**Knowledge Distillation for GSNR Prediction**

# **Objective**

The goal of this task is to explore and implement active learning techniques to improve the prediction of General Signal-to-Noise Ratio (GSNR). The objective is to iteratively select the most informative data points from the European or USA dataset to train a model, aiming to reduce the amount of labeled data needed to achieve high performance.

# **Understanding Knowledge Distillation**

Knowledge distillation is a machine learning technique where a smaller, simpler model (student) is trained to mimic the behavior of a larger, more complex model (teacher). This approach helps in transferring knowledge from a complex model to a simpler one, maintaining performance while reducing the model's size and computational requirements.

## **Principles**

* **Teacher Model**: A large, well-performing model trained on the original dataset.
* **Student Model**: A smaller model that learns to replicate the performance of the teacher model.
* **Distillation Process**: The student model is trained using both the ground truth labels and the soft targets (probability distributions) provided by the teacher model.

## **Benefits**

* **Model Compression**: Reduces the size of the model, making it more suitable for deployment on resource-constrained devices.
* **Inference Speed**: Smaller models often have faster inference times, which is beneficial for real-time applications.
* **Performance Retention**: Maintains a high level of performance despite the reduction in model complexity.
* **Resource Efficiency**: Lower memory and computational requirements, which can lead to cost savings in both training and deployment phases.

## **Uses**

* **Mobile and Edge Computing**: Deploying models on devices with limited computational power, such as smartphones and IoT devices.
* **Real-time Applications**: Enhancing the speed of applications requiring quick responses, such as video streaming, gaming, and autonomous driving.
* **Transfer Learning**: Adapting models to new tasks by transferring knowledge from a pre-trained teacher model to a student model.
* **Privacy-preserving Machine Learning**: Reducing the amount of data that needs to be shared or stored, which is beneficial in privacy-sensitive applications.

## **Disadvantages**

* **Loss of Interpretability**: The student model might be less interpretable than the teacher model, especially if it focuses on mimicking performance rather than understanding underlying data patterns.
* **Initial Cost**: Training the teacher model can be resource-intensive and time-consuming.
* **Hyperparameter Tuning**: Requires careful tuning of parameters such as the temperature in soft distillation, which can be challenging and time-consuming.
* **Potential Performance Degradation**: In some cases, the student model might not achieve the same level of performance as the teacher model, particularly if the distillation process is not well-executed.

## **Conclusion**

Knowledge distillation is a powerful technique for transferring knowledge from complex models to simpler ones, offering significant benefits in terms of model size, speed, and resource efficiency. However, it also comes with challenges, including potential performance trade-offs and the need for careful tuning. Understanding and effectively implementing knowledge distillation can lead to more efficient and scalable machine learning solutions.

# **Knowledge Distillation for European Dataset**

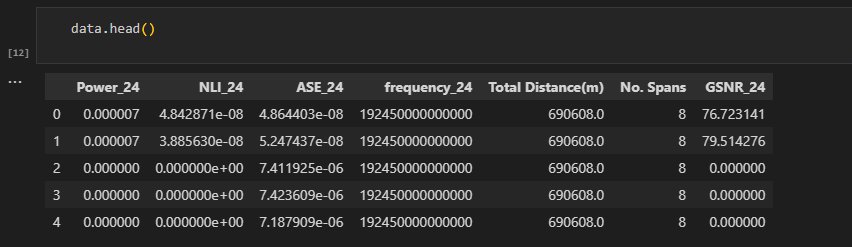
## **1. Data Preprocessing**

Data preprocessing is an important step, as it refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific task.

**We used channel 24 (gsnr\_24) as the target variable of European dataset.**

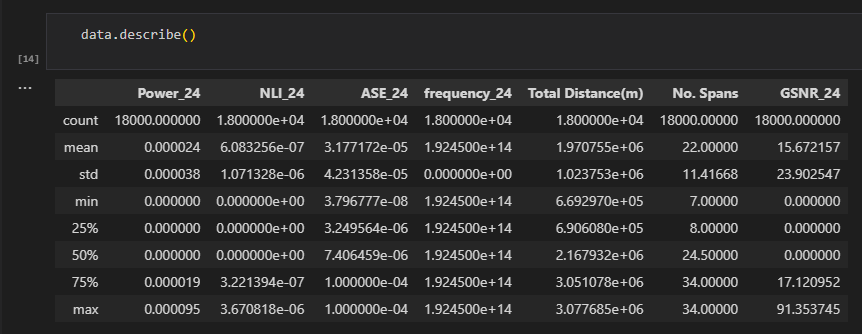
### **Examine first few rows and column types**

We started by examining the first few rows of the dataset using the data.head() method to get an initial overview of the data. All the columns were float64 except frequency and number of spans that were int.



