A Universal Model Combining Differential Equations and Neural Networks for Ball Trajectory Prediction

Zhiwei Shi[a,*], Chengxi Zhu[a], Fan Yang[b], Jun Yan[c], Zheyun Qin[a], Songquan Shi[d] and Zhumin Chen[a]

[a]School of Computer Science and Technology, Shandong University, 72 Binhai Highway, Qingdao, Shandong Province, Qingdao, 266000, Shandong, China
[b]DeepCode Robotics, No. 488 Hongxing Road, Xiaoshan District, Hangzhou, 310000, Zhejiang, China
[c]College of Electronic and Information Engineering, Tongji University, 4800 Cao'an Highway, Jiading District, Shanghai, China, 201804, Shanghai, China
[d]Shaoxing University, No. 1077, Chengnan Avenue, Shaoxing, 312000, Zhejiang, China

ARTICLE INFO

Keywords:
Universal Method
Physics and Data-Driven
Ball Trajectory Prediction
Differential Equations

ABSTRACT

This paper presents a data-driven universal ball trajectory prediction method integrated with physics equations. Existing methods are designed for specific ball types and struggle to generalize. This challenge arises from three key factors. First, learning-based models require large datasets but suffer from accuracy drops in unseen scenarios. Second, physics-based models rely on complex formulas and detailed inputs, yet accurately obtaining ball states, such as spin, is often impractical. Third, integrating physical principles with neural networks to achieve high accuracy, fast inference, and strong generalization remains difficult. To address these issues, we propose an innovative approach that incorporates physics-based equations and neural networks. We first derive three generalized physical formulas. Then, using a neural network and observed trajectory points, we infer certain parameters while fitting the remaining ones. These formulas enable precise trajectory prediction with minimal training data: only a few dozen samples. Extensive experiments demonstrate our method's superiority in generalization, real-time performance, and accuracy. The trajectories data is available on: https://github.com/any11name/trajectories

1. Introduction

The integration of technology into sports training has significantly improved performance analysis and skill development through wearable sensors [1, 2], vision-based analysis systems [3, 4, 5, 6], and robotic training systems [7, 8, 9]. However, existing trajectory prediction models are highly specialized, designed for specific sports and ball dynamics, which limits their adaptability across different scenarios. To address this limitation, we propose a universal ball trajectory prediction algorithm that integrates physical modeling with data-driven learning, enabling accurate predictions across diverse sports. This advancement is essential for developing more versatile and intelligent training systems, ultimately enhancing athlete performance across a wide range of disciplines.

Developing a universal ball trajectory prediction algorithm presents several challenges. First, learning-based methods require large amounts of training data, yet their prediction accuracy deteriorates significantly when applied to scenarios beyond the training distribution. Additionally, as shown in Table 1, the ball's high flight speed demands a model with rapid inference capabilities. Second, physics-based models rely on complex physical equations and detailed input parameters, but accurately obtaining crucial ball states, such as spin, is often difficult in real-world scenarios.

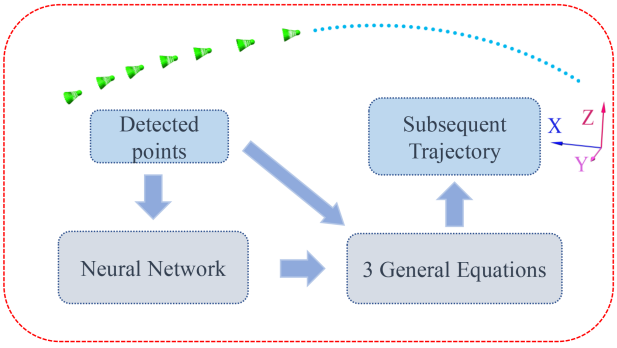


Figure 1: The Architecture of our Method. This data-driven universal ball trajectory prediction method primarily uses a neural network and three universal equations to predict the subsequent trajectory of the ball. X-Y-Z represents the world coordinate system defined by the user.

Third, integrating physics-based modeling with data-driven learning is inherently challenging, as the algorithm must achieve both high accuracy and computational efficiency to ensure real-time performance. Addressing these challenges is critical for developing a robust and versatile trajectory prediction system applicable across multiple sports.

Existing ball trajectory prediction algorithms can be broadly classified into three main categories: physics-based models, data-driven models, and hybrid models. Physics-based models leverage equations of motion and aerodynamic principles to predict trajectories [11, 12, 13]; however, they

*Corresponding author
202220775@mail.sdu.edu.cn (Z. Shi); 202235207@mail.sdu.edu.cn (C. Zhu); sail308@126.com (F. Yang); yanjun@ieee.org (J. Yan); zheyunqin@gmail.com (Z. Qin); sqshi@usx.edu.cn (S. Shi); chenzhumin@sdu.edu.cn (Z. Chen)
ORCID(s):

Table 1
Average travel time and speed of different balls [10]

Types	table tennis	badminton	tennis	Pickleball
Time (ms)	500	800	1500	1300
Speed (km/h)	40	80	70	50

struggle to account for complex effects such as spin and environmental variations. Data-driven models, including neural networks and regression-based approaches, learn trajectory patterns from empirical data [14, 15]. While these models offer adaptability, they often lack interpretability and generalization beyond the training distribution. Hybrid models integrate physics-based constraints with machine learning techniques to enhance both accuracy and generalization [16, 17]. However, their application to ball trajectory prediction remains largely unexplored. Despite these advancements, most existing methods are tailored to specific sports, underscoring the need for a universal trajectory prediction framework.

To address these challenges, we propose a universal trajectory prediction method, as illustrated in Fig. 1. Our approach is grounded in rigorous physical derivations, leading to the formulation of three generalized mathematical equations. By integrating these equations with a lightweight neural network, our method can effectively predict the trajectories of a wide range of ball types. This approach achieves high accuracy, strong generalization capability, and fast inference speed, making it well-suited for trajectory prediction across diverse scenarios.

Our contributions are summarized as follows:

- We develop a universal algorithm capable of accurately predicting ball trajectories across different sports.
- By integrating physical modeling (three equations) with a data-driven learning model (MLP), we achieve a balance between accuracy, adaptability, and efficiency, making our method highly versatile and capable of handling not only normal trajectories but also defective and spinning ball trajectories.
- Our approach achieves high prediction accuracy using only 90 training samples for each type of balls, demonstrating its data efficiency and generalization capability.
- Real-world experiments show that the accuracy of our method meets the requirements.

