An Optimized Metamodel for **Predicting Oil Outflow**

Tanmoy Das, Floris Goerlandt and Kristjan Tabri

Background

- **Arctic oil spills** can have devastating impact on delicate species & Indigenous community
- Several **factors** for spill response effectiveness: remoteness, harsh arctic conditions e.g. presence of ice, cold temperature

- **Decision Support Tool (DST)**
  - Integrates key elements of spill responses
  - Captures Arctic conditions
  - Leverages data analytics for better response

DST: a computer-based tool equipped with modelling and analytical capabilities

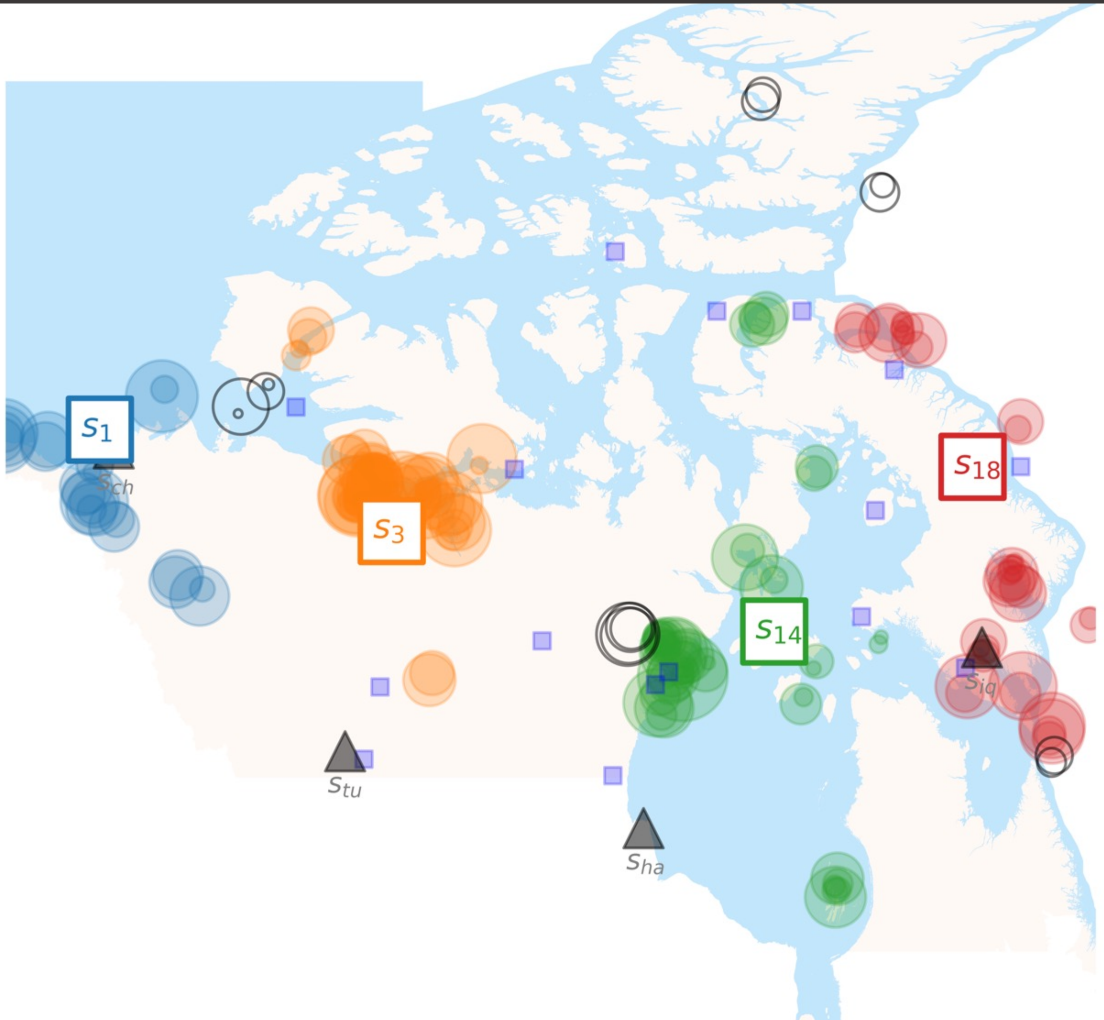


Fig 1. Hypothetical oil spills and response facilities in Canadian Arctic

NNeMPOO model (PI)
Deep Neural Network to Predict Spill size

Dataset

Monte Carlo Simulation to generate data based on ADSAM-C model

Response variables: **Volume of Oil Spill**, Damage Extents

Predict oil spill volume

NNeMPOO: Neural Network Model to **Predict Oil Outflow**

Metamodels

- Deep Neural Network
- Polynomial Regression
- Gradient Boosted Regression Tree

Performance Metric: R^2, RMSE, MAE, Runtime

Multiobjective Optimization

Improve performance of metamodels by tuning hyperparameters

Select Optimal metamodel using Pareto Frontier

Data (Oil Spill Scenarios)

- **Challenges**
  - No relevant data for Canadian Arctic
  - ADSAM-C Engineering model

- **Data Generation**
  1. Distributions of variables are used from literature
