Convolutional Neural Networks for Particle Diffusometry in the Presence of Flow and with Defocused Particles

Pranshul Sardana[1,*] and Steve Wereley[1]

1: Purdue University, Department of Mechanical Engineering, West Lafayette, USA
*Corresponding author: psardana@purdue.edu

Keywords: Particle Diffusometry, Diffusion Coefficient Measurement, Convolutional Neural Networks, Particle-based measurements, Deep Learning.

ABSTRACT

Diffusion is a natural phenomenon in fluids. Its measurement can be done optically by seeding an otherwise featureless fluid with tracer particles and observing their motion. However, existing algorithms for particle-based diffusion coefficient measurement have multiple failure modes especially when the fluid has a flow or the particles are defocused. We present a method based on Convolutional Neural Networks (CNNs) for predicting diffusion coefficients in the presence of these real-world effects. The CNNs were trained, validated, and tested on crops from four-frame temporally averaged images. The study includes three simulated datasets: Gaussian-shaped particles under no fluid flow condition, Gaussian-shaped particles under an arbitrary flow condition, and defocused particles under no fluid flow condition. The results show that the CNNs have a low Mean Absolute Error (MAE) of $0.12\,\mu m^2/s$ between the true and predicted diffusion coefficient values for the dataset with Gaussian-shaped particles under no fluid flow condition. The CNNs have a slightly higher MAE of $0.19\,\mu m^2/s$ for Gaussian-shaped particles under arbitrary flow and defocused particles under no fluid flow conditions. The performance of the CNNs was also benchmarked against four conventional algorithms on these simulated datasets. The results show that the conventional algorithms perform better than the CNNs when the particles are assumed to be Gaussian and the fluid has no flow. However, the CNNs outperform conventional methods in the presence of flow or if the particles are defocused. Finally, the outputs of CNNs were compared against the outputs of conventional algorithms on experimental datasets. This provides uncertainty in the range of $0.19\,\mu m^2/s$ - $0.47\,\mu m^2/s$, which is slightly larger than the errors obtained from simulated datasets. Hence, the study shows that CNNs can be used to reliably predict diffusion coefficients from complex particle datasets with as little as four frames.

1. Introduction

Diffusion is a process caused by the random motion of molecules. It is quantified by diffusion coefficient (D) and its measurement is known as diffusometry. Real-time diffusometry has applications such as pathogen detection (Clayton et al., 2019; Moehling et al., 2020), nanoparticle characterization (Clayton et al., 2016), and pharmaceuticals (Cagno et al., 2018). However, measuring the diffusion coefficient is challenging as the optical properties of the liquids do not change with D. Optical diffusometry can be performed by seeding this otherwise featureless fluid with reflecting tracer particles and imaging it under a microscope.

These particle images can be processed using conventional methods such as correlation-based techniques (Olsen & Adrian, 2000) and single-particle tracking (Ernst & Köhler, 2012). These techniques can reliably predict the diffusion coefficient if the underlying fluid has no flow or the particles are assumed to be Gaussian-shaped, a common assumption with macro-scale PIV setups. However, removing flow from microfluidic chips is extremely difficult, especially in point-of-care settings. Additionally, microfluidic particle image setups use high magnifications, which create a narrow depth of field but highly illuminated particles which causes a defocus ring around the imaged particles. Recent approaches (Ahmadzadegan et al., 2020) have been proposed to work with non-Gaussian particle shapes. However, their simulated particles followed a Bessel function where the size of the particles in the image does not change with their depth. Hence, the conventional algorithms either have multiple failure modes for microfluidic particle image data processing or have not been tested on defocused particle datasets, and better solutions and testing are needed.

Deep learning (DL) algorithms have recently been used for particle image data processing (Sardana & Wereley, 2023; Cai et al., 2020; Barnkob et al., 2021). These studies often employ convolutional neural networks (CNNs), a subset of DL algorithms. CNNs are specialized units that can extract information from n-dimensional tensors such as particle images. These methods do not assume specific fluid flow types or particle shapes, learning instead from large datasets. In Particle Diffusometry, this approach is termed Deep Particle Diffusometry (DPD) (Sardana & Wereley, 2023). However, these methods have only been applied to Gaussian particles, and their performance on datasets with defocused particles is unknown.

