Application of machine learning in the determination of impact parameter in the ^{132}Sn + ^{124}Sn system

Fupeng Li,[1,2] Yongjia Wang,[1,*] Zepeng Gao,[3,1] Pengcheng Li,[4,1] Hongliang Lü,[5] Qingfeng Li,[1,6,†] C. Y. Tsang,[7] and M. B. Tsang[7]

[1]*School of Science, Huzhou University, Huzhou 313000, China*
[2]*College of Science, Zhejiang University of Technology, Hangzhou 310014, China*
[3]*College of Physics Science and Technology, Shenyang Normal University, Shenyang 110034, China*
[4]*School of Nuclear Science and Technology, Lanzhou University, Lanzhou 730000, China*
[5]*HiSilicon Research Department, Huawei Technologies Co., Ltd., Shenzhen 518000, China*
[6]*Institute of Modern Physics, Chinese Academy of Sciences, Lanzhou 730000, China*
[7]*National Superconducting Cyclotron Laboratory and the Department of Physics and Astronomy, Michigan State University, East Lansing, Michigan 48824, USA*

(Received 11 May 2021; accepted 17 August 2021; published 7 September 2021)

Background: ^{132}Sn + ^{124}Sn collisions at a beam energy of 270 MeV/nucleon were performed at the Radioactive Isotope Beam Factory (RIBF) in RIKEN to investigate the nuclear equation of state. Reconstructing the impact parameter is one of the important tasks in the experiment as it relates to many observables.
Purpose: In this work, we employ three commonly used algorithms in machine learning, the artificial neural network (ANN), the convolutional neural network (CNN), and the light gradient boosting machine (LightGBM), to determine the impact parameter by analyzing either the charged particle spectra or several features simulated with events from the ultrarelativistic quantum molecular dynamics (UrQMD) model.
Method: To closely imitate experimental data and investigate the generalizability of the trained machine learning algorithms, incompressibility of nuclear equation of state and the in-medium nucleon-nucleon cross sections are varied in the UrQMD model to generate the training data.
Results: The mean absolute error Δb between the true and the predicted impact parameter is smaller than 0.45 fm if training and testing sets are sampled from the UrQMD model with the same parameter set. However, if training and testing sets are sampled with different parameter sets, Δb would increase to 0.8 fm.
Conclusion: The generalizability of the trained machine learning algorithms suggests that these machine learning algorithms can be used reliably to reconstruct the impact parameter in experiments.

DOI: 10.1103/PhysRevC.104.034608

I. INTRODUCTION

Heavy-ion collisions (HICs) provide a unique opportunity to explore the nuclear equation of state (EoS), which remains a key requirement for understanding nuclear reactions, nuclear structure, as well as neutron star properties [1–7]. In recent decades, both experimentalists and theorists have made major efforts to obtain information of the EoS. These studies reveal that the uncertainty of the nuclear EoS is the largest in the density-dependent term at high densities. For this purpose, ^{132}Sn + ^{124}Sn, ^{112}Sn + ^{124}Sn, and ^{108}Sn + ^{112}Sn collisions at a beam energy of 270 MeV/nucleon were performed at the Radioactive Isotope Beam Factory (RIBF) in RIKEN [8].

Usually in an experiment, the centrality or impact parameter is reconstructed by using the relationship between observed quantities and the collision geometry [9–13]. Recently, the field of artificial intelligence (AI) has received unprecedented attention, and prodigious progress has been made in the application of AI techniques; see, e.g., Ref. [14] and references therein. Machine learning (ML), which is a subset of AI, is an interdisciplinary subject, involving probability theory, statistics, approximation theory, convex analysis, algorithm complexity theory, and other subjects. Due to its powerful learning and induction ability, ML approaches are widely employed in physical science [15–29]. Of particular relevance to this work, there are several applications of ML in reconstructing the impact parameter in HICs [30–34]. For example, in Refs. [30–33], an artificial neural network (ANN) or support vector machine is used to reconstruct the impact parameter from final state observables or the particle momentum distributions. In our previous work [35], we utilized a convolutional neural network (CNN) and a light gradient boosting machine (LightGBM) to determine the impact parameter from two-dimensional transverse momentum and rapidity spectra of protons on an event-by-event basis. It was found that the average difference between the true impact parameter and the estimated one obtained with modern ML algorithms, i.e., CNN and LightGBM, is much smaller than that obtained

*Corresponding author: wangyongjia@zjhu.edu.cn
†Corresponding author: liqf@zjhu.edu.cn

with simple neutral networks adopted 25 years ago [35]. In addition, modern ML algorithms have much stronger big-data processing and learning capabilities, as well as stronger generalizability and explainability, which may help us to find new insight into the existing data. All these aforementioned studies revealed the capability of ML methods in reconstructing the impact parameter. We note that in these studies the training data are usually generated with theoretical models. For example, the quantum molecular dynamics (QMD) model was used to generate data in Refs. [30–32], and a classical molecular dynamics (CMD) model was used in Ref. [33]. When applying these ML methods to analyze real experimental data, the reliability should be evaluated as none of the theoretical models represent real experimental data perfectly. Using data generated from different physical models or different model parameter sets from the same model should give a good estimation of ML's capability. For this purpose, the ultrarelativistic quantum molecular dynamics (UrQMD) model with different nuclear EoS and different in-medium nucleon-nucleon cross sections (two of the main ingredients in the transport model) is used to generate data, and the generalizability of ML methods is investigated by generating training and testing sets with different model parameters.

The paper is organized as follows. In Sec. IV A, we will briefly introduce the UrQMD model and different datasets in this study. We continue with Sec. IV B in which ANN, CNN, and LightGBM algorithms will be described. In Sec. IV C, we discuss the results and generalizability of the three algorithms in detail. We end with Sec. V, which is dedicated to summary and outlook.

