University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: Computer Science	Spring semester, 2018 Open / ~~Restricted access~~
Writer: Fredrik Haugsand	[illegible] (Writer's signature)
Faculty supervisor: Prof. Chunming Rong External supervisor(s): Anton Shchipanov, Senior Research Engineer, International Reasearch Institute of Stavanger	
Thesis title: Testing Different Ways to Analyze Data from Well Sensors Using Neural Networks and Image Processing	
Credits (ECTS): 30	
Key words: Neural Network, Image Processing, Feature Extraction, Data Analysis	Pages: 19 Stavanger, 15/06/2018 Date/year

Testing Different Ways to analyse Data from Well Sensors Using Neural Networks and Image Processing

Fredrik Haugsand, UiS

Abstract— **This thesis focuses on evaluation and testing different approaches of image recognition, neural networks and data analytics with application to the analysis of data collected from sensors installed in wells operated in the oil and gas industry. Simple image recognition algorithms are compared with a newly implemented approach for feature extraction. Two different neural networks are also described and implemented, to compare against the image recognition. Image recognition algorithms had a limited amount of success due to the sample size of images, while neural networks and feature extraction are viable methods to analyse and classify pressure transients.**

I. INTRODUCTION

Using and implementing neural networks in the petroleum industry was widely used in the late 90's and early 00's [15]. Lately, implementing these networks has gained resurgence because of new big data sets accumulated in the industry with the installation of different sensors in oil fields. This resurgence is mirrored across the industry, which makes it important to find a cutting-edge technology, not only to reduce costs, but to improve safety and data collection.

II. BACKGROUND & OBJECTIVES

In this thesis, we will consider synthetic cases simulating an injection well with a Permanent Downhole Pressure Gauge (PDG). The PDG sensors are typically measuring pressure with high accuracy, while well production or injection rate is sometimes measured for a cluster of wells, where a problem with rate allocation between the wells occurs. In this thesis, we will try to address the rate allocation problem in the sense of separating different well pressure transient responses. Such a difference is observed in any long well history and may be caused by (1) superposition effects, where sequential pressure transients cumulative effect from previous changing rate periods, (2) change in well boundary conditions, e.g. interference with a nearby well or approaching a flow barrier, (3) change of well (like skin) and reservoir (like permeability) parameters with time, and many others [14]. The main objective of the thesis is to (1) evaluate and test different pattern recognition approaches to separate pressure transient responses in a well history based on change in "pressure" pattern behavior. (2) Consistency analysis of the well surveillance data as example of pressure and rate measurements. (3) Automate routine procedures of pressure transient analysis.

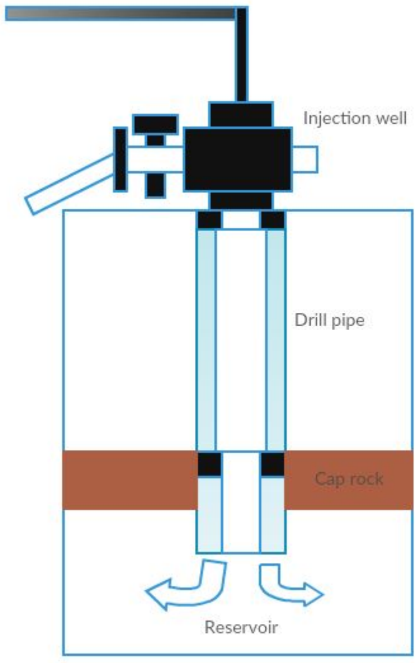

Fig. 1. Injection well [4]

III. RELATED WORK

The methods discussed and implemented in this thesis have been tried in other fields of study, mainly in medical studies and data analysis of exponential data. Some of the methods described in these papers gave inspiration for a similar implementation. A few important papers of note, both in data analytics in oil fields and pattern recognition are listed and shortly described below.

J.R. Parker, University of Calgary, uses curves plotted from data collected on a respirometer to create objects one can analyse using computer vision [17]. Some of the noteworthy algorithms used to classify data in this paper are geometrical classifiers, slope histograms, image moments and outline features. Not all of these classifiers are viable to be implemented into this thesis, but outline features, geometrical classifiers and the method of creating objects are tested and incorporated into this thesis.

Paul L. Rosin, Cardiff University, describes how to calculate geometrical shapes using old and new algorithms (2002) [19]. While it describes and compares algorithms for different geometrical shapes, the algorithms this thesis incorporates is the algorithm to calculate ellipticity. This paper proposes two methods for calculating ellipticity, the classical method and the Euclidian method. This thesis will use the classical method, as the Euclidian method works better on rounder objects.

Karthik Raghupathy, Cornell University, discusses how to detect and analyse curves in images by using a Generalized Radon Transform [18]. This was not implemented into the thesis, but the structured way of analysing images helped in understanding how an image can be processed.

Roland Horne & Chuan Tian, recurrent neural networks(RNN) mainly uses two recurrent neural networks, NARX and a regular RNN [21]. The goals of these networks are to learn how pressure responds as a function of time and rate. They successfully achieve this and is tested on the provided data in this thesis.

Roland Horne & Chuan Tian, machine learning applied to multiwell test analysis and flow rate reconstruction [20]. This paper uses machine learning to reconstruct flow rate both in single-well and multiwell systems. They successfully recreate rate as a function of time and pressure, both with perfect and noisy data.

IV. DATA PRE-PROCESSING

All tables and algorithms are in the appendixes. If a table or algorithm is referenced, seek the appendix. The original data set consisted of two columns, time[Hr] and pressure[PSI] in one continuous reading, as seen in Table IV. Data stored in this fashion is useful for showing how the well operates over a period of time, but does not contain enough information to use in a neural network or image processing efficiently. This section describes how the data is pre-processed into a form easily read by the chosen neural networks. Pressure as a function of time, with changes in rate displayed as vertical lines (figure 2).

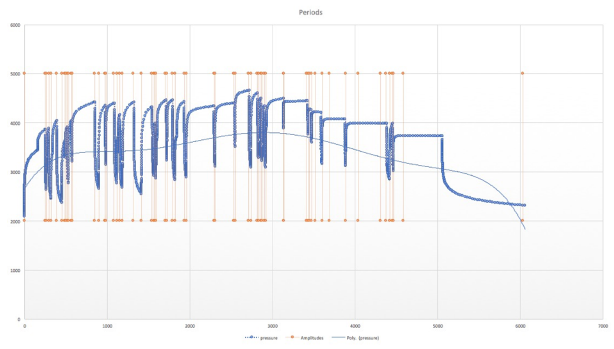

Fig. 2. Syntethic data

To contain data on changes in rate and where a neural network might expect a change to happen, rate[STB/D] is added as a column to the data set. Adding rate to the data allows for further processing, as a change in rate signifies the beginning of a new time window.

For each change of rate, time and pressure are divided into windows based on the rate history, as seen in table V. If the time and pressure window has a constant rate, this is referred to as a pressure transient. For each pressure transient, the last instance of time in the previous window is subtracted from the initial time to create a time window to analyse.