  2. ADSAM Tool is applied in a Monte Carlo Simulation

- **Shape of dataset**
  - 11 features
    - $type_A, type_B$
    - $velocity_A, velocity_B$
    - $angle_{collision}$
    - loc_imp_B
    - $disp_A, disp_B$
    - $length_B, width_B$
    - oil_{id}
  - 7 targets
    1. **Volume of spill** (after 15 min, 1hr, and 24hr)
    2. Height and length of inner & outer hull damage
  - 104,000 oil spill scenarios

Sample Dataset

$type_A$	$type_B$	$velocity_A$	$velocity_B$	$angle_{collision}$	loc_imp_B	$disp_A$	$disp_B$	$length_B$	$width_B$	oil_{id}
1	1	10.750	7.11	76.463	133.387	7896141.8	61292607.4	176.54	32.54	764
1	1	5.005	4.55	96.703	89.047	7896141.8	61292607.4	176.54	32.54	764
1	1	6.863	2.55	146.278	111.347	7896141.8	61292607.4	176.54	32.54	908.9
1	1	8.718	5.05	131.717	110.597	7896141.8	61292607.4	176.54	32.54	764
1	1	14.222	4.42	97.889	122.627	7896141.8	61292607.4	176.54	32.54	908.9

$\mathrm{volume}_{oo.15m}$	$\mathrm{volume}_{oo.1hr}$	$\mathrm{volume}_{oo.1day}$	$\mathrm{height}_{d.ih}$	$length_{d.ih}$	$\mathrm{height}_{d.oh}$	$length_{d.oh}$
12860.6257	12860.6257	**12860.62**	7.8544396	3.35171428	7.8544396	3.35171428
1513.19813	2381.59121	**3161.70**	2.43383642	1.61695674	6.2602418	2.58609917
0	0	**0**	0	0	2.7040273	1.51363073
811.747427	811.747427	**811.74**	1.08030608	1.34840256	5.4297069	2.37223512
25721.2515	25721.2515	**25721.25**	7.8544396	3.28974832	7.8544396	3.28974832

