Deep learning for flow observables in ultrarelativistic heavy-ion collisions

H. Hirvonen, K. J. Eskola, and H. Niemi
University of Jyväskylä, Department of Physics, P.O. Box 35, FI-40014 University of Jyväskylä, Finland and Helsinki Institute of Physics, P.O. Box 64, FI-00014 University of Helsinki, Finland

(Received 28 March 2023; accepted 28 June 2023; published 13 September 2023)

We train a deep convolutional neural network to predict hydrodynamic results for flow coefficients, average transverse momenta, and charged particle multiplicities in ultrarelativistic heavy-ion collisions from the initial energy density profiles. We show that the neural network can be trained accurately enough so that it can reliably predict the hydrodynamic results for the flow coefficients and, remarkably, also their correlations like normalized symmetric cumulants, mixed harmonic cumulants, and flow-transverse-momentum correlations. At the same time the required computational time decreases by several orders of magnitude. To demonstrate the advantage of the significantly reduced computation time, we generate 10^7 initial energy density profiles from which we predict the flow observables using the neural network, which is trained using 5×10^3, and validated using 9×10^4 events per collision energy. We then show that increasing the number of collision events from 9×10^4 to 10^7 can have significant effects on certain statistics-expensive flow correlations, which should be taken into account when using these correlators as constraints in the determination of the quantum chromodynamics matter properties.

DOI: 10.1103/PhysRevC.108.034905

I. INTRODUCTION

Probing the properties of the strongly interacting matter close to a zero net-baryon density is the primary goal of the highest-energy ultrarelativistic heavy-ion collision experiments. One of the most important tools in interpreting the experimental data is relativistic hydrodynamics. In the hydrodynamic limit the matter behavior is controlled by the matter properties like equation of state and transport coefficients, such as shear and bulk viscosity. It has been well established that in heavy-ion collisions flow-like signatures are seen in azimuthal angle spectra of produced particles. This indicates that a small droplet of deconfined phase of quantum chromodynamics (QCD) matter called quark-gluon plasma (QGP) is created in these collisions, and that it exhibits a fluid-like behavior [1–4].

Comparing the measurements with the predictions of hydrodynamic computations gives then a possibility to determine the QCD matter properties. A reliable estimate of the QCD matter properties with well-defined error bars demands a global analysis of as many experimental observables and collision systems as possible. In the recent years, such global analyses have given constraints on the QCD transport properties [5–14]. In particular, the shear viscosity near the QCD transition temperature $T \approx 155$ MeV is rather well constrained. For the full temperature dependence of shear viscosity, and especially bulk viscosity, the uncertainties are significantly larger.

A way to improve the analysis is to consider more observables. One challenge here is that in practice it is necessary to compute the hydrodynamic evolution event by event, i.e., for each collision event separately, so that the computed observables are obtained as averages over a large number of collisions to closely match with the actual measurements. The nontrivial dependence of the final observables on the equation of state, transport coefficients, initial conditions, and the details of the conversion of the fluid to particles together with numerically demanding hydrodynamic simulations makes the global analysis a very CPU intensive task. In particular, this is the case when the global analysis takes into account observables that require high statistics obtained by accumulating a large number of computed collision events.

The most basic experimental observables quantifying the magnitude and details of the flow-like behavior are the Fourier coefficients of an azimuthal hadron spectrum, which are usually referred to as flow coefficients v_n. They are measured as multiparticle correlations. The increased luminosity in recent measurements, especially at the CERN Large Hadron Collider (LHC), has enabled precision measurements of multiparticle correlations between flow coefficients all the way up to the eight-particle level. Obtaining reliable estimates of these correlations from the fluid dynamical simulation can require gathering statistics from about 10^6 collision events. Obtaining such high statistics is computationally very expensive and performing computations gets even more expensive when sampling the $\mathcal{O}(15)$-dimensional parameter space of a global analysis, where statistics should be obtained for around 300 different parametrizations. Typically one event needs about 30 min computing time from a CPU and thus the total

time it would take to perform high statistic global analysis is around $0.5 \times 10^6 \times 300 \approx 10^8$ CPU hours.

One way to decrease the computation time would be to convert the codes to GPU and use a modern GPU based supercomputer to do the computing. Even though this would significantly speed up the simulations, the task would still require a significant amount of computing time. Another possibility is to simplify the complicated fluid dynamical computations and construct fast estimators that can give good estimates of the final state observables from the initial state alone. The simple version of such an estimator for flow coefficients could be constructed, for example, by assuming a linear relation between initial state eccentricities and corresponding flow coefficients. As shown in Refs. [6,15,16] this kind of linear relation works reasonably well for v_2 in central collisions, but nonlinear effects start to get noticeable in more peripheral collisions and even more so in the case of higher-order flow coefficients for which this kind of estimator would not work well even to begin with.

In this article we present a way to estimate p_T-integrated flow observables and correlators directly from the initial energy density profile based on deep convolutional neural networks (CNN). The convolutional neural networks have been proven to be very efficient and accurate tools when it comes to image classification and computer vision tasks. During the past decade, network architectures have evolved towards deeper and deeper networks, i.e., a typical network contains more layers than before. A modern CNN architecture can contain hundreds of layers and tens of millions trainable parameters. Neural networks and deep learning have been utilized before in the context of heavy-ion collisions for various different applications, such as impact parameter estimation, identifying quenched jets, or determination of the QCD matter phase transition [17–21]. In Ref. [22] it was shown that the neural network can also model full hydrodynamic evolution on short time periods, $\Delta\tau \approx 2$ fm, but this kind of method has not yet been applicable for modeling a complete space-time evolution of QGP. The deep neural network was also applied to estimating v_2 from the kinematic information of particles in the context of the AMPT model [23]. However, until the current study, neural networks have not been successfully trained to predict flow observables and correlators from the initial state energy density.

The basic setup here is the perturbative QCD based EKRT (Eskola-Kajantie-Ruuskanen-Tuominen) gluon saturation model [24,25] for the computation of initial conditions that, when supplemented by relativistic hydrodynamic evolution [6,26], gives a good overall description of the available flow data from heavy-ion collisions at the BNL Relativistic Heavy Ion Collider (RHIC) and LHC [27–29]. The neural network constructed here is, however, not restricted to this particular model, but can in principle be applied to any similar framework.

This paper is organized in the following way. In Sec. II we briefly go through the structure of the used neural network and give details about how it is implemented in practice. In Sec. III we validate the accuracy of the neural network by showing that the results obtained by the network match well with the hydrodynamic simulations. The main results are then shown in Sec. IV, where we present the neural network predictions for various different correlators with 10^7 generated collision events. The summary and conclusions are then given in Sec. V.

TABLE I. The structure of the used DenseNet network.

