CS230

Generating Synthetic Well Logs from Real-time Drilling Data

Mehrdad Yousefzadeh
mehrdady@stanford.edu

Obai Shaikh
shaikhon@stanford.edu

Rayan Kanfar
kanfar@stanford.edu

Abstract

The objective of this project is to predict subsurface rock properties in wells from real-time drilling data. Rock properties such as porosity, density, stiffness, saturation, and permeability can be calculated or measured from wireline logs and are of paramount importance in subsurface resource estimation and exploitation. Drilling information are available in every well while wireline petrophysical measurements are expensive and scarce, especially in delineation, production and injection wells. Drilling parameters are not directly correlated to wireline logs. This problem is non-linear and the performance of three sequence based learning algorithms are investigated, namely, Inception Convolutional Neural Networks, Clock Work Recurrent Neural Networks, and Long-Short Term Memory. The first two models seem to be more predictive while LSTM suffers from over-fitting to the mean of the data.

1 Introduction

Drilling parameters are available in every drilled well while rock properties are acquired using expensive wireline logging tools inserted into the well, which is not available at all depths (the annual capital spent on wireline logging is more than $2.6 bilions). For example, in exploratory wells, more data is acquired at the well location as opposed to a production wells. The objective of this project is to predict rock properties while drilling to help management make real time decisions about borehole measurements.

Rock properties such as porosity, permeability, saturation, density, etc. are of paramount importance for subsurface resource assessment and exploitation. Examples of wireline logs of interest include neutron porosity, resistivity, gamma ray, density, compressional wave velocity, etc. Real-time drilling information (i.e. rate of penetration, torque, weight on drill bit, etc.) reflect the responses of drilling subsurface rock layers and are correlated to the bulk properties of rocks. Equinor, previously known as Statoil, made data from the North Sea oil fields available for research in June, 2018. The Volve field dataset include wireline logs, mud logs, production data, seismic volumes, and more. Given real-time drilling information, we wish to predict rock properties at wells as if it had been measured by wireline logging tools.

We explore three sequence based learning algorithms, specifically, Inception convolutional neural networks, clockwork recurrent networks, and long-short term memory. Such a feature prediction model offers an attractive solution for production and delineation wells, where the acquisition of data is limited.

CS230: Deep Learning, Winter 2019, Stanford University, CA.

2 Related work

We found some existing papers and projects focusing on the wireline log generation and using drilling data to investigate the lithology of the subsurface. The most relevant papers can be categorized in two sets:

1. Predicting wireline logs using wireline logs. In [ZYJ18] the authors used an LSTM network to predict some sections of the logs based on other logs. Fully connected networks have been used in [PMBH18, BBSNP13] to preform the same task. General regression neural networks are used in the work by [RMA$^+$09] to predict logs.

2. Identification of the subsurface lithology using the drilling data has been investigated by [MH15]. They used a basic fully connected network and classified the lithology of different layers based on drilling data and some wireline logs.

To the best of our knowledge, there is no work or publication on predicting logs from drilling data.

3 Dataset and Features

We have a total of 24 logged wells. Each well consists of real-time drilling data and wireline logs. Six drilling logs are selected for training (Depth, rate of penetration (ROP), weight on bit (WOB), revolutions per minute (RPM), torque, and Mechanical Specific Energy (MSE)). The corresponding logs containing the y labels we wish to predict are gamma ray, density, and compressional velocity. The main data processing steps were:

1. To convert logs of interest from DLIS to readable LAS ascii format
2. Merge logs from separate files into one matrix, 3) Clean the logs from null values
3. Interpolate bad and missing data
4. Normalization
5. Data augmentation was carried out by extracting subsets from the logs with a specific window size and stride. Each subset is considered as a training example. This implementation preserves the relative depth relation and provides different snapshots of each well log. By the end of this process, we generated 29040 training examples from 17 well logs using a window length of 32, and stride of 4.

