MSc Project

Release 0.1

Ian Wan

Aug 31, 2022

CONTENTS:

CHAPTER

ONE

WAKEMODEL

1.1 WakeModel package

1.1.1 Submodules

1.1.2 WakeModel.CNN_model module

class WakeModel.CNN_model.**Generator**(*nr_input_var*, *nr_filter*)

Bases: Module

The class is the Neural Network that generates the flow field around one or more wind turbines. The network uses the pytorch framwork and uses fully connected and transpose convolutional layers. The methods of this class include the training of the network, testing of the accuracy and generaton of the training data.

static create_floris_dataset(*size*, *u_range*, *ti_range*, *yaw_range*, *u_list=None*, *ti_list=None*, *yawn_list=None*, *TimeGen=True*, *floris_init_path='./'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*, *normalisation=12*, *image_size=163*, *seed=1*)

Function to generate the dataset needed for training using FLORIS. The flowfield around a turbine is generated for a range of wind speeds, turbulent intensities and yaw angles. The 2d array and correspoding init conditions are saved for training.

Function can be used to generated training, validation and test sets.

Parameters

- **size** (*int*) – Size of the dataset
- **image_size** (*int*) – Size of the flow field outputs that are generated, this depends on the neural network used, should be 163.
- **u_range** (*list*) – Bound of u values [u_min, u_max] used.
- **ti_range** (*list*) – Bound of TI values [TI_min, TI_max] used.
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used.
- **u_list** (*list, optional*) – Uniform distribution of wind speed values.
- **ti_list** (*list, optional*) – Uniform distribution of turbulence intensity values.
- **yawn_list** (*list, optional*) – Uniform distribution of yaw angle values.
- **TimeGen** (*bool, optional*) – print out data generation time in seconds if True. Defaults to True.
- **floris_init_path** (*str, optional*) – Path to the FLORIS file. Defaults to "./".

- **floris_name** (*str, optional*) – name of the FLORIS file without .json or .yaml. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json file. False if FLORIS file is yaml file. Defaults to False.
- **normalisation** (*int, optional*) – Normalise CNN output so that wind speed lies between 0 and 1. Defaults to 12.
- **image_size** – Size of the flow field outputs that are generated,
- **163.** (*this depends on the neural network used. Defaults to*) –
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns

Tensor of size (size, image_size, image_size)
which includes all the generated flow fields. The flow fields are normalised to help training

x (torch.tensor): Tensor of size (size, 1, 3) which includes the
flow conditons of the correspoding flow field in the x tensor.

Return type
y (torch.tensor)

static create_pywwake_dataset(*size*, *u_range*, *ti_range*, *yaw_range*, *u_list=None*, *ti_list=None*, *yawn_list=None*, *load_floris=True*, *x_list=[0.0]*, *y_list=[0.0]*, *floris_init_path='./inputs/'*, *floris_name='FLORIS_input_jensen'*, *model='Jensen'*, *TimeGen=True*, *legacy=False*, *image_size=163*, *normalisation=12*, *seed=1*)

Function to generate the dataset needed for training using py_wake. Wind turbine object in py_wake is created from FLORIS input file. Another option is to create in-built py_wake wind turbine object. The flowfield around a turbine is generated for a range of wind speeds, turbulent intensities and yaw angles. The 2d array and correspoding init conditions are saved for training. Squared sum superposition model, Jiminez wake deflection model, and Crespo Hernandez turbulence model are fixed in the wind farm model.

Function can be used to generated training, validation and test sets.

Parameters

- **size** (*int*) – Size of the dataset.
- **u_range** (*list*) – Bound of wind speed values [u_min, u_max] used.
- **ti_range** (*list*) – Bound of turbulence intensity values [TI_min, TI_max] used.
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used.
- **u_list** (*list, optional*) – Uniform distribution of wind speed values.
- **ti_list** (*list, optional*) – Uniform distribution of turbulence intensity values.
- **yawn_list** (*list, optional*) – Uniform distribution of yaw angle values.
- **load_floris** (*bool, optional*) – Load turbine information from FLORIS file if True. Defaults to True.
- **x_list** (*list, optional*) – list of x coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **y_list** (*list, optional*) – list of y coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **floris_init_path** (*str, optional*) – Path to the FLORIS file. Defaults to "./inputs/".

- **floris_name** (*str, optional*) – name of the FLORIS file without .json or .yaml. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json file. False if FLORIS file is yaml file. Defaults to False.
- **model** (*str, optional*) – wake deficit model. Default to "Jensen".
- **TimeGen** (*bool, optional*) – print out data generation time in seconds if True. Defaults to True.
- **image_size** (*int*) – Size of the flow field outputs that are generated,
- **163.** (*this depends on the neural network used. Defaults to*) –
- **normalisation** (*int, optional*) – Normalise CNN output so that wind speed lies between 0 and 1. Defaults to 12.
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns

Tensor of size (size, image_size, image_size)
which includes all the generated flow fields. The flow fields are normalised to help training

x (torch.tensor): Tensor of size (size, 1, 3) which includes the
flow conditons of the correspoding flow field in the x tensor.

