This article is part of the

Proceedings of the 16th Minisymposium Verfahrenstechnik and 7th Partikelforum (TU Wien, Sept. 21/22, 2020)

Title:
Deep Neural Network-based View factor Modelling of Radiative Heat Transfer between Particles

Corresponding author:
Josef Tausendschön (TU Graz), josef.tausendschoen@tugraz.at

Date of submission:
28.02.20

Date of revision:
10.09.20

Date of acceptance:
10.09.20

Chapter ID:
MoV5-(03)

Length:
3 pages

ISBN (full book):
978-3-903337-01-5

Deep Neural Network-based View factor Modelling of Radiative Heat Transfer between Particles

Josef Tausendschön[1*], Stefan Radl[1]

1: Dept. of Process and Particle Technology, Graz University of Technology, Austria
* Correspondent author: josef.tausendschoen@tugraz.at

Keywords: Radiative Heat Transfer, Data-driven Modeling, Machine Learning, Deep Neural Networks

1. Introduction

Above 700 °C radiative heat transfer is the most dominating heat transfer mechanism. Such high temperature processes occur in a wide range of industrial applications. Examples are: pebble bed reactors, laser sintering and high-temperature particle oxidation or reduction processes [1].

The modelling of heat radiation is an ongoing challenge in many simulation fields, e.g. Computational Fluid Dynamics (CFD) and the Discrete Element Method (DEM). In this study we investigate radiative heat modelling from a particle perspective. In principle the emitted heat flux $\dot{Q}_{i,j}$ between two particles depends on the following factors: particles temperatures T, surfaces A, emissivity ϵ and the view factor ε_{i-j} between these particles, as well as the Boltzmann constant σ_s (see Eq.1). The view factor is the ratio of the radiation leaving a surface i and the radiation that is striking surface j.

$$\dot{Q}_{i,j} = \frac{\sigma_s \left(T_i^4 - T_j^4\right)}{\frac{1-\epsilon_i}{A_i\,\epsilon_i} + \frac{1-\epsilon_j}{A_j\,\epsilon_j} + \frac{1}{\varepsilon_{i-j}\,A_i}} \qquad (1)$$

View factors can be found by analytical or numerical integration of the solid angles under which the surfaces can see each other. Typical industrial processes feature billions of particles and even in small research devices millions of particles are interacting. Therefore, the view factor calculation based on integration cannot be performed for real-sized systems due to the astronomical computational cost. Walker et al. [2] presented the so-called “Monte Carlo Raytracing” that is derived from the scientifically widely adopted Monte Carlo algorithm to determine view factors. Another (computationally much more efficient) option is the “Projection Method” from Forgber & Radl [3], where a certain number of test points is distributed on the particle surfaces and then by using the solid angle between the particles, the area fraction which the absorbing particle actually sees is determined. The area fraction is composed by an assemble of the distributed test points. The view factor is then the number of test points on surface j divided by the total number of test points distributed on the emitting surface i.

The Monte Carlo Raytracing and the Projection Method reflect the in simulation science widespread tradeoff between accuracy and speed. The Monte Carlo raytracing achieves very accurate view factors, but takes up high computational cost. The projection method provides precise view factors in specified systems at significantly less computational effort, although the demand is still too big to adopt the method in general for heat radiation modelling in DEM simulations.

To overcome this tradeoff in the presented approach we use Machine Learning (ML) techniques to create a pre-trained heat radiation model based on a deep neural net (DNN), that predicts accurate view factors at high speed. The training data is generated with the Monte Carlo Raytracing method from a small particle bed (see Figure 1).

The outcome of the presented approach will be compared to the projection method and a simple regression analysis in terms of accuracy and computational cost.

