By:

APGCE ASIA PETROLEUM GEOSCIENCE CONFERENCE & EXHIBITION
'Asia's Premier Geoscience Event'

PETRONAS

ASIA PETROLEUM GEOSCIENCE CONFERENCE & EXHIBITION (APGCE) 2022

Evolving Geoscience for a Sustainable Future

28-29 November 2022
Kuala Lumpur Convention Centre
Malaysia

Convolutional Neural Networks to Augment PSDM Velocity Model Building

Greg Cameron, MDSA
Rob Vestrum, PhD

TBI

Organised By: ICEP INTERNATIONAL CONFERENCE & EXHIBITION PROFESSIONALS
In Collaboration With: EAGE EUROPEAN ASSOCIATION OF GEOSCIENTISTS & ENGINEERS

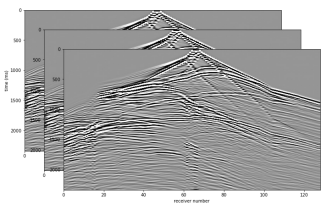
Visit the website to find out more at
icep.com.my/apgce

ASIA PETROLEUM GEOSCIENCE CONFERENCE & EXHIBITION (APGCE) 2022
Evolving Geoscience for a Sustainable Future
28-29 November 2022
Kuala Lumpur Convention Centre
Malaysia

APGCE ASIA PETROLEUM GEOSCIENCE CONFERENCE & EXHIBITION
'Asia's Premier Geoscience Event'

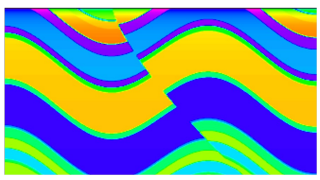
By: PETRONAS

Can we estimate PSDM velocity models using Convolutional Neural Networks?



Shot Records

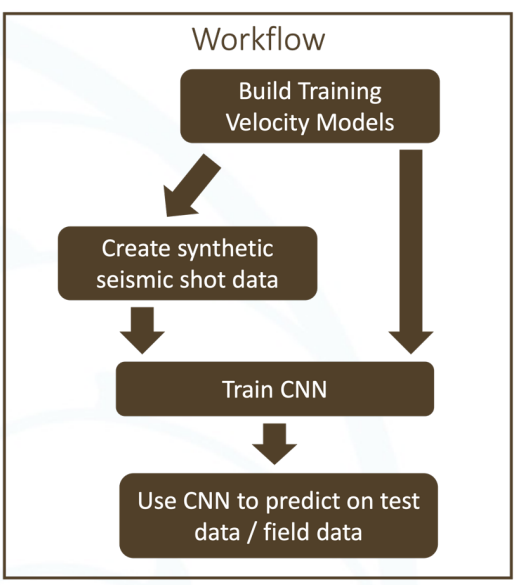
Input → Convolutional Neural Network (CNN) → Predict



Velocity Model

We investigate using Convolutional Neural Networks (CNN) to augment PSDM velocity model building. Our goal was to train a CNN to be able to predict a PSDM velocity model from seismic shot records.

The motivation for this was straightforward. PSDM velocity model building in complex structure land environments is difficult. We often have limited geological constraints, with few (if any) wells and an incomplete understanding of the subsurface structure. The seismic data in these areas is usually sparse due to economic and environmental constraints. The complex near surface results in data with poor signal/noise and frequency content. Due to these factors, automated methods such as reflection tomography and full waveform inversion often struggle. Can a CNN do any better?



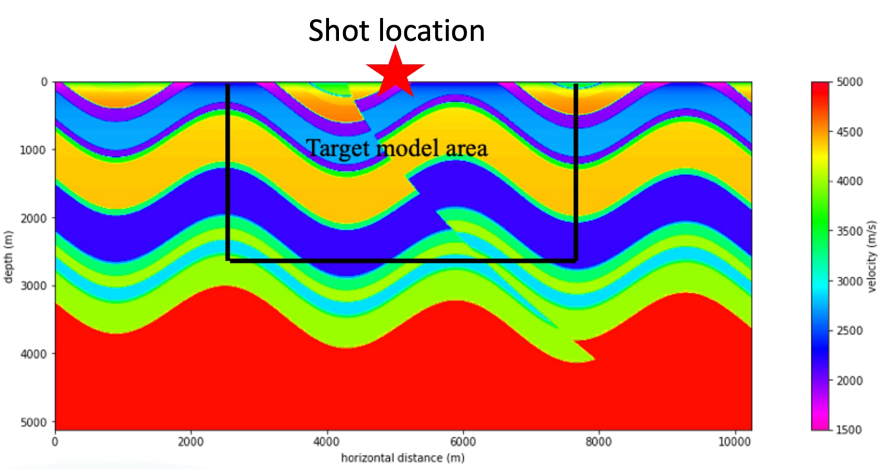
We show two 2D land data examples: a simple synthetic dataset and a field dataset from the Canadian foothills. For both datasets, we followed the same workflow. We first built representative training velocity models. Next, using finite-difference acoustic modeling, we created synthetic seismic data. The seismic data (as input) and velocity models (as target) were then used to train the CNN. Finally, the trained CNN was used to predict velocity models from test or field data.

Synthetic Example – Data Overview

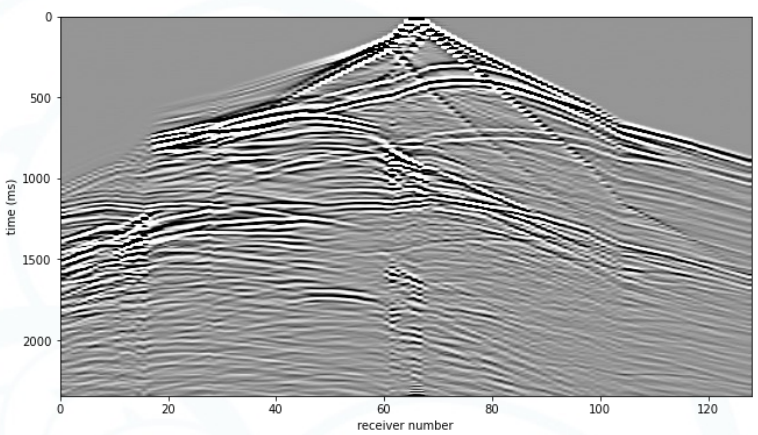
The velocity models for the synthetic example were created automatically, containing between 10 and 40 layers of varying thickness and dip. The layers were sinusoidal in shape, with varying period and amplitude. Layer velocities varied randomly from 2000-5000m/s. Each model contained a single listric thrust fault with varying orientation and dip. The velocities were reduced in the shallow to account for weathering and reduced lithostatic load.

