Research On Optimal Control Strategies Based On Deep Reinforcement Learning Under Differential Algebraic Equation

Guoxing Si*

Basic Department of Zhengzhou Vocational College of Industrial Safety, Zhengzhou, Henan, 451192, China

** Corresponding author. E-mail: zhengzhousi66@126.com*

Received: Mar. 12, 2026; Accepted: Jun. 19, 2026

Optimal control of differential-algebraic equations represents a major challenge due to the intricate relationship between the state dynamics and the algebraic constraints, especially for large-scale power transmission systems with discrete topology actions. The limitations of conventional controllers and unconstrained deep reinforcement learning algorithms lie in the failure to ensure the feasibility, safety, and long-term stability of the system while maintaining the DAE simulation consistency. To overcome the limitations of the existing solutions, the present work proposes a structured DRL approach, which integrates the Attentive Differentiable Structure for Control Optimization and the Policy Embedded Gradient Field Navigation algorithms. The ADSCO method represents the grid observations and the topology actions using a shared latent space, while the attentive energy modeling ensures the state-action compatibility using feasibility-aware signals from the DAE constraints. The PEGFN method refines the policy outputs using the constraint-informed gradient field defined on the relaxed action simplex. The effectiveness of the proposed method was tested using the L2RPN and RL2Grid test cases and proved to give better results compared to both previous methods and the current best reinforcement learning algorithms. Specifically speaking about the paper under discussion, 'simulation consistency with respect to DAEs' is the property that ensures that all control measures that are implemented within the proposed framework are consistent with the dynamics of the power system represented by DAEs.

Keywords: Optimal control (OC), DRL, differential-algebraic systems, interpretable AI, intelligent systems

© The Author('s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are cited.

http://dx.doi.org/10.6180/jase.202610_33.058

1. Introduction

As the complexity of engineering systems grows, there is a need to develop new methods for controlling Dynamical Systems where Differential Algebraic Equations (DAEs) [1] form the basis of modeling. The main reason for such a need is the combination of the Differential and Algebraic sections in DAEs modeling, making it difficult to apply conventional control approaches as well as the system's stability analysis [2]. Conventional Model-based Control approaches also fail to tackle high-dimensional non-linear systems under feasibility constraints [3]. As a result, adaptive control approaches should be developed for coping with algebraic constraints alongside enhancing the system performance by interacting with the environment [4]. DRL approach is suitable for solving this issue because of its learning capacity through trial-and-error interactions [5, 6].

The first research on the control of the DAE system has been carried out using mostly model-based methods that use concrete mathematical models and solid theoretical foundations [7, 8]. The model-based method gives the possibility of designing the controller through the application of pre-defined models of the systems and provides reliable theoretical guarantees for stability, feasibility, model predictive control, feedback linearization, and Lyapunov stability theory [9]. The model-based method loses its efficiency

in the case of inaccuracies in modeling, disturbance, uncertainty, and real-time computation [10]. The increase in the number of variables, algebraic equations, and stiffness complicates the realization of the method in CPS and DAE [11, 12].

This requirement gave rise to the development of approximate methods which included statistical modeling and light-weight prediction algorithms [13]. These methods helped address inflexibility of the analytical control through surrogate modeling of the dynamics [14]. On the other hand, for DAE constrained control problems, hybrid approaches that included adaptive learning of differential dynamics and algebraic constraint handling using projection, regularization or optimization were developed [15]. While flexible, especially in the case of partial observability, these methods have their disadvantages too. For example, getting an accurate surrogate model is possible only if there is enough data available [16]. Besides, lack of prior structural knowledge reduces generalizability in nonlinear highdimensional systems and may result in infeasible solutions due to poorly handled constraints [17].

DRL applied to differential-algebraic equations (DAEs) makes autonomous control more flexible and robust [18]. Deep networks learn to map the high-dimensional state space of DAEs into actions by learning the nonlinear temporal relationships between them without a model during interactions [19, 20]. Constraint handling in DRL based on DAE can be done using penalties or specific architectures [21, 22]. Issues associated with DRL based on DAE include gradient variance, hyperparameters, and reward sparsity [23].

Using techniques such as Deep Q Networks, Proximal Policy Optimization, and Soft Actor-Critic, the deep reinforcement learning technique facilitates learning of nonlinear control policy in complex and high-dimensional control applications [24]. The technique has been applied to robotic systems, automated driving, power system engineering, and other stochastic control applications [25]. Although deep reinforcement learning does not depend on system dynamic models and thus is applicable to hard dynamic control applications, there are some problems in applying deep reinforcement learning in control engineering in terms of efficiency, stability, and safety of the process. These challenges can be addressed through using the synergy of the reinforcement learning techniques and other methodologies such as Lyapunov techniques, Hamilton-Jacobi-Bellman formulation, robust control, randomization, imitation learning, and curriculum learning [26]. Safe reinforcement learning refers to the process of optimizing policies considering both state and action constraints. There are several review articles concerning constrained formulations and algorithms [27], constrained Markov Decision Process [28], safety reward optimization and convergence properties [29], real-time stabilization with state constraints [29], and continuous-time nonlinear system and Barrier Lyapunov Function approach [30]. In the field of related studies in DAE and control, there are studies dealing with optimal control for linear-quadratic infinite dimensional DAE [31], DAE theory, observers, optimal control and algebraic aspects [32], and the extension of linear-quadratic control in Hilbert-space DAE with Popov operator [33].

With constraints imposed on a system, e.g., power systems, chemical process systems, and mechanical systems with kinematic constraints, we can derive differential-algebraic equations. DAEs consist of both algebraic and differential equations with algebraic equations modeling system constraints in contrast to ordinary differential equations (ODEs). The solution of DAEs and its analysis and control become difficult processes [34]. DAEs can be classified according to index that defines the number of differentiations to convert the DAE to an ODE; higher index DAEs tend to be more difficult to solve because of their sensitivity to constraint satisfaction [35]. Conventional control methods like projection methods, index reduction, and geometric control require accurate models and are therefore not suitable for many applications [36]. On the other hand, control of DAE requires constraint satisfaction approaches that meet optimization and constraint satisfaction requirements, like linear and nonlinear MPC, feedback linearization, and differential geometric control [37].

Condensed version with references preserved:

Integrating deep reinforcement learning with DAE-constrained systems is important for learning control policies that satisfy structural constraints. A key challenge is ensuring that policies respect algebraic constraints during both training and deployment, since violations can produce infeasible control actions and make the problem more difficult than unconstrained DRL [38]. Existing methods address this through constraint-enforcing simulation environments, reward-penalty terms, and constrained reinforcement learning or optimization techniques based on Karush-Kuhn-Tucker conditions, although penalty-based approaches often require tuning and may not guarantee constraint satisfaction [39]. For DAEconstrained systems, differentiable simulators and physics-informed neural networks offer stronger constraint-satisfaction guarantees, but challenges remain in stability and generalization [33, 34]. Related studies include Hammad Saeed et al. [40] improved DC microgrid current sharing and voltage control using gray-wolf-optimized adaptive droop control, and

Nurpatsha [41] compared SNN, Bi-LSTM, CNN, and GRU for Alexa voice command classification, showing strong CNN performance in recall and accuracy and SNN performance in precision and AUC.

