Development of Uncertainty-Guided Deep Learning with Application to Thermal Fluid Closures

Chih-Wei Chang, Nam Dinh

NEKVA/NUC Workshop
“Multiphysics Model Validation”
June 27-29, 2017
Outline

• Introduction
• Technical background overview
• Formulation of the framework
• Case study
• Type-II machine learning (work in progress)
Motivation

• Sub-grid-scale (SGS) physics models (or so-called closure relations) determines the accuracy of thermal-fluid modeling, and it is essential for simulation codes.

• "Big" data in thermal-hydraulics are available with the invention of measurement equipment such as inferred and PIV (Particle image velocimetry) camera.

• Deep learning (DL) is a universal approximator [Hinton, 1989], and can discover the underlying correlations behind the data to achieve the cost-effective closure development for
  – new geometries, new coolants or system conditions, particularly in newly designed systems.
Physics-Constrained ML for Thermal Fluids

aaS: as a Service
(D, M): (Data, Method)
SGS: Sub-Grid-Scale
Objectives

- We will develop the uncertainty guided deep learning framework for developing fluid dynamics closures.
  - Allow thermo-fluid codes to have the features of
    - Robustness
    - Reliability
    - Adaptability
    - Extensibility
  - Benefit not only system-level codes but also computational multi-fluid dynamics simulations (not DNS).
Outline

• Introduction
• Technical background overview
• Formulation of the framework
• Case study
• Type-II machine learning (work in progress)
Introduction

Closure Relations

Field Equations

Machine Learning
Conservation Equations

- Two-phase mixture model (TMM) [Doster, 2015]
  
  - Mass
    \[ A_x \frac{\partial \rho}{\partial t} + \frac{\partial \rho v A_x}{\partial z} = 0 \]
  
  - Momentum
    \[ A_x \frac{\partial \rho v}{\partial t} + \frac{\partial \rho v v A_x}{\partial z} = -A_x \frac{\partial P}{\partial z} - \tau_w P_w + \mu g_z A_x - \frac{\partial}{\partial z} \left[ \frac{\alpha_g \rho_g \alpha_i \rho_l}{\rho} (v_g - v_l)^2 \right] A_x \]
  
  - Energy
    \[ A_x \frac{\partial \rho u}{\partial t} + \frac{\partial \rho u v A_x}{\partial z} = -P \frac{\partial A_x}{\partial z} + q_w P_w - \frac{\partial}{\partial z} \left[ \frac{\alpha_g \rho_g \alpha_i \rho_l}{\rho} (v_g - u_t)(v_g - v_l) \right] A_x + P \frac{\partial}{\partial z} \left[ \frac{\alpha_g \rho_g \alpha_i \rho_l}{\rho} (v_g - v_l)(v_g - v_l) \right] A_x \]
Data assimilation by Machine Learning

- The data assimilation in this work is different from the definition by meteorologists.
- Our goal is to have a relax function form that can assimilate the data to enable cost-effective closure model development.

<table>
<thead>
<tr>
<th>Tribe</th>
<th>Problem</th>
<th>Master Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbolists</td>
<td>Knowledge Composition</td>
<td>Inverse Deduction</td>
</tr>
<tr>
<td>Evolutionaries</td>
<td>Structure Discovery</td>
<td>Genetic Programming</td>
</tr>
<tr>
<td>Analogizers</td>
<td>Similarity</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>Connectionists</td>
<td>Credit Assignment</td>
<td>Backpropagation</td>
</tr>
<tr>
<td>Bayesians</td>
<td>Uncertainty</td>
<td>Probabilistic Inference</td>
</tr>
</tbody>
</table>
Deep learning - model structure

- Neural networks (NNs) with more than two layers are deep learning (DL) [Heaton, 2015].
- More layers are expected to capture data behaviors well, but it may suffer from overfitting issues.
  - No theorem to guide the selection of NN hyperparameters
- Multilayer NNs are universal approximators [Hornik, 1989].