Block	Output size	Layers
Convolution	$134 \times 134 \times 64$	7×7 conv, stride 2
Pooling	$67 \times 67 \times 64$	3×3 max pool, stride 2
Dense block	$67 \times 67 \times 256$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition layer	$67 \times 67 \times 128$ $33 \times 33 \times 128$	1×1 conv 2×2 average pooling, stride 2
Dense block	$33 \times 33 \times 512$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition layer	$33 \times 33 \times 256$ $16 \times 16 \times 256$	1×1 conv 2×2 average pooling, stride 2
Dense block	$16 \times 16 \times 896$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 20$
Transition layer	$16 \times 16 \times 448$ $8 \times 8 \times 448$	1×1 conv 2×2 average pooling, stride 2
Dense block	$8 \times 8 \times 1216$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$
Output layer	$1 \times 1 \times 1216$	8×8 global average pooling
	N_{out}	Fully connected layer with ReLU activation

II. MODEL SETUP

A. DenseNet

The evolution of CNN architectures towards deeper networks has caused challenges to their design [30]. Very deep networks can easily lose some information about the input. Additionally, when propagating the gradient information from the output back to the input, the gradients can start to approach zero. Therefore, the optimizer leaves the network weights close to the input nearly unchanged so that the loss function won't converge to the global minima. This makes the training of a model slow and inaccurate. To solve the vanishing gradient and feature loss problem a dense convolutional network or DenseNet was introduced [31]. The DenseNet consists of two major building blocks: dense blocks and transition layers. The dense block solves the vanishing gradient and feature loss problems by reusing features from the previous layers via concatenation, so that all the proceeding layers in the dense block use feature maps from the previous layers as inputs. This makes it possible to maintain low complexity features while also taking advantage of the deep network's ability to probe very complex features of the training data. Such a property makes the DenseNet a great choice when the data

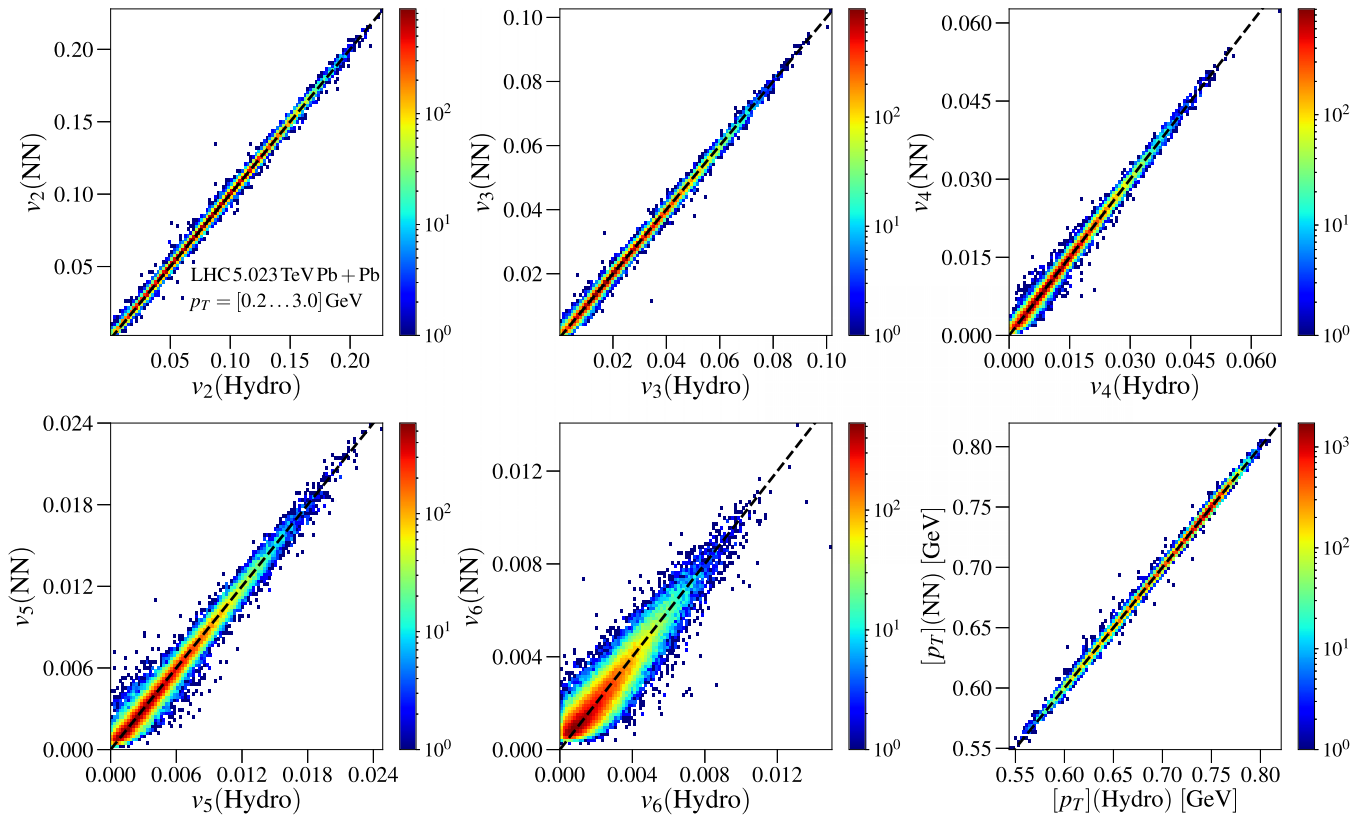

FIG. 1. The event-by-event neural network (NN) predictions versus the results from the hydrodynamic simulations for the validation events in the 0–80% centrality range.

set is somewhat limited and overfitting becomes an issue. The transition layers are then used to reduce the input size. It uses a 1×1 convolutional layer followed by a 2×2 average pooling layer.

In this study we use the DenseNet-BC variant which applies a 1×1 convolutional bottleneck layer before each 3×3 convolution layer in the dense blocks and compression to the transition layer with compression parameter $\theta = 0.5$, which reduces the number of feature-maps by a factor of 2. The growth rate is set to $k = 32$. The DenseNet is originally designed for computer vision tasks and to adapt it to a regression task we change the softmax activation function of the output layer to a linear activation function. The exact structure of the model is shown in Table I, where each convolutional layer contains convolutional layer + batch normalization + ReLU activation. Compared to the original DenseNet model we have changed 3×3 and 7×7 convolution layers with depthwise separable convolution layers, which seems to improve the stability and slightly decrease the validation loss of the model.

B. Implementation

The DenseNet model is trained using midrapidity observables obtained from the hydrodynamic simulations of heavy-ion collisions computed in Ref. [29]. The initial energy density profiles for the hydrodynamic evolution are calculated from the EKRT model [6,26], where the event-by-event fluctuations emerge from the random positions of nucleons inside the colliding nuclei. The computation of the initial profiles is very fast and takes a negligible amount of CPU time compared to the computation of the hydrodynamic evolution and the corresponding physical observables for each event. It is quite easy to generate millions of initial conditions corresponding to different collision events.

As an input, the DenseNet model uses discretized initial energy density in the transverse-coordinate (x, y) plane calculated from the EKRT-model with a grid size 269×269 and a resolution of 0.07 fm. The DenseNet model is trained to reproduce a set of final state p_T integrated observables v_n, average transverse momentum $[p_T]$, and charged particle multiplicity $dN_{\rm ch}/d\eta$ for each event. The input energy density is normalized in such a way that the training data set has a mean of zero and a standard deviation of one.