3000 models were created: 2780 for training, 200 for training validation, and 20 test models that were kept separate to evaluate the CNN after training was completed. One of the training models is shown below. The CNN was trained to predict the target model area, about 5km wide and 2.5km deep. The rest of the model is to provide aperture for the modeling.

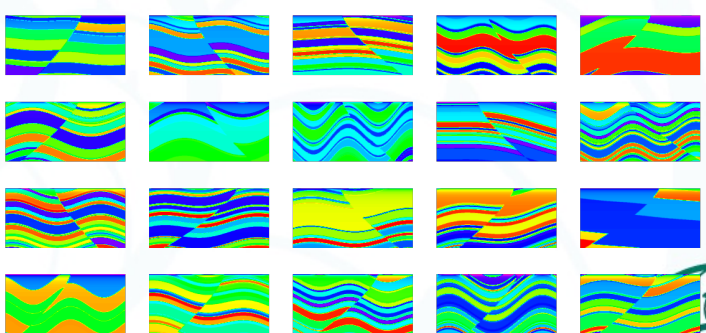
The seismic shot records were modeled using finite difference acoustic modeling using a 25Hz Ricker wavelet. The geometry consisted of 64 shots spaced evenly at 80m across the target model area, with each shot containing 128 receivers spaced at 40m. This is sparse geometry, especially given the relatively shallow model, providing challenging data for the CNN.



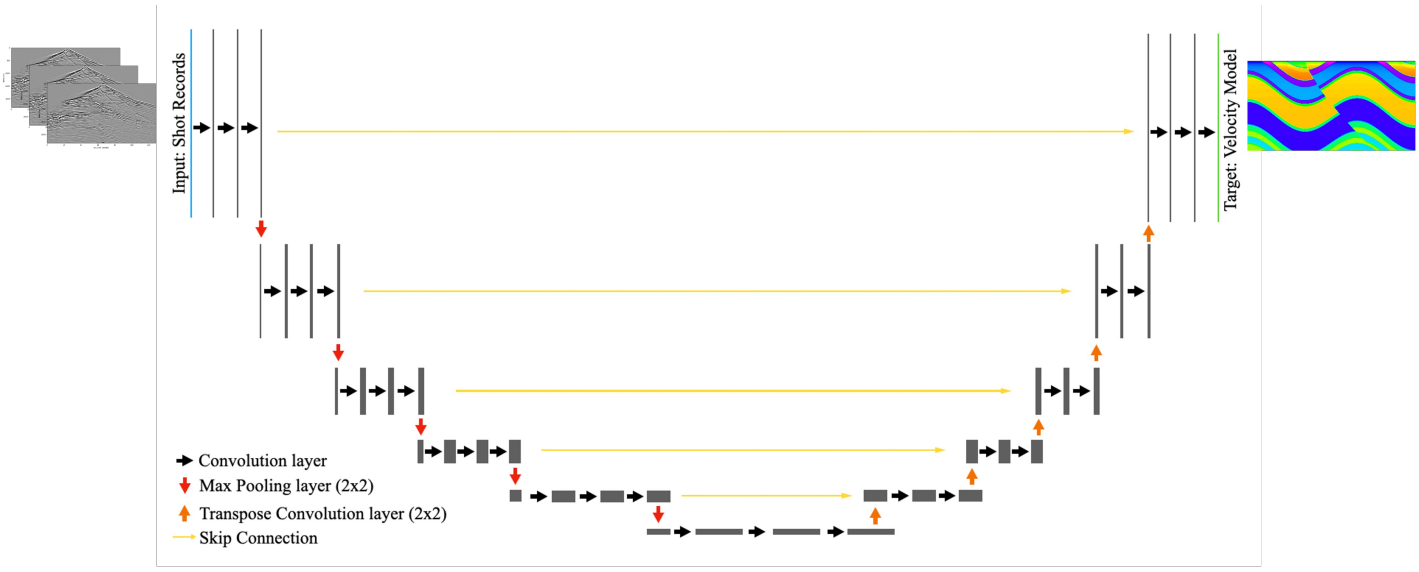
A synthetic shot record from the centre of this model is shown on the right. Note the complexity of the reflectivity on the left side of the the shot record, where the wavefield crosses the fault. It would be very difficult to pick first arrivals on this shot, resulting in poor refraction statics. Also, with the 25 Hz Ricker wavelet we have about 12-50Hz bandwidth. The lack of low frequencies would make FWI difficult to implement without a very good starting model.



More training models are shown on the right to give an example of diversity.



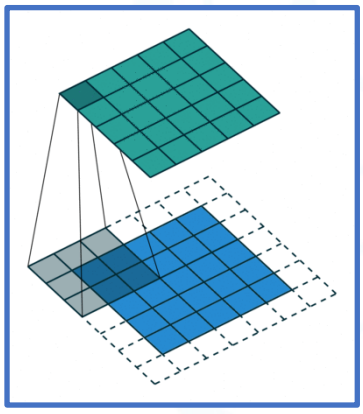
Convolutional Neural Network



The CNN is shown in the schematic above. It follows a U-net architecture, with shot records input on the left and target velocity models output on the right. The rectangles represent the data as it travels through the network. The black right arrows represent 3x3 convolutional layers, the red down arrows represent max pooling layers, and the orange up arrows represent transpose convolutional layers. There are also skip connections, shown in yellow, that allow higher resolution features to skip part of the network.

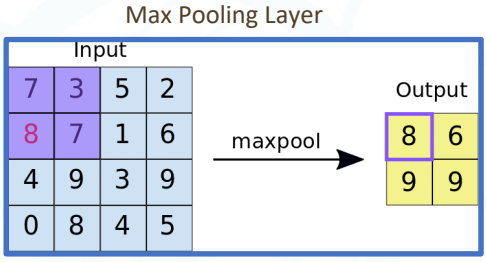
The animations below describe the main layer types of the CNN. The 3x3 convolutional layer convolves the blue input to the green output, maintaining the same shape by padding the input. The max pooling layer reduces the dimensionality of the data by outputting the maximum value of each 2x2 patch of data. The transpose convolutional layer is similar to the convolutional layer but with padding between each row and column, increasing the dimensionality through convolution. The goal of the training process is to learn the convolutional weights that minimize the least square error between the predicted and target velocity models.

