Journal of Petroleum Science and Engineering 206 (2021) 109006

Contents lists available at ScienceDirect

Journal of Petroleum Science and Engineering

journal homepage: www.elsevier.com/locate/petrol

At-bit estimation of rock density from real-time drilling data using deep learning with online calibration

Mikkel Leite Arnø [a,*], John-Morten Godhavn [b], Ole Morten Aamo [a]

[a] Department of Engineering Cybernetics, Norwegian University of Science and Technology, 7491 Trondheim, Norway
[b] Equinor Research Center, 7053 Ranheim, Norway

ARTICLE INFO

Keywords:
At-bit virtual density log
Deep learning
Streaming learning

ABSTRACT

We present a novel streaming learning approach, utilizing a deep neural network (DNN) to learn from data available during operation to estimate at-bit density using drilling parameters. Since every wellbore is different, the relationship between drilling parameters and at-bit density varies. Equipment used, well trajectory, friction and bit wear are examples of conditions that affect this relationship and makes a pre-trained model unable to represent an accurate input/output mapping applicable to all wells. However, using delayed density log measurements, continuously supervising updates to the model is possible during operation. The algorithm has been tested on drilling data from wells on a field operated by Equinor and compared to a standard deep learning approach, where results show that a streaming learning approach outperforms the traditional method. Statistical analyses have been performed to verify the statistical significance and effect size on the data sets. Data visualizations using a t-distributed Stochastic Neighbor Embedding (t-SNE) indicate that the relationship between drilling parameters and density log indeed vary between wellbores, making generalizability an issue for a traditional supervised learning approach to this problem, and motivating a streaming learning approach. Using the proposed method, more accurate at-bit estimates can be made, providing preliminary indications ahead of the tool placed 20–30 m behind the bit, which, dependent on rate of penetration (ROP), will be available 20–120 min later.

1. Introduction

The drilling operation is complex. It is also subject to significant uncertainty, which needs to be managed in order to ensure a safe and efficient drilling operation. One such source of uncertainty is at-bit lithology, which is a central part of the environment the drilling system is in interaction with. Lithology evaluation is relevant for best-practice selection of suitable drilling parameters, evaluation of reservoir structure and evaluation of well placement, to mention a few. Logging while drilling (LWD) tools (Arps and Arps, 1964) are typically used by experts to classify lithology. However, they are placed some distance behind the bit, making at-bit lithology evaluation directly from LWD tools impossible. The density log is an LWD tool that can be used to separate harder lithologies such as stringers from softer lithologies. Although not a conclusive lithology indicator on its own, it provides valuable information on downhole conditions, and in combination with other LWD logs an accurate understanding of downhole lithology can be achieved. This tool is typically mounted 20–30 m behind the bit. Due to the placement of these tools, drilling parameters are the earliest indicators for changes in at-bit lithology, although they are difficult for humans to interpret manually. Deep learning specializes in finding patterns in data, and can be used to find a mapping from drilling parameters to density log, although every wellbore is different with case-specific conditions such as equipment used, well trajectory, friction and bit wear. Bit wear, for example, is very difficult to estimate due to vast uncertainty. During drilling, as the bit is gradually worn, the bit-rock interaction will change (Waughman et al., 2002). A static model might misinterpret the dulled bit as a harder lithology, and thus overestimate bulk density.

LWD tools and deep neural networks (DNNs) have been combined before in Rolon et al. (2009), where a DNN takes as input a set of LWD logs and outputs an estimate for another logging tool. They present the results for three different input/output combinations, in which the estimated logs are: resistivity, density and neutron. As LWD tools are expensive, companies do not always use all of them, resulting in a less complete image of lithology properties, and Rolon et al. (2009) was motivated by reducing these costs, and to estimate logs when missing. In Zhang et al. (2018) a long short-term memory (LSTM) model is used for virtual log generation. They present an experiment

* Corresponding author.
E-mail address: mikkel.l.arno@ntnu.no (M.L. Arnø).

https://doi.org/10.1016/j.petrol.2021.109006
Received 22 March 2021; Received in revised form 1 May 2021; Accepted 21 May 2021
Available online 27 May 2021

Abbreviations

ANFIS	Adaptive Network-Based Fuzzy Inference System
BHA	Bottomhole Assembly
DTC	Compressional-Slowness
CVAE	Conditional Variational Autoencoder
DNN	Deep Neural Network
Q	Flow
GRU	Gated Recurrent Unit
HKLD	Hook Load
HMSE	Hydro-Mechanical Specific Energy
LWD	Logging While Drilling
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MSE	Mechanical Specific Energy
PDC	Polycrystalline Diamond Compact
ROP	Rate of Penetration
SVM	Support Vector Machine
RPM	Surface Drill string Rotation
T	Surface Torque
t-SNE	t-distributed Stochastic Neighbor Embedding
WOB	Weight on Bit

where the neutron porosity, delta-time shear, and array induction two-foot resistivity are estimated based on gamma ray and delta-time compressional. Results include comparisons between the LSTM and DNN, where they found the LSTM to be superior for their study. Another study (Osarogiagbon et al., 2020) simulates missing data in a wellbore, and presents results for several machine learning algorithms attempting to estimate gamma ray log from drilling parameters. In Jeong et al. (2021), a conditional variational autoencoder (CVAE) is used to estimate shear-slowness (DTS) from other logs such as gamma ray, neutron porosity, bulk density and compressional-slowness (DTC). Their results are compared to that of an LSTM and a bi-directional LSTM. Gowida et al. (2020) presents experiments using both a DNN and an adaptive network-based fuzzy inference system (ANFIS) for estimation of density log using drilling parameters as input. The models in this study are trained and tested on 2400 observations from the same well, where missing data is simulated for a section of the well, on which the trained models perform with high accuracy. If a well has available log data for only certain sections, these standard deep learning methods could be applied to fill in sections of missing log data. A novel Bayesian neural network approach, using neutron porosity, gamma ray, deep resistivity, photoelectric factor and density logs to estimate sonic log, is presented in Feng et al. (2021b). Their results are comparable to that of traditional neural networks, and in addition offers quantification of uncertainty in the predictions. Lithology classification based on LWD logs using data-driven methods has been presented in several previous publications. Examples of methods include scalable gradient boosted decision trees (Dev and Eden, 2019), support vector machines (SVMs) (Al-Anazi and Gates, 2010) and bidirectional gated recurrent units (GRUs) (Zeng et al., 2020; Tian et al., 2021). Evaluation studies (Xie et al., 2018) have also been presented. Other downhole characteristics have also been addressed using machine learning. In Tunkiel et al. (2021), a streaming learning system for inclination prediction in directional drilling using a GRU is presented. This work is motivated by avoiding delayed corrective actions, thus improving well placement. It is also argued that standard machine learning approaches without online retraining fail to generalize well to different wellbores. In Feng et al. (2021a), a Bayesian approach is applied to fault detection using a convolutional neural network (CNN), allowing risk evaluation through uncertainty quantification.

The contribution of this work is two-fold. First, we present a novel streaming learning approach, using a DNN to solve the problem of estimating at-bit density log from drilling parameters. A pre-trained model continuously learns from the data stream available during operation, allowing the model to adapt to case specific conditions. The method is applied to data from several wellbores on a field operated by Equinor to demonstrate performance, and compared to the performance of a baseline model, which represents the traditional approach to log estimation. Next, an unsupervised learning data analysis is performed, which visualizes how data from different wellbores are structured differently. This analysis indicates the weakness in using a pre-trained and static model to estimate at-bit density using drilling parameters, motivating the streaming learning approach.

The paper is structured as follows: Section 2 presents the data used, along with the different methods and algorithms used in this study, including our own n-bin experience replay buffer. Section 3 presents results for data visualizations and at-bit density estimates versus measurements. Lastly, Section 4 offers conclusions and suggestions for further work.

