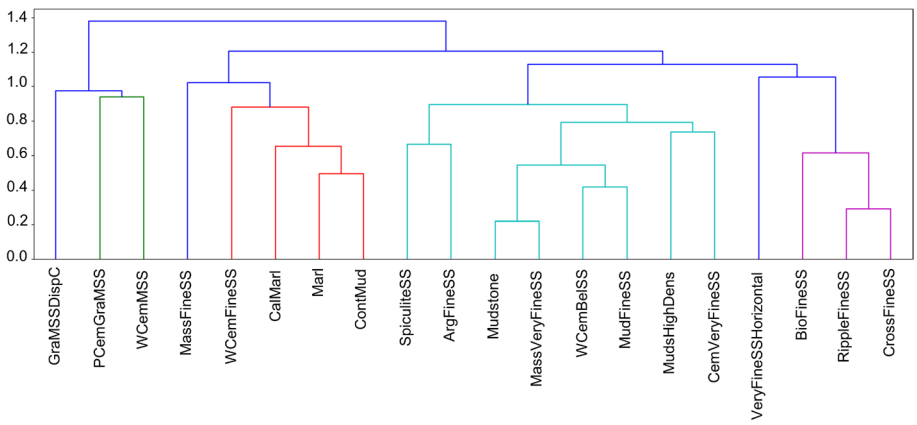

Fig. 14 Dendrogram of the process of clustering the lithofacies classes together using as a distance metric the confusion matrix that has been calculated by the proposed CNN classification algorithm

confusion space mostly results in grouping of lithofacies with similar grain sizes, textures and transport properties. However, as an example, we notice that the well-cemented medium-grained sandstone (WCemMSS) class is first merged with the poorly cemented granule-rich sandstone class (PCemGraMSS), and at a slightly higher level they merge with granule-rich sandstone with dispersed cementation (GraMSSDispC). Recall then that it was previously shown that the granule-rich lithofacies core measurements spread out in the regions with high permeability values ranging from 30 mD up to 50 Darcy (i.e., the green ellipsoid in Fig. 11), where the majority of samples exhibit porosity and permeability values above 0.20 and 1 Darcy, respectively. On the other hand, the well-cemented sandstone samples (WCemMSS) are characterized by porosity and permeability values less than 0.10 and 5 mD, respectively. Therefore, merging these classes, with completely different transport properties, does not seem reasonable.

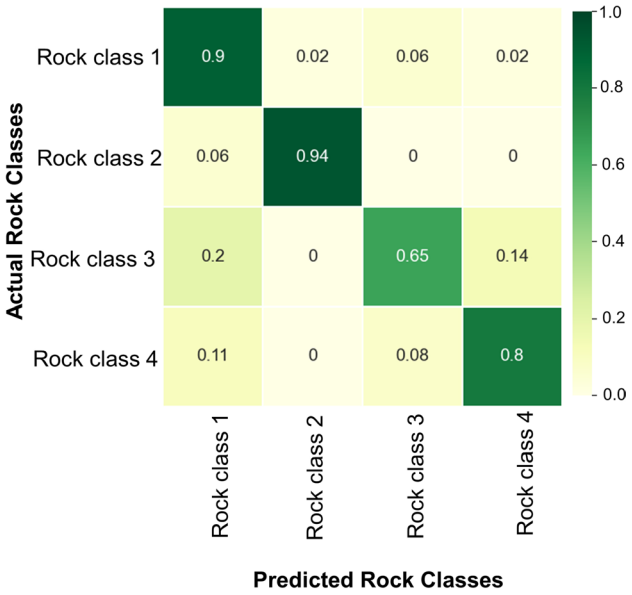

Fig. 15 Confusion matrix on the test set using the coarsened classifier, where the original 20 lithofacies classes are merged into four rock classes

6.2 Lithofacies prediction using the Coarsened CNN classifier

As mentioned in the previous subsection, the current lithofacies classification task can be coarsened with respect to the number of classes by merging similar misclassified lithofacies classes. More specifically, based on the porosity–permeability relationships and hierarchical clustering results, we propose grouping the lithofacies classes into four rock classes, as presented in Table 6.