Deep means multilayer

**Activation Functions**

- **Sigmoid**
  \[ f(x) = \frac{1}{1 + e^{-x}} \]
- **ReLU (Rectified Linear Unit)**
  \[ f(x) = \max(0, x) \]

**Hidden Units (Neurons)**

\[ HU_{11}(x) = \sigma \left( \sum_{i=1}^{2} w_{1i}x_i + b_{11} \right) \]
Deep learning - model structure

- Neural networks (NNs) with more than two layers are deep learning (DL) [Heaton, 2015].
- More layers are expected to capture data behaviors well, but it may suffer from overfitting issues.
  - No theorem to guide the selection of NN hyperparameters
- Multilayer NNs are universal approximators [Hornik, 1989].

\[ f(x) = \frac{1}{1 + e^{-x}} \]

**Activation Functions**

- **Sigmoid**
  \[ f(x) = \frac{1}{1 + e^{-x}} \]

- **ReLU (Rectified Linear Unit)**
  \[ f(x) = \max(0, x) \]

\[ HU_{11}(x) = \sigma(\sum_{i=1}^{2} w_{1i}x_i + b_{11}) \]

\[ HU_{21}(x) = \sigma(\sum_{i=1}^{3} w_{2i}HU_{1i} + b_{21}) \]
Deep learning - model structure

- Neural networks (NNs) with more than two layers are deep learning (DL) [Heaton, 2015].
- More layers are expected to capture data behaviors well, but it may suffer from overfitting issues.
  - No theorem to guide the selection of NN hyperparameters
- Multilayer NNs are universal approximators [Hornik, 1989].
Deep learning - model structure

\[ HU_{11}(x) = \sigma\left( \sum_{i=1}^{2} w_{i1}x_i + b_{11} \right) \]

\[ HU_{21}(x) = \sigma\left( \sum_{i=1}^{3} w_{2i}HU_{1i} + b_{21} \right) \]

\[ o_1(x) = \sigma\left( \sum_{i=1}^{3} w_{oi}HU_{2i} + b_{o1} \right) \]

\[ \hat{y}(x) = o_1(x) \]

L2 norm (loss function)

\[ L = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \]

Stochastic Gradient Descent

\[ w^{n+1} = w^n - \eta \frac{\partial L}{\partial w^n} \]
Deep learning - workflow

**Recipe of Deep Learning**

1. **Step 1: Network Structure**
2. **Step 2: Learning Target**
3. **Step 3: Learn!**

- **Good Results on Training Data?**
  - **NO**
  - **Overfitting!**
  - **NO**
  - **Good Results on Testing Data?**
    - **YES**
    - **Other methods do not emphasize this.**

Lee, 2016
# Deep learning – frameworks

<table>
<thead>
<tr>
<th>Configuration File</th>
<th>Programmatic Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Caffe</strong></td>
<td>Theano</td>
</tr>
<tr>
<td>Microsoft Cognitive Toolkit</td>
<td>Tensorflow</td>
</tr>
<tr>
<td></td>
<td>Torch</td>
</tr>
</tbody>
</table>

## Table of Comparisons

<table>
<thead>
<tr>
<th>Framework</th>
<th>Languages</th>
<th>Tutorials and training materials</th>
<th>CNN modeling capability</th>
<th>RNN modeling capability</th>
<th>Architecture: easy-to-use and modular front end</th>
<th>Speed</th>
<th>Multiple GPU support</th>
<th>Keras compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theano</td>
<td>Python, C++</td>
<td>++</td>
<td>+++</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Tensor-Flow</td>
<td>Python</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Torch</td>
<td>Lua, Python (new)</td>
<td>+</td>
<td>+++</td>
<td>++</td>
<td>++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Caffe</td>
<td>C++</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>MXNet</td>
<td>R, Python, Julia, Scala</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>Neon</td>
<td>Python</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>CNTK</td>
<td>C++</td>
<td>+</td>
<td>+</td>
<td>+++</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

[http://www.svds.com/getting-started-deep-learning/]
There is no ML application for thermal-hydraulics yet.

