# **ONT | Task-3**

## **Transfer Learning**

## **Introduction**

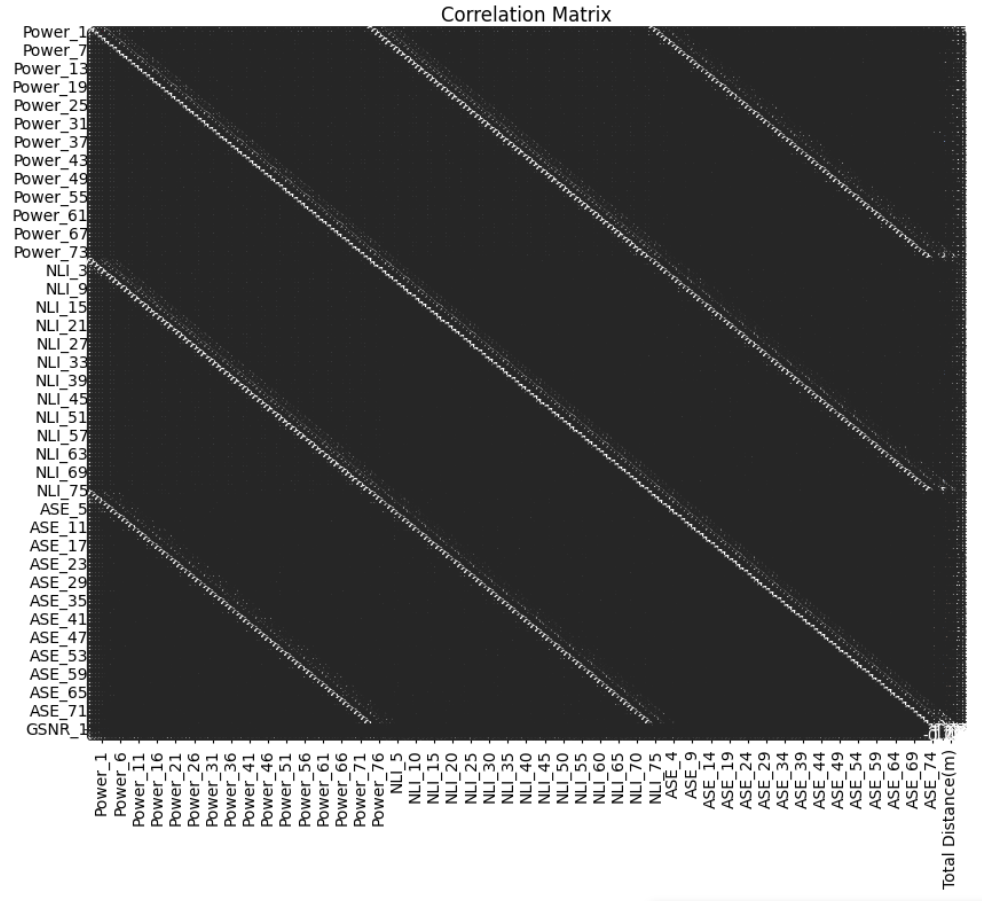
In this project, we explored the application of transfer learning techniques to predict General Signal-to-Noise Ratio (GSNR) using datasets from Europe and the USA. Transfer learning leverages pre-trained models and adapts them to new, yet related tasks, thus reducing the need for extensive training on large datasets. We specifically employed two primary techniques: feature extraction and fine-tuning.

## **Datasets**

Two datasets were utilized:

1. **European Dataset**: Served as the source dataset to train the initial model.
2. **USA Dataset**: Used for fine-tuning the pre-trained model to predict GSNR accurately in a different geographical context.

Both datasets contained features such as power, NLI (Nonlinear Interference), ASE (Amplified Spontaneous Emission) etc. which were essential for predicting GSNR.



## **Data Preparation**

**Preprocessing**

* **Missing Values**: Handled missing values by either dropping rows or filling them with mean values.
* **Feature Selection**: Dropped irrelevant columns such as 'frequency\_1 to frequency\_76' and 'GSNR\_2 to GSNR\_76', as well as 'Source', 'Destination', and 'Number of ON channels'.
* **Normalization**: StandardScaler was used to scale both the features and target variables to ensure uniformity and enhance model performance.