Exploratory Data Analysis & Feature Engineering

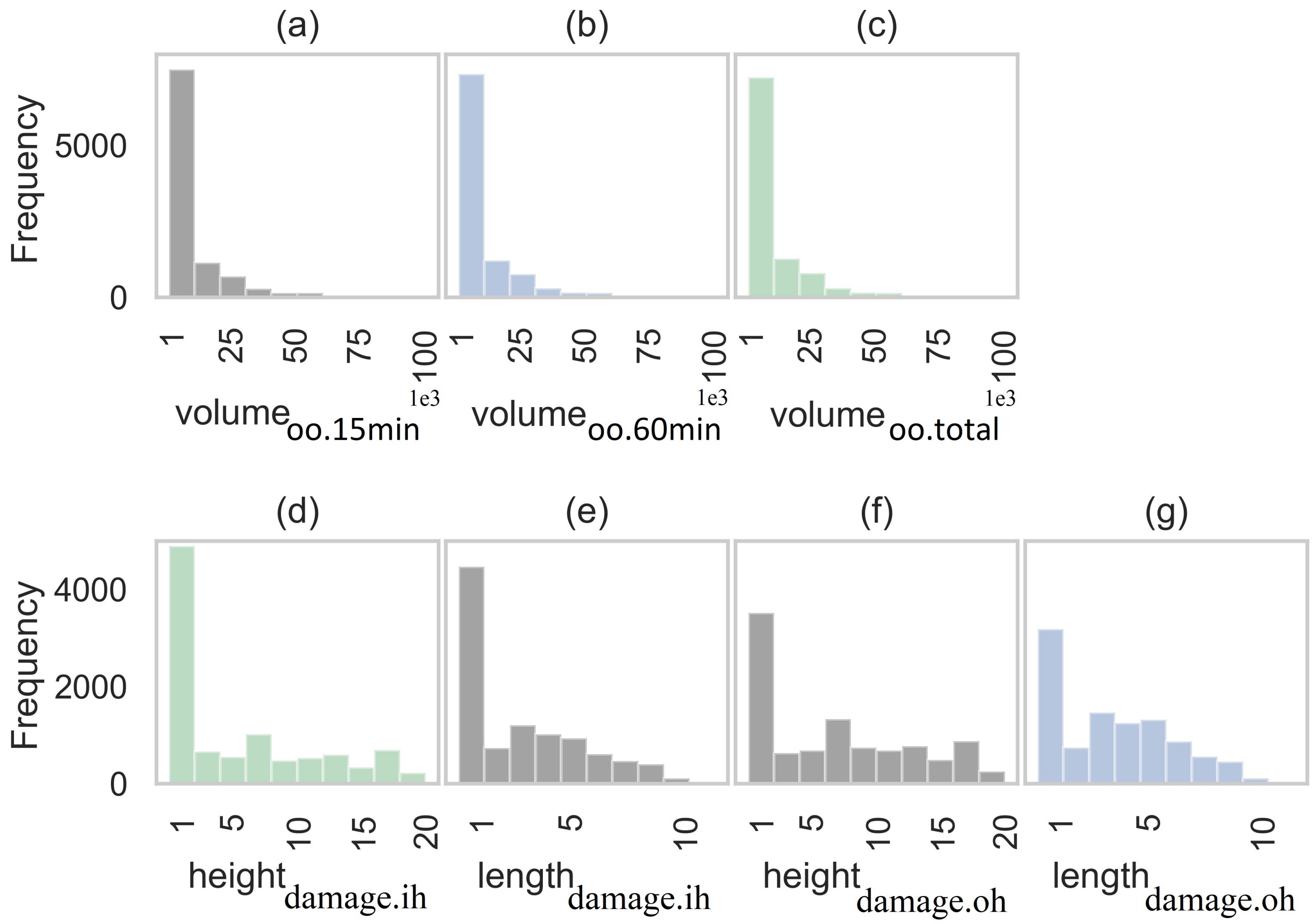


Figure 3. Histograms of outputs

Feature Engineering

- Normalization:
  - all input and output variables are normalized between zero to one before modeling as DNN, PR, and GBRT expect normalized data.[61]
- Encoding
  - Since regression models will not correctly handle categorical inputs, a ordinal encoder is used to convert them into numerical values.

