EnerGNN

A Graph Neural Network library for real-life complex Energy systems.

Dr. Balthazar Donon, RTE R&D

14.04.2026 | LFE TAC

EnerGNN

Current Status

- License MPL-2.0.
- Wish to enter LFE at Sandbox stage.
- Clean code.
- Comprehensive Sphinx documentation.
- Integration and unit tests.

EnerGNN

Current GNN libraries have not been designed for large-scale and real-life complex industrial systems.

Complex Data. Industrial systems are hyper heterogeneous multi graphs.

Varying Structure. Outages, infrastructure expansion and object renaming / reordering alter the topological structure of data over time.

Learning Modalities. If labelled data are missing (e.g., for complex optimization problems), self-supervision is a promising avenue for training (see Amortized Optimization).

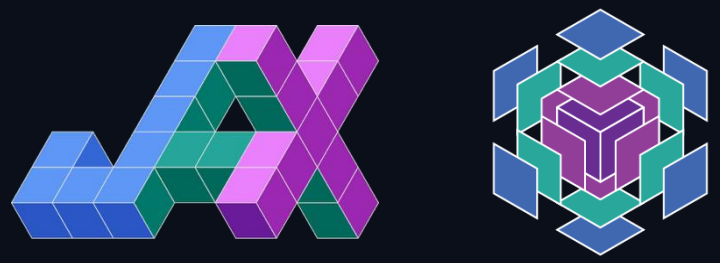
EnerGNN

Goal

Bridge the gap between research and industrial operation of energy networks.

Ecosystem

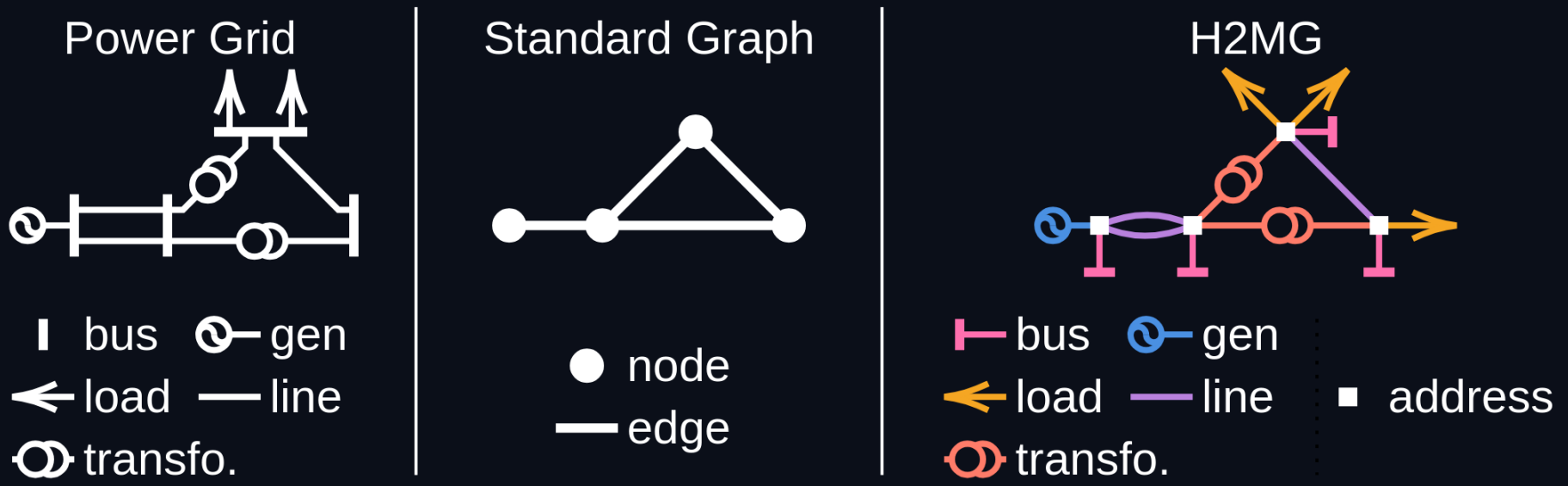
Built on JAX and Flax for performance, automatic differentiation and scalability.



EnerGNN

Industrial systems are not simple graphs, they are:

- Hyper graphs. Made of hyper-edges that can connect more than 2 entities.
- Heterogeneous graphs. Multiple component types (lines, transformers, etc.).
- Multi graphs. Multiple components can be collocated.