Return type
y (torch.tensor)

epoch_training(*criterion*, *optimizer*, *dataloader*, *device*)

Trains the model for one epoch data provided by dataloader. The model will be updated after each batch and the function will return the train loss of the last batch

Parameters

- **criterion** (*torch.nn.criterion*) – Loss function used to train model
- **optimizer** (*torch.optim.Optimizer*) – Optimizer for gradient descent
- **dataloader** (*torch.utils.DataLoader*) – Dataloader to store dataset
- **device** (*torch.device*) – Device on which model/data is stored, cpu or cuda

Returns
Loss of training set defined by criterion

Return type
training loss (float)

error(*x_eval*, *y_eval*, *device*, *image_size=163*)

Calculate the average pixel wise percentage error of the model on a evaluation set. The error function calculates:

$$1/set\_size * \sum_{n=0}^{set\_size} (1/image\_size^2 * \sum_{i=0}^{image\_size^2} (100 * abs(y\_eval_{n,i} - prediction_{n,i})/max(y\_eval_{n,i})))$$

Parameters

- **x_eval** (*torch.tensor*) – Tensor of size (set_size, 1, 3) which includes the flow conditons of the correspoding flow field in the x_eval tensor. set_size is the size of the evaluation set.

- **y_eval** (*torch.tensor*) – Tensor of size (set_size, image_size, image_size) which includes all the generated normalised flow fields. set_size is the size of the evaluation set.
- **device** (*torch.device*) – Device to store and run the neural network on, either cpu or cuda.
- **image_size** (*int, optional*) – Size of the flow field outputs that are generated. Default to 163.

Returns
average pixel wise percentage error

Return type
mean_error (float)

forward(*x*)

Functions defines a forward pass though the network. Can be used for a single input or a batch of inputs

Parameters
x (*torch.tensor*) – input tensor, to be passed through the network

Returns
Output of network

Return type
flow_fields (torch.tensor)

initialize_weights()

Initilize weights using a normal distribution with mean = 0,std2 = 0.02 which has helped training. Loop over all modules, if module is convolutional layer or batchNorm then initialize weights.

Parameters
model (*torch model*) – Neural network model defined using Pytorch

layer(*in_filters*, *out_filters*, *kernel_size*, *stride*, *padding*)

One layer of the CNN which consits of ConvTranspose2d, a batchnorm and LRelu activation function. Function is used to define one layer of the network

Parameters

- **in_filters** (*int*) – Nr. of filters in the previous layer
- **out_filters** (*int*) – Nr. of output filters
- **kernel_size** (*int*) – Size of the ConvTranspose2d layer
- **stride** (*int*) – Stride of the ConvTranspose2d layer
- **padding** (*int*) – Padding used in this layer

Returns
Pytroch Sequential container that defines one layer

Return type
nn.Sequential

load_model(*path='.'*, *device='cpu'*)

Function to load model from a pt file into this class.

Parameters

- **path** (*str, optional*) – path to saved model. Default to '.'.
- **device** (*torch.device, optional*) – Device to load onto, cpu or cuda. Default to 'cpu'.

save_model(*name*, *path*)

Function to save current model paramters so that it can be used again later. Needs to be saved with as .pt file

Parameters

- **name** (*str*) – name of .pt file that stores the model.
- **path** (*str*) – path to directory to which the .pt file is saved.

training: bool

WakeModel.CNN_model.**uniform_distribution**(*size: int*, *param_range: list*, *seed: int = 1*)

Return list of values following uniform distribution given the minimum and maximum values

Parameters

- **size** (*int*) – Size of the uniformly distributed list
- **param_range** (*list*) – Minimum and maximum values of the uniformly distributed list
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns
Uniformly distributed values

Return type
param_list (list)

1.1.3 WakeModel.SetSeed module

WakeModel.SetSeed.**set_seed**(*seed*)

Set all the random seeds to a fixed value to ensure reproducible results.

Parameters
seed (*int*) – Random number seed.

1.1.4 WakeModel.SetParams module

WakeModel.SetParams.**ChangeParams**(*gen: Generator*, *num_layers: int = 1*) → None

Change model parameters num_layers from the last layer to requires_grad=True for transfer learning.

Parameters

- **gen** (*Generator*) – CNN model to be modified.
- **num_layers** (*int, optional*) – number of layers from the last layer which model parameters will be changed. Defaults to 1.

WakeModel.SetParams.**LoadModel**(*model_path: str*, *model_name: str*) → *Generator*

Load a previously saved CNN model to an instantiated Generator object.

Parameters

- **model_path** (*str*) – Path to directory to store the saved model.
- **model_name** (*str*) – Name of the trained saved model (needs be .pt).

Returns
Previously saved CNN model.