II. UrQMD MODEL

The UrQMD model is a many-body microscopic transport model which has been successfully extended to describe HICs with beam energy from tens of MeV per nucleon up TeV per nucleon available at the CERN Large Hadron Collider (LHC) [36–41]. In the UrQMD model, each nucleon is represented by a Gaussian wave packet in phase space. The coordinates r_i and momentum p_i of particles i are propagated according to Hamilton's equation of motion:

$$\dot{\mathbf{r}}_i = \frac{\partial\langle H\rangle}{\partial \mathbf{p}_i}, \quad \dot{\mathbf{p}}_i = -\frac{\partial\langle H\rangle}{\partial \mathbf{r}_i}. \tag{1}$$

Here, $\langle H\rangle$ is the total Hamiltonian function. It consists of the kinetic energy T and the potential energy U with $U = \sum_{i\neq j} V_{ij}$. The following density and momentum dependent potential has been widely employed in QMD-like models [42–47]:

$$V_{ij} = \alpha\left(\frac{\rho_{ij}}{\rho_0}\right) + \beta\left(\frac{\rho_{ij}}{\rho_0}\right)^{\eta} + t_{md}\ln^2[1 + a_{md}(\mathbf{p}_i - \mathbf{p}_j)^2]\frac{\rho_{ij}}{\rho_0}. \tag{2}$$

In this work, the parameter sets which yield a soft (hard) and momentum dependent equation of state with the incompressibility $K_0 = 200$ MeV ($K_0 = 380$ MeV) are considered. From now on we refer to the soft and hard EoS as SM and HM respectively. Even though K_0 has been constrained to a relatively narrow range [48–52]), SM and HM are still considered in this work to generate data with large differences. Further, although we know the in-medium nucleon-nucleon elastic cross section (σ_{NN}) is suppressed when compared to the free one, the degree of this suppression is still not completely pinned down [53,54]. We use the FU3FP1 parametrization of σ_{NN} as in our previous works [53,54]. We also consider the free σ_{NN} in this study. All together, four parameter sets of the UrQMD model listed in Table I are used. Their influences on five observable quantities—the nuclear stopping power (vartl) from free protons, the directed flow $v_1 = \langle\frac{p_x}{p_t}\rangle$, the elliptic flow $v_2 = \langle\frac{p_x^2 - p_y^2}{p_x^2 + p_y^2}\rangle$, yield of free protons, and multiplicity for central ($0 \leqslant b \leqslant 2$ fm) collisions obtained with different model parameter sets—are listed in Table II. Clearly, observables are affected by model parameters. For example, v_1 increases by 70% if the flow obtained from SM-I is compared to that from HM-F. The isospin-dependent minimum span tree (iso-MST) algorithm is used in UrQMD model to recognize clusters. The yields of free protons and clusters are very sensitive to cluster recognition parameters (i.e., the maximum distance and relative momentum between two nucleons). To consider this issue, calculations with parameter sets different from the nominal ones (see caption for details) are also presented as SM-I(MST)*. As listed in Table II, the number of free protons and M_{ch} also vary a lot with the cluster recognition parameters.

TABLE I. Four parameter sets of the UrQMD model with different mean-field potential and nucleon-nucleon elastic cross section.

EoS	Cross section	Mode
SM	free	SM-F
SM	in medium	SM-I
HM	free	HM-F
HM	in medium	HM-I

TABLE II. Observable quantities (i.e., v_1 slope and v_2 of free protons at midrapidity, the yield and vartl of free protons, the total charged multiplicity M_{ch}) obtained with different UrQMD parameter sets. We note here that the results listed in this table cannot be compared directly with experimental data, because events with flat b dependence are simulated. To compare with experimental results, calculations with b-weighted events should be used.

Mode	v_1 slope	v_2	Yield	vartl	M_{ch}
SM-F	0.14	-0.0046 ± 0.0011	44.98	0.94	87.60
SM-I	0.11	-0.0024 ± 0.0012	43.97	0.91	86.35
SM-I(MST)[a]	0.097	-0.0043 ± 0.0013	36.95	0.92	81.84
HM-F	0.19	-0.0077 ± 0.0010	50.07	0.97	90.51
HM-I	0.15	-0.0043 ± 0.0010	49.14	0.89	89.24

[a]This is SM-I mode in combination with MST algorithm (two nucleons with relative distance $\Delta r \leqslant 4.8$ fm and relative momentum $\Delta p \leqslant 0.25$ GeV/c are considered to belong to the same cluster) to recognition fragments. In other cases, the isospin dependent MST algorithm with $\Delta r^{pp} \leqslant 2.8$ fm, $\Delta r^{nn} \leqslant 3.8$ fm, $\Delta r^{np} \leqslant 3.8$ fm, and $\Delta p \leqslant 0.25$ GeV/c is used.

For each parameter set, 60 000 events of ^{132}Sn + ^{124}Sn collisions with a uniform impact parameter distribution in $0 \leqslant b \leqslant 7$ fm at 270 MeV/nucleon are simulated. Data obtained from 50 000 of these events are classified as the training data while the remaining 10 000 events are the testing data. Normally, the $b\,db$ weighted distribution (i.e., the number of events with an impact parameter b being proportional to b) due to the collision gefometry is used in transport model simulations. In Refs. [31,34,35], large bias between predicted and true impact parameter was observed for the central collisions because a very small fraction of the total events are central collisions[1] [55–57]. To avoid this issue, events with flat distribution of impact parameter are simulated.