2. Data and methodology

2.1. Data

Data from the reservoir sections of 9 different wells on a homogeneous field operated by Equinor were gathered for this work, for a total of 1.75 million observations. The training set contained 5 wells, for a total of approximately 740 000 observations, and the validation set consisted of 1 well with approximately 365 000 observations. The test set contained 3 wells, with a total of 645 000 observations. Data cleaning was performed by visual inspection. Obviously erroneous measurements were removed by logic specifying reasonable values. Next, data was centered and scaled to have zero mean and unit variance before training commenced.

2.2. Measurements description

The drilling system is composed of several subsystems, where data acquisition is different for different subsystems. For the purpose of this work, we divide measurements into two groups: *surface measurements* and *downhole measurements*. Surface measurements are taken on the rig, and examples are drilling parameters like hook load (HKLD), surface torque (T), surface drill string rotation (RPM), flow (Q) and hook height. Weight on bit (WOB) is closely related to, and derived from HKLD, while bit depth, hole depth and rate of penetration (ROP) are derived from hook height. In this work, downhole measurements refers to the density log. The density logging tool measures density of lithology perpendicular to the drill string by emitting gamma rays and detecting backscatter, giving a measure of average electron density in the lithology, which is strongly correlated with bulk density. In other words, it gives an indirect measurement of the bulk density. These measurements are communicated to the rig by mud pulse telemetry.

It is established (Bourgoyne and Young, 1974) that lithologies with different properties yield different bit-rock interactions, meaning that ROP is dependent on rock properties and input actuation like WOB, T and RPM. This means that information regarding at-bit conditions like density should be latently available in real-time through these surface measurements. The density logging tool on the other hand is installed in the bottomhole assembly (BHA), 20–30 m behind the bit. Dependent on ROP, this amounts to significant time delays on logs in ranges of typically 20–120 min. To eliminate this delay, we propose to estimate the at-bit density from surface measurements.

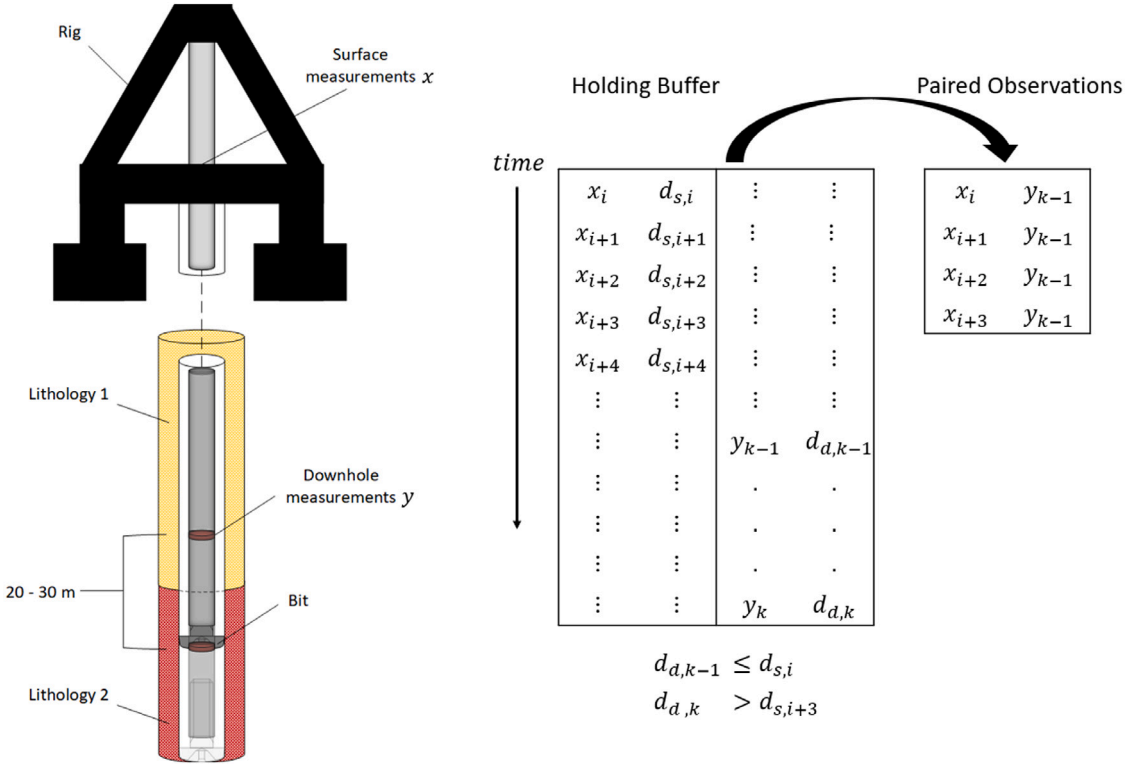

Fig. 1. Schematics illustrating the drilling operation and the availability of measurements.

2.3. Depth correction & resampling

Surface measurements are sampled with a sampling period of typically 2–3 s, while the density log is sampled at a lower rate, typically every 10–15 s. Also, surface measurements and downhole measurements recorded at a given time provide information at different depths. For these reasons, a method to correct for depth and differing sampling rates was required. This was done using a *holding buffer* that stored incomplete observations. As Fig. 1 illustrates, an observation can be completed once the distance between bit and tool has been drilled, and the label corresponding to some set of surface measurements is obtained. We start by denoting the sampling periods for surface measurements x, and the density log y, as T_s and T_d, respectively. The ith surface measurements are taken at time iT_s, so that $x_i = x(iT_s)$. Similarly, the ith density log measurement is taken at time iT_d, so that $y_i = y(iT_s)$. x_i provides information at the bit, at depth $d_{s,i} = d_s(iT_s)$, while y_i, measured at a known distance behind the bit, d_{tool}, provides information at depth $d_{d,i} = d_s(iT_d) - d_{tool}$. Once we obtain the first density log measurement y_k, where $d_{d,k} > d_{s,j}$ for any indices j in the holding buffer, we can pair observations so that each x_j in the holding buffer is paired with the previous density log measurement, y_{k-1}, where $d_{d,k-1} \leq d_{s,j}$. In addition to correcting for depth, this procedure is equivalent to resampling the y's using the forward fill method.

2.4. Variable selection

The variables used as input to the DNN were selected based on domain knowledge and availability. To eliminate the density log delay, we limit ourselves to measurements providing at-bit information, which eliminates other LWD tools. In addition to using drilling parameters, we can perform some feature engineering. Mechanical specific energy (MSE) quantifies the amount of energy required to remove a unit volume of rock. It is a function of ROP, WOB, T and RPM, and known to be different for different lithologies (Dupriest and Koederitz, 2005). MSE is given by:

$$\mathrm{MSE} = \frac{\mathrm{WOB}}{A_b} + \frac{120\pi \cdot \mathrm{RPM} \cdot \mathrm{T}}{A_b \cdot \mathrm{ROP}}, \tag{1}$$

Table 1
Default values set for unknown parameters.

Parameter	Default value
Bit type	Polycrystalline Diamond Compact (PDC)
Junk slot area	14 (in.^{2})
Flow area	0.12 (in.^{2})
Mud weight	20 (ppg)

where units are WOB (lb), T (lb-ft), ROP (ft/h), and A_b is bit area (in.^{2}). Another parameter of interest is the hydro-mechanical specific energy (HMSE), which also accounts for the weakening of the rock ahead of the bit due to flow. HMSE is given by:

$$\mathrm{HMSE} = \frac{\mathrm{WOB}}{A_b} + \frac{120\pi \cdot \mathrm{RPM} \cdot \mathrm{T}}{A_b \cdot \mathrm{ROP}} + \frac{1154\eta \cdot \Delta P_b \cdot \mathrm{Q}}{A_b \cdot \mathrm{ROP}}, \tag{2}$$

where η is the hydraulic energy reduction factor, ΔP_b is the bit pressure drop at the nozzle (psi), and Q is the flow rate (gpm). Some parameters related to the bit and mud were not available to compute the hydraulic contribution of HMSE (Osarogiagbon et al., 2020). Default parameter values were set for these, as described in Table 1.

