Deep Learning for Computer Graphics

Niloy Mitra  Iasonas Kokkinos  Paul Guerrero  Vladimir Kim  Nils Thuerey  Leonidas Guibas

UCL/Adobe  UCL/Ariel AI  UCL/Adobe  Adobe  TU Munich  Stanford University/FAIR
Course Organizers

Niloy Mitra
Course Organizers

Niloy Mitra  
Iasonas Kokkinos
Course Organizers

Niloy Mitra  Iasonas Kokkinos  Paul Guerrero
Course Organizers

Niloy Mitra
Iasonas Kokkinos
Paul Guerrero
Vladimir Kim
Course Organizers

Niloy Mitra  Iasonas Kokkinos  Paul Guerrero  Vladimir Kim  Nils Thuerey
Course Organizers

Niloy Mitra  
Iasonas Kokkinos  
Paul Guerrero  
Vladimir Kim  
Nils Thuerey  
Leonidas Guibas
Course Organizers

Niloy Mitra  Iasonas Kokkinos  Paul Guerrero  Vladimir Kim  Nils Thuerey  Leonidas Guibas
## Timetable

<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
<th>Niloy</th>
<th>Iasonas</th>
<th>Paul</th>
<th>Nils</th>
<th>Leonidas</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00</td>
<td>Introduction</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~9:15</td>
<td>Neural Network Basics</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~9:50</td>
<td>Supervised Learning in CG</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~10:20</td>
<td>Unsupervised Learning in CG</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>~10:55</td>
<td>Learning on Unstructured Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>~11:35</td>
<td>Learning for Simulation/Animation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>12:05</td>
<td>Discussion</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Course Objectives
Course Objectives

• Provide an overview of the popular **ML algorithms** used in CG

• Provide a quick overview of **theory** and **CG applications**
Course Objectives

• Provide an overview of the popular ML algorithms used in CG

• Provide a quick overview of theory and CG applications
  • Many extra slides in the course notes + example code

• Summarize progress in the last 3-5 years
Course Objectives

• Provide an overview of the popular **ML algorithms** used in CG

• Provide a quick overview of **theory** and **CG applications**
  • Many extra slides in the course notes + example code

• Summarize progress in the last 3-5 years
  • We have attempted to organize them
  • Discuss the main **challenges and opportunities** specific to CG
Help Us Improve
Help Us Improve

• Our aim is to convey what we found to be relevant so far

• You are invited/encouraged to give feedback
  • Speakup. Please send us your criticism/comments/suggestions
Help Us Improve

• Our aim is to convey what we found to be relevant so far

• You are invited/encouraged to give feedback
  • Speak up. Please send us your criticism/comments/suggestions
  • Ask questions, please!
Help Us Improve

- Our aim is to convey what we found to be relevant so far

- You are invited/encouraged to give feedback
  - Speakup. Please send us your criticism/comments/suggestions
  - Ask questions, please!

- Thanks to the many who helped so far with slides/comments
Representations in Computer Graphics
Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)
Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)
- Animation (e.g., skeletal positions over time; cloth dynamics over time)
Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)
- Animation (e.g., skeletal positions over time; cloth dynamics over time)
- Physics simulations (e.g., fluid flow over space-time, multi body interaction)
Problems in Computer Graphics

- Feature detection (image features, point features) \( \mathbb{R}^{m \times m} \rightarrow \mathbb{Z} \)
- Denoising, Smoothing, etc. \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- Embedding, Metric learning \( \mathbb{R}^{m \times m, m \times m} \rightarrow \mathbb{R}^{d} \)
- Rendering \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- Animation \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- Physical simulation \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- Generative models \( \mathbb{R}^{d} \rightarrow \mathbb{R}^{m \times m} \)
Problems in Computer Graphics

- Feature detection (image features, point features) \( \mathbb{R}^{m \times m} \rightarrow \mathbb{Z} \)
- Denoising, Smoothing, etc. \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- Embedding, Metric learning \( \mathbb{R}^{m \times m, m \times m} \rightarrow \mathbb{R}^{d} \)
- Rendering \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- Animation \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- Physical simulation \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- Generative models \( \mathbb{R}^{d} \rightarrow \mathbb{R}^{m \times m} \)
# Problems in Computer Graphics

