## Graph Neural Networks in Power Forecasting

### Overview

This report, adapted from a presentation on Graph Neural Networks (GNNs), explores their use in power market modelling, focusing on forecasting the Japan day-ahead to imbalance price spread. We provide a brief overview of GNN fundamentals, highlight two areas where they excel, weather forecasting and the optimal power flow problem and present a case study of Japan, comparing GNN-based approaches with traditional models across node and edge-level prediction tasks.

### 1. Introduction & Motivation

Energy systems such as power grids, weather fields, and energy flows are inherently networked, with complex spatial and temporal dependencies. Traditional independent-node or purely time-series models often ignore these connections, limiting their ability to capture interactions and propagation effects across the network.

#### Three motivating examples:

* 1. **Weather forecasting**: Weather forecasts are important for power markets, but the data can be tricky spread over irregular grids, from many sources, and sometimes biased. New GNN-based models like GraphCast and Pangu-Weather have beaten leading weather prediction systems on many measures, showing they can capture the complex spatial and temporal patterns in the atmosphere.
  2. **Optimal Power Flow**: Optimal Power Flow (OPF) finds the most efficient way to run a power grid while meeting demand and staying within limits. Traditional solvers can be too slow for large systems, so operators use simplified versions that lose accuracy. Recent GNN-based models like DeepMind’s CANOS solve OPF much faster and handle changes in grid topology, making them promise for real-time use on large grids.
  3. **Electricity price prediction**: Lastly, we present a case study of the Japan power market, using GNNs for node classification, node regression, and edge classification to forecast properties that are central to market performance and system reliability.

The remainder of this report first reviews the basic concepts behind GNNs, then discusses their role in weather forecasting and optimal power flow, before presenting the Japan case study and comparing GNN performance with traditional modelling approaches.

### 2. Background and related work

#### 2.1 From Neural Networks to Graph Neural Networks

Artificial neural networks (ANNs) learn mappings from inputs to outputs by composing layers of parameterized transformations. In a feedforward network, a fixed-size input vector is passed through one or more hidden layers with nonlinear activations to produce an output vector. For an item u at layer k, this can be written as

,

Advances in architectures such as convolutional neural networks (CNNs) for images and recurrent neural networks (RNNs) for sequences have driven major progress in areas like image recognition, natural language processing, and speech synthesis ***(Schmidhuber, 2015).***

These architectures, however, assume highly structured inputs: CNNs work on regular grids, while RNNs require strictly ordered sequences. Many real-world systems such as power grids, transportation networks, molecular structures, and weather meshes are best represented as graphs, where entities are connected by complex, often irregular relationships. Flattening these into fixed-size vectors typically discards valuable structural information.

Graph Neural Networks (GNNs) generalize deep learning to graph-structured data, producing node or graph representations that depend on both node features and the graph’s connectivity. Rather than processing each entity in isolation, GNNs propagate information along edges via *message passing*, enabling each node to incorporate context from its neighbours and the wider network. This makes them well suited to systems with rich relational dependencies, such as the spatial and temporal interactions in energy markets.

#### 2.2 Graph Representation Basics

A graph consists of a set of nodes and a set of edges describing relationships between nodes. In energy applications:

* **Nodes** may represent physical entities such as market regions, weather stations, substations, or generation units.
* **Edges** capture relationships between nodes, such as transmission lines in a grid, spatial adjacency in a weather mesh, or flow connections between markets.
* **Node features** are variables measured or engineered for each node (e.g., historical demand, temperature, wind speed, local generation).
* **Edge features** describe properties of connections (e.g., line capacity, impedance, historical congestion frequency).

Graphs can take different forms:

* **Static vs dynamic** – Static graphs keep the same nodes and edges over time; dynamic graphs allow topology or features to change, such as line outages or new connections.
* **Directed vs undirected** – Directed graphs encode asymmetric relationships (e.g., HVDC links with directional limits), while undirected graphs assume symmetry.
* **Weighted vs unweighted** – Weighted graphs assign a numerical value to each edge (e.g., capacity, distance, strength of correlation), while unweighted graphs simply record whether an edge exists.

In power system modelling, graphs are typically **sparse** (each node connects to only a small subset of others), **weighted** (edges have capacities), and often **dynamic** (reflecting outages, maintenance schedules, or seasonal network changes).