From the default parameters, we can compute HMSE, and further define the input vector of predictors for the DNN, $\mathbf{x}$, as:

$$\mathbf{x} = [\mathrm{ROP},\ \mathrm{RPM},\ \mathrm{WOB},\ \mathrm{T},\ \mathrm{MSE},\ \mathrm{HMSE}]^\top. \tag{3}$$

The output of the DNN, $\hat{\mathbf{y}}$, is simply at-bit density. As a sanity check, we wish to investigate the correlations between the density log and the inputs to the model, which are illustrated in Fig. 2. The correlations are presented separately for each wellbore, and it can be seen that the strength of the linear relationships varies between wellbores. Correlations between density and ROP are the most consistent. MSE and HMSE are also quite strongly correlated with density for most wellbores, although the strength varies more. Especially for training wells 3 and 5, these relationships are weaker. RPM, WOB and T correlations vary more, although for some wells, these can be seen to be strongly correlated with density. Several of the drilling parameters are controlled by

the driller. Autodrillers can be set to maintain constant WOB or ROP. When the driller suspects that a stringer is being drilled, the WOB is routinely increased while RPM is reduced. This will typically lead to a reduction in T as well. For training wells 2–5, the correlations for RPM, WOB and T in Fig. 2 support this. For training well 1, however, the signs of the correlations for these parameters are inverted. For the validation well, the sign of the correlation for T is inverted. Since these parameters are controlled by the driller, correlations with density will be highly affected by the driller's choices. As an example, if the drilling strategy is to increase both WOB and RPM for stringers, the resulting ROP might increase as well. Isolated, increased ROP for stringers might seem counterintuitive, but would be explained by the overall drilling strategy and the actions performed by the driller. MSE and HMSE on the other hand, eliminate the driller's actions, and define the input energy required to drill through the rock. The correlations between density and these parameters are consistently positive, indicating that more energy is required to drill denser lithology.

2.5. Deep neural networks

The specifics of the DNN used in this work are outlined in this section. First, we provide notation for the DNN:

L : number of layers in the DNN
m : number of observations
f : number of features
x : drilling parameters/features
y : density log/target variable
s_l : number of neurons in layer $l \in 1, \dots, L$
(x_i, y_i) : ith training example, $i \in 1, \dots, m$
$\mathbf{w}^{[l]}$: trainable weight matrix for layer l
$\mathbf{b}^{[l]}$: trainable bias vector for layer l

and

$$\mathbf{x} = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x_1 & x_2 & \dots & x_m \\ \vdots & \vdots & & \vdots \end{bmatrix} \in \mathbb{R}^{f \text{ x } m}, \tag{4}$$

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_m \end{bmatrix} \in \mathbb{R}^{1 \text{ x } m}, \tag{5}$$

$$\mathbf{w}^{[l]} = \begin{bmatrix} \dots & w_1^{[l]\top} & \dots \\ \dots & w_2^{[l]\top} & \dots \\ & \vdots & \\ \dots & w_{s_{l-1}}^{[l]\top} & \dots \end{bmatrix} \in \mathbb{R}^{s_{l-1} \text{ x } s_l}, \tag{6}$$

$$\mathbf{b}^{[l]} = \begin{bmatrix} b_1^{[l]} \\ b_2^{[l]} \\ \vdots \\ b_{s_l}^{[l]} \end{bmatrix} \in \mathbb{R}^{s_l \text{ x } 1}. \tag{7}$$

For a given layer l, $\mathbf{z}^{[l]}$ denotes the linear combination of activations from the previous layer, $\mathbf{a}^{[l-1]}$, determined by the trainable weights $\mathbf{w}^{[l]}$ and biases $\mathbf{b}^{[l]}$:

$$\mathbf{z}^{[l]} = \mathbf{w}^{[l]\top}\mathbf{a}^{[l-1]} + \mathbf{b}^{[l]}\mathbf{1}, \tag{8}$$

where $\mathbf{1}$ is a row vector of ones. Eq. (8) describes the linear component of the forward propagation. Next, $\mathbf{z}^{[l]}$ is passed through the layer activation function $g^{[l]}$, which is the nonlinear component of the forward propagation:

$$\mathbf{a}^{[l]} = g^{[l]}(\mathbf{z}^{[l]}). \tag{9}$$

The leaky ReLU activation function is utilized for every hidden layer, so that

$$\mathbf{a}^{[l]} = \max\{\gamma\mathbf{z}^{[l]}, \mathbf{z}^{[l]}\}, \quad l = 1, \dots, L-1. \tag{10}$$

γ is the slope in the left half plane. Note that $\mathbf{a}^{[0]} = \mathbf{x}$. Finally, the DNN outputs the estimated density at-bit, which is a quantitative output, resulting in a regression layer:

$$\hat{\mathbf{y}} = \mathbf{a}^{[L]} = \mathbf{z}^{[L]}. \tag{11}$$

Eqs. (8)–(11) describe the forward propagation of the DNN. The model can be visualized by a layered model with connected nodes, as shown in Fig. 3. The weight initialization is He normal (He et al., 2015), which mitigates exploding and vanishing gradients by managing the variance of the activations throughout the layers of the network. This is done by pulling the weights in each layer from a truncated normal distribution with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{\frac{1}{s_{[l-1]}}}$. The weight initialization for layer l is then given by:

$$\mathbf{w}^{[l]} \in \mathbb{R}^{s_{l-1} \text{ x } s_l} \sim \mathcal{N}\left(\left[0, (s_{[l-1]})^{-1}\right]\right). \tag{12}$$

The biases, $\mathbf{b}^{[l]}$, are initialized as zeros. The optimizer used was Adam optimization (Kingma and Ba, 2015), which adaptively estimates appropriate momentum for the gradient updates. The Adam optimization algorithm is given by Algorithm 1.

Algorithm 1 Adam Optimization

$V_{\mathbf{dw}} = 0, S_{\mathbf{dw}} = 0, V_{\mathbf{db}} = 0, S_{\mathbf{db}} = 0,$
for each sampled mini-batch **do**
  Forward prop on $\mathbf{x}$:
    $\mathbf{z}^{[1]} = \mathbf{w}^{[1]\top}\mathbf{x} + \mathbf{b}^{[1]}\mathbf{1}$
    $\mathbf{a}^{[1]} = g^{[1]}(\mathbf{z}^{[1]})$
    $\mathbf{z}^{[2]} = \mathbf{w}^{[2]\top}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}\mathbf{1}$
    $\mathbf{a}^{[2]} = g^{[2]}(\mathbf{z}^{[2]})$
    $\vdots$
    $\mathbf{a}^{[L]} = \mathbf{z}^{[L]}$
    $\hat{\mathbf{y}} = \mathbf{a}^{[L]}$
  Compute cost J
  Backpropagate using the chain rule to compute gradients **dw** and **db**
  $V_{\mathbf{dw}} = \beta_1 V_{\mathbf{dw}} + (1-\beta_1)\mathbf{dw}$
  $V_{\mathbf{db}} = \beta_1 V_{\mathbf{db}} + (1-\beta_1)\mathbf{db}$
  $S_{\mathbf{dw}} = \beta_2 S_{\mathbf{dw}} + (1-\beta_2)\mathbf{dw}^2$
  $S_{\mathbf{db}} = \beta_2 S_{\mathbf{db}} + (1-\beta_2)\mathbf{db}^2$
  $V^c_{\mathbf{dw}} = \frac{V^c_{\mathbf{dw}}}{1-\beta_1^r}$
  $V^c_{\mathbf{db}} = \frac{V^c_{\mathbf{db}}}{1-\beta_1^r}$
  $S^c_{\mathbf{dw}} = \frac{S^c_{\mathbf{dw}}}{1-\beta_2^r}$
  $S^c_{\mathbf{db}} = \frac{S^c_{\mathbf{db}}}{1-\beta_2^r}$
  $\mathbf{w} = \mathbf{w} - \alpha\frac{V^c_{\mathbf{dw}}}{\sqrt{S^c_{\mathbf{dw}}}+\varepsilon}$
  $\mathbf{b} = \mathbf{b} - \alpha\frac{V^c_{\mathbf{db}}}{\sqrt{S^c_{\mathbf{db}}}+\varepsilon}$
end for