2. Related work

Ball trajectory prediction involves estimating the future motion of a ball based on an athlete's movements or the ball's current trajectory. In sports robotics, precise trajectory prediction is essential for enabling robots to successfully intercept or strike the ball. Existing trajectory prediction methods can be broadly classified into three categories: physics-based models [18, 9, 13], learning-based models [14, 15], and hybrid models [16, 17].

Physics-based models: Physics-based models primarily predict ball trajectories by applying aerodynamic equations to the ball's current state. These models have been widely used in various applications. Huang et al. [12] developed a physical model capable of accurately predicting the trajectory and bounce of spinning balls, laying a foundation for ball behavior analysis. Mehta [11] investigated the aerodynamic properties of sports balls in flight, providing critical insights into ball performance and contributing to the optimization of sports techniques. Additionally, Ji et al. [9] integrated aerodynamic modeling with clustering algorithms, significantly reducing trajectory prediction errors. These studies collectively underscore the importance of both theoretical modeling and computational approaches in advancing trajectory prediction research.

Learning-based models: Unlike physics-based models, learning-based approaches leverage not only existing trajectory data but also additional information, such as athlete movements [19, 3] and racket sensor data [20], to predict ball trajectories in advance. However, these methods typically require large amounts of data for training. Key applications include the following: Xiao et al. [15] introduced an innovative framework for real-time tennis ball trajectory prediction, utilizing a multi-camera setup and a factor graph model to achieve high-precision ball tracking and state estimation. Gomez et al. [14] proposed a deep learning-based trajectory prediction approach, employing an encoder-decoder network to accurately forecast the ball's future path. Additionally, Zhang et al. [3] developed an advanced sports analytics platform that enables early trajectory prediction by integrating motion sensor data with vision-based systems for shot type classification. Collectively, these studies highlight the versatility and effectiveness of learning-based methods in ball trajectory prediction.

Hybrid models: Currently, research on integrating physical models with neural networks for ball trajectory prediction remains limited, and practical applications are relatively scarce. While some progress has been made. Some relevant studies include Zhang et al. [16], who proposed a hybrid approach combining physics-based modeling with machine learning to improve motion prediction accuracy. Similarly, Moya et al. [17] explored the integration of computational mechanics and physics-informed machine learning to enhance the understanding of physical phenomena. These studies highlight the potential of combining domain knowledge with data-driven models to achieve more reliable predictions. However, further research is needed to refine these integration methods and explore their applications in real-world scenarios.

In conclusion, ball trajectory prediction remains a challenging task, as existing methods often struggle to meet practical requirements in terms of accuracy, computational efficiency, and adaptability.

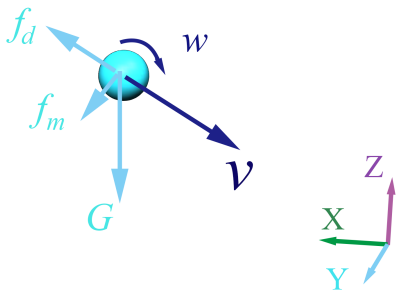


Figure 2: The force diagram of the ball during its flight. During its flight, the ball is primarily subjected to gravity (G), air resistance (f_d), the Magnus force caused by rotation (f_m), and lift force from the air, among others.

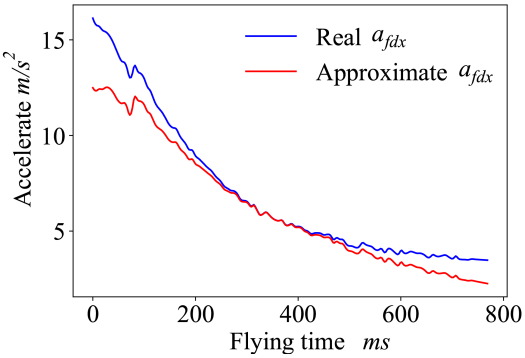


Figure 3: Comparison Between the Real Value and the Approximate Value of a_{fdx}. The horizontal axis represents the time elapsed since the shuttlecock was hit, and the vertical axis represents the value of a_{fdx} over time.

3. Method Design

In this section, we describe the overview of this universal solution for ball trajectory prediction method and give an explanation of each component.

3.1. Problem Analysis

Ball trajectory prediction relies on its current state or known trajectory points to forecast subsequent motion. However, accurately obtaining the ball's state—such as its velocity, acceleration, and particularly its spin—remains challenging. Moreover, the forces acting on the ball are highly complex. During flight, the ball is subjected not only to gravity (G) and aerodynamic drag (f_d) but also to the Magnus force (f_m) induced by spin, among other factors, as illustrated in Fig. 2. To address these challenges, we propose three generalized universal formulas integrated with a neural network, enabling accurate trajectory prediction while circumventing many complex computations and reasoning processes.

3.2. X-axis Direction

As shown in Fig. 2, the X-axis direction is the primary flight direction of the ball. Since the ball's velocity is high, the main force acting on it is air resistance, while the other forces are relatively small and can be temporarily ignored here. The formula for the acceleration of the ball caused by air resistance is Eq. (1) [11], where k is a constant that depends only on the type of ball and v is the velocity of the ball. The component of a_{fd} in the x-axis (a_{fdx}), is shown in Eq. (2), where v_x is the velocity of the ball in the X-axis. Relatively speaking, this formula is quite complex. Due to factors such as gravity, the flight angle of the ball changes continuously over time, making calculations difficult in such cases. To simplify this process, we reduce Eq. (2) to Eq. (3). Since the ball is mainly influenced by the resistance of air in the x-direction, the values of a_x and a_{fdx} are approximately equal. This simplification process does not significantly affect accuracy. Fig. 3 shows a comparison curve between the real a_{fdx} and the simplified a_{fdx} at different flight times for a badminton trajectory, from which it can be seen that the loss of accuracy is not substantial.

