Accelerating granular dynamics simulations: A graph neural network surrogate for complex high-energy ball milling

Nuñez, Santiago Garrido; Schott, Dingena L.; Padding, Johan T.

DOI
10.1016/j.powtec.2025.121653

Publication date
2026

Document Version
Final published version

Published in
Powder Technology

Contents lists available at ScienceDirect

Powder Technology

journal homepage: www.journals.elsevier.com/powder-technology

Accelerating granular dynamics simulations: A graph neural network surrogate for complex high-energy ball milling

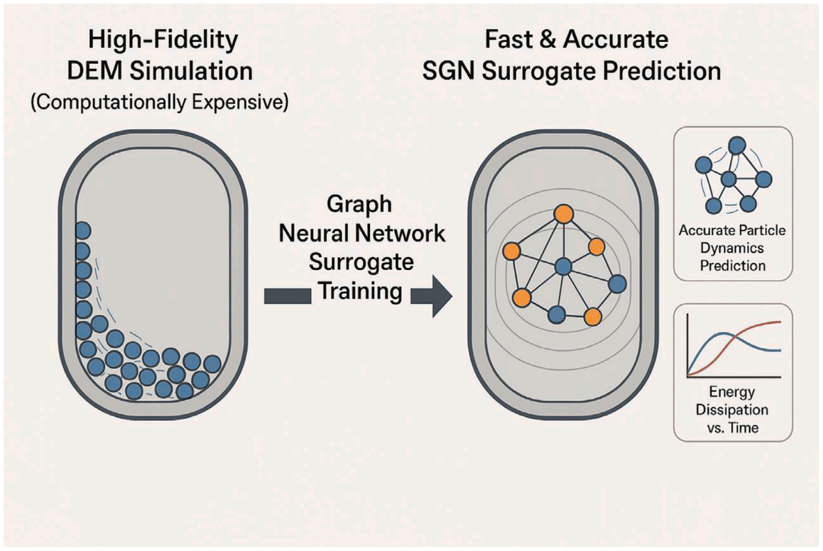
Santiago Garrido Nuñez [a,*], Dingena L. Schott [b], Johan T. Padding [a]

[a] Department of Process and Energy, Delft University of Technology, Leeghwaterstraat 39, Delft, 2628 CB, The Netherlands
[b] Department of Maritime and Transport Technology, Delft University of Technology, Mekelweg 2, Delft, 2628 CD, The Netherlands

HIGHLIGHTS

- The surrogate captures complex particle dynamics in a high-energy ball mill.
- Runs with a timestep 100× larger than DEM.
- Predicts cumulative energy dissipation, a relevant mechanochemical metric.
- Strong transferability to unseen motions and moderate jar geometry alterations.

GRAPHICAL ABSTRACT


High-Fidelity DEM Simulation
(Computationally Expensive)
Graph Neural Network Surrogate Training
Fast & Accurate SGN Surrogate Prediction
Accurate Particle Dynamics Prediction
Energy Dissipation vs. Time

ARTICLE INFO

Keywords:
Graph neural network
Discrete element method
Mechanochemistry
Surrogate model
Artificial intelligence

ABSTRACT

While the Discrete Element Method (DEM) provides high-fidelity insights into granular processes like high-energy ball milling, its computational cost can become prohibitive when exploring extensive parameter spaces required for scale-up. This limitation hinders the rapid design and optimization cycles crucial for emerging applications, like mechanochemistry. Surrogate modeling offers a promising path to overcome these computational barriers, yet existing approaches often struggle to accurately represent the complex, moving boundary conditions typical of milling equipment. In this work, we leverage a Signed Distance Function Graph Neural Network (SGN) surrogate tailored to the high-energy, moving-boundary regime of the Emax mill. Trained on DEM data, the SGN jointly predicts particle kinematics for recursive roll-out and a mechanochemistry-relevant global quantity, the global dissipated energy. The model exhibits strong generalization to unseen motion trajectories and moderate jar-shape edits without retraining, while operating with a timestep over 100x larger than required by DEM. In a CPU-only comparison, it achieves a minimum of 6.6× wall-clock speedup. This approach provides a powerful and promising technique for the simulation, analysis, and design optimization of high-energy ball milling equipment.

* Corresponding author.
E-mail address: s.garridonunez@tudelft.nl (S. Garrido Nuñez).

https://doi.org/10.1016/j.powtec.2025.121653
Received 25 May 2025; Received in revised form 4 September 2025; Accepted 13 September 2025
Available online 19 September 2025

1. Introduction

The rapid advancement of artificial intelligence can enable significant progress in scientific computing. In particular, surrogate models can approximate complex physical phenomena, such as particle interactions in granular systems or particle-based fluid representations, at much lower computational cost than traditional methods like the discrete element method (DEM) or smoothed particle hydrodynamics (SPH) [1–3].

Such surrogates are especially appealing in high-energy ball milling, where DEM is a well-established method for a wide range of applications, such as mechanochemistry [4–6], mechanical alloying [7], ultra-fine milling, and particle breakage [8,9]. Although each of these application fields faces its own set of distinct challenges for producing a valuable model, they share some critical commonalities, namely they require a combination of a large number of particles (discrete elements), a very small timestep to accurately numerically integrate the underlying equations of motion, or possibly a combination of both [10,11]. These conditions may be manageable when reproducing well-defined granular flows on lab-scale. However, as attention shifts towards industrial-scale applications, the computational cost needed to resolve numerous particle–particle and particle–wall contacts at small timesteps can become prohibitively expensive. This is especially problematic for emerging technologies such as mechanochemistry, where there are no clear connections to larger-scale machinery [12]. The intensive iterative design and implementation required slows the adoption of mechanochemistry, delaying its potential contributions to sustainability goals as defined by green chemistry principles [13]. Consequently, surrogate models are an appealing alternative in this context, offering a potential pathway to accelerate progress.

Among data-driven surrogates, graph neural networks (GNNs) have rapidly become a compelling paradigm for learned particle simulators, representing particles as nodes and interactions as edges, and rolling out dynamics via message passing [14]. The seminal Graph Network-based Simulator (GNS) demonstrated accurate, long-horizon roll-outs over fluids, rigid bodies, and simple granular settings, with strong generalization in particle count and initial conditions. [1]. Building on this, Choi & Kumar developed GNN surrogates specifically for granular flows (e.g., column collapse), reporting hundreds-fold speedups relative to high-fidelity solvers while preserving key flow features and scaling to larger domains than seen in training [15]. There is also emerging work coupling GNS with inverse design/optimization to tune DEM parameters or device settings efficiently [16].

A persistent challenge in learned granular simulators is boundary handling in complex geometry while preserving accurate physics. Early approaches either encoded distances to simple box-like boundaries or introduced virtual/ghost entities in so-called boundary GNNs to approximate walls. While these approaches have improved model generality, the introduction of virtual entities brings additional complexity and potential inaccuracies in wall-interaction physics [17–24]. To address these limitations, Li and Sakai proposed a signed distance function-based GNN (SGN) that encodes arbitrarily shaped boundaries as continuous distance fields, allowing the network to handle complex geometries without virtual particles [2]. In parallel, physics-informed GNN architectures have been introduced to embed hard constraints from mechanics. For instance, Sharma and Fink enforce conservation of linear and angular momentum at individual collisions via a special message-passing scheme, yielding stable long-term predictions for 3D granular systems with inelastic impacts [25].

Despite these advances, no prior work has applied GNN-based surrogates to the extreme dynamic regime of high-energy ball milling. In such systems, particles collide violently under rapidly evolving boundary conditions (e.g. tumbling jars and moving reactor walls). Existing surrogate models have yet to demonstrate they can capture this highly dynamic, dissipative environment. In fact, prior data-driven studies have focused on macro-scale performance metrics of low-speed mills, such as predicting particle size distribution, bed height, or mixing quality, rather than simulating the detailed collision dynamics [17,24,26]. Related SDF-GNN work (e.g., Li & Sakai, 2024 [2]) has also targeted static or low-speed boundaries. Thus, this gap motivates the present work to develop a surrogate approach that can faithfully emulate the physics of a high-energy milling process.

To achieve this, we develop a surrogate model capable of handling complex dynamic boundaries, involving oscillatory and translational motion, characteristic of high-energy ball milling equipment used in powder processing. Specifically, we adapt a SGN to accurately capture complex energy input mechanisms and intense, high-energy dynamics and collisions, beyond the low-velocity regimes of prior studies. We extend the model by introducing moving boundaries and a secondary output for cumulative energy dissipation alongside local particle kinematics, enabling a dual prediction that is critical for mechanochemistry. Furthermore, our approach shows promising generalization to unseen motions and modified geometries, facilitating systematic study and design. Although our primary focus is on mechanochemical applications, this methodology can be adapted to virtually any high-energy ball milling scenario. Because the boundary kinematics and regimes differ compared to previous studies, we do not pursue a numerical head-to-head comparison; our contribution is complementary, extending SDF-based surrogates to time-varying, fast-moving wall regimes.

All associated code is freely available, including scripts for data extraction, transformation, and loading, as well as those used for model construction, training, and generative simulations, at the following location: `github.com/sgarridonunez/SGN_ball_milling`. In the subsequent sections, we detail the specific SGN architecture and training procedure using DEM simulation data, present validation results comparing surrogate predictions against ground truth values for particle dynamics and energy dissipation, and discuss the model's performance and generalization capabilities.

2. Methodology

2.1. Discrete Element Method (DEM) and simulation setup

The Discrete Element Method (DEM) is used to generate the data needed for training the surrogate model and serves as ground truth. In this study, Altair® EDEM™ 2021.2 was used as the DEM solver, and Python 3.9.12 was used for data post-processing. EDEM™ follows a soft sphere approach by calculating the contact forces for each particle interaction using Hertz and Mindlin's contact model, which can capture the non-linear interactions that arise when particle–particle or particle–wall collisions occur [27,28].

Newton's equations of motion are solved numerically to predict the evolution of the (angular) velocity of each particle:

$$m_i \frac{d\mathbf{V}_i}{dt} = \mathbf{F}_{c,i} + m_i \mathbf{g} \tag{1}$$

$$I_i \frac{d\boldsymbol{\omega}_i}{dt} = \boldsymbol{\tau}_i \tag{2}$$

where m_i, I_i, $\mathbf{V}_i$ and $\boldsymbol{\omega}_i$ are the mass, moment of inertia, velocity, and angular velocity, respectively, of particle i. $\mathbf{F}_{c,i}$ and $\boldsymbol{\tau}_i$ represent the total contact force and total contact torque (relative to the particle's center of mass), respectively. The total force and torque are determined by summing over all neighbors in contact with particle i.

Each discrete element has its own radius R, mass m, Young's modulus Y, shear modulus G, coefficient of restitution e, and Poisson ratio ν. The contact force $\mathbf{F}_{c,ij}$ on a particle i due to its interaction with another particle j (or wall) is the vector sum of a normal force $\mathbf{F}_{n,ij}$ and tangential force $\mathbf{F}_{t,ij}$:

$$\mathbf{F}_{c,ij} = \mathbf{F}_{n,ij} + \mathbf{F}_{t,ij} = (K_n \delta_{n,ij} - \gamma_n \mathbf{V}_{n,ij}) + (K_t \boldsymbol{\delta}_{t,ij} - \gamma_t \mathbf{V}_{t,ij}) \tag{3}$$

with:

$$K_n = \frac{4}{3} Y^* \sqrt{R^* \delta_n} \tag{4}$$

$$\gamma_n = -2\sqrt{\frac{5}{6}}\beta\sqrt{S_n m^*} \geq 0 \tag{5}$$
$$K_t = 8G^*\sqrt{R^*\delta_n} \tag{6}$$
$$\gamma_t = -2\sqrt{\frac{5}{6}}\beta\sqrt{S_t m^*} \geq 0 \tag{7}$$
$$S_n = 2Y^*\sqrt{R^*\delta_n} \tag{8}$$
$$S_t = 8G^*\sqrt{R^*\delta_n} \tag{9}$$
$$\beta = \frac{\ln(e)}{\sqrt{\ln^2(e)+\pi^2}} \tag{10}$$
$$\frac{1}{Y^*} = \frac{(1-\nu_1^2)}{Y_1} + \frac{(1-\nu_2^2)}{Y_2} \tag{11}$$
$$\frac{1}{G^*} = \frac{2(2-\nu_1)(1+\nu_1)}{Y_1} + \frac{2(2-\nu_2)(1+\nu_2)}{Y_2} \tag{12}$$
$$\frac{1}{R^*} = \frac{1}{R_1} + \frac{1}{R_2} \tag{13}$$
$$\frac{1}{m^*} = \frac{1}{m_1} + \frac{1}{m_2} \tag{14}$$

Here, $\mathbf{V}_{n,ij}$ and $\mathbf{V}_{t,ij}$ denote the relative velocities in the normal and tangential directions between particles i and j at the point of contact. The vectors $\boldsymbol{\delta}_{n,ij}$ and $\boldsymbol{\delta}_{t,ij}$ represent the normal and tangential overlaps between the particles, with the tangential overlap obtained by integrating the relative tangential velocity over time and projecting it onto the current tangential direction. The constants K_n and K_t are the elastic coefficients for normal and tangential contacts, respectively, while γ_n and γ_t correspond to the viscoelastic damping coefficients for these contacts.

On the right-hand side of Eq. (3), the expression within the first set of parentheses represents the normal force, and the expression in the second set corresponds to the tangential force. Specifically, the normal force comprises two components: a spring force and a normal damping force $\mathbf{F}_{n,d}$, while the tangential force is made up of a shear force and a tangential damping force $\mathbf{F}_{t,d}$. The magnitude of the tangential force F_t is limited according to the Coulomb friction law: if $F_t \geq \mu_f F_n$, where μ_f is the friction coefficient and F_n the magnitude of the normal force, then F_t is set equal to F_n (while still oriented in the tangential direction).

