



Learning Spatio-Temporal Aggregations for Large-Scale Capacity Expansion Problems

Aron Brenner
abrenner@mit.edu
Massachusetts Institute of Technology
USA

Rahman Khorramfar
khorram@mit.edu
Massachusetts Institute of Technology
USA

Saurabh Amin
amins@mit.edu
Massachusetts Institute of Technology
USA

ABSTRACT

Effective investment planning decisions are crucial to ensure that critical cyber-physical infrastructures satisfy performance requirements over an extended time horizon. Computing these decisions often requires solving Capacity Expansion Problems (CEPs). In the context of regional-scale energy systems, these problems are prohibitively expensive to solve owing to large network sizes, heterogeneous node characteristics (e.g., electric power and natural gas energy vectors), and a large number of operational periods. To maintain tractability, traditional approaches resort to aggregating network nodes and/or selecting a set of representative time periods. Often, these reductions do not capture the supply-demand variations that crucially impact the CEP costs and constraints, leading to suboptimal decisions. Here, we propose a novel graph convolutional autoencoder approach for spatiotemporal aggregation of a generic CEP with heterogeneous nodes (CEPHN). Our autoencoder architecture leverages graph pooling to identify nodes with similar characteristics and minimizes a multi-objective loss function. This loss function is specifically tailored to induce desirable spatial and temporal aggregations in terms of tractability and optimality of CEPHN. In particular, the output of the graph pooling provides a *spatial aggregation* while clustering the low-dimensional encoded representations yields a *temporal aggregation*. We apply our approach to generation expansion planning of coupled power and natural gas system in New England. The resulting spatiotemporal aggregation leads to a simpler CEPHN with 6 nodes (as opposed to 88 nodes in the original system) and a small set of representative days selected from a full year. We evaluate aggregation outcomes over a range of hyperparameters governing the loss function, and compare resulting upper bounds on the original problem with those obtained using previously known methods. The results from our case study show that this approach provides solutions that are 33% (resp. 10%) better those than obtained from standard spatial (resp. temporal) aggregation approaches.

CCS CONCEPTS

• **Computing methodologies** → **Cluster analysis**; • **Applied computing** → *Multi-criterion optimization and decision-making*.

KEYWORDS

Dimensionality Reduction, Clustering, Graph Neural Networks, Mixed Discrete/Continuous Optimization, Power System Planning

ACM Reference Format:
Aron Brenner, Rahman Khorramfar, and Saurabh Amin. 2023. Learning Spatio-Temporal Aggregations for Large-Scale Capacity Expansion Problems. In *ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023) (ICCPS '23), May 9–12, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3576841.3585920



This work is licensed under a Creative Commons Attribution International 4.0 License.
ICCPS '23, May 9–12, 2023, San Antonio, TX, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0036-1/23/05.
https://doi.org/10.1145/3576841.3585920

1 INTRODUCTION

Capacity Expansion Problems (CEPs) are a class of optimization problems that determine the optimal timing, location, and sizing of future expansion for a portfolio of assets to support the growing demand [1, 10]. CEPs have found a broad range of applications in critical infrastructure networks, including communication systems [29], railway network [30], urban water supply infrastructure [8], and energy systems [3, 17, 22, 37]. These applications usually involve making decisions for a set of network locations over a planning horizon. In many settings, expansion planning requires modeling the coupling between strategic level investment decisions with operational decisions, making CEPs a large-scale mathematical programs. For instance, generation expansion problems (GEPs) in power systems involve computing long-term generation expansion decisions, while accounting for operational costs and constraints (e.g., dispatch, power flow, load control, and storage). Recently, the scope of these problems has been expanded to joint planning for multi-vector energy systems; in particular the coupling with natural gas (NG) and hydrogen [6, 13]. Thus, an outstanding challenge is to determine the spatial and temporal resolution of CEPs that enable us to utilize modern optimization solvers to obtain high-quality solutions for realistic problems instances. Here, we develop machine learning (ML) based approach to address this challenge. Specifically, we build on recent advances in deep learning for graph-structured data and develop a spatio-temporal aggregation approach for a class of CEPs.

Spatial and/or temporal aggregation of these CEPs has been the subject of many recent studies that seek to make problems analytically tractable for single-vector energy systems. Examples of temporal reduction strategies are rolling-horizon approach and representative period selection. In the first approach, the problem is solved iteratively over a manageable size of the planning horizon until all the planning periods are covered [5]. The latter approach is usually employed in power system CEPs where the planning problem is solved for a small number of appropriately chosen time periods and the corresponding weights [40]. On the other hand, spatial aggregation involves finding network structures such that

the problem can be tractably solved over a subset of nodes or a reduced-sized network [26, 34]. However, prior work on aggregation methods for CEPs, mostly relies on ad hoc techniques rather than systematically exploring the space of spatial and temporal aggregations. For example, in GEP, the bus-level network is aggregated into state or zone-level network [13, 18].

In CEPs with an underlying network structure, such as GEPs, an ideal parameter aggregation must yield a formulation that can be solved efficiently, while adequately capturing the parameters across spatial and temporal dimensions. Given that the planning decisions depend crucially on the network structure and planning horizon, the choice of aggregation can significantly impact the CEP's solution [25, 32]. Previous work has explored the use of ML methods to learn high-quality temporal aggregations [41]. However, learning *spatial* aggregation in conjunction with temporal aggregation is an unexplored problem, especially in the context of multivector energy systems. Our graph convolutional autoencoder approach attempts to fill this gas by considering CEPs with underlying network structure and heterogeneous nodes (HN) that are characteristics of multivector energy systems (e.g., electricity power generation and consumption nodes or NG supply and demand nodes). Importantly, nodes may differ in terms of expansion cost parameters and operational constraints arising from technological differences and modeling resolution of individual subsystems. We refer to such planning problem as CEPHN. Our approach can be potentially applied to expansion planning for new cyber-physical systems, enabling refinement of classical approaches in facility location problems.

The proposed graph convolutional autoencoder approach captures (i) spatial correlations between input parameters (ii) physical interdependencies between the heterogeneous nodes and (iii) heterogeneous granularity of data for two or more coupled networks for power and NG nodes as our test case. The graph convolutional architecture accounts for the network interactions by training both within and across heterogeneous nodes autoencoder that minimizes a multi-objective loss function for node-specific input data. Moreover, we incorporate graph pooling to automatically learn spatial aggregations that group nodes exhibiting similar input patterns. The training procedure is based on the loss function that reconstruction loss and clustering objective. Our work provides a new direction to make CEPHNs tractable by utilizing the automatically learned spatio-temporal aggregations, leanding to high-quality planning decisions for critical infrastructure networks.

Specifically, we focus on aggregating and solving an otherwise intractable joint power-gas planning problem formulated as a GEP. The network model underlying this formulation is fairly detailed, capturing various power-gas operating constraints and decarbonization policy directives over a given planning horizon. The problem has heterogeneous nodes, each with a different temporal resolution [13]. We demonstrate that our autoencoder model is well-suited to learning latent embeddings of the spatio-temporal patterns in this GEP and yield better solutions than previous techniques. The investment decisions of the GEP we consider here are over a planning horizon with yearly granularity, while the operational decisions (e.g., unit commitment, power production, load shedding, and energy storage) require hourly or sub-hourly resolution. In our case, the computational difficulty in solving the GEP increases further because we model both power and NG networks. Thus, considering the demand information at a day-to-day granularity becomes prohibitively expensive from a computational viewpoint. Prior work has tried to tackle such issues by aggregating power system nodes (buses) within a geographical neighborhood (power zone) to a single node [13, 18] and by solving the GEP for a set of representative periods (e.g., days) [12, 40]. In contrast, our approach accounts for the spatio-temporal variability in supply and demand patterns in determining the reduced dimensional network stricture and also provides the set of representative days. This yields a tractable CEPHN formulation that can be solved to obtain high-quality planning decisions

Previous work on selecting sets of representative days using variants of k-means [2, 19, 24, 39], k-medoids [33, 39], and hierarchical clustering [21, 39]. The distance data used in clustering algorithms are usually constructed based on a set of time series inputs such as load data and variable renewable energies (VRE) capacity factors [12, 18]. Notably, these approaches neither account for demand data with multiple time resolutions nor spatial interdependencies, which we exploit with our graph convolutional approach. Hence, they cannot be readily extended to extract representative days for joint power-NG systems, an aspect that is crucial for realism and tractability in joint planning models for decarbonizing these systems. Spatial aggregation methods in power systems employ various techniques including bus elimination, node clustering based geographical boundaries, or other heuristic methods such as Ward reduction [26, 34]. However, these studies do not apply a fully data-driven approach to aggregating buses in the presence of another energy vector across nodes. We believe that our approach addresses these limitations.