#### 2.3 Message Passing & Spatiotemporal GNNs

The content in this section follows the formulations and concepts from key works in the field, including the original Graph Neural Network framework (Scarselli et al., 2009), the Graph Attention Network (Veličković et al., 2018), and the overview in *Graph Representation Learning* (Hamilton, 2020).

The core idea behind Graph Neural Networks is message passing: each node updates its representation by combining its own features with information aggregated from its neighbours. At iteration k, the basic update is:

where is the set of neighbours of node u, and are trainable matrices, is a bias, and σ is a non-linear activation.

Two important properties follow:

* **Permutation equivariance** – the output does not depend on how nodes are ordered in the input.
* **Receptive field growth** – with each layer, a node’s embedding incorporates information from one more hop in the graph; after K layers it contains information from all nodes within K hops.

In practice, adding many layers often leads to **over smoothing**, where node embeddings become nearly identical and local distinctions are lost. Complexity is usually increased not by depth, but by making the **aggregation** step richer through normalization, pooling, or attention.

One common extension is the **Graph Attention Network (GAT)**, which learns a weight for each neighbour before aggregation:

and is a trainable attention vector. This allows the model to assign different importance to different connections.

**From embeddings to predictions:** After K message-passing steps, each node u has a final embedding that encodes both its own features and information from its neighbourhood. These embeddings can be fed into decoders tailored to the prediction task:

**Node-level tasks**

* Node regression – predict a continuous value (e.g., price, demand):
* Node classification – predict a discrete class (e.g., congestion event, reserve shortfall):

**Edge prediction / classification** – compute a score for an edge between nodes u and v using their embeddings:

* Dot product decoder:
* Bilinear decoder:

These decoders allow a single GNN architecture to be adapted to a wide range of tasks, from predicting values at each node to forecasting the likelihood of a connection (edge) existing or having certain properties.

**Extensions**: For datasets with strong temporal dynamics such as weather evolution or electricity prices GNNs are often combined with temporal encoders (RNNs, TCNs, Transformers) that process sequences of node features before spatial aggregation. This allows the model to capture both time evolution and spatial dependencies without relying solely on deeper message passing.

### 3.GNN Applications in Energy Systems

We now turn to two application areas where GNNs have achieved strong results in energy modelling: weather forecasting and optimal power flow.

#### 3.1 Weather Forecasting with GNNs

**Models and timeline.**  
The first widely adopted global GNN forecaster was GraphCast (DeepMind, 2023): a multi-scale graph model that delivers 10-day deterministic forecasts and beat ECMWF-HRES on most verification targets. Its successor, GenCast, is a *probabilistic* (ensemble) model based on diffusion that outperforms ECMWF’s ENS on the vast majority of targets up to 15 days.

**Inputs and initialization.**  
GraphCast/GenCast are trained on ERA5; operational runs initialize from analysis fields (e.g., ECMWF HRES or national systems). GraphCast consumes two 3-D states (t−6 h, t) and rolls forward in 6-hour steps. Public work has shown effective fine-tuning to other analysis systems, including Canada’s GDPS and NOAA’s GDAS/GFS (EAGLE), enabling drop-in use outside the ECMWF pipeline.

**Where they perform best and limits.**  
GNN weather models excel at synoptic-scale skill in the 1–15-day window and give earlier, spatially consistent signals for coherent extremes (e.g., cyclone tracks/placement), with GenCast adding calibrated spread. Known gaps include underestimation of fine-scale intensities (e.g., convective rain, peak winds) and regional/variable-dependent rankings where other ML models may lead in specific domains.

**Adaptation for trading/research.**  
For power market use, GNN forecasts don’t have to be used in isolation they can be blended or adapted for higher-value decisions:

1. **Switching models by conditions** – Use GNN forecasts when they outperform traditional NWP (e.g., for extreme events or coherent large-scale systems) and fall back to conventional models when those have better skill (e.g., small-scale convection or fine-detail local effects).
2. **Regional fine-tuning** – Adapt a pretrained GraphCast/GenCast to a local analysis system (e.g., GDAS, GDPS) through curriculum fine-tuning. This reduces local bias and sharpens resolution in the target market area.
3. **Variable-specific optimisation** – Re-weight losses for key variables in a given region or asset, such as optimising 10 m wind forecasts for a specific wind park or solar irradiance for a PV site.