This work extends the current DPD algorithms to datasets with defocused particles. The CNNs were fed PIV-style interrogation crops from a temporally averaged image obtained by averaging consecutive frames ($n > 1$) and provided the diffusion coefficient as output. The mean square error between the real and predicted diffusion coefficients was minimized during training. The CNNs were trained and tested on Gaussian particles with no flow, an arbitrary flow, and defocused particles with no flow conditions. The performance of the CNNs was compared against four conventional algorithms.

The rest of the paper is organized as follows. Section 2 describes the different kinds of datasets created and used in the study. Section 3 describes the CNN training, validation, and testing workflow. This section also describes the architectural and hyperparameter ranges and optimization information used for training the CNNs. Section 4 describes the performance of the CNNs on both simulated and experimental datasets. This section also provides a benchmark of current algorithms against conventional algorithms. Section 5 concludes the current work and provides direction for future work.

2. Creating datasets

The CNNs were trained on simulated datasets and tested on both simulated and experimental datasets. Training was performed on both Gaussian and defocused particles (Rossi, 2019). The simulated datasets contained 271 possibilities for diffusion coefficient ranging from $0.3\,\mu\mathrm{m}^2/\mathrm{s}$ to $3\,\mu\mathrm{m}^2/\mathrm{s}$ with a step size of $0.01\,\mu\mathrm{m}^2/\mathrm{s}$. For defocused particles, the maximum distance from the focal plane was $100\,\mu\mathrm{m}$. As the trained CNNs should predict the diffusion coefficient regardless of flow and particle shape, three families of datasets were created.

First, the Gaussian particles under no flow was the case where the particles were modeled as 2D Gaussians and moved purely because of diffusion. Next, the Gaussian particles under an arbitrary flow was the case where the particles were modeled as 2D Gaussians but could move because of both diffusion and flow. The arbitrary flow dataset had 4 flow conditions: no flow, Uniform flow, Couette flow, and Poiseuille flow. The flows had a maximum velocity of $2\,\mathrm{m}/\mathrm{s}$ in the horizontal direction and no velocity in the vertical direction. This provides a Peclet number from 0 to 2.67×10^6. Finally, the defocused particles under no flow was the case where the particles are defocused based on their distance from the focal plane and move purely because of diffusion. The defocused particles were formed using ray tracing and an approximated model of a spherical lens (Rossi, 2019). The particles closest to the focal plane had a Gaussian-like shape, whereas the particles further from the focal plane had a defocus ring caused by imaging at the out-of-focus location of these particles. Figure 1 shows sample particle shapes obtained from the two simulations. It can be seen that the in-focus particles obtained from the ray-tracing approach look very similar to the Gaussian particles. However, the out-of-focus particles obtained from the ray tracing have a defocus ring around them and can not be assumed as Gaussian-shaped anymore.

