# Flowrate Estimation from Sensor Readings for Remote Well Monitoring


Daniel Helm & Joe Pauly | Department of Computer Science at the University of Oklahoma

CS 5043 — Advanced Machine Learning | Dr. Andrew Fagg

*In cooperation with Plow Technologies, LLC*

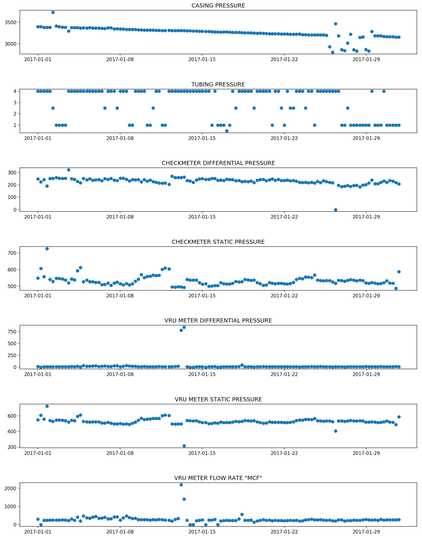

## Introduction

Through boom and bust cycles of commodity prices, the oil and gas industry adapts to stay alive. Most recently, the bust of 2014 catalyzed industry efforts to drive down operational costs through efficiencies in operations. These efficiencies are a direct result from oilfield digitalization. Energy companies collect streaming data across offshore platforms, drilling rigs, onshore wellpads, pipelines, water treatment facilities. Plow Technologies, LLC provides streaming services to energy operators. In an effort to increase confidence in data capture and alert operators of monitoring or machine malfunctions, machine learning can be applied on the time series data collected by Plow Tech.

## Problem

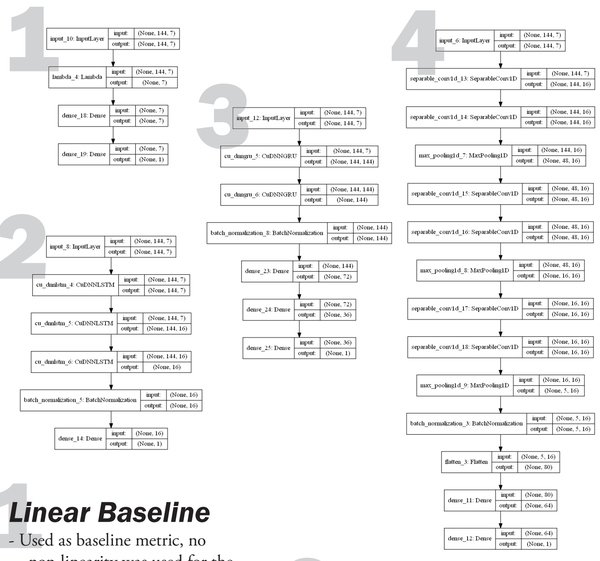
***Using sensor data from a site, can we estimate flowrate?***

- Goal: Use time series data collected from monitoring sensors on the well to estimate and confirm flowrate measurements.
- Checkmeter flowrate measurments will be used for training the model, but in prediction, the model will see have no previous measurements of flowrate.
- Inputs: Sensor measurments for casing pressure, tubing pressure, checkmeter differential pressure, checkmeter static pressure, VRU meter differential pressure, VRU meter static pressure, and VRU meter flow rate for 2017 to date. All other data from the site is ignored.
- Inputs are bucketed 5 minute averages for each sensor on a chosen pad. Sensors have been heavily filtered before being logged to make noise a non-issue.
- Input values correspond to real-world units, and as such, vary drastically in scale. As such, we scale and center the data around 0 in pre-processing.
- Our model output is the flowrate at the checkmeter, as measured in MCF (thousand cubic feet), with a goal of minimizing the model's MAE (mean absolute error).




## Approach

- Working with time-series data drove much of our approach. We tested the most prominent deep learning architectures for sequential data: Recursive Neural Nets (RNN) & 1-Dimensional Convolutional Neutal Nets (1D Convnet)
- Took "best practice" advice into much of our decision making. Among these is use of ReLU for the output all dense layers using an activation function.
- All models use 144 time steps (about 12 hours) to make predictions. Training done using 500 randomized batches per epoch, each consisting of 128 samples.
- Specific architectures were arrived at by training several different implementations, trained on 2017 data from January through August. Validation and testing was done on remaining months. Mean Absolute Error was used as our success metric.




### 1 Linear Baseline

- Used as baseline metric, no non-linearity was used for the activation function, only the final timestep in a sequence is looked at.