Following this classification, the groundtruth labels derived from manual core description can be modified so to reflect the four superclasses above instead of the original 20 ones. This implies that one can retrain the original CNN classifier proposed above using this new set of labels and also perform a new round of testing. The resulting confusion matrix is shown in Fig. 15, from which we see that the classifier is able to predict rock classes 1, 2, and 4 with high recall values.

However, rock class 3 is still predicted with a relatively low recall (0.65), and it is mostly confused with rock classes 1 and 4. To inspect why, consider the corresponding rock classes, shown in Fig. 16. The plot shows that the coarsened classifier generalizes well and predicts individual rock classes with high accuracy. The classifier even shows higher pixel-wise precision in detecting thin layers and bed boundaries to the point that it is able to detect thin layers that are not picked by the manual core description. As an example, in Fig. 16, the 2D CT image cross section shows a clear change in the gray scale values in the section marked by the green rectangle in Fig. 16A. Here, we see that the more porous and permeable layer (characterized by darker grayscale values) is underlain by a tighter layer marked by the red rectangle. The tight layer is characterized by brighter gray scale values compared to the layers above and below, but this was not picked during manual core description. At the same time, this layer is accurately detected by the CNN classifier. More investigation of this

Table 6 Proposed rock classes resulted from merging similar lithofacies classes

Rock classes	Clustered lithofacies	Description
Rock class 1	Marl, CalMarl, ContMud, WCemFineSS, Mudstone, MudsHighDens, MudFineSS, WCemBelSS, WCemMSS, CemVeryFineSS, MassVeryFineSS	Very fine- to medium-grained sandstones, well-cemented very fine- to medium-grained sandstones, marl and mudstones
Rock class 2	GraMSSDispC, PCemGraMSS	Medium-grained granule-rich sandstones, poorly cemented / with dispersed calcite cementation
Rock class 3	SpiculiteSS, ArgFineSS	Fine-grained spiculite sandstones and fine-grained argillaceous sandstones
Rock class 4	RippleFineSS, CrossFineSS, BioFineSS, MassFineSS, VeryFineSSHorizontal	Fine-grained sandstones with different types of laminations

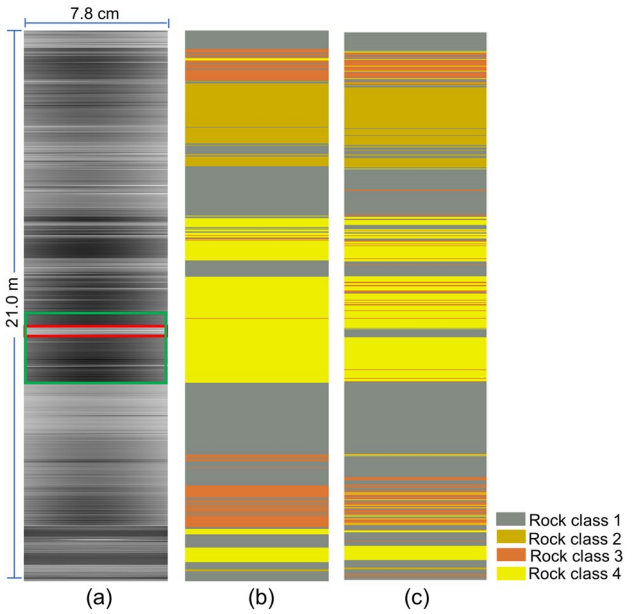

Fig. 16 Predicted rock classes on the test section of the well (approximately 21 meters) (**c**), shown with actual rock classes (**b**) and the 2D cross section of the input CT images (**a**). The scaled-up classifier is predicting the rock classes with high accuracy

