JUSTC

Article

http://justc.ustc.edu.cn

Received: May 06, 2025; Accepted: June 07, 2025

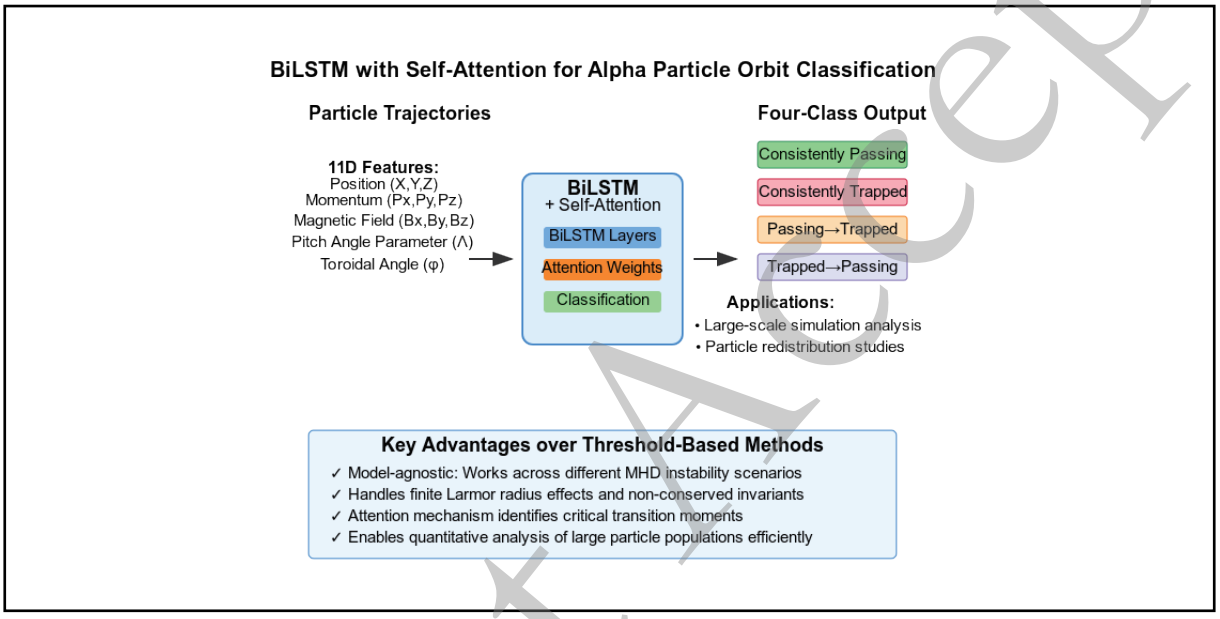
Classifying Alpha Particle Orbit Transitions in Tokamak Fusion Plasmas Using a BiLSTM with Self-attention Mechanism

Junyi Xu[1], Baofeng Gao[2], and Jian Liu[3] ✉

[1]*School of Nuclear Science and Technology, University of Science and Technology of China, Hefei, Anhui, 230026, China;*
[2]*Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250103, China;*
[3]*Weihai Institute for Interdisciplinary Research, Shandong University, 180 Culture West Road, Weihai, Shandong, 264209, China*

✉Correspondence: Jian Liu, E-mail: jliuphy@ustc.edu.cn
© 2025 The Author(s). This is an open access article under the CC BY-NC-ND 4.0 license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Graphical abstract



Public summary

- Developing a bidirectional LSTM neural network with self-attention mechanism for automatic classification of alpha particle orbit transitions in tokamak fusion plasmas. The system processes 11-dimensional time-series data to distinguish between four particle states, overcoming limitations of traditional threshold-based methods when finite Larmor radius effects become significant.

- Creating a machine learning framework that analyzes high-energy alpha particle behavior during magnetohydrodynamic instabilities by learning temporal patterns from multidimensional trajectory data. This approach enables efficient classification of billions of particles from small labeled datasets, providing new tools for understanding particle confinement and redistribution patterns.

- Implementing self-attention mechanism that automatically identifies critical moments in particle trajectories, offering interpretable insights into potential physical mechanisms driving particle state changes. The attention weights highlight key temporal regions that may correspond to important transitions between trapped and passing states in tokamak plasmas.

Citation: Xu J Y, Gao B F, Liu J. Classifying Alpha Particle Orbit Transitions in Tokamak Fusion Plasmas Using a BiLSTM with Self-attention Mechanism. *JUSTC*, **2025**, 55(0): . CSTR: 32290.14.JUSTC-2025-0054. DOI: 10.52396/JUSTC-2025-0054

Just Accepted

JUSTC

Article

http://justc.ustc.edu.cn

Received: May 06, 2025; Accepted: June 07, 2025

Classifying Alpha Particle Orbit Transitions in Tokamak Fusion Plasmas Using a BiLSTM with Self-attention Mechanism

Junyi Xu[1], Baofeng Gao[2], and Jian Liu[3] ✉

[1]*School of Nuclear Science and Technology, University of Science and Technology of China, Hefei, Anhui, 230026, China;*
[2]*Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250103, China;*
[3]*Weihai Institute for Interdisciplinary Research, Shandong University, 180 Culture West Road, Weihai, Shandong, 264209, China*

✉Correspondence: Jian Liu, E-mail: jliuphy@ustc.edu.cn
© 2025 The Author(s). This is an open access article under the CC BY-NC-ND 4.0 license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Cite This: *JUSTC*, **2025**, 55(X): (13pp) Read Online SI Supporting Information

Abstract: In tokamak plasmas, particles undergo transitions between trapped and passing states under magnetohydrodynamic instabilities. We developed a four-class bidirectional LSTM neural network with self-attention that categorizes particles as consistently passing, consistently trapped, passing-to-trapped, or trapped-to-passing. Using 11-dimensional time-series data (position, momentum, magnetic field, pitch angle parameter, and toroidal angle), the model generates attention weights that highlight critical moments in particle trajectories. Trained on a small labeled subset, our classifier efficiently scales to large-scale simulations, supporting detailed analysis of transition dynamics. We demonstrate this approach using alpha particles with significant finite Larmor radius effects, and the methodology can be adapted to different instability scenarios through retraining.

Keywords: burning plasma; trapped-passing transitions; magnetohydrodynamic instabilities; LSTM; attention mechanism

CLC number: **Document code:** A

1 Introduction

Transitional particles, which undergo changes between trapped and passing states, play a crucial role in particle transport and spatial redistribution in tokamak plasmas. These transitions directly influence the overall particle distribution patterns, as trapped (banana) orbits and passing orbits have fundamentally different spatial characteristics, particularly in their high-field side (HFS) and low-field side (LFS) distributions. Previous studies[1-5] have established the importance of these transitional particles, with research by[6, 7] demonstrating that passing and trapped energetic ions influence magnetic hydrodynamic instabilities differently. Accurate classification of these particle states and transitions is therefore crucial for understanding their collective behavior and impact on plasma dynamics, requiring quantitative analysis of transition statistics to determine how many particles undergo each type of transition and how these transitions contribute to particle redistribution between HFS and LFS regions.

Threshold-based classification methods face fundamental limitations when finite Larmor radius effects and steep electromagnetic field gradients from MHD instabilities become significant. In static magnetic fields, where particle motion is constrained by invariants (magnetic moment μ and kenetic energy E_k), the pitch angle parameter

$$\Lambda = \frac{\mu B_0}{E_k} = \frac{mv_\perp^2/2}{E_k} \cdot \frac{B_0}{B} \tag{1}$$

serves as an effective criterion[8, 9] to distinguish trapped particles[10-12] from passing particles, where $v_\perp$ are the velocity components perpendicular to the magnetic field direction, the constant B_0 is the reference magnetic field strength (in tokamaks typically taken as the magnetic field at the axis). Trapped particles are characterized by:

$$\Lambda > \frac{1}{B_m/B_0}, \tag{2}$$

where B_m is the maximum magnetic field strength along the guiding center magnetic field line. For example, numerical simulations reveal that alpha particles[13-15] with significant finite Larmor radius effects during Double Tearing Mode (DTM) instabilities[16] exhibit complex transitional dynamics between trapped (banana) and passing states that cannot be accurately captured by simple threshold criteria. The complex dynamics invalidate the underlying assumptions: magnetic moment μ is no longer conserved, kinetic energy E_k varies, and the reference magnetic field line for determining B_m becomes ill-defined due to large gyroradius excursions comparable to the characteristic scales of magnetic perturbations.

