Proc. of the 8th International Symposium on Geotechnical Safety and Risk (ISGSR)
Edited by Jinsong Huang, D.V. Griffiths, Shui-Hua Jiang, Anna Giacomini and Richard Kelly
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-5182-7_08-016-cd

Using Convolutional Neural Networks and Monte-Carlo Dropout to Generate Synthetic Well Logs with Accurate Uncertainty Estimation

Yian Wong[1] and Sau-Wai Wong[2]

[1]Computer Science Department, The University of Texas at Austin, United States.
Email: yian@utexas.org

[2] rybkarock, Texas, USA; Visiting Professor, University of Newcastle, Australia.
Email: sauwai.wong@rybkarock.com, sau.wong@newcastle.edu.au

Abstract: The common and most acquired well logs are the basic petrophysical and lithological logs which usually include density, porosity, resistivity, gamma-ray, and customarily, the caliper measurement of borehole diameter to provide an indication of borehole and log quality. Acoustic log acquisition may not be routine, especially in a cost-constrained environment and in older wells. Previous works have developed methods to generate synthetic acoustic logs using petrophysical-based models. However, their reliability and applicability heavily depend on the complexity, variability, and uncertainty of subsurface rocks and fluids. The challenge is compounded by rock heterogeneity such as the highly variable and laminated unconventional shales. Generation of synthetic acoustic log values is considerably more challenging when rock formation is highly heterogeneous. Predictive models that perform well on relatively homogenous rock formations often struggle to predict accurately for heterogeneous rock formations such as the highly laminated shales and mudrocks. In this paper, we use two datasets, one set of well logs is taken from a highly heterogeneous shale formation, and the other is from a much more homogenous rock formation. We introduce and evaluate robust deep learning models and quantify the uncertainty of prediction using Monte-Carlo dropout. These models are classes of time-series neural network models: the Multi-Layer Perceptron, Long-Short Term Memory model, and Convolutional Neural Networks. We test some common machine learning approaches as baseline models on our datasets and show that the deep learning architectures consistently outperform the baseline models. Furthermore, the Monte-Carlo dropout approach is adopted and integrated into the deep learning models, which provides an estimate of the prediction uncertainty. Generally, the Convolutional Neural Network coupled with Monte Carlo dropout provides the most robust results with adequate quantification of prediction uncertainty.

Keywords: acoustic well log; rock heterogeneity; LSTM; Monte-Carlo dropout; convolutional neural network

1 Introduction

Petrophysical and lithological well logs are often essential tools for determining the physical properties and characteristics of subsurface rocks and formation fluids. However, it is often the case that acoustic well logs are missing or incomplete due to measurement failure or cost constraints. In many measurements, values such as the gamma-ray (GR), density (DEN), or resistivity (RT) are collected regularly, while dipole sonic acoustic logs can only be acquired for a smaller subset of wells and depths due to economic and feasibility constraints. This gives rise to the critical problem of generating synthetic acoustic well log values.

For the sake of economic efficiency and log accuracy, several methods have been developed to reconstruct these acoustic values. One approach uses theoretical physical models and empirical correlations to determine these values. These approaches aim to fit these wells to highly constrained models (Bateman, 2012). However, these approaches usually rely on many broad assumptions and empirical data. They often lack the expressivity to be generally applicable across different geological and lithological settings. Other methods frame the problem of generating synthetic acoustic well log values as a regression task, where they are regressing between known values of a well (like GR, DEN, or RT) as inputs and the desired acoustic values (like the compressional wave velocity and shear wave velocity, or the sonic transit travel time DTCO and DTSM, which are the inverse of compressional and shear wave velocities, respectively) as outputs.

Past works (Maleki et al. 2014) on the application of neural networks and decision trees for this type of problem did not attempt to quantify or estimate model uncertainty. For the problem of synthetic acoustic log generation or the general characterization of subsurface rocks and fluid, an indication of prediction uncertainty will be useful and desirable. Uncertainty estimation may provide guidance for well re-logging; attention can be given to areas/wells with high prediction uncertainty, for closer examination or re-logging. For example, additional data or logs may be acquired for areas where model uncertainty is particularly high while forgoing logging in areas where the model is more confident in its predictions. In these types of situations, expressing model uncertainty can perhaps facilitate a more informed, efficient, and economically optimized data acquisition plan.

In this paper, we devise three sequential neural network architectures, each based on a major subclass of neural networks: Multi-Layer Perceptrons (MLP), Long-Short Term Memory Model (LSTM), and Convolutional Neural Networks (CNNs). These model architectures are well suited to utilize the spatial and sequential nature of well logs to improve their prediction accuracy. We compare these neural network models to some common 'baseline' models; specifically, Support Vector Machines (SVM), Random Forests (RF), and Gradient Boosting Trees (GBT). In addition, the Monte-Carlo dropout (MC dropout) inference method is adopted to estimate the prediction uncertainty of neural networks.

2 Related Works

Akinnikawe et al. (2018) explore common machine learning methods on a specific well log dataset. Their results show that random forests and neural networks perform the best, which aligns with the idea that they can capture non-linear and, in some cases, discrete relationships. Both models are extremely expressive models that can capture the complexity of well logs, or more precisely - subsurface rock and fluid variability. Our paper expands on this work by not only evaluating MLPs, but also LSTMs and CNNs. Additionally, we use Monte-Carlo dropout to add uncertainty estimation to these deep learning models.

Zhang et al. (2018) used traditional LSTM models and showed that they perform better than regular neural networks and other machine learning methods. However, they did not mention or utilize 'bi-directional' LSTMs. This means that at a given depth level, they only used data from above this depth level to make a prediction, while bi-directional LSTMs (Bi-LSTMs) would incorporate information from above and below this depth level. Therefore, the present Bi-LSTM is an extension of LSTM with the incorporation of bi-directionality and the application of MC dropout to capture model uncertainty.

Chen (2020) explored 'ensemble' LSTMs (EnLSTMs), which are regular LSTMs trained using ensemble randomized maximum likelihood. This algorithm is based on Bayes' theorem, and it specializes in tasks with a limited amount of data because one can inject one's beliefs about the data in the form of probabilistic priors. EnLSTMs require two perturbation methods, which we believe could be highly case-specific and hence make the model less generalizable to different tasks and datasets. Comparisons of our Bi-LSTM model with less common models like EnLSTMs may be explored in future works.

3 Dataset

We evaluate the models using two vastly different datasets. The first dataset is taken from the Tertiary Basin in the Gulf of Mexico, publicly available through the SEAM[1] phase 1 project. A prominent feature is the salt formation, embedded in a sedimentary system. Although not entirely homogeneous, it is considerably much more homogeneous when compared to the second dataset taken from an unconventional resource formation consisting of mainly shale/mudrock. It is expected that the highly heterogeneous rock formation would present challenges for in generating high quality well log prediction. Each dataset contains a suite of well logs from the same basin with similar geology and rock formation; they typically consist of gamma-ray, density, resistivity, porosity, and the transit time of compressional wave to predict the 'missing' compressional wave logs in a nearby well. All these wells are near-vertical wells, and their distance from each other varies from 1+ km to several km.

For simplicity and the present discussion, we will focus only on the generation and prediction of the compressional acoustic wave, or the more commonly used compressional wave transit travel time, DTCO. The same algorithms for predicting acoustic shear waves could simply follow the same approach[2].

4 Neural Networks

In this paper, we will survey the multi-layer perceptron (MLP), Bi-directional long short-term memory model (Bi-LSTM), and convolutional neural network (CNN). These neural networks are all trained via gradient descent and backpropagation, where the gradient of the model parameters is determined by calculating the gradient of some objective function, which is the so-called loss function:

---

[1] SEAM: The SEG Advanced Modeling Project - a collaborative industrial research effort dedicated to large-scale leading-edge geophysical numerical simulation.

[2] Shear waves can be easily predicted from compressional waves using empirical correlations as well as machine learning methods (Maleki et al, 2014) but it is predicated on the availability of compressional wave log in the first place, which is not our assumption in this paper.

$$\nabla_\theta L = \frac{\partial L}{\partial \theta} \tag{1}$$

where L is the loss function and θ is the neural network parameters. We iteratively shift the parameter values in the negative direction of the gradient of our loss function, which analytically is the same as the direction greatest descent for the loss function value. In the case of regression, our objective is to predict output values as close as possible to the ground truth. Hence, our loss function is the mean squared error (MSE) between the prediction of the model and the ground truth. To train on sequential data, we unpack individual sequences of length l for every instance from the dataset. This means that for every output y_i, we use inputs $x_{i-(\frac{l}{2})}, \dots, x_i, \dots, x_{i+(\frac{l}{2})}$.

To ensure that we have inputs at the edges of the dataset, we pad the dataset with the beginning instance and ending instance. We also employ mini batching when performing gradient descent. Rather than computing the gradient of the loss function across the entire dataset, we instead randomly batch instances together and compute the gradient on this smaller batch. This technique, called stochastic gradient descent, allows for faster convergence since we are taking many steps by approximating the gradient on subsets of the datasets, rather than computing the gradient over the entire dataset and taking one step.

In practice, we use a variant of stochastic gradient descent, called *Adam*. This is a standard heuristic algorithm that is widely used for deep learning which adapts the step sizes used during gradient descent to allow for faster convergence. We have adopted a sequence length of 32 and a batch size of 64.

4.1 Multi-Layer Perceptron

MLPs are also known as artificial neural networks or deep neural networks. They are modeled as layers: starting from the input layer, then into hidden layers, and finally into output layers. Each hidden layer and output layer consists of multiple nodes. Each node performs a unique logistic regression from the outputs of nodes of the previous layer. That is, each node computes the following:

$$f(\sum_i x_i w_i + b) \tag{2}$$

where x_i is the i^{th} value from the previous layer, w is a weight vector corresponding to the node, and b is a bias value. The function f is a non-linear function, which aims to introduce non-linear behavior into the neural network. In all our neural networks, we use the rectified linear units (ReLU) function as our non-linearity function, as it has seen wide success in a vast array of tasks.

4.2 Long-Short Term Memory Models

LSTMs are a type of recurrent neural network (Hochreiter & Schmidhuber, 1997). Recurrent neural networks (RNNs) are particularly well suited to handle sequential data. Typically, RNNs have a repeating component in their architecture, where its output gets fed into itself. This component is called its 'hidden state' and contains compressed information about all the past inputs, which can help the neural network make a better prediction.

Traditional RNNs have difficulty in learning long-term dependencies because of vanishing gradients. This deficiency has motivated the development and application of the LSTM structure. An LSTM unit has four gates, an input gate, a forget gate, an output gate, and a tanh (hyperbolic tangent) gate. The input gate, i_t, and the forget gate, f_t, are in the form:

$$i_t = \sigma(W_i(h_{t-1}, x_t) + b_i) \tag{3}$$

$$f_t = \sigma(W_f(h_{t-1}, x_t) + b_f) \tag{4}$$

where W and b are corresponding weight and bias vectors for each gate, h_{t-1} is the hidden state of the LSTM from the previous timestep $t-1$, and x_t is the input from the current timestep. The tanh layer is in the form:

$$C_t{}' = tanh(W_c(h_{t-1}, x_t) + b_c) \tag{5}$$

Hence, the next hidden state is determined by these three layers. The forget gate is multiplied by the last hidden state to potentially add or remove information from previous input. The input gate proposes potential elements of the hidden state to be updated, while the tanh layer proposes potential changes to the hidden state:

$$C_t = f_t C_{t-1} + i_t C_t' \tag{6}$$

The output gate represents the output of the LSTM and is given by:

$$o_t = \sigma(W_o(h_{t-1}, x_t) + b_o) tanh(C_t) \tag{7}$$

The structure of LSTMs is especially useful in the case of well logs because different sections of the well can be characterized by different rock 'types', and these rock types could be identified by 'long-term' dependencies that the LSTM can potentially identify. Rather than having just one LSTM model that models sequences forward in time, our proposed Bi-LSTMs also use another LSTM model that models the sequence backward in time. Therefore, inputs from before and after a certain depth position are involved in predicting the well log response.

4.3 Convolutional Neural Networks

CNNs take advantage of the spatial arrangement of data, such as the depth readings of well logs. CNNs are typically used in two dimensions, usually for image processing. In our use-case, we perform one-dimensional convolutions. CNNs consist of many filter vectors that all have a given size. These filters slide across the input dataset sequentially. In each slide, the filter takes a stride to shift the filter to cover a different area of the dataset. The area the filter covers at a given moment is called the receptive field. To extract a feature, a CNN computes the dot product between the filter and its receptive field to create a resultant vector with dimensions smaller than the input.

Compared to traditional neural networks, intermediate features of convolution's resultant sequence represent only a small part of the original sequence, whereas in a neural network one feature is dependent on the entire sequence in its hidden layers. This localized feature extraction allows for fewer overall model parameters than a regular neural network and gives the CNN the ability to learn more robust patterns without overfitting.

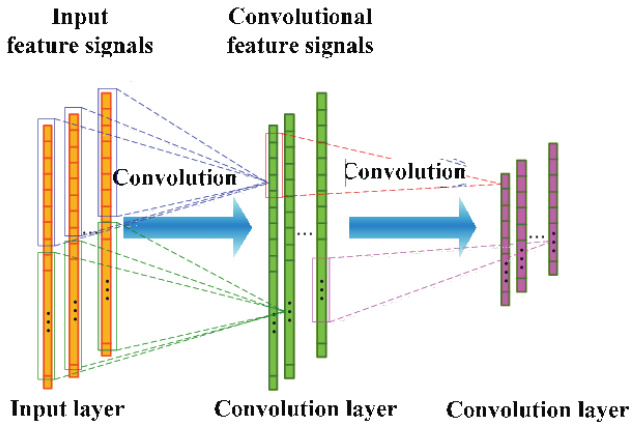

Figure 1. An illustration of CNN. The network takes as input different feature signals and performs convolutions on the input to ultimately predict the output.

5 Monte Carlo Dropout

MC Dropout can be seen as a method of treating dropout as a Bayesian approximation (Gal & Ghahramani, 2016). To denote a predictive distribution of a Bayesian probabilistic model of an output y, an input x, and a dataset $D = (x_i, y_i)_{i=1}^N$ with N instances we use:

$$p(y|x, D) \tag{8}$$

In Bayesian probability, we typically arrive at a predictive distribution by learning a distribution over functions, or equivalently a distribution over a generalized function's parameters, $p(\theta|\, D)$. We treat the learned parameters from stochastic gradient descent of a neural network as a posterior distribution over its parameters, where we can sample from it using random dropout, randomly zeroing out parameters with probability p to effectively sample the output of a random subset of the model.

$$\theta_t \sim q(\theta|D) \tag{9}$$

We use these samples to approximate the variational inference of the neural network's predicted likelihood:

$$p(y|x) = \int p(y|x,\theta)q(\theta|D)d\theta \tag{10}$$

$$p(y|x) = \frac{1}{T}\sum_{t=1}^{T} p(y|x,\theta_t)\ s.t.\ \theta_t \sim q(\theta|D) \tag{11}$$

In practice, we can sample with random dropout for T times, giving us T point estimates of a predictive distribution of our model. We denote the mean value of these point estimates as μ and the variance as σ^2. For simplicity's sake, we assume the predictive distribution is normally distributed, hence:

$$p(y|x) = N(\mu, \sigma^2) \tag{12}$$

The variance of this predictive distribution is what we use to express the uncertainty of all neural networks in our experiments. We use this technique and consider probabilities with $p = 0.1$ and $p = 0.2$ and sample each data instance with 50 forward passes for samples.

5.1 Uncertainty Estimation

Measuring performance in uncertainty estimation can be unintuitive at first glance. It is difficult to think of a score that values both accuracy and uncertainty, where low uncertainty should be rewarded when the predicted value is accurate and punished when it is inaccurate. We employ the log-likelihood of the true value in the predictive distribution as a metric for evaluating uncertainty estimation. In the context of Monte Carlo dropout, that is:

$$\log\ [p(y_i \mid x_i, \theta)] \tag{13}$$

where y_i and x_i are the i^{th} output and inputs of the dataset.

In Figure 2, the value -1 is the ground truth. There are three example predictive distributions. The red line on each distribution graph shows the likelihood (horizontal line) of the ground truth value (vertical line) in the given distribution. The y-axis has likelihood values for corresponding output values on the x-axis. The likelihood value is highest for distributions that are both accurate and certain, but lowest for distributions that are inaccurate and certain. There is a middle ground in the likelihood value for when the distribution is both inaccurate and uncertain. In practice, we will compare models by their log-likelihood (i.e., the natural log of the likelihood values) to make for an easier comparison. This is because log-likelihood is bounded by [-∞,0], while the likelihood is bounded by [0,1]. The sparser output bounds of log-likelihood make a comparison between models easier.

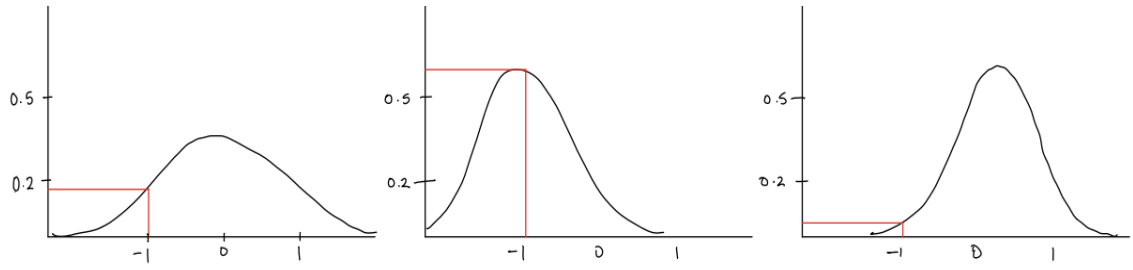

Figure 2. Comparison of multiple probability distributions. Left: An inaccurate, but high uncertainty distribution. Center: An accurate and low uncertainty distribution. Right: An inaccurate and low uncertainty distribution.

6 Baseline Models

We survey both decision tree methods (random forests (RF) and gradient boosting trees (GBT)) and support vector machines (SVMs) as baseline models. To appropriately assess the predictive performance of advanced deep learning neural network models, we use decision trees and SVMs as a baseline to provide an assessment of performance. These baseline models are commonly used in all types of regression tasks and hence are implemented in Sci-Kit Learn, a popular scientific computing Python package (Pedregosa et al. 2011). We use these implementations in our experiments with their default parameter settings.

7 Analysis

We assess the performance of every model on both the homogenous and heterogenous datasets. Since both datasets have multiple wells, we perform an experiment for every well. We pick out a target well, and we train the model on the other wells, and test its performance on the target well. We do this for every well in the dataset in a manner like k-fold cross validation.

For our baseline models, we use the GR, DEN, and RT values at depth d as an input to predict the DTCO at depth d. In the case of our proposed sequential neural network models, we use all the GR, DEN, and RT values at depth $d - \frac{l}{2}, \ldots, d, \ldots, d + \frac{l}{2}$. We also add dropout layers after before every layer in every model both for training and inference using MC dropout.

A variant of stochastic gradient descent, called Adam, is used. This is a standard heuristic algorithm that is widely used for deep learning which adapts the step sizes used during gradient descent to allow for faster convergence. We also use a sequence length of 32 and a batch size of 64.

All models are compared using the average mean squared error (MSE) of predicted DTCO values for all wells. For our neural networks, we will compare their log-likelihoods to assess the performance of their uncertainty estimation capabilities. The predicted uncertainties are represented in well logs along with the model's predictions and actual measure log data (ground truth) to provide a sense of how uncertainty prediction looks in practice.

For all experiments and the sake of consistent training, we scaled all inputs and outputs by their inner-quartile range to create normalized inputs for neural network models, as the scaling of inputs and outputs affects the gradients in deep learning.

8 Results

In Table 1, the deterministic results of all baseline models and the neural network models for both the heterogenous (set A) and homogenous (set B) well log datasets are presented. The wells in the 'heterogeneous' formation are labeled from A1 to A4, and the wells in the 'homogenous' formation are labeled B1 to B5. The best-predicted results for each well log as quantified by the lowest MSE are in bold. MSE is the mean squared error of the prediction measured against that specific well's actual data, the 'ground truth'.

These results shown in Table 1 compare the output of all models deterministically, hence we do not account for the neural network's predicted uncertainties, and instead use the means of their predictive distributions as maximum likelihood estimates of their predictions. Neural network models perform similarly to each other, and one of the three neural network models outperforms the baseline models in 7 of the 9 wells in total. In datasets B3 and B5, decision tree models outperform our neural network models. However, neural network models still perform very well both in absolute terms and relative to the decision tree performance (within an error of 4).

Table 1. Deterministic comparison of neural network models and baseline models in synthetic well log generation. Measured using mean squared error (lower is better).

Well	GBT	SVM	RF	MLP	Bi-LSTM	CNN
A1	51.59	54.40	57.97	47.08	**45.14**	54.23
A2	39.23	62.09	41.72	28.65	32.89	**28.45**
A3	28.44	38.65	36.41	37.56	30.12	**28.64**
A4	35.83	36.97	40.76	39.61	36.67	**32.98**
B1	9.867	13.38	10.81	**4.84**	8.23	6.652
B2	6.578	10.55	6.936	**4.83**	5.029	5.144
B3	2.774	14.54	**1.916**	3.28	16.49	3.407
B4	17.12	17.43	18.32	15.36	20.23	**14.42**
B5	**.532**	14.40	1.849	2.64	17.54	4.291

CNNs outperform all models consistently in the heterogenous well dataset, achieving significantly lower errors than the rest of the models. These results demonstrate that CNNs can share parameters by localizing features and allow the models to generalize easily without overparameterizing the objective.

We also evaluate our models coupled with MC Dropout. To do this, the log-likelihood of the ground truth in the predictive probability distribution is analyzed. Since we are taking the log of a probability, the value will be negative, and therefore, a log-likelihood value that is closer to 0, or less negative, is better. Table 2 shows the average log-likelihood measured across each well for the three neural network models using MC dropout. Higher log-likelihood values are indications of a combination of superior uncertainty estimation and prediction accuracy. A log-likelihood value that is very close to zero means that the predicted distribution assigns a very high probability to the true value. If the model is uncertain, then its predicted distribution could still assign a reasonably high log-likelihood even if its distribution is centered at an inaccurate value. In this case, the model's log-likelihood would be much higher than a model prediction that is both inaccurate and overconfident.