Despite some improvements being witnessed in the development of sliding mode and super twisting control strategies, some weaknesses still remain in the form of a lack of guarantee of feasibility, scalability to high-dimensional spaces, and the capability of dealing with scenarios where there is partial observability of the system states. In addition, the traditional approach uses models to achieve the required control, which poses a challenge when dealing with nonlinearities, high dimensionality, or incomplete knowledge of the system dynamics.

In an attempt to address the problems faced by earlier research, this methodology integrates the ideas of deep reinforcement learning and DAE constraints. Deep reinforcement learning is used to learn the latent embeddings and the attention scores for checking the feasibility of state for discrete topological actions. However, the DAE constraints help enhance the performance of the policy via gradient updates on the constrained action simplex space. The proposed approach deviates from the existing approaches such as constrained reinforcement learning and energy-based models.

2. Materials and methods

2.1. Overview

This section presents a combined approach to solve the optimal topology control of the power transmission grid with DAE constraints. It is very simple to describe the power transmission grid using DAEs, but the topology control process needs some discrete actions such as switching. It would be hard to solve this problem with traditional modeling and combinatorial optimization algorithms considering the dimensionality, randomness, and realtime nature of the problem. Although the DRL is suitable to perform sequential decision tasks, the constrained conditions of the DAEs can be ignored while using the conventional DRL algorithm. Thus, this problem is formulated as a constrained sequential decision problem as the conventional environments L2RPN and RL2Grid.

2.2. Preliminaries

Several assumptions are made to simplify modeling and support effective algorithm training. Quasi-steady-state dynamics are assumed between control actions, allowing a DAEbased model without high-frequency transients. Algebraic constraints represent physical limits such as voltage magnitudes, power balance, and line capacities, while topology changes are treated as instantaneous control actions. Uncertainty in load and renewable generation is handled through scenario-based modeling rather than probabilistic modeling.

Component interactions are captured by the DAE simulator, and disturbances are modeled by adding small Gaussian noise to observations during training.

Formalizing the topology control problem has begun for power transmission networks under DAE constraints. Let $\mathcal{X} \subseteq \mathbb{R}^n$ denote the continuous dynamic state space (e.g., generator or other slow system states), and let $\mathcal{Y} \subseteq \mathbb{R}^q$ denote algebraic variables associated with network constraints (e.g., bus voltage magnitudes and phase angles). The control process evolves over a finite decision horizon indexed by discrete time steps $t = 0, 1, \ldots, T$. The power system dynamics were governed by a DAE of the form of Eq. (1) and Eq. (2):

$$\dot{\mathrm{x}}(t)\& = f(\mathrm{x}(t), \mathrm{y}(t), \mathrm{a}(t)) \tag{1}$$

$$0\& = g(\mathrm{x}(t), \mathrm{y}(t), \mathrm{a}(t)) \tag{2}$$

Formulation: $f(\cdot)$ is the differential equation of the system, while $g(\cdot) = 0$ is the algebraic equation of the system such as power flow equations and Kirchhoff's law. The control variable $\mathrm{a}(t) \in \mathcal{A}$ is a discrete topology change, such as line switching or busbar switching. In order to obtain the value of the algebraic variables $\mathrm{y}(t)$, it need to solve Eq. [eq: dae_alg] corresponding to each state-action pair $(\mathrm{x}(t), \mathrm{a}(t))$. Thus, the topology control problem can be formulated as a decision-making problem under constraint where the state $s_t \in \mathcal{S}$ represents line loading, voltage profile, topology, and contingencies, and the action $a_t \in \mathcal{A}$ results in state transition as per Eq. (3):

$$s_{t+1} \sim P\left(\cdot \mid s_t, a_t\right) \tag{3}$$

where the transition kernel P is defined implicitly by DAE-consistent power system simulation based on [eq:dae diff]-[eq:dae alg]. The control objective is to learn a policy $\pi(a \mid s)$ that maximizes the expected cumulative operational reward as Eq. (4):

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \gamma^t \, r(s_t, a_t) \right] \tag{4}$$

where $r\left(s_t, a_t\right)$ captures grid operation objectives such as overload mitigation and system survivability, and $\gamma \in (0, 1)$ is a discount factor. Operational and safety constraints are expressed through cost functions as Eq. (5):

$$c^{(k)}\left(s_t, a_t\right) \geq 0, \quad k = 1, \ldots, K \tag{5}$$

which quantify violations of physical or security limits (e.g., line overloads or voltage violations). These constraints define a constrained Markov decision process (CMDP), where feasible control policies must satisfy prescribed budget limits on cumulative constraint costs.

To support the proposed modeling and optimization strategy, the feasible control manifold was defined as Eq. (6):

$$M := \Big\{ (s,a) \in S \times A \;\Big|\; \text{the DAE (1)–(2) admits a physically feasible solution} \Big\} \tag{6}$$

This manifold characterizes the subset of topology actions that are consistent with the underlying power system physics and serves as the foundation for the constraint-aware policy design introduced in the subsequent sections.

2.3. Attentive Differentiable Structure for Control Optimization (ADSCO)

The method ADSCO is applicable for DAE-constrained topology control of power grid in domains such as L2RPN and RL2Grid, where the action spaces consist of discrete topology actions and the validity of each action is determined by DAE simulation. This method improves the learning of policies by incorporating the representation of grid observation and topology action in a common latent space as well as employing a differentiable score function with constraint awareness to eliminate invalid topology actions.

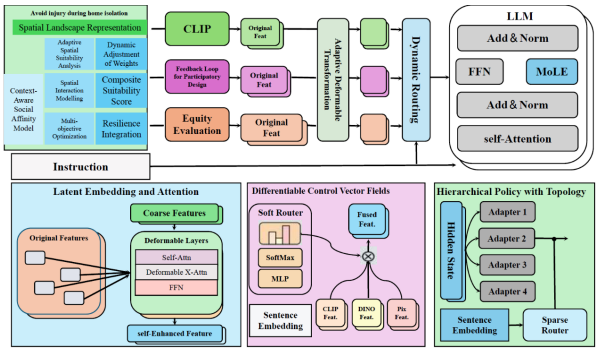

Fig. 1. Illustration of ADSCO for topology control

Fig. 1 summarizes the ADSCO pipeline for DAE-constrained power grid operation, including (i) latent embedding of grid observations and topology actions, (ii) attention-based action compatibility scoring, and (iii) a constraint-aware objective that favors feasible and secure topology decisions.