- **Feature detection** (image features, point features) \( \mathbb{R}^{m \times m} \rightarrow \mathbb{Z} \)
- **Denoising**, **Smoothing**, etc. \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- **Embedding**, **Metric learning** \( \mathbb{R}^{m \times m, m \times m} \rightarrow \mathbb{R}^d \)
- **Rendering** \( \mathbb{R}^{m \times m} \rightarrow \mathbb{R}^{m \times m} \)
- **Animation** \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- **Physical simulation** \( \mathbb{R}^{3m \times t} \rightarrow \mathbb{R}^{3m} \)
- **Generative models** \( \mathbb{R}^d \rightarrow \mathbb{R}^{m \times m} \)
Goal: Learn a Parametric Function

\[ f_\theta : \mathbf{X} \rightarrow \mathbf{Y} \]

\( \theta \): function parameters, these are learned

\( \mathbf{X} \): source domain

\( \mathbf{Y} \): target domain
Goal: Learn a Parametric Function

\[ f_\theta : X \rightarrow Y \]

\( \theta \): function parameters, these are learned

\( X \): source domain

\( Y \): target domain

Examples:
Goal: Learn a Parametric Function

\[ f_\theta : X \rightarrow Y \]

\( \theta \): function parameters, these are learned

\( X \): source domain  \( Y \): target domain

Examples:

Image Classification:

\[ f_\theta : \mathbb{R}^{w \times h \times c} \rightarrow \{0, 1, \ldots, k - 1\} \]

\( w \times h \times c \): image dimensions  \( k \): class count
Goal: Learn a Parametric Function

\[ f_\theta : \mathbb{X} \rightarrow \mathbb{Y} \]

\( \theta \): function parameters, these are learned  \( \mathbb{X} \): source domain  \( \mathbb{Y} \): target domain

Examples:

Image Classification:

\[ f_\theta : \mathbb{R}^{w \times h \times c} \rightarrow \{0, 1, \ldots, k - 1\} \]

\( w \times h \times c \): image dimensions  \( k \): class count

Image Synthesis:

\[ f_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^{w \times h \times c} \]

\( n \): latent variable count  \( w \times h \times c \): image dimensions
Semantic Segmentation

Semantic Segmentation

Classification + Localization

Object Detection

Instance Segmentation

The Legend of Tarzan
Pose Detection using CNNs
Image Denoising

[Chaitanya et al. 2017, Siggraph]
Image Denoising

[Chaitanya et al. 2017, Siggraph]
Image Translation Problems

[Isola et al. 2017, CVPR]
DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling
Sketch to Face!

DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling

[Han et al. 2017, Siggraph]
Real Images
Real Images
Machine Learning 101: **Linear Classifier**

Each data point has a class label:

\[ y^i = \begin{cases} 1 & \text{(red)} \\ 0 & \text{(blue)} \end{cases} \]

\[ f_\theta : \mathbb{R}^n \rightarrow \{0, 1\} \]
Machine Learning 101: Linear Classifier

Each data point has a class label:

\[ y^i = \begin{cases} 
1 & (\text{red}) \\
0 & (\text{blue}) 
\end{cases} \]

\[ f_\theta : \mathbb{R}^n \rightarrow \{0, 1\} \]
Machine Learning 101: **Linear Classifier**

Each data point has a class label:

\[
y^i = \begin{cases} 
1 & \text{if } w x + b \geq 0 \\
0 & \text{if } w x + b < 0 
\end{cases}
\]

\[
f_\theta : \mathbb{R}^n \rightarrow \{0, 1\}
\]

\[
f_\theta (x) = \begin{cases} 
1 & \text{if } w x + b \geq 0 \\
0 & \text{if } w x + b < 0 
\end{cases}
\]
Machine Learning 101: **Linear Classifier**