The DenseNet model gives then a full event-by-event distribution of these observables, and it allows us to build a set of measurable quantities, such as event-averaged N-particle flow coefficients $v_n\{N\}$, normalized symmetric cumulants $NSC(m, n)$, normalized mixed harmonic cumulants $nMHC(n, m)$, and flow-transverse-momentum correlations $\rho(v_n^2, [p_T])$. Note that these observables are different moments of the full $\mathcal{P}(v_n, [p_T], dN_{\rm ch}/d\eta)$ distribution, e.g., two-particle flow coefficient $v_n\{2\}$ is a root-mean-square event average of v_n. It is nontrivial that the network can be trained to a sufficient accuracy to reproduce these observables, the correlators in particular. The definitions of all these

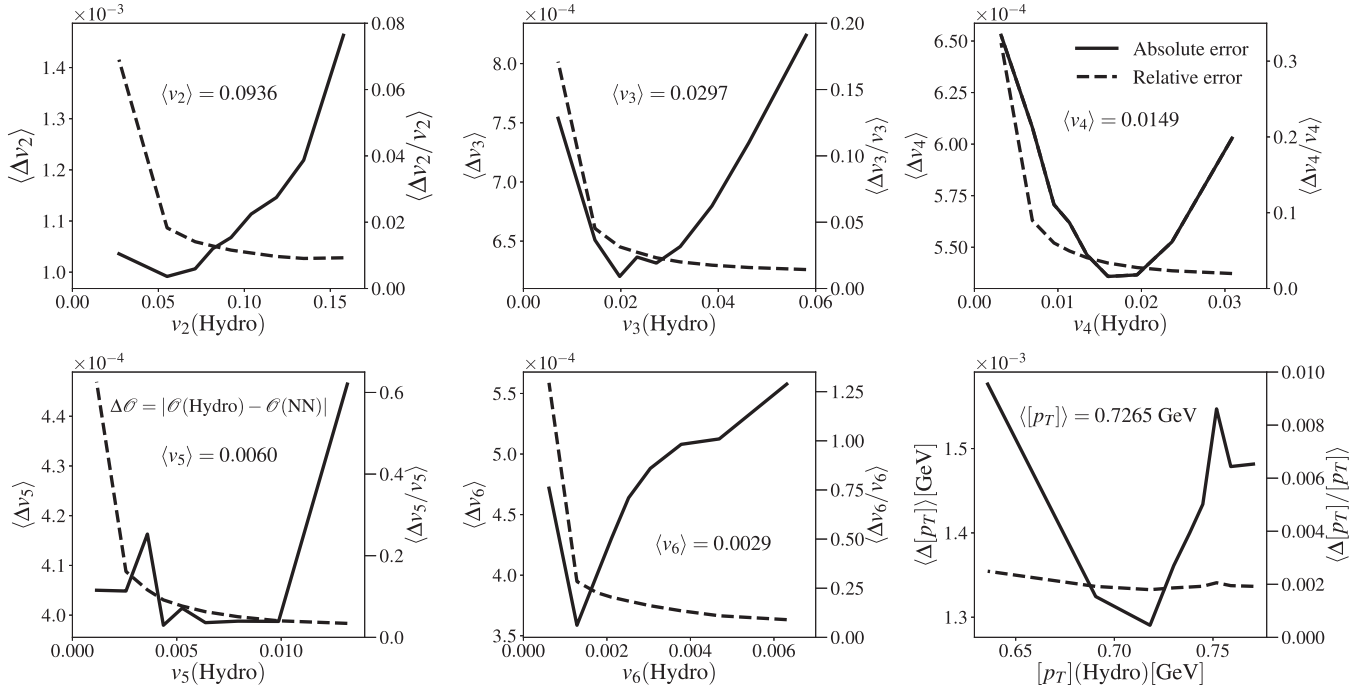

FIG. 2. The mean absolute and relative errors between the neural network predictions and the results from the hydrodynamic simulations for the validation events in the 0–80% centrality range.

observables and the details of the EKRT-model and hydrodynamic computations can be found from Refs. [6,29].

We note that here all the events in a training data set use the same parameters for hydrodynamic evolution, meaning that, currently, the trained neural network cannot predict results from hydrodynamic simulations that use for example different viscosity parametrizations.

We train a separate neural network for each of the flow coefficient v_2, v_3, v_4, v_5, v_6, for the average transverse momentum $[p_T]$, and multiplicity $dN_{\text{ch}}/d\eta$ outputs using in total of 2×10^4 hydrodynamic events in the training. However, one network can give multiple outputs (N_{out} in Table I) of the same observable with different p_T integration ranges. This is necessary since different measurements use different p_T ranges when measuring the observables.

The training events are distributed evenly (5000 events each) between 200 GeV Au + Au, 2.76 TeV Pb + Pb, 5.023 TeV Pb + Pb, and 5.44 TeV Xe + Xe collision systems. The outputs of different neural networks are normalized with a constant such that the typical value of a given output observable is O(1). This makes possible to set the same learning rate for different observables without affecting the quality of the training too much. The exception to this is the charged particle multiplicity network for which the output is not normalized because it uses a different loss function than the other networks. The training data are heavily augmented by applying random rotations (rotation angle from 0 to 2π), flips and translations (shifts from -0.92 fm to 0.92 fm in both x and y directions) to the input during the training.

All the network models above are trained using the Adam optimizer [32] for 120 epochs with a batch size of 64. Using larger batch sizes made the training phase faster, but at the same time significantly decreased the accuracy of the networks. The learning rate is initially set to 0.001, except in the case of the charged particle multiplicity where the initial learning rate is 0.01, and it is divided by a factor of 10 at epochs 75 and 110. Even though the use of a decaying learning rate is not completely necessary because of the adaptive nature of the Adam optimizer, we noticed that adding a learning rate decay made the training faster without sacrificing accuracy. Additionally, the batch normalization momentum is set to 0.1. As a regularization method we tried both the dropout and L2 regularization, but they did not give any improvements for the validation accuracy or made it worse. This is most likely due to a heavy data augmentation which in itself acts as an efficient regularization method.

For all observables except charged particle multiplicity, we use a mean squared error (MSE) loss function which is defined as

$$\text{Loss(MSE)} = \frac{1}{N}\sum_i (y_{\text{i,true}} - y_{\text{i,pred}})^2, \tag{1}$$

where the sum is over all events in the training batch, N is the number of events in a training batch, and $y_{\text{i,true}}$ and $y_{\text{i,pred}}$ are the true and predicted values of an observable, respectively. For the charged particle multiplicity we use a mean squared logarithmic error (MSLE) loss function,

$$\text{Loss(MSLE)} = \frac{1}{N}\sum_i (\ln(y_{\text{i,true}} + 1) - \ln(y_{\text{i,pred}} + 1))^2. \tag{2}$$

The training is done using the Nvidia Tesla V100 GPU, which has 32 GB of VRAM and 640 tensor cores. The training time for one network is ca. 80 min. The neural network code

