Deblending using convolutional neural networks

Zhan Niu* and Daniel Trad

December 11, 2019

NSERC CRSNG
UNIVERSITY OF CALGARY
FACULTY OF SCIENCE
Department of Geoscience

Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

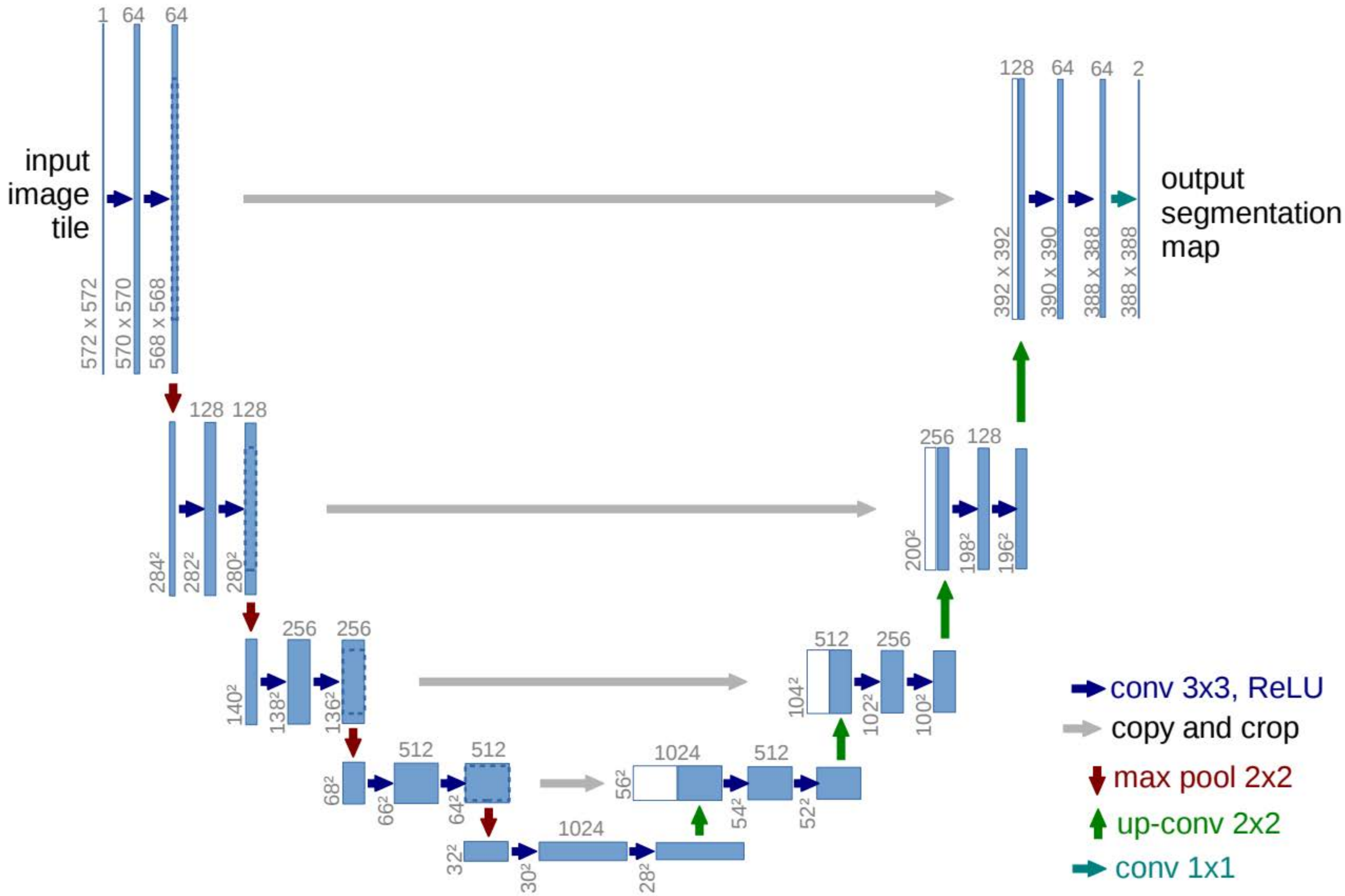
Motivation

- Programming level:
  - Modern GPUs’ high performance of parallelization
  - The new booming techniques in machine learning
- Geophysics level:
  - Deblending requires removing signal that is incoherent

Content

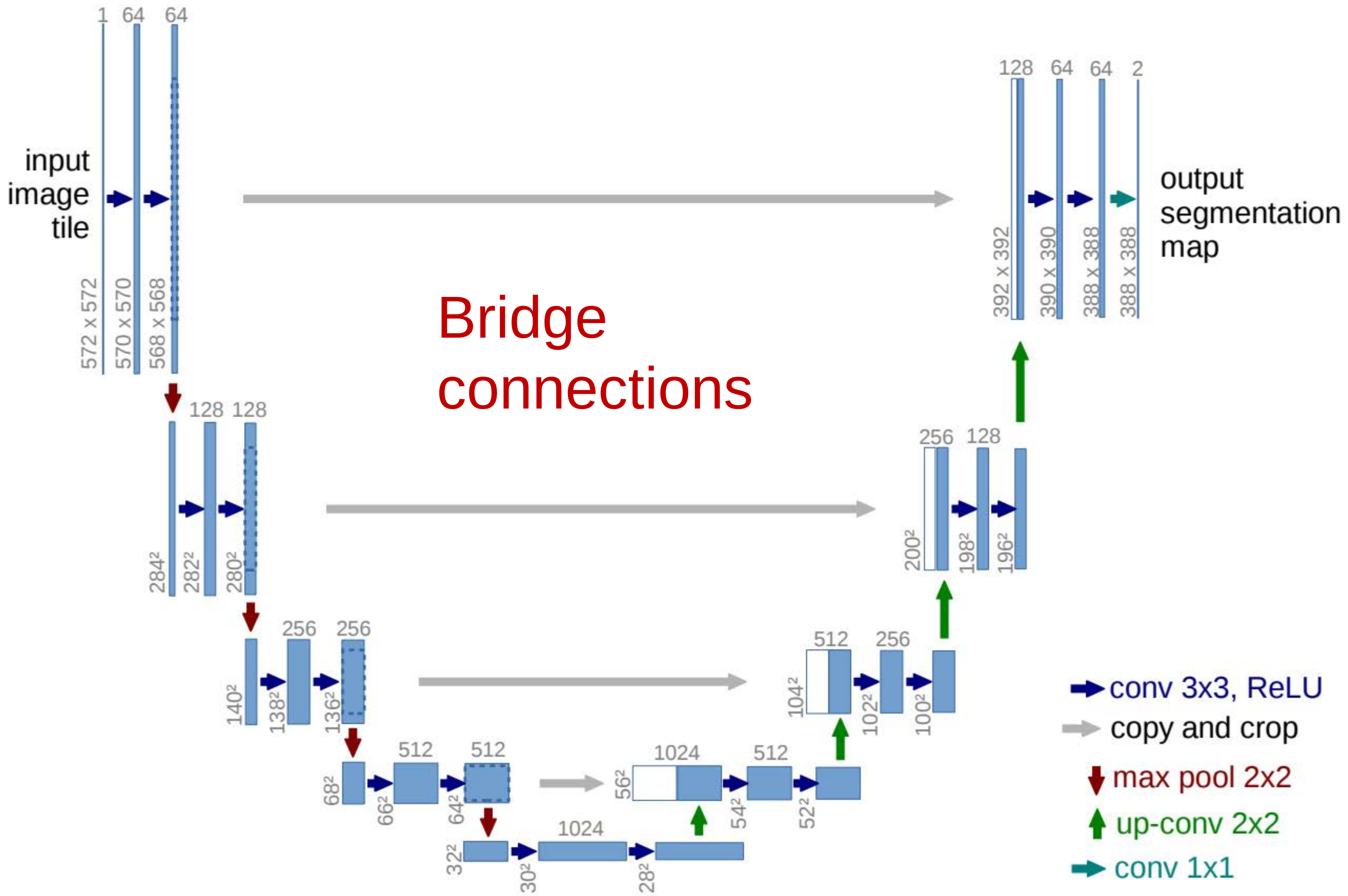
- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

Theory – U-Net

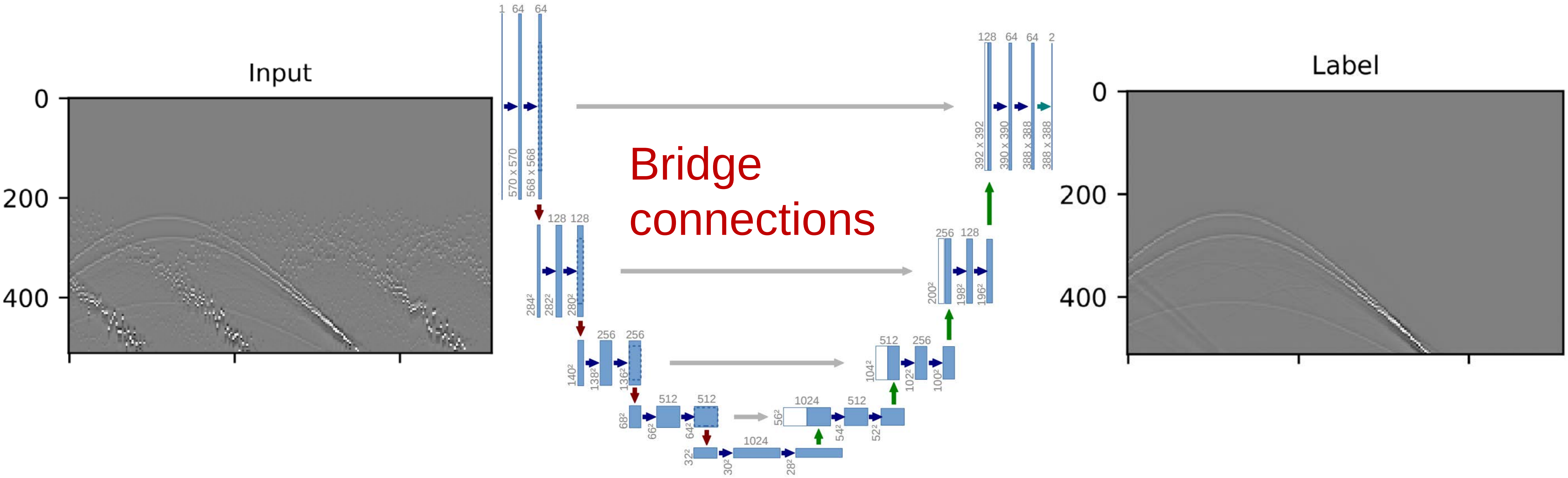


Modified from Ronneberger et al. (2015)