β_1, β_2 and ε are tunable hyperparameters for Adam optimization, and r is the current iteration number. $V_{\mathbf{dw}}$ and $V_{\mathbf{db}}$ are biased first moment estimates. $S_{\mathbf{dw}}$ and $S_{\mathbf{db}}$ are biased second raw moment estimates. Superscript c denotes their bias-corrected counterparts. **dw** and **db** are $\frac{\partial J}{\partial \mathbf{w}}$ and $\frac{\partial J}{\partial \mathbf{b}}$, respectively. α is the learning rate. Cost J is defined as

$$J = \frac{1}{2m_b}\left|\hat{\mathbf{y}} - \mathbf{y}\right|^2, \tag{13}$$

where m_b denotes the number of observations in a mini-batch.

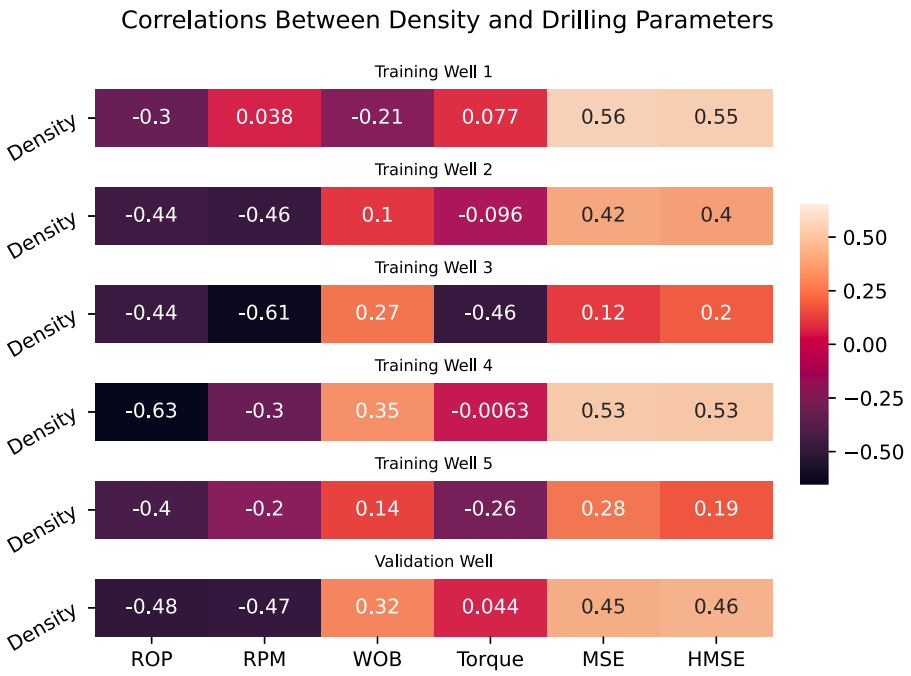

Fig. 2. Correlations between density log and drilling parameters for training wells and validation well.

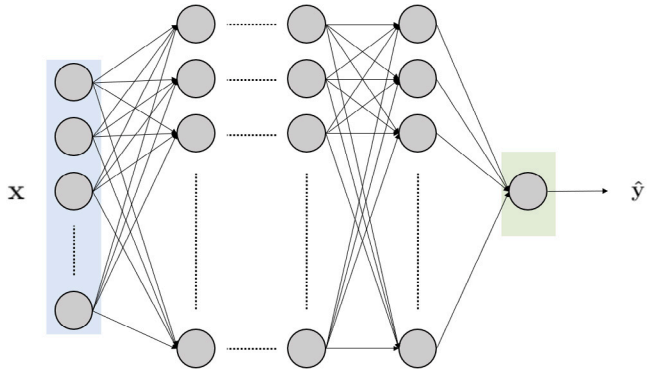

Fig. 3. Deep neural network. **Input:** x. **Output:** ŷ.

2.6. *n*-bin prioritized experience replay

In conventional supervised learning, a model is typically trained on a fixed data set for multiple passes, aiming to converge towards a well-performing model for unseen data assumed to come from the same distribution. However, this simple assumption breaks down in many cases due to a variety of factors, such as shift in the independent or dependent variables, or due to an evolving underlying process. This phenomenon is commonly known as nonstationarity or concept drift, and is harmful to the predictive power of such models (Ditzler et al., 2015; Elwell and Polikar, 2011). The aim of streaming learning is to continuously update the model to correct for these effects. However, a common problem in streaming learning is catastrophic forgetting (McCloskey and Cohen, 1989), where old representations are forgotten due to adaptation to the non-stationary environment. In our attempt to adapt to drifting concepts while mitigating catastrophic forgetting, an n-bin prioritized experience replay $D \in \mathbb{R}^{n \times N}$ buffer was developed. The prediction space $y \in [y_{min}, y_{max}]$ is divided into n bins, and each bin in the buffer contains N observations. This configuration ensures that the replay buffer always contains observations covering the span of the prediction space, and thus that the model is better equipped to give accurate estimates overall, and not be biased by the distribution of the latest available observations. A similar configuration has been used for multi-class classification, with one buffer for each class (Hayes et al., 2019). When learning from a data stream, an observation (x_i, y_i) is allocated to a bin in the experience replay buffer based on the value of y_i, and consequently, the oldest observation of that bin is discarded. The mini-batch used for backpropagation is sampled from the experience replay by prioritization using the softmax function. At every update of the model, the observations are given a probability of being sampled for the next update by:

$$p_i = \frac{e^{(y_i - \hat{y}_i)^2}}{\sum_{c=1}^{C} e^{(y_c - \hat{y}_c)^2}}, \tag{14}$$

where $C = nN$ is the total number of observations in the buffer. It can be seen that a higher model error on an observation results in a higher probability of being sampled for the next training step. This method is well known in the reinforcement learning field, and has shown to be an improvement over sampling observations from a uniform distribution (Schaul et al., 2016), due to added focus on areas where the model performs poorly. The n-bin prioritized experience replay algorithm is given formally as:

Algorithm 2 n-Bin Prioritized Experience Replay

```
Load n-bin replay buffer D filled with historical data
while learning do
    if new observations (x, y) available then
        Enqueue observations (x, y) in appropriate bin in D
        Dequeue appropriate bin
        Calculate p_i, i = 1, ..., C
        Randomly sample K observations by probabilities p_i from D
        Backpropagate on sampled observations
    end if
end while
```

Retention of observations from the entire range of the prediction space in this fashion allows smaller replay buffers. Rather than retaining all previous observations for further training, a small subset is kept, making this method memory efficient. In addition, dividing the prediction space into bins ensures that historical observations in one

bin are available as long as no new observations stream into that bin. Thus, the algorithm can take into account older representations while making updates during operation.

2.7. Streaming learning system

The streaming learning system is based on a pre-trained and validated model, which is loaded as the *baseline model*, on which to iteratively perform updates during operation. As surface measurements become available during operation, they are first fed to the DNN to estimate at-bit density. Subsequently, they are stored in the holding buffer until its corresponding label, the density log measurement, is available. At this point, the completed observation is moved from the holding buffer to the experience replay buffer, and then used for further training. Note that the learning phase begins at time τ when $d_{d,\tau} \geq d_{s,0}$, meaning that we are not interested in the density log measurements above the first available drilling parameter measurements. Algorithm 3 summarizes the method.

