Daanish M Mohammed

US Citizen | 470-815-3490 | daanishmd98@gmail.com | github.com/dmo98

Education

Georgia Institute of Technology, Atlanta, GA Aug 2021 – May 2023
MS in Computer Science, Specialization: Machine Learning GPA: 3.9/4.0

Birla Institute of Technology and Science (BITS Pilani), Hyderabad, India Aug 2016 – May 2020
BE in Electronics and Communications Engineering GPA: 8.22/10.0

Skills

Languages: Python, Java, MATLAB, C, SQL, HTML, CSS
Libraries: PyTorch, TensorFlow, NetworkX, Ray Tune, OpenCV, scikit-learn, NumPy, Flask
Technologies & Tools: Git, Docker, Kubernetes, AWS SageMaker, Google Cloud Platform, MySQL, Tableau

Experience

Data Science Intern May 2022 – Aug 2022
Koch Industries (Georgia-Pacific LLC) Atlanta, GA
- Worked on implementing a License Plate Reader using Computer Vision Algorithms.
- Implemented the following tasks: (1) Annotating a dataset of images of trucks. (2) Training a YOLOv5 model on the truck dataset and using the predicted bounding box to extract license plates via cropping. (3) Perform a homography on the cropped license plates and run Optical Character Recognition (OCR) on it to extract the text.

Graduate Teaching Assistant Jan 2022 – May 2023
Georgia Institute of Technology, CS 4641/7641 Machine Learning Atlanta, GA
- Responsibilities include creating and grading homework assignments, solving conceptual and programming doubts on EdStem and in weekly one-to-one Office Hour sessions, and hosting project seminars for students

Undergraduate Researcher Aug 2019 – Dec 2019
BITS Pilani Hyderabad, India
- Worked on Time-Frequency domain detection of heart valve disorders from PCG signals using a CNN.
- Pre-processed raw PCG audio data from the PASCAL Heartsound Database to extract cardiac cycles.
- Generated Time-Frequency matrices by applying the Smoothed Pseudo Wigner-Ville Distribution to the cardiac cycles.
- Trained a CNN on the Time-Frequency matrices to classify PCG signals into 4 classes based on the heart valve disorder present. It achieved a training accuracy of 83.75% and a validation accuracy of 84.37%.

Projects

Evaluation of GNNs for Graph Classification without Node Attributes (Python, PyTorch, Ray Tune) Link
- Trained several Graph Neural Network models, such as GCN, GAT and GraphSAGE, on the REDDIT-BINARY dataset to learn to distinguish and classify the interaction between users on a Reddit post based on the underlying graph structure into 2 categories: (i) Question/Answer-based subreddit posts and (ii) Discussion-based subreddit posts.
- Evaluated the performance of these models on the Graph Classification task using accuracy and AUCROC score as metrics. We found that GraphSAGE (with 'add' as the aggregation operation) achieved the best performance, with an accuracy of 76.7% and an AUCROC score of 0.840.

Generation of Car Images using Generative Models (VAEs and GANs) (Python, TensorFlow, OpenCV) Link
- Built and trained VAE and DCGAN models on the Stanford Cars Dataset to generate synthetic images of cars.
- Applied pre-processing steps such as cropping, resizing, and rescaling to prepare the image data for the models. Used latent vectors of size 512 and 100 for the VAE and DCGAN models respectively.

Pneumonia Detection from Chest X-Ray Images (Python, TensorFlow) Link
- Built and trained a Deep CNN to classify Chest X-Ray images as 'Normal' or 'Pneumonia'. Performed hyper-parameter optimization to obtain a model configuration with a train accuracy of 97.6%, and validation accuracy of 92.6%
- Used Transfer Learning to fine-tune pre-trained AlexNet, VGG-16, and GoogLeNet networks on the data. Compared the performance of these networks using the validation accuracy, precision, recall, f1-score and false negative rate as the metrics.

Airport Management System (Python, mysql-connector-python, Flask, flask-mysqldb) Link
- Created a Database application to demonstrate certain functionalities an Airport Manager would be in charge of.
- Constructed an EER Diagram and mapped it to a Relational Schema. Created and populated the database. Designed a simple interface and had the backend access the database to run queries.