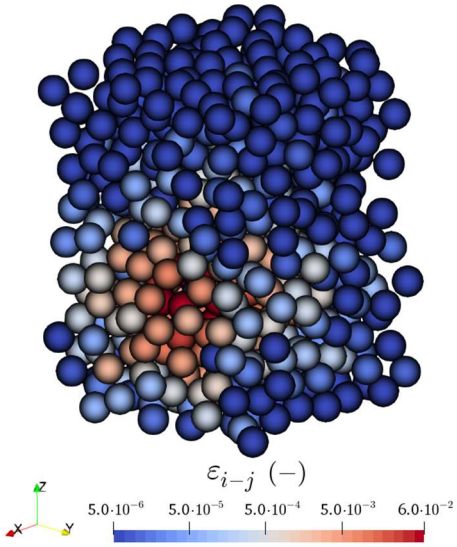

Figure 1: View factors in a typical random particle packing

2. Neural Net Model Parameters

DNNs consist of an input layer, hidden layer and an output layer. Each layer consists of an arbitrary number of nodes and a DNN contains an arbitrary number of hidden layers. Therefore, two important decisions are made when implementing a DNN: the number of hidden layers and the number of nodes for each of these layers.

Basically, a neural net with zero hidden layers can represent linear separable functions or decisions. However, without a hidden layer the neural net is not considered to be “deep” and cannot model nonlinear behavior. With one hidden layer a neural net can approximate any function that contains a continuous mapping from one finite space to another. Using two hidden layers enables the DNN to model an arbitrary decision boundary to an arbitrary accuracy [4]. Therefore, two versions of each DNN with different input layer will be investigated: a one hidden layer version and a two hidden layer version.

The input layer is defined by the features of the training data, that are often called markers. Consequently, the number of nodes for the input layer is the number of markers of your training data. The output layer is then defined by the desired output and the number of nodes for that reason equals one, namely the view factor. The choice of the activation layers and the used optimizer are also considered as neural network model design. The Stochastic Gradient Descent (SGD) optimizer is used as standard optimizer and the Adam optimizer as advanced option [5]. In regression tasks no activation function is applied to the output layer. The input layer and the hidden layer are typically followed by the Rectifying Linear Unit (ReLU) activation function in regression tasks [6]. To avoid overfitting of the neural net two options are tested: using so-called “Dropout Layers” or applying the early stopping method [7]. The learning rate and if used the dropout rate are called the hyperparameters of the neural network. Together the model design and the hyperparameters form the neural net model parameters. To achieve the best possible

generalization with the DNN-based model it is essential to optimize these model parameters. The optimization is performed by the randomized search approach [8].

3. Training of the deep neural net

As previously mentioned, the view factor data is generated with the Rayfactor tool from a randomly-positioned monodisperse particle bed. The particle bed contains of 587 particles. The particle volume fraction of the system is 0.40. Therefore, a dataset with 586·586 can be generated since every particle is emitting and absorbing. To ease up the training a dataset with 33 emitting particles is considered. The dataset is split into a training set, validation set and test set, where 80 % is used for training and respectively 10 % for testing and validation.

Figure 2 shows exemplarily the view factors that are calculated by the Rayfactor tool for one emitting particle and a simple regression over the dimensionless surface to surface distance S_{i-j}.

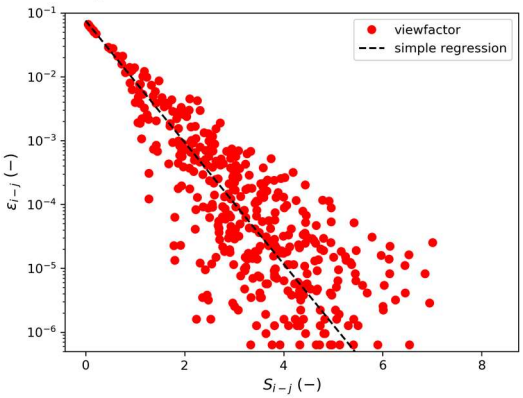

Figure 2: Exemplary view factor data for one emitting particle in the particle bed shown in Figure 1.

The creation and the training of the DNN is performed within the Keras® framework in Python®. The preparation of the raw data and the randomized search optimization is made by using the Scikit-learn® environment. One has to note that very small view factors lead to negligibly small heat fluxes and are set to zero if they are below a threshold value of $5 \cdot 10^{-6}$. The thresholding is applied to the training data and also to the prediction of the DNN. The mean squared error (MSE) between the target and the prediction is one metric to describe the performance of the DNN. Because the MSE is significantly more influenced by big view factors and does not reflect the overall quality of the prediction, the calculated view factors are correlated to the view factors determined via Monte Carlo Raytracing. The coefficient of determination (R^2) is then used to describe this correlation. For the use in dense settled particle beds with very high particle volume fractions, separate datasets need to be created. The derived DNN-based model can be used for non-settled particle beds without limitation. Particle distances or system sizes are also not restrictive, since input markers are typically normalized before being fed to the neural net.