Return type
gen (*Generator*)

WakeModel.SetParams.**SplitList**(*param_list: list*, *ratio: float*, *seed: int = 1*) → list

Reduce the size of param_list while preserving the proportion of values over the uniform distribution. Used to maintain consistency of input data for CNN models when the input dataset size is changed.

Parameters

- **param_list** (*list*) – list of uniformly distributed values
- **ratio** (*float*) – new size / original size
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns

list of indices from param_list to construct the new list.
Has length of new size.

Return type
indices (list)

WakeModel.SetParams.**get_params_to_update**(*model:* Generator) → list

Returns list of model parameters that have required_grad=True. Taken from Machine Learning module implementation lecture 4. https://github.com/ese-msc-2021/ML_module/blob/master/implementation/4_CNN/4_CNN_afternoon/Transfer_Learning_Solutions.ipynb

Parameters
model (Generator) – CNN model to be iterated over.

Returns
list of model parameters that have requires_grad=True.

Return type
params_to_update (list)

WakeModel.SetParams.**set_parameter_requires_grad**(*model:* Generator, *requires_grad: bool = False*) → None

Change requires_grad attribute of model parameters. Taken from Machine Learning module implementation lecture 4. https://github.com/ese-msc-2021/ML_module/blob/master/implementation/4_CNN/4_CNN_afternoon/Transfer_Learning_Solutions.ipynb

Parameters

- **model** (Generator) – CNN model to be altered.
- **requires_grad** (*bool, optional*) – torch.autograd record operations if True. Defaults to False.

1.1.5 WakeModel.Pywake module

WakeModel.Pywake.**GetHorizontalGrid**(*wt*, *domain_x=[0, 3000]*, *domain_y=[- 200, 200]*, *dim=163*)

Return the domain as HorizontalGrid object.

Default to $x \in [0, 3000]$ and $y \in [-200, 200]$.

Parameters

- **wt** (*WindTurbine*) – wind turbine object.
- **domain_x** (*list, optional*) – domain of x. Default to [0, 3000].

- **domain_y** (*list, optional*) – domain of y. Default to [-200, 200].
- **dim** (*int, optional*) – dimension of horizontal grid. Default to 163.

Returns

domain of x=[0, 3000], y=[-200,200]
at dimension 163x163.

Return type
grid (HorizontalGrid)

WakeModel.Pywake.**GetTurbineFromFloris**(*floris_path='inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*)

Create a py_wake WindTurbine object from FLORIS file. Assume FLORIS file contains power thrust table, rotor diameter, and hub height.

Parameters

- **floris_path** (*str*) – Path to directory storing FLORIS file. Defaults to "inputs/".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json file. False if FLORIS file is yaml file. Defaults to False.

Returns
py_wake WindTurbine object with physical properties from FLORIS file

Return type
wt (WindTurbine)

WakeModel.Pywake.**GetUniformSite**(*ti*, *ws*, *p_wd=[1]*)

Create a py_wake UniformSite object, a wind turbine site that has constant, uniform speed.

Parameters

- **ti** (*float*) – turbulence intensity.
- **ws** (*float*) – wind speed.
- **p_wd** (*list*) – wind speed probability distribution. Default is [1].

WakeModel.Pywake.**GetWindFromFlowMap**(*flow_map*)

Extract effective wind speed from flow_map.

Parameters
flow_map (*FlowMap*) – effective local wind speed and effective local turbulence intensity at all grid points, generated from simulation.

Returns
effective local wind speed in m/s at all grid points.

Return type
ews (numpy.ndarray)

WakeModel.Pywake.**GetWindfarmLayout**(*floris_path='inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*)

Load x and y coordinates of wind turbines in wind farm from FLORIS file. Assume layout is specified in FLORIS file.

Parameters

- **floris_path** (*str*) – Path to directory storing FLORIS file. Defaults to "inputs/".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json file. False if FLORIS file is yaml file. Defaults to False.

Returns
x and y coordinates of wind turbines respectively.

Return type
x_pos, y_pos (list, list)

WakeModel.Pywake.**PlotFlowMapContour**(*flow_map*)

Create contour plot of effective wind speed from FlowMap object. Uses cmap=coolwarm.

Parameters
flow_map (*FlowMap*) – effective local wind speed and effective local turbulence intensity at all grid points, generated from simulation.

WakeModel.Pywake.**PywakeWorkflow**(*ti*, *ws*, *yaw_angle*, *wd=270*, *load_floris=True*, *x_list=[0.0]*, *y_list=[0.0]*, *floris_path='inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*, *model='Jensen'*, *show_fig=True*)

Full workflow from start to finish: load data from FLORIS file to create wind turbine(s), create a site for the wind turbine(s), simulate airflow and create contour plot.

