Introduction

The integration of multiple geophysical well logs together with seismic data can greatly reduce the ambiguity of geological interpretation and help to construct a better hydrocarbon reservoir model. However, not all types of logs are available at every single well in an area of interest because of cost limitations or borehole problems.

Several statistical and empirical models have been used to estimate missing logs. Gardner's equation provides a reasonable relationship between sonic and density for brine-saturated rock types (Gardner et al., 1974). Castagna et al. (1985); Greenberg and Castagna (1992) propose empirial relationships to calculate S-wave velocities (Vs) from P-wave velocities (Vp). These models can produce a useful predicted log; however, they are interval-based, depend on rock types, and often their calibration requires time and expertise. An alternative data-driven method is to use deep learning for missing well-log prediction. The method predicts missing logs at a specific well based on nearby training wells incorporating all available well log types. Rolon et al. (2009); Salehi et al. (2017) use fully connected neural networks (FCNNs) for predicting non-recorded logs from existing logs. However, FCNNs only produce a point-to-point mapping from input logs to output logs. Rock properties often demonstrate a trend with depths at a specific well, which are important for geological studies. Recurrent neural networks (RNNs) consider both internal input from previous step (e.g. trend) and external inputs. Zhang et al. (2018) use RNNs to generate synthetic well logs; however, the uncertainty of the predicting model is not quantified.

We propose a method to estimate missing logs (in this case Vs logs from other 8 types of logs) by using bidirectional LSTM cascaded with FCNN. We further add a dropout layer to quantify the uncertainty of the model. Output from our method is a Monte-Carlo distribution of log curves whose variance is the uncertainty of the model. We apply our method to 35 Central North Sea wells and show performance on three blind wells.

Theory

RNNs are an internal self-looped deep learning architecture used for natural language processing with sequential data. The output of RNN at each time step is affected by the input of the current step and the input from previous steps (Figure 1a). Let x be the input sequence and y be the output sequence with length T. $x^{<t>}$ and $y^{<t>}$ are the samples at time t of input and output sequence respectively, $a^{<t>}$ is the activation output of the network. The forward propagation of a recurrent neural network is

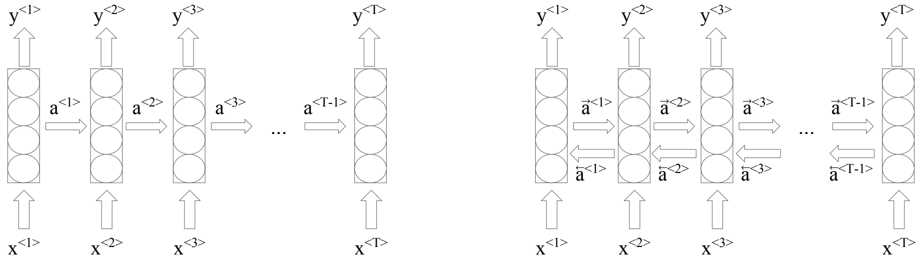
$$\begin{aligned} a^{<t>} &= g(W_{aa}\, a^{<t-1>} + W_{ax}\, x^{<t>} + b_a) \\ y^{<t>} &= g(W_{ya}\, a^{<t>} + b_y) \end{aligned} \tag{1}$$

where W_{aa}, W_{ax}, W_{ya}, b_a, b_y are trainable parameters; and g is an activation which normally is the hyperbolic tangent function. Bidirectional RNNs (BRNNs) are a development of RNNs, which use inputs from both earlier steps and later steps in the sequence (Schuster and Paliwal, 1997). BRNN is an acyclic graph which has a recurrent layer going backward in time (Figure 1b). The output of the network at each time step is calculated by activation outputs $\vec{a}^{<t>}$ and $\overleftarrow{a}^{<t>}$

$$\begin{aligned} \vec{a}^{<t>} &= g(\vec{W}_{aa}\, \vec{a}^{<t-1>} + \vec{W}_{ax}\, x^{<t>} + \vec{b}_a) \\ \overleftarrow{a}^{<t>} &= g(\overleftarrow{W}_{aa}\, \overleftarrow{a}^{<t-1>} + \overleftarrow{W}_{ax}\, x^{<t>} + \overleftarrow{b}_a) \\ y^{<t>} &= g(\vec{W}_{ya}\, \vec{a}^{<t>} + \overleftarrow{W}_{ya}\, \overleftarrow{a}^{<t>} + b_y) \end{aligned} \tag{2}$$