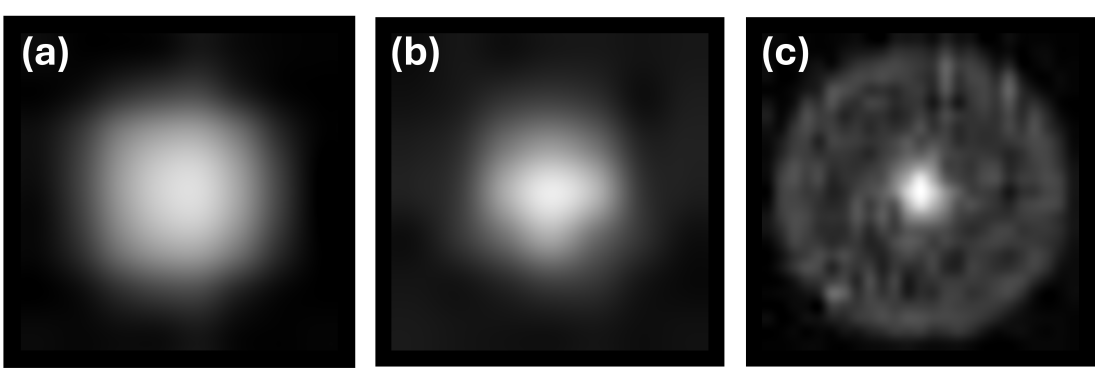

Figure 1. Various simulated particle shapes: (a) Gaussian particle, (b) In-focus particle obtained from the ray-tracing approach, (c) Out-of-focus particle at a distance of $20\,\mu\mathrm{m}$ from the focal plane obtained from the ray-tracing approach.

For each simulation, three sets of particle images were created: one for training, one for validation, and one for testing. The simulated test dataset was also used to benchmark the CNNs against four single-particle and correlation-based algorithms. The CNNs were also tested on Gaussianized experimental videos obtained from 500 nm particle images recorded at 40x magnification (Lee et al., 2022). The experimental sequences were subsampled to obtain multiple sequences for each video, with diffusion coefficients in the range the CNNs were trained.

3. Network Training, Validation and Testing

The CNNs were trained, validated, and tested on simulated particle images. As the ground truth for the simulated datasets is known, the performance of the trained models can be found without any uncertainty.

The CNNs were fed crops from temporally averaged images in all these steps. During training, the CNNs were fed a single random crop from the averaged images. This was done to provide diversity so that the CNN sees the different perturbations of the input and does not learn to map fixed inputs to outputs. The temporally averaged image contained 4 frames so that the CNNs had enough temporal information in the input to distinguish between motion because of flow and motion because of diffusion. Each crop had a size of 256x256 pixels.

The goal during the training process was to learn a generalized mapping from the given temporally averaged image and predict the diffusion coefficient. If the training goes on for too long, the CNNs start to memorize this mapping. This is known as overfitting to the training set and it degrades the performance of the CNNs on new, unseen datasets. Overfitting was avoided in the current datasets by using a validation dataset. A validation step was performed after every training epoch and if the validation loss was more than the minimum validation loss for five consecutive epochs, the training was stopped, and the model obtained from the least validation loss was used as the best model. This also helps speed up the training process as the models being trained on bad hyperparameter combinations are stopped early.

During testing, one needs to be data efficient and deterministic with the results. So, the whole image was broken down into smaller crops, each of which was fed to the CNNs individually, and diffusion coefficient values were obtained. The final diffusion coefficient was the mean of values obtained from the individual predictions. Hence, the entire image can be used to obtain the final diffusion coefficient, without any spatial stochasticity in the predictions. Similarly, the temporal stochasticity of the predictions can be reduced by using multiple sets of averaged images.

The CNNs were trained on 500 hyperparameters and architectural combinations. During the search, the architectural parameters were treated as any other hyperparameter and these were optimized jointly. The initial choices and the search space that were defined are provided in table 1. The next hyperparameter combination was chosen by Tree-structured Parzen Estimator (Bergstra et al., 2011) (TPE) which is a Bayesian method and uses the performance on the current hyperparameter combination to select the next set of hyperparameters. These choices were made to minimize the mean absolute error (MAE) between the true diffusion coefficient values and values predicted by the best model for that hyperparameter combination on the validation dataset.

Table 1. Convolutional Neural Network (CNN) architecture and hyperparameter choices and search space used for training. The architectural parameters are treated as hyperparameters during the search process and the search was performed jointly.