Additionally, the contact torque, $\boldsymbol{\tau}_{ij}$, acting on particle i as a result of its interaction with particle (or wall element) j is determined by the cross product of the vector $\mathbf{R}_{ij}$—which extends from the center of mass of particle i to the contact point with particle j—and the tangential contact force $\mathbf{F}_{t,ij}$. Given that the particles experience continuous rolling motion, particularly in interactions with a wall, it is essential to account for any slight deviations from perfect sphericity. This is achieved by introducing a rolling torque, $\boldsymbol{\tau}_{r,ij}$, which is computed using the coefficient of rolling friction μ_r, the magnitude of the normal contact force $F_{n,ij}$, the distance R_{ij} from the center of mass to the contact point, and the orientation of the particle's relative angular velocity, $\boldsymbol{\omega}_{rel}$. These relationships are described by:

$$\boldsymbol{\tau}_{ij} = \mathbf{R}_{ij} \times \mathbf{F}_{t,ij} + \boldsymbol{\tau}_{r,ij} \tag{15}$$
$$\boldsymbol{\tau}_{r,ij} = -\mu_r F_{n,ij} R_{ij} \frac{\boldsymbol{\omega}_{rel}}{\omega_{rel}} \tag{16}$$

Finally, the energy dissipated over the time interval from t_1 to t_2, due to the damping effects characterized by γ_n and γ_t, is calculated as follows:

$$E_n = \int_{t_1}^{t_2} \mathbf{F}_{n,d} \cdot \mathbf{V}_{n,ij}\, \mathrm{d}t \tag{17}$$
$$E_t = \int_{t_1}^{t_2} \mathbf{F}_{t,d} \cdot \mathbf{V}_{t,ij}\, \mathrm{d}t \tag{18}$$

The DEM model has been calibrated and validated for the mechanochemical regeneration of sodium borohydride $NaBH_4$ in the *Emax* high-energy ball mill produced by the German company Retsch. We refer to our previous work for more details [4,29]. The crucial properties relevant to this work can be found in Table 1.

Table 1
Properties used for the milling balls and jar obtained from Retsch, corresponding to steel X46Cr13 while accounting for the presence of $NaBO_2 \cdot 4\,H_2O$ ($\geq$99% - Sigma-Aldrich) and MgH_2 ($\geq$99.9% - Nanoshel).

Parameter	Value
Particle diameter	0.01 m
Restitution coefficient	0.3
Friction coefficient	0.3
Rolling friction coefficient	0.045
Density	7700 [kg/m^3] [30]
Young's modulus	2.05 [GPa] [30]
Poisson's ratio	0.235 [30]
Simulation time step	9.5×10^{-7} [s]
Total simulation time	15 [s]
Time integration method	Euler

The machine can allocate proprietary grinding jars with 125 ml of volume that follow a circular motion with a rotational speed n up to 2000 revolutions per minute with an amplitude (radius) A of 1.7 cm, see Fig. 1. The movement of the jar has been replicated in our simulations. An STL file was built and imported into EDEM™ to represent the geometry of the milling jar accurately. The STL was validated as watertight to ensure an unambiguous SDF sign (inside/outside); closed internal cavities would be handled correctly under the same assumption.

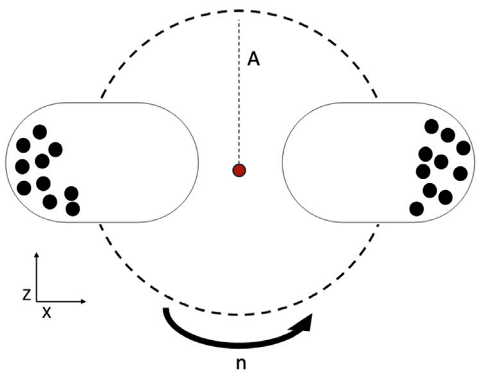
The system is initially set up by generating all the discrete media over a five-second interval, which allows them to settle into their resting positions within the jar before any motion begins. After this initialization phase, the simulation runs for an additional 10 s with a rotational speed of 300 rpm to capture the dynamic behavior of the system. A fill ratio of 10% is used, corresponding to a total of 24 milling balls. To reduce computational complexity, the model is simplified by including only milling balls as discrete elements. This simplification is justified because the influence of the processed material can be effectively represented by calibrating the friction and restitution coefficients [5,31,32].

2.2. Graph neural network and surrogate model

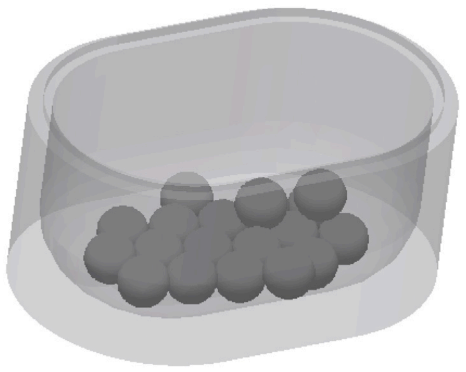
A graph is a representation composed of a set of nodes and a set of edges that connect pairs of these nodes [33]. This structure models relationships or interactions between objects in various domains, such as computer networks, social networks, biological systems, and, in the context of this paper, granular systems (see Fig. 2). Moreover, graphs provide a natural framework for message passing, where nodes exchange information with their neighbors along the edges [34]. This capability is fundamental in graph neural network architectures, enabling iterative aggregation of local information to capture complex, global patterns within the graph.

2.2.1. Architecture

In this study, node and edge features encode the dynamic state and geometric context of the system. Specifically, the dynamic state is encoded through particle velocity, and the geometric context is dictated by a Signed Distance Function (SDF), leading to the definition of an SDF-based graph neural network (SGN) as proposed by Li et al. [2]. As such, we follow the same terminology for consistency. The SDF is a fixed field computed from the watertight, triangulated jar STL (see Fig. 3). It provides a per-particle distance channel for node features and a proximity vector on particle–wall edges. The mill's boundary motion is applied to the wall: for each time step the STL is rigidly translated in space by the time-dependent center of mass (CoM), and the SDF is evaluated relative to the moved wall. Particle states are not directly forced by this motion; they are updated by integrating predicted smooth accelerations. Conditioning on the moving boundary allows inference under unseen motion trajectories, provided the training data spans comparable kinematic/energy regimes.

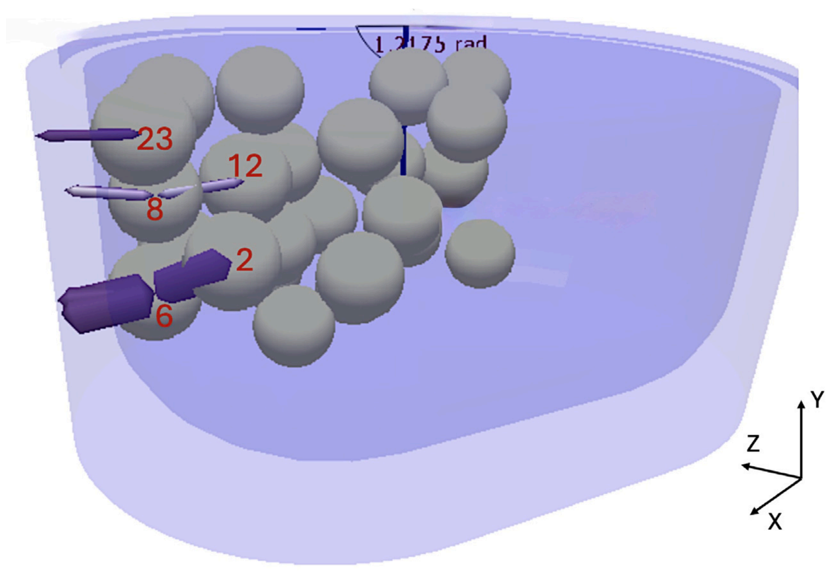


(a)

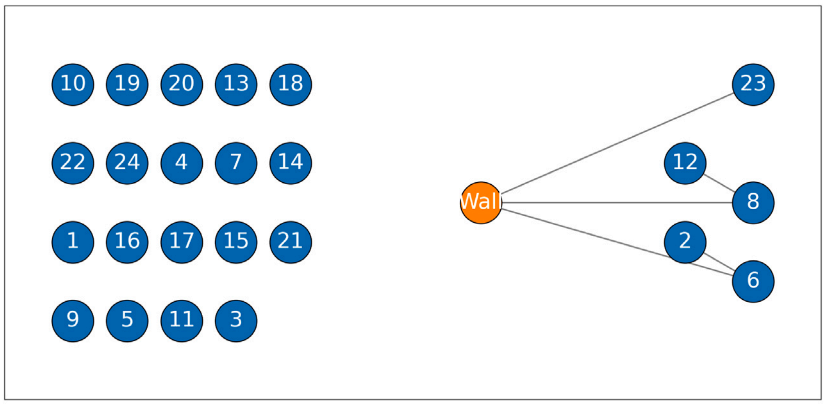


(b)

Fig. 1. (a) Schematic of jar movement (A = 1.7 cm, n = 300 rpm) (b) 3D model of the milling jar [4].



(a)



(b)

Fig. 2. (a) DEM connectivity example, (b) Graph representation of DEM timestep. Note that each particle has an individual ID used to define connectivity at any given timestep.

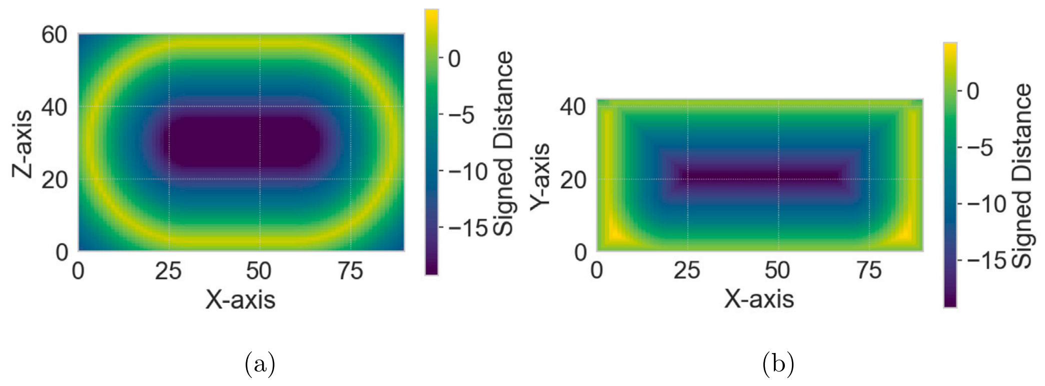


Fig. 3. Geometry's SDF field slices (a) XZ plane, (b) XY plane. In our system, negative values indicate positions inside the jar. Thus, the most negative values represent positions furthest away from the wall.

For closed (watertight) surfaces, the inside/outside sign is unambiguous and internal cavities are supported; open holes or non-manifold seams can introduce sign ambiguity and should be repaired before SDF generation. For new jar shapes, we simply recompute the SDF field for the new STL; no retraining is required for inference, whereas new training datasets require regenerating the SDF-derived features.

The granular system at a given time is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The set of nodes $\mathcal{V}$ includes nodes v_p representing each individual particle ($p = 1, \dots, N_p$) and a single, dedicated node v_w representing properties of the wall boundary. Note that the node v_w does not dictate any spatial context information for the particles; it is simply established to define variables relevant to the geometry, such as its center of mass, and to be able to keep track of collisions between particles and the wall, which are crucial in ball milling.

The set of edges $\mathcal{E}$ comprises two subsets: particle–particle edges $\mathcal{E}_{pp} = \{e_{ij}\}$ and particle–wall edges $\mathcal{E}_{pw} = \{e_{iw}\}$. For generating the graphs used during offline training, the edge sets $\mathcal{E}_{pp}$ and $\mathcal{E}_{pw}$ are constructed directly from the contact pairs reported by the source high-fidelity DEM simulation (i.e. from Altair® EDEM™) at each corresponding time step. This implicitly defines the interaction range learned by the model from the training data.

To capture temporal dependencies, features for particle and wall nodes incorporate information over a history window covering the current and τ preceding time steps, giving a total window size (W) of $\tau + 1$. For a particle node v_i at time step t_n, the input node feature vector $\epsilon_{v_i}(t_n)$ is constructed by concatenating features related to particle kinematics and boundary interactions:

$$\epsilon_{v_i}(t_n) = \text{concat}\left(\{\mathbf{V}_p(v_i, t_j)\}_{j=n-\tau}^{n}, \{\phi_{SDF}(v_i, t_j), \nabla\phi_{SDF}(v_i, t_j)\}_{j=n-\tau}^{n}\right) \tag{19}$$

where $\mathbf{V}_p(v_i, t_j)$ is the translational velocity of particle i at time t_j. Crucially, the SDF value $\phi_{SDF}(v_i, t_j)$ and its gradient $\nabla\phi_{SDF}(v_i, t_j)$ are calculated for all particle positions relative to the time-dependent boundary geometry (since the geometry is in oscillatory motion) at each time step t_j in the history window. This provides a continuous geometric and contact context to every particle node. The feature vector for the wall node, $\epsilon_{v_w}(t_n)$, includes information about the wall's state over the time window, such as its Center of Mass (CoM) position and rotational velocity, padded with zeros to match the dimensionality of particle node features for each snapshot in the window.