Usually, the transverse momentum $p_t = \sqrt{p_x^2 + p_y^2}$ and rapidity $y_z = \frac{1}{2}\ln[\frac{E+p_z}{E-p_z}]$ of charged particles[2] can be measured in heavy-ion experiments. In Ref. [58], the reduced rapidity $y_0 = y_z/y_{\rm pro}$ is used instead of y_z. Here, $y_{\rm pro}$ denotes the rapidity of the projectile in the center-of-mass system. In order to minimize preprocessing, the two-dimensional p_t and y_0 spectrum of all charged particles with 30×30 grid is also used as the input dataset. p_t ranges from 0 to 1 GeV/c and y_0 ranges from -2 to 2. This two-dimensional p_t and y_0 spectrum of all charged particles is labeled as *DATASET1*.

For *DATASET2*, we use seven input features or observables obtained from ^{132}Sn + ^{124}Sn at 270 MeV/nucleon.

Five of the seven features are the number of deuteron, triton, and helium isotopes, $N(d, t, \mathrm{He})$; the averaged transverse momentum of deuteron, triton, and helium isotopes, $N(d, t, \mathrm{He})p_t$; the number of free protons at mid-rapidity ($|y_0| \leqslant 0.5$), Np; and the averaged transverse momentum of free protons at mid-rapidity, Np_t. The remaining two features are ERAT for free protons, defined as

$$\mathrm{ERAT} = \frac{\sum_i [p_{ti}^2/(2m + E_i)]}{\sum_i [p_{zi}^2/(2m + E_i)]}, \tag{3}$$

and the transverse kinetic energy $E_\perp$ for light charged particles with the charge number $Z = 1$ and $Z = 2$, defined as

$$E_\perp = \sum_{Z=1,2} \frac{p_t^2}{2m}. \tag{4}$$

The above variables are selected not only because they are correlated to impact parameter but also because the measurement of these variables in experiments on the event-by-event basis is feasible. In addition, for a fixed impact parameter, variables with smaller event-by-event fluctuations are also of benefit to ML algorithms. In this context, variables like directed and elliptic flows are not used because of their large event-by-event fluctuations. From a theoretical point of view, the size of the spectator fragments, or the largest fragment with projectile (target) rapidity, is also a good candidate for determining the impact parameter. In most experiments that focus on central collisions, the large projectile-like fragments are rejected to enhance the detection of central collision events which have much smaller cross sections. Therefore, the size of spectator fragments is not used as well.

We use three different ML algorithms with dataset as listed in Table III. To assess the accuracy of the reconstruction of the impact parameter, the performance of different algorithms can be quantified by the mean absolute error:

$$\Delta b = \frac{1}{N_{\rm event}} \sum_{i=1}^{N_{\rm events}} \left| b_i^{\rm true} - b_i^{\rm pred} \right|. \tag{5}$$

Here, $b_i^{\rm true}$ is the true impact parameter of each event and $b_i^{\rm pred}$ is the predicted one from different algorithms.

TABLE III. Four different ML algorithms with dataset.

Algorithm	Dataset	Label
CNN	*DATASET1*	CNNa
LightGBM	*DATASET1*	LightGBMa
ANN[a]	*DATASET2*	ANNb
LightGBM	*DATASET2*	LightGBMb

[a]ANN is more suitable for data with seven input features than CNN. See details in Sec. IV B.

III. ANN, CNN, AND LightGBM ALGORITHMS

In this work, we use three most representative algorithms, ANN, CNN, and LightGBM, to determine the impact parameter; the detailed parameter sets of these methods are the same as that in our previous work [35]. When ANN solves a problem, it converts the input data into a one-dimensional vector. The number of fitting parameters increases with dimensions. Therefore many more parameters are needed to handle input data with larger dimension [59,60]. For image data, ANN easily loses its spatial characteristics, resulting in unsatisfactory training results. To avoid this problem, newer neural networks (such as CNN) have been developed. CNN algorithm is one of feed-forward neural networks that includes convolution calculations and has a deep structure [61,62]. Because of the introduction of local receptive fields and shared weights, it requires many fewer parameters compared with ANN. LightGBM is a new gradient boosting tree framework developed by Microsoft; it is highly efficient and scalable and can support many different algorithms. Its advantages include (1) faster training efficiency, (2) low memory usage, (3) higher accuracy, and (4) ability to tackle large-scale data [63,64]. For high-dimensional or weakly correlated input data, usually, the performance of decision-tree-based algorithms is not good as that of neural networks. However, for solving problems of physical nature, decision-tree-based algorithms are usually favored because of their high interpretability. In general, ANN is more suitable for tabular data such as *DATASET1*, CNN is much more powerful for handling image-like data (i.e., *DATASET2*), while LightGBM is suitable for both *DATASET1* and *DATASET2*.

[1]Another possible reason is that physical fluctuation is larger in central collisions than in peripheral ones.

[2]Throughout this paper, transverse momentum per nucleon is used instead of transverse momentum for clusters.

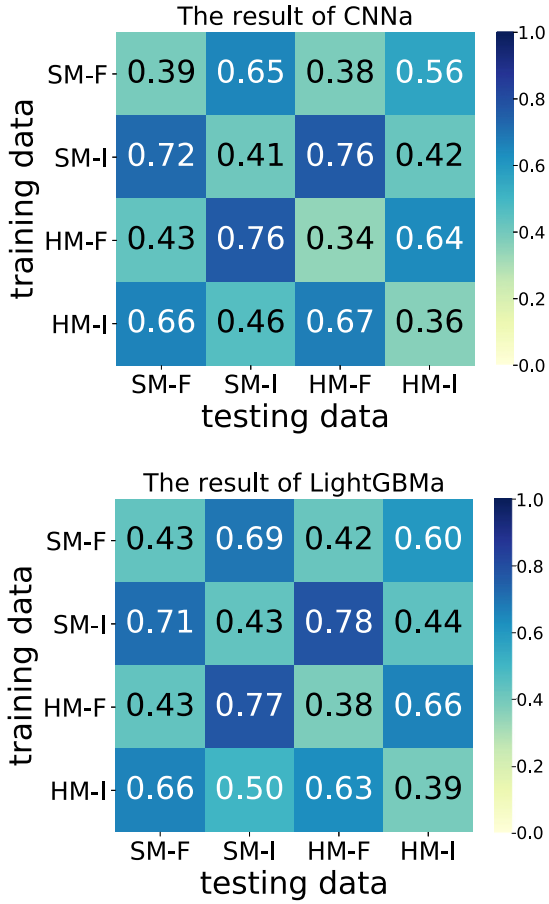

FIG. 1. The results of the CNNa and LightGBMa algorithms when *DATASET1* is used. The number in each cell denotes Δb for the testing data (generated with the vertical labeled mode) by using the training data (generated with the horizontal labeled mode). The statistical error due to the randomness in the testing data was estimated to be smaller than 1% by comparing parallel testing data, and is therefore negligible.