-64 Params, trained 25 epochs

### 2 LSTM

-Simple, narrow approach stacking LSTMs. Uses Nadam optimizer and Xavier uniform initial weights

-Batch Normalization used to regularize data before dense layer

-4,305 Params, trained 30 epochs

### 3 GRU

- 2 Layer approach, stacking wide GRUs. Uses RMSProp optimizer and Xavier uniform initial weights

-Batch Normalization used to regularize data before 2 wide dense layers

-205,057 Params, trained 15 epochs

### 4 1D Convnet

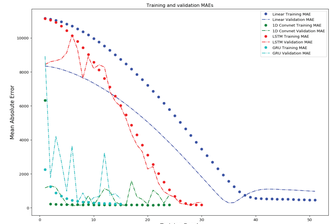
-Deep, narrow approach taken to stacking 1D separable convolution layers. 1 time step stride and window size of 5 time steps. Uses Nadam optimizer and Xavier uniform initial weights

-Max Pooling of size 3 used every 2 Convolutional layers, with Batch Normalization done after the convolutional layers.

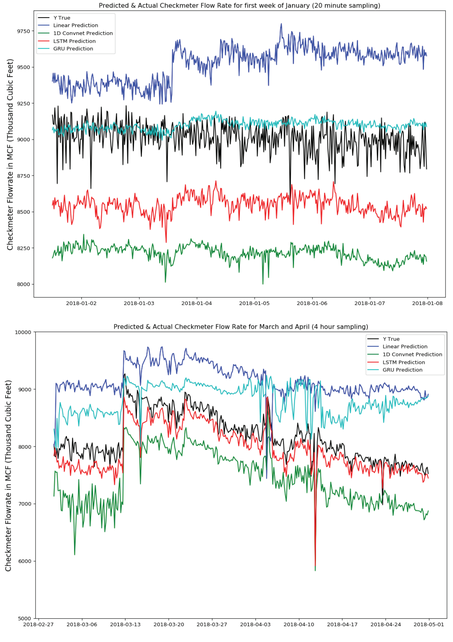
-7,236 Params, trained 24 epochs

## Model Performance

Because of the structure of our data, model performance was gagued on new data that was unused during our initial investigations into model architecture. Being time-series data unique to a site and with sections of time not being independent (future oil resevoirs depend on current levels), we are unable to do n-fold cross validation, but rather must rate performance based on our linear baseline model.




Our validation scores were much more "jumpy" compared to our exploratory tests, indicating over-fitting of the model might be more apparent when there is greater separation in time between the training and test sets.




As seen above, all our models track the basic movement of the flowrate. As the test set gets more temporally removed from the training set, the GRU and Linear models stray, while the LSTM model seems to sustain it performance.

Final Test MSEs for 2018 YTD data were --

| | |
|---|---|
| Linear: | 747.22 |
| 1DConvnet: | 757.86 |
| LSTM: | 461.25 |
| GRU: | 483.35 |

## Conclusion

- As our models behaved differently when our dataset changed, we'd need to provide more protection in our models against overfitting. The training curve for 1DConvnet drastically changed and as a result it went from the best performing model to the worst.
- The LSTM model had the fewest parameters of the RNN architectures, yet scored the highest.
- There was a drastic change in the flowrate in March 2018, which probably indicates work being done on the site. All models successfully adapted to the rapid change.
- Depending on the commercial application, our models currently might not be worth the training costs in comparison with more simple estimators like Linear Regression.
- Further research will need to include the applicability of these architectures to other sensor measurments and different wellpads in order to create a more comprehesive understanding of how a site behaves during normal operation.
- Ultimately, we hope this research will allow service providers to automate alarms for when a wellsite operates outside of a model's learned parameters.

## References


Chollet, François. *Deep Learning with Python.* Manning Publications Co., 2018.

Chollet, François et al. "Keras: The Python Deep Learning Library." *Keras Documentation*, 2015, keras.io/.

Dozat, Timothy. "Incorporating Nesterov Momentum into Adam." (2015).

Géron, Aurélien. *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* 1st ed., OReilly, 2017.

Glorot, Xavier and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." *AISTATS* (2010).


*Photo from http://themilliondollarway.blogspot.com/2015/07/another-18-well-super-pad-hess-en.html*

## Abstract


Automation and real time monitoring drastically reduces operational costs in the oil field from running a drilling rig on a tight schedule to daily production operations. Plow Technologies, LLC collects streaming data on individual well pads for oil and gas operators. Flowrate of the production stream is the most important data on a wellpad. Machine learning can be applied to estimate production flowrate. Multiple measurements from the wellhead, checkmeter, and vapor recovery unit (VRU) are used as inputs to estimate production flowrate in real time. Deep learning architectures were explored for this problems and tested against a linear regression baseline performance.

Three architectures were explored, each a different method of dealing with sequential data: a GRU stack, an LSTM stack and a 1D CNN. The GRU and LSTM models outperformed the linear baseline, but it is unclear if any performed well enough of commercial implementation. Each model behaved differently and scored more poorly than in the exploratory phases of the research, so more exploration into model specifics is necessary.