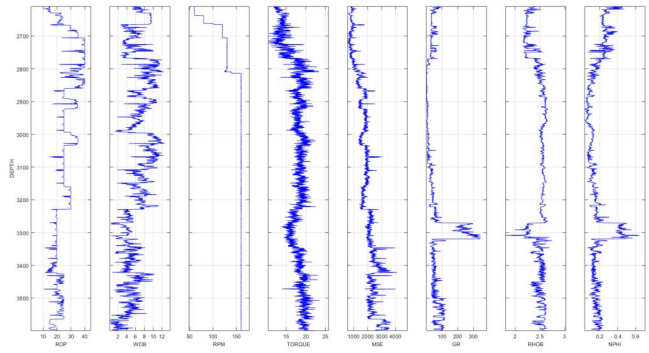

Figure 1: Example of drilling data and well logs for one well

4 Methods

For this project we investigated several network models, namely, CNN-Inception model, recurrent neural networks and its variations and finally a specific type of RNN called Clockwork-RNN [KGGS14]. The RNN and its variations (LSTM, GRU and Bi-LSTM) were not predictive for this

project. They mainly suffered from sticking to the mean of the output data, therefore, we didn't include them to this report. However, the codes and results are available in our GitHub repository [YKS19]. In the following we explain two other methods.

4.1 Inception Convolution Neural Network

The inception model is a convolutional neural network that includes layers with stacked convolutional output volumes. This is achieved by padding the input such that the convolved output has the same dimension as its input, no matter what filter size is used. The original purpose of this model is to have deep networks that avoid vanishing gradients by having wide layers. Theoretically, this model is useful in our problem because we wish to capture patterns at different scales. Geologic features have multiscale correlations. The cost function used in learning is the mean square error.

We use 1D convolutions in our model. In other words, the sequences of different data types such as rate of penetration, weight on bit, torque, etc, are set as channels in the inputs. We do 1D convolutions because although "valid" convolutions of 1D and 2D inputs are the same, "same" convolutions produce different results. While "same" 1D convolutions pad only the inputs only vertically, "same" 2D convolutions pad around the whole input image producing more expensive computations and more bias results.

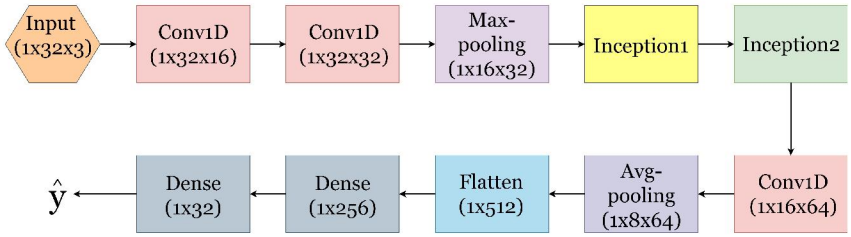

Figure 2: CNN architecture

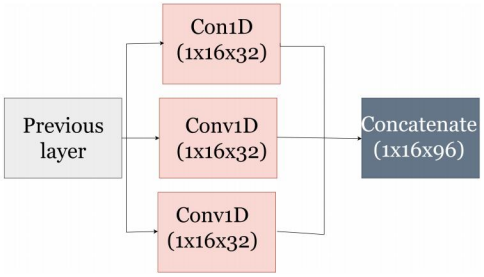

Figure 3: Inception layer architecture

4.2 Clockwork Recurrent Neural Network

Clockwork RNN (CW-RNN) like Siamese Recurrent Networks (SRNs) consist of input, hidden and output layers. There are forward connections from the input to hidden layer, and from the hidden layer to the output layer. However, the neurons in the hidden layer are partitioned into k modules. A clocking period $T_n \in \{T_1, \cdots, T_k\}$ is assigned to each module. Each module in fully interconnected internally, but recurrent connections between module j to module i are established only if period T_i is smaller than T_j.

At time step t, the RNN output, $y_O^{(t)}$, is calculated using the following formulae:

$$y_H^{(t)} = f_H(W_H \cdot y^{(t-1)+W_I \cdot x^{(t)}}) \quad (1)$$

$$y_O^{(t)} = f_O(W_O \cdot y_H^{(t)}) \quad (2)$$

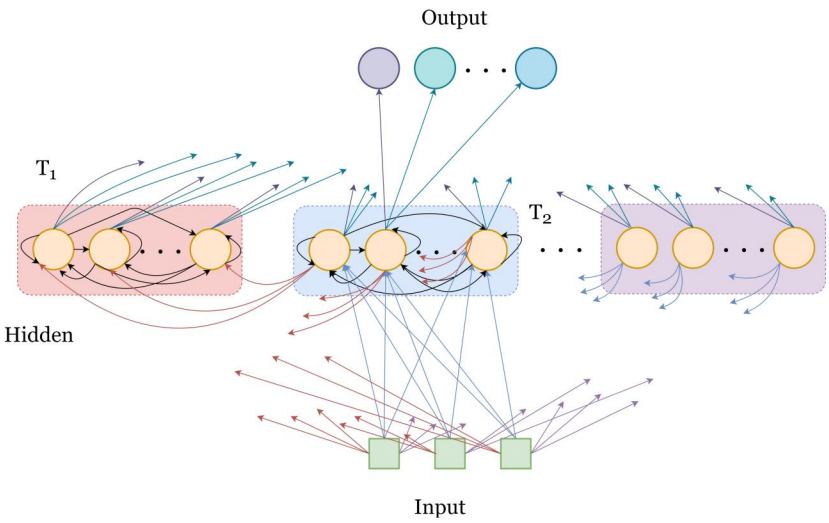

Figure 4: CW-RNN Architecture

The matrix W_H is a block-upper triangular matrix, where each block-row consists of block-columns. At each forward pass time step, only the block-rows corresponding to the executed modules are used to evaluate (1):

$$W_{Hi} = \begin{cases} W_{Hi} & \text{for } (t \text{ MOD } T_i) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

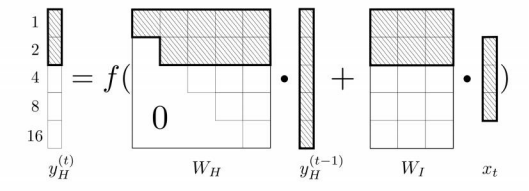

Figure 5: Calculation of the hidden unit activations at time step t = 6 in CW-RNN according to equation1

Hence, resulting in the corresponding parts of the output vector, y_H, being updated at each time step. Whereas, the other parts retain their output values from previous time steps. Therefore, low-clock-rate modules memorize long-term information from the input sequences, without being distracted by faster modules. This way, high-speed modules process local high-frequency information.

The back propagation of gradients is similar to regular RNNs. The only difference is that gradients propagate only from the modules that were executed at time step t. Therefore, CW-RNN runs faster than a regular simple RNN with the same number of hidden neurons, since not all modules are evaluated at each time step.

5 Experiments/Results/Discussion

5.1 CW-RNN

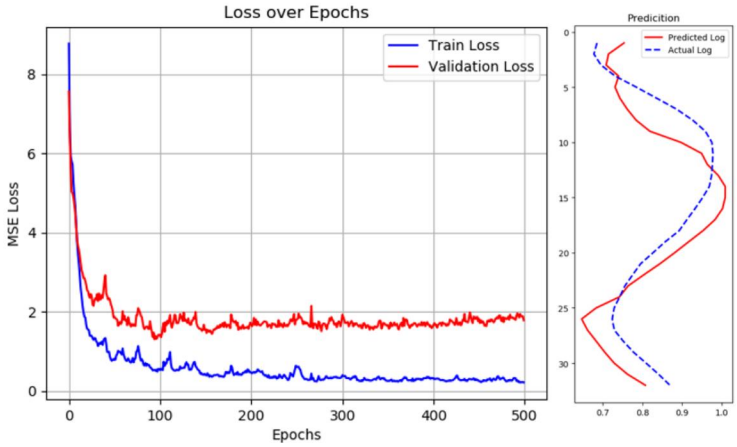

Figure 6: CW-RNN Loss Over Epochs and Prediction

CW-RNN training was carried out using 27588 training examples, and 1452 samples for testing. The model ran for 500 epoch, and optimized using an ADAM optimizer. The learning rate was 0.001, and the cost function is the mean squared error. Predictions of bulk density are superimposed on the actual log 6.

5.2 Hyperparameter Tuning

There are 3 hyperparameters need to be tuned for CW-RNN: Number of hidden units, number of clocking periods, and value of each clocking period. A hyperparameter sensitivity study was conducted, and the results are summarized in Tables 1 and 2. The model is most sensitive to the number of hidden units by which the hidden weight matrix W_H is partioned into.