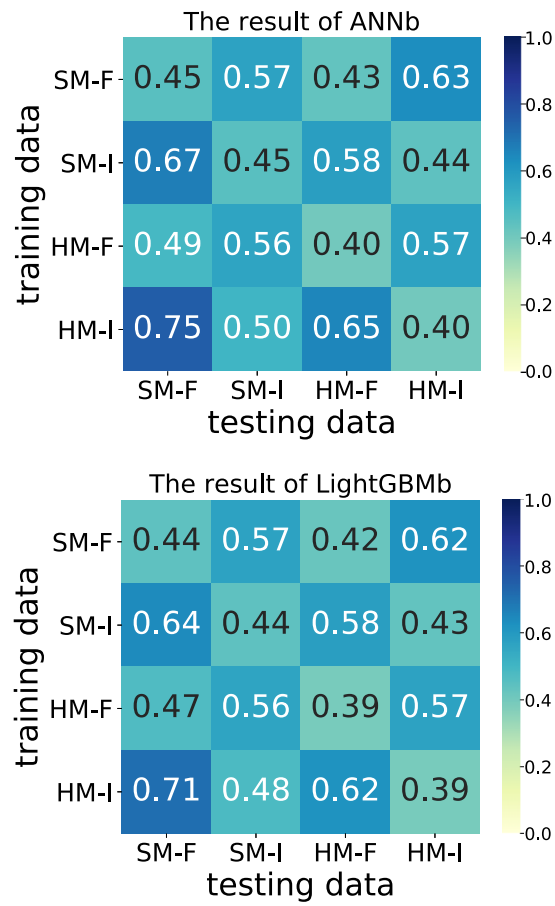

FIG. 2. The results of the ANNb and LightGBMb algorithms by using *DATASET2*. The number in each cell denotes Δb for the testing data (generated with the vertical labeled mode) by using the training data (generated with the horizontal labeled mode).

IV. RESULTS AND DISCUSSIONS

A. Reconstruction results of *DATASET1* and *DATASET2*

The results of CNNa and LightGBMa in which *DATASET1* serves as the input training data are displayed in Fig. 1. As can be seen, Δb is about 0.3–0.4 fm (numbers along the diagonal) if both the training data and testing data are generated from the same UrQMD model parameter set. By using training and testing data obtained from different parameter sets (off diagonal), e.g., the two-dimensional p_t and y_0 spectrum of all charged particles generated with SM-I serves as the training data while simulation data generated with HM-F serve as the testing data, Δb is increased to about 0.8 fm. This is understandable due to parameters in both the mean-field potential and collision terms (two of the main ingredients of the transport model) being different in SM-I and HM-F modes. In addition, it can be found that Δb is affected much more by cross section than by K_0. For example, Δb for testing data obtained from SM-F by using ML algorithms trained with data from SM-I mode is about 0.7 fm, while by using ML algorithms trained with data from HM-F it is about 0.4 fm. This is due to the fact that both b and σ_{NN} strongly affect the number of collisions and the final observed particle spectra. Thus, the fingerprint of impact parameter on particle spectra is erased to some extent by varying σ_{NN}. Furthermore, Δb in most cases obtained with CNNa is slightly smaller than that obtained with LightGBMa, indicating that CNN has a better performance on *DATASET1* than LightGBM. However, considering the fact that LightGBM is at least ten times faster than CNN and does not require a GPU, LightGBM is a better choice for all practical purposes.

Figure 2 shows Δb obtained with ANNb and LightGBMb algorithms by using *DATASET2*. For both training data and testing data generated from the same parameter set, Δb are about 0.4–0.45 fm which is slightly larger than the corresponding values displayed in Fig. 1. We observe the same trend that the diagonal numbers are smaller than off diagonal. In addition, Δb for training and testing data generated from parameter sets with the same mean-field potential but different σ_{NN} are also larger than other cases, indicating again σ_{NN} has a stronger effect than the mean-field potential. However,