The results show that CNNs are about twice as good as the next best model in the heterogenous well log dataset. They also perform the best or the next best in the homogenous rock formation dataset. While MLPs outperform CNNs in well logs B1, B2, and B3, their average log-likelihood are very similar. Additionally, the log-likelihood scores of the CNNs in all wells fluctuate very little compared to the MLP or Bi-LSTM. We believe this quality demonstrates the robustness of the model, and its ability to make strong generalizations about the data it is given and express uncertainty adequately and appropriately in both datasets.

Table 2. Probabilistic comparison of neural network models in synthetic well log generation. Measured using log-likelihood probability (higher/closer to zero is better).

Well	MLP	Bi-LSTM	CNN
A1	-17.22	-10.54	**-5.735**
A2	-22.90	-14.03	**-5.839**
A3	-23.51	-15.35	**-5.427**
A4	-27.06	-19.28	**-7.473**
B1	**-2.162**	-2.399	-2.367
B2	**-1.762**	-2.290	-2.387
B3	**-1.710**	-2.474	-2.220
B4	-5.067	-3.017	**-2.918**
B5	-1.713	-2.304	**-2.275**

Figure 3 shows the predictive distributions of each respective model in a subset of a well from the homogenous dataset. These graphs show the common behavior and potential problems with each model with MC dropout implementation. It can be observed that CNN predicts the ground truth to be within at most one standard deviation of the predictive mean throughout this entire segment, at the cost of considerably higher predicted uncertainty as shown by the wider uncertainty ranges in the graph. MLPs and Bi-LSTMs both tend to predict much lower uncertainties, but consequentially predict the ground truth to be over 3 standard deviations away from the mean in many cases. In some instances, the ground truth even falls outside of the third standard deviation away from the predictive mean, indicating model overconfidence.

Figure 4 demonstrates similar model behavior in the heterogeneous dataset. The deterministic error in all models is observable, exemplifying the difficulties and challenges in modeling heterogeneous acoustic well logs. We can see that in the heterogenous well subsection, the CNN exceeds the Bi-LSTM and MLP in both deterministic and uncertainty estimation. Almost all the ground truth DTCO values fall within the first standard deviation range of the uncertainty prediction in the CNN results. Bi-LSTMs and MLPs predict distributions that are extremely overconfident, and most ground truth labels fall outside of the three standard deviation range.

From the qualitative graphs and quantitative results, it can be observed that CNN is very reasonable in expressing uncertainty even for the highly heterogeneous rock formation and provide a bound for possible model inaccuracy.

MLPs and Bi-LSTMs appear to consistently underestimate uncertainty. MLP and Bi-LSTM models may have too many parameters and are overfitting to their training data. The consequence of such overfitting is for the models to become unreasonably overconfident in their predictions.