In the remainder of the paper we first present a generic CEP with heterogeneous nodes in Section 2. We define spatio-temporal aggregation and present relevant notation in Section 3. We review the autoencoder technique and give details of the graph convolutional neural network approach in Section 4. Section 5 presents the details of the case study, and Section 6 describes the setup for our numerical analysis. The results and discussion is provided in Section 7. Finally, we conclude the paper and provide future directions in Section 8.

2 CAPACITY EXPANSION MODEL

In this section, we present a generic CEPHN for a network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with each node $n \in \mathcal{N}$ belonging to $s \in \mathcal{S}$, the set of node types. In the context of generation expansion models, different node types can represent nodes associated with each energy vector (e.g., power nodes, NG nodes, and storage nodes). We assume that the planning horizon for all nodes is the same, but the node types can differ in their time resolution required for operational planning. To incorporate heterogeneous times-scale of different node types, we assume that larger timer resolution are all multipliers of smallest one. For example, we can model power system operation on an hourly basis and NG network operations on a daily basis. For a given class of nodes, s, we denote by $\mathcal{N}^s$ the set of nodes. Likewise, we denote by $\mathcal{T}^s$ the set of operational periods, e.g. hours or days. We introduce $\boldsymbol{x}^s$ as node-wise strategic-level decision variables, such as investment in or establishment of a new asset for class s. Similarly, we define $\boldsymbol{y}^s$ to be node-wise operational decisions for

class s over the set of planning periods $\mathcal{T}^s$. The CEPHN problem can then be formulated as:

$$\min_{\boldsymbol{x}^s, \boldsymbol{y}^s} \quad \sum_{s\in\mathcal{S}} \sum_{n\in\mathcal{N}^s} \left(\mathbf{f}_n^s x_n^s + \sum_{t\in\mathcal{T}^s} \mathbf{g}_{nt}^s y_{nt}^s \right) \tag{1a}$$

$$\text{s.t.} \quad \mathbf{A}^s \boldsymbol{x}^s + \mathbf{B}^s \boldsymbol{y}^s \le \mathbf{b}^s, \quad \forall s \in \mathcal{S} \tag{1b}$$

$$\sum_{s\in S} \mathbf{C}^s \boldsymbol{y}^s \le \mathbf{h}^s \tag{1c}$$

$$\boldsymbol{x}^s \in \mathbb{Z}^{p^s} \times \mathbb{R}^{q^s}, \quad \forall s \in \mathcal{S} \tag{1d}$$

$$\boldsymbol{y}^s \in \mathbb{R}^{l^s}, \quad \forall s \in \mathcal{S} \tag{1e}$$

The objective function (1a) minimizes the total cost of investments and operations summed over the planning horizon. (1b) defines constraints in each class. The constraint (1c) couples operational constraints across all types of nodes. Without loss of generality, we assume that the shared variables only appear in the coupling constraint which is defined in terms of the parameters of that particular node type. In a multivector energy system, the coupling constraints model the exchange of energy between node types. They also capture policy constraints such as system-wide emissions reduction goals that limit the emission of CO_2 induced by all node types [6, 13, 44]. In particular, our formulation can model a broad range of CEPHNs from infrastructure planning [9] to supply chain design [23]. For most applications, this is a large-scale mixed integer programs and is not tractable even with the power of modern optimization solvers. Our spatio-temporal aggregation approach seeks to address this challenge.

3 SPATIO-TEMPORAL AGGREGATION

Conceptually, the aggregation process is a set of mathematical operations designed to reduce the complexity of the problem (1) by reducing the network size and number of operational periods, and consequently, reducing the number of variables and/or constraints in the original formulation. An ideal spatio-temporal aggregation provides a reduced-form model whose solution and objective values are close to those of the original formulation. However, obtaining an exact solution for the original problem is, by assumption, intractable. As such, "proxy" methods are proposed in the literature in order to evaluate the quality of spatio-temporal aggregations. Examples include i) comparison of the solution of the aggregated problem to a lower bound of the original problem [46]; ii)evaluating specific parameters (e.g., susceptance of transmission lines) [26, 34]; iii) comparing objective values for feasible solutions to the original problem obtained from the aggregated problem; [24, 42].

The literature on spatial aggregation is relatively sparse and rather ad hoc. These include aggregating nodes by geographical boundaries [13, 18], and exploiting the underlying technical structure of the problem [26, 34]. On the other hand, the literature on temporal aggregation for large-scale CEPs is more developed. Here, rolling-horizon and clustering of planning periods are common methods. Although rolling-horizon approach results in a feasible solution to the original problem, it does not capture the variation in demand and supply parameters over the entire planning horizon. Therefore, the absence of extreme days in some look-ahead periods can render an inferior solution, limiting the applicability of this method to realistic planning problems.

On the other hand, clustering methods aim to group the parameters associated with each operational period. For these methods, a distance metric, e.g., Euclidean distance, must be specified to measure dissimilarity between the parameters associated with any pair of operational periods. Variants of this method are employed in energy system planning problems, in particular k-means, k-medoids, and heirarchical clustering [40]. It is also common in power systems applications to construct a derived feature set by solving the full problem for a small number of planning periods and obtain different cost components or other output features for each planning period [2, 40]. These derived features are then included in the distance calculation used by the clustering algorithm. In addition to the model parameters and derived features, clustering can also utilize other exogenous data that impact temporal variation in the parameters. In the power system context, such data can include ambient temperature, and electricity prices [47], which are known to have a significant impact on the realized power loads.

To build the case of our spatio temporal aggregation approach, we now illustrate clustering-based temporal aggregation on the generic CEPHN introduced in Section 2. The first step is to select a node type s and its associated time resolution t. Recall that we assumed all time resolutions are multiplier of the smallest one. For each $s \in \mathcal{S}$, we construct the vector $\mathbf{x}_n^{(t)}$ for node n at time t as follows:

$$\mathbf{x}_n^{(t)} = \left(\text{vec}(\tilde{\mathbf{P}}_n^{(t)}) \parallel \text{vec}(\tilde{\mathbf{O}}_n^{(t)}) \parallel \text{vec}(\tilde{\mathbf{E}}_n^{(t)}) \right),$$

where $\parallel$ and $\text{vec}(\cdot)$ are the concatenation and vectorization operators, respectively. The elements of $\mathbf{x}_n^{(t)}$ include time varying information that captures the inherent variability across operational periods; the element $\tilde{\mathbf{P}}_n^{(t)}$ denote a subset of CEPHN model parameters, $\tilde{\mathbf{O}}_n^{(t)}$ denote the aforementioned derived features, and $\tilde{\mathbf{E}}_n^{(t)}$ are exogenous data relevant for aggregation. We will refer to the elements of $\mathbf{x}_n^{(t)}$ henceforth as the observed *features* of node n at time t. We then construct a matrix of features for each subsystem $s \in \mathcal{S}$ by concatenating as follows