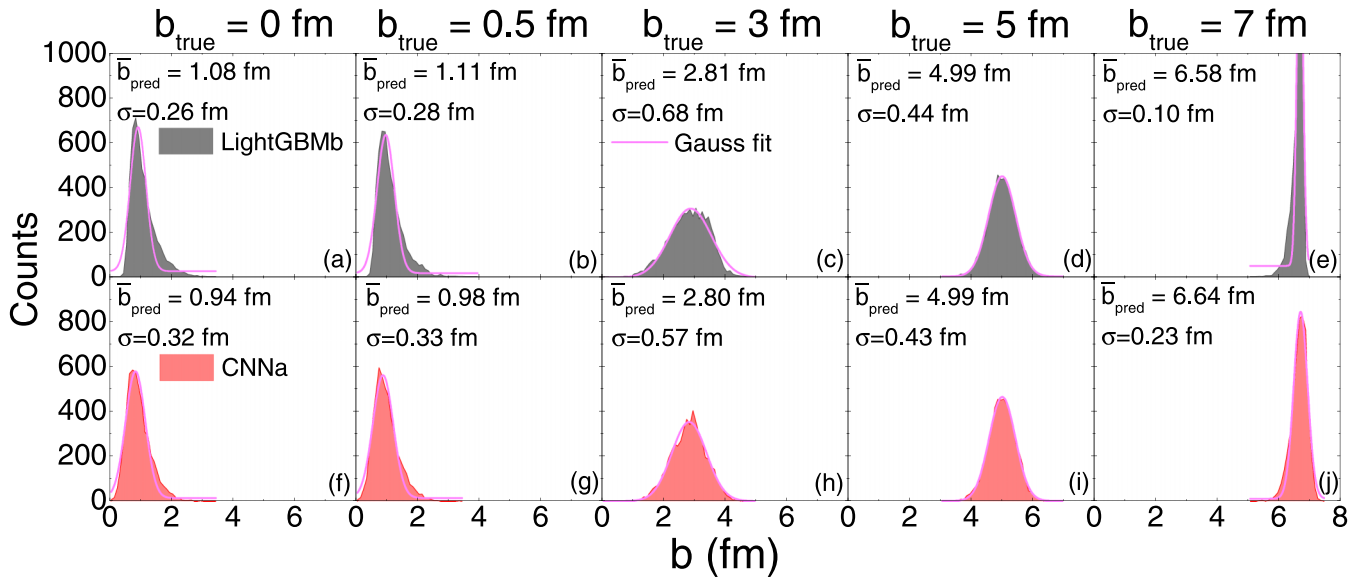

FIG. 3. The distribution of the predicted impact parameter from LightGBMb and CNNa algorithms. Both the training data and testing data are generated with SM-I mode. 5000 ^{132}Sn + ^{124}Sn collision events for each impact parameter (from left panel to right, $b = 0$, 0.5, 3, 5, and 7 fm) are tested. The pink lines represent the Gaussian fitting ($y = y_0 + \frac{A}{\sigma\sqrt{2\pi}} \exp^{-} \frac{(x-\mu)^2}{2\sigma^2}$) of the distribution. $\bar{b}_{\rm pred}$ and σ represent the averaged value of the predicted b and its standard deviation, respectively.

even for training data and testing data generated from these different parameter sets, Δb is still smaller than 0.8 fm obtained from CNNa even in the worst case.

Regarding the influence of the cluster algorithm, by using the LightGBMb algorithm trained with data from SM-F, SM-I, HM-F, and HM-I, the Δb are 0.72, 0.50, 0.70, and 0.51 fm for testing data obtained with SM-I(MST), respectively. Overall, Δb is smaller than 0.8 fm regardless of the model parameters or cluster recognition algorithms sets used to generate data.

B. Impact parameter dependence

It is observed that Δb depends on the impact parameter, and Δb is larger in central collisions [31,34,35]. Figure 3 shows the distributions of the predicted impact parameter obtained with LightGBMb (top panels) and CNNa (bottom panels) algorithms. Above 1 fm, the averaged value of $\bar{b}_{\rm pred}$ is close to the true value. For $b_{\rm true} = 0$ and 0.5 fm, $\bar{b}_{\rm pred}$ are about 1.0 fm, much larger deviations from $b_{\rm true}$. The random nucleon-nucleon collision processes are much more abundant when b is small, therefore fingerprint of the impact parameter on various observables might be washed out by the stochastic process. If the outcomes of collisions with $b = 0$ and 1 fm are naturally indistinguishable, but collisions with $b > 1$ fm are distinguishable, the $b_{\rm pred}$ for events with $b_{\rm true} < 1$ fm given by the ML algorithm would be close to 1 fm in order to get the smallest global loss, because $b_{\rm true}$ varies from 0 to 7 fm. When $\bar{b}_{\rm pred}$ obtained from LightGBMb and CNNa are compared, the latter performs much better in the most central collisions.

C. Explanation of the LightGBM algorithm

LightGBM is very explainable whereas CNN is often treated as a black box. Explainable ML algorithms are usually preferred, especially when they are applied to solve physical problems [65,66] because understanding what happens when ML algorithms make predictions could help us make better use of the outputs. To understand how the LightGBM algorithm gives a particular result and to develop insight into what the ML algorithm has learned, *feature importance* technology of LightGBM and the method of Shapley additive explanations (SHAP) [67] are applied to show which features have the greatest effect on the determination of impact parameter.

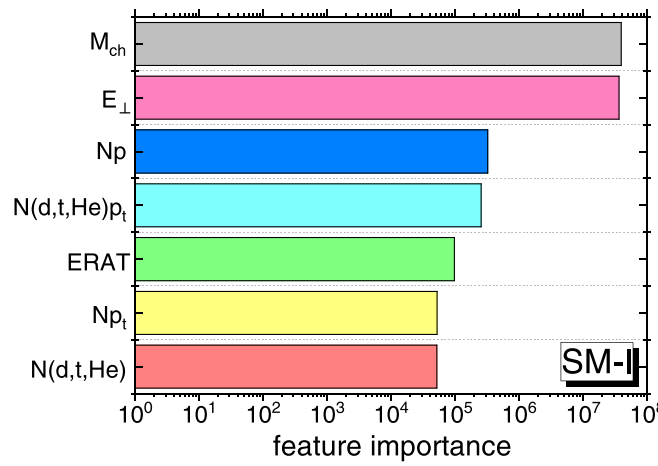

FIG. 4. The importance of the seven features obtained with the *feature importance* technology of LightGBM algorithm.