Neural Network Model to predict oil outflow (NNeMPOO)

Modeling Neural Network

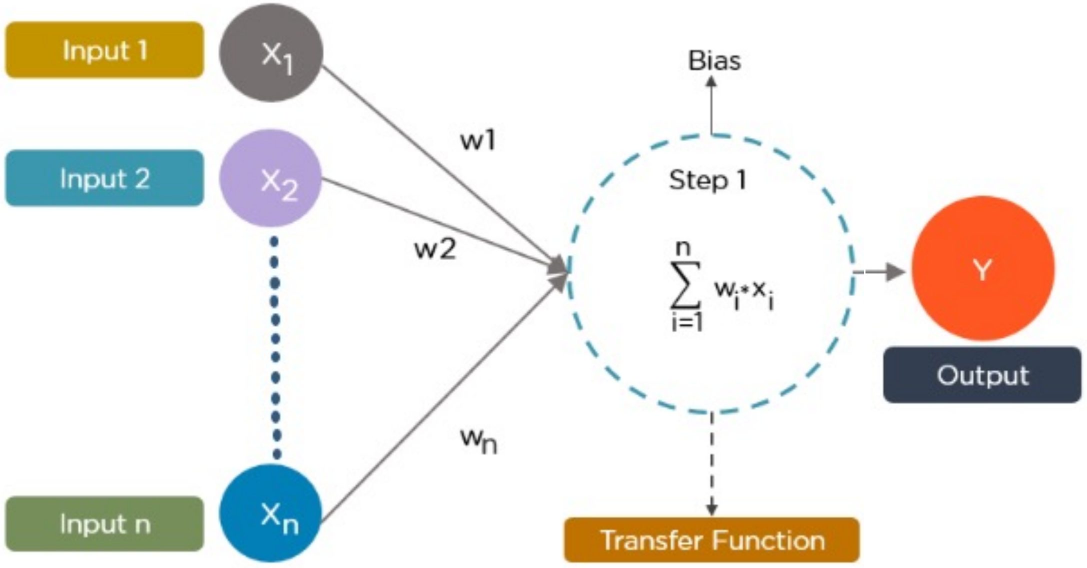
A multilayer perceptron (a Feed forward neural network)

Input layer → Hidden layer → Output

- Training:
  - Minimizing MSE Loss function
  - Weight term: Wi = Wi + (n * (t — o)) Where Wi represents the weight to be updated, n denotes the learning rate, t is the target output, o is the actual output.
  - ReLU Activation function: $f(x) = \max(0, x)$
- Hyperparameter tuning
  - Number of layers, number of neurons, learning rate, activation function, epoch, Alpha
- Select the final architecture

Steps in Training:

1. Initialize the weight vector with random values.
2. Iteratively apply the perceptron to each training example.
3. Modify the perceptron's weights whenever it misclassifies an example.
4. Continue this process until all training examples are correctly classified



PR & GBRT

Polynomial Regression (PR)

- A regression model
- Relationship between inputs and outputs as a polynomial of degree k as shown in Eq. 2 where β is the estimator of the model, x,y,e are the inputs, outputs, and error term.

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \ldots + \beta_k x_i^k + e_i$$

- Hyperparameters*: Number of layers, number of neurons, learning rate, activation function, epoch, Alpha*

Gradient Boosted Regression Tree (GBRT)

- An ensemble technique to sequentially fit a simple model and improve learning (DT is our base learner)
- Iteratively adds weak learners to construct a model $f : X \rightarrow R$

$$f_0(x) = \gamma, \ldots, f_t(x) = f_{t-1}(x) + \eta\, m_t(x)$$

- Hyperparameters: *number of estimators*, the *maximum depth*

Performance Metric

- ***R^2*** - variance of outputs explained by the inputs
- **RMSE** - average difference between values predicted by a model and the actual values, shown in Eq. 4 where n is the sample size; $\hat{y}_i$ is the predicted values for the observed values y_i
- **MAE** - sum of absolute error divide by the sample size
- **Runtime** of the model

- Note: A higher value of R^2 and lower values of RMSE, MAE and runtime represent a better model

$R^2 = 1 - \frac{\textit{unexplained variation}}{\textit{Total variation}}$	Eq. 3
$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$	Eq. 4
$MAE = \frac{1}{n}\sum_{i=1}^{n}\lvert(y_i - \hat{y}_i)\rvert$	Eq. 5