Rate is processed accordingly, with an extra constraint to eliminate unwanted data collection, as shown in Table VI. If the rate of the current window is larger than the previous, the later rate is subtracted from the former. This creates a rate not affected by the previous rate, as leaving the rate in the original fashion would create an artificially high rate. If the rate of the current window is smaller than the former, rate is set to null. Data of rate drops is not relevant for this thesis, as it does not symbolize a pressure transient.

$$p[i] - p@dt[i] = 0 \quad (1)$$

Where i is the specific instance of the pressure in a transient. $p@dt$ is the reference number for that specific transient. For example, the first transient in the data set has 2000 as the reference number, as it is the very first pressure in that specific transient.

Pressure is initially processed the same way as time and rate, but requires another step to normalize the data. Pressure is the one type of data that is normalized against an initial reference point based on the rate. After the last instance of the previous window is subtracted from the first in the current window, the rates of these are compared. If there is a pressure transient present, pressure is required to be normalized. To create this reference point, the initial pressure of the first window is used as a guideline. All later windows of the same pressure transient should have an initial value and similar growth as this window.

The reference point pressure is normalized against is calculated by the division of rate between the current and former rate window. Using this method, two types of pressure transients were observed.

The automated process normalizing this data is dependent on unprocessed time and pressure, and on a list of rate changes. The following image shows the input/output-processing of the data. Algorithm 1 in appendix 2 shows pseudocode for how the data is processed.

$$\Delta p, \Delta p' = d(\Delta dP[i])/d(ln(\Delta dQ[i])) \quad (2)$$

where i is the instance of the pressure in a specific transient. This is the Bourdet derivative, created to find the derivative in pressure [14].

$$dP[i]/dQ[i] \quad (3)$$

Where i is the specific instance of the pressure in a transient. dQ[i] is the corresponding rate for the pressure in a transient. Removal of all nil values in the rate is vital for this operation to work.

A. DATA TYPES ANALYSIS

The previous section mentioned two different pressure transients. This subsection briefly shows and describes the differences between those two.

Figure 3 displays a stereotypical pressure transient of each type. As seen in the image, the slope of each pressure transient has a different reaction over time.

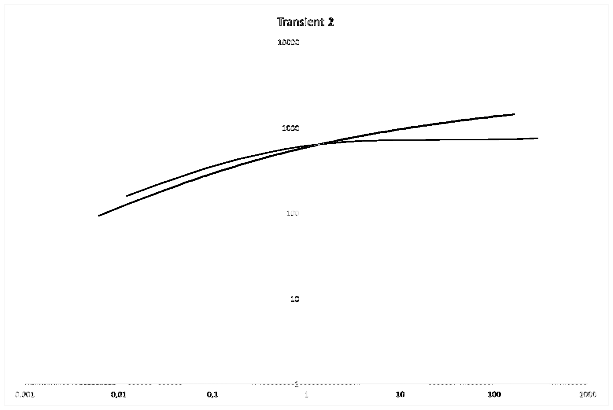

Fig. 3. Pressure transients curve compared

The data set has a total of 94 windows, where 38 are considered not to be of any transient, as rate is set to 0. There are 46 type A and 10 transient type B. Table VII shows the initial window, while table VIII illustrates the first pressure transient of type B, of the pressure transients displayed in figure 3. Looking at the pressure and pressure reference, one can see that when the initial window is the reference point, it has a reference of 1.

The tables and image show that the pressure transients have a different starting point, because the reference number for transients of type B is smaller than type A.

These differences cannot be considered as conclusive differences between the two transients, as there are outliers in both cases. These outliers might make some of the transitions longer or shorter than the average. If a transient is shorter than average, there are fewer data points to measure and visually distinguish between them. The average length of the windows, as well as the amount of windows with a rate smaller than 100, is displayed below. The amount of rate history with a length between 0 and 40 is set as 0 in the calculation of reference points, as this rate is a fluctuation of 0 STB/D.

Function	Result	Unit
Average:	55,9	Hr
Mean:	23,1	Hr
Min:	0,036	Hr
Max:	336,2	Hr
Rate: 0-10:	15	
Rate: 10-20:	6	
Rate: 20-30:	2	
Rate: 30-40:	4	
Rate: 40-:	16	

TABLE I
STATISTICS DATA

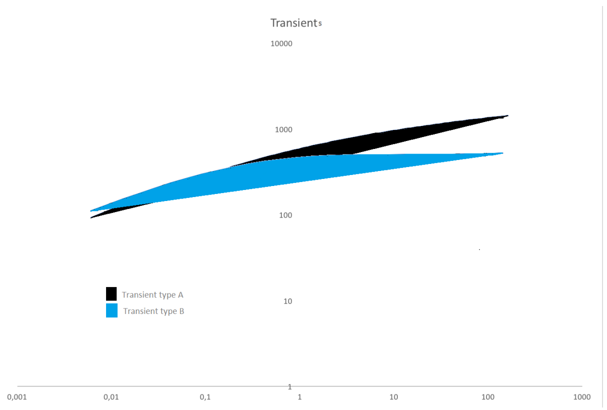

Fig. 4. Pressure transient object compared

B. CONSIDERED TYPES OF SOLUTIONS

- Image processing
- Neural networks
- Data analysis
- Feature extraction

These solutions were considered because between them, it should be possible to classify objects with slight differences. Not all of the solutions are able to distinguish between objects created by pressure transients reliably. Image processing will create and use objects and compare these to each other, to train a model to classify the objects effectively. Neural networks will do the same, but will use the pre-processed data instead of objects. This thesis incorporates one new method; a method based on segmented feature extraction. This will combine feature extraction and data analysis to classify each pressure transient processed.

V. IMAGE PROCESSING

A. Circularity

Circularity is a measure of how closely the shape of an object approaches that of a mathematically perfect circle. Circularity is a function of the objects roundness and is applied in two dimensions, such as the cross-sectional circles along a cylindrical object [19].

$$F_{circ} = \frac{4\pi * A}{P^2} \quad (4)$$

Where P is the perimeter of a circle.

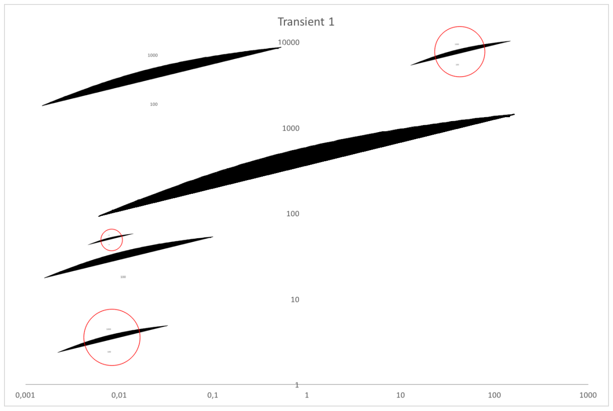

Fig. 5. Circularity detected

Roundness is calculated by the gross features of the shape, rather than the definition of edges and corners [13], [19]. This means that an object can have a poor performance in circularity based on the eccentricity and number of sides. Function 4 shows how circularity is calculated.