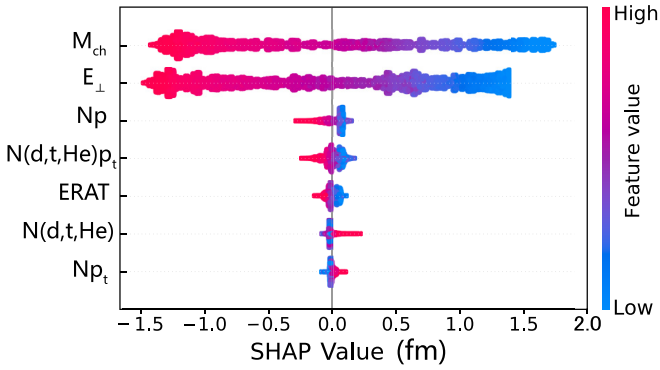

FIG. 5. Importance ranking for the seven input features obtained with SHAP package. Each row represents a feature, and the x axis is the SHAP value, which shows how important a feature is for a particular prediction. Each point represents a sample, and the color represents feature value (red is high, blue is low).

Figures 4 and 5 illustrate the ranking of importance of the seven features. In both figures, M_{ch} and $E_\perp$ are ranked as the two most important features, while the importance values of the other five features are similar and very weak. To understand the feature importance, the correlations between the impact parameter and the seven input features are plotted in Fig. 6. The impact parameter b are much more correlated with M_{ch} and $E_\perp$ than the others, which implies that they can serve as good candidates for determining b. In addition, we calculate their Pearson correlation coefficient (PCC). PCC is often used to measure the linear correlation between two variables in statistics. PCCs between b and M_{ch}, as well as b and $E_\perp$ are close to 1, implying a strong linear correlation between them. Meanwhile, one may find some inconsistencies in Figs. 4–6 and Table IV. For example, PCC between b and $N(d,t,\mathrm{He})$ is the third largest one, but $N(d,t,\mathrm{He})$ ranks as almost the most irrelevant feature in Figs. 4 and 5. This could occur if the ML algorithm learns not only the linear but also the nonlinear multifaceted relationship between the output and the input features.

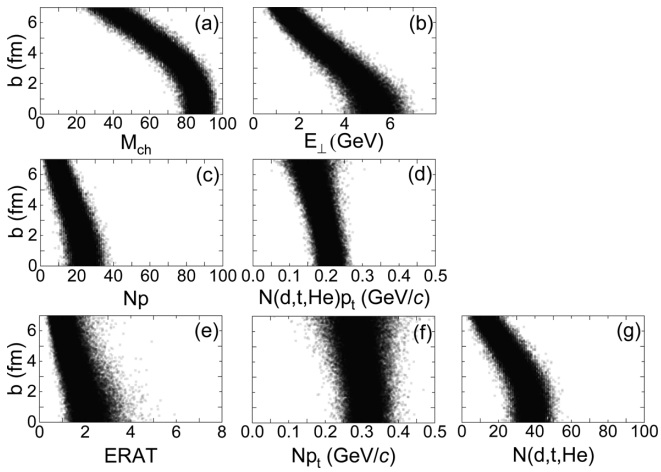

FIG. 6. The correlations between the impact parameter and the seven input features.

TABLE IV. The Pearson correlation coefficient among the seven features and the impact parameter.

	$E_\perp$	M_{ch}	$N(d,t,\mathrm{He})$	Np	$N(d,t,\mathrm{He})p_t$	ERAT	Np_t
b	0.94	0.93	0.86	0.82	0.68	0.67	0.29

SHAP is a model interpretation package developed by Python that interprets the output of ML model. For each test sample, the predicted impact parameter for the ith sample can be obtained with $b_{pred,i} = b_{\mathrm{base}} + f(x_{i1}) + f(x_{i2}) + \cdots + f(x_{ik})$, where $f(x_{ik})$ is the SHAP value of feature x_{ik}. Here x_{ik} represents the value of the kth input feature of the ith sample, and b_{base} is the mean value of the all samples. $f(x_{ik})$ represents the contribution of feature x_{ik} to the prediction $b_{pred,i}$; it tells us how to fairly distribute the prediction among the features. For a certain sample, the larger $f(x_{ik})$ is, the more important is x_{ik}. In the present work, as the output b is uniformly distributed from 0 to 7 fm, the b_{base} is about 3.49 fm. Figure 7 displays the contribution of each feature to a certain prediction. The results for five random samples for each impact parameter ($b = 1$ and 7 fm) are displayed. When the impact parameter is less (greater) than b_{base}, the SHAP value of each feature is basically negative (positive). As observed in Fig. 6, for a smaller b, values of both M_{ch} and $E_\perp$ are larger. It is understandable as more particles and transverse energy maybe produced from the more central collision. This is also the reason why the SHAP values of the red dots (samples with high values of M_{ch} and $E_\perp$) in the first two rows are more negative in Fig. 5.

Overall, it can be found that M_{ch} and $E_\perp$ are the two most important input features for reconstructing the impact parameter while $N(d,t,\mathrm{He})$ and Np_t are listed as being the most

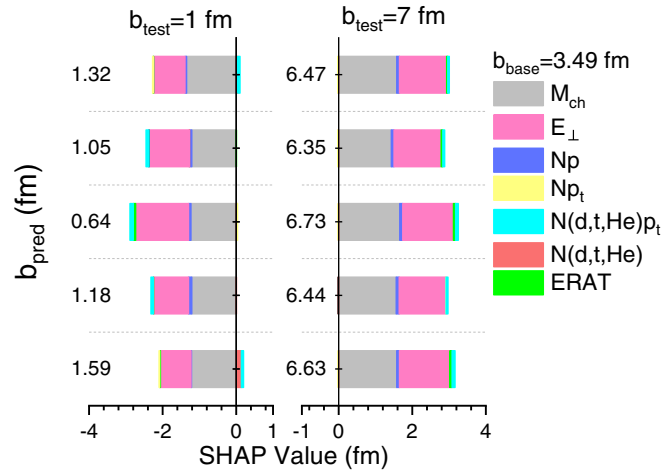

FIG. 7. The contribution of each feature to a prediction (b_{pred}) obtained with the SHAP algorithm, which pushes the prediction of the model from the base value to the final value (model output b_{pred}). The base value is the mean value of the model predicted value on the training data, here $b_{\mathrm{base}} = 3.49$ fm. Results from five random events for each tested impact parameter ($b_{\mathrm{test}} = 1$ and 7 fm) are illustrated as examples.

irrelevant features. Based on the ranking importance, we can reduce the number of features by taking a subset of the most important features. We have checked that the performance does not change if $N(d, t, \mathrm{He})$ and Np_t are not included as features in the training.