Parameters

- **ti** (*float*) – turbulence intensity.
- **ws** (*float*) – wind speed.
- **yaw_angle** (*float*) – yaw angle
- **wd** (*float*) – wind direction. Default to 270.
- **load_floris** (*bool, optional*) – Load physical properties of wind turbine from FLORIS file if True. Defaults to True.
- **x_list** (*list, optional*) – list of x coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **y_list** (*list, optional*) – list of y coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **stf** (*bool, optional*) – switch turbulence model to STF2017. Default to False.
- **model** (*str, optional*) – wake deficit model. Default to Jensen.
- **show_fig** (*bool, optional*) – show contour plot of effective wind speed. Default to True.

Returns
effective local wind speed in m/s at all grid points.

Return type
ews (numpy.ndarray)

WakeModel.Pywake.**TestSuperposition**(*ti*, *ws*, *yaw_angle*, *wd=270*, *floris_path='inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*)

Generate flow field plots for different superposition models. STF2017 turbulence model, Jiminez wake deflection model, and Fuga wake deficit model are fixed.

For more details on superposition models, please see: https://topfarm.pages.windenergy.dtu.dk/PyWake/notebooks/EngineeringWindFarmModels.html#Superposition-models

Parameters

- **ti** (*float*) – turbulence intensity.
- **ws** (*float*) – wind speed.
- **yaw_angle** (*float*) – yaw angle
- **wd** (*float, optional*) – wind direction. Default to 270.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.

WakeModel.Pywake.**TestTurbulence**(*ti*, *ws*, *yaw_angle*, *wd=270*, *floris_path='inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*)

Generate flow field plots for different turbulence models. Squared sum superposition model, Jiminez wake deflection model, and Fuga wake deficit model are fixed.

For more details on turbulence models, please see: https://topfarm.pages.windenergy.dtu.dk/PyWake/notebooks/EngineeringWindFarmModels.html#Turbulence-models

Parameters

- **ti** (*float*) – turbulence intensity.
- **ws** (*float*) – wind speed.
- **yaw_angle** (*float*) – yaw angle
- **wd** (*float, optional*) – wind direction. Default to 270.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.

WakeModel.Pywake.**TestWakeDeficit**(*ti*, *ws*, *yaw_angle*, *wd=270*, *floris_path='./inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*)

Generate flow field plots for different wake deficit models. Squared sum superposition model, Jiminez wake deflection model, and Crespo Hernandez turbulence model are fixed.

For more details on wake deficit models, please see: https://topfarm.pages.windenergy.dtu.dk/PyWake/notebooks/EngineeringWindFarmModels.html#Wake-deficit-models

Parameters

- **ti** (*float*) – turbulence intensity.
- **ws** (*float*) – wind speed.
- **yaw_angle** (*float*) – yaw angle
- **wd** (*float, optional*) – wind direction. Default to 270.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs/".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.

1.1.6 WakeModel.TrainFloris module

WakeModel.TrainFloris.**floris_file_v2_to_v3**(*floris_path='./inputs/'*, *floris_name='FLORIS_input_gauss.json'*)

Converts FLORIS file from json to yaml so that it can be loaded into FlorisInterface in FLORIS version 3.

Parameters

- **floris_path** (*str, optional*) – Path to directory storing FLORIS json file. This is be the same path that the yaml file will be saved. Defaults to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS json file. Defaults to "FLORIS_input_gauss.json".

WakeModel.TrainFloris.**train_CNN_floris**(*nr_epochs*, *learning_rate*, *batch_size*, *train_size*, *val_size*, *u_range*, *ti_range*, *yaw_range*, *nr_workers=0*, *ML_model=None*, *x_train=None*, *y_train=None*, *x_eval=None*, *y_eval=None*, *change_params=False*, *num_layers=1*, *floris_path='./inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*, *model_path='./TrainedModels/'*, *model_name='FlorisJensen.pt'*, *save_model=True*, *plot_fig=True*, *train_fig_path='./TrainingFigures/'*, *train_fig_name='FlorisJensen.png'*, *seed=1*, *nr_filters=16*, *image_size=163*)

Create a new model and train it for a certain number of epochs using a newly generated dataset. Dataset is generated from FLORIS. Hyper-parameters such as model size or lr can be changed as input to the function. After training the model error for all epochs is plotted and the model performance will be evaluated on a test set. Finally, the model will saved as the model_name which needs to add as .pt file

Parameters

- **nr_epochs** (*int*) – Nr. of training epochs
- **learing_rate** (*float*) – Model learning rate
- **batch_size** (*int*) – Training batch size
- **train_size** (*int*) – Size of the generated training set
- **val_size** (*int*) – Size of the generated validation set
- **u_range** (*list*) – Bound of u values [u_min, u_max] used