2.3.1. *Latent Embedding and Attention*

Let $s \in \mathcal{S}$ denote the observable grid state returned by the simulator, and let $a \in \mathcal{A}$ denote a discrete topology action. ADSCO defines an action-scoring function $\mathcal{F} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ that maps a state-action pair to a scalar energy (lower is better). Instead of treating this mapping as a black box, a structured embedding approach with modular latent components is adopted as Eq. (7):

$$\mathcal{F}(s,a) = \psi(\phi(s), \theta(a), \gamma(s,a)) \tag{7}$$

where $\phi : \mathcal{S} \to \mathbb{R}^d$ encodes the grid state into a d-dimensional latent space, $\theta : \mathcal{A} \to \mathbb{R}^d$ encodes the topology action, and $\gamma : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ captures attention-based interaction features between states and actions. The attention mechanism prioritizes topology actions that are most compatible with the current grid state. An attention kernel $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$ is defined by a SoftMax-normalized similarity as Eq. (8):

$$\kappa(\mathrm{p},\mathrm{q}) = \frac{\exp\left(\mathrm{p}^\top \mathrm{q}/\tau\right)}{\sum_{a' \in \mathcal{A}} \exp\left(\mathrm{p}^\top \theta\left(a'\right)/\tau\right)} \tag{8}$$

where $\tau > 0$ is a temperature, $\mathrm{p} = \phi(s)$, and $\mathrm{q} = \theta(a)$. This kernel normalizes compatibility scores across the discrete action set. The attention-modulated interaction map is obtained by aggregating action embeddings weighted by compatibility as Eq. (9):

$$\gamma(s,a) = \kappa(\phi(s), \theta\vartheta(a)) \cdot \vartheta(a) \tag{9}$$

The problem formulation requires the representations of both states of the system and actions to be in the same latent space, enabling the assessment of the actions through comparison of their similarity to the present grid state. In such a manner, those topological actions that are most similar to the state of the system will be preferred. The selection of actions is performed by minimizing the action energy, see Eq. (10):

$$\hat{a} = \underset{a \in \mathcal{A}}{\operatorname{argmin}} \mathcal{F}(s,a) \tag{10}$$

For differentiable learning, the corresponding Boltzmann policy is induced by the energy as Eq. (11):

$$\pi(a \mid s) = \frac{\exp(-\mathcal{F}(s,a))}{\sum_{a' \in \mathcal{A}} \exp\left(-\mathcal{F}\left(s,a'\right)\right)} \tag{11}$$

To enhance the smoothness of the action-energy landscape and stabilize learning, an auxiliary regularization term over the energy scores was introduced as Eq. (12):

$$\mathcal{E}(s) = \lambda_{\text{smooth}} \cdot \sum_{a \in \mathcal{A}} \left\| \nabla_{\theta(a)} \mathcal{F}(s,a) \right\|_2^2 \tag{12}$$

Where $\lambda_{\text{smooth}} > 0$ controls the strength of smoothness regularization on the action embedding space.

The above-mentioned scoring technique provides an energy score for every possible activity. This implies that activities with low energies conform more to the constraints of the system. Activities are selected in a random manner on the basis of the provided scores.

2.3.2. *Differentiable Control Vector Fields*

Topology actions are discrete; therefore, vector-field refinement is performed on a continuous relaxation of the action distribution rather than directly on the action itself. Let $\mathrm{q}(s) \in \Delta^{|\mathcal{A}|}$ be a relaxed action probability vector on the simplex, with components $q_a(s)$ corresponding to actions $a \in \mathcal{A}$. The expected energy was defined under q as Eq. (13):

$$\mathcal{V}_\beta(s,\mathrm{q}) = \sum_{a\in\mathcal{A}} q_a(s)\,\mathcal{F}(s,a) + \beta_0 + \sum_{k=1}^{K} \beta_k \cdot \sigma_k(\phi(s)) \quad (13)$$

$\mathcal{V}_\beta$ can be represented as per Eq. (13) through a summation operation involving the products of a set of basic functions $\{\sigma_k\}_{k=1}^{K}$ and parameters $\{\beta_k\}_{k=1}^{K}$. In regard to the basic functions σ_k, they have been defined as RBFs whose center points are selected as the representative state-action pairs and fall within the domain of latent embeddings encountered by the neural network during training. Parameters β_k, on the other hand, have been learned alongside other neural network parameters by means of gradient descent.

Where $\{\sigma_k\}$ are smooth basis functions that shape the potential in state-embedding space. The induced corrective direction is defined as the negative gradient of the potential with respect to the relaxed distribution, as Eq. (14):

$$\mathrm{g}(s,\mathrm{q}) = -\nabla_{\mathrm{q}}\mathcal{V}_\beta(s,\mathrm{q}) = -[\mathcal{F}(s,a)]_{a\in\mathcal{A}} \quad (14)$$

Starting from an initial distribution q_t, ADSCO applies an iterative refinement step as Eq. (15):

$$\mathrm{q}_{t+1} = \Pi_\Delta\left(\mathrm{q}_t + \alpha \cdot \mathrm{g}\left(s_t, \mathrm{q}_t\right)\right) \quad (15)$$

Where $\alpha > 0$ is the step size and Π_Δ denotes projection onto the simplex to ensure $\mathrm{q}_{t+1} \in \Delta^{|\mathcal{A}|}$. To avoid overly sensitive representations, Lipschitz-style constraints on the encoders are imposed as Eq. (16):

$$\|\nabla_s \phi(s)\|_2 \le L_s, \quad \|\theta(a)\|_2 \le L_a \quad (16)$$

Where L_s and L_a control sensitivity of state embeddings and the magnitude of action embeddings, respectively.

2.3.3. *DAE-Feasibility Consistency Loss*

To ensure that the learned scoring function aligns with DAE feasibility, violation signals are returned by the simulator after applying an action. Let $v(s,a) \ge 0$ be a normalized violation measure (e.g., overload or voltage violation magnitude). A feasibility consistency loss is defined as Eq. (17):

$$\mathcal{C}(s,a) = \left(\mathcal{H}_\delta(s,a) - v(s,a)\right)^2 \quad (17)$$

Which penalizes a mismatch between the learned constraint score and the simulatordetected violations. This encourages ADSCO to assign a higher penalty to actions that lead to DAE-infeasible or insecure operating points.

2.3.4. *Hierarchical Policy with Topology*

The final component introduces a constraint-aware hierarchical policy Π_δ that explicitly accounts for topological feasibility. Rather than enforcing feasibility as an external post-processing step, this was embedded into a differentiable constraint representation (As shown in Fig. 2).

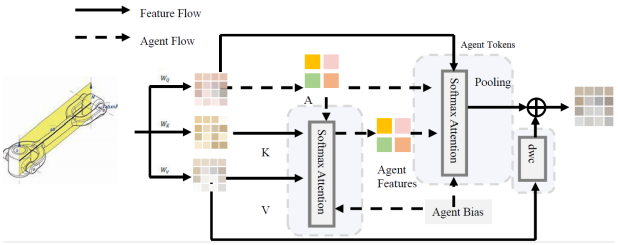

Fig. 2. Illustration of Hierarchical Policy with Topology

Fig. 2 summarizes the constraint-aware action scoring and aggregation mechanism for discrete topology control, where feasibility-related features are integrated into the policy objective via differentiable constraint heads.

