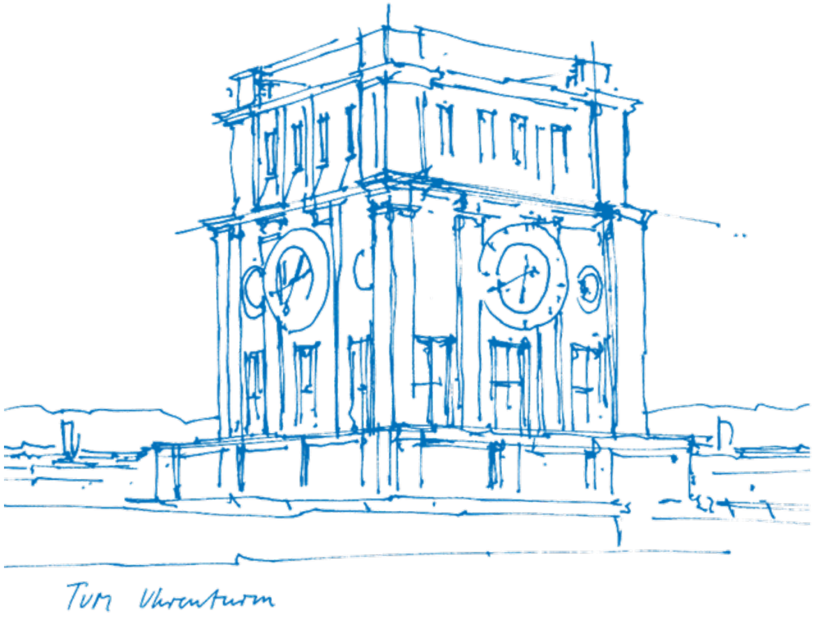
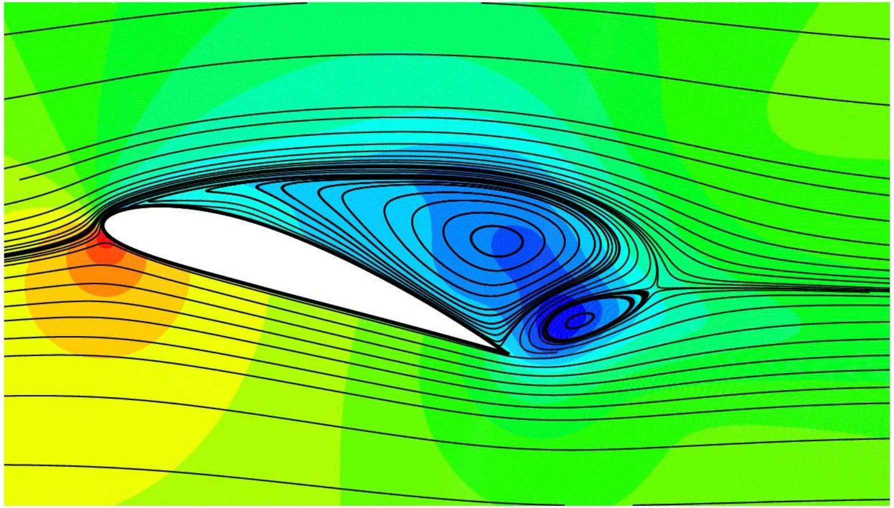
Deep Learning Methods for Reynolds-Averaged Navier-Stokes Simulations of Airfoil Flows



Julian Hohenadel
Technical University of Munich
Chair of Computer Graphics and Visualization
Munich, 11. May 2020

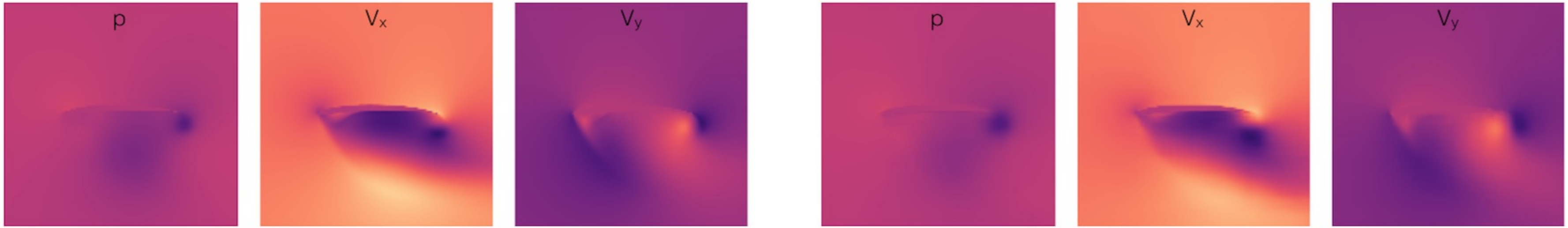
What this paper is about

- Inference of Reynolds-Averaged Navier-Stokes (RANS) solutions
- Pressure and velocity distributions
- Airfoil shapes
- Deep learning – U-Net architecture derivative
- Generalization



Taken from `https://www.pinterest.ch/pin/615163630322034457/`

Teaser



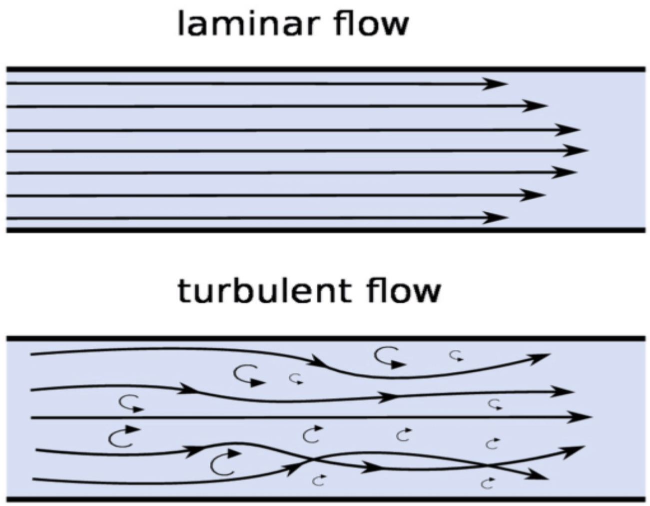
Target / ground truth solution

Solution inferred by trained CNN

Taken from `https://github.com/thunil/Deep-Flow-Prediction`

Background – RANS

- Nonlinear partial differential equation (PDE) system
- Based on Navier-Stokes equations
- Used for the modeling of turbulent incompressible flows
- Averages over time component



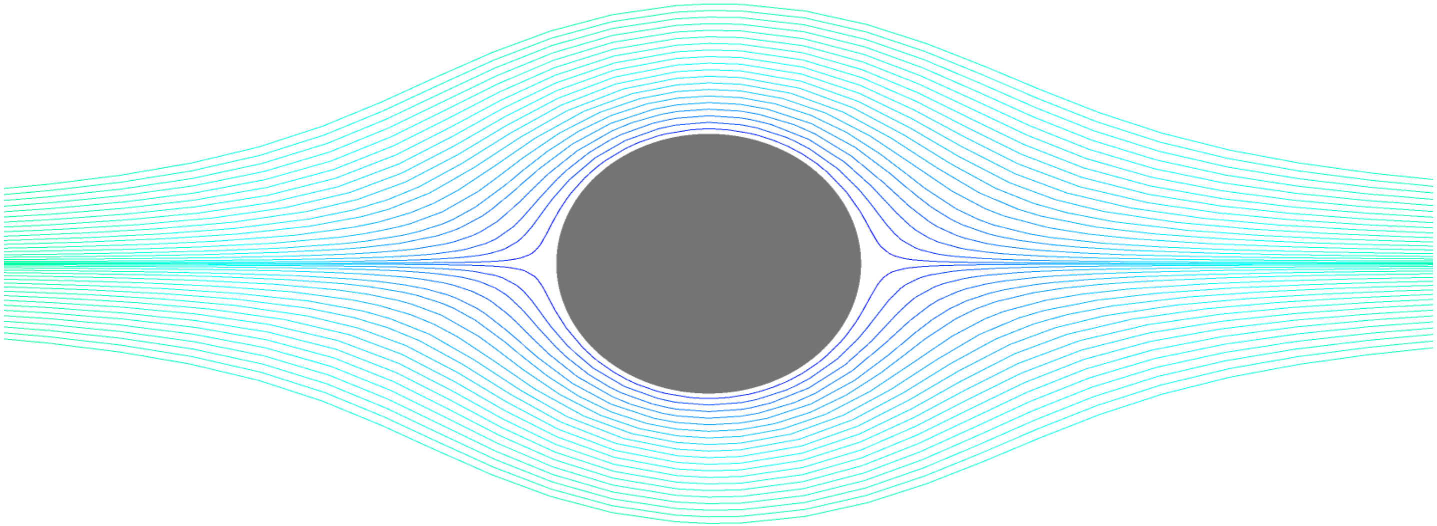
Taken from `https://diffzi.com/laminar-flow-vs-turbulent-flow/`

Background – RANS

Reynolds number *Re*:

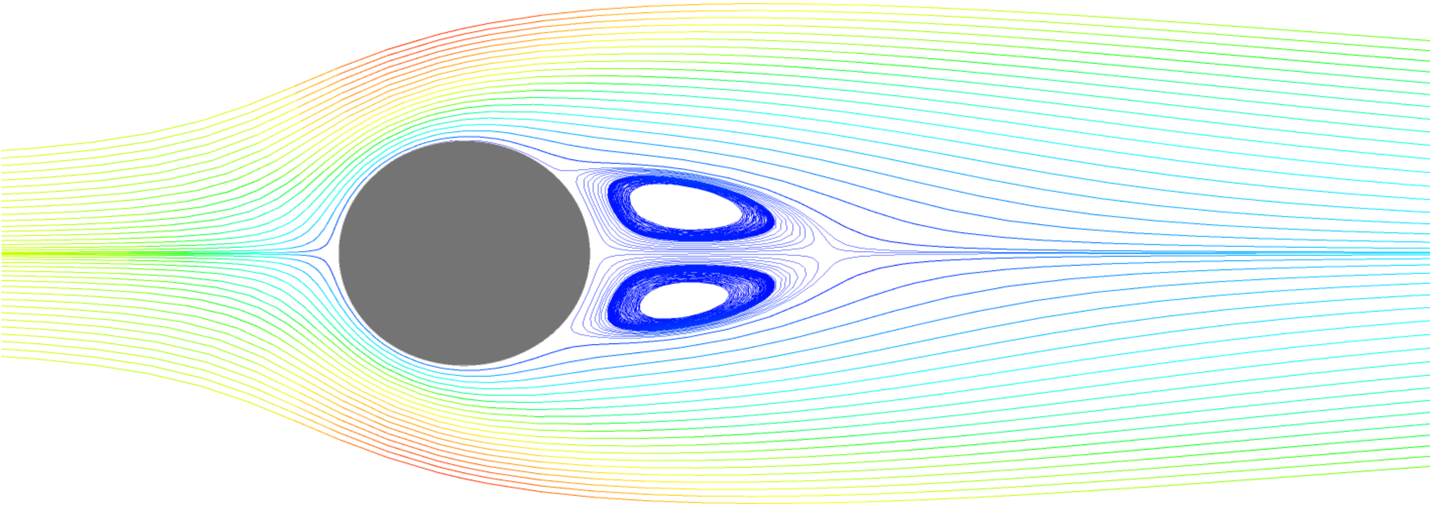
- dimensionless constant
- needed for calculation of turbulence models
- magnitude decides flow (laminar, turbulent)
- affects lift and drag coeffcients