Machine learning techniques are increasingly being integrated into plasma physics research[17-19], offering powerful tools for pattern recognition in complex systems. To address the fundamental limitations of threshold-based methods, we develop a universal, data-driven approach that circumvents threshold ambiguity entirely. Our methodology learns to identify transition patterns directly from well-defined physical quantities without relying on assumption-based criteria. By

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

incorporating these fundamental physical quantities into a bi-directional Long Short-Term Memory (LSTM) neural network with self-attention mechanism[20-27], we successfully trained a four-class classifier capable of distinguishing between different particle state types: consistently passing, consistently trapped, passing-to-trapped, and trapped-to-passing. This automated pipeline enables quantitative analysis of transition statistics in large-scale simulations: automatically determining how many particles undergo each type of transition (passing-to-trapped vs trapped-to-passing), and systematically evaluating how these transitions contribute to particle redistribution between HFS and LFS regions. This data-driven approach provides a robust tool for understanding the collective impact on plasma confinement.

The remainder of this manuscript is structured as follows: Section 2 details our simulation framework and the architecture of our LSTM-attention neural network, including simulation parameters and machine learning methodology. Section 3 presents our analysis of alpha particle transition dynamics, featuring detailed case studies of both passing-to-trapped and trapped-to-passing transitions, and demonstrates the correlation between attention weights and critical events. Section 4 concludes with a discussion of our neural network methodology and outlines directions for future large-scale simulations of alpha particle dynamics in tokamak plasmas.

2 Methodology

2.1 Simulation setup and particle tracking

Our study employs a test particle approach to investigate alpha particle dynamics in electromagnetic fields obtained from MHD simulations. Since 3.5 MeV alpha particles have much higher energy than the background plasma and operate on much shorter characteristic timescales, they cannot be effectively modeled within the MHD framework. Additionally, as alpha particles are far fewer in number compared to the background plasma, the test particle approach is appropriate.

The temporal evolution of magnetic and electric fields is calculated using the M3D code, which solves the resistive MHD equations. For simplicity, the simulation is performed with low beta and uniform background plasma density, using a q-profile with weak negative shear as illustrated in Fig. 1. While tearing modes themselves originate from resistivity, the evolution of tearing modes spans 0-4000 Alfvén times. Within the brief 200 Alfvén time window, the collision effects between background plasma and α particles remain negligible, with electromagnetic forces dominating. We exclusively simulate α particle dynamics during the saturation phase of tearing modes between 2000-2200 Alfvén times. As shown in Figs. 2 and 3, the Poincaré plots of magnetic field lines at the $\phi = 0$ poloidal cross-section at 2000 and 2200 Alfvén times provide a visualization of the magnetic island structures during this phase. The interaction between the two islands can contribute to their mutual growth and energy release.

The simulations are performed using ITER-like parameters: major radius $R_0 = 6.2$ m, minor radius $a = 2.0$ m, elongation $\kappa = 2.0$, and triangularity $\delta = 0.5$. The magnetic field strength

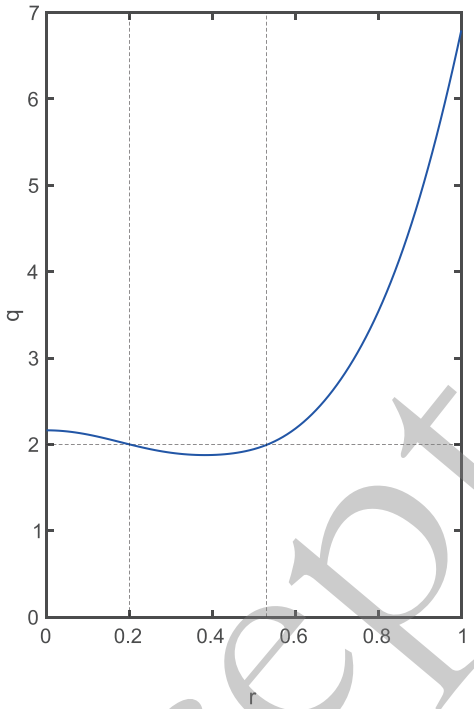


Fig. 1. Safety factor profile with weak negative shear. r is the normalized poloidal magnetic flux.

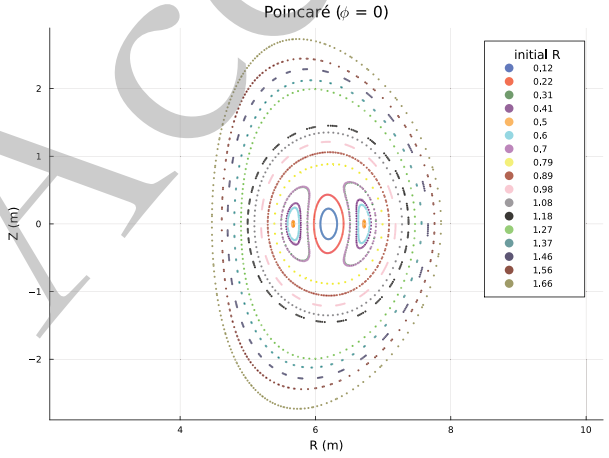


Fig. 2. Poincaré plot of DTM magnetic field lines at $\phi = 0$ cross-section, $t = 2000\tau_A$.

at the axis is $B_0 = 5.18$ T, resulting in an Alfvén velocity of $v_A = 9.76 \times 10^6$ m/s and Alfvén time $\tau_A = 6.35 \times 10^{-7}$ s. For 3.5 MeV alpha particles, the cyclotron frequency is $\omega_c = 2.5 \times 10^8$ s^{-1}.

The electromagnetic field evolution is simulated over 4000 Alfvén times. For our analysis, we select a representative time window: 2000-2200 Alfvén times corresponding to the saturated perturbation phase, as the tearing mode reaches saturation around 2000 Alfvén times. The Double Tearing Mode instabilities are characterized by a perturbation magnetic field amplitude of $\sim 10^{-5}$ T and operate in the low-frequency regime ($\omega = 0$ in the plasma frame). These parameters are typical for resistive MHD instabilities in ITER-like conditions, where the perturbation remains small compared to the equilibrium field ($\delta B/B_0 \sim 10^{-6}$) while still significantly affecting

JUSTC Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

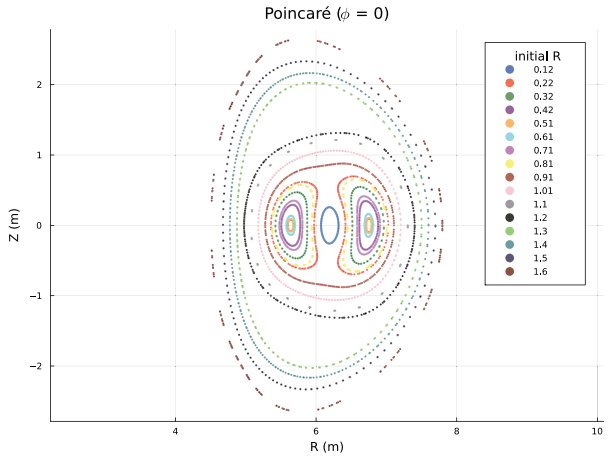


Fig. 3. Poincaré plot of DTM magnetic field lines at $\phi = 0$ cross-section, $t = 2200\tau_A$.

particle trajectories. Initially, the electric field perturbation is zero, with its evolution tied to the development of the tearing mode (see Eq. (A2)).