A constraint scoring function is defined $\mathcal{H}_\delta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ parameterized by δ as Eq. (18):

$$\mathcal{H}_\delta(s,a) = \delta^\top \tanh\left(W[\phi(s) \bigoplus \theta(a)]\right) \quad (18)$$

Where W is a trainable matrix and $\tanh(\cdot)$ ensures bounded and smooth constraint features. This mapping captures non-linear feasibility and security surfaces induced by the underlying grid topology. The hierarchical policy is then defined by minimizing an augmented objective over discrete actions as Eq. (19):

$$\Pi_\delta(s) = \underset{a\in\mathcal{A}}{\operatorname{argmin}} \left\{\mathcal{F}(s,a) + \lambda \cdot \mathcal{H}_\delta(s,a)\right\} \quad (19)$$

Where $\lambda > 0$ balances operational performance and feasibility. To ensure stability near constraint boundaries, a soft-margin regularization is introduced as Eq. (20):

$$\mathcal{R}_{\text{margin}}(s,a) = \log\left(1 + \exp\left(\gamma \cdot \mathcal{H}_\delta(s,a)\right)\right) \tag{20}$$

Where $\gamma > 0$ controls the softness of constraint enforcement. Furthermore, the constraint scoring is extended to multiple heads and aggregates them by attention as Eq. (21):

$$\mathcal{H}_\delta^{\text{agg}}(s,a) = \sum_{i=1}^{M} \alpha_i(s,a) \cdot \mathcal{H}_{\delta_i}(s,a) \tag{21}$$

Where $\alpha_i(s,a)$ are attention scores satisfying $\sum_{i=1}^{M} \alpha_i(s,a) = 1$. The final hierarchical policy thus jointly minimizes the augmented objective as Eq. (22):

$$\begin{aligned}\Pi_\delta^\star(s) = \underset{a\in\mathcal{A}}{\operatorname{argmin}}\Big\{ & \mathcal{F}(s,a) + \lambda\, \mathcal{H}_\delta^{\text{agg}}(s,a) \\ & + \beta\, \mathcal{R}_{\text{margin}}(s,a)\Big\}\end{aligned} \tag{22}$$

Where $\beta > 0$ is a regularization coefficient.

Incorporation of such constraint cues is done via attention to the most salient elements affecting the situation at hand, thus ensuring priority of action on the most important aspects of the situation.

2.4. Policy-Embedded Gradient Field Navigation (PEGFN)

Based on the findings of ADSCO, the new algorithm for solving the problem of DAE constraint in discrete topology control is PEGFN which builds upon the procedure of action selection by taking into consideration the previous policy while constructing the gradient field from the constrained discrete action space. This ensures safe exploration without any infeasible topology as shown in Fig. 3.

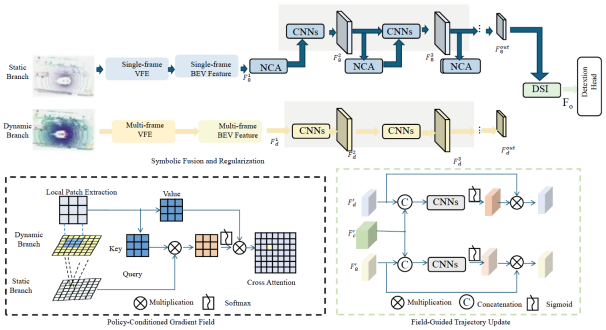

Fig. 3. Illustration of PEGFN for topology control

Figure 3 shows the mechanism of PEGFN through the use of the gradient field approach that uses policy conditions, feasibility masking, projection, and robustness regularization under the constraints of DAE. The controller design ensures that actions taken by representing the system state including generated power, bus voltage, and network topology in latent space using the architecture of two layers of ReLU MLP. Actions are represented in latent space that is evaluated for feasibility and relaxed to feasible action space under the constraints of DAE. The entire process of controller design consists of two stages; ADSCO produces strong candidate actions in action energy space, while PEGFN projects onto feasible action space, as shown in Fig. 4.

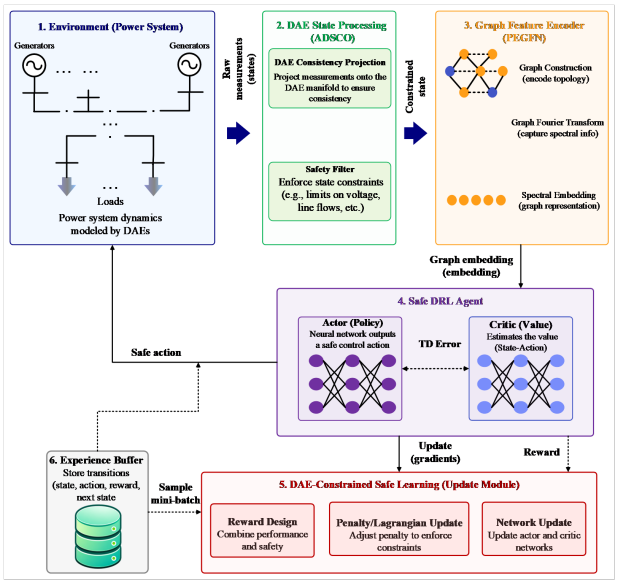

Fig. 4. Workflow of the ADSCO + PEGFN Framework

2.4.1. *Policy-Conditioned Gradient Field*

Let $s_t \in \mathcal{S}$ denote the observed grid state at time t, and let $\mathcal{A}$ be the discrete set of topology actions. ADSCO provides an energy score $\mathcal{F}(s_t,a)$ for each $a \in \mathcal{A}$. The nominal action distribution induced by the energy model, as Eq. (23):

$$q^{(0)}(a \mid s_t) = \frac{\exp\left(-\mathcal{F}(s_t,a)\right)}{\sum_{a'\in\mathcal{A}} \exp\left(-\mathcal{F}(s_t,a')\right)} \tag{23}$$

To enable differentiable refinement, the distribution as a simplex vector is represented $\mathrm{q}^{(0)}(s_t) \in \Delta^{|\mathcal{A}|}$. A policy-conditioned potential over q is defined as Eq. (24):

$$\mathcal{E}(s_t,\mathrm{q}) = \sum_{a\in\mathcal{A}} q(a)\mathcal{F}(s_t,a) \tag{24}$$

and the corresponding gradient field on the simplex as Eq. (25):

$$\mathcal{G}(s_t,\mathrm{q}) := -\nabla_{\mathrm{q}}\mathcal{E}(s_t,\mathrm{q}) = -\left[\mathcal{F}(s_t,a)\right]_{a\in\mathcal{A}}. \tag{25}$$

This field points toward distributions that reduce expected energy, thus favoring topology actions with lower operational cost and lower predicted constraint penalty.

2.4.2. State-Dependent Gating

Not all states require equal correction. The scalar risk indicator $\kappa(s_t)$ computed from the observation (e.g., maximum line loading ratio or the number of active violations returned by the simulator). PEGFN uses a smooth gating function $\eta : \mathcal{S} \to [0,1]$ as Eq. (26):

$$\eta(s_t) = \frac{1}{1 + \exp(-\rho \cdot (\kappa(s_t) - \kappa_0))} \tag{26}$$

Where κ_0 is a fixed risk threshold, and $\rho > 0$ controls sharpness. When the grid operates near security limits, $\eta(s_t)$ approaches 1 , and PEGFN applies stronger refinement. An amplification factor is defined $\alpha(s_t) \in \mathbb{R}_+$as Eq. (27):

$$\alpha(s_t) = \alpha_0 + \alpha_1 \cdot \eta(s_t) \tag{27}$$

Where $\alpha_0 > 0$ and $\alpha_1 \geq 0$ are fixed hyperparameters.

2.4.3. Field-Guided Refinement with Feasibility Projection

Refinement of PEGFN takes place in one step for each decision iteration, with the starting point being $\mathrm{q}^{(0)}(s_t)$ as shown in Eq. (28). While prior simulations included refinements through five steps, it was realized that such benefits, if any, offered by such refinements were insignificant, while there was added computation cost involved, without improving any other aspect of the system:

$$\tilde{\mathrm{q}}(s_t) = \Pi_\Delta\left(\mathrm{q}^{(0)}(s_t) + \alpha(s_t)\,\eta(s_t)\,\mathcal{G}\left(s_t, \mathrm{q}^{(0)}(s_t)\right)\right) \tag{28}$$

Despite the use of the iterative improvement approach by both ADSCO and PEGFN, there is a certain difference between how the two algorithms operate. More precisely, while ADSCO (Equation 15) uses iterative improvement of action energy scores in the latent space of policies for generating action preferences and building a proper energy landscape for decision making, PEGFN (Equation 28) improves the probability distribution of discrete topology actions through the use of policy-dependent gradient fields and action masks from operations.