Hyperparameter	Search Type	Range	Initial Value
Learning Rate	Log Uniform	[1e-5 - 1e-3]	1e-4
Batch Size	Choice	2, 4, 8, 16, 32	8
Kernel size of First Convolution Filter	Choice	3, 5, 7	4
Number of Convolutional Layers	Choice	7, 13, 25, 49	13

The learning rate defines the change in each model parameter value during backpropagation. These parameters are internal to the model and are different than the hyperparameters used to train the models. The batch size is the number of inputs used in each iteration of the stochastic gradient descent. The kernel size of the first convolution filter defines the receptive field of the first convolutional layer. This controls the amount of information fed to the CNN. The number of convolutional layers defined the depth of the CNN. The current study aimed to explore a larger range of CNN depths to give the outer loop a choice between shallower and deeper networks.

In each run of the inner loop, a hyperparameter combination was selected and the CNN was trained for a maximum of 40 epochs with an early stop of 5 epochs. The CNNs were trained with the Adam optimizer and the mean squared error (MSE) loss was used to guide the backpropagation.

4. Results

The CNNs were tested on both simulated and experimental datasets. The CNNs were also compared against conventional methods. Finally, the outputs from the CNNs were compared against those from traditional approaches on experimental data to get the prediction uncertainty. 20 frames were used for every test. The CNNs were trained on 4 frame averaged images, hence 5 averaged images were created with a temporal stride of 4 frames, leading to 20 frames.

4.1. Performance on Gaussian Particles under no Flow Condition

The simplest case to solve was Gaussian particles under a no fluid flow condition. Here, the particles had a simple shape that did not change with the distance from the focal plane. The particles move purely because of diffusion, and no other factors change their location. Figure 2 shows the performance of CNNs when trained and tested on these datasets.

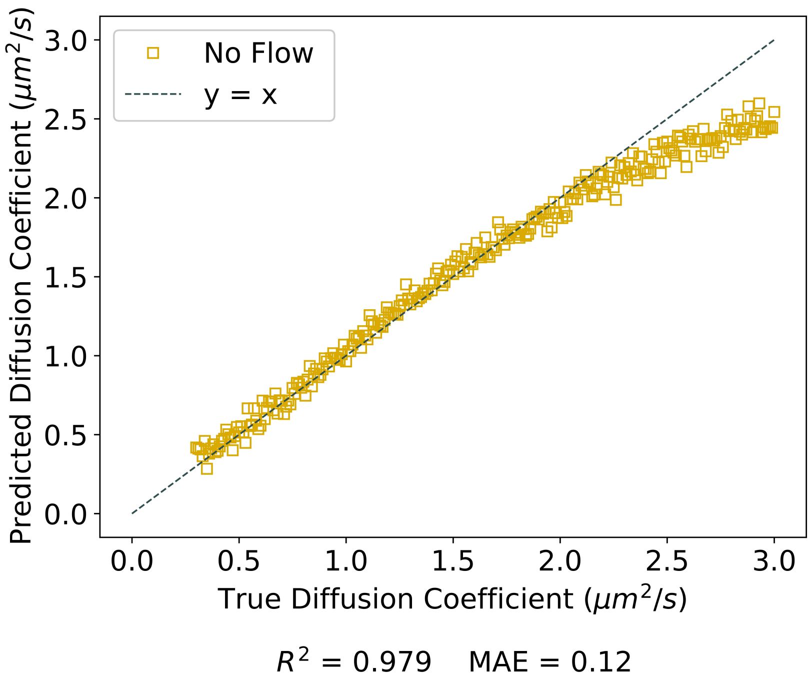

Figure 2. Performance of CNNs when trained and tested with Gaussian particles under a no flow condition.

It can be seen that the CNNs can learn to predict the diffusion coefficient with a high R^2 and a low MAE for this case. The CNN had a close fit for diffusion coefficients below $2.3\,\mu m^2/s$ and under predict for higher diffusion coefficients.

4.2. Performance on Gaussian Particles under Arbitrary Flow Condition

Next, the CNNs were trained and tested on Gaussian particles under the arbitrary flow condition. In this case, the particles change position because of diffusion and one of the four flow conditions. The information about the type and amount of flow was not provided to the CNNs and the task was still to predict diffusion coefficient purely from the four-frame averaged image. Figure 3 shows the performance of CNNs when trained and tested on these datasets.