Circularity was not a successful classifier for differentiating between the two different pressure transients, as the objects created were too big to calculate circularity on. To calculate circularity, the objects had to be shrunk by 200%. While objects were detectable when small enough, it is not possible to distinguish between pressure transients. Therefore, circularity was not developed further, as the variables needed to create enough variables to distinguish between transients would need to be hard-coded into the classification. This will not allow for any leeway in case of abnormal objects, as seen in function 4.

B. Bag of words, vector space model

Bag of words, or vector space models is a classification method to distinguish between images using multiple small features in a set of images. These features are commonly called words, from a similar method in natural language processing(NLP) [7].

Using an example from NLP to describe the method, a bag of words is a collection of features/words in a document, disregarded for position, grammar or word order. This allows for a machine learning method to be trained on the number of words found in the document. Implemented for image classification, it counts the words over a histogram, to find the most likely class the image belongs to. The model containing the words is in image recognition trained by using a codebook generator.

A codebook generator converts each feature to a vector-representation that maps features to a certain image. This vector representation is made using speeded up robust features(SURF). SURF detects interest points in an image and creates an integer approximation of each feature represented by a word.

$$c^* = argmax_c p(c) \prod_{n=1}^{N} p(w_n \,|c) \quad (5)$$

w = each patch w is a V-dimensional vector that has a single component that equals to one and all other components equal to zero (for k-means clustering setting, the single component equal one indicates the cluster that w belongs to [7]). c = category of the image

To test and implement the bag of words model, it was conducted on three different image sets. These image sets contained images based on proven pressure transients, and were used to optimize the return of data and results.

The three different image sets created were image sets based on the full length of the pressure transient, the average length and the shortest period. These lengths were chosen to see if there were differences in the images created by the pressure transient, based on the objects created by the pressure transients.

The best performing image set was created by converting pressure transients into an object spanning from the curve to the x-axis. This creates a series of objects corresponding to the size of the pressure transient. The objects created are therefore not of a predetermined size, which can lead to underperforming training sessions. To determine the accuracy reliably under these conditions, the model is trained 5 times to find the average accuracy. This is seen in table IX, where the 5 times are listed accordingly.

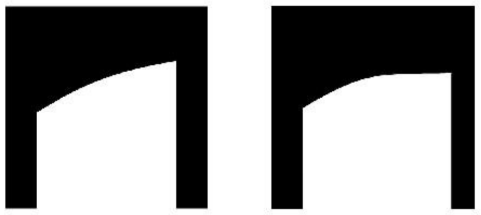

Fig. 6. Full size pressure transient object

The average and short image sets are created by limiting the scale of time while creating the objects. Limiting this scale makes sure that all of the objects in these image sets start and end at the same coordinates. This removes excess, unwanted data from the images, which reduces accuracy. Considering that all of the images are of the same size, it limits the amount of pressure transient objects available in the average set, as not all objects are long enough to create an object that matches the decided coordinates. The image sets containing the full and short objects contain 10 images of each transient. The average image set contains 6 objects of each object. This means that the reliability of the results attained using the image sets containing 10 objects is higher than the set containing 6. This because the accuracy or error of the image set with fewer images will be artificial high in case of mistaken classification. As seen in table X, the accuracy of the bag of words model on the short image sets, the average accuracy over 5 iterations is 0.486. This classification accuracy is expected, as much of the information in the objects is found by comparing the last halves of the objects.

The average image set contains in total 12 images, in which there are 6 images of each pressure transient. Both image sets contain 6 images because the model is trained on an equal number of images from each class. Using this method gives an advantage the short image classifier did not have. These objects contain all of the data available up until the average length of the curve. This means that the model should be able to train the model to a higher degree than the short image model. While there is more data to train the model, it also has a higher fluctuation in accuracy when it classifies discrepancies due to the increased effect each image

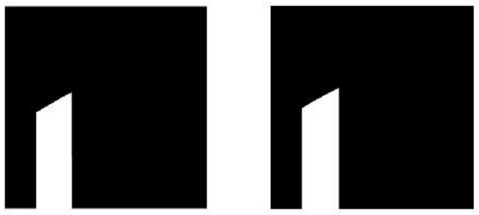

Fig. 7. Limited pressure transient object

has on the classification. The average accuracy for this model is at 0.66, which is high due to the 5 times the model was trained, it correctly classified all of the images once, as seen in table XI.

While this result is good, it is not consistent enough to implement further. If the data set was larger, more pressure transients with this length could be found. This would theoretically even out the accuracy of the model, possibly to an accuracy higher than 0.66. However, since the used data set is not large enough to find more pressure transients of this length to increase the sample size, this method will not be developed further.

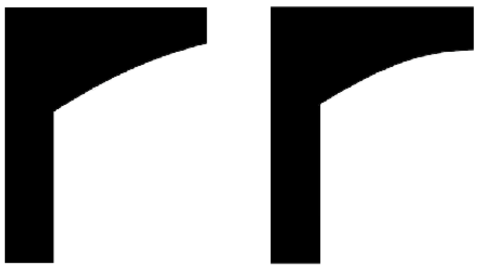

Fig. 8. Average pressure transient object

C. Blob detection

$$g(x,y,t) = \frac{1}{2\pi rt} e^{\frac{x^2+y^2}{2t}} \quad (6)$$

Where t is a scale space representation [8].

In computer vision, blob detection methods are aimed at detecting regions in a digital image that differ in properties, such as brightness or color, compared to surrounding regions. Informally, a blob is a region of an image in which some properties are constant or approximately constant; all the points in a blob can be considered in some sense to be similar to each other. The most common method for blob detection is convolution [8].

Given some property of interest, expressed as a function of position on the image, there are two main classes of blob detectors: (1) differential methods, which are based on derivatives of the function with respect to position, and (2) methods based on local extrema, which are based on finding the local maxima and minima of the function. With the more recent terminology used in the field, these detectors can also be referred to as interest point operators, or alternatively interest region operators.

There are several motivations for studying and developing blob detectors. One main reason is to provide complementary information about regions, which is not obtained from edge detectors or corner detectors. In early work within the area, blob detection was used to obtain regions of interest for further processing. These regions could signal the presence of objects or parts of objects in the image domain with application to object recognition and/or object tracking. In other domains, such as histogram analysis, blob descriptors can also be used for peak detection with application to segmentation. Another common use of blob descriptors is as main primitives for texture analysis and texture recognition. In more recent work, blob descriptors have found increasingly popular use as interest points for wide baseline stereo matching and to signal the presence of informative image features for appearance-based object recognition based on local image statistics. There is also the related notion of ridge detection to signal the presence of elongated objects [8].

In this thesis blob detection is used to train other models, to improve their performance and increase reliability. Bag of words, object recognition and deep learning object classification all use blob detection to rotate and compare the objects. This is most prominent in object recognition, as blob detection is a vital part of comparing the objects in these images.

D. Object recognition

We hoped that using object recognition with blob detection as a method to process the images before comparisons, would differentiate between different pressure transient objects. This was conducted by comparing the edges of an object. The edges are compared to edges in another image to see how similar the objects are. This implementation was not a deep learning model, as it was implemented using deep learning object classification. There are some advantages in not using a deep learning image classificator (1) objects are compared directly to each other, (2) similar objects will get a better score and (3) implementation requires less than a deep learning model.