Basic RNNs, however, are not very good at capturing very long-term dependencies due to vanishing gradients problem, when working with long and dense-sampled well logs. Therefore, we replace a normal RNN architecture with a long short-term memory (LSTM) architecture to solve this problem (Hochreiter and Schmidhuber, 1997). In LSTM, the activation outputs are affected by different gates which decide whether to remove or add information to the cell states. A candiate value of memory cell state at each time step is computed from the activation output of the previous time step and external input at current time step

$$\tilde{c}^{<t>} = g(W_{ca}\, a^{<t-1>} + W_{cx}\, x^{<t>} + b_c) \tag{3}$$



(a)

(b)

***Figure 1** (a) RNN architecture. (b) BRNN architecture.*

The update and forget gate decide whether to update the cell state with the candidate

$$\begin{aligned} \Gamma_u &= \sigma(W_{ua}\, a^{<t-1>} + W_{ux}\, x^{<t>} + b_u) \\ \Gamma_f &= \sigma(W_{fa}\, a^{<t-1>} + W_{fx}\, x^{<t>} + b_f) \\ c^{<t>} &= \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * x^{<t-1>} \end{aligned} \tag{4}$$

where σ is the sigmoid function. The activation output at the current time step is calculated using the output gate Γ_o

$$\begin{aligned} \Gamma_o &= \sigma(W_{oa}\, a^{<t-1>} + W_{ox}\, x^{<t>} + b_o) \\ a^{<t>} &= \Gamma_o * \tanh(c^{<t>}) \end{aligned} \tag{5}$$

The output of the network is calculated similar to the basic RNNs. LSTM can be combined with BRNN to capture the long-term dependencies and to take inputs from both previous and future time steps. The architecture we use in this work has a BRNN followed by a FCNN. The BRNN has three forward LSTM layers and three backward LSTM layers with 200, 300 and 400 neurons respectively. There are four FCNN layers with 2048, 1024, 512, and 1 neurons respectively. The training dataset has 32 wells with gamma ray (GR), density (RHOB), Vp, resistivity (RT), volume of shale (Vsh), water saturation (Sw), neutron porosity (NPHI), depth (D), and Vs log. We calculate the minimum length of all wells in the training dataset and take samples from the begining of each well to the maximum multiple of minimum length. We then concatenate all types of logs except the Vs log along the last dimension. The training data is resampled to batches with “minimum-length” samples in each batch. This configuration is to avoid mixing samples of different depths from different wells in a batch. We pre-process the data by removing the median and scaling according the interquartile range, which is robust to the outliers and removes the effects of erroneous spikes in the training data.

Dropout layer can be used during training as a regularization and during test time to encourage the model learn the Bernouli distribution imposed over the paramters (Srivastava et al., 2014; Gal and Ghahramani, 2016). The Monte-Carlo sampling creates a distribution of the output prediction whose mean is taken as the final prediction and variance is used to quantify the model uncertainty. Our prediction model in this work has a dropout layer between the BRNN and the FCNN with probability $p = 70\%$. We take 100 Monte-Carlo samples when applying the model to construct the distribution.

Results

The network is trained with 100 epochs in which all data from all intervals are used (i.e. the process is not interval-based). The predictions at two of the blind wells are in Figure 2 and Figure 3. There is a good match between the deep-learning prediction and the recorded Vs log and also with the predictions based on rock physics models which was done on an interval by interval basis. The 7^{th} track in the left image is the predicted Vs (red curve) with 95% and 5% quantile (blue curves) obtained from Monte-Carlo simulation. The black curve in the same pannel is recorded Vs in the left image and the rock-physics based estimation in the middle and also in the zoomed section on the right. Figure 4 shows a close match between our proposed deep learning approach and the measured Vs at selected intervals of interests. The correlation coefficient between predicted Vs log and measured Vs is 94% (Figure 5).