$$a_{fd} = k_x \cdot v^2 \tag{1}$$

$$a_{fdx} = -v_x/v \cdot k_x \cdot v^2 \tag{2}$$

$$a_x \approx a_{fdx} \approx k_x \cdot v_x^2 \tag{3}$$

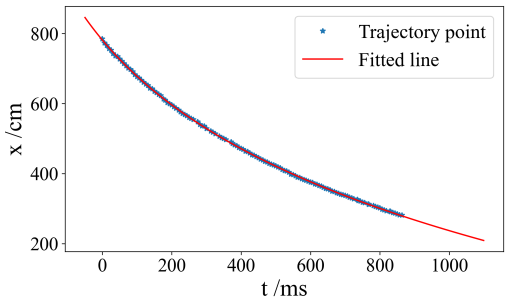
Eq. (3) is a typical second-order differential equation with respect to time (t) and can be expressed in the form of Eq. (4). So, the general solution of this second-order differential equation is Eq. (5), where c_{x2} and c_{x3} are arbitrary constants [21, 22]. This formula represents the relationship between the ball's coordinate on the X-axis and the flight time. In the previous reasoning process, we used a lot of approximations, which will lead to loss of accuracy. Therefore, we take $-1/k_x$ as an unknown quantity c_{x1} and obtain Eq. (6). In this way, we can obtain the values of c_{x1}, c_{x2} and c_{x3} by fitting the known points of the trajectory and predict the subsequent trajectory of the ball in the X-axis direction.

$$a_x = k_x \cdot v_x^2 \Leftrightarrow \frac{d^2x}{dt^2} = k_x \cdot (\frac{dx}{dt})^2 \tag{4}$$

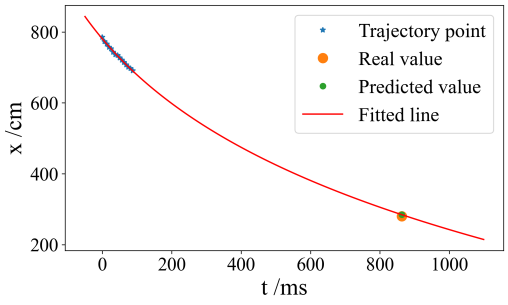
$$x = -\frac{1}{k_x} ln(t + c_{x2}) + c_{x3} \tag{5}$$

$$x = c_{x1} \cdot ln(t + c_{x2}) + c_{x3} \tag{6}$$

Based on our experimental validation, Eq. (6) can effectively fit the existing trajectory points, as shown in Fig. 4(a). It can be seen that the fitting error is very small. However,



(a)



(b)

Figure 4: Points Fitting and Prediction Figure. This is a badminton trajectory, used as an example to validate the fitting and prediction performance of Eq. (6). The horizontal axis represents time and the vertical axis represents the X-axis coordinate of shuttlecock. (a) is fitting all trajectory points by Eq. (6). (b) is fitting the first 15 trajectory points by Eq. (6) in the X-axis when c_{x3} is known, the yellow point in this figure is the real position of shuttlecock at 864.3 ms, the green point is the predicted value by Eq. (6) at 864.3 ms.

in this case, using Eq. (6) to predict subsequent trajectories results in significant prediction errors, especially for long-term trajectory predictions. Through experiments, we found that when c_{x3} is known, Eq. (6) exhibits excellent predictive capabilities, as demonstrated in Fig. 4(b). In this figure, assuming the value of c_{x3} is known, we used only the first fifteen points of the trajectory to fit the values of c_{x1} and c_{x2}. Subsequently, we predict the ball's position along the X-axis at $t = 864.3ms$, achieving a prediction error of only $4.48cm$, indicating high predictive accuracy.

Through further analysis and experimentation, we conclude that the value of c_{x3} is closely related to the type of ball and its flight state. Therefore, we employ a simple Multi-Layer Perceptron (MLP) network to predict the value of c_{x3}. The input of the MLP model is the current flight state of the ball, such as its position, velocity, and acceleration. The output of MLP model is the value of c_{x3}. This completes the trajectory prediction of the ball along the x-axis as follows:

- Obtain the current state of the ball by fitting existing trajectory points using the polynomial method.
- Enter the current state of the ball into the MLP network to obtain the value of c_{x3}.
- Fit the existing trajectory points using Eq. (6) to obtain the values of c_{x1} and c_{x2}.
- Predict the subsequent trajectory of the ball along the x-axis using Eq. (6).

3.3. Y-axis Direction

Unlike the X-axis direction, the Y-axis direction is not the primary flight direction of the ball. Therefore, the magnitudes of air resistance and other forces are comparable and cannot be directly ignored. Based on our research, when the ball's speed is relatively low, the relationship between the ball's speed and air resistance is generally linear [23], and the relationship between the Magnus force and the ball's speed is also linear [11]. In addition, other forces, such as buoyancy, are treated as a constant value (k_{y2}). Based on this, we can derive the primary force equation for the ball in the Y-axis direction as follows: $a_y = k_{y1} \cdot v_y + k_{y2}$. Where k_y is a coefficient related to the state of the ball. This formula can be expressed in the form of a second-order differential equation, as shown in Eq. (8). Here, c and c_{y4} are arbitrary constants. By further organizing, we can obtain Eq. (9), where t represents the flight time of the ball, y denotes the corresponding Y-axis coordinate of the ball at time t, and $c_{y1}, c_{y2}, c_{y3}, c_{y4}$ are parameters that need to be calculated.