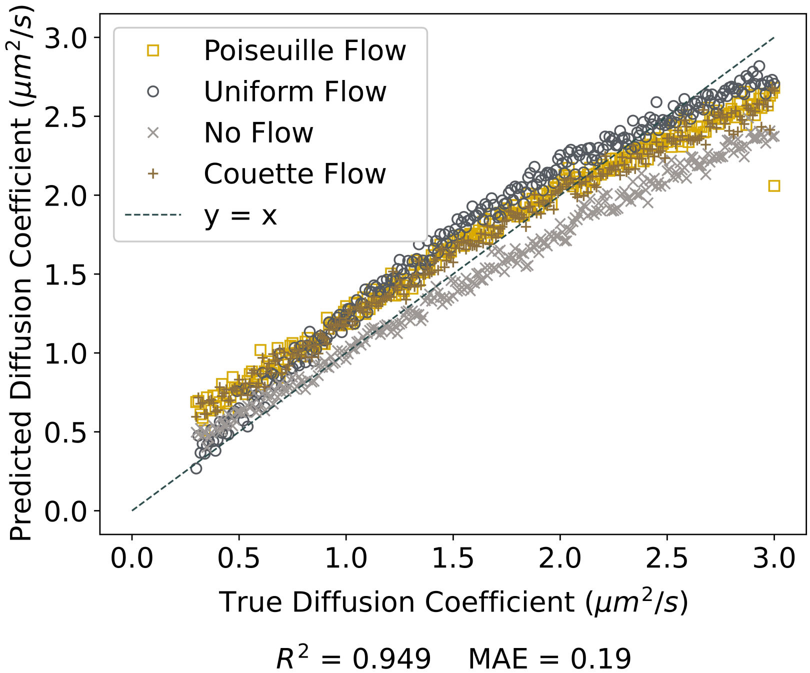

Figure 3. Performance of CNNs when trained and tested with Gaussian particles under the arbitrary flow condition. The arbitrary flow can either be uniform flow, Couette flow, Poiseuille, or no flow.

It can be seen that CNNs can learn to predict diffusion coefficients even in the presence of flow. Even though the type of flow was not explicitly mentioned while training the CNNs, their predictions from the same flow type were stacked together showing that the type of flow influences the CNN prediction. Specifically, this particular model under predicted diffusion coefficient values for the no flow case compared to any of the cases with a flow. Also, the CNN over predicted the diffusion coefficient values for uniform flow especially at larger diffusion coefficients.

4.3. Performance on Defocused Particles under no Flow Condition

Next, the CNNs were trained and tested on defocused particles under the no-flow conditions. Even though the particle motion was simple in this case, the particles had a complex shape that depended on their distance from the focal plane. As the particles undergo 3D diffusion, their distance from the focal plane changes, hence changing the imaged particle shape. Figure 4 shows the performance of CNNs when trained and tested on these datasets.