$$\mathbf{X}_s^{(t)} = \begin{bmatrix} \mathbf{x}_{s_1}^{(t)} \\ \vdots \\ \mathbf{x}_{s_{|\mathcal{N}^s|}}^{(t)} \end{bmatrix} \in \mathbb{R}^{|\mathcal{N}^s| \times d_s},$$

where $\{s_1, \ldots, s_{|\mathcal{N}^s|}\}$ is the set of nodes within subsystem s. In concatenation of features, if data of node type $s' \in \mathcal{S}\backslash s$ has coarser time resolution than s, then we repeat its features for each time period t. For example, suppose a CEPHN has two nodes types with hourly and daily resolution. If we choose to aggregate at an hourly temporal resolution, then data with a daily resolution will be repeated in all matrices, $\mathbf{X}_s^{(t)}$, for all one-hour periods t within the same day. We also denote by $\mathcal{T}$ the set of time periods corresponding to the temporal resolution at which we aim to aggregate. After constructing the dataset $\{\mathbf{X}_s^{(t)} : s \in \mathcal{S}, t \in \mathcal{T}^s\}$ as described above, the parameters across all subsystems can be further vectorized and concatenated to construct

$$\mathbf{x}^{(t)} = \left(\text{vec}(\mathbf{X}_1^{(t)}) \parallel \ldots \parallel \text{vec}(\mathbf{x}_{|\mathcal{S}|}^{(t)}) \right),$$

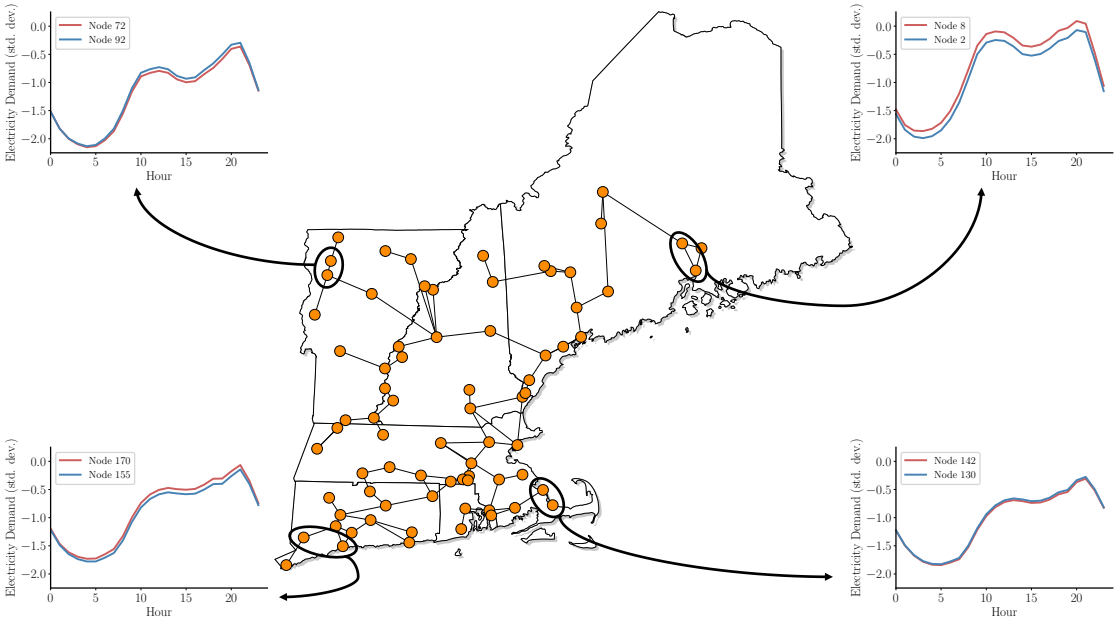


Figure 1: Demand nodes in the same neighborhood tend to demonstrate similar variations in electrical load over the course of the day. Spatial dependencies in CEPHN parameters, like those shown in this figure, are modeled explicitly by graph convolutional layers in the proposed autoencoder architectures.

which has dimensionality $\sum_{s\in\mathcal{S}} |\mathcal{N}^s| d_s$. Then, a generic clustering algorithm such as k-medoids clustering can be applied to minimize the objective

$$\min_{C,\mathbf{z}} \sum_{k=1}^{K} \sum_{t\in C_k} \|\mathbf{x}^{(t)} - \mathbf{z}_k\| \tag{2}$$

by assigning each of the parameter vectors to one of K representative periods, where $\{\mathbf{z}_1,\dots,\mathbf{z}_K\} \subset \{\mathbf{x}^{(1)},\dots,\mathbf{x}^{|\mathcal{T}|}\}$ denote the cluster medians, C_k denotes the set of periods assigned to representative period $\mathbf{z}_k$, and $\|\cdot\|$ denotes a user-specified dissimilarity metric.

In practice, however, the performance of clustering algorithms deteriorate as the dimensionality of its input data increase [2]. This is due to overwhelming effect of random noise on distance calculations in high-dimensional settings, which poses a challenge to extracting meaningful distances between operational periods. [7] and [2] note that resolving this "curse of dimensionality" for aggregating large-scale CEPHNs, where large network sizes and high temporal resolutions of operational periods yield a very high-dimensional dataset.

4 LEARNING AGGREGATION ARCHITECTURE

To resolve the challenges associated with the high dimensionality of the dataset associated with the CEPHN problem, we develop a graph convolutional neural network (GCN) approach for spatio-temporal aggregation. Specifically, we train a graph convolutional autoencoder with a pooling operation (see Section 4.2) to minimize a multi-objective loss function, which can be tuned to trade off desirable properties of spatial and temporal aggregations (see Section 4.3). By incorporating graph pooling, we constrain the autoencoder to identify groups of nodes that demonstrate similar patterns in their corresponding parameters (e.g., nodal demands, production capacities, etc.). This allows us to extract the learned pooling assignments as a *spatially aggregated* network. Simultaneously, the pooled features can be extracted from the autoencoder, which yields low-dimensional representations of the data on which we can apply a clustering algorithm to obtain a *temporal aggregation* [27]. The resulting spatio-temporal aggregation yields a computationally tractable CEPHN formulation for a subset of operational periods and on a reduced network size.

4.1 Overview of Autoencoders

Autoencoders are deep neural networks that are trained to encode (compress) and decode (decompress) high-dimensional inputs. Unlike principal components analysis (PCA), autoencoders are non-linear models that are trained in a supervised fashion to minimize the error between input data and its corresponding reconstruction following compression [11]. Importantly, these models learn to map high-dimensional inputs into lower-dimensional *latent spaces*, and they are frequently used in practice to learn latent distances among high-dimensional data points while limiting the confounding effects of random noise.

Given a high-dimensional input such as a set of spatio-temporal CEPHN parameters, $\mathbf{x}^{(t)} \in \mathbb{R}^d$, an autencoder can be trained to jointly learn an encoder, $E_\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$, and a decoder, $D_\theta : \mathbb{R}^{d'} \to$

$\mathbb{R}^d$ that minimize the reconstruction loss function,

$$\mathcal{L} = \sum_{t \in \mathcal{D}} \left\| \mathbf{x}^{(t)} - \hat{\mathbf{x}}^{(t)} \right\|_2^2$$

over the dataset $\mathcal{D}$, where $\hat{\mathbf{x}}^{(t)} = D_\theta(E_\phi(\mathbf{x}^{(t)}))$ is the reconstructed input. Here, $d' \ll d$ denotes the dimension of the learned latent space, or more informally, the "tightness" of the compression bottleneck. Lower values of d' will yield higher errors in reconstructing inputs, but are likely to be more effective in reducing the impact of noise on calculating distances to be used in clustering applications [27]. Considering the temporal aggregation approach discussed in Section 3, we have $d = \sum_{s \in \mathcal{S}} |\mathcal{N}^s| d_s$ and $\mathcal{D} = \mathcal{T}$. Then, one can instead aim to minimize (2) using distances computed from $\{E_\phi(\mathbf{x}^{(1)}), \ldots, E_\phi(\mathbf{x}^{(|\mathcal{T}|)})\}$ rather than $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(|\mathcal{T}|)}\}$.