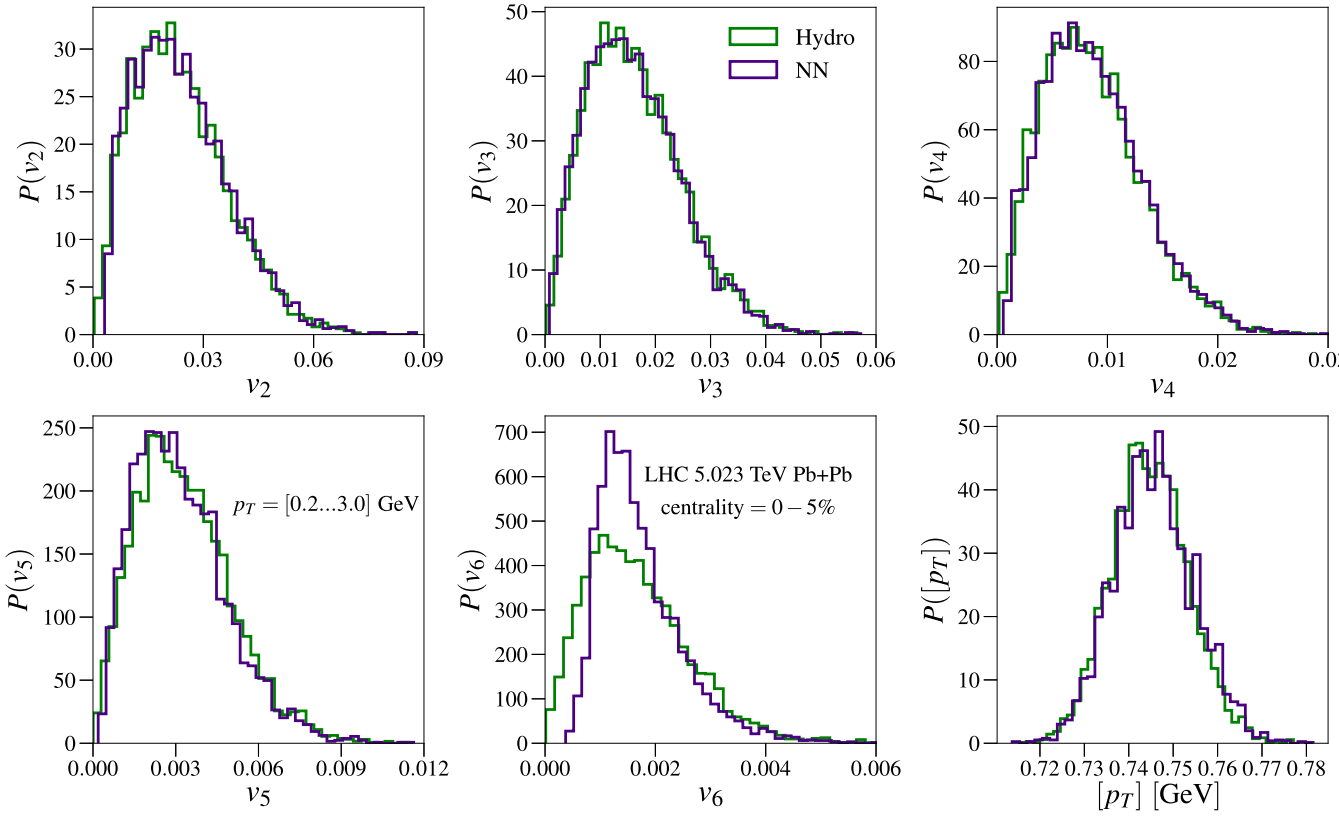

FIG. 3. The distributions of flow observables from the neural network predictions and hydrodynamic simulations for the validation events in the 0–5% centrality range.

is written in PYTHON and it is implemented using the Keras Deep Learning API v2.10.0 [33] together with the Tensorflow v2.10.0 library [34]. The pretrained networks and the code that can be used to generate EbyE flow observables from EKRT-model initial energy density are available as the Supplemental Material [35].

III. VALIDATION

After the training, the accuracy of the neural network needs to be tested with an independent validation data set. Here, we only focus on results for a 5.023 TeV Pb + Pb collision system, but the performance of the neural network is similar for other systems as well. The testing is done by generating 9×10^7 initial energy density profiles and comparing neural network predictions for different observables against those obtained from hydrodynamic simulations. We remind that only 5000 5.023 TeV events were used in the training of the network.

In Fig. 1, we show a two-dimensional (2D) histogram comparing the neural network predictions against hydrodynamic computations event by event for the flow coefficients v_n $(n = 2, 3, 4, 5, 6)$ and average transverse momenta $[p_T]$. The color bar indicates the number of events in each histogram bin and the dashed black line indicates where hydrodynamic computations and neural network predictions match exactly. Because the observables we are interested in are inside the 0–80% centrality range, we only show events from this centrality range in the histogram.

For v_2 we see an excellent agreement between the neural network and hydrodynamic results. The accuracy of the network starts to slowly decrease when moving towards higher-order flow coefficients and in the cases of v_5 and v_6 we already start to see clear deviations from the hydrodynamic results. This behavior is expected since the lower-order flow coefficients and initial-state eccentricities have quite linear dependence and they are not as sensitive to nonlinear effects arising from hydrodynamic evolution as higher-order flow coefficients. For the average transverse momentum the neural network seems to predict the hydrodynamic results very accurately. However, one needs to note that event-by-event fluctuations of $[p_T]$ are very small compared to the absolute value of $[p_T]$. This means that relatively small errors are not necessarily a guarantee of that the network can correctly predict correlations involving $[p_T]$.

To complement the information in Fig. 1 and to give more quantitative estimates of errors, we show the mean absolute and the relative errors for different observables in Fig. 2. Here, we can confirm that the relative error is indeed increasing when increasing the order of the flow coefficients. The errors are not very sensitive to the value of an observable but typically the absolute errors are smallest close to the average value of the observable. We can also notice that the relative error is the largest when the value of an observable is small. The small values of flow coefficients