Algorithm 3 System Overview

```
Pre-train and validate model on historical data
Load model as baseline model
Load experience replay buffer filled with historical data
while learning do
    Receive x_i, estimate ŷ_i
    Store x_i along with d_{s,i} in holding buffer
    if y_k available then
        for all observations in holding buffer do
            Find indices j for all observations satisfying d_{d,k} > d_{s,j}
            Pair x_j with y_{k−1}, ∀j
        end for
        Remove these observations from holding buffer, and update experience replay buffer
        Backpropagate on observations sampled from experience replay buffer
    end if
end while
```

3. Results

3.1. Pre-training and validation

Pre-training and validation of the baseline model was an iterative approach. Hyperparameters such as DNN architecture (neurons and layers), learning rate, mini-batch size and number of epochs were tuned in an informal search based on validation set performance. Upon arrival on a satisfactory model, the streaming learning hyperparameters were tuned on the same data set. Table 2 shows the hyperparameter settings. The resulting baseline model from pre-training and validation had 3 hidden layers, each with 12 neurons. Streaming learning hyperparameters refers to the settings for streaming learning during operation. It can be seen that here, the learning rate and mini-batches sampled from the experience replay are smaller than during pre-training. Experience replay bin limits refers to the limits on the density log used in Algorithm 2 to select the appropriate bin. Density log for the data used in this study was in the range 2.0–2.7 (g/cm^3), so that observations below the first bin limit would belong to the *low* bin, observations between the two limits belonged to the *mid* bin, and lastly, observations above the second bin limit belonged to the *high* bin. The hyperparameter values presented in Table 2 should make decent initial values for similar problems. However, note that the experience replay bin limits in particular will be very case-dependent. The bin limits ensure retention of observations in different parts of the prediction space, making an understanding of the dependent variable key. We suggest consideration of important areas in the prediction space when tuning these parameters. If several areas are important, the number of bins could also be increased.

Table 2
Summary of hyperparameters.

Pre-training hyperparameter	Value
Learning rate	$7.5 \cdot 10^{-5}$
Hidden layers	3
Neurons in hidden layers	12
Mini-batch size	128
Epochs	25
Optimizer	Adam
Streaming learning hyperparameter	**Value**
Learning rate	$7.5 \cdot 10^{-6}$
Experience replay bins	3
Experience replay bin sizes	256
Experience replay bin limits	[2.115, 2.535]
Mini-batch size	16
Epochs	1
Optimizer	Adam

3.2. Streaming learning results

This subsection is dedicated to the presentation and evaluation of the performance of the streaming learning approach compared to the baseline model. We provide results for the validation well, along with the 3 test wells. Along with each plot, the mean absolute error (MAE) is presented. MAE is given by:

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m} |y_i - \hat{y}_i|. \tag{15}$$

Although the raw data used for the algorithm was time data, the results are converted into depth data with equidistant points at a resolution of 1 m by downsampling. At every integer depth, data points within 0.5 m are averaged. Fig. 4 shows the comparison between the baseline and streaming learning approach on the validation well. For the baseline model, one can see that the model suffers from bias on low-density observations throughout the wellbore, and that the model gradually overestimates density from 6000 m and towards the end. An inspection of the raw data revealed that for this wellbore, the torque gradually increased which indicates concept drift. The streaming learning approach can be seen to mitigate both the bias and the drift, resulting in an MAE decrease from 0.1087 g/cm^3 to 0.0615 g/cm^3. This is a relative decrease of approximately 43%.

Figs. 5 and 6 show the same comparisons for test wells 1 and 2. On these wellbores, the baseline model performs quite well. Still, on test set 1 at approximately 5000–5500 m depth, the baseline model is visibly off measurements. On test set 2, the baseline model overestimates low density observations. For test wells 1 and 2, the streaming learning approach reduces these errors, resulting in 21% and 7% decreases in MAE, respectively. For test well 2, the resulting improvement from the streaming learning algorithm is incremental, and the added complexity of method implementation compared to the baseline model counterpart may not be worthwhile. For wellbores with similar input/output relationships to those of the training wells, the baseline model will perform well without the need for continuous updates. However, in many cases it is not known in advance whether or not this will be the case. Such considerations should be a part of the validation process. If it is found that a static model performs well during validation, a traditional supervised learning approach might be sufficient. If, however, heterogeneity between wellbores is found during this process, a streaming learning approach might be warranted to correct for drifts and shifts in concepts.

Lastly, Fig. 7 shows the results for test well 3. It can be seen that the baseline model is very flat, and unable to capture any trends in the at-bit density. This indicates that the mapping from drilling parameters to at-bit density is significantly different from that of the wellbores in the training set. We can call this a shift in concepts. Although the streaming learning approach is not as good as for the previous wells, it is to a much larger degree able to capture the trends by adapting to

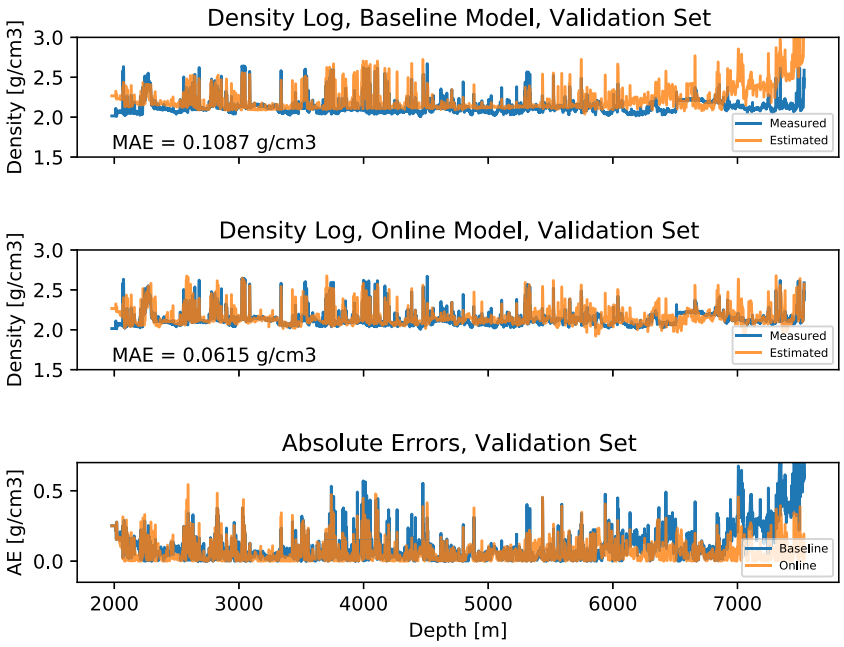

Fig. 4. Measured and estimates on the validation set. **Top:** Baseline model performance. **Middle:** Streaming learning performance. **Bottom:** Absolute errors (AE).