- **ti_range** (*list*) – Bound of TI values [TI_min, TI_max] used
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used
- **nr_workers** (*int, optional*) – Nr. of worker to load data. Defaults to 0.
- **ML_model** (*Generator, optional*) – CNN model passed as an argument. Defaults to None.
- **x_train** (*torch.tensor, optional*) – Tensor of size (train_size, 1, 3) which includes the flow conditons of the correspoding flow field. To be used for training.
- **y_train** (*torch.tensor, optional*) – Tensor of size (train_size, image_size, image_size) which includes all the generated normalised flow fields. To be used for training.
- **x_eval** (*torch.tensor, optional*) – Tensor of size (val_size, 1, 3) which includes the flow conditons of the correspoding flow field. To be used for validation.
- **y_eval** (*torch.tensor, optional*) – Tensor of size (val_size, image_size, image_size) which includes all the generated normalised flow fields. To be used for validation.
- **change_params** (*bool, optional*) – Set requires_grad=False in CNN layers if True. Defaults to False.
- **num_layers** (*int, optional*) – Number of layers left as requires_grad=True. Defaults to 1.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Defaults to "./inputs/".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **model_path** (*str, optional*) – Path to directory to store the saved model.
- **model_name** (*str, optional*) – Name of the trained saved model (needs be .pt).
- **save_model** (*bool, optional*) – Save trained model if True. Defaults to True.
- **plot_fig** (*bool, optional*) – Create plot of validation over epochs. Defaults to True.
- **train_fig_path** (*str, optional*) – Path to directory to store plot of validation errors over epochs during training.
- **train_fig_name** (*str, optional*) – File name of the saved plot.
- **seed** (*int, optional*) – Random number seed. Default to 1.
- **nr_filters** (*int, optional*) – Nr. of filters used for the conv layers. Defaults to 16.
- **image_size** (*int*) – Size of the data set images, needs to match the model output size for the current model this is 163 x 163.

Returns

Trained CNN model. loss (float): Training loss defined by the loss function. val_error (float): Percentage error on the validation set. error_list (list): List of percentage errors on validation set.

Return type

gen (*Generator*)

1.1.7 WakeModel.mftl_floris module

WakeModel.mftl_floris.**MFTL_FLORIS**(*HF_nr_epochs*, *HF_learning_rate*, *HF_batch_size*, *HF_train_size*, *HF_val_size*, *u_range*, *ti_range*, *yaw_range*, *load_model*, *LF_train_size*, *LF_nr_epochs=25*, *LF_learning_rate=0.003*, *LF_batch_size=50*, *LF_val_size=30*, *nr_workers=0*, *image_size=163*, *nr_filters=16*, *nr_input_vars=3*, *floris_path='./inputs/'*, *LF_floris_name='FLORIS_input_jensen'*, *LF_legacy=False*, *HF_floris_name='FLORIS_input_cc'*, *HF_legacy=False*, *save_LF_model=False*, *save_HF_model=True*, *LF_model_name='FLORISJensen.pt'*, *HF_model_name='FLORISJensenFuga.pt'*, *train_fig_name='FLORISJensenFuga.png'*, *train_fig_path='./TrainingFigures/'*, *model_path='./TrainedModels/'*, *load_model_path='./TrainedModels/'*, *seed=1*)

Implementation of multi-fidelity transfer learning: train a CNN model with low-fidelity data, then re-train the same CNN model with high-fidelity data. Both datasets are generated using FLORIS. The final trained model and the plot of validation error over epochs is saved.

Parameters

- **HF_nr_epochs** (*int*) – Nr. of training epochs for high-fidelity data.
- **HF_learning_rate** (*float*) – Model learning rate while training using high-fidelity data.
- **HF_batch_size** (*int*) – Batch size while training using high-fidelity data.
- **HF_train_size** (*int*) – Size of the generated high-fidelity training set.
- **HF_val_size** (*int*) – Size of the generated high-fidelity validation set.
- **u_range** (*list*) – Bound of u values [u_min, u_max] used.
- **ti_range** (*list*) – Bound of TI values [TI_min, TI_max] used.
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used.
- **load_model** (*bool*) – True if model is loaded from a .pt file.
- **LF_train_size** (*int*) – Size of the generated low-fidelity training set.
- **LF_nr_epochs** (*int, optional*) – Nr. of training epochs for low-fidelity data. Defaults to 25.
- **LF_learning_rate** (*float, optional*) – Model learning rate while training using low-fidelity data. Defaults to 0.003.
- **LF_batch_size** (*int, optional*) – Batch size while training using low-fidelity data. Defaults to 50.
- **LF_val_size** (*int, optional*) – Size of the generated low-fidelity validation set. Defaults to 30.
- **nr_workers** (*int, optional*) – Nr. of worker to load data. Defaults to 0.
- **image_size** (*int, optional*) – Size of the data set images, needs to match the model output size for the current model, which is 163 x 163.
- **nr_filters** (*int, optional*) – Nr. of filters used for the conv layers. Defaults to 16.
- **nr_input_vars** (*int, optional*) – Nr. of input variables. Defaults to 3.

- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Defaults to "./inputs/".
- **LF_floris_name** (*str, optional*) – Name of the low-fidelity FLORIS file without .json or .yaml. Defaults to "FLORIS_input_jensen".
- **LF_legacy** (*bool, optional*) – True if low-fidelity FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **HF_floris_name** (*str, optional*) – Name of the high-fidelity FLORIS file without .json or .yaml. Defaults to "FLORIS_input_cc".
- **HF_legacy** (*bool, optional*) – True if high-fidelity FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **save_LF_model** (*bool, optional*) – Save the low-fidelity model if True. Defaults to False.
- **save_HF_model** (*bool, optional*) – Save the high-fidelity model if True. Defaults to True.
- **LF_model_name** (*str, optional*) – Name of the saved low-fidelity model (needs to be .pt).
- **HF_model_name** (*str, optional*) – Name of the saved high-fidelity model (needs to be .pt).
- **train_fig_name** (*str, optional*) – File name of the saved plot of validation error over epochs during training.
- **train_fig_path** (*str, optional*) – Path to directory to store plot of validation errors over epochs.
- **model_path** (*str, optional*) – Path to directory to store the saved CNN model.
- **load_model_path** (*str, optional*) – Path to directory to the saved CNN model to be loaded.
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns
CNN model trained with low-fidelity data. HFgen (Generator): CNN model undergone multi-fidelity transfer learning.

Return type
LFgen (*Generator*)

1.1.8 WakeModel.TrainPywake module

WakeModel.TrainPywake.**train_CNN_pywake**(*nr_epochs*, *learning_rate*, *train_size*, *val_size*, *batch_size*, *u_range*, *ti_range*, *yaw_range*, *u_list=None*, *ti_list=None*, *yaw_list=None*, *x_train=None*, *y_train=None*, *x_eval=None*, *y_eval=None*, *x_list=[0.0]*, *y_list=[0.0]*, *nr_input_vars=3*, *nr_filters=16*, *image_size=163*, *nr_workers=0*, *ML_model=None*, *change_params=False*, *num_layers=1*, *load_floris=True*, *floris_path='./inputs/'*, *floris_name='FLORIS_input_jensen_1x3'*, *legacy=False*, *model='Jensen'*, *model_path='./TrainedModels/'*, *model_name='PywakeJensen.pt'*, *save_model=True*, *plot_fig=True*, *train_fig_path='./TrainingFigures/'*, *train_fig_name='PywakeJensen.png'*, *seed=1*)

Create a new model and train it for a certain number of epochs using a newly generated dataset. Dataset is generated using py_wake. Hyper-parameters such as model size or learning rate can be changed as input to the function. After training the model error for all epochs is plotted and the model performance will be evaluated on a test set. Finally, the model will saved as the model_name which needs to add as .pt file

Parameters

- **nr_epochs** (*int*) – Nr. of training epochs
- **learing_rate** (*float*) – CNN learning rate.
- **train_size** (*int*) – Size of the generated training set
- **val_size** (*int*) – Size of the generated validation set
- **batch_size** (*int*) – Training batch size
- **u_range** (*list*) – Bound of wind speed values [u_min, u_max] used.
- **ti_range** (*list*) – Bound of turbulence intensity values [TI_min, TI_max] used.
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used.
- **u_list** (*list, optional*) – Uniform distribution of wind speed values.
- **ti_list** (*list, optional*) – Uniform distribution of turbulence intensity values.
- **yaw_list** (*list, optional*) – Uniform distribution of yaw angle values.
- **x_train** (*torch.tensor, optional*) – Tensor of size (train_size, 1, 3) which includes the flow conditons of the correspoding flow field. To be used for training.
- **y_train** (*torch.tensor, optional*) – Tensor of size (train_size, image_size, image_size) which includes all the generated normalised flow fields. To be used for training.
- **x_eval** (*torch.tensor, optional*) – Tensor of size (val_size, 1, 3) which includes the flow conditons of the correspoding flow field. To be used for validation.
- **y_eval** (*torch.tensor, optional*) – Tensor of size (val_size, image_size, image_size) which includes all the generated normalised flow fields. To be used for validation.
- **x_list** (*list, optional*) – list of x coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **y_list** (*list, optional*) – list of y coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **nr_input_vars** (*int, optional*) – Nr. of input variables. Defaults to 3.
- **nr_filters** (*int, optional*) – Nr. of filters used for the conv layers. Defaults to 16.
- **image_size** (*int, optional*) – Size of the data set images, needs to match the model output size for the current model, which is 163 x 163.
- **nr_workers** (*int, optional*) – Nr. of worker to load data. Defaults to 0.
- **ML_model** (Generator, *optional*) – CNN model passed as an argument. Defaults to None.
- **change_params** (*bool, optional*) – Set requires_grad=False in CNN layers if True. Defaults to False.
- **num_layers** (*int, optional*) – Number of layers left as requires_grad=True. Defaults to 1.