3x3 Convolutional Layer



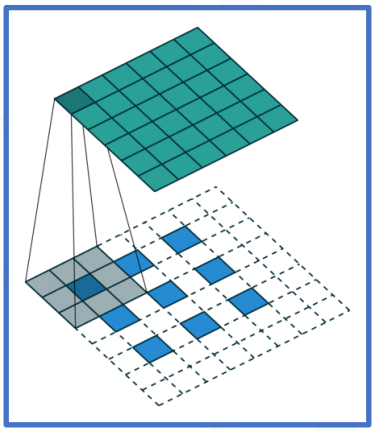
https://github.com/vdumoulin/conv_arithmetic

Max Pooling Layer



https://nico-curti.github.io/NumPyNet/NumPyNet/layers/maxpool_layer.html

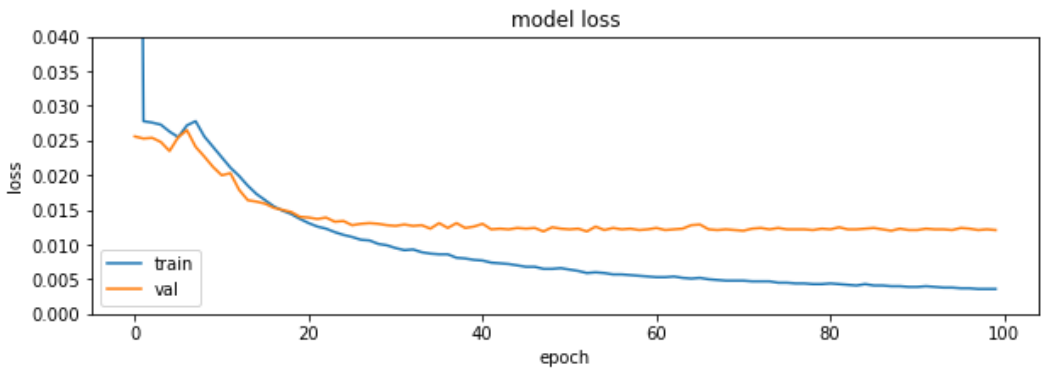
3x3 Transpose Convolutional Layer



https://github.com/vdumoulin/conv_arithmetic

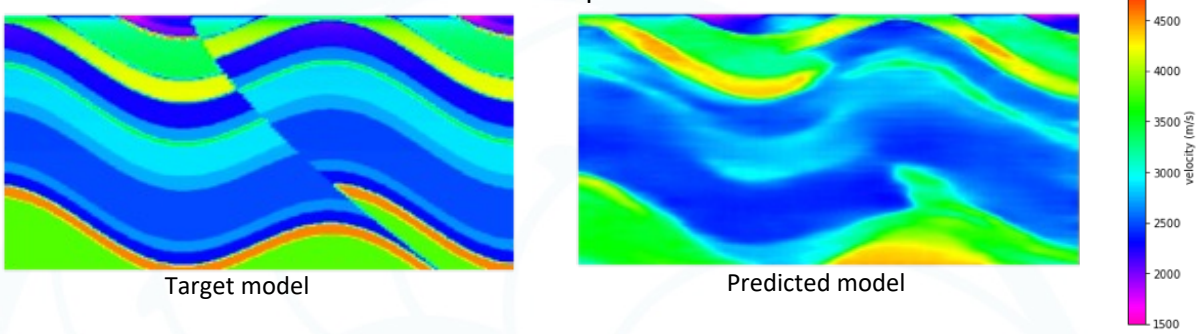
Synthetic Example - Results

The CNN was trained over 100 epochs, learning 188 million convolutional weights. The loss curves for the training are shown below. The blue curve is the training loss while the orange curve is the validation loss. The validation data were not involved in the training but were used to evaluate the CNN after each epoch. Both training and validation loss converge to a low error. However, the convergence of the validation data slows significantly after 20 epochs while the training data continues to converge. This suggests that there is some overfitting, where features learned on the training dataset don't generalize to other models. The amount of overfitting could be reduced with more training data or by improving the network.



The images below show the results of the CNN on a test dataset. This test example was not used in the training process. The target model is shown on the left for comparison. While the predicted model is not perfect, the major velocity boundaries are correctly identified and the fault location is accurate.