$$\frac{d^2y}{dt^2} = k_{y1} \cdot \frac{dy}{dt} + k_{y2} \tag{7}$$

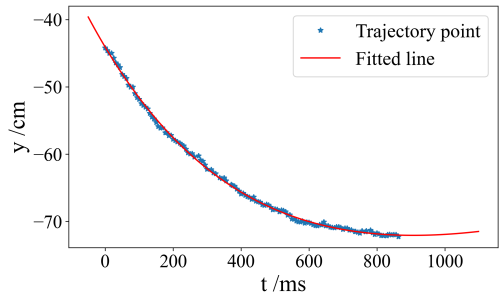
$$y = c/k_{y1} \cdot e^{k_{y1} \cdot t} - k_{y2}/k_{y1} \cdot t + c_{y4} \tag{8}$$

$$y = c_{y1} \cdot e^{c_{y2} \cdot t} + c_{y3} \cdot t + c_{y4} \tag{9}$$

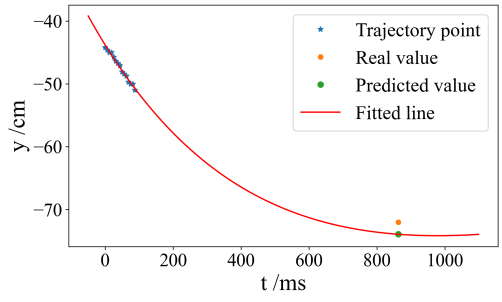
Based on our experimental validation, Eq. (9) can effectively fit the existing trajectory points, as shown in Fig. 5(a). It can be seen that the fitting error is very small. However, there are four unknown parameters here and with too many unknowns, our tests reveal that the differences in c_{y2} across various trajectories are minimal. So, we adopt a fixed value for $c_{y2} = -0.00187$ (-0.00187 is the average value derived from 200 fitted trajectories). Similarly to the X-axis, we also use a simple MLP network to predict the value of c_{y4}. When c_{y4} is known, Eq. (9) exhibits excellent predictive capabilities, as demonstrated in Fig. 5(b). In this figure, c_{y4} is known, we used only the first fifteen points of the trajectory to fit the values of c_{y1} and c_{y2}. Subsequently, we predicted the ball's position along the Y-axis at $t = 864.3ms$, achieving a prediction error of only $1.94cm$, indicating high predictive accuracy.

3.4. Z-axis Direction

Unlike the X-axis and Y-axis directions, due to the effect of gravity, the range of ball speed variation in the Z-axis direction is significantly larger. Therefore, we generally

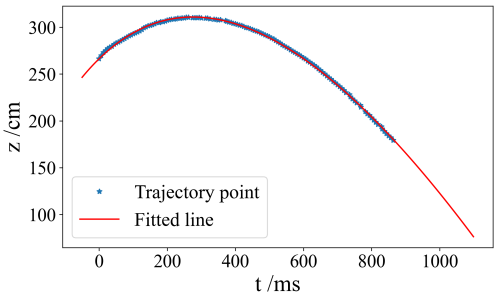


(a)

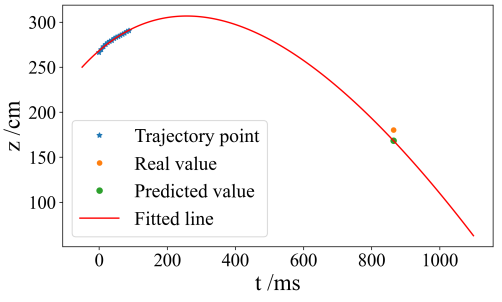


(b)

Figure 5: Points Fitting and Prediction Figure. This is a badminton trajectory, used as an example to validate the fitting and prediction performance of Eq. (9). The horizontal axis represents time and the vertical axis represents the Y-axis coordinate of shuttlecock. (a) is fitting all trajectory points by Eq. (9). (b) is fitting the first 15 trajectory points by Eq. (9) in the Y-axis when c_{y4} is known, the yellow point in this figure is the real position of shuttlecock at 864.3 ms, the green point is the predicted value by Eq. (9) at 864.3 ms.



(a)



(b)

Figure 6: Points Fitting and Prediction Figure. This is a badminton trajectory, used as an example to validate the fitting and prediction performance of Eq. (11). The horizontal axis represents time and the vertical axis represents the Z-axis coordinate of shuttlecock. (a) is fitting all trajectory points by Eq. (11). (b) is fitting the first 15 trajectory points by Eq. (11) in the Z-axis when c_{z4} is known, the yellow point in this figure is the real position of shuttlecock at 864.3 ms, the green point is the predicted value by Eq. (11) at 864.3 ms.

assume that the relationship between ball speed and air resistance is quadratic. Since in most cases, a velocity of $v_z = 0$ occurs at some point along a trajectory, at these points the air resistance in the Z-axis direction is zero. So, forces such as the Magnus force, which have a linear relationship with the ball speed, cannot be neglected. Thus, we can conclude that: $a_z = k_{z1} \cdot v_z^2 + k_{z2} \cdot v_z - g$. In most cases, this formula can be expressed in the form of Eq. (10). k_{z1} and k_{z2} are coefficients to be determined, which are related to the type of the ball and its spin, among other factors, and $g = 9.8$ is the acceleration due to gravity. In Eq. (10), z is ball's position in Z-axis, t is time, k_{z3} and k_{z4} are coefficients. Eq. (10) is a second-order differential equation, and its general solution is given by Eq. (11). Where c_{z1}, c_{z2}, c_{z3}, c_{z4} are parameters that need to be calculated.

$$\frac{d^2z}{dt^2} + k_{z3} \cdot (\frac{dz}{dt} + k_{z4})^2 = 0 \tag{10}$$

$$z = c_{z1} \cdot ln(t + c_{z2}) + c_{z3} \cdot t + c_{z4} \tag{11}$$

$$v_z = \frac{c_{z1}}{t + c_{z2}} + c_{z3} \tag{12}$$

By taking the first derivative of Eq. (11), we obtain Eq. (12), which represents the relationship between the ball's velocity (v_z) and time (t). As $\lim_{t\to\infty}$, v_z converges to c_{z3}, a constant that is solely dependent on the type of the ball. For shuttlecock, $c_{z3} = -1.2cm/ms$. Similar to the aforementioned scenario, c_{z4} can also be obtained through an MLP model. In this case, Eq. (11) also exhibits excellent fitting and predictive capabilities, as illustrated in Fig. 6. In this figure, c_{z4} is known, we used only the first fifteen points of the trajectory to fit the values of c_{z1} and c_{z2}. Subsequently, we predicted the ball's position along the Z-axis at $t = 864.3ms$, achieving a prediction error of only $11.8cm$, indicating high predictive accuracy.