Where Π_Δ denotes projection onto the probability simplex $\Delta^{|\mathcal{A}|}$. To enforce hard feasibility for topology actions, an action feasibility mask is applied to $\mathrm{m}(s_t) \in \{0,1\}^{|\mathcal{A}|}$ derived from operational rules and simulator-provided constraints (e.g., forbidden switch operations and known infeasible actions). The projected feasible distribution is as Eq. (29):

$$q\,\mathrm{proj}(a \mid s_t) = \frac{\tilde{q}(a \mid s_t)\; m(a \mid s_t)}{\sum_{a' \in \mathcal{A}} \tilde{q}(a' \mid s_t)\; m(a' \mid s_t) + \epsilon} \tag{29}$$

Where $\epsilon > 0$ is a small constant for numerical stability. The final topology action is selected as Eq. (30):

$$a_t = \underset{a \in \mathcal{A}}{\mathrm{argmax}} q\,\mathrm{proj}(a \mid s_t) \tag{30}$$

The environmental transition $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ is then produced by DAE-consistent power system simulation, as defined in Section 3.2. PEGFN does not apply Euler integration, since the simulator already implements physically consistent transitions under the DAE constraints.

This stage can be seen as the adjustment period of the suggested policy, where the outcomes will be polished based on the energy expectations gradient usage, and then mapped to the space of actions.

2.4.4. Robustness Regularization

To stabilize decision making and encourage robustness under uncertainty, the refined distribution is regularized by an entropy term as Eq. (31):

$$\mathcal{H}(s_t) = -\sum_{a \in \mathcal{A}} q^{\mathrm{proj}}(a \mid s_t) \log q_{\mathrm{proj}}(a \mid s_t) \tag{31}$$

Additionally, the feasibility risk of using simulator violation signals is penalized. Let $v(s_t, a_t) \geq 0$ be the normalized violation magnitude returned after applying a_t (0 if feasible).

A feasibility penalty is defined as Eq. (32):

$$\mathcal{R}(s_t, a_t) = v(s_t, a_t) \tag{32}$$

The PEGFN-refined decision is trained jointly with ADSCO under the following stepwise objective as Eq. (33):

$$\mathcal{L}_{\mathrm{PEGFN}} = \mathbb{E}\left[\mathcal{F}(s_t, a_t) + \lambda_1 \mathcal{R}(s_t, a_t) - \lambda_2 \mathcal{H}(s_t)\right] \tag{33}$$

Where $\lambda_1 > 0$ and $\lambda_2 > 0$ are fixed weights. This objective encourages low-energy and feasible topology decisions while maintaining sufficient exploration.

2.5. Component-wise Decomposition

Simplified description of architecture: This proposed architecture has been divided into five interconnected modules for solving the problem of optimal control for the task with constraints from DAE. Module C1 represents high-dimensional grid observations and topology actions together in a single latent space using the attention mechanism to get actions corresponding to the current state of the grid. Module C2 builds an energy landscape for control with relaxed action distributions which allows one to

optimize the policy without backpropagation through the DAE solver. C3 module includes topology constraints and feasibility information received from the simulation using DAE into the policy. C4 optimizes the actions received from the policy with the help of policy-guided gradient fields in the action space. C5 module provides global constraints and regularization to prevent infeasibility and instability.

3. Results and discussion

In this section, an assessment of ADSCO+PEGFN method for control of power grid topologies using L2RPN and RL2Grid benchmark sets will be done. The purpose of experiments is to evaluate performance in terms of control effectiveness, operational safety, computational efficiency, and generalizability to novel scenarios.

3.1. Task Definition

Problem statement can be viewed as the topology control of power transmission networks under DAE constraints. Input variables are: observable grid state s_t : loading factors, voltages, topology status, contingency flags. Output variables are: chosen discrete topology actions $a_t \in \mathcal{A}$: lines switch, busbar topology change, etc. Problem statement is formulated as reinforcement learning problem, where the policy is learned to choose actions with maximum long-term operation benefit in view of the safe grid. The signal for learning is obtained from the interaction with the simulator, but not explicitly provided by labels. Every time step, the simulator returns rewards and constraint violation signals (overload, voltages violation, etc.).

3.2. Dataset and Data Preprocessing

Datasets

Some examples of benchmarking platforms for controlling power grid via reinforcement learning are L2RPN and RL2Grid. In the simulation-based setting of L2RPN, the state transition is attained via DAE consistent power flow simulation without any use of traces.

L2RPN takes advantage of chronics that refer to large scale operation of power grid with time series data regarding load, generation, and contingencies. Each chronic is an instance of one RL episode that may last hundreds or even thousands of steps but will be ended earlier when security constraint is violated. On the other hand, the grid control problems in RL2Grid can be defined in a unified way with topology control having a fixed-horizon episode with discrete actions.

3.2.1. Data Preprocessing

Equal pre-processing of the data is applied to both L2RPN and RL2Grid datasets for the purpose of comparing them. As the data in both domains is obtained through simulators, there is no need for removing the duplicated instances. The samples that contain missing data from the simulator, have NaNs, have unsolved solver problems at the beginning of an episode or do not have full set of observations features are removed, whereas episodes that have been stopped because of an unsuccessful simulation due to extraordinary circumstances are considered as examples of poor feedback in reinforcement learning. Continuous features, such as line loading ratio, voltages and power flows are normalized via min-max normalization using training-set statistics, whereas topology features are left as they are. Temporal information and order of components are retained by the simulator without any predefined prediction horizon, windowing, stride sampling or negative sampling.

3.3. Evaluation Metrics and Baseline

3.3.1. Metrics Definition

The below-mentioned four metrics are applied to assess the performance of all the considered methods:

1. Average Episodic Reward to evaluate system performance;
2. Constraint Violation Frequency to determine safety violations (loads and voltage violations);
3. Average Episode Duration to determine stability and robustness till failure;
4. Total Load Curtailment during Episode as an economic and reliability metric.

Computation time is evaluated using model hyperparameters and floating-point operations per inference step.

3.3.2. Evaluation Protocol

The evaluation procedure is performed off-policy based on pre-defined splits of L2RPN and RL2Grid. At each step, actions are sampled from the full action space without any Top- K estimation at evaluation time. The results are averaged over complete episodes, wherein the seen environments are distributed according to the training set, whereas the unseen environments consist of excluded chronics. Agents are terminated either normally or fail.

3.3.3. Statistical Settings

To ensure statistical reliability, each method is trained and evaluated using $N = 5$ independent runs with different random seeds. Results are indicated as the mean standard deviation of the runs. The significance of the difference between the method and the most effective baseline was determined by a two-sided paired t-test, with differences being significant at $p < 0.05$.