Result

Hyperparameter tuning

	Opt. method	Initial Search Space		
DNN		Number of hidden layers	Learning rate	Activation function
	RS	[128, 64, 32,16,8,2]	linspace(0.020,.25,5)	relu, sigmoid
	TPOT	(64,32),(32,32), (32, 16)	linspace(0.015,.03,10)	relu, sigmoid
	GS	(64,32, 32), (64,64), (64, 32)	0.02, 0.01	relu
	BO	uniform(128, 256)	uniform(0.01, .1)	relu
PR		Degree	Interaction	Include Bias?
	RS	Range(1,10,1)	Yes, No	Yes, No
	GS	3, 4, 5	No	Yes
GBRT		Number of estimator	Learning rate	Max depth
	RS	(2,3,4,5,10,20)	linspace(.01,.8,10)	3,5,10,15,20
	TPOT	100	linspace(.01,.9,100)	range(2, 11,4)
	GS	1,3,4,5,10	0.001, 0.1,.2,0.5,0.9	3,5,10
	BO	uniform(4,20)	uniform(0.001, 0.1)	uniform(1,20)

Better combination of Hyperparameters			
# of hidden layer	Learning rate	Activation func	R^2
(128)	0.02	relu	0.82
(32, 32)	0.015	relu	0.937
(64,32,32)	0.02	relu	0.927
(125)	0.013	relu	0.85
Degree	Interaction	Bias?	R^2
4	No	Yes	0.88
4	No	Yes	0.88
Number of estimator	Learning rate	Max depth	R^2
5	0.448	10	0.91
100	0.12	10	0.89
10	.5	10	0.92
14	0.08	14	0.84

Note: relu: Rectified Linear Unit; sigmoid: a ‘S’ shaped monotonic curve to approximate the activation function of DNN; uniform(min_value, max_value) will produce random values that follow Uniform distribution.

Model evaluation and model Selection

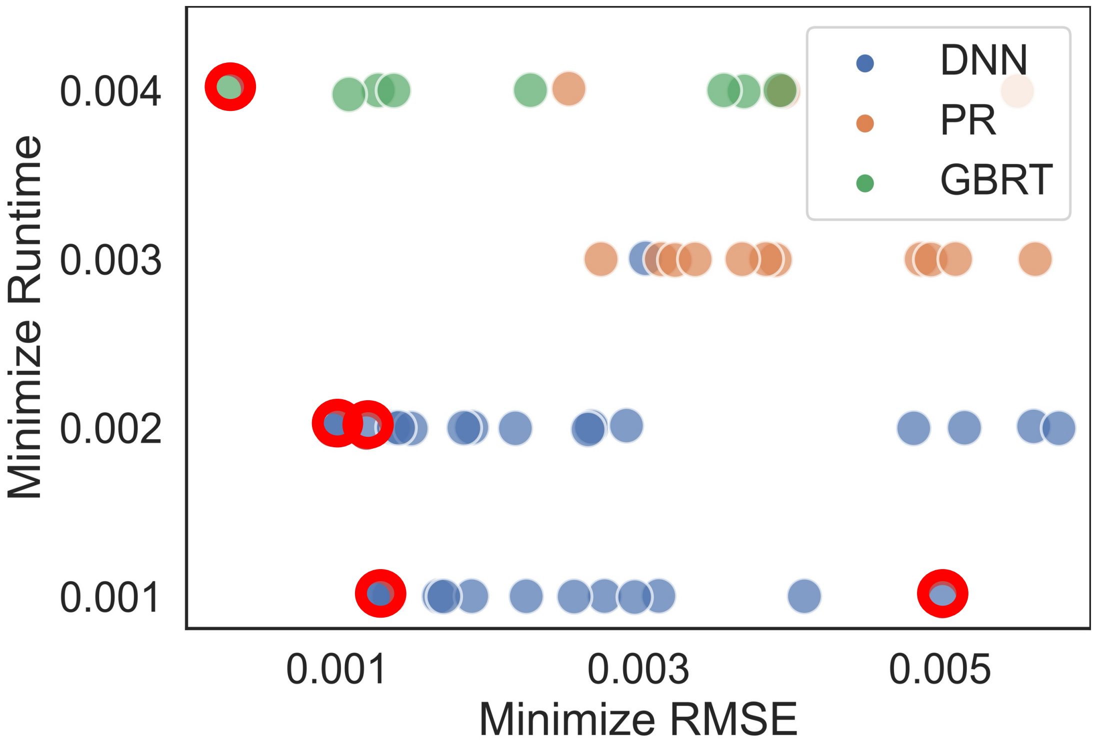


Figure 8. Trade-off between predictive accuracy vs. computational time; red circles construct Pareto Front