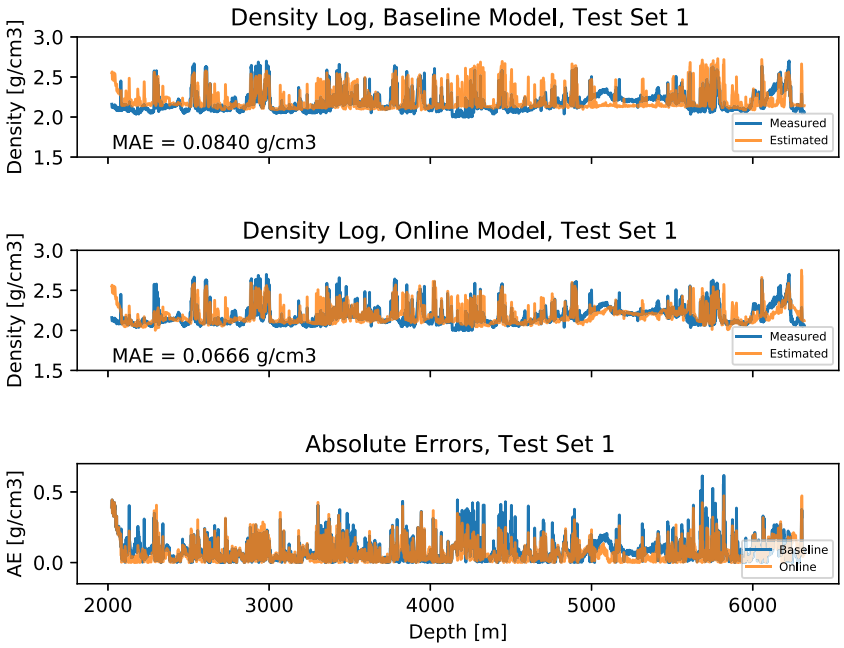

Fig. 5. Measured and estimates on test set 1. **Top:** Baseline model performance. **Middle:** Streaming learning performance. **Bottom:** Absolute errors (AE).

the concept shift, resulting in an MAE decrease of 36%. Even though the maximum recorded density for these wellbores is 2.7 g/cm^3, it can be seen that the online models estimates densities above this value at depths 5030 m and 5110 m. At depth 5030 m, an unusually low value for RPM was measured, along with a high WOB. At 5110 m, an unusually high MSE was recorded, indicating an unusual combination of drilling parameters. As neural networks do not extrapolate well, these events result in erroneously high density estimates. The flattening effect apparent for this wellbore can also to some extent be observed in the other wellbores. In the other wellbores this takes the form of a cut-off effect, so that low-density observations are not estimated well by the baseline model. This is likely due to heterogeneity between the training wellbores. Since the input/output relationships in the wells differ, the baseline model is trained to fit an "average" of these, resulting in a compressive effect on the predictions.

The rate of adaption to newly available data can be seen to differ for the different data sets. For the validation, test 1 and test 2 sets, we can see that the baseline model overestimates the low-density observations in the beginning of the drilling operation. For the streaming learning approach, we can see that this is quickly mitigated by the online re-training. Also for test set 3, the performance is improved, although the corrections are not as fast. From the poor baseline model performance, we could argue that this wellbore is the most different from the training sets, and that the speed of the online retraining is dependent on how

Fig. 6. Measured and estimates on test set 2. **Top:** Baseline model performance. **Middle:** Streaming learning performance. **Bottom:** Absolute errors (AE).

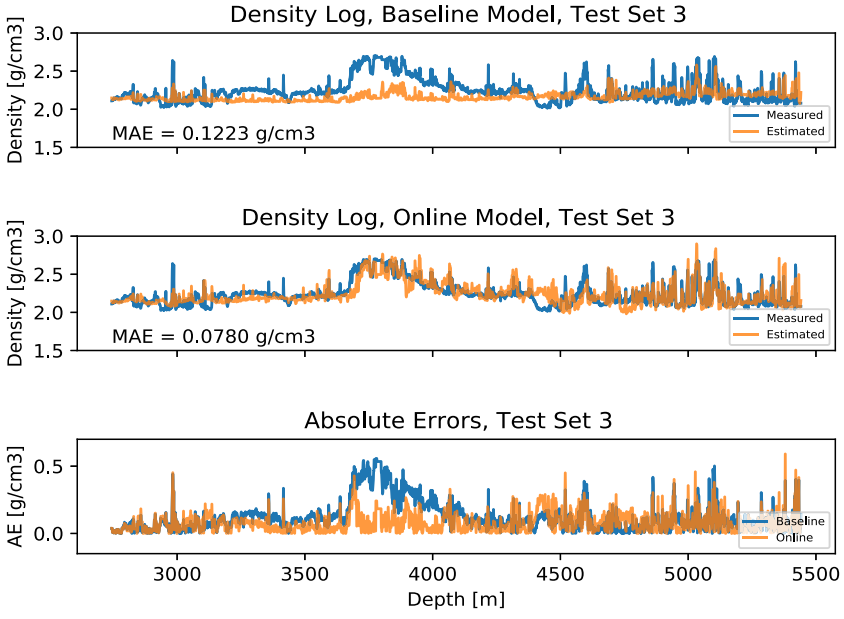

Fig. 7. Measured and estimates on test set 3. **Top:** Baseline model performance. **Middle:** Streaming learning performance. **Bottom:** Absolute errors (AE).

much the model must be corrected to fit well to the newly available data. The learning rate of the system determines the size of the gradient descent updates, however, setting this hyperparameter too high can lead to unstable optimization. Thus, the online retraining rate must be a trade-off between speed and stability.

From the absolute errors on the depth converted data sets, we can investigate the statistical significance of the difference in performance between the two approaches, along with effect size and statistical power. We can perform paired, two-tailed t-tests for each wellbore to determine statistical significance. The null hypothesis is $H_0 : \mu_1 = \mu_2$, where μ_1 is the population mean absolute error for the baseline model, and μ_2 is the population mean absolute error for the streaming learning approach. We also determine Cohen's d, which is a measure of standardized effect size. The proposed interpretation of d is along a continuum, with conventional *small*, *medium* and *large* effect sizes at approximately 0.2, 0.5 and 0.8, respectively. Combined with a statistical significance level $\alpha_0 = 0.05$, which is the accepted probability that we are failing to reject a false null hypothesis (type I error), we can find the statistical power, $1-\beta$, where β quantifies the probability of rejecting the null hypothesis given that it is actually correct (type II error) (Portney, 2020). In Table 3 we present m, the number of observations in each depth converted data set, MAE_b, the mean absolute error using the baseline model, MAE_s the mean absolute error using the streaming learning approach, $\Delta\text{MAE} = \text{MAE}_b - \text{MAE}_s$, Cohen's d, p-values and statistical power, $1-\beta$. We can see from the p-values that the difference in performance are statistically significant for all wellbores. Thus, we reject H_0. From Cohen's d, we see that the effect size on the validation and test 3 wellbores are medium to large. The effect size for

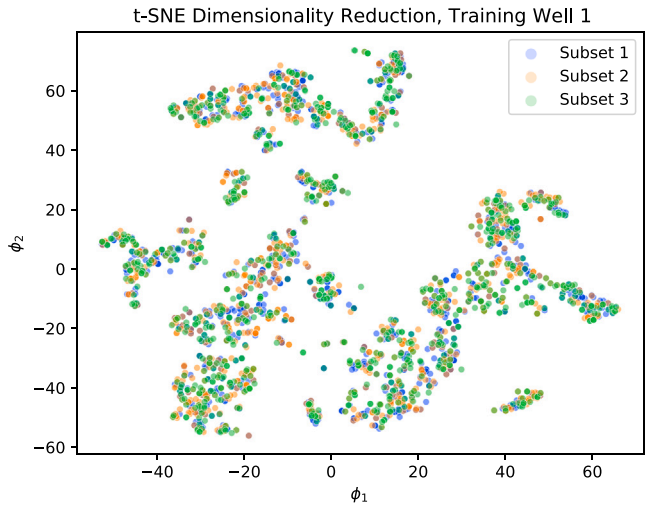

Fig. 8. t-SNE plot for 3 random subsets from training set 1.