V. SUMMARY AND OUTLOOK

In this work, three popular ML algorithms, ANN, CNN and LightGBM, are applied to determine the impact parameter by using either the proton spectra or seven features generated with the UrQMD model. To test the generalizability of the trained ML algorithms, four different UrQMD model parameter sets are applied to generate the data. It is found that the mean absolute error between the true impact parameter and the estimated one Δb can be smaller than 0.45 fm if training and test sets are taken from the UrQMD model with the same parameter set, while Δb increases to 0.8 fm if the training and testing data are taken from different parameter sets in the UrQMD model. Furthermore, the feature importance is obtained with LightGBM algorithm based on *feature importance* technology and SHAP. The total number of charged particles M_{ch} and the transverse kinetic energy $E_\perp$ for light charged particles are the two most relevant features for determining the impact parameter, and this can be understood from the distribution of impact parameter as functions of M_{ch} and $E_\perp$.

The generalizability of the trained ML algorithms is tested by using training and testing data generated from different model parameter sets. Although observables are strongly affected by the model parameters, the extracted Δb is still smaller than 0.8 fm, implying the trained ML algorithms are robust approaches. This gives us confidence that the trained ML algorithms with data generated by theoretical models can be applied to determine the impact parameter in real experimental data as shown in Ref. [68].

ACKNOWLEDGMENTS

The authors acknowledge support in the form of access to the computing server C3S2 in Huzhou University. The work is supported in part by the National Natural Science Foundation of China, Grants No. U2032145, No. 11875125, and No. 12047568; the National Key Research and Development Program of China under Grant No. 2020YFE0202002; and the "Ten Thousand Talent Program" of Zhejiang province (No. 2018R52017). This work was partly supported by the U.S. Department of Energy under Grants No. DE-SC0021235 and No. DE-NA0003908, and by the U.S. National Science Foundation under Grant No. PHY-1565546. The contributions of H.L. are non-Huawei achievements.

[1] B. A. Li, L. W. Chen, and C. M. Ko, Phys. Rep. **464**, 113 (2008).
[2] M. B. Tsang, J. R. Stone, F. Camera, P. Danielewicz, S. Gandolfi, K. Hebeler *et al.*, Phys. Rev. C **86**, 015803 (2012).
[3] J. Xu, Prog. Part. Nucl. Phys. **106**, 312 (2019).
[4] B. A. Li, B. J. Cai, L. W. Chen, and J. Xu, Prog. Part. Nucl. Phys. **99**, 29 (2018).
[5] B. A. Li, P. G. Krastev, D. H. Wen, and N. B. Zhang, Eur. Phys. J. A **55**, 117 (2019).
[6] M. Colonna, Prog. Part. Nucl. Phys. **113**, 103775 (2020).
[7] A. Ono, Prog. Part. Nucl. Phys. **105**, 139 (2019).
[8] G. Jhang, J. Estee, J. Barney *et al.*, Phys. Lett. B **813**, 136016 (2021).
[9] A. Andronic, J. Lukasik, W. Reisdorf, and W. Trautmann, Eur. Phys. J. A **30**, 31 (2006).
[10] J. Adamczewski-Musch *et al.* (HADES Collaboration), Eur. Phys. J. A **54**, 85 (2018).
[11] J. D. Frankland *et al.* (INDRA Collaboration), arXiv:2011.04496.
[12] S. J. Das, G. Giacalone, P. A. Monard, and J. Y. Ollitrault, Phys. Rev. C **97**, 014905 (2018).
[13] P. Li, Y. Wang, Q. Li, J. Wang, and H. Zhang, J. Phys. G **47**, 035108 (2020).
[14] Y. LeCun, Y. Bengio, and G. Hinton, Nature (London) **521**, 436 (2015).
[15] Z. A. wang, J. Pei, Y. Liu, and Y. Qiang, Phys. Rev. Lett. **123**, 122501 (2019).
[16] P. Morfouace, C. Y. Tsang, Y. Zhang *et al.*, Phys. Lett. B **799**, 135045 (2019).
[17] L. Yang, C. J. Lin, Y. X. Zhang *et al.*, Phys. Lett. B **807**, 135540 (2020).
[18] J. E. Bernhard, J. S. Moreland, and S. A. Bass, Nat. Phys. **15**, 1113 (2019).
[19] L. G. Pang, K. Zhou, N. Su, H. Petersen, H. Stöcker, and X. N. Wang, Nat. Commun. **9**, 210 (2018).
[20] H. Liu, D. Han, Y. Ma, and L. Zhu, Sci. China Phys. Mech. Astron. **63**, 112062 (2020).
[21] C. Bertulani, Sci. China Phys. Mech. Astron. **63**, 112063 (2020).
[22] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborova, Rev. Mod. Phys. **91**, 045002 (2019).
[23] A. Radovic, M. Williams, and D. Rousseau *et al.*, Nature (London) **560**, 41 (2018).
[24] M. R. Hush, Science **355**, 580 (2017).
[25] J. Carrasquilla and R. Melko, Nat. Phys. **13**, 431 (2017).
[26] J. Steinheimer, L. Pang, K. Zhou *et al.*, J. High Energy Phys. **12** (2019) 122.
[27] Y. L. Du, K. Zhou, J. Steinheimer *et al.*, Eur. Phys. J. C **80**, 516 (2020).
[28] K. Zhou, G. Endrődi, L. G. Pang, and H. Stöcker, Phys. Rev. D **100**, 011501(R) (2019).
[29] Y. Kvasiuk, E. Zabrodin, L. Bravina *et al.*, J. High Energy Phys. **07** (2020) 133.
[30] C. David, M. Freslier, and J. Aichelin, Phys. Rev. C **51**, 1453 (1995).
[31] S. A. Bass, A. Bischoff, J. A. Maruhn, H. Stocker and W. Greiner, Phys. Rev. C **53**, 2358 (1996); S. A. Bass, A. Bischoff, C. Hartnack *et al.*, J. Phys. G **20**, L21 (1994).
[32] F. Haddad, K. Hagel, J. Li *et al.*, Phys. Rev. C **55**, 1371 (1997).
[33] J. De Sanctis, M. Masotti, M. Bruno *et al.*, J. Phys. G **36**, 015101 (2009).