3.5. Method Workflow

Our method operates iteratively, updating the algorithm whenever a new trajectory point is detected. The workflow of the algorithm is illustrated in Fig.7. First, we apply a polynomial fitting method to the latest 20 trajectory points to estimate the current state of the ball (or all available points if fewer than 20 are detected). Next, the ball's state (position, velocity, and acceleration) is fed into a multilayer perceptron (MLP) to compute the parameters c_{x3}, c_{y4}, and c_{z4}. Subsequently, the remaining parameters of Eq.(6), (9),

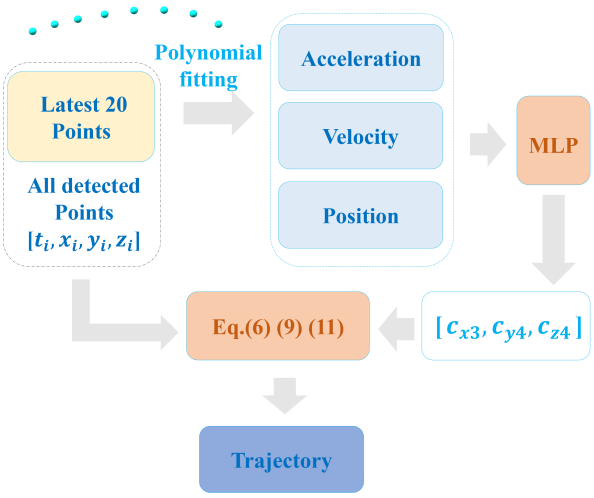


Figure 7: Data Processing Flowchart. In our method, we first use polynomial fitting to fit the latest 20 trajectory points to obtain the current state of the ball, and then predict the subsequent trajectory through MLP and Eq. (6), (9), (11).

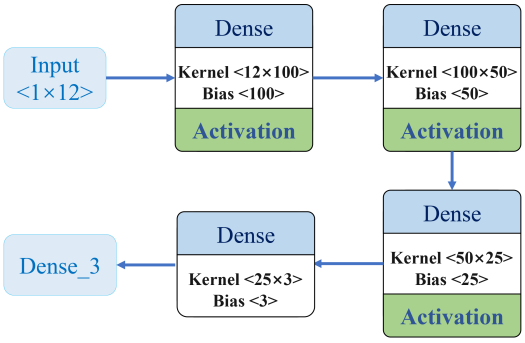


Figure 8: The Structure of MLP in Our Method. This is the structure of MLP, in this figure, the model's input is 1x12, and the output is 1x3.

and (11) are determined by fitting all trajectory points to these equations. This process enables accurate prediction of the ball's subsequent trajectory based on the derived parameters.

4. System Implementation

Our method is designed and implemented as follows. The ball's position is captured using a stereo camera system operating at a frame rate of 160 frames per second (fps), which is increased to 220 fps for table tennis applications. This section primarily focuses on the introduction of the MLP model and key algorithmic details.

4.1. MLP

In our method, the MLP model is employed to predict the parameters c_{x3}, c_{y4} and c_{z4} in Eq. (6), (9), and (11). The model utilizes the ReLU activation function due to its computational efficiency and ability to mitigate the vanishing gradient problem. Training is conducted using the Adam optimizer, which provides an adaptive learning rate and rapid convergence. To ensure precise regression, the model is optimized with the mean squared error (MSE) loss function. As a result, the proposed approach achieves both high inference speed and prediction accuracy, making it well-suited for real-time trajectory prediction tasks, as illustrated in Fig. 8.

The input dimension of the model is 1×12, representing the ball's current position, velocity, and acceleration along the X, Y, and Z axes. These features are derived by fitting the latest 20 trajectory points $[t_i, x_i, y_i, z_i]$ using cubic polynomials. Specifically, t_i is fitted separately against x_i, y_i and z_i using cubic polynomial regression. If fewer than 20 trajectory points are available, all existing points are used for fitting. The model then outputs the estimated values of c_{x3}, c_{y4} and c_{z4}.

To train a robust model, only a few dozen trajectory data points are needed. However, data augmentation is essential to enhance diversity. For instance, a single trajectory may encompass numerous states corresponding to different ball positions, yet the values of c_{x3}, c_{y4} and c_{z4} remain unchanged. In such cases, data diversity can be effectively increased through transformations such as translation and rotation.

4.2. Techniques for Reducing Prediction Errors

From our method design, it is evident that accurate trajectory prediction is crucial for the robot's ability to successfully hit the ball. However, prediction errors are inevitable, especially in our approach, where an MLP is used to estimate c_{x3}, c_{y4} and c_{z4} in Eq. (6), (9), and (11). Errors in these predicted parameters can propagate and significantly amplify subsequent trajectory prediction inaccuracies.

To mitigate this issue, we employ a simple yet effective technique: Suppose that we have detected 37 trajectory points. We first fit a polynomial to the latest 20 points (from the 18th to the 37th) and use MLP to estimate c_{x3}, c_{y4} and c_{z4}. Next, we fit the first 24 points (instead of all 37 points) to calculate the values of c_{x1}, c_{x2}, c_{y1}, c_{y3} and c_{z1}, c_{z2} (c_{y2} and c_{z3} are constant values). Using Eq. (6), (9), and (11), we then predict the ball's position along the X, Y and Z-axis at t_{35}, t_{36} and t_{37}, and compare these predictions with the actual coordinates to determine the prediction error. This error primarily stems from the inaccuracy of c_{x3}, c_{y4} and c_{z4}. Based on the error, we iteratively adjust these values (either increasing or decreasing its value) until the error falls below a predefined threshold.