Comparing these images were done using OpenCV(Open Source Computer Vision). OpenCV is a library with support for real-time object detection and recognition. It is one of the most popular libraries used in computer vision [10].

$$I_1(A,B) = \sum_{i=1...7} \left| (\frac{1}{m_i^A}) - (\frac{1}{m_i^B}) \right| \quad (7)$$

$$I_3(A,B) = \sum_{i=1...7} \frac{\left|(m_i^A) - (m_i^B)\right|}{|m_i^A|}$$

$$m_i^A = sign(h_i^A) * log h_i^A$$

$$m_i^B = sign(h_i^B) * log h_i^B$$

where

$$h_i^B, h_i^A$$

are the Hu moments of A and B, respectively [6].

This function has three different functions to compute similarity. Function 7 shows how the Hu moments are calculated for the two most reliable functions. A Hu moment is a weighted average of the image pixels intensity, to describe a method after segmentation. The performances of these functions are listed in Table XIII, whereas a lower number means that the image is similar. The performance of CV_I1(I1) is the highest of the three functions, but it shares the same limitation as the other two functions. While CV_I1 can recognize objects of the same pressure transient, it will also wrongfully recognize the other pressure transient as the same. This error is most prominent in this function, as the other two functions struggle to recognize pressure transients of the same type. This error is less predominate using CV_I3, but the rate of error using CV_I1 to find similar pressure transients is also higher. This error is most likely due to the overlap between the different types of transient being so large, as seen in figure 9.

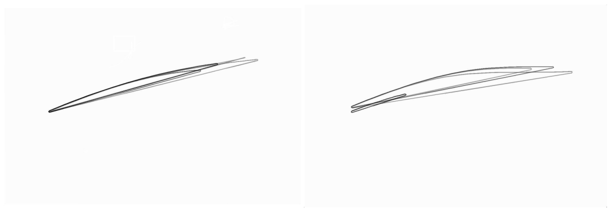
Fig. 9. Overlap pressure transients

This rate of error means that while it is possible to recognize objects of the same pressure transients, it is not reliable enough for further development, when deep learning object recognition libraries can be implemented.

E. Deep learning object classification

Deep learning object classification uses a large pool of negative images with a pre-trained convolutional neural network(CNN) to extract features from images. This feature extraction can be compared to a bag of words model, but with more advanced classifiers. The model is trained against the pool of negative images to increase accuracy. If the amount of images increases, the accuracy should theoretically also increase. To ease the training curve and increase accuracy, deep learning object classification uses a learning framework called ResNet(Deep residual learning for image recognition). ResNet increases the references to the layer inputs, which has been tested to work with an error rate of 0.0357 on ImageNets training set. This training set contains 14.1 million images over 21814 categories.

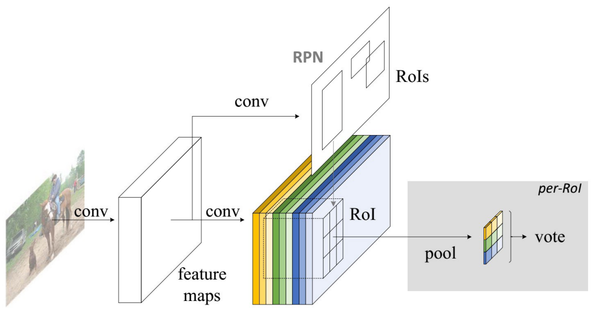

Fig. 10. Deep learning object classification [23]

By importing the three image sets compared in the bag of words section into the deep learning object classifiers, the image sets are trained against each other.

After the model is trained, it uses select images from each of the image sets to test the classifier. The accuracy of this classifier is interesting, as it has a limited amount of images to train against. Table XIV shows the results of the deep learning object classificator [23]. The image set with the best performance was as in the bag of words model, the image set with full sized pressure transients. The average accuracy over 5 iterations is 0.642, which means that it is possible to use object classification to some degree. However, the accuracy of this model is concentrated on pressure transient type A. It classifies pressure transients of type A with 0.8571 efficiency, and pressure transient type B with 0.4285. This incline in accuracy towards transient type A is not expected, but is intelligible as the images in transient type A has a consistent high length. The accuracy for the short and average image set is 0.5, which means that deep learning object classification cannot be used on these image sets.

VI. Neural Network

For the scope of testing and implementing neural networks and comparing them against the methods previously discussed, the networks were implemented and tested in Matlab. This was opted into, because it features a series of neural networks ready to use. Other options than Matlab were Tensorflow and Keras, where Keras is better for time series as Tensorflow converts all input data to tensors. Matlab also offers features to optimize an NN based on the available data. While this will not specify how many hidden layers and neurons the NN has, it will quickly find a network with a close approximation to the best available response.

Therefore, this section has two parts. The first part is where the tool is used on the image set. The second part is where different types of RNNs are used to classify and predict processed data. While these two parts are based on the same original data, they cannot be compared with each other, as the usage is fundamentally different.

A. Neural network classifier

The Matlab neural network classifier lets you insert a data set and find the best match [1]. This was conducted on a set of data composed of 10 images of transient type A and B. Using this data allows for a direct comparison against image recognition and feature extraction. By running this classificator, there are two main issues you have to consider. The first being that you have to make sure the data is a good fit for the classification before you start, as this classificator will not see errors in the data set. The second is how deep you want to run the comparison, while a light comparison between the neural networks will give a rough guide to the end result, an in-depth comparison might find a network well suited for the data.

As one can see in the figure below, the best classification network is a coarse tree. This means that the optimal solution is a neural decision tree [12]. A neural decision tree is a symbiotic melt between a neural network and a decision tree.

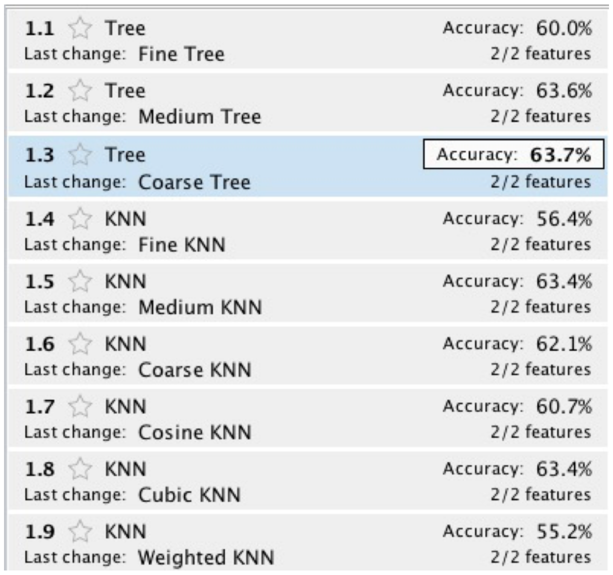
1.1 Tree — Last change: Fine Tree — Accuracy: 60.0% — 2/2 features
1.2 Tree — Last change: Medium Tree — Accuracy: 63.6% — 2/2 features
1.3 Tree — Last change: Coarse Tree — Accuracy: 63.7% — 2/2 features
1.4 KNN — Last change: Fine KNN — Accuracy: 56.4% — 2/2 features
1.5 KNN — Last change: Medium KNN — Accuracy: 63.4% — 2/2 features
1.6 KNN — Last change: Coarse KNN — Accuracy: 62.1% — 2/2 features
1.7 KNN — Last change: Cosine KNN — Accuracy: 60.7% — 2/2 features
1.8 KNN — Last change: Cubic KNN — Accuracy: 63.4% — 2/2 features
1.9 KNN — Last change: Weighted KNN — Accuracy: 55.2% — 2/2 features

Fig. 11. Neural network classifier

Comparing this to the optimal solution in image recognition, it is slightly more reliable. This means that a solution working on the data behind the objects will provide better results than a pure image-based classificator. The following images show the data represented as a function of pressure and time and the confusion matrix of the neural decision tree. The confusion matrix should show a perfect green line in the diagonal, but since it is not continuous, the rate of false negatives are too numerous.