- **load_floris** (*bool, optional*) – Load turbine information from FLORIS file if True. Defaults to True.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Defaults to "./inputs/".
- **floris_name** (*str, optional*) – Name of the FLORIS file without .json or .yaml. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **model** (*str, optional*) – Wake deficit model. Default to "Jensen".
- **model_path** (*str, optional*) – Path to directory to store the saved model.
- **model_name** (*str, optional*) – Name of the trained saved model (needs be .pt).
- **save_model** (*bool, optional*) – Save the trained model to model_path with model_name if True. Defaults to True.
- **plot_fig** (*bool, optional*) – Plot validation error over epochs and save plot. Defaults to True.
- **train_fig_path** (*str, optional*) – Path to directory to store plot of validation errors over epochs during training.
- **train_fig_name** (*str, optional*) – File name of the saved plot of validation error over epochs.
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns

Trained CNN model. loss (float): Training loss defined by the loss function. val_error (float): Percentage error on the validation set. error_list (list): List of percentage errors on validation set.

Return type

gen (*Generator*)

1.1.9 WakeModel.mftl_pywake module

WakeModel.mftl_pywake.**MFTL_PyWake**(*HF_nr_epochs*, *HF_learning_rate*, *HF_batch_size*, *HF_train_size*, *HF_val_size*, *u_range*, *ti_range*, *yaw_range*, *load_model*, *LF_train_size*, *LF_nr_epochs=25*, *LF_learning_rate=0.003*, *LF_batch_size=50*, *LF_val_size=30*, *nr_workers=0*, *image_size=163*, *nr_filters=16*, *nr_input_vars=3*, *load_floris=True*, *x_list=[0.0]*, *y_list=[0.0]*, *floris_path='./inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*, *LF_model='Jensen'*, *LF_model_name='PywakeJensen.pt'*, *save_LF_model=False*, *HF_model='Fuga'*, *HF_model_name='PywakeJensenFuga.pt'*, *save_HF_model=True*, *train_fig_name='PywakeJensenFuga.png'*, *train_fig_path='./TrainingFigures/'*, *model_path='./TrainedModels/'*, *load_model_path='./TrainedModels/'*, *seed=1*)

Implementation of multi-fidelity transfer learning: train a CNN model with low-fidelity data, then re-train the same CNN model with high-fidelity data. Both datasets are generated using py_wake. The final trained model and the plot of validation error over epochs is saved.

Parameters

- **HF_nr_epochs** (*int*) – Nr. of training epochs for high-fidelity data.

- **HF_learning_rate** (*float*) – Model learning rate while training using high-fidelity data.
- **HF_batch_size** (*int*) – Batch size while training using high-fidelity data.
- **HF_train_size** (*int*) – Size of the generated high-fidelity training set.
- **HF_val_size** (*int*) – Size of the generated high-fidelity validation set.
- **u_range** (*list*) – Bound of u values [u_min, u_max] used.
- **ti_range** (*list*) – Bound of TI values [TI_min, TI_max] used.
- **yaw_range** (*list*) – Bound of yaw angles [yaw_min, yaw_max] used.
- **load_model** (*bool*) – True if model is loaded from a .pt file.
- **LF_train_size** (*int*) – Size of the generated low-fidelity training set.
- **LF_nr_epochs** (*int, optional*) – Nr. of training epochs for low-fidelity data. Defaults to 25.
- **LF_learning_rate** (*float, optional*) – Model learning rate while training using low-fidelity data. Defaults to 0.003.
- **LF_batch_size** (*int, optional*) – Batch size while training using low-fidelity data. Defaults to 50.
- **LF_val_size** (*int, optional*) – Size of the generated low-fidelity validation set. Defaults to 30.
- **nr_workers** (*int, optional*) – Nr. of worker to load data. Defaults to 0.
- **image_size** (*int, optional*) – Size of the data set images, needs to match the model output size for the current model, which is 163 x 163.
- **nr_filters** (*int, optional*) – Nr. of filters used for the conv layers. Defaults to 16.
- **nr_input_vars** (*int, optional*) – Nr. of input variables. Defaults to 3.
- **load_floris** (*bool, optional*) – Load turbine information from FLORIS file if True. Defaults to True.
- **x_list** (*list, optional*) – list of x coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **y_list** (*list, optional*) – list of y coordinates of non-FLORIS wind turbines. Defaults to [0.].
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Defaults to "./inputs/".
- **floris_name** (*str, optional*) – Name of the FLORIS file without .json or .yaml. Defaults to "FLORIS_input_jensen".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **LF_model** (*str, optional*) – Low-fidelity model used to generate low-fidelity training and validation set. Defaults to Jensen.
- **LF_model_name** (*str, optional*) – Name of the saved low-fidelity model (needs to be .pt).
- **save_LF_model** (*bool, optional*) – Save the low-fidelity model if True. Defaults to False.