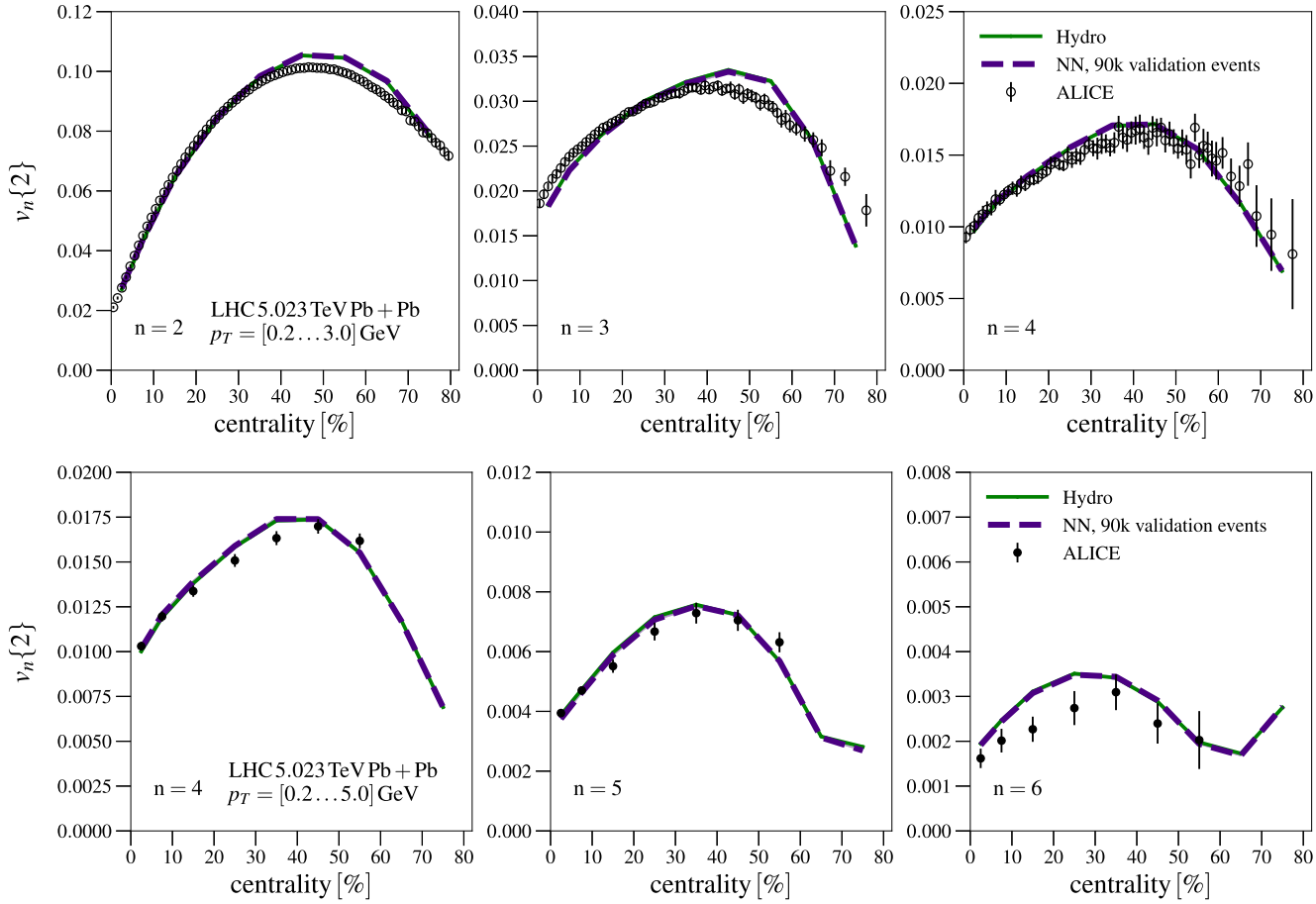

FIG. 4. The comparison of the flow coefficients $v_n\{2\}$ between the neural network predictions and hydrodynamic computations. The experimental data are from the ALICE Collaboration [36,37].

usually correspond to the most central or the most peripheral collisions.

To see where the growing relative errors at the smallest values of v_n start to play a role, we compare distributions of flow observables between the neural network predictions and hydrodynamic computations in the most central collisions. The results are shown in Fig. 3, where we can see that the distributions are nearly identical except for the flow coefficient v_6. In this case the distribution given by the neural network prediction is narrower than the one obtained from the hydrodynamic computation while the location of the peak value is very similar in both cases. This indicates that the neural network might be able to reproduce the average values of v_6 quite well but it

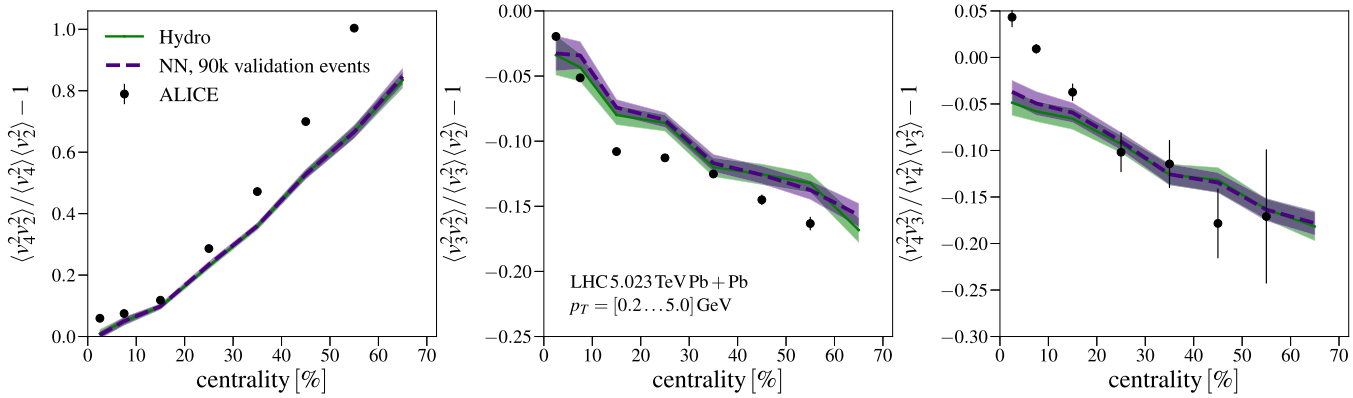

FIG. 5. The comparison of normalized symmetric cumulants between the neural network predictions and hydrodynamic computations. The experimental data are from the ALICE Collaboration [38].

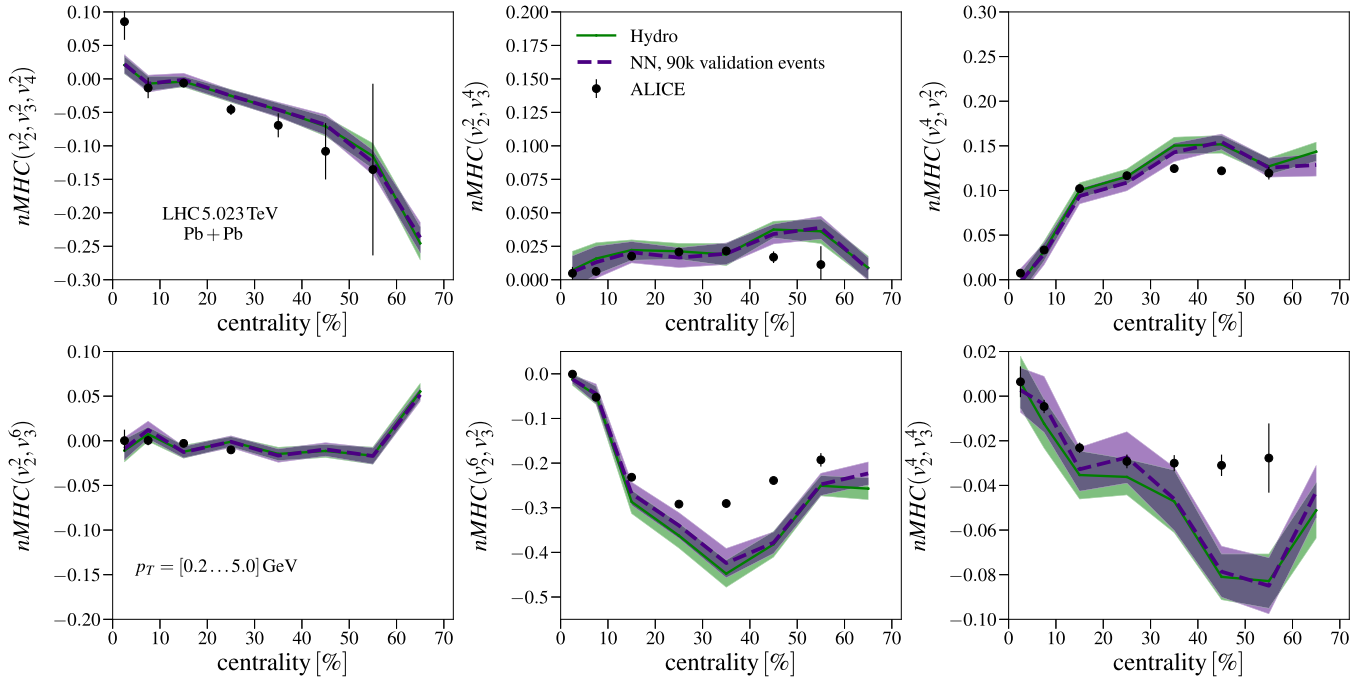

FIG. 6. The comparison of normalized mixed harmonic cumulants between the neural network predictions and hydrodynamic computations. The experimental data are from the ALICE Collaboration [38].

cannot be guaranteed to reliably predict the correlations involving v_6.