Recent work on graph representation learning has facilitated the extension of deep unsupervised learning to high-dimensional graph-structured data [15]. In Section 4.2, we introduce relevant ideas from modeling with graph neural networks. Then, in Section 4.4 and 4.3, we describe our autoencoder architecture and training process.

4.2 Graph Representation Learning

To apply graph convolution operations, it is first necessary to encode a relevant graph structure over the nodes $\mathcal{N}$ with an affinity matrix, $\mathbf{A} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{N}|}$. Importantly, the graph convolutions do not need to utilize the exact adjacencies, $\mathcal{E}$, given by the CEPHN network topology, $\mathcal{G}$. Rather, an appropriate affinity structure to encode is one that captures underlying interdependencies between nodes. For instance, one can choose to construct an affinity matrix based on geospatial distance, weather-related similarity, or demand patterns. We encode these affinities between any two node n and n' as

$$\mathbf{A}_{nn'} = \exp\left(-\frac{\text{dist}(n, n')^2}{\sigma^2}\right),$$

where $\text{dist}(n, n')$ denotes a distance metric of choice between nodes n and n' (e.g., the Euclidean distance between their corresponding geographical coordinates), and σ denotes the standard deviation of distances in the network [35]. We also construct the diagonal degree matrix $\mathbf{D}$ such that $\mathbf{D}_{nn} = \sum_{n'} \mathbf{A}_{nn'}$.

Following [16], we utilize *Chebyshev convolutional filters*, which approximate spectral convolutions to learn latent node features as weighted local averages of observed and learned features for adjacent nodes. This is ideal for learning low-dimensional representations of networks as neighborhoods of nodes typically exhibit related (either similar or complementary) spatio-temporal patterns in demand, production capacity, etc. (see Fig. 1). Chebyshev filters operate on the "renormalized" graph Laplacian $\tilde{\mathbf{L}} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}}$, where $\tilde{\mathbf{D}} = \mathbf{I} + \mathbf{D}$ and $\tilde{\mathbf{A}} = \mathbf{I} + \mathbf{A}$, and perform a form of Laplacian smoothing [20, 38]. We initialize $\mathbf{H}_0 \in \mathbb{R}^{|\mathcal{N}| \times d}$ to be equal to our input matrix for $|\mathcal{N}|$ nodes with d node features and apply convolutional filters to learn subsequent node features as follows:

$$\mathbf{H}_{l+1} = \sigma(\tilde{\mathbf{L}} \mathbf{H}_l \mathbf{\Theta}_l),$$

where $\mathbf{\Theta}_l$ is a trainable weight matrix and $\mathbf{H}_l$ is a matrix of node embeddings in layer l. $\sigma(\cdot)$ is typically a nonlinear activation function, such as ReLU or tanh.

In each layer, GCNs aggregate features from the immediate neighborhood of each node. Deep GCNs stack multiple layers with nonlinear activations to learn latent node features as nonlinear functions of both local and global observed node features. In contrast, [31] propose a simpler graph autoencoder model, which they demonstrated to have competitive performances with multilayer GCNs on standard benchmark datasets despite being limited to linear first-order interactions. Shallow neural architectures are also better suited for settings where data availability is more limited, which is commonly the case for CEPHN applications. For example, CEPHN formulations for power systems often utilize thousands of parameters to represent daily power demand over the course of one day for a large network. However, the number of such observations may be on the order of several hundreds of days. Consequently, one must select the depth of the autoencoder keeping in mind this trade-off between generalization in highly limited data settings and capacity of the model to learn potentially richer node representations.

Driven by the need for dimensionality reduction and coarsening in graph-level ML tasks, multiple approaches for graph pooling have been developed in recent years [4, 43, 45]. A typical approach to graph pooling is to train a GCN block, or a series of graph convolutional layers, with a softmax output activation for node grouping assignments as part of a larger neural network. Given $|\mathcal{N}|$ nodes and $|\mathcal{N}'| < |\mathcal{N}|$ desired node groups, the output of this pooling assignment block is $\mathbf{S} \in \mathbb{R}^{|\mathcal{N}| \times |\mathcal{N}'|}$, a matrix of soft assignments in which $\mathbf{S}_{nn'} \in (0, 1)$ encodes the degree to which node n is assigned to group n'. To pool the node features after the l-th layer, $\mathbf{H}_l$, we simply compute $\mathbf{Z} = \mathbf{S}^\top \mathbf{H}_l$. Conversely, we can disaggregate the pooled feature matrix by computing $\mathbf{SZ}$.

For example, one can imagine this block as taking in a time series of electricity demands for each node, $\mathbf{X}_s^{(t)}$, and returning a "predicted" membership distribution, parameterized by $\mathbf{S}^{(t)}$, which assigns all $|\mathcal{N}|$ nodes to one of the $|\mathcal{N}'|$ groups. Although there is no observed membership data with which to verify node group assignments, the quality of the assignments is evaluated implicitly by the reconstruction error $\mathcal{L}_R$ (which we define formally in Section 4.3), and consequently the reconstruction loss is backpropagated to the pooling block during training of the autoencoder. To minimize reconstruction loss, the pooling mechanism will learn to pool nodes that exhibit similar electricity demands in each day.

For our application, we utilize the MinCutPool operator proposed by [4], which corresponds to including the objective

$$\mathcal{L}_P = \underbrace{-\frac{\text{Tr}(\mathbf{S}^\top \tilde{\mathbf{A}} \mathbf{S})}{\text{Tr}(\mathbf{S}^\top \tilde{\mathbf{D}} \mathbf{S})}}_{\mathcal{L}_C} + \underbrace{\left\| \frac{\mathbf{S}^\top \mathbf{S}}{\|\mathbf{S}^\top \mathbf{S}\|_F} - \frac{\mathbf{I}_{|\mathcal{N}'|}}{\sqrt{|\mathcal{N}'|}} \right\|_F}_{\mathcal{L}_O}. \tag{3}$$

as part of a multi-objective loss function while training the autoencoder (see (6) in Section 4.3). This MinCutPool objective $\mathcal{L}_P$ is the sum of the *cut loss*, $\mathcal{L}_C$, and the *orthogonality loss*, $\mathcal{L}_O$. Minimizing the cut loss yields a pooling assignment that groups strongly connected nodes, while minimizing the orthogonality loss yields pooling assignments that are orthogonal and of similar sizes, i.e., each node is fully assigned to one group. The orthogonality loss term is included to discourage convergence to degenerate minima of the cut loss such as the uniform assignment of all nodes to all groups.

Moreover, it is often advantageous to ensure that node grouping assignments yield aggregated networks that are relatively uniform across certain parameters of interest. For example, one might prefer that the total demand of each node group (after summing over constituent nodes) does not differ too much from that of other groups. To this end, we also introduce the following *negative entropy loss* objective

$$\mathcal{L}_H = (\mathbf{S}^\top \mathbf{H}_0 \mathbf{1})^\top \log(\mathbf{S}^\top \mathbf{H}_0 \mathbf{1}), \tag{4}$$

where $\mathbf{1} \in \mathbb{R}^{d_s}$ is a vector of ones. Note that this negative entropy loss is minimized when the sums of input features for all node groups are identical.