Test example



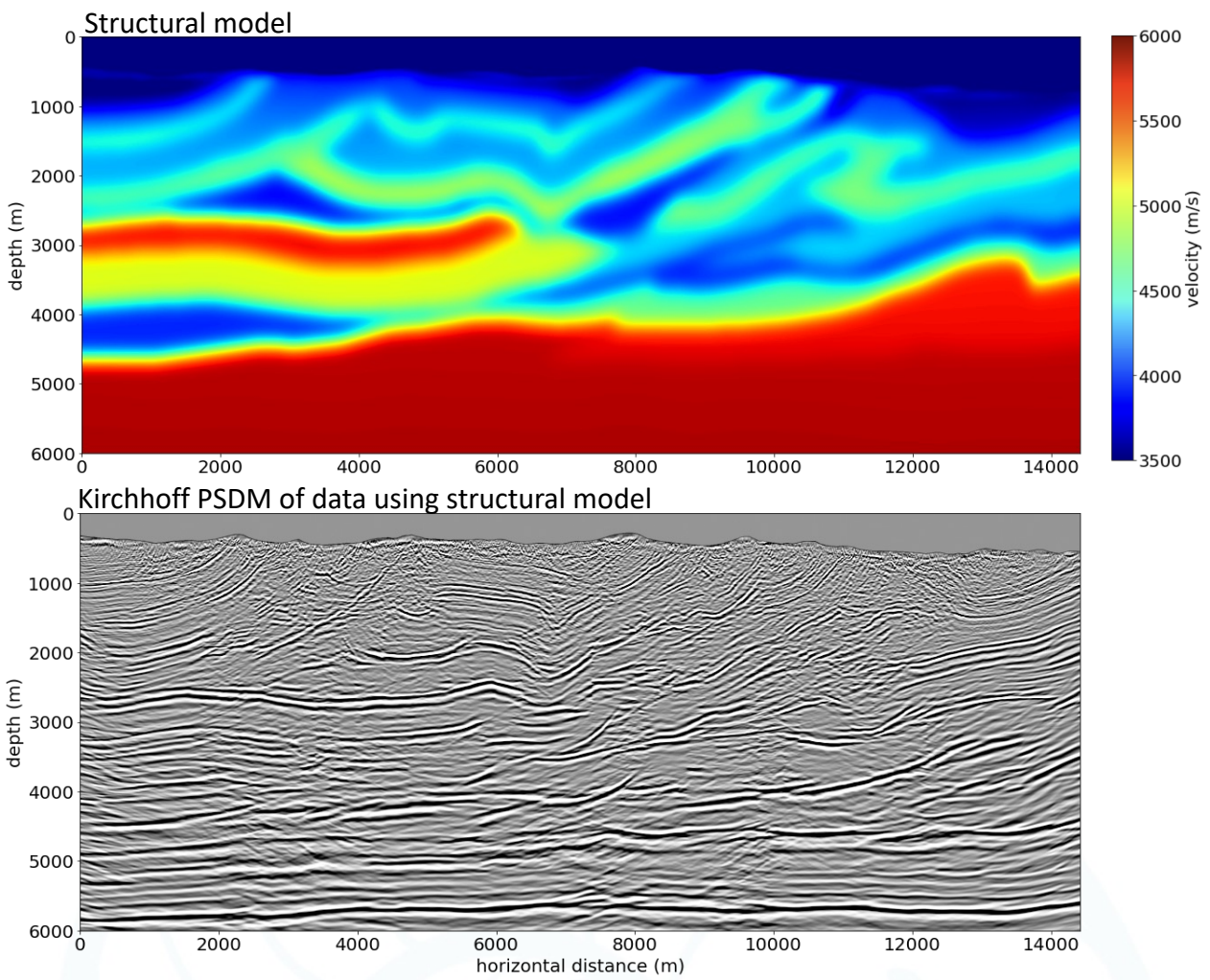
Target model

Predicted model

We could have spent more time refining this example. However, the results were compelling enough that we wanted to try this on a field data example, which we describe on the following slides.

Field Data Example

This field data example is a 2D line from the Canadian Foothills. We have processed this line previously using a structural model building approach, updating the model over several iterations using interpretive guidance and seismic diagnostics. Our best model and the corresponding Kirchhoff PSDM are shown below. The goal of this test is to predict a velocity model using the CNN and benchmark both the model and PSDM against these structural model building results.

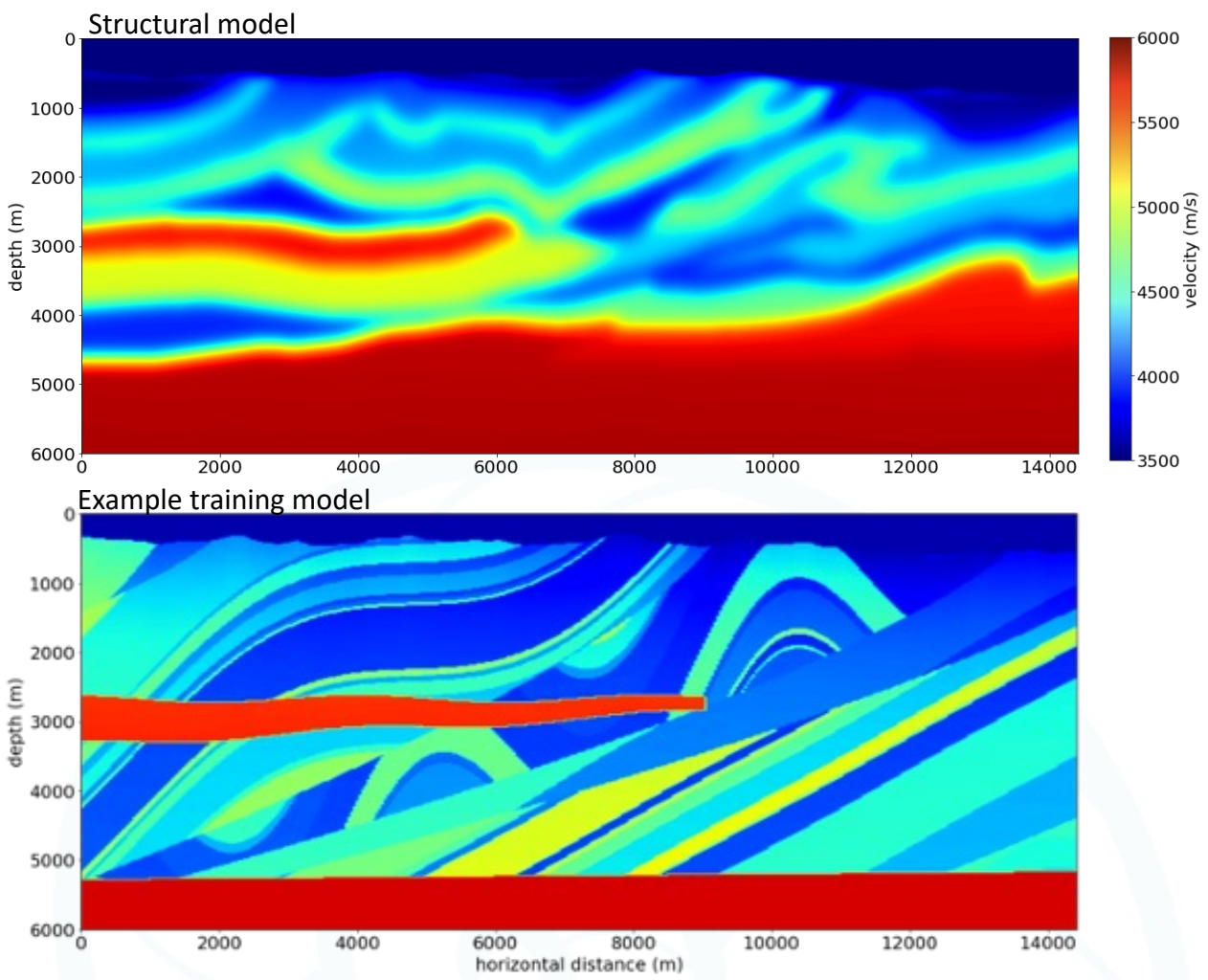


The acquisition of this line was sparse with 142 shots at 100m spacing and 736 receivers at 20m spacing. There were 300 channels per shot, providing a maximum offset of 3000m and a nominal cdp fold of 30.

The Canadian foothills consists of old, high-velocity rock with clastic rocks having a velocity of about 4000m/s, carbonates about 5200m/s, and crystalline basement about 6000m/s.