Each data point has a class label:

\[
y^i = \begin{cases} 
1 & \text{if } w x + b \geq 0 \\
0 & \text{if } w x + b < 0 
\end{cases}
\]

\[
f_\theta : \mathbb{R}^n \longrightarrow \{0, 1\}
\]

\[
f_\theta(x) = \begin{cases} 
1 & \text{if } w x + b \geq 0 \\
0 & \text{if } w x + b < 0 
\end{cases}
\]

\[
\theta = \{w, b\}
\]

CreativeAI: Deep Learning for Computer Graphics
Data-driven Algorithms (**Supervised**)

Labelled data
(supervision data)
Data-driven Algorithms (Supervised)

Labelled data (supervision data) → ML algorithm → Trained model

CreativeAI: Deep Learning for Computer Graphics
Data-driven Algorithms (Supervised)

- Labelled data (supervision data) → ML algorithm → Trained model
- Test data (run-time data) → Trained model
Data-driven Algorithms (Supervised)

Labelled data (supervision data) → ML algorithm → Trained model → Prediction

Test data (run-time data) → Trained model
Data-driven Algorithms (Supervised)

- Labelled data (supervision data) → ML algorithm
- converged? → Trained model → Prediction

Validation data (supervision data)

Test data (run-time data)
Data-driven Algorithms (Supervised)

1. Labelled data (supervision data)
2. ML algorithm
3. converged?
4. Validation data (supervision data)
5. Test data (run-time data)
6. Trained model
7. Prediction
Data-driven Algorithms (Supervised)

Labelled data (supervision data) → ML algorithm → converged? → Validation data (supervision data)

Test data (run-time data) → Trained model → Prediction

Implementation Practice: Training: 70%; Validation: 15%; Test 15%
Training versus Validation Loss/Accuracy

- Underfitting
- Overfitting

Error vs. Model Parameter

Training error

Validation error
Training versus Validation Loss/Accuracy

underfitting

overfitting

validation error

training error

error

model parameter

CreativeAI: Deep Learning for Computer Graphics
Data-driven Algorithms (Unsupervised)

Training data → ML algorithm → converged? → Validation data → Test data (run-time data) → Trained model → Prediction

Implementation Practice: Training: 70%; Validation: 15%; Test 15%
Various ML Approaches (Supervised approaches)

Various ML Approaches (Supervised approaches)

Various ML Approaches (Supervised approaches)

Various ML Approaches (Supervised approaches)

Various ML Approaches (Supervised approaches)

Rise of Learning

• 1958:  Perceptron
• 1974:  Backpropagation
• 1981:  Hubel & Wiesel wins Nobel prize for ‘visual system’
• 1990s: SVM era
• 1998:  CNN used for handwriting analysis
• 2012:  AlexNet wins ImageNet
What is Special about CG?
What is Special about CG?

1. **Regular data structure** and easy to parallelize (e.g., image translation)

2. Many sources of input data — **model building** (e.g., images, scanners, motion capture)
What is Special about CG?

1. **Regular data structure** and easy to parallelize (e.g., image translation)

2. Many sources of input data — **model building** (e.g., images, scanners, motion capture)

3. Many sources of **synthetic data** — can serve as supervision data (e.g., rendering, animation)
What is Special about CG?

1. Regular data structure and easy to parallelize (e.g., image translation)

2. Many sources of input data — model building (e.g., images, scanners, motion capture)

3. Many sources of synthetic data — can serve as supervision data (e.g., rendering, animation)

4. Many problems in generative models and need for user-control
Main Challenges and Scope for Innovation
Main Challenges and Scope for Innovation

1. **Representation**: How is the data organised and structured?

2. **Training data**: Is it synthetic or real, or mixed?
Main Challenges and Scope for Innovation

1. **Representation:** How is the data organised and structured?

2. **Training data:** Is it synthetic or real, or mixed?

3. **User control:** End-to-end or in small steps?
Main Challenges and Scope for Innovation

1. **Representation:** How is the data organised and structured?

2. **Training data:** Is it synthetic or real, or mixed?

3. **User control:** End-to-end or in small steps?