4.3 Autoencoder Loss

Because CEPHNs consider different associated costs for investment and operational decisions across the different node classes (see Section 2), we introduce hyperparameters $\{\alpha_1, \ldots, \alpha_{|\mathcal{S}|}\}$ to tune the trade-off between the multiple reconstruction objectives, which gives us the total reconstruction loss objective

$$\mathcal{L}_R = \sum_{t \in \mathcal{T}} \sum_{s \in \mathcal{S}} \frac{\alpha_s}{|\mathcal{D}|} \|\mathbf{X}_s^{(t)} - \hat{\mathbf{X}}_s^{(t)}\|_F^2, \tag{5}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Then, the autoencoder can be trained to minimize the multi-objective loss function

$$\mathcal{L} = \alpha_R \mathcal{L}_R + \alpha_P \mathcal{L}_P + \alpha_H \mathcal{L}_H. \tag{6}$$

The hyperparameters $\{\alpha_R, \alpha_P, \alpha_H\}$ weight the reconstruction, pooling, and entropy losses respectively. Altogether, we must specify up to $3 + |\mathcal{S}|$ hyperparameters to weight the various objectives considered in this loss function, including those corresponding to reconstruction losses for different node classes. The class-specific reconstruction loss hyperparameters can be specified to reflect the contribution of each subsystem towards the total cost. Alternatively, we can choose to experiment with a range of these hyperparameters to determine the appropriate trade-offs between the multiple reconstruction losses and the pooling and entropy losses by solving the corresponding aggregated CEPHN and evaluating performance using its objective or an upper bound that it provides on the original (i.e., non-aggregated) CEPHN instance. In our case study, we evaluate a small range of these hyperparameter according to upper bounds on the original CEPHN given by solutions to aggregated instances, in which the aggregations are given by our autoencoder.

4.4 GCN Architecture

We now present the proposed GCN architecture for spatial and temporal aggregation, which is illustrated in Fig. 2. To train the autoencoder, we first construct the input

$$\mathbf{X}^{(t)} = \begin{bmatrix} \mathbf{X}_1^{(t)} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{X}_2^{(t)} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{X}_{|\mathcal{S}|}^{(t)} \end{bmatrix} \in \mathbb{R}^{|\mathcal{N}| \times (\sum_{s \in \mathcal{S}} d_s)},$$

where $\mathbf{X}_s^{(t)} \in \mathbb{R}^{|\mathcal{N}^s| \times d_s}$ denotes the matrix of features corresponding to all nodes in class s at time t. This block diagonal matrix effectively assigns a feature vector to each node with class-specific features for a subset of elements and zeros for all others. By applying graph convolutions, the autoencoder learns latent node features for a node in class s using observed features from neighboring nodes across *all classes* in $\mathcal{S}$. One may also include node-specific one-hot features in the input matrix, which is a common practice for graph convolutional modeling [14].

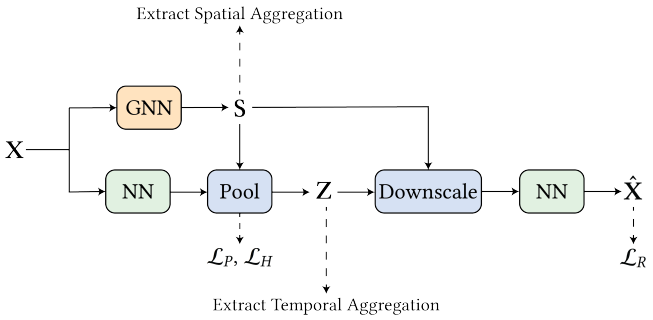


Figure 2: The autoencoder leverages a graph pooling operation to simultaneously learn spatial aggregations as node groupings and construct low-dimensional embeddings that can be used for identifying temporal aggregations via generic clustering algorithms. Values of component loss functions are extracted from different parts of the autoencoder, which must be balanced during the training process.

The encoder passes the input $\mathbf{X}^{(t)}$ in parallel through (1) a GCN block that outputs a node grouping assignment matrix, $\mathbf{S}^{(t)} \in (0,1)^{|\mathcal{N}| \times |\mathcal{N}'|}$, and (2) a second block that learns low-dimensional latent node features, $\mathbf{H}_l^{(t)} \in \mathbb{R}^{|\mathcal{N}| \times d'}$, using either fully connected layers or graph convolutions. Together, the outputs of these blocks are passed to a pooling layer, which aggregates the low-dimensional embeddings according to the learned grouping assignments, resulting in the low-dimensional node-aggregated embedding $\mathbf{Z}^{(t)} = \mathbf{S}^{(t)\top} \mathbf{H}_l^{(t)}$. This block also computes the MinCutPool loss, (3) and the negative entropy loss, (4). The pooled embedding matrix $\mathbf{Z}^{(t)}$ and node grouping assignment $\mathbf{S}^{(t)}$ are passed to the decoder.

To reconstruct the input, the decoder downscales the pooled feature matrix by computing $\mathbf{S}^{(t)} \mathbf{Z}^{(t)} \in \mathbb{R}^{|\mathcal{N}| \times d'}$. This matrix is then passed through several layers of graph convolutions and/or fully connected layers, which outputs the reconstructed parameter matrix $\hat{\mathbf{X}}^{(t)}$.

4.5 Retrieving Spatio-Temporal Aggregations

Once the autoencoder has been trained, each of the $|\mathcal{T}|$ operational periods can be passed through the encoder to retrieve the low-dimensional node-pooled feature matrices $\{\mathbf{Z}^{(1)}, \ldots, \mathbf{Z}^{(|\mathcal{T}|)}\}$. These feature matrices can then be vectorized into $(d'|\mathcal{N}'|)$-dimensional vectors, and a generic clustering method, e.g. k-Medoids clustering, can be applied to identify a set of representative operational periods and their corresponding weights. Here, the clustering algorithm seeks to minimize the objective introduced in (2) using distances calculated from the $(d'|\mathcal{N}'|)$-dimensional latent feature vectors $\{\mathbf{Z}^{(1)}, \ldots, \mathbf{Z}^{(|\mathcal{T}|)}\}$ rather than the $(\sum_s |\mathcal{N}^s| d_s)$-dimensional observed feature vectors.

Then, $|\mathcal{T}|$ spatial aggregations can be retrieved by passing each of the $|\mathcal{T}|$ observations through the pooling GCN block. From here, a single spatial aggregation can be chosen from the $|\mathcal{T}|$ candidates (many of which may be identical). One approach, which we chose for our case study, is to assign each node's respective group by taking a "vote" of group memberships over the $|\mathcal{T}|$ operational periods. Another possible approach would be to weight the vote based on the learned temporal aggregation (i.e., cluster median weights).

We note that the spatial and temporal aggregations do not need to be retrieved from the same trained autoencoder. Rather, one can select different inputs, hyperparameters, and architectures for spatial and temporal aggregation. Indeed, it is preferable to limit the inputs to only the most relevant parameters for aggregation as deep neural models can be trained more efficiently and with higher performance on lower dimensional inputs.

5 CASE STUDY

We built on the joint power-NG planning problem proposed in [13]. Although their model is a generation and transmission expansion problem, here we consider a simpler problem that can be treated as a GEP; see Supplementary Information (SI) [36] for full description of the problem. The problem is to determines the minimum investment and operational costs of co-optimizing electricity and NG systems for the year 2050 under various investment, operational, and policy constraints. The investment decisions for the power system include establishing new plants and decommissioning existing plants. The decisions for the NG include establishing new pipelines. Our model accounts for major planning constraints such as minimum stable production, ramping, energy balance, and storage. The interdependency between the two systems is captured two sets of constraints. The first coupling constraint captures the flow of NG to power system to enable the operation of gas-fired power plants. The second constraint imposes a system-wide emission constraint, limiting the emissions of CO_2 from both systems to a pre-specified value.