- **HF_model** (*str, optional*) – High-fidelity model used to generate high-fidelity training and validation set. Defaults to Fuga.
- **HF_model_name** (*str, optional*) – Name of the saved high-fidelity model (needs to be .pt).
- **save_HF_model** (*bool, optional*) – Save the high-fidelity model if True. Defaults to True.
- **train_fig_name** (*str, optional*) – File name of the saved plot of validation error over epochs during training.
- **train_fig_path** (*str, optional*) – Path to directory to store plot of validation errors over epochs.
- **model_path** (*str, optional*) – Path to directory to store the saved model.
- **load_model_path** (*str, optional*) – Path to directory to the saved CNN model to be loaded.
- **seed** (*int, optional*) – Random number seed. Default to 1.

Returns

CNN model trained with low-fidelity data. HFgen (Generator): CNN model undergone multi-fidelity transfer learning.

Return type

LFgen (*Generator*)

1.1.10 WakeModel.VisualiseFlows module

WakeModel.VisualiseFlows.**CompareFlows**(*u*, *ti*, *yaw_angle*, *package='py_wake'*, *pywake_model='Jensen'*, *nr_input_var=3*, *nr_filters=16*, *image_size=163*, *model_path='./TrainedModels/'*, *model_name='FlorisJensen.pt'*, *load_floris=True*, *x_list=[0.0]*, *y_list=[0.0]*, *floris_path='./inputs/'*, *floris_name='FLORIS_input_jensen'*, *legacy=False*, *fig_path='./results/'*, *fig_name='FlorisJensen.png'*, *show_fig=False*, *save_fig=True*)

Loads a trained model from a .pt file, use the model and selected package to predict the flow field around the wind turbine. Contour plots of flow fields from package, model and relative difference are shown.

Parameters

- **u** (*float*) – wind speed in m/s.
- **ti** (*float*) – turbulence intensity.
- **yaw_angle** (*float*) – yaw angle.
- **package** (*str*) – Package used for training the model. Either 'FLORIS' or 'py_wake'. Default to 'py_wake'.
- **pywake_model** (*str, optional*) – model used to generate flow field in py_wake.
- **nr_input_var** (*int, optional*) – Nr. of inpu variables for the model. Default to 3.
- **nr_filters** (*int, optional*) – Nr. of filters used for the conv layers. Default to 16.
- **image_size** (*int*) – Size of the data set images, needs to match the model output size which is 163 x 163
- **model_path** (*str, optional*) – Path to directory to store the saved model

- **model_name** (*str*) – Name of the trained saved model (needs be .pt)
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **legacy** (*bool, optional*) – True if FLORIS file is json. False if FLORIS file is yaml. Defaults to False.
- **fig_path** (*str, optional*) – Path to directory to store the figure
- **fig_name** (*str, optional*) – File name of the stored figure
- **show_fig** (*bool, optional*) – Show figure on window if True. Defaults to False.
- **save_fig** (*bool, optional*) – Save figure to fig_path with fig_name. Defaults to True.

Returns

flow field prediction
from FLORIS and model respectively

Return type
y, prediction (numpy.ndarray, numpy.ndarray)

WakeModel.VisualiseFlows.**LF_trained_CNN**(*model_list*, *u*, *ti*, *yaw*, *model_path*, *floris_path*, *floris_name*, *fig_path*)

Create figures for CNNs that has been trained using a single model py_wake package. Compare flow fields generated by those CNNs with flow fields generated from py_wake. Assume turbines are loaded from FLORIS input files. Used for processing experiment runs.

Parameters

- **LF_list** (*list*) – low-fidelity py_wake models.
- **u** (*float*) – wind speed in m/s.
- **ti** (*float*) – turbulence intensity.
- **yaw_angle** (*float*) – yaw angle.
- **model_path** (*str, optional*) – Path to directory to store the saved model.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **fig_path** (*str*) – Path to directory to store the figure.

WakeModel.VisualiseFlows.**mftl_CNN**(*LF_list*, *u*, *ti*, *yaw*, *model_path*, *floris_path*, *floris_name*, *fig_path*, *HF_model='Fuga'*)

Create figures for CNNs that has undergone transfer learning using py_wake package. Compare flow fields generated by those CNNs with flow fields generated from py_wake Fuga. Assume turbines are loaded from FLORIS input files. Used for processing experiment runs.

Parameters

- **LF_list** (*list*) – low-fidelity py_wake models.
- **u** (*float*) – wind speed in m/s.
- **ti** (*float*) – turbulence intensity.

- **yaw_angle** (*float*) – yaw angle.
- **model_path** (*str, optional*) – Path to directory to store the saved model.
- **floris_path** (*str, optional*) – Path to directory storing FLORIS file. Default to "./inputs".
- **floris_name** (*str, optional*) – name of the FLORIS file. Defaults to "FLORIS_input_gauss".
- **fig_path** (*str*) – Path to directory to store the figure.
- **HF_model** (*sstr, optional*) – high-fidelity model used in transfer learning. Defaults to Fuga.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

W

INDEX

U

W