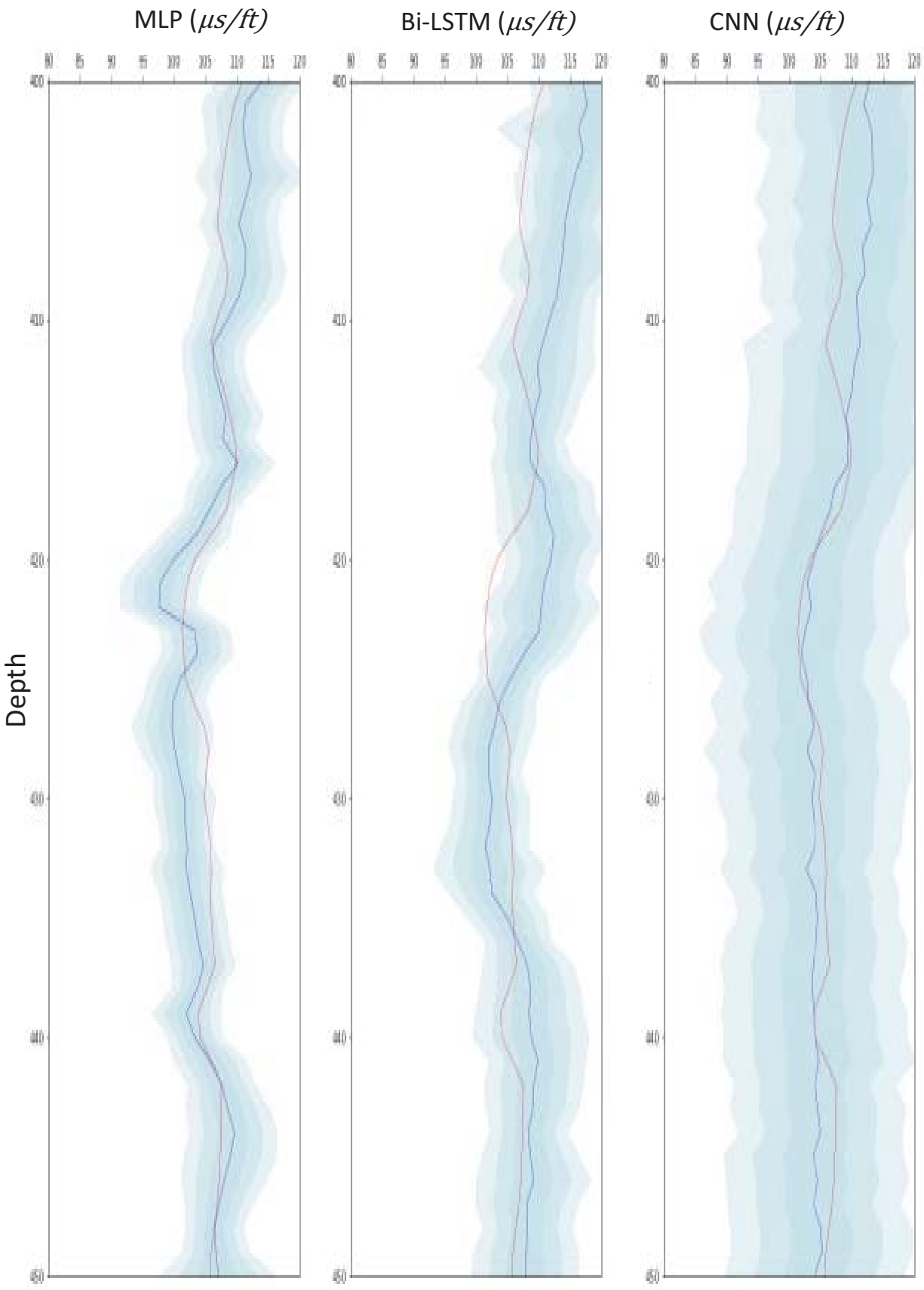

Figure 3. Predictive mean and uncertainties on the **homogenous** dataset for MLP (left), LSTM (middle), CNN (right). In red is the ground truth recorded DTCO. In blue is the predictive mean of the model's predictive distribution. The darkest blue shade represents the range of one standard deviation away from the mean. Subsequent shades of blue represent one extra standard deviation.