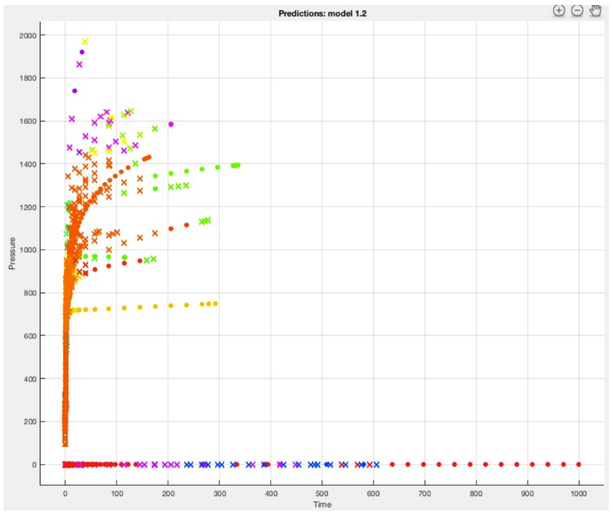

Fig. 12. Pressure transients

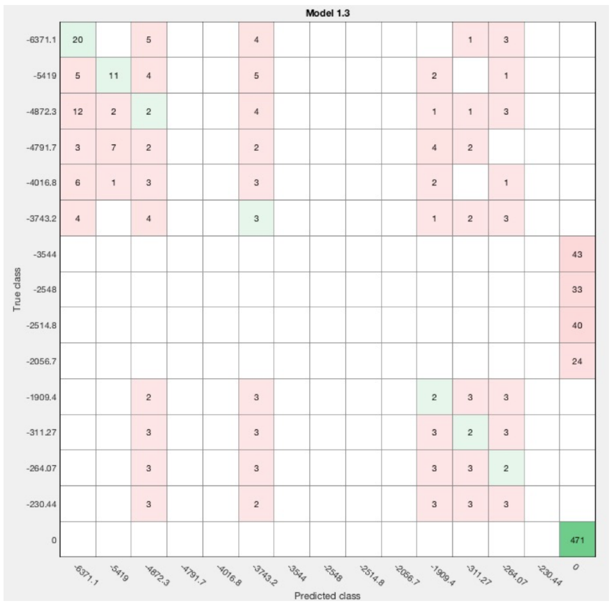

Fig. 13. Confusion matrix for an coarse tree

B. Recurrent Neural network

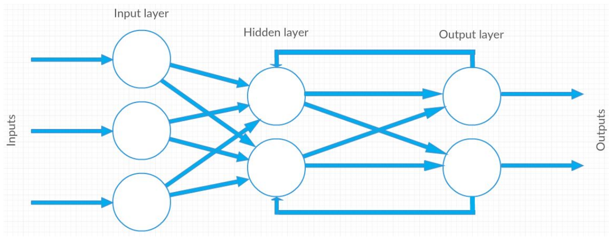

Fig. 14. Recurrent neural network [5]

Neural networks can be sorted into two different categories. These categories are feed-forward and feedback neural networks. The difference between them is how they behave in comparison to each other. Feed-forward can only send data forward through the network, from input to output. This means that there are no loops in the network. For a feed-forward network to learn, they require a series of networks to send the data forward into the chain [3].
A feedback (recurrent) network can send data both ways through the network by using local loops. Feedback networks are usually more complicated than feed-forward, as the setup is more advanced. Feedback networks can save a certain amount of data from one state to the other, which allows them to memorize the previous states. This is why feedback networks are used in translation software.

In this thesis, the used recurrent neural networks are multilayer perceptron (MLP) and nonlinear autoregressive neural network with external input (NARX). These types of networks were chosen because it gives the opportunity to compare the most common recurrent neural network, with a network designed for time series prediction.

Multilayer perceptron(MLP)
This is one of the most regular ANN in use today, as it is easy to expand and add new layers. MLP consists of at least three layers with nodes; input, hidden and output. MLPs are usually used in fitness approximation, as they have a strong ability to solve problems stochastically. Some examples are image and voice recognition. The previously discussed MNIST database often uses MLP [24].

Time delay neural network(TDNN)
TDNN is specifically made to classify patterns where time is a vital component of the data. An example of this is voice recognition with underlying phonetic features. TDNN is a backpropagational network [11].

Nonlinear autoregressive neural network with external input(NARX)
NARX is a dynamical neural architecture used for input-output modeling of nonlinear dynamical systems. When applied to time series prediction, the NARX network is designed as a TDNN, without the feedback loop of delayed outputs [22] [16].

$$y(t) = f(x(t-1), ..., x(t-d), y(t-1), ..., y(t-d)) \quad (8)$$

where d are past values of y(t), given another series x(t).

The classical MLP neural network was tested by two different inputs, to see whether there was a correlation between all three of the factors in the data. The input-output scheme of the first test was time, pressure as input and rate as output. The test was run to investigate if there was a direct correlation between rate and pressure. If there is a direct correlation between these variables, one can use this to create a method to return one variable to recreate rate based on the input. The plot of the fit for time-pressure /rate shows that there is no direct correlation between these data. If there was, the black dots would cross the plots in a diagonal manner.

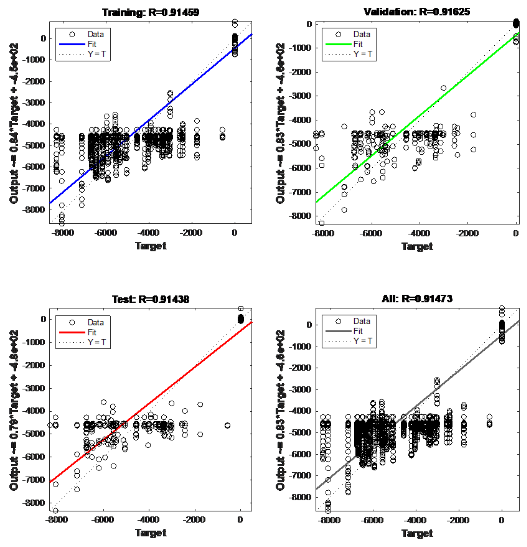

Fig. 15. MLP fit as a function of rate

The second MLP test had the most concise results with time and rate as input and pressure as output. The results show that even though the training is not perfect, it might be due to many different factors such as lack of data, not optimized performance of the neural net or a bad batch of segmented data. The following shows a correlation between the rate as a function of time against pressure. This was expected, as the rate is constant for each change in pressure.