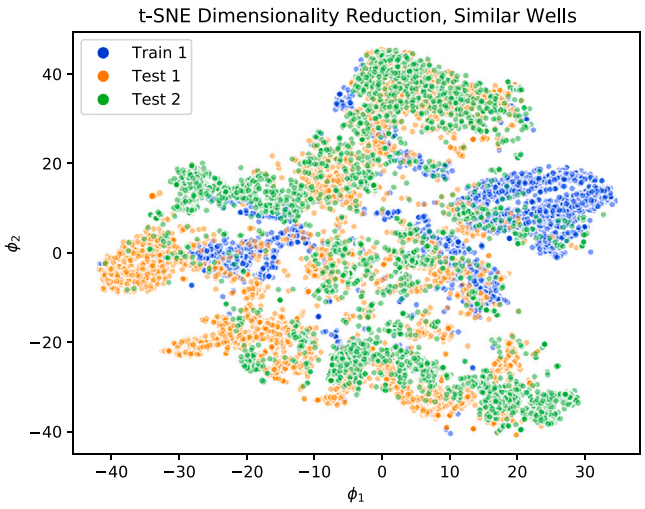

Fig. 9. t-SNE plot for 3 random subsets from training set 1, test set 1 and test set 2.

the test 1 wellbore is medium, and small for the test 2 wellbore. For all wellbores, the probability of committing a type II error, is ≈ 0.

3.3. Data visualization with t-SNE

Data visualizations with t-SNE in two dimensions are provided to complement the results in Figs. 4–7 (see Appendix for a brief summary of the t-SNE method). From these plots, we have observed several different scenarios: drift, minor offsets and significant concept shift. We wish to visualize the data to better understand these effects. In the t-SNE analyses, a set of data points in 7 dimensions, taken as:

$$\theta_i = [\mathrm{ROP}_i, \mathrm{RPM}_i, \mathrm{WOB}_i, \mathrm{T}_i, \mathrm{MSE}_i, \mathrm{HMSE}_i, \mathrm{Density\ log}_i], \tag{16}$$

is reduced to the set of data points $[\phi_1, \phi_2]$ in two dimensions. Using this setup, we can identify the structures of the data in the wellbores, for example if similar drilling parameters result in similar or different density log measurements for different wellbores. Fig. 8 illustrates the two-dimensional mapping for 3 random subsets from the same training well, each containing one third of the observations. As expected, these subsets occupy similar spaces in the plot. This indicates that a model trained on one of these subsets could perform well on the other two. Fig. 9 illustrates the same analysis on training well 1, and test sets 1 and 2. The baseline model was found to perform quite well for these test sets, and from this t-SNE analysis, we can see that the observations from these wellbores indeed overlap reasonably well with the training well in the two-dimensional mapping. Lastly, we inspect Fig. 10, which shows the two-dimensional mapping for training well 1, along with the validation well and test well 3. The validation well and test well 3 exhibited concept drift and shift respectively, and the baseline model performed poorly on them. From the plot, we can observe natural clustering within each wellbore, which might be attributed to the fact that low-density and high-density observations should be different. However, it can be seen that several clusters from the validation well and test well are isolated from each other and from the training well. As they form separate clusters, this indicates that all 3 wellbores belong to their own natural grouping. Because of this, any baseline model regardless of model type and architecture, cannot be trained on this training well and be expected to generalize well to the others. In other words, streaming learning is essential for obtaining acceptable performance in this case.

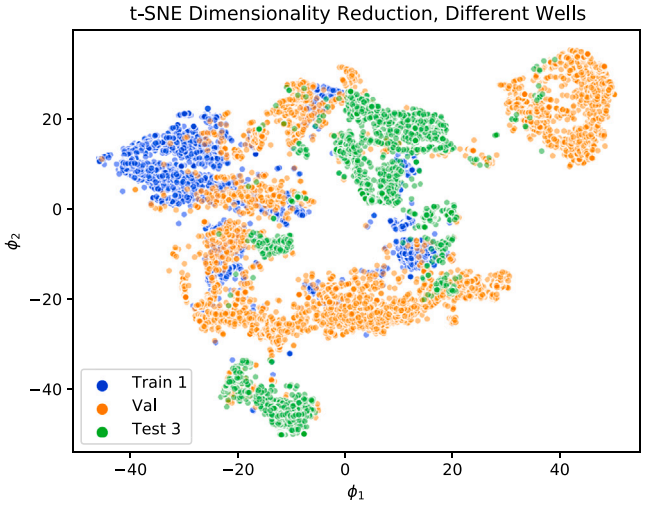

Fig. 10. t-SNE plot for 3 random subsets from training set 1, validation set and test set 3.

4. Conclusions & further work

Since the density log is typically mounted 20–30 m behind the bit, the driller is rendered blind to at-bit conditions. In the current work, a DNN is used to estimate at-bit density from drilling parameters to eliminate this delay. The DNN is pre-trained on historical data from wells on a field operated by Equinor, serving as a baseline model. Using delayed density log measurements, the model is continuously updated during operation. This streaming learning approach allows adjustments to changing conditions that are not explicitly included in the model as variables. Comparisons of the results for the baseline model and the streaming learning approach indeed show that performance in terms of mean absolute error can be greatly improved using a streaming learning approach. This is especially true for wellbores where the relationships between drilling parameters and density log are significantly different from what the model has seen before during training. The method gives preliminary at-bit density log estimates that are available in real-time, while adapting to change, thus increasing generalizability so that the model is applicable to a wider range of cases. t-SNE is used to visualize the data from different wellbores and shows that the data sets are structurally different. This indicates that a pre-trained model, regardless of model architecture, will be unable to generalize to all

Table 3
Results of power analysis.

Data set	m	MAE_b [g/cm^3]	MAE_s [g/cm^3]	ΔMAE [g/cm^3]	d	p	$1-\beta$
Val	5556	0.1087	0.0615	0.0472	0.61	$1.2 \cdot 10^{-206}$	≈ 1
Test 1	4296	0.0840	0.0666	0.0174	0.42	$9.1 \cdot 10^{-82}$	≈ 1
Test 2	3780	0.0762	0.0712	0.005	0.14	$9.3 \cdot 10^{-10}$	≈ 1
Test 3	2698	0.1223	0.0780	0.0443	0.58	$3.4 \cdot 10^{-92}$	≈ 1

the wellbores used in the analysis. It also serves as motivation for a streaming learning approach.

For further work, attempts should be made at a streaming learning approach for other LWD tools, as having several at-bit logs would further nuance the bottomhole information.

CRediT authorship contribution statement

Mikkel Leite Arnø: Conceptualization, Methodology, Software, Validation, Formal analysis, Visualization, Writing - original draft, Writing - review & editing. **John-Morten Godhavn:** Conceptualization, Methodology, Writing - review & editing, Supervision, Project administration. **Ole Morten Aamo:** Conceptualization, Methodology, Writing - review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors wish to thank Equinor, Norway and NTNU, Norway for funding this project, and the Troll license for sharing their drilling data. We also want to thank colleagues at Equinor who have provided valuable domain expertise, among them the REAL team.