Baselines

The proposed ADSCO+PEGFN framework is compared against six fixed baselines that cover traditional approaches, mainstream DRL models, recent state-of-the-art methods, and lightweight practical solutions. The baselines are: Rule-Based Topology Controller, DC-OPF Controller, DQN, PPO, GNN-PPO, and A2C. All baselines operate on the same observation and action spaces and are trained and evaluated under identical data splits and computational settings to ensure fair comparison.

3.4. Implementation Details

3.4.1. Training Details of the Proposed Method

Experiments are conducted on a system configured with Ubuntu 20.04 LTS, which has two AMD EPYC 7543 processors, 256 GB of memory, and a single NVIDIA A100 40GB graphics processing unit. The model is built based on Python 3.9, PyTorch 2.0, CUDA 11.7, and cuDNN 8.6. On the other hand, L2RPN and RL2Grid provide consistent simulations of the power grid for DAE while NumPy 1.23 and SciPy 1.10 are used for numerical computations. No use of any pre-trained models or third-party control solver algorithms is involved.

3.4.2. Model Configuration

For the model of ADSCO+PEGFN, the hyperparameters of structure are set to be fixed across all the experiments. The state encoder $\phi(\cdot)$ maps the raw grid states into the latent 128-dimensional space using a two-layer ReLU MLP. The discrete topology action is mapped into the 128 - dimensional embedding space $\theta(\cdot)$. Single head scaled dot product attention mechanism is applied to compute the compatibility score between the state and action with a constant temperature. The energy function $\mathcal{F}(s, a)$ is a lightweight feed-forward network which takes state embedding, action embedding, and interaction vector from the attention module as the inputs and outputs energy scores. For PEGFN, the action distribution is optimized by a single gradient update on the action probability simplex.

3.4.3. Fairness Statement

All techniques use the same interface for environments, observation representations, and actions as defined by the benchmarks. Preparation and assessment are done with the same training, validation, and test data splits and under the same hardware and software conditions. All techniques' hyperparameters are optimized only on the validation set, and the reported metrics are calculated from evaluation on the separate test set without further modifications.

3.5. Results on L2RPN Benchmark

The L2RPN is the standard test for the performance of the long-horizon control of power grid topology under real-world constraints and DAE simulations. It is clear from the results shown in Table 1that ADSCO+PEGFN outperforms all other baseline algorithms on the basis of all measures that are comparable to theirs in the same experimental setting. ADSCO+PEGFN has a mean episodic return of 336.8 which is 20.9 higher than that of GNNPPO and it also lowers down the constraint violation ratio from 5.4% to 3.9%. Moreover, it improves the stability with an increased survival time of 637 timesteps and decreases the load shedding by 18.4% compared to GNN-PPO. over $N = 5$ independent runs.

The results of RL2Grid are reported in Table 2, where the attention is paid to generalization and robustness of the algorithms in new operational settings. The suggested algorithm ADSCO+PEGFN demonstrates superiority to all other approaches in relation to all metrics. In particular, it provides the largest average episodic return, which equals 311.4 and is 21.8 better compared to the GNN-PPO approach. Moreover, the algorithm demonstrates better performance in terms of safety because it reduces the violation rate to 4.6% compared to 6.8% of the GNN-PPO and even 7.5% for PPO and A2C. It happens because of the PEGFN action refinement through the constraints and energy-based representation of ADSCO, which includes the hints about the feasibility of DAE within the policy learning. ADSCO + PEGFN also performs better in terms of stability providing the average survival time equal to 602 and the minimal load shedding value per episode equal to 97.8 that is 19.2% less than GNN-PPO.

deviation over $N = 5$ independent runs.

3.6. Main Ablation Analysis

The absence of any element from the ADSCO+PEGFN system results in a bad result on all the metrics, as can be seen from Table 4. In case when the attention-based state-action interaction is ignored, the expected return drops to 301.7 compared to 324.1 before, whereas constraint violations increase from 4.2% to 6.1%. Thus, we have shown the importance of accounting for action compatibility in different environments. The absence of PEGFN affects the stability the most, reducing the survival length by 88 actions and increasing constraint violations and load shedding. Thus, the use of gradient refinement, accounting for the constraints is vital for safe policies. The lack of feasibility penalties and entropy leads to even more unstable behavior, with constraint violations being equal to 7.5% and load shedding to 134.2. over $N = 5$ runs.

Table 1. Experimental results on the L2RPN benchmark. Results are reported as mean ± standard deviation

Method	Avg. Return	Violation Rate (%)	Survival Steps	Load Shedding
Rule-Based Controller	214.6 ± 18.9	12.7 ± 1.6	382 ± 41	186.4 ± 22.1
DC-OPF Controller	241.3 ± 21.4	9.8 ± 1.2	426 ± 37	163.7 ± 19.5
DQN	276.8 ± 24.6	7.4 ± 1.0	498 ± 45	139.2 ± 17.8
PPO	298.5 ± 19.7	6.2 ± 0.8	542 ± 39	121.6 ± 15.4
GNN-PPO	315.9 ± 17.3	5.4 ± 0.7	578 ± 33	112.3 ± 14.1
A2C	287.1 ± 22.5	6.8 ± 0.9	521 ± 41	128.9 ± 16.7
ADSCO+PEGFN (Ours)	336.8 ± 14.9	3.9 ± 0.5	637 ± 28	91.7 ± 11.6

Table 2. Experimental results on the RL2Grid benchmark. Results are reported as the mean ± standard

Method	Avg. Return	Violation Rate (%)	Survival Steps	Load Shedding
Rule-Based Controller	198.3 ± 20.6	14.9 ± 1.8	361 ± 44	201.7 ± 24.9
DC-OPF Controller	226.5 ± 22.1	11.6 ± 1.4	402 ± 39	176.8 ± 21.3
DQN	254.7 ± 26.4	8.9 ± 1.2	471 ± 48	148.6 ± 19.7
PPO	271.9 ± 21.8	7.6 ± 1.0	509 ± 41	133.4 ± 17.2
GNN-PPO	289.6 ± 18.9	6.8 ± 0.8	544 ± 36	121.1 ± 15.6
A2C	263.2 ± 23.7	8.1 ± 1.1	486 ± 43	141.9 ± 18.4
ADSCO+PEGFN (Ours)	311.4 ± 16.2	4.6 ± 0.6	602 ± 31	97.8 ± 12.8

Table 3. Computational efficiency comparison on L2RPN and RL2Grid benchmarks

Method	Params	FLOPs (per step)
DQN	1.6 M	0.42 G
PPO	2.3 M	0.61 G
GNN-PPO	6.8M	1.94 G
A2C	2.1 M	0.58G
ADSCO+PEGFN (Ours)	3.4M	0.89G

Table 4. Main ablation results on L2RPN and RL2Grid. Results are reported as the mean ± standard deviation

Variant	Avg. Return		Violation Rate (%)	Survival Steps	Load Shedding
Full ADSCO+PEGFN	324.1 15.6	±	4.2 ± 0.5	619 ± 29	94.8 ± 12.2
w/o Attention-Based Interaction	301.7 18.9	±	6.1 ± 0.7	563 ± 35	118.6 ± 15.4
w/o Policy-Embedded Refinement	287.9 20.3	±	6.8 ± 0.8	531 ± 38	126.9 ± 16.8
w/o Feasibility Penalty and Entropy Regularization	292.4 19.6	±	7.5 ± 0.9	548 ± 36	134.2 ± 17.9