The initial alpha particle distribution is designed to reflect realistic fusion plasma conditions. Spatially, particles are distributed following a parabolic profile with respect to the normalized poloidal magnetic flux ψ, defined by the density function $\rho(\psi) \propto (1-\psi)^2$. This profile accounts for the higher alpha particle production rate in denser plasma regions near the magnetic axis. The distribution spans the entire tokamak volume, with uniform sampling across all toroidal angles $\phi \in [0, 2\pi)$. All alpha particles are initialized with an energy of 3.5 MeV, typical of fusion-born alpha particles. While the energy is uniform, the momentum directions are randomly distributed, ensuring isotropic coverage of velocity space.

Particle trajectories are computed using the Accurate Particle Tracker (APT)[28] with a time step of 0.1 gyroperiod, ensuring accurate resolution of particle motion even in regions of steep field gradients. The electromagnetic field data is updated every 10 Alfvén times during particle pushing. We employ full-orbit particle tracking rather than guiding-center approximations because the latter assumes $\rho_L \ll L_{\text{system}}$, where ρ_L is the Larmor radius and L_{system} represents characteristic system length scales. For 3.5 MeV alpha particles in ITER-like parameters, $\rho_L \approx 5$ cm, which becomes comparable to the width of magnetic islands and steep gradient regions, violating the guiding-center assumption. The simulation results have been cross-verified using an independent full-orbit particle tracer, ParticleTracker.jl[29], confirming consistent magnetic moment variations. This level of μ variation is expected in full-orbit simulations of energetic particles with significant finite Larmor radius effects, as previously observed in runaway electron studies[30].

2.2 Neural network architecture: BiLSTM with self-attention

For the neural network analysis, we sample 200 equally spaced points from each trajectory, providing sufficient temporal resolution to capture the transition dynamics while maintaining computational efficiency. Each sampled state is characterized by an 11-dimensional vector consisting of: position coordinates ($\mathbf{X}$), momentum ($\mathbf{P}$), local magnetic field ($\mathbf{B}$), pitch angle parameter (Λ), and toroidal angle (ϕ). The collection of these 200 samples forms what we call a particle trajectory, resulting in a data structure with shape $(200, 11)$. This representation approach borrows from the concept of word embeddings[31, 32], aiming to capture the contextual clues of causally-related time series for particles.

Although the pitch angle parameter can be derived from other inputs, we deliberately include it as a separate feature in our input vector for several important reasons. In static magnetic fields, Λ serves as an invariant that directly reflects particle motion characteristics, making it a physically significant feature. By explicitly including this input, we provide the network with a feature that has clear physical interpretation under equilibrium conditions. This approach is particularly beneficial for the unsupervised learning of attention weights, which helps identify critical transition moments in particle trajectories.

In electromagnetic perturbation environments, the physical behavior becomes significantly more complex and can't be described by simple mathematical formulas. The pitch angle parameter undergoes substantial oscillations. Our research aims to capture these complex relationships that are difficult to express using analytical methods. By including Λ alongside its constituent variables, we help the neural network better understand particle dynamics under perturbation. One purpose of using interdependent data is to encourage the neural network to learn non-linear feature dependencies in its hidden layers. LSTM networks can learn these temporal relationships, while cross-entropy loss[33] helps identify complex patterns across multiple time series simultaneously. This enables the model to detect subtle correlations between different inputs that collectively determine transition outcomes.

From a deep learning practice perspective, providing redundant information to neural networks can be beneficial for classification tasks. Different representations of related information can help the network build more robust internal representations.

2.2.1 LSTM Core Design

The choice of bidirectional LSTM[23] with self-attention mechanism is motivated by the nature of our trajectory classification task. Unlike sequential prediction tasks where models generate outputs step-by-step, our goal is to classify each complete 200-timestep trajectory into one of four categories based on the particle's behavior across the entire sequence. In addition, the bidirectional architecture is particularly advantageous for scalability to larger datasets, allowing future expansion of the training set to further improve network accuracy. Convolutional Neural Networks (CNNs), while effective for spatial pattern recognition, are fundamentally limited for this application due to several factors: (1) CNNs primarily capture local temporal patterns within fixed receptive fields, struggling with long-term dependencies across 200-timestep sequences spanning multiple orbit periods; (2) CNNs are inherently order-agnostic, whereas temporal sequence is critical for distinguishing passing-to-trapped from trapped-to-passing transitions; (3) For trajectory classification tasks requiring global temporal context, CNNs cannot effectively integrate information across the entire sequence.

JUSTC Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

Our architecture detailed in Fig. 4 enables comprehensive analysis of trajectories from both chronological perspectives, crucial for elucidating the causal mechanisms underlying state transitions. The network processes time-series data comprising 200 temporal steps, with each step encoding the complete 11-dimensional state vector.

2.2.2 Self-Attention Mechanism

In addressing the classification of alpha particle orbits, we utilize the entire trajectory's time-series information but recognize that not all temporal moments contribute equally to transition dynamics. Rather than treating each time step with equal importance, we incorporate a self-attention mechanism[21, 34], that learns to identify and weight the critical moments determining particle state transitions.

The self-attention mechanism allows the model to identify and weight the relative importance of different time segments in determining transition outcomes. The additive attention module computes weights α_t for each time step t according to:

$$\alpha_t = \text{softmax}(W_2 \tanh(W_1 h_t + b_1) + b_2) \tag{3}$$

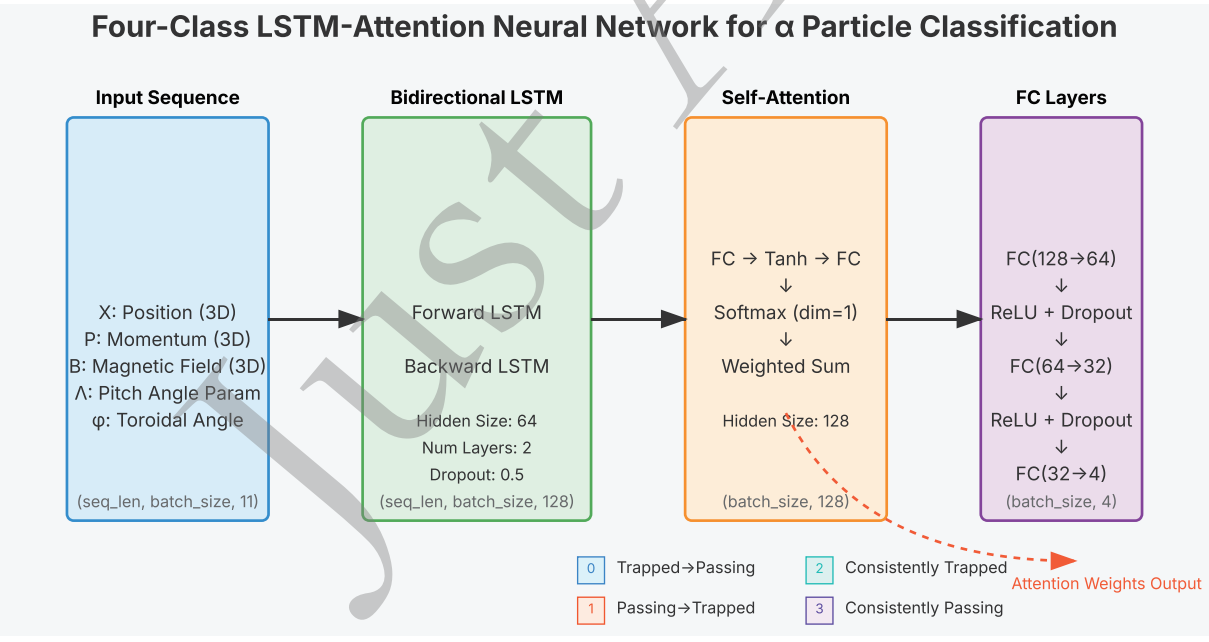
where h_t represents the hidden state at time t, and W_1, W_2, b_1, b_2 are learnable parameters. This mechanism proves particularly valuable in identifying critical moments in particle trajectories that determine transition outcomes, assigning higher weights to physically significant temporal features.

This mechanism dynamically weights different temporal segments of particle trajectories based on their relevance to the classification task, creating a more flexible and interpretable representation. Unlike standard LSTMs that rely solely on sequential processing, this additive attention approach enables the model to directly learn correlations between arbitrary time steps, regardless of their temporal distance.

The attention weights revealed in our analysis are not directly optimized through supervised learning, but rather emerge as an indirect consequence of the classification task. While training the neural network to distinguish between four distinct classes (consistently passing, consistently trapped, passing-to-trapped, and trapped-to-passing states), the model implicitly learns to identify the most informative temporal regions for making these classifications. This unsupervised emergence of attention distributions provides valuable physical insights, as the network independently discovers temporal correlations without being explicitly instructed to do so. The attention mechanism thus serves as an interpretable window into the model's decision-making process, revealing which aspects of particle trajectories are most relevant for determining their eventual fate.

2.3 Training Methodology

The dataset comprises trajectories of 5 million particles simulated using APT[35], from which a subset was carefully selected for manual labeling through phase space visualization. This labeled subset, consisting of several hundred trajectories, was partitioned into training (70%), validation (15%), and test (15%) sets. Each trajectory was classified as passing particle (label 3), trapped particle (label 2), passing-to-trapped particle (label 1) or trapped-to-banana particle (label 0). The neural network model was optimized using cross-entropy loss with an Adam optimizer[36] initialized at a learning rate of 5×10^{-4} and weight decay of 10^{-5}. To enhance convergence characteristics and prevent overfitting, we implemented: (1) early stopping with a patience threshold of 20 epochs based on valida-



The model processes multidimensional time-series data of particle trajectories through several components: (1) Input layer accepting position (**X**), momentum (**P**), magnetic field (**B**), pitch angle parameter (Λ), and toroidal angle (ϕ); (2) Bidirectional LSTM with 2 layers and 32 hidden units per direction to capture temporal dependencies; (3) Self-attention mechanism that identifies critical time steps for transition determination; (4) Fully connected layers with ReLU activations and dropout.

Fig. 4. Neural network architecture for α particle transition classification.

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

tion accuracy; (2) gradient norm clipping[37] with a maximum norm of 1.0 to mitigate gradient explosion; (3) dropout regularization with a rate of 0.5 applied between LSTM layers and within fully connected layers.

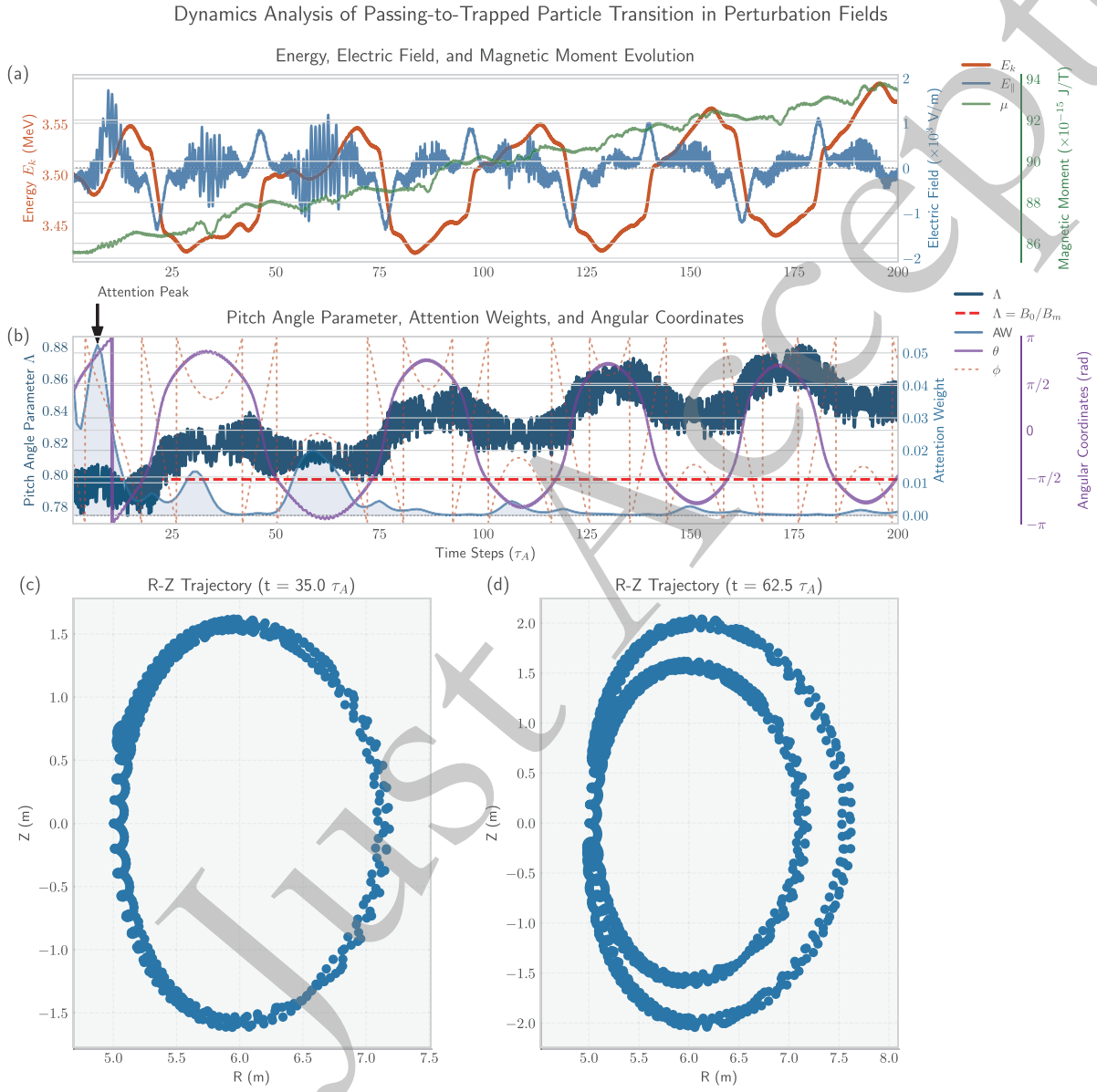
Post-training analysis focuses on two key aspects: First, the classification accuracy, and second, the interpretability of attention mechanisms.

3 Results

3.1 Particle Dynamics Analysis

The pitch angle parameter Λ is defined in Eq. (1) under static magnetic field conditions, where Λ is an invariant. Passing particles ($\Lambda < B_0/B_m$) are characterized by connected trajectories on the left side with near-elliptical poloidal orbits, as shown in Fig. 5(c) before 35 Alfvén times. Trapped particles ($\Lambda > B_0/B_m$), conversely, undergo "reflection" as demonstrated in Fig. 6(c) before 170 Alfvén times.

If a particle experiences a strong time-varying electric field before reaching the tip of a banana orbit (reflection point) and gains sufficient parallel acceleration before reaching the reflection point, then this particle will no longer reflect at its previous position. Readers can understand this using a magnetic mirror as an example: if a particle suddenly gains paral-

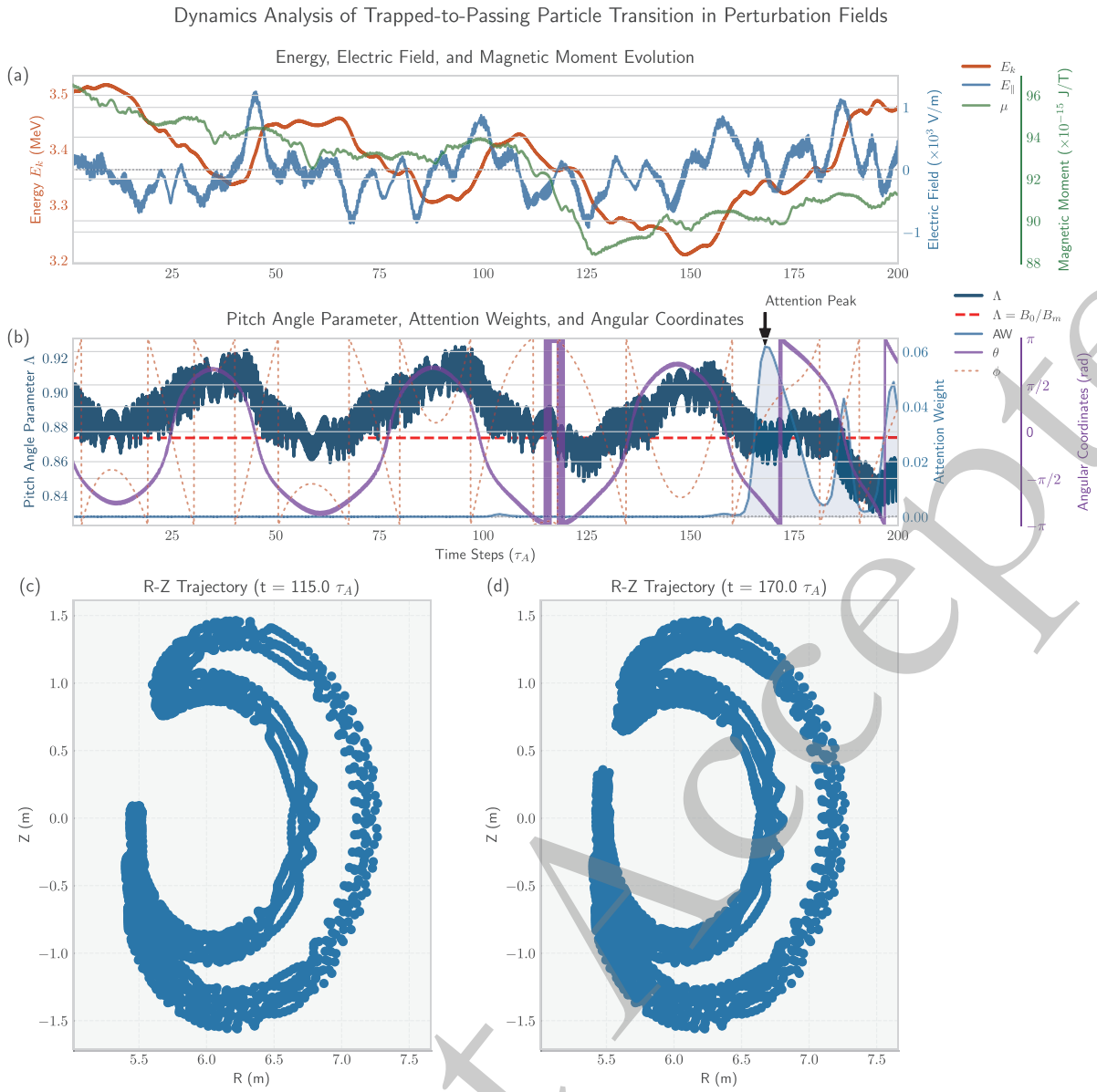


(a) Parallel electric field $E_\parallel$ (light blue), particle kinetic energy E_k (tan), and magnetic moment μ (green). (b) Pitch angle parameter $\Lambda = \mu B_0/E_k$ (blue), angular coordinates θ and φ (purple and red), and attention weights (light blue shading). Attention peaks coincide with orbit transitions: initial passing state, first reflection at approximately $35\tau_A$, and second reflection at $\sim 62.5\tau_A$. A small spike occurs when Λ exceeds the threshold at $\sim 20\tau_A$. (c) $R-Z$ trajectory showing counterclockwise motion until first reflection at $35\tau_A$. (d) After the first reflection, the particle completes the outer side of the banana orbit in a clockwise direction. At $\sim 62.5\tau_A$, it undergoes a second reflection at the lower "tip" of the banana orbit.

Fig. 5. Dynamics analysis of passing-to-trapped state transition for a 3.5 MeV alpha particle under electromagnetic perturbations.

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):



The representation of physical quantities is similar to that in Fig. 5. Initially, the pitch angle parameter $\Lambda > B_0/B_m$, indicating a trapped particle. The attention weights (blue shading) accurately mark the particle's transition points ($t = 170\tau_A$), rather than when the poloidal angle (purple) temporarily stays at π and $-\pi$ at $t = 115\tau_A$. The poloidal angle θ ranges from $-\pi$ to π, with π and $-\pi$ representing the same physical position in the poloidal plane.

Fig. 6. Dynamics analysis of trapped-to-passing particle transition for a 3.5 MeV α particle under electromagnetic perturbations.

Just Accepted

lel velocity upon reaching the reflection point, it can pass through the original reflection point.

DTM accompany magnetic reconnection to form magnetic islands. According to Faraday's law of electromagnetic induction, the "breaking" and reconnection of magnetic field lines generate a strong electric field, with the electric field strength proportional to the magnetic reconnection rate. The peak electric field near magnetic islands is very large and will significantly affect particle motion. For a typical trapped particle with a banana orbit period of approximately 50 Alfvén times, if its reflection point is near a magnetic island, the electromagnetic field at the original reflection point position will be very different from what it experienced previously at the same location.

The simulations in this paper demonstrate that particles undergoing transitions all have initial pitch angle parameters Λ near the threshold. For particles with Λ already close to the threshold, under the influence of electric fields, trapped particles may be accelerated to sufficient parallel velocity to pass through reflection points and enter passing orbits. This is called de-trapping. Similarly, passing particles can also transform into trapped particles under electromagnetic field effects, meaning that kinetic energy E_k is not an invariant in these conditions, as shown in Fig. 5(a). Over 200 Alfvén times, particles complete approximately 5000 gyration cycles. On average, the magnetic moment μ changes only by 0.001% per gyration cycle. However, as alpha particles exhibit significant finite Larmor radius effects, the total magnetic moment fluctuation (green line) reaches 8% over the entire 200 Alfvén times as shown in Fig. 5(a) and Fig. 6(a). Since both μ and E_k

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

are not invariants, this leads to continuous oscillation in Λ.

Under perturbations, when the average value of Λ oscillations crosses the threshold value, particles have a high probability of undergoing state transitions if finite Larmor radius effects are negligible. Threshold-based classification relies on finding B_m along the magnetic field line passing through the particle's guiding center. However, when finite Larmor radius effects become significant (ρ_L comparable to DTM scale), the guiding center approximation breaks down, leading to non-conservation of magnetic moment μ. Meanwhile, this breakdown undermines the very foundation of threshold-based classification, as the reference magnetic field line becomes ill-defined, rendering the B_0/B_m criterion fundamentally ambiguous.

The magnitude of this effect is clearly demonstrated in our simulations: in Fig. 6(c), at $t = 115\tau_A$, the particle is located at $\theta \approx \pi$ (or $-\pi$), $Z \approx 0.0$ m, and exhibits a large gyroradius (evident from the width of the blue trajectory segments). This gyroradius is significantly larger than the typical assumption of small Larmor radius ($\rho_L \ll$ characteristic length of DTM). Such large gyroradius excursions make it physically meaningless to define a single "guiding center magnetic field line" for threshold determination, as the particle samples vastly different magnetic field regions during its gyromotion.