### **Summary Statistics and look for null values**

Summary statistics were generated using data.describe(), which provided insights into the central tendency, dispersion, and overall distribution of the data. We also check for null values but there were no null values at all.



### **Feature Scaling**

Feature scaling is a crucial preprocessing step in many machine learning workflows. It ensures that all features contribute equally to the model's performance by putting them on a similar scale. This is particularly important for algorithms sensitive to the magnitude of feature values, such as Support Vector Machines, k-Nearest Neighbors, and neural networks.

In this task, we used the StandardScaler from the scikit-learn library to perform feature scaling. The StandardScaler standardizes features by removing the mean and scaling to unit variance, which transforms the data to have a mean of 0 and a standard deviation of 1.

By scaling the features, we enhance the performance of our machine learning models, ensuring they converge faster and perform better by treating all features equally, regardless of their original scale.

## **2. Model Training (Teacher Model)**

### **Model Architecture**

The teacher model was designed as a feedforward neural network with the following architecture:

* **Hidden Layer 1**: Dense layer with 128 units and ReLU activation function
* **Hidden Layer 2**: Dense layer with 64 units and ReLU activation function
* **Hidden Layer 3**: Dense layer with 32 units and ReLU activation function
* **Output Layer**: Dense layer with 1 unit and linear activation function

### **Compilation**

* **Optimizer**: Adam
* **Loss Function**: Mean Squared Error (MSE)
* **Metrics**: Mean Absolute Error (MAE)

### **Training Details**

* **Epochs**: 50
* **Batch Size**: 32
* **Validation Data**: Provided for evaluation during training

### **Training Results**

The teacher model was trained on the scaled training dataset and validated on the scaled validation dataset. The performance metrics on the validation set were recorded as follows:

* **Validation Loss**: 2.401
* **Validation MAE**: 0.584

### **Evaluation**

The validation loss and MAE indicate the model's performance on unseen data. The validation loss of 2.401 suggests that the model's predictions have a moderate degree of error when compared to the actual values. The MAE of 0.584 reflects the average absolute error of the model's predictions, indicating that, on average, the predictions deviate by 0.584 units from the true values.

## **3. Hard Knowledge Distillation**

### **Model Architecture**

* **Hidden Layer 1**: Dense layer with 64 units and ReLU activation
* **Hidden Layer 2**: Dense layer with 32 units and ReLU activation
* **Output Layer**: Dense layer with 1 unit and linear activation

### **Training Details**

* **Epochs**: 50
* **Batch Size**: 32
* **Validation Data**: Used for evaluation

### **Performance Metrics**

* **Validation Loss**: 2.928
* **Validation MAE**: 0.688

### **Conclusion**

The student model achieved a validation loss of 2.928 and a validation MAE of 0.688. These results suggest that the student model's performance is lower compared to the teacher model's metrics, indicating a need for further optimization or a different model configuration.

## **4. Soft Knowledge Distillation**

The soft knowledge distillation process was evaluated using different temperatures to soften the outputs from the teacher model. The performance of the student models was assessed based on validation loss and mean absolute error (MAE) for different temperature values.

### **Temperature values: 1, 5, 10**

### **Student Model Architecture:**

* Dense layers with ReLU activation
* Output layer with linear activation

### **Training Details**

* Epochs: 50
* Batch Size: 32
* Validation Split: 20%

### **Results**

|  |  |  |
| --- | --- | --- |
| Temperature | Validation loss | Validation MAE |
| 1 | 192.317 | 7.682 |
| 5 | 190.435 | 7.736 |
| 10 | 195.368 | 7.714 |

### **Analysis**

* **Temperature 1**: Provides the lowest validation loss but also results in the lowest MAE among the tested temperatures.
* **Temperature 5**: Shows a slight increase in loss and MAE compared to temperature 1 but still performs reasonably well.
* **Temperature 10**: Results in the highest validation loss and MAE, indicating that a higher temperature might not be as effective in this case.

### **Conclusion**

Among the tested temperatures, a temperature of 1 yields the best performance in terms of validation loss and MAE, making it the most effective configuration for soft knowledge distillation in this instance.

## **5. Time Comparison**

The performance of model training in terms of time and memory usage was assessed for the teacher model and student models using hard and soft knowledge distillation techniques.

|  |  |  |
| --- | --- | --- |
| Model Type | Training Time (s) | Memory Usage (MiB) |
| Teacher Model | 14.4286 | 986.0586 |
| Student Model (Hard Distillation) | 30.0792 | 986.8203 |
| Student Model (Soft Distillation) | 4.4004 | 987.2227 |

### **Analysis**

* **Teacher Model**: Takes the least time for training (14.43 seconds) but uses significant memory (986.06 MiB).
* **Student Model (Hard Distillation):** Requires the most time (30.08 seconds) and slightly more memory (986.82 MiB) compared to the teacher model.
* **Student Model (Soft Distillation):** Demonstrates the fastest training time (4.40 seconds) but has the highest memory usage (987.22 MiB), likely due to the combination of soft targets and ground truth.