4. Marker Selection

As can be seen in Figure 2 a certain distance S_{i-j} between the emitting and absorbing particle can lead to varying view factors. A neural net with a single input node always represents the same output for the same input. Therefore, additional markers that can be fed into the neural net need to be found within the particle data.

The solid angle between the interacting particles is an important quantity in the analytical integration of view factors. For that reason, the solid angle will be investigated as marker. Representative for a local particle volume fraction the volume of Voronoi cells achieved from Voronoi tessellation is also considered as marker. Shadows between particles significantly influence the radiative heat transfer, therefore an algorithm that detects particles that are between the emitting and the absorbing particle in a certain volume was implemented. The number of particles in that volumetric area is counted and the significance of that marker for the prediction is also investigated. To adjust the results to polydisperse systems additional markers, e.g., the ratio of the radii could be used.

5. Results

The Projection Method (number of test points = 400) achieves an MSE of $7.8 \cdot 10^{-7}$ on the test dataset. It took 16.24 s to distribute the test points and 24.55 s to calculate the view factors on a single CPU core. The coefficient of determination is 0.986.

A simple linear regressor derived from the total dataset (for an example regression see Figure 2) achieves an MSE of $7.6 \cdot 10^{-7}$ on the test dataset at a basically instantaneous speed. However, as can be seen quantitively in Figure 2, the simple regression cannot accurately predict the spread of the view factors for large distances between the particles.

Figure 3 shows the correlation of the DNN view factors with that of the Monte Carlo Raytracing method. The DNN-based Model achieves an MSE of around $5 \cdot 10^{-7}$ while taking around 0.30 s to calculate the view factors and 0.55 s to load the pretrained model. The R^2 value is 0.9906.

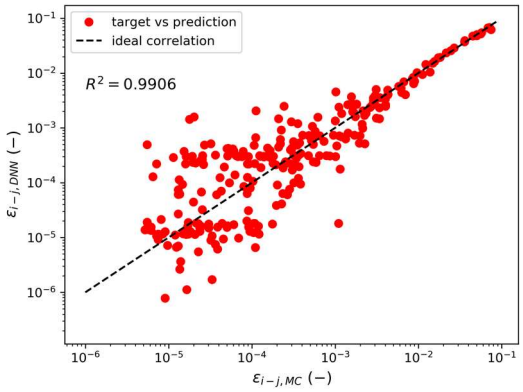

Figure 3: Prediction of the DNN-based model vs. target from Monte Carlo Raytracing of a test dataset

It is demonstrated that a pretrained DNN can model view factors at higher accuracy and with significantly less computational effort than other models in literature.

[1] M.F. Modest, Radiative Heat Transfer - Second Edition, Academic Press, 2003. https://doi.org/10.1017/CBO9781107415324.004.

[2] T. Walker, S.C. Xue, G.W. Barton, Numerical determination of radiative view factors using ray tracing, J. Heat Transfer. 132 (2010) 1–6. https://doi.org/10.1115/1.4000974.

[3] T. Forgber, S. Radl, A novel approach to calculate radiative thermal exchange in coupled particle simulations, Powder Technol. 323 (2018) 24–44. https://doi.org/10.1016/j.powtec.2017.09.014.

[4] J. Heaton, Artficial Intelligence For Humans, Volume 3: Deep Learning and Neural Networks, Heaton Research, Inc., 2015.

[5] D.P. Kingma, J.L. Ba, Adam: A method for stochastic optimization, 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc. (2015) 1–15.

[6] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT Press, 2016.

http://www.deeplearningbook.org/.
[7] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958. https://doi.org/10.1016/0370-2693(93)90272-J.
[8] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2012) 281–305.