Field Data Example – Representative Training Data

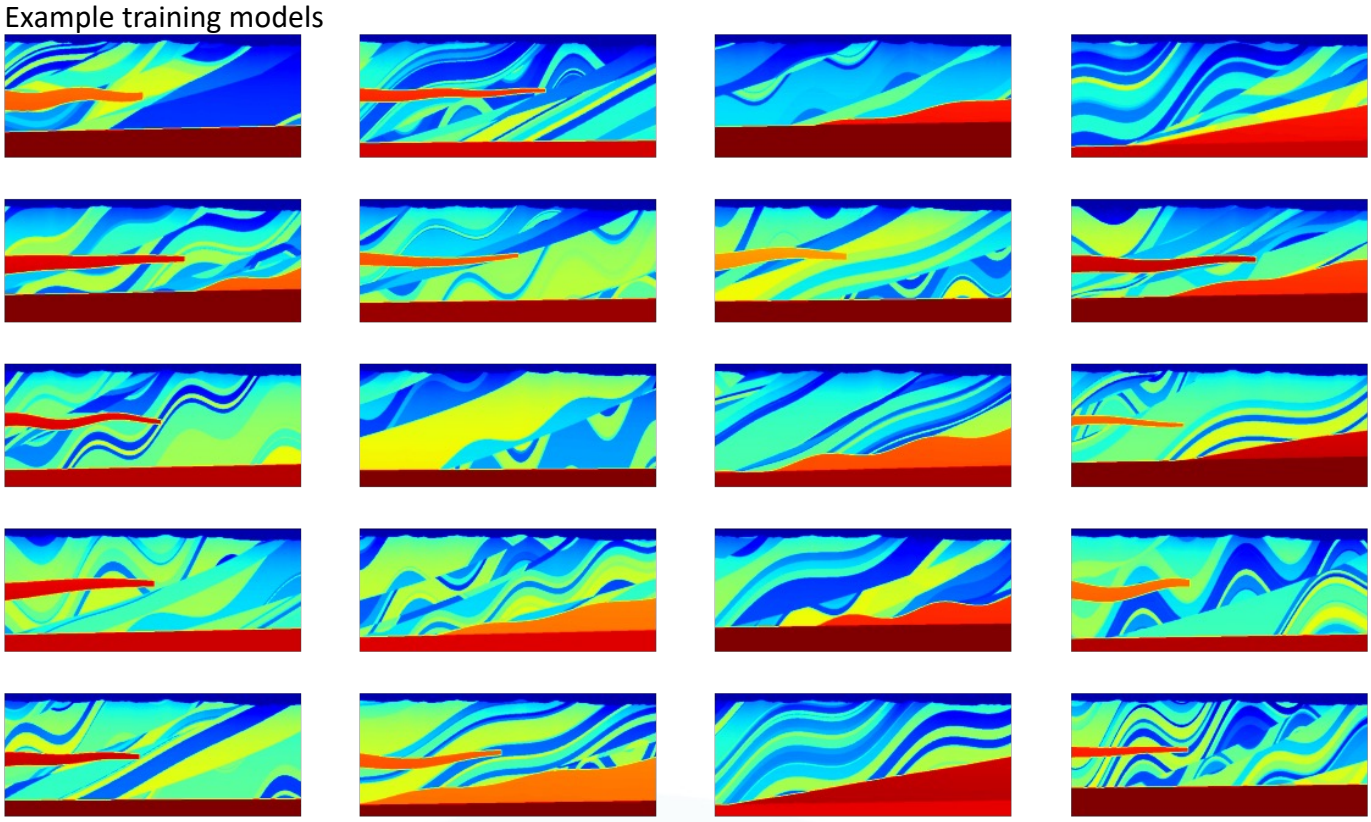
As with the synthetic example, the first step in this test was to build the training velocity models. To ensure optimal training of the CNN, it was important that the training models were representative of what we expect in the field. This is where we add human intelligence into the deep learning process. By constraining the training data to a reasonable range of velocities, dips, and faults, we provided the CNN with a focused set of training data. The images below compare our structural model to one of the training models. Note that while this training model does not look geologic, it contains similar velocities and dips to the structural model.



Field Data Example - Modelling

We created 4800 training samples, 100 validation samples, and several hundred test samples. The images below show the variety of our training samples.

Example training models

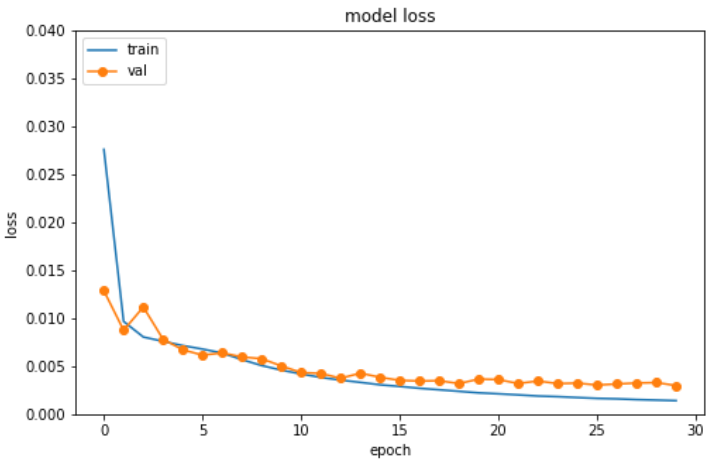


The models used in this test were more complicated than the synthetic example in several ways. We incorporated topography, with the sources and receivers located at the same location and elevation as the field data. The models were also anisotropic (TTI). We used constant values of epsilon (0.12) and delta (0.03) while dips were computed parallel to bedding. The training velocity models were created with up to 5 listric faults with different geology across the faults. Some models included a carbonate layer on the left of the model at about 3 km depth and/or on the right of the model above the basement.