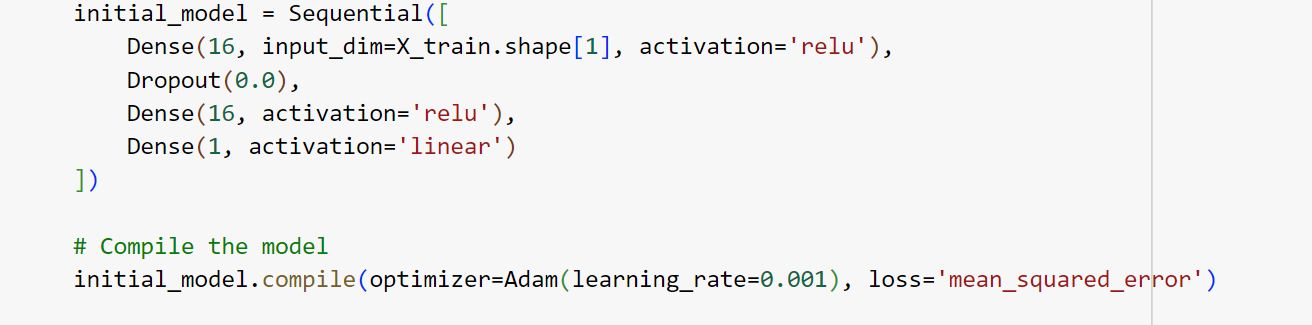
**Techniques and Libraries**

1. **Pandas**: For data manipulation and preprocessing.
2. **Scikit-Learn**: For data splitting, scaling, and evaluation metrics.
3. **TensorFlow and Keras**: For building, training, and fine-tuning deep learning models.
4. **Matplotlib and Seaborn**: For visualizing data distributions, correlations, and model performance.

## **Model Training**

**Initial Model Training on European Dataset**

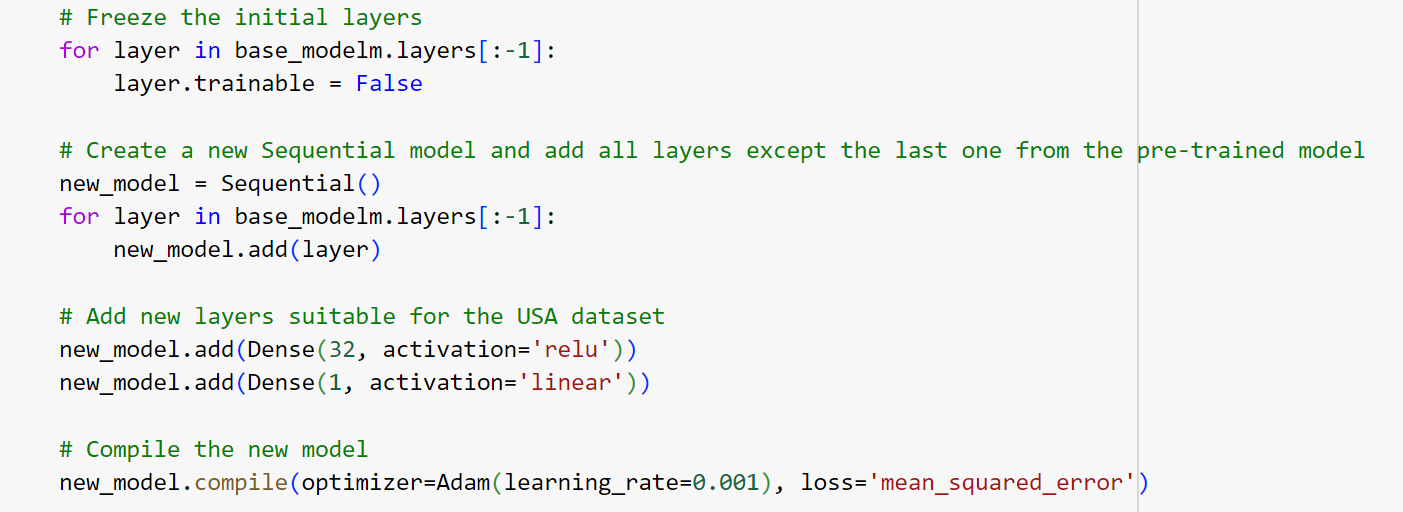
* **Architecture**: A neural network with three layers was used.
  + Input layer: Dense layer with 64 units and ReLU activation.
  + Hidden layer: Dense layer with 32 units and ReLU activation.
  + Output layer: Dense layer with 1 unit and linear activation.
* **Optimization**: Adam optimizer with a learning rate of 0.001.
* **Loss Function**: Mean Squared Error (MSE).
* **Training**: The model was trained for 100 epochs with a batch size of 32.