Theory – U-Net

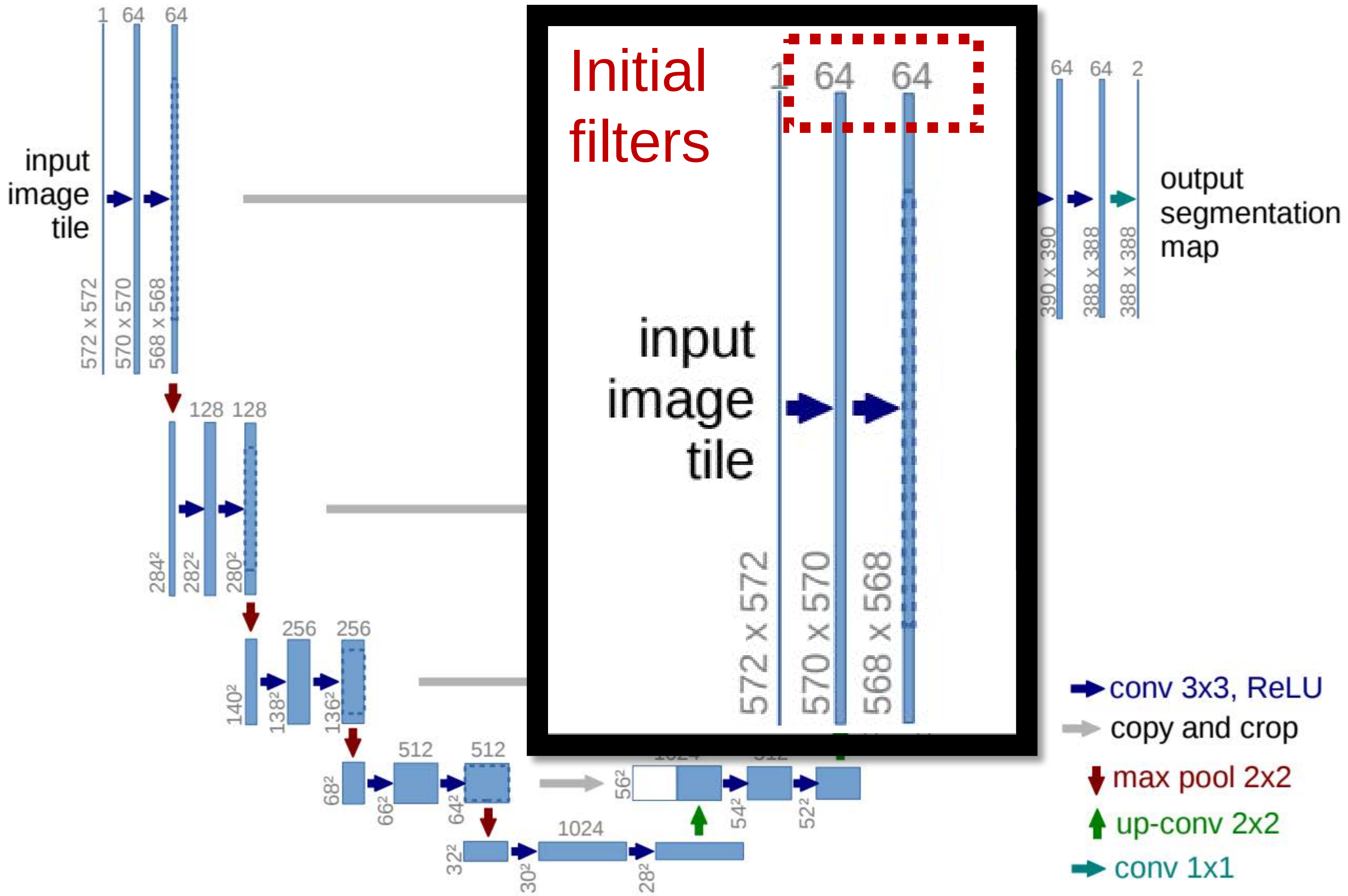


Modified from Ronneberger et al. (2015)



Modified from Ronneberger et al. (2015)

Theory – U-Net



Modified from Ronneberger et al. (2015)

Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/ target definition
  - Validation
  - Test
- Conclusion

Theory – Workflow

Entire dataset

- Training dataset (80%)
  - Directly used for model update
- Validation dataset (20%)
  - Assist on selecting the best model
  - Metric

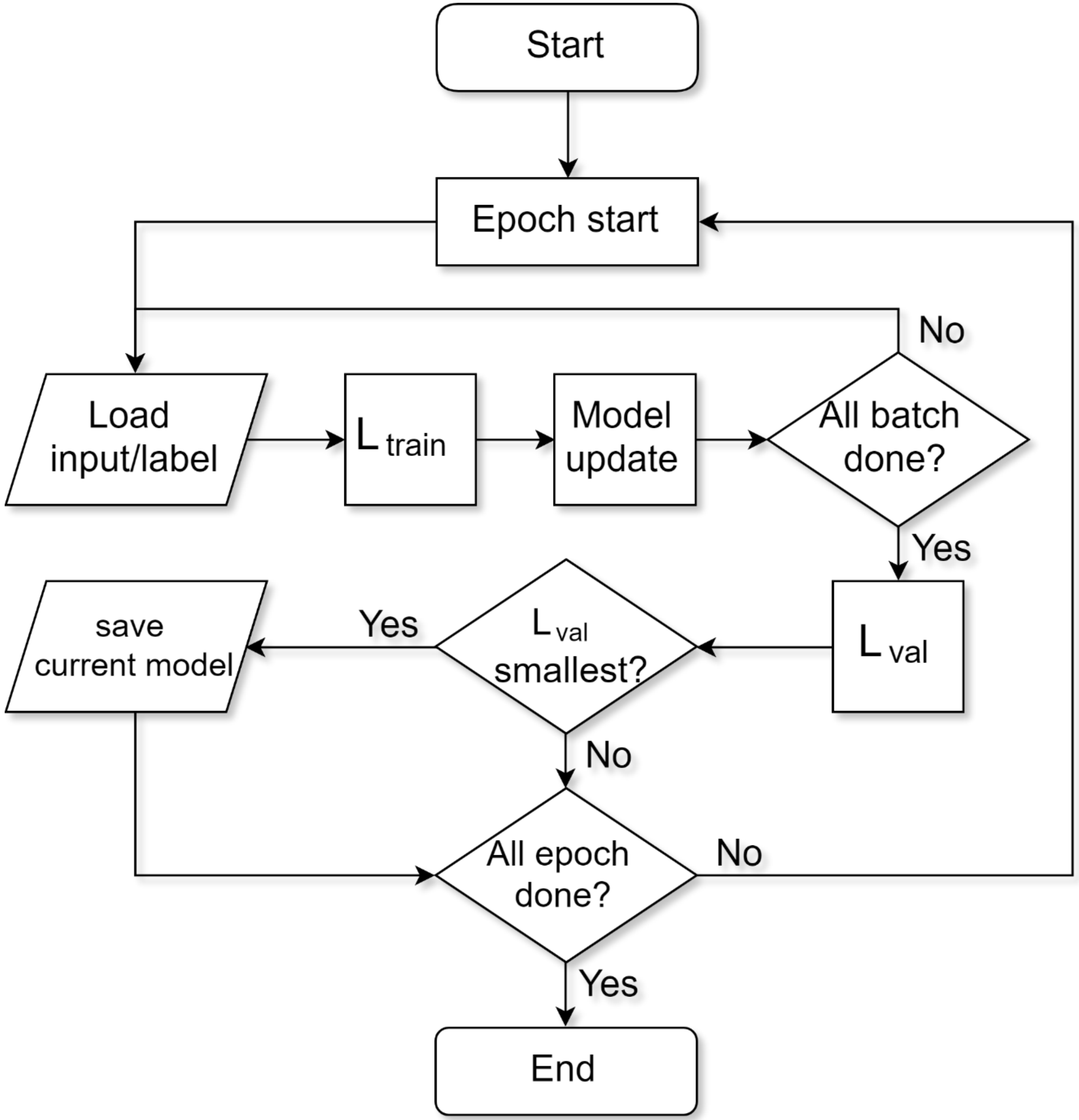
The loss function – Mean Square Error (L2 square)

$$L = \text{mean}\left(\left\|Y - Y_{pred}\right\|_2^2\right)$$

The optimizer – ADAM

$$\boldsymbol{v}_i = \beta_1 \boldsymbol{v}_{i-1} + (1 - \beta_1)\boldsymbol{g}_i$$
$$\boldsymbol{s}_i = \beta_2 \boldsymbol{s}_{i-1} + (1 - \beta_2)\boldsymbol{g}_i^2$$
$$\widehat{\boldsymbol{v}}_i = \frac{\boldsymbol{v}_i}{1 - \beta_1^i}, \widehat{\boldsymbol{s}}_i = \frac{\boldsymbol{s}_i}{1 - \beta_2^i}$$
$$\boldsymbol{h}_i = \boldsymbol{h}_{i-1} - \alpha \frac{\widehat{\boldsymbol{v}}_i}{\sqrt{\widehat{\boldsymbol{s}}_i} + \epsilon}$$