It should be noted that regardless of the validity of the guiding center approximation, the magnetic moment μ can always be unambiguously defined for particles in full-orbit simulations, and therefore the pitch angle parameter Λ remains well-defined, albeit continuously oscillating. The challenge lies not in calculating Λ, but in determining the appropriate threshold B_0/B_m for classification when the reference magnetic field line becomes ambiguous. In contrast, the neural network learns to identify transition patterns through supervised learning from the temporal evolution of these fundamental quantities via data-driven pattern recognition, without relying on any assumption-based threshold criteria.

To ensure accurate classification, our neural network inputs include position coordinates ($\mathbf{X}$), momentum ($\mathbf{P}$), magnetic field ($\mathbf{B}$), and toroidal angle (ϕ). We also excluded poloidal angle (θ) because DTM makes the magnetic axis position ambiguous, no longer precisely at ITER's nominal $R = 6.2$ m, creating calculation difficulties. However, we still used $R = 6.2$ m as the center when plotting the poloidal angle (purple) in Fig. 5(b) and Fig. 6(b), solely to help readers visualize the direction of particle motion in the R-Z plane.

Notably, particles like Fig. 5(c) and Fig. 6(c) with similar initial conditions may not experience transitions, indicating chaotic behavior due to DTM influence. This chaotic sensitivity fundamentally undermines simple threshold-based criteria, as transition outcomes cannot be determined by any single threshold crossing. Therefore, representing particles with 11-dimensional vectors is necessary to capture the full complexity of transitions in such a chaotic system.

It should be emphasized that our data-driven machine learning approach is designed to be model-agnostic — the neural network learns to identify transition patterns from temporal evolution of well-defined physical quantities, independent of the specific underlying wave-particle interaction mechanisms. This methodology can be broadly applied to different MHD instability scenarios without requiring detailed knowledge of perturbation characteristics, making it particularly valuable for large-scale simulation analysis where multiple instability modes may coexist.

3.2 Convenient Classification of Particle Transitions using the Neural Network

Having established the limitations of threshold-based methods in the presence of significant finite Larmor radius effects, we now demonstrate the effectiveness of our neural network approach for particle transition classification. Particle state transitions in perturbed fields are fundamentally pathway-dependent rather than state-dependent, requiring analysis of the complete temporal evolution rather than instantaneous state evaluation. Our bidirectional Long Short-Term Memory (LSTM) layers with a self-attention mechanism effectively process multidimensional time-series data of particle trajectories, learning to recognize complex patterns that indicate state transitions through the temporal dynamics of fundamental physical quantities. Training this neural network is computationally efficient (only minutes on a PC with no GPU), achieving a final loss of 0.1 after 40 epochs.

In terms of accuracy, our model achieved 99% classification accuracy on an expanded test set of 200 particle trajectories (50 per class), with only 2 misclassifications. Both errors involved trapped-to-passing transition particles that were incorrectly classified as consistently trapped particles, occurring when the transition happened near the end of the 200 Alfvén time observation window, making it difficult for the classifier to distinguish between particles that underwent late-stage transitions versus those that were consistently in passing states throughout the entire period. Despite these minor errors, our 99% accuracy represents a dramatic improvement over threshold-based classification methods, which perform poorly in the presence of strong electromagnetic perturbations. The accuracy achieved here is sufficient for our primary objective: reliable statistical analysis of particle redistribution through accurate classification of transition proportions in large-scale simulations. This enables systematic quantification of how different transition types contribute to particle redistribution between high-field side and low-field side regions.

Although comprehensive evaluation of neural network generalization is not the central focus of this work, our classifier demonstrates robust capabilities across varied conditions. The classification architecture presented herein exhibits significant transferability — when trained on data from the 2000-2200 Alfvén time window, it effectively classifies particle transitions during the 3800-4000 Alfvén time window in ITER simulations, despite the substantial evolution of tearing modes between these periods. This suggests the network has captured fundamental physical distinctions between transition types rather than memorizing specific perturbation patterns. The strong generalization capabilities likely stem from our approach of processing discretely sampled trajectories that are generated by high-precision particle pusher simulations, rather than relying on an instantaneous state. Our classifier facilitates efficient analysis of large-scale simulations by enabling comprehensive classification of millions of particles

JUSTC ACCEPTED

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

after training on only a modest labeled subset.

3.3 Interpretability and Utility of Attention Weights

Beyond the accuracy, our LSTM-attention neural network provides valuable insights into the physical mechanisms driving particle transitions. The light blue shading in Figs. 5 and 6 illustrates the attention weight distributions, revealing which temporal regions most significantly influence transition outcomes.

Fig. 5 illustrates a passing-to-trapped transition for a 3.5 MeV alpha particle. The particle initially exists in a passing state with clockwise motion before experiencing its first reflection at approximately 35 Alfvén times with a big attention weight peak. Before 35 Alfvén times, magnetic moment increases with a dramatic reduction in total kinetic energy, so Λ has a climb and exceeds B_0/B_m. This may be the reason why the small attention weight spike occurs at this time. The subsequent bigger attention peak coincides with the particle's second reflection point at approximately 62.5 Alfvén times. After that, the neural network can make sure that the particle is trapped, so there are no more attention peaks. These temporal correlations between attention weights and critical physical events demonstrate that the neural network learns to identify the most relevant features for classification unsupervised.

Fig. 6 presents the trapped-to-passing transition for a 3.5 MeV α particle. Transition occurs late in the simulation and the biggest attention weight peak appears at approximately 170 Alfvén times when the particle passing the position $\theta = \pi$ in the R-Z plane. The poloidal angle θ temporarily stays at π and $-\pi$ at $t = 115\tau_A$ but attention weight is low before 170 Alfvén times.

By analyzing the temporal distribution of attention weights, we can identify which time periods in a particle's trajectory most strongly influence its classification outcome automatically. The interpretability provided by the attention mechanism further enhances the value of this approach, allowing us to focus on physical meaning of key events in the long evolution of particle trajectories.

Beyond providing physical insights, the attention weights offer practical value for large-scale simulation optimization. In billion-particle simulations, I/O operations represent the primary computational bottleneck, making it impractical to save particle data at every timestep. The attention mechanism can identify critical time periods in particle trajectories, enabling targeted high-resolution re-simulation of these specific time windows rather than uniformly detailed simulation across entire trajectories. This approach is particularly valuable given that APT simulations can handle billions of particles, but the associated I/O overhead necessitates strategic data saving strategies.

4 Conclusion and Future Work

Our straightforward, data-driven method analyzes millions of particles after training on only a small labeled subset through two key steps:

(1) **Dataset curation:** Manually label a minimal set of representative particle trajectories.

(2) **Feature extraction:** Selected transition-relevant physical quantities as inputs.

The trained model provides two primary capabilities:

(1) **Efficient classification:** Rapidly identifies particle types across numerous trajectories.

(2) **Physical insights:** Generates attention weights that highlight critical moments in transition dynamic.

Future work will focus on applying our trained classifier to large-scale simulations to quantify transition statistics: how many particles undergo different types of transitions, and how these transitions alter particle distribution between high-field side and low-field side regions. By conducting larger-scale simulations with increased particle counts, we will obtain precise measurements of transition-induced density redistribution patterns, ultimately enhancing our understanding of energetic particle dynamics in fusion plasmas.

Conflict of Interest

The authors declare that they have no conflict of interest.

References

[1] Harvey R W, Dendy R O. A trapped-passing fluid model for tokamak neoclassical transport. *Physics of Fluids B: Plasma Physics*, **1992**, *4* (4): 902–910.

[2] García-Muñoz M, Martin P, Fahrbach H U, et al. NTM induced fast ion losses in ASDEX Upgrade. *Nuclear Fusion*, **2007**, *47*: L10–L15.

[3] Brizard A J, Duthoit F X. Canonical transformation for trapped/passing guiding-center orbits in axisymmetric tokamak geometry. *Physics of Plasmas*, **2014**, *21*: 052509.

[4] Bland J, Varje J, Gorelenkov N N, et al. Interplay between beam-driven chirping modes and plasma confinement transitions in spherical tokamak ST40. *Nuclear Fusion*, **2023**, *63*: 016024.