This method requires the batch size to be so large that it at least contains the change in rate, so that the network can learn from the previous period.

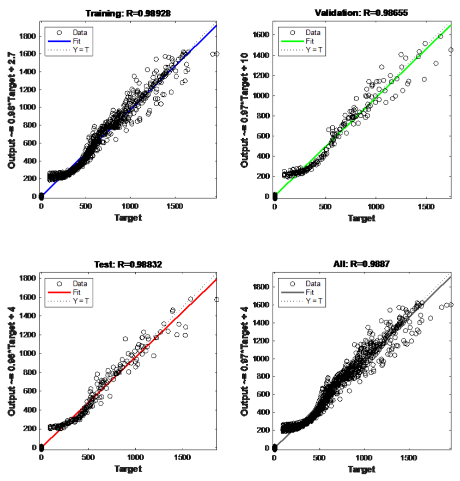

Fig. 16. MLP fit as a function of pressure

While the results with time-rate as input showed a certain correlation, it is not the optimal approach for a dynamic time series. The optimal approach for this solution is a network that trains the network over time. Some of these networks were mentioned in the neural network description, but the network used and tested here is the NARX network.

The graph below shows the output and targets as a function of time, as well as the rate of error. Seeing the output and targets against each other is interesting, but the graph of note is the one containing the error. This error spikes every time there is a shift in rate. This is not necessarily a negative, as it shows room for improvement. This particular network was tested 5 times for every change in hidden layers and time delays to ensure reliability.

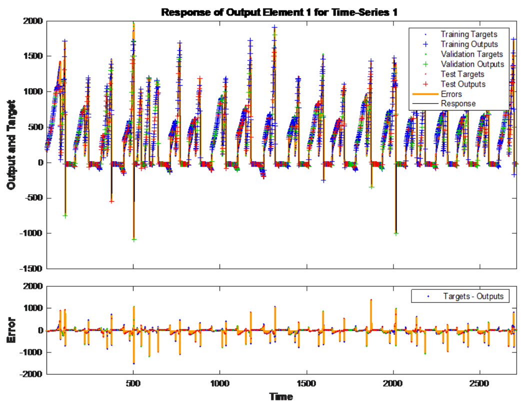

Fig. 17. NARX response and error

Table XIV shows the average mean square error(MSE) and regression for the data set proven most optimal in the MLP network. The table shows the MSE and R as a function of the amount of layers used. The solution with the least amount of error was 20 hidden layers. Even though this MSE seems extremely high, this is the total MSE for testing, training and validation. The lower the MSE, the better the solution. Considering that it is so high, the solution is not optimal. This could be a consequence of the data set not being large enough, or the training function not being optimized for this data.

Layers&Delay	MSE	R
10:2	12810,46411	9,556634
20:2	12674,86273	9,555272
30:2	13086,3958	9,545762
40:2	14401,4155	9,501542

TABLE II

LAYERS, DELAYS AND MSE

VII. FEATURE EXTRACTION

The final method showed the most promising results.

Feature extraction involves reducing the amount of resources required to describe a large set of data. When conducting analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power.It may also cause a classification algorithm to overfit training samples and generalize poorly to new samples. Feature extraction is a general term for methods constructing combinations of variables to get around these problems while still describing the data with sufficient accuracy. Many machine learning practitioners believe that properly optimized feature extraction is the key to effective model construction [9].

Feature extraction is an area that can be expanded into as many subparts as there is data for. Our approach to feature extraction is unique in the way it processes the data and in the computations done to compare the transients. While feature extraction has been used in oil well analysis before, it has not to our knowledge been used in this specific way. Our method differs from existing methods by separating the data into segments. As previously mentioned in the data processing part, the amount of data as a function of time is decreasing from the first logging. This means that when the data is segmented, it has to preserve the data and even out the number of data points per segment. If the initial segment has 50 points of data, and the later segments have 10, this will be a possible point of error and improvement. While the amount of data has to be segmented into gradually equal parts, it also needs enough points of data to get a useful reading of the data. If the initial segment is divided too far, negative consequences by the curve reading as linear instead of exponential will show.

After analysing the data, the segmentation was decided to happen at 7 different points. The following table shows where this division is, as well as how many of the transients have a maximum length inside that period of time.

Points	amount
0-5:	13
5-10:	3
10-50:	9
50-100:	5
100-200:	6
200-:	6

TABLE III

DATA SEGMENTATION POINTS

After this division, we have a series of calculations run on each segment. These calculations show the main point of expansion and whether the method should be extended at a later stage. The initial thought is to make each function as simple as possible, to extract the essence of the transient curve. The functions used in this thesis are length, degree and slope.

$$Length = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1}^{2)}$$

$$Slope = \frac{(y_2 - y_1)}{(x_2 - x_1)}$$

$$Degree = tan^{-1}\left(\frac{(y_2 - y_1)}{(x_2 - x_1)}\right), \quad (9)$$

where x1 and y1 are the initial coordinates of the curve, respectively to the axis's.

Using these functions on all of the segments, we collect the data and compare the results against the other corresponding segments of the other transients. While the data is not conclusive for one of the segments, the analysis can distinguish between curves if there is sufficient data.

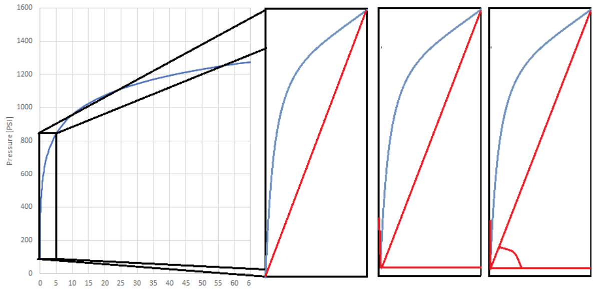

Fig. 18. Feature extraction on Segmentation

For each segment and later each transient, the results of the functions are weighted against a predetermined limit for division between the different transients. This weighting is how the final result is decided for each of the total transients processed. To make this weight fair in terms of how the weights are added, it is calculated by an average of the difference between each transient used on the initial processing.
To find these limits, 10 transients of each type was processed manually, to find the length, degree and slope of each segment. This processing is also where the average total length as a function of time was disclosed.

Algorithm 2 in Appendix II shows how the program sorts through the data. The algorithm describes how the data is processed against the weights and how the data is read in. Pre-processed is not included in this algorithm, as it was described in section 3.

The following figures show the differences between the two main transient types as a function of time and the chosen variables. One notices that the major differences is not how the length changes from segment to segment between the two types. While length can be a deciding factor for the weighting of a transient, it will in most cases be a neutral factor. The deciding factors for distributing the weights across the transients are degree and slope, as these functions have the largest characteristic contrasts. One can argue that since degree is a function of the arctan of the slope, it will always be different, but the scope of the different changes is larger on the arctan than slope.

