

# Code Description:

## Optimization Proxies using Limited Labeled Data and Training Time—A Semi-Supervised Bayesian Neural Network Approach

### Overview

This document describes the implementation of a **Bayesian Neural Network (BNN)** for solving **Optimal Power Flow (OPF)** problems. The model uses a semi-supervised learning approach to optimize power grid operations by leveraging uncertainty quantification and integrating unlabeled data. The code supports various power grid datasets and configurations for training and testing.

### Usage

You can run the main script by executing:

```
python src/main.py [OPTIONS]
```

For Analysis:

```
python src/analysis.py
```

### Options

The table below describes the available command-line options, which can be obtained using

```
python src/main.py --help
```

Table 1: Command-line Options for BNN-OPF

Option	Short	Type	Description
--datapath	-p	TEXT	Path to the dataset directory.
--case	-c	TEXT	Power grid case to solve, e.g., <code>pglib_opf_case118_ieee</code> .
--config	-o	TEXT	Path to the configuration file.
--numgroups	-n	INTEGER	Number of data groups, each with 15,000 points.
--train	-r	INTEGER	Number of training points per group.
--test	-e	INTEGER	Number of testing points per group.
--runtype		TEXT	Type of run (e.g., <code>semisupervisedBNN</code> ).
--trackloss		Flag	Track and save all losses.
--debug		Flag	Enable debug mode.
--warn		Flag	Enable warning mode.
--error		Flag	Enable error flag.
--onlydl		Flag	Only download the data and exit.
--install-completion		Flag	Install shell completion for the current shell.
--show-completion		Flag	Show shell completion options.
--help		Flag	Show help information and exit.

### Example

To run the program using a specific dataset, case, and configuration file:

```
python src/main.py --case pglib_opf_case118_ieee --config config.json --train 512
```

This example uses the `pglib_opf_case118_ieee` dataset with 512 training points for training Sandwich BNN.

```

config.json

{
  "batch_size": 1024,
  "initial_learning_rate": 1e-3,
  "decay_rate": 1e-4,
  "max_epochs": 200000,
  "sandwich_rounds": 10,
  "max_training_time": 600,
  "early_stopping_trigger_supervised": 25,
  "early_stopping_trigger_unsupervised": 30,
  "patience_supervised": 3,
  "patience_unsupervised": 5
}

```

## Requirements

The project dependencies are specified as follows:

```

python = "^3.11,<3.13"
typer = "^0.12.3"
logging = "^0.4.9.6"
numpy = "^1.26.0"
scipy = "^1.14.0"
gridx-egret = {path = "Egret"}
torch_geometric =
{git = "https://github.com/pyg-team/pytorch_geometric.git", branch="master"}
tqdm = "^4.66.5"
numpyro = "^0.15.2"
optax = "^0.2.3"
scikit-learn = "^1.5.1"

```

## License

This project is licensed under the MIT License. See the LICENSE file for details.