Comparing the neural network and hydrodynamic results event by event gives information about the accuracy of the network, but the measurements average over a large number of events in centrality bins. Consequently, it is crucial to test the performance of the network in these cases as well. To get a comprehensive view of the network's ability we check its performance for two-particle flow coefficients $v_n\{2\}$, normalized symmetric cumulants $NSC(m, n)$, normalized mixed harmonic cumulants $nMHC(n, m)$, and flow-transverse-momentum correlations $\rho(v_n^2, [p_T])$.

The flow coefficients $v_n\{2\}$ are shown in Fig. 4 as a function of centrality. We can see that the neural network results seem to match the hydrodynamic results nearly exactly. This is true even in the cases of v_5 and v_6 where the event-by-event accuracy of networks was not as good.

Much more challenging quantities to predict are the different correlations between the flow coefficients. In Fig. 5 we show the centrality dependence of the normalized symmetric cumulants $NSC(m, n)$. The statistical errors are estimated via jackknife resampling as in Ref. [29]. The normalized symmetric cumulants are four-particle correlations between two flow harmonics and thus are more sensitive to event-by-event fluctuations than the flow coefficients $v_n\{2\}$. This makes it more challenging to predict them using the neural network. Nevertheless, in the case of $NSC(4, 2)$ we get an almost exact agreement between the neural network and the hydrodynamic results. For $NSC(3, 2)$ and $NSC(4, 3)$ there are some visible differences between the two, but deviations are still quite small compared to the statistical errors.

The normalized mixed harmonic cumulants $nMHC(n, m)$, which are six- or eight-particle correlations, are shown in Fig. 6. The agreement between the neural network predictions and the hydrodynamic computation is again good, even in the cases where the correlation is very weak. Finally, in Fig. 7, we show the flow-transverse-momentum correlations $\rho(v_n^2, [p_T])$ as a function of the number of participant nucleons. In this observable the biggest challenge for the neural network is not the accuracy of the flow coefficients as one might naively expect, but instead the accuracy of the mean transverse momentum. This is due to the fact that the correlation is very sensitive to the mean transverse momentum fluctuations and, as discussed earlier, catching these fluctuations requires a very good precision from the neural network. Nevertheless, as can be seen from Fig. 7, the neural network predictions agree well with the hydrodynamic results.

IV. HIGH-STATISTICS PREDICTIONS

Now that the accuracy of the neural network has been established, we can use it to estimate what happens to the above correlations at a high-statistics limit. To do so we generate 10^7 events using the neural network, which takes around 20 h with the GPU. This is a very substantial difference compared to doing full hydrodynamic simulations using CPU, which would take about 5×10^6 CPU hours.

The effect of increased statistics for the normalized symmetric cumulants can be seen in Fig. 8. In the case of $NSC(4, 2)$ we see slight deviations in the most central and peripheral collisions, but the centrality dependence is very similar to the lower statistics hydrodynamic results. This is not surprising since the statistical errors are already relatively small with 9×10^4 events. The situation is quite different for $NSC(3, 2)$ where the statistical errors are of considerable size with 9×10^4 events. Here, we see that

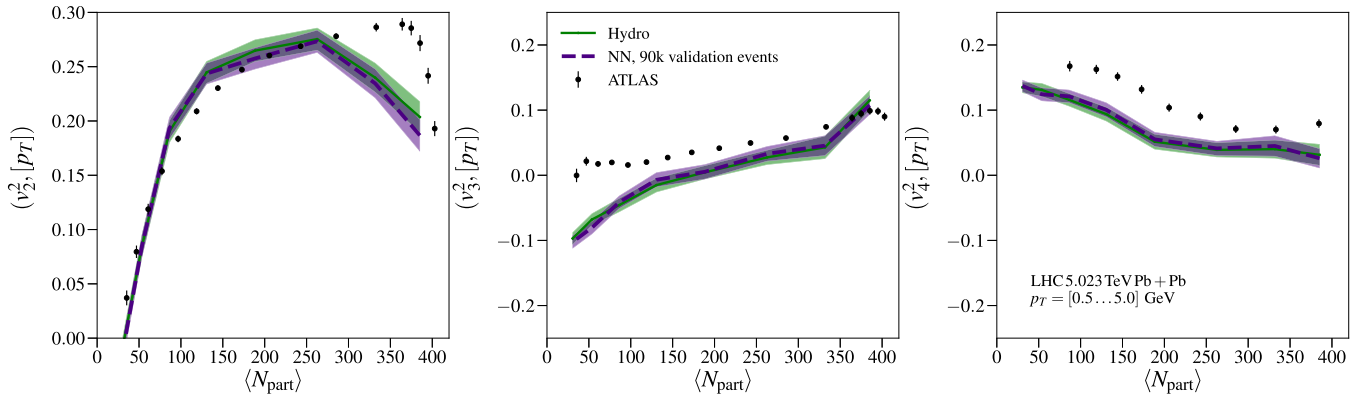

FIG. 7. The comparison of flow-transverse-momentum correlations between the neural network predictions and hydrodynamic computations. The experimental data are from the ATLAS Collaboration [39].

with 10^7 events the statistical fluctuations are negligible, revealing the true centrality dependence from the model, and it now gives a very similar shape as the ALICE measurements, even though the neural network prediction (i.e., the underlying hydrodynamic simulation with which the network was trained) underestimates the amount of anticorrelation. For $NSC(4,3)$ we also see some deviations from the lower-statistics hydrodynamic result in the most central and peripheral collisions. We note that in the most central collisions we see a somewhat similar difference between the neural network and the hydrodynamic result also in the validation data set, which might indicate that this difference can be a systematic error caused by the inaccuracy of the neural network.

In principle, the normalized mixed harmonic cumulants in Fig. 9 should be even more sensitive to the increased event number, since correlations are usually weaker than in the case of the normalized symmetric cumulants. For $nMHC(v_2^2, v_3^4)$ the neural network prediction with 10^7 events is inside the statistical errors of the hydrodynamic results, but in the central collisions the increased number of events reveals a very different kind of centrality dependence which seems to agree well with the ALICE measurements. In the cases of $nMHC(v_2^4, v_3^2)$ and $nMHC(v_2^6, v_3^2)$ we see statistically significant differences between the hydrodynamic results and 10^7 event predictions, which signals that the jackknife resampling can sometimes significantly underestimate the statistical errors. For $nMHC(v_2^2, v_3^6)$ we see that increasing the number of events from 9×10^4 to 10^7 removes the sharp changes between the correlation and anticorrelation and the high statistic result is nearly zero except in the most peripheral collisions. This is again in line with the ALICE measurements.

The flow-transverse-momentum correlations for the 10^7 neural network prediction are shown in Fig. 10. The increased statistics makes it now possible to use exactly the same centrality bins as the ATLAS measurements without completely ruining the accuracy. For $\rho(v_2^2, [p_T])$ the 10^7 event result differs substantially from the 9×10^4 event hydrodynamic result only in the most central collisions. This effect is mostly a combination of different centrality binning and the fact that correlation decreases very quickly when moving from 375 to 400 participants. The effect of statistics can be better seen in the case of $\rho(v_3^2, [p_T])$, where in the central collisions the 10^7 event result has different dependence on participant number than the 9×10^4 event hydrodynamic result. In this region the 10^7 event neural network result also agrees better with the ALICE measurements.