Theory – Workflow

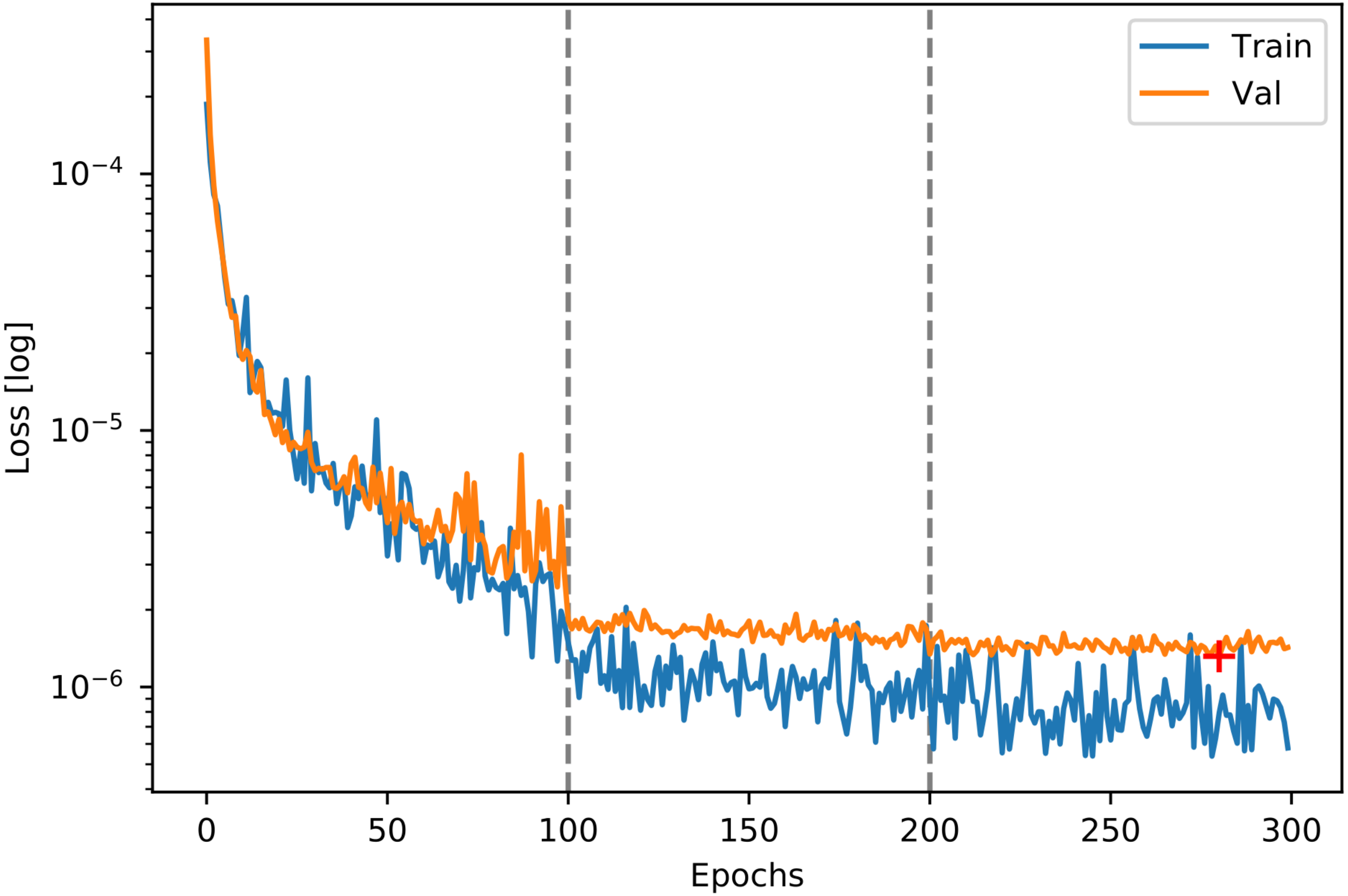


Algorithm 2 Training workflow.

Require: $\mathcal{L}(\cdot)$, model$(\cdot)$, optim$(\cdot)$

- **for** each epoch **do**
  - **for** each minibatch **do**
    - zero the gradients
    - load X and Y
    - $Y_{\text{pred}} \leftarrow \text{model}(X)$
    - $L \leftarrow \mathcal{L}(Y_{\text{pred}}, Y)$
    - $g \leftarrow BP(L)$
    - $\text{model}(\cdot) \leftarrow \text{model}(\cdot) + \text{optim}(g)$
  - $Y_{\text{val}} \leftarrow \text{model}(X_{\text{val}})$
  - $L_{\text{val}} \leftarrow \mathcal{L}(Y_{\text{val}}, Y)$
  - **if** L_{val} is the smallest **then**
    - save the model$(\cdot)$

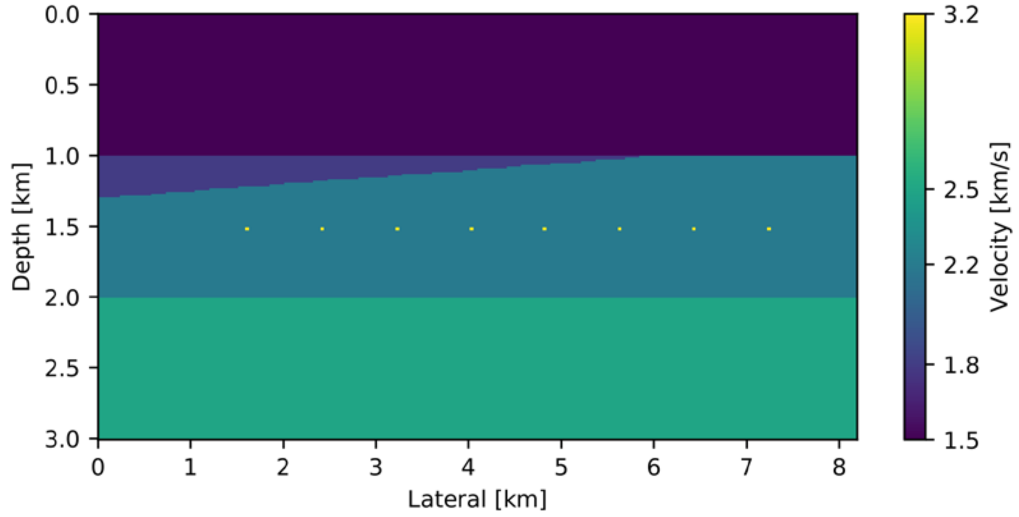
Theory – Workflow

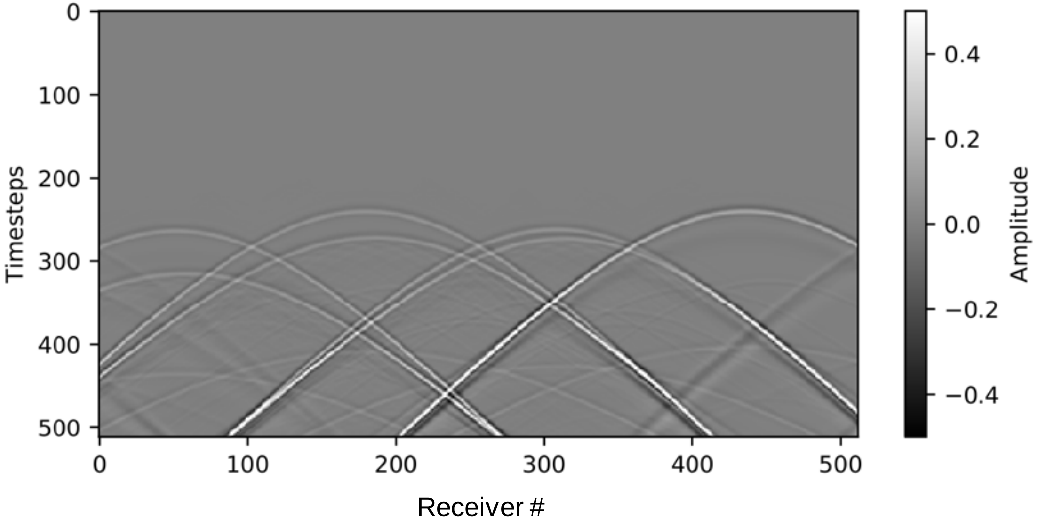


Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

Results – Input/target definition



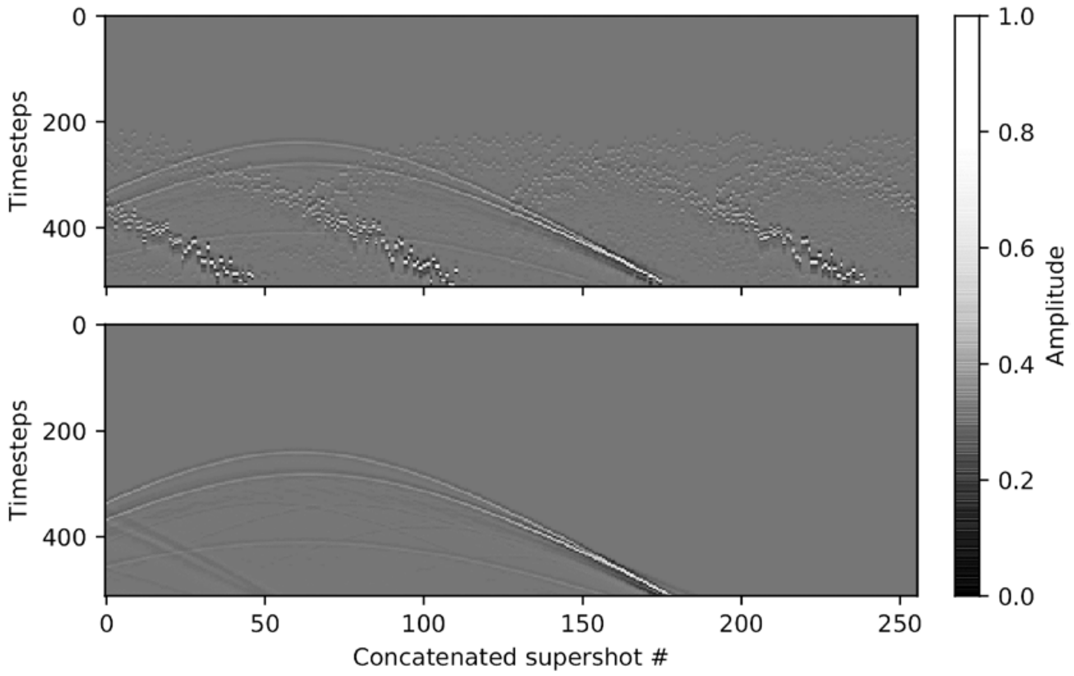


Seismic (pseudo-blended)

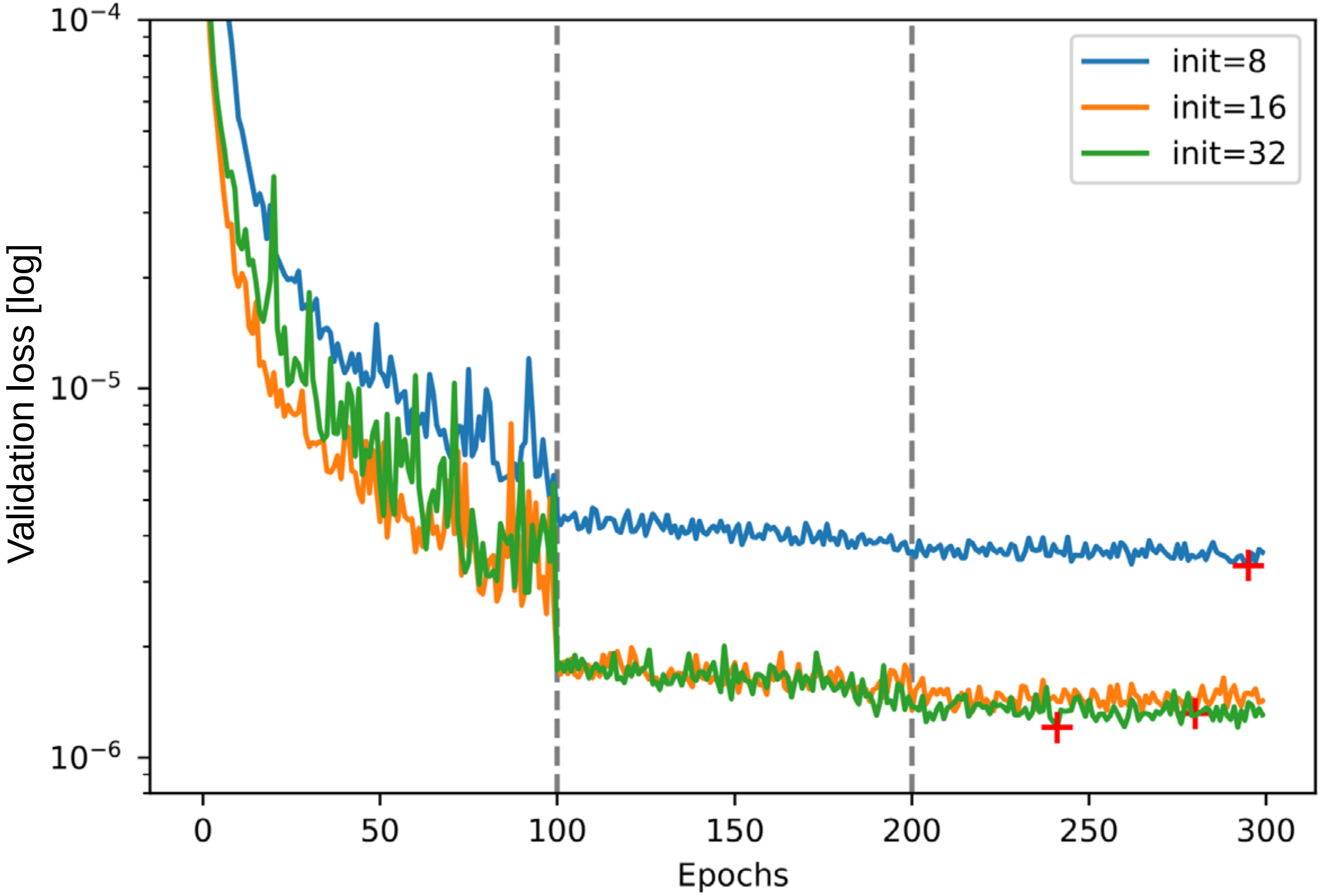
64 supershots × 4 blended
512 receivers
512 timesteps

Machine learning

512 Samples
256 Width
512 Height

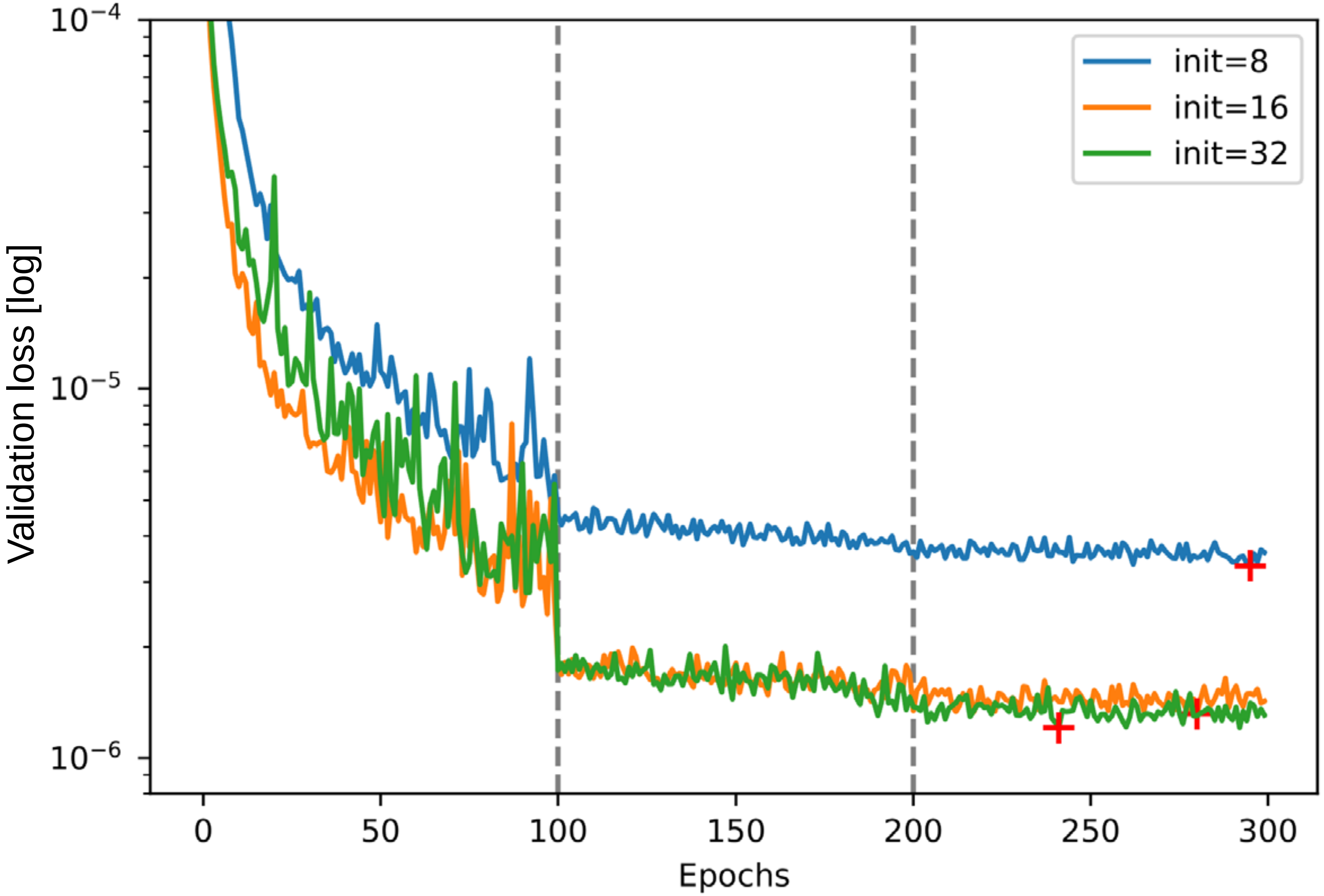


Results – Best initial filters

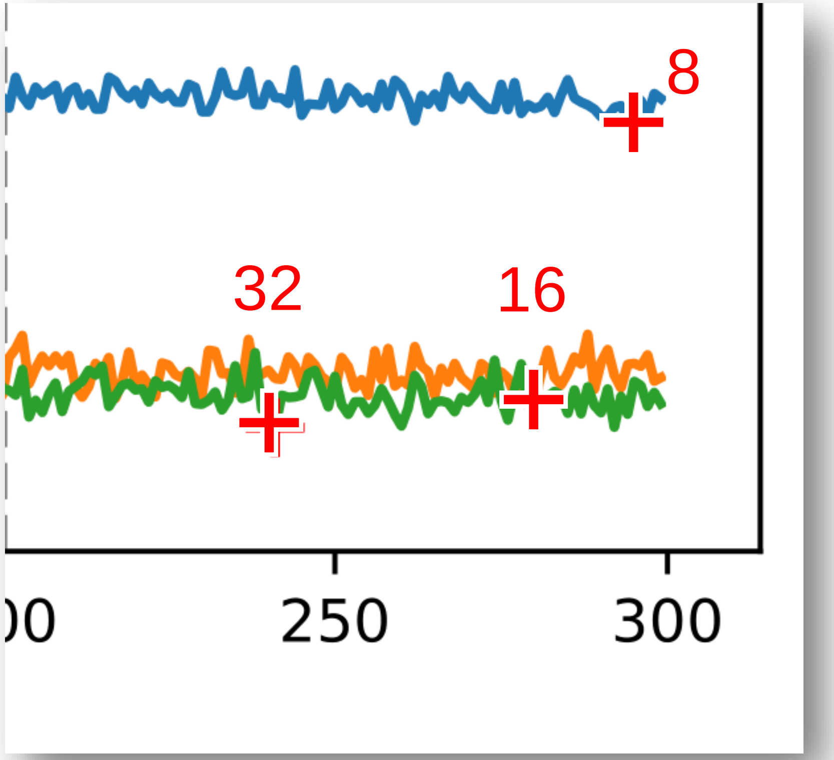


# filters	$\min(L_{\text{val}})$	Best epoch
8	3.321×10^{-6}	295
16	1.318×10^{-6}	280
32	1.207×10^{-6}	241

Results – Best initial filters



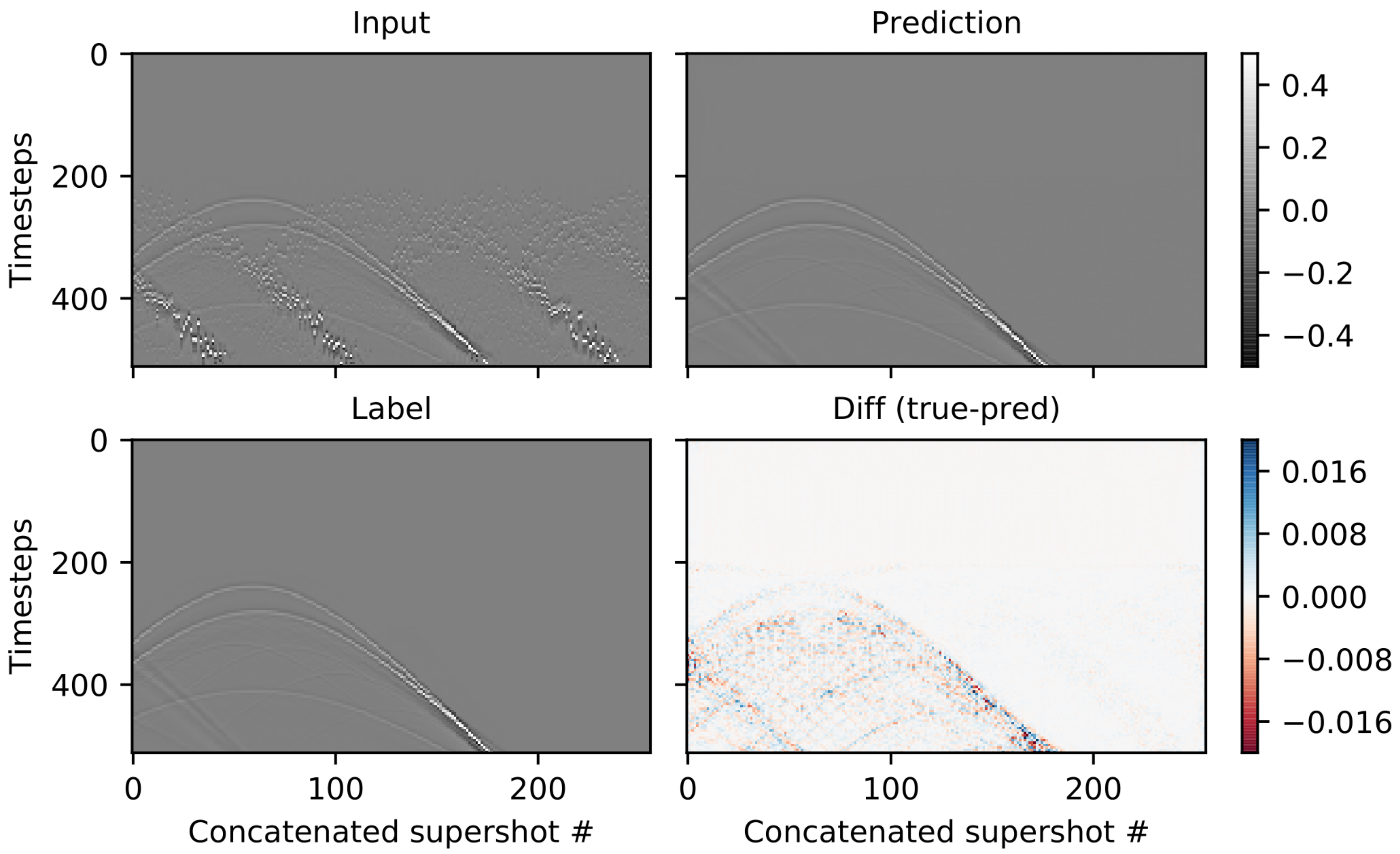
# filters	$\min(L_{\text{val}})$	Best epoch
8	3.321×10^{-6}	295
16	1.318×10^{-6}	280
32	1.207×10^{-6}	241



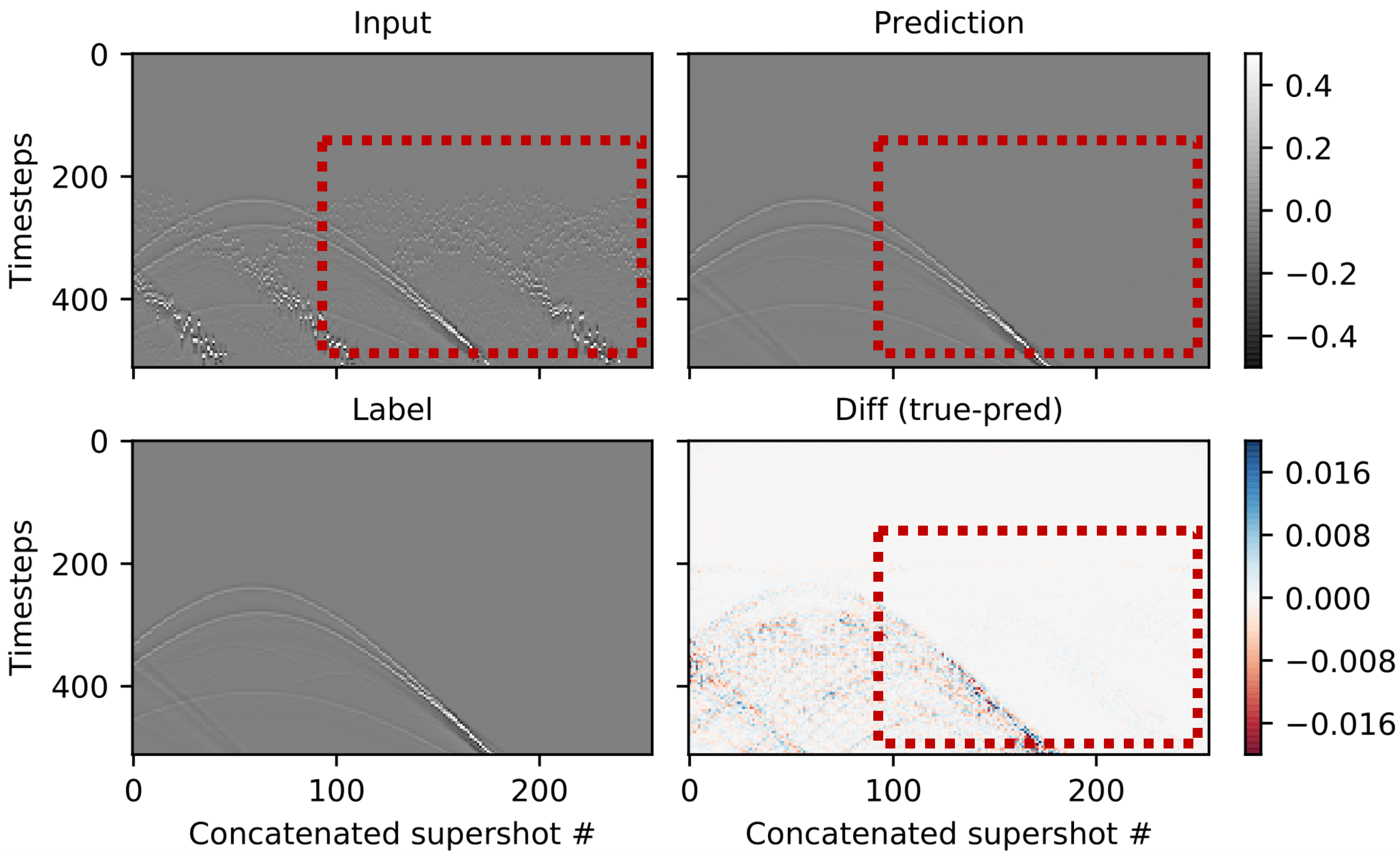
Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

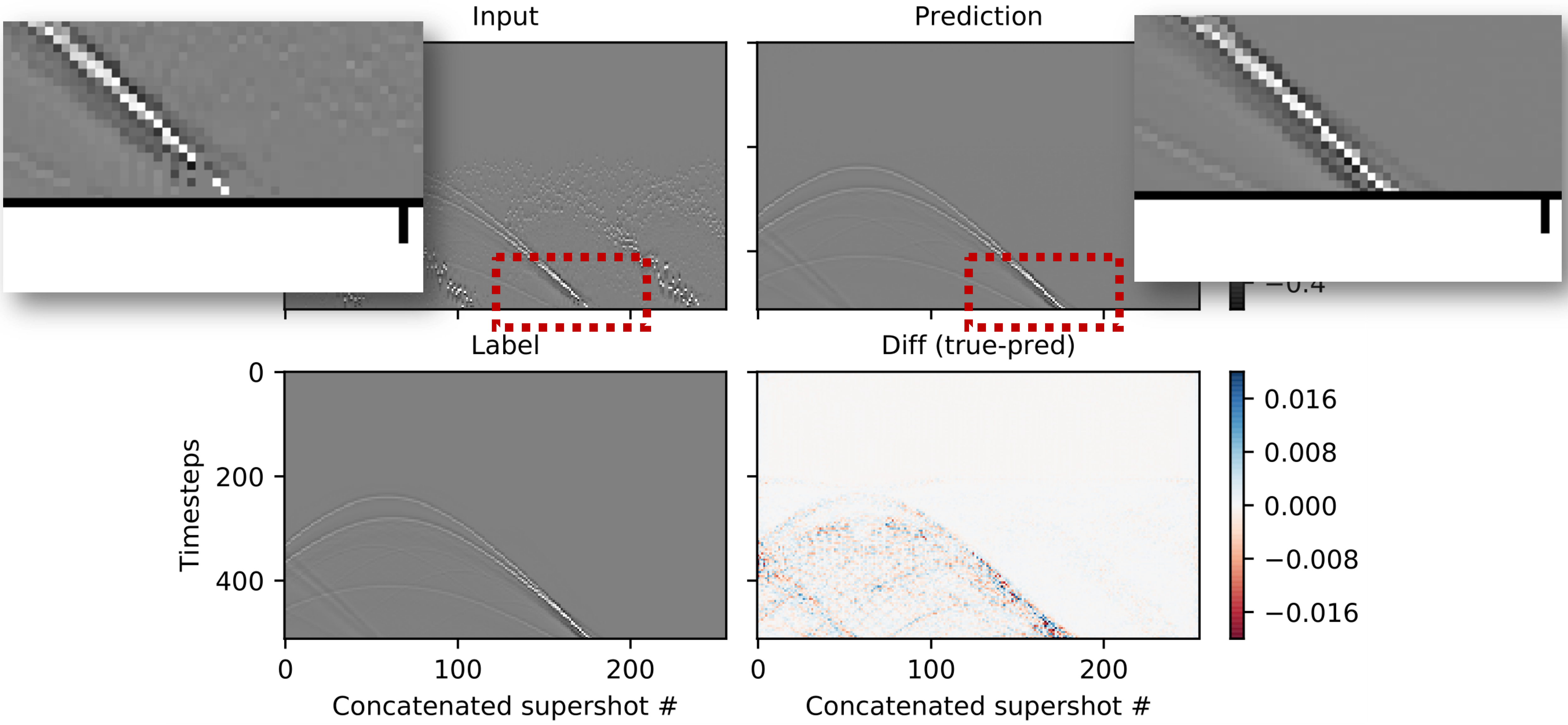
Results – Validation



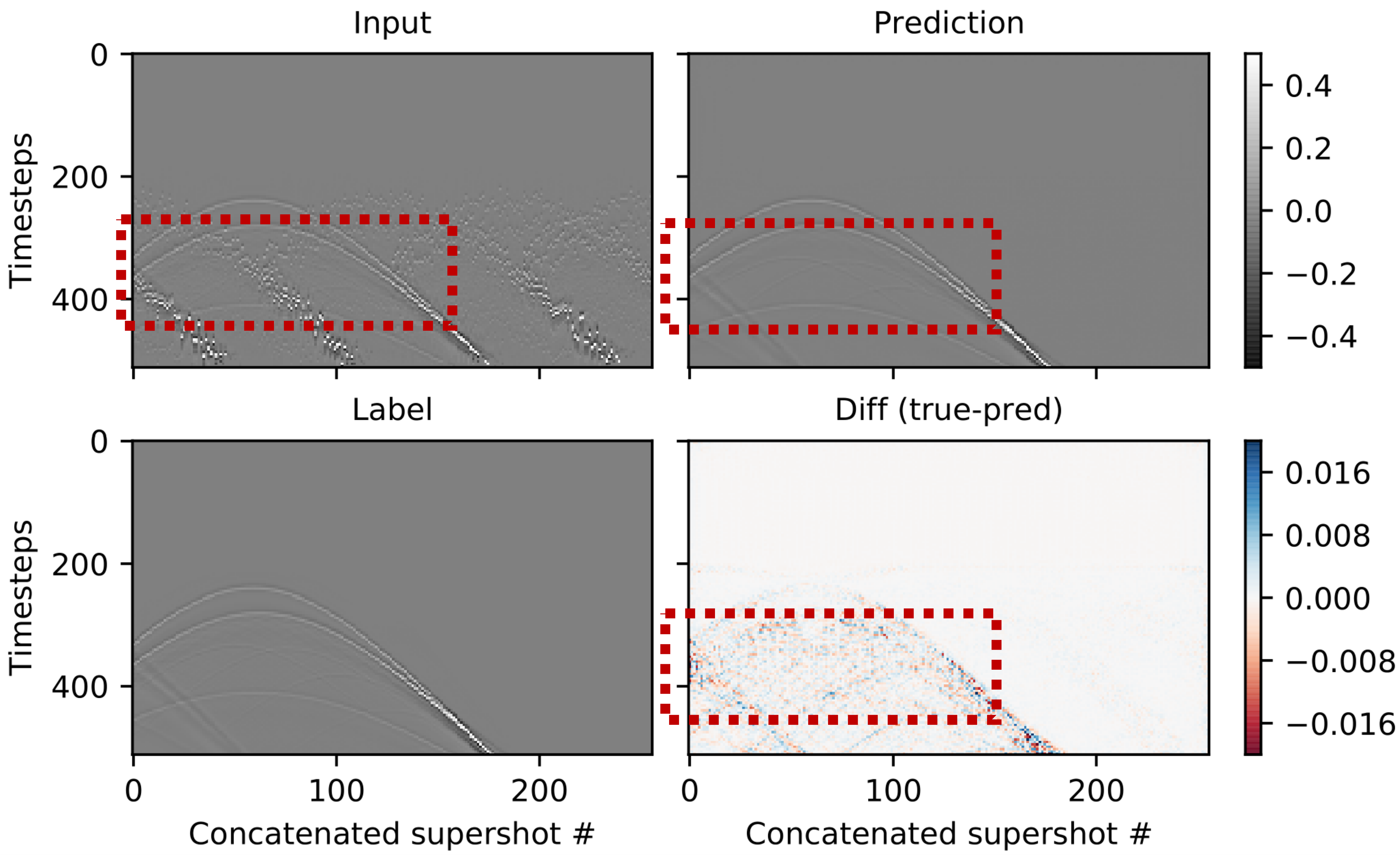
Results – Validation



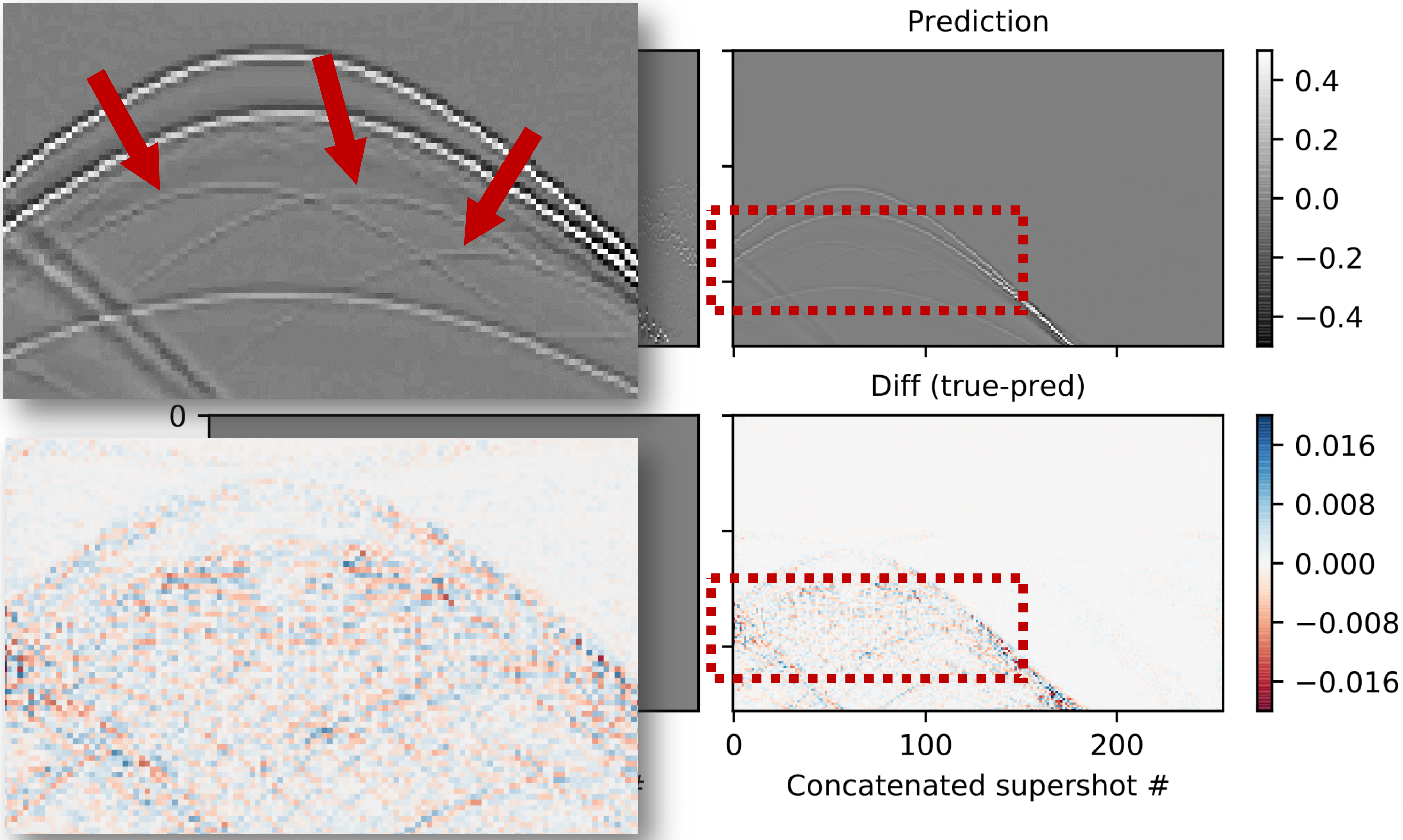
Results – Validation



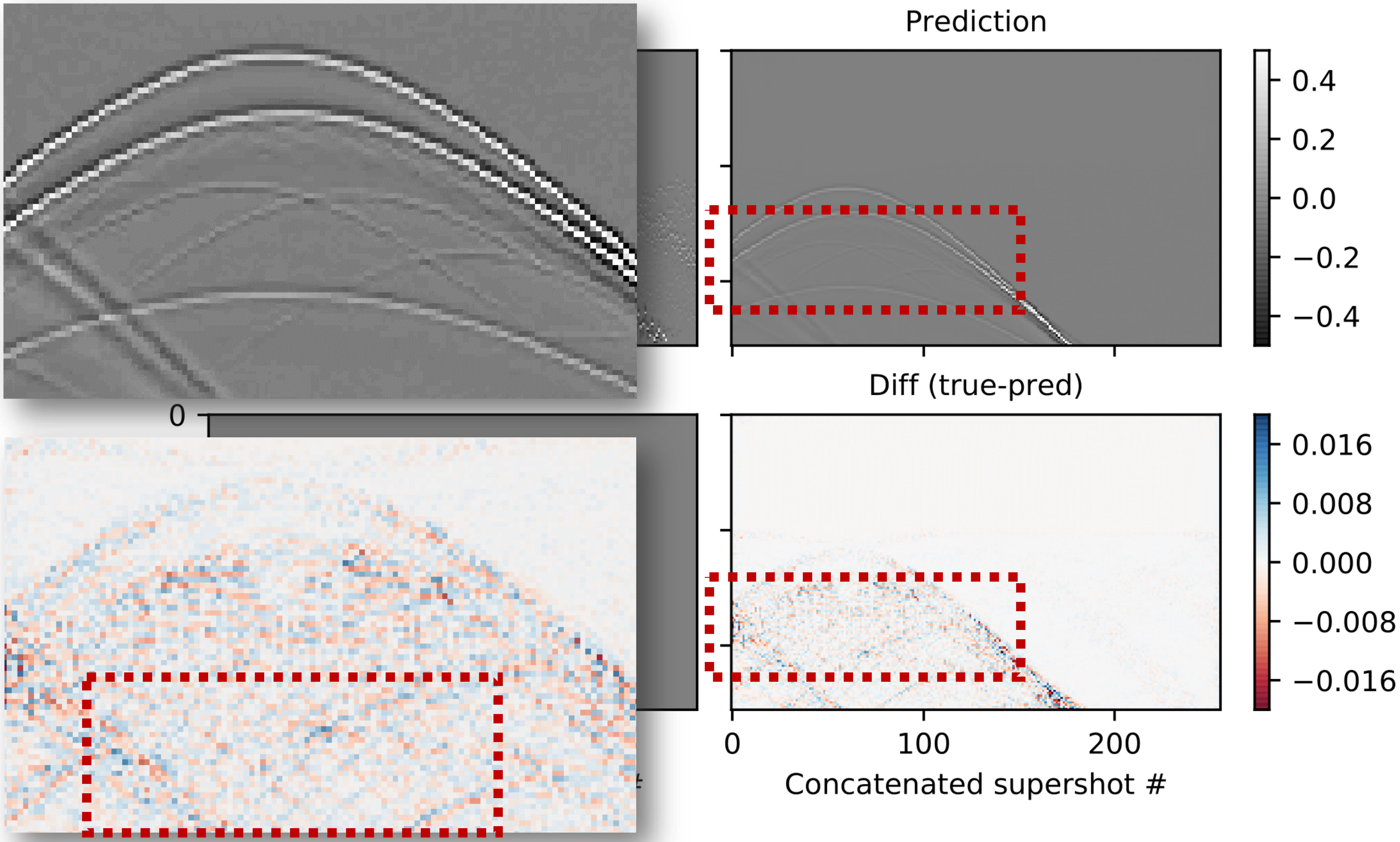
Results – Validation



Results – Validation



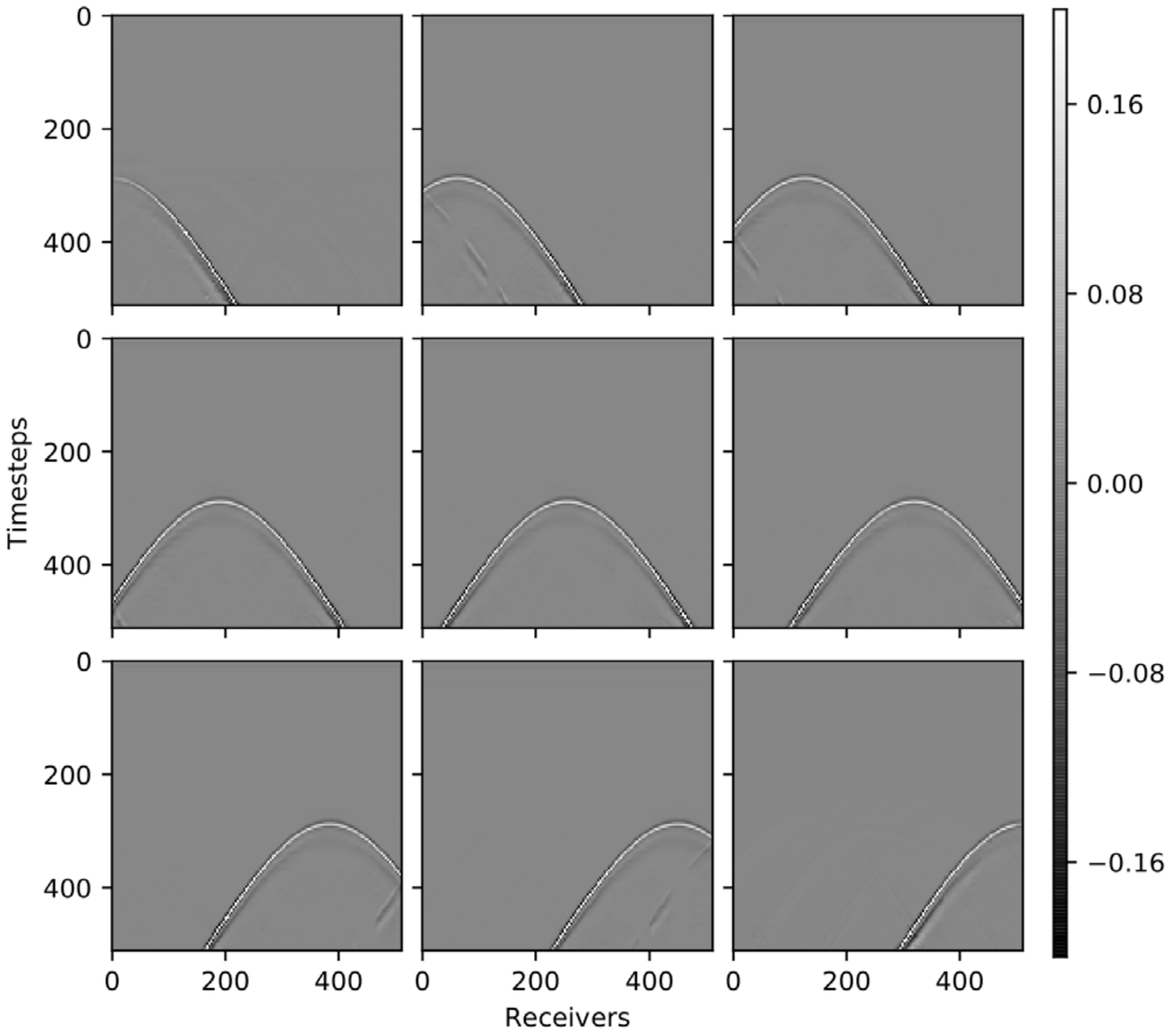
Results – Validation



Content

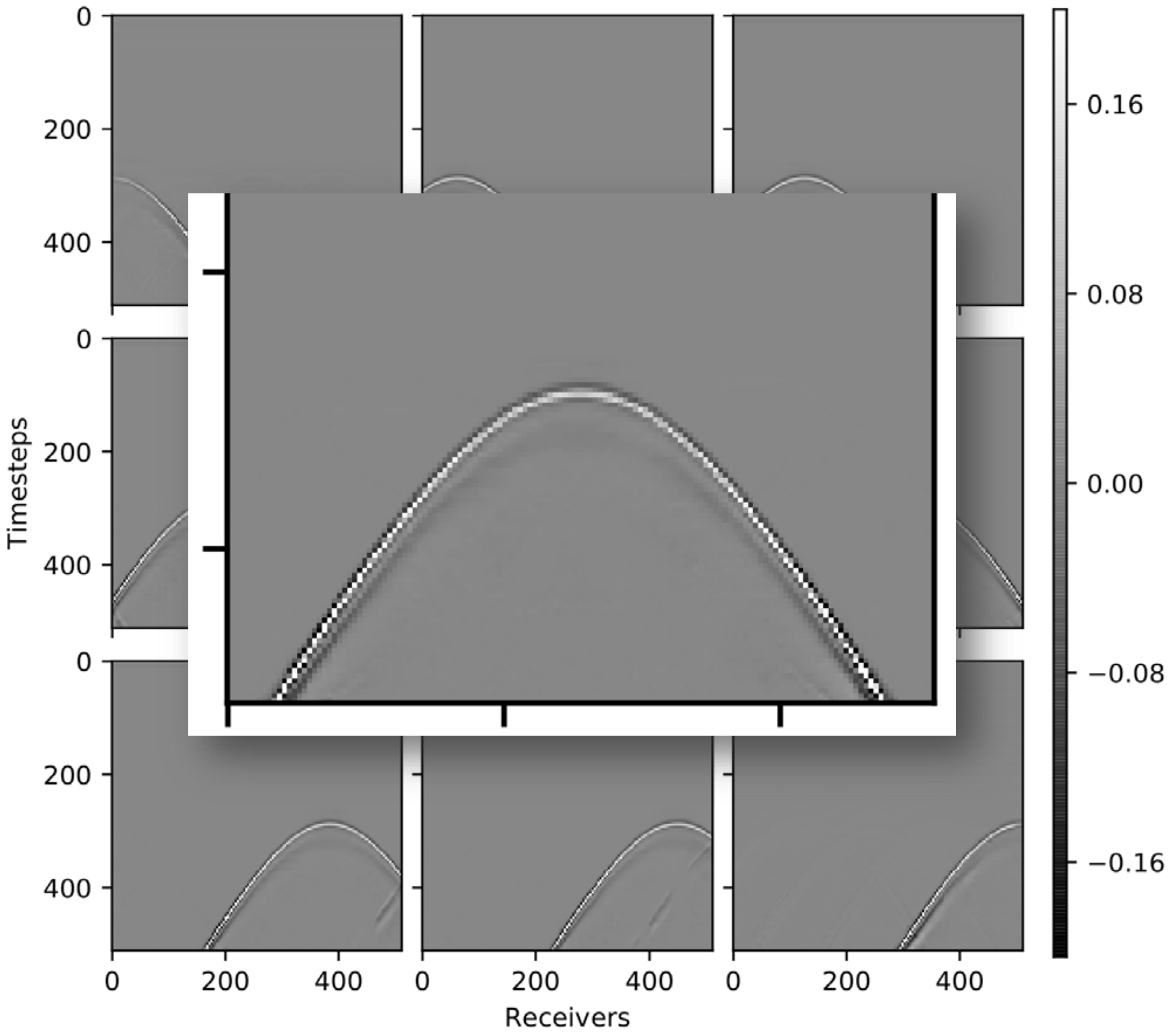
- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