No. Hidden Units	Clocking Periods	Train MSE	Test MSE
64	[1, 2, 3, 5, 8, 12, 18, 24]	4.90	7.63
128	[1, 2, 3, 5, 8, 12, 18, 24]	2.73	5.15
256	[1, 2, 3, 5, 8, 12, 18, 24]	1.76	3.04
512	[1, 2, 3, 5, 8, 12, 18, 24]		

Table 1: Sensitivity to Hidden Units

No. Hidden Units	Clocking Periods	Train MSE	Test MSE
256	[1, 2]	1.67	2.89
256	[1, 2, 3, 5]	1.31	3.23
256	[1, 2, 3, 5, 8, 12]	1.29	2.99
256	[1, 2, 3, 5, 8, 12, 18, 24]	1.14	2.69

Table 2: Sensitivity to Clocking Periods

6 Conclusion/Future Work

For the first time a model has been proposed to predict well logs from drilling data. The model needs to be tested and improved on other fields data sets before deployment. CWRNN for predicting

wireline logs from drilling data works better than the other investigated models, however, 1D CNN with inception layers are somehow predictive. Other variations of RNN (LSTM, GRU and Bi-LSTM) suffer from representing the mean of Sequence. The power of CWRNNs lies within their capability of memorizing data much better than Elman RNNs and LSTMs as they have a structured hidden layer that does not enforce representing the mean of all inputs (running average in case of the LSTM). Ideally a hybrid CWRNN-CNN may be the choice for this application. Another reason for superiority of CWRNN over other RNN is the lower number of parameters that needs to be learned compared to other RNNs, while they have an embedded memory.

For the future we are planning to acquire more data train more complex models. Due to the noisy nature of these data sets a multi-level wavelet transform will be investigated. We also noticed some non-physical values in the data sets, therefore we will seek consulting from experts to overcome these discrepancy of data.

7 Contributions

Rayan developed and hypertuned the inception model. He also visualized the open source dataset including wells, seismic volumes and seismic interpretations on OpendTect to have a better understanding of the field.

Mehrdad developed and tuned diffrenet RNN models. He implemented the deep RNN, GRU, LSTM and Bi-directional version of them. He also did the literature review and suggested to use MSE (mechanical specific energy) to better represent some human controlled data such as RPM and WOB.

Obai developed and hypertuned the CWRNN model. He found this method in literature and suggested using that, since we had the *mean* problem with other methods. He also performed the data prepossessing and augmentation in MATLAB.

References

This section should include citations for: (1) Any papers mentioned in the related work section. (2) Papers describing algorithms that you used which were not covered in class. (3) Code or libraries you downloaded and used. This includes libraries such as scikit-learn, Tensorflow, Pytorch, Keras etc. Acceptable formats include: MLA, APA, IEEE. If you do not use one of these formats, each reference entry must include the following (preferably in this order): author(s), title, conference/journal, publisher, year. If you are using TeX, you can use any bibliography format which includes the items mentioned above. We are excluding the references section from the page limit to encourage students to perform a thorough literature review/related work section without being space-penalized if they include more references. Any choice of citation style is acceptable as long as you are consistent.

References

References

[BBSNP13] M Baneshi, M Behzadijo, M Schaffie, and H Nezamabadi-Pour. Predicting log data by using artificial neural networks to approximate petrophysical parameters of formation. *Petroleum Science and Technology*, 31(12):1238–1248, 2013.

[KGGS14] Jan Koutnik, Klaus Greff, Faustino Gomez, and Juergen Schmidhuber. A clockwork rnn. *arXiv preprint arXiv:1402.3511*, 2014.

[MH15] Alireza Moazzeni and Mohammad Ali Haffar. Artificial intelligence for lithology identification through real-time drilling data. *Journal of Earth Science & Climatic Change*, 6(3):1–4, 2015.

[PMBH18] George Parapuram, Mehdi Mokhtari, and Jalel Ben Hmida. An artificially intelligent technique to generate synthetic geomechanical well logs for the bakken formation. *Energies*, 11(3):680, 2018.

[RMA+09] Luisa Rolon, Shahab D Mohaghegh, Sam Ameri, Razi Gaskari, and Bret McDaniel. Using artificial neural networks to generate synthetic well logs. *Journal of Natural Gas Science and Engineering*, 1(4-5):118–133, 2009.

[YKS19] Mehrdad Yousefzadeh, Rayan Kanfar, and Obai Shaikh. Cs230-project. `https://github.com/mehrdadyo/CS230-Project`, 2019.

[ZYJ18] Dongxiao ZHANG, CHEN Yuntian, and MENG Jin. Synthetic well logs generation via recurrent neural networks. *PETROLEUM EXPLORATION AND DEVELOPMENT-ELSEVIER-*, 45(4):629–639, 2018.