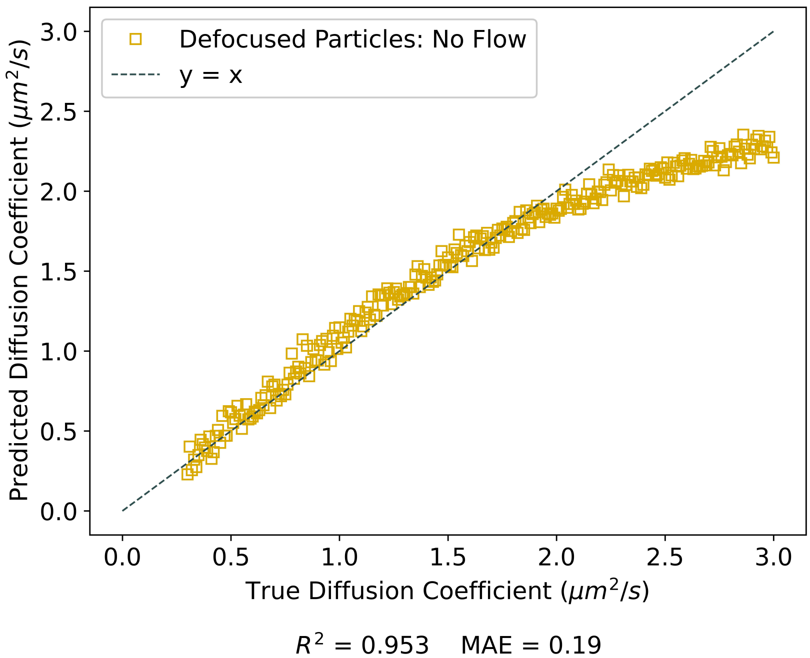

Figure 4. Performance of CNNs when trained and tested with defocused particles under a no flow condition. The shape of defocused particles changes in each frame with their distance from the focal plane, making it more difficult for CNNs to learn these distributions.

It can be seen that even though the quantitative performance in this case was similar to the performance of CNNs trained on Gaussian particles under an arbitrary flow, the performance degrades at higher diffusion coefficients. This degradation in performance comes at a sooner diffusion coefficient when compared to CNNs trained on Gaussian data, highlighting the limitation of the current workflow when the particle shapes change in each frame.

4.4. Benchmarking against conventional algorithms

To evaluate the performance of the current CNNs, these were compared against four conventional algorithms: TrackPy (Allan et al., 2023), correlation-EMSD(Sardana & Wereley, 2023), Olsen-Adrian(Olsen & Adrian, 2000), and iPED(Ahmadzadegan et al., 2020). The comparison was done based on the average Mean Average Error (MAE) between true and predicted diffusion coefficient values for various simulated datasets. Figure 5 compares the performance of current CNNs against the four conventional algorithms on the three simulated datasets using Mean Absolute Error (MAE) as the performance metric.

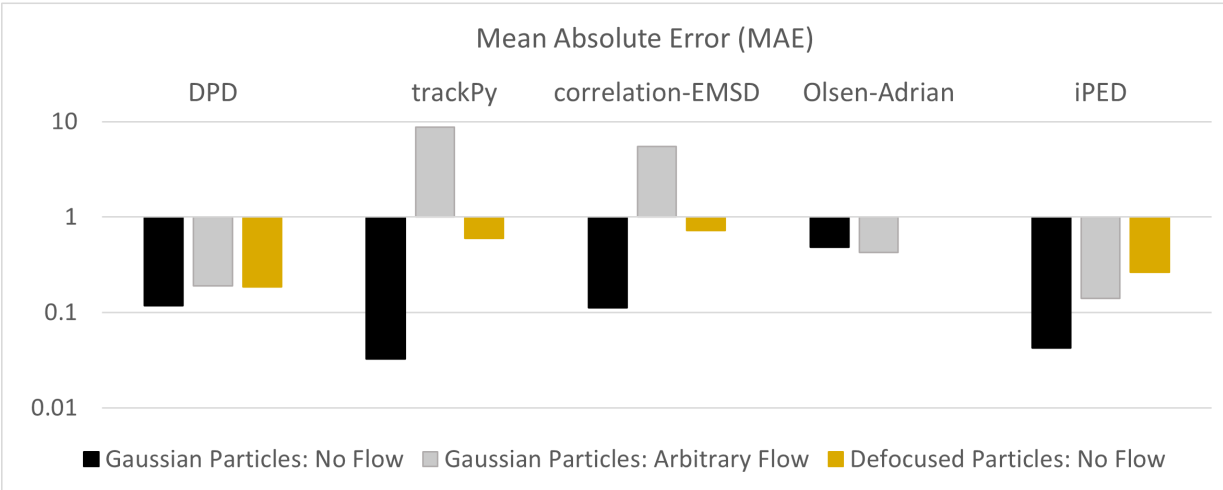

Figure 5. Comparison of the trained CNNs against the four conventional algorithms on the simulated datasets. The lower the Mean Absolute Error (MAE), the better the algorithm performance. The conventional algorithms used here were TrackPy (Allan et al., 2023), correlation-EMSD (Sardana & Wereley, 2023), Olsen-Adrian (Olsen & Adrian, 2000), and iPED (Ahmadzadegan et al., 2020)

