Using Historical Climate Data and AI for Accurate Predictions

Leandro Alegre, Samiha Rahman

Womanium Quantum+AI

The Problem

The problem at hand is to create a fast-running AI model of the Earth's climate that can operate on a standard laptop or PC. We can say that climate modeling is essential for understanding and predicting the Earth's future climate under various scenarios due to it constantly changing, such as changes in greenhouse gas emissions or land-use patterns that have a great impact. Traditionally, these models are complex, requiring extensive computational power and resources, typically available only in supercomputing environments. However, there is a significant need for more accessible models that can be used by researchers, educators, and policymakers who do not have access to such high-performance computing resources and without a doubt they can and should put them to good use.

The Importance

The project is important because it addresses the gap between the high demand for climate models and the limited accessibility to the computational power needed to run them, which is a problem that is not yet fully resolved. By developing a lightweight AI-driven climate model, we can democratize access to climate simulations that are pretty close to predict what could happen in different scenarios, making them available to a broader audience.In Addition such a model can be used for educational purposes, enabