CS230
SPR 19

Engineer Level Automated Interpretation of Geothermal Well Logs

Ahinoam Pollack, ahinoamp@Stanford.edu

Stanford
SCHOOL OF EARTH, ENERGY & ENVIRONMENTAL SCIENCES

1. What am I predicting? geothermal reservoir properties

- Diagnosing reservoir properties is crucial for making engineering decisions for a geothermal power plant.
- Temperature and pressure well log data is collected by sensors that are lowered along a wellbore, recording measurements as a function of depth (Fig. 1).
- This data is used to ascertain five reservoir properties, shown in Fig. 1: top feed zone (1), bottom feed zone (2), reservoir temperature (3), reservoir pressure (4), and depth of reservoir pressure (5).
- Classically, this analysis is done visually by a geothermal reservoir engineer.
- We train a neural network once, which can then supply engineer-level results within seconds.

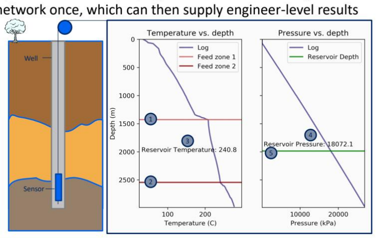


Figure 1. (left) a sensor collects subsurface information from a wellbore. (right) An example of data collected by the sensor as a function of depth (the log data) and the five characteristics determined by a reservoir engineer based on the well data.

2. What is the input and output for each sample?

- Input: temperature and pressure well logs – two "depth – series".
- Additional extracted input feature: the temperature derivative w.r.t. depth.
- Output: five reservoir properties.

3. What is the available data?

We have three sets of data:

- Training data: 10,000 generated (synthetic) temperature and pressure series, of length 200. Shape: [10000,200,3].
- Validation data: 12 real sets of temperature and pressure series, used for hyper-parameter tuning. The data was resampled to be of length 200 and normalized to have values between 0 and 1. Shape: [12,200,3]
- Testing data: 9 real sets of temperature and pressure series, resampled to be of length 200 and normalized to have values between 0 and 1. Shape: [9,200,3].

Input Channels [m,200,3]

Temperature
$$\begin{bmatrix} T_1^{[1]} & \cdots & T_{200}^{[1]} \\ \vdots & \ddots & \vdots \\ T_1^{[m]} & \cdots & T_{200}^{[m]} \end{bmatrix}$$

Pressure
$$\begin{bmatrix} P_1^{[1]} & \cdots & P_{200}^{[1]} \\ \vdots & \ddots & \vdots \\ P_1^{[m]} & \cdots & P_{200}^{[m]} \end{bmatrix}$$

Temperature derivative
$$\begin{bmatrix} \partial T_1^{[1]} & \cdots & \partial T_{200}^{[1]} \\ \vdots & \ddots & \vdots \\ \partial T_1^{[m]} & \cdots & \partial T_{200}^{[m]} \end{bmatrix}$$

Predict →

Five reservoir properties [m, 5]
$$\begin{bmatrix} P_1^{[1]} & \cdots & P_5^{[1]} \\ \vdots & \ddots & \vdots \\ P_1^{[m]} & \cdots & P_5^{[m]} \end{bmatrix}$$

4. Architecture: convolutional neural network

- I used a Convolution Neural Network (CNN) for this prediction task since CNNs have been proven to be effective for spatial data, as shown by the success of the ImageNet algorithm [1].
- General architecture:
  1. Divide the data into batches
  2. Execute initial convolution.
  3. Execute middle convolution block *n* times:
     - One-dimensional convolution
     - Optional Batch norm
     - Activation function
     - Optional Max pooling and Optional dropout
  4. Flatten
  5. Fully connected layer with five outputs
- **Cost**: mean squared error. **Metrics**: mean absolute error.
- **Optimization algorithm**: Adam. **Learning rate**: exponential learning rate decay.
- **Early stopping**: save model when validation error is lowest

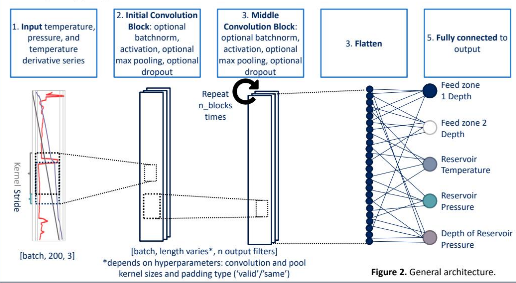


Figure 2. General architecture.

5. Hyperparameter tuning

- There were a total of 25 hyperparameters to choose for this setup.
- Hyperparameter choice was done via a stochastic algorithm that sampled from the range of hyper parameters, and then evaluated the results.
- 200 training iterations with different hyperparameters were performed to find the best network.
- A sensitivity analysis in Fig. 3 shows that the most important parameter is the number of convolution blocks.
- Fig. 4 shows that the mean error decreases as a function of the number of convolution blocks.
- The best model had 4 convolution blocks, 'valid' initial padding, and an initial kernel size of 12.

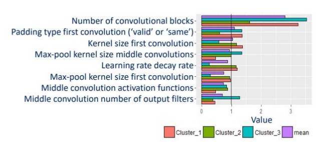


Figure 3. Sensitivity analysis of hyperparameters. Values over 1 indicate a high influence of the parameter on the error.

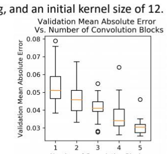
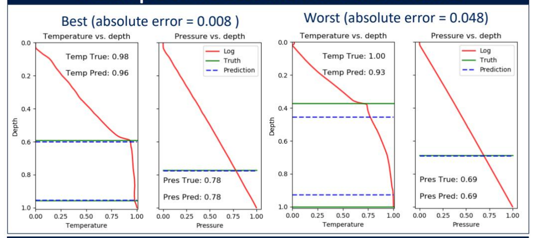


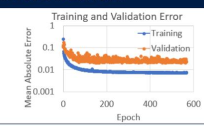
Figure 4. Influence of the number of convolution blocks on the mean absolute error.

7. Example of the best and worst of the results



8. Summary results of selected network

Training Error	Validation error	Test error
0.008	0.020	0.030



9. Discussion

- The CNN framework successively predicted the reservoir characteristics for easy pressure and temperature series in the training, testing and validation sets.
- In the real data, the solution may sometimes be ambiguous even for an engineer and therefore the algorithm does not perform as well as expected.
- Therefore, I suggest implementing an algorithm which gives several possible interpretations for an ambiguous input and a single interpretation for an easy input.

10. Future/current – providing several possible solutions in ambiguous cases

- I have started working on the above suggestion, using a similar architecture as that presented in section 4. However, instead of a fully connected layer in the end with five properties, there are several possible outputs that branch out:
  - An output that detects the first feed zone that is a (200,1) layer followed by a softmax activation. Turning this into a classification problem.
  - A similar output that detects the second feed zone.
- Similar to Google Maps, the algorithm would then suggest the classifications with the highest confidence. It could be just one answer or many interpretations based on the relative confidence levels.
- An initial result of this multiple output algorithm is shown in Fig. 5. On the right, is an example of relative confidence of the algorithm in the solution. On the left, the algorithm suggests many interpretations.
- The interpretations can be limited to not be too close to each other.

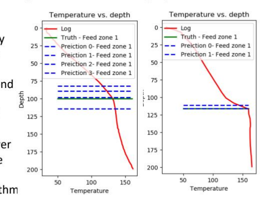


Figure 5. The algorithm provides multiple predictions for the input.

13. References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutionalneural networks. InAdvances in neural information processing systems, pages 1097–1105, 2012.