Appendix. *t-distributed Stochastic Neighbor Embedding*

t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton, 2008) falls within the category of unsupervised learning algorithms. It is typically used for visualization of high-dimensional data by dimensionality reduction. It is a nonlinear method capable of preserving the local structure of high-dimensional data while revealing global structures such as clusters. When converting the high-dimensional data $\vartheta = \{\theta_1, \theta_2, \dots \theta_m\}$ to a low-dimensional mapping $\varphi = \{\phi_1, \phi_2, \dots, \phi_m\}$, t-SNE starts by converting the high-dimensional Euclidean distances between data points to similarities $p_{i|j}$, quantifying the conditional probability that θ_i would pick θ_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at θ_i. This similarity is given as:

$$p_{i|j} = \frac{\exp\left(-\|\theta_i - \theta_j\|^2/2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|\theta_i - \theta_k\|^2/2\sigma_i^2\right)}. \tag{17}$$

From these, the joint probabilities are defined to be symmetrized conditional probabilities, that is:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}. \tag{18}$$

σ_i is the variance of the Gaussian centered at θ_i. This parameter can be indirectly tuned by the user through the *perplexity* hyperparameter. For a user-specified perplexity, t-SNE performs a binary search for the value of σ_i that produces a probability distribution P_i over all the other data points with the same perplexity $Perp(P_i)$. This is defined as:

$$Perp(P_i) = 2^{H(P_i)}, \tag{19}$$

where $H(P_i)$ is the Shannon entropy measured in bits:

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{i|j}. \tag{20}$$

Perplexity can be viewed as a smoothing measure for the number of effective neighbors, and t-SNE is robust to changes in this parameter. For the low-dimensional mappings, the similarities q_{ij} are computed using a Student t-distribution with one degree of freedom, resulting in:

$$q_{ij} = \frac{(1 + \|\phi_i - \phi_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|\phi_k - \phi_l\|^2)^{-1}}. \tag{21}$$

Note that p_{ii} and q_{ii} are set to 0 since t-SNE is only interested in modeling pairwise similarities. Next, t-SNE minimizes the Kullback–Leibler divergence between the two probability distributions P and Q through gradient descent. The Kullback–Leibler divergence is given by:

$$C = KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \tag{22}$$

from which the gradient w.r.t the low-dimensional map can found to be:

$$\frac{\partial C}{\partial \phi_i} = 4 \sum_j (p_{ij} - q_{ij})(1 + \|\phi_i - \phi_j\|^2)^{-1}(\phi_i - \phi_j), \tag{23}$$

which can be used to update the low-dimensional mapping φ from an initial value.

References

Al-Anazi, A., Gates, I.D., 2010. On the capability of support vector machines to classify lithology from well logs. Nat. Resour. Res. 19 (2), 125–139. http://dx.doi.org/10.1007/s11053-010-9118-9.

Arps, J.J., Arps, J.L., 1964. The subsurface telemetry problem-a practical solution. J. Pet. Technol. 16 (05), 487–493. http://dx.doi.org/10.2118/710-PA.

Bourgoyne, Jr., A.T., Young, Jr., F.S., 1974. A multiple regression approach to optimal drilling and abnormal pressure detection. Soc. Pet. Eng. J. 14 (04), 371–384. http://dx.doi.org/10.2118/4238-PA.

Dev, V.A., Eden, M.R., 2019. Formation lithology classification using scalable gradient boosted decision trees. Comput. Chem. Eng. 128, 392–404. http://dx.doi.org/10.1016/j.compchemeng.2019.06.001.

Ditzler, G., Roveri, M., Alippi, C., Polikar, R., 2015. Learning in nonstationary environments: A survey. IEEE Comput. Intell. Mag. 10 (4), 12–25. http://dx.doi.org/10.1109/MCI.2015.2471196.

Dupriest, F.E., Koederitz, W.L., 2005. Maximizing drill rates with real-time surveillance of mechanical specific energy. In: Proceedings of the SPE/IADC Drilling Conference, Amsterdam, Netherlands. Society of Petroleum Engineers, http://dx.doi.org/10.2118/92194-MS, SPE-92194-MS.

Elwell, R., Polikar, R., 2011. Incremental learning of concept drift in nonstationary environments. IEEE Trans. Neural Netw. 22 (10), 1517–1531. http://dx.doi.org/10.1109/TNN.2011.2160459.

Feng, R., Grana, D., Balling, N., 2021a. Uncertainty quantification in fault detection using convolutional neural networks. Geophysics 86 (3), M41–M48. http://dx.doi.org/10.1190/geo2020-0424.1.

Feng, R., Grana, D., Balling, N., 2021b. Variational inference in Bayesian neural network for well log prediction. Geophysics 86 (3), 1–45. http://dx.doi.org/10.1190/geo2020-0609.1.

Gowida, A., Elkatatny, S., Abdulraheem, A., Shehri, D.A., 2020. Synthetic well-log generation: New approach to predict formation bulk density while drilling using neural networks and fuzzy logic. In: Proceedings of the International Petroleum Technology Conference, Dhahran, Kingdom of Saudi Arabia. OnePetro, http://dx.doi.org/10.2523/IPTC-19787-MS, IPTC-19787-MS.

Hayes, T.L., Cahill, N.D., Kanan, C., 2019. Memory efficient experience replay for streaming learning. In: Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada. http://dx.doi.org/10.1109/ICRA.2019.8793982.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile. https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.123.

Jeong, J., Park, E., Emelyanova, I., Pervukhina, M., Esteban, L., Yun, S.-T., 2021. Application of conditional generative model for sonic log estimation considering measurement uncertainty. J. Pet. Sci. Eng. 196, 108028. http://dx.doi.org/10.1016/j.petrol.2020.108028.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations (ICLR), San Diego, California, USA. https://arxiv.org/abs/1412.6980.

McCloskey, M., Cohen, N.J., 1989. Catastrophic interference in connectionist networks: The sequential learning problem. Psychol. Learn. Motiv. 24, 109–165. http://dx.doi.org/10.1016/S0079-7421(08)60536-8.

Osarogiagbon, A.U., Oloruntobi, O., Khan, F., Venkatesan, R., Butt, S., 2020. Gamma ray log generation from drilling parameters using deep learning. J. Pet. Sci. Eng. 195, 107906. http://dx.doi.org/10.1016/j.petrol.2020.107906.

Portney, L.G., 2020. Foundations of Clinical Research: Applications to Evidence-Based Practice, fourth ed. FA Davis Company, Pennsylvania, United States.

Rolon, L., Mohaghegh, S.D., Ameri, S., Gaskari, R., McDaniel, B., 2009. Using artificial neural networks to generate synthetic well logs. J. Nat. Gas Sci. Eng. 1 (4–5), 118–133. http://dx.doi.org/10.1016/j.jngse.2009.08.003.

Schaul, T., Quan, J., Antonoglou, I., Silver, D., 2016. Prioritized experience replay. In: Proceedings of the 4th International Conference on Learning Representations, ICLR, San Juan, Puerto Rico. https://arxiv.org/abs/1511.05952v4.

Tian, M., Omre, H., Xu, H., 2021. Inversion of well logs into lithology classes accounting for spatial dependencies by using hidden markov models and recurrent neural networks. J. Pet. Sci. Eng. 196, 107598. http://dx.doi.org/10.1016/j.petrol.2020.107598.

Tunkiel, A.T., Sui, D., Wiktorski, T., 2021. Training-while-drilling approach to inclination prediction in directional drilling utilizing recurrent neural networks. J. Pet. Sci. Eng. 196, 108128. http://dx.doi.org/10.1016/j.petrol.2020.108128.

Van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9 (11), 2579–2605.

Waughman, R.J., Kenner, J.V., Moore, R.A., 2002. Real-time specific energy monitoring reveals drilling inefficiency and enhances the understanding of when to pull worn PDC bits. In: Proceedings of the IADC/SPE Drilling Conference, Dallas, Texas. Society of Petroleum Engineers, http://dx.doi.org/10.2118/74520-MS, SPE-74520-MS.

Xie, Y., Zhu, C., Zhou, W., Li, Z., Liu, X., Tu, M., 2018. Evaluation of machine learning methods for formation lithology identification: A comparison of tuning processes and model performances. J. Pet. Sci. Eng. 160, 182–193. http://dx.doi.org/10.1016/j.petrol.2017.10.028.

Zeng, L., Ren, W., Shan, L., 2020. Attention-based bidirectional gated recurrent unit neural networks for well logs prediction and lithology identification. Neurocomputing 414, 153–171. http://dx.doi.org/10.1016/j.neucom.2020.07.026.

Zhang, D., Yuntian, C., Jin, M., 2018. Synthetic well logs generation via recurrent neural networks. Pet. Explor. Dev. 45 (4), 629–639. http://dx.doi.org/10.1016/S1876-3804(18)30068-5.