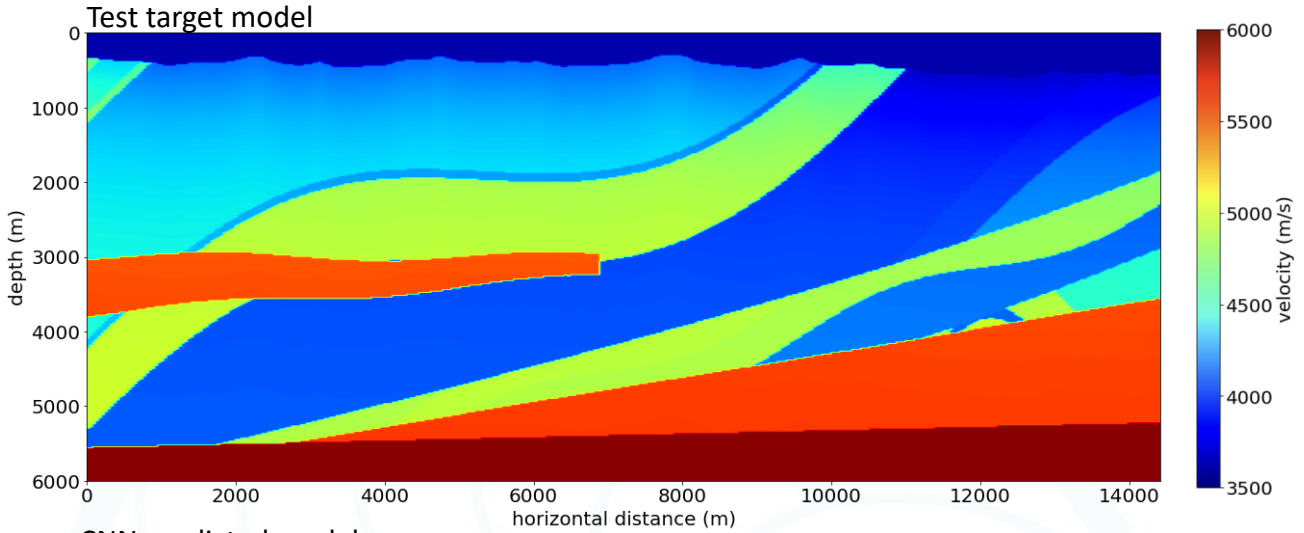
Field Data Example – Results Synthetic

We used the same CNN architecture as with our first example. The CNN was trained over 30 epochs. The loss curves are shown on the right. Both training and validation losses converge to low error with only a small amount of overfitting.

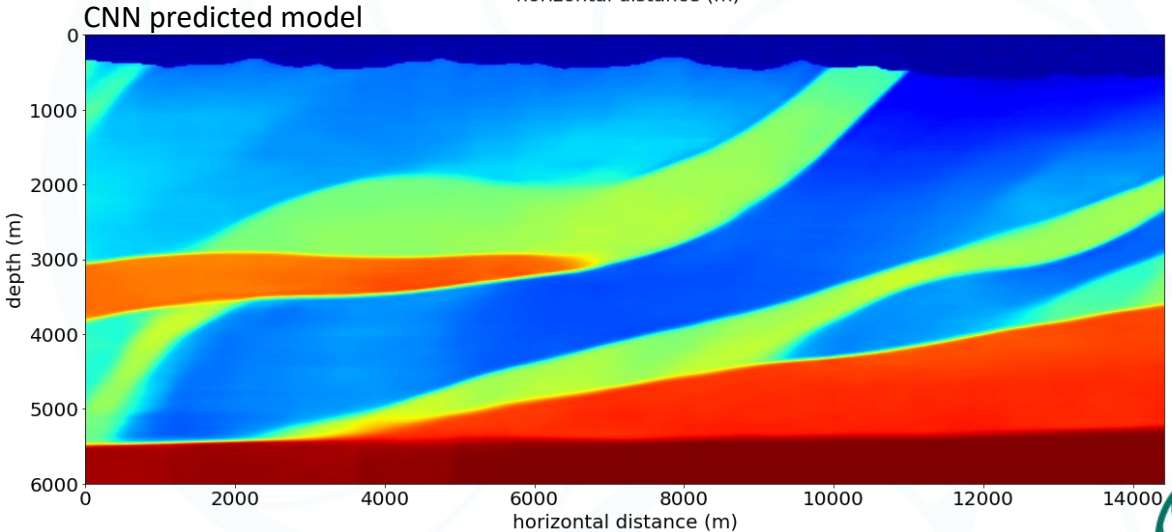
We evaluated our CNN on several of our test samples. The images below compare a test target model to the CNN predicted model. In this case the predicted model is of good quality with major velocity boundaries clearly defined and velocities close to the true values.