EnerGNN

API Overview

EnerGNN

energnn.graph

- Core data representation API designed for H2MG.
- Transmission grid data are seamlessly imported from *PyPowSyBl*.

```
>>> print(graph)
Mass
          addresses  features
            node_id    weight         x         y         z
object_id
0               0.0  5.322265  0.202435  0.202435  0.242032
1               1.0  3.496568  0.962326  0.962326  0.306690
2               2.0  3.535864  0.060886  0.060886  0.094170
3               3.0  7.213709  0.984766  0.984766  0.068853
Spring
          addresses           features
           node1_id node2_id         k
object_id
0               0.0      1.0  0.020424
1               1.0      2.0  0.037591
2               2.0      3.0  0.045405
Registry
[0. 1. 2. 3.]
```

EnerGNN

`energnn.model`

Modular and parametrizable GNN library designed for H2MG data, robust to structure variations, and made of interchangeable `flax.nnx.Module` bricks.

1. **Normalizer.** Robust handling of real-world multi-modal and multi-scale data distributions.
2. **Encoder.** Embeds input features into a latent space.
3. **Coupler.** Propagates information throughout the input graph (message passing, neural ode, etc.).
4. **Decoder.** Converts latent variables into meaningful predictions.

EnerGNN

energnn.problem

Interface for business-driven use-case.

- Each use-case should implement its own `problem` and `problem_loader`.
- Decouples business logic (graph data sampling, gradient computation, etc.) from the GNN logic (forward pass and back-prop).

```
for problem in problem_loader:
    context = problem.get_context()              # (1) Use-case input graph
    decision = model(context)                    # (2) GNN forward pass
    gradient = problem.get_gradient(decision)    # (3) Use-case direction of improvement
    model.update_weights(gradient)               # (4) GNN back-propagation and weight update

# Fun fact: context, decision and gradient are all hyper heterogeneous multi graphs.
```

EnerGNN

energnn.trainer

Trains a GNN model over a use-case specific problem loader.

- Orchestrated training loop using *optax*.
- Easy integration with experiment trackers.
- Checkpoint management and persistence via *orbax*.

```
from energnn.trainer import SimpleTrainer

trainer = SimpleTrainer(model=..., train_loader=..., val_loader=...)
trainer.train(n_epochs=10)
```

EnerGNN

Use Cases @ RTE

EnerGNN

Contingency Screening | TRL 5

For a given operating condition, predict which contingencies (e.g. line disconnection) will cause an overflow.

- Full-scale real-life data.
  - One years of snapshots.
  - French HV-EHV transmission grid, 7k+ buses.
  - Wide variety of object classes (lines, transformers, generators, loads, shunts, etc.).
  - Real-life topological variations, object renaming.
- Supervised classification.
- 99% of risky outages correctly predicted.
- 86% of non-risky outages correctly predicted.

EnerGNN

Tertiary Voltage Control | TRL 5

For a given operating condition, minimize voltage violations.

- Continuous and discrete drivers for action.
- Full-scale real-life data.
  - Two years of snapshots.
  - French HV-EHV transmission grid, 7k+ buses.
  - Wide variety of object classes (lines, transformers, generators, loads, shunts, etc.).
  - Real-life topological variations, object renaming.
- Self-supervised learning. The GNN learns by trial-and-error.
- ~50% of voltage violations are avoided.

EnerGNN

Topology Optimization | TRL 3

For a given operating condition, modify switch states to improve exchange capacity between two regions.

- Self-supervised learning. The GNN learns by trial-and-error.
- Small-scale toy system. 12 substations, 57 switches.
- Detailed topology. Lines, switches, loads and generators.
- GNN, ~9.3% mean capacity improvement in ~200ms.
- MILP baseline, ~13.7% mean capacity improvement in ~10min.

EnerGNN

Team & Experience

EnerGNN

- 7+ years of experience in GNNs for Power Systems @ RTE.
- Partnerships with academics (Université de Liège, Université Paris-Saclay, Mines Paris-PSL, University College Dublin).
- Upcoming collaboration with InstaDeep.
- Main contributors & supervisors:
  - Geoffroy JAMGOTCHIAN (RTE)
  - Hugo KULESZA (RTE)
  - Steve NOUATIN (RTE / DataStorm)
  - Balthazar DONON (RTE)
  - Louis WEHENKEL (ULiège)

EnerGNN

Open to new use cases, model improvements, and API feedback.

Code: https://github.com/energnn

Contact: balthazar.donon@rte-france.com