This GEP considers both electric power and NG nodes, each having their own input parameters and operating on different time resolutions; hence the problem is an example of CEPHN. The power system operates on hourly basis whereas the NG system operations are carried out on daily resolution. In the formulation presented in the SI, set of node types consist of power and NG (i.e., $\mathcal{S} = \{1, 2\}$). The investment decisions for the power system $\boldsymbol{x}^1$ include investment in the generation and storage technologies at each node and storage. Part of the investment decisions such as investment in thermal generation is assumed to be integer decisions, and all other variables are defined as continuous. The operational decisions for the power system include power generation, storage and load shedding which collectively form $\boldsymbol{y}^1$. The variable $\boldsymbol{x}^2$ for NG system include investment in pipeline expansion which is a binary variable. The operational variables $\boldsymbol{y}^2$ include fuel supply, flow, and load shedding.

We used the input data provided in [13]. The authors construct the New England power and NG networks by publicly available data. The power network consists of 188 buses located in 88 distinct nodes (locations). The NG network consists of 18 NG nodes. The input data considers 12 power plants types including 5 existing and 7 new plant types. The plants types represent a range of generation options such as various gas-fired plants, wind, solar, hydro and nuclear power generators.

6 EXPERIMENTAL SETUP

We use the case study in Section 5 to evaluate the performance of our approach and consider power demand, NG demand, and capacity factor (CF) as input data. We obtain five instances of the aggregated problem each with as follows:

- Aggregation based on U.S. states; nodes that are located in the same state are represented as a single node. This aggregation is common practice in the literature and serves as a benchmark for our case study.
- Aggregation based on pooling loss only (PL); values of α_R, α_P, and α_H in Eq. (6) are set to 0, 1, and 0 respectively.
- Aggregation based on pooling and reconstruction losses (PRL); values of α_R, α_P, and α_H in Eq. (6) are set to 1, 1, and 0 respectively.
- Aggregation based on pooling and entropy losses (PHL); values of α_R, α_P, and α_H in Eq. (6) are set to 0, 1, and 1 respectively.
- Aggregation based on pooling, reconstruction, and entropy losses (PRHL); values of α_R, α_P, and α_H in Eq. (6) are set to 1, 1, and 1 respectively.

For temporal aggregation, we consider 5, 10, 20, and 40 representative operational periods out of 365 days. We obtain four values for each number of representative days. We use k-medoid to cluster the raw data which includes demands and CFs. We also applied PCA to the raw data and clustered the resulting data. These two methods are commonly used in the literature. We also consider two outcomes from the proposed approach. In the first outcome (A1), we only consider power demand, whereas the second outcome (A2) is a result of considering power demand, NG demand, and CFs. This experimental setup results in 80 instances.

All instances are implemented in Python using Gurobi 10.0 and are available at the GitHub page [36]. All instances are run on the MIT Supercloud system, which uses an Intel Xeon Platinum 8260 processor with up to 48 cores and 192 GB of RAM [28]. We limit the CPU time to 3 hours for all instances, by which point all models were solved to optimality with a mixed integer gap of 1% or lower.

7 RESULTS AND DISCUSSION

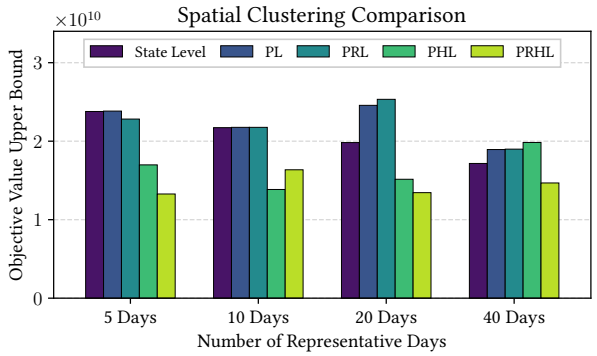
To evaluate each instance of the spatio-temporal aggregation, we propose a heuristic approach that generates a feasible solution for the full problem (88 nodes and the entire year). The approach consists of three steps.

(1) We first solve the spatially and temporally aggregated problem and retrieve investment decisions for each cluster of nodes.
(2) We then consider the full network for only two representative days and add constraints to ensure that the sum of investment decisions over all nodes represented by a group do not exceed the number of investment decisions for the group in the first step. For example, suppose that five solar plants are established in the first step for node group 7, and

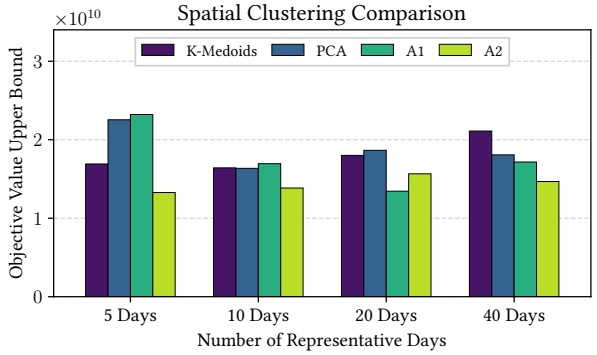
assume that nodes 3, 6, and 44 are represented by group 7. Then in the second step, we constrain the sum of established solar plants across nodes 3, 6, and 44 to be at most five. The result of this step provides investment decisions for each node.

(3) Finally, we consider the full network over the whole year and fix the investment decisions returned by Step 2. This renders the problem as a linear program in which the only decision variables correspond to continuous-valued operational decisions.

The solution to the third step provides a feasible solution to the original CEPHN, and consequently is an upper bound (UB).



(a) UB comparison for spatial aggregation methods. Each bar corresponds to the *lowest* UB computed for the given spatial aggregation method across *all* temporal aggregation methods.



(b) UB comparison for temporal aggregation methods. Each bar corresponds to the *lowest* UB computed for the given temporal aggregation method across *all* spatial aggregation methods.

Fig. 3a illustrates the best performance of different spatial reduction methods across four temporal aggregation methods. We note that spatial aggregations from the proposed approach outperform in all representative days sizes. PRHL, which is trained to minimize the pooling, reconstruction, and negative entropy losses, outperforms all other methods including the state level aggregation commonly used in the literature for all but 10 representative days. Its performance is followed by PHL, which is trained to minimize the pooling and negative entropy losses. Overall, the upper bound obtained by instances that are spatially aggregated using PRHL is 33%, 40%, 39%, and 10% better than the upper bounds obtained from aggregating using state level, PL, PRL, and PHL, respectively (after averaging across all temporal aggregation methods and numbers of representative days). The result suggest that the inclusion of the negative entropy loss in conjunction with the reconstruction loss can provide superior spatial aggregations as evaluated by our 3-step upper bound procedure.