### **Conclusion**

The student model using soft distillation provides the fastest training time, whereas hard distillation takes the longest. Memory usage is relatively similar across all models, with soft distillation slightly higher.

# **Knowledge Distillation for USA Dataset**

## **1. Data Preprocessing**

Data preprocessing is an important step, as it refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific task.

**We used channel 24 (gsnr\_24) as the target variable of USA dataset.**

### **Examine first few rows and column types**

We started by examining the first few rows of the dataset using the usa\_data.head() method to get an initial overview of the data. All the columns were float64 except frequency and number of spans that were int.

### **Summary Statistics and look for null values**

Summary statistics were generated using usa\_data.describe(), which provided insights into the central tendency, dispersion, and overall distribution of the data. We also check for null values but there were no null values at all.

### **Feature Scaling**

Feature scaling is a crucial preprocessing step in many machine learning workflows. It ensures that all features contribute equally to the model's performance by putting them on a similar scale. This is particularly important for algorithms sensitive to the magnitude of feature values, such as Support Vector Machines, k-Nearest Neighbors, and neural networks.

In this task, we used the StandardScaler from the scikit-learn library to perform feature scaling. The StandardScaler standardizes features by removing the mean and scaling to unit variance, which transforms the data to have a mean of 0 and a standard deviation of 1.

By scaling the features, we enhance the performance of our machine learning models, ensuring they converge faster and perform better by treating all features equally, regardless of their original scale.

## **2. Model Training (Teacher Model)**

### **Model Architecture**

The teacher model was designed as a feedforward neural network with the following architecture:

* **Hidden Layer 1**: Dense layer with 256 units and ReLU activation function
* **Dropout Layer**: Dropout layer with rate equals 0.2
* **Hidden Layer 2**: Dense layer with 128 units and ReLU activation function
* **Dropout Layer**: Dropout layer with rate equals 0.2
* **Hidden Layer 3:** Dense layer with 64 units and ReLU activation function
* **Dropout Layer:** Dropout layer with rate equals 0.2
* **Hidden Layer 4**: Dense layer with 32 units and ReLU activation function
* **Hidden Layer 5**: Dense layer with 16 units and ReLU activation function
* **Output Layer**: Dense layer with 1 unit and linear activation function

### **Compilation**

* **Optimizer**: Adam
* **Loss Function**: Mean Squared Error (MSE)
* **Metrics**: Mean Absolute Error (MAE)

### **Training Details**

* **Epochs**: 50
* **Batch Size**: 32
* **Validation Data**: Provided for evaluation during training

### **Training Results**

The teacher model was trained on the scaled training dataset and validated on the scaled validation dataset. The performance metrics on the validation set were recorded as follows:

* **Validation Loss**: 0.802
* **Validation MAE**: 0.309

### **Evaluation**

The validation loss and MAE indicate the model's performance on unseen data. The validation loss of 0.902 suggests that the model's predictions have a better degree of error when compared to the actual values. The MAE of 0.309 reflects the average absolute error of the model's predictions, indicating that, on average, the predictions deviate by 0.309 units from the true values.

## **3. Hard Knowledge Distillation**

### **Model Architecture**

* **Hidden Layer 1**: Dense layer with 64 units and ReLU activation
* **Dropout Layer:** Dropout layer with rate equals 0.2
* **Hidden Layer 2**: Dense layer with 32 units and ReLU activation
* **Dropout Layer:** Dropout layer with rate equals 0.2
* **Output Layer**: Dense layer with 1 unit and linear activation

### **Training Details**

* **Epochs**: 25
* **Batch Size**: 64
* **Validation Split**: 20%

### **Performance Metrics**

* **Validation Loss**: 0.598
* **Validation MAE**: 0.257

### **Conclusion**

The student model achieved a validation loss of 0.598 and a validation MAE of 0.257. These results suggest that the student model's performance is lower compared to the teacher model's metrics, indicating a need for further optimization or a different model configuration.

## **4. Soft Knowledge Distillation**

The soft knowledge distillation process was evaluated using different temperatures to soften the outputs from the teacher model. The performance of the student models was assessed based on validation loss and mean absolute error (MAE) for different temperature values.

