Shri Keshavinee R

Contact:
Email: shrikeshavinee@gmail.com
LinkedIn: linkedin.com/in/Keshavinee
Github: github.com/Keshavinee

Education:
I'm pursuing an undergraduate degree in computer science and engineering in Sri Sivasubramaniya Nadar College of Engineering, Chennai. My CGPA is 9.054. The courses attended are Algebra and Calculus, Complex Functions and Laplace Transforms, Discrete Mathematics, Programming in Python, C, Java, Data Structures, Digital Principles and System Design, Unix and Shell Programming, Operating Systems, Computer Organization & Architecture.

I'm also pursuing an online bachelor's degree in Data Science in Indian Institute of Technology, Madras. I've completed a diploma in programming with CGPA 8.5. The courses attended are Probability and Statistics, Mathematics for Machine Learning, Computational Thinking, Modern Application Development, Database Management Systems, Machine learning foundations, Tools in Data Science, Business Data Science and Machine learning techniques.

Skills:
- Languages: C,C++, Python, Java, HTML, CSS, JavaScript, SQL, Bash
- Frameworks: Numpy, Scikit, Pandas, Java Swing, Flask
- Tools: GIT, PostgreSQL, MySQL, SQLite, Tableau
- Platforms: Linux, Windows

I'm proficient in the above mentioned skills.

Projects:
1. I participated in the ImageCLEF task (Multimedia Retrieval in CLEF - 2022) and published a paper on "A Fusion Approach for Web Search Result Diversification Using Machine Learning Algorithms". Result diversification provides a broader view of a search topic while also increasing the possibilities of finding relevant information. It has been shown to increase user satisfaction in recommender systems and web searches. Many

different approaches have been proposed in the related literature for the diversification problem. Since the web search result is enormous, it is essential to have an efficient fusion approach. Hence, the objective of this paper is to propose the implementation of a fusion model based on KNN, CART, and SVR regressors. This fusion model aims to improve the accuracy and reduce the error value of the generated result. To improve the accuracy of the results of the inducers, three regressors were implemented in the voting regressor. The model was trained on data from 56 different inducers, containing 167,139 training values, and tested on data from 56 inducers, containing 175,591 testing values. The base regressors obtained MAE values of 0.004 for KNN, 0.003 for CART, and 0.085 for SVR. The voting regression yielded an MAE of 0.017. Among the implemented regressors, CART provided the optimized result. Of the 10 best submissions, the best F1 score and CR score are 0.5634 and 0.4414 respectively. After publishing this paper, I had great learning in different machine learning models. Link to the paper

2. It is about bringing together a cane, smartphone camera, and basic projection system to turn any wall in the classroom into a smart wall. Teachers who use a basic projector (NOT SMARTBOARDS) to teach in the classroom are bound to the computer to control and navigate a lesson. Sometimes the systems do not face the classroom forcing teachers to have their back to the classroom making it difficult to manage the environment. We developed a tool by leveraging computer vision libraries, "OpenCV", and "media pipe" to enable a simple computer and projection systems to behave like a smartboard leveraging commonly available local tools The solution should include the following features:

   1) Leverage Machine Vision enabling computers that are controlled by human hands by recognizing writing on the board using hands and such and digitally recognizing them as inputs.

   2) Detects the hands like a mouse pointer to click or interact with what is displayed.

   This project helped me to learn Machine learning concepts and Python very well. Link to the project

3. I did a project on a Graphical Password Authentication system with my friends. We developed a web application. I took care of the backend part,

used the Django framework (written in Python) and wrote code in javascript for client-side scripting. This project helped me to learn Python, HTML, CSS, and Javascript. We used a recognition-based graphical password scheme. It creates a platform for the user to select pictures from a variety of images provided, during authentication the user is asked to recognize the previously selected images to gain access. We are yet to publish the paper.

Awards:

Winner at Smart India Hackathon 2022 - won cash prize of Rs. 100000

Research Interests:

I'm planning to do an MS in Robotics and Computer Vision. As the first step toward my immediate academic goal, I took on a project on building a robotic arm for the removal of weeds in a farm field. In the Indian agricultural industry, weedicides are sprayed on the crops collectively without taking into consideration whether weeds are present. Due to this, crop-yielding plants are also damaged. To avoid such situations and to improve food quality and productivity, it is necessary to have a system that removes only the weeds. With the help of an automatic weed removal system using a robotic arm, it is possible to remove the weed more precisely rather than spray weedicide over the whole field. First, a mapping module will be designed using Robot Operating System (ROS) and a camera with raspberry pi. Then an image segmentation system will be designed using a deep learning model for semantic segmentation of a captured image. The position of weeds will be determined using ROS. And a robotic arm will be employed to pluck the weeds.