4. **Loss functions:** Hand-crafted or learned from data?
Data is the New Currency

• **Synthetic** data
  • Generative model + photo-realistic **rendering**
  • Object geometry + physical **simulation**
  • Object geometry + synthetic materials + realistic simulations

• **Real** data
  • Collected from images, scans, mocap sessions
  • Collected using specialized equipments (e.g., light-field, pressure gloves)
End-to-end: **Learned Features**
End-to-end: **Learned Features**

- **Before**
  - Handcrafted feature extraction, e.g., edges or corners (hand-crafted)
  - Mostly with linear models (PCA)
- **Now**
End-to-end: **Learned Features**

- **Before**
  - Handcrafted feature extraction, e.g., edges or corners (hand-crafted)
  - Mostly with linear models (PCA)

- **Now**
  - End-to-end
  - Move away from hand-crafted representations
End-to-end: Learned Loss
End-to-end: Learned Loss

• Before
  • Evaluation came after
  • It was a bit optional
    • You might still have a good algorithm without a good way of quantifying it
    • Evaluation helped publishing

• Now
End-to-end: Learned Loss

• **Before**
  • Evaluation came after
  • It was a bit optional
    • You might still have a good algorithm without a good way of quantifying it
    • Evaluation helped publishing

• **Now**
  • It is essential and build-in
  • If the loss is not good, the result is not good
  • (Extensive) Evaluation happens automatically

• While still much is left to do, this makes graphics much more reproducible
End-to-end Training: Real/Generated Data
End-to-end Training: Real/Generated Data

• *Before*
  • Test with some toy examples
  • Deploy on real stuff
  • Maybe collect some performance data later

• *Now*
End-to-end Training: Real/Generated **Data**

- **Before**
  - Test with some toy examples
  - Deploy on real stuff
  - Maybe collect some performance data later

- **Now**
  - Test and deploy need to be as identical *(in distribution)*
  - Need to collect data first
  - No two steps
Course Plan

• Understand **common ML methods** (supervised and unsupervised) used in CG

• Understand the **building blocks**
  • Commonly used architectures, loss function, training advice

• Opportunities to develop **new methods**
  • ML methods for CG-specific domains (e.g., points, meshes, graphs)
  • How to mix synthetic/real data (and distributions)
Code Examples

PCA/SVD basis
Linear Regression
Polynomial Regression
Stochastic Gradient Descent vs. Gradient Descent
Multi-layer Perceptron
Edge Filter ‘Network’
Convolutional Network
Filter Visualization
Weight Initialization Strategies
Colorization Network
Autoencoder
Variational Autoencoder
Generative Adversarial Network

http://geometry.cs.ucl.ac.uk/creativeai/
Other Courses at Siggraph 2019

- **Deep Learning: A Crash Course**
  Andrew Glassner
  *Sunday 9:00-12:15*

- **Geometric Computing with Python**
  Sebastian Koch, Teseo Schneider, Francis Williams, Daniele Panozzo
  *Tuesday 2:00-3:30*

- **Differential Graphics with Tensorflow**
  Sofien Bouaziz, Martin Wicke, Julien Valentin, Paige Bailey, Josh Gordon, Christian Haene, Alexander Mordvintsev, Shan Carter
  *Thursday 9:00-12:15*
Examples in Graphics

Geometry

Image manipulation

Animation

Rendering
Examples in Graphics

Image manipulation
- Sketch simplification
- Colorization
- BRDF estimation

Rendering
- Real-time rendering
- Denoising

Geometry
- Mesh segmentation
- Procedural modelling

Animation
- Facial animation
- Boxification

Fluid
- Learning deformations

Procedural modelling
- PCD processing

CreativeAI: Deep Learning for Computer Graphics
Examples in Graphics

- Sketch simplification
- Colorization
- Procedural modelling
- Mesh segmentation
- Learning deformations
- Real-time rendering
- BRDF estimation
- Animation
- Boxification
- Denoising
- Fluid
- Facial animation
- PCD processing

CreativeAI: Deep Learning for Computer Graphics
Course Information (slides/code/comments)

http://geometry.cs.ucl.ac.uk/creativeai/