BTEC 424: Deep Learning Applications for Biomedical Image Analysis

Course Designer:
Jeffrey Robinson, MS, PhD
Translational Life Science Technology Program
UMBC at Universities at Shady Grove
9636 Gudelsky Dr., Rockville, MD, 20850
Phone: 301-335-4851
Email: jrobin2@umbc.edu

BTEC 424 (4 credit): Deep Learning Applications for Biomedical Image Analysis.
This is a senior-level capstone course for the UMBC TLST Bioinformatics Track. The course provides knowledgebase and practical experience in analysis of biomedical images and applications for Deep Learning/Neural Networks applications for biomedical image data. Images we will study include immunofluorescence and confocal microscopy images, clinical radiology images from X-rays and MRI, and tumor immunohistopathology. Class will use the open-source NIH ImageJ software to perform some common methods in image analysis such as nuclei segmentation and counting.

Deep Learning applications will be taught in two phases:
1. *Fundamentals of Deep Learning*. This will be taught by application of simple programming examples to standardized image datasets. Standardized datasets include the MNIST handwritten digits dataset, and the real-estate images and statistics dataset to generate simple but powerful neural network predictive models. Students will use Python programming with the Keras package to generate neural networks for image recognition with these datasets.

2. *Applications for Deep Learning algorithms for Biomedical image data*. Convolutional neural networks (CNNs) and Generative Adversarial Networks (GANs) will be trained on clinical histopathology data and immunofluorescence microscopy data, respectively. Python code and applications are adapted from the NIH Data Science open-source training materials. In this class we will use the NSF Jetstream cloud computing platform or AWS for these coding projects.

Prerequisites: BTEC330 (Software Applications), BTEC350/STAT350 (Biostatistics), BTEC395 (Bioinformatics), BTEC362 (Python programming).

Course Objectives:

A. **Module A**. Understand the types of biomedical images, the imaging technologies, and basic analyses for biomedical images and data.

B. **Module B**. Understand how to generate neural networks for image recognition and predictive modelling by applying Python programming language, the Keras package, and standardized datasets.

C. **Module C**. Apply Deep Learning methods to image analysis for clinical and biomedical images.

Textbook:
Deep Learning with TensorFlow 2 and Keras: Regression, ConvNets, RNNs, NLP, and more with TensorFlow 2 and the Keras API, 2nd. A. Gulli, A. Kapoor, S. Pal. Packt Publishing.

Grading:

Exam 1 (Week 4): 25%
Project 1 (Week 11): 25%
Project 2 (Finals Week): 25%
Homework: 20%
Attendance and Participation (at least five question/comment inside or outside of class): 5%

Course Outline (16 weeks):

A. **Module A: Introduction to Biomedical image data (Weeks 1-4).**

1. Immunofluorescent microscopy, histopathology, radiology. Image acquisition technology and data formats.
2. Basic nuclei segmentation and counting with the Open-source ImageJ software.

Exam1 (Week 4, 25%) – This exam will test the student's understanding of imaging technology in both research and clinical settings, and the understanding of basic image analysis algorithms such as nuclei segmentation and counting.

B. **Module B: Generate simple Neural Network models using standardized image datasets (Weeks 5-10).**
Students will learn to generate deep learning models by using standard datasets. These will include 1) generate a model that recognizes handwritten digits, and 2) Generate a model that predicts real estate values from a combination of numerical and image data. Assessments are example-based, students will work through the examples and post their code on GitHub and will be graded based on those submissions.

1. Generate a Convolutional Neural Network to identify handwritten digits using the MNIST dataset. Use the Python Keras package to generate a Convolutional Neural Network (CNN), teach of hand written characters (0-9).

   Exercises adapted from:
   a) "How to Develop a CNN for Handwritten Digit Classification". Machinelearningmastery.com. https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/

   b) "MNIST Dataset Python Example Using CNN". Towardsdatascience.com. https://towardsdatascience.com/mnist-cnn-python-c61a5bce7a19

2. Combine categorical, numerical, and image data into a single dataset for predicting real estate prices: Use the *house prices dataset* with the Python Keras package to create an integrated deep learning predictive model combining both image and numerical data.

   Exercises adapted from:
   a) "Regression with Keras". Pyimagesearch.com. https://www.pyimagesearch.com/2019/01/21/regression-with-keras/

   b) "Regression with Keras and CNNs". Pyimagesearch.com. https://www.pyimagesearch.com/2019/01/28/keras-regression-and-cnns/

   c) "Keras: Multiple Inputs and mixed data". Pyimagesearch.com. https://www.pyimagesearch.com/2019/02/04/keras-multiple-inputs-and-mixed-data/

Project 1 (Due Week 11, 25%) – Students will use Python, the Keras API, Jupyter Notebooks and GitHub to generate a deep learning model based on the examples above. The open-source functionality and coding practices will be graded by functionality and presentation of code that students develop on GitHub.

C. **Module C: Deep Learning Applications for Biomedical Research (Weeks 11-16).**

Exercises adapted from:

1. Bioimage segmentation using Convolutional Neural Networks (CNN).
   a) Content and homeworks adapted from: https://hpc.nih.gov/training/handouts/DL_by_Example1_20191031.pdf

2. Bioimage synthesis using Generative Adversarial Networks (GAN).
   a) Content and homeworks adapted from: https://hpc.nih.gov/training/handouts/DL_by_Example4_20200312.pdf

Project 2 (Due Finals Week, 25%) – Students will use Python and the Keras API to build predictive models based on clinical images by adapting the provided code into their own application, using command-line skills and NSF Jetstream or AWS cloud computing systems. Students will be graded on the performance of their code. Projects will be graded on the basis of the functionality of their code and application of proper coding practices as delineated in the assignment description.