Test target model



CNN predicted model

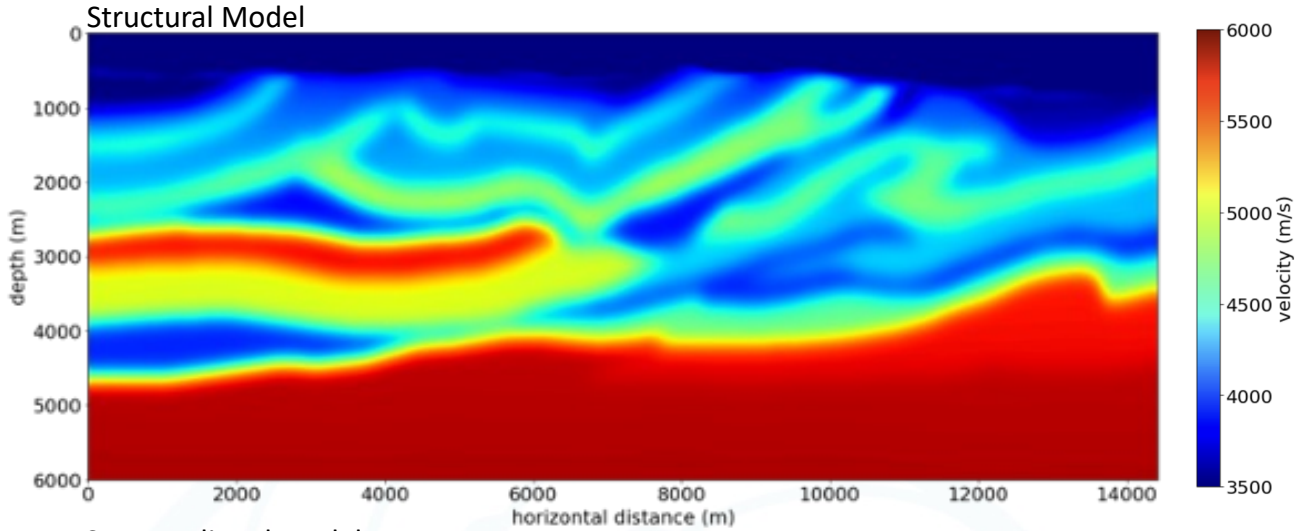


Field Data Example – Results Model

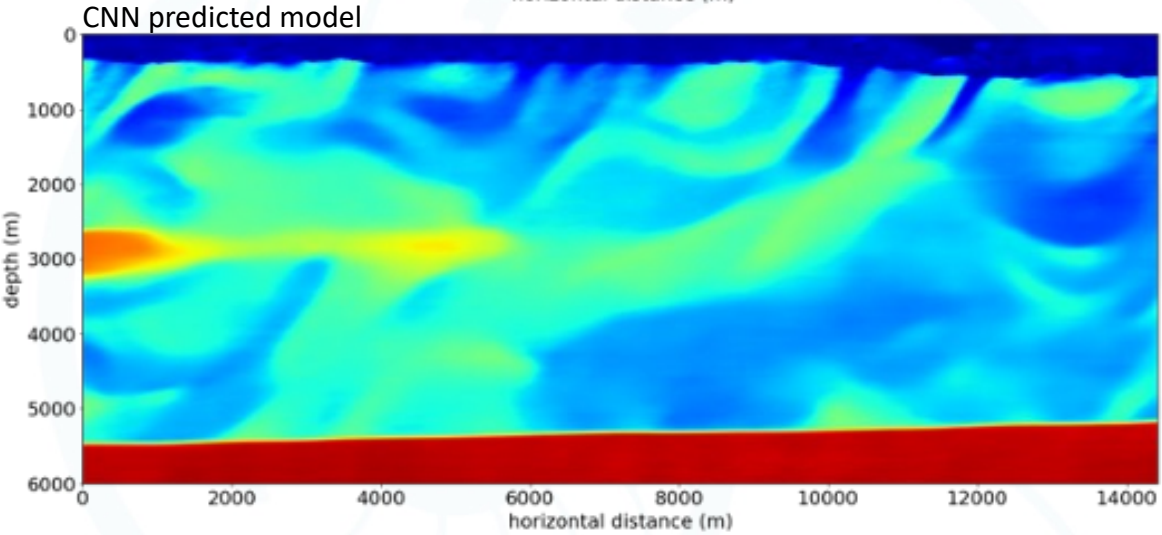
Next, we evaluated the CNN on the field data. The results below compare our structural model with the model predicted by the CNN. The structural model is not the ground truth; it is simply a model we built to optimize the imaging and is presented here for comparison purposes.

The predicted model identifies many similar features as the structural model with some noticeable differences. The overall velocity range is similar. The CNN does not predict the carbonate layer above the basement. The near-surface features differ with steeper dips and more detail predicted on the CNN model.

Structural Model



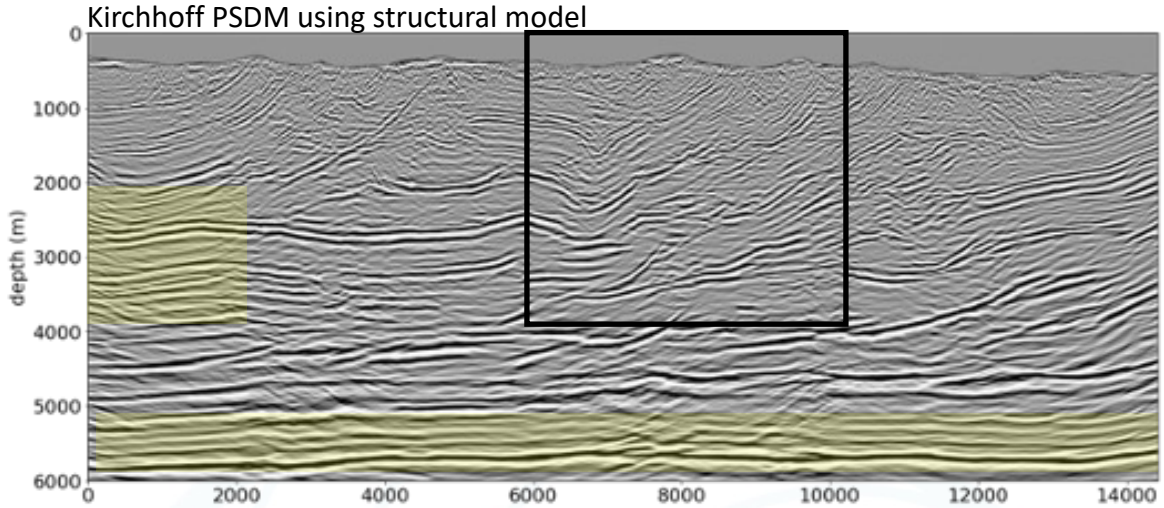
CNN predicted model



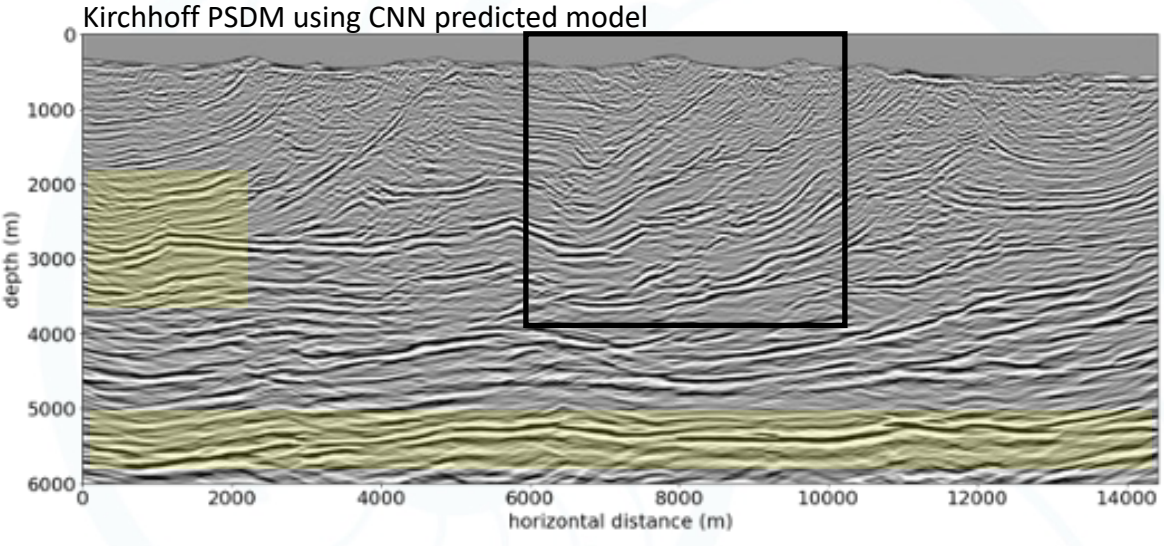
Field Data Example – Results PSDM

To further evaluate the prediction, we migrated the data with the two models shown on the previous slide. The results are shown below. Both images show similar structures. There are a few areas shown in yellow where the structural model produces a much better image. We expect the basement to be a planar reflector, with a slight dip to the west. This is much better imaged using the structural model. The poor basement imaging on the CNN model is largely due to the missing deep carbonate layer. Above the basement, the quality of the deep imaging is similar between the two migrations. On the left, there is a velocity anomaly on the predicted model that causes an imaging artifact. However, the shallow imaging shown in the black box is improved on the CNN model, as shown on the following slide.