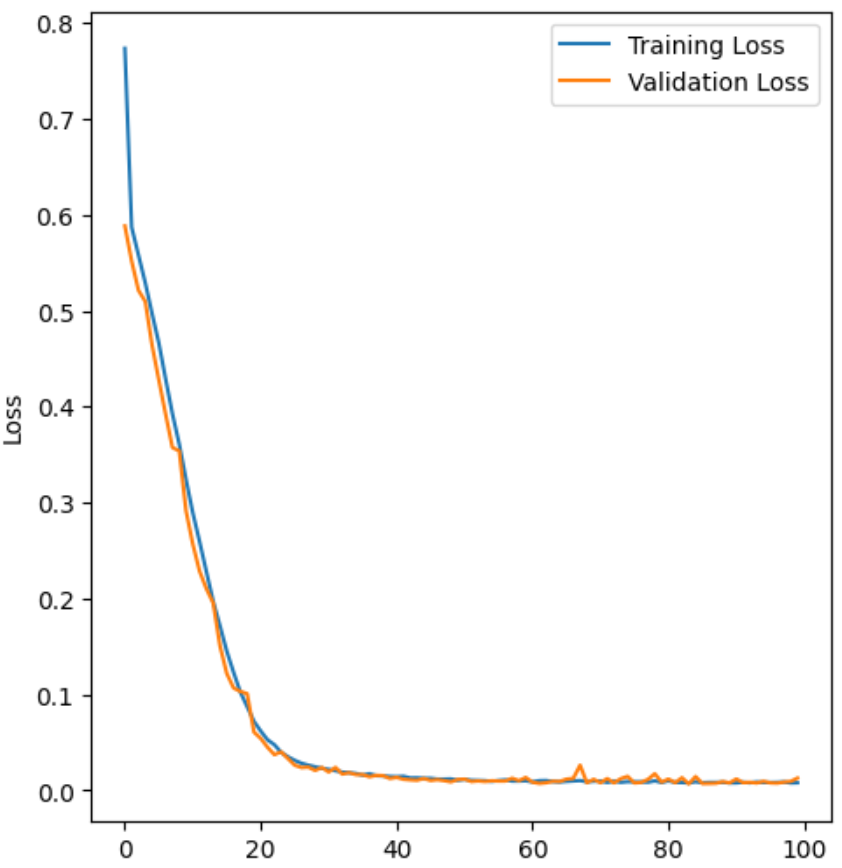
## **Transfer Learning**

**Feature Extraction**

* **Procedure**:
  + The initial model's layers, except for the last layer, were frozen.
  + New layers were added: a dense layer with 32 units and ReLU activation, followed by an output layer with 1 unit and linear activation.
  + The new model was compiled and trained on the USA dataset for 100 epochs.