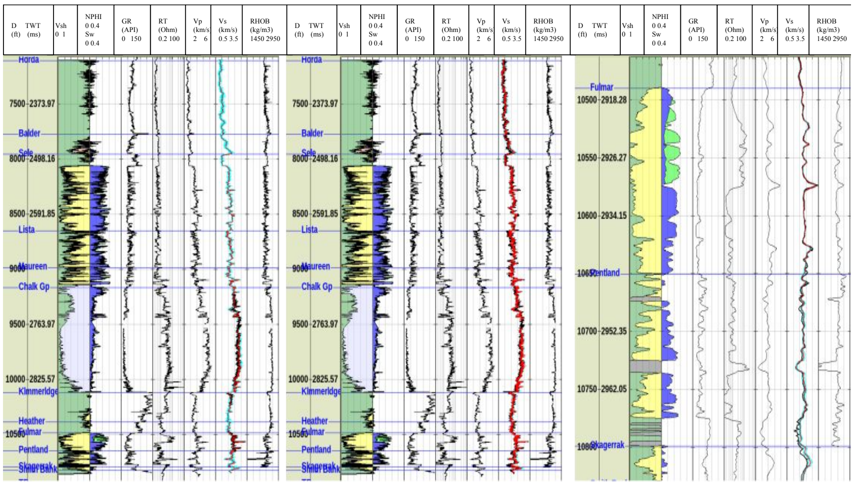


Figure 2 *Blind well 1: Left: Our predictions in red, the corresponding uncertainties in blue, and measured Vs in black. Middle: Our predictions in red whilst rock physics based estimations in black. Right: Same as middle but zoomed on the area of interest.*

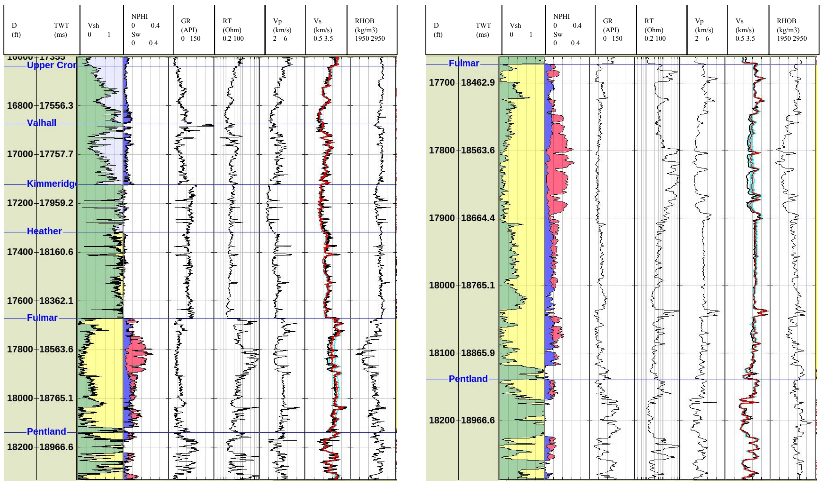
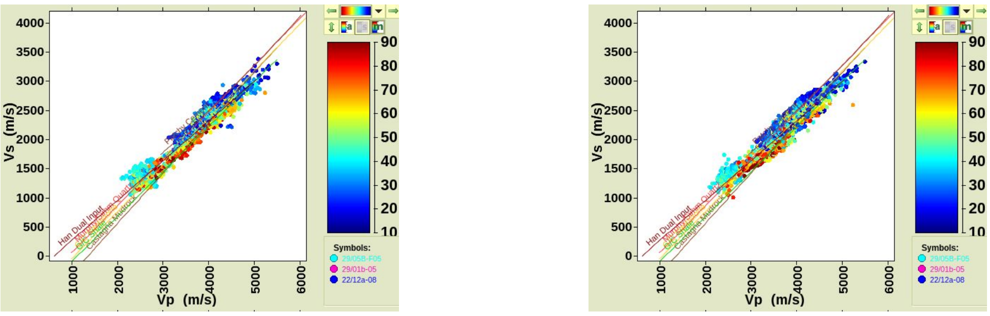