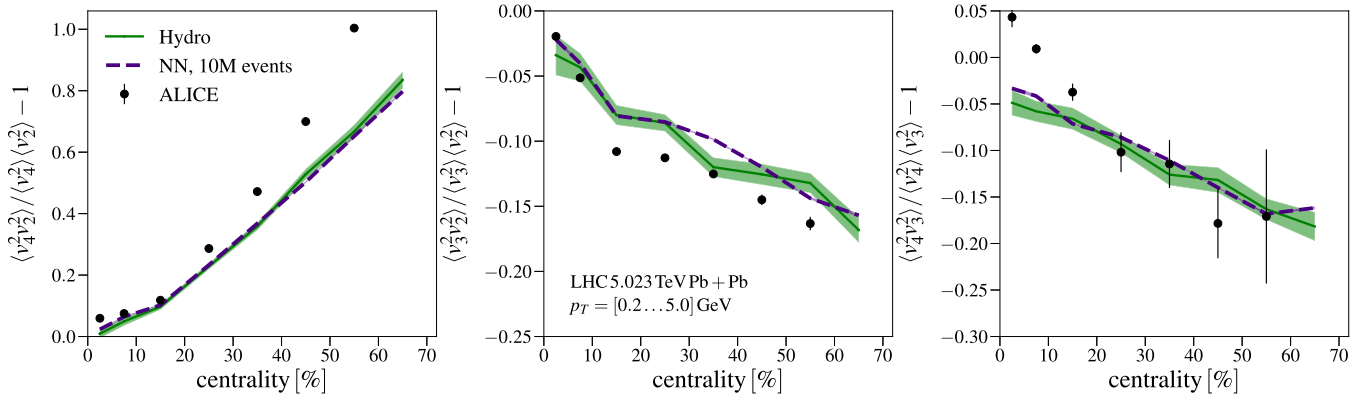

FIG. 8. The neural network prediction of normalized symmetric cumulants with 10^7 collision events compared with the hydrodynamic results from 9×10^4 collision events. The experimental data are from the ALICE Collaboration [38].

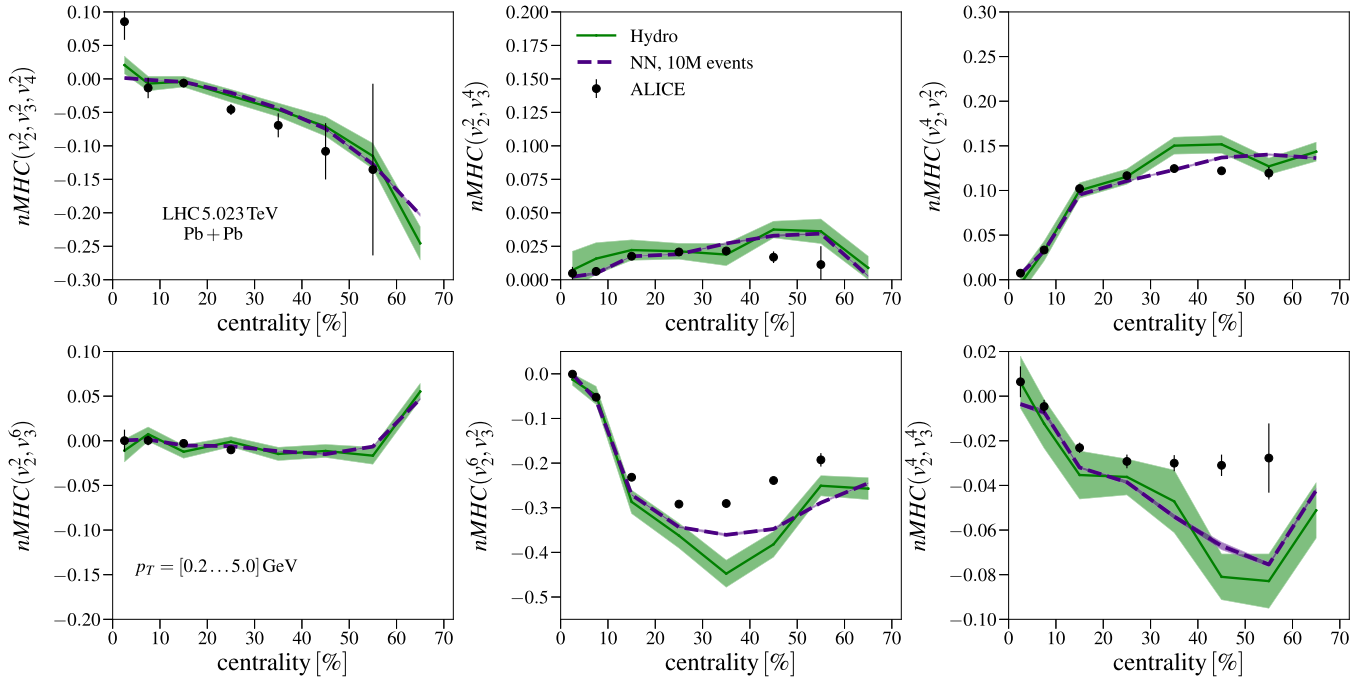

FIG. 9. The neural network prediction of normalized mixed harmonic cumulants with 10^7 collision events compared with the hydrodynamic results from 9×10^4 collision events. The experimental data are from the ALICE Collaboration [38].

V. CONCLUSIONS AND SUMMARY

We have trained a deep convolutional neural network to predict a variety of flow observables from the initial state energy density profiles. The training was done using 2×10^4 training events from 200 GeV Au + Au, 2.76 TeV Pb + Pb, 5.023 TeV Pb + Pb, and 5.44 TeV Xe + Xe collision systems, with 5000 events for each collision system. The training data were computed using viscous relativistic hydrodynamics with initial conditions from the EKRT model, and using the model and viscosity parameters from Ref. [29].

The accuracy of the network was tested against the results from hydrodynamic simulations for two-particle flow coefficients $v_n\{2\}$, normalized symmetric cumulants $NSC(m, n)$, normalized mixed harmonic cumulants $nMHC$, and flow-transverse-momentum correlations $\rho(v_n^2, [p_T])$. We emphasize that this is a nontrivial test for the accuracy of the network, especially with the correlators. The validation tests used in total of 9×10^4 events for each collision system, independent of the training data, and in all of the cases the neural network was able to predict hydrodynamic results quite reliably. This is already a significant improvement in terms of computational time, as only 5000 events were used per collision system to train the network.

The neural network was then used to predict the same flow observables but this time with 10^7 generated events. This took around 20 GPU hours of computing time which is many orders of magnitude faster than doing the same number of hydrodynamic simulations using CPU. The increased number of events made statistical errors negligibly small and allowed us to estimate the observables with a higher precision. In many

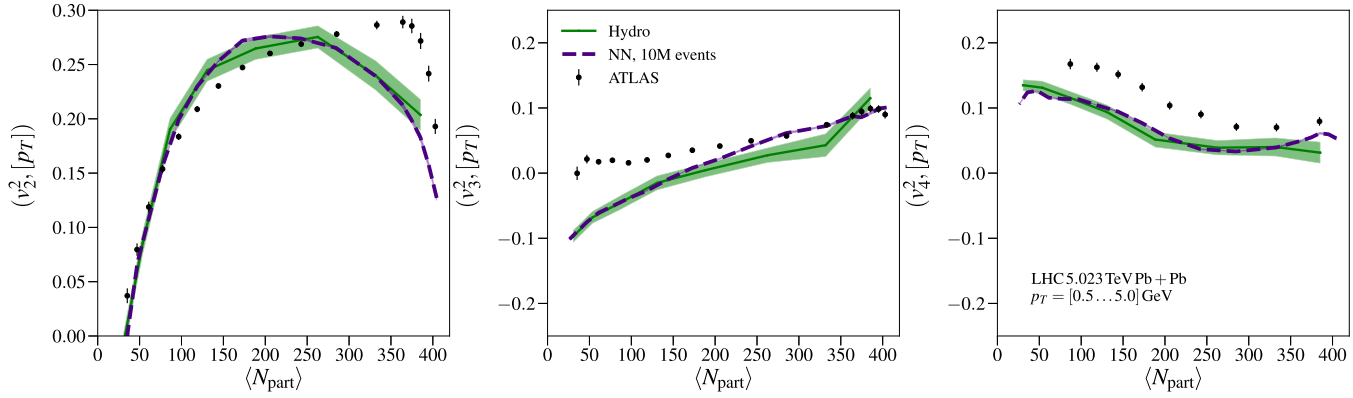

FIG. 10. The neural network prediction of flow-transverse-momentum correlations with 10^7 collision events compared with the hydrodynamic results from 9×10^4 collision events. The experimental data are from the ATLAS Collaboration [39].

cases the 10^7 event neural network prediction differed from the 9×10^4 event hydrodynamic computations by a quite large margin emphasizing the importance of a large event statistics when comparing simulations with the measurements.