Results – Test on different velocity model



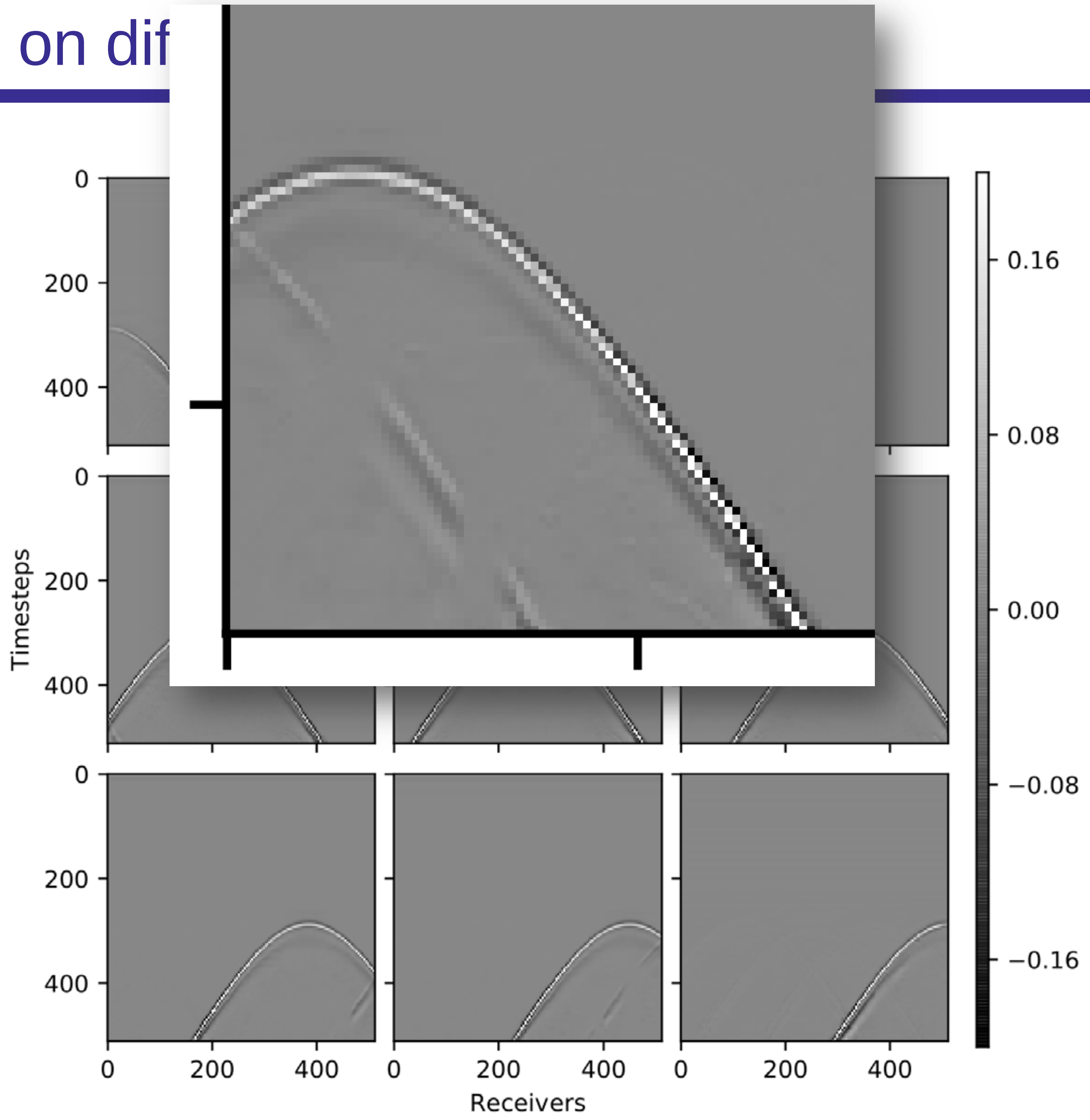
Two layer model

Results – Test on different velocity model



Two layer model

Results – Test on dif



Two layer model

Content

- Motivation
- Theory
  - U-Net
  - Workflow
- Results
  - Input/target definition
  - Validation
  - Test
- Conclusion

Conclusion

- We trained a U-Net for solving deblending problems and the best combination of the hyper-parameters was found
- The network performs well on data with the same distribution of the training set.

Future work:

- Gradient boosting
- Training on patches
- More randomized velocity models (more data!)
- More advance neural network structures

Acknowledgments

- All CREWES sponsors
- NSERC (CRDPJ 461179-13)
- Jian Sun, Marcelo Guarido and Hongliang Zhang for valuable discussions

and thank you!