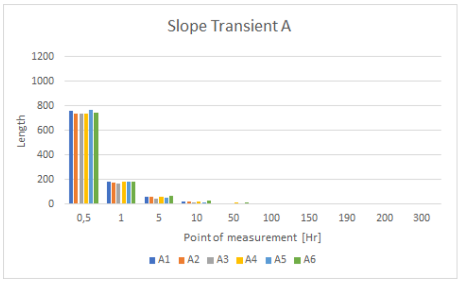

Fig. 19. Slope trasient A

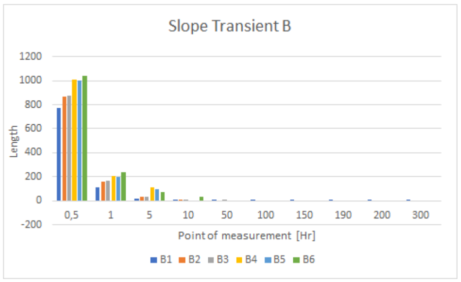

Fig. 20. Slope trasient B

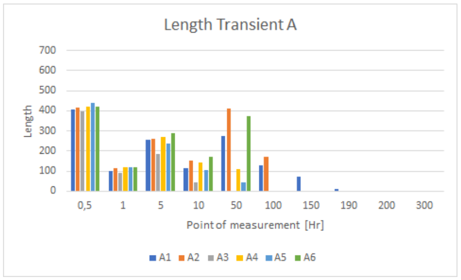

Fig. 21. Length transient A

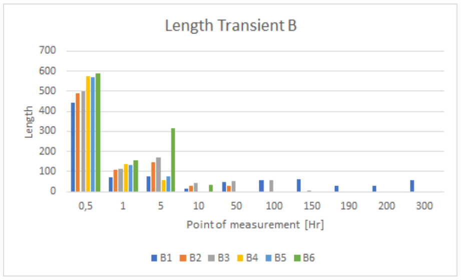

Fig. 22. Length transient B

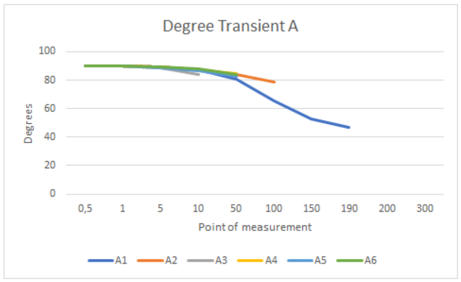

Fig. 23. Degree transient A

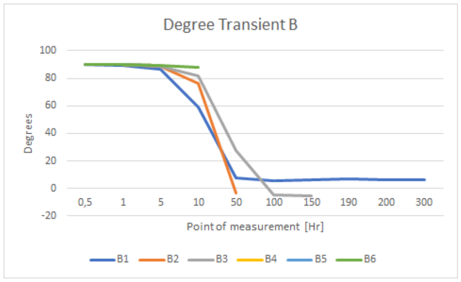

Fig. 24. Degree transient B

VIII. Results and Discussion

Throughout the thesis, the results of each section have been discussed shortly, but the methods have not been compared. To summarize the results of each method, four methods of image recognition were used. Deep learning was the most successful method. Deep learning had an average accuracy of 0.637 for the best method. While this result shows that it is possible to distinguish between the transients in using, it is not promising enough to warrant further development. Especially since the tool used to achieve this result runs the data on a variety of networks to find the best fit.
Testing with neural networks was limited, but the results displayed are promising to a certain degree. While a neural network can be promising, the conditions for effectivity is not an issue this thesis has tried to find. Therefore, while one can claim that the results can be tested more effectively, this span of development belongs to another project.
Feature extraction in collaboration with segmentation is the newly developed concept that has not been tested on extended data sets. The data set used to develop the concept works reasonably well if the length of the transient is higher than 10 hours. After 10 hours the clear differences between the two transients become more distinguished and feature extraction performs well.

Overall the thesis has gone well, with multiple initial goals, in which most have been fulfilled.

Code used/developed is published on Github: https://github.com/Fhaug/masterthesis [2].

IX. Further Work

With the results displayed in this thesis, it comes as a natural extension that there could be multiple new projects with this thesis as a base. The most logical follow up to this thesis is developing the program used to clean the data and check which type of transient the current period contains to make it versatile enough to receive data not pre-processed to a certain degree. This will naturally improve a neural network, since it will make more data available for an eventual deep learning approach.

Improving feature extraction can also be a viable project, if it is extended to a degree in which the results are improved by finding more basic functions one can use.
On an academic note, the thesis can also be used as a base for a paper in an oil conference, since ANN and predicting response has had a recent resurgence in popularity.

Acknowledgment

Chunming Rong, main counsellor and a source of inspiration.
Anton Shcipanov, for giving me an opportunity to make this thesis a reality and always being available.
Cristina Hegdehus, counsellor and helping with everything from venting ideas to showing how an NN works.
Antorweep Chakravorty, for the weekly meetings.

List of Figures

LIST OF TABLES

APPENDIX I
TERMS AND EXPLANATIONS

NN: Neural Network
ML: Machine Learning
TF: TensorFlow
ANN: Artificial Neural Network
PBU: Pressure Build Up
GRNN: Generalized Regression Neural Network
TDNN: Time Delay Neural Network
MLP: Multilayer Perceptron
LSTM: Long Short-Term Memory
NARX: Nonlinear autoregressive neural network with external input
Window: One specific type of PBU
Period: A general PBU
Transient: A period of constant rate
SURF: Speeded Up Robust Features
RESNET: Deep Residual Learning for Image Recognition
BOW: Bag of words
MSE: Mean Square Error
R: Regression

APPENDIX II
TABLES

Time	Pressure
0	2000
0,006058868	2092,738568
0,012117736	2125,95201
0,018176604	2149,77305
0,024235472	2168,851204
0,03029434	2184,962183
0,036353208	2199,005331
...	...
158,3387793	3429,98346
162,9916663	3434,93584
162,9977252	3459,30313
163,0037841	3468,03412
163,0098429	3474,29788
163,0183459	3481,10295
163,0308266	3488,81815

TABLE IV
ORIGINAL SYNTHETIC DATA

Time	Pressure	Rate
0	2000	-6742,395145
0,006058868	2092,738568	-6742,395145
0,012117736	2125,95201	-6742,395145
0,018176604	2149,77305	-6742,395145
0,024235472	2168,851204	-6742,395145
0,03029434	2184,962183	-6742,395145
0,036353208	2199,005331	-6742,395145
...	...	...
158,3387793	3429,98346	-6742,395145
162,9916663	3434,93584	-6742,395145
162,9977252	3459,30313	-8513.514103
163,0037841	3468,03412	-8513.514103
163,0098429	3474,29788	-8513.514103
163,0183459	3481,10295	-8513.514103
163,0308266	3488,81815	-8513.514103

TABLE V

DATA WITH RATE

Time	Pressure	Pressure reference
0,006058868	116,2940324	0,746117418
0,012117736	157,2203992	0,746117418
0,018176604	186,2593974	0,746117418
0,026679606	217,451352	0,746117418
...	...	...
235,8400236	742,1599869	0,746117418
266,1343638	745,6384904	0,746117418
279,1921819	747,1229674	0,746117418
292,25	748,5980009	0,746117418