Figure 3 *Blind well 2: Left: Our predictions in red, the corresponding uncertainties in blue, and measured Vs in black. Middle: Our predictions in red whilst rock physics based estimations in black. Right: Same as middle but zoomed on the area of interest. A slightly higher uncertainty range is evident in the right plot around the gas saturated zone.*



(a)

(b)

Figure 4 *Blind wells' Vp-Vs crossplots colored by GR. The left plot uses rock physics based estimations whilst the right plot uses our predicted Vs. All the corresponding rock physics models are shown for reference.*

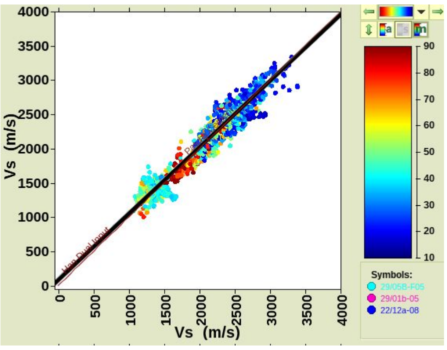


Figure 5 *Measured Vs versus our predicted Vs at blind wells. Red line: perfect prediction. Black line: best-fit test results.*

Conclusions

We propose a method for estimating missing well logs using a deep learning model consisting of BRNN with LSTM blocks cascaded with FCNN. An application of this technique was shown to predict Vs logs from Vsh, GR, NPHI, RT, D, Vp and Sw logs. Our model produces comparable predictions with rock physics based estimations facilitated by a Monte-Carlo simulation and the implementation of a probabilistic dropout layer produces a distribution of the predicted log from which the model uncertainty can be quantified. This is an important feature allowing a more confident decision from the predictions of a deep learning model. In this paper, we focus on estimating missing Vs logs; however, this approach can be extended to estimating any other kinds of missing logs using different input log types.

Acknowledgments

We thank Ikon Science for allowing to publish the results of this work and providing mannual petrophysical interpretations for comparison. The computations in this paper are done using the Tensorflow open source machine learning framework (Abadi et al., 2016).

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I.J., Harp, A., Irving, G., Isard, M., Jia, Y., Józefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D.G., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P.A., Vanhoucke, V., Vasudevan, V., Viégas, F.B., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. and Zheng, X. [2016] TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, **abs/1603.04467**.

Castagna, J.P., Batzle, M.L. and Eastwood, R.L. [1985] Relationships between compressional-wave and shear-wave velocities in clastic silicate rocks. *Geophysics*, **50**(4), 571–581.

Gal, Y. and Ghahramani, Z. [2016] Dropout As a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16. JMLR.org, 1050–1059.

Gardner, G., Gardner, L. and Gregory, A. [1974] Formation velocity and density—The diagnostic basics for stratigraphic traps. *Geophysics*, **39**(6), 770–780.

Greenberg, M.L. and Castagna, J.P. [1992] Shear-Wave Velocity Estimation in Porous Rocks: Theoretical Formulation, Preliminary Verification and applications. *Geophysical Prospecting*, **40**, 195–209.

Hochreiter, S. and Schmidhuber, J. [1997] Long Short-Term Memory. *Neural Comput.*, **9**(8), 1735–1780.

Rolon, L., Mohaghegh, S.D., Ameri, S., Gaskari, R. and McDaniel, B. [2009] Using artificial neural networks to generate synthetic well logs. *Journal of Natural Gas Science and Engineering*, **1**(4), 118 – 133.

Salehi, M.M., Rahmati, M., Karimnezhad, M. and Omidvar, P. [2017] Estimation of the non records logs from existing logs using artificial neural networks. *Egyptian Journal of Petroleum*, **26**(4), 957 – 968.

Schuster, M. and Paliwal, K. [1997] Bidirectional Recurrent Neural Networks. *Trans. Sig. Proc.*, **45**(11), 2673–2681.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. [2014] Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, **15**(1), 1929–1958.

Zhang, D., Chen, Y. and Meng, J. [2018] Synthetic well logs generation via Recurrent Neural Networks. *Petroleum Exploration and Development*, **45**(4), 629 – 639.