There is no PDE-constrained DL application yet.

Although 2-layer NN has been applied to the isothermal bubbly flow case, the previous work does not indicate the method to select
– NN hyperparameters
– and the requirements of datasets.
Outline

- Introduction
- Technical background overview
- Formulation of the framework
- Case study
- Type-II machine learning (work in progress)
Introduction (1/2)

Traditional Framework

Taking years to decades

Experiments

Model Development (Semi-empirical)

Mechanistic Description

Related Models and Insights

Simulation

Assessment (Validation, UQ)

Application (Lessons Learned)

Knowledge Base
Introduction (2/2)

Data-Driven Modeling Framework

- Experiments/Simulations
- Data Analysis
- Physics-Constrained Machine Learning
- Related Models and Insights
- Simulations
- Assessment (Validation, UQ)
- Application (Lessons Learned)
- Knowledge Base

Traditional Framework

- Taking years to decades
- Mechanistic Description
- Experiments
- Model Development (Semi-empirical)
- Simulation
- Assessment (Validation, UQ)
- Application (Lessons Learned)
- Knowledge Base
Method of manufactured data (MMD)

• To evaluate the framework, the present work proposes a method of manufactured data (MMD) as surrogates for actual datasets in real-life applications.

• The MMD applies a computer code with high-fidelity models to generate numerical solutions
  – for training and benchmark purposes.

• The “high fidelity” here refers to models which have been more extensively adjusted and assessed to be trustworthy for conditions under consideration.

• Both training and benchmark datasets can be generated with
  – different degree of detail (homogenization, amount).
  – controlled uncertainty (“manufactured errors and biases”).
Physics-constrained machine learning

- PCML problems involve PDE models with sub-grid-scale (SGS) physics models that are scale separable, for which closure relations are local.
Issues of using DL in thermal fluid (1/6)

• Questions
  – What is the uncertainty in DL due to different weight initialization?
  – How to select Tikhonov regularization parameters to increase the predictability?
  – What is the uncertainty due to different DNN structures?
  – How to increase the training accuracy for DNN while using multiple datasets?
  – What are the criteria to search the optimal DL-based fluid closures for field equations?
Issues of using DL in thermal fluid (2/6)

• Understand the uncertainties in DL by different weight initialization.
  – “Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful,” George E.P. Box.
  – The goal of uncertainty quantification is to evaluate and minimize uncertainties associated with experiments and models.
  – DL with the same structure can result in different predictabilities due to parameter uncertainties. The following figures identify this fact by using DL-based friction closures with different parameters.
Issues of using DL in thermal fluid (3/6)

• Understand how to use Tikhonov regularization to increase the predictability.
  – DL model tends to over fit the noisy dataset.
  – The following figures shows the regularization can increase the prediction for DL-based friction closures. The question is how to systematically determine the regularization parameters to obtain DL-based fluid closures with robustness.

Application, 4-Layer DL, Regularization Strength = 0.0e+00

Application, 4-Layer DL, Regularization Strength = 1.0e-06
Issues of using DL in thermal fluid (4/6)

- Understand the uncertainty due to different DNN structures and datasets.
  - The Euclidean loss of DNN with the layer number more than 3 will converge to a larger value than DNN with less layers. Fig. 1 depicts the results by 1-training dataset, and the wiggle in the figure is due to lack of training datasets.