Kirchhoff PSDM using structural model



Kirchhoff PSDM using CNN predicted model

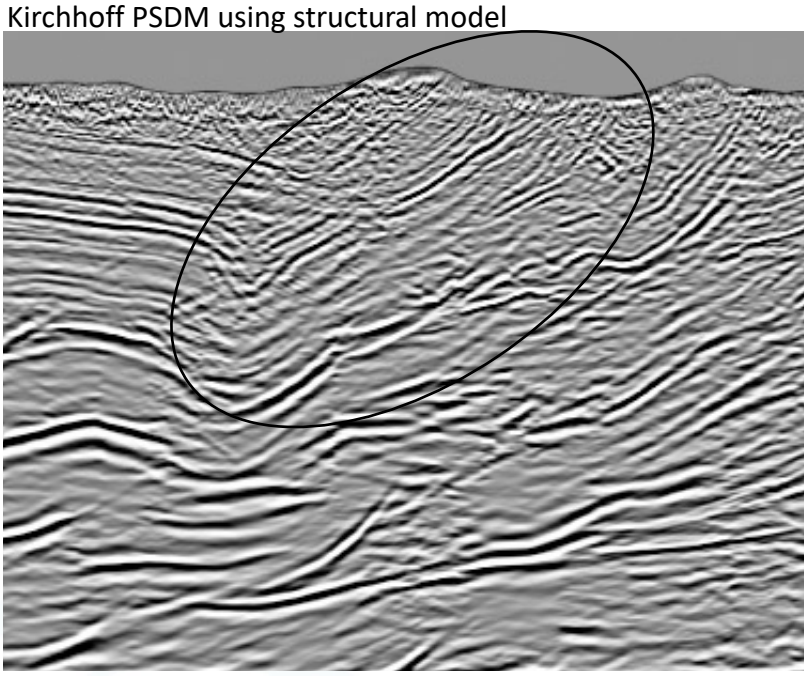


Field Data Example – Results PSDM (zoom)

The zoomed images below show areas of improved imaging on the CNN predicted model, particularly the synclines and reflectors shown in ovals. The improved reflector continuity and interpretability gives us confidence that the CNN is predicting a reasonable shallow model. This improved imaging is despite a much simpler processing workflow, as outlined below.

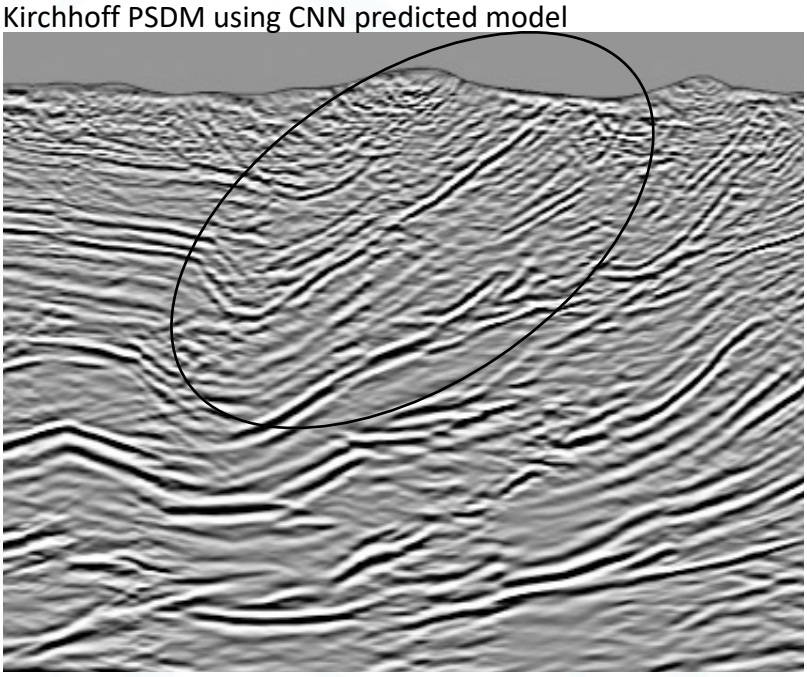
- Workflow
  - First arrival tomography for shallow model
  - Structural model for deep
  - Many (10+) model iterations
  - Reflection statics

Kirchhoff PSDM using structural model



- Workflow
  - CNN model building

Kirchhoff PSDM using CNN predicted model



Conclusions

We have shown that a Convolutional Neural Network can be trained to predict PSDM velocity models from shot records. The training data need to be representative and sufficient. It is important to use any available geological constraints to provide a focused set of training data to the CNN.

The synthetic data showed good results, with clear velocity boundaries and accurate velocities. The field data showed some promising results. There were some problems with the model in the deep and on the edges. However, the shallow model produced better imaging in places than the mature structural model, despite a much simpler workflow.

Moving forward, we plan to spend some time investigating more advanced and more efficient forward-modeling methods. We also see an opportunity to refine the CNN architecture. We plan to test this method on several more field data examples to establish a robust workflow.

References

- Faust, L.Y. [1951] Seismic velocity as a function of depth and geologic time, *Geophysics,* **16,** 192-206.
- Kingma, D. and Ba, J. [2014] Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*
- Krizhevsky, A., Sutskever, I. and Hinton, G. [2012] ImageNet Classification with Deep Convolutional Neural Networks, *Neural Information Processing Systems,* **25**, 10.1145/3065386.
- Ronneberger, O., Fischer, P. and Brox, T. [2015] U-Net: Convolutional net- works for biomedical image segmentation, *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234– 241.
- Yang, F. and Ma, J. [2019] Deep-learning inversion: A next-generation seismic velocity model building method, *Geophysics*, **84,** 583-599