Table 5. Sensitivity and robustness analysis on L2RPN and RL2Grid

Setting	Avg. Return	Violation Rate (%)	Survival Steps	Load Shedding	Notes
Default Configuration	324.1 ± 4.2	15.6 ± 0.5	619 ± 29	94.8 ± 12.2	Baseline
Reduced Step Size	317.6 ± 4.6	16.4 ± 0.6	603 ± 31	101.9 ± 13.5	Slight degradation
Increased Entropy Weight	312.8 ± 4.9	17.1 ± 0.6	591 ± 33	107.3 ± 14.1	More exploration
State Noise Injection	309.4 ± 5.3	18.7 ± 0.7	584 ± 35	112.6 ± 15.8	Strongest disturbance

3.7. Sensitivity and Robustness Analysis

As indicated in Table 5, the robustness of ADSCO+PEGFN with respect to hyperparameter setting and perturbation is also confirmed. The more frequent the refinement step, the less the return and survival time will be; yet, at the same time, the safety will be guaranteed. The higher the entropy regularization term, the more conservative the strategy will be; the rate of violations and load shedding will increase slightly.

Table 6. Nomenclature

Abbreviations		**Symbols**	
ACIL	Adaptive Continuous-time Interval Learning (used for continuous-time control)	$f(\cdot)$	The differential component of the system
ADSCO	Attentive Differentiable Structure for Control Optimization	$g(\cdot)$	Algebraic constraints of the system
AI	Artificial Intelligence	s	The observable grid state
AUC	Area Under the Curve	W	A trainable matrix
CNN	Convolutional Neural Network	$\mathcal{X}$	The continuous dynamic state
DAE	Differential-Algebraic Equation	y	Algebraic variables associated with the system
DRL	Deep Reinforcement Learning	γ	A discount factor
L2RPN	Learning to Run a Power Network	ϕ	The grid state
OC	Optimal Control	θ	The topology action
PEGFN	Policy-Embedded Gradient Field Navigation	τ	A temperature parameter
RL2Grid	Reinforcement Learning to Grid	λ	The strength of smoothness regularization
symbol	Discrete topology action	β	A regularization coefficient

4. Competing of interests

The author declares no competing of interests.

5. Authorship contribution statement

Guoxing Si: Writing-Original draft preparation, Conceptualization, Project administration.

6. Data availability

On Request

7. Declarations

Not applicable

8. Conflicts of interest

The author declares that there is no conflict of interest regarding the publication of this paper.

9. Author statement

The manuscript has been read and approved by the author, the requirements for authorship, as stated earlier in this document, have been met, and the author believes that the manuscript represents honest work.

10. Funding

Not applicable

11. Ethical approval

The author has been personally and actively involved in substantial work leading to the paper, and will take public responsibility for its content.

References

[1] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang. "Informer: Beyond efficient transformer for long sequence time-series forecasting". In: *Proceedings of the AAAI conference on artificial intelligence*. ***35***. 12. 2021, 11106–11115. DOI: https://doi.org/10.1609/aaai.v35i12.17325.

[2] A. Angelopoulos, E. Candes, and R. J. Tibshirani, (2023) *"Conformal pid control for time series prediction"* **Advances in neural information processing systems *36***: 23047–23074. DOI: https://ojs.aaai.org/index.php/AAAI/article/view/17325.

[3] L. Shen and J. Kwok. "Non-autoregressive conditional diffusion models for time series prediction". In: *International Conference on Machine Learning*. PMLR. 2023, 31016–31029. DOI: https://proceedings.mlr.press/v202/shen23d.html.

[4] M. S. Sial, Q. Fu, and T. Vianna Brugni, (2022) *"Allocation of Interline Power Flow Controller-Based Congestion Management in Deregulated Power System"* **Advances in Engineering and Intelligence Systems *1***(01): 65–74. DOI: https://doi.org/10.22034/aeis.2022.148478.

[5] X. Wen and W. Li, (2023) *"Time series prediction based on LSTM-attention-LSTM model"* **IEEE access *11***: 48322–48331. DOI: https://doi.org/10.1109/ACCESS.2023.3276628.

[6] L. Ren, Z. Jia, Y. Laili, and D. Huang, (2023) *"Deep learning for time-series prediction in IIoT: progress, challenges, and prospects"* **IEEE transactions on neural networks and learning systems *35***(11): 15072–15091. DOI: https://doi.org/10.1109/TNNLS.2023.3291371.

[7] Y. Li, K. Wu, and J. Liu, (2023) *"Self-paced ARIMA for robust time series prediction"* **Knowledge-Based Systems *269***: 110489. DOI: https://www.sciencedirect.com/science/article/pii/S0950705123002393.

[8] Y. Wang, (2024) *"Enhancing transient stability of DFIG-based wind turbine systems using deep learning-controlled resistance-based fault current limiters"* **Advances in Engineering and Intelligence Systems *3***(01): 10–22. DOI: https://doi.org/10.22034/aeis.2024.423904.1139.

[9] D. M. Durairaj and B. K. Mohan, (2022) *"A convolutional neural network based approach to financial time series prediction"* **Neural Computing and Applications *34***(16): 13319–13337. DOI: https://doi.org/10.1007/s00521-022-07143-2.

[10] L. Yin, L. Wang, T. Li, S. Lu, J. Tian, Z. Yin, X. Li, and W. Zheng, (2023) *"U-Net-LSTM: time series-enhanced lake boundary prediction model"* **Land *12***(10): 1859. DOI: https://doi.org/10.3390/land12101859.

[11] C. Yu, F. Wang, Z. Shao, T. Sun, L. Wu, and Y. Xu. "Dsformer: A double sampling transformer for multivariate time series long-term prediction". In: *Proceedings of the 32nd ACM international conference on information and knowledge management*. 2023, 3062–3072. DOI: https://doi.org/10.1145/3583780.361485.

[12] W. Zheng and G. Chen, (2021) *"An accurate GRU-based power time-series prediction approach with selective state updating and stochastic optimization"* **IEEE Transactions on Cybernetics *52***(12): 13902–13914. DOI: https://doi.org/10.1109/TCYB.2021.3121312.

[13] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, (2021) *"A survey on long short-term memory networks for time series prediction"* **Procedia Cirp *99***: 650–655. DOI: https://doi.org/10.1016/j.procir.2021.03.088.

[14] R. Chandra, S. Goyal, and R. Gupta, (2021) *"Evaluation of deep learning models for multi-step ahead time series prediction"* **Ieee Access *9***: 83105–83123. DOI: https://doi.org/10.1109/ACCESS.2021.3085085.

[15] J. Fan, K. Zhang, Y. Huang, Y. Zhu, and B. Chen, (2023) *"Parallel spatio-temporal attention-based TCN for multivariate time series prediction"* **Neural Computing and Applications *35***(18): 13109–13118. DOI: https://link.springer.com/article/10.1007/s00521-021-05958-z.

[16] M. Hou, C. Xu, Z. Li, Y. Liu, W. Liu, E. Chen, and J. Bian. "Multi-granularity residual learning with confidence estimation for time series prediction". In: *Proceedings of the ACM Web Conference 2022*. 2022, 112–121. DOI: https://doi.org/10.1145/3485447.351205.