Problem statement:

The nature of dark matter remains one of the fundamental mysteries of the universe. Strong gravitational lensing offers a promising probe of the substructure of dark matter, but identifying images containing such substructure has proven challenging. Deep learning methods have shown potential in accurately identifying images containing substructure, but further development is needed to differentiate between various dark matter models. This project aims to further develop the DeepLense pipeline to accurately estimate dark matter properties using deep regression techniques.

The plan is to conduct an extensive literature review to gain a deeper understanding of strong gravitational lensing and deep learning methods used in identifying substructure of dark matter. Next, strong lensing simulation data from lenstronomy will be acquired and prepared for deep learning model training. Deep regression models will be designed and developed to accurately estimate population-level quantities and properties of dark matter particle candidates, and the models will be trained and evaluated using appropriate metrics. Model hyperparameters will be optimized, and the trained models will be interpreted to gain insights into the features important for accurately estimating dark matter properties. Finally, a detailed report and a presentation of the findings will be prepared.

The deliverables for this project include a set of deep regression models that accurately estimate dark matter properties, a report detailing the methodology, results, and conclusions, and a presentation of the findings to the team and other stakeholders. The models will be designed to differentiate between various dark matter models, including WIMP particle dark matter, vortex substructure of dark matter condensates, and superfluids. Additionally, the project will contribute to the development of the DeepLense pipeline and lenstronomy simulations, which can be used for further research in the field of astrophysics.

Timeline:

Week 1: Literature review

- Conduct an extensive literature review to gain a deeper understanding of strong gravitational lensing and deep learning methods used in identifying substructure of dark matter
- Familiarize with the DeepLense pipeline and lenstronomy simulations

Week 2-3: Data preparation

- Acquire and prepare strong lensing simulation data from lenstronomy for deep learning model training
- Develop techniques to differentiate between various dark matter models, including WIMP particle dark matter, vortex substructure of dark matter condensates, and superfluids.

Week 4-5: Deep learning model development

- Design and develop deep regression models for accurately estimating population-level quantities and properties of dark matter particle candidates

- Experiment with various deep learning architectures and regularization techniques

Week 6-8: Model training and evaluation

- Train the developed models on the prepared data set
- Evaluate the model performance using appropriate metrics such as mean squared error and R-squared

Week 9-10: Hyperparameter tuning

- Optimize model hyperparameters using techniques such as grid search and random search to improve model performance

Week 11: Model interpretation and visualization

- Interpret the trained models to gain insights into the features important for accurately estimating dark matter properties
- Develop visualizations to aid in interpreting the results

Week 12: Project wrap-up

- Write a detailed report of the project, including the methodology, results, and conclusions
- Prepare a presentation and present the findings to the team and other stakeholders

Technical details:

I trained a convolutional neural network (CNN) model using the Keras API for TensorFlow for the given dataset.

Here is a brief overview of the model:

- The first layer is a convolutional layer with 32 filters, a filter size of 3x3, and ReLU activation function. The input shape of the layer is the shape of one sample in the training data.
- The second layer is a max pooling layer with a pool size of 2x2.
- The third layer is another convolutional layer with 64 filters, a filter size of 3x3, and ReLU activation function.
- The fourth layer is another max pooling layer with a pool size of 2x2.
- The fifth layer is another convolutional layer with 128 filters, a filter size of 3x3, and ReLU activation function.
- The sixth layer is another max pooling layer with a pool size of 2x2.

- The seventh layer is a flatten layer that flattens the output of the previous layer into a 1D vector.
- The eighth layer is a dense layer with 64 neurons and ReLU activation function.
- The ninth and final layer is a dense layer with 1 neuron and no activation function specified.

```
#Model architecture
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu',input_shape=X_train.shape[1:]))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(1))
```

I used the following code to train a neural network model using the Adam optimizer with a learning rate of 0.001 and the mean squared error loss function. The model is trained for 25 epochs with a batch size of 32 and then evaluated on a separate test set to compute the model's performance in terms of the MSE metric. Finally, the trained model is saved to a file for future use.

```
# Compile the model
model.compile(optimizer=Adam(lr=0.001), loss='mean_squared_error')

# Train the model
model.fit(X_train, y_train, batch_size=32, epochs=25)

# Save the trained model
model.save('lensing_model.h5')

# Evaluate the model on the test set
mse = model.evaluate(X_test, y_test)
print('Test MSE:', mse)
```

This model provided a mean squared error of 30.11.