### **Temperature values: 1, 5, 10**

### **Student Model Architecture:**

* Dense layers with ReLU activation
* Dropout layers with rate equals 0.2
* Output layer with linear activation

### **Training Details**

* Epochs: 25
* Batch Size: 64
* Validation Split: 20%

### **Results**

|  |  |  |
| --- | --- | --- |
| Temperature | Validation loss | Validation MAE |
| 1 | 1.058 | 0.778 |
| 5 | 0.988 | 0.767 |
| 10 | 1.052 | 0.772 |

### **Analysis**

### **Temperature 1:** Results in a validation loss of 1.058 and an MAE of 0.778. This temperature provides reasonable performance but is not the best for both metrics.

### **Temperature 5:** Achieves the lowest validation loss (0.988) and MAE (0.767). This configuration performs best overall, suggesting it strikes a good balance between model accuracy and error.

### **Temperature 10:** Exhibits a slightly higher validation loss (1.052) and MAE (0.772) compared to temperature 5. This indicates that increasing the temperature beyond 5 results in decreased model performance.

### **Conclusion**

## Temperature 5 yields the best performance in terms of both validation loss and MAE, making it the most effective temperature setting for soft knowledge distillation in this case. Temperatures 1 and 10 do not perform as well as temperature 5, with temperature 5 providing the optimal balance for distillation.

## **5. Time Comparison**

The performance of model training in terms of time and memory usage was assessed for the teacher model and student models using hard and soft knowledge distillation techniques.

|  |  |  |
| --- | --- | --- |
| Model Type | Training Time (s) | Memory Usage (MiB) |
| Teacher Model | 30.3243 | 2437.9922 |
| Student Model (Hard Distillation) | 26.4381 | 2461.3125 |
| Student Model (Soft Distillation) | 26.7504 | 2501.7500 |

### **Analysis**

### **Teacher Model:** Requires the most time for training (30.32 seconds) and uses the least memory (2437.99 MiB) among the models tested. This indicates that although the teacher model is more time-consuming, it is more efficient in terms of memory usage compared to the student models.

### **Student Model (Hard Distillation):** Takes slightly less time (26.44 seconds) and uses more memory (2461.31 MiB) than the teacher model. This suggests that hard distillation is relatively efficient in terms of time but has a higher memory demand.

### **Student Model (Soft Distillation):** Shows a similar training time to hard distillation (26.75 seconds) but with the highest memory usage (2501.75 MiB). This higher memory usage could be attributed to the complexity of handling soft targets combined with ground truth data.

### **Conclusion**

The teacher model, despite having the longest training time, is the most memory-efficient. In contrast, the student models, particularly the one using soft distillation, exhibit higher memory usage but similar training times. Among the student models, soft distillation has the highest memory consumption, while hard distillation is more time-efficient compared to the soft distillation model.

# **Challenges Faced and Solutions**

## **Challenges**

**1. Model Performance Variation**: Different temperatures for soft distillation yielded varying results in terms of validation loss and MAE. This variation made it challenging to select the optimal temperature for best performance.

* **Solution**: We systematically tested multiple temperatures and evaluated their impact on validation metrics. Based on this evaluation, we identified that a temperature of 5 provided the best balance between validation loss and MAE.

**2. Time and Memory Efficiency:** The models exhibited significant differences in training time and memory usage, with the teacher model being more memory-efficient while the student models required more resources.

* **Solution**: We measured and compared training times and memory usage for both hard and soft distillation methods. This allowed us to make informed decisions about the trade-offs between training efficiency and resource consumption.

# **Summary and Suggestions for Future Work**

## **Summary**

* **Model Performance**: Among the tested temperatures for soft knowledge distillation, a temperature of 5 provided the optimal performance with the lowest validation loss and MAE. This suggests that this temperature setting strikes the best balance for the soft distillation process.
* **Time and Memory Efficiency**: The teacher model was the most memory-efficient, although it had the longest training time. The student models, particularly those using soft distillation, showed higher memory usage but comparable training times. Hard distillation was slightly more time-efficient than soft distillation.

## **Potential Improvements and Future Work**

* **Hyperparameter Tuning**: Further exploration of different temperature values and other hyperparameters could refine the performance of both soft and hard distillation methods. This might involve testing additional values or combinations of parameters.
* **Model Optimization**: Investigate ways to optimize the student models to reduce memory usage without compromising performance. Techniques such as model pruning or quantization could be explored.
* **Extended Evaluation:** Conduct additional experiments with different model architectures or additional datasets to assess the generalizability of the findings and improve the robustness of the models.
* **Enhanced Training Strategies**: Implement advanced training techniques, such as adaptive learning rates or more sophisticated regularization methods, to potentially enhance both training efficiency and model performance.