TABLE VIII

DELTA PRESSURE TRANSIENT B

Time	Pressure	Rate
0,006058868	2092,738568	-6742,395145
0,012117736	2125,95201	-6742,395145
...	...	...
153.685892337064	1420.99252766959	-6742.39514525
158.338779362954	1426.078613439	-6742.39514525
162.991666388843	1431.02508116768	-6742.39514525
-	-	-
0.00605886803802491	92.4820093068201	-1771.11895846
0.0121177360750266	125.625192445101	-1771.11895846
0.0181766041130231	149.429059372595	-1771.11895846
0.026679606204027	175.245055402413	-1771.11895846

TABLE VI

TIME, PRESSURE AND RATE WINDOWS

Time	Pressure	Pressure reference
0,006058868	92,73856774	1
0,012117736	125,9520098	1
0,018176604	149,7730505	1
0,024235472	168,851204	1
...	...	...
136,9726957	1405,201218	1
153,6858923	1424,883655	1
158,3387794	1429,983461	1
162,9916664	1434,935841	1

TABLE VII

DELTA PRESSURE TRANSIENT A

Full image		
Known	AS	BS
AS	0.00	1.00
BS	0.57	0.43
average	0.21	
AS	0.71	0.29
BS	0.29	0.71
average	0.71	
AS	0.86	0.14
BS	0.71	0.29
average	0.57	
AS	0.14	0.86
BS	0.29	0.71
average	0.57	
AS	0.71	0.29
BS	0.43	0.57
average	0.57	
Total average	0.512	

TABLE IX

FULL TRANSIENT

Cut image		
Known	AS	BS
AS	0.57	0.43
BS	0.71	0.29
average	0.43	
AS	0.00	1.00
BS	0.14	0.86
average	0.43	
AS	0.71	0.29
BS	0.57	0.43
average	0.57	
AS	0.14	0.86
BS	0.29	0.71
average	0.57	
AS	0.71	0.29
BS	0.86	0.14
average	0.43	
Total average	0.486	

TABLE X

CUT/SHORT TRANSIENT

TABLE XII

OBJECT RECOGNITION

Method	AB	A2B	A3B	AB2
CV_l1	0.0049	0.0374	0.1069	0.0599
CV_l2	0.0140	0.1271	0.4073	0.2292
CV_l3	0.0051	0.0364	0.1058	0.0554
	AA	AA2	AA3	AA4
CV_l1	0.0	0.0325	0.1019	0.0666
CV_l2	0.0	0.1130	0.3932	0.2461
CV_l3	0.0	0.0301	0.0910	0.0600
	BB	BB2	BB3	BB4
CV_l1	0.0	0.0649	0.0290	0.3424
CV_l2	0.0	0.2433	0.1023	1.7111
CV_l3	0.0	0.0580	0.0262	0.3033

TABLE XIII

OBJECT RECOGNITION

Average transient		
Known	AS	BS
AS	1.00	0.00
BS	1.00	0.00
average	0.5	
AS	1.00	0.00
BS	0.33	0.67
average	0.83	
AS	1.00	0.00
BS	1.00	0.00
average	0.50	
AS	1.00	0.00
BS	0.00	1.00
average	1.00	
AS	1.00	0.00
BS	1.00	0.00
average	0.50	
Total average	0.66	

TABLE XI

AVERAGE TRANSIENT

Deep learning object classificator results		
Known	AS	BS
Original AS	0.8571	0.1428
Original BS	0.5714	0.4285
average	0.642	
Cut AS	0.00	1.00
Cut BS	0.00	1.00
average	0.5	
Average AS	0.00	1.00
Average BS	0.00	1.00
average	0.5	
Total average	0.547	

TABLE XIV

DEEP LEARNING OBJECT CLASSIFICATION RESULTS

Appendix III
Algorithms

```
Algorithm 1: Data processing
  Result: Data processed
  Load data into List;
  while While transient segment do
      initialTime = 0;
      for i in length(data) do
          c
      end
      ompare time against time where rate changes;
      if time equals rate change then
          initialTime = time;
          listTime.append (time[i] - initialTime) ;
          listPressure.append
            (pressure[i]-2000)/(rate[i]/rate[-1]));
      else
          listTime.append (time[i] - initialTime) ;
          listPressure.append
            (pressure[i]-2000)/(rate[i]/rate[-1]));
      end
  end
  for i in length(listTime) do
      p
  end
  rint(listTime[i], listPressure[i], listRate[i]) ;
```

```
Algorithm 2: Segmentation algorithm
  Result: Weighting of segments
  initialization;
  while While transient segment do
      initialize weight;
      initialize T(transient variable);
      if Length longer than X then
          Weight +=1;
      else
          Weight -= 1;
      end
      if Slope higher than Y then
          Weight +=1;
      else
          Weight -=1;
      end
      if Degree lower than Z then
          Weight +=1;
      else
          Weight -=1;
      end
  end
  if weight >=T then
      Transient = A;
  else
      Transient = B;
  end
```

References

[1] Feature extraction.

[2] Fredrik haugsands github repository.

[3] Neural network toolbox.

[4] Oil well image.

[5] Recurrent neural network image.

[6] Structural analysis and shape descriptors.

[7] Bag-of-words model in computer vision, Jun 2018.

[8] Blob detection, Jun 2018.

[9] Feature extraction, Jun 2018.

[10] Opencv, Jun 2018.

[11] Time delay neural network, May 2018.

[12] Balestriero and Randall. Neural decision trees, Mar 2017.

[13] Ariel Balter. How to calculate circularity, Apr 2017.

[14] Dominique Bourdet. *Well test analysis the use of advanced interpretation models*. Elsevier, 2002.

[15] A.o. Kumoluyi. Higher-order neural networks in petroleum engineering. *SPE Western Regional Meeting*, 1994.

[16] Jos Maria P. Menezes and Guilherme A. Barreto. Long-term time series prediction with the narx network: An empirical evaluation. *Neurocomputing*, 71(16-18):33353343, 2008.

[17] J.r. Parker. Scientific curve classification by combining simple algorithms. *Proceedings First IEEE International Conference on Cognitive Informatics*.

[18] Karthik Raghupathy. Curve tracing and curve detection in images. 06 2018.

[19] Paul L. Rosin and Jovia Uni. 2d shape measures for computer vision. *Handbook of Applied Algorithms*, page 347371.

[20] Chuan Tian and Roland N. Horne. Machine learning applied to multiwell test analysis and flow rate reconstruction. *SPE Annual Technical Conference and Exhibition*, 2015.

[21] Chuan Tian and Roland N. Horne. Recurrent neural networks for permanent downhole gauge data analysis. *SPE Annual Technical Conference and Exhibition*, 2017.

[22] Hang Xie, Hao Tang, and Yu-He Liao. Time series prediction based on narx neural networks: An advanced approach. *2009 International Conference on Machine Learning and Cybernetics*, 2009.

[23] Joyce Xu. Deep learning for object detection: A comprehensive review, Sep 2017.

[24] Zhang, Ren, Sun, and Jian. Deep residual learning for image recognition, Dec 2015.