Background – Reynolds number: < 1



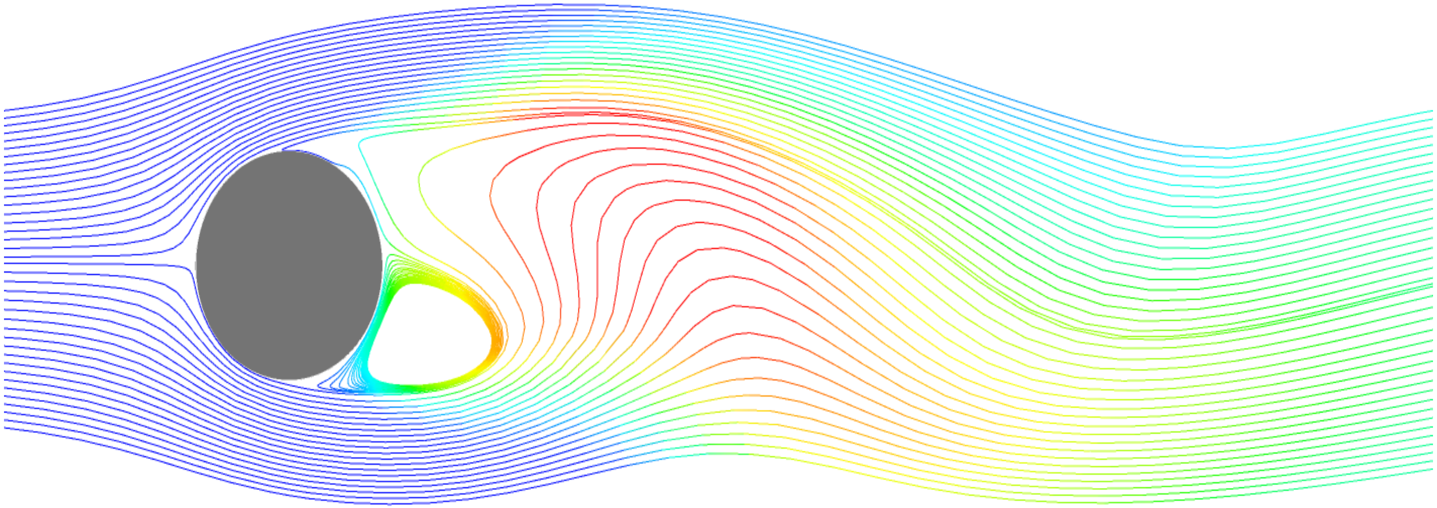
Taken from https://www.computationalfluiddynamics.com.au/reynolds-number-cfd/

Background – Reynolds number: ≈ 10



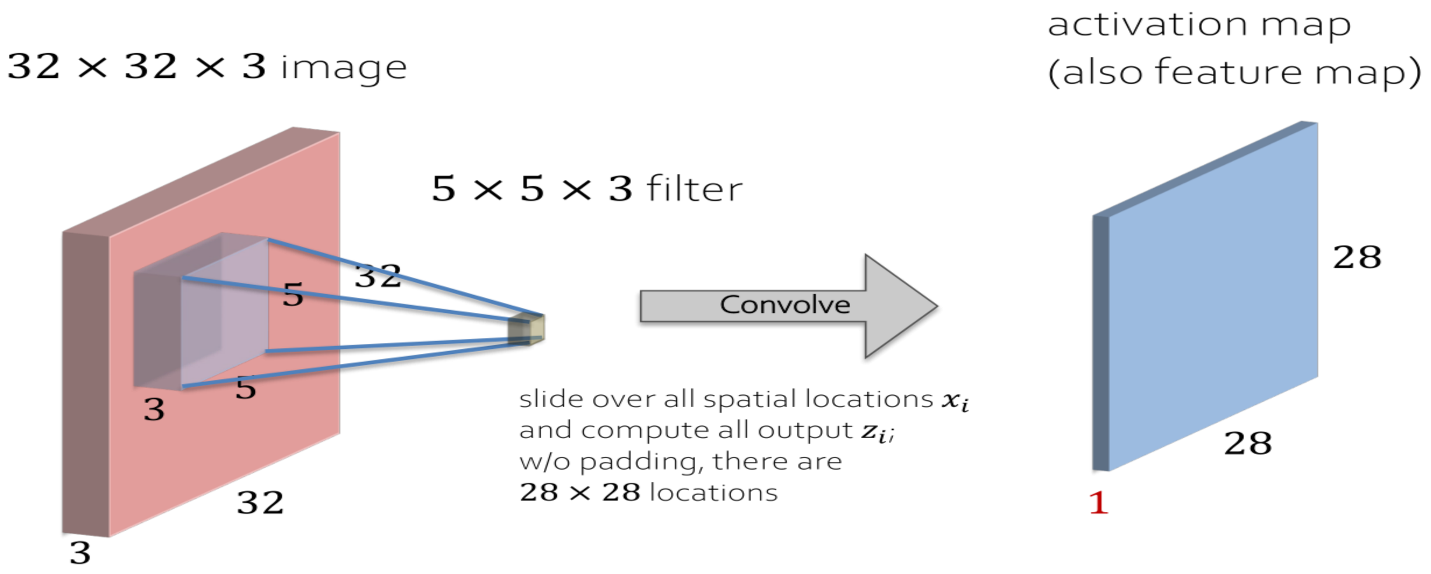
Taken from `https://www.computationalfluiddynamics.com.au/reynolds-number-cfd/`

Background – Reynolds number: $\approx 1 \cdot 10^5$



Taken from `https://www.computationalfluiddynamics.com.au/reynolds-number-cfd/`

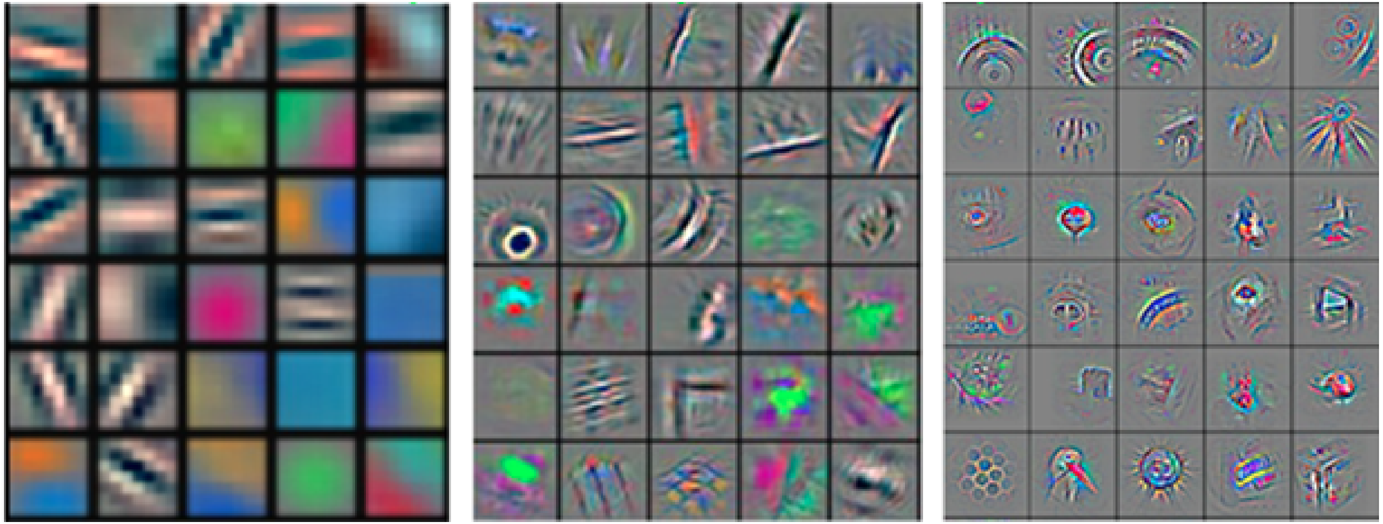
Background – Convolutions



Taken from I2DL WS19/20 (TUM)

Background – Convolutions

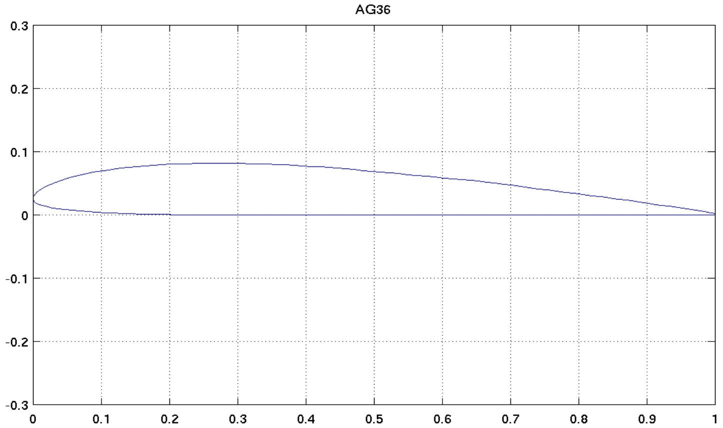
Low-Level Features, Mid-Level Features, High-Level Features: each filter captures different characteristics



Taken from `https://arxiv.org/pdf/1311.2901.pdf`

Data – Generation

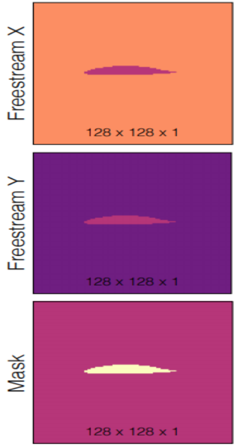
- Airfoil shapes from UIUC database
- Reynolds number: $[0.5, 5] \cdot 10^6$ (highly turbulent)
- Angle of attack: $[-22.5, 22.5]$
- Ground truth generated with OpenFOAM
  (pressure, x velocity, y velocity)
- Training data resolution: $3 \times 128 \times 128$
  (Inference region $<$ full simulation domain)



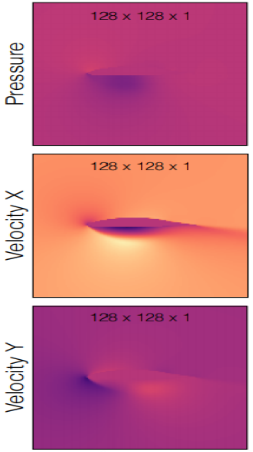
Taken from `https://m-selig.ae.illinois.edu/ads/afplots/ag35.gif`

Data – Structure

Input channels



Target channels



Reynolds number encoded as differently scaled freestream velocity vectors wrt. their magnitude

Data from the RANS solution

Pre-processing – Normalization

Motivation: Flatten space of solutions, accelerate learning by simplifing the learning task for the NN

Bernoulli equation for incompressible (**laminar**) flow:

$$\frac{\mathbf{v}^2}{2} + gz + \frac{\mathbf{p}}{\rho} = constant$$

- v: velocity
- g: acceleration (constant)
- z: elevation (constant)
- p: pressure
- ρ: density (constant)

$\implies v^2 \sim p$ – e.g. double the speed quadruples the pressure

Pre-processing – Normalization

Normalization of target channels by division with freestream magnitude:

$\tilde{v}_o = \frac{v_o}{\|v_i\|}, \quad \tilde{p}_o = \frac{p_o}{\|v_i\|^2}$ – important to remove quadratic scaling of pressure

Pre-processing – Normalization

All units dissapear $\implies$ really dimensionless:

- Pressure: $[p]_{SI} = 1\,Pa = 1\,\frac{kg}{m \cdot s^2}$
- Density: $[\rho]_{SI} = 1\,\frac{kg}{m^3}$ – constant in incompressible flow
- Velocity: $[v]_{SI} = \frac{m}{s}$

Pre-processing – Offset removal & value clamping

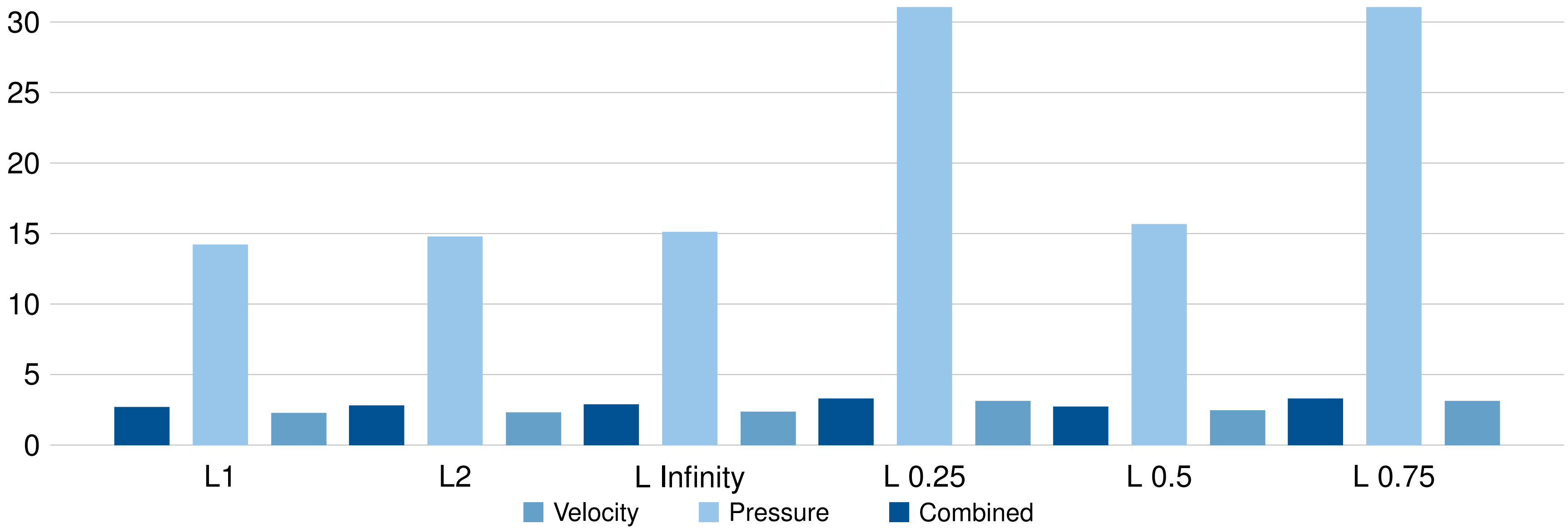
Motivation: eliminate ill-posed learning goal & improve numerical precision

- RANS typically only needs ∇_p for computation
- Spatially move pressure distribution into the origin
- $\hat{p}_o = \tilde{p}_o - p_{mean}$
- Clamp both input and target channels into $[-1, 1]$ range

Pre-processing – Evaluation

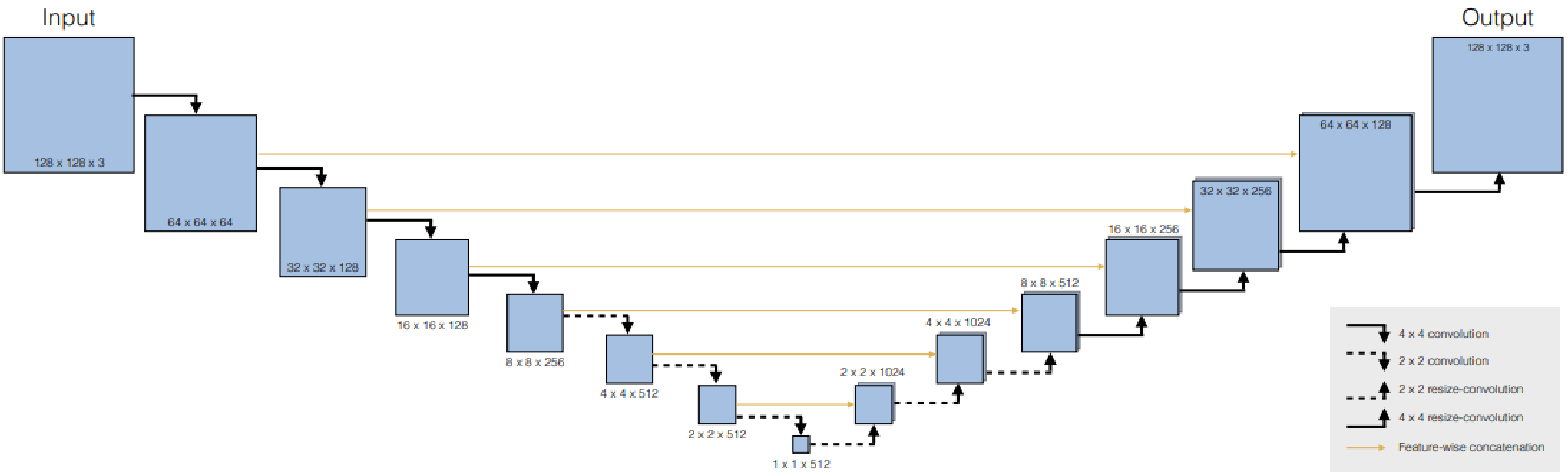
Vector norms used in pre-processing comparision wrt. error, default: L2 (in %)

L1 normalization achieves the best error rates (p, vel, combined: **14.19**%, **2.251**%, **2.646**% – L2: 14.76%, 2.291%, 2.780%)



Architecture

U-Net derivative proposed in the paper:



Taken from `https://arxiv.org/pdf/1810.08217.pdf`

Architecture – Convolutional blocks

Encoder

1. Activation – Leaky ReLu (0.2)
2. Convolution – Width down, Depth up
3. Batch normalization
4. Dropout (1%)

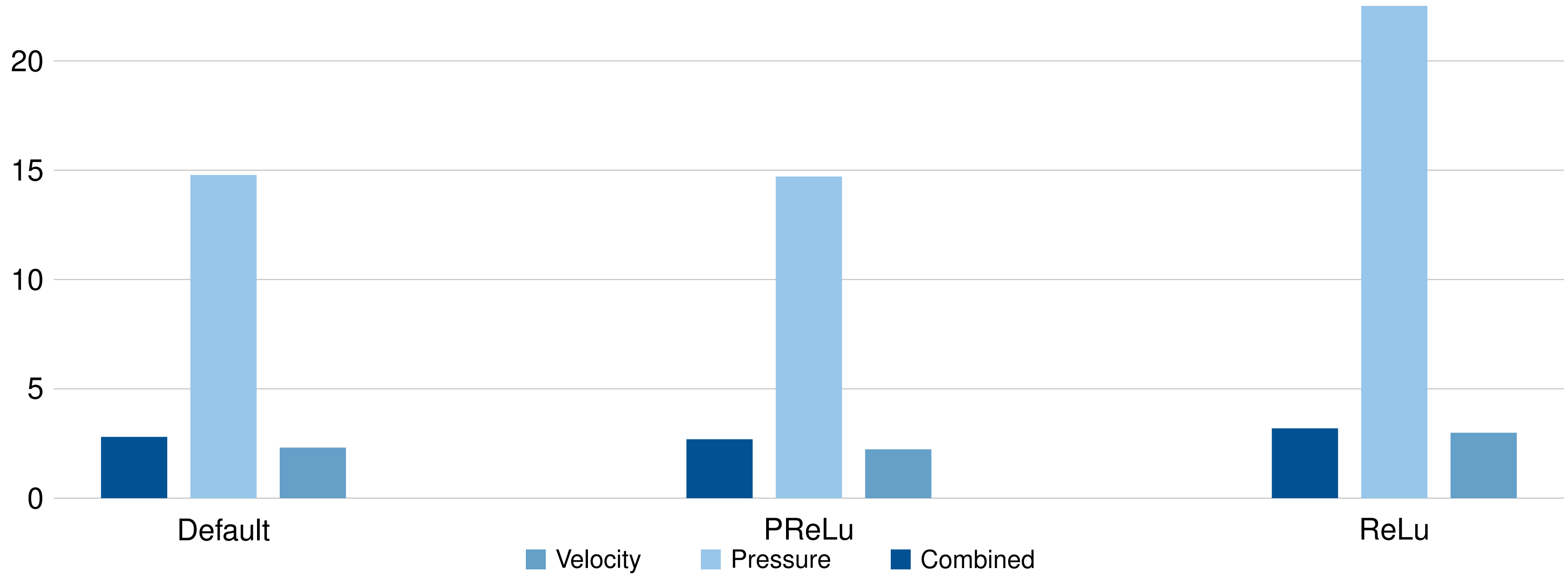
Decoder

1. Activation – ReLu
2. Upsampling – linear (2.0)
3. Convolution – Width up, Depth down
4. Batch normalization
5. Dropout (1%)

Architecture – Evaluation

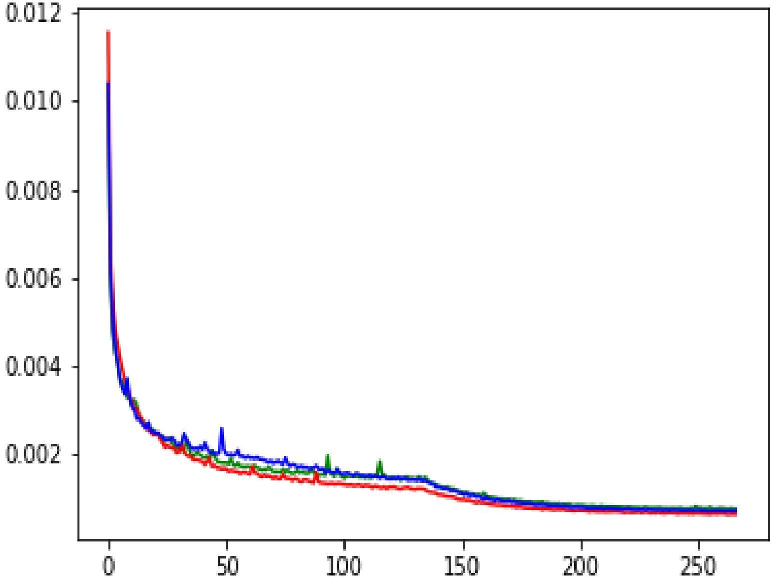
Error percentage of different activation functions after 160k iterations (266 epochs).

PReLu achieves the best error rates (p, vel, combined: **14.69**%, **2.216**%, **2.676**% – Default: 14.76%, 2.296%, 2.787%)

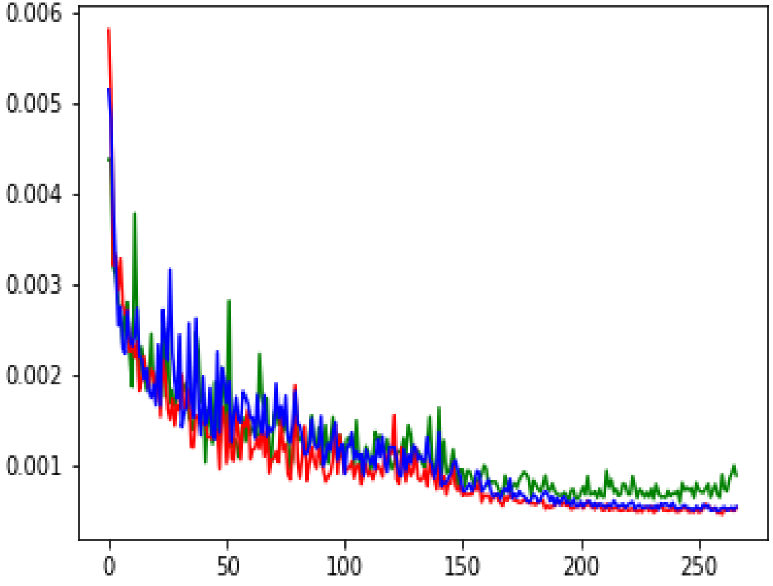


Architecture – Evaluation

Training loss (Default: b, PReLu: r, ReLu: g)



Validation loss (Default: b, PReLu: r, ReLu: g)



Transfer

Motivation: Can the network architecture adapt to other PDE systems & how will it perform?

Another use case for PDE systems: predicting wave propagation on shallow water

Governed by Saint-Venant equations (related with Navier-Stokes equations)

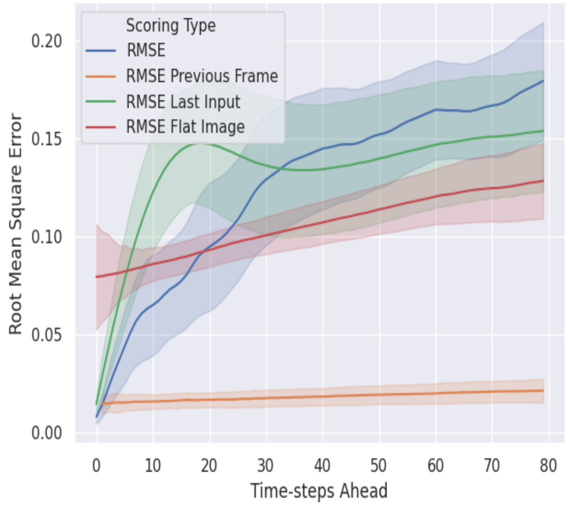
Transfer

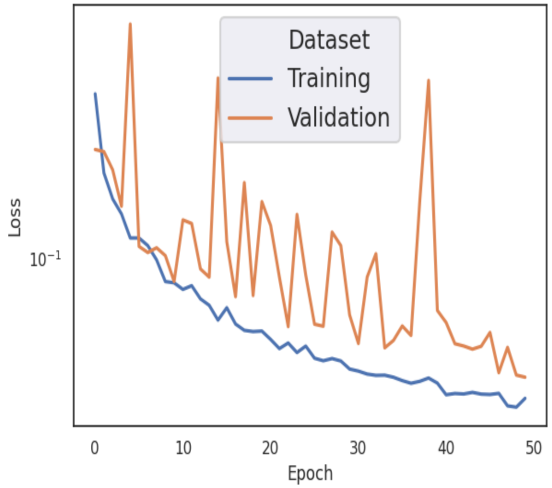
U-Net architecture changes:

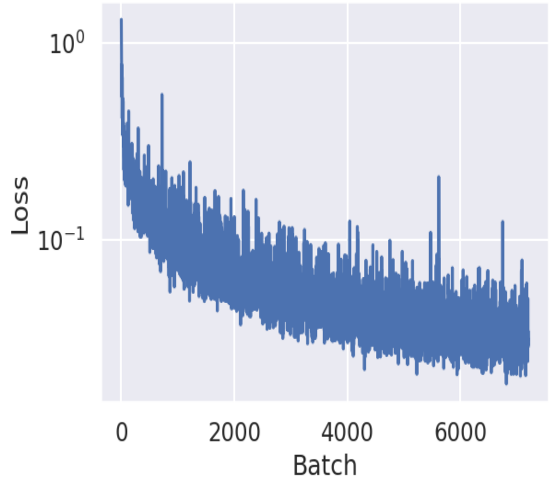
- Input channels – contain the last *n* time steps
- Output channels – predict the next *m* time steps
- Output is refeeded as input to predict time series

Transfer – Evaluation

RMSE with variance, validation loss and batch loss on Bigger Tub environment:





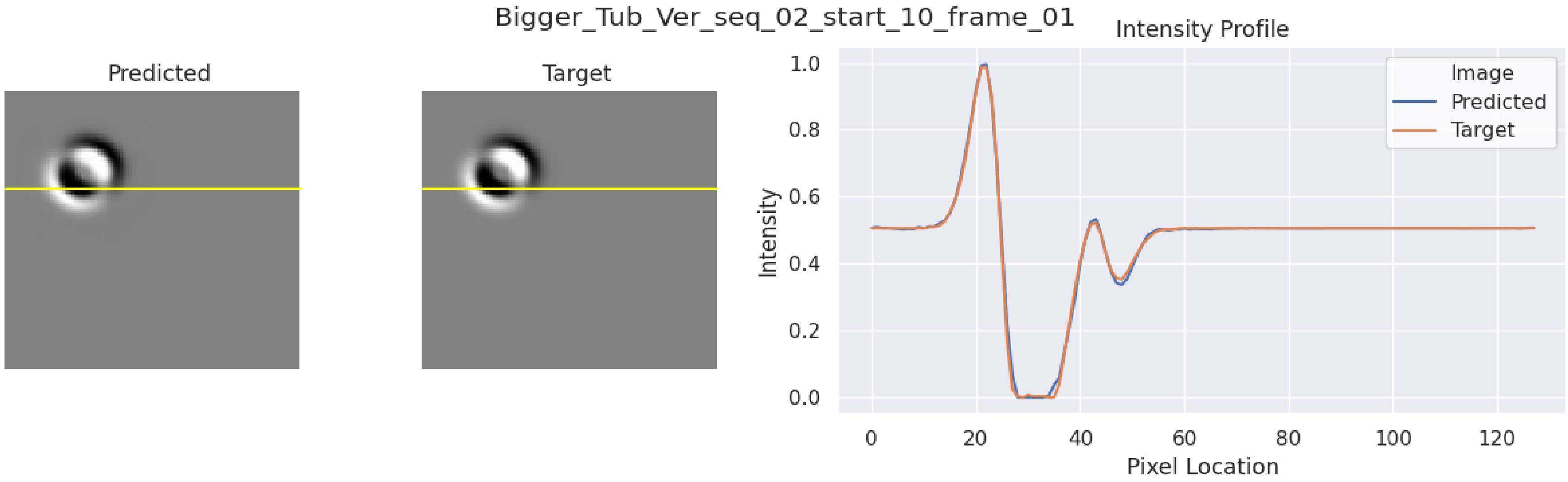


All plots and training in Transfer were made with `https://github.com/stathius/wave_propagation`

Transfer – Evaluation

Wave propagation prediction

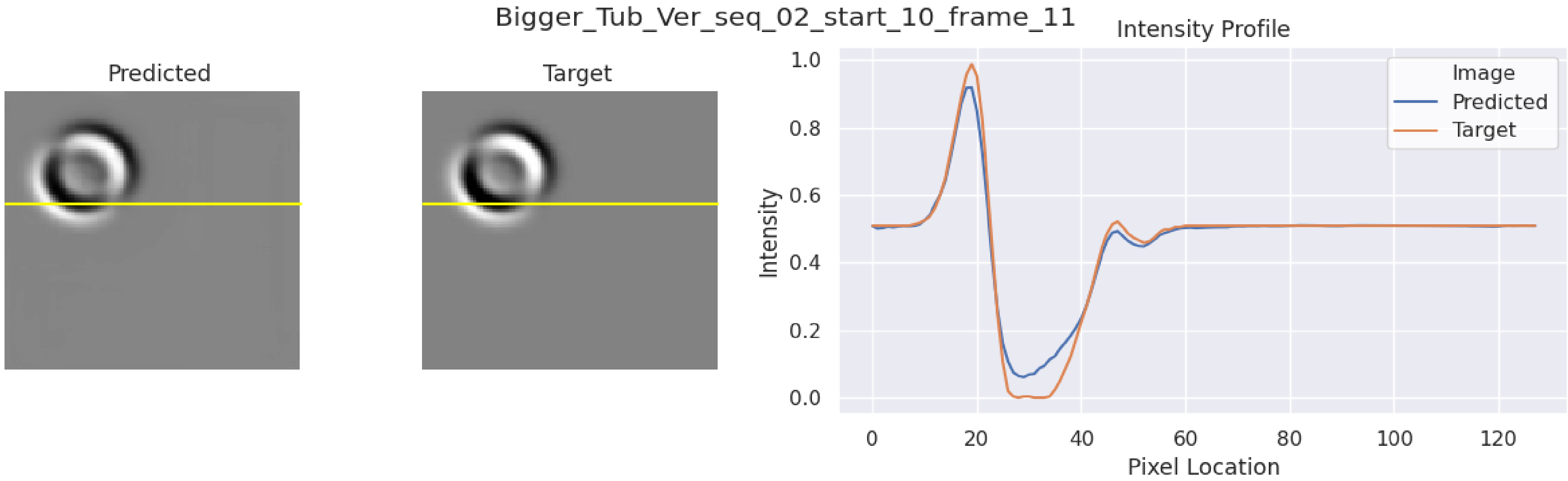
Intensity profile on scanline – Frame 1



Transfer – Evaluation

Wave propagation prediction

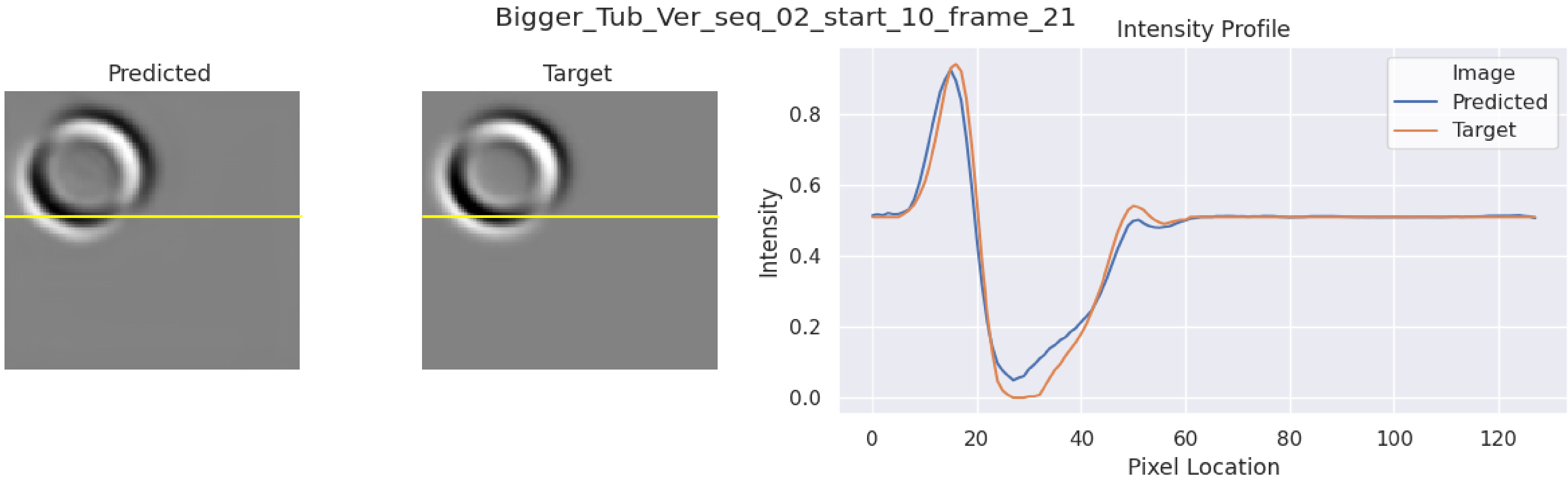
Intensity profile on scanline – Frame 11



Transfer – Evaluation

Wave propagation prediction

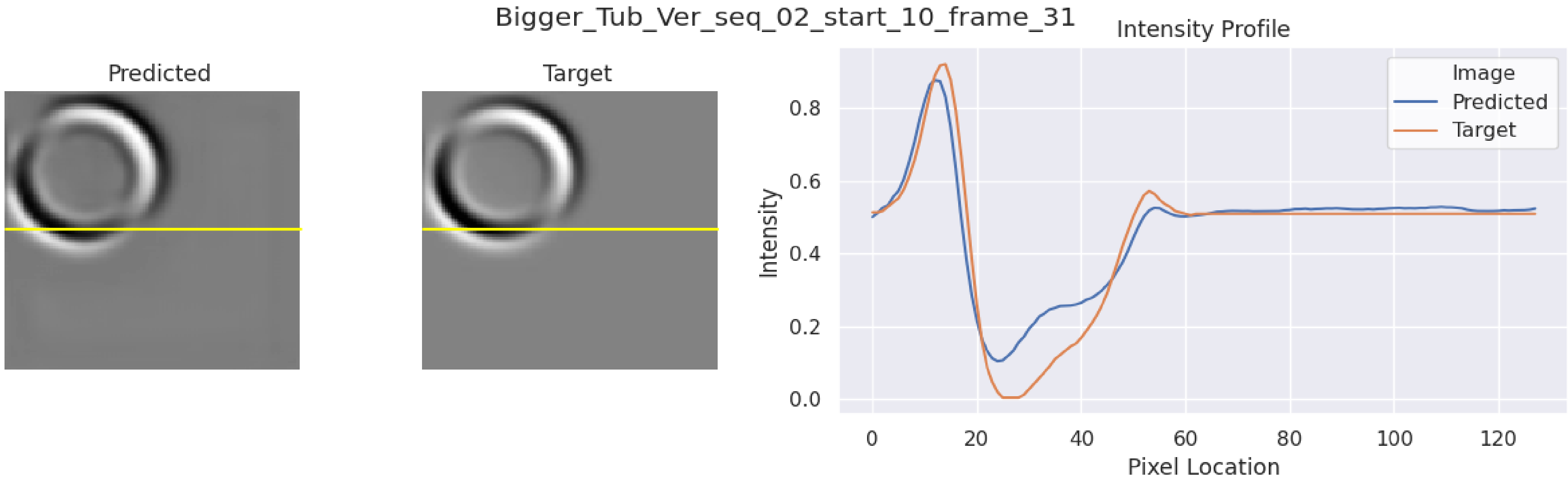
Intensity profile on scanline – Frame 21



Transfer – Evaluation

Wave propagation prediction

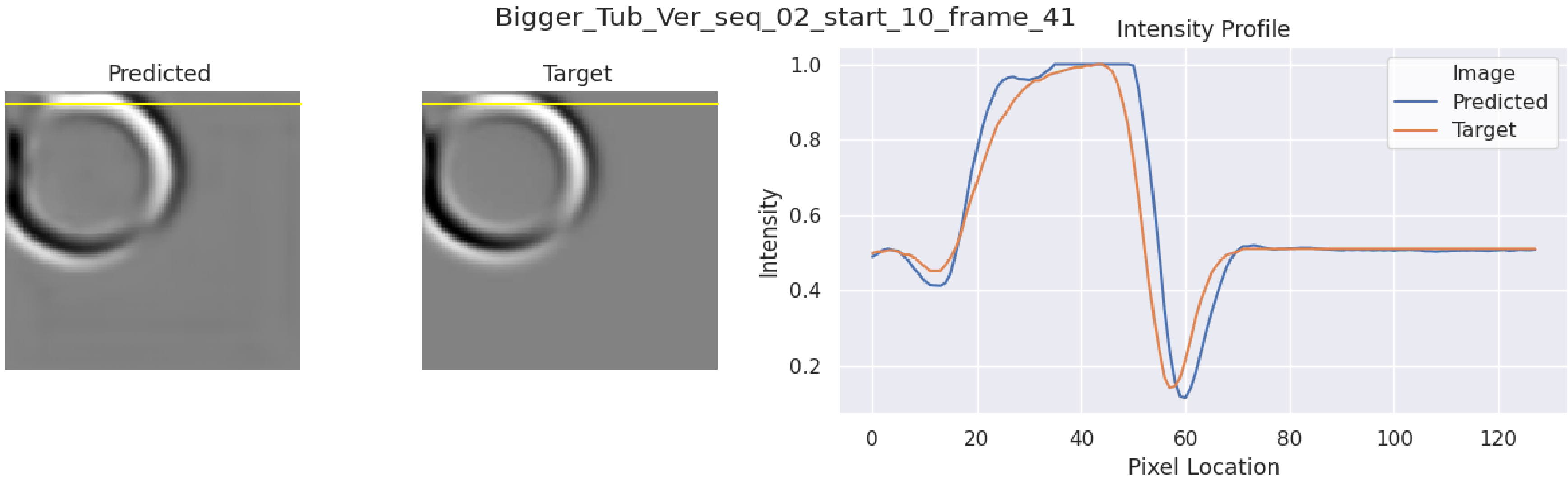
Intensity profile on scanline – Frame 31



Transfer – Evaluation

Wave propagation prediction

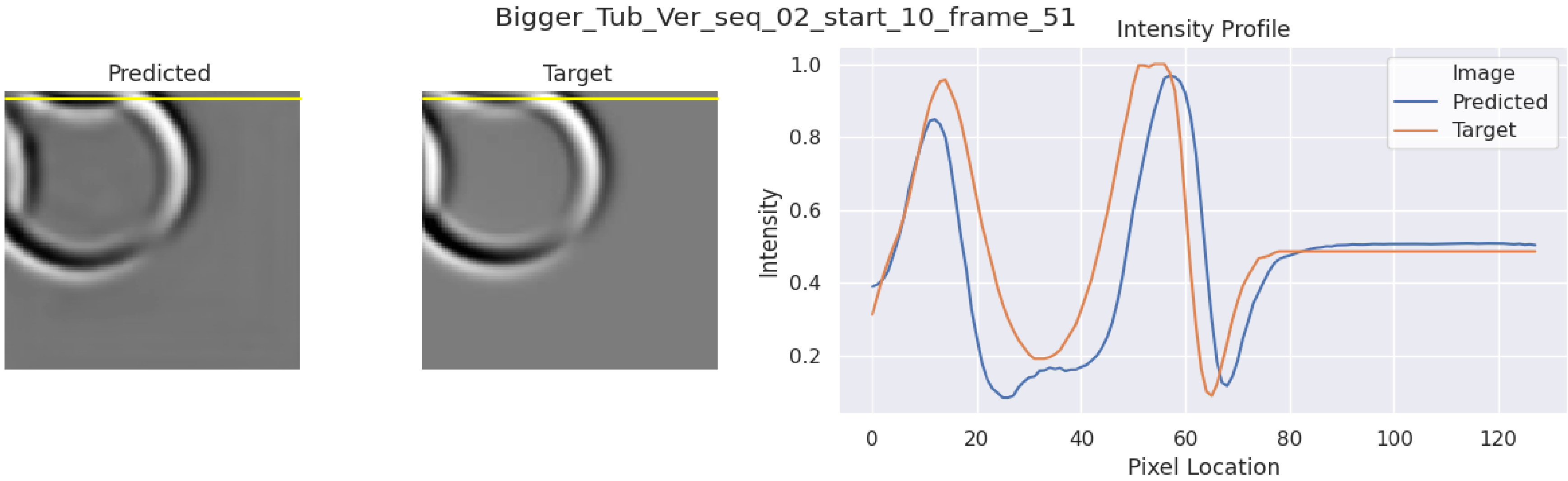
Intensity profile on scanline – Frame 41



Transfer – Evaluation

Wave propagation prediction

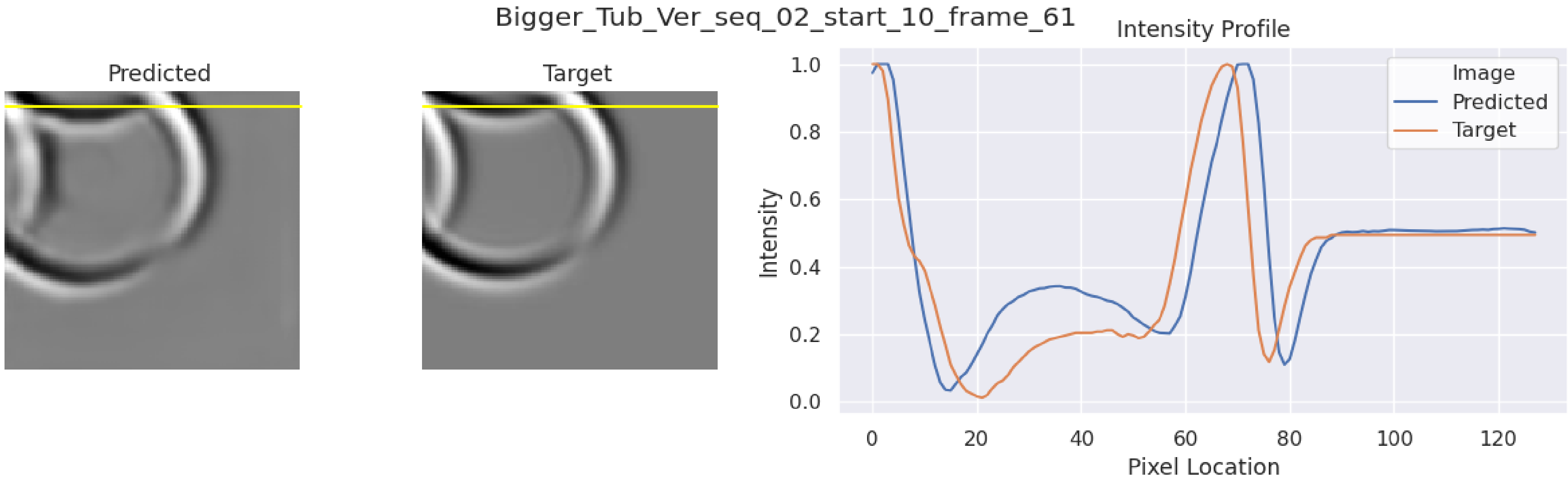
Intensity profile on scanline – Frame 51



Transfer – Evaluation

Wave propagation prediction

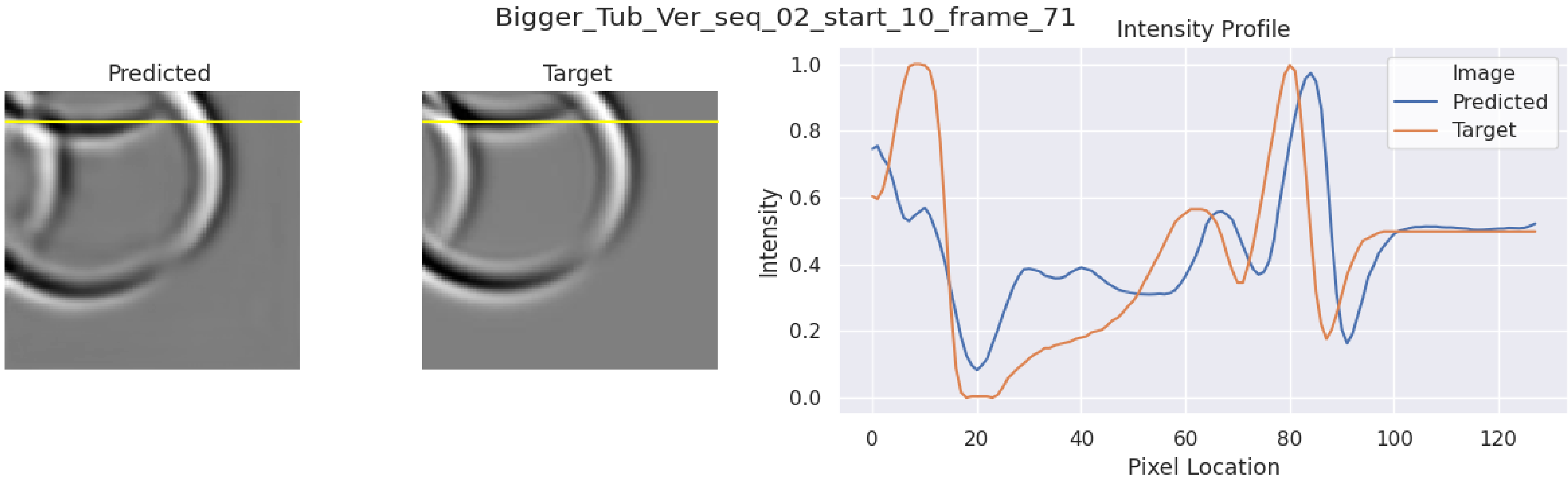
Intensity profile on scanline – Frame 61



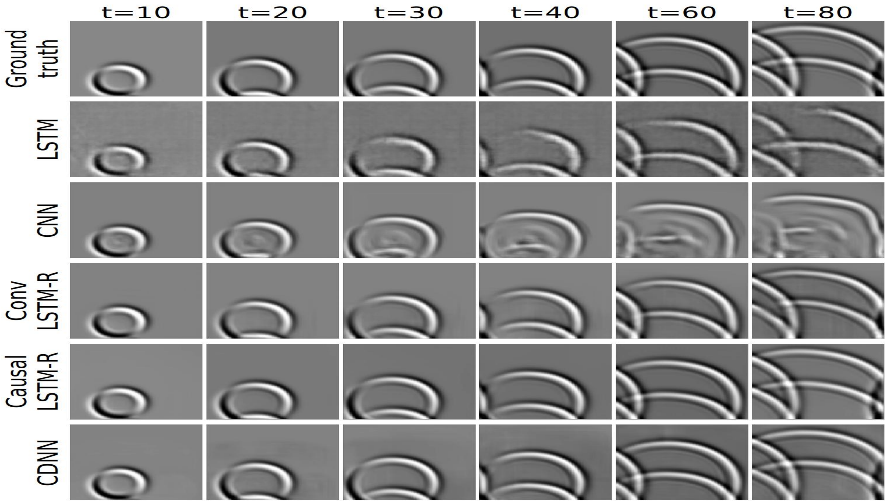
Transfer – Evaluation

Wave propagation prediction

Intensity profile on scanline – Frame 71



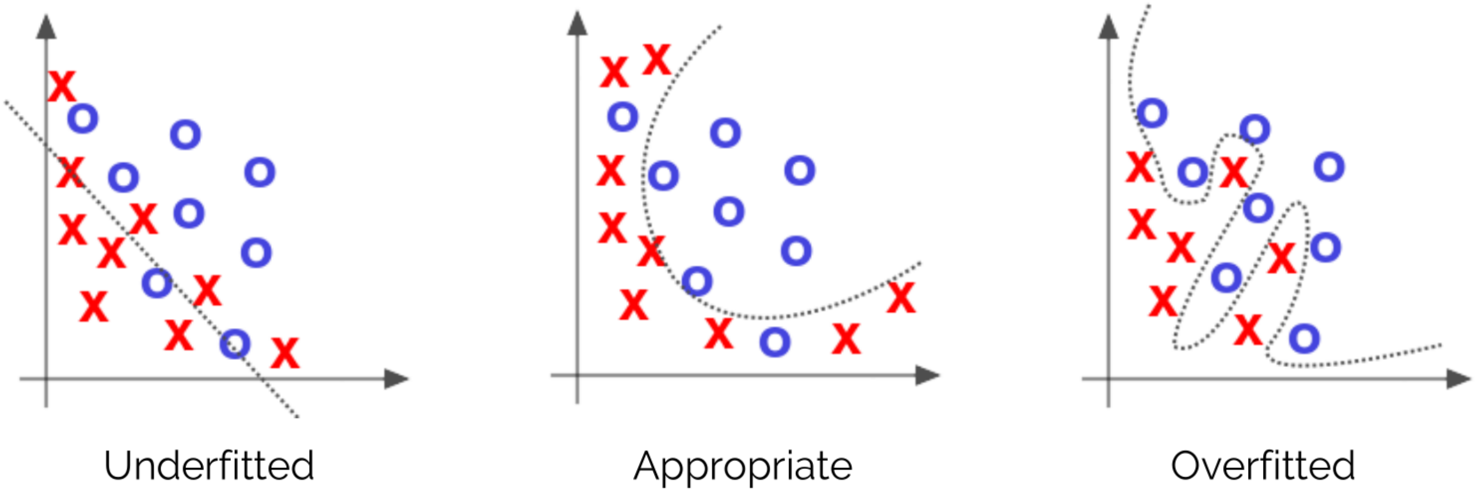
Transfer – Evaluation



Taken from `https://github.com/stathius/wave_propagation`

Generalization

Motivation: Key question in deep learning: How well does my NN perform on unseen data?



Taken from Deep Learning by Adam Gibson, Josh Patterson, O'Reily Media Inc., 2017

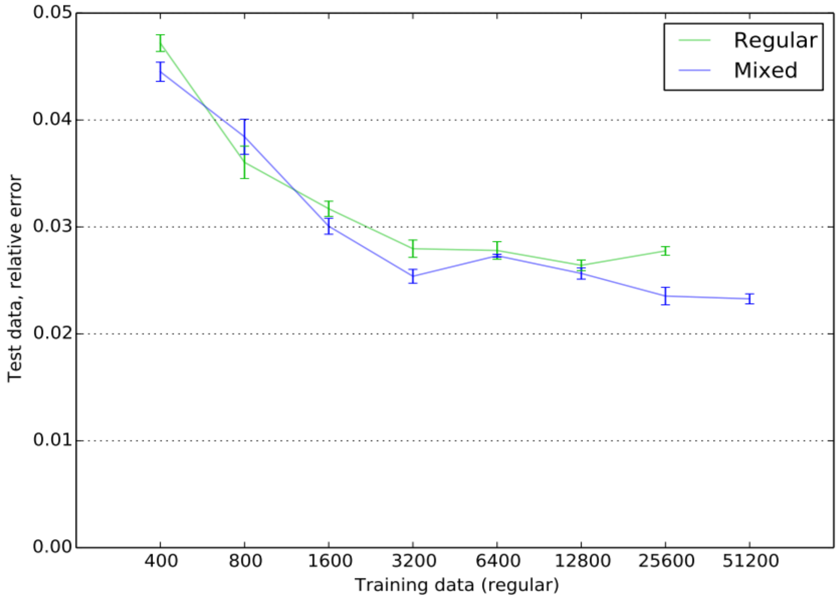
Generalization

Splitted up training data:

- Regular
- Mixed (50% regular, 50% sheared ($\pm$15 degrees))

The plot shows training with a $30.9 \cdot 10^6$ parameter model

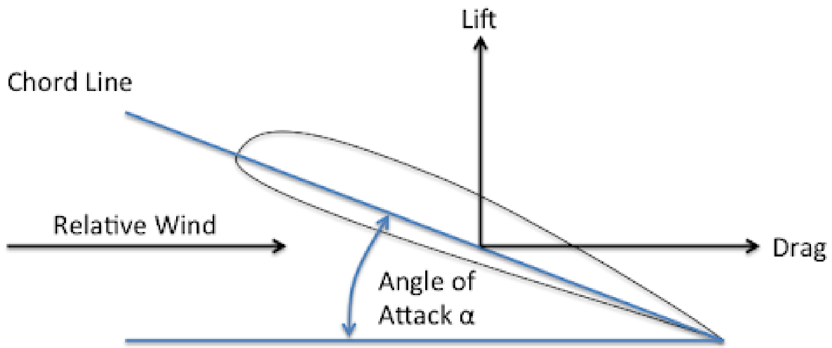
The high capacity supports training with the mixed dataset, achieving a even lower error



Taken from `https://arxiv.org/pdf/1810.08217.pdf`

Generalization – Evaluation

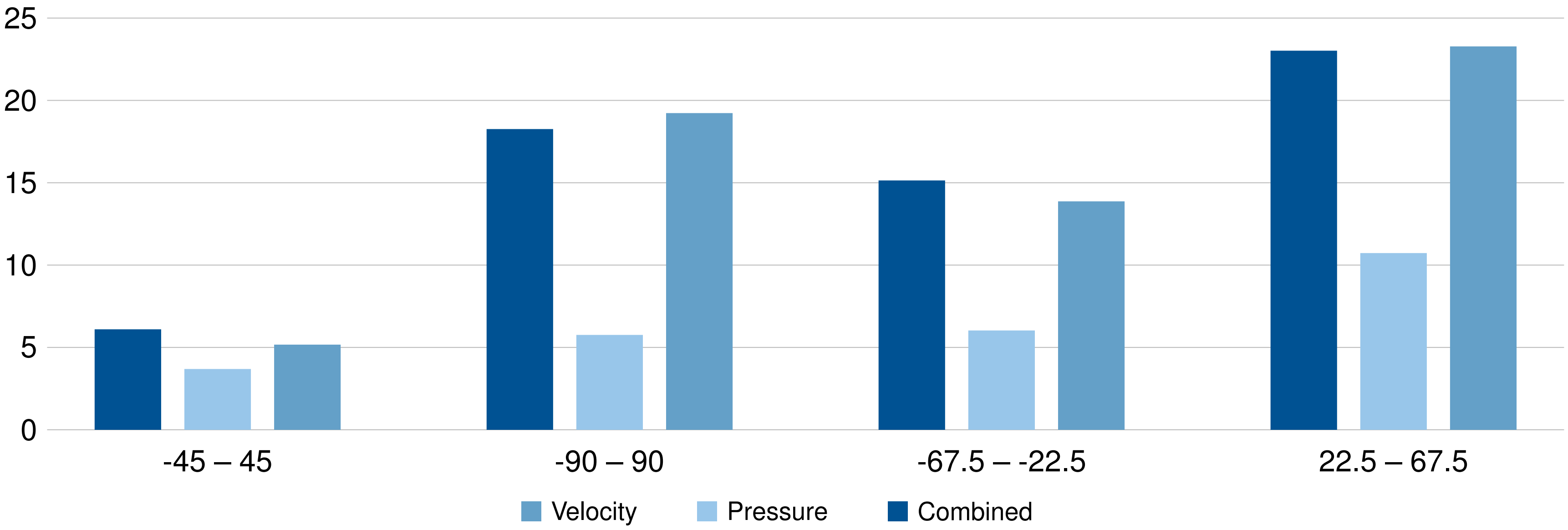
Generalization with different angels of attack:



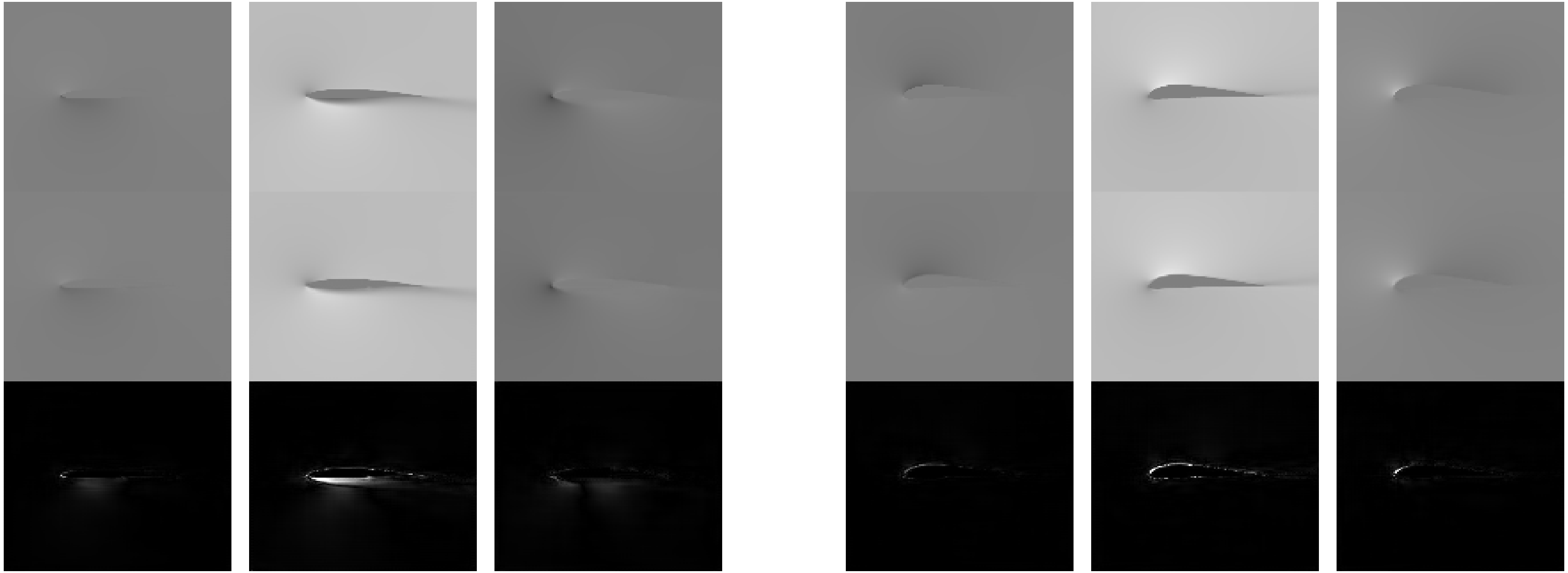
Taken from `http://www.aviationchief.com/angle-of-attack.html`

Generalization – Evaluation

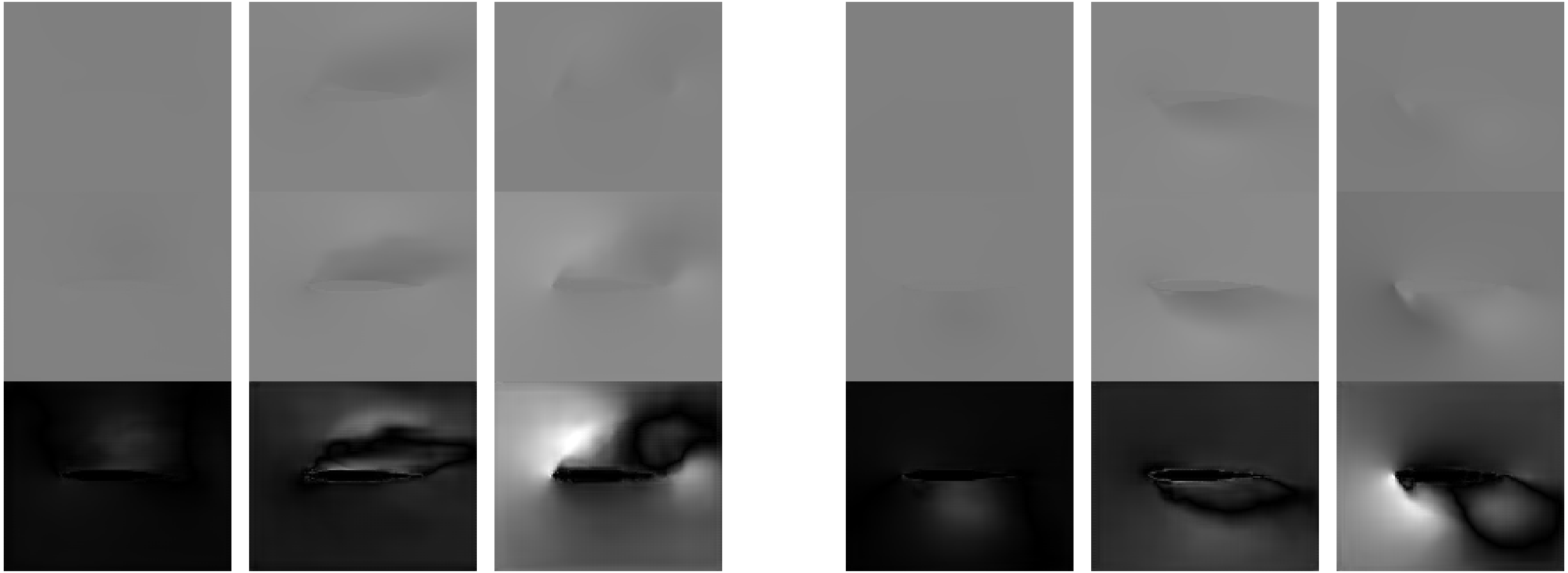
Error increase of different angle of attack intervals as test set wrt. ground truth $[-22.5, 22.5]$



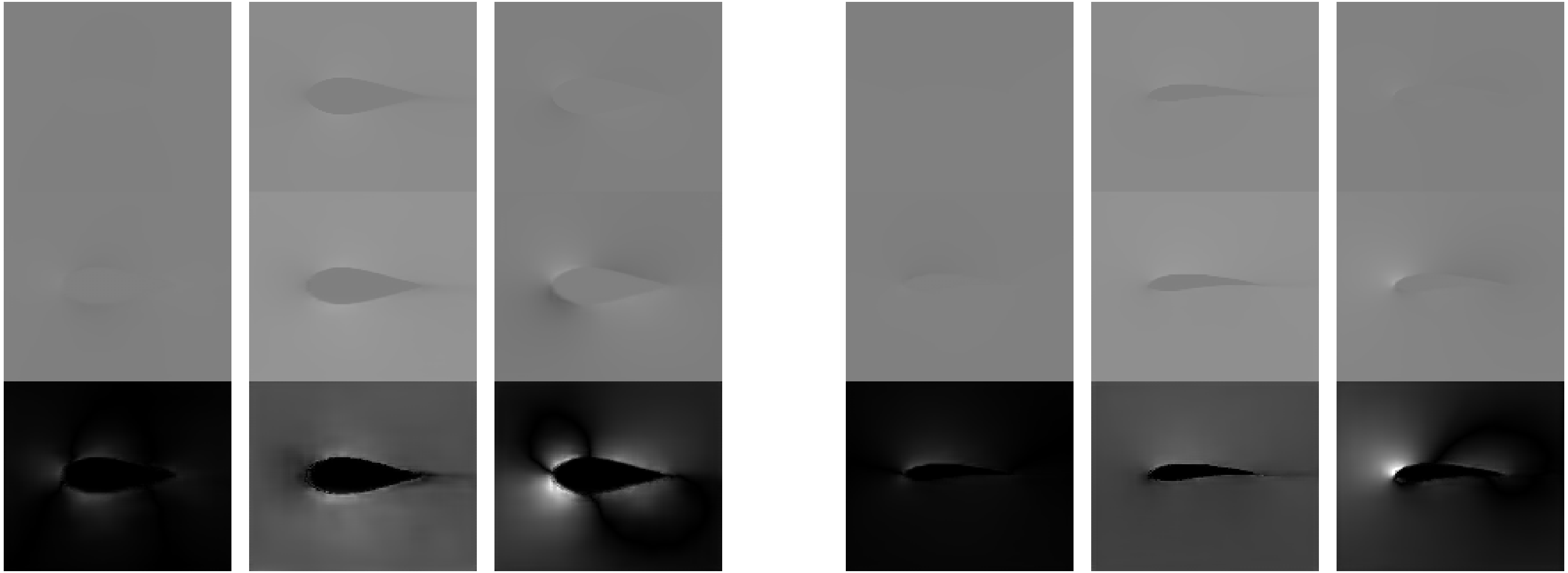
Generalization – $[-22.5, 22.5]$



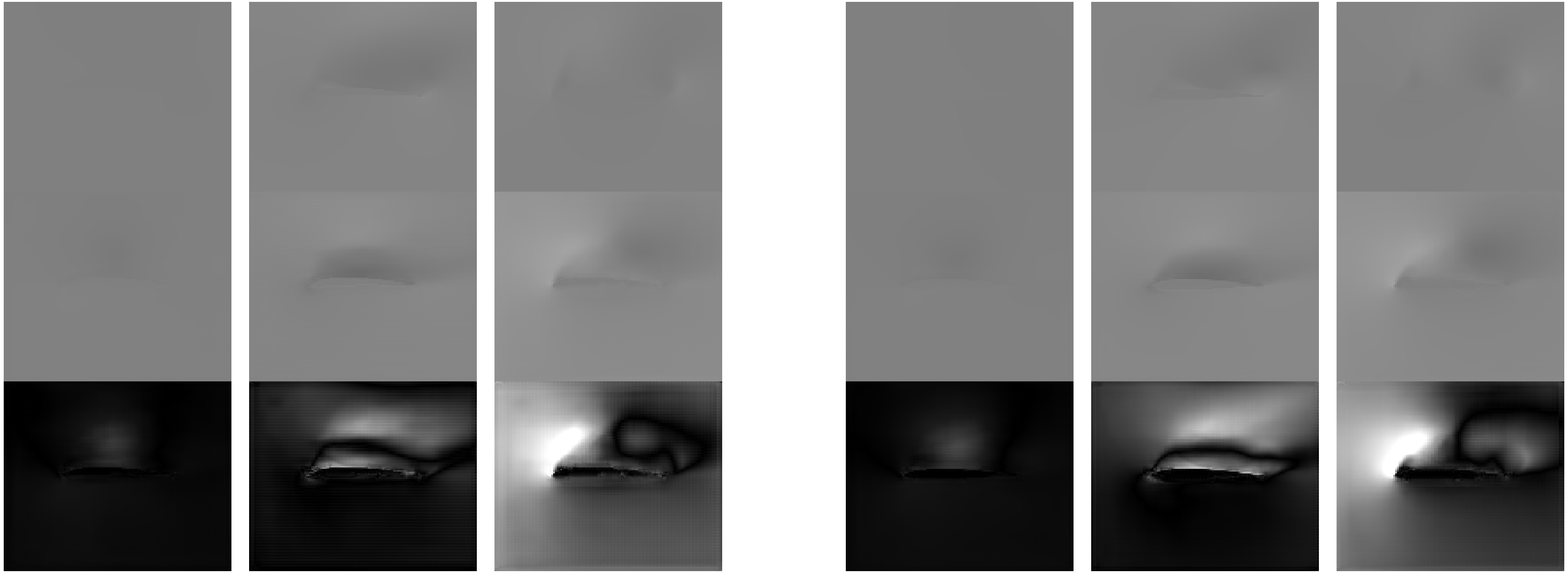
Generalization – $[-45, 45]$



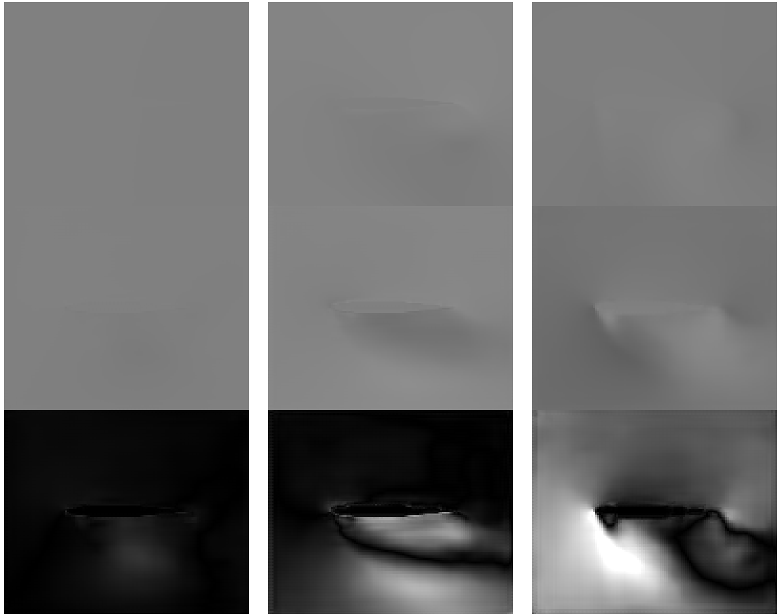
Generalization – $[-90, 90]$

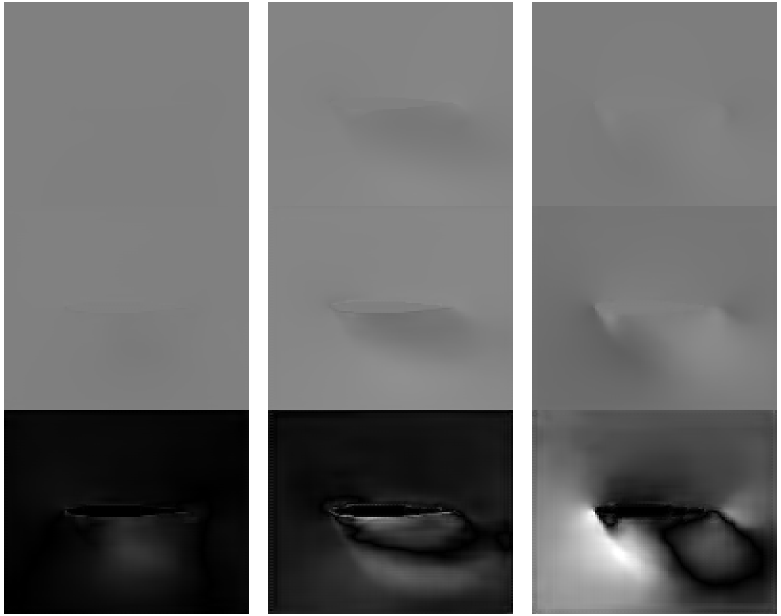


Generalization – $[-67.5, -22.5]$



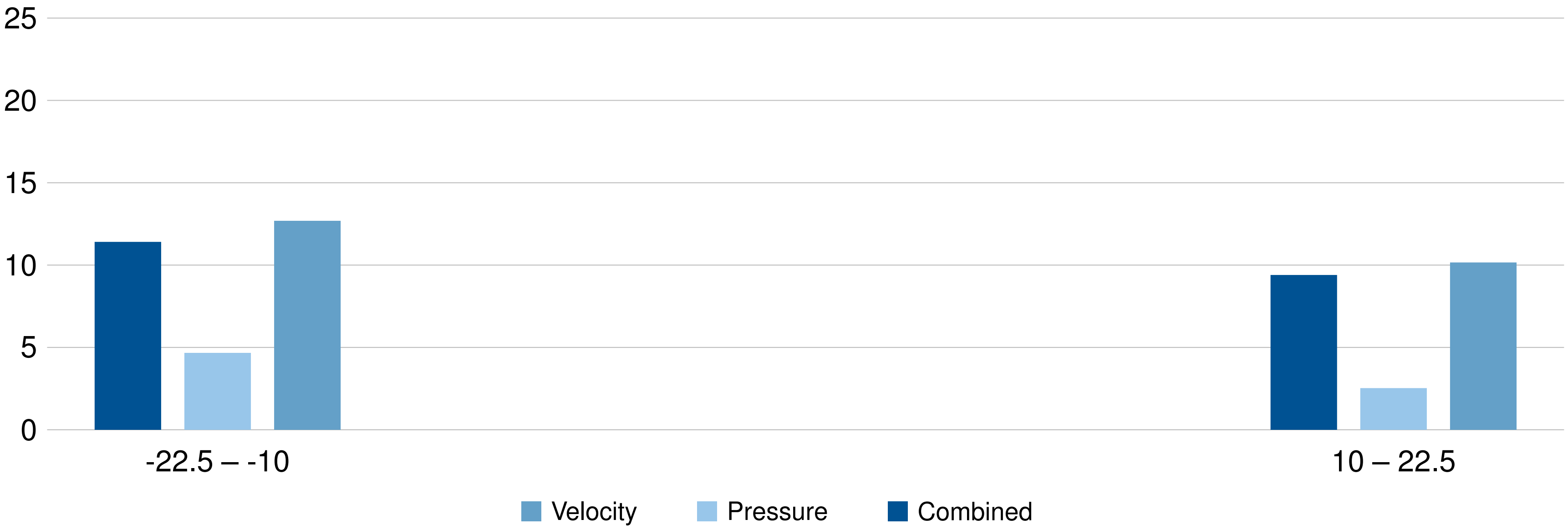
Generalization – $[22.5, 67.5]$



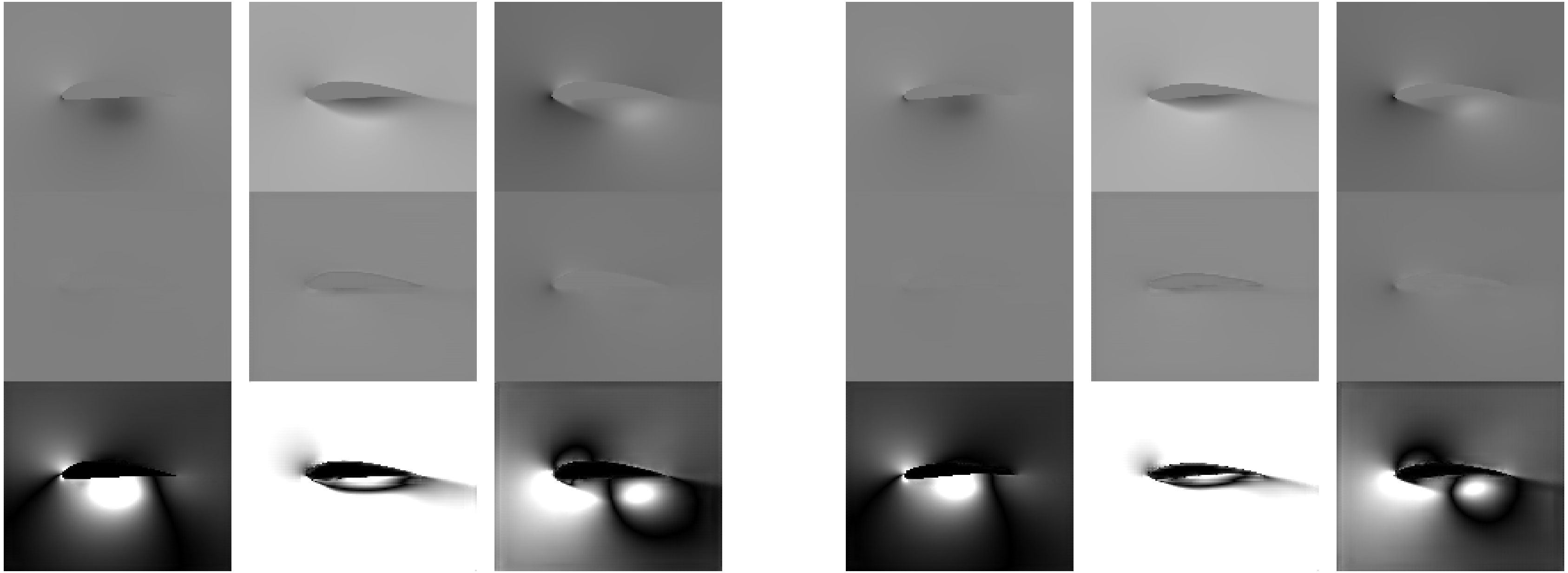


Generalization – Evaluation

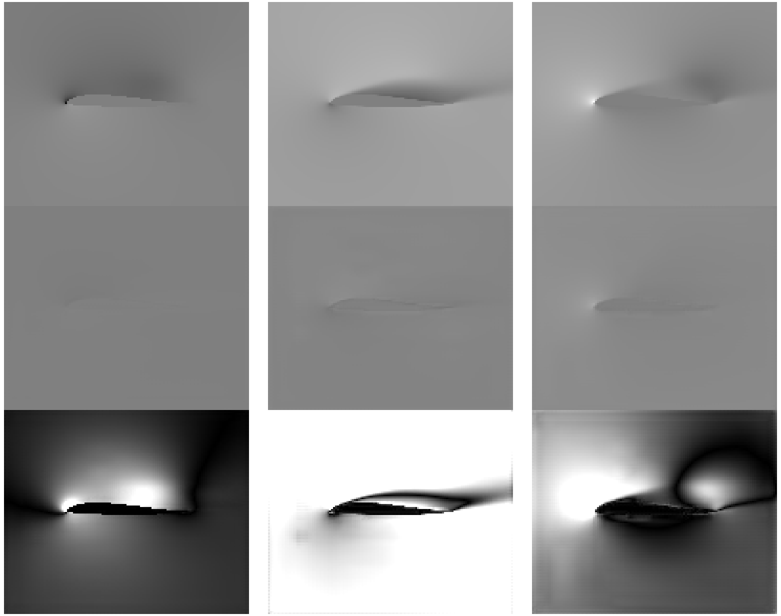
Error increase of different angle of attack interpolations (test set) wrt. ground truth $[-22.5, 22.5]$

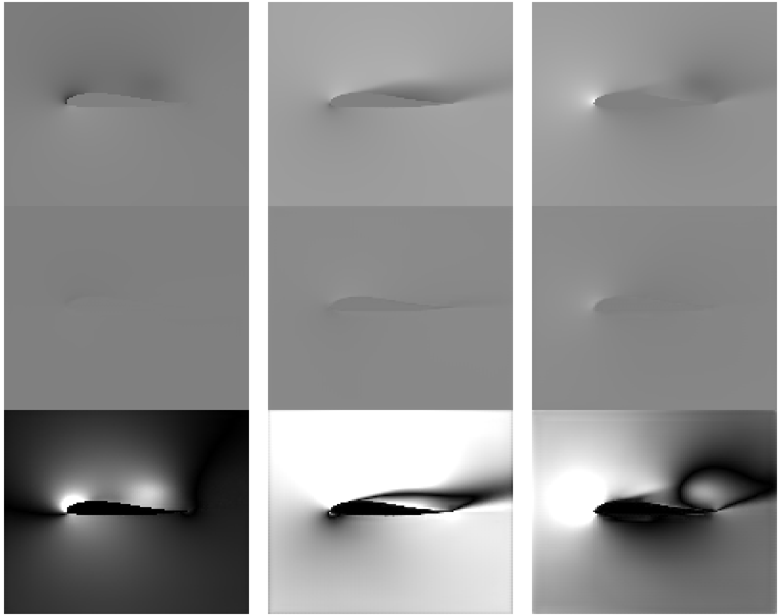


Generalization – training: $[-10, 22.5]$, testing: $[-22.5, -10]$



Generalization – training: $[-22.5, 10]$, testing: $[10, 22.5]$





Discussion

Positive

- Relative error $< 3\%$
- U-Net's catch regions of interest fast and reliable
- U-Net's can outperform LSTM's
- Capacity
- Inference speed ($1000\times$)
- Accuracy improvements possible

Negative

- Prior knowledge
- Solvers needed (data generation)
- Generalization
- Trade off: training speed – grid resolution
- Possible data loss from transformation
- no guarantee for correctness

Summary

Investigate the accuracy of U-Net models for the inference of Reynolds-Averaged Navier-Stokes solutions

Data

- Input (encodes Reynolds number): Bit Mask, x & y velocity
- Target (RANS solution): Pressure, x & y velocity

Pre-Processing

- Make data dimensionless, flatten space of solutions
- Pressure offset removal, numerical precision

Architecture

- U-Net – Encoder - Bottleneck - Decoder structure
- Activations highly depend on task

Transfer

- U-Net as time-series prediction NN for wave propagation
- Input: last *n* frames, Output: next *m* frames, refeed

Generalization

- NN performance on unseen data: Different angels of attack
- underwhelming performance

Discussion

- low error (improvable), speed up, even with low capacity
- prior knowledge and solvers needed, poor generalization

Backup slides

Backup slides – Training Setup

Adam optimizer ($\beta_1 = 0.5, \beta_2 = 0.999$)

Learning rate: 0.0004

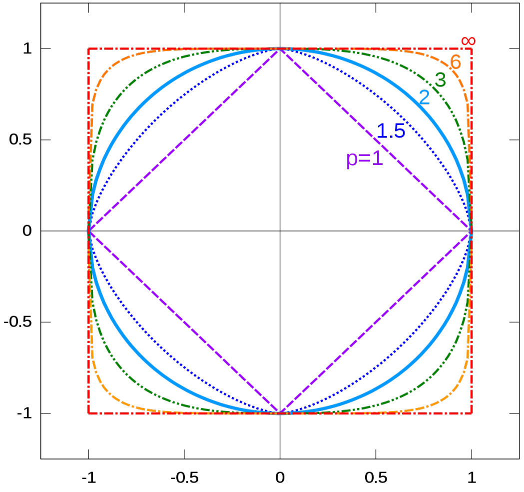
Learning rate decay: On

Batch size: 10

Iterations: 80000

Model parameters:
122.979, 487.107, 1.938.819, 7.736.067, 30.905.859

Backup slides – Norms on unit circle



Taken from: `https://de.wikipedia.org/wiki/Norm_(Mathematik)#/media/Datei:Vector-p-Norms_qtl1.svg`

Backup slides – Navier-Stokes

Navier-Stokes equation for incompressible flow:

$$\frac{\partial u}{\partial t} + (u \cdot \nabla)u - \nu \nabla^2 u = -\nabla(\frac{p}{\rho_0}) + g$$

- u: velocity
- p: pressure
- ν: kinematic viscosity
- ρ_0: uniform density
- g: gravitational acceleration

Backup slides – Saint-Venant

Saint-Venant equations for incompressible flow (1D):

$$\frac{\partial A}{\partial t} + \frac{\partial (Au)}{\partial x} = 0$$
$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial \zeta}{\partial x} = -\frac{P}{A}\frac{\tau}{\rho}$$

- x: coordinate
- t: time
- $A(x,t)$: cross-sectional area of the flow at x
- $u(x,t)$: flow velocity
- $\zeta(x,t)$: free surface elevation
- $\tau(x,t)$: wall shear stress along the wetted perimeter P(x,t) of the cross section at x
- ρ: (constant) fluid density
- g: gravitational acceleration

Backup slides – Previous work

Previous work done in the area of physics simulations with deep learning:

- Accelerating Eulerian Fluid Simulation With Convolutional Networks
  `https://arxiv.org/pdf/1607.03597.pdf`
- tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow
  `https://arxiv.org/pdf/1801.09710.pdf`
- Deep Neural Networks for Data-Driven Turbulence Models
  `https://arxiv.org/pdf/1806.04482.pdf`
- Data-driven discretization: a method for systematic coarse graining of partial differential equations
  `https://arxiv.org/pdf/1808.04930v1.pdf`
- Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data
  `https://arxiv.org/pdf/1808.04327.pdf`