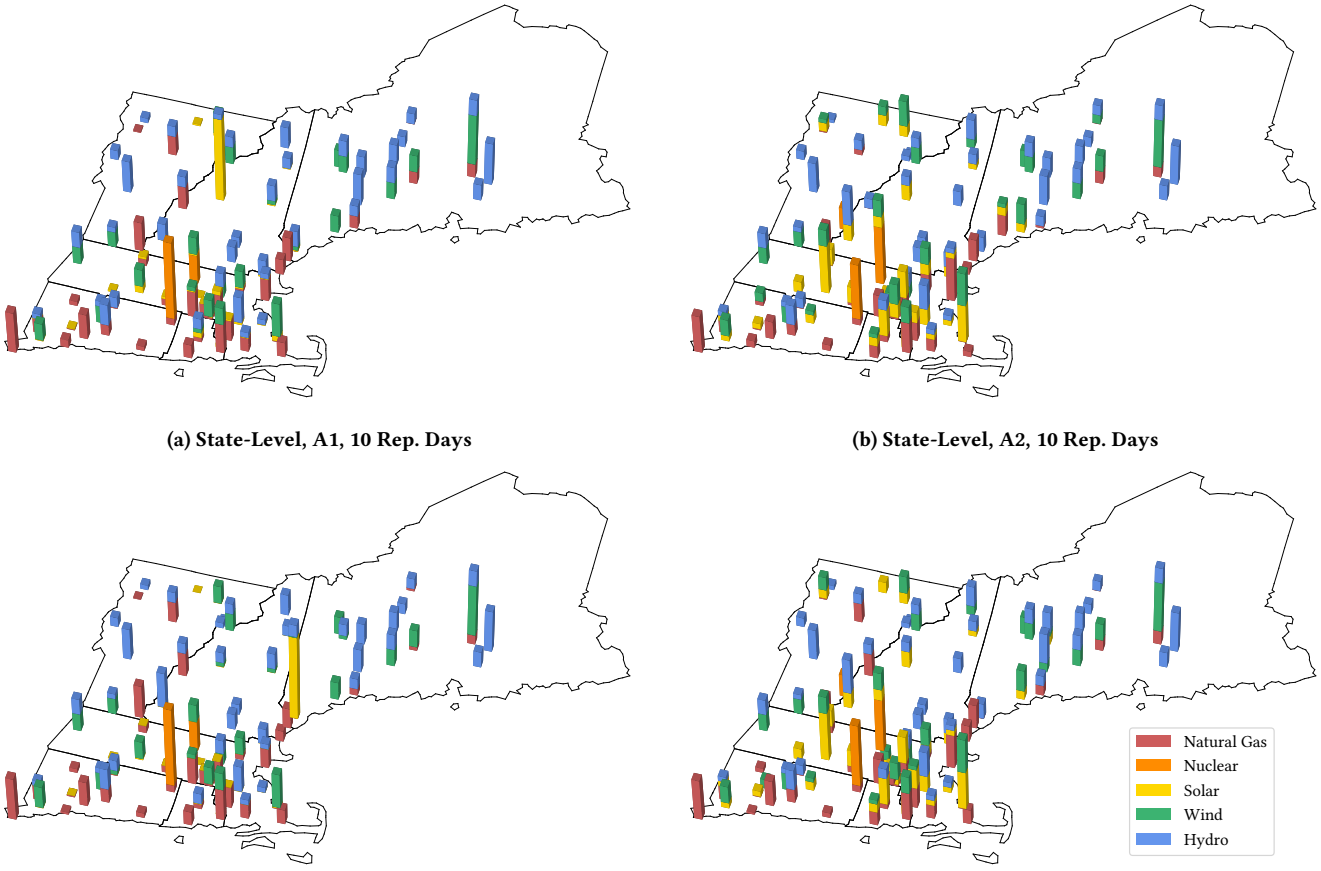
The best performance observed for the four temporal aggregation methods across the considered spatial aggregations is shown in Fig. 3b. A2, which is trained to minimize reconstruction error for power demands, NG demands, and CFs, outperforms the other temporal aggregation methods for all but 20 representative days. The feasible solution corresponding to the temporal aggregation from A1 yields the worst upper bound for 5 representative days, but A2 performs significantly better for the same number of representative days, highlighting the significance of including NG demand and CF data in training the autoencoder. Furthermore, A2 demonstrates more consistent upper bounds across different number of representative days. This is significant, as some large-scale CEPHNs can only be solved for a small number of representative days. Consequently, a temporal aggregation method that poorly captures the variability of supply-demand can have significant implications for planning and operations [3]. Overall, the investment decisions retrieved from solving the temporal aggregation using A2 yields an upper bound that is 10%, 9%, and 7% better than k-medoid, PCA and A1, respectively (after averaging across all spatial aggregation methods and numbers of representative days).

Generation mix is another result that significantly impacts planning outcomes. Fig. 4 illustrates power generation across different sources as solved for using two spatial and two temporal aggregation methods for 10 representative days. We compare state-level and PRHL with A1 and A2 to compare the generation outcomes from the proposed approach to aggregations commonly used in the literature. We find that the choice of temporal aggregation has a greater impact than the method for spatial aggregation. In particular, the inclusion of NG demand and CFs in A2 significantly increases the share of solar generation and distributes its generation more uniformly across different nodes, whereas in A1, we observe solar generation is concentrated in a smaller set of locations. We also observe, albeit to a smaller degree, differences in generation from wind and NG resulting from the two temporal aggregation methods.

8 CONCLUSION

In this paper, we present a graph convolutional autoencoder approach to spatio-temporally aggregate capacity expansion problems with heterogeneous nodes. We leverage a graph pooling operation to automatically identify nodes exhibiting similar characteristics. Moreover, our proposed architecture utilizes graph convolutions to exploit network interactions within and across subsystems. We also introduce a multi-objective training loss which can be tuned to capture varying subsystem relevance and desirable qualities of spatial aggregations.

We apply our approach to a generation expansion problem for the New England joint power-NG network. Specifically, we experiment with reducing the original problem size from 88 to 6 nodes using various trade-offs between components of the multi-objective



(a) State-Level, A1, 10 Rep. Days

(b) State-Level, A2, 10 Rep. Days

(c) PRHL, A1, 10 Rep. Days

(d) PRHL, A2, 10 Rep. Days

Figure 4: Generation mix comparison across four spatio-temporal aggregations. To make comparison easier, the height of each bar is normalized according to the total generation for its corresponding source.

loss function. We also experiment with applying our temporal aggregation approach to retrieve two sets of representative days, one of which is obtained using only power demand data, while the other incorporates power demands, NG demands, and capacity factors for variables renewable generators (i.e., solar and wind). We consider common spatial and temporal aggregation methods from the literature and compare the solutions associated with each of these methods. We find that solutions resulting from our approach significantly outperform those obtained from existing methods for our case study.

Future studies may extend the approach and compute additional experiments made in this work in several directions. For one, additional experiments can be performed with different CEPHN formulations to more comprehensively understand the relationship connecting aggregation resolution to both solution quality and sensitivity of the learned planning decisions. Moreover, the sensitivity of aggregations, and consequently planning decisions, with respect to changes in the dataset should be investigated.

ACKNOWLEDGMENTS

This work is supported by funding from MIT Energy Initiative Future Energy Systems Center, MIT Climate Grand Challenges grant for "Preparing for a new world of weather and climate extremes", MIT School of Engineering Stockham Fellowship, and C3.ai grant for "CLAIRE: Causal reasoning for real-time attack localization in cyber-physical systems".

REFERENCES

[1] Shabbir Ahmed and Nikolaos V Sahinidis. 2003. An approximation scheme for stochastic integer programs arising in capacity expansion. *Operations Research* 51, 3 (2003), 461–471.

[2] Marc Barbar and Dharik S Mallapragada. 2022. Representative period selection for power system planning using autoencoder-based dimensionality reduction. *arXiv preprint arXiv:2204.13608* (2022).

[3] Jeffrey A Bennett, Claire N Trevisan, Joseph F DeCarolis, Cecilio Ortiz-García, Marla Pérez-Lugo, Bevin T Etienne, and Andres F Clarens. 2021. Extending energy system modelling to include extreme weather risks and application to hurricane events in Puerto Rico. *Nature Energy* 6, 3 (2021), 240–249.

[4] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. 2020. Spectral Clustering with Graph Neural Networks for Graph Pooling. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Article 82, 10 pages.

[5] Espen Flo Bødal, Audun Botterud, and Magnus Korpås. 2022. Capacity Expansion Planning with Stochastic Rolling Horizon Dispatch. *Electric Power Systems Research* 205 (2022), 107729.

[6] Espen Flo Bødal, Dharik Mallapragada, Audun Botterud, and Magnus Korpås. 2020. Decarbonization synergies from joint planning of electricity and hydrogen production: a Texas case study. *international journal of hydrogen energy* 45, 58 (2020), 32899–32915.

[7] Aron Brenner, Rahman Khorramfar, Dharik Mallapragada, and Saurabh Amin. 2022. Graph Representation Learning for Energy Demand Data: Application to Joint Energy System Planning under Emissions Constraints. *arXiv preprint arXiv:2209.12035* (2022).