[17] H. V. Dudukcu, M. Taskiran, Z. G. C. Taskiran, and T. Yildirim, (2023) *"Temporal Convolutional Networks with RNN approach for chaotic time series prediction"* **Applied soft computing *133***: 109945. DOI: https://doi.org/10.1016/j.asoc.2022.109945.

[18] I. Amalou, N. Mouhni, and A. Abdali, (2022) *"Multivariate time series prediction by RNN architectures for energy consumption forecasting"* **Energy Reports *8***: 1084–1091. DOI: https://doi.org/10.1016/j.egyr.2022.07.139.

[19] Y. Xiao, H. Yin, Y. Zhang, H. Qi, Y. Zhang, and Z. Liu, (2021) *"A dual-stage attention-based Conv-LSTM network for spatio-temporal correlation and multivariate time series prediction"* **International Journal of Intelligent Systems *36***(5): 2036–2057. DOI: https://doi.org/10.1002/int.22370.

[20] J. Wang, W. Jiang, Z. Li, and Y. Lu, (2021) *"A new multi-scale sliding window LSTM framework (MSSW-LSTM): a case study for GNSS time-series prediction"* **Remote Sensing *13***(16): 3328. DOI: https://doi.org/10.3390/rs13163328.

[21] M. Xu, M. Han, C. P. Chen, and T. Qiu, (2018) *"Recurrent broad learning systems for time series prediction"* **IEEE transactions on cybernetics *50***(4): 1405–1417. DOI: https://doi.org/10.1109/TCYB.2018.2863020.

[22] W. Zheng and J. Hu, (2022) *"Multivariate time series prediction based on temporal change information learning method"* **IEEE Transactions on Neural Networks and Learning Systems *34***(10): 7034–7048. DOI: https://doi.org/10.1109/TNNLS.2021.3137178.

[23] Y. Sun, S. Dang, W. Fang, W. Huang, and Y. Xu, (2025) *"Design and analysis of electricity anomaly detection and diagnosis models based on generative adversarial networks"* **International Journal of High Speed Electronics and Systems *34***(04): 2540330. DOI: https://doi.org/10.1142/S0129156425403304.

[24] W. R. Moskolaı, W. Abdou, A. Dipanda, and Kolyang, (2021) *"Application of deep learning architectures for satellite image time series prediction: A review"* **Remote Sensing *13***(23): 4822. DOI: https://doi.org/10.3390/rs13234822.

[25] Z. Karevan and J. A. Suykens, (2020) *"Transductive LSTM for time-series prediction: An application to weather forecasting"* **Neural Networks *125***: 1–9. DOI: https://doi.org/10.1016/j.neunet.2019.12.030.

[26] S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Zhang, and J. Zhou. "Timemixer: Decomposable multiscale mixing for time series forecasting". In: *International conference on learning representations*. ***2024***. 2024, 38626–38652. DOI: https://proceedings.iclr.cc/paper_files/paper/2024/hash/a7ac8a21e5a27e7ab31a5f42a0117bdb-Abstract-Conference.html.

[27] A. Wachi, X. Shen, and Y. Sui, (2024) *"A survey of constraint formulations in safe reinforcement learning"* **arXiv preprint arXiv:2402.02025**: DOI: https://doi.org/10.48550/arXiv.2402.02025.

[28] W. Zhao, T. He, R. Chen, T. Wei, and C. Liu, (2023) *"State-wise safe reinforcement learning: A survey"* **arXiv preprint arXiv:2302.03122**: DOI: https://doi.org/10.48550/arXiv.2302.03122.

[29] H. Wang, S. Bo, S. M. Fazeli, A. Yavari, B. Liu, Q. Zhao, and J. Liu, (2026) *"Constrained stabilization control via safe reinforcement learning and control Lyapunov value function"* **Control Engineering Practice *174***: 107065. DOI: https://doi.org/10.1016/j.conengprac.2026.107065.

[30] S. Bandyopadhyay and S. Bhasin, (2025) *"Lagrangian-based online safe reinforcement learning for state-constrained systems"* **Automatica *179***: 112458. DOI: https://doi.org/10.1016/j.automatica.2025.112458.

[31] M. Opmeer and O. Staffans, (2024) *"On linear quadratic optimal control and algebraic Riccati equations for infinite-dimensional differential-algebraic equations"* **DAE Panel 2**: DOI: https://doi.org/10.52825/dae-p.v2i.1388.

[32] A. Ilchmann, T. Reis, L. Biegler, S. Campbell, C. Führer, R. März, S. Trenn, P. Kunkel, R. Riaza, V. H. Linh, et al. *Differential-algebraic equations forum*. ***2017***. Springer, 2013. DOI: https://doi.org/10.1007/978-3-642-27555-5.

[33] H. Gernandt and T. Reis, (2024) *"Linear-quadratic optimal control for abstract differential-algebraic equations"* **IFAC-PapersOnLine *58***(17): 310–315. DOI: https://doi.org/10.1016/j.ifacol.2024.10.187.

[34] J. Wang, Z. Peng, X. Wang, C. Li, and J. Wu, (2020) *"Deep fuzzy cognitive maps for interpretable multivariate time series prediction"* **IEEE transactions on fuzzy systems *29***(9): 2647–2660. DOI: https://doi.org/10.1109/TFUZZ.2020.3005293.

[35] S. Karasu and A. Altan, (2022) *"Crude oil time series prediction model based on LSTM network with chaotic Henry gas solubility optimization"* **Energy *242***: 122964. DOI: https://doi.org/10.1016/j.energy.2021.122964.

[36] J. Wen, J. Yang, B. Jiang, H. Song, and H. Wang, (2020) *"Big data driven marine environment information forecasting: a time series prediction network"* **IEEE transactions on fuzzy systems *29***(1): 4–18. DOI: https://doi.org/10.1109/TFUZZ.2020.3012393.

[37] M. A. Morid, O. R. L. Sheng, and J. Dunbar, (2023) *"Time series prediction using deep learning methods in healthcare"* **ACM Transactions on Management Information Systems *14***(1): 1–29. DOI: https://doi.org/10.1145/3531326.

[38] H. Zhang, J. Zeng, J. Ma, Y. Fang, C. Ma, Z. Yao, and Z. Chen, (2021) *"Time series prediction of microseismic multi-parameter related to rockburst based on deep learning"* **Rock Mechanics and Rock Engineering *54***(12): 6299–6321. DOI: https://doi.org/10.1007/s00603-021-02614-9.

[39] H. Widiputra, A. Mailangkay, and E. Gautama, (2021) *"Multivariate CNN-LSTM model for multiple parallel financial time-series prediction"* **Complexity *2021***(1): 9903518. DOI: https://doi.org/10.1155/2021/9903518.

[40] M. Hammad Saeed, S. Iqbal, M. Sohail, and U. Akram, (2023) *"An optimal adaptive control of DC-Microgrid with the aim of accurate current sharing and voltage regulation according to the load variation and gray-wolf optimization"* **Advances in Engineering and Intelligence Systems 2**(03): 50–60. DOI: https://doi.org/10.22034/aeis.2023.406515.1110.

[41] S. Nurpatsha, (2025) *"A New Analysis of Web Customer Service Text Classification of Alexa Virtual Assistant Commands Using a Deep Learning Model"* **Journal of Artificial Intelligence and System Modelling *3***(02): 76–90. DOI: https://doi.org/10.22034/jaism.2025.515660.1120.