[34] M. Omana Kuttan, J. Steinheimer, K. Zhou, A. Redelbach, and H. Stoecker, Phys. Lett. B **811**, 135872 (2020).
[35] F. Li, Y. Wang, H. Lü *et al.*, J. Phys. G **47**, 115104 (2020).
[36] S. A. Bass, M. Belkacem, M. Bleicher *et al.*, Prog. Part. Nucl. Phys. **41**, 255 (1998).
[37] M. Bleicher, E. Zabrodin, C. Spieles *et al.*, J. Phys. G **25**, 1859 (1999).
[38] Q. Li, C. Shen, C. Guo, Y. Wang, Z. Li, J. Lukasik, and W. Trautmann, Phys. Rev. C **83**, 044617 (2011).
[39] Q. Li, G. Graf, and M. Bleicher, Phys. Rev. C **85**, 034908 (2012).
[40] Q. Li and Z. Li, Sci. China Phys. Mech. Astron. **62**, 972011 (2019).
[41] Y. Wang and Q. Li, Front. Phys. (Beijing) **15**, 44302 (2020); Y. Zhang, N. Wang, Q. F. Li *et al.*, *ibid.* **15**, 54301 (2020).
[42] J. Aichelin, Phys. Rep. **202**, 233 (1991).
[43] Q. Li, Z. Li, S. Soff, M. Bleicher, and H. Stoecker, J. Phys. G **32**, 151 (2006).
[44] C. Hartnack, R. K. Puri, J. Aichelin *et al.*, Eur. Phys. J. A **1**, 151 (1998).
[45] L. Tong, P. Li, F. Li *et al.*, Chin. Phys. C **44**, 074103 (2020).
[46] H. Yu, D. Q. Fang, and Y. G. Ma, Nucl. Sci. Tech. **31**, 61 (2020).
[47] Z. Q. Feng, Nucl. Sci. Tech. **29**, 40 (2018).
[48] Y. Wang, C. Guo, Q. Li *et al.*, Phys. Lett. B **778**, 207 (2018).
[49] C. Hartnack, H. Oeschler, and J. Aichelin, Phys. Rev. Lett. **96**, 012302 (2006).
[50] Z. Q. Feng, Phys. Rev. C **83**, 067604 (2011).
[51] A. Le Fèvre, Y. Leifels, W. Reisdorf, J. Aichelin, and C. Hartnack, Nucl. Phys. A **945**, 112 (2016).
[52] P. Danielewicz, R. Lacey, and W. G. Lynch, Science **298**, 1592 (2002).
[53] P. Li, Y. Wang, Q. Li, and H. Zhang, Nucl. Sci. Tech. **29**, 177 (2018).
[54] P. Li, Y. Wang, Q. Li, C. Guo, and H. Zhang, Phys. Rev. C **97**, 044620 (2018).
[55] C. A. Ogilvie, D. A. Cebra, J. Clayton, S. Howden, J. Karn, A. VanderMolen, G. D. Westfall, W. K. Wilson, and J. S. Winfield, Phys. Rev. C **40**, 654 (1989).
[56] L. Phair, D. R. Bowman, C. K. Gelbke *et al.*, Nucl. Phys. A **548**, 489 (1992).
[57] J. P. Sullivan, J. Péter, and D. Cussol *et al.* Phys. Lett. B **249**, 8 (1990).
[58] W. Reisdorf *et al.*, (FOPI Collaboration), Nucl. Phys. A **876**, 1 (2012).
[59] D. Rumelhart, G. Hinton, and R. Williams, Nature (London) **323**, 533 (1986).
[60] J. Schmidhuber, Neural Networks **61**, 85 (2015).
[61] S. Ioffe and C. Szegedy, in *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, ICML'15 Vol. 37 (JMLR.org, 2015), pp. 448–456.
[62] S. Nitish, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, J. Mach. Learn. Res. **15**, 1929 (2014).
[63] G. Ke, Q. Meng, T. Fianley *et al.*, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2017), pp. 3149–3157.
[64] LightGBM, https://lightgbm.readthedocs.io/en/latest/index.html.
[65] F. Emmert-Streib, O. Yli-Harja, and M. Dehmer, WIREs Data Mining Knowl. Discovery **10**, e1368 (2020).
[66] F. Y. Xu, H. Uszkoreit, Y. Z. Du *et al.*, in *Natural Language Processing and Chinese Computing, NLPCC 2019*, Lecture Notes in Computer Science, Vol. 11839 (Springer, Cham, 2019), pp. 563–574.
[67] S. Lundberg and S. I. Lee, *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems* (2017), pp. 4768–4777.
[68] C. Y. Tsang, Y. Wang, M. B. Tsang *et al.*, arXiv:2107.13985.