5. Experiments

This section evaluates the performance of our proposed ball trajectory prediction method through a series of experiments from five key perspectives. First, we compare the prediction accuracy of different neural network architectures in estimating c_{x3}, c_{y4} and c_{z4}. Second, we assess the prediction

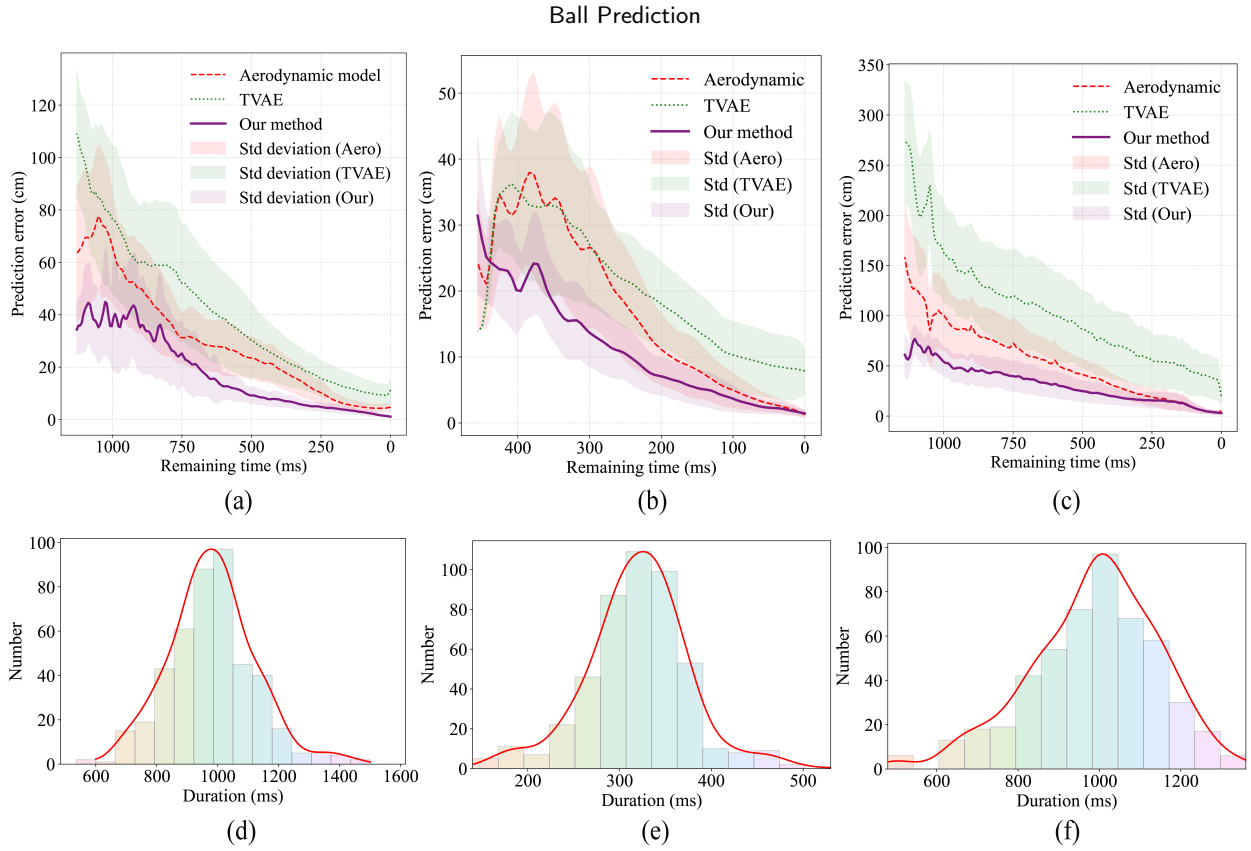


Figure 9: The Comparison between our method, TVAE[14], and the aerodynamic model [10] in the prediction error of last trajectory point. Here, green represents the aerodynamic model, red denotes TVAE, and purple indicates our method. (a) is the average prediction error of badminton, (b) is the average prediction error of table tennis, (c) is the average prediction error of tennis. (d), (e), and (f) are the distribution of trajectory durations for badminton, table tennis, and tennis used in the tests, respectively.

performance of our fitting-based model across badminton, tennis, and table tennis by benchmarking it against existing methods. Third, we examine the trajectory prediction accuracy under specific conditions, such as when a badminton shuttlecock lacks feathers. Fourth, we extend our approach to pickleball and evaluate its predictive accuracy. Fifth, we compare the inference speed of different methods to assess their suitability for real-time applications. Our method requires only a small amount of training data while still achieving strong generalization performance. Specifically, we train our model using just 90 trajectory samples per sport, including badminton, table tennis, tennis, and pickleball. Experimental results demonstrate that our model achieves an average inference time of 0.97 ms, satisfying the real-time performance requirements.

5.1. Comparison of Different Neural Networks

As previously discussed, our method employs a neural network to predict c_{x3}, c_{y4} and c_{z4}. Table 2 presents a comparative analysis of the prediction errors for these parameters, along with the inference time of various models. The results indicate that the MLP surpasses both LSTM and CNN in terms of prediction accuracy and computational efficiency. Notably, since the values of c_{x3} and c_{z4} typically exceed 2000, the relative prediction error of the MLP remains low.

Table 2
Comparison of Different Neural Networks

Model	Prediction error			Inference time(ms)
	c_{x3}	c_{y4}	c_{z4}	
LSTM [24]	185.92	16.28	163.59	1.8
CNN [25]	176.41	13.04	161.74	5.68
MLP [26]	**173.73**	**10.59**	**152.85**	0.018

5.2. The Last Point Prediction Error

In this section, we primarily evaluate the prediction error of the last trajectory point. While prediction accuracy is typically assessed based on the landing point, practical applications may encounter challenges such as occlusions, making it undetectable at times. To address this, we use the last trajectory point before the bounce, denoted as $p_l = [x_l, y_l, z_l]$, as the ground truth and compute prediction errors at different timestamps. Specifically, the predicted last point