As there are still considerable uncertainties in determining QCD matter properties from the experimental data, it is important to be able to use as many measurements as possible to constrain the properties. In particular, the current measurements at the LHC give a wealth of different flow correlations with tight error bars that provide independent information about the matter properties. Many of the measured correlators are rather weak, and can require millions of computed hydrodynamic events in order to get similar statistical errors as in the experiments. To use these quantities as a constraint to the QCD properties, it is then necessary to have a computationally efficient way to generate such a large set of events, and this is exactly what the neural network presented here can do.

Currently the neural network can predict flow observables for different initial energy density profiles, but the predictions always describe a hydrodynamic evolution that is identical to the one used in the training data set, i.e., it is not possible to change the viscosity parametrization after the network has been trained. The network should next be constructed to be more versatile and take viscosity parameters as additional inputs, making the neural network an even more efficient tool in global analysis. This is left as future work.

ACKNOWLEDGMENTS

We acknowledge the financial support from the Jenny and Antti Wihuri Foundation, and the Academy of Finland Project No. 330448 (K.J.E.). This research was funded as a part of the Center of Excellence in Quark Matter of the Academy of Finland (Project No. 346325). This research is part of the European Research Council Project No. ERC-2018-ADG-835105 YoctoLHC. The Finnish IT Center for Science (CSC) is acknowledged for the computing time through Project No. jyy2580.

[1] U. Heinz and R. Snellings, Annu. Rev. Nucl. Part. Sci. **63**, 123 (2013).
[2] C. Gale, S. Jeon, and B. Schenke, Int. J. Mod. Phys. A **28**, 1340011 (2013).
[3] P. Huovinen, Int. J. Mod. Phys. E **22**, 1330029 (2013).
[4] C. Shen, Nucl. Phys. A **1005**, 121788 (2021).
[5] C. Gale, S. Jeon, B. Schenke, P. Tribedy, and R. Venugopalan, Phys. Rev. Lett. **110**, 012302 (2013).
[6] H. Niemi, K. J. Eskola, and R. Paatelainen, Phys. Rev. C **93**, 024907 (2016).
[7] J. E. Bernhard, J. S. Moreland, S. A. Bass, J. Liu, and U. Heinz, Phys. Rev. C **94**, 024907 (2016).
[8] S. A. Bass, J. E. Bernhard, and J. S. Moreland, Nucl. Phys. A **967**, 67 (2017).
[9] J. E. Bernhard, J. S. Moreland, and S. A. Bass, Nat. Phys. **15**, 1113 (2019).
[10] D. Everett *et al.* (JETSCAPE Collaboration), Phys. Rev. C **103**, 054904 (2021).
[11] G. Nijs, W. van der Schee, U. Gürsoy, and R. Snellings, Phys. Rev. C **103**, 054909 (2021).
[12] J. Auvinen, K. J. Eskola, P. Huovinen, H. Niemi, R. Paatelainen, and P. Petreczky, Phys. Rev. C **102**, 044911 (2020).
[13] J. E. Parkkila, A. Onnerstad, and D. J. Kim, Phys. Rev. C **104**, 054904 (2021).
[14] J. E. Parkkila, A. Onnerstad, S. F. Taghavi, C. Mordasini, A. Bilandzic, M. Virta, and D. J. Kim, Phys. Lett. B **835**, 137485 (2022).
[15] F. G. Gardim, F. Grassi, M. Luzum, and J. Y. Ollitrault, Phys. Rev. C **85**, 024908 (2012).
[16] H. Niemi, G. S. Denicol, H. Holopainen, and P. Huovinen, Phys. Rev. C **87**, 054901 (2013).
[17] C. David, M. Freslier, and J. Aichelin, Phys. Rev. C **51**, 1453 (1995).
[18] S. A. Bass, A. Bischoff, J. A. Maruhn, H. Stöcker, and W. Greiner, Phys. Rev. C **53**, 2358 (1996).
[19] P. Xiang, Y. S. Zhao, and X. G. Huang, Chin. Phys. C **46**, 074110 (2022).
[20] L. Liu, J. Velkovska, and M. Verweij, J. High Energy Phys. 04 (2023) 140.
[21] L. G. Pang, K. Zhou, N. Su, H. Petersen, H. Stöcker, and X. N. Wang, Nat. Commun. **9**, 210 (2018).
[22] H. Huang, B. Xiao, Z. Liu, Z. Wu, Y. Mu, and H. Song, Phys. Rev. Res. **3**, 023256 (2021).
[23] N. Mallick, S. Prasad, A. N. Mishra, R. Sahoo, and G. G. Barnaföldi, Phys. Rev. D **105**, 114022 (2022).
[24] K. J. Eskola, K. Kajantie, P. V. Ruuskanen, and K. Tuominen, Nucl. Phys. B **570**, 379 (2000).
[25] R. Paatelainen, K. J. Eskola, H. Holopainen, and K. Tuominen, Phys. Rev. C **87**, 044904 (2013).
[26] R. Paatelainen, K. J. Eskola, H. Niemi, and K. Tuominen, Phys. Lett. B **731**, 126 (2014).
[27] H. Niemi, K. J. Eskola, R. Paatelainen, and K. Tuominen, Phys. Rev. C **93**, 014912 (2016).
[28] K. J. Eskola, H. Niemi, R. Paatelainen, and K. Tuominen, Phys. Rev. C **97**, 034911 (2018).
[29] H. Hirvonen, K. J. Eskola, and H. Niemi, Phys. Rev. C **106**, 044913 (2022).
[30] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, J. Big Data **8**, 53 (2021).
[31] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, arXiv:1608.06993.
[32] D. P. Kingma and J. Ba, arXiv:1412.6980.
[33] F. Chollet *et al.*, https://keras.io/.
[34] M. Abadi *et al.*, arXiv:1603.04467, https://www.tensorflow.org/.
[35] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevC.108.034905 for the pretrained neural networks and the event generation code.
[36] S. Acharya *et al.* (ALICE Collaboration), J. High Energy Phys. 07 (2018) 103.
[37] S. Acharya *et al.* (ALICE Collaboration), J. High Energy Phys. 05 (2020) 085.
[38] S. Acharya *et al.* (ALICE Collaboration), Phys. Lett. B **818**, 136354 (2021).
[39] G. Aad *et al.* (ATLAS Collaboration), Eur. Phys. J. C **79**, 985 (2019).