Fig. 1 DNN with 1-training dataset

Fig. 2 DNN with 9-training dataset
Issues of using DL in thermal fluid (5/6)

- Understand the difficulty of training DNN while using multiple datasets.
  - The Euclidean loss of DNN with more layers and training datasets tend to saturate at a higher value than DNN with less layers and training data.
  - Fig. 1 depicts the comparison of Euclidean losses by using different datasets and different DNN structures. The black and red dash lines show the results by fully connected feedforward NN with 6 layers.
  - In order to obtain accurate results for DNN with training dataset more than 1, we implement batch normalization (BN) [Ioffe, 2015] into DNN model. The blue line shows that BN can help DNN to achieve lower Euclidean loss.
Issues of using DL in thermal fluid (6/6)

• Understand how to define the criteria to search the optimal DL-based fluid closures for field equations.
  – Inferring a model from data belongs to ill-posed problem that the solution may not exist.
  – Hadamard’s principles on well-posedness
    • A solution should exist.
    • The solution should be unique.
    • The solution should continuously depend on the data.
  – Requirements of well-posedness
  – Requirements of minimum training datasets

Structure?
Requirements of well-posedness

- The output behavior of DL-based model should change continuously with the insight for the given problems.
  - “Insight” is the current best knowledge toward the problems

- We define model complexity (MC) and model-insight consistence (MIC) factors as:

  Model Complexity (MC) = \left| \frac{\partial M}{\partial Par} \right| - \left| \frac{\partial I}{\partial Par} \right|

  Model Insight Consistence (MIC) = 1 - MC

- Model insight consistence factor dose not represent the model accuracy; instead, it is an indicator to check if the DL-based model achieves the given requirement.

Ex. Friction Factor

\[ I \propto \text{Re}^{-1} \]

\[ M = \text{sigmoid} \left( \sum_{i=1}^{5} HU_{k-1,i}w_{kji} + b_{kj} \right) \]
Outline

- Introduction
- Technical background overview
- Formulation of the framework
- Case study
  - Requirements of formulating well-posed DL-based closures for fluid simulations
- Type-II machine learning (work in progress)
The purpose of the testing task

• Questions
  – What is the uncertainty in DL due to different weight initialization?
  – How to select Tikhonov regularization parameters to increase the predictability?
  – What is the uncertainty due to different DNN structures?
    • What the DNN structure can produce optimal results?
  – How to increase the training accuracy for DNN while using multiple datasets?
  – What are the criteria to search the optimal DL-based fluid closures for field equations?
    • Can model-insight consistence (MIC) factor be the screening criterion to recognize whether DL-based closures are well-posed?
**Problem formulation**

- Applying manufactured data
  - Assuming the global model is valid for local application
- Developing a DL-based friction closure

\[
\frac{\partial (\rho A)}{\partial t} + \frac{\partial (\rho A v)}{\partial x} = 0
\]

\[
\frac{\partial (\rho v A)}{\partial t} + \frac{\partial (\rho v^2 A)}{\partial x} = -A \frac{\partial p}{\partial x} - F_F - A \rho g \frac{\partial z}{\partial x}
\]

\[
F_F = \frac{1}{2} \rho v |v| f S
\]

\[f = a Re^b\]

\[f = DL(Re)\]

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipe length (m)</td>
<td>2</td>
</tr>
<tr>
<td>Pipe diameter (m)</td>
<td>(6.25 \times 10^{-3})</td>
</tr>
<tr>
<td>Flow rate (kg/s)</td>
<td>(4.06 \times 10^{-4} - 8.6 \times 10^{-3})</td>
</tr>
<tr>
<td>Re (experiment)</td>
<td>(100 - 2000)</td>
</tr>
<tr>
<td>Re (training)</td>
<td>(300 - 1800)</td>
</tr>
</tbody>
</table>

S: circumference

f: Fanning friction factor

DL: deep learning
Data processing and results (1/2)

- Regularize DL-based closures by insights
- Prevent model outputs from a sudden gradient change

Model Complexity ($MC$) = \[ \left| \frac{\partial M}{\partial Par} \right| - \left| \frac{\partial I}{\partial Par} \right| \]

Model Insight Consistence ($MIC$) = $1 - MC$

\[ I \propto Re^{-1} \]

\[ M = \sigma \left( \sum_{i=1}^{5} HU_{k-1,i} w_{kj} + b_{kj} \right) \]
Data processing and results (2/2)

Full pipe pressure drop in both training and predicting domains by DL-based friction closures

Residual of full pipe pressure drop in both training and predicting domains by DL-based friction closures
Main findings

• Model-insight consistence (MIC) factor can be a screening criterion that guides the search for the optimal DL model.