[5] Beer M A, Hammett G W. Bounce averaged trapped electron fluid equations for plasma turbulence. *Physics of Plasmas*, **1996**, *3*: 4018–4022.

[6] Takahashi R, Brennan D P, Kim C C. Kinetic effects of energetic particles on resistive MHD stability. *Physical Review Letters*, **2009**, *102*: 135001.

[7] Cai H, Wang S, Xu Y, et al. Influence of energetic ions on tearing modes. *Physical Review Letters*, **2011**, *106*: 075002.

[8] jcandy. Magnetic confinement in a tokamak. https://physics.stackexchange.com/questions/411283/magnetic-confinement-in-a-tokamak, 2018.

[9] Allen R J. A demonstration of the magnetic mirror effect. *American Journal of Physics*, **1962**, *30*: 867–869.

[10] Goldston R J, White R B, Boozer A H. Confinement of high-energy trapped particles in tokamaks. *Physical Review Letters*, **1981**, *47*: 647–649.

[11] Graves J P, Hastie R J, Hopcraft K I. The effects of sheared toroidal plasma rotation on the internal kink. *Plasma Physics and Controlled Fusion*, **2000**, *42*: 1049.

[12] Chang C S. Neoclassical ion thermal conductivity modified by finite banana effects in a tokamak plasma. *Physics of Plasmas*, **1997**, *4*: 2241–2245.

[13] Garcia J, Kazakov Y, Coelho R, et al. Stable Deuterium-Tritium plasmas with improved confinement in the presence of energetic-ion instabilities. *Nature Communications*, **2024**, *15* (1): 7846.

[14] Kiptily V G, Challis C D, Dumont R, et al. Observation of alpha-particles in recent D–T experiments on JET. *Nuclear Fusion*, **2024**, *64*: 086059.

[15] Rimini F G, Contributors J, Team T E T E. 40 years of JET operations: A unique contribution to fusion science. *Plasma Physics*

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

and Controlled Fusion, **2025**, *67*: 033001.
[16] Ishii Y, Azumi M, Kurita G, et al. Nonlinear evolution of double tearing modes. *Physics of Plasmas*, **2000**, *7*: 4477–4491.
[17] Kates-Harbeck J, Svyatkovskiy A, Tang W. Predicting disruptive instabilities in controlled fusion plasmas through deep learning. *Nature*, **2019**, *568*: 526–531.
[18] Seo J, Kim S, Jalalvand A, et al. Avoiding fusion plasma tearing instability with deep reinforcement learning. *Nature*, **2024**, *626*: 746–751.
[19] Degrave J, Felici F, Buchli J, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, **2022**, *602*: 414–419.
[20] Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Computation*, **1997**, *9*: 1735–1780.
[21] Cheng J, Dong L, Lapata M. Long short-term memory-networks for machine reading. arXiv: 1601.06733, **2016**.
[22] Gers F A, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM. *Neural Computation*, **2000**, *12*: 2451–2471.
[23] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM networks. In: *Proceedings of 2005 IEEE International Joint Conference on Neural Networks, 2005*. IEEE, **2005**: 2047–2052.
[24] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv: 1409.0473, **2014**.
[25] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Guyon I, Von Luxburg U, Bengio S, et al. editors. Advances in Neural Information Processing Systems, New York: Curran Associates, Inc., **2017:** 30.
[26] Sanchez-Gonzalez A, Godwin J, Pfaff T, et al. Learning to simulate complex physics with graph networks. In: Proceedings of the 37th International Conference on Machine Learning, New York: ACM, **2020**: 8459–8468.
[27] Pfaff T, Fortunato M, Sanchez-Gonzalez A, et al. Learning mesh-based simulation with graph networks. arXiv: 2010.03409, **2020**.
[28] Wang Y, Liu J, Qin H, et al. The accurate particle tracer code. *Computer Physics Communications*, **2017**, *220*: 212–229.
[29] Junyi Xu. AccurateTracker.jl: A Julia package for particle tracking in electromagnetic fields, 2024. GitHub repository.
[30] Wang Y, Qin H, Liu J. Multi-scale full-orbit analysis on phase-space behavior of runaway electrons in tokamak fields with synchrotron radiation. *Physics of Plasmas*, **2016**, *23*: 062505.
[31] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model. *Journal of machine learning research*, **2003**, *3*: 1137–1155.
[32] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space. arXiv: 1301.3781, **2013.**
[33] Shannon C E. A mathematical theory of communication. *The Bell System Technical Journal*, **1948**, *27*: 379–423.
[34] Lin Z, Feng M, Nogueira dos Santos C, et al. A structured self-attentive sentence embedding. arXiv: 1703.03130, **2017**.
[35] Wang Y, Liu J, Qin H, et al. The accurate particle tracer code. *Computer Physics Communications*, **2017**, *220*: 212–229.
[36] Kingma D P, Ba J. Adam: A method for stochastic optimization. arXiv: 1412.6980, **2014**.
[37] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. In: Proceedings of the 30th International Conference on Machine Learning, PMLR, **2013**, 28(3): 1310-1318.
[38] Zhang R, Liu J, Qin H, et al. Volume-preserving algorithm for secular relativistic dynamics of charged particles. *Physics of Plasmas*, **2015**, *22* (4): 044501.

A Appendix: Particle Dynamics and MHD Equations

A.1 Particle Dynamics Equations

The equations of motion for high-energy charged particles in electromagnetic fields are governed by the Lorentz force:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\mathbf{p}}{m_0\gamma}, \tag{A1a}$$

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = q\left(\mathbf{E} + \frac{\mathbf{p}\times\mathbf{B}}{m_0\gamma}\right), \tag{A1b}$$

where m_0 and q represent the particle's rest mass and charge, respectively, $\mathbf{B}$ and $\mathbf{E}$ represent the magnetic and electric fields, $\mathbf{x}$ and $\mathbf{p}$ represent the position and momentum vectors, and $\gamma = \sqrt{1/(1-v^2/c^2)}$ is the Lorentz factor with c being the speed of light in vacuum. For numerical integration of these equations, we employed a relativistic volume-preserving algorithm[38].

A.2 MHD Equations for M3D Code

Generalized Ohm's Law In resistive MHD, we retain only the resistive term:

$$\mathbf{E} + \mathbf{u}\times\mathbf{B} = \eta\mathbf{J}, \tag{A3}$$

where η is plasma resistivity.

Complete MHD Equation Set The MHD equations implemented in the M3D code are:

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{v}) = \nabla\cdot(D\nabla\rho), \tag{A3}$$

$$\rho\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \mathbf{J}\times\mathbf{B} - \nabla p + \nu\nabla^2\mathbf{v}, \tag{A4}$$

$$\frac{\mathrm{d}p}{\mathrm{d}t} = -\gamma p\nabla\cdot\mathbf{v} + \rho\nabla\cdot\boldsymbol{\kappa}\cdot\nabla\frac{p}{\rho}, \tag{A5}$$

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}, \tag{A6}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{A7}$$

B Normalized Perturbed Field in Simulations

In the M3D simulation code, the variables represented with a hat, $\hat{E}$, refer to normalized quantities. The velocity normalization is expressed as:

$$\mathbf{v} = \epsilon v_A \hat{\mathbf{v}} \tag{B8}$$

where $\epsilon = a/R_0 = 0.32$ is the aspect ratio, and the Alfvén velocity is $v_A = 9.76 \times 10^6\,\mathrm{m/s}$.

Similarly, the magnetic field is normalized as:

$$\mathbf{B} = B_0 \hat{\mathbf{B}} \tag{B9}$$

where $B_0 = 5.18\,\mathrm{T}$ serves as the reference magnetic field.