Edge features encode the relative spatial configuration or interaction properties between connected nodes identified by the DEM simulation during data generation. For particle–particle edges $e_{ij} \in \mathcal{E}_{pp}$, the features $\epsilon_{e_{ij}}(t_n)$ include the relative distance vector:

$$\epsilon_{e_{ij}}(t_n) = \{\boldsymbol{r}_{ij}(t_n)\} \quad \text{where } \boldsymbol{r}_{ij}(t_n) = \boldsymbol{X}_j(t_n) - \boldsymbol{X}_i(t_n) \tag{20}$$

For particle–wall edges $e_{iw} \in \mathcal{E}_{pw}$, the features $\epsilon_{e_{iw}}(t_n)$ represent the particle–wall interaction, using the SDF distance vector derived from the particle's SDF features:

$$\epsilon_{e_{iw}}(t_n) = \phi_{SDF}(v_i, t_n)\frac{\nabla\phi_{SDF}(v_i, t_n)}{\|\nabla\phi_{SDF}(v_i, t_n)\|} \tag{21}$$

These specific input features (velocity history, SDF history, relative positions) are chosen because they can be readily updated or recalculated during the recursive simulation phase using only the model's outputs (acceleration integrated to velocity and position) and the known boundary motion, enabling a closed-loop prediction while sliding the history window (W). This contrasts with features like contact forces or tangential overlaps, which are outputs of the DEM simulation but cannot be directly calculated during the surrogate's recursive loop without making further assumptions or predictions of unknown future contacts. All input node (particle and wall) and edge (particle–particle and particle–wall) features are normalized using the mean and standard deviation derived from the training dataset prior to being processed by the network. Separate normalization statistics are maintained for particle nodes, wall nodes, particle–particle edges, and particle–wall edges. Additionally, Gaussian noise with a standard deviation of 0.005 ($\sigma = 0.005$) is added to the normalized velocity features during training to enhance robustness during recursive inference.

The SGN architecture follows the established Encoder-Processor-Decoder paradigm. First, the Encoder employs independent Multi-Layer Perceptrons (MLPs), MLP_v and MLP_e, with ReLU activations to map the input node and edge features to initial latent embeddings, $\boldsymbol{h}_v^0$ and $\boldsymbol{h}_e^0$, respectively:

$$\boldsymbol{h}_v^0 = \text{MLP}_v(\epsilon_v) \quad \forall v \in \mathcal{V} \tag{22}$$

$$\boldsymbol{h}_e^0 = \text{MLP}_e(\epsilon_e) \quad \forall e \in \mathcal{E} \tag{23}$$

Note that while a single MLP_v is shown, distinct initial layers or feature handling could be applied to particle vs. wall nodes if necessary. Similarly, MLP_e processes both edge types. Second, the Processor consists of L_p interaction layers, performing iterative message passing to refine node representations by propagating information through the graph. Within each layer l:

1. An edge-update MLP, ψ^e, computes messages based on the embeddings of connected nodes and the edge itself:

$$\boldsymbol{m}_{e_{ij}}^l = \psi^e(\boldsymbol{h}_{v_i}^l, \boldsymbol{h}_{v_j}^l, \boldsymbol{h}_{e_{ij}}^l) \tag{24}$$

2. An aggregation function, $\bigoplus$ (specifically, element-wise mean in our implementation), pools incoming messages for each node v_i (including the wall node v_w) from its neighborhood $\mathcal{N}(i)$:

$$\bar{\boldsymbol{m}}_{v_i}^l = \bigoplus_{j \in \mathcal{N}(i)} \boldsymbol{m}_{e_{ij}}^l \tag{25}$$

Table 2
Summary of SGN graph inputs and targets. Per-snapshot node features are concatenated over a window $W = \tau + 1$.

Component	Value	
Node features		
Particle (per snapshot)	$[\mathbf{V},\ \phi_{\mathrm{SDF}},\ \nabla\phi_{\mathrm{SDF}}]$	(Dim. 7)
Wall (per snapshot)	$[\mathbf{CoM},\ \mathrm{RPM}]$ (zero-padded)	(Dim. 7)
Edge features		
PP edge (particle–particle)	$\boldsymbol{r}$	(Dim. 3)
PW edge (particle–wall)	$\phi_{\mathrm{SDF}} \cdot \hat{\nabla}\phi_{\mathrm{SDF}}$	(Dim. 3)
Targets		
Node (per particle)	$\mathbf{A}_{norm}$	(Dim. 3)
Global (per graph)	ΔE_{norm}	(Dim. 1)

Symbols: $\mathbf{V} = [v_x, v_y, v_z]$ (velocity); ϕ_{SDF} (signed distance to wall surface); $\nabla\phi_{\mathrm{SDF}} = [\partial_x\phi_{\mathrm{SDF}}, \partial_y\phi_{\mathrm{SDF}}, \partial_z\phi_{\mathrm{SDF}}]$ (SDF gradient); $\hat{\nabla}\phi_{\mathrm{SDF}} = \nabla\phi_{\mathrm{SDF}}/\|\nabla\phi_{\mathrm{SDF}}\|$ (unit normal); $\mathbf{CoM} = [\mathrm{CoM}_x, \mathrm{CoM}_y, \mathrm{CoM}_z]$ (jar center-of-mass position); $\boldsymbol{r} = \boldsymbol{X}_j - \boldsymbol{X}_i$ (PP separation vector); $\mathbf{A}_{norm} = [a_x, a_y, a_z]$ (normalized acceleration); ΔE_{norm} (per-step normalized dissipated energy).

3. A node-update MLP, ψ^v, updates the node embedding using its previous state and the aggregated message:

$$\boldsymbol{h}_{v_i}^{l+1} = \psi^v(\boldsymbol{h}_{v_i}^{l}, \bar{\boldsymbol{m}}_{v_i}^{l}) \tag{26}$$

Third, the Decoder utilizes an MLP, MLP_d, to map the final node embeddings from the processor, $\boldsymbol{h}_v^{L_p}$, to the target outputs. For particle nodes v_p, the primary target is the normalized particle acceleration $\boldsymbol{A}_{\mathrm{norm}}(v_p, t_n)$, as this allows the model to drive the system's dynamics through integration. The output for the wall node v_w is also computed, but disregarded for the primary task.

$$\boldsymbol{A}_{\mathrm{norm}}(v_p, t_n) = \mathrm{MLP}_d(\boldsymbol{h}_{v_p}^{L_p}) \tag{27}$$

Additionally, a Global Readout branch aggregates the final node embeddings $\boldsymbol{h}_v^{L_p}$ (via mean pooling across all particle nodes with an edge (i.e. particles undergoing a collision)) and passes the result through a separate MLP to predict a normalized global system property, specifically for this work, the incremental energy dissipation $\Delta E_{\mathrm{norm}}(t_n)$ (result from adding Eqs. (17) and (18)). Because ΔE_{norm} is learned from DEM targets, calibration across operating regimes depends on training coverage of dissipation magnitudes and collision statistics (e.g., RPM, amplitude, fill ratio, materials). This global variable was selected due to its established relevance in characterizing the potential outcomes of mechanochemical processes [4]. Table 2 summarizes the graph inputs and targets used in this work.

2.2.2. Training

The network's learnable parameters θ are trained offline using supervised learning on data generated by high-fidelity DEM simulations, which are sampled at a fine time resolution (Δt_{DEM}). Accurately defining Δt_{DEM} is essential for enabling the model to capture the dynamics effectively. In this work, because the median collision duration is approximately 0.0005 s, we selected a Δt_{DEM} of 0.0001 s to ensure that the model can accurately learn the evolution of collisions. Although this parameter can be fine-tuned depending on the application, in systems where individual collisions are critical for realistic granular flow, it is advisable not to exceed the median collision duration. It is also important to note that the training data utilized comprises snapshots taken only after the initial particle generation phase is complete and the motion of the milling jar has commenced, focusing the model on the relevant dynamic interactions. Then, the first 4.5 s are used for training, resulting in a total of 45,000 snapshots.

The model was trained on one Nvidia A100 [35] with a batch size of 2 and a learning rate initially set to 1e−4, managed by an Adam optimizer and an exponential scheduler targeting a final rate of 1e−6 with a maximum of 2000 training epochs.

The objective is to minimize a suitable loss function between the SGN's predictions ($\boldsymbol{A}_{\mathrm{norm}}, \Delta E_{\mathrm{norm}}$) and the corresponding ground truth values derived from the DEM data. Specifically, the total loss function $\mathcal{L}_{total}$ is a weighted sum of the loss calculated for the primary task (node acceleration prediction, $\mathcal{L}_{node}$) and the loss for the auxiliary global prediction task (energy dissipation, $\mathcal{L}_{global}$) as seen in Eq. (28):

$$\mathcal{L}_{\mathrm{total}} = \alpha\,\mathcal{L}_{\mathrm{node}} + \mathcal{L}_{\mathrm{global}} \tag{28}$$

with

$$\mathcal{L}_{\mathrm{node}} = \frac{1}{N}\sum_{k=1}^{N} H_\delta\big(\mathbf{A}_{\mathrm{norm},k}^{\mathrm{pred}} - \mathbf{A}_{\mathrm{norm},k}^{\mathrm{gt}}\big), \tag{29a}$$

$$\mathcal{L}_{\mathrm{global}} = \frac{1}{B}\sum_{b=1}^{B} H_\delta\big(\Delta E_{\mathrm{norm},b}^{\mathrm{pred}} - \Delta E_{\mathrm{norm},b}^{\mathrm{gt}}\big). \tag{29b}$$

where B is the number of graphs in the mini-batch, N is the total number of particle-node acceleration components in the batch (wall node excluded), $\mathbf{A}_{\mathrm{norm}}^{\mathrm{pred}}$ and $\mathbf{A}_{\mathrm{norm}}^{\mathrm{gt}}$ are the predicted and ground-truth normalized per-particle accelerations (the index k runs over all particle components), and $\Delta E_{\mathrm{norm}}^{\mathrm{pred}}$ and $\Delta E_{\mathrm{norm}}^{\mathrm{gt}}$ are the predicted and ground-truth normalized per-step dissipated-energy increments (the former produced by the global head from pooled node embeddings over particle nodes with at least one incident edge). We use the Huber penalty (Eq. (30)) with threshold $\delta = 2$:

$$H_\delta(e) = \begin{cases} \dfrac{1}{2\delta}\,e^2, & |e| < \delta, \\ |e| - \dfrac{\delta}{2}, & |e| \geq \delta. \end{cases} \tag{30}$$

The weighting factor α (set to 3.0 in our implementation) allows for prioritizing the accuracy of the particle dynamics prediction during training relative to the global energy prediction. In this work, both $\mathcal{L}_{node}$ and $\mathcal{L}_{global}$ utilize the Huber loss function. Due to the high-energy collisions inherent in ball milling, the ground truth acceleration distribution can exhibit large spikes relative to median values. This occurs not only because of the large contact forces during impacts but also because sampling the DEM simulation at Δt_{DEM} can alias high-frequency events. Collision dynamics occurring at the DEM's finer internal timestep might not be fully resolved between samples, leading to apparent discontinuities or spikes in the calculated acceleration used for training. Consequently, the Huber loss is employed for its robustness to such outliers, combining the benefits of L2 loss (mean squared error, MSE) for small errors and L1 loss (mean absolute error, MAE) for large deviations. Specifically, for errors below a predefined threshold δ, it penalizes deviations quadratically, ensuring smooth convergence, while for errors above the threshold, it applies a linear penalty, thereby reducing the influence of extreme values on the overall training process.

2.2.3. Recursive stage

Once trained, the SGN model enables efficient online recursive simulation. The process (Fig. 5) starts by initializing a state window with $\tau + 1$ snapshots from DEM data. Then, for each subsequent time step t_{n+1}:

1. Input features are constructed from the current window ending at t_n.
2. The SGN predicts normalized acceleration $\boldsymbol{A}_{\mathrm{norm}}(t_n)$ and energy increment $\Delta E_{\mathrm{norm}}(t_n)$.
3. Predictions are denormalized to physical units $\boldsymbol{A}(t_n)$ and $\Delta E(t_n)$.
4. Particle states are advanced using a numerical integrator (e.g., Euler) with the surrogate simulation time step Δt:

$$\boldsymbol{V}(t_{n+1}) = \boldsymbol{V}(t_n) + \boldsymbol{A}(t_n)\Delta t \tag{31}$$

$$\boldsymbol{X}(t_{n+1}) = \boldsymbol{X}(t_n) + \boldsymbol{V}(t_n)\Delta t \tag{32}$$

The definition of Δt is critical for ensuring stability during the recursive simulation stage. We adopt a timestep equal to the training dataset's sample frequency (0.0001 s). Using larger timesteps leads to stability issues, as collisions may be missed or excessive penetration between particles and the wall may occur, resulting in exponential error accumulation. This limitation primarily arises from the high-energy dynamics, which inherently involve high velocities and accelerations that make spatial definition overly sensitive to small changes. Nonetheless, this surrogate solving timestep represents a 104× relaxation compared to the DEM solving timestep used in Altair® EDEM™ (see Table 1).

5. The wall boundary CoM for t_{n+1} is obtained via extrapolation (using pre-calculated periodic splines).
6. New SDF values are recomputed from the updated particle positions $\boldsymbol{X}(t_{n+1})$ and the updated wall geometry. The graph connectivity is then rebuilt: particle–particle edges $e_{ij} \in \mathcal{E}_{pp}$ are added if $\|\boldsymbol{X}_j(t_{n+1}) - \boldsymbol{X}_i(t_{n+1})\| \leq r_c$ (we recommend $r_c \leq 1.5\,R$), and particle–wall edges $e_{iw} \in \mathcal{E}_{pw}$ are added if $\phi_{\mathrm{SDF}}(v_i, t_{n+1}) \geq \phi_{\mathrm{pw}}$ with $\phi_{\mathrm{SDF}} < 0$ representing zones inside the surface. The choice of ϕ_{pw} is guided by the DEM contact distribution (e.g., capturing $\geq$90% of ground-truth PW contacts; see Fig. 4). In this work, we set the value at -0.0052 m, but it will vary according to the ball's kinetic energy and physical properties, as they will dictate the depth of penetration. At sharp edges/vertices the nearest-point direction can be ambiguous; since edge creation uses only ϕ_{SDF}, this does not rely on normals, which are used to compute edge features. Residual near-wall noise is handled in Step 7.
7. Optional correction (snap-back) is applied only for shallow, near-wall penetrations: if $\phi_{\mathrm{SDF}}(v_i, t_{n+1}) \geq \phi_{\mathrm{sb}}$ with $\phi_{\mathrm{pw}} < \phi_{\mathrm{sb}} < 0$, we project to the threshold value:

$$\mathbf{x}_i(t_{n+1}) \leftarrow \mathbf{x}_i(t_{n+1}) - \left(\phi_i - \phi_{\mathrm{sb}}\right)\mathbf{n}_i, \quad \text{where } \phi_i = \phi_{\mathrm{SDF}}(v_i, t_{n+1}),$$
$$\mathbf{n}_i = \nabla\phi_i / \|\nabla\phi_i\|.$$

For deeper overlaps ($\phi_{\mathrm{SDF}} < \phi_{\mathrm{sb}}$) no snap-back is used; the contact dynamics resolve the interaction (parameters ϕ_{pw} and ϕ_{sb} are listed in Table 3). The optional correction's sole purpose is to prevent nonphysical interpenetration from accumulating due to prediction error or integration overshoot.
8. A new snapshot dictionary for t_{n+1} is assembled using the updated states and recalculated geometric features (including the re-determined contacts/edges).
9. The time window is advanced by removing the oldest snapshot and adding the new one.

This iterative process can be visualized in Fig. 5 and allows the surrogate model to generate the system's stable evolution over extended periods, driven solely by its own predictions after initialization. A summary of the parameters used for the architecture of the SGN and the recursive stage is presented in Table 3. The SGN was implemented using Pytorch 2.1.

3. Results and discussion

In this section, the performance of the SGN surrogate model is evaluated using three distinct assessment methods. First, we measure how accurately the model predicts the bulk dynamics of the standard high-energy ball milling process in the *Emax* machine. Second, we assess performance using a mechanochemistry-specific variable: the global energy dissipation of the system, which continuously increases as collisions occur. Our previous work has shown that this variable can effectively characterize a mechanochemical process from a mechanical standpoint [4]. Third, we evaluate the model's generalization by testing its ability to handle unseen motions and modifications to the base geometry. These evaluations are crucial to demonstrate the potential of the method in the iterative design processes required to scale up and optimize processes. A sensitivity analysis of critical hyperparameters in Table 3, together with an ablation study on the relevance of the global loss $\mathcal{L}_{\mathrm{global}}$, can be found in Appendix, where we show how the predictive capacity and stability of the model are affected.

Table 3
Key parameters for SGN model architecture and recursive simulation loop.

Parameter	Value	Unit/Description
Model architecture		
History window Size ($\tau + 1$)	7	Time steps
Hidden dimension	256	–
MLP layers	4	–
Interaction layers (L_p)	1	–
Huber loss threshold (δ)	2	–
Loss weighting factor (α)	3	–
Recursive simulation loop		
Time step (Δt)	1×10^{-4}	s
Integration type	Euler	–
PP contact threshold (r_c)	0.0015	m
PW contact threshold (ϕ_{pw})	−0.0052	m
Snap-back threshold (ϕ_{sb})	−0.0049	m

Contact rules: We adopt $\phi_{\mathrm{SDF}} < 0$ inside the geometry. PP edges are added when $\|\Delta\mathbf{x}\| \leq r_c$. PW edges are added when $\phi_{\mathrm{SDF}} \geq \phi_{pw}$. Snap-back is applied only when $\phi_{\mathrm{SDF}} \geq \phi_{sb}$, with $\phi_{pw} < \phi_{sb} < 0$.

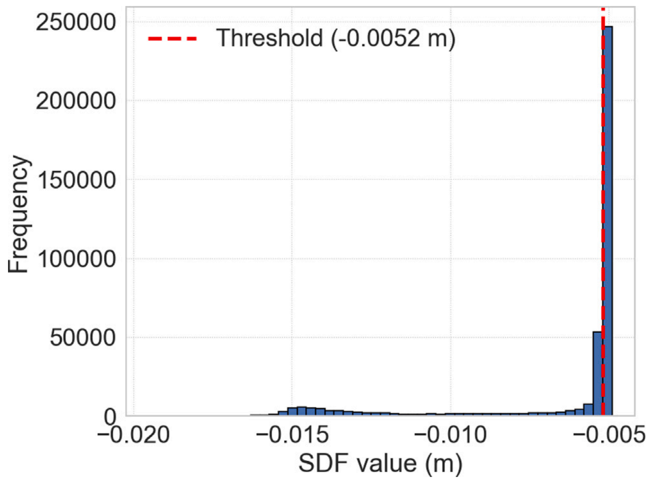


Fig. 4. Distribution of SDF values for particle–wall contacts. Values closer to zero indicate proximity to the wall. The defined threshold of −0.0052 covers over 90% of all particle–wall contacts.

3.1. Standard high-energy milling process

The model was trained on 4.5 s of high-resolution data generated by a DEM simulation, which also serves as the initial benchmark (Fig. 6). To evaluate the accuracy and stability of the model when predicting the bulk motion of particles in the system, it is run recursively (online) for 15 s. At each available time step t, we compare the predicted ball positions $\hat{\mathbf{X}}_t$ to the ground-truth reference $\mathbf{X}_t$ (see Fig. 7), and compute spatial MSE according to Eq. (33). We report the MSE because, while the Huber loss was used during training to reduce the influence of occasional high-frequency acceleration spikes, MSE provides a single, widely understood scalar that directly quantifies average squared deviations in particle positions for straightforward benchmarking of bulk-dynamics accuracy.

$$\mathrm{MSE}_t = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{\mathbf{X}}_t^{(i)} - \mathbf{X}_t^{(i)}\right\|^2 \tag{33}$$

Although this is the simplest objective of the model, it is crucial to ensure stability and accurate global predictions, which rely on accurate

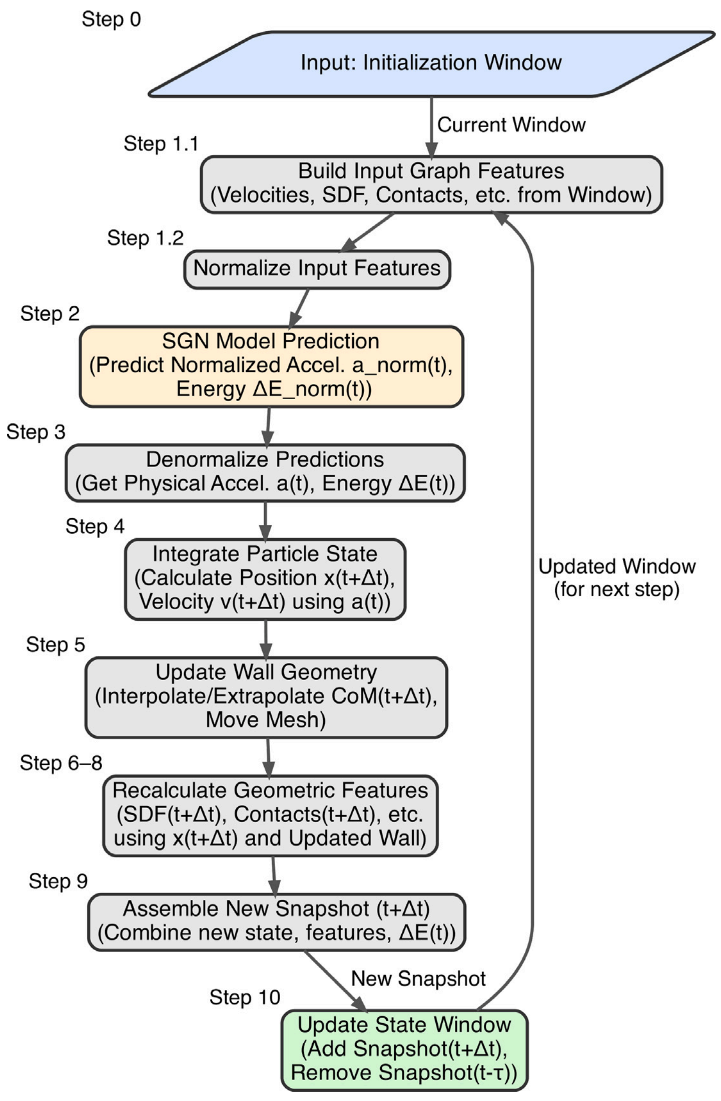

Step 0
Input: Initialization Window
Current Window
Step 1.1
Build Input Graph Features (Velocities, SDF, Contacts, etc. from Window)
Step 1.2
Normalize Input Features
Step 2
SGN Model Prediction (Predict Normalized Accel. a_norm(t), Energy ΔE_norm(t))
Step 3
Denormalize Predictions (Get Physical Accel. a(t), Energy ΔE(t))
Step 4
Integrate Particle State (Calculate Position x(t+Δt), Velocity v(t+Δt) using a(t))
Step 5
Update Wall Geometry (Interpolate/Extrapolate CoM(t+Δt), Move Mesh)
Step 6–8
Recalculate Geometric Features (SDF(t+Δt), Contacts(t+Δt), etc. using x(t+Δt) and Updated Wall)
Step 9
Assemble New Snapshot (t+Δt) (Combine new state, features, ΔE(t))
New Snapshot
Step 10
Update State Window (Add Snapshot(t+Δt), Remove Snapshot(t-τ))
Updated Window (for next step)

Fig. 5. SGN recursive loop flowchart.

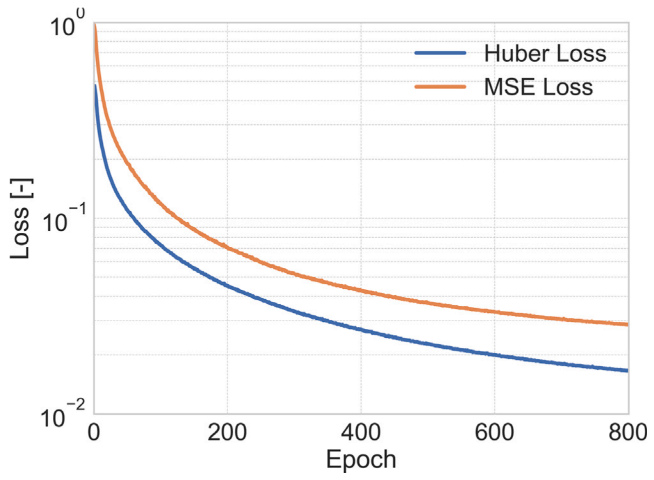


Fig. 6. Standardized loss history for the SGN surrogate model. Note that the MSE loss is shown for comparison purposes.

bulk dynamics and proper collision identification. Looking at Fig. 7, it is possible to visualize the oscillatory motion of the system and its effect on the predictions. The peaks of these curves coincide with the moments where the milling jar changes direction in the x-axis. Then, the predicted global energy dissipation can be compared in Fig. 8.

Thus, the combination of Figs. 7 and 8 shows that the SGN model is capable of accurately representing the bulk motion of the high-energy system and maintaining an accurate track of the energy dissipation that occurs in the system with a stable relative error of 2.78% while using a solving timestep 104× larger than the original DEM simulation.

To provide a more intuitive representation of the accuracy of the predictions, we used Blender 4.3.2 to reproduce the motion of the milling balls predicted by the SGN and compare them to the original DEM visualization. This can be seen in Fig. 9.

3.2. Unseen motions

Upon establishing that the model can accurately represent the original bulk dynamics and energy dissipation, we now test the model with two new, unseen motions that have a direct effect on the dynamics of the milling balls. To define these arbitrary motions, we use two different Lissajous curves as they operate under the cyclic behavior that

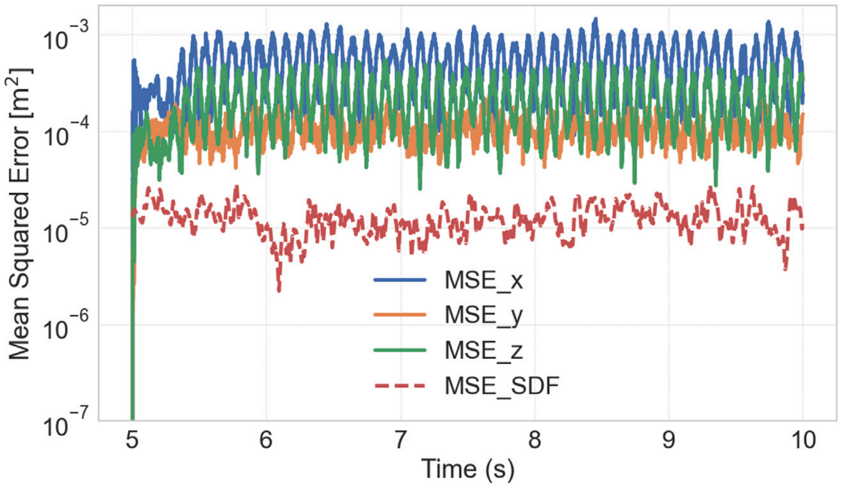


Fig. 7. MSE loss history for the recursive (online) stage of the SGN. Here, time is measured from $t = 5$ s, marking the start of motion in the DEM simulation, which provides reference data only up to $t = 10$ s. Beyond this point, the model is let run recursively until $t = 15$ s.

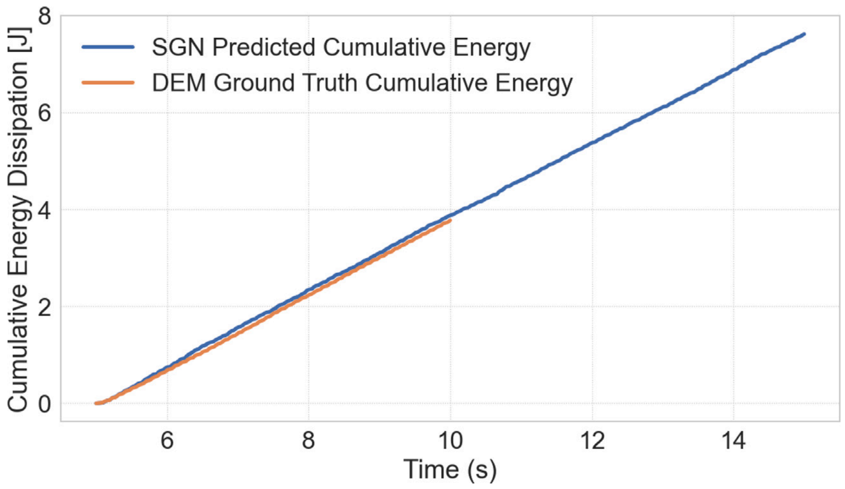


Fig. 8. Cumulative energy dissipation: comparison of SGN-predicted versus DEM ground-truth energy dissipation. Note that DEM simulation stops at $t = 10$ s.

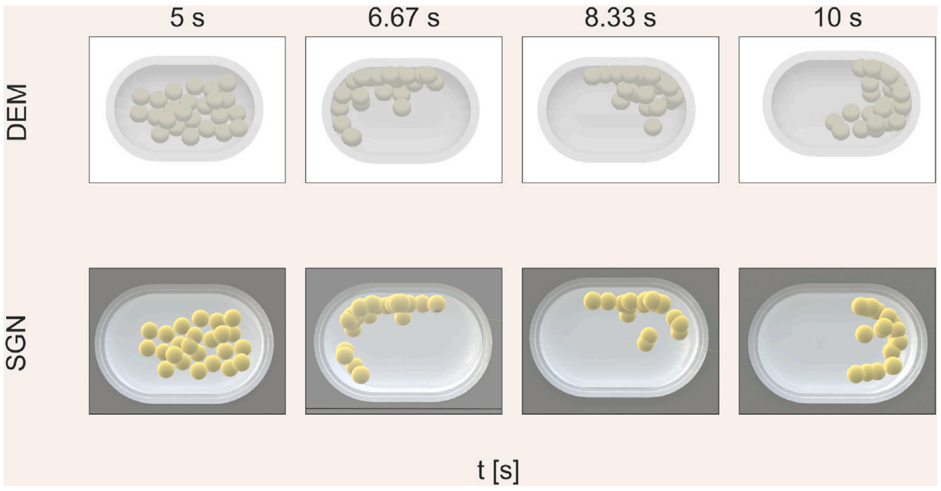


Fig. 9. Comparison grid of DEM and SGN simulation results.

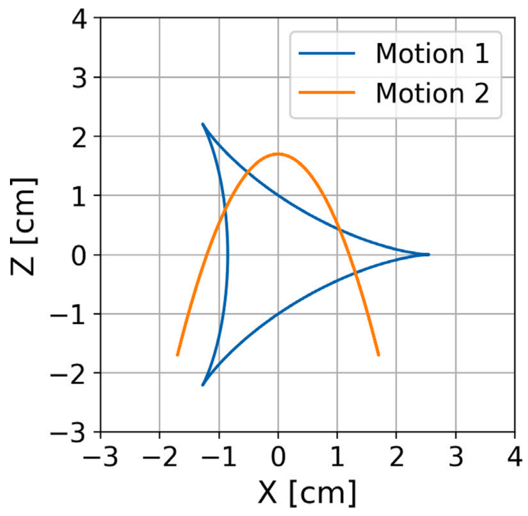


Fig. 10. Lissajous motions used for testing the SGN.

a typical high-energy ball mill experiences. The trajectories of the two tested motions are shown in Fig. 10, and where implemented in Altair® EDEM™ according to Eqs. (34) and (35).

Motion 1

$$x_1(t) = 1.7\ \sin\left(10\pi\, t\right)\ -\ 0.85\ \cos\left(20\pi\, t\right),$$
$$z_1(t) = 1.7\ \cos\left(10\pi\, t\right)\ -\ 0.85\ \sin\left(20\pi\, t\right). \tag{34}$$

Motion 2

$$x_2(t) = 1.7\ \sin\left(10\pi\, t\right),$$
$$z_2(t) = 1.7\ \cos\left(20\pi\, t\right). \tag{35}$$

The model is capable of reproducing accurate and stable predictions of the bulk dynamics for both unseen motions, with MSE errors comparable to those of the standard case presented in the previous section, as per Fig. 11 and can be visualized in Figs. 12 and 13.

The model's ability to predict energy dissipation is substantially weaker than its performance on bulk dynamics. For Motion 1, the predictions maintain a constant relative error of 23.84%. For Motion 2, the error rises to a steady 46.62%. Nonetheless, the shape of the time series is captured remarkably well. In both cases, the simulation follows not only the overall trend but also the short-time-scale wiggles (i.e., the small, rapid oscillations super-imposed on the mean growth, so the predicted and reference curves rise and fall almost in lock step). This can be quantified by the Pearson correlation coefficient ($r = 0.9998$ for both cases). Pearson's r measures linear association between two variables: an r of 1 means every peak, dip, and inflection in one series occurs at exactly the same relative level in the other (perfect synchrony of the wiggles), while an r of 0 would indicate no consistent linear pattern. Because r is insensitive to uniform scaling or offsets, the coefficient can be close to 1 even when the absolute magnitudes are biased, as we see here. In other words, the model slightly underestimates the magnitude of each dissipation event, but it gets the timing and relative spacing of those events almost perfectly right. This can be visualized in Figs. 14 and 15. To address this scaling bias, one could expand the training set to cover a broader spectrum of dissipation magnitudes, introducing both lower and higher energy cases. This would encourage the network to learn appropriate scaling factors across broader operational ranges, reducing systematic bias and improving calibration of its outputs. Finally, by exposing the surrogate to diverse collision magnitudes and scenarios, its transferability to new systems should improve, potentially eliminating the need for manual post-processing adjustments. We intend to explore this in a future study.

3.3. Modifications to original geometry

To test the hypothesis that the underestimation of energy dissipation magnitude in unseen motions arises from a mismatch in the distribution of known collision and velocity features, and at the same time probe the model's ability to handle unseen geometric features, we introduce a slightly tweaked design to boost the collision frequency. Specifically, we insert a cylindrical barrier at the center of the jar (see Fig. 16) and drive the system with Motion 1 (see Fig. 10).

By combining this geometry modification with Motion 1, we can also verify whether the surrogate still generates physically plausible particle trajectories and remains numerically stable when both jar shape and motion lie outside its training reference. To illustrate our hypothesis in action, we compare the model's predicted cumulative energy dissipation for the modified geometry + Motion 1 case against the original geometry + Motion 1 ground-truth curve (see Fig. 17). Although this reference no longer corresponds to the actual physics of the modified geometry, it serves as a controlled experiment: if the barrier boosts collision frequency, thus increasing the net impact statistics, then, when we compare its predictions to the original geometry baseline, the underestimation bias should shrink.

Indeed, we observe a substantial drop in relative error to 5.97%, confirming that the original dissipation bias stems from a mismatch in feature distributions. The numerical agreement of this comparison has no physical validity; it exists solely to validate our distribution-matching hypothesis. Furthermore, the snapshots in Fig. 18 are presented solely for illustration; they demonstrate that the model accurately handles modifications to the original geometry while remaining stable over time.

3.4. Runtime

On an Apple M1 Max (10-core CPU; CPU-only to match the DEM run requirements), an SGN rollout of the Emax case (15 s physical time, $\Delta t = 10^{-4}$ s) completed in 3443 s (~57 min), whereas the corresponding DEM simulation to solver completion took 22,734 s (~379 min). Thus, the SGN was about 6.6× faster with an 84.9% shorter runtime. This comparison excludes the subsequent post-processing of DEM data required to compute the dissipation target ΔE; including it would further increase the DEM wall clock as it requires manual processing, so we report the solver time only. We observe similar speedups across all our tested cases. In profiling, SDF re-evaluation dominates SGN runtime due to the rapidly moving boundary; this cost is hard to avoid because accurate particle–wall spatial information must be maintained each step for stability. Additionally, increasing Δt leads to missed contacts or boundary escapes given the high rotational speed.

4. Conclusions

In this work, we developed and validated a Signed-Distance-Function Graph Neural Network (SGN) that serves as a faithful surrogate for Discrete Element Method (DEM) simulations of high-energy ball milling. By embedding the jar geometry directly through an SDF field that supports dynamic translational motion, the model overcomes the static, non-translational boundary limitations of previous surrogates. When coupled with a message-passing graph network, it captures both particle–particle and particle–wall interactions effectively. In contrast with previous surrogates, the SGN is also specifically designed to handle the high-impact velocities and collision frequencies characteristic of high-energy milling processes. Training on 45,000 high-resolution DEM snapshots, the SGN simultaneously learns local accelerations and a global energy dissipation metric, providing a physics-aware description that goes beyond purely kinematic fits. This secondary metric is especially informative for mechanochemistry because cumulative dissipated energy directly tracks the mechanical work that activates solid-state reactions, but it could be changed to accommodate other

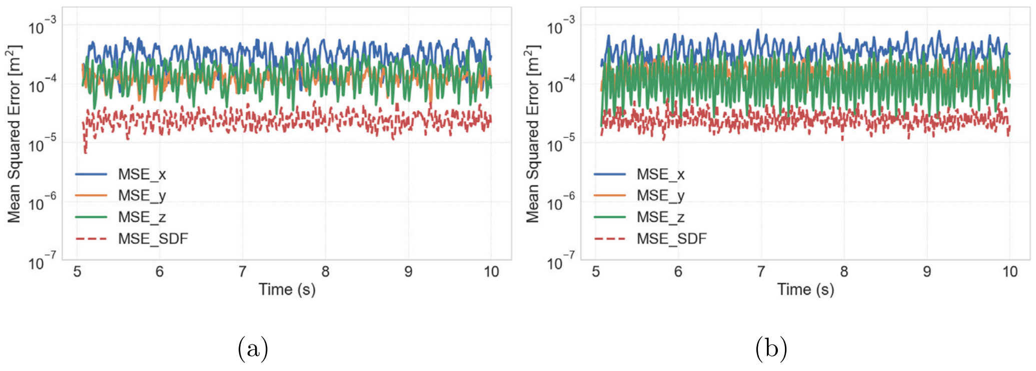


Fig. 11. MSE loss history comparison for (a) Motion 1 and (b) Motion 2.

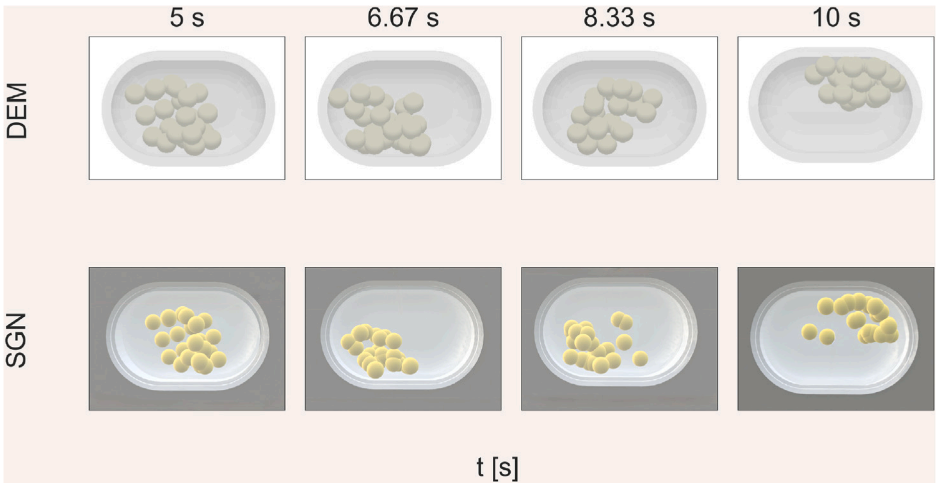


Fig. 12. Comparison grid of DEM and SGN simulation results for unseen Motion 1.

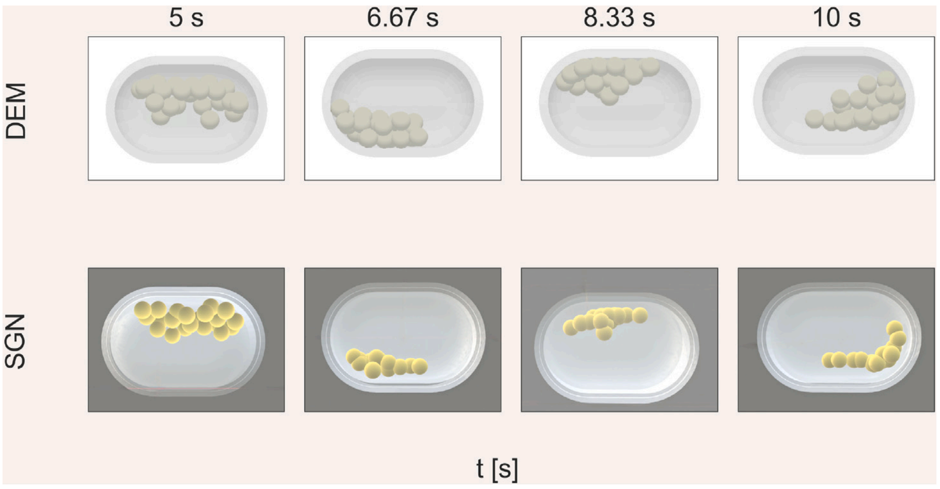


Fig. 13. Comparison grid of DEM and SGN simulation results for unseen Motion 2.

applications. For instance, in fluidized bed reactors the model could instead output granular temperature to monitor mixing efficiency; in hopper or silo flows, it could report stress accumulation to predict clogging by swapping the global target from ΔE_{norm} to a wall-region stress (e.g., von Mises stress in a region of interest) and time-integrating it over the rollout; and in continuous granulation processes it could track particle residence time to optimize throughput.

Benchmarking against a reference DEM simulation of the *Emax* mill revealed that the surrogate reproduces bulk motion with a mean squared error plateau of $\sim 2 \times 10^{-4}$ m^2 and tracks cumulative energy dissipation with a stable 2.8% relative error. Importantly, these results are obtained with a time step of 1×10^{-4} s, equivalent to a 104× relaxation over the DEM solver step (9.5×10^{-7} s). The ability to function with such relaxed temporal resolution while remaining numerically stable makes the SGN a promising drop-in replacement for exploratory studies, sensitivity scans and digital-twin applications. On runtime, the CPU-only rollout was ∼6.6× faster than the DEM solver, and this comparison excludes the additional DEM post-processing required to

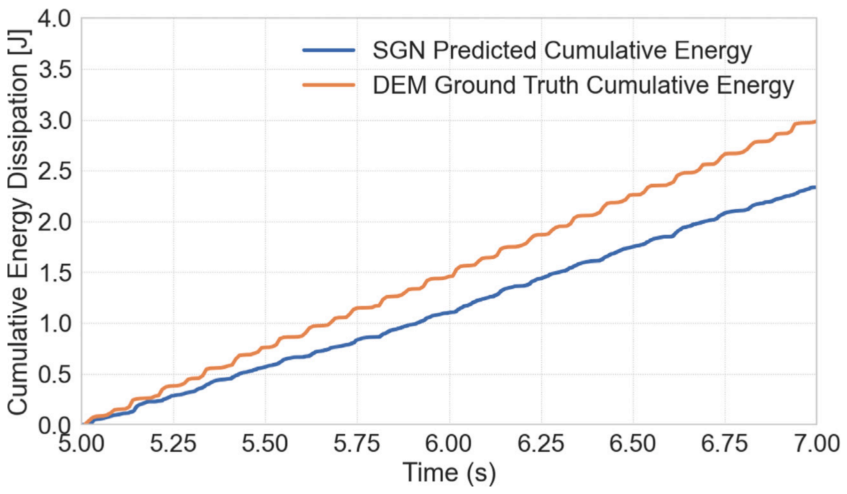


Fig. 14. Cumulative energy dissipation for unseen Motion 1: comparison of SGN-predicted versus DEM ground-truth energy dissipation. Note that we plot until $t = 7$ s to facilitate the observation of the dissipation's evolution.

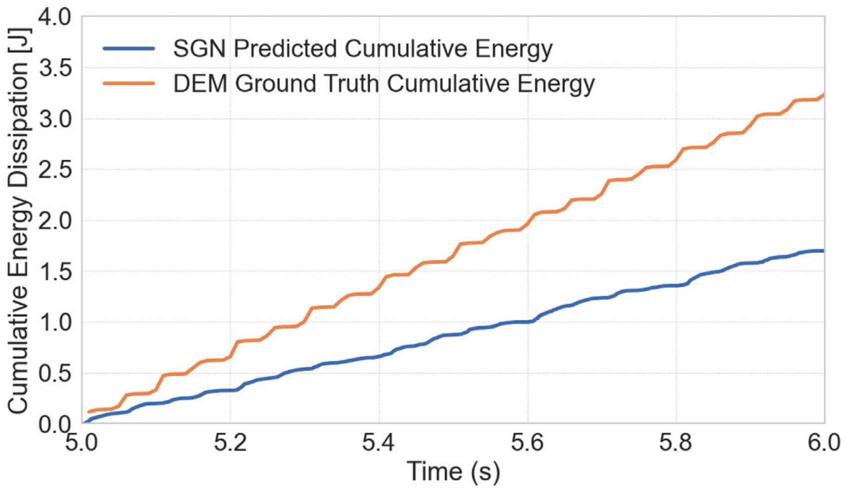


Fig. 15. Cumulative energy dissipation for unseen Motion 2: comparison of SGN-predicted versus DEM ground-truth energy dissipation. Note that we plot until $t = 6$ s to facilitate the observation of the dissipation's evolution.

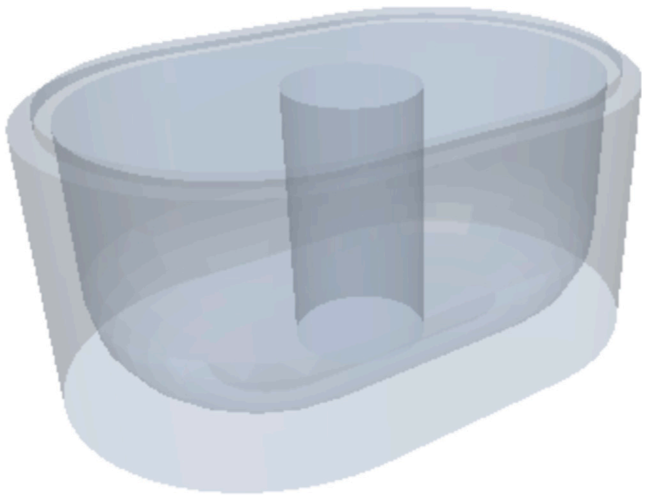


Fig. 16. Modification to original geometry by adding a cylinder at its center.

compute ΔE_{total}. Profiling shows that SDF evaluation dominates cost because fast boundary motion requires maintaining accurate per-particle spatial context each step, and increasing Δt to reduce these calls proved unviable at high RPM due to missed contacts and boundary escapes; further wall-clock gains are therefore most likely from SDF-derived calculation optimization

The surrogate also exhibits strong generalization when driven by two previously unseen motions. It preserved stable dynamics and kept trajectory errors within the same bounds observed for the trained motion. Although the absolute scale of energy dissipation was under-predicted (about 24% and 47%, respectively), the temporal evolution was captured with near perfect correlation ($r \approx 0.9998$), indicating that the model internalizes the underlying physics but needs broader training data to calibrate energy magnitudes outside of its original set.

Geometric robustness was tested by inserting a cylindrical obstacle, absent from the training set, and combining it with an unseen driving motion. The surrogate remained numerically stable under this combined distribution and, once the barrier increased the ball's collision frequency, its energy-dissipation error (measured against the original-geometry baseline used for hypothesis testing) fell to roughly 6%. While this comparison is not physically meaningful for the altered jar, it

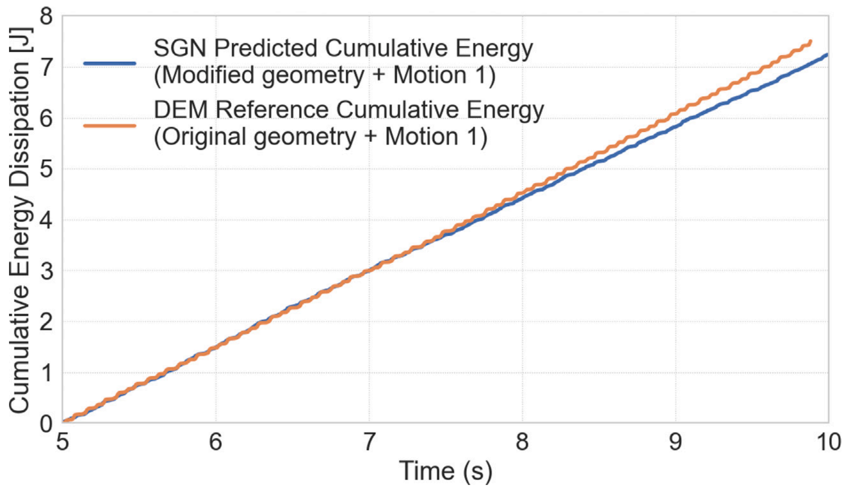


Fig. 17. Cumulative energy dissipation distribution-matching experiment: comparison of SGN-predicted energy versus the DEM baseline reference (original geometry + Motion 1). Note that this ground-truth comparison has no physical validity but serves to illustrate how slowing particle kinematics reduces the under-prediction bias.

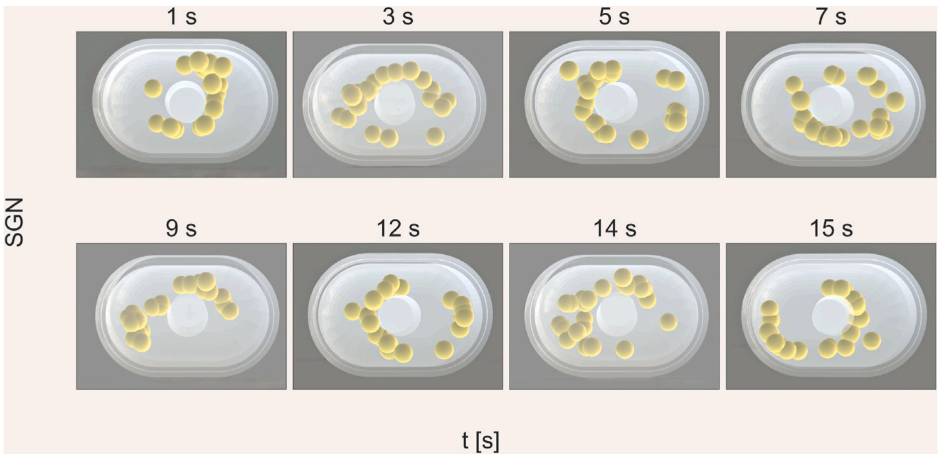


Fig. 18. SGN simulation results for modified geometry and Motion 1.

supports the idea that the earlier under-prediction stemmed from a distribution shift rather than a fundamental limitation of the model.

Overall, these findings demonstrate that SDF-based graph surrogates can compress high-fidelity DEM physics into a lightweight neural simulator that is both fast, transferable and can handle complex motions and geometries. Such capability opens avenues for iterative milling jar optimization, large-scale parameter sweeps for mechanochemical scale-up, and closed-loop control strategies.

Several challenges remain:

- *Material diversity*: the current network is trained on a single material system; extending the feature set to particle radius, fill ratio and restitution distributions is a logical next step.
- *Energy calibration*: the bias observed under out-of-distribution motions points to the need for data augmentation spanning a wider velocity and energy spectrum.
- *Uncertainty quantification*: ensemble or Bayesian message-passing variants would provide a direct performance indicator by relating the model's predictive variance to deviations from ground-truth DEM data, yielding confidence intervals around mean predictions. This is crucial because granular processes are inherently stochastic and sensitive to initial conditions, so quantifying predictive uncertainty helps detect out-of-distribution scenarios and supports risk-aware decision making in industrial deployment.

The present study marks an advance toward data-driven acceleration of granular-process simulations. By releasing all code and preprocessing tools as open source, we hope to catalyze community adoption, foster reproducibility, and ultimately shorten the innovation cycle for sustainable mechanochemical technologies.

CRediT authorship contribution statement

Santiago Garrido Nuñez: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Dingena L. Schott:** Writing – review & editing, Supervision. **Johan T. Padding:** Writing – review & editing, Supervision, Conceptualization.

Funding

This work was supported by the project SH2IPDRIVE: Sustainable Hydrogen Integrated Propulsion Drives, funded by the RVO (Rijksdienst voor Ondernemend) under grant MOB21013.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Johan T. Padding reports financial support was provided by Dutch Research Council. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Hyperparameter sensitivity and global-loss ablation

In this section, we analyze how the performance of the surrogate model is affected by a different selection of hyperparameters from those reported in Table 3, and an ablation of the global loss $\mathcal{L}_{\text{global}}$ (see Eq. (28)). Since the model has two objective predictions, we employ the following rationale: if the model maintains a stable error when predicting particle dynamics, then we test if the global energy dissipation is predicted accurately. We assess the influence of these hyperparameters based on the model's capacity to predict the standard milling process in the *Emax* machine.

History window (W)

The selection of the history window size is critical to allow the model to learn sufficient information about how a typical collision evolves in the system. Too long of a window will effectively introduce noise that the model will not be able to resolve and lead to unstable dynamic predictions (see Fig. A.2). Our results also indicate that selecting a window size that covers less than the median collision duration can lead to stable dynamic prediction, but will result in less accurate energy dissipation predictions (see Fig. A.1). Thus, we recommend selecting a window size that encompasses at least the median duration, and does not exceed this time by more than 40%. In the case of this work, the median collision duration is 0.0005 s, and each window frame contributes 0.0001 s.

Interaction layers (L_p)

The number of interaction layers defines how many times node messages are passed and aggregated before making a prediction. Intuitively, deeper interaction modules allow the model to capture more complex multi-body effects, but given the small number of particles in the system, the number of collisions with more than 2 elements involved is relatively scarce. As a result, using more than 1 layer leads to unstable predictions. When using 2 interaction layers, the dynamic predictions of the particle manage to stay within the jar bounds, but they become chaotic, leading the energy dissipation predictions to grow without bound (see Fig. A.3). Using even more layers (i.e. 4) leads to unstable dynamic predictions (see Fig. A.4). Nonetheless, we suspect that a system with significantly more multi-body interactions will necessitate more interaction layers. We recommend consulting the studies mentioned in the introduction since they cover systems where multi-body interactions are more prevalent.

Hidden dimension (neuron number)

The number of neurons in the model's hidden layers defines its representational capacity for capturing the nonlinear dynamics of particle collisions. Too few neurons constrain the model's ability to predict energy dissipation accurately, although the dynamics remain stable and precise (see Fig. A.5).

Conversely, an excessively large hidden dimension increases the risk of overfitting to training noise, which can manifest as unstable long term predictions or reduced generalization capability. However, determining the precise network width at which overfitting first appears would require a broader hyperparameter sweep, which was unnecessary for this work.

MLP layer number

The number of MLP layers defines the depth of successive nonlinear transformations applied to each node's aggregated features, thereby controlling the model's capacity to approximate complex mappings between the current particle states and their future dynamics. Using too little hidden layers (i.e. 2) leads to highly unstable dynamic predictions (Fig. A.6).

Similarly, using too many hidden layers can introduce vanishing or exploding gradient issues during training, increase the model's susceptibility to overfitting, and substantially raise computational cost. In our experiments, since four MLP layers achieved stable convergence and accurate predictions, we did not investigate deeper architectures.

PW contact threshold value (ϕ_{pw})

The PW contact threshold (ϕ_{pw}) defines the near-wall region in which a particle–wall edge is created (edges added when $\phi_{\text{SDF}} \geq \phi_{\text{pw}}$, with $\phi_{\text{SDF}} < 0$ inside the jar). A shallower threshold (less negative, closer to zero) narrows this band and can miss near-wall interactions or lead to particle escape from the domain. A deeper threshold (more negative) widens the band, increasing PW edge density. We probe with two perturbations around the baseline $-0.0052\,\text{m}$: a shallower $-0.0049\,\text{m}$ and a deeper $-0.0055\,\text{m}$, adjusting the snap-back level as $\phi_{\text{sb}} = \phi_{\text{pw}} + 0.0003\,\text{m}$ to maintain $\phi_{\text{pw}} < \phi_{\text{sb}} < 0$.

In both cases, the kinematics of the system remain stable, but the energy dissipation prediction behavior differs. With a PW threshold

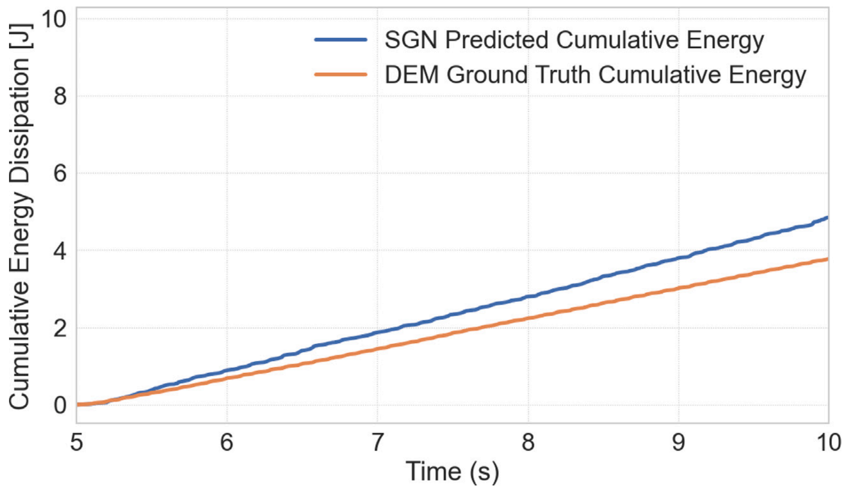


Fig. A.1. Cumulative energy dissipation for window size = 3.

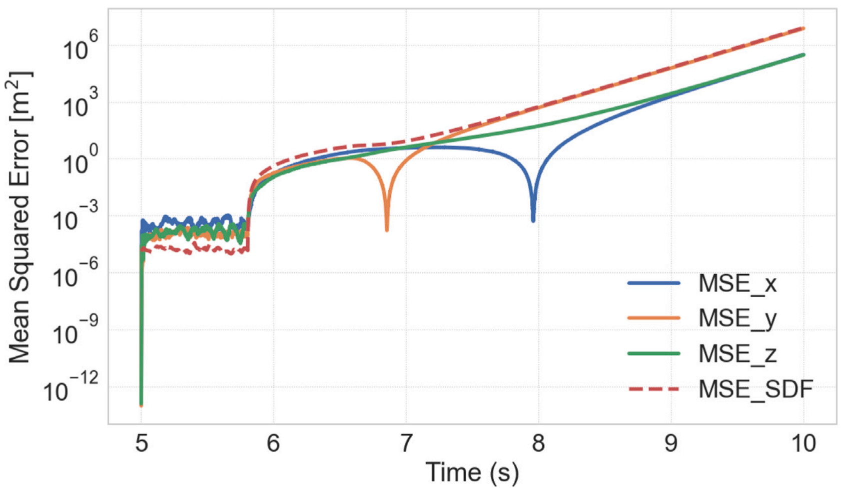


Fig. A.2. MSE loss history for window size = 9.

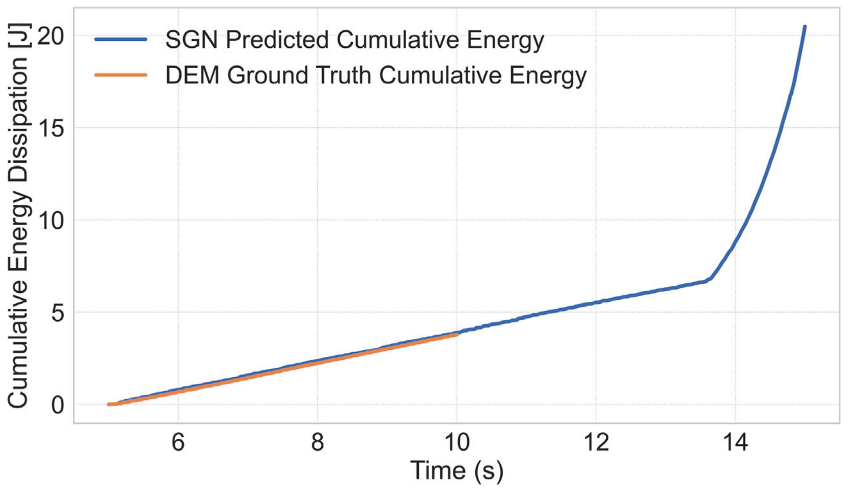


Fig. A.3. Cumulative energy dissipation for interaction layer size = 2.

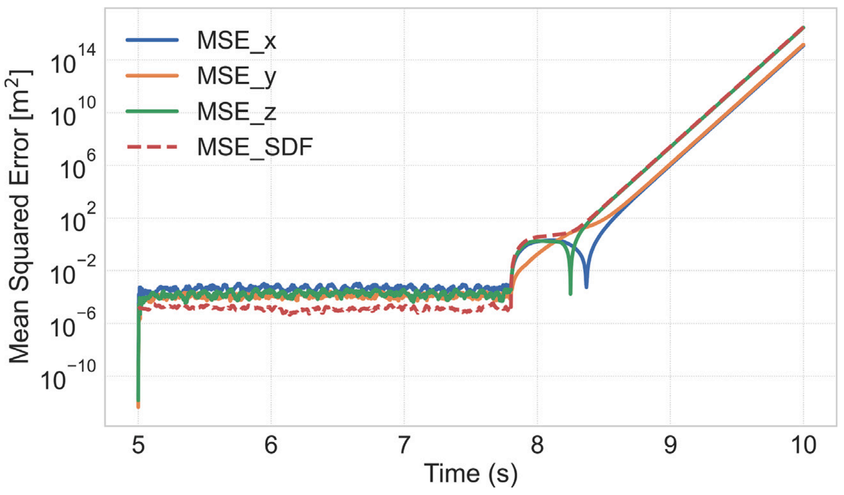


Fig. A.4. MSE loss history for interaction layer size = 4.

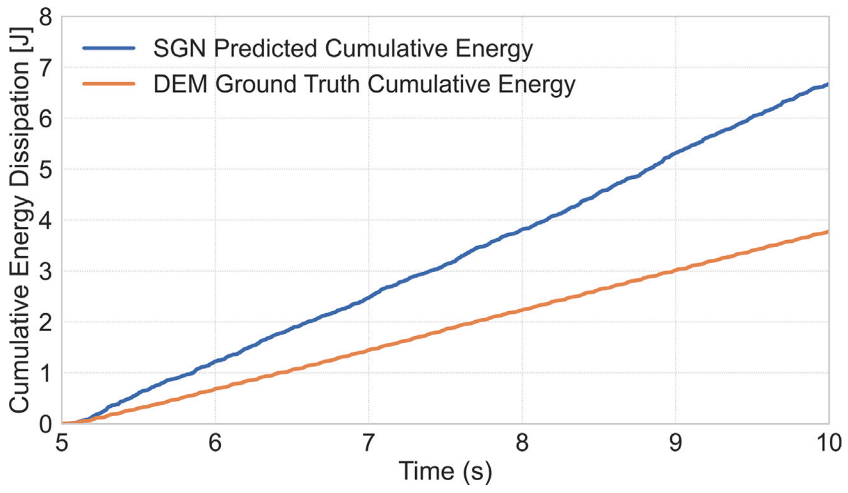


Fig. A.5. Cumulative energy dissipation for neuron number = 64.

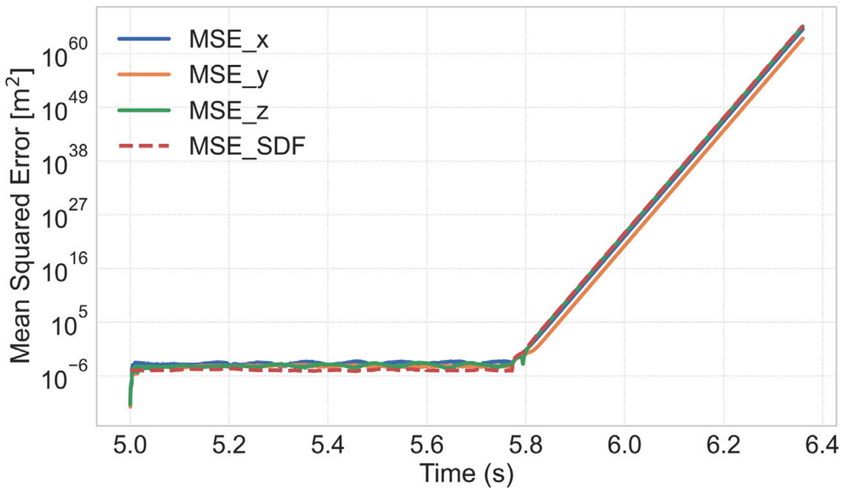


Fig. A.6. MSE loss history for MLP layer size = 2.

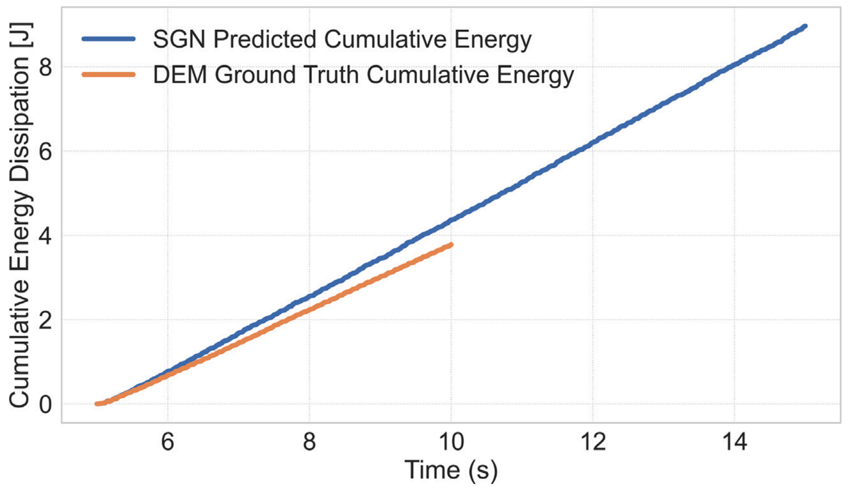


Fig. A.7. Cumulative energy dissipation for shallower ϕ_{pw} = −0.0049 m.

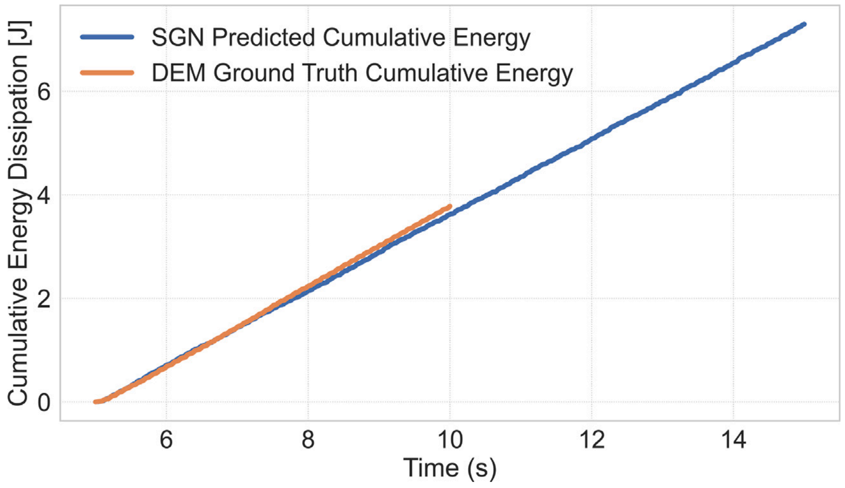


Fig. A.8. Cumulative energy dissipation for shallower $\phi_{pw} = -0.0055$ m.

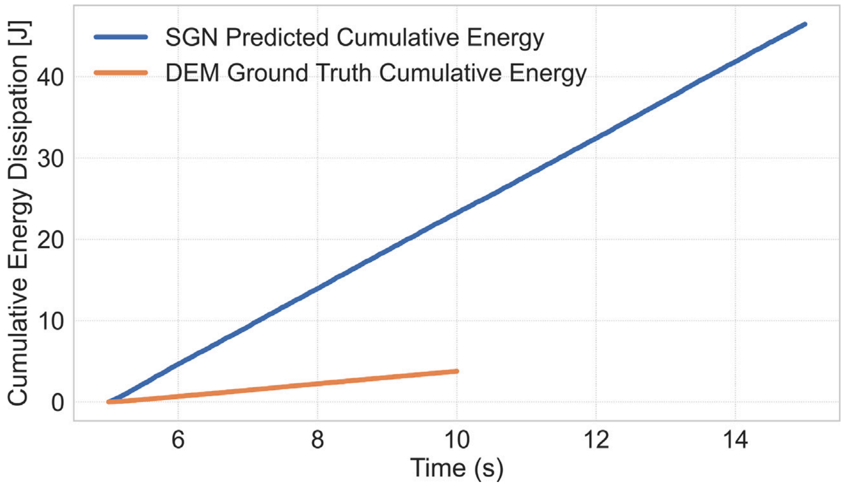


Fig. A.9. Ablation of the global loss $\mathcal{L}_{\text{global}}$.

closer to zero, grazing contacts are minimized, and the active PW-edge set shrinks to only the most wall-proximal, high-intensity events. This stronger set inflates the predicted per-step dissipation, so the cumulative energy overshoots the DEM ground truth; in this case, the relative error reaches 15.45% (see Fig. A.7).

Conversely, when the PW threshold becomes more negative, more grazing interactions are detected, the active PW-edge set enlarges, and mean pooling over this larger, lower-intensity set reduces the predicted increment. It should be noted that deeper thresholds increase the risk of false positive detections. In this case, the prediction undershoots the DEM reference, with a relative error of 4.18% (Fig. A.8). For a safe rollout, we recommend selecting a threshold that recovers $\geq 90\%$ of DEM PW ground truth contacts.

Global-loss ablation ($\mathcal{L}_{global}$)

Finally, we perform a ablation directly from Eq. (28) by setting $\mathcal{L}_{\text{global}} = 0$, so the objective reduces to $\mathcal{L}_{\text{total}} = \alpha\,\mathcal{L}_{\text{node}}$ (all other settings in Table 3 unchanged). Without supervision on the global head, the predicted per-step dissipation becomes severely miscalibrated: the cumulative curve overshoots DEM with a relative error 514.35% at the end of the rollout, even though the kinematics remain stable (see Fig. A.9). This confirms that $\mathcal{L}_{\text{global}}$ is necessary to calibrate the magnitude of dissipation; otherwise, its scale is unconstrained and drifts upward.

Data availability

The data and code will be made available in the link mentioned in the paper.

References

[1] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, P.W. Battaglia, Learning to simulate complex physics with graph networks, 2020, arXiv:2002.09405. URL https://arxiv.org/abs/2002.09405.

[2] S. Li, M. Sakai, Advanced graph neural network-based surrogate model for granular flows in arbitrarily shaped domains, Chem. Eng. J. 500 (2024) 157349, http://dx.doi.org/10.1016/j.cej.2024.157349, URL https://www.sciencedirect.com/science/article/pii/S1385894724088405.

[3] Y. Zhao, H. Li, H. Zhou, H.R. Attar, T. Pfaff, N. Li, A review of graph neural network applications in mechanics-related domains, 2024, arXiv:2407.11060. URL https://arxiv.org/abs/2407.11060.

[4] S. Garrido Nuñez, D.L. Schott, J.T. Padding, Predictive models for energy dissipation in mechanochemical ball milling, Powder Technol. 457 (2025) 120919, http://dx.doi.org/10.1016/j.powtec.2025.120919, URL https://www.sciencedirect.com/science/article/pii/S0032591025003146.

[5] C. Burmeister, M. Hofer, P. Molaiyan, P. Michalowski, A. Kwade, Characterization of stressing conditions in a high energy ball mill by discrete element simulations, Processes 10 (2022) 692, http://dx.doi.org/10.3390/pr10040692.

[6] S. Garrido Nuñez, D.L. Schott, J.T. Padding, Linking mechanics and chemistry: machine learning for yield prediction in NaBH4 mechanochemical regeneration, RSC Mechanochem. (2025) http://dx.doi.org/10.1039/D5MR00076A.

[7] W. Chen, M. Schoenitz, T. Ward, R. Dave, E. Dreizin, Numerical simulation of mechanical alloying in a shaker mill by discrete element method, KONA Powder Part. J. 23 (2005) 152–162, http://dx.doi.org/10.14356/kona.2005018.

[8] A. Oliveira, V. Rodriguez, R. de Carvalho, M. Powell, L. Tavares, Mechanistic modeling and simulation of a batch vertical stirred mill, Miner. Eng. 156 (2020) 106487, http://dx.doi.org/10.1016/j.mineng.2020.106487, URL https://www.sciencedirect.com/science/article/pii/S0892687520303071.

[9] V.A. Rodriguez, L. Ribas, A. Kwade, L.M. Tavares, Mechanistic modeling and simulation of a wet planetary ball mill, Powder Technol. 429 (2023) 118901, http://dx.doi.org/10.1016/j.powtec.2023.118901, URL https://www.sciencedirect.com/science/article/pii/S003259102300685X.

[10] M. Sakai, How should the discrete element method be applied in industrial systems?: A review, KONA Powder Part. J. 33 (2016) 169–178, http://dx.doi.org/10.14356/kona.2016023.

[11] N. Govender, D.N. Wilke, C.-Y. Wu, R. Rajamani, J. Khinast, B.J. Glasser, Large-scale GPU based DEM modeling of mixing using irregularly shaped particles, Adv. Powder Technol. 29 (10) (2018) 2476–2490, http://dx.doi.org/10.1016/j.apt.2018.06.028, URL https://www.sciencedirect.com/science/article/pii/S0921883118303054.

[12] V. Martinez, T. Stolar, B. Karadeniz, I. Brekalo, K. Užarević, Advancing mechanochemical synthesis by combining milling with different energy sources, Nat. Rev. Chem. 7 (1) (2023) 51–65, http://dx.doi.org/10.1038/s41570-022-00442-1.

[13] J.-L. Do, T. Friščić, Mechanochemistry: A force of synthesis, ACS Central Sci. 3 (1) (2017) 13–19, http://dx.doi.org/10.1021/acscentsci.6b00277, PMID: 28149948 arXiv:https://doi.org/10.1021/acscentsci.6b00277.

[14] M. Fransen, A. Fürst, D. Tunuguntla, D.N. Wilke, J. Brandstetter, J. Ooi, et al., Towards scientific machine learning for granular material simulations: Challenges and opportunities, Arch. Comput. Methods Eng. (2025) http://dx.doi.org/10.1007/s11831-025-10322-8.

[15] Y. Choi, K. Kumar, Graph neural network-based surrogate model for granular flows, 2023, http://dx.doi.org/10.48550/arXiv.2305.05218, arXiv arXiv:2305.05218. URL https://arxiv.org/abs/2305.05218.

[16] Y. Jiang, X. Chen, coauthors, Integrating graph neural network-based surrogate modeling with inverse design for granular flows, Ind. Eng. Chem. Res. (2024) http://dx.doi.org/10.1021/acs.iecr.4c00692, URL https://pubs.acs.org/doi/10.1021/acs.iecr.4c00692.

[17] A. Mayr, S. Lehner, A. Mayrhofer, C. Kloss, S. Hochreiter, J. Brandstetter, Boundary graph neural networks for 3D simulations, 2023, arXiv:2106.11299. URL https://arxiv.org/abs/2106.11299.

[18] Z. Xie, X. Gu, Y. Shen, A machine learning study of predicting mixing and segregation behaviors in a bidisperse solid–liquid fluidized bed, Ind. Eng. Chem. Res. 61 (24) (2022) 8551–8565, http://dx.doi.org/10.1021/acs.iecr.2c00071.

[19] L. Lu, X. Gao, J.-F. Dietiker, M. Shahnam, W.A. Rogers, Machine learning accelerated discrete element modeling of granular flows, Chem. Eng. Sci. 245 (2021) 116832, http://dx.doi.org/10.1016/j.ces.2021.116832, URL https://www.sciencedirect.com/science/article/pii/S0009250921003973.

[20] X. Guo, C. Hu, Y. Dai, H. Xu, L. Zeng, Learning dense gas-solids flows with physics-encoded neural network model, Chem. Eng. J. 485 (2024) 150072, http://dx.doi.org/10.1016/j.cej.2024.150072, URL https://www.sciencedirect.com/science/article/pii/S1385894724015584.

[21] N. Kishida, H. Nakamura, S. Ohsaki, S. Watano, Development of ultra-fast computing method for powder mixing process, Chem. Eng. J. 475 (2023) 146166, http://dx.doi.org/10.1016/j.cej.2023.146166, URL https://www.sciencedirect.com/science/article/pii/S1385894723048970.

[22] L. Benvenuti, C. Kloss, S. Pirker, Identification of DEM simulation parameters by artificial neural networks and bulk experiments, Powder Technol. 291 (2016) 456–465, http://dx.doi.org/10.1016/j.powtec.2016.01.003, URL https://www.sciencedirect.com/science/article/pii/S003259101630002X.

[23] A. Hajisharifi, F. Romanò, M. Girfoglio, A. Beccari, D. Bonanni, G. Rozza, A non-intrusive data-driven reduced order model for parametrized CFD-DEM numerical simulations, J. Comput. Phys. 491 (2023) 112355, http://dx.doi.org/10.1016/j.jcp.2023.112355, URL https://www.sciencedirect.com/science/article/pii/S0021999123004503.

[24] P. Zhi, Y. Wu, Graph neural networks for accelerating the discrete element simulation of granular flow, in: A. Korobenko, M. Laforest, S. Prudhomme, R. Vaziri (Eds.), Proceedings of the 16th World Congress on Computational Mechanics (WCCM 2024) and 4th Pan American Congress on Computational Mechanics (PANACM 2024), Track: Data Science, Machine Learning and Artificial Intelligence, Scipedia, Vancouver, Canada, 2024, pp. 1–11, http://dx.doi.org/10.23967/wccm.2024.123, URL https://www.scipedia.com/public/Zhi_Wu_2024a.

[25] V. Sharma, O. Fink, Dynami-CAL GraphNet: A physics-informed graph neural network conserving linear and angular momentum for dynamical systems, 2025, arXiv preprint arXiv:2501.07373.

[26] Y. Li, J. Bao, T. Chen, A. Yu, R. Yang, Prediction of ball milling performance by a convolutional neural network model and transfer learning, Powder Technol. 403 (2022) 117409, http://dx.doi.org/10.1016/j.powtec.2022.117409.

[27] DCS Computing GmbH, JKU Linz and Sandia Corporation., Gran model hertz model. URL https://www.cfdem.com/media/DEM/docu/gran_model_hertz.html.

[28] Altair Engineering Inc., The Hertz-Mindlin (no slip) model. URL https://help.altair.com/EDEM/Creator/Physics/Base_Models/Hertz-Mindlin_(no_slip.htm).

[29] S. Garrido Nuñez, D.L. Schott, J.T. Padding, Optimization of operational parameters in the mechanochemical regeneration of sodium borohydride (NaBH4), Int. J. Hydrog. Energy 97 (2025) 640–648, http://dx.doi.org/10.1016/j.ijhydene.2024.11.360, URL https://www.sciencedirect.com/science/article/pii/S0360319924050511.

[30] Lucefin Group, X46Cr13 technical card. URL https://www.lucefin.com/wp-content/files_mf/1.4034a420c25.pdf.

[31] P. Santhanam, E. Dreizin, Predicting conditions for scaled-up manufacturing of materials prepared by ball milling, Powder Technol. 221 (2012) 403–411, http://dx.doi.org/10.1016/j.powtec.2012.01.037.

[32] S. Rosenkranz, S. Breitung-Faes, A. Kwade, Experimental investigations and modeling of the ball motion in planetary ball mills, Powder Technol. 212 (2011) 224–230, http://dx.doi.org/10.1016/j.powtec.2011.05.021.

[33] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI Open 1 (2020) 57–81, http://dx.doi.org/10.1016/j.aiopen.2021.01.001, URL https://www.sciencedirect.com/science/article/pii/S2666651021000012.

[34] J. Gilmer, S.S. Schoenholz, P.F. Riley, O. Vinyals, G.E. Dahl, Neural message passing for quantum chemistry, 2017, arXiv:1704.01212. URL https://arxiv.org/abs/1704.01212.

[35] Delft High Performance Computing Centre (DHPC), DelftBlue supercomputer (phase 2), 2024, https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2.