For day-ahead to week-two horizons can combine these strategies: start from GraphCast (deterministic) or GenCast (ensemble) outputs, apply regional tuning where needed, and selectively switch to or blend with traditional models based on expected skill for the market-relevant weather regime.

**3.2 Optimal Power Flow (OPF)**

* Brief definition: DC/AC OPF, objective, constraints.
* DeepMind’s work: learning OPF surrogates; performance vs classical solvers.
* Benefits of GNNs: locality, scalability, generalization to unseen topologies.
* Relation to forecasting tasks: e.g., predicting congestion, reserve shortfalls.

**3.2 Optimal Power Flow (OPF)**

**What OPF is.**  
OPF chooses generator set-points (and other controls) to minimise operating cost while satisfying network physics and operational limits. **AC-OPF** enforces full nonlinear power-flow equations (voltages, reactive power, thermal/angle limits) and is hard at scale; **DC-OPF** is a fast linear approximation used widely in market workflows but can be infeasible for AC constraints. Security-constrained OPF (SCOPF) adds **N-1** reliability by requiring feasibility under any single outage.

**Classical solutions (baseline).**  
Industry and research tools solve AC-OPF with nonlinear programming (e.g., interior-point methods). They are accurate and provide feasibility certificates when they converge, but can be slow on very large systems or under tight N-1 constraints. DC-OPF is fast and scalable, but ignores voltage/reactive effects and can misestimate congestion in some regimes.

**What DeepMind did (CANOS).**  
CANOS is a **GNN-based AC-OPF surrogate**: given a grid state, it predicts near-optimal set-points very quickly and is designed to be **robust to N-1** topology changes. It scales to large networks and delivers solutions close to AC-OPF quality in milliseconds rather than seconds/minutes. The trade-off is that, like other learned surrogates, it does not guarantee AC feasibility on every instance—so a light **feasibility repair** (power-flow/post-processing) or occasional fallback to a classical solver is recommended.

**How it compares and how to use it.**

* **Versus AC NLP solvers:** orders faster, but classical solvers remain the gold standard for exact feasibility/certification.
* **Versus DC-OPF:** typically more accurate with respect to AC objectives/constraints while retaining speed; still benefits from a PF clean-up step.
* **Practical integration:** use CANOS as a **surrogate/warm-start** for AC-OPF, or for **rapid scenario screening** (contingencies, outages, redispatch) to explore congestion/LMP impacts; keep guardrails (violation checks, fallback triggers).

*Implication for trading/research:* a fast, graph-based OPF surrogate enables large-scale “what-if” analysis and congestion forecasting at interactive speeds. Pair it with post-processing and clear fallbacks to manage feasibility risk.

**4. Case Study: Japan Power Markets with GNNs**

**4.1 Market & Data Overview**

* Japan’s market structure: regions, interties, frequency split.
* Data sources: market, weather, outages.
* Graph construction: nodes, edges, weights, dynamic status.

**4.2 Tasks & Problem Formulation**

* **Node regression**: price/reserve rate prediction.
* **Node classification**: scarcity/reserve events.
* **Edge classification/regression**: congestion probability or flow amount.

**4.3 Models Evaluated**

* Baselines (time-series, tabular ML).
* GNN architectures (GCN, GraphSAGE, GAT with temporal modules).

**4.4 Evaluation Setup**

* Metrics for regression, classification, and edge tasks.
* Train/val/test protocol and ablations (topology, features, architectures).

**4.5 Results & Analysis**

* Performance tables by task/horizon.
* Horizon-vs-skill plots, calibration plots.
* Ablation charts (impact of topology/feature set).
* Interpretation: attention maps, edge importances.

**5. Discussion & Broader Perspective**

* How lessons from Japan case study apply to weather forecasting and OPF.
* When GNNs give a clear advantage; when simpler models suffice.
* Potential extensions:
  + Region-specific weather model fine-tuning.
  + Dynamic graph prediction (outages, capacity changes).
  + Scaling to ERCOT/Europe-scale grids.

**6. Conclusion**

* Recap of key findings and implications for research & industry.
* Limitations and next steps.

**References**

* 10–15 curated papers across weather forecasting, OPF, and energy market GNNs.
* Include graph cast / graph gen paper
* Include Deep mind optimal power flow paper
* Schmidhuber, 2015.
* Original graph Neural Network Paper
* Graph Representation Learning Book