As a lower MAE means better performance, it can be seen that many conventional algorithms perform better than the CNN models if the particles were assumed to be Gaussian-shaped and the system was under a no flow condition. However, most of the conventional algorithms performed worse than the CNNs if there was a flow or the particles were defocused. The performance of Olsen-Adrian was so bad that it was removed from the graph to improve visualization for the rest of the results. iPED was the only non-CNN algorithm with reasonable performance for all three cases. But, its performance was still worse than the CNNs when the particles were defocused.

4.5. Evaluation on Experimental Dataset

The CNN models were also used for predicting diffusion coefficients on experimental data collected under the no flow case. As the true values of experimental datasets were unknown, the CNNs predictions were compared against the outputs from trackPy. The particle locations from the experimental datasets were Gaussianized before feeding to the CNNs.

Table 2. Mean Absolute Uncertainty in $\mu m^2/s$ between the CNN prediction when compared with trackPy outputs and iPED outputs.

Concentration (mg/ml)	trackPy	iPED
20	0.229	0.461
15	0.216	0.476
10	0.194	0.371

It can be seen that the experimental uncertainties between the outputs from CNN models and trackPy were quite similar to errors in the simulated case. These uncertainties become a bit higher when comparing the outputs of CNN models and iPED. Still, these errors are quite low and the CNN models can be used for interference on experimental datasets.

5. Conclusion and Future Work

This paper provides a CNN-based method for predicting diffusion coefficients from particle images. The proposed method has a high performance even when the particles have complex defocused shapes or the underlying flow has a flow. These are the cases where the performance of existing algorithms degrades significantly. This benchmarking was done on various simulated datasets. The proposed method also provides low uncertainties in experimental data.

Even though the proposed methods have low prediction errors, there is still a large scope for improvement. The performance of the current CNNs degrades for high diffusion coefficients when the particles are defocused. In these high diffusion coefficients, the particle shapes change significantly between each frame making it harder for the current CNNs to extract the diffusion coefficients. Additionally, the CNNs are not trained and tested for the case when there is a flow along with defocused particles. It is the most complicated case and is closest to the real world. Finally, the performance of the CNNs can be improved by changing how data is fed and exploring better search spaces for hyperparameters and architectural parameters. More work is needed in these directions.

References

Ahmadzadegan, A., Ardekani, A. M., & Vlachos, P. P. (2020, 9). Estimation of the probability density function of random displacements from images. *Physical Review E*, *102*, 033305. Retrieved from `https://journals.aps.org/pre/abstract/10.1103/PhysRevE.102.033305` doi: 10.1103/PHYSREVE.102.033305/FIGURES/10/MEDIUM

Allan, D. B., Caswell, T., Keim, N. C., van der Wel, C. M., & Verweij, R. W. (2023, 2). *soft-matter/trackpy: v0.6.1.* Retrieved from `https://zenodo.org/record/7670439` doi: 10.5281/ZENODO.7670439

Barnkob, R., Cierpka, C., Chen, M., Sachs, S., Mader, P., & Rossi, M. (2021, 6). Defocus particle tracking: a comparison of methods based on model functions, cross-correlation, and neural networks. *Measurement Science and Technology*, *32*, 094011. Retrieved from `https://iopscience.iop.org/article/10.1088/1361-6501/`

abfef6https://iopscience.iop.org/article/10.1088/1361-6501/abfef6/meta doi: 10.1088/1361-6501/ABFEF6

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 24). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf

Cagno, M. P. D., Clarelli, F., Vabenø, J., Lesley, C., Rahman, S. D., Cauzzo, J., ... Stein, P. C. (2018, 4). Experimental determination of drug diffusion coefficients in unstirred aqueous environments by temporally resolved concentration measurements. *Molecular Pharmaceutics*, *15*, 1488-1494. Retrieved from https://pubs.acs.org/doi/abs/10.1021/acs.molpharmaceut.7b01053 doi: 10.1021/ACS.MOLPHARMACEUT.7B01053/SUPPL_FILE/MP7B01053_SI_001.PDF

Cai, S., Liang, J., Gao, Q., Xu, C., & Wei, R. (2020, 6). Particle image velocimetry based on a deep learning motion estimator. *IEEE Transactions on Instrumentation and Measurement*, *69*, 3538-3554. doi: 10.1109/TIM.2019.2932649

Clayton, K. N., Moehling, T. J., Lee, D. H., Wereley, S. T., Linnes, J. C., & Kinzer-Ursem, T. L. (2019, 2). Particle diffusometry: An optical detection method for vibrio cholerae presence in environmental water samples. *Scientific Reports 2019 9:1*, *9*, 1-12. Retrieved from https://www.nature.com/articles/s41598-018-38056-7 doi: 10.1038/s41598-018-38056-7

Clayton, K. N., Salameh, J. W., Wereley, S. T., & Kinzer-Ursem, T. L. (2016, 9). Physical characterization of nanoparticle size and surface modification using particle scattering diffusometry. *Biomicrofluidics*, *10*. Retrieved from https://pubmed.ncbi.nlm.nih.gov/27703593/ doi: 10.1063/1.4962992

Ernst, D., & Köhler, J. (2012, 12). Measuring a diffusion coefficient by single-particle tracking: statistical analysis of experimental mean squared displacement curves. *Physical Chemistry Chemical Physics*, *15*, 845-849. Retrieved from https://pubs.rsc.org/en/content/articlehtml/2013/cp/c2cp43433dhttps://pubs.rsc.org/en/content/articlelanding/2013/cp/c2cp43433d doi: 10.1039/C2CP43433D

Lee, D. H., Madsen, E. A., Linnes, J. C., & Wereley, S. T. (2022, 12). Temporally and spatially resolved micro-rheometry of a transient viscous polymer formation. *Measurement Science and Technology*, *34*, 035301. Retrieved from https://iopscience-iop-org.ezproxy.lib.purdue.edu/article/10.1088/1361-6501/aca993https://iopscience-iop

-org.ezproxy.lib.purdue.edu/article/10.1088/1361-6501/aca993/meta doi: 10.1088/1361-6501/ACA993

Moehling, T. J., Lee, D. H., Henderson, M. E., McDonald, M. K., Tsang, P. H., Kaakeh, S., ... Linnes, J. C. (2020, 11). A smartphone-based particle diffusometry platform for sub-attomolar detection of vibrio cholerae in environmental water. *Biosensors and Bioelectronics*, *167*, 112497. doi: 10.1016/J.BIOS.2020.112497

Olsen, M. G., & Adrian, R. J. (2000, 10). Brownian motion and correlation in particle image velocimetry. *Optics Laser Technology*, *32*, 621-627. doi: 10.1016/S0030-3992(00)00119-5

Rossi, M. (2019, 10). Synthetic image generator for defocusing and astigmatic piv/ptv. *Measurement Science and Technology*, *31*. Retrieved from https://iopscience-iop-org.ezproxy.lib.purdue.edu/article/10.1088/1361-6501/ab42bbhttps://iopscience-iop-org.ezproxy.lib.purdue.edu/article/10.1088/1361-6501/ab42bb/meta doi: 10.1088/1361-6501/AB42BB

Sardana, P., & Wereley, S. T. (2023, 11). Deep particle diffusometry: Using deep learning to measure diffusion coefficient from particle images. *Measurement Science and Technology*. Retrieved from https://iopscience.iop.org/article/10.1088/1361-6501/ad108bhttps://iopscience.iop.org/article/10.1088/1361-6501/ad108b/meta doi: 10.1088/1361-6501/AD108B