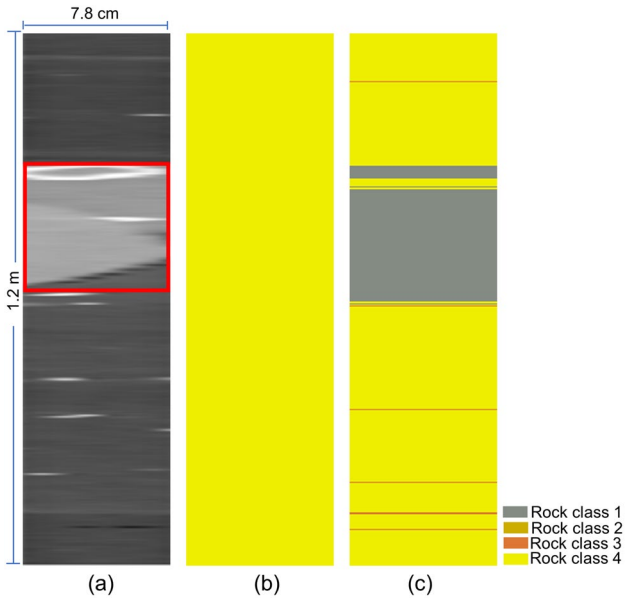

Fig. 17 Zoomed interval of the test set (approximately 1.2 meters), where the CNN classifier is able to pick the calcite nodule. **a** 2D cross section of the input CT images, **b** Rock classes from manual core description, **c** Predicted rock classes using the scaled-up classifier

interval reveals that the tight layer is actually a big calcite nodule encapsulated within the massive fine-grained sandstone lithofacies (Fig. 17). This calcite nodule is classified as rock class 1 that contains lithofacies classes with high amount of calcite cementation, most probably due to similar grayscale values.

7 Conclusions

In this study, the capability of CNN to classify lithology, based on the 2D whole core CT image slices, was investigated, and its performance was characterized in detail.

A CNN classifier was trained to learn features associated with 20 various lithofacies classes derived from manual core descriptions. The trained classifier was then used to predict lithofacies on the unseen test set images.

The preliminary results revealed that the trained classifier showed lithofacies-dependent performance and it misclassified, to various degrees, specific lithofacies classes with similar grain size, gray-scale values, and transport properties.

The obtained prediction confusion matrix was then utilized as a valuable tool to understand the performance limits of the CNN classifier and to combine the similar lithofacies into rock classes using an automatic hierarchical clustering approach.

Applying the CNN classifier on these clustered classes shows that the new approach generalizes well and predicts the rock classes with high recall values. Moreover, it shows higher pixel-wise precision, in detecting thin layers, compared to expert-derived core description, thereby providing higher resolution information than the one extracted during the manual labeling process.

The proposed classifier is trained based on data from a single well with imbalanced distribution of lithofacies classes. This might result in lower prediction performance on the classes with lower proportions. Adding more training images for those classes, preferable from other wells with similar lithology, might have a positive impact on the performance of the classifier.

As expected, uncertainties associated with manual core description, interchanging and dipping lithofacies can also affect the training process and generalization capability of the trained classifier.

It is worth to mention that the network architecture might affect the results, but it is not expected to change the conclusions in this study. For comparison purposes, the VGG16 architecture [45] was tested out and its

performance was compared with the proposed architecture. However, this change of the CNN architecture had minor impact on the acquired results.

Acknowledgements This research is a part of BRU21 – NTNU Research and Innovation Program on Digital and Automation Solutions for the Oil and Gas Industry (www.ntnu.edu/bru21) supported by Equinor in Norway. Carl Fredrik Berg acknowledges support from the Research Council of Norway through its Centre of Excellence funding scheme with Project No. 262644. We would like to thank Equinor for providing the data and the permission for publishing this work.

Funding This study was supported by Equinor.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Availability of data and material Due to confidentiality agreements, the data supporting the findings of this study cannot be made available.

Disclaimer The views and opinions expressed in this paper are those of the authors and do not necessarily reflect those of the Joint Industry Research Program members or example data owners.