• Optimal DL models are defined as:
  – They have the maximal predictive capability.
  – By Occam’s razor principle, they are deep neural networks with the simplest structure that captures the insights and the data within the uncertainty range.

• This case study is limited to model fully developed laminar flow using the simple closure model with a single scaling parameter, Re.
  – A broader case study is a must to characterize usefulness of the MIC factor in more complex processes
  – “Insight” is subject to multiple scaling parameters and various sources of uncertainty such as pressure loss due to spacer grids.
Outline

• Introduction
• Technical background overview
• Formulation of the framework
• Case study
• Type-II machine learning (work in progress)
Type-II machine learning (work in progress)

- Classification of machine learning (ML)

**Type-I ML, Physics-Constrained ML**

**Type-II ML, Physics-Informed ML**
Technical approach (1/2)

- 2D steady-state heat conduction equation with $k(T)$

$$\frac{\partial}{\partial x} \left( k \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left( k \frac{\partial T}{\partial y} \right) = 0$$

- Geometry
  - square geometry with length equal to 1 cm

- Model assumptions
  - Temperature dependent thermal conductivity [W/(cm-K)]
    $$k = \frac{1}{11.75 + 0.0235T}$$
    - Dirichlet boundary conditions

- Numerical discretization (Patankar, 1980)
  - 21 grids on each side ($\Delta x = \Delta y = \Delta h = 0.05 cm$)
  - piecewise-linear temperature and $k$ profile between the grids
  - centered formula of order $O(h^2)$

$$T_{i,j}^{n+1} = \left( T_{i+1,j}^n + T_{i-1,j}^n + T_{i,j+1}^n + T_{i,j-1}^n \right) \frac{k_{i,j}^n}{k_{i+1,j}^n + k_{i-1,j}^n + k_{i,j+1}^n + k_{i,j-1}^n + k_{i+1,j+1}^n + k_{i+1,j-1}^n + k_{i-1,j+1}^n + k_{i-1,j-1}^n + 4k_{i,j}^n}$$

Technical approach (2/2)

• Deep learning library - Tensorflow (v 1.0.0)
  – Train DNN to construct $k$ from data
  – Accelerate PDE solutions by GPU
• 2D steady-state heat diffusion

Heat Equation
$$T_{i,j}^{k+1} = \frac{1}{4} \left[ T_{i,j}^k + T_{i+1,j}^k + T_{i-1,j}^k + T_{i,j+1}^k + T_{i,j-1}^k + k^k_i T_{i,j}^k + 4k^k_j \right]$$
Method of manufactured data (MMD)

• Compare training and predicting datasets.
  – different temperature profiles
Results of type-I and type-II ML with $k(T)$

- Compare predicting results between type-I and type-II training

**Fig. 1** Predicting the temperature profiles by type-I training

**Fig. 2** Predicting the temperature profiles by type-II training
Main findings (work in progress)

- Tensorflow can accelerate the solution of PDE by GPU that make the type-II training becomes possible.

- Type-II training makes the temperature dependent conductivity model tightly couple with PDE. The results is more consistent with the data than type-I training.

- This nonlinear task confirms that the stochastic gradient descent (SGD) algorithm can search the optimal solution for type-II ML problems.
Thank you very much!

Chih-Wei Chang
cchang11@ncsu.edu

Nam Dinh
ntdinh@ncsu.edu