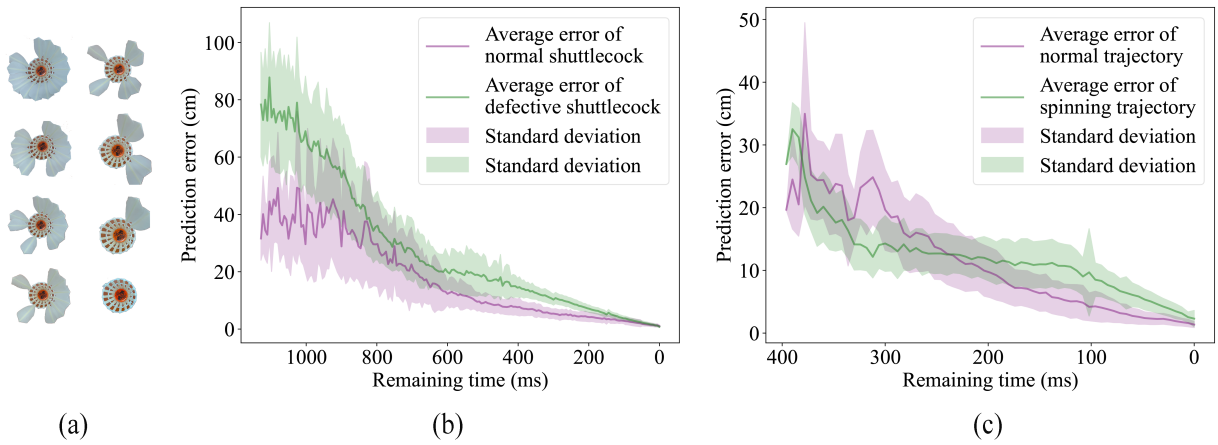


Figure 10: The Prediction Error in Special Conditions. (a) refers to defective shuttlecocks where one or more feathers of a shuttlecock are cut off. (b) is the prediction error of the last trajectory point between normal shuttlecocks and defective shuttlecocks. (c) is the prediction error of the last trajectory point between non-spinning table tennis and spinning table tennis. The horizontal axis represents the remaining time until the last trajectory point, and the vertical axis represents the prediction error.

$p_p = [x_p, y_p, z_p]$ is defined as the intersection of the predicted trajectory with the plane $z = z_l$, and the prediction error is calculated as $\sqrt{(x_p - x_l)^2 + (y_p - y_l)^2}$.

Furthermore, we compare our approach with the aerodynamic model [10] and the learning-based model TVAE [14]. Experiments are conducted using 445 badminton trajectories, 470 table tennis trajectories, and 500 tennis trajectories. As shown in Fig. 9 (d), (e), and (f), the trajectory duration distribution is relatively uniform. All trajectories are collected from real matches. The final results, presented in Fig. 9 (a), (b), and (c), demonstrate that our method achieves the highest prediction accuracy.

5.3. The Generalizability of Our Method

In this section, we assess the generalizability of our method in two challenging scenarios: (1) predicting the last trajectory point when a badminton shuttlecock experiences feather loss, and (2) predicting the last trajectory point for a spinning table tennis ball.

For the first scenario, we collected 8 sets of 520 trajectories, where shuttlecock feathers were randomly clipped, resulting in two, four, six, etc., feathers missing, as illustrated in Fig. 10 (a). Each group contains 71, 71, 72, 73, 77, 73, 70, and 13 trajectories, respectively. The prediction errors for the last trajectory point are compared between normal and defective shuttlecocks in Fig. 10 (b). While the prediction error for normal shuttlecocks is initially smaller, the errors for both cases converge closely when the remaining time is less than 800 ms. For the second scenario, we collected 320 spinning ball trajectories using a table tennis ball launcher. These include topspin, backspin, left spin, and right spin, with two sets for each type at rotation rates of 10.23 revolutions per second (rev/s) and 20.95 rev/s, respectively. The prediction errors for the last trajectory point between non-spinning and spinning balls are shown in Fig. 10 (c). The results indicate that the prediction errors under both conditions are very similar.

In both cases, the MLP model remains unchanged, and the training data does not include abnormal shuttlecock trajectories or spinning ball trajectories. These experiments demonstrate the strong generalizability of our method to unseen conditions.

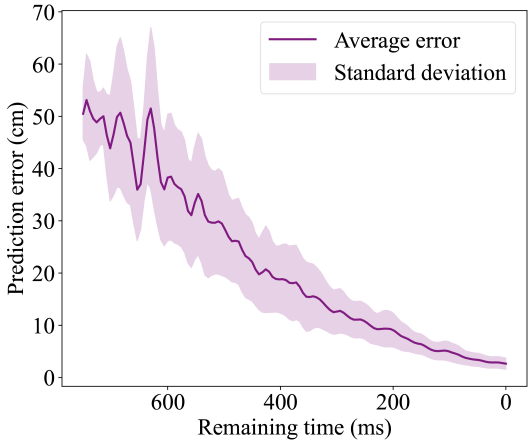


Figure 11: The Prediction Error in pickleball. This figure shows the prediction error of the last trajectory point over time. The horizontal axis represents the remaining time until the last trajectory point, and the vertical axis represents the prediction error.

5.4. The Extension of Our Method to Pickleball

To evaluate the extensibility of our method, we conduct experiments on pickleball. The model is trained using 90 pickleball trajectories and tested on 103 trajectories to assess its prediction accuracy. The test trajectory durations ranged from 300 ms to 1000 ms. Fig. 11 illustrates the prediction error of the last trajectory point over time. Experimental

results indicate that our method achieves a satisfactory level of prediction accuracy.

5.5. The Inference Time of Our Model

We utilize a hybrid approach that combines physical equations with neural networks, where the inference speed is primarily influenced by the neural network and iterative optimization. According to our tests, the average inference time of our method is 0.97 ms, whereas the inference time of TVAE ranges from 4 to 8 ms. Therefore, our method fully meets real-time performance requirements.

5.6. Summary

From these experiments, we can get several takeaways: 1) Our method performs well in terms of generalization and real-time performance. 2) Our method achieves excellent results by integrating neural networks with physical principles. 3) Our design is effective; it can overcome the above difficulties and challenges.

6. Conclusion

This paper introduces a universal ball trajectory prediction method that seamlessly integrates physical principles with neural networks to achieve accurate real-time predictions for various types of balls. We begin by deriving three generalized equations that govern ball flight through rigorous physical analysis. A neural network is then employed to estimate specific parameters within these equations, while the remaining parameters are determined by fitting observed trajectory points, enabling precise forecasting of future motion. Remarkably, the neural network requires only a limited number of trajectory samples for training. Extensive experiments validate the proposed method's effectiveness in terms of prediction accuracy, adaptability, and real-time performance.

7. Limitation and Discussion

However, our method is still in the prototype stage, and there are several areas that require further refinement and validation to enhance its overall reliability and applicability in real-world scenarios.

1) Our method currently predicts ball trajectories only before or after the bounce, as it lacks a bounce model. Next, we aim to incorporate a bounce model to enhance the comprehensiveness of the trajectory prediction system.

2) Our method employs an MLP network that currently requires separate training for each type of ball. While the training data requirements are minimal, this limits the model's generalizability. To address this, we plan to refine the architecture to develop a more universal model capable of predicting trajectories for various ball types.

