UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

STATISTICAL AND DEEP LEARNING METHODS FOR GEOSCIENCE PROBLEMS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By
SAURABH SINHA
Norman, Oklahoma
2021

STATISTICAL AND DEEP LEARNING METHODS FOR GEOSCIENCE PROBLEMS

A DISSERTATION APPROVED FOR THE

SCHOOL OF GEOSCIENCES

BY THE COMMITTEE CONSISTING OF

Dr. Kurt J Marfurt, Chair

Dr. Heather Bedle

Dr. Matthew Pranter

Dr. Mashhad Fahes

Dr. Sumit Verma

Acknowledgements

I want to thank my mother and my father, Lila Sinha, and Rakesh Sinha, as well as my siblings and my extended family for their support for this project. I would like to express my deepest gratitude and appreciation to my advisor Dr. Kurt Marfurt for his expertise, motivation, and continuous support throughout my PhD. To the committee members, Dr. Matthew Pranter, Dr. Sumit Verma, Dr. Heather Bedle and Dr. Mashhad Fahes, I am extremely grateful for being part of my committee, for the valuable feedback, advise and encouragement. I wish to extend my gratitude to Dr. Rafael Pires De Lima for being a great friend and the partner in all the projects we did together. To Rebecca Fay for helping me with numerous forms and paperwork that I was drowning in day after day; without her help I will never be able to get through the enormous amount of paperwork needed to reach this stage.

To Dr. Yuliana Zapata, my fiancée, thank you for all your love and support for the last seven years and many more years to come. Without you, I will never be able to be able to achieve what I did in these years.

Table of Contents

List of Tables

List of Figures

Abstract

Machine learning is the new frontier for technology development in geosciences and has developed extremely fast in the past decade. With the increased compute power provided by distributed computing and Graphics Processing Units (GPUs) and their exploitation provided by machine learning (ML) frameworks such as Keras, Pytorch, and Tensorflow, ML algorithms can now solve complex scientific problems. Although powerful, ML algorithms need to be applied to suitable problems conditioned for optimal results. For this reason ML algorithms require not only a deep understanding of the problem but also of the algorithm's ability. In this dissertation, I show that Simple statistical techniques can often outperform ML-based models if applied correctly.

In this dissertation, I show the success of deep learning in addressing two difficult problems. In the first application I use deep learning to auto-detect the leaks in a carbon capture project using pressure field data acquired from the DOE Cranfield site in Mississippi. I use the history of pressure, rates, and cumulative injection volumes to detect leaks as pressure anomaly. I use a different deep learning workflow to forecast high-energy electrons in Earth's outer radiation belt using in situ measurements of different space weather parameters such as solar wind density and pressure. I focus on predicting electron fluxes of 2 MeV and higher energy and introduce the ensemble of deep learning models to further improve the results as compared to using a single deep learning architecture.

I also show an example where a carefully constructed statistical approach, guided by the human interpreter, outperforms deep learning algorithms implemented by others. Here, the goal is to correlate multiple well logs across a survey area in order to map not only the thickness, but also to characterize the behavior of stacked gamma ray parasequence sets. Using tools including maximum likelihood estimation (MLE) and dynamic time warping (DTW) provides a means of

generating quantitative maps of upward fining and upward coarsening across the oil field. The ultimate goal is to link such extensive well control with the spectral attribute signature of 3D seismic data volumes to provide a detailed maps of not only the depositional history, but also insight into lateral and vertical variation of mineralogy important to the effective completion of shale resource plays.

Introduction

This dissertation consists of my work as a Ph.D. student at the University of Oklahoma and a student intern at Los Alamos National Laboratory in Los Alamos, NM. Chapters in this dissertation are organized based on the journal papers published or submitted, but all chapters maintain the first-person plural. Despite great help from all the co-authors, I was responsible for most of the code development, writing the papers, and developing the figures.

Chapter 1 introduces the deep learning architectures I frequently use. This chapter introduces the terminologies and concepts when using deep learning architectures such as multi-layer perceptron, convolutional neural networks, long-term short memory, and a combination of these. I also introduce the intuitions behind these architectures in this study and their appropriate use along with mathematical formulations and the relevant figures that explain these concepts. I place a lot of emphasis on the fundamental concepts in this chapter instead of many different variants of similar architectures that use same basic structure with minimal conceptual changes.

Chapter 2 is presented as a submitted publication to the journal*Interpretation* and is a refined version of the abstracts (Sinha et al., 2018; Sinha et al., 2019) and is submitted for publication in the SEG/AAPG journal Interpretation. This chapter shows statistical segmentation and dynamic time warping for the semi-supervised well log correlation. The main body of this chapter demonstrates an application to semi – automate the picking of the parasequences on well logs and the quantitative estimation of the parasequences in the form of well log attributes. This chapter includes two elaborate appendices for the mathematical formulation of the algorithms for interested readers.

Chapter 3 is presented as it was published in the Journal of Greenhouse Gas Control (Sinha et al., 2020a), an expanded abstract at SPE article (Sinha et al., 2020b). This chapter demonstrates the use of deep learning architectures to automate the critical task of leak detection in carbon sequestration projects using pressure field data. The problem of leak detection in carbon sequestration projects is posed as an anomaly detection instead of a classification problem. In this chapter I test a total of 13 deep learning models for pressure forecasting and rank them based on their ability to accurately detect the leaks.

Chapter 4 is presented as it was published in AGU Space Weather Journal (Sinha et al., 2021) and as an EGU expanded abstract (Chen et al., 2021). This chapter shows how to use deep learning architectures to predict high-energy electron fluxes that cause solar storms and affect satellites and other spacecraft. This study also introduces deep learning ensemble models for space weather prediction tasks in the PREMEV series of models developed jointly by Los Alamos National Laboratory and the National Aeronautics and Space Administration (NASA). (NASA).

References

Chen, Y., R. Pires de Lima, S. Sinha, and Y. Lin, 2021, PreMevE: A Machine-Learning Based Predictive Model for MeV Electrons inside Earth's Outer Radiation Belt: EGU General Assembly Conference Abstracts, EGU21-8545.

Sinha, S., R. P. de Lima, Y. Lin, A. Y. Sun, N. Symons, R. Pawar, and G. Guthrie, 2020a, Normal or abnormal? Machine learning for the leakage detection in carbon sequestration projects using pressure field data: International Journal of Greenhouse Gas Control, **103**, 103189.

Sinha, S., R. Pires De Lima, Y. Lin, A. Y Sun, N. Symon, R. Pawar, and G. Guthrie, 2020b, Leak Detection in Carbon Sequestration Projects Using Machine Learning Methods: Cranfield Site, Mississippi, USA: SPE Annual Technical Conference and Exhibition.

Sinha*, S., R. Kiran, J. Tellez, and K. Marfurt, 2019, Identification and Quantification of Parasequences Using Expectation Maximization Filter: Defining Well Log Attributes for Reservoir Characterization: Unconventional Resources Technology Conference, Denver, Colorado, 22-24 July 2019, 62–73.

Sinha, S., R. P. de Lima, J. Qi, L. Infante-Paez, and K. Marfurt, 2018, Well-log attributes to map upward-fining and upward-coarsening parasequences: 2018 SEG International Exposition and Annual Meeting.

Sinha, S., Y. Chen, Y. Lin, and R. P. de Lima, 2021, PreMevE Update: Forecasting Ultra-relativistic Electrons inside Earth's Outer Radiation Belt: ArXiv Preprint ArXiv:2104.09055.

Chapter 1: Introduction to deep learning architectures

Introduction

Neural networks are often regarded as the "black box" and share many terminologies with various scientific disciplines. This overlap and different terminologies often confuse a non-expert reader such that critical aspects of the project are lost in jargon. In this chapter, I provide an intuitive understanding of neural networks and define the terms used in this dissertation.

The artificial neuron

A biological neuron inspires an artificial neuron (Figure 0-1). The biological neuron in Figure 0-1a) receives input at the dendrites, processes the signal in the neuron's main body, and outputs the results to the brain via the axons. Figure 0-1b) shows a simple an artificial neuron that takes an input, multiplies it with some model weights, and then produces an output. An artificial neuron is also referred to as a perceptron in some literature. In this dissertation, I will use the word as a neuron.

The input or the input features in a neuron can be a scalar or a vector; however, most geoscience problems require inputs to be a vector. Examples of inputs to a neuron might be the pressure at the bottom of a well (Vaferi et al., 2015; Fath et al., 2020), gamma-ray log API values (Khandelwal and Singh, 2010; Ouadfeul and Aliouane, 2012; Kushwaha et al., 2020) or the darkness value of a pixel in a greyscale image (Asif and Choi, 2001; Ahmadi and Akbarizadeh, 2018). On every input, there is an associated weight that can be adjusted to tune the neuron in the training process. An activation function carries out the decision-making task in the neurons.

A bias shown as the vector $\mathbf{b}=\mathbf{w}_0$ in Figure 0-1 b is added to the weights term avoid the case where all the initial weights are zero and the model produces zero values irrespective of the weight. With this understanding, we represent the jth neuron mathematically as

$$z_j = \frac{1}{N}\sum_{n=1}^{N} w_{nj} x_n + w_{0j} \qquad 1\text{-}1,$$

where $\mathbf{z_j}$ is the output value, x_n are the N input values the values w_{nj}, n=1,2,…,N are the model weights and $\mathbf{w_{0j}}$ is the bias value. In general, the number of output values can be different than the number N of input data values. Equation 1-1 is a convolution of the input data **x** with weights **w** and is the fundamental equation that governs the operation of an artificial neuron.

Figure 0-2 shows multiple units of such neurons connected to form a layer of a network. The first layer that receives the inputs is called the input layer, the last layer that produces the output is called the output layer, and the layers in between are called the hidden layers. The name *hidden layers* suggest these layers never see the initial input (I) or the final output (O). In other words, the layers are "hidden" from the I/O operation. Any network with three or more hidden layers is called a “deep network." There is already one input and one output layer, such that a "deep" network must have at least five layers of neurons to be classified as deep. The neurons are fully connected to each other and network such as in Figure 0-2 is called a densely connected neural network (DNN). The neurons might not always be fully connected for all the architectures such as a convolutional layer (Shi et al., 2016) in the CNN architecture discussed later in this chapter.

Figure 0-3 shows a sigmoid activation function. Notice the values of the sigmoid function (Vail and Wornardt Jr, 1991; Wanto et al., 2017) ranges between zero and one. Hence, the output

is TRUE, i.e., if the value of the output produced by the network approaches 1, and FALSE if the output approaches zero. The sigmoid activation function shown in Figure 1-3 is represented as:

$$\varphi(z_j) = \frac{1}{1+e^{-z_j}} \qquad 1\text{- }2,$$

where $\mathbf{z_j}$ is computed from Equation 1-1 and is a function of inputs **x** and the weight and the biases of the model. A more complete list of common activation functions is summarized in Figure 0-4.

Just like any other statistical model, neural networks must be able to compare the results of the model to the true values. Let's assume the true values are denoted by **y** and the neuron's prediction, **z,** via the activation function is

$$a_j = \varphi\left(z_j\right). \qquad 1\text{- }3.$$

For a simplest case such as in Figure 0-1b with one neuron, the output a_j is compared with the true value **y** to generate a measurement of the prediction error.

For the case shown in Figure 0-5, where *L* layers of neurons are available, equation 1-4 can be generalized. The output from an activation function for the l^{th} layer is $\mathbf{a}^{(l)}=\varphi\mathbf{[z}^{(l)}]$ where the vector $\mathbf{z}^{(l)}$ are the predicted values from the (l-1)th layer. If the true value is **y,** then the error for each term is

$$\varepsilon_j \equiv y_j - a_j^{(L)}$$

and the mean squared error (Zhang et al., 2019b) can be written as:

$$E = \frac{1}{N(L)} \sum_{j=1}^{N(L)} \varepsilon_j^2 \qquad . \qquad 1\text{- }4,$$

Note that in equation 1-5 that larger errors are penalized more as the error terms are squared, resulting in a relatively slow learning process. Instead of minimizing the mean squared error, we can minimize the cross-entropy (Qin et al., 2019; Bosman et al., 2020) which uses a logarithmic metric. For a binary classification, the cross- entropy can be written as:

$$E_{(cros-entropy)} = -[y_j \log(p) + (1-y_j)\log(1-p_j)] \qquad 1\text{-}6$$

where, the model predicts a probability distribution $p(y=i)$ for each class i=1,2.

Although we have defined the number of layers and neurons, we have not yet determined the values of the weights, $\mathbf{w}^{(l)}$. The optimum values of these weights will minimize squared error, cross entropy, or other "cost function". To obtain these values we use the backpropagation technique, drawn from basic calculus.

The cost function and computed errors between the actual value and the output layer depend on the model parameters. From equation 1-1, the model parameters are simply weights and biases on each neuron in each layer. The output is more sensitive to some weights and biases than to others. To quantify how sensitive, the cost function E is to the weights and biases, we compute the first partial derivative of the cost function with each element of the vector $\mathbf{w}$. Figure 1-5 shows a simplistic neural network with L layers numbered from 1 through L with one neuron for each layer. The output at the first layer is shown in equation 1-3 as $\varphi^{(1)}[\mathbf{z}^{(1)}]$. Similarly, the final output at the output layer is simply $\varphi^{(L)}[\mathbf{z}^{(L)}]$ Hence the partial derivative of cost function E with respect to the weights and biases $\mathbf{w}$ can be written as:

$$\frac{\partial E}{\partial w_n^{(L)}} \quad . \qquad 1\text{-}7$$

Equation 1-7 can be expanded using the chain rule to be a function of the variables at layer L-1. The layer L-1 is a function of the variables at layer L-2 and so on, back propagating the gradient through the layers to the first layer which has the original data **x** as input. Once, a set of weights and biases are obtained that minimizes the error, they are adjusted for the next training step.

A single iteration from the input with the adjustment of the weights and the biases to produce a new estimate of the output is called a training "epoch". It is common to plot the mean squared error (MSE) of all the training samples at each epoch to evaluate the performance of the neural network process Analyzing these errors allows the data analyst to fine tune the model parameters such as the number of neurons for each layer, type of activation function used, and so on.

We now derive the full formulation for the backpropagation for a general case in summation notation which is easy to understand and implement in a python program if reader wishes to implement it separately. Majority of this derivation is modified from the Makin (2016). Interested readers can refer to Makin (2016) for more details.

Figure 1-13 depicts the general case of a MFNN architecture. The order of activation of the layer is from left to right and the order of error propagation is from right to left (backpropagation). The layers are named right to left for easy notation to derive backpropagation equations. In Figure 1-13, we show a fully connected network, but the derivation is completely generalized for a partially connected network or a dropout case. Notice in Figure 1-13, we show the indexing of the layers from right to left to explain backpropagation. All layers are referred to as i, j,k for indexing purpose. So, for example layer j has a neuron sitting at j^{th} position where j is a subset of total number of neurons in that layer shown by capital letter J. Hence, $j \in \{0,1,\ldots\ldots J\}$.

Similarly, i^{th} layer has a total of I neurons. The connection of one layer of neuron to the another is addressed as follows: As the layers may or may not be fully connected, let's say from one-layer *j* to another layer *i* there are total of K possible connections in a fully connected network. But, if there are only *j* number of neurons connected to the i^{th} layer, we can denote the number of connections as K_j.

With this in mind lets rewrite the output for the layer *i* using the inputs from layer *j* where K_j neurons are connected from layer *j* to layer *i*. This can be written as:

$$x_j = \sum_{k \in K_j} w_{kj} z_k \qquad 0\text{-}8$$

where, w_{kj} is the weights from j^{th} layer to i^{th} layer for the connected neurons and z_k is the output from the sigmoid activation of *k* neurons in the j^{th} layer. This is to be summed over the *k* connections to include all the weights and the neurons.

There is an important derivation we will show that will come in handy during the following equations. The derivation is simply the derivative of the sigmoid function itself. The sigmoid function is given as:

$$\emptyset(z) = \frac{1}{1+e^{-z}} \quad . \qquad 0\text{-}9$$

Taking the first derivative of equation 1-9 with respect to z, we get,

$$\frac{d\emptyset}{dz} = \frac{1}{1+e^{-z}} \left(\frac{e^{-z}}{1+e^{-z}}\right), \qquad 0\text{-}10$$

equation 1-10 can be rearranged into

$$\frac{d\emptyset}{dz} = \emptyset(z)[1 - \emptyset(z)] \quad . \qquad 0\text{-}11$$

We will use the equation 1-11 further into the derivation.

Now let us define the error *E* which is the measure between the true value and the predicted value at layer j.

$$E = \frac{1}{2}\sum_{j=1}^{J}(t_j - z_j) \quad , \qquad \text{0-12}$$

where, t_j is the target value and z_j is the predicted value. In equation 1-12, we scale the error by dividing it by ½ as our error is a square error and the derivative of the error shall cancel the ½ scaling factor. This is a neat trick to keep the equations clean when we compute the derivative of the error.

The weight change for a node in layer *j* depends on the weights from *k* nodes connected from layer *i* to layer *j*. This can be expressed as a partial derivative with respect to weights as:

$$\Delta w_{kj} = -\alpha \frac{\partial E}{\partial w_{kj}} . \qquad \text{0-13}$$

The α in equation 1-13 is called a learning rate or learning parameter of a MFNN network.

Expanding equation 1-13 with the help of chain rule, we get,

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial z_i}\frac{\partial z_i}{\partial x_j}\frac{\partial x_j}{\partial w_{kj}} \qquad \text{0-14}$$

We know the term x_j is the output from the k^{th} layer into the j^{th} node. Hence, we can write the term $\frac{\partial x_j}{\partial w_{kj}}$ as simply the output from the layer *k* or z_k as:

$$\frac{\partial x_j}{\partial w_{kj}} = z_k. \qquad \text{0-15}$$

The first two terms are often lumped together as a single quantity and often referred to as the "error term" of the backpropagation and denoted by δ. In this case we can write it as:

$$\delta_j = -\frac{\partial E}{\partial z_j}\frac{\partial z_j}{\partial x_j}. \qquad \text{0-16}$$

where $\frac{\partial z_i}{\partial x_j}$ is simply the derivative of the output of sigmoid function. From equation 1-11, we know the derivative of the sigmoid function can be reduced in a compact form. Hence, we can write,

$$\frac{\partial z_j}{\partial x_j} = z_j(1 - z_j) \ . \qquad \text{0-17}$$

Now we focus our attention on the first term of equation 1-16 i.e., $\frac{\partial E}{\partial z_j}$. Here, we want to see how the error has propagated from the layer *j* to layer *i*. This requires another chain rule application which can be written as:

$$\frac{\partial E}{\partial z_j} = \sum_{i \in I_j} \frac{\partial E}{\partial z_i} \frac{\partial z_i}{\partial x_i} \frac{\partial x_i}{\partial z_j}. \qquad \text{0-18}$$

The first two terms in equation 1-17 are simply the partial derivatives from equation 1-16.The third term of the equation 1-18 i.e., $\frac{\partial x_i}{\partial z_j}$ is simply the derivative of the equation 1-8 or ,

$$\frac{\partial x_i}{\partial z_j} = w_{ji} \qquad . \qquad \text{0-19}$$

Substituting from equation 1-16 and equation 1-19, equation 1-18 reduces to,

$$\frac{\partial E}{\partial z_j} = \sum_{i \in I_j} \delta_i w_{ji} \qquad , \qquad \text{0-20}$$

such that equation 1-14 becomes

$$\frac{\partial E}{\partial w_{kj}} = -\sum_{i \in I_j} (\delta_i w_{ji})\, z_j(1 - z_j) z_k \ . \qquad \text{0-21}$$

Equation 1-21 is the summation notation for backpropagation. In compact form equation 1-21 can also be written as:

$$\frac{\partial E}{\partial w_{kj}} = -\delta_j z_k \qquad . \qquad \text{0-22}$$

Now we see how this can be implemented in a network with the help of simple formulae.

The key aspect of backpropagation is to determine how much weight changes much be applied on every neuron at each layer which is given as:

$$\Delta w_{kj}(n) = \alpha \delta_j z_k + \eta \Delta w_{kj}(n-1), \quad \text{0-23}$$

where, n is the iteration index also called epoch and is described before in this chapter , α is the learning rate which is usually kept between (0,1]. z_k is the activation at node in the first layer k, η is called the momentum and takes a real value between [0,1). δ_j is the error term described before associated with the node after the weight.

MFNN architecture

As the name suggests, multi-layer feed forward networks or MFNN are a layered set of neurons described in the previous section and shown in Figure 1-2. The term multi-layer refers to more than one layer and as the inputs in one-layer feeds to the next layer via the activation function or "feeding forward". MFNN is the simplest of all NN architectures and easiest to understand. Di et al. (2018) applied MFNN network for fault detection. Ross and Cole (2017) used MFNN along with other NN architectures for classification of seismic facies. Hart et al. (2000) applied the MFNN architecture for time to depth conversion using seismic attributes.

The MFNN architectures complexity and hence training times increases as the data size increases requiring more neurons for training or deeper networks. Hence, for two- and three-dimensional seismic interpretation applications, the use of MFNN is limited. Another class of NN architectures is more suitable for seismic data known as CNN and is discussed later in this chapter. The data used in this dissertation is time series only. Chapter 2 uses gamma ray well logs, Chapter 3 uses oil field pressure and rates at the well head and Chapter 4 uses electron fluxes. MFNN is effective in modeling time series data and hence included in this discussion. The data preparation for all architectures specific to time series data is discussed at the end of this chapter.

LSTM architecture

LSTM falls under the class of recurrent neural networks (RNN). The RNN are suitable particularly for sequences such as time series data. Some applications are forecasting reservoir pressure over time, time lapse seismic, financial stock prices over time and even natural language processing. RNNs are an excellent fit for any sequences which have some dependency on the data prior to the forecasting interval or the future values depend on the past history of the inputs.

A recurrence in a network is created by modifying the perceptron shown in Figure 0-1b. Figure 0-6 shows a neuron modified with a feedback loop. Notice the "unrolling" of the neuron in Figure 0-6. The neuron tries to produce the output at time t-1 and this output is fed back into the same neuron at time *t* and the output at time *t* is fed back into the same neuron as an input for epoch *t*+1 and so on. This is different from an “epoch” of training that refers to the one full training iteration for the whole architecture.

At each time step, the neuron receives the input from the current time and the input from previous time steps. The idea of creating the neurons with a feedback loop can now be extended to create layers of such neurons such as shown in Figure 0-7. Hence, a whole layer can now be unrolled as a function of the number iterations just like a single neuron. Because the output at any time step depends on all the previous epochs, the RNN now have a "memory" component to it achieved by "unrolling" them in time. To distinguish these neurons from a simple neuron such as the one used for a MFNN network, they are referred as "cells" instead of a neuron.

RNNs suffer from the phenomenon called vanishing gradients (Hochreiter, 1998a, 1998b). As discussed in the previous section, the neural networks update themselves by back propagating the gradient throughout the network and as the networks gets deeper, the effect of individual gradients become smaller and smaller. This problem in RNNs can be fixed by choosing the

appropriate activation function such as a leaky rectified linear unit (RELU) (Zhang et al., 2017) , gradient clipping (Zhang et al., 2019a; Chen et al., 2020) or batch normalization (Santurkar et al., 2018).

Another major issue with RNNs arise when the networks get deeper and hence when the network unrolls, where a part of the memory is lost due to the vanishing gradients. Hence, a robust network with long term memory is needed to accomplish this task. A more sophisticated network that can alleviate these issues by making some changes in the cell design called as a LSTM cell or simply an LSTM network.

Figure 0-8 shows a typical LSTM cell which consists of a number of gates in a LSTM cell. The gates decide what inputs and previous information to be retained at every time step and what information can be discarded to determine a "cell state". Then only the most important information is passed onto the next time step. Many such cells can be connected in series. The notation in Figure 0-8 can be broken down into multiple parts as follows: the LSTM cell at epoch *t* has the input at time epoch as $\mathbf{x}_t$ and the cell state from the previous epoch *t*-1 as $\mathbf{c}_t$-1.

The purpose of a LSTM cell is to take these inputs and convert them to the cell state at epoch *t* as $\mathbf{c}_t$ and the output for the next time step *t* as $\mathbf{h}_t$. The information to be discarded is decided by a "forget gate" denoted by $\mathbf{f}_t$. The forget gate accomplishes this by a sigmoid layer shown in the left part of the cell in Figure 0-8. As a sigmoid always outputs zero or one, the value one implies the information will be kept and the value zero implies the information will be discarded. Immediately following the forget gate there is another sigmoid layer and a hyperbolic tangent layer that decide on what information must be added to this cell to the next time step. Once its decided what new information must be added, the old cell state is now updated with the new cell state. The

new cell state is nothing but convolution of old cell state with the forget gate and the new information added.

These steps can be written in a set of equations as:

1. *Linear transformation of the old cell state with the forget gate with weights and biases on the forget gate layer as:*

$$\mathbf{f}_t = \emptyset(\mathbf{w}_f . [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f \quad , \qquad 1\text{- }24$$

where,

$\mathbf{h}_{t-1}$ is the output from the previous cell state,

$\mathbf{x}_t$ is the input at the current cell state,

$\mathbf{b}_f$ are the biases for the forget gate, and

$\mathbf{f}_t$ is the output from the forget gate.

2. *Adding new information at current cell state:*

This part of the cell contains a sigmoid layer and a hyperbolic tangent layer. The output from the sigmoid ($\mathbf{i}_t$) can be written as :

$$\mathbf{i}_t = \emptyset(\mathbf{w}_i . [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i \quad , \qquad 1\text{- }25$$

Similarly, the output $\tilde{\mathbf{c}}_t$ from the hyperbolic tangent layer can be written as ,

$$\tilde{\mathbf{c}}_t = \emptyset(\mathbf{w}_i . [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i \quad . \qquad 1\text{- }26$$

3. *Determining the total information using old cell state($\boldsymbol{c}_{t-1}$) and the new cell state($\boldsymbol{c}_t$):*

$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + i_t * \tilde{\mathbf{c}}_t \text{ , and} \qquad 1\text{- }27$$

4. *The final output from the cell ($\boldsymbol{o}_t$) at time t:*

$$\mathbf{o}_t = \emptyset(\mathbf{w}_0.[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_0, \quad 1\text{-}28$$

and hence the current cell state update as

$$\mathbf{h}_t = \mathbf{o}_t * \tanh(\mathbf{c}_t) \quad . \quad 1\text{-}29$$

LSTM cells provide a unique advantage in geoscience problems, especially for dynamic properties such as reservoir pressure. As geological layers are deposited sequentially as per Steno's principle, predicting properties in advance is critical for geosciences. LSTM networks are suited to handle problems involving sequences. Zhong et al. (2019a) used a version of the LSTM and CNN for pressure prediction in the carbon sequestration approach using a reservoir simulator. Wang and Chen (2019) used the LSTM networks for pressure well test interpretation in tight reservoirs where longer shut-in times are often not an option due to low permeability of the reservoir. Xu et al.(2020) used LSTM network for production forecasting of the coal bed methane reservoirs which are also tight reservoirs. Song et al. (2020) provided the general framework of simultaneous prediction of rates and pressures using the LSTM networks.

Well logs such as gamma-ray (GR) logs indicate the stacking pattern of the geological sequences. The geological layers are deposited sequentially, and the GR log is a proxy for such sequences. It is intuitive to apply the LSTM networks to predict the following sequence of GR log if, for any reason, a section is missing on the log. Pham et al. (2020) demonstrated this approach for the prediction of missing logs by training an LSTM on previous log values. In extreme cases, where a full well log is missing in a well but, logs are available in the neighboring area, a synthetic log can be reconstructed using the logs from the nearby wells. Zhang et al.(2018) described one such case study in a tight reservoir by predicting missing lithology logs by borrowing lithology logs from the neighboring wells.

CNN architecture

Similar to the simple neuron of an MFNN network, the CNN is also inspired by the biological neuron, especially the vision aspect of the biological neuron. Figure 1-9 shows the cartoon, the conceptualization of the CNN with respect to the human vision analogy.

In Figure 1-9a notice that the visual cortex only sees a part of a complete image at any given time. A neuron is activated when it detects some unique feature, such as an edge. Hence, the neuron only perceives certain features from its surroundings, and the entire image is reconstructed in the brain from the features collected from small patches of the data. This observation implies that for a more extensive data set, identifying patterns can be achieved by performing computations on more minor sub-data set, dramatically decreasing the computing needs. The task of detecting the critical features is accomplished by filters called convolutional kernels. When convolved with the input, they produced a filtered image, and an activation function is only activated if an important feature is detected. LeCun et al. (1998) used this observation to the first CNN architecture in 1998 called LeNet (LeCun et al., 1998b), the first real-world application of one such network. Yann LeCun received the "Turing award" for this development in 2019.

Before introducing the structure of the CNN, a few terminologies are necessary to understand the structure of the CNN. We define some of these below as:

1. *Filter size:* A filter size is the number of parameters in a filter. In Figure 1-10, for a one-dimensional convolution, notice that the model uses only two weights w1 and w2, the filter size is two.
2. *Stride:* In Figure 1-10, the stride of the filter is two as the filter is shifted two neurons at a time.

3. *Pooling:* Figure 1-10 is a simplified one-dimensional convolution. However, often the CNN is used for two-dimensional and three-dimensional data. For images of larger sizes or just for a large number of images, it can still be computationally expensive to train a CNN model. Hence, some re-sampling or data decimation is required. The pooling layer accomplished this task of re-sampling. Thus, the pooling layer is just a step for data decimation.

4. *Dropout:* A dropout layer implies dropping random neurons to avoid reinforcing some of the neuron pathways to avoid overtraining. In this technique, to avoid overtraining the network, random neurons are dropped after every training epoch.

This idea of one-dimensional convolution can be extended to higher-dimensional data where each dimension serves as a channel of input. For an image, the height (*H*) and the width (*W*) depict two channels. If the image is colored, there is another channel to the input: color (*c*). These channels are passed onto convolutional filters, which detect various features in the data. Then data is decimated. Data decimation can is done before and after the feature extraction, and it is also not uncommon to add some DNN layers after the CNN network to improve the training results. Selection of such choices like the placement of decimation layer falls under the architecture design and depends on the problem statement. Often, tailor-made architectures are required to solve a specific problem. Figure 1-11 shows the famous CNN diagram, which is now very straightforward to understand. An input or input channel is fed into multiple convolutional filters, decimated and extracted features are often passed into dense layers in series.

Popular CNN architectures include ResNet (Targ et al., 2016), AlexNet (Krizhevsky et al., forthcoming), and Le-Net (LeCun et al., 1998b). All such networks have a typical CNN architecture such as stride, number of pooling layers, number of filters, type of filters, etc. Once the data is passed through these layers, a final classification of the data can be achieved on the

training samples. CNNs are a natural fit for geosciences as many problems in geosciences require pattern recognition in one, two or three dimensions. Pires de Lima et al. (2019) has used CNNs to classify core images into different lithofacies by segmenting the core images and then classifying them into different lithofacies. Duarte-Coronado et al. (2019) used the CNN to estimate porosity in thin sections. (Das et al., 2019) used the CNN for seismic impedance inversion. Xiong et al. (2018) applied the CNN networks to automate fault detection in seismic amplitude volumes. Sinha et al. (2020) used the CNN to automate the task of leak detection in carbon capture projects using field pressure data. Zhong et al. (2019b) applied CNN for leak detection by using time lapse seismic amplitude data.

CONV-LSTM

CONV-LSTM combines the features of CNN and the LSTM. The CNN can extract spatial features while LSTM can preserve the temporal aspect. One such analogy can be drawn from a finite difference reservoir simulator (Sinha and Devegowda, 2017). The reservoir simulator mimics the physics of fluid flow by using a geological model and then updates the model one step at a time.

Like coupling a CNN to an LSTM network, the CNN can also be coupled with any RNN network (Shih and Wu, 2020). Figure 1-11 shows one such network where the input is passed through a CNN first and then through LSTM before one training step is finished. This process is repeated over multiple training epochs. Shih and Wu (2020) have applied this methodology to create surrogate reservoir models for reservoir simulation.

The use of CONV-LSTM is not as popular as CNN or LSTM separately due to high computational requirements. Many CONV-LSTM practical applications require advanced networks such as RESNET (Targ et al., 2016). Advanced networks such as RESNET often require

parallelization on multiple GPUs, and such resources are not always available, and the results of CONV-LSTM versus CNN are marginally better (Sinha et al., 2020).

Time series data for deep learning

Classic time series models can outperform the deep learning architectures in many cases. Chapter 2 of this dissertation applies a combination of classical statistical methods that outperforms complex networks such as CNN. However, for multivariate time series analysis, depending on the complexity of the problem, sometimes it is necessary to use complex models.

This section presents the data preparation we use for multivariate time series analysis in Chapters 3 and 4 of this dissertation. There is a plethora of information available online to apply deep learning networks in 2D scenarios and some in 3D scenarios (Zhao and Mukhopadhyay, 2018; Duarte-Coronado et al., 2019; Pires de Lima et al., 2019). Lima (2019) demonstrated the process of data preparation, training and then testing of CNN for core lithofacies identification using two dimensional core images. Lima (2019) also summarized the workflow and the use of such networks and their constraints. Zhao (2018) proposed a workflow for fault detection in 3D seismic volumes by coupling CNN with Laplace of Gaussian(LOG) filtering to improve the automated fault detection. But little information available for the time series data to perform similar analysis in one dimension. The reason is simple. In many cases, just the statistical methods will outperform a deep learning network; the complex architectures are considered an "overkill" for the time series data. However, every day new data is being acquired in digital form with simple devices such as a phone or a digital watch on personal health in medical sciences (Huang and Kinser, 2002; Al Rahhal et al., 2016). More emphasis is being laid to one-dimensional data such as well logs in the oil and gas industry, thus driving deep learning architectures.

It is important to first prepare our data for the supervised learning task. This task is done by windowing the time series data into different sets of windows to predict the following sample or a set of samples. The data for the CNN and LSTM networks requires an additional step of a number of steps (timesteps) in most architectures such as Keras because the architectures are primarily developed for image processing. Hence, we need to" trick" the architectures for time series forecasting. An input to a deep learning network consists of [samples, timesteps, features]. The sample is just the number of data points in a sequence like time series. The time step is one window of observation, and features are the number of columns is the number of features. Hence, one data window is a one-time step, and then the window is slid forward one sample at a time. We implement this using a separate function and reshape our time series into a windowed time series in the form [samples, timesteps, features]. The window size selection depends on the data available for training and the size of the features. For a large dataset, large window size is appropriate. For a small dataset, small window size is a necessity. An analogy can be drawn from the seismic resolution and sampling rate. A higher sampling rate is preferred but not always possible. Larger window size is preferred, but it does not leave many training steps to train the network. Hence, the window size versus the network complexity is a tradeoff that can be decided by iterating various window sizes.

Conclusions

This chapter discusses the basics of the deep learning architectures necessary for Chapters 3 and 4 of this dissertation. I discuss a total of four deep learning architectures, including MFNN, CNN, LSTM, and CONV-LSTM. We also discuss the data preparation for time series data for these architectures and the architecture design.

Figures for Chapter 1

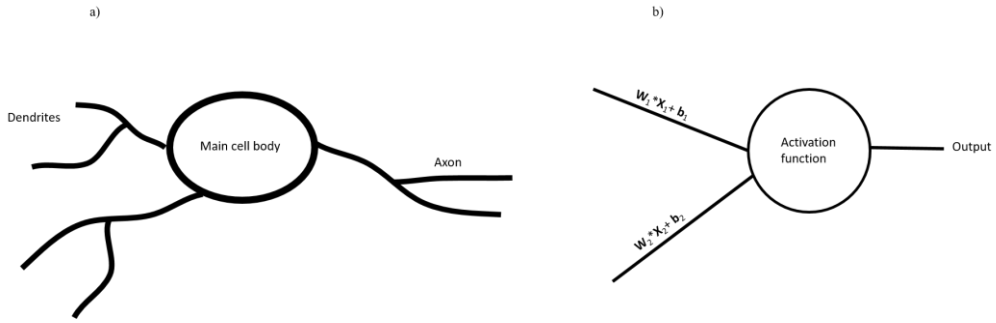


Figure 0-1 a) A biological neuron b) A computational neuron/ perceptron. The biological neuron takes input from the environment. The inputs are processed in main cell body and an output signal is communicated to the brain. A perceptron takes similar inputs, the inputs are processed by an activation function and an output is produced.

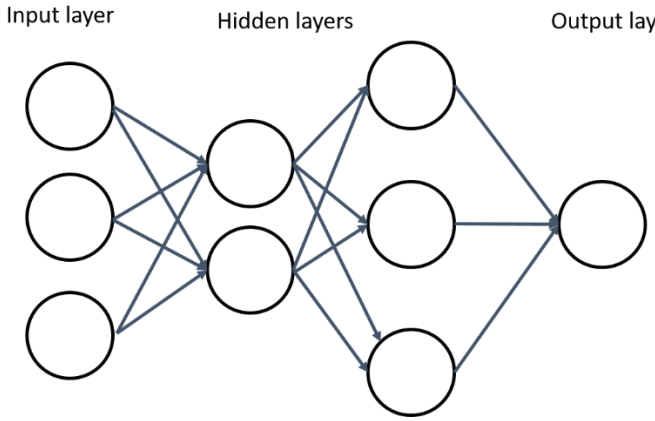


Figure 0-2 The multi layered neurons showing the input layer , the output layer and the hidden layers in-between. To qualify as a deep network a network must have at least three hidden layers. Notice that the hidden layers never see the input(I)/output(O) operations and hence *hidden* from the I/O operation and called as such.

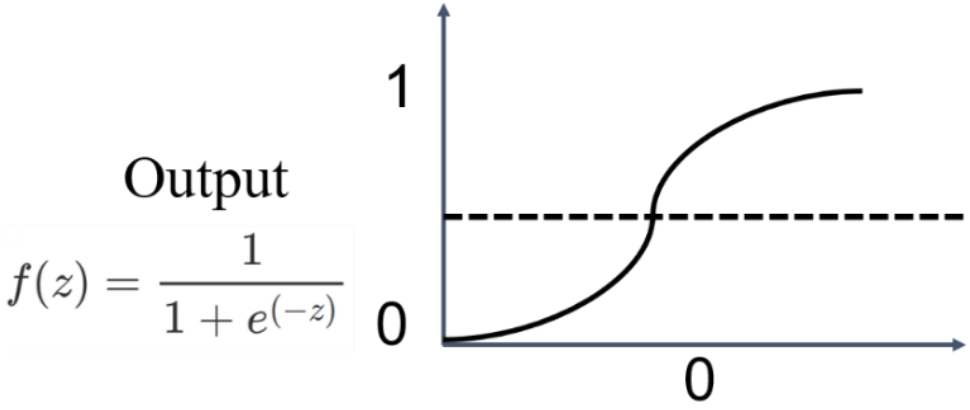


Figure 0-3 The sigmoid activation function that takes z= wx+b as an input and generates an output for the neuron . The output is bounded between 0 and 1 using a smooth function called the sigmoid function depicted by f(z).

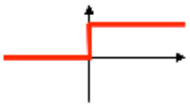
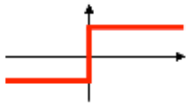
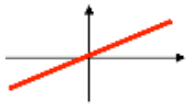
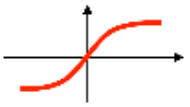
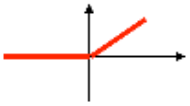
Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	

Figure 0-4 Most common activation functions used for classification and regression problems (Raschka, 2015).

$$\mathbf{z^L = w^L a^{L-1} + b^L}$$

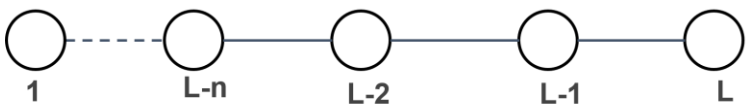


Figure 0-5 The generalized case of L layer neural network. For simplicity only one neuron per layer is used. The activation function takes the input **z** and produces an output **a**.

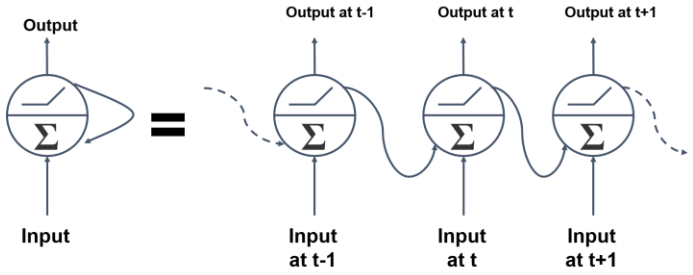


Figure 0-6 Unrolling of a neuron (Portilla, 2021). The neuron at every time step takes inputs from all previous time steps creating the memory feature in RNN networks.

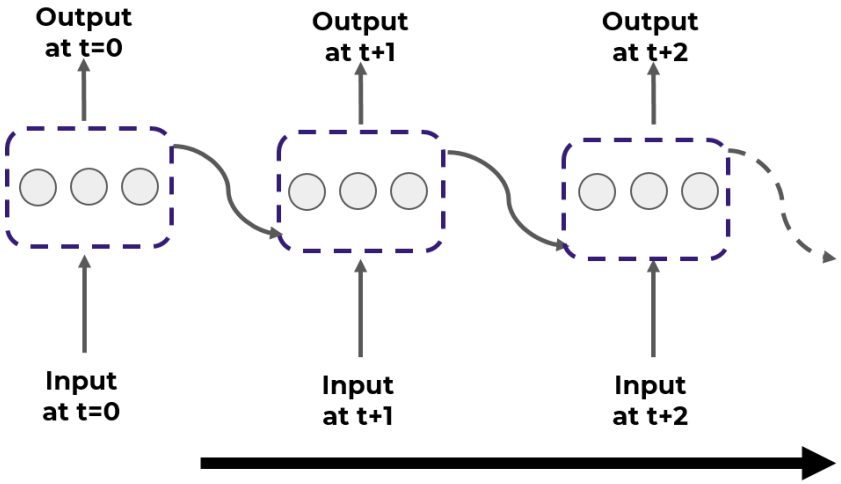


Figure 0-7 Unrolled layer of neurons in series (Portilla, 2021).

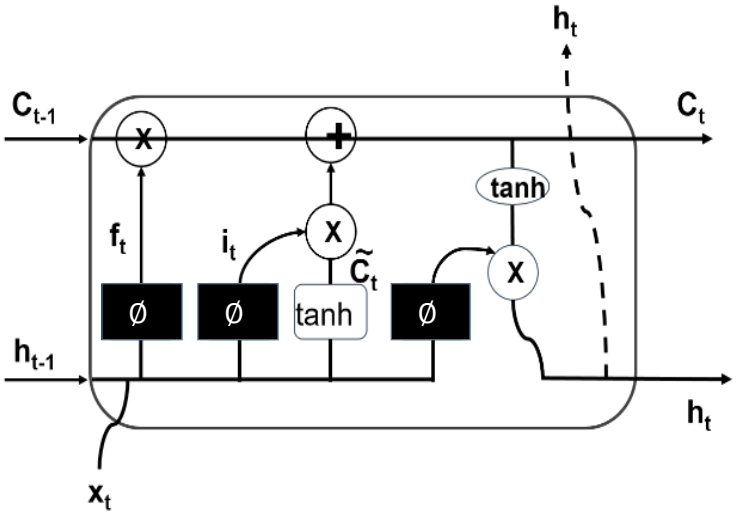


Figure 0-8 An LSTM cell showing the input, forget and remember gates (Portilla, 2021).

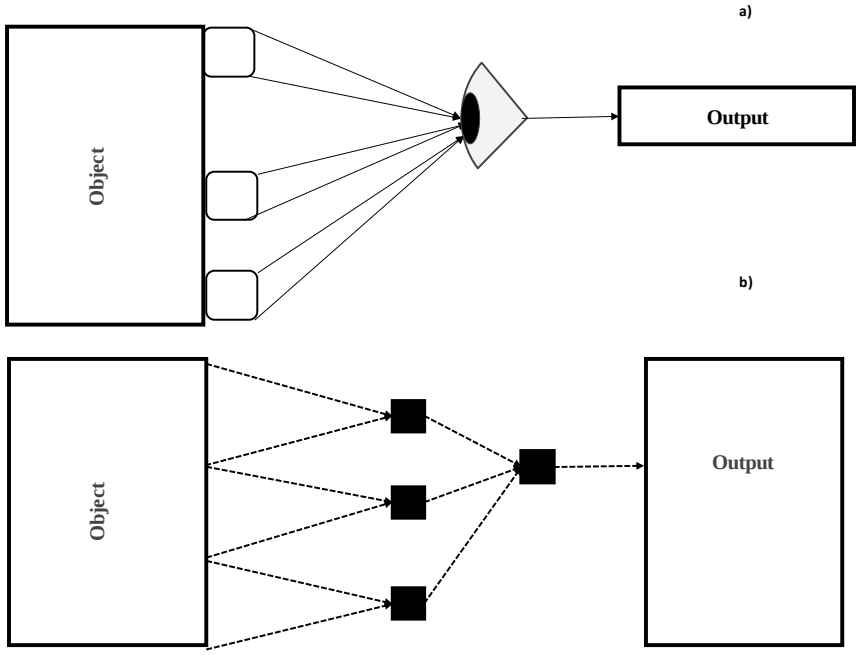


Figure 0-9 Conceptualization of a biological neuron using the visual cortex and a CNN equivalent. The object is read into patches of data and a neuron is only activated if certain features such as an edge is detected. The full image of the object is then created in the brain by using sparse data of such features. This is implemented in an artificial CNN by using convolution filters and the activation functions. Activation functions are only activated if a specific feature is detected.

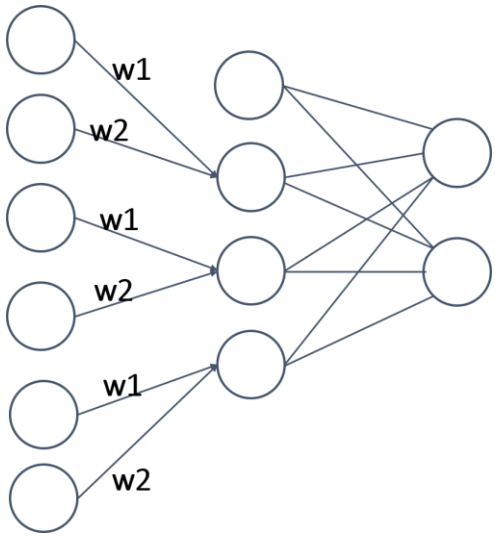


Figure 0-10 A simplified version of a 1D convolution. A single filter is shown with a filter size of two and a stride of two.

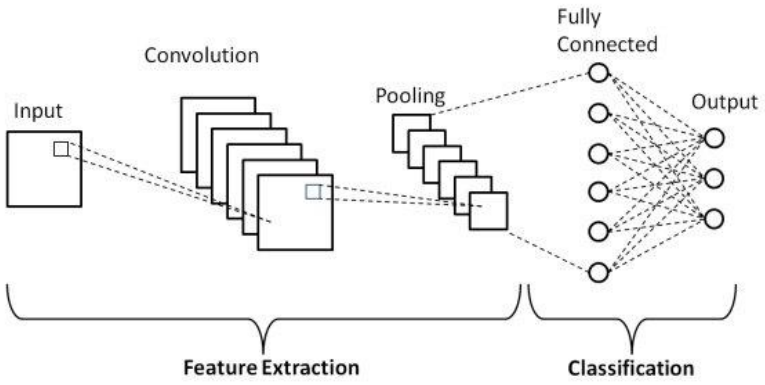


Figure 0-11 The simplified CNN architecture (Phung and Rhee, 2018).

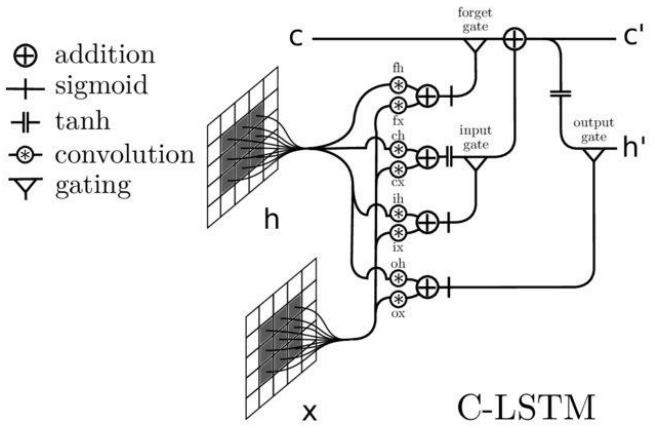


Figure 0-12 CONV-LSTM architecture. (Phung and Rhee, 2018).

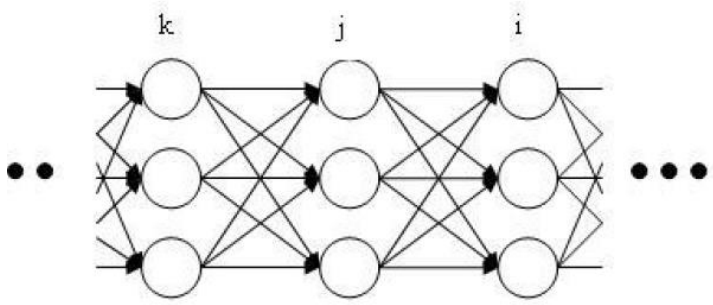


Figure 0-13 Generalized MFNN architecture with indices used for backpropagation derivation (Makin, 2016).

References

Al Rahhal, M. M., Y. Bazi, H. AlHichri, N. Alajlan, F. Melgani, and R. R. Yager, 2016, Deep learning approach for active classification of electrocardiogram signals: Information Sciences, **345**, 340–354.

Bosman, A. S., A. Engelbrecht, and M. Helbig, 2020, Visualising basins of attraction for the cross-entropy and the squared error neural network loss functions: Neurocomputing, **400**, 113–136.

Chen, X., S. Z. Wu, and M. Hong, 2020, Understanding gradient clipping in private SGD: a geometric perspective: Advances in Neural Information Processing Systems, **33**.

Das, V., A. Pollack, U. Wollner, and T. Mukerji, 2019, Convolutional neural network for seismic impedance inversion: Geophysics, **84**, R869–R880.

Di, H., M. Shafiq, and G. AlRegib, 2018, Patch-level MLP classification for improved fault detection, *in* SEG Technical Program Expanded Abstracts 2018, Society of Exploration Geophysicists, 2211–2215.

Duarte-Coronado, D., J. Tellez-Rodriguez, R. Pires de Lima, K. Marfurt, and R. Slatt, 2019, Deep convolutional neural networks as an estimator of porosity in thin-section images for unconventional reservoirs, *in* SEG Technical Program Expanded Abstracts 2019, Society of Exploration Geophysicists, 3181–3184.

Hart, D., R. Balch, W. Weiss, and S. Wo, 2000, Time-to-Depth Conversion of Nash Draw L Seismic Horizon using Seismic Attributes and Neural Networks: SPE Permian Basin Oil and Gas Recovery Conference.

Hochreiter, S., 1998a, The vanishing gradient problem during learning recurrent neural nets and problem solutions: International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, **6**, 107–116.

Hochreiter, S., 1998b, Recurrent neural net learning and vanishing gradient: International Journal Of Uncertainity, Fuzziness and Knowledge-Based Systems, **6**, 107–116.

Horn, R. A., 1990, The hadamard product: Proc. Symp. Appl. Math, **40**, 87–169.

Huang, B., and W. Kinsner, 2002, ECG frame classification using dynamic time warping: IEEE CCECE2002. Canadian Conference on Electrical and Computer Engineering. Conference Proceedings (Cat. No. 02CH37373), **2**, 1105–1110.

Krizhevsky, A., I. Sutskever, and G. E. Hinton, Forthcoming ImageNet Classification with Deep Convolutional Neural Networks (AlexNet).

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner, 1998a, Gradient-based learning applied to document recognition: Proceedings of the IEEE, **86**, 2278–2324.

Pham, N., X. Wu, and E. Zabihi Naeini, 2020, Missing well log prediction using convolutional long short-term memory network: Geophysics, **85**, WA159–WA171.

Phung, V. H., and E. J. Rhee, 2018, A deep learning approach for classification of cloud image patches on small datasets: Journal of Information and Communication Convergence Engineering, **16**, 173–178.

Pires de Lima, R., F. Suriamin, K. J. Marfurt, and M. J. Pranter, 2019, Convolutional neural networks as aid in core lithofacies classification: Interpretation, **7**, SF27–SF40.

Portilla Jose, 'Complete Guide to TensorFlow for Deep Learning with Python', Pierian Data Inc, 1 Oct 2021, https://courses.pieriandata.com/

Qin, Z., D. Kim, and T. Gedeon, 2019, Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator: ArXiv Preprint ArXiv:1911.10688.

Raschka, S., 2015, Python Machine Learning: Packt publishing ltd.

Ross, C. P., and D. M. Cole, 2017, A comparison of popular neural network facies-classification schemes: The Leading Edge, 36, 340–349.

Santurkar, S., D. Tsipras, A. Ilyas, and A. Mądry, 2018, How does batch normalization help optimization? Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2488–2498.

Shi, W., J. Caballero, L. Theis, F. Huszar, A. Aitken, C. Ledig, and Z. Wang, 2016, Is the deconvolution layer the same as a convolutional layer? ArXiv Preprint ArXiv:1609.07009.

Shih, C. Y., and X. Wu, 2020, A CNN-RNN based machine learning model for carbon storage management: AGU Fall Meeting Abstracts, **2020**, H048-07.

Sinha, S., and D. Devegowda, 2017, Quantification of Recovery Factors in Downspaced Shale Wells: Application of a Fully Coupled Geomechanical EOS compositional Simulator: Unconventional Resources Technology Conference, Austin, Texas, 24-26 July 2017, 3342–3354.

Sinha, S., R. P. de Lima, Y. Lin, A. Y. Sun, N. Symons, R. Pawar, and G. Guthrie, 2020, Normal or abnormal? Machine learning for the leakage detection in carbon sequestration projects using pressure field data: International Journal of Greenhouse Gas Control, **103**, 103189.

Song, X., Y. Liu, L. Xue, J. Wang, J. Zhang, J. Wang, L. Jiang, and Z. Cheng, 2020, Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model: Journal of Petroleum Science and Engineering, **186**, 106682.

Targ, S., D. Almeida, and K. Lyman, 2016, Resnet in resnet: Generalizing residual architectures: ArXiv Preprint ArXiv:1603.08029.

Vail, P. R., and W. Wornardt Jr, 1991, An integrated approach to exploration and development in the 90s: Well log-seismic sequence stratigraphy analysis: .

Wang, S., and S. Chen, 2019, Application of the long short-term memory networks for well-testing data interpretation in tight reservoirs: Journal of Petroleum Science and Engineering, **183**, 106391.

Wanto, A., A. P. Windarto, D. Hartama, and I. Parlina, 2017, Use of binary sigmoid function and linear identity in artificial neural networks for forecasting population density: IJISTECH (International Journal of Information System & Technology), **1**, 43–54.

Xiong, W., X. Ji, Y. Ma, Y. Wang, N. M. AlBinHassan, M. N. Ali, and Y. Luo, 2018, Seismic fault detection with convolutional neural network: Geophysics, **83**, O97–O103.

Xu, X., X. Rui, Y. Fan, T. Yu, and Y. Ju, 2020, Forecasting of coalbed methane daily production based on T-LSTM neural networks: Symmetry, **12**, 861.

Zhang, D., C. Yuntian, and M. Jin, 2018, Synthetic well logs generation via Recurrent Neural Networks: Petroleum Exploration and Development, **45**, 629–639.

Zhang, J., T. He, S. Sra, and A. Jadbabaie, 2019a, why gradient clipping accelerates training: A theoretical justification for adaptivity: ArXiv Preprint ArXiv:1905.11881.

Zhang, N., S.-L. Shen, A. Zhou, and Y.-S. Xu, 2019b, Investigation on performance of neural networks using quadratic relative error cost function: IEEE Access, **7**, 106642–106652.

Zhang, X., Y. Zou, and W. Shi, 2017, Dilated convolution neural network with LeakyReLU for environmental sound classification: 2017 22nd International Conference on Digital Signal Processing (DSP), 1–5.

Zhao, T., and P. Mukhopadhyay, 2018, A fault detection workflow using deep learning and image processing: 2018 SEG International Exposition and Annual Meeting.

Zhong, Z., T. R. Carr, X. Wu, and G. Wang, 2019a, Application of a convolutional neural network in permeability prediction: A case study in the Jacksonburg-Stringtown oil field, West Virginia, USA: Geophysics, **84**, B363–B373.

Zhong, Z., A. Y. Sun, Q. Yang, and Q. Ouyang, 2019b, A deep learning approach to anomaly detection in geological carbon sequestration sites using pressure measurements: Journal of Hydrology, **573**, 885–894.

Chapter 2 : Semi-Supervised well log correlation

Saurabh Sinha[1], Kurt J Marfurt [1], Rafael Pires De Lima[2], Sumit Verma[3], Thang Ha[1], Javier Tellez[4]

[1]School of Geosciences, The University of Oklahoma, Norman, OK, USA

[2]Geological Survey of Brazil, São Paulo, Brazil

[3] University of Texas at Permian Basin, Odessa, TX, USA

[4] Colorado Mesa University, Grand Junction, CO, USA

Preface

This chapter is presented as submitted to the AAPG-SEG journal *Interpretation,* and is based on previously presented SEG and the URTEC expanded abstracts (Sinha et al., 2018; Sinha et al., 2019). This chapter shows the semi supervised automatic identification and quantification of the parasequences using changepoint analysis and the dynamic time warping algorithms.

References

Sinha, S., R. Kiran, J. Tellez, and K. Marfurt, 2019, Identification and Quantification of Parasequences Using Expectation Maximization Filter: Defining Well Log Attributes for Reservoir Characterization: Unconventional Resources Technology Conference, Denver, Colorado, 22-24 July 2019, 62–73.

Sinha, S., R. P. de Lima, J. Qi, L. Infante-Paez, and K. Marfurt, 2018, Well-log attributes to map upward-fining and upward-coarsening parasequences: 2018 SEG International Exposition and Annual Meeting.

Abstract

Lithological correlation is an essential part of lithostratigraphy and has several practical applications, such as reconstructing layering patterns in sedimentary basins, and developing a sequence stratigraphic framework for the shale plays. Typically, the indirect lithostratigraphic correlations are performed with data acquired in wells such as gamma ray, density, sonic, and resistivity logs. With recent advances in hydrocarbon shale plays and the digitization of legacy data, thousands of well logs are now available for human interpretation. Performing lithostratigraphic correlation on all these logs by a human interpreter is economically impractical. Hence, more often than not, most technical or economic decisions in shale plays are guided by "factory-made" field development plans, where all wells are completed with the exact pre-designed drilling and completion plans. We show a workflow to automate the process of lithostratigraphic correlation and then quantify the correlated sequences using "well log attributes.". We divide our workflow into three significant parts. We first segment the well logs into lithostratigraphic units using a maximum likelihood-based segmentation technique called changepoint analysis. Then, we correlate the units automatically from one well to another using the dynamic time warping algorithm. After the units are correlated, we use linear fits to show automatically identified upward fining and upward coarsening lithological units. We plot the slope of these trends as well log attributes develop them on a three-dimensional model. We show data from an unconventional and a conventional play to demonstrate our workflow. The results show high fidelity, reliable well log correlations independent of the type of hydrocarbon play in a fraction of the time required by a human interpreter. Faster identification and correlation of the units can help critical decision-making for plays with higher data volumes such as shales. Our three-dimensional model shows the gradients of the lithological units that can be useful for depositional system interpretation.

Introduction

Veeken (2013) defines lithostratigraphy, a subdivision of stratigraphy, as the systematic organization and description of rocks into distinctive units identified by the lithological characteristics of the rocks and their stratigraphic relations. Lithostratigraphic correlation is the process of connecting different lithostratigraphic units from one well to another. Once correlated, well lithostratigraphy can be used to generate two and three-dimensional stratigraphic models. These models are used in numerous geological workflows, from geo-cellular modeling to flow simulations. Additional details such as chronomarkers and biomarkers, the stratigraphic models provide a means to construct sequence stratigraphic frameworks.

Lithostratigraphy can be implemented using direct and indirect methods using Steno's principles (Byers, 1982; Kravitz, 2014). The direct method compares the lithology from one location to another, for example, using core samples or outcrop rocks, to construct a cross-section. Given the expense of coring, most geoscientists studying the subsurface use an indirect approach that correlates electrical well logs. The most common well log used for this purpose is a gamma-ray (GR) log that measures a formation's natural radioactivity in American Petroleum Institute (API) units along the well. The natural radioactivity of different formations is plotted against well depth and used to infer the stacking pattern in the formation. Similar lithofacies can also be obtained in a core GR and can be correlated from the one obtained in a well GR log. Singh (2008) correlated the core GR and defined them as gamma-ray parasequences (GRP) in the Barnett Shale. As our area of study is similar to Singh (2008), we use the same classification as Singh (2008).

Singh (2008) identified three distinct types of GRPs on the GR log and correlated them mineralogically with the thin sections from the core available in the area as:

1. Upward decreasing API interval: This trend is observed during the decrease in relative sea level or progradation during late stage lowstand systems tracts.

2. Upward increasing API interval: pattern is observed during gradual rise in relative sea level towards highstand systems tracts.

3. Constant API interval: Sea level remains stable during this period.

Singh (2008) identifies the GR rapid changes bounding these intervals as log-derived flooding surfaces and defines them as parasequences. Parasequences form a relatively conformable succession of genetically related beds or bed-sets bounded by marine-flooding surfaces conformal to the classic definition provided by Van Wagoner et al. (1988). Gamma rays provide only a partial measure of lithology; we follow this convention for the GR-derived patterns in this study and refer to these lithological sequences as gamma ray parasequences (GRP). This convention is held throughout this study.

Figure 2-1 from Slatt (2016) shows the general conceptualization of sequence stratigraphic framework with the help of a cartoon along with the GR response showing a lowstand systems tract (LST), transgressive systems tract (TST), and a highstand systems tract (HST). Figure 2-1 also shows the relationship between the depositional systems and the relative sea level. As the sea level falls to reach a low stand, the shoreline is exposed, eroded, and accompanied by more detrital sediments. In contrast, as the sea level increases to form a TSTs, the previously shallow area is now deeper and sees finer sediments deposited. The trend boundary between these two sediment tracts manifests itself as a spike in the GR response shown by the bold black curve. Figure 2-2 from Singh (2008) shows a similar analysis of a GR log measured from a Fort Worth Basin core and the correlated GRPs, the interpreted sea levels, mineralogy, the mineralogically based facies .

Note that the GRPs are closely related to the mineralogy and the depositional environment, suggesting that if the GRPs can be correctly identified and correlated for all the wells in a shale play that we can link the relative proportion of clay to other minerals through a depositional model.

With the development of shale resource plays, the number of recorded well log data has increased significantly, with almost all pilot wells and many lateral wells being logged each year. Simultaneously, a mammoth amount of legacy data is also continuously digitized, making a plethora of data available for geological analysis. Properties are sold and acquired each year, each of which may contain thousands of wells with logs that are either un-interpreted, or previously interpreted using a different geologic model than the new property owner. Interpreting these well logs before additional wells are drilled or critical technical or economic decisions are made becomes requires significant interpretation manpower. Analysis like the ones in Figure 2-1 and Figure 2-2 over the whole shale play are often skipped in favor of simpler "factory-made" drilling and completion plan.

Let’s assume the interpreter needs to locate two different boundaries (two picks) to identify a GRP, and that there are five such units per well. Thus, ten picks are required for the interpretation of a single well. Now, let’s assume the human interpreter can expertly pick one top in one minute and works eight hours in a day. The time needed to finish the interpretation of a varying number of wells is shown in Table 2-1.

More complete lithostratigraphic models of shale reservoirs can high grade well locations, minimizing the drilling and completion of less productive wells. Defining and interpolating GRPs for all the wells in the reservoir and correlating them to core is a key component in constructing such a lithostratigraphic model. We therefore need a workflow that can accelerate and automate GRP identification and correlation. A few attempts were made in this direction. Zhang et al.

(2019) used convolutional neural networks (CNN) to correlate 7,000 well logs from the Songliao Basin in northeast China. They found that their CNN-based workflow required extensive human interpretation to in the form of geologic cross-sections to properly construct the training data Because of the rapid lateral change of lithology in their fluvial reservoir, the prediction provided low, medium, and high accuracy results under different conditions.

Number of wells	Picks required	Total picks	Time (in minutes)	Time (in days)
10	10	100	100	0.3
100	10	1000	10000	~21
1000	10	10000	100000	~208
10000	10	100000	1000000	~2082

Table 2-1 Estimated time taken by a human interpreter for manual picking of well tops. Assuming five GRPs are interpreted, and two picks are required for each GRP using a linear time assumption and the human interpreter works eight hours in a day.

Figure 2-3 shows the same effort measured in number of days as the number of wells increase, where it can take almost 100,000 hours to interpret 1000 wells, which is an unacceptable time frame for any critical decision making.

As an alternative, we reevaluate using the more robust algorithm called dynamic time warping (DTW) used by in the past, but not without its shortcomings. For example, Wheeler(2015) used a DTW variant to correlate multiple wells using density logs. However, Wheeler's version suffers from constraints in the conjugate gradient method. Moreover, the density logs utilized by Wheeler are not as commonly acquired as gamma ray logs. Behdad (2019) applied DTW to aid in the interpretation of the well tops by plotting the power spectrum of continuous wavelet transform (CWT) to gamma ray logs. Unfortunately, the results require intensive post-processing and yield relatively subjective predictions. Fang et al. (2021) proposed a DTW workflow where they replaced the simple (Euclidean) absolute difference between the measurements in two wells difference b with a semblance metric. The workflow requires the manual interpretation of

reference wells to guide the correlations. As the number of wells grows, there is an increase in the number of reference wells needed for interpretation. Additionally, if multiple sequences are to be picked, all reference wells require manual interpretations for all the sequences.

We approach the problem of interpreting many well logs at a fundamental level and try to overcome the shortcomings from previous studies to take the workflow towards a more automated and reliable correlation stage. We first segment the wells into different lithostratigraphic units/ GRPs. We then search for the required unit only in different segments one by one. We hypothesize that for any algorithm such as DTW to provide high fidelity results, either there must be enough features to conclusively and accurately distinguish one segment from another, or the search space should be small.

We follow this introduction and motivation with a description of a three-part workflow. First, we segment all the well logs using a reference or "type well" into different GRPs. Second, we correlate the GRPs across all wells using a DTW algorithm. Third, we estimate the slope of the upward fining or upward coarsening GR log for each GRP using a least-squares fit. After having defined our workflow, we apply it to two data sets – an unconventional Barnett Shale play from the Fort Worth Basin, Texas, and a conventional fluvial deltaic play from the North Slope, Alaska. We discuss the advantages and limitations of our proposed workflow, followed by concluding observations. An appendices provides details of the changepoint analysis and the dynamic time warping (DTW) algorithm.

Workflow

Segmentation of an electric well log

The first step in the lithological correlation is to break down an electrical log into sub-sections where each section represents a unique GRP. In this study, we only use GR logs, and hence we use the terminology "GR log" instead of a more general electric well log throughout this manuscript. Nonetheless, a similar analysis can be performed on other one-dimensional well logs if the lithological units are well delineated and if the data are available in all the wells.

Figure 2-4 shows our overall workflow for the segmentation, which is the first step in cross-correlation. A well log trace is first de-spiked using a sliding window approach (Chu, 1995; Keogh et al., 2001). A fixed-length window with 90% overlap is slid from top to bottom of the well log trace. We then compute the median in this window. Any sample with a value larger than the value ***t*** times the median value is replaced by the median (m) computed in the window. The rejected data are stored as a well log attribute for analysis. Once the log is de-spiked and the rejected data are stored as an attribute, we use changepoint analysis to segment the well log (Killick et al., 2012; Killick and Eckley, 2014). A full formulation of the changepoint analysis is presented in appendix A.

Referring to Figure 4, a GR well log varying with depth z can be written as:

$$\boldsymbol{y} = \mathbf{GR}(\mathrm{z}) \tag{1}$$

The running window filtered output is then

$$f(z) = \begin{cases} y(z); & \text{if } GR(z) \leq tm \\ m\,, & \text{if } y(z) \geq tm \end{cases} \tag{2}$$

where, m is the computed median in the window and *t times m* is a threshold defined for what is a spike. Hence, if a spike is found greater than threshold times the median, the spike is replaced by

the median. Optimizing the window size is requires analyzing multiple parameters over several wells; however, once a window size has been selected, it can be used for all the wells in the area if the geology is fairly consistent.

The scale of the stratification present in the reservoir provides a guide on the optimum window size. If the reservoir shows multiple small-scale sequences on the GR log, then a slightly smaller window size will better delineate the sequences. We found a threshold of 1.5 times the median to be a good starting point. Nevertheless, the threshold should be chosen based on the actual GR ray response and likely varies across different datasets. Similar to the window size, once the parameter *t* is selected, it can be kept constant for all the wells containing a similar stratigraphy.

McGonigle et al. (2021) review the more successful changepoint analysis algorithms. Chen and Gupta (2011) favor the parametric formulation of changepoint analysis, while Sanderson et al. (2010) prefer using a local wavelet for non-stationary time series. Cho and Fryzlewicz (2015) address multiple change-point detection for a higher dimensional time series consisting of multiple logs (e.g., gamma ray, density, and resistivity) in each well. We limit our discussion here to a maximum likelihood changepoint algorithm that we find effective in GR log segmentation. A full formulation of the changepoint analysis is presented in appendix A. Here we only present a brief idea of changepoint analysis. We use the formulation presented by Chen and Gupta (2011) for the detection in mean of the distribution using a likelihood approach. Changepoint analysis is a statistical that analyses a null hypothesis against an alternative hypothesis. We treat the GR log as a time series data where every sample is assumed to follow a normal distribution. A normal distribution is completely defined by a mean and a variance of the distribution. Introducing a changepoint in the time series divides the changepoint into two segments with different means but

same assumed variance. This builds the alternative hypothesis. We compute the maximum likelihood estimators (MLE) for null and alternative hypotheses. We use the MLE to compute maximum likelihood for the null and the alternative hypothesis. A ratio test for MLE such as a Wilk's test can be used to maximize the ML ratio to obtain the location of the changepoint.

When first changepoint is located, the location of the changepoint divides the GR log into two segments where the same analysis can be applied recursively until there are no more points left. This approach is called a binary segmentation (Cho and Fryzlewicz, 2015) . As number of changepoints increases the number of recursive loops required to obtain the changepoints also increase which can be alleviated by using a dynamic programming technique called Pruned Exact Linear Time (PELT) (Wambui et al., 2015). In this study we use only detect changes in mean and assume the variance/covariance to be the same with the maximum likelihood ratio test and PELT. A more exhaustive approach for multivariate formulation is presented in Appendix A. A formulation to include both changes in mean and variance is presented by Chen and Gupta (2011) and applied for a time series by Killick et al.(2012).

In this manuscript, we refer to these change points as GRP boundaries defined by Singh (2008) and discussed in the introduction section and could be interpreted as the flooding surfaces. We assume that these trend boundaries separate two GRP trends, such as upward increasing/decreasing API, which can be segmented at these boundaries. We use type well to tune the parameters to identify the GRPs of interest and then apply them to all the wells in the study area.

Cross Correlation using dynamic time warping (DTW)

In this section we only introduce the intuition behind using DTW as algorithm of choice for cross correlating the segmented sections from one well to another. Appendix B summarizes the detailed mathematical formulation for the DTW algorithm and can be referred to for greater details. The idea behind using well logs for well to well cross-correlation can be broken down into few simple steps as follows:

Step 1 (the cause): The geological layers are deposited over one another following Steno's principles during cyclical sea levels.

Step 2 (the effect): The vertical stacking pattern thus produced is of particular interest to geoscientists who want to incorporate this information for decision-making.

Step 3 (the inference): The stacking pattern can be inferred by discreetly sampling the natural radioactivity of these layers vertically, i.e., a GR log. Often, a particular formation or a set of formations exhibit unique characteristics in a GR log.

For example, a human interpreter first identifies a unique formation signature: a flooding surface (FS) right above a dominant carbonate deposition. If this feature is present in most of the wells in the area, this is correlated first in all the wells. After this step, the whole GR log trace is automatically segmented above and below this unique layer. In other words, having a reference reduces the search space for the human interpreter. The interpreter then picks the following prominent sequence and gradually picks smaller and smaller sequences reducing the search space at each step logically.

There are multiple challenges in mimicking this workflow, even for a segmented GR log to cross-correlate within different wells. For example, the rate of deposition of a GRP can vary

due to available accommodation space. This means the same lithology can be deposited rapidly or slowly depending on the area in the basin where the well is located. The effect of rapid deposition on a GR log is a "stretched" version of the reference GRP. In the area of slower deposition/limited accommodation space, the same GRP might be "squeezed". The same sequence might also be partially eroded or, in other words, may have fewer or more samples as compared to a type well.

Humans are extremely good at identifying patterns. Hence, for an algorithm to mimic it, the algorithm must be able to compensate for the stretch and squeeze and should also be able to look for a sequence with an unequal number of samples. The segmentation itself reduces the search space for the algorithm as now an algorithm has to search in a small portion of the GR log instead of a whole well log.

DTW provides the following advantages over other alternatives such as simple cross-correlations:

1. DTW does not require the same number of samples in two signals to be compared,
2. DTW is less affected by noise and spikes in the data, and
3. DTW can stretch and squeeze the signal for comparison.

The DTW algorithm consists of two sets of signals (Myers et al., 1980). A reference signal and a query/test signal. As the name suggests, the reference signal is a signal used as a "template" that is used for comparison against a new signal (query) that can be stretched and squeezed. A best match is obtained, and the "effort" required to obtain a best match is recorded as a distance metric which is a Euclidean distance in our case. The more similar are the reference and the query signal, the less effort is required for the match and hence lower value of the distance metric.

In our study, we first segment the well log into different GRPs in the type well. Similarly, all other wells in the play are segmented into different GRPs by borrowing segmentation parameters from the type well. A GRP to be identified is selected from the type well and acts as a reference signal. This signal is then compared to each well in the play, and a distance metric is obtained and plotted as an attribute alongside with the GR log. The segment with the lowest distance required for the match is then selected and highlighted. The exact process is then repeated over all GRPs.

Intuitively, we first select a more extensive section on the log, such as the whole lower and upper Barnett that can be easily correlated from well to well, as the first section to be identified. Once identified, we can easily pick smaller sections within the bigger section as the search space for the DTW is dramatically reduced to identify the subsequent sequences.

Well log attributes

Once the intervals are identified and cross-correlated, we normalize the data in every GRP using a Z-score transform. As the absolute value of API differs from well to well it is essential to normalize the data before obtaining any curve fitting. Once normalized, it is expected that the GRPs, which are bounded by the flooding surfaces/discontinuities and the fits, could be affected at the edges for the slope estimates. To overcome this challenge, we use a Random sample consensus (RANSAC) regressor instead of ordinary least-squares (OLS) regression. RANSAC regressor described by Fischler and Bolles (1981) is less sensitive to the outliers and hence provides accurate slope estimates in the presence of extreme values, i.e., flooding surfaces at both ends of GRPs in our case. After a RANSAC linear fit is obtained, we use this slope to quantify and

identify it as a second well log attribute. We focus on this well log attribute only in this study to interpret the depositional system.

Case studies

Barnett Shale example

Figure 2-5 shows the stratigraphy of Barnett shale and the major stratigraphic sections from Montgomery et al.(2005). Note in Figure 2-5 that the Barnett sits on top of Viola limestone. The Forestburg limestone runs NE- SW through the section, decreasing thickness from NE to SW, ultimately pinching towards SW. The Forestburg limestone is sandwiched between lower and upper Barnett in the NE section.

Figure 2-5 also shows the typical type log with the GR response of the key formations. The lower Barnett is the area of interest for most operators due to high total organic carbon (TOC) and optimal fracking conditions. The GRPs in both lower Barnett and upper Barnett are shown in Figure 2-2 from Singh (2008), identified and picked by an expert interpreter.

Figure 2-6 shows our area of study in the Barnett shale. We have selected 106 wells present in the area. The type log used in our study is shown in Figure 2-7 a) with major formation tops are identified with the help of grey arrows. A zoomed-in section of the lower section is also shown in Figure 2-7 b) with the manually interpreted GRPs color-coded as red and green. The red arrows indicate upward decreasing API, and the green arrows show upward increasing API. This convention is held throughout this study.

Figure 2-8 shows the segmentation results from the changepoint algorithm described in previous section. The segmentation can be performed with various segment lengths and can be adjusted to pick the appropriate GRPs required by the interpreter. One such example is demonstrated in Figure 2-8 panel b versus the panel c. In panel b, the minimum segmentation length is chosen as 100 samples, and in panel c the minimum segment length is chosen as 25 samples. With a sampling rate of 1 sample/ Ft, it implies we do not want to pick a GRP less than 100 Ft in panel b and 25 ft. in panel c.

Once a well log is segmented, we can fit the RANSAC regressor and compute the slopes. This is shown in Figure 2-9 for the type well. Notice that the GRPs are identified, segmented, fitted and a quantitative measure of the GRP gradient is also obtained in Figure 2-9. Panel a of Figure 2-9 shows the original type curve with the identified GRPs with a red curve. Panel b shows the automated upward increasing and upward decreasing API trends with the same color coding as Figure 2-2 by an expert interpreter. Panel c shows the respective slopes of the GRPs. After the logs are segmented and interpreted on the type log, we correlate these segments on all 106 wells in the study area. Once all the wells are segmented and interpreted, the interpreted GRPs need to be correlated to tie the analysis together. We do this using the DTW algorithm.

We discussed in previous section that DTW utilizes a reference signal and a query signal. A query signal is compared against a reference signal and the best match is obtained by stretching and squeezing the query signal. The effort required to stretch and squeeze is then logged as a Euclidean distance metric. Figure 2-10 shows this methodology where a random GRP segment is selected in a coarsely segmented type well. Figure 2-10 a) shows the segmented well log, and Figure 2-10 b) shows the GR segment highlighted in a grey box. This segment is then compared

to all segments in the type well, and the associated DTW distance is plotted in Figure 2-10 c). Notice that when the segment is compared to itself, it requires no adjustment, and hence the cost associated with it is zero which is expected and observed. To highlight the segment identified as the best fit, we use a simple filter on the DTW distance (Φ) where the minimum distance is set to one (TRUE) and the rest of the log is set to zero (FALSE). This is shown in panel d.

To test this methodology in a new well, we compare the reference signal from the type well to a test well which is first segmented with the exact same parameters as the type well and then queried against the reference signal segment by segment. The results are shown in Figure 2-11. Figure 2-11a shows the segmented test well log, and the DTW distance is shown in Figure 2.11b. Notice in panel b, the distance observed for the best match is non-zero. The highlighted match is shown in Figure 2-11 c. The DTW correctly identifies the segment in the test well log. The full correlation between the well logs is shown in Figure 2-12. Notice in Figure 2-12 the autocorrelated wells for one segment/GRP. Notice in Figure 2-12 the correlation is near perfect.

In Figure 2-11 and Figure 2-12, notice that as we are trying to pick a very small GRP segment over a complete well log and as the GRPs become more and more similar, it will be hard for the DTW algorithm to identify these GRPs correctly. Consider the difference between the DTW distance in the adjacent segments (the correct one and the one right above it) in Figure 2-11 is minimal.

Note that we show a small portion of the well log is shown in the image. There are approximately 10,000 samples per log, and this can lead to a common pitfall in the GRP cross-correlation where the DTW incorrectly identifies another segment in a test well. To overcome this issue, we use the following strategy: we first pick a big chunk of the lower and upper Barnett with

a distinctive Forestburg sandwiched between the two formations. This is shown in Figure 2-13 a). We pick a distinctive section from the type log consisting lower and upper Barnett, segment all the wells and pick this section first. We then repeat the process for Forestburg limestone (green), three more GRPs and the viola limestone (light blue). We do not pick all the small GRPs as they are not continuous in our area of study and are not present in all the wells. Once picked, we cross correlate using DTW and extract top and bottom of these formations. The top and bottom are then used in a commercial software to first build the surfaces that can then be used to construct three – dimensional GRP zone model where every zone is a GRP. This is intuitive and similar to the strategy adopted by a human interpreter, where an easily identifiable formation is first selected, and then other GRPs are picked by association. We then use the density porosity available from 16 wells in the area to create parasequences guided porosity volumes shown in lower right in Figure 2-13b).

Once a zone model is constructed, we use the slope estimate from RANSAC repressor in each interval for simple interpolate between the wells and show the results of these slopes in the area of interest. We repeat the process for all the zones bounded by these GRPs. We use one layer per zone and use sequential indicator simulation (SIS) for the interpolation between the wells. The final sloped for two GRP's is shown in Figure 2-14 a). Notice the rate of changes in the sequences for two parasequences over time. The images obtained can be used with other geological attributes such as seismic curvature, aberrancy and entropy to gain more details on the depositional settings. We plan to add more such attributes for quantitative analysis in the near future.

North slope Alaska example

We show our workflow in the previous section for an unconventional reservoir. To show that our workflow is independent of the reservoir type, we apply our methodology to a conventional reservoir example. In this example we use a type well and test wells used for exploration of Brookman sandstone reservoirs in the Nanshuk and Torok formation on the North slope Alaska (Bhattacharya et al., 2020). Nanshuk formation is a fluvial-deltaic shelf succession and Torok formation is its basinward equivalent. Both formations are genetically related and composed of clastic rocks. Figure 2-15 shows our workflow applied on the GR log data from (Bhattacharya et al., 2020). Figure 2-15a shows the formations of interest, i.e., Nanshuk and Torok picked by the expert interpreter and the upward increasing API and upward decreasing API intervals. In Figure 2-15, panel b is the machine-interpreted results which show similar results. In Panel b, it can be noticed that the machine can pick finer scale sequences along with the quantitative estimate of the slopes.

We pick a sequence in the test well similar to the Barnett example and extract a reference signal. We then repeat the process for a type well and then on a test well. The results are shown in Figure 2-16. Notice in Figure 2-16, panel a, the reference signal shows a zero cost for a match compared to itself, which is expected. In panel b, the same segment is identified on a random test well. As the number of wells is minimal (four wells in total), an analysis like the Barnett example cannot generate a three-dimensional GRP zone model. However, it can be seen in Figure 2-15 and Figure 2-16 that our workflow is independent of the type of reservoir.

Discussion

In this section we discuss some of the key aspects of the usage of our workflow in practical field applications.

The "semi" in semi-supervised

Our work is titled semi-supervised workflow, and it is a critical distinction from a fully automated workflow. The "semi" consists of the inputs and intuitions from an expert interpreter. Machines are extremely good at performing repetitive tasks with high accuracy. However, machines lack intuition like a human interpreter, and hence we have designed this workflow to incorporate the intuitions from a human interpreter, and machines can perform the repetitive tasks. Following are the instances where the "semi" part of our workflow is significant:

1. Selection of the type well: The type well is a representative well in the area and hence represents the major formations encountered in the area and are essential for the geoscientists. A type well must be an actual representation for the majority of the wells. Multiple type wells can be used if one such well is not available.

2. The machine must find a GRP: Once a GRP is selected, the algorithm must find it in all the wells. Hence, if a GRP is entirely absent from one of the wells, the algorithm will pick up the "next best thing" and will produce erroneous results. If such a scenario is encountered, the tops can be adjusted in commercial software as a post-processing step. This process is no different from an automated seismic horizon picking algorithm, where few bad picks are changed later in an interactive approach.

3. Selection of appropriate parameters: The segmentation workflow requires the human interpreter to adjust some input parameters such as the minimum number of samples, the regularization of the cost function, and others. A detailed understanding of the mathematical formulation is not essential for the expert interpreter. Still, playing with various parameters, an interpreter must choose the

formation they want to pick and the scale of the formation. The depositional cycles exhibit a fractal behavior, i.e., within a bigger sea-level change/cycle, there are multiple small-scale changes and hence different orders of deposition. To pick multiple orders of sequences, the same process can be repeated where a lower order cycle is picked first and then higher and higher order. The results can then be combined in most commercial software.

The stacking of technologies

In this workflow, we have stacked two technologies together to produce state of the art results. The first step is the segmentation, and second step is the DTW. The segmentation alone is useful for identification of the GRPs however these GRPs need to be correlated manually without DTW. DTW alone can be used for correlations but does not produce desired accuracy levels. We have identified it is because of lack of features to conclusively identify a formation in a large search space, i.e., a full well log. Segmentation with DTW stacked together become an extremely powerful tool.

Competing technologies such as CNNs might be an obvious answer for similar problems where multiple well logs are available. The authors have tried CNN and it failed to produce accurate results with only GR log. However, as number of logs increase, the accuracy also increases. End users of the technologies have to beware of these facts while picking one technology over the other. Simpler workflows such as segmentation with DTW can produce far superior results than more complex machine learning techniques such as CNN if used wisely.

Extending the analogy

In this study, we demonstrate of segmenting and correlating on a GR log. However, the workflow can be extended to any one-dimensional data such as any well log, daily production rates and pressure data. For non-stationary signals such as pressure data in injection mode a linear detrending may be required before segmenting but after the data is made stationary, similar process can be repeated to find and correlate patterns in one dimensional data.

Limitations and possible solutions

The workflow described in this study have certain limitations. In this section we discuss some of these and the workaround with the limitations of the workflow.

1. The area in which a parasequnces is missing: the workflow proposed assigns one parasequnces per well based on the value of the DTW distance. Hence, the parasequence must find it in every well. To resolve this issue, one can simply divide the area into different sectors. A type well can be selected for each area and similar process can be repeated. If the sequence is missing only in a few wells, a surface can be first interpolated with the tops and then error points can be deleted and surfaces can be interpolated again. This post processing step fits naturally in the current manual picking of the tops.
2. The area with missing sections or repeat sections: In faulted or folded areas respectively, the similar solution as bullet point one can be adopted. For faulted areas, the area of study can be divided on both sides of the folds and the analysis could be tied up with manual interpretation at the fault. For heavily folded areas, the algorithm is not very useful. It must find a unique match which could be any of the repeat section and that remains a limitation of this workflow. A workaround can be found if the repeat sections are far apart in the well

logs, the well logs can be sliced with a depth cutoff, the user can then pickup first section. Then similar analysis can be performed with other half of the well log.

3. In areas of complex geology and extremely small parasequnces: In areas where it is not possible to distinguish between two sequences or they are very close an expert interpreter has to come up with a strategy to pick up the sequences. For example, before picking smaller thickness sequences, it is important to pick bigger sequences to narrow down the search. In the bigger sequences, the smaller sequences can be picked easily. Intuitively in case of Barnett shale it is easier to pick a Forestburg limestone sandwiched between upper and lower Barnett. Similarly, any unique feature can be used to narrow down the search and it is up to the expert interpreter's judgement to design the picking strategy.

Conclusions

In this manuscript, we present a semi-supervised workflow to automate the identification and quantification of GRP in conventional and unconventional reservoirs. There is often just GR log available for analysis in all the wells in a study area, especially when the study includes legacy data. The search space with GR log only implies searching for a small unit with only one feature, i.e., a univariate time-series signal. This often results in erroneous and produces low to average accuracy irrespective of the algorithm used as the search space is too large to identify the correct unit conclusively. However, if a log is segmented first and each segment is compared, the workflow is much more effective. In our study we segment the GR log using changepoint analysis and then correlate them from one well to another using DTW algorithm. We show high fidelity, reproducible results that can be interpreted and visualized in two and three dimensions. We introduce the "well log attributes" analogous to seismic attributes for the quantitative estimation

of the GRP which may be helpful in the geological workflows such as interpretation of depositional environments. Our workflow is statistical and hence is independent of the type of reservoir.

Acknowledgments

We want to thank the Attribute-Assisted Seismic Processing and Interpretation (AASPI) consortium at the University of Oklahoma for funding this research. We would also like to thank Devon Energy for a license to use their Barnett Shale data and Schlumberger for their generous donation of the Petrel software for use in research and education.

Appendix A- Changepoint analysis

Changepoint analysis is a technique that provides a means to detect anomalous, discreet steps in an array of random variables. The algorithm detects hypothesized changepoints by applying a statistical test that measures changes in the distributional behavior of the random variables. For example, a normal distribution is completely defined by its mean and variance. Simple changepoint tests may then consist of detecting changes in the mean, in the variance or in both mean and variance. This same analysis can easily be extended to a multivariate case with detecting mean vectors and covariance matrices of multiple random variables, such as computing change points in 3-element triple-combo well logs.

Changepoint analysis evaluates measurements for or against the null hypothesis of there not being a significant change in the data. Instead of testing the two hypotheses separately, changepoint analysis computes the likelihood ratio of the null and alternative hypothesis using metrics such as the Shapiro–Wilk test or Hoteling's T^2 test (Sullivan, 2015).The maximum ratio between the null and the alternative hypothesis (Hotelling's T^2 metric) for each candidate

changepoint provides the changepoint location. If the value of this ratio exceeds a user-defined threshold, the changepoint is accepted.

Figure 2-17 shows a univariate sequence exhibiting a normal distribution. The random variables in Figure 2-17 are sampled in two stages. First a normal distribution is defined for all the points. Then a candidate changepoint is chosen, and the normal distribution computed for the points lying to the left and the points lying to the right of the changepoint. The statistics of these new sub-distributions are then compared to the statistics for the distribution as a whole. The visual changes in the points in Figure 2-17 are evident to a human observer. The task is to define a test that can effectively separate the two sets of sequences by placing a boundary between them.

For the gamma ray analysis analyzed in this paper, we find that comparing the means provides an excellent estimate of the changepoints. Chen and Gupta (2011) show how to detect changepoints using the variance and both mean and variance together. (Qi et al., 2015)found the inverse of the coefficient of variation, μ/σ to be an excellent metric to map changepoints in 3D windows of fixed size in the application of Kuwahara filters.

Detecting changes in the mean for scalar univariate distribution

As we are only detecting changes in the mean, we will assume the variance of the sub distributions to be that of the complete distribution. Hence the random variables can be assumed to be defined as $\mathbf{x}_m(\mu_k, \sigma^2)$, where k=2,3,…, J-1 and represents the data (x_1, x_2,,…,x_k) to the left and including sample k. The null hypothesis is that all such distributions exhibit the same mean or

$$H_0 = \mu_1 = \mu_2 = \mu_3 = \ldots\ldots\ldots\ldots = \mu_n, \qquad \text{A1}$$

If a changepoint exists at index *k*, then everything on the left of the changepoint has equal means and everything on the right side has another mean. Hence, for all the indices from 1 to *k* has same mean and from *k+1* to *J* has a different mean. This alternative hypothesis can be written as:

$H_1 = \mu_1 = \mu_2 = \mu_3 = \ldots = \mu_{k \neq} \mu_{k+1} \ldots = \mu_J,$ A2

Because we are assuming that the variance σ is same for all the candidate changepoints, let's assume $\sigma^2 = 1$ without the loss of generality.

Hence for hypothesis H_0 the likelihood function can be written as:

$$L_0(\mu) = \frac{1}{\sqrt{2\pi}} exp^{-\sum_{i=1}^{i=J} \frac{(x_i - \mu)^2}{2}}$$ A3

The maximum likelihood estimator (MLE) for μ in equation B3 is simply the mean given by:

$$\hat{\mu} = \bar{x} = \frac{1}{J} \sum_{j=1}^{J} x_j$$ A4

Under the alternative hypothesis H_1, the likelihood estimator is:

$$L_1(\mu_1, \mu_2) = \frac{1}{(\sqrt{2\pi})^J} exp^{-\{\frac{\left[\sum_{i=1}^{k} (x_j - \mu_1)^2 + \sum_{k+1}^{J} \left(x_j - \mu_J\right)^2\right]}{2}\}}$$ A5

Now the two MLE's lying to the left and right of the changepoint *k* can be written as:

$$\widehat{\mu_1} = \overline{x_k} = \frac{1}{J} \sum_{j=1}^{k} x_j, \text{ and}$$ A6

$$\widehat{\mu_J} = \overline{x_{J-k+1}} = \frac{1}{J-k} \sum_{j=k+1}^{J} x_j$$ A7

The sum of the squared errors can now be written for the null and the alternative hypothesis.

For the null hypothesis with the MLE estimator:

$$S = \sum_{j=1}^{J}(x_j - \bar{x})^2, \quad \text{A8}$$

whereas for a change point at *k*,

$$S_k = \sum_{i=1}^{J}(x_{ij} - \overline{x_k})^2 + \sum_{i=k+1}^{J}(x_i - \overline{x_{J-k}})^2 \quad \text{A9}$$

The likelihood procedure maximizing the likelihood of the null hypothesis over the alternative hypothesis:

$$U^2 = \max_{2\leq k\leq J-1}(S - S_k). \quad \text{A10}$$

Hawkings (1977) derived the exact test statistic for equation B10 as T_k = (S-S_k) given as:

$$T_k = \sqrt{\frac{J}{k(J-k)}}[\sum_{i=1}^{k}(x_i - \bar{x})^2]. \quad \text{A11}$$

The location of the changepoint is then determined as:

$$k = \arg(\max_{2\leq k\leq J-1}|T_k|).$$

In other words, a changepoint is detected when the value of S- S_k becomes significant. The threshold can be specified by a user on the significance value.

Detecting changes in the mean for vector multivariate distribution

For a multivariate problem, the variance needs to be replaced by a covariance matrix also referred to as auto-covariance matrix, dispersion matrix, variance matrix, or variance–covariance matrix in some of the statistics literature. Also, a new parameter M needs to be added that represents the number of elements for the random vector variable in a hyperdimensional space. For example, every depth point in a well log can have a GR log API value, a resistivity value and a neutron porosity value, such that M=3. Hence for J samples with M well logs, with a fixed covariance of C, the null hypothesis can be written as:

$H_0 = \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}_3 = \ldots\ldots\ldots\ldots = \boldsymbol{\mu}_n = \boldsymbol{\mu}$, A12

where the elements of the vector means are the means of each of the well logs.

Introducing a change point at index k, the alternative hypothesis can be written as:

$H_1 = \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}_3 = \ldots = \boldsymbol{\mu}_{k \neq} \boldsymbol{\mu}_{k+1} \ldots = \boldsymbol{\mu}_n$, A13

Under H_0 the likelihood function can be written as,

$$L_0(\boldsymbol{\mu}, \mathbf{C}) = (2\pi)^{-\frac{MJ}{2}} |\hat{\mathbf{C}}|^{-J/2} \exp\{\frac{-1}{2} \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})^{\mathrm{T}} (\mathbf{C}^{-1})(\mathbf{x}_i - \boldsymbol{\mu})\} \quad \text{A14}$$

The maximum likelihood estimate for the mean and the covariance is now

$$\hat{\boldsymbol{\mu}} = \bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{J} \mathbf{x}_i, \text{ and} \quad \text{A15}$$

$$\hat{\mathbf{C}} = \bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{J} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T \quad , \quad \text{A16}$$

the MLE for the null hypothesis is

$$L_0(\hat{\boldsymbol{\mu}}, \hat{\mathbf{C}}) = (2\pi)^{-\frac{mn}{2}} |\widehat{\boldsymbol{\Sigma}\mathbf{C}}|^{-J/2} \exp^{-MJ/2}, \quad \text{A17}$$

and the MLE for the alternative hypothesis is

$$L_1(\boldsymbol{\mu}_1, \boldsymbol{\mu}_n, \mathbf{C}) = (2\pi)^{-\frac{MJ}{2}} |\hat{\mathbf{C}}|^{-J/2} \exp \frac{-1}{2} \Big\{ [\sum_{i=1}^{k} (\mathbf{x}_i - \boldsymbol{\mu}_1)^{\mathrm{T}} (\mathbf{C}^{-1})(\mathbf{x}_i - \boldsymbol{\mu}_1)] + \sum_{i=k+1}^{J} \big(\mathbf{x}_i - \boldsymbol{\mu}_J\big)^{\mathrm{T}} (\mathbf{C}^{-1})\big(\mathbf{x}_i - \boldsymbol{\mu}_J\big)] \Big\}. \quad \text{A18}$$

The MLE for $\boldsymbol{\mu}_1, \boldsymbol{\mu}_n, \mathbf{C}$ *are now given as* ,

$$\widehat{\boldsymbol{\mu}_1} = \overline{\mathbf{x}_k} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{x}_i \quad , \quad \text{A19}$$

$$\widehat{\boldsymbol{\mu}_J} = \overline{\mathbf{x}_{J-k}} = \frac{1}{J-(k+1)} \sum_{i=k+1}^{J} \mathbf{x}_i, \quad \text{A20}$$

$$\hat{\mathbf{C}} = \frac{1}{n}\left[\sum_{i=1}^{k}(\mathbf{x}_i - \overline{\mathbf{x}_k})(\mathbf{x}_i - \overline{\mathbf{x}_k})^T + \sum_{i=k+1}^{J}\left(\mathbf{x}_i - \overline{\mathbf{x}_{J-k}}\right)\left(\mathbf{x}_i - \overline{\mathbf{x}_{J-k}}\right)^T\right] \quad . \qquad \text{A21}$$

The MLE under the alternative hypothesis can now be written as,

$$L_1\left(\widehat{\boldsymbol{\mu}_1}, \widehat{\boldsymbol{\mu}_J}, \hat{\mathbf{C}}\right) = (2\pi)^{-\frac{MJ}{2}} |\hat{\mathbf{C}}|^{-J/2} \exp^{-MJ/2} \qquad \text{A22}$$

Using Hotelling's T^2 test, the maximum likelihood ratio test for null versus the alternative hypothesis can be written as,

$$\lambda_k{}^2 = \mathbf{y}_k{}^T \mathbf{W^{-1}}_k \mathbf{y}_k \,, \qquad \text{A23}$$

where $\mathbf{y}_k$ is the standardized difference similar between the null and the alternative hypothesis computed as:

$$\mathbf{y}_k = \sqrt{\frac{k(J-k)}{J}}\left(\bar{\mathbf{x}}_{\boldsymbol{k}} - \bar{\mathbf{x}}_{\boldsymbol{J-k}}\right) \quad , \qquad \text{A24}$$

and the elements $\mathbf{w}_k$ are computed as,

$$\mathbf{w}_k = \frac{1}{J-2}\left[\sum_{i=1}^{k}\left((\mathbf{x}_i - \overline{\mathbf{x}_k})(\mathbf{x}_i - \overline{\mathbf{x}_k})^T\right) + \sum_{i=k+1}^{J}\left(\mathbf{x}_i - \overline{\mathbf{x}_{J-k}}\right)\left(\mathbf{x}_i - \overline{\mathbf{x}_{J-k}}\right)^T\right]. \text{B25}$$

The candidate changepoint k is obtained by maximizing the λ^2

$$k = \arg(\max_{1 \le k \le n-1} \lambda^2) \qquad \text{A26}$$

If the maximum value of λ^2 exceeds a threshold value c, a changepoint is detected. c is decided by the user for the significance level. This can be iterated to find the best set of picks.

Detecting changes in the mean of a gamma ray log

We follow the procedure for changepoint detection for GR log using the generalized formulation for the vectors using equation B26. The changepoint detection then reduces to following steps:

1. Read the GR log and the depth into memory.
2. Using the equations above, locate the most significant changepoint *k* for the array or sub array being analyzed.
3. If the changepoint is accepted as being significant divide the GR log is divided into two sub segments. If the changepoint is not accepted, this part of the well log is deemed to be continuous; proceed to any remaining segments that need to be analyzed.
4. Repeat the steps 2-3 for individual segments until either of the two conditions are reached:
    a. The segment array is empty i.e., the entire log is segmented or
    b. The minimum segment length is reached. This condition implies if the interpreter doesn't want to accept sequences less than a given minimum thickness.

Appendix B – dynamic time warping

Dynamic time warping or DTW is a process that stretches and squeezes a new time series to match as much as possible a baseline series. In our case using gamma ray logs, the data are measured in depth rather than time, and the baseline series is commonly called the type log for the formation of interest. The stretch and squeeze process is similar to the impedance log when matching well log synthetics to measured seismic data during a model-based inversion technique. A common everyday life example can be drawn from the farrier placing an iron horseshoe on the hoof of a horse. A piece of iron is heated and deformed on an anvil and is matched against the

specific hoof. If a suitable match is obtained, the process ends. However, if further adjustments are required, the iron horseshoe is heated again until it fits or a near-perfect fit for the horse hoof is obtained. The closer the shape of the horseshoe with the hoof to begin with, the easier it is for the farrier to match the shoe, and hence, fewer attempts of heating and reheating are required. Thus, the associated "cost" is low when the match is close and high when it is poor.

The process of DTW for one-dimensional data, such as well logs, requires a pair of one-dimensional data series. The first series, known as the reference signal, serves as a baseline to compare any other series known as a test or a query signal. In the horseshoe example, the hoof's shape can be serves as the reference signal and the shape of the horseshoe as the query signal. The query signal is stretched or squeezed to match the reference signal. The corresponding "effort" to match the signal is recorded as a cost metric that is nothing but a pointwise computed distance matrix. Although common metrics include Euclidean, Manhattan, and Mahalanobis distances, because we are comparing gamma ray logs to gamma ray logs in the same formation in a given oil field, the simple Euclidean distance is sufficient. Unlike many machine learning techniques, the advantage of DTW is that it doesn't require both the reference and the test time signal to be of equal size, This property of the DTW algorithm makes it a suitable algorithm in applications that require frequent comparisons between two signals of unequal lengths, such as occurs in speech recognition where some speakers speak slowly and others more rapidly (Rabiner et al., 1978).

Wei et al., (2006) find that for speech recognition, two individuals can pronounce the same word a little differently. The word "hello" and "hellooo" spoken by two individuals are essentially the exact words where the latter individual stresses on the alphabet "o." Although the lengths of the time series are different, when the second time series is squeezed, it closely resembles word hello. Wei et al. (2006) and Dhingra et al., (2013) serve as excellent references on this topic.

This appendix discusses the basic mathematical formulation of the DTW algorithm. A complete, more comprehensive formulation can be found in Rabiner et al. (1978) and Myers et al (1980). Figure B1 shows a typical warping problem from Rabiner et al. (1978).

Figure 2-18 shows the discrete reference signal $\mathbf{R}(n)$, $0 \leq n \leq N$ and test signal $\mathbf{T}(m)$, $0 \leq m \leq M$ where N_1 and N_2 are end points of the reference signal and M_1 and M_2 are the endpoints of the test signal. The objective of the time warping is to map the indices of the reference signal $\mathbf{R}(n)$ over the test signal $\mathbf{T}(m)$. This mapping is represented with the help of a warp function $\mathbf{w}$ defined as,

$$m = \mathbf{w}(n) \qquad \text{B1}$$

In the simplest warp case, we can assume that the endpoints of the reference signal and test signal match, thereby imposing a boundary condition. This assumption can be relaxed in some cases. Dhingra et al(2013) provide details on how to the boundary conditions. For our work, we have already delimited the candidate parasequence boundaries using equation A 11 such that the endpoints of both series match and can be written as:

$$M_1 = \mathbf{w}(N_1) \quad \text{, and} \qquad \text{B2}$$

$$M_2 = \mathbf{w}(N_2) \quad . \qquad \text{B3}$$

This type of boundary condition is called a constrained endpoint (CE). The warping function can now be constrained to satisfy the following set of continuity equations:

$$\mathbf{w}(n+1) - \mathbf{w}(n) = \begin{cases} 0,1,2 & \text{if } \mathbf{w}(n) \neq \mathbf{w}(n-1) \\ 1,2 & \text{if } \mathbf{w}(n) = \mathbf{w}(n-1) \end{cases} , \qquad \text{B4}$$

Equation B4 requires $\mathbf{w}(n)$ to be monotonically increasing, with a maximum slope of two, and a minimum slope of zero except in the case where preceding frame is zero. In the case where

preceding frame is zero, the slope is set to one. Using the boundary conditions in B2 and B3 and the continuity condition of B4, the warping function **w** now lies in a parallelogram in the (n, m) plane which is shown in Figure B2a. The vertices of this parallelogram can be obtained as:

Point A:

$$m - 1 = 2(n - 1),$$

$$m - M = (n - N)/2, \qquad \text{B5}$$

Point B:

$$m - M = (n - N)/2 \text{ , and}$$

$$m - M = (n - N) \qquad \text{B6}$$

Hence, the warp function **w**, is constrained to follow a path inside the shaded region shown in Figure B2a. The full function can be defined by a distance function **D**(n, m) for every pair of points (n, m). Given a distance function **D(**n, m**)**, the warping path **w**(n) is chosen to minimize the pairwise accumulated distance, or the sum of the all the distance pairs:

$$\mathbf{D}_T \equiv \min_{\mathbf{w}(n)} \sum_{n=1}^{N} \mathbf{D}[\mathbf{R}(n), \mathbf{T}(\mathbf{w}(n))] \qquad \text{B7}$$

We use dynamic programing to obtain the optimal warping path **w**(n) given by equation B7 The accumulated distance to any grid point can be written in a recursive manner as

$$\mathbf{D}_A(n, m) = \mathbf{D}(n, m) + \min_{q \le m} \mathbf{D}_\mathbf{A}(n - 1, q) \qquad \text{B8}$$

where, $\mathbf{D}_A$ is the optimized minimum distance to the grid point (m, n) inside the parallelepiped shown in Figure B2. $\mathbf{D}_A$ has the form

$$\mathbf{D}_A(n, m) = \sum_{p=1}^{n} \mathbf{D}\{[\mathbf{R}(p), \mathbf{T}[(\mathbf{w}(p))]\} \qquad \text{B9}$$

$\mathbf{D}_A$ can be solved with the continuity equation by writing it in the form:

$$\mathbf{D}_A(n,m) = \mathbf{D}(n,m) + \min[\mathbf{D}_A(n-1,m)\mathbf{g}(n-1,m) + \mathbf{D}_A(n-1,m-1), \mathbf{D}_A(n-1,m-2)] \quad \text{B10}$$

where $\mathbf{g}(n, m)$ is a weight of the form ,

$$\mathbf{g}(n,m) = \begin{cases} 1, & \text{if } \mathbf{w}(n) \neq \mathbf{w}(n-1) \\ \infty, & \text{if } \mathbf{w}(n) = \mathbf{w}(n-1) \end{cases} \quad \text{B11}$$

$\mathbf{D}_T$ can now be solved as:

$$\mathbf{D}_T = \mathbf{D}_A(N,M) \quad \text{B12}$$

Equations B8-B12 define the full dynamic programming formulation for the DTW process. Myers et al. (1980) provides an excellent the derivation with the solution under various constraints. The "DTW distance" or the minimum warp distance in the parallelepiped is the distance attribute used in our analysis to determine which block of data on the test log best matches a target block of data on the reference log.

The DTW is implemented in our workflow as follows:

1. Select a GRP from the type log to be identified over all the wells.
2. Use this GRP to stretch and squeeze and match against all the segments of the next well and plot the minimum cost as a function of depth for every segment. For one segment, there is one DTW distance.
3. Select the minimum of all these segments as the match and correlation is complete
4. Repeat over all the wells in the area
5. Select next GRP from the type well and repeat steps 1-4.

Figures for chapter 2

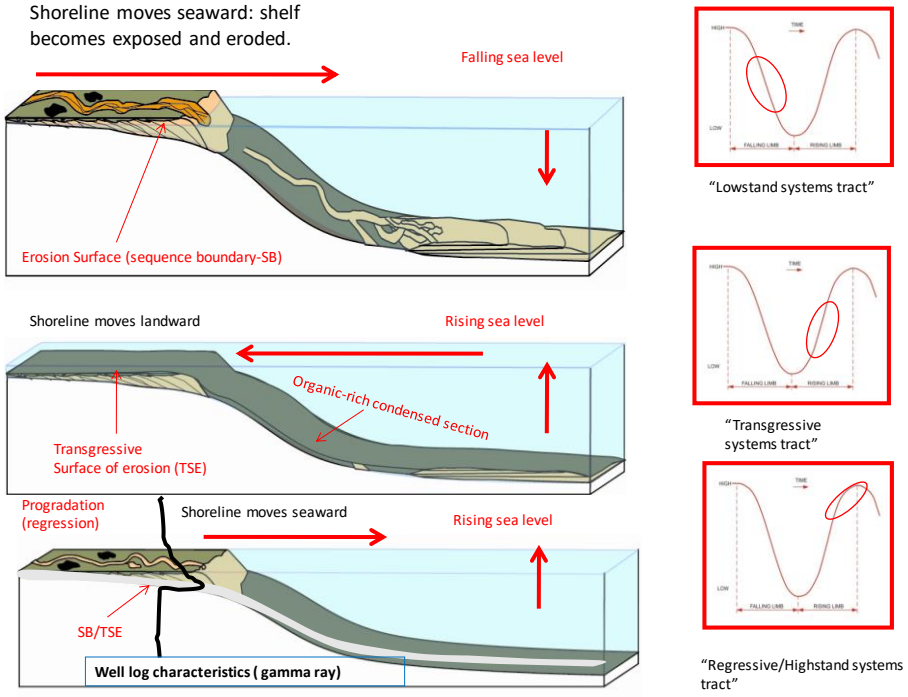


Figure 2-1 Cartoon showing the relation between the depositional environments and the gamma ray response for a conventional sequence stratigraphic interpretation (modified from Slatt, 2016). Notice that for the condensed section, a GR peak is observed and as the system is exposed to erosional elements, the GR decreases a.k.a coarsening upward trend. An opposite trend is observed when the sea level is rising a.k.a , a fining upwards trend.

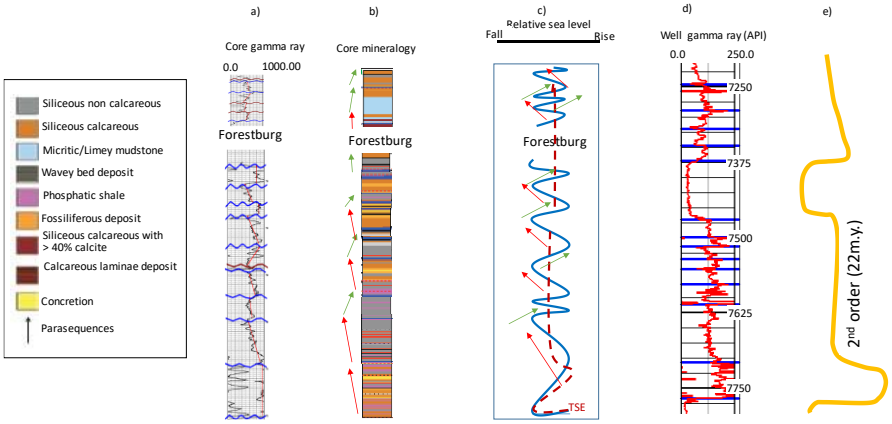


Figure 2-2 Data from a Barnett Shale core and corresponding electric logs: (a) The gamma ray log measured from the core , manually segmented by an expert interpreter. (b) Lithofacies distribution in a lithology log constructed by Singh (2008). Red arrows indicate an increase and green arrows a decrease in carbonate facies with the direction of the arrow. (c) Interpreted relative sea level curve. Red arrows indicate sea level fall and green arrows sea level rise. (d) The same gamma ray parasequences correlated on a gamma ray measured in the well. (e) The interpreted second order sequence of 22 million years. The arrows are closely related showing the lithofacies trends can be used to derive other information such as depositional environment and the relative hydrocarbon potential. The cyclicity in the upward

fining/ coarsening sequences, variation in sea level and toxic and anoxic depositional environment is ultimately correlated to the eustatic sea level curve. (After Slatt ,2012)

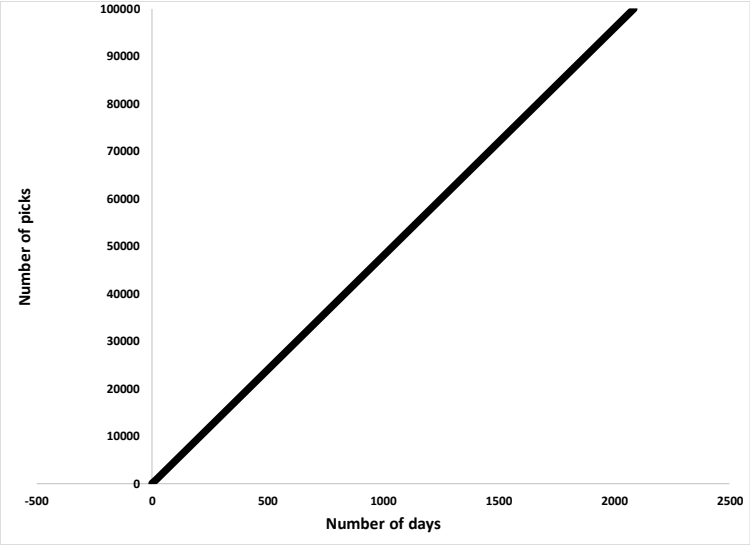


Figure 2-3 Estimated time taken by a human interpreter to manually pick formation top on well logs as the number of well logs increases. For a small number of wells such as an offshore turbidite reservoir the interpretation can be done manually. However, for a typical shale play with a thousands of wells, it can thousands of interpreter hours to generate the complete results. This make the manual interpretation of all the wells prohibitively expensive.

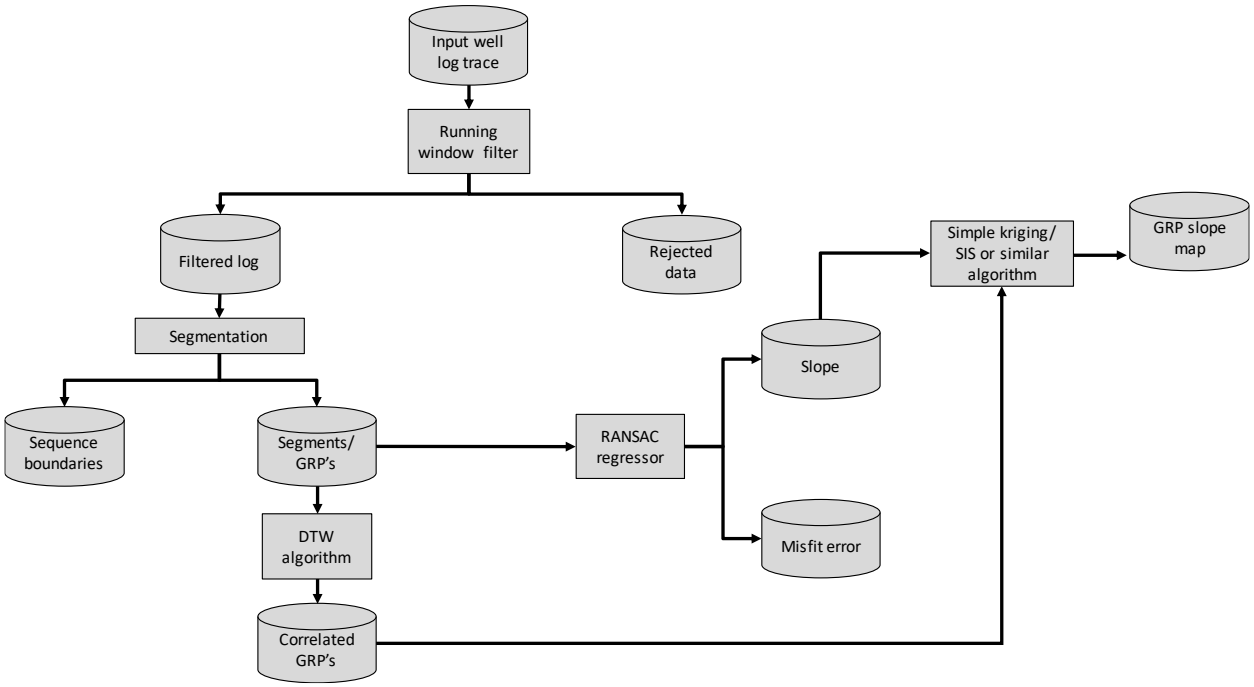


Figure 2-4 The GR well log segmentation and interpretation workflow. The input well log is first processed with a running window filter that rejects spikes in the data, where the rejected data are stored as an attribute. The filtered log is then segmented into different gamma ray parasequences (GRPs) which are then correlated using a DTW algorithm. Next, a robust least-squares RANSAC regressor represents each GRP for each well by a mean and slope, where the deviation from a linear trend provides a measure of the model accuracy. The slope measurements mimic manual picks of upward fining and upward coarsening trends. Finally, the correlated slopes from multiple wells are kriged to produce a slope map for each GRP that can be used to map lateral changes in sediment deposition.

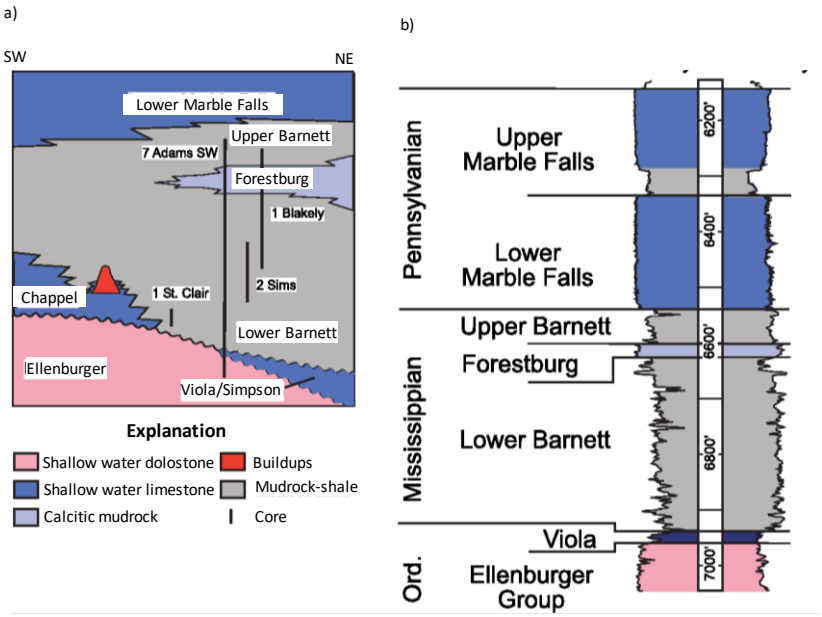


Figure 2-5 (a) Stratigraphy of the Fort Worth Basin in North Texas. (b) Well log from TP Sims#2 well, showing the major formations including the Marble Falls, the Upper Barnett, Forestburg, Lower Barnett, Viola, and Ellenburger Group. In the Wise Co. study area described in this paper, there is no Chappel limestone, and the Marble Falls and Viola limestones form the upper and lower hydraulic fracture barriers. The Lower Barnett Shale has a high quartz content resulting in better completion than the quartz-poor Upper Barnett Shale. The Forestburg limestone is too thin to form an effective hydraulic fracture barrier. The Ellenburger aquifer is isolated from the Lower Barnett by the tighter Viola limestone. (After Montgomery et al., 2005).

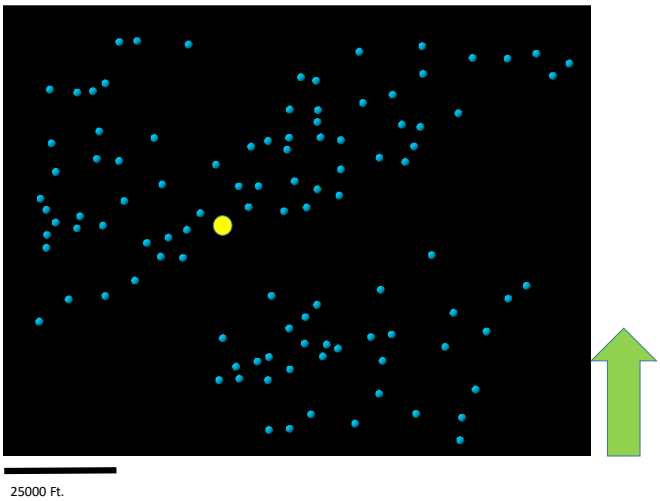


Figure 2-6 Relative location of the 106 wells (blue circles) in our study. The wells are shown by blue spheres and the north is indicated by the green arrow. The yellow sphere shows the location of the type well in the area.

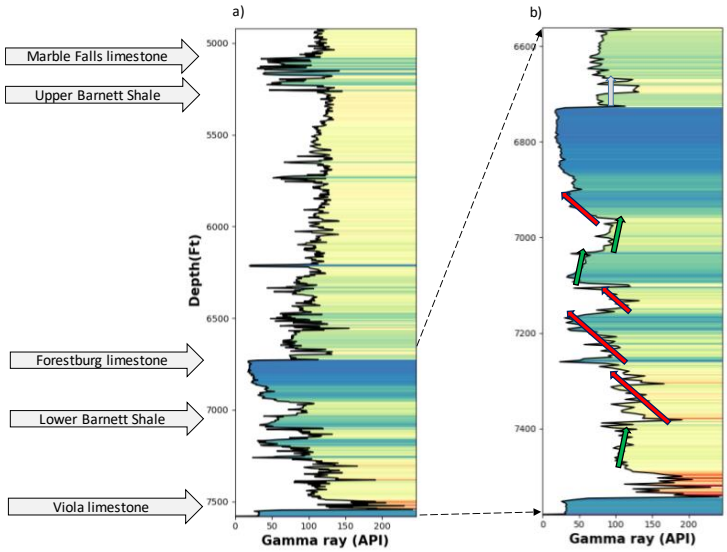


Figure 2-7 (a) Formation tops and the type gamma ray log for geologic section analyzed in this paper. (b) Zoomed section showing the gamma ray parasequence trends. Following the color scheme used by Slatt (2012) in Figure 3, green arrows indicate upward increasing API and red arrows upward decreasing API.

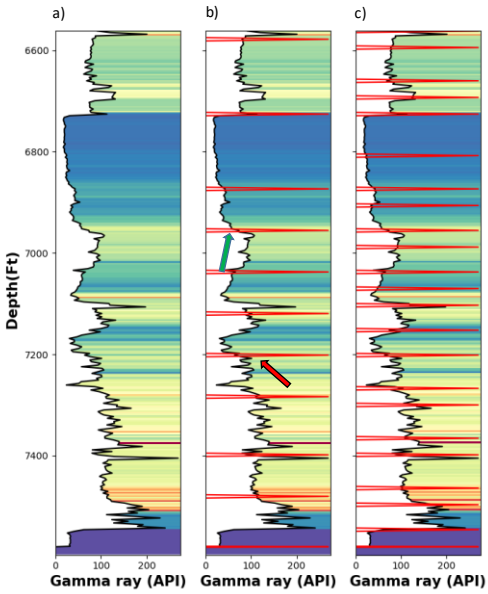


Figure 2-8 Segmentation performed on a well log using the parameters provided in Table 2 showing the (a) filtered log, and the results using a (b) coarse segmentation and (c) fine segmentation. Notice the GRP segments identified by the algorithm. Because different order parasequences may be required for a specific interpretation objective, the segmentation algorithm can be used to pick small scale as well as large scale segments. An interpreter may then choose to manually pick green (upward increasing API) and red (upward decreasing API) arrows. Our goal is to automate this process.

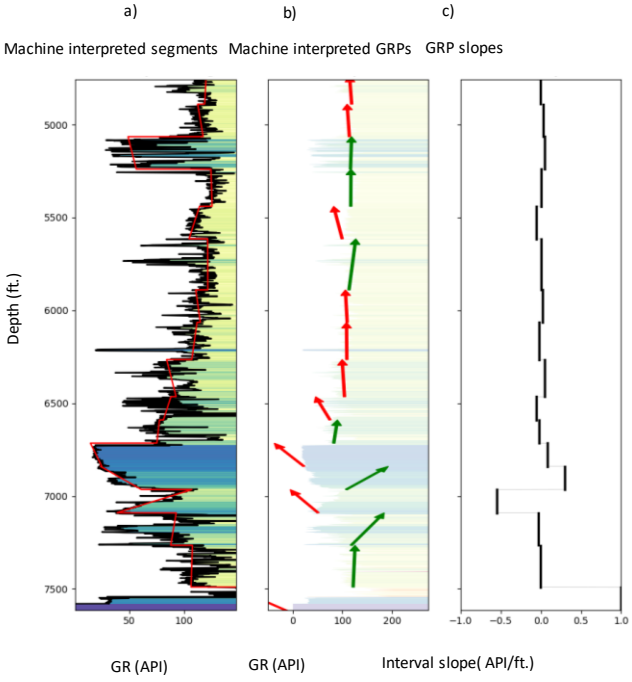


Figure 2-9 The results of the workflow shown in Figure 4 on type well showing: (a) the gamma ray well log and the linear RANSAC regressor fit to each segment/GRP, (b) the interpreted depositional trends (upward increasing/ decreasing API) or sequences. (c) the slope of the RANSAC regressor fit for each GRP. Notice that the interpretation is similar to an expert interpreter and can be obtained in a fraction of time for the full well log trace over thousands of

wells. A quantitative measure of the GRPs is also available along with the upward fining/coarsening trends. The increase in API upwards is shown by a green arrow and an upward decrease in the API is shown by a red arrow.

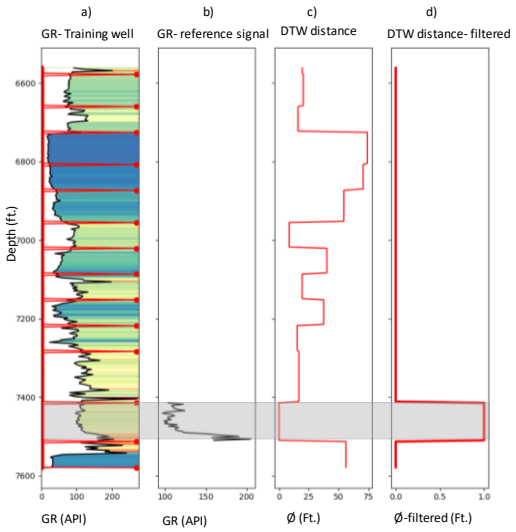


Figure 2-10 Dynamic time warping (DTW) results over the type well in Barnett Shale. From left to right (a) Gamma ray weathering profile showing major parasequences obtained with segmentation and the parameters described in Table 3. The lower Barnett GRP to be correlated is highlighted in the gray box. (b) The reference signal/ GRP to be queried across all the segments (c) The minimum warp Euclidian distance required to obtain the best match. (d) The filtered distance. The segment with minimum distance is set at True; other segments are set to be False to highlight the match and is referred to as the filtered DTW distance. The GRP segment from lower Barnett is selected as the reference signal. This signal is then compared against all segmented pieces on all the other wells using "stretch" and "squeeze" operations. The associated cost is plotted as a distance metric and the segment with the least cost to obtain the best possible match is recorded. In this case the segment is from the given log itself and hence the cost of matching is zero, which is expected. This is analogous to "autocorrelation" to show the validity of the argument.

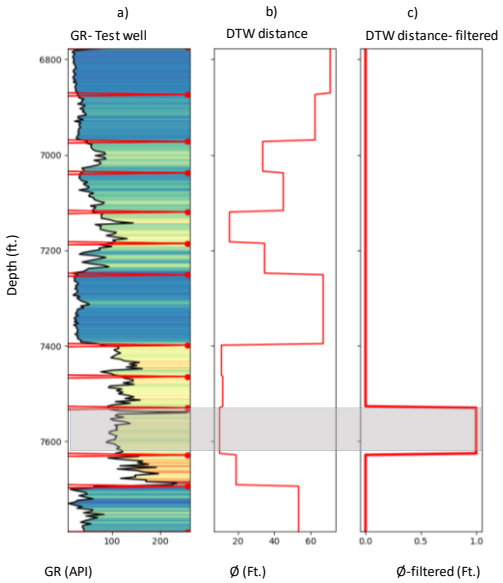


Figure 2-11 Dynamic time warping (DTW) results over the test well in Barnett Shale showing (a) Gamma ray weathering profile showing major parasequences obtained using the same segmentation parameters as the type log/ training well shown in the previous figure. (b) The minimum warp Euclidian distance required to obtain the best match between the reference signal from type well/ training well. (c) The filtered distance. The segment with the minimum distance is set to be True and the others False to highlight the match. Notice that when the reference signal is taken from the type well and tested on a new well, it correctly identifies the GRP in the new well (highlighted in grey box) even though the thickness and shape of the GRP are different. The DTW distance is no longer zero but rather minimum over all the segments. This process can be repeated over thousands of wells in a very short amount of time. Once correlated the interpreter can develop two- and three-dimensional models with properties assigned to each of the parasequences.

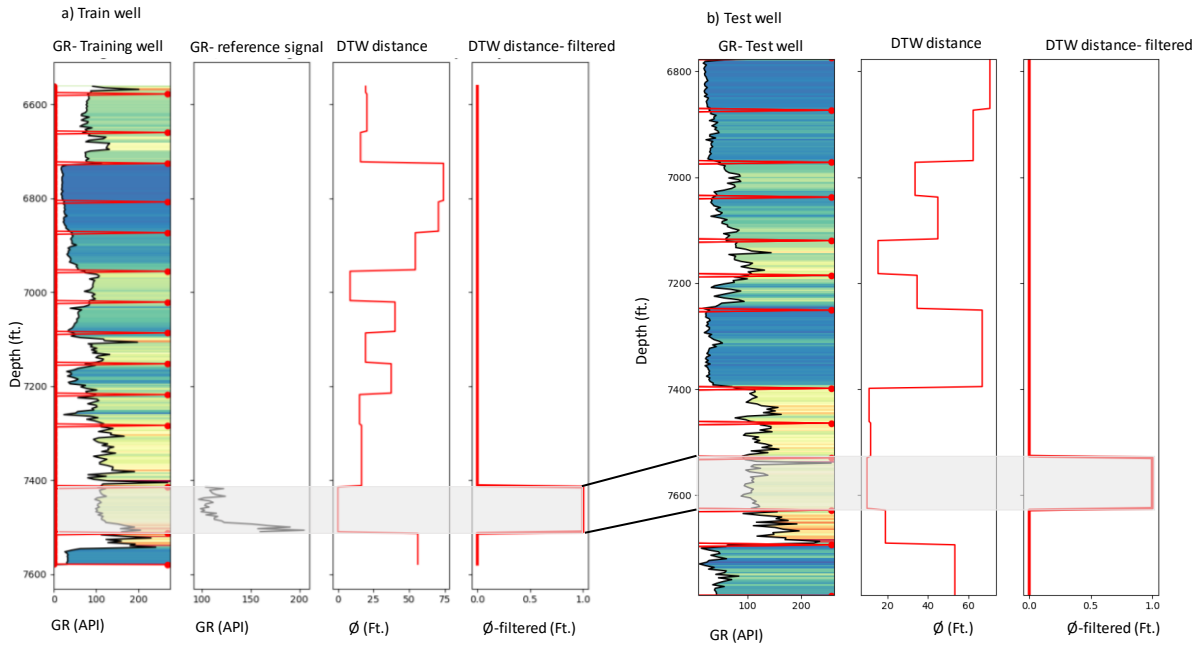


Figure 2-12 Automated correlation of (a) a training well and (b) a representative test well. We extract a template signal (in black, second panel in a) and stretch and squeeze it to match with every segment of the test well shown in (b). The segment with the lowest cost is highlighted and assigned as a match. Notice that the algorithm has correctly identified the GRP in the test well although the signal depth, thickness, thickness, and shapes are different.

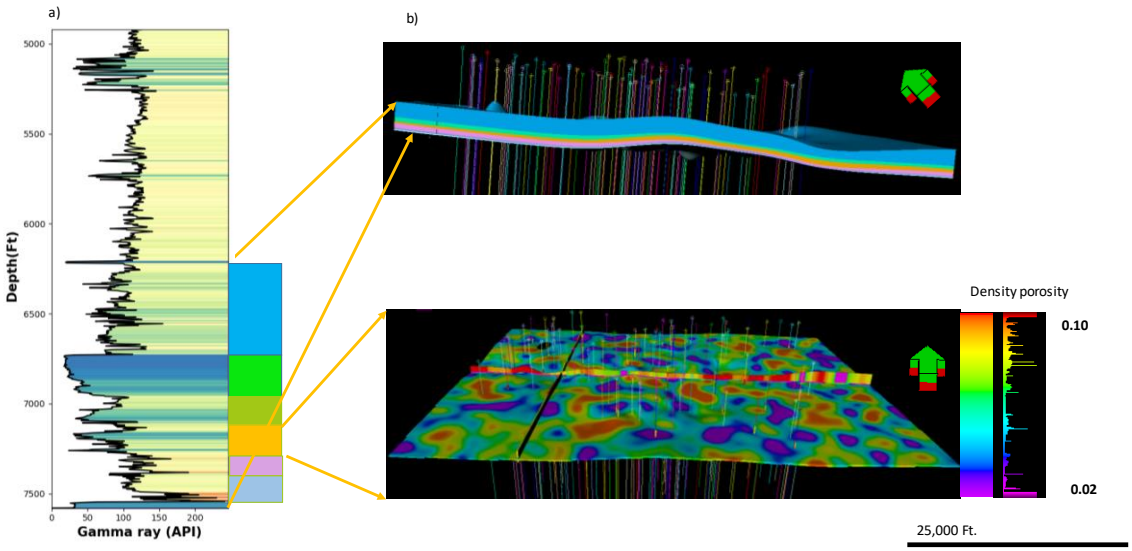


Figure 2-13 a) The type well and the highlighted sections picked on a GR log. Small GRPs are not picked as they are not continuous throughout the study area. Also, some GRPs are automatically delineated if they fall between the picked GRPs. We begin by (1) picking the entire package of interest. Then (2) we pick the Forestburg limestone. Then (3) we pick one of the larger GRP and (4) the Viola limestone. (b) The three-dimensional GRP zone model generated from the well tops and three GRP zone model guided density porosity.

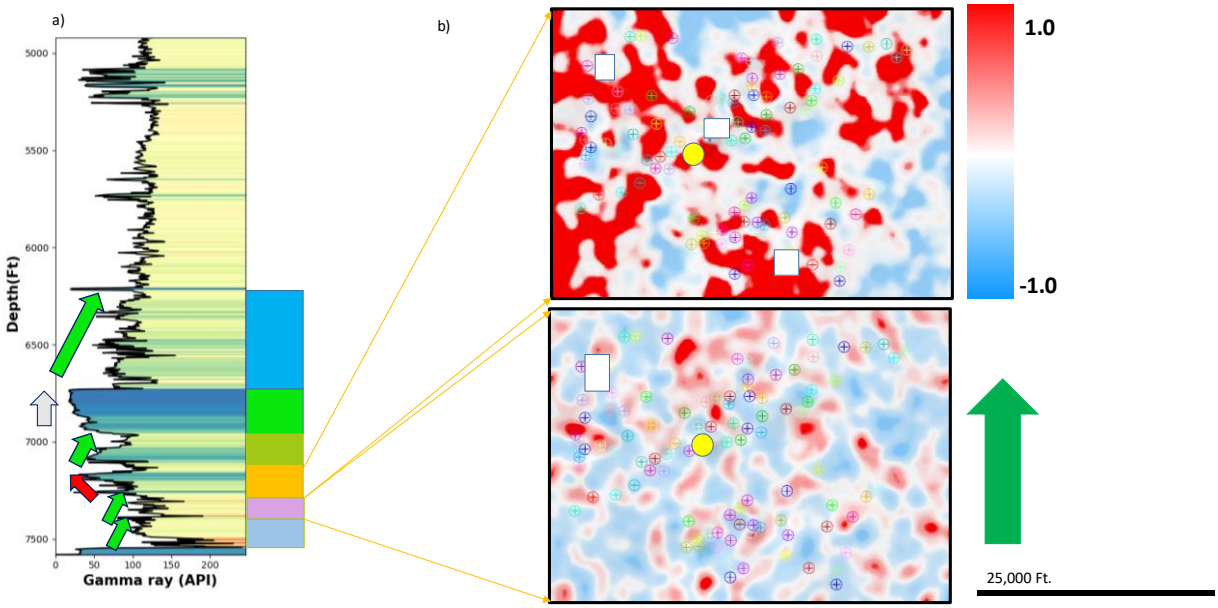


Figure 2-14 a) The type well and the highlighted sections picked on a GR log. (b) Two-dimensional GRP slopes for zones 4 and 5 show the changes in the rate of vertical deposition for two GRPs in time. The white boxes show the points where the interpolation is not reliable due to small thickness of the GRP.

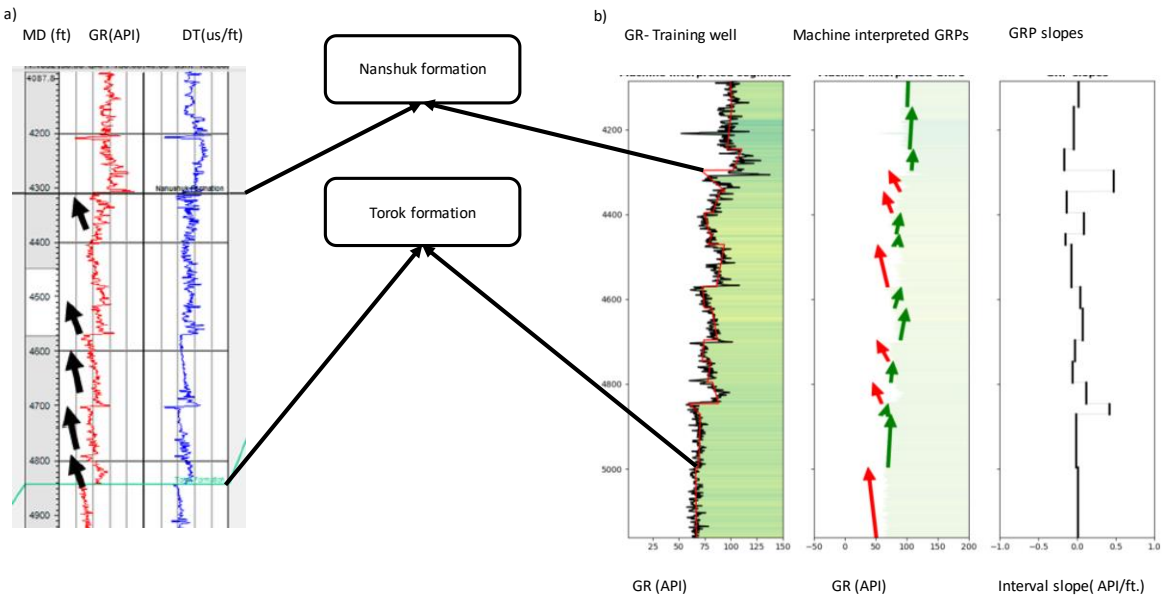


Figure 2-15 A comparison of manual picks generated by a skilled human interpreter and the results from the workflow described in Figure 4. (a) Manual interpretation of the Nanushuk and Torok Formations on the North Slope, Alaska from Bhattacharya and Verma (2020). (b) Machine interpreted parasequences. Notice the fine scale picking of parasequences over full well log provided by the automated workflow in (b). The algorithm correctly identifies the sequence boundaries, and the interpretation is at par with the human interpreter. The machine can pick finer scale parasequences in many cases missed by a human interpreter. In addition to the interpretation of parasequences, a quantitative measure of relative slope is added as an aid to the interpretation.

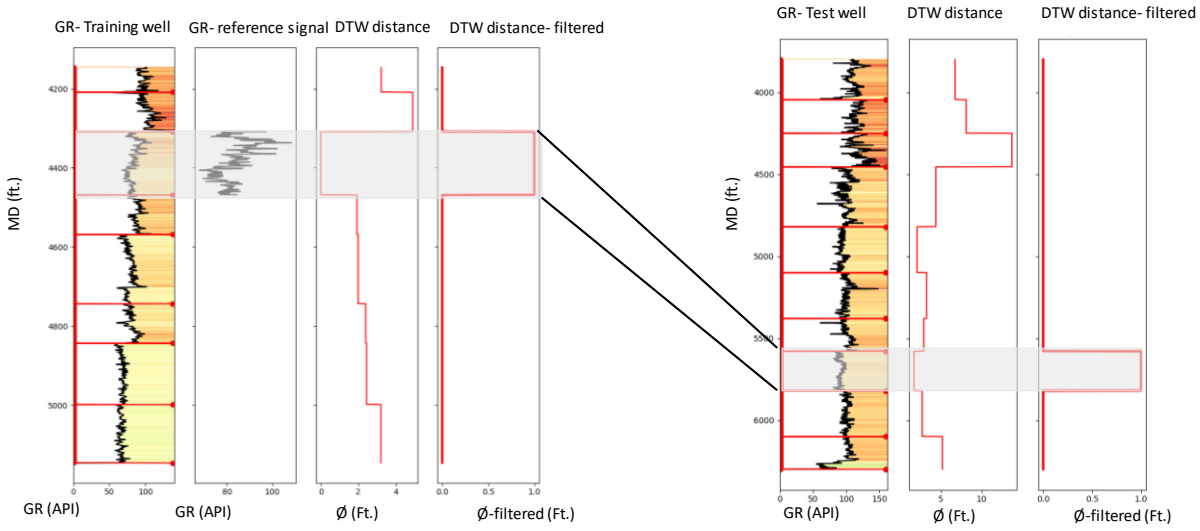


Figure 2-16 Automated correlation of the parasequences for a suite North Slope Alaska gamma ray logs showing (a) the training (type) well and (b) a representative test well. test well. Notice that the algorithm has correctly identified the highlighted parasequences in the test well. In this case, the signal shapes are similar.

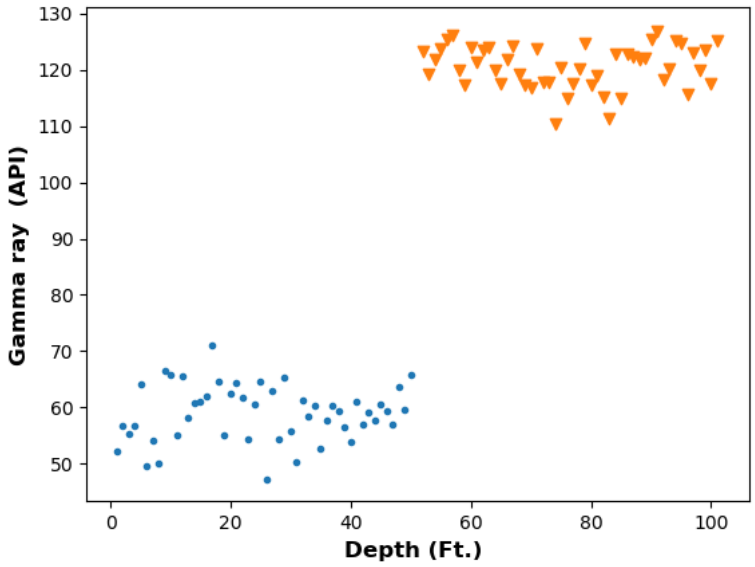


Figure 2-17 The lateral changes in the mean from μ_{Left}=60 API (blue circles) and μ_{Right}=120 API (orange triangles) with a variance of 25 API in each case.

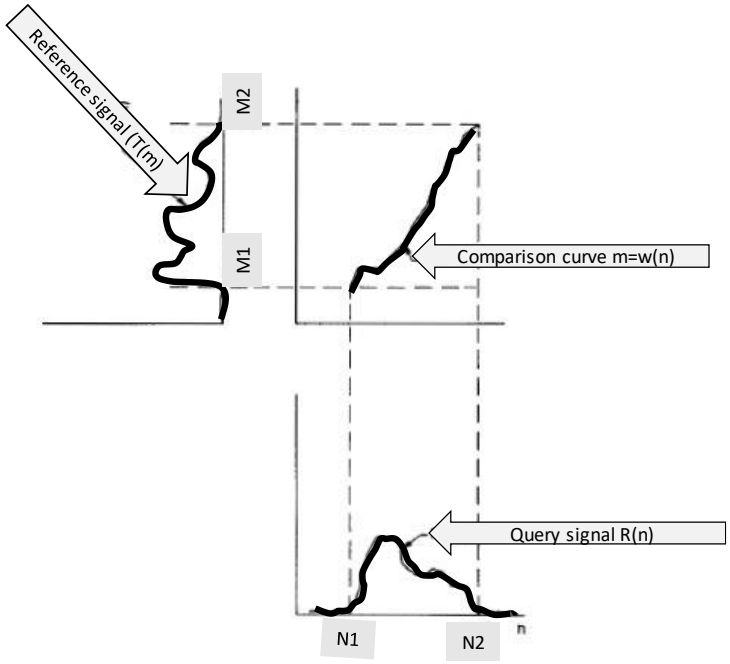


Figure 2-18 DTW illustration showing a reference and a query signal under the boundary condition that the end points M_1 and M_2 of the reference signal match the endpoints N_1 and N_2 of the reference signal. (after Rabiner et al., 1978).

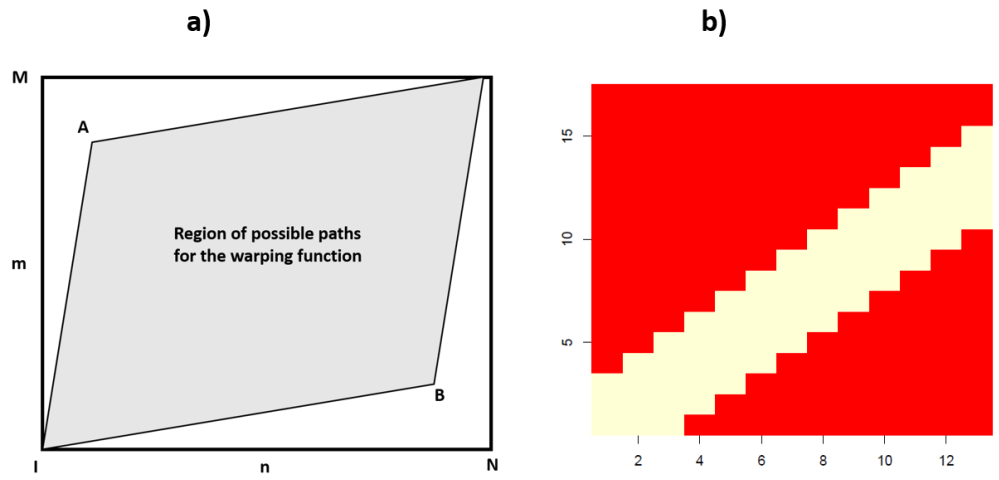


Figure 2-19 (a) Warping function modified after Rabiner et al. (1978) b) The minimum distance solved using dynamic programming in a Python program. (After Giorgino (2009).

References

Behdad, A., 2019, A step toward the practical stratigraphic automatic correlation of well logs using continuous wavelet transform and dynamic time warping technique: Journal of Applied Geophysics, **167**, 26–32.

Bhattacharya, S., S. Verma, and J. R. Rotzien, 2020, 3D seismic imaging of the submarine slide blocks on the North Slope, Alaska: Interpretation, **8**, SR37–SR44.

Chen, J., and A. K. Gupta, 2011, Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance: Springer Science & Business Media.

Cho, H., and P. Fryzlewicz, 2015, Multiple-change-point detection for high dimensional time series via sparsified binary segmentation: Journal of the Royal Statistical Society: Series B: Statistical Methodology, 475–507.

Chu, C.-S. J., 1995, Time series segmentation: A sliding window approach: Information Sciences, **85**, 147–173.

Dhingra, S. D., G. Nijhawan, and P. Pandit, 2013, Isolated speech recognition using MFCC and DTW: International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, **2**, 4085–4092.

Fang, H., Y. Lou, B. Zhang, H. Xu, and M. Lu, 2021, Mimicking the process of manual sequence stratigraphy well correlation: Interpretation, **9**, T667–T684.

Fischler, M. A., and R. C. Bolles, 1981, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography: Communications of the ACM, **24**, 381–395.

Keogh, E., S. Chu, D. Hart, and M. Pazzani, 2001, An online algorithm for segmenting time series: Proceedings 2001 IEEE International Conference on Data Mining, 289–296.

Killick, R., and I. Eckley, 2014, changepoint: An R package for changepoint analysis: Journal of Statistical Software, **58**, 1–19.

Killick, R., P. Fearnhead, and I. A. Eckley, 2012, Optimal detection of changepoints with a linear computational cost: Journal of the American Statistical Association, **107**, 1590–1598.

Montgomery, S. L., D. M. Jarvie, K. A. Bowker, and R. M. Pollastro, 2005, Mississippian Barnett Shale, Fort Worth basin, north-central Texas: Gas-shale play with multi–trillion cubic foot potential: AAPG Bulletin, **89**, 155–175.

Myers, C., L. Rabiner, and A. Rosenberg, 1980, Performance tradeoffs in dynamic time warping algorithms for isolated word recognition: IEEE Transactions on Acoustics, Speech, and Signal Processing, **28**, 623–635.

Qi*, J., M. Cahoj, A. AlAli, L. Li, and K. Marfurt, 2015, Segmentation of salt domes, mass transport complexes on 3D seismic data volumes using Kuwahara windows and multiattribute cluster analysis, *in* SEG Technical Program Expanded Abstracts 2015, Society of Exploration Geophysicists, 1821–1825.

Rabiner, L., A. Rosenberg, and S. Levinson, 1978, Considerations in dynamic time warping algorithms for discrete word recognition: IEEE Transactions on Acoustics, Speech, and Signal Processing, **26**, 575–582.

Sanderson, J., P. Fryzlewicz, and M. Jones, 2010, Estimating linear dependence between nonstationary time series using the locally stationary wavelet model: Biometrika, **97**, 435–446.

Singh, P., 2008, Lithofacies and sequence stratigraphic framework of the Barnett Shale, Northeast Texas: .

Sullivan, P. J., 2015, Biomeasurement: A Student's Guide to Biological Statistics: .

Van Wagoner, J., H. Posamentier, R. Mitchum, P. Vail, J. Sarg, T. Loutit, and J. Hardenbol, 1988, An overview of the fundamentals of sequence stratigraphy and key definitions: .

Wambui, G. D., G. A. Waititu, and A. Wanjoya, 2015, The power of the pruned exact linear time (PELT) test in multiple changepoint detection: American Journal of Theoretical and Applied Statistics, **4**, 581.

Wei, L., E. Keogh, and X. Xi, 2006, Saxually explicit images: Finding unusual shapes: Sixth International Conference on Data Mining (ICDM'06), 711–720.

Wheeler, L. F., 2015, Automatic and Simultaneous Correlation of Multiple Well Logs: Colorado School of Mines.

Chapter 3 : Normal or abnormal? Machine learning for the leakage detection in carbon sequestration projects using pressure field data

Saurabh Sinha[1,3], Rafael Pires De Lima[2], Youzuo Lin[3], Alexander Y. Sun[4], Neil Symons[3], Rajesh Pawar[3], George Guthrie[3]

[1]School of Geosciences, The University of Oklahoma, Norman, OK, USA

[2]Geological Survey of Brazil, São Paulo, Brazil

[3] Los Alamos National Laboratory, Los Alamos, NM, USA

[4]Bureau of Economic Geology – University of Texas at Austin, Austin, TX, USA

Preface

This chapter is presented as published in the journal of greenhouse gas control (Sinha et al., 2020a), which itself is based on the expanded abstracts published with SPE (Sinha et al., 2020b). This chapter shows the results from an anomaly detector based on deep learning framework. The anomaly detector is designed to detect leaks in carbon sequestration projects with minimal human intervention.

References

Sinha, S., R. P. de Lima, Y. Lin, A. Y. Sun, N. Symons, R. Pawar, and G. Guthrie, 2020a, Normal or abnormal? Machine learning for the leakage detection in carbon sequestration projects using pressure field data: International Journal of Greenhouse Gas Control, **103**, 103189.

Sinha, S., R. Pires De Lima, Y. Lin, A. Y Sun, N. Symon, R. Pawar, and G. Guthrie, 2020b, Leak Detection in Carbon Sequestration Projects Using Machine Learning Methods: Cranfield Site, Mississippi, USA: SPE Annual Technical Conference and Exhibition.

Abstract

The international commitments for atmospheric carbon reduction will require a rapid increase in carbon capture and storage (CCS) projects. The key to any successful CCS project lies in the long-term storage and prevention of leakage of stored carbon dioxide (CO_2). In addition to being a greenhouse gas, CO_2 leaks reaching the surface can accumulate in low-lying areas resulting in a serious health risk. Among several alternatives, some of the more promising CCS storage formations are depleted oil and gas reservoirs, where the reservoirs had good geological seals prior to hydrocarbon extraction. With more CCS wells coming online, it is imperative to implement permanent, automated monitoring tools. We apply machine learning models to automate the leakage detection process in carbon storage reservoirs using rates of (CO_2) injection and pressure data measured by simple harmonic pulse testing (HPT). To validate the feasibility of this machine learning based workflow, we use data from HPT experiments carried out in the Cranfield oil field, Mississippi, USA. The data consist of a series of pulse tests conducted with baseline parameters and with an artificially introduced leak. Here, we pose the leakage detection task as an anomaly detection problem where deviation from the predicted behavior indicates leaks in the reservoir. Results show that different machine learning architectures such as multi-layer feed forward network, Long Short-Term Memory, and convolutional neural network are able to identify leakages and can provide early warning. These warnings can then be used to take remedial measures.

Introduction

Carbon capture and storage (CCS) refers to the process of permanently capturing the carbon dioxide (CO_2) emitted primarily by large point sources (e.g., power plants, cement processing facilities, and other fixed industrial assets) into geological formations (Selma et al.,

2014). Depleted hydrocarbon reservoirs serve as an excellent target for CCS (Bachu, 2000, 2003) due to their higher storage capacity with available infrastructure in place. Older hydrocarbon production/injection wells can be repurposed as injection wells for the CO_2 injection.

Although complex, the physics of fluid flow in geological reservoirs is fairly well understood by petroleum engineering community. CO_2 injection for enhanced oil recovery (EOR) projects has been used for decades in the oil industry. The injected CO_2 can take a significant time to convert into stable form by either dissolution, precipitation, or other chemical processes (Moore et al., 2005; Oelkers et al., 2008; Gaus, 2010). Hence, long term storage of the CO_2 without leaks is the cornerstone of every CCS project. Leakage of CO_2 out of the subsurface into the atmosphere completely negates the purpose of a CCS project. Moreover, the injected CO_2 migration out of the target formation into surrounding geological layers can pose a different set of environmental problems. CO_2 leaked out of the target formation can mix with the water table and increase its acidity of the aquifer, it can also affect severely plant growth at surface soil levels and soil microbiology(Smith et al., 2013; Fernández-Montiel et al., 2015).

The process of CO_2 injection is not just operationally challenging, but also time dependent. In a CCS project, CO is injected into a target storage formation or reservoir, increasing the reservoir pressure and hence changing the stresses in the reservoir and potentially reactivating faults in the area (Ivanova et al., 2012; Rutqvist, 2012; Castelletto et al., 2013). If the faults slip, the stored CO_2 in the reservoir may migrate to other geological formations and eventually to the surface. The detection of CO_2 leakage in CCS projects can be accessed via several technologies. (Gal et al., 2019) and (May and Waldmann, 2014), measured CO_2 concentrations in surface soils. (Shao et al., 2019a) found carbon isotopes in the soil served as a proxy for the CO_2 leak, whereas (Shao et al., 2019b) showed that surface measurements of tracers injected with the CO_2 can

indicate leaks. Although most of these methods are economically viable and are direct indicators of CO_2 leak, they do not provide an early warning signal to prevent the leak during the operational stage. By the time the leak is detected, a significant amount of CO_2 may already have reached to the surface.

Two reliable and well-established field methods for continuous monitoring of subsurface properties are via seismic monitoring and the hydrocarbon well pressure monitoring. Bergmann et al., (2010), Roach et al., (2015), Macquet et al., (2017), and, Stork et al., (2018) showed the usage of seismic data for leak detection in the CCS projects utilizing hydrocarbon reservoirs. Three-dimensional (3D) seismic data provide a 3D image of the changes in velocities and density in the subsurface caused by a CO_2 plume. A time-lapse analysis of 3D seismic can provide valuable insights on plume development and migration. However, the acquisition, processing, and interpretation of seismic data are expensive and time-consuming even for passive monitoring (Verdon et al., 2010a, 2010b; Dondurur, 2018). Additionally, seismic data are often band limited, providing relatively low vertical resolution images and hence offering only an approximate estimate (e.g., within 50-100 m) on the spatial location of any leak. The quality of the seismic interpretation is dependent on data acquisition, data processing, and the initial guess of the geological model itself.

An alternative to the time-lapse seismic data monitoring is pressure data monitoring. The formation pressure is measured with a pressure gauge which is both inexpensive and easy to install. Multiple pressure gauges can be deployed at a variety of perforation intervals, as well as across multiple locations in a field. The gauge continuously provides high frequency, high resolution data in real time via a telemetry system(Reeves et al., 2011; Hawthorn et al., 2017). The injection of

CO_2 causes a pressure perturbation and can be induced intentionally in a process called a "well test". One such well test is a harmonic pulse testing (HPT).

A typical HPT consists of alternative cycles of fluid injection and well shut-in causing multiple pressure perturbations or sampling the reservoir at multiple frequencies (Brigham, 1970; Fokker and Verga, 2011; Sun et al., 2015; Fokker et al., 2018). The pressure data are obtained at the bottom of the well and hence exhibit a high signal to noise ratio. For the gauges installed at the surface, additional noise is introduced in the well tubing due to multi-phase flow and the vibrations. A gauge installed at the bottom of the well avoids the additional noise by recording pressure directly at the sandface. HPT is also used to build a dominant frequency in the signal, thus boosting the signal to noise ratio. The HPT typically requires at least two wells: the first is an injection well at which this perturbation is induced by the harmonic injection of a fluid into the reservoir; the second is an observation well which records the response of the reservoir due to these pulses. Analyzing the pressure signal at the observation well provides various insights about the reservoir and the fluid properties such as reservoir connectivity, fault proximity, permeability, etc. Sun et al. (2016) demonstrated the efficacy of the pressure HPT in distinguishing the pressure response of a leak versus the non-leak case in a field test in Cranfield, Mississippi, USA. The fact that a pulse test can distinguish between the leak and no-leak scenarios lays the foundation of this work.

Analysis of HPT data from a single well by human interpreters is not an extremely complicated task. A skilled human interpreter can easily distinguish the difference between leak and non-leak responses. However, a large CCS project across multiple depleted oil fields may incorporate many injection wells, and many abandoned wells can act as leakage paths from the storage formation. Each well can be instrumented, with pulse tests being conducted frequently along with continuous pressure data monitoring. Analysis of so many live pressure feeds can be a

daunting task for human interpreters. In contrast, computers are extremely efficient at repetitive tasks and intuitively, are best suited for this purpose. The core goals for our work are to demonstrate that an HPT can distinguish between a leak and non-leak scenarios and, if so, to identify appropriate ML approaches to automate the process.

Recent advances in algorithms and computing, and the availability of more and better data, provide an opportunity for remarkable progress in the solution to effective detection of CO_2 leakage. Sun et al., (2014) proposed segregating the leak signatures from the baseline using spectral analyses of the pressure response. Zhou et al., (2018, 2019) demonstrated the use of convolutional neural networks (CNN) with synthetically modeled seismic data to monitor the CO_2 volume. Zhong et al., (2019) showed the use of reservoir simulator images for a convolutional-Long Short-Term Memory (Conv-LSTM architecture). de Lima et al., (2019a) showed the usage of various deep learning architectures to detect leaks from multiple targets in seismic data. de Lima et al., (2019b) transformed the seismic images to pseudo RGB images for improved computational efficiency. All of the above methods are either based on seismic data or two-dimensional images which either suffer from resolution issues or are computationally expensive. In many cases, the data required for training, testing, and or the software's required to do these analyses are not available from the field, limiting the practicality of these methods.

The aim of this work is to create an early warning detection system for leakages in CCS projects using pressure data along with rate and cumulative injection volume of CO_2. We used currently available state-of-the-art deep learning models and adapt them to this application. The data consist of time series data of injection rates and measured pressures from a set of pulse tests conducted at the Cranfield reservoir, Mississippi, USA. As we do not utilize any two-dimensional

images or three-dimensional voxels in this study, our methodology is computationally efficient while providing state-of-the-art results.

We evaluated deep learning architectures such as multi-layer feed forward network (MFNN), long short-term memory (LSTM), convolutional neural networks (CNN), and a combination of CNN and LSTM, abbreviated as CONV-LSTM. The results show that if trained on a particular set of data, a machine learning method can identify anomalous behavior from the learned data and flag it as an anomaly. Sophisticated architectures such as CNN and LSTM can learn and retain features present in typical HPT pressure response. We also show the adaptability of the models for different pulse tests as well as their limitations. The results can be achieved in a fraction of time using simple inputs but, sophisticated architectures. The final product of this analysis is an anomaly detector which can differentiate between expected versus the anomalous pressure behavior.

This paper is divided into six sections. Sections 1 and 2 provide background information about the data used in this manuscript, the problem formulation, and data pre-processing for our workflow. Section 3 discusses the problem statement and its formulation as the anomaly detection problem, as well as a basic introduction of the neural network architectures used in the study. Section 4 contains results from the neural network models and the discussion of these results. Section 5 presents the model extension to different scenarios. We present the conclusions in Section 6.

Background

Cranfield site data acquisition

The data utilized in this study were obtained in the Cranfield reservoir, Mississippi, USA. The Cranfield reservoir is primarily a sandstone reservoir that was used for CO_2 enhanced oil recovery project after the primary and secondary recovery (Sun et al., 2016). The CO_2 injection for the purpose of sequestration started in July 2008. The series of HPT experiments used in this study were carried out in January 2015, the total injected volume into the reservoir at the time of the test was more than one million metric tons. Due to the high injection volumes continuously for seven years prior to the test, it is assumed that in-situ brine effects are negligible for any practical purposes and supercritical CO_2 is the only in situ fluid present in the reservoir. Figure 3-1 shows the setup used in the pulse test as described by Sun et al., (2016). Three wells are utilized in the study: F1, F2, and F3. The distance between the wells is also shown in Figure 3-1. The CO_2 is injected in well F1 which is the injection well, well F2 is the monitoring well, and F3 is a well that can be used to add an artificial leak by venting off CO_2 via a surface valve to simulate a real-life leakage scenario.

Hereafter, we name "baseline" the cases on which the pulse tests are conducted without the introduction of a leak, and an alternating sequence of injection and shut-in is carried out. Two baseline tests, one of 90-min and one of 150-min were conducted on 19th January and January 20th of 2015, respectively. "Leak" tests are similar to baseline tests, but with an artificial leak introduced at well F3 by opening a surface valve. Two leak tests were conducted ten days after the baseline tests, one on 30th of January 2015 and another on 31st of January 2015 with a duration of 90-min and 150-min respectively. Well F2 remains shut-in as an observation well during all tests and pressure data acquired at well F2 is used in our analysis. Table 3.1 summarizes the pulse test schedule for all four tests. Fig. 3.2 shows the measured pressure recorded at well F2 during the Baseline - 150 min test, along with the injection rates at well F1. We only provided the

necessary details about the data for the objectives of this study. A complete description of the data acquisition is provided by Sun et al., (2016).

The pressure gauge sampling rate used in the baseline tests is one sample every two seconds (i.e., a sampling rate of 0.5 samples/s). The sampling rates used in the leak tests are one sample every five seconds, i.e., a sampling rate of 0.2 samples/s. Before any analysis, we matched the sampling rates of both baseline and leak data to 1 sample/s. As very limited data are available to train the models, we used the Fourier method to re-sample the signal to 1 sample/second instead of decimating the signal (Heideman et al., 1984).

Once we re-sampled the data, and both baseline and leak tests have the same sampling rates, we de-spiked the data. To de-spike the data, we used a Hann window of 21 samples and convolved the samples in the Hann window with the scaled data. A Hann function of length L (Harris, 1978) is defined as:

$$w_o(x) \triangleq \begin{cases} \frac{1}{2}\left(1+\cos\frac{2\pi x}{L}\right) = \cos^2\frac{\pi x}{L}, & |x| < \frac{L}{2} \\ 0, & |x| \geq \frac{L}{2} \end{cases}, \qquad 3.1$$

Where, x is the discreet signal. The Hann function can then be re-sampled for a discrete signal as:

$$w_o(x) = \begin{cases} \frac{1}{2}\left(1-\cos\frac{2\pi n}{N}\right) = \cos^2\frac{\pi n}{N}, & 0 \leq n \leq N \end{cases}, \qquad 3.2$$

Where, N+1 is the length of the window. For a signal, s(t) this window can then be convolved over the signal as:

$$s'(t) = s(t) * w(N+1)\,, \qquad 3.3$$

where s'(t) is the smooth signal in the time domain, and w(N+1) is a Hann window of samples. As injection rates remain the same for all tests, the rates alone cannot be used to predict the different pressures for the same injection rates. To overcome the issue, we created an additional feature of

injected cumulative volume. This feature was then added as an input along with the injection rates to predict the given pressure response. Cumulative volume feature can be computed by integrating injection rates over a given period as:

$$C = \int_0^T q\, dt, \qquad 3.4$$

where C is the cumulative injected volume, q is the instantaneous rate, and T is the time until injection is completed. We used de-spiked surface pump rates as the injection rates to compute the cumulative injection rate.

Figure 3-2 shows that there is a linear upward trend in the measured pressure in red and hence a detrending of the data is required. Without detrending, the machine learning models would have to extrapolate the input behavior, which is not ideal for anomaly detection tasks. Figure 3-3 shows the resampled, scaled, and detrended data. Notice that the number of samples are now higher and the linear upward trend is removed. As a final step for the input preparation, we used the min–max scalar from the Python 3.7 distribution Scikit-Learn library (Pedregosa et al., 2011) to rescale the inputs for modeling.

Figure 3-3 shows a comparison between the measured pressures in monitoring well F2 for the 90-min baseline and 90-min leak cases. The data in Figure 3-4 shows that the pulse test is an effective method to distinguish between the leak and baseline scenarios and two pressure signals can be segregated. However, the absolute difference in pressure baseline and leak tests is minimal 0.5 Psi. Hence, the modeling techniques for prediction requires excellent accuracy.

Leakage detection – an anomaly detection problem

Anomaly detection

The process of classifying the baseline and leak data are similar to a real life lie detector methodology. A baseline response of the subject's body signals – such as heartbeat – is set by asking questions which are expected to yield definite true answers. Answers to the follow up questions are then compared to the subject's baseline body signals and deviation is recorded. If the deviation is beyond a preset threshold, it is classified as an anomaly. In our case, we have a pressure response for a baseline and a leak test. The baseline test can be used to set a baseline for "true values" and all future responses can be compared to true values along with a threshold to classify data points as "on trend" or an anomaly. Formally, an anomaly is defined as patterns in data that are significantly different than the "normal behavior". Anomalous behavior can be classified as contextual, point or collective anomalies (Chandola et al., 2009). Yu et al., (2014) presented a detailed discussion on the various classifications of the anomalies for a time series data.

A human interpreter can analyze the baseline and leak pressure responses and can observe the differences easily. However, due to computational cost and memory limitations, a computer can only read "chunks" of data. All comparisons must be made on different segments of the data which are essentially a subset of the overall data which is a multi-dimensional time series in our case. A simple method to read the data for signal processing is a sliding window. A sliding window reads segments of data from the given time series and the segments are then used for further processing such as an anomaly detection task. The window is then moved forward and the process is repeated. Yu et al., (2014) applied sliding window for outlier detection in hydrological time series data which are non-stationary, time variant, and have stochastic components superimposed on deterministic trends such as our data. We have applied a sliding window approach in our study where we use 1000 samples of our multi-dimensional input time series to predict the future output.

To compare the responses, we need a statistical metric. In our study, we defined anomaly using the squared error (Ɛ) from the model predicted value to the actual observation. To further refine the analysis, we set thresholds on the (Ɛ) to highlight the regions of anomaly. For all the analyses, the (Ɛ) is defined as:

$$\text{Ɛ} = (y_i - \hat{y})^2, \qquad 3.5$$

where y_i is the measured value and $\hat{y}$ is the predicted value from a regression model.

Anomaly detection using neural network has been studied by many authors. Callegari et al., (2014) studied the anomalous traffic detection using multilayer feedforward neural network (MFNN) architecture using inputs such as traffic flows and comparing the results with a threshold. Bontemps et al., (2016) studied the anomaly detection for unknown intrusion detection in computer networks using Long Short-Term Memory (LSTM) networks. Kaur et al., (2013) compared major computer based anomaly detection methods such as fuzzy logic based systems, genetic algorithms, neural networks and Bayesian systems. Kaur et al., (2013) highlighted major advantages of neural networks comparable algorithms as ability to generalize patterns from limited data. The pressure data in a CCS project is non-stationary, multi-temporal, and often has stochastic components to it due to field operational changes such as surface pump rates. But there are intrinsic deterministic patterns in the pressure waveform such as increasing and decreasing pressures with increase in injection rates and shut-in respectively. Hence, we choose neural networks combined with a sliding window and squared error as a metric. We set a threshold of 1e-03 on the squared error metric for the anomaly determined based on our results.

Modeling terminology used in the study

For all the deep learning architectures, we used a sliding window (Glumov et al., 1995; Vafaeipour et al., 2014) of 1000 samples to make the future prediction. Sliding window technique is key step in any supervised time series forecasting. A sliding window converts a sequential supervised problem into a classical supervised learning problem (Dietterich, 2002).The windowing of input data using a window classifier h_w creates a window of width which then maps onto an individual output, which is in our case the future pressure value. The half width of a window of size w is defined as d below:

$$d = (w - 1)/2\,, \qquad 3.6$$

Intuitively, larger is the window size, more features in the data can be accommodated. However, in our case the data is limited for training and hence a trade-off must be made on window size versus the size of the training data. A simple method to optimize the window size is to minimize the training MSE over a range of different window sizes. In our study we found by testing various window sizes, that a window of 1000 samples can sufficiently represent the cyclical features present in the pressure data. Smaller windows are incapable of capturing the cyclical events properly, while larger windows do not allow enough data for training and validation.

The inputs for all models are pressure measured at the monitoring well F2, injection rates obtained from injection well F1, and cumulative injection volumes which are computed from the injection rates at well F1. Hence, the input in all cases is a three-dimensional vector of 1000 samples each, and the output is one or multiple pressure data points ahead in the future.

We used 150-min baseline and 90-min leak data for the analyses. The 150-min leak test was not included in this workflow. We implemented an approximately 39,000 data points for analysis after re-sampling of the data. Out of these, roughly 26,000 samples belong to 150-min baseline test and 14,000 samples belong to 90-min leak test. According to best machine learning

practices, we divide our data into train, validation, and test sets. For training, we used 15,000 samples from 150-min baseline test to train the models – roughly 40% of the data. For the validation set, we used the remainder of 11,000 samples from 150-min baseline test. For test set we use all 14,000 samples from 90-min leak test. These data points are separated by various color arrows showing the train, test, baseline and leak boundaries on all result plots. The color coding of the data is discussed in the results section.

The models are not trained on the leak test data, because the ultimate goal of our study is to determine the anomalous pressure behavior as compared to a baseline test. If the leak test was included in the analysis, the model would not be able to distinguish between the baseline and leak test. In addition, we used mean squared error (MSE) as a loss metric in all models. The chronological order of the tests shown in Table 3.1 shows that the 90-min leak test follows the 150-min baseline test. As the reservoir pressure is a dynamic property and changes with injection, we used the 150-min baseline and 90-min leak in the analysis in chronological order.

The neural networks are intrinsically stochastic hence, we generated 20 realizations of each model to report the average training MSE and computational time. In all the architectures described in the following sections we tested the model performance in terms of MSE and computational time to achieve best results in the least computational times. To avoid over-fitting, we used 20% dropout in hidden layers (Srivastava et al., 2014). Also, we attempted to forecast multiple points in the future in a multivariate and multi-step fashion, to test the ability of a model to forecast more than one sample in the future.

In our study, field data showing pressure behavior with baseline and leak scenario are available. For our models to be field-deployable they should detect anomaly during a pulse test as well as live continuous pressure feeds from permanent down-hole pressure gauges. To compute

the anomaly, an actual pressure and a predicted pressure is required according to Eq. 3.5. Hence, we use a multi-step forecast. A model which can forecast 1000 samples in the future with low MSE loss in training and validation has a better scalability than a model that can only forecast just 100 samples. If a model can forecast multiple samples reliably on different pulse tests, this forecast can be used as a proxy for expected reservoir pressure behavior. When the pressure gauge records the future samples, it can then be used to compute the anomaly. If the model observes no anomalous behavior under given error threshold, samples recorded by pressure gauge can be included in training and model can be updated and the next forecast can be made. This process eliminates the need for a reservoir simulator to generate the forecasts for complicated reservoirs with dynamic operational conditions such as injection rates.

Multilayer feedforward neural network (MFNN)

An MFNN is a class of feed forward artificial neural network (ANN) which uses neurons as the basic computational unit (Yilmaz and Kaynar, 2011; Rynkiewicz, 2012; Amid and Mesri Gundoshmian, 2017). An artificial neuron is a computation unit consisting of weighted inputs and outputs. The output signal is further modified by an activation function. The weights on the inputs are analogous to the regression coefficients in simple linear regression. These are often initialized with random values and updated during the "learning process" of the model and are the desired model parameters.

An activation function provides the non-linear component in ANN models. An array of activation functions have been developed over the years such as sigmoid, hyperbolic tangent, rectified linear unit (ReLU), leaky ReLU, Exponential Linear Unit (ELU), etc. (Nwankpa et al., 2018)) discussed these activation functions in detail.

A network of artificial neurons form a layer and modern deep learning architectures use multiple layers of neurons to form complex multi-layer models. One such network architecture is MFNN. The network must have an input layer that receives the inputs, one or multiple hidden layers, and an output layer. The output layer is the layer which produces the desired prediction. The network learns by using backpropagation algorithm(Mozer, 1995; Sathyanarayana, 2014) based on some loss function metric such as MSE. The learning of the network is optimized by an algorithm that reduces the loss according to the gradients computed.

After testing both the number of layers and the number of neurons per layer, we selected a MFNN model composed of three hidden layers, respectively with 20, 20, and 10 neurons. The model is built with rectified linear unit (ReLU) as an activation function (Behnke, 2003). The final output is the pressure at the next time step, and hence the last layer has just one output. We used ADAM optimizer(Kingma and Ba, 2014) with a learning rate of 1e–04.

To extend this idea of predicting one point forward in time, we attempted to forecast multiple points in the future to see the ability of the model to predict multiple points before the window is slid to the next set of input data. This technique is known as multivariate, multi-point prediction. The network architecture is similar to the one in multivariate single point described above. We used the 1000 samples to predict the first one, then 10, and finally 1000 samples in the future. The results of the model are summarized in section three. The prediction is not included in the input data to predict the future samples, but the inputs are divided into 1000 samples window and then 1, 10, 100 and 1000 samples in the future are predicted. We use similar multi-step forecasts for all neural network architectures in this study.

Long short-term memory (LSTM)

LSTM falls under recurrent neural network architecture. Recurrent neural networks (RNN) intrinsically have a feedback loop allowing for memory retention in longer sequences of data (Hochreiter et al., 2001; LeCun and others, 2015; Nguyen et al., 2019). RNN family of network architectures thus, are naturally suited for time series prediction. The network consists of memory blocks instead of a single neuron which connect into sequential layers to make dense layers. The memory block is "gated" where each gate has its own activation functions. The gates decide which information to keep or discard. A typical block has three gates, namely, input gate, output gate, and the forget gate. The forget gate decides which information to keep/discard. The input gate updates the memory state, and the output gate decides the output of that memory block. A series of gated memory blocks/cells allows the network to read sequences of data and not just single point.

The LSTM model we use is composed of 100 LSTM units with tanh as the activation function and ADAM as the optimizer, as well as a learning rate of 1e04. Furthermore, to determine the model scalability we followed a similar methodology as in the MFNN architecture, using a window of 1000 samples to forecast one, 10, 100 and then 1000 samples.

Convolutional neural networks (CNN)

A CNN architecture commonly consists of three set of layers: convolutional layers, pooling layers, and fully connected layers. A convolutional layer convolves the input with a series of filters to produce a filtered output. These filters help to extract complex features from the data such as sinusoidal behavior of pressure in our data. The filtered result then is commonly used as input to a pooling layer which reduces the dimensionality of the data. The output from the pooling layer is flattened and passed to a fully connected dense layer or a series of layers (Simonyan and Zisserman, 2014; Zeiler and Fergus, 2014; Gu et al., 2018). The neurons in a CNN architecture,

unlike MFNN, will only be connected to a small region of the output from the previous neurons, but only from a selected “patch” from the previous layer hence reducing the chance of overfitting as compared to the MFNN.

The CNN model consists of a single convolutional layer with filter of size three, and RELU activation function, a max pooling layer of size two and stride two, followed by two fully connected layers comprising 50 neurons and RELU activation function. The output of the network is a single neuron that contains the pressure at a future time step. We included ADAM as the optimizer with a learning rate of 1e04.

Convolutional-LSTM (CONV-LSTM)

As discussed earlier, the merit of LSTM lies in keeping the temporal aspect of sequence data. CNN on the other hand is a powerful architecture for extracting features in both image and sequence data. Therefore, we selected a hybrid version of CNN and LSTM for the pressure forecasting. CNN was implemented to extract complex shapes from the time series data, and LSTM to keep track of these features over longer sequences. Hybrid models are very efficient in fulfilling both tasks (Xingjian et al., 2015a).

We divided the 1000 samples window of three features into ten subsets of 100 samples, each which is fed first to the CNN for feature extraction. CNN then forwards this information to LSTM for temporal updates. To achieve this task, we used time distributed layers in the architecture(Xingjian et al., 2015b). The CONV-LSTM model is composed of one convolutional layer with 64 filters with size three and RELU activation function, followed by a max-pooling layer, the output then is flattened and passed as input to a LSTM layer with 50 units that outputs a

sequence. The final LSTM sequence is then used as input to the output layer for the predictions of the pressure data.

Results

Evaluation metrics and terminology

In this section we discuss the results from the various neural network architectures implemented. We used three different types of diagnostic plots to summarize our findings for every model for visual inference. We also calculated MSE for the training, validation, and test data. In each diagnostic plot, the end of the training set is shown by a black arrow and the beginning of 90-min leak test is shown by a brown arrow. The end of the 90-min leak test is shown by a violet arrow. Hereafter, the terminology used for multi-step forecast varies according to the number of samples predicted in the future. For example, MFNN_100 represents the case where 1000 samples fixed window is used to predict 100 samples in the future.

In the first diagnostic plot we compared the actual versus the measured value of the pressure. This plots the pressure prediction to the true value of the pressure. The second plot is the key output of the study which is the point-wise squared error (Eq. (5)) also referred as anomaly plot throughout this section. Higher error between prediction and true value indicates an anomaly. We filtered this error by setting all errors less than 1e03 to zero and amplifying the errors higher than 1e03 by a factor of 100 thus, suppressing the smaller errors and highlighting the larger error greater than the 1e03 threshold. This is referred to as filtered error in the diagnostic plot. The third and final plot show the training losses (MSE) during the model training. Training, validation, and test losses are reported in Table 3.2. We use the following definition of MSE to compute the errors as:

$$MSE = \frac{1}{n}\sum_{i=1}^{i=n}(y_i - \hat{y})^2 \quad , \qquad 3.7$$

where MSE is the mean squared error, y_i is the actual pressure measurement and $\hat{y}$ is the predicted pressure. Higher training MSE values imply a less accurate model during training. We made two different types of comparisons to evaluate the models:

1. Percentage difference between training + validation MSE versus the test MSE: we used the 15,000 (4 h) samples from 150-min baseline to compute the training MSE and the remaining 11,000 samples from 150-min baseline test as validation MSE. We computed the average of these two and calculated the percentage difference with the test set. The test set is the 90-min leak test data of 14,000 samples. Higher difference in MSE between the train + validation versus the test set translates into better segregation of baseline versus leak cases, a desirable condition for an anomaly detector.
2. Models which are scalable: for this criterion, we computed the percentage difference in errors between the one sample forecast and 1000 samples forecast in each architecture category. Lower error spread between the extreme cases implies higher model scalability.

Based on these two criteria, we generated a ranking system for the models assigning the same weight to both the MSE and the scalability criteria (50% each). For the first criterion, we computed a percentage difference of a model with respect to the train and validation average versus test as shown in Eq. (8). Moreover, to determine the normalized difference, we divided all scores by the highest score:

$$Diff = \frac{Model(train+validation\ MSE) - Model\ (test\ MSE)}{test\ MSE} * 100\ \%\ , \qquad 3.8$$

Conversely, for the scalability criterion we calculated the error spread over one sample forecast versus a 1000 sample forecast in each model category as shown in Eq. (9). As higher spread implies

poor scalability, to assign appropriate scores to the models we defined an inverse error spread as presented in Eq. (10). The models are then assigned a rank based on the overall score based on Eq. (11) and reported in Table 3.3. We ignored the computational time for this ranking as it is evident from Table 3.2 that all models except CONV-LSTM require similar computational times:

$$Error\ spread = \frac{Model_1(MSE) - Model_{100}(MSE)}{Model_1(MSE)}, \text{ and} \quad 3.9$$

$$Inverse\ error\ spread = \frac{1}{Error\ spread}. \quad 3.10$$

MFNN results

The results from the MFNN architecture are presented in Figure 3-5. Figure 3-5 a shows the pressure prediction for the MFNN architecture. Results show that MFNN predictions are very close to the true values and the model scales excellently from MFNN_1 through MFNN_1000. MFNN performance is also reported in Tables 3.2 and 3.3 through the MSE spread in different cases ranging from MFNN_1 through MFNN_1000. Figure 3-5 b shows that the baseline and the leakage portions of the data can be easily separated for all the cases throughout MFNN_1 to MFNN_1000. The false alarms are minimal and exist as random spikes in the baseline data as compared to continuous high amplitude in case of leak data. The training losses in Figure 3-5c show that the model converges rapidly over 100 training epochs.

Table 3.2 shows the results summary for all the cases of MFNN. Notice from Table 3.2 that the model has low training and validation MSE. Figure 3-5 shows that the model converges rapidly with higher sample forecasts having higher overall MSE as expected. Table 3.2 shows that the training and validation MSE are consistently low in contrast to the test MSE reflecting the results visually observed in Figure 3-5. The computing times are one of the lowest among all the

models. Table 3.2 shows that MFNN_10 ranks at third position in overall model rankings attributed to the low computational cost and ability to separate the baseline versus the leak data.

CNN results

Figure 3-6 shows the results from the training and the test data from the CNN architecture. Results in Figure 3-6 a show the pressure predictions for the CNN architecture. Notice the CNN predictions in Figure 3-6 accurately match the sinusoidal behavior of the pressure. Also, Figure 3-6 a shows that the pressure predictions deteriorate as number of future samples prediction increases. For example, the model prediction is excellent in case of CNN_100 but poor for CNN_1000 suggesting limited model scalability. This is further reflected in Figure 3-6 b where CNN_1000 errors are shown in black. This fact is further bolstered from Figure 3-6c where training errors are higher for CNN_1000.

Tables 3.2 and 3.3 capture these results – with the exception of CNN_1000 – all CNN models have excellent performance; with low MSE for training and validation and higher test MSE values which make these effective models for anomaly detection. From Table 3.3 of model rankings CNN_10 ranks second out of thirteen models tested overall due to its reasonable computing times and ability to separate the baseline versus the leak tests.

LSTM results

The LSTM results are summarized in Figure 3-7. Figure 3-7a shows the pressure prediction from the LSTM architecture. The predicted pressures are extremely close to the true pressure for training and validation. The MSE increases gradually for higher samples multi-step forecast. Notice in Figure 3-7b the sharp contrast in anomaly behavior of LSTM_1 in baseline versus leak case. Although the computational times are higher for LSTM compared to the MFNN architecture,

the baseline and leak sections are better highlighted with low training and validation MSE (see Table 3.2).

From Table 3.3 of model rankings, LSTM_1 is the top performer among all the thirteen models tested, primarily due to its ability to differentiate between the baseline and leak tests. Table 3.2 shows the model is scalable and training and test MSE are consistent with the model variants. For example, the error increases gradually from LSTM_1 to LSTM_1000 as compared to CNN architecture where errors increase sharply after CNN_100. The false alarms in LSTM are minimal. However, one patch of false alarm can be noticed in Figure 3-7b. Comparing it to Figure 3-7a, note the sharp change in the pressure in this region between four to five hours into the 150-min baseline test. These changes are often operational and caused due to pump shutdown and pump shut-ins at surface. Notice in Figure 3-7 a and b these changes are captured very efficiently from CNN_1 through CNN_100 suggesting a combination of the CNN and LSTM architectures can be a possible solution providing best of CNN and LSTM architectures.

CONV-LSTM results

The results of CONV-LSTM are presented in Figure 3-8. In Figure 3-8, we observed that CONV-LSTM is able to capture the sinusoidal features of the pressure signal efficiently. As the architecture utilizes features of both LSTM and CNN, the training MSE is lowest among all the thirteen models tested in this study (see Table 3.2). However, the validation and test MSE are higher which suggests overfitting. The computational time for CONV-LSTM is the highest of the architectures discussed in this study. Although the model converges rapidly and requires fewer training epochs it still takes significantly more time than the other algorithms. Notice in Figure 3-8c CONV-LSTM provides training MSE one order smaller compared to any other architecture used in this study in just 25 epochs as compared to 100 epochs utilized for training of other

architectures such as MFNN, CNN, and LSTM. The filtered error anomaly shows that the architecture can differentiate between the baseline and the leak test efficiently (Figure 3-8 b). Notice the false alarm patch present in LSTM is now reduced but not completely eliminated. The CONV-LSTM is not ranked with other models in the study as the computational time alone makes the architecture worst among all the models.

Detection model generalization to different scenarios

We discussed earlier in the manuscript the dynamic nature of the hydrocarbon reservoir under CO injection. As the injection continues, the average reservoir pressure increases, and the stresses across the sealing faults, or abandoned wells, change as do rock properties such as porosity and permeability. In addition to rock properties such as porosity and permeability. These properties can change as function of the stress regime as well. Hence, a natural question arises: can our workflow be applied to any other pulse test?

To answer this question, we included another baseline test. Thus, the updated data consists of a 90-min baseline test, a 150-min baseline test and a 90-min leak test. Previously, we considered for the analysis only a 150-min baseline test and a 90-min leak test. The first 10,000 samples used come from the 90-min baseline test to train the MFNN model, the end of the training is shown by the black arrow in Fig. 3.9a. In addition, we indicated in the trained model from first 10,000 samples belonging to 90-min baseline, we predicted the pressure behavior for both 150-min baseline test and the 90-min leak test. The results are summarized in Figure 3-9a and b. We noticed that when the model is trained only on 90-min baseline test, it identifies the 150-min baseline as well as the 90-min leak test, both as an anomaly. MSE is higher in the case of the 90-min leak test than 150-min baseline test, but both are much greater than the 90-min baseline test on which the

model is trained. Hence, as expected, a model can only learn from the training data distribution and will address everything else as an anomaly.

Now, a second question arises, can the model learn from a set of new data and incorporate the additional features and reservoir behavior at different states? To answer this question, we included only one-half cycle of the 150-min baseline test into our training in addition to 10,000 samples from 90-min baseline test. This is shown in Figure 3-8 c and d. Results show that the model can now easily distinguish between the baseline and leak tests effectively. Thus, if a small portion of a different test data set is included in the training, the model is capable of learning new features, demonstrating that deep learning architectures are powerful enough to learn new features quickly. However, as expected, they can only make a prediction based on the data used for training. Deep learning models lack intuition, unlike human interpreters. Hence, for full-scale field deployment, the models have to be trained on a particular set of test parameters.

Based on the model's capability on multi-step forecasting and results discussed throughout this study, we can infer that a deep learning model can be used for future pressure prediction with confidence (see Tables 3.2 and 3.3). As the system is dynamic, once a forecast with the high degree of confidence interval has already occurred, these data must be included in the training, and the model has to be updated. As our workflow utilizes time-series data, the additional training is fast and efficient, making the model practically scalable and field-deployable.

Conclusions

In this work we showed that anomaly detectors based on neural networks can be useful to provide early warning for CO_2 leaks which can be an invaluable tool to prevent further leakage and improved the safety of carbon sequestration projects. We demonstrated the applications of various neural network architectures for anomaly detection tasks such as MFNN, CNN, LSTM,

and CONV-LSTM. To compare all the models, we have designed a ranking system based on multiple quantitative and qualitative criteria such as MSE, scalability of the model, and practicality in a field deployment scenario. In our study LSTM is ranked highest followed by CNN then MFNN. The MFNN architecture performs well for training, validation, and test data but raises some false flags. We observed that the LSTM architecture performs the best overall, but the CNN makes an excellent case for capturing the sinusoidal waveform like features in our pulse test data. The hybrid architecture CONV-LSTM provides the best of both worlds, however, the computational times for CONV-LSTM can be a limitation. Carbon sequestration projects are a dynamic system affected by complex geology and operational parameters and all machine learning algorithms, including deep learning, learn from training data and are limited by the availability of the training data. We show that deep learning architectures can adapt to new set of data and can be used in wide variety of monitoring projects and field conditions making deep learning-based anomaly detectors a cheap and efficient alternative to human resources.

Acknowledgments

We would like to thank U.S. DOE Office of Fossil Energy's Carbon Storage program to provide funding for this project.

Figures for chapter 3

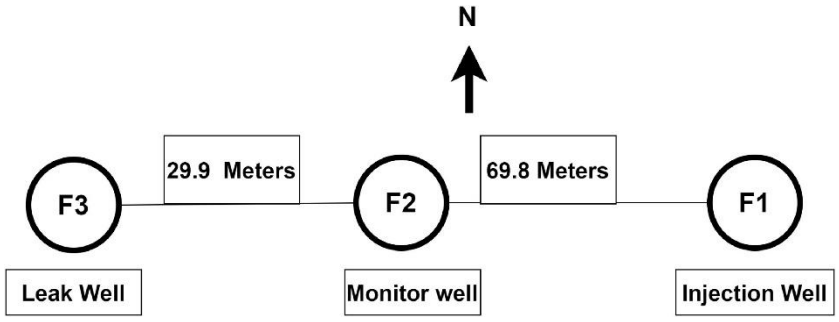


Figure 3-1 Schematic illustration of the wells used in this study and their configuration. F1 is the injection well, F2 is the monitoring well and F3 is the well where leak is introduced. Pressure utilized in this study is obtained from the well, F2. Leakage rate at well F3 is 60 kg/min or 724 Bbl/min. The injection rate is 300 kg/min or 3621 Bbl/day.

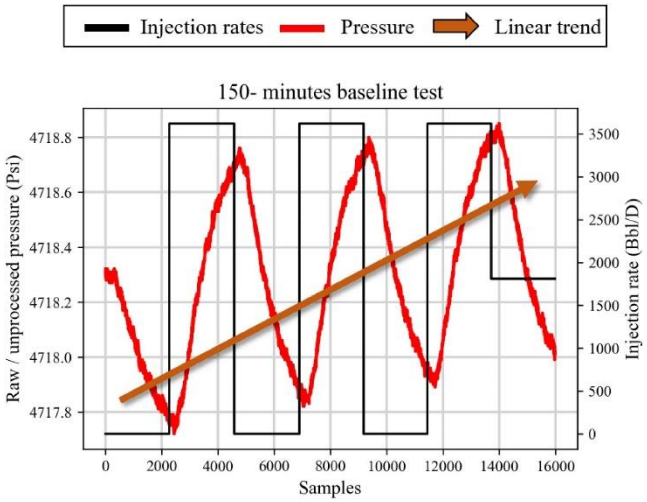


Figure 3-2 Unprocessed/raw pressure data obtained from the pressure gauge installed at well F2. The rates are plotted on secondary axis. The pressures exhibit a linear upward trend due to continued injection. A detrending is required for the pressure data.

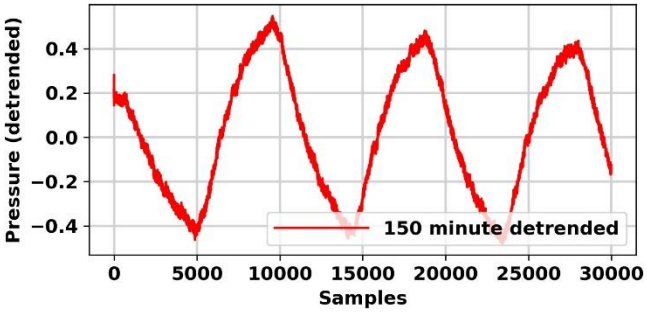


Figure 3-3 150-min baseline test pressure after detrending and re-sampling the data. Notice that the linear trend from present in Fig. 3.3-2 is now removed.

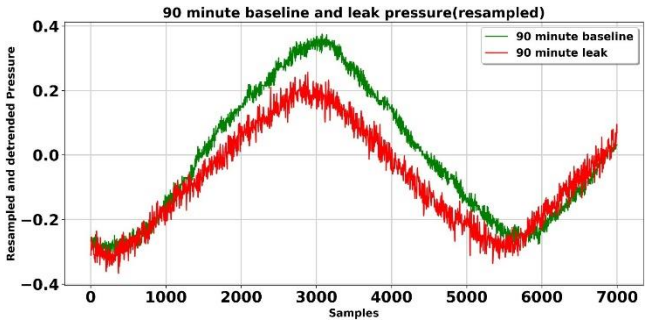


Figure 3-4 Comparison of pressure response between the leak versus non leak 90-min test obtained from pressure gauge installed at well F2. Figure shows that the pulse test pressure response can distinguish between the baseline versus leak tests.

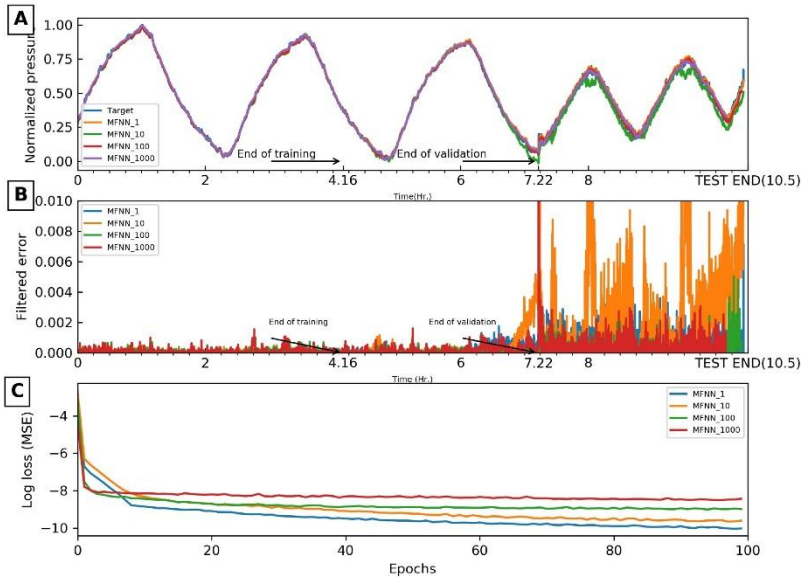


Figure 3-5 MFNN models results summary. (a) Pressure predictions versus the true pressure value. (b) Pressure anomaly. (c) Training losses. The false alarms manifest as random spikes in the data, hence, can be easily identified by a human interpreter.

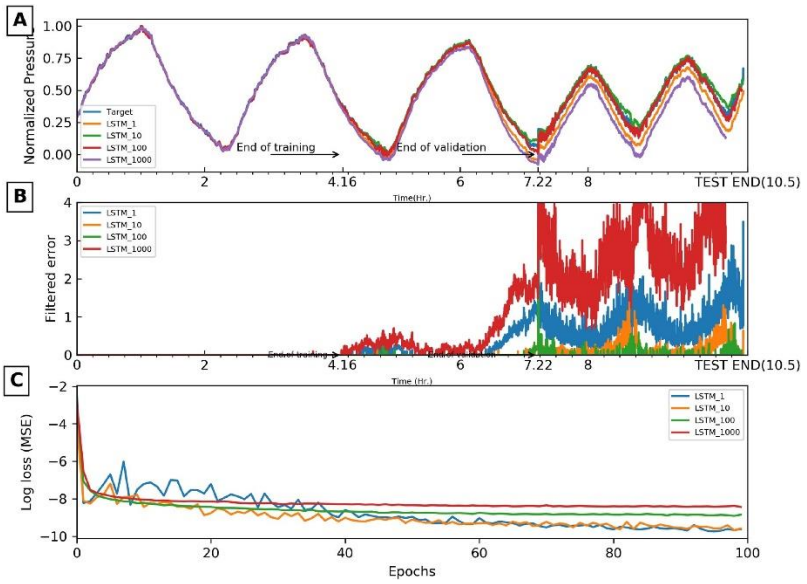


Figure 3-6 CNN results. (a) Pressure predictions versus the true pressure value. (b) Pressure anomaly. (c) Training losses. Notice that CNN can capture the sinusoidal behavior of pressure efficiently. Also notice the sharp change in the training MSE from CNN_100 to CNN_1000.The sharp change in the training MSE at epoch 33 is caused by mini-batch gradient descent in the optimizer ADAM.

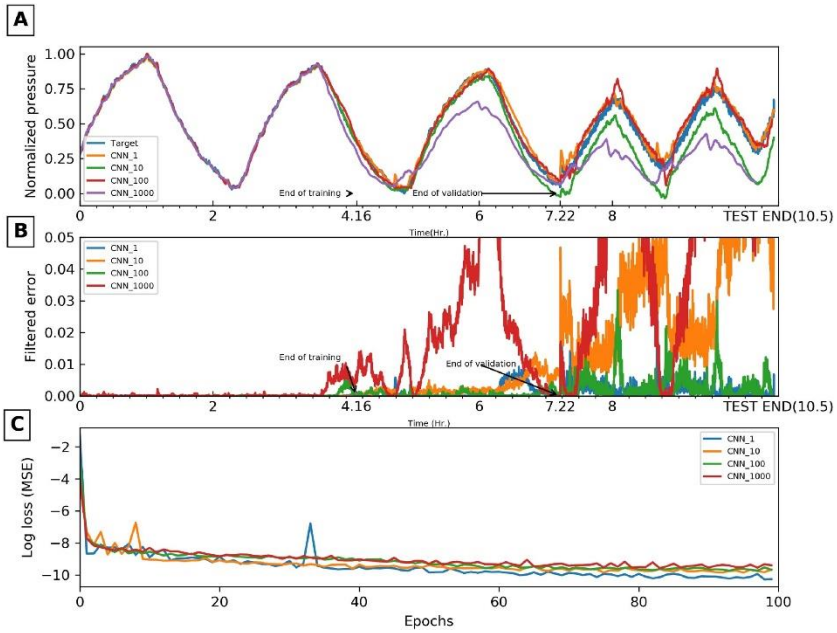


Figure 3-7 LSTM results summary. (a) Pressure prediction from LSTM architecture. (b) Pressure anomaly. (c) Training losses. The predictions are highly accurate in case of LSTM and there is a sharp contrast in baseline versus leak data. A patch of false leak is indicated in the anomaly plot.

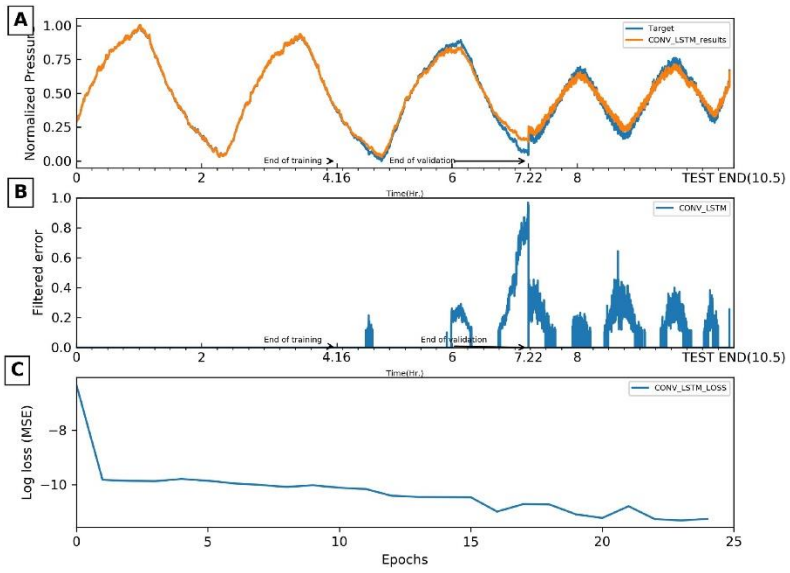


Figure 3-8 CONV-LSTM pressure predictions. (a) Pressure predictions versus the true pressure value. (b) Pressure anomaly. (c) Training losses. Notice the training losses with number of epochs which drop sharply for CONV-LSTM as compared to any other architecture. The false alarm patch present in LSTM results is now reduced but not completely eliminated.

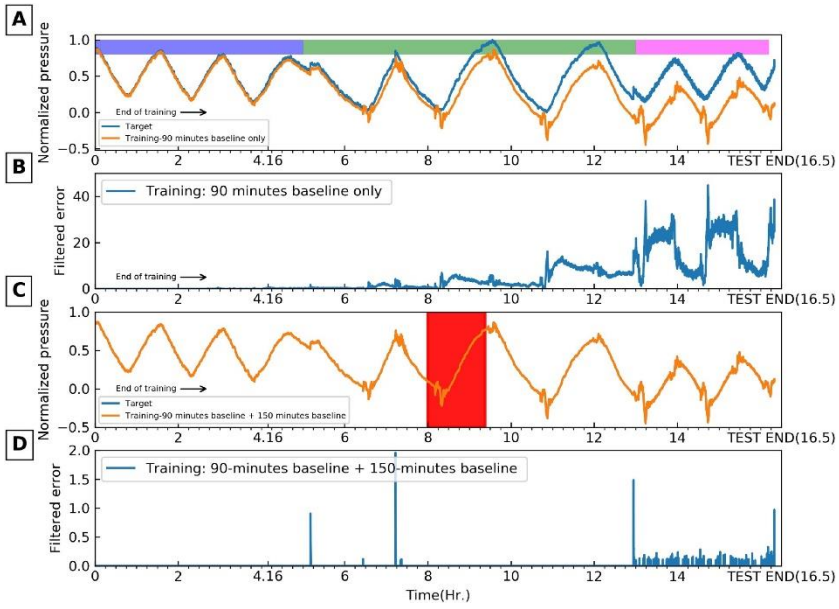


Figure 3-9 Results summary for extension scenarios. (a) Pressure prediction using MFNN_1 when only 90-min baseline test is used for training. (b) Pressure anomaly when only 90-min baseline test is used for training. (c) Pressure prediction when 5000 samples from 150-min baseline are added to existing training data. (d) Pressure anomaly when

5000 samples from baseline are added to the training data. The additional 5000 samples used for training are highlighted using a red box in panel C. The 90-min baseline test, 150-min baseline test and 90-min leak test is shown by blue, green and magenta color bars in panel A. Notice that when only 90-min baseline test is used for training, the model identifies 150-min baseline test as anomaly in addition to the 90-min leak test. When a small number of extra training samples from 150-min baseline test are added to training, the results improve significantly.

Tables for chapter 3

Test	Date	Rate	Pulse
Baseline – 90 min	19th January 2015	3620	45
Baseline – 150 min	20th January 2015	3620	75
Leak – 90 min	30th January 2015	3620	45
Leak – 150 min	31st January 2015	3620	75

Table 3-1 Injection schedule for the baseline and leak test. Pulse duration is identical in respective baseline and leak tests. All rates are measured in Bbl/D and the pulse half cycle times are in minutes.

Model	Training MSE	Validation MSE	T + V MSE	Test MSE	Epochs	Time (s)
MFNN_1	3.60E−05	6.52E−04	3.44E−04	8.99E−04	100	82
MFNN_10	7.07E−05	6.66E−04	3.68E−04	2.65E−03	100	83
MFNN_100	6.60E−05	**9.77E−05**	**8.18E−05**	**3.15E−04**	**100**	**81**
MFNN_1000	1.06E−04	6.78E−04	3.92E−04	4.77E−04	100	80
CNN_1	6.34E−05	1.17E−03	6.17E−04	2.17E−03	100	328
CNN_10	4.89E−05	2.76E−03	1.41E−03	3.25E−02	100	332
CNN_100	1.88E−04	6.60E−04	4.24E−04	3.53E−03	100	334
CNN_1000	1.94E−03	2.80E−02	1.50E−02	3.80E−02	100	328
LSTM_1	4.06E−05	1.99E−03	1.01E−03	3.28E−02	100	226

LSTM_10	3.96E−05	2.24E−04	1.32E−04	1.45E−03	100	213
LSTM_100	4.67E−05	1.65E−04	1.06E−04	4.78E−04	100	209
LSTM_1000	9.16E−05	5.17E−03	2.63E−03	2.66E−02	100	193
CONV-LSTM_1	**7.93E−06**	1.18E−03	5.96E−04	1.19E−03	25	3800

Table 3-2 Results summary for all models used in the study. T + V refers to training and validation averaged MSE. All models are run using an NVIDIA GTX super 2580 GPU unit.

Model	Error Spread	Inverse spread	Percentage Difference (T + V versus test)	Normalized difference	Score	Rank
LSTM_1	**126**	**7.94E−03**	**3139**	**1.25E+01**	**6.26E+00**	**1**
CNN_10	2960	3.38E−04	2211	8.81E+00	4.40E+00	2
MFNN_10	40845	2.45E−05	1116	4.45E+00	2.22E+00	3
LSTM_10	126	7.94E−03	998	3.98E+00	1.99E+00	4
LSTM_1000	126	7.94E−03	911	3.63E+00	1.82E+00	5
CNN_100	2960	3.38E−04	732	2.92E+00	1.46E+00	6
LSTM_100	126	7.94E−03	352	1.40E+00	7.04E−01	7
MFNN_100	40845	2.45E−05	285	1.13E+00	5.67E−01	8
CNN_1	2960	3.38E−04	251	1.00E+00	5.00E−01	9
MFNN_1	40845	2.45E−05	162	6.44E−01	3.22E−01	10
CNN_1000	2960	3.38E−04	154	6.14E−01	3.07E−01	11
MFNN_1000	40845	2.45E−05	22	8.59E−02	4.30E-02	12
CONV-LSTM_1	NA	99	3.94E-01	NA	NA	

Table 3-3 Summary of the results for all models used in the study. T + V refers to training and validation averaged MSE. LSTM model outperforms all models in overall ranking.

References

Amid, S., and T. Mesri Gundoshmian, 2017, Prediction of output energies for broiler production using linear regression, ANN (MLP, RBF), and ANFIS models: Environmental Progress & Sustainable Energy, **36**, 577–585.

Bachu, S., 2000, Sequestration of CO\ensuremath_2 in geological media: criteria and approach for site selection in response to climate change: Energy Conversion and Management, **41**, 953–970.

Bachu, S., 2003, Screening and ranking of sedimentary basins for sequestration of CO\ensuremath_2 in geological media in response to climate change: Environmental Geology, **44**, 277–289.

Behnke, S., 2003, Hierarchical Neural Networks for Image Interpretation: Springer.

Bergmann, P., U. Lengler, C. Schmidt-Hattenberger, R. Giese, and B. Norden, 2010, Modelling the geoelectric and seismic reservoir response caused by carbon dioxide injection based on multiphase flow simulation: Results from the CO_2SINK project: Chemie Der Erde - Geochemistry, **70**, 173–183.

Bontemps, L., V. L. Cao, J. McDermott, and N.-A. Le-Khac, 2016, Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks, *in* Future Data and Security Engineering, Springer International Publishing, 141–152.

Brigham, W. E., 1970, Planning and Analysis of Pulse-Tests: Journal of Petroleum Technology, **22**, 618–624.

Callegari, C., S. Giordano, and M. Pagano, 2014, Neural network based anomaly detection: 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD).

Castelletto, N., G. Gambolati, and P. Teatini, 2013, Geological CO_2 sequestration in multi-compartment reservoirs: Geomechanical challenges: Journal of Geophysical Research: Solid Earth, **118**, 2417–2428.

Chandola, V., A. Banerjee, and V. Kumar, 2009, Anomaly detection: ACM Computing Surveys, **41**, 1–58.

Dietterich, T. G., 2002, Machine learning for sequential data: A review: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), 15–30.

Dondurur, D., 2018, Acquisition and Processing of Marine Seismic Data: Elsevier.

Fernández-Montiel, I., M. Touceda, A. Pedescoll, R. Gabilondo, A. Prieto-Fernández, and E. Bécares, 2015, Short-term effects of simulated below-ground carbon dioxide leakage on a soil microbial community: International Journal of Greenhouse Gas Control, **36**, 51–59.

Fokker, P. A., and F. Verga, 2011, Application of harmonic pulse testing to water–oil displacement: Journal of Petroleum Science and Engineering, **79**, 125–134.

Fokker, P. A., E. S. Borello, F. Verga, and D. Viberti, 2018, Harmonic pulse testing for well performance monitoring: Journal of Petroleum Science and Engineering, **162**, 446–459.

Gal, F., Z. Pokryszka, N. Labat, K. Michel, S. Lafortune, and A. Marblé, 2019, Soil-Gas Concentrations and Flux Monitoring at the Lacq-Rousse CO\ensuremath_2-Geological Storage Pilot Site (French Pyrenean Foreland): From Pre-Injection to Post-Injection: Applied Sciences, **9**, 645.

Gaus, I., 2010, Role and impact of CO\ensuremath_2–rock interactions during CO\ensuremath_2 storage in sedimentary rocks: International Journal of Greenhouse Gas Control, **4**, 73–89.
Glumov, N., E. Kolomiyetz, and V. Sergeyev, 1995, Detection of objects on the image using a sliding window mode: Optics & Laser Technology, **27**, 241–249.
Gu, J., Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and others, 2018, Recent advances in convolutional neural networks: Pattern Recognition, **77**, 354–377.
Harris, F. J., 1978, On the use of windows for harmonic analysis with the discrete Fourier transform: Proceedings of the IEEE, **66**, 51–83.
Hawthorn, A., S. Aguilar, and others, 2017, New Wireless Acoustic Telemetry System Allows Real-time Downhole Data Transmission through Regular Drillpipe: SPE Annual Technical Conference and Exhibition.
Heideman, M., D. Johnson, and C. Burrus, 1984, Gauss and the history of the fast Fourier transform: IEEE ASSP Magazine, **1**, 14–21.
Hochreiter, S., Y. Bengio, P. Frasconi, J. Schmidhuber, and others, 2001, Gradient flow in recurrent nets: the difficulty of learning long-term dependencies: .
Ivanova, A., A. Kashubin, N. Juhojuntti, J. Kummerow, J. Henninges, C. Juhlin, S. Lüth, and M. Ivandic, 2012, Monitoring and volumetric estimation of injected CO\ensuremath_2 using 4D seismic, petrophysical data, core measurements and well logging: a case study at Ketzin, Germany: Geophysical Prospecting, **60**, 957–973.
Kaur, H., G. Singh, and J. Minhas, 2013, A review of machine learning based anomaly detection techniques: ArXiv Preprint ArXiv:1307.7286.
Kingma, D. P., and J. Ba, 2014, Adam: A method for stochastic optimization: ArXiv Preprint ArXiv:1412.6980.
LeCun, Y. and others, 2015, LeNet-5, convolutional neural networks: URL: Http://Yann. Lecun. Com/Exdb/Lenet, **20**, 14.
de Lima, R. P., Y. Lin, and K. J. Marfurt, 2019a, Transforming seismic data into pseudo-RGB images to predict CO_2 leakage using pre-learned convolutional neural networks weights, *in* SEG Technical Program Expanded Abstracts 2019, Society of Exploration Geophysicists, 2368–2372.
de Lima, R. P., Y. Lin, K. J. Marfurt, and others, 2019b, Transforming seismic data into pseudo-RGB images to predict CO_2 leakage using pre-learned convolutional neural networks weights: SEG International Exposition and Annual Meeting.
Macquet, M., D. C. Lawton, J. Donags, and J. Barraza, 2017, Feasibility study of time-lapse-seismic monitoring of CO_2 sequestration: EAGE/SEG Research Workshop 2017, cp-522.
May, F., and S. Waldmann, 2014, Tasks and challenges of geochemical monitoring: Greenhouse Gases: Science and Technology, **4**, 176–190.
Moore, J., M. Adams, R. Allis, S. Lutz, and S. Rauzi, 2005, Mineralogical and geochemical consequences of the long-term presence of CO\ensuremath_2 in natural reservoirs: an example from the Springerville–St. Johns Field, Arizona, and New Mexico, USA: Chemical Geology, **217**, 365–385.
Mozer, M. C., 1995, A focused backpropagation algorithm for temporal: Backpropagation: Theory, Architectures, and Applications, **137**.
Nguyen, M., T. He, L. An, D. C. Alexander, J. Feng, B. T. Yeo, A. D. N. Initiative, and others, 2019, Predicting Alzheimer's disease progression using deep recurrent neural networks: BioRxiv, 755058.

Nwankpa, C., W. Ijomah, A. Gachagan, and S. Marshall, 2018, Activation functions: Comparison of trends in practice and research for deep learning: ArXiv Preprint ArXiv:1811.03378.

Oelkers, E. H., S. R. Gislason, and J. Matter, 2008, Mineral carbonation of CO\ensuremath_2: Elements, **4**, 333–337.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, 2011, Scikit-learn: Machine Learning in Python: Journal of Machine Learning Research, **12**, 2825–2830.

Reeves, M. E., P. L. Camwell, J. McRory, and others, 2011, High speed acoustic telemetry network enables real-time along string measurements, greatly reducing drilling risk: Offshore Europe.

Roach, L. A., D. J. White, and B. Roberts, 2015, Assessment of 4D seismic repeatability and CO\ensuremath_2 detection limits using a sparse permanent land array at the Aquistore CO\ensuremath_2 storage site: Geophysics, **80**, WA1–WA13.

Rutqvist, J., 2012, The geomechanics of CO\ensuremath_2 storage in deep sedimentary formations: Geotechnical and Geological Engineering, **30**, 525–551.

Rynkiewicz, J., 2012, General bound of overfitting for MLP regression models: Neurocomputing, **90**, 106–110.

Sathyanarayana, S., 2014, A gentle introduction to backpropagation: Numeric Insight, **7**, 1–15.

Selma, L., O. Seigo, S. Dohle, and M. Siegrist, 2014, Public perception of carbon capture and storage (CCS): A review: Renewable and Sustainable Energy Reviews, **38**, 848–863.

Shao, H., D. A. Ussiri, C. G. Patterson, R. A. Locke II, H. Wang, A. H. Taylor, and H. F. Cohen, 2019a, Soil gas monitoring at the Illinois Basin–Decatur Project carbon sequestration site: International Journal of Greenhouse Gas Control, **86**, 112–124.

Shao, H., D. A. Ussiri, C. G. Patterson, R. A. Locke II, H. Wang, A. H. Taylor, and H. F. Cohen, 2019b, Soil gas monitoring at the Illinois Basin–Decatur Project carbon sequestration site: International Journal of Greenhouse Gas Control, **86**, 112–124.

Simonyan, K., and A. Zisserman, 2014, Very deep convolutional networks for large-scale image recognition: ArXiv Preprint ArXiv:1409.1556.

Sinha, S., R. P. de Lima, Y. Lin, A. Y. Sun, N. Symons, R. Pawar, and G. Guthrie, 2020a, Normal or abnormal? Machine learning for the leakage detection in carbon sequestration projects using pressure field data: International Journal of Greenhouse Gas Control, **103**, 103189.

Sinha, S., R. Pires De Lima, Y. Lin, A. Y Sun, N. Symon, R. Pawar, and G. Guthrie, 2020b, Leak Detection in Carbon Sequestration Projects Using Machine Learning Methods: Cranfield Site, Mississippi, USA: SPE Annual Technical Conference and Exhibition.

Smith, K. L., M. D. Steven, D. G. Jones, J. M. West, P. Coombs, K. A. Green, T. S. Barlow, N. Breward, S. Gwosdz, M. Krüger, S. E. Beaubien, A. Annunziatellis, S. Graziani, and S. Lombardi, 2013, Environmental impacts of CO\ensuremath_2 leakage: recent results from the ASGARD facility, UK: Energy Procedia, **37**, 791–799.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014, Dropout: a simple way to prevent neural networks from overfitting: The Journal of Machine Learning Research, **15**, 1929–1958.

Stork, A. L., C. Allmark, A. Curtis, J.-M. Kendall, and D. J. White, 2018, Assessing the potential to use repeated ambient noise seismic tomography to detect CO_2 leaks: Application to the Aquistore storage site: International Journal of Greenhouse Gas Control, **71**, 20–35.

Sun, A. Y., J. Lu, and S. Hovorka, 2015, A harmonic pulse testing method for leakage detection in deep subsurface storage formations: Water Resources Research, **51**, 4263–4281.

Sun, A. Y., A. Kianinejad, J. Lu, and S. Hovorka, 2014, A frequency-domain diagnosis tool for early leakage detection at geologic carbon sequestration sites: Energy Procedia, **63**, 4051–4061.

Sun, A. Y., J. Lu, B. M. Freifeld, S. D. Hovorka, and A. Islam, 2016, Using pulse testing for leakage detection in carbon storage reservoirs: A field demonstration: International Journal of Greenhouse Gas Control, **46**, 215–227.

Vafaeipour, M., O. Rahbari, M. A. Rosen, F. Fazelpour, and P. Ansarirad, 2014, Application of sliding window technique for prediction of wind velocity time series: International Journal of Energy and Environmental Engineering, **5**, 105.

Verdon, J. P., J.-M. Kendall, and S. C. Maxwell, 2010a, A comparison of passive seismic monitoring of fracture stimulation from water and CO\ensuremath_2 injection: Geophysics, **75**, MA1–MA7.

Verdon, J. P., J.-M. Kendall, and S. C. Maxwell, 2010b, A comparison of passive seismic monitoring of fracture stimulation from water and CO\ensuremath_2 injection: Geophysics, **75**, MA1–MA7.

Xingjian, S., Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W. Woo, 2015a, Convolutional LSTM network: A machine learning approach for precipitation nowcasting: Advances in Neural Information Processing Systems, 802–810.

Xingjian, S., Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W. Woo, 2015b, Convolutional LSTM network: A machine learning approach for precipitation nowcasting: Advances in Neural Information Processing Systems, 802–810.

Yilmaz, I., and O. Kaynar, 2011, Multiple regression, ANN (RBF, MLP) and ANFIS models for prediction of swell potential of clayey soils: Expert Systems with Applications, **38**, 5958–5966.

Yu, Y., Y. Zhu, S. Li, and D. Wan, 2014, Time series outlier detection based on sliding window prediction: Mathematical Problems in Engineering, **2014**.

Zeiler, M. D., and R. Fergus, 2014, Visualizing and understanding convolutional networks: European Conference on Computer Vision, 818–833.

Zhong, Z., A. Y. Sun, Q. Yang, and Q. Ouyang, 2019, A deep learning approach to anomaly detection in geological carbon sequestration sites using pressure measurements: Journal of Hydrology, **573**, 885–894.

Zhou, Z., Y. Lin, Y. Wu, Z. Wang, R. Dilmore, and G. Guthrie, 2018, Spatial-temporal densely connected convolutional networks: An application to CO\ensuremath_2 leakage detection: Proceeding of Society of Exploration Geophysics, 2136–2140.

Zhou, Z., Y. Lin, Z. Zhang, Y. Wu, Z. Wang, R. Dilmore, and G. Guthrie, 2019, A data-driven CO\ensuremath_2 leakage detection using seismic data and spatial–temporal densely connected convolutional neural networks: International Journal of Greenhouse Gas Control, **90**, 102790.

Chapter 4 : PreMevE Update: Forecasting Ultra-Relativistic Electrons inside Earth's Outer Radiation Belt

Saurabh Sinha[*1,2], Yue Chen[†1], Youzuo Lin[1], and Rafael Pires de Lima[3]

[1]Los Alamos National Laboratory, Los Alamos, New Mexico, USA

[2]University of Oklahoma, Norman, OK, USA

[3]Geological Survey of Brazil, São Paulo, Brazil

Preface

This chapter is presented as published in AGU space weather journal (Sinha et al., 2021) which itself is based on the EGU abstract (Chen et al., 2021). This chapter presents the deep learning methods to predict the ultra-relativistic electron flux distributions during MeV electron events. This new PreMevE-2E model makes reliable 1- and 2- day ensemble forecasts of ≥2 MeV electrons inside Earth's outer radiation belt.

References

Chen, Y., R. Pires de Lima, S. Sinha, and Y. Lin, 2021, PreMevE: A Machine-Learning Based Predictive Model for MeV Electrons inside Earth's Outer Radiation Belt: EGU General Assembly Conference Abstracts, EGU21-8545.

Sinha, S., Y. Chen, Y. Lin, and R. P. de Lima, 2021, PreMevE Update: Forecasting Ultra-relativistic Electrons inside Earth's Outer Radiation Belt: ArXiv Preprint ArXiv:2104.09055.

Abstract

Energetic electrons inside Earth's outer Van Allen belt pose a major radiation threat to space-borne electronics that often play vital roles in our modern society. Ultra-relativistic electrons with energies greater than or equal to two Megaelectron-volt (MeV) are of particular interest due to their high penetrating ability, and thus forecasting these ≥2 MeV electron levels has significant meaning to all space sectors. Here we update the latest development of the predictive model for MeV electrons inside the Earth's outer radiation belt. The new version, called PreMevE-2E, focuses on forecasting ultra-relativistic electron flux distributions across the outer radiation belt, with no need for in-situ measurements of the trapped MeV electron population except at geosynchronous (GEO) orbit. Model inputs include precipitating electrons observed in low-Earth-orbits by NOAA satellites, upstream solar wind conditions (speeds and densities) from solar wind monitors, as well as ultra-relativistic electrons measured by one Los Alamos GEO satellite. We evaluated a total of 32 supervised machine learning models that fall into four different classes of linear and neural network architectures, and also successfully tested ensemble forecasting by using groups of top-performing models. All models are individually trained, validated, and tested by in-situ electron data from NASA's Van Allen Probes mission. It is shown that the final ensemble model generally outperforms individual models at most L-shells, and this PreMevE-2E model is able to provide 25-hr (~1-day) and 50-hr (~2-day) forecasts with high mean performance efficiency and correlation values. Our results also suggest this new model is dominated by non-linear components at low L-shells (< ~4) for ultra-relativistic electrons, which is different from the dominance of linear components at all L-shells for 1 MeV electrons as previously discovered.

Introduction

Since their discovery in 1958, energetic particles inside the Earth's Van Allen radiation belts have been a top concern for space operations, including the Apollo missions in the early years of the Space Age. Since then, our interests in these magnetically trapped electrons and protons have been repeatedly refreshed, and our understanding of these belt particles has deepened through six decades of observations. It is well recognized that these particles usually present in a two-belt distribution—an inner belt in the region with equatorial distances (i.e., L-shells or simply L) within ~2-3 Earth radii and an outer belt with $\sim 3 < L < 8$ separated by the slot region in between. This general picture for the electron belts has been widely adopted by the aerospace industry, as specified by empirical models such as the AE8 (Vette, 1991). Starting in 1990, observations from the Combined Radiation Release and Effects Satellite—in particular the deep injection of Megaelectron-volt (MeV) electrons during the March 1991 event (Blake et al., 1992)—reignited research interest in understanding the dynamics of outer belt electrons, whose intensities may vary up to several orders of magnitude during magnetic storms. Recently, observations from Van Allen Probes (also called RBSP) again surprised the space community by showing the persistent absence of MeV electrons inside the inner belt (Fennell et al., 2015 and Claudepierre et al., 2015).

Indeed, for satellites operating in geosynchronous orbit (GEO), geosynchronous-transfer-orbit (GTO), medium- and high-earth-orbits (MEO and HEOs) with high apogees, energetic electrons inside the outer belt pose a major space radiation risk, not only in term of the ionizing dose, but also due to the deep dielectric charging and discharging phenomena (Reagan et al. 1983). When space systems are irradiated, some of the electrons are energetic enough to penetrate through satellite surfaces (e.g., ranges of 2.0 and 3.0 MeV electrons inside Aluminum are 4.53 and 6.92 mm, respectively), stop, and bury themselves inside the dielectric materials of electronic parts on board. During major MeV

electron events when electron intensities across the outer belt are greatly enhanced and sustained high levels, these buried electrons accumulate faster than they can dissipate, and thus build up high electric fields (a process called "charging", with the potential differences reaching as high as multiple kilovolts), until eventually sudden intense breakdowns occur which result discharge arcs that may cause catastrophic failure to individual electronics or to the satellite as a whole.

Consequently, understanding and forecasting MeV electron events have been a central research topic for radiation belt studies. Recent mounting evidence, particularly from Van Allen Probes, has suggested that local wave-particle interactions play a critical role in energizing seed electrons to MeV energies and above for individual events, but that radial diffusion can also be important (e.g., Li and Hudson, 2019 and references therein). Based on this theoretical framework, a list of first-principles three-dimensional diffusive models have been developed which have shown successes as well as limitations in describing the MeV electron dynamics (see Chen et al., 2016 for a brief review). A different approach has also been proposed and explored which uses precipitating low-energy electrons observed in low-Earth-orbits (LEO) as a proxy for the wave-particle interactions. This new idea of predicting MeV electron events inside the outer belt was first proposed by Chen et al. (2016) using observations over ~260 days to demonstrate feasibility. Chen et al. (2019) then successfully constructed the first PREdictive MEV Electron (PreMevE) model based on simple linear predictive filters. The follow-up study by Pires de Lima et al. (2020) (abbreviated to P2020 hereinafter) advanced the model to PreMevE 2.0, by fusing machine learning (ML) algorithms, which is able to reliably forecast 1 MeV electron distributions across the outer belt. This current study further expands forecasts to electrons with higher energies ≥ 2 MeV, the population with high beta ratio (velocity over light speed) values > ~0.98 and thus also called ultra-relativistic electrons in this work. Other notable work in this area is of Claudepierre and O'Brien (2020) that used multilayer feedforward networks to specify

350 keV and 1 MeV radiation belt electron flux distributions. Their SHELLS model also takes data from a LEO satellite as inputs and is demonstrated to make specifications with high correlations and low errors when compared to observed fluxes.

Another new component of this study is the utilizing of ensemble forecasting, a predictive skill that has been widely adopted in meteorology forecasts. Ensemble forecasting is a numerical method that uses multiple predictions from slightly different initial conditions, or different forecast models, to generate a broad sample of the possible future states of a dynamical system (Knipp, 2016). The instances of different conditions or different models are called "members." In the forecast cycle each member starts with a current state of the system based on a combination of observations and a background model, followed by a calculation of the system evolution over time. Outputs from the members are then combined and analyzed for trends and uncertainty ranges. Recently, ensemble methods have been applied to a list of space research topics, ranging from predicting new solar cycles and coronal mass ejections to magnetospheric reactions, for which a brief review was given by Knipp (2016) and references therein.

The purpose of this paper is to report how PreMevE has been upgraded to make predictions of ultra-relativistic electron flux distributions across the outer radiation belt. Still, with no requirement for in situ measurements of trapped MeV electrons except for at GEO, this unique model, named PreMevE-2E where *E* stands for both ensemble forecast and enhancement, has enhanced its capability to meet the predictive requirements for penetrating outer-belt electrons during the post Van Allen Probes era. In the next section, data and parameters to be used for this study are briefly described, as well as the selected ML algorithms. Section 3 explains in detail how the model is trained, validated and tested to forecast the distributions of >2 MeV electron integral fluxes, followed by the forecasts

of 2 MeV electron differential fluxes in Section 4. This work is concluded in Section 5 with a summary of findings and possible future directions.

Data, Parameters, and Machine Learning Algorithms

Most data, model parameters and ML algorithms used in this work have been previously described in detail by Chen et al. (2019) and P2020, and here is a brief recap focusing on the differences. Ultra-relativistic electron flux distributions are from the in-situ observations made by the Relativistic Electron-proton Telescope (REPT, Baker et al., 2012) experiment aboard one Van Allen Probe spacecraft (RBSP-a) at $L \leq 6$, and by the Energy Spectrometer for Particles (ESP, Meier et al., 1996) instrument carried by one Los Alamos National Laboratory (LANL) GEO satellite LANL-01A at L = 6.6. The ESP instrument measures relativistic electrons between 0.7 and 10 MeV, and its flux data previously has been calibrated with other particle instruments (e.g., Friedel et al., 2005) and used for scientific studies of GEO electron dynamics (e.g., Sicard-Piet et al., 2008 and Boynton et al., 2014). As presented in Figure 4-1A, integral fluxes of >2 MeV electrons are the target data set that is a function of L-shell over a 1289-day interval starting from 2013 February 20. These >2 MeV electron data are used for model training, validation and test, and are not needed as model input (except for at GEO) for making predictions.

Model input parameters include low-energy precipitating electrons measured by one NOAA Polar Operational Environmental Satellite (POES) NOAA-15 and upstream solar wind conditions over the same time interval. As shown in Figure 4-1, POES electron data used here are the same as in P2020, and thus the same nomenclature is adopted: E2, E3, and P6 refer to the count rates of >100, >300, and >1000 keV electrons from different POES channels, as shown in Panels B – D, respectively. Hereinafter, all electron intensities for the target, E2, E3, and P6 data are in logarithmic values unless otherwise specified. Upstream solar wind conditions include the speeds that have been tested in

P2020, and solar wind densities (SWD) as the new model input parameter. All electron intensities as well as solar wind parameters in Figure 4. 1 are binned in 5-hour increments, and electrons are also binned every 0.1 L-shell. We standardized the solar wind speeds and densities by first subtracting their mean values and then dividing the results with the standard deviations.

The same four supervised ML algorithms tested in P2020 are used due to their previous success, including linear regression, feedforward neural networks (FNNs), long short-term memory (LSTM), and convolutional neural networks (CNNs). Briefly, linear regression models seek the optimized linear relationship between input parameters and targets. FNNs use layers of neurons to process inputs with linear transformations followed by nonlinear activation functions to optimize outputs. LSTM networks consist of connected memory cells that learn the sequential and temporal dynamics from the previous time steps to make predictions, and CNNs rely upon a convolution kernel to filter the data and explore the local patterns inside. All FNNs, LSTM and CNNs are trained with the objective to digest inputs and minimize a specified loss function. More details of these four ML algorithms can be found in Section 3 of P2020 and references therein.

For model development, data in Figure 4-1 are split for training, validation, and test, with portions of 65% (~835 days), 14% (175), and 21% (267), respectively. Models are trained for each individual L-shell between 2.8 and 6 as well as at GEO (6.6) in the outer belt region, with the optimization goal of minimizing the root-mean-square error between the target values $\boldsymbol{y}$ (electron fluxes in logarithm) and predicted values $\boldsymbol{f}$. Different parameter combinations and temporal window sizes are tested for model inputs. We also compare the performance of models using different ML algorithms as in P2020. Model performance is gauged by Performance Efficiency (PE), which quantifies the accuracy of predictions by comparing to the variance of the target. Naming $\boldsymbol{y}$ and $\boldsymbol{f}$ both with size $\boldsymbol{M}$, PE is defined

as $PE = 1 - \frac{\sum_{j=1}^{M}(y_j - f_j)^2}{\sum_{j=1}^{M}(y_j - \bar{y})^2}$, where $\bar{y}$ is the mean of $\boldsymbol{y}$. PE does not have a lower bound, and its perfect score is 1.0, meaning all predicted value perfectly match observed data, or that $\boldsymbol{f} = \boldsymbol{y}$.

Forecasting >2 MeV Electron Flux Distributions

Models in this section predict the integral fluxes of >2 MeV electrons. Key results are summarized in Tables 1 and 2, followed by detailed discussions. Table 1 lists PE values of all 32 models for 25 hr (or called 1-day) forecasts, and Table 2 are for 50 hr (2-day) forecasts. In each Table, there are eight input and window size combinations for each of the four ML algorithms, where the model names follow the convention of P2020. For example, FNN-64-32-elu are FNNs composed of two hidden layers—the first one has 64 neurons and the second has 32 neurons, and the neurons use Exponential Linear Unit (ELU, Clevert et al., 2015) as the activation function; LSTM-128 models have one layer with 128 memory cells; and conv-64-32-relu are CNN models composed of two convolutional layers—the first one contains 64 kernels and the second contains 32 kernels, and the kernels use Rectified Linear Unit (ReLU, Hahnloser et al., 2000; Nair & Hinton, 2010) as an activation function. All four categories of models in Tables 1 and 2 are the same as those in Tables 2 and 3 of P2020 and also have the same model hyperparameters. "Window size" refers to how many 5 - hr time bins of input data are needed by the models, e.g., a window size of four corresponds to a time history of 20 hours. The Input Parameters column specifies the features needed for each model, dE2 refers to the normalized temporal derivatives of E2 fluxes, E246 indicates E2 fluxes at L = 4.6 are used as inputs for all L-shells, and SW and SWD refer to solar wind speeds and densities, respectively. Here E246 is used for other L-shells due to the high cross-Lshell correlation observed previously (e.g., Figure 4. 2C in Chen et al., 2019) as well as the demonstrated positive effects on forecasts (e.g., see models with highest PE values in Table 2 of P2020). Because of different selection of input features, the size of the

input layer changes accordingly. For example, the size of the input layer for model 10 in Table 1 uses a window size of 16 and data from E2, E3, P6, and SW channels. This corresponds to an input size of 64 – four data channels multiplied by 16 data points for each channel. The last row in each Table is for the ensemble model that will be discussed later in this section.

First, we examined the effects of feature selection (i.e., input combinations and window sizes) as in Figure 4-2, which uses linear and LSTM models as the examples and plots the out-of-sample PE values (for validation and test data) as a function of L-shells. In Panel A, the general trend can be observed for linear models that PE increases with the increasing number of input parameters and window sizes. All curves have similar shapes with the highest PE at L~4.0 and decreasing in both directions, while PE values at GEO go above 0.6. Note each PE curve has data points located at L-shells from 3.0 to 6.0 with an increment of 0.1 as well as at GEO, per the way PreMevE was constructed as specified in Chen et al., 2019. The high PE values at GEO can be explained by the inclusion of >2 MeV electron fluxes in-situ measured by LANL-01A satellite, as previously seen in P2020. Here we highlight three examples: models 6 and 8 have different input parameters but the same window size, while models 7 and 8 have the same input parameters but different window sizes (see Table 1). It is seen that model 8 has the highest PE with SWD included in inputs. In Panel B, LSTM models have very different PE curves with large variations. Several LSTM models show a local minimum in PE with L-shell at ~4 and a plateau at L between 3.1-3.8. In addition, the inclusion of SWD to models 23 and 24 indeed decrease their PE at $L \leq 6$ compared to those of model 22 (also see Table 1). PE values can drop below zero at small $L < 3.0$, particularly for the linear models, mainly due to the lack of training events, which is discussed later in this section. Therefore, hereinafter we confine our discussions on PE only for $L \geq 3.0$.

To get an idea of model performance, we first inspected Table 1 for 1-day forecasts comparing models' mean PE values, which are averaged over all L-shells except for GEO for individual models. Based on the mean out-of-sample PE values for combined validation and test data, one can rank models' performance from high to low. For instance, in the linear category, model 8 is the top performer with the highest mean PE of 0.523, followed by model 6 with a PE value of 0.509. For the top performer model 8, its out-of-sample PE at GEO is 0.629, also the highest in the category and thus in bold and underscored. Similarly, the top and second performers in other categories are picked out with their mean PE in bold font and underscored. In Table 1, mean PE values of the four top (second) performers are 0.523 (0.509) for linear, 0.553 (0.488) for FNN, 0.537 (0.521) for LSTM, and 0.479 (0.477) for CNN, while their PE values at GEO are 0.629 (0.625), 0.630 (0.603), 0.600 (0.581), and 0.598 (0.566), which are not necessarily the highest of each category. Note among the four top performers, only the linear model 8 has SWD in model inputs, while at GEO three out of the four models with the highest PE, i.e., models 8, 15 and 23, have SWD included.

Similarly, in Table 2 for 2-day forecasts, mean out-of-sample PE values for the four top (second) performers are 0.438 (0.431), 0.460 (0.416), 0.456 (0.451), and 0.423 (0.408), while their PE at GEO are 0.428 (0.431), 0.423 (0.419), 0.390 (0.384), and 0.402 (0.345) which are often not the highest in the category. For 2-day forecasts, SWD are not needed for the top four performers, while at GEO the only exception is the linear model 8. Therefore, unlike the important role of SW as demonstrated here and in P2020, SWD is not necessary for model input except for 1-day linear forecasts at GEO. Also, in both Tables 1 and 2, top FNN and LSTM models marginally outperform top linear models, suggesting the significance of non-linear component for >2 MeV electrons, in sharp contrast to P2020 models for 1 MeV electrons in which top linear ones always have the highest (or next to the highest)

PE values. Additionally, PE values at GEO are ~0.1 higher than the mean PE at L≤6 for 1-day forecasts, while for 2-day forecasts PE values are slightly lower at GEO.

PE curves for the top two performers in each category for one- and two-day forecasts are further compared in Figure 4-3 as a function of L-shell. First, note in both panels there is no one individual model that outperforms others over all L-shells. For example, linear models (i.e., the solid gray curves) have higher PE at L-shells above ~3.8, while the top FNN (red) and LSTM (brown) models perform better at small L-shells than the quickly degrading linear ones. Plus, the PE curves for the top linear model 6 of PreMevE 2.0 for 1 MeV electrons (see Tables 2 and 3 in P2020) are plotted in dashed gray for comparison. It can be seen that for this new model PE curves for the linear ones in solid gray and magenta have higher PE at L-shells >4.5 for 1-day (>4.0 for 2-day) but lower PE at smaller L-shells than the dashed gray curve.

An overview of 1-day forecasted flux distributions from the four top models are presented in Figure 4-4, showing similar dynamics compared to those observed in target data. Over the entire interval, most MeV electron events are captured well in terms of both intensities and L-shell ranges, e.g., the areas in red and yellow colors. Exceptions include the significant electron dropouts, e.g., the vertical blue strip on days ~ 1080 at L > 5, and the deep electron injections into small L-shells below 3.0. To highlight the differences between forecasts and target, error ratios for the four models are plotted in Figure 4-5, in which green color indicates perfect predictions while blue (red) means over- (under-) predictions. For example, in the validation and test periods, the lack of vertical red strips suggests the onsets of >2 MeV electron events are well predicted, while the vertical blue strips reflect the predicted high fluxes during dropouts, which is acceptable since this model aims to predict the enhancements of energetic electrons. Again, the reddish areas at small L-shells ~ 2.8 and 2.9 during the validation and test periods, particularly in Panel A, indicate models under par performance in the

area. This is due to the fact that at these low L-shells training data is dominated by background and the ML algorithms can learn only from the single major event starting on day ~758, while there are up to three events during the validation and test periods. To keep the comparison standardized, we chose not to modify the training set for L-shells 2.8 and 2.9, and hence predictions at these L-shells are not as good as others. For the same reason, we excluded these two L-shells and only counted 3.0 and higher to calculate the mean PE values over L-shells as in Tables 1 and 2.

Alternatively, models' performance on 1-day forecasts can also be examined from scatter plots of flux data points over the entire 1289-day interval, as shown in Figure 4-6 for the top four models. In each 2D histogram, the position of each pixel compares the predicted and target fluxes and the pixel color counts the occurrences over the interval. In each panel, the diagonal indicates a perfect match, and the dark gray (light gray) dashed lines on both sides mark error factor ratios of 3 (5) and 1/3 (1/5) between predicted and observed fluxes (original flux values not in logarithm). The majority of the points (in red) fall close to the diagonal and are well contained between the two factor-3 lines, particularly the points in the upper right quarter during MeV electron events. The two percentages in the lower right show how many data points fall with the two pairs of factor lines, and the red number in the second row is the correlation coefficient (CC) value. It is seen that all models have high CC values from 0.90 to 0.91, and 71 - 78% (87-90%) forecasts have error ratios within the factors of 3 (5). In addition to PE, all these CC values and percentages help further quantify performance of the top four models.

The overview plot for 2-day forecasts is given by Figure 4-7 for the four top performers as identified in Table 2. Similar features can be seen here as in Figure 4-4, including the resemblance between forecasts and observations as well as the misses at low L-shells of 2.8 and 2.9. One noticeable feature in Panel D is the "patchiness" at L=4.2, where predicted fluxes by LSTM model 22 are

persistently lower than those in neighboring L-shells. This feature corresponds to the local minimum at L=4.2 in the solid brown PE curve for the same model in Figure 4-3B. A similar feature is also visible in Figure 4-7 E at the same L-shell and corresponds to the local minimum in the solid orange PE curve in Figure 4-3B. Considering the L-shell dependent performance of models as shown in Figure 4-3, we decided to test ensemble forecasts for optimization, using a combination of linear and non-linear models. Indeed, as mentioned in Section 1, the ensemble forecast has been widely applied for weather forecasting (e.g., see the review by Cheung et al., 2001), and studies have shown that the ensemble mean can act as a non-linear filter with a skill statistically higher than any ensemble individual member (Toth and Kalnay, 1997).

We first tested 1-day forecasts using a small ensemble group. As shown in the last row of Table 1, ensemble members include linear model 8, FNN model 13, LSTM model 22 and CNN model 29, which are the top four performing models in each of the four categories. At each time step the ensemble prediction of electron fluxes at one L-shell is the median of all four-member model outputs, and standard deviation of the outputs is the measure of uncertainty. One such example is shown in Figure 4-8 , where at four individual L-shells the ensemble forecasts (in red) closely track the increments and decays of >2 MeV electron fluxes (black) observed during MeV electron events, and the gray strips in the background represent the uncertainties from this ensemble group. Similarly, another ensemble group was constructed for 2-day forecasts, with the same member models except for the linear one being replaced by model 8 (see the last row in Table 2), and the ensemble forecasts are also shown to closely trace observations at four individual L-shells as in Figure 4. 9.

The plots in Figure 4-10 compare the observed and ensemble forecasted flux distributions over the entire interval. There are noticeable improvements, including the better predictions of low fluxes at L-shells ~3.5, e.g., the blue area centered on day 552 during the training in Panel B, and the deep

injections to low L-shells during the validation and test periods when compared to the linear model in Figure 4. 4B. Also, the "patchiness" previously observed in 2-day forecasts in LSTM model has been much alleviated here in Panel D. In general, however, it is not easy to tell the difference just by eyeballing and comparing to distributions in Figure 4-4, 5 and 7.

Therefore, we again use the PE to quantify model performance, comparing the ensemble PE curves to those of group members as a function of L-shell in Figure 4-11. First, it is seen that in both panels, the ensemble PE curves (in red) almost always stay to the rightmost for all L-shells, including at GEO, when compared to PE curves from four member models. The outperformance of ensemble models is consistent with previous results from other fields and justifies the usage of ensemble forecasting in this new model. Second, when compared to the PE curves in dashed gray from the linear model of PreMevE 2.0, our ensemble forecasts either have at least comparable performance in Panel A for 1-day or have even better performance in Panel B for 2-day, in particular at medium or high L-shells. Looking back to Tables 1 and 2, the ensemble models have a mean PE value of 0.612 for 1-day and 0.521 for 2-day at $L \leq 6$, and 0.677 and 0.572 at GEO, and all these values are significantly higher than those from individual top performer models. Therefore, our test demonstrates the advantage of ensemble forecasting, and thus this new model is named PreMevE-2E for its adoption of ensemble forecasting to enhance prediction capability. To put this model's performance into context, the operational Relativistic Electron Forecast Model (REFM) at NOAA has PE values of 0.72 and 0.49 at GEO for 1‐ and 2‐day predictions for daily averaged fluence of >2 MeV electrons, and our model has PE values of ~0.68 and ~0.57 at GEO for 1- and 2-day forecasts of >2 MeV electron fluxes with 5 hr time resolution. In addition, besides GEO our model also has similar predictive performance across L-shells between 3 and 6 in the heart of the outer belt.

Note that here we only tested with a small ensemble group, and obviously there are other possible options. For example, the ensemble group may include more members, and not necessarily the same number of models from all categories. Besides, particularly at GEO the ensemble group can be different, for instance by selecting the models with the highest out-of-sample PE values at GEO. Furthermore, another possibility is to construct a "hybrid" model that uses the best performing model(s) at each individual L-shell, instead of selecting the same model(s) for all L-shells. For example, this hybrid model may combine nonlinear models at small L-shells (e.g., L<3.5 as in Figure 4-3) and include more linear ones at large L-shells depending on their ranks in PE. From this sense, here the ensemble test is an initial test of the hybrid model—only with a small model group though—and more extensive studies are expected in the future.

Forecasting 2 MeV Electron Flux Distributions

This section explains how PreMevE-2E predicts differential flux distributions of 2 MeV electrons. The methodology is identical to those used for >2 MeV electrons as described in Section 3, tested models are the same as in Tables 1 and 2, and here we summarize the results. First, the effects of model input parameters and window sizes are examined, and the mean PE values for individual models are presented in Tables 3 and 4 for 1- and 2-day forecasts, respectively. In Table 3, mean PE values of the four top (second) performers are 0.600 (0.590) for linear, 0.549 (0.548) for FNN, 0.549 (0.533) for LSTM, and 0.525 (0.518) for CNN, while their PE values at GEO are 0.566 (0.568), 0.461 (0.535), 0.509 (0.539), and 0.459 (0.437), which are lower than the highest for each category. In Table 4, mean PE values of the four top (second) performers are 0.512 (0.506) for linear, 0.474 (0.461) for FNN, 0.438 (0.435) for LSTM, and 0.439 (0.425) for CNN, while their PE values at GEO are 0.234 (0.244), 0.186 (0.105), 0.138 (0.106), and 0.102 (0.125), which are often not even close to the highest value for each category. It is interesting to notice that the top (and second) linear models have higher mean

PE than all the remaining top performers for both 1- and 2-day forecasts. Based on the rank of mean PE values, in the last row of both Tables, ensemble forecasting models are constructed including the top performers from each of the four categories.

The overview plots in Figure 4-12 compare the observed and ensemble forecasted flux distributions over the entire interval. The similarity between 1-day ensemble forecasts (Panel B) and target distributions (Panel A) is impressive, but the vertical red strips in the error ratio distribution (Panel C) at L > ~4 also suggest the forecasts often miss the very beginning of the onsets of MeV electron events. Similar features are seen in Panels D and E for 2-day ensemble forecasts.

In Figure 4-13, model performance is quantified by comparing the ensemble PE curves to those of group members as a function of L-shell. First, as seen in Figure 4-11, in both panels the ensemble PE curves (in red) almost always stay to the rightmost for all Lshells, except for at GEO for 2-day forecasts, when compared to the PE curves from four member models. Therefore, the outperformance of ensemble models is confirmed for 2 MeV electrons. Second, when compared to the PE curves in dashed gray from the linear model of PreMevE 2.0, our ensemble forecasts have comparable (Panel B) or even higher (A) PE values in average but not at GEO. From the last rows of Tables 3 and 4, the ensemble models have a mean PE value of 0.624 for 1-day and 0.521 for 2-day at $L \leq 6$, and 0.564 and 0.186 at GEO, and all these mean PE values are higher than those from individual top performer models but not at GEO. To show more details, for 1-day forecasts, Figure 4-14 shows that at four individual L-shells the ensemble forecasts (in red) closely track the ups and downs of 2 MeV electron fluxes (black), and similarly Figure 4-15 shows the 2-day forecast results. It is noticeable that in Panel D the 2-day forecasts at GEO often have values much lower than those observed peak flux values, in particular during the several major events, which may explain the low PE value of 0.186 at GEO.

There are two motivations for us to test predicting differential flux distributions of 2 MeV electrons. The first is to have a counterpart in comparing with PreMevE 2.0 model for 1 MeV electrons, and it turns out the two models have very similar performance in term of PE (except for 2-day at GEO). The second motivation is that, with both integral and differential fluxes available, one can further determine a single-parameter energy spectrum shape (e.g., in an exponential form) that can be helpful to quantify radiation effects (e.g., ionizing doses) with a given satellite geometry and shielding design.

Summary and Conclusions

Using electron data from NASA's Van Allen Probes mission, we have trained, evaluated and tested a set of supervised machine learning models to forecast ≥2 MeV electron fluxes. After evaluating the performance of these models, ensemble forecasting has proven overall to perform better than any individual model in different categories. After the completeness of training, our model has demonstrated its ability to make forecasts with no more need of in-situ electron measurements from Van Allen Probes.

This new PreMevE-2E model can be used for predicting dynamic distributions of ultra-relativistic electrons, with measurement inputs that are made available from satellites operating in LEO, GEO, and at the Lagrangian 1 point of the Sun-Earth system. In this work we have evaluated and tested: 1) the effects of different parameter combinations, including solar wind densities, as well as the window sizes for model performance; 2) four categories of linear and neural network models; and 3) the adoption of ensemble forecasting. PreMevE-2E has enhanced its forecasting capability by extending to the ultra-relativistic electron energy range. Model predictions over a 14 months out-of-sample period demonstrate that this model provides high-fidelity 1-day (2-day) forecasts of ≥2 MeV electron flux distributions: the mean PE values are above 0.61 (0.52) for both integral and differential fluxes across L-shells from 3 to 6; at GEO, model PE values are ~0.68 and

~0.57 for 1- and 2-day forecasts of >2 MeV integral fluxes, and ~0.56 and ~0.19 for the differential fluxes of 2 MeV electrons. Therefore, we believe this newly updated PreMevE-2E model lays down another step stone towards fully predicting severe MeV electron events in the future.

Acknowledgements

The authors declare no conflicts of interest. We gratefully acknowledge the support of NASA Heliophysics Space Weather Operations to Research Program (18-HSWO2R18-0006), the NASA Heliophysics Guest Investigators program (14-GIVABR14_2-0028), and LANL internal funding. We want to acknowledge the PIs and instrument teams of NOAA POES SEM2 and RBSP REPT for providing measurements and allowing us to use their data. Thanks to CDAWeb for providing OMNI data. RBSP and POES data used in this work were downloadable from the missions' public data websites (https://www.rbsp-ect.lanl.gov and http://www.ngdc.noaa.gov), while LANL-01A electron data are provided as supplementary material of this work. We would also like to thank Misa Cowee for her great help with proofreading and copyediting the manuscript.

Figures for chapter 4

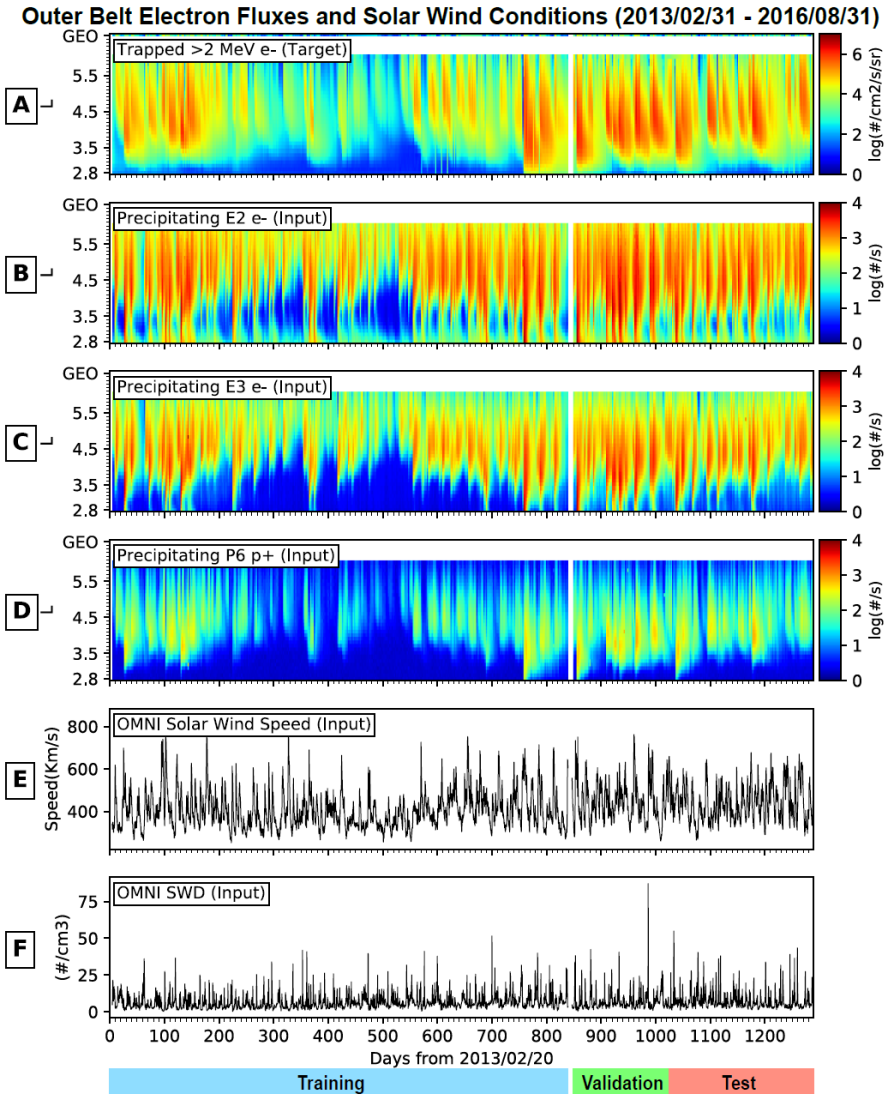


Figure 4-1 **Overview of electron observations and solar wind conditions.** All panels present the same 1289-day interval starting from 2013/02/20. **A)** Flux distributions of >2 MeV electrons, the variable to be forecasted (i.e., targets). **B to D**) Count rates of precipitating electrons measured by NOAA-15 in LEO, for E2, E3, and P6 channels, respectively. **E)** Solar wind speeds measured upstream of the magnetosphere from the OMNI data set. **F)** Solar wind densities. Data in Panels **B** to **F** serve as model inputs, i.e., predictors. The bottom color bars indicate the portions of data used for training, validation, and test.

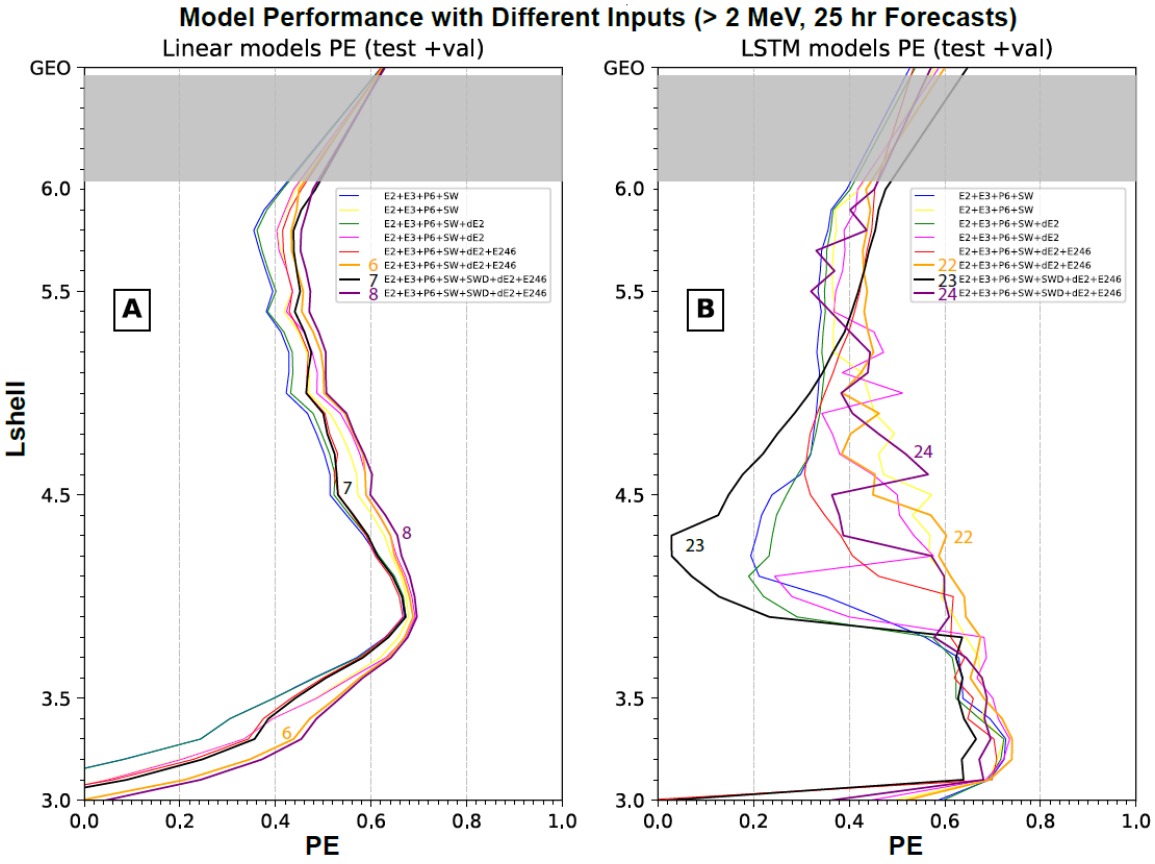


Figure 4-2 **PE values for the combined validation and test sets are presented as a function of L-shell for linear and LSTM models as in Table 1. A)** Comparison of eight linear regression models with the last three being numbered. **B)** Comparison of eight LSTM models also with the last three being numbered. The models are numbered in the way as in Table 1. Note all linear models behave similarly, but LSTM models vary greatly with different input parameters and window sizes. Also note there is no data points on each PE curve inside the shaded L-shell range (i.e., 6.0 < L < GEO).

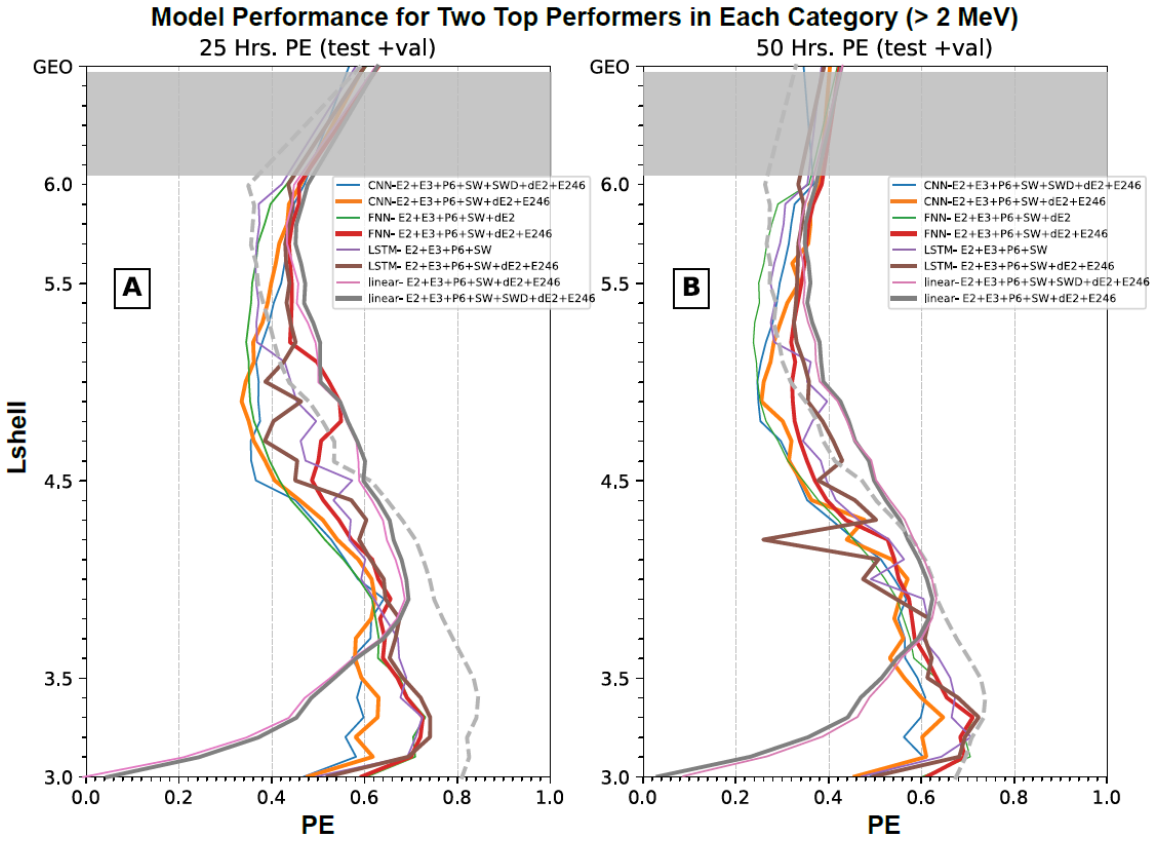


Figure 4-3 **Model PE values for validation and test data are presented as a function of L-shell for the top two performers in each category forecasting > 2 MeV electrons. A)** Top two performers of each category for 1-day (25 hr) forecasts as listed in Table 1. In each category, the thick (thin) curve is for the top (second) performer. PE curve for the top linear model in PreMevE 2.0 making 1-day forecasts of 1 MeV electrons (P2020) is plotted in dashed gray for comparison. **B)** Top two performers of each category for 2-day (50 hr) forecasts as listed in Table 2. PE curves are in the same format as in Panel A. PE curve for the top linear model in PreMevE 2.0 making 2-day forecasts of 1 MeV electrons (P2020) is also plotted in dashed gray for comparison.

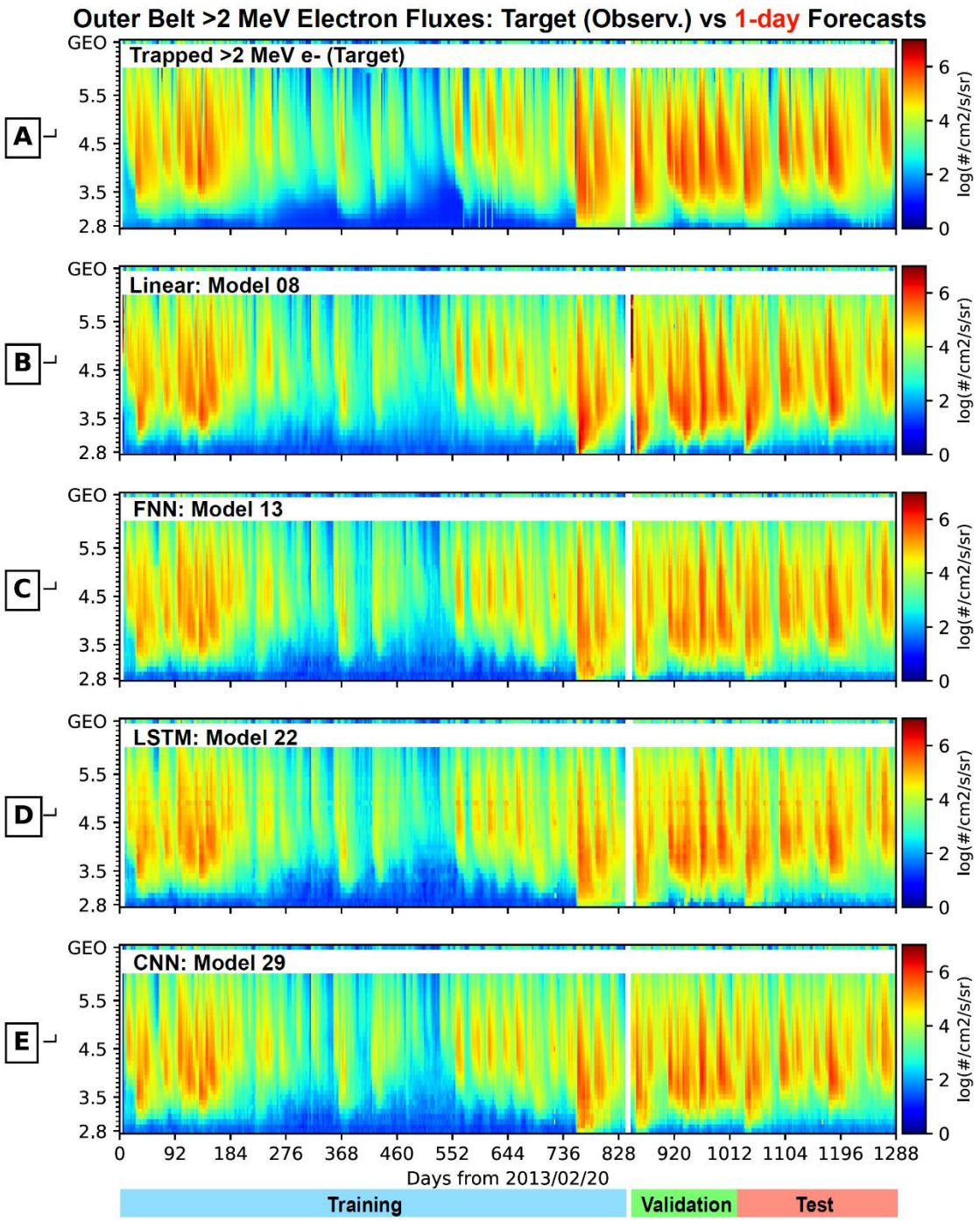


Figure 4-4 **Overview of target and 1-day forecasted > 2 MeV electron fluxes across all L-shells for the entire 1289-day interval. A**) Observed flux distributions to be forecasted for >2 MeV electrons. Panels **B**) to **E**) show 1-day forecasted flux distributions by the four top performers, each with the highest out-of-sample PE from one category, including the linear regression model 8, FNN model 13, LSTM model 22, and CNN model 29 as listed in Table 1.

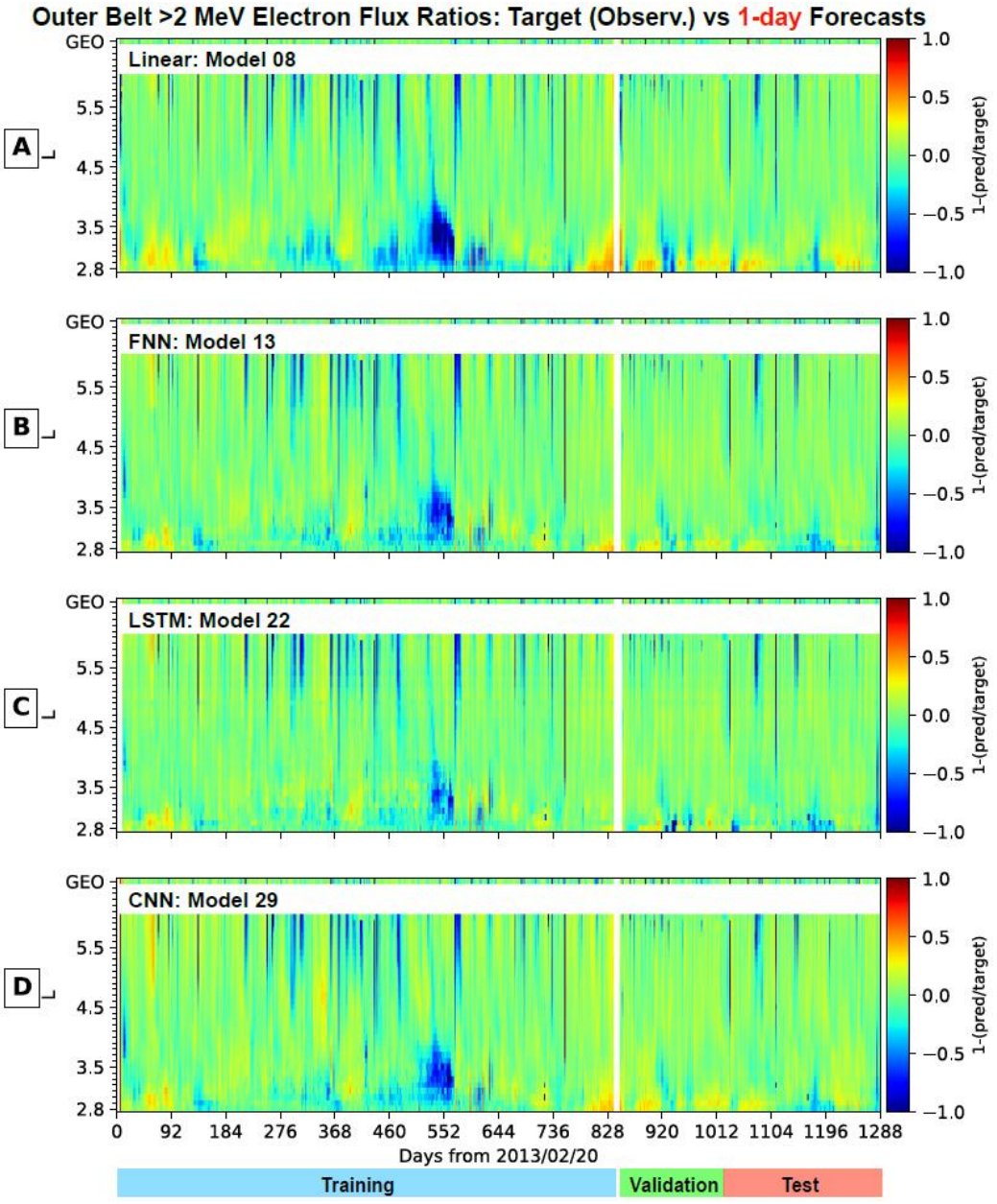


Figure 4-5 **Relative error ratios of 1-day forecasts across all L-shells for >2 MeV electrons.** Panels A to D plot the deviations ratios, defined as targets minus forecasts and then divided by the targets, as a function of L-shell and time for linear regression model 8, FNN model 13, LSTM model 22, and CNN model 29, respectively, the four top performers as listed in Table 1. Green color depicts perfect predictions, and red (blue) indicates under-predictions (over-predictions).

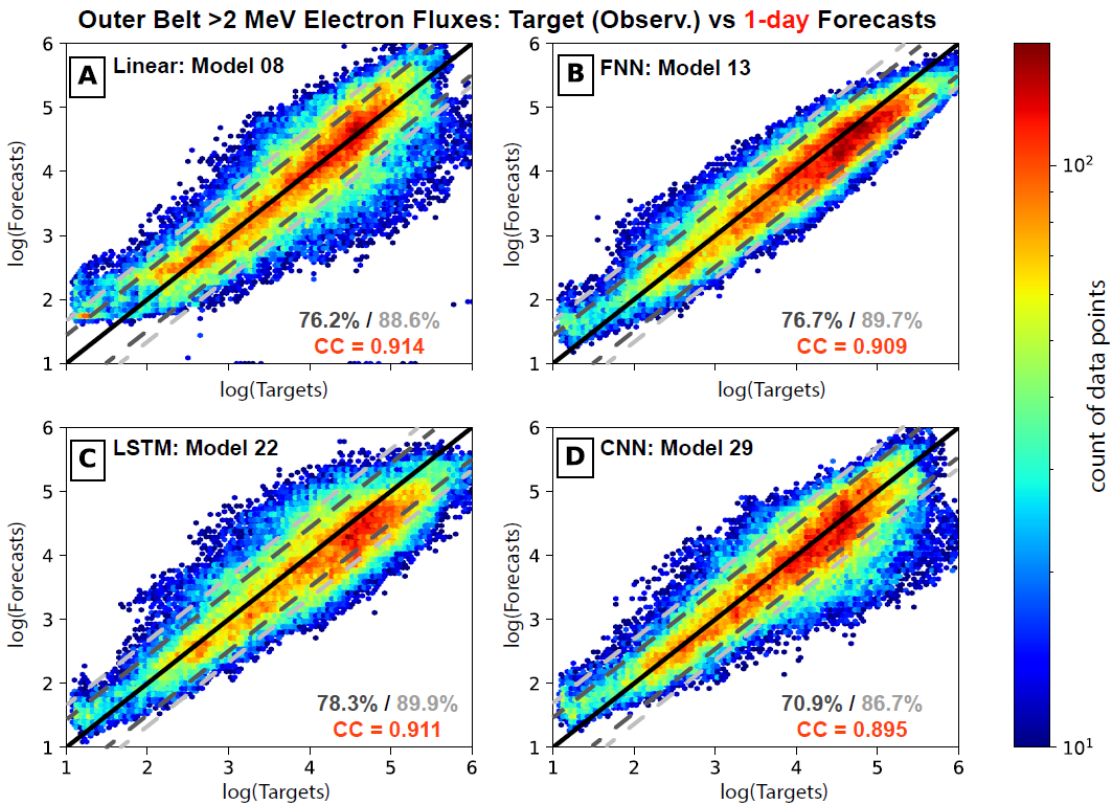


Figure 4-6 **Model prediction vs. target 2D histograms for 1-day forecasted >2 MeV electron fluxes across all L-shells. A**) Histogram of the fluxes predicted by LinearReg model 8 (the linear top performer as in Table 1) vs. the target >2 MeV electron fluxes. The color bar indicates the count of points in bins of size 0.1 x 0.1. Similarly, panels **B**) to (**D**) show predictions vs >2 MeV target for FNN model 13, LSTM model 22, and CNN model 29, the top performers as in Table 1. In each panel, the diagonal line for perfect matching is shown in solid black curve, and the dashed dark gray (and light gray) lines indicate ratio—between original fluxes—factors of 3 (and 5). The dark gray (light gray) number in lower right is the percentage of points falling within the factors of 3 (5), and the red number shows the correlation coefficient.

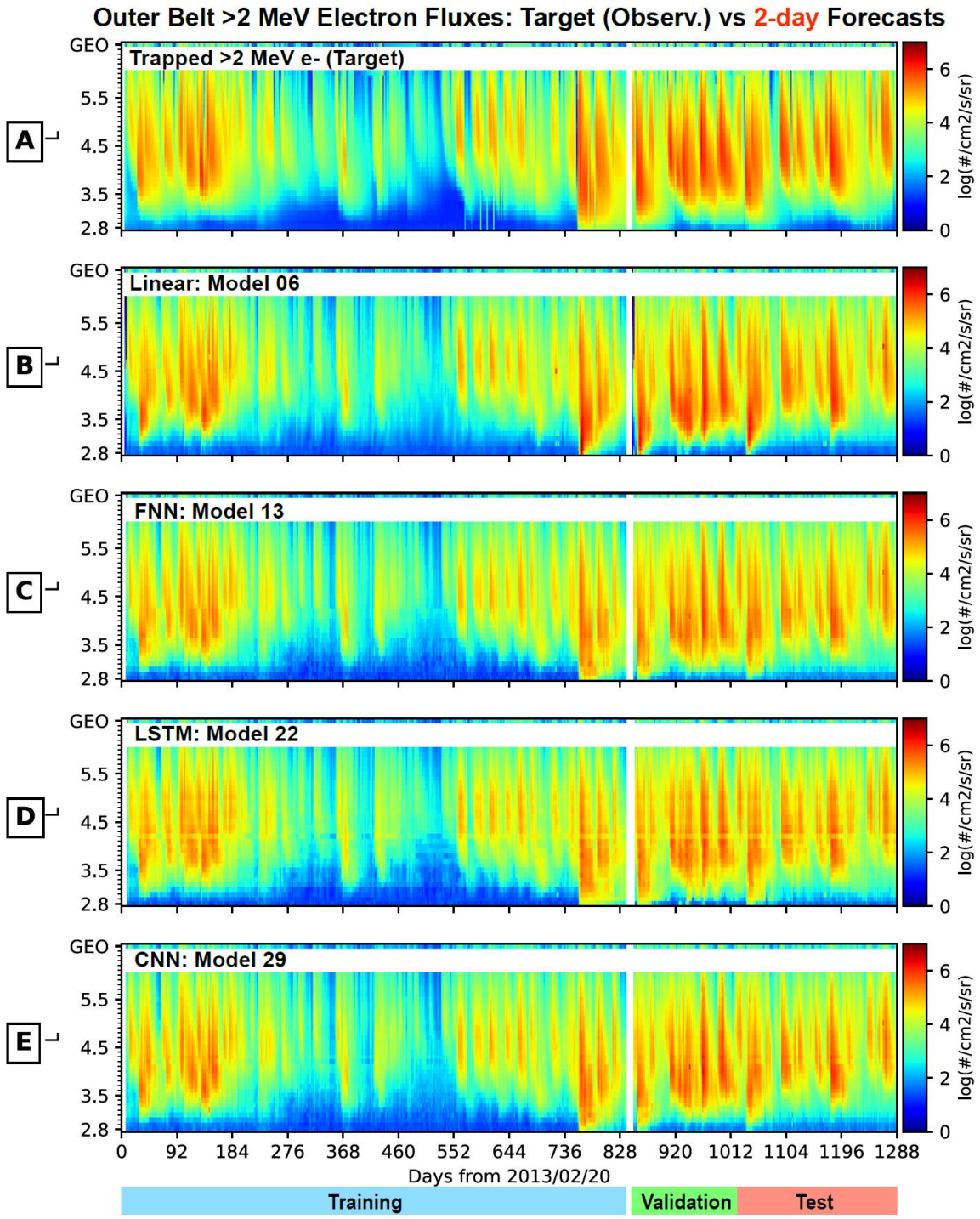


Figure 4-7 **Overview of target and 2-day forecasted fluxes across all L-shells.** Panel (A) shows the observed flux distributions to be forecasted for >2 MeV electrons. Panels (B) to (E) show forecasts from the four top performers, each with the highest out-of-sample PE from one category, including linear regression model 6, FNN model 13, LSTM model 22, and CNN model 29 as listed in Table 2.

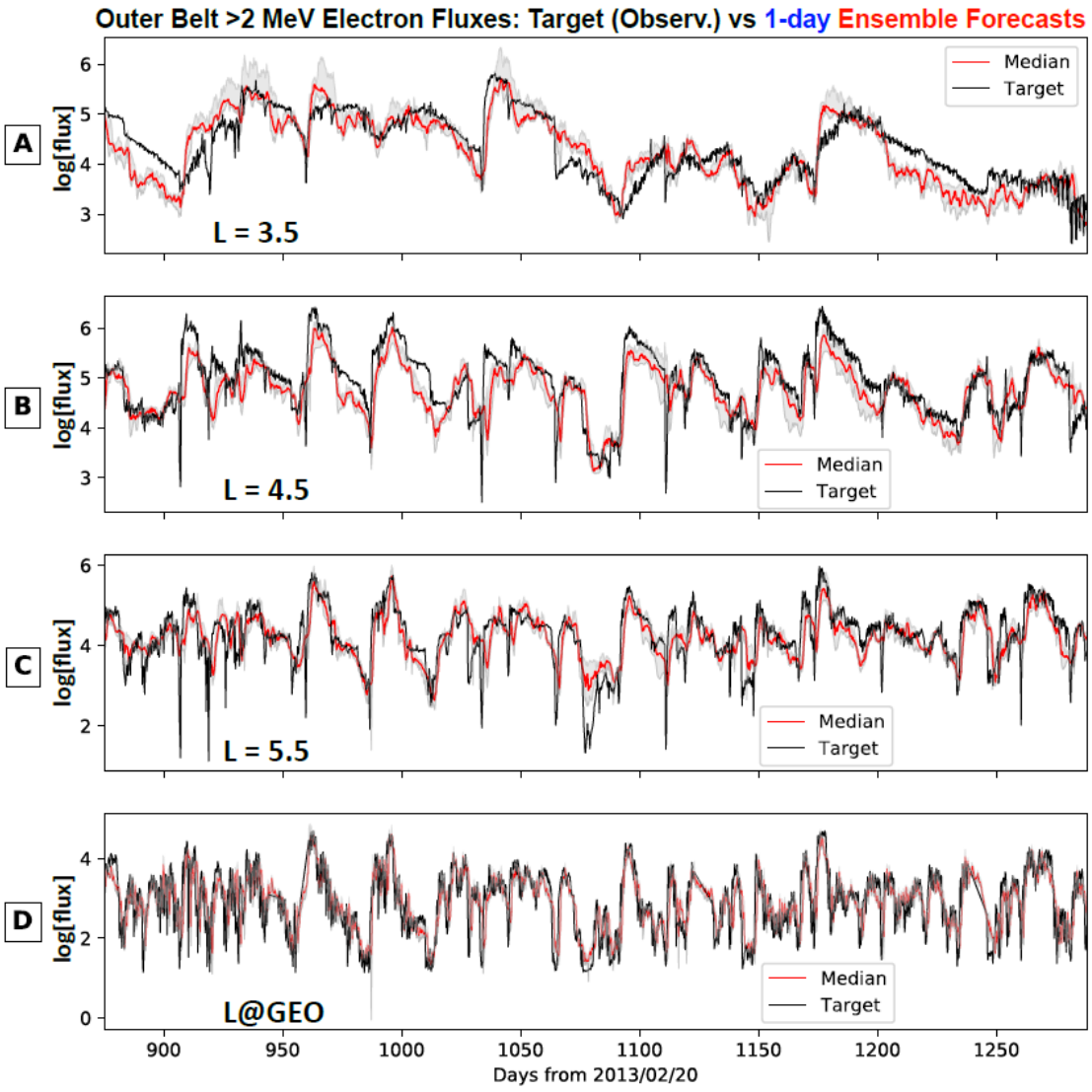


Figure 4-8 **One-day ensemble forecasting results for > 2 MeV electron fluxes over individual L-shells.** Results are shown for the validation and test periods, and panels from the top to bottom are for L-shells at 3.5, 4.5, 5.5, and GEO (6.6), respectively. In each panel, the target is shown in black, and the gray strip shows the uncertainty ranges (or standard deviations) from the ensemble group, and the median from the ensemble predictions is shown in bright red color. Note that the uncertainties from the ensemble models vary both spatially and temporally, however the median values follow the targets closely.

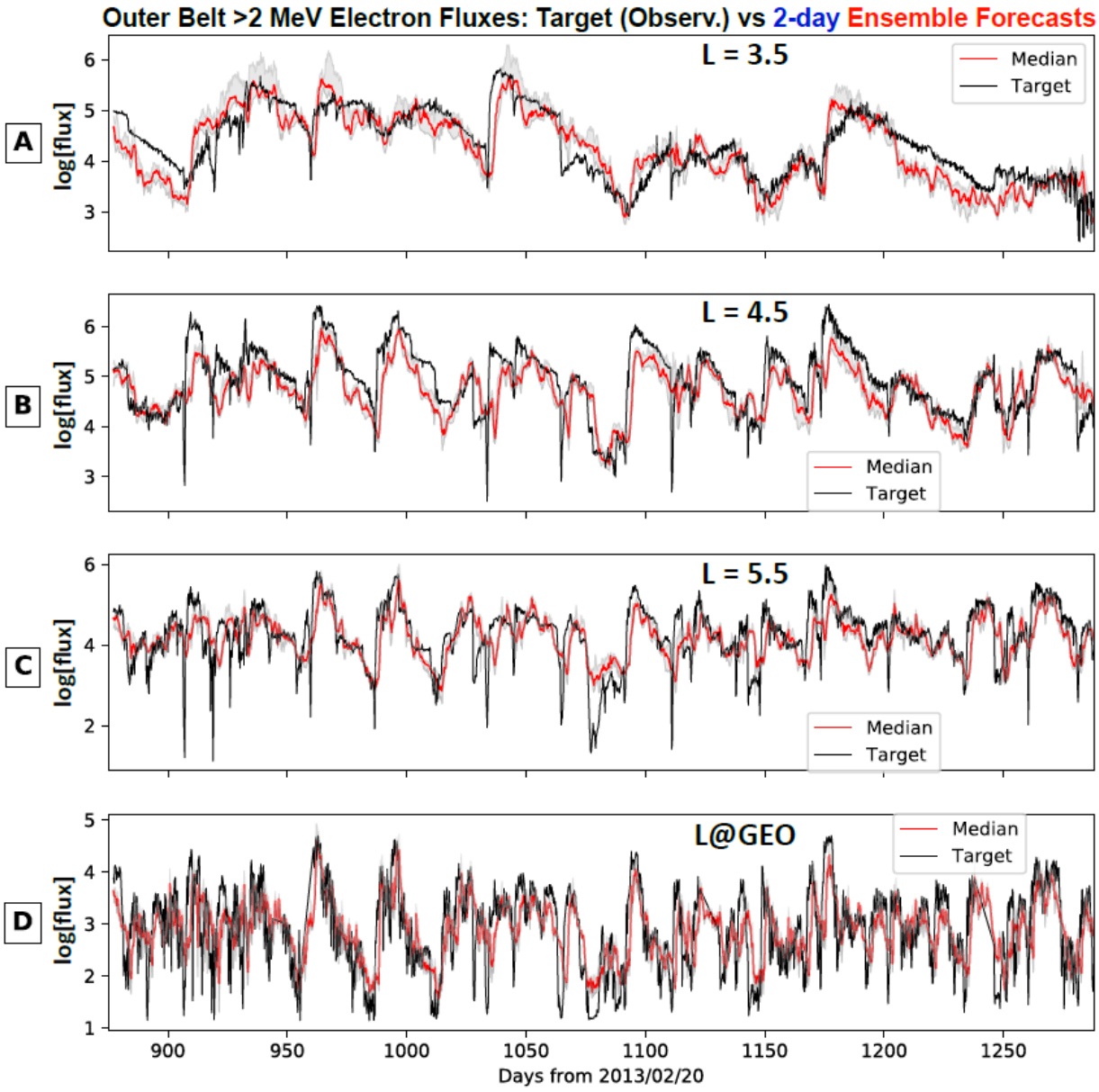


Figure 4-9 **Two-day ensemble forecasting results for >2 MeV electron fluxes over individual L-shells.** Results are shown for the validation and test periods, and in the same format as Figure 4.8.

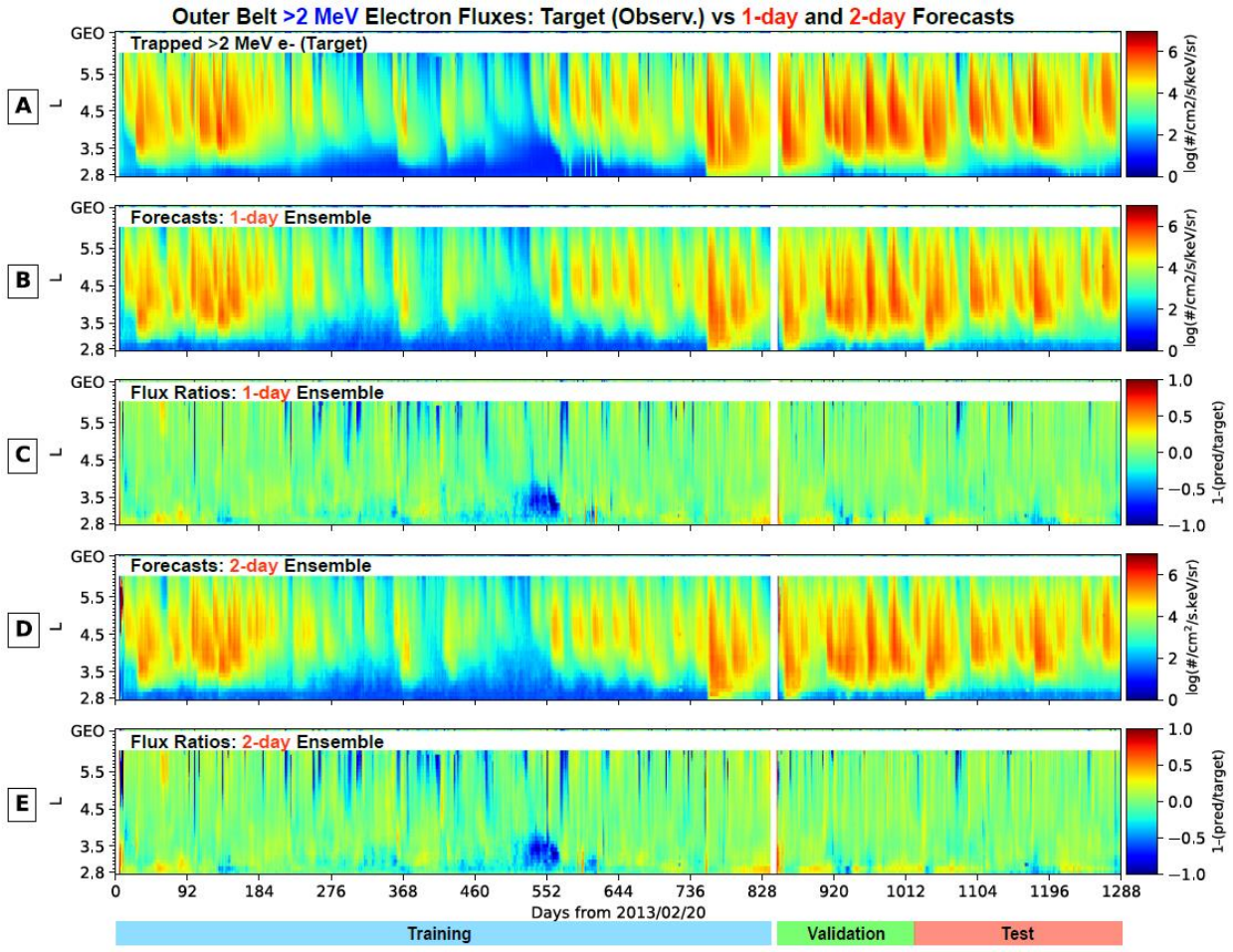


Figure 4-10 **Overview of target vs 1- and 2-day ensemble forecasted >2 MeV electron fluxes across all L-shells for the entire 1289-day interval. A)** Observed flux distributions. **B)** One-day predicted flux distributions from the ensemble model. **C)** Deviation ratios between the target and 1-day predicted fluxes. **D** and **E)** Same format as B and C but for 2-day ensemble forecasts.

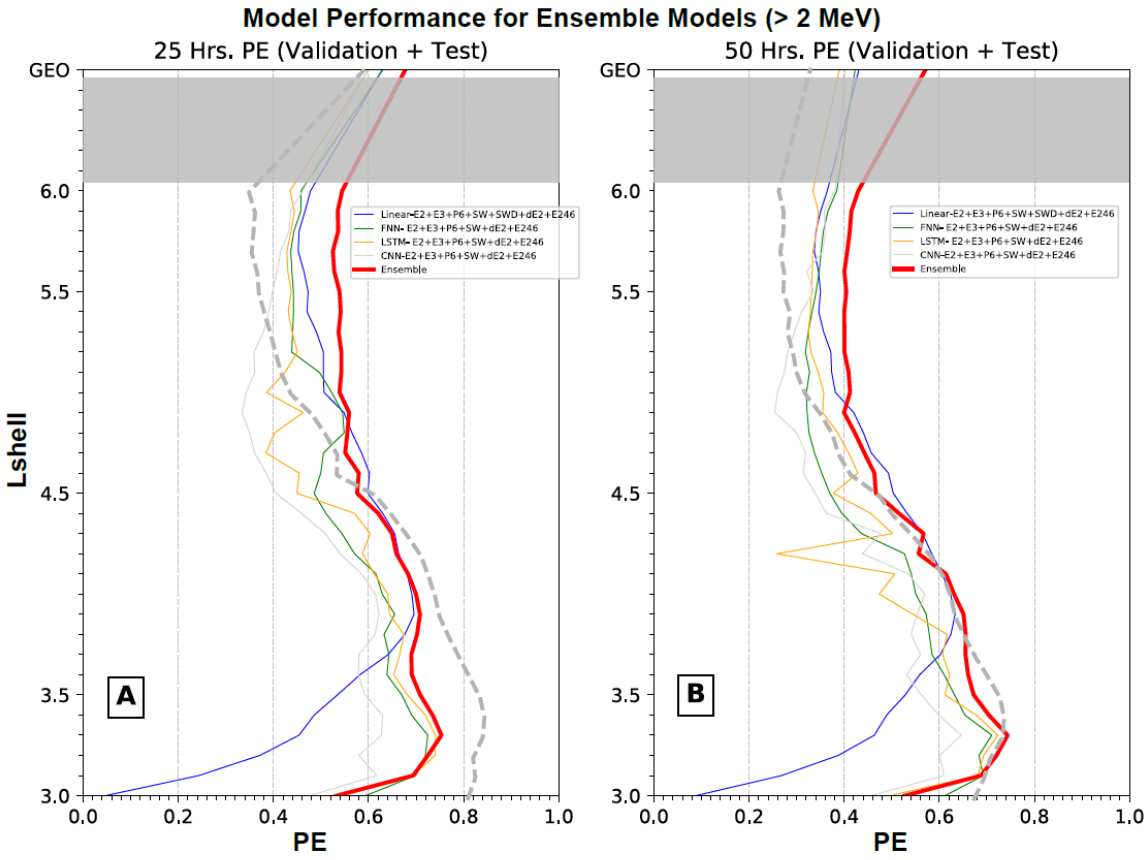


Figure 4-11 **Model PE values for validation and test data are presented as a function of L-shell for ensemble models forecasting >2 MeV electrons. A)** PE curves for 1-day (25 hr) forecasting models. The thick red curve is for the ensemble model compared to four individual ensemble member models (the top performers as defined in Table 1) in different colors. PE curve for the top linear model in PreMevE 2.0 making 1-day forecasts of 1 MeV electrons (P2020) is plotted in dashed gray for comparison. **B)** PE curves for 2-day (50 hr) forecasting models. The red curve is for the ensemble model and other four curves are for ensemble member models (as defined in Table 2). The PE curve for the top linear model in PreMevE 2.0 making 2-day forecasts of 1 MeV electrons (P2020) is plotted in dashed gray for comparison.

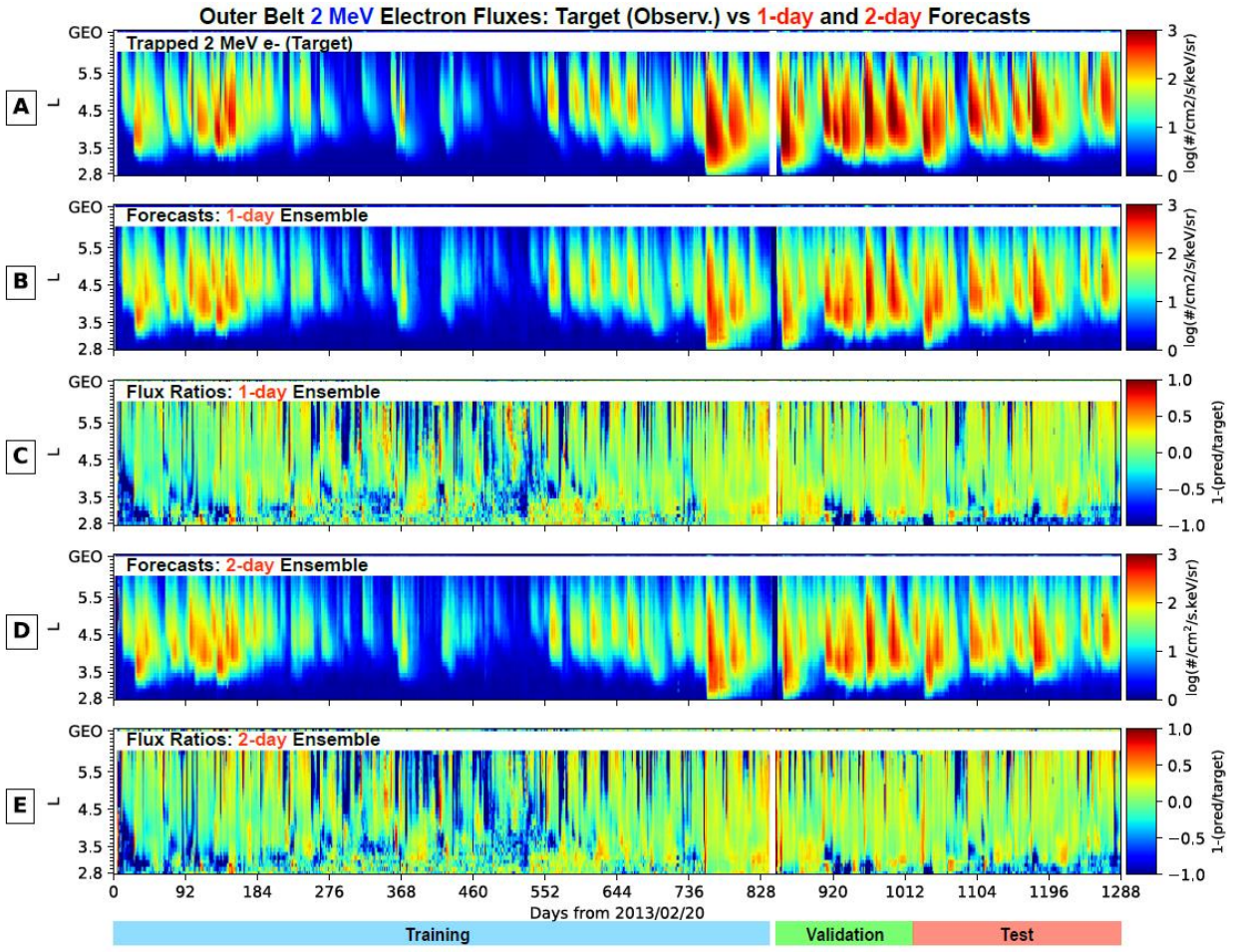


Figure 4-12 **Overview of target vs 1- and 2-day ensemble forecasted 2 MeV electron fluxes across all L-shells for the entire 1289-day interval.** All panels are in the same format as in Figure 4. 10.

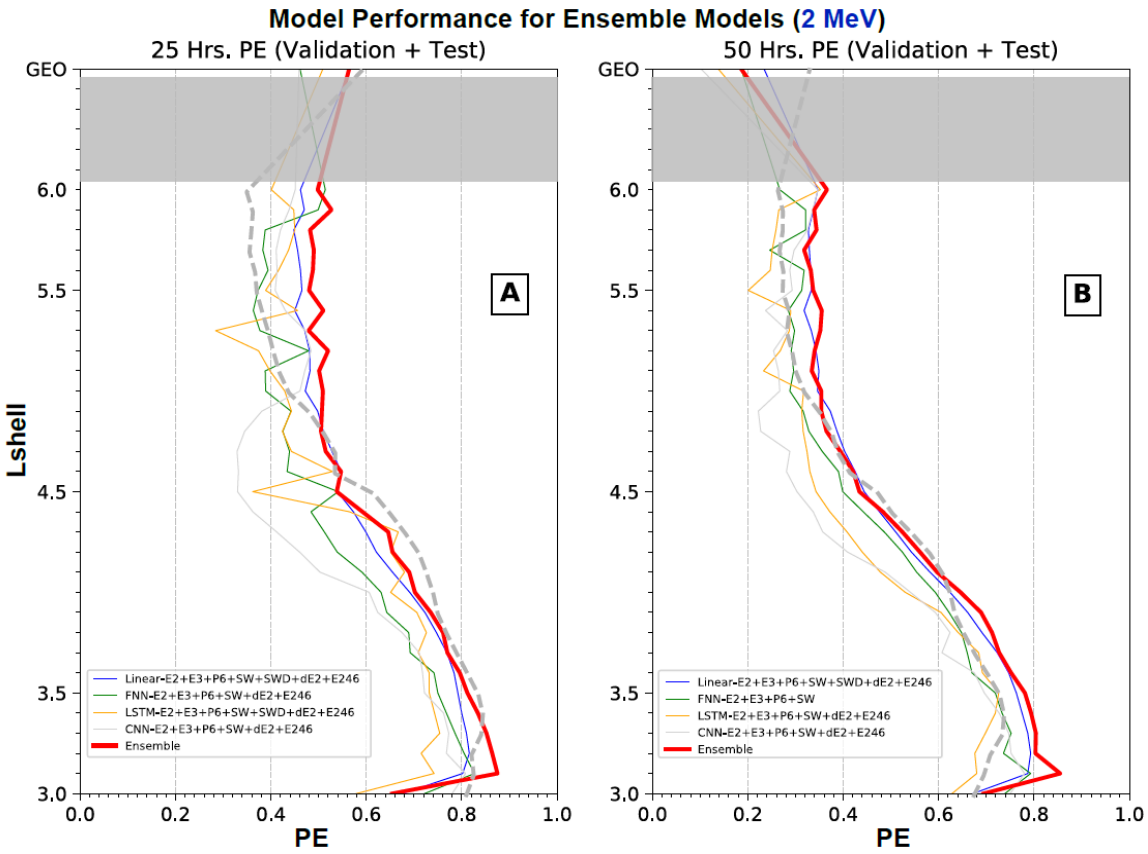


Figure 4-13 **Model PE values for validation and test data are presented as a function of L-shell for models forecasting 2 MeV electron fluxes. A)** PE curves for 1-day (25 hr) forecasting models. The thick red curve is for the ensemble model (in red) compared to those for four individual ensemble member models (the top performers defined in Table 3) in different colors. **B)** PE curves for 2-day (50 hr) forecasting models. The red curve is for the ensemble model and the other four curves are for the ensemble member models (defined in Table 4). The PE curves for the top linear model in PreMevE 2.0 making 1- and 2-day forecasts of 1 MeV electrons (P2020) are plotted for comparison.

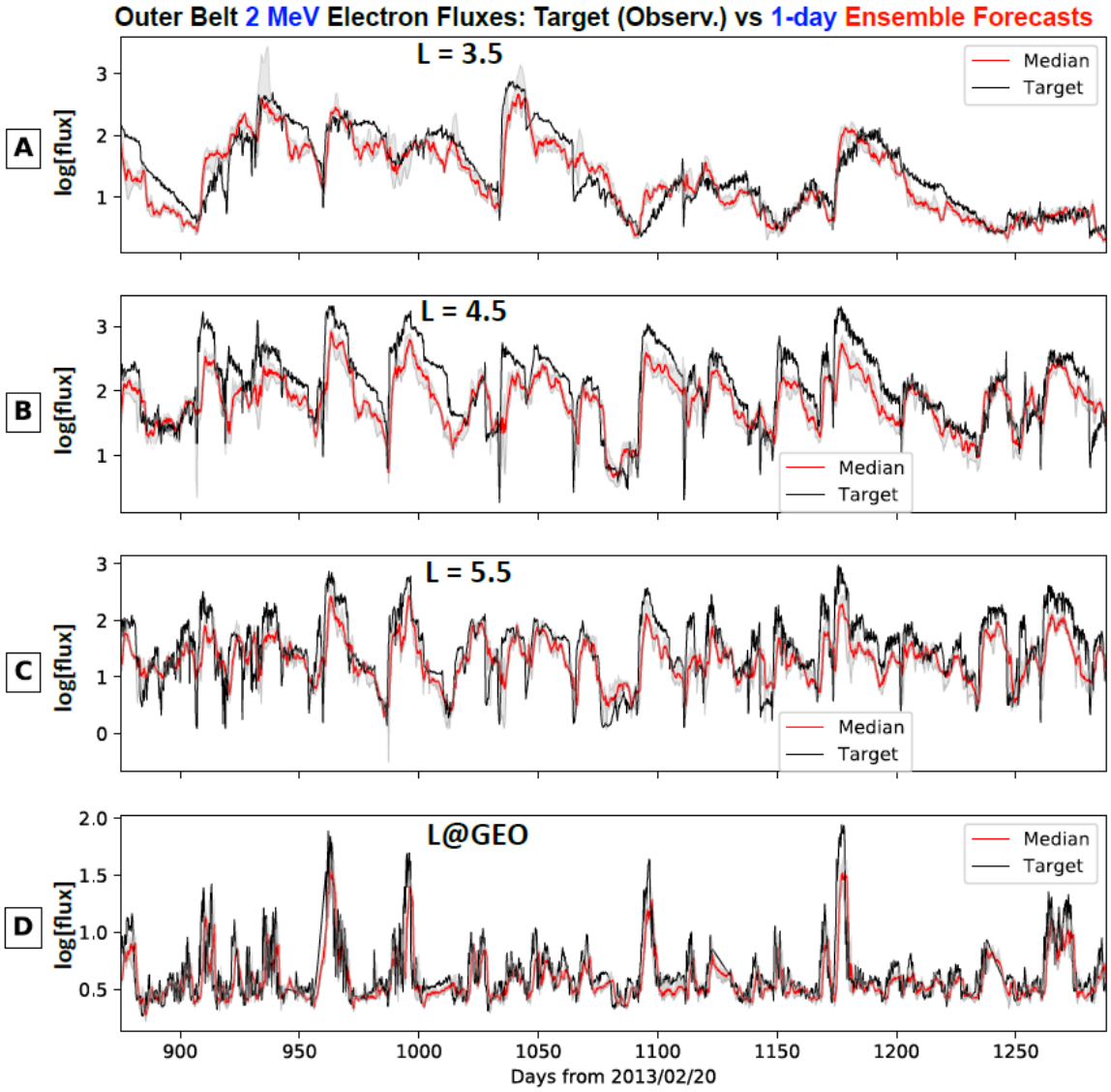


Figure 4-14 **One-day ensemble forecasting results for 2 MeV electron fluxes over individual L-shells.** Results are shown for the validation and test periods, and panels from the top to bottom are for Lshell at 3.5, 4.5, 5.5, and GEO (6.6), respectively. In each panel, the target is shown in black, and the gray strip shows the uncertainty ranges (or standard deviations) from the ensemble group, and the median from the ensemble predictions is shown in bright red color.

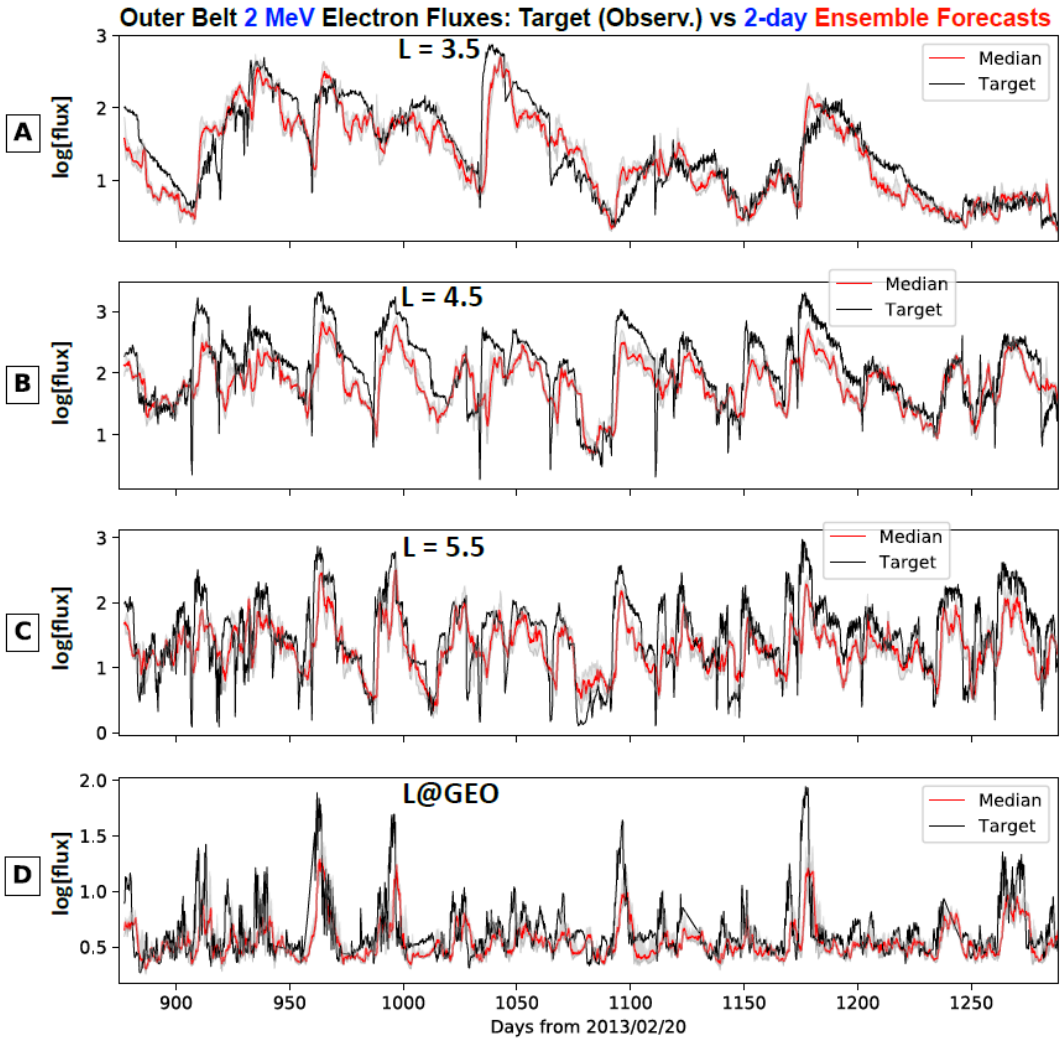


Figure 4-15 **Two-day ensemble forecasting results for 2 MeV electron fluxes over a range of L-shells.** Results are shown for the validation and test periods, and in the same format as Figure 4.14.

Tables for chapter 4

Table 4-1**Performance of models in four categories (>2 MeV) for 1-day (25 hr) forecasts.** Models in the 2nd column belong to four categories: linear regression, feedforward neural networks (FNN), long-short-term memory (LSTM), and convolutional neural networks. Window size column tells the number of 5-hourly bin data points needed as input, Input Parameters are model input combinations, and the rest columns show mean PE values for different intervals. Among the eight models in each category, the top performer—ranked by the out-of-sample performance efficiency (PE) values in the (PE val + test) column—has its model number in bold font and underscored, and the second performer has its number in bold. In the last column for PE at GEO for validation and test data sets, the highest PE value for each category is also in bold and underscored, and it may not be the same as the one from the top performer (which always has its GEO PE value underscored). The last row (model 33) is for the ensemble model PE values.

Index	Models	Window size	Input Parameters	PE train	PE validation	PE test	PE val + test	PE all	PE GEO val+test
1	LinearReg	4	E2+E3+P6+SW	0.712	0.108	0.454	0.414	0.707	0.621
2	LinearReg	16	E2+E3+P6+SW	0.742	0.194	0.509	0.470	0.736	0.623
3	LinearReg	4	E2+E3+P6+SW+dE2	0.714	0.112	0.461	0.420	0.709	0.622
4	LinearReg	16	E2+E3+P6+SW+dE2	0.747	0.197	0.523	0.479	0.741	0.625
5	LinearReg	4	E2+E3+P6+SW+dE2+E246	0.736	0.188	0.486	0.456	0.731	0.622
6	LinearReg	16	E2+E3+P6+SW+dE2+E246	0.763	0.255	0.548	0.509	0.757	0.625
7	LinearReg	4	E2+E3+P6+SW+SWD+dE2+E246	0.741	0.193	0.502	0.466	0.736	0.627
8	LinearReg	16	E2+E3+P6+SW+SWD+dE2+E246	0.770	0.266	0.568	**0.523**	0.764	**0.629**
9	FNN-64-32-elu	4	E2+E3+P6+SW	0.686	0.202	0.463	0.426	0.690	0.631
10	FNN-64-32-elu	16	E2+E3+P6+SW	0.699	0.331	0.459	0.460	0.704	0.620
11	FNN-64-32-elu	4	E2+E3+P6+SW+dE2	0.644	0.216	0.403	0.395	0.658	0.630
12	FNN-64-32-elu	16	E2+E3+P6+SW+dE2	0.716	0.319	0.511	0.488	0.720	0.603
13	FNN-64-32-elu	4	E2+E3+P6+SW+dE2+E246	0.766	0.404	0.566	**0.553**	0.765	0.630
14	FNN-64-32-elu	16	E2+E3+P6+SW+dE2+E246	0.704	0.200	0.441	0.408	0.699	0.624
15	FNN-64-32-elu	4	E2+E3+P6+SW+SWD+dE2+E246	0.713	0.266	0.456	0.446	0.713	**0.646**
16	FNN-64-32-elu	16	E2+E3+P6+SW+SWD+dE2+E246	0.715	0.195	0.392	0.384	0.703	0.621
17	LSTM-128	4	E2+E3+P6+SW	0.662	0.208	0.445	0.414	0.673	0.527
18	LSTM-128	16	E2+E3+P6+SW	0.750	0.366	0.537	0.521	0.747	0.581
19	LSTM-128	4	E2+E3+P6+SW+dE2	0.665	0.198	0.440	0.410	0.675	0.538
20	LSTM-128	16	E2+E3+P6+SW+dE2	0.740	0.287	0.526	0.489	0.737	0.588
21	LSTM-128	4	E2+E3+P6+SW+dE2+E246	0.700	0.282	0.472	0.459	0.706	0.535
22	LSTM-128	16	E2+E3+P6+SW+dE2+E246	0.781	0.401	0.545	**0.537**	0.771	0.600
23	LSTM-128	4	E2+E3+P6+SW+SWD+dE2+E246	0.671	0.140	0.387	0.365	0.674	**0.648**
24	LSTM-128	16	E2+E3+P6+SW+SWD+dE2+E246	0.799	0.348	0.507	0.499	0.777	0.571
25	Conv-64-32-relu	4	E2+E3+P6+SW	0.702	0.289	0.462	0.453	0.705	0.593
26	Conv-64-32-relu	16	E2+E3+P6+SW	-0.178	-3.765	-2.170	-2.333	-0.341	-0.002
27	Conv-64-32-relu	4	E2+E3+P6+SW+dE2	0.710	0.292	0.477	0.462	0.711	0.596
28	Conv-64-32-relu	16	E2+E3+P6+SW+dE2	0.186	-2.251	-1.138	-1.268	0.078	-0.081
29	Conv-64-32-relu	4	E2+E3+P6+SW+dE2+E246	0.719	0.324	0.480	**0.479**	0.722	**0.598**
30	Conv-64-32-relu	16	E2+E3+P6+SW+dE2+E246	0.110	-2.382	-1.334	-1.398	0.006	-0.168
31	Conv-64-32-relu	4	E2+E3+P6+SW+SWD+dE2+E246	0.749	0.285	0.497	0.477	0.742	0.566
32	Conv-64-32-relu	16	E2+E3+P6+SW+SWD+dE2+E246	0.065	-2.861	-1.636	-1.733	-0.080	0.074
33	**Ensemble: models 8 + 13 + 22 + 29**			**0.782**	**0.393**	**0.625**	**0.612**	**0.783**	**0.677**

Table 4-2 **Performance of models in four categories (>2 MeV) for 2-day (50 hr) forecasts.** In the same format as Table 4.1.

Index	Models	Window size	Input Parameters	PE train	PE validation	PE test	PE val + test	PE all	PE GEO val+test
1	LinearReg	4	E2+E3+P6+SW	0.675	0.049	0.381	0.352	0.671	0.417
2	LinearReg	16	E2+E3+P6+SW	0.702	0.120	0.433	0.400	0.696	0.421
3	LinearReg	4	E2+E3+P6+SW+dE2	0.678	0.055	0.390	0.358	0.674	0.422
4	LinearReg	16	E2+E3+P6+SW+dE2	0.707	0.127	0.444	0.409	0.701	0.427
5	LinearReg	4	E2+E3+P6+SW+dE2+E246	0.701	0.128	0.411	0.391	0.695	0.422
6	LinearReg	16	E2+E3+P6+SW+dE2+E246	0.727	0.189	0.468	**0.438**	0.720	0.428
7	LinearReg	4	E2+E3+P6+SW+SWD+dE2+E246	0.703	0.141	0.420	0.399	0.697	0.429
8	LinearReg	16	E2+E3+P6+SW+SWD+dE2+E246	0.606	0.450	0.414	0.431	0.577	**0.431**
9	FNN-64-32-elu	4	E2+E3+P6+SW	0.658	0.140	0.407	0.372	0.661	0.428
10	FNN-64-32-elu	16	E2+E3+P6+SW	0.669	0.251	0.392	0.393	0.671	0.403
11	FNN-64-32-elu	4	E2+E3+P6+SW+dE2	0.624	0.174	0.367	0.360	0.638	**0.438**
12	FNN-64-32-elu	16	E2+E3+P6+SW+dE2	0.686	0.236	0.436	0.416	0.687	0.419
13	FNN-64-32-elu	4	E2+E3+P6+SW+dE2+E246	0.715	0.315	0.460	**0.460**	0.715	0.423
14	FNN-64-32-elu	16	E2+E3+P6+SW+dE2+E246	0.670	0.109	0.373	0.340	0.664	0.433
15	FNN-64-32-elu	4	E2+E3+P6+SW+SWD+dE2+E246	0.679	0.208	0.388	0.387	0.679	0.425
16	FNN-64-32-elu	16	E2+E3+P6+SW+SWD+dE2+E246	0.681	0.133	0.322	0.322	0.668	0.385
17	LSTM-128	4	E2+E3+P6+SW	0.634	0.141	0.386	0.359	0.644	**0.425**
18	LSTM-128	16	E2+E3+P6+SW	0.711	0.290	0.461	0.451	0.708	0.384
19	LSTM-128	4	E2+E3+P6+SW+dE2	0.640	0.139	0.387	0.360	0.649	0.420
20	LSTM-128	16	E2+E3+P6+SW+dE2	0.706	0.238	0.453	0.428	0.701	0.360
21	LSTM-128	4	E2+E3+P6+SW+dE2+E246	0.668	0.193	0.394	0.385	0.672	0.366
22	LSTM-128	16	E2+E3+P6+SW+dE2+E246	0.739	0.307	0.457	**0.456**	0.729	0.390
23	LSTM-128	4	E2+E3+P6+SW+SWD+dE2+E246	0.644	0.110	0.342	0.328	0.647	0.418
24	LSTM-128	16	E2+E3+P6+SW+SWD+dE2+E246	0.743	0.252	0.405	0.407	0.723	0.335
25	Conv-64-32-relu	4	E2+E3+P6+SW	0.676	0.227	0.396	0.394	0.676	0.403
26	Conv-64-32-relu	16	E2+E3+P6+SW	-0.115	-3.656	-2.088	-2.227	-0.283	0.048
27	Conv-64-32-relu	4	E2+E3+P6+SW+dE2	0.684	0.237	0.407	0.404	0.683	**0.403**
28	Conv-64-32-relu	16	E2+E3+P6+SW+dE2	0.209	-2.242	-1.139	-1.254	0.094	-1.397
29	Conv-64-32-relu	4	E2+E3+P6+SW+dE2+E246	0.699	0.269	0.415	**0.423**	0.698	0.402
30	Conv-64-32-relu	16	E2+E3+P6+SW+dE2+E246	0.181	-2.220	-1.240	-1.287	0.070	-0.105
31	Conv-64-32-relu	4	E2+E3+P6+SW+SWD+dE2+E246	0.711	0.236	0.411	0.408	0.703	0.345
32	Conv-64-32-relu	16	E2+E3+P6+SW+SWD+dE2+E246	0.125	-2.594	-1.530	-1.588	-0.020	-0.328
33	**Ensemble: models 6 + 13 + 22 + 29**			**0.738**	**0.299**	**0.532**	**0.521**	**0.738**	**0.572**

Table 4-3 **Performance of models in four categories (2 MeV) for 1-day (25 hr) forecasts.** In the same format as Table 4.1.

Index	Models	Window size	Input Parameters	PE train	PE validation	PE test	PE val + test	PE all	PE GEO val+test
1	LinearReg	4	E2+E3+P6+SW	0.746	0.358	0.591	0.538	0.744	0.549
2	LinearReg	16	E2+E3+P6+SW	0.769	0.427	0.628	0.583	0.768	0.561
3	LinearReg	4	E2+E3+P6+SW+dE2	0.748	0.363	0.596	0.543	0.747	0.554
4	LinearReg	16	E2+E3+P6+SW+dE2	0.773	0.432	0.637	0.590	0.772	0.568
5	LinearReg	4	E2+E3+P6+SW+dE2+E246	0.761	0.383	0.594	0.550	0.755	0.553
6	LinearReg	16	E2+E3+P6+SW+dE2+E246	0.781	0.440	0.632	0.590	0.776	**0.568**
7	LinearReg	4	E2+E3+P6+SW+SWD+dE2+E246	0.768	0.385	0.595	0.551	0.759	0.555
8	LinearReg	16	E2+E3+P6+SW+SWD+dE2+E246	0.792	0.450	0.645	**0.600**	0.784	0.566
9	FNN-64-32-elu	4	E2+E3+P6+SW	0.763	0.396	0.582	0.548	0.756	0.535
10	FNN-64-32-elu	16	E2+E3+P6+SW	0.721	0.298	0.481	0.451	0.710	0.471
11	FNN-64-32-elu	4	E2+E3+P6+SW+dE2	0.679	0.105	0.411	0.344	0.663	**0.582**
12	FNN-64-32-elu	16	E2+E3+P6+SW+dE2	0.700	0.238	0.476	0.426	0.693	0.505
13	FNN-64-32-elu	4	E2+E3+P6+SW+dE2+E246	0.769	0.417	0.571	**0.549**	0.760	0.461
14	FNN-64-32-elu	16	E2+E3+P6+SW+dE2+E246	0.727	0.249	0.455	0.422	0.708	-0.735
15	FNN-64-32-elu	4	E2+E3+P6+SW+SWD+dE2+E246	0.782	0.389	0.554	0.529	0.762	0.572
16	FNN-64-32-elu	16	E2+E3+P6+SW+SWD+dE2+E246	0.730	0.168	0.418	0.369	0.695	0.487
17	LSTM-128	4	E2+E3+P6+SW	0.705	0.295	0.493	0.456	0.702	**0.578**
18	LSTM-128	16	E2+E3+P6+SW	0.751	0.387	0.539	0.518	0.742	0.539
19	LSTM-128	4	E2+E3+P6+SW+dE2	0.713	0.295	0.503	0.463	0.708	0.551
20	LSTM-128	16	E2+E3+P6+SW+dE2	0.744	0.360	0.519	0.496	0.731	0.509
21	LSTM-128	4	E2+E3+P6+SW+dE2+E246	0.757	0.362	0.539	0.511	0.746	0.577
22	LSTM-128	16	E2+E3+P6+SW+dE2+E246	0.791	0.423	0.525	0.525	0.764	0.516
23	LSTM-128	4	E2+E3+P6+SW+SWD+dE2+E246	0.782	0.395	0.556	0.533	0.764	0.539
24	LSTM-128	16	E2+E3+P6+SW+SWD+dE2+E246	0.836	0.453	0.551	**0.549**	0.795	0.509
25	Conv-64-32-relu	4	E2+E3+P6+SW	0.762	0.309	0.548	0.496	0.744	0.437
26	Conv-64-32-relu	16	E2+E3+P6+SW	0.602	-0.596	-0.096	-0.186	0.494	-0.549
27	Conv-64-32-relu	4	E2+E3+P6+SW+dE2	0.770	0.340	0.566	0.518	0.754	**0.437**
28	Conv-64-32-relu	16	E2+E3+P6+SW+dE2	0.637	-0.467	0.031	-0.057	0.545	-0.490
29	Conv-64-32-relu	4	E2+E3+P6+SW+dE2+E246	0.782	0.373	0.556	**0.525**	0.762	0.459
30	Conv-64-32-relu	16	E2+E3+P6+SW+dE2+E246	0.638	-0.422	-0.050	-0.090	0.533	-0.898
31	Conv-64-32-relu	4	E2+E3+P6+SW+SWD+dE2+E246	0.801	0.329	0.540	0.500	0.766	0.430
32	Conv-64-32-relu	16	E2+E3+P6+SW+SWD+dE2+E246	0.670	-0.537	-0.086	-0.170	0.529	-0.631
33	**Ensemble: models 8 + 13 + 24 + 29**			**0.810**	**0.476**	**0.640**	**0.624**	**0.796**	**0.564**

Table 4-4 **Performance of models in four categories (2 MeV) for 2-day (50 hr) forecasts.** In the same format as Table 4.1.

Index	Models	Window size	Input Parameters	PE train	PE validation	PE test	PE val + test	PE all	PE GEO val+test
1	LinearReg	4	E2+E3+P6+SW	0.701	0.288	0.500	0.461	0.700	0.186
2	LinearReg	16	E2+E3+P6+SW	0.721	0.339	0.535	0.497	0.720	0.227
3	LinearReg	4	E2+E3+P6+SW+dE2	0.703	0.294	0.506	0.466	0.702	0.200
4	LinearReg	16	E2+E3+P6+SW+dE2	0.725	0.349	0.544	0.506	0.724	0.244
5	LinearReg	4	E2+E3+P6+SW+dE2+E246	0.717	0.309	0.500	0.469	0.710	0.198
6	LinearReg	16	E2+E3+P6+SW+dE2+E246	0.735	0.354	0.534	0.502	0.729	**0.244**
7	LinearReg	4	E2+E3+P6+SW+SWD+dE2+E246	0.720	0.323	0.504	0.475	0.714	0.212
8	LinearReg	16	E2+E3+P6+SW+SWD+dE2+E246	0.743	0.364	0.546	**0.512**	0.735	0.234
9	FNN-64-32-elu	4	E2+E3+P6+SW	0.718	0.333	0.494	**0.474**	0.713	0.186
10	FNN-64-32-elu	16	E2+E3+P6+SW	0.669	0.212	0.395	0.370	0.661	0.222
11	FNN-64-32-elu	4	E2+E3+P6+SW+dE2	0.642	0.041	0.348	0.286	0.627	**0.258**
12	FNN-64-32-elu	16	E2+E3+P6+SW+dE2	0.661	0.143	0.394	0.346	0.650	-0.044
13	FNN-64-32-elu	4	E2+E3+P6+SW+dE2+E246	0.718	0.332	0.472	0.461	0.710	0.105
14	FNN-64-32-elu	16	E2+E3+P6+SW+dE2+E246	0.685	0.153	0.373	0.341	0.665	-0.891
15	FNN-64-32-elu	4	E2+E3+P6+SW+SWD+dE2+E246	0.725	0.306	0.446	0.436	0.706	0.081
16	FNN-64-32-elu	16	E2+E3+P6+SW+SWD+dE2+E246	0.681	0.087	0.327	0.289	0.647	0.146
17	LSTM-128	4	E2+E3+P6+SW	0.667	0.197	0.407	0.373	0.660	0.191
18	LSTM-128	16	E2+E3+P6+SW	0.699	0.270	0.435	0.416	0.687	0.204
19	LSTM-128	4	E2+E3+P6+SW+dE2	0.673	0.191	0.409	0.373	0.663	**0.265**
20	LSTM-128	16	E2+E3+P6+SW+dE2	0.689	0.237	0.435	0.404	0.678	0.224
21	LSTM-128	4	E2+E3+P6+SW+dE2+E246	0.702	0.261	0.439	0.418	0.691	0.201
22	LSTM-128	16	E2+E3+P6+SW+dE2+E246	0.735	0.322	0.440	**0.438**	0.712	0.138
23	LSTM-128	4	E2+E3+P6+SW+SWD+dE2+E246	0.715	0.271	0.426	0.413	0.696	0.247
24	LSTM-128	16	E2+E3+P6+SW+SWD+dE2+E246	0.772	0.327	0.436	0.435	0.730	0.106
25	Conv-64-32-relu	4	E2+E3+P6+SW	0.716	0.248	0.463	0.425	0.700	**0.125**
26	Conv-64-32-relu	16	E2+E3+P6+SW	0.562	-0.694	-0.195	-0.279	0.448	-0.332
27	Conv-64-32-relu	4	E2+E3+P6+SW+dE2	0.720	0.236	0.451	0.413	0.699	0.105
28	Conv-64-32-relu	16	E2+E3+P6+SW+dE2	0.670	-0.359	0.101	0.031	0.582	-1.124
29	Conv-64-32-relu	4	E2+E3+P6+SW+dE2+E246	0.740	0.294	0.456	**0.439**	0.717	0.102
30	Conv-64-32-relu	16	E2+E3+P6+SW+dE2+E246	0.584	-0.605	-0.203	-0.245	0.464	-0.980
31	Conv-64-32-relu	4	E2+E3+P6+SW+SWD+dE2+E246	0.745	0.219	0.425	0.393	0.708	0.076
32	Conv-64-32-relu	16	E2+E3+P6+SW+SWD+dE2+E246	0.680	-0.530	-0.117	-0.173	0.532	-1.216
33	**Ensemble: models 8 + 9 + 22 + 29**			**0.743**	**0.361**	**0.543**	**0.521**	**0.736**	**0.186**

References

Baker, D. N. et al. (2012). The Relativistic Electron-Proton Telescope (REPT) instrument on board the Radiation Belt Storm Probes (RBSP) spacecraft: Characterization of Earth's radiation belt high-energy particle populations. *Space Sci. Rev.*, 10.1007/s11214-012-9950-9.

Blake, J.B., W.A. Kolasinski, R.W. Filius, and E.G. Mullen (1992), Injection of Electrons and Protons with Energies of Tens of MeV into L<3 on March 24, 1991, *Geophys. Res. Lett.*, **19**, 821

Boynton, R. J., M. A. Balikhin, and D.Mourenas (2014), Statistical analysis of electron lifetimes at GEO: Comparisons with chorus-driven losses, *J. Geophys. Res.* Space Physics, **119**, 6356–6366, doi:10.1002/2014JA019920

Chen, Y., G. D. Reeves, G. S. Cunningham, R. J. Redmon, and M. G. Henderson, 2016, Forecasting and remote sensing outer belt relativistic electrons from low earth orbit: *Geophysical Research Letters*, **43**, 1031–1038

Chen, Y., G. D. Reeves, X. Fu, and M. Henderson, 2019, PreMevE: New predictive model for megaelectron-volt electrons inside earths outer radiation belt: *Space Weather*, **17**, 438–454.

Cheung, K.K.W. (2001), A review of ensemble forecasting techniques with a focus on tropical cyclone forecasting, *Meteorol. Appl.*, **8**, 315-332

Claudepierre, S. G. et al. (2015). A background correction algorithm for Van Allen Probes MagEIS electron flux measurements. *J. of Geophys. Res.*, **120**, 5703–5727. 10.1002/2015JA021171

Claudepierre, S. G., & O'Brien, T. P. (2020). Specifying high-altitude electrons using low-altitude LEO systems: The SHELLS model. Space Weather, **18**, e2019sw002402. https://doi.org/10.1029/2019sw002402

Clevert, D.‐A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (ELUs). ArXiv E‐Prints, arXiv, 1511, 07289.

Fennell, J. F., S. G. Claudepierre, J. B. Blake, T. P. O'Brien, J.H. Clemmons, D. N. Baker, H. E. Spence, and G. D. Reeves (2015), Van Allen Probes show that the inner radiation zone contains no MeV electrons: ECT/MagEIS data, *Geophys. Res. Lett.*, **42**, 1283–1289, doi:10.1002/2014GL062874

Friedel, R. H. W., S. Bourdarie, and T. E. Cayton (2005), Intercalibration of magnetospheric energetic electron data, *Space Weather*,**3**, S09B04, doi:10.1029/2005SW000153

Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., & Seung, H. S. (2000). Digital selection and analogue amplification coexist in a cortex‐inspired silicon circuit. Nature, 405(6789), 947−951. https://doi.org/10.1038/35016072

Jin, Y., 2006, Multi-objective machine learning: Springer Science & Business Media, 16.

Knipp, D. J. (2016), Advances in Space Weather Ensemble Forecasting, *Space Weather*, **14**, 52–53, doi:10.1002/2016SW001366.

Kuepfer, L., M. Peter, U. Sauer, and J. Stelling, 2007, Ensemble modeling for analysis

of cell signaling dynamics: Nature biotechnology, 25, 1001–1006

Li, W., & Hudson, M. K. (2019). Earth's Van Allen radiation belts: From discovery to the Van Allen Probes era. *Journal of Geophysical Research*, **124**, 8319–8351. https://doi.org/10.1029/2018JA025940

Meier, M. M., Belian, R. D., Cayton, T. E., Christensen, R. A., Garcia, B., Grace, K. M., et al. (1996). The energy spectrometer for particles (ESP): Instrument description and orbital performance. In Workshop on the Earth's trapped particle environment (Vol. 383, pp. 203–210). New York: Am. Inst. Phys. Conf. ROC.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In Proceedings of the 27th International Conference on International Conference on Machine Learning (pp. 807–814). USA: Omnipress. Retrieved from http://dl.acm.org/citation.cfm?id = 3104322.3104425

Pham, B. T., T. V. Phong, T. Nguyen-Thoi, K. Parial, S. K. Singh, H.-B. Ly, K. T.

Nguyen, L. S. Ho, H. V. Le, and I. Prakash, 2020, Ensemble modeling of landslide

susceptibility using random subspace learner and different decision tree classifiers:

Geocarto International, 1–23.

Pires de Lima, R., Y. Chen, and Y. Lin, 2020, Forecasting megaelectron-volt electrons inside earth’s outer radiation belt: PreMevE 2.0 based on supervised machine learning algorithms: *Space Weather*, **18**, e2019SW002399, https:// doi.org/10.1029/2019SW002399

Reagan, J.B., R.E. Meyerott, E.E. Gaines, R.W. Nightingale, P.C. Filbert and W.L. Imhof (1983), Space charging currents and their effects on spacecraft systems, *IEEE Trans. Electr. Insul.*, **18**, 354.

Sicard-Piet, A., S. Bourdarie, D. Boscher, R. H. W. Friedel, M. Thomsen, T. Goka, H. Matsumoto, and H. Koshiishi (2008), A new international geostationary electron model: IGE-2006, from 1 keV to 5.2 MeV, *Space Weather*, **6**, S07003, doi:10.1029/2007SW000368

Toth, Z. & Kalnay, E. (1997). Ensemble forecasting at NCEP and the breeding method. *Mon. Wea. Rev.*, **125**: 3297−3319

Vette, J. (1991), The AE-8 trapped electron model environment, *NASA Technical Memorandum*, NASA-TM-107820, https://ntrs.nasa.gov/citations/19920014985

Conclusions and final remarks

Deep learning is a powerful technique that promises to extract and use patterns seen in data to solve a great many previously unsolvable problems. In this dissertation I have shown that deep learning provides excellent results in predicting leaks from oil-field wellhead data and the prediction of high energy electron fluxes. When I started this dissertation I hypothesized that deep learning would also solve the problem of mapping parasequence sets from hundreds of gamma-ray logs. I tried alternative deep learning architectures to solve this problem to no avail. Instead, the application of more traditional (though state-of-the-art) statistical techniques guided by the human interpreter provided superior results. Even in chapter 4, linear regression produces equally good results for the lower to mid L shells for all but the ensemble deep learning model. .

The nature of the problem, amount of data available and ultimate aim of the project are extremely important junctions in the project life span to spend time before jumping into the details of a sophisticated architecture. I show in chapter 3 an application of automating the task of leak detection by posing the problem as anomaly detection. In general, machines are extremely good at performing repetitive tasks with high accuracy while humans are extremely good at recognizing new patterns. At the time of writing this dissertation, there are already claims of a "singularity" being achieved where machine intelligence supersedes the human intellect. However, most of these tools are still either not commonly available or the claims of singularity itself disputed among scientists. Nonetheless, I find in the geosciences applications addressed in this dissertation that machine learning holds great promise in automating repetitive tasks, but find little evidence of machines superseding human "intuition".