References

1. Al-Anazi A, Gates I (2010a) On the capability of support vector machines to classify lithology from well logs. Nat Resour Res 19(2):125–139
2. Al-Anazi A, Gates I (2010b) A support vector machine algorithm to classify lithofacies and model permeability in heterogeneous reservoirs. Eng Geol 114(3–4):267–277
3. Al-Obaidi M, Heidari Z, Casey B, Williams R, Spath J et al (2018) Automatic well-log-based fabric-oriented rock classification for optimizing landing spots and completion intervals in the midland basin. Society of Petrophysicists and Well-Log Analysts
4. Anjos CE, Avila MR, Vasconcelos AG, Neta AMP, Medeiros LC, Evsukoff AG, Surmas R, Landau L (2021) Deep learning for lithological classification of carbonate rock micro-ct images. Comput Geosci 25(3):1–13
5. Ball K, Arbus T, Odi U, Sneed J (2017) The rise of the machines, analytics, and the digital oilfield: Artificial intelligence in the age of machine learning and cognitive analytics. Unconventional Resources Technology Conference
6. Baraboshkin EE, Ismailova LS, Orlov DM, Zhukovskaya EA, Kalmykov GA, Khotylev OV, Baraboshkin EY, Koroteev DA (2020) Deep convolutions for in-depth automated rock typing. Comput Geosci 135:104330
7. Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13(1):281–305
8. Bize-Forest N, Lima L, Baines V, Boyd A, Abbots F, Barnett A et al (2018) Using machine-learning for depositional facies prediction in a complex carbonate reservoir. Society of Petrophysicists and Well-Log Analysts, pp 1–11
9. Ceci M, Hollmén J, Todorovski L, Vens C, Džeroski S (2017) Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, vol 10535. Springer, Berlin
10. Chawshin K, Gonzales A, Berg CF, Varagnolo D, Heidari Z, Lopez O (2021) Classifying lithofacies from textural features in whole-core ct-scan images. SPE Reserv Eval Eng 24(02):341–357
11. Chollet F (2015) Keras: The python deep learning library. Astrophysics Source Code Library: ascl-1806
12. De Lima R, Suriamin F, Marfurt KJ, Pranter MJ (2019a) Convolutional neural networks as aid in core lithofacies classification. Interpretation 7(3):SF27–SF40
13. De Lima RP, Bonar A, Coronado DD, Marfurt K, Nicholson C (2019b) Deep convolutional neural networks as a geological image classification tool. Sediment Rec 17:4–9
14. De Lima RP, Duarte D, Nicholson C, Slatt R, Marfurt KJ (2020) Petrographic microfacies classification with deep convolutional neural networks. Comput Geosci 142:104481
15. Dubois MK, Bohling GC, Chakrabarti S (2007) Comparison of four approaches to a rock facies classification problem. Comput Geosci 33(5):599–617
16. Godbole S (2002) Exploiting confusion matrices for automatic generation of topic hierarchies and scaling up multi-way classifiers. Annual Progress Report. Indian Institute of Technology, Bombay, India
17. Gonzalez A, Kanyan L, Heidari Z, Lopez O et al (2019) Integrated multi-physics workflow for automatic rock classification and formation evaluation using multi-scale image analysis and conventional well logs. Society of Petrophysicists and Well-Log Analysts
18. Hall B (2016) Facies classification using machine learning. Lead Edge 35(10):906–909
19. Hall BJ, Govert A, Energy C (2016) Techniques for Using Core CT Data for Facies Identification and Analysis. SPE/AAPG/SEG Unconventional Resources Technology Conference
20. Hinton G, Srivastava N, Swersky K (2012) Neural networks for machine learning lecture 6a overview of mini-batch gradient descent
21. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012) Improving neural networks by preventing co-adaptation of feature detectors. CoRR, abs/1207.0580
22. Horrocks T, Holden EJ, Wedge D (2015) Evaluation of automated lithology classification architectures using highly-sampled wireline logs for coal exploration. Comput Geosci 83:209–218
23. Hutter F, Hoos HH, Leyton-Brown K (2011) Sequential model-based optimization for general algorithm configuration. In: International Conference on Learning and Intelligent Optimization. LION 2011: Learning and Intelligent Optimization, pp 507–523
24. Ian Goodfellow YBaAC (2016) Deep Learning. MIT Press.
25. Imamverdiyev Y, Sukhostat L (2019) Lithological facies classification using deep convolutional neural network. J Petrol Sci Eng 174:216–228

26. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of ICML, pp 448–456
27. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM computing surveys (CSUR) 31(3):264–323
28. King B (1967) Step-wise clustering procedures. J Am Stat Assoc 62(317):86–101
29. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. In: International Conference on Learning Representations
30. Kirichev M, Slavov T, Momcheva G (2021) Fuzzy u-net neural network architecture optimization for image segmentation. IOP Conf Ser Mater Sci Eng 1031(1):012077
31. Korjani M, Popa A, Grijalva E, Cassidy S, Ershaghi I et al (2016) A new approach to reservoir characterization using deep learning neural networks. Presented at the SPE Western Regional Meeting, Anchorage, Alaska, USA, SPE-180359-MS
32. Li L, Jamieson K, DeSalvo G, Rostamizadeh A, Talwalkar A (2017) Hyperband: A novel bandit-based approach to hyperparameter optimization. J Mach Learn Res 18(1):6765–6816
33. Malki H, Baldwin J, Kwari M et al (1996) Estimating permeability by use of neural networks in thinly bedded shaly gas sands. SPE Comput Appl 8(02):58–62
34. Mustra M, Delac K, Grgic M (2008) Overview of the dicom standard. In: 50th International Symposium ELMAR, 1, pp 39–44
35. Nagy G (1968) State of the art in pattern recognition. Proc IEEE 56(5):836–863
36. Odi U, Nguyen T (2018) Geological facies prediction using computed tomography in a machine learning and deep learning environment. Unconventional Resources Technology Conference, Society of Exploration Geophysicists, pp 336–346. URTEC-2901881-MS
37. O'Malley T, Bursztein E, Long J, Chollet F, Jin H, Invernizzi L (2019) Keras Tuner. https://github.com/keras-team/keras-tuner
38. Rafik B, Kamel B (2017) Prediction of permeability and porosity from well log data using the nonparametric regression with multivariate analysis and neural network, hassi r'mel field, algeria. Egypt J Petrol 26(3):763–778
39. Renter JA (1989) Applications of computerized tomography in sedimentology. Marine Georesour Geotechnol 8(3):201–211
40. Rogachev A, Melikhova E (2020) Automation of the process of selecting hyperparameters for artificial neural networks for processing retrospective text information. IOP Conf Ser Earth Environ Sc 577:012012
41. Rogers S, Chen H, Dt K-M, Fang J (1995) Predicting permeability from porosity using artificial neural networks. AAPG Bulletin 79(12):1786–1797
42. Rogers SJ, Fang J, Karr C, Stanley D (1992) Determination of lithology from well logs using a neural network. AAPG Bulletin 76(5):731–739
43. Schneider CA, Rasband WS, Eliceiri KW (2012) Nih image to imagej: 25 years of image analysis. Nat Methods 9(7):671–675
44. Scholkopf B, Smola AJ (2018) Learning with kernels: support vector machines, regularization, optimization, and beyond. Adaptive Computation and Machine Learning series, MIT Press, Cambridge MA
45. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations (ICLR)
46. Sneath PH, Sokal RR et al (1973) Numerical taxonomy. The principles and practice of numerical classification. W. H. Freeman, San Francisco, CA
47. Snoek J, Larochelle H, Adams RP (2012) Practical bayesian optimization of machine learning algorithms. Advances in neural information processing systems, pp 2960–2968
48. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
49. Tschannen V, Delescluse M, Rodriguez M, Keuper J (2017) Facies classification from well logs using an inception convolutional network. Computer Vision and Pattern Recognition
50. Valentin MB, Bom CR, Coelho JM, Correia MD, Márcio P, Marcelo P, Faria EL (2019) A deep residual convolutional neural network for automatic lithological facies identification in brazilian pre-salt oilfield wellbore image logs. J Petrol Sci Eng 179:474–503
51. Ward JH Jr (1963) Hierarchical grouping to optimize an objective function. J Am stat Assoc 58(301):236–244
52. Wong P, Henderson D, Brooks L et al (1998) Permeability determination using neural networks in the ravva field, offshore india. SPE Reserv Eval Eng 1(02):99–104
53. Yamashita R, Nishio M, Do RKG, Togashi K (2018) Convolutional neural networks: an overview and application in radiology. Insights Imaging 9(4):611–629