3) Currently, our experiments are focused exclusively on small balls, such as shuttlecocks, and the accuracy of trajectory prediction for larger balls, like basketballs and soccer balls, has yet to be verified. In the next phase, we will conduct experiments to evaluate and refine our method for these larger ball scenarios.

CRediT authorship contribution statement

Zhiwei Shi: System implementation, Methodology, Software, Writing. **Chengxi Zhu:** Experiment . **Fan Yang:** Writing and Experiment. **Jun Yan:** Writing . **Zheyun Qin:** Writing . **Songquan Shi:** Writing. **Zhumin Chen:** Conceptualization of this study.

References

[1] Manju Rana and Vikas Mittal. Wearable sensors for real-time kinematics analysis in sports: A review. *IEEE Sensors Journal*, 21(2):1187–1207, 2020.

[2] Akash Anand, Manish Sharma, Rupika Srivastava, Lakshmi Kaligounder, and Divya Prakash. Wearable motion sensor based analysis of swing sports. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 261–267, 2017.

[3] Cheng Zhang, Fan Yang, Gang Li, Qiang Zhai, Yi Jiang, and Dong Xuan. Mv-sports: a motion and vision sensor integration-based sports analysis system. In *IEEE INFOCOM 2018-IEEE conference on computer communications*, pages 1070–1078. IEEE, 2018.

[4] Shuainan Ye, Chen Zhu-Tian, Xiangtong Chu, Yifan Wang, Siwei Fu, Lejun Shen, Kun Zhou, and Yingcai Wu. Shuttlespace: Exploring and analyzing movement trajectory in immersive visualization. *IEEE transactions on visualization and computer graphics*, 27(2):860–869, 2020.

[5] Kuang-Da Wang, Yu-Tse Chen, Yu-Heng Lin, Wei-Yao Wang, and Wen-Chih Peng. The coachai badminton environment: Bridging the gap between a reinforcement learning environment and real-world badminton games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 23844–23846, 2024.

[6] Paul Liu and Jui-Hsien Wang. Monotrack: Shuttle trajectory reconstruction from monocular badminton video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3513–3522, 2022.

[7] Zulfiqar Zaidi, Daniel Martin, Nathaniel Belles, Viacheslav Zakharov, Arjun Krishna, Kin Man Lee, Peter Wagstaff, Sumedh Naik, Matthew Sklar, Sugju Choi, et al. Athletic mobile manipulator system for robotic wheelchair tennis. *IEEE Robotics and Automation Letters*, 8(4):2245–2252, 2023.

[8] Shotaro Mori, Kazutoshi Tanaka, Satoshi Nishikawa, Ryuma Niiyama, and Yasuo Kuniyoshi. High-speed and lightweight humanoid robot arm for a skillful badminton robot. *IEEE Robotics and Automation Letters*, 3(3):1727–1734, 2018.

[9] Yunfeng Ji, Xiaoyi Hu, Yutao Chen, Yue Mao, Gang Wang, Qingdu Li, and Jianwei Zhang. Model-based trajectory prediction and hitting velocity control for a new table tennis robot. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2728–2734. IEEE, 2021.

[10] Fan Yang, Zhiwei Shi, Sixian Ye, Jiazhong Qian, Wenjie Wang, and Dong Xuan. Varsm: Versatile autonomous racquet sports machine. In *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*, pages 203–214. IEEE, 2022.

[11] Rabindra D Mehta. Aerodynamics of sports balls. *Annual Review of Fluid Mechanics*, 17(1):151–189, 1985.

[12] Yanlong Huang, De Xu, Min Tan, and Hu Su. Trajectory prediction of spinning ball for ping-pong player robot. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3434–3439. IEEE, 2011.

[13] Tianxiao Zhang, Xiaohan Zhang, Yiju Yang, Zongbo Wang, and Guanghui Wang. Efficient golf ball detection and tracking based on convolutional neural networks and kalman filter. *arXiv preprint arXiv:2012.09393*, 2020.

[14] Sebastian Gomez-Gonzalez, Sergey Prokudin, Bernhard Schölkopf, and Jan Peters. Real time trajectory prediction using deep conditional generative models. *IEEE Robotics and Automation Letters*, 5(2):970–976, 2020.

[15] Qingyu Xiao, Zulfiqar Zaidi, and Matthew Gombolay. Multi-camera asynchronous ball localization and trajectory prediction with factor graphs and human poses. *arXiv preprint arXiv:2401.17185*, 2024.

[16] Zhibo Zhang, Yanjun Zhu, Rahul Rai, and David Doermann. Pimnet: Physics-infused neural network for human motion prediction. *IEEE Robotics and Automation Letters*, 7(4):8949–8955, 2022.

[17] Beatriz Moya, Alberto Badías, David González, Francisco Chinesta, and Elias Cueto. Computational sensing, understanding, and reasoning: an artificial intelligence approach to physics-informed world modeling. *Archives of Computational Methods in Engineering*, 31(4):1897–1914, 2024.

[18] Hongyang Zhao, Shuangquan Wang, Gang Zhou, and Woosub Jung. Tenniseye: tennis ball speed estimation using a racket-mounted motion sensor. In *Proceedings of the 18th international Conference on information Processing in Sensor Networks*, pages 241–252, 2019.

[19] Erwin Wu and Hideki Koike. Futurepong: Real-time table tennis trajectory forecasting using pose prediction network. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–8, 2020.

[20] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

[21] Hossein Jafari, Babak Fazli Malidareh, and Vahid Reza Hosseini. Collocation discrete least squares meshless method for solving nonlinear multi-term time fractional differential equations. *Engineering analysis with boundary elements*, 158(Jan.):107–120, 2024.

[22] Chao Zhang, Zhuojia Fu, and Yaoming Zhang. A novel global rbf direct collocation method for solving partial differential equations with variable coefficients. *Engineering Analysis with Boundary Elements*, 160:14–27, 2024.

[23] George Gabriel Stokes et al. On the effect of the internal friction of fluids on the motion of pendulums. 1851.

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[25] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, and Daniel J. Inman. 1d convolutional neural networks and applications: A survey. 2019.

[26] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.