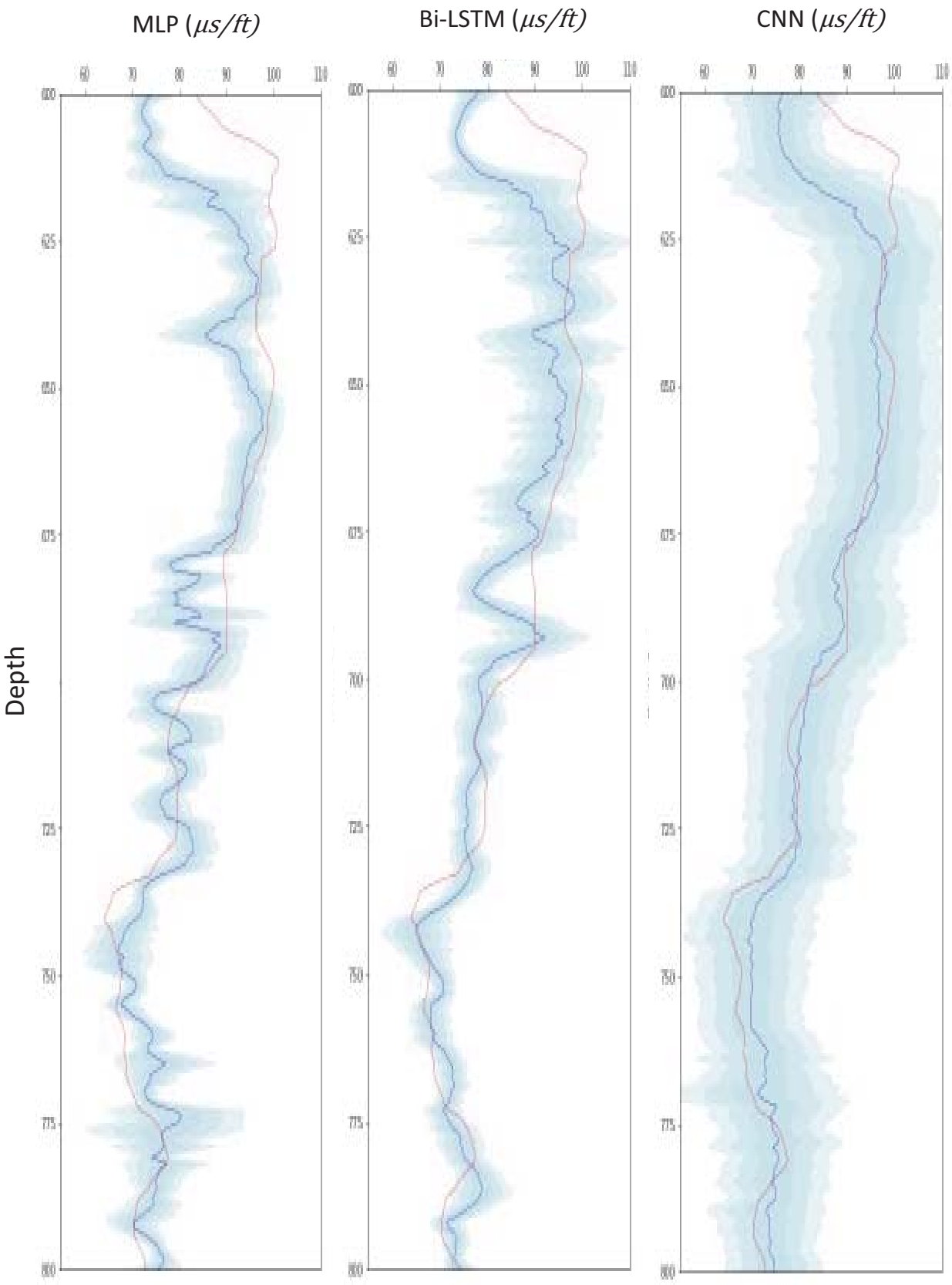

Figure 4. Predictive mean and uncertainties on the **heterogenous** dataset for MLP (left), LSTMs (middle), CNNs (right). In red is the ground truth recorded DTCO. In blue is the predictive mean of the model's predictive distribution. The darkest blue shade represents the range of one standard deviation away from the mean. Subsequent shades of blue represent one extra standard deviation.

9 Conclusion

Many of the state-of-the-art models in well log generation have leveraged the power of deep learning neural networks. In this paper, we have explored a range of models and identified a consistently better model than common methods such as the traditional neural networks or random forests. Our proposed methodology of combining the deep learning CNN model and MC dropout for uncertainty estimates, not only outperforms these common baseline models significantly but is also able to quantify the uncertainty in model prediction. Crucially, our approach does not increase the computational complexity of the model to estimate prediction uncertainty. We believe that capturing uncertainty in model prediction will continue to be a valuable task not just for synthetic well log generation but more generally, in evaluating subsurface geological data prediction. This is the subject of ongoing research and future publications.

References

Akinnikawe, O., Lyne, S., & Roberts, J. (2018). Synthetic well log generation using machine learning techniques. *Proceedings of the 6th Unconventional Resources Technology Conference*. https://doi.org/10.15530/urtec-2018-2877021

Bateman, R. M. (2012). *Openhole log analysis and formation evaluation*. Published by Society of Petroleum Engineers. ISBN 13:9781613991565.

Breiman, L. (2001). *Random forests - machine learning*. SpringerLink. Retrieved June 13, 2022, from https://link.springer.com/article/10.1023/A:1010933404324

Chen, Y., & Zhang, D. (2020). Well log generation via ensemble long short-term memory (enlstm) network. *Geophysical Research Letters, 47(23).* https://doi.org/10.1029/2020gl087685

Gal, Y., & Ghahramani, Z. (2016, October 4). *Dropout as a Bayesian approximation: Representing model uncertainty in Deep Learning*. arXiv.org. Retrieved March 26, 2022, from https://arxiv.org/abs/1506.02142

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Maleki, S., Moradzadeh, A., Riabi, G.R., Gholami, R. and Sadeghzadeh, F. (2014). Prediction of shear wave velocity using empirical correlations and artificial intelligence methods. *NRIAG Journal of Astronomy and Geophysics. Vol 3(1).* https://doi.org/10.1016/j.nrjag.2014.05.001

Pedregosa, F., Profile, V., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, É., & Metrics, O. M. V. A. (2011, February 1). *Scikit-Learn: Machine learning in Python*. The Journal of Machine Learning Research. Retrieved June 13, 2022, from https://dl.acm.org/doi/10.5555/1953048.2078195

Zhang, D., Chen, Y., & Meng, J. (2018). Synthetic well logs generation via recurrent neural networks. *Petroleum Exploration and Development, 45(4), 629–639.* https://doi.org/10.1016/s1876-3804(18)30068-5