[8] Caetano CS Fraga, Josué Medellín-Azuara, and Guilherme F Marques. 2017. Planning for infrastructure capacity expansion of urban water supply portfolios with an integrated simulation-optimization approach. *Sustainable cities and society* 29 (2017), 247–256.

[9] Michel Gendreau, Jean-Yves Potvin, Ali Smires, and Patrick Soriano. 2006. Multi-period capacity expansion for a local access telecommunications network. *European Journal of Operational Research* 172, 3 (2006), 1051–1066.

[10] Na Geng and Zhibin Jiang. 2009. A review on strategic capacity planning for the semiconductor manufacturing industry. *International journal of production research* 47, 13 (2009), 3639–3655.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[12] Maximilian Hoffmann, Leander Kotzur, Detlef Stolten, and Martin Robinius. 2020. A review on time series aggregation methods for energy system models. *Energies* 13, 3 (2020), 641.

[13] Rahman Khorramfar, Dharik Mallapragada, and Saurabh Amin. 2022. Electric-Gas Infrastructure Planning for Deep Decarbonization of Energy Systems. *arXiv preprint arXiv:2212.13655* (2022).

[14] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. https://doi.org/10.48550/ARXIV.1609.02907

[15] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308* (2016).

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. https://openreview.net/forum?id=SJU4ayYgl

[17] Nikolaos E Koltsaklis and Athanasios S Dagoumas. 2018. State-of-the-art generation expansion planning: A review. *Applied energy* 230 (2018), 563–589.

[18] Can Li, Antonio J Conejo, Peng Liu, Benjamin P Omell, John D Siirola, and Ignacio E Grossmann. 2022. Mixed-integer linear programming models and algorithms for generation and transmission expansion planning of power systems. *European Journal of Operational Research* 297, 3 (2022), 1071–1082.

[19] Can Li, Antonio J Conejo, John D Siirola, and Ignacio E Grossmann. 2022. On representative day selection for capacity expansion planning of power systems under extreme operating conditions. *International Journal of Electrical Power & Energy Systems* 137 (2022), 107697.

[20] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. , 8 pages.

[21] Yixian Liu, Ramteen Sioshansi, and Antonio J Conejo. 2017. Hierarchical clustering to find representative operating periods for capacity-expansion modeling. *IEEE Transactions on Power Systems* 33, 3 (2017), 3029–3039.

[22] Hanan Luss. 1982. Operations Research and Capacity Expansion Problems: A Survey. *Oper. Res.* 30 (1982), 907–947.

[23] Hongguang Ma, Xiang Li, and Yankui Liu. 2020. Multi-period multi-scenario optimal design for closed-loop supply chain network of hazardous products with consideration of facility expansion. *Soft Computing* 24 (2020), 2769–2780.

[24] Dharik S Mallapragada, Dimitri J Papageorgiou, Aranya Venkatesh, Cristiana L Lara, and Ignacio E Grossmann. 2018. Impact of model resolution on scenario outcomes for electricity sector system expansion. *Energy* 163 (2018), 1231–1244.

[25] James H Merrick. 2016. On representation of temporal variability in electricity capacity planning models. *Energy Economics* 59 (2016), 261–274.

[26] HyungSeon Oh. 2009. A new network reduction methodology for power system planning studies. *IEEE Transactions on Power Systems* 25, 2 (2009), 677–684.

[27] Lance Parsons, Ehtesham Haque, and Huan Liu. 2004. Subspace Clustering for High Dimensional Data: A Review. *SIGKDD Explor. Newsl.* 6, 1 (jun 2004), 90–105. https://doi.org/10.1145/1007730.1007731

[28] Albert Reuther, Jeremy Kepner, Chansup Byun, Siddharth Samsi, William Arcand, David Bestor, Bill Bergeron, Vijay Gadepally, Michael Houle, Matthew Hubbell, et al. 2018. Interactive supercomputing on 40,000 cores for machine learning and data analysis. , 6 pages.

[29] Morten Riis and Kim Allan Andersen. 2004. Multiperiod capacity expansion of a telecommunications connection with uncertain demand. *Computers & operations research* 31, 9 (2004), 1427–1436.

[30] Francisca Rosell and Esteve Codina. 2020. A model that assesses proposals for infrastructure improvement and capacity expansion on a mixed railway network. *Transportation Research Procedia* 47 (2020), 441–448.

[31] Guillaume Salha, Romain Hennequin, and Michalis Vazirgiannis. 2019. Keep it simple: Graph autoencoders without graph convolutional networks.

[32] Bruno U. Schyska, Alexander Kies, Markus Schlott, Lueder von Bremen, and Wided Medjroubi. 2021. The sensitivity of power system expansion models. *Joule* 5, 10 (2021), 2606–2624.

[33] Ian J Scott, Pedro MS Carvalho, Audun Botterud, and Carlos A Silva. 2019. Clustering representative days for power systems generation expansion planning: Capturing the effects of variable renewables and energy storage. *Applied Energy* 253 (2019), 113603.

[34] Di Shi. 2012. *Power system network reduction for engineering and economic analysis*. Arizona State University.

[35] David Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2012. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Processing Magazine* 30 (10 2012).

[36] SI-Code. 2022. Supplementary material available at: https://github.com/RahmanKhorramfar91/ICCPS-2023. https://github.com/RahmanKhorramfar91/ICCPS-2023

[37] Kavinesh J Singh, Andy B Philpott, and R Kevin Wood. 2009. Dantzig-Wolfe decomposition for solving multistage stochastic capacity-planning problems. *Operations Research* 57, 5 (2009), 1271–1286.

[38] Gabriel Taubin. 1995. A Signal Processing Approach to Fair Surface Design. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. Association for Computing Machinery, New York, NY, USA, 351–358. https://doi.org/10.1145/218380.218473

[39] Holger Teichgraeber and Adam R Brandt. 2019. Clustering methods to find representative periods for the optimization of energy systems: An initial framework and comparison. *Applied energy* 239 (2019), 1283–1293.

[40] Holger Teichgraeber and Adam R Brandt. 2022. Time-series aggregation for the optimization of energy systems: Goals, challenges, approaches, and opportunities. *Renewable and Sustainable Energy Reviews* 157 (2022), 111984.

[41] Holger Teichgraeber and Adam R Brandt. 2022. Time-series aggregation for the optimization of energy systems: Goals, challenges, approaches, and opportunities. *Renewable and Sustainable Energy Reviews* 157 (2022), 111984.

[42] Holger Teichgraeber, Constantin P Lindenmeyer, Nils Baumgärtner, Leander Kotzur, Detlef Stolten, Martin Robinius, André Bardow, and Adam R Brandt. 2020. Extreme events in time series aggregation: A case study for optimal residential energy supply systems. *Applied energy* 275 (2020), 115223.

[43] Anton Tsitsulin, John Palowitch, and Bryan Perozzi. 2020. Graph Clustering with Graph Neural Networks. https://arxiv.org/abs/2006.16904

[44] Gregory Von Wald, Kaarthik Sundar, Evan Sherwin, Anatoly Zlotnik, and Adam Brandt. 2022. Optimal Gas-Electric Energy System Decarbonization Planning. *Advances in Applied Energy* (2022), 100086.

[45] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 4805–4815.

[46] Ryohei Yokoyama, Yuji Shinano, Yuki Wakayama, and Tetsuya Wakui. 2019. Model reduction by time aggregation for optimal design of energy supply systems by an MILP hierarchical branch and bound method. *Energy* 181 (2019), 782–792.

[47] Matteo Zatti, Marco Gabba, Marco Freschini, Michele Rossi, Agostino Gambarotta, Mirko Morini, and Emanuele Martelli. 2019. k-MILP: A novel clustering approach to select typical and extreme days for multi-energy systems design optimization. *Energy* 181 (2019), 1051–1063.