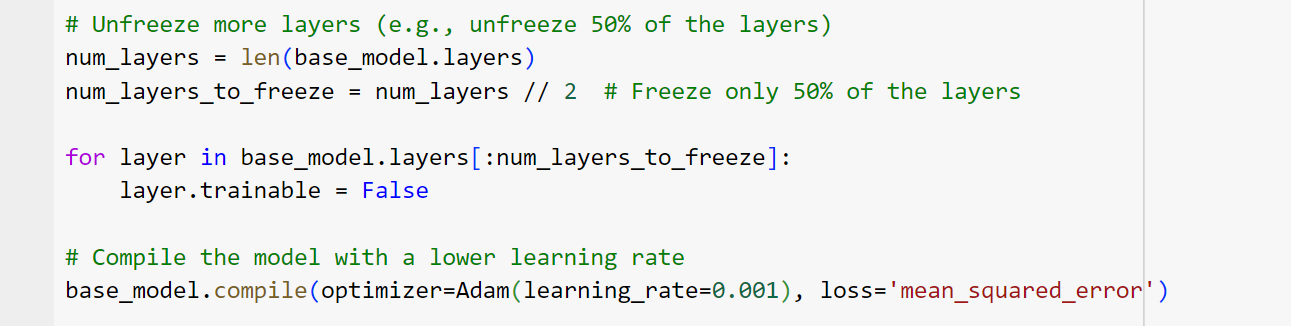


* **Results**:
  + MSE: 0.0128
  + R²: 0.9868

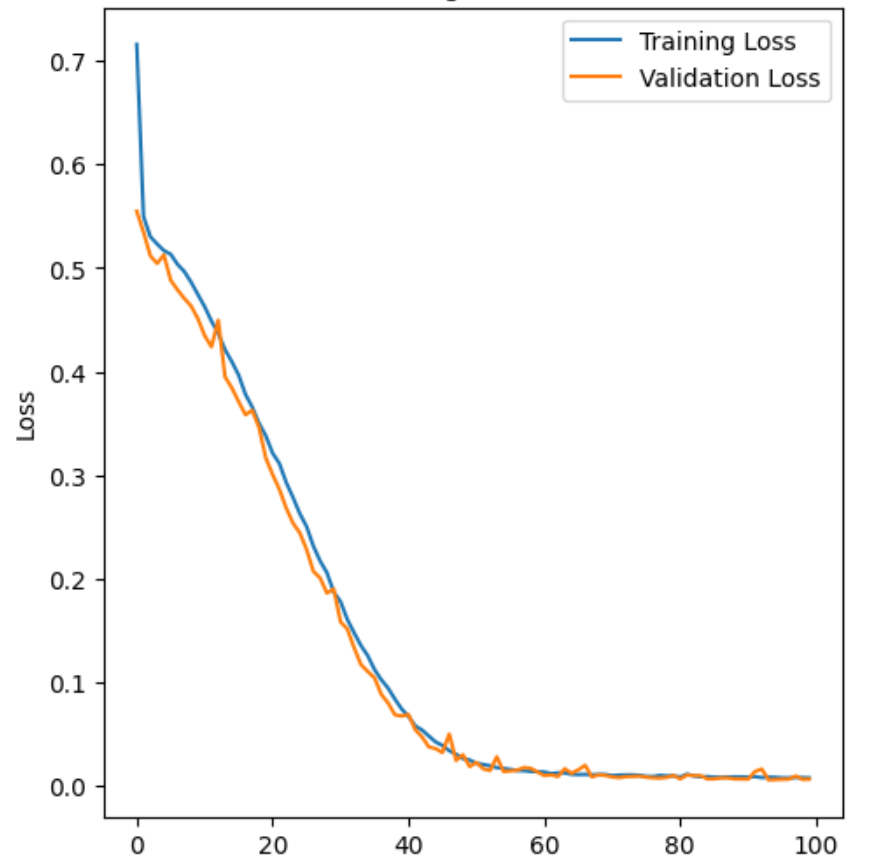


**Fine-Tuning**

* **Procedure**:
  + Some of the initial layers were unfrozen to allow for re-training.
  + The entire model was fine-tuned on the USA dataset with a reduced learning rate (0.001).
  + Training was conducted for 100 epochs.



* **Results**:
  + MSE: 0.0064
  + R²: 0.9934



## **Evaluation**

Both techniques, feature extraction and fine-tuning, were evaluated based on Mean Squared Error (MSE) and R² score on the validation set. The fine-tuning method outperformed feature extraction, achieving a lower MSE and higher R² score, indicating better predictive accuracy and generalization to the USA dataset.

## **Conclusion**

The project demonstrated the effectiveness of transfer learning in enhancing model performance with limited data availability for the target task. Fine-tuning, in particular, significantly improved the model's predictive capabilities by allowing selective re-training of the pre-trained model. The use of advanced deep learning libraries like TensorFlow and Keras, along with robust preprocessing techniques, was crucial in achieving these results.