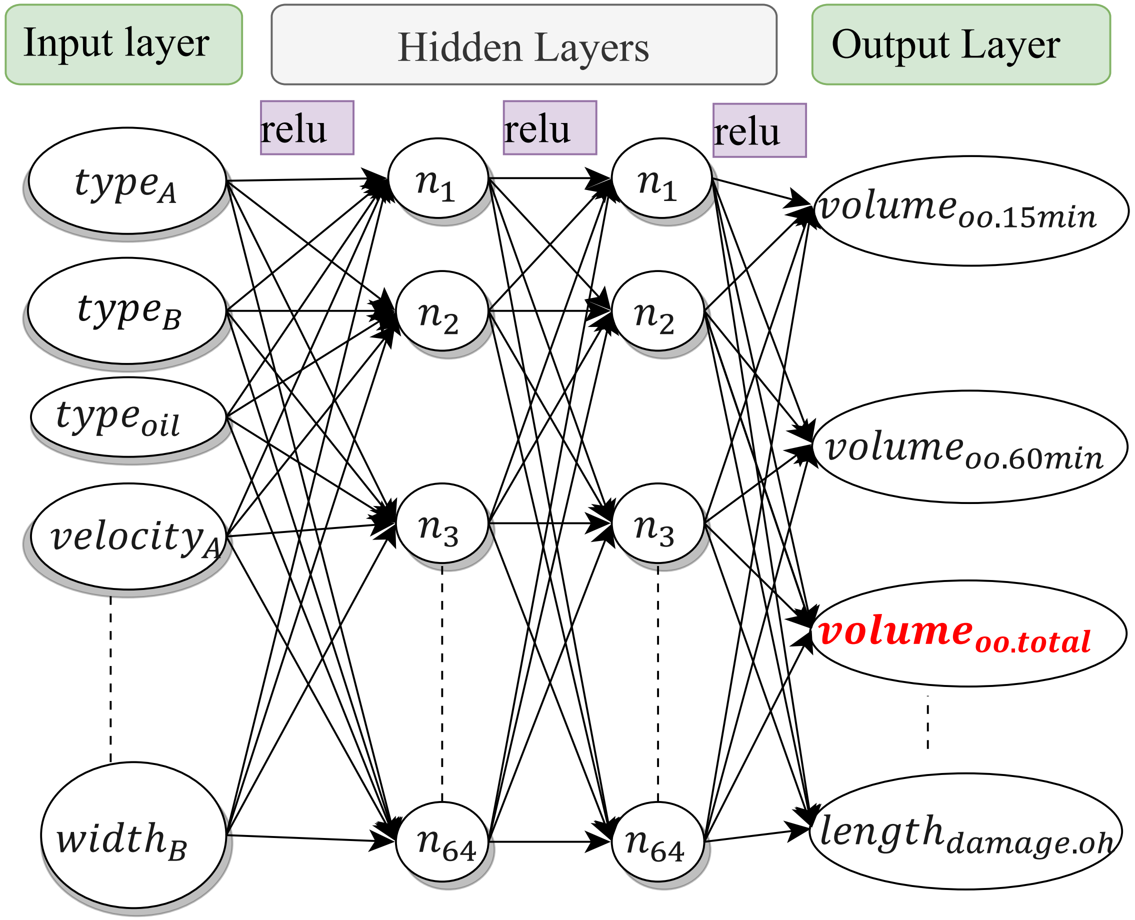


Figure 9. Structure of NNeMPOO model

Managerial Insights

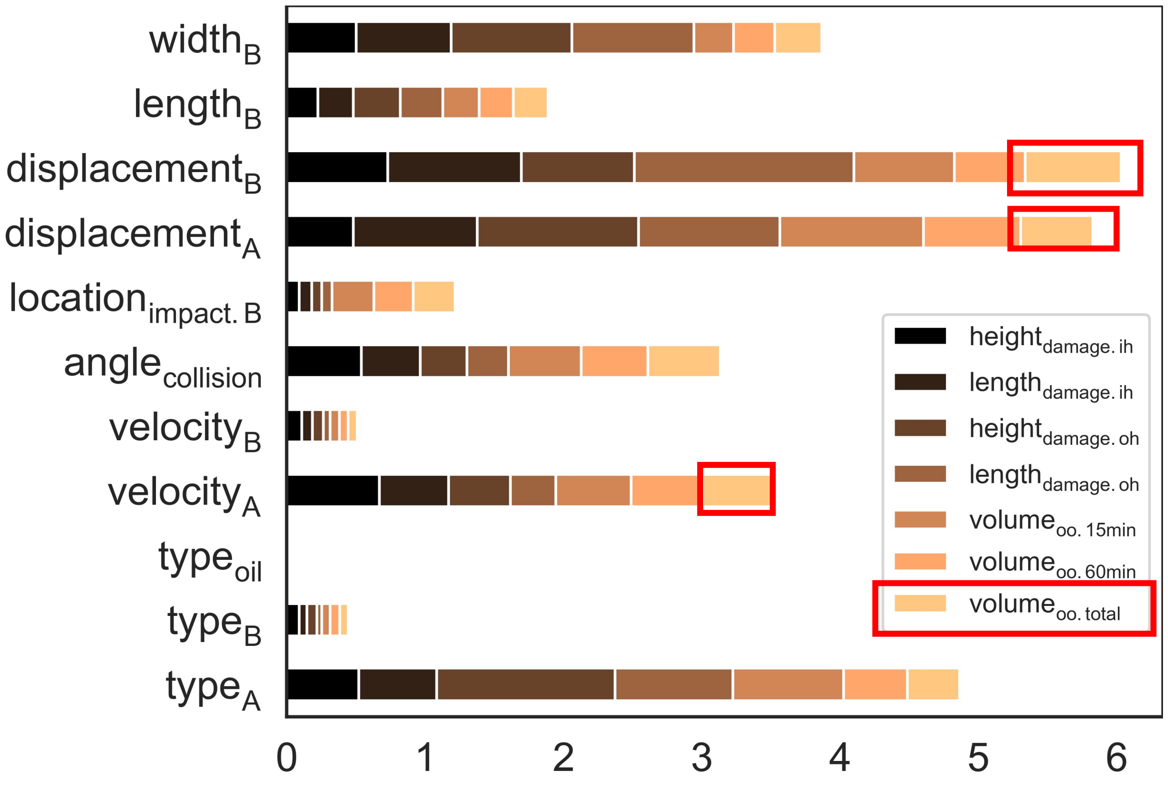


Figure 10. Feature importance plot of DNN metamodeling

- **Implementation**
  - **NNeMPOO** requires significantly fewer input parameters to predict spill size
  - Simple, easy to implement, scalable
- **Speed & Execution**
  - 18x faster than existing engineering model
    - Speed makes the model useful for integration in a strategic risk assessment
- **Impact**
  - A critical sub-model for a DST for Canadian Arctic spill response
  - Source code: https://github.com/tanmoyie/Deep-Neural-Network
  - Published in JEME – Part M (link)