The electric field is normalized as:

$$\mathbf{E} = E_0 \hat{\mathbf{E}} = \epsilon v_A B_0 \hat{\mathbf{E}} \tag{B10}$$

C Deep Learning Architecture for Alpha Particle Transition Classification

This appendix details the implementation of our deep learning model for classifying alpha particle orbit transitions. The architecture combines bidirectional Long Short-Term Memory (LSTM) networks with a self-attention mechanism, enabling effective analysis of complex multidimensional time-series data from particle trajectories.

C.1 Data Preprocessing and Feature Engineering

Each particle trajectory is represented by a multidimensional time series spanning 200 time steps. The feature vector at each time step consists of:

- Position components ($\mathbf{X}$): Three-dimensional Cartesian coordinates (x, y, z)
- Momentum components ($\mathbf{P}$): Three-dimensional momentum vector (p_x, p_y, p_z)
- Magnetic field components ($\mathbf{B}$): Three-dimensional magnetic field vector (B_x, B_y, B_z)
- Pitch angle parameter (Λ): The ratio $\mu B_0/E_k$, where μ is the magnetic moment, B_0 is the reference magnetic field strength, and E_k is the kinetic energy
- Toroidal angle (ϕ): Calculated as arctan (y/x) to capture the particle's angular position in the tokamak

C.2 Neural Network Architecture

Our model architecture consists of three main components: a bidirectional LSTM encoder, a self-attention mechanism, and a fully connected classification network.

C.2.1 Bidirectional LSTM Layer

The bidirectional LSTM processes the input sequence in both forward and backward directions:

$$\mathbf{h}_t = \mathrm{BiLSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{C11}$$

where $\mathbf{x}_t$ represents the input feature vector at time step t, and $\mathbf{h}_t$ is the hidden state. We used a two-layer BiLSTM with 32 hidden units and dropout ($p = 0.5$) between layers to prevent overfitting. The bidirectional nature allows the model to capture both past and future contextual information for each time step.

C.2.2 Self-Attention Mechanism

The self-attention mechanism identifies the most relevant time steps for classification:

$$e_t = \mathbf{v}^T \tanh(\mathbf{W}\mathbf{h}_t + \mathbf{b}) \tag{C12}$$

$$\alpha_t = \frac{\exp(e_t)}{\sum_{i=1}^{T} \exp(e_i)} \tag{C13}$$

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

$$\mathbf{c} = \sum_{t=1}^{T} \alpha_t \mathbf{h}_t \tag{C14}$$

where $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{v}$ are learnable parameters. The attention weights α_t represent the importance of each time step, and $\mathbf{c}$ is the context vector that aggregates information across the entire sequence. This mechanism allows the model to focus on specific regions of the trajectory that are most informative for determining the particle classification outcome.

The implementation of the self-attention mechanism is shown below:

```
class SelfAttention(nn.Module):
    def __init__(self, hidden_size):
        super(SelfAttention, self).__init__()
        self.hidden_size = hidden_size
        self.attention = nn.Sequential(
            nn.Linear(hidden_size, hidden_size),
            nn.Tanh(),
            nn.Linear(hidden_size, 1)
        )
    def forward(self, encoder_outputs):
        # encoder_outputs: (batch_size, seq_len, hidden_size)
        batch_size, seq_len, _ = encoder_outputs.size()
        # (batch_size, seq_len, 1)
        attention = self.attention(encoder_outputs)
        # (batch_size, seq_len, 1)
        attention_weights = torch.softmax(attention, dim=1)
        # (batch_size, 1, seq_len) * (batch_size, seq_len, hidden_size)
        # -> (batch_size, 1, hidden_size)
        context = torch.bmm(
            attention_weights.transpose(1, 2),
            encoder_outputs
        )
        # (batch_size, hidden_size)
        return context.squeeze(1), attention_weights
```

C.2.3 Four-Class Classification Network

The classification network processes the context vector to produce predictions for four distinct classes:

- Class 0: Trapped→Passing transitions
- Class 1: Passing→Trapped transitions
- Class 2: Consistently trapped particles
- Class 3: Consistently passing particles

The network outputs class logits that are transformed into probabilities:

$$\hat{\mathbf{y}} = \text{softmax}(\text{MLP}(\mathbf{c})) \tag{C15}$$

where MLP represents a multi-layer perceptron with ReLU activations and dropout between layers, and softmax produces a probability distribution over the four classes. The complete model implementation is provided below:

```
class FourClassTransitionClassifier(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers, dropout=0.5):
        super(FourClassTransitionClassifier, self).__init__()
        # Bidirectional LSTM layers
        self.lstm = nn.LSTM(
            input_size=input_size,
            hidden_size=hidden_size,
            num_layers=num_layers,
            batch_first=True,
            bidirectional=True,
            dropout=dropout if num_layers > 1 else 0
        )
        # Attention layer
        self.attention = SelfAttention(hidden_size * 2)
        # Fully connected layers for classification
```

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

```
        self.fc = nn.Sequential(
            nn.Linear(hidden_size * 2, hidden_size),
            nn.ReLU(),
            nn.Dropout(dropout),
            nn.Linear(hidden_size, hidden_size // 2),
            nn.ReLU(),
            nn.Dropout(dropout),
            nn.Linear(hidden_size // 2, 4) # 4 classes output
        )
    def forward(self, x):
        # x: (batch_size, seq_len, input_size)
        # Pass through LSTM layers
        outputs, _ = self.lstm(x)
        # Apply attention mechanism
        context, attn_weights = self.attention(outputs)
        # Generate class logits through fully connected layers
        logits = self.fc(context)
        return logits, attn_weights
```

C.3 Training Procedure

The model was trained using the cross-entropy loss function for multi-class classification:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c}) \tag{C16}$$

where $y_{i,c}$ is the true label for sample i and class c (using one-hot encoding), $\hat{y}_{i,c}$ is the predicted probability for that class, N is the number of samples, and $C = 4$ is the number of classes.

We employed the Adam optimizer with an initial learning rate of 5×10^{-4} and weight decay of 10^{-5} for regularization. To prevent gradient explosion, gradient clipping was applied with a maximum norm of 1.0.

A learning rate scheduler was implemented to reduce the learning rate by a factor of 0.5 when validation accuracy plateaued for 10 consecutive epochs. Early stopping was employed with a patience of 20 epochs to prevent overfitting.

```
def train_model(model, train_loader, val_loader, criterion,
                optimizer, scheduler=None, num_epochs=100,
                patience=15, device='cuda'):
    model = model.to(device)
    history = {'train_loss': [], 'val_loss': [], 'val_acc': []}
    best_val_acc = 0
    no_improve_epochs = 0
    for epoch in range(num_epochs):
        # Training phase
        model.train()
        train_loss = 0
        for inputs, labels in train_loader:
            inputs, labels = inputs.to(device), labels.to(device)
            # Forward pass
            logits, _ = model(inputs)
            loss = criterion(logits, labels)
            # Backward pass and optimization
            optimizer.zero_grad()
            loss.backward()
            # Gradient clipping
            torch.nn.utils.clip_grad_norm_(
                model.parameters(),
                max_norm=1.0
            )
            optimizer.step()
            train_loss += loss.item()
        # Validation phase
        model.eval()
```

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):

```
val_loss = 0
correct = 0
total = 0
with torch.no_grad():
    for inputs, labels in val_loader:
        inputs, labels = inputs.to(device), labels.to(device)
        logits, _ = model(inputs)
        loss = criterion(logits, labels)
        val_loss += loss.item()
        _, predicted = torch.max(logits, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
# Calculate average loss and accuracy
train_loss = train_loss / len(train_loader)
val_loss = val_loss / len(val_loader)
val_acc = 100 * correct / total
# Update learning rate
if scheduler is not None:
    scheduler.step(val_acc)
# Update history records
history['train_loss'].append(train_loss)
history['val_loss'].append(val_loss)
history['val_acc'].append(val_acc)
```

Just Accepted

CSTR: 32290.14.JUSTC-2025-0054
DOI: 10.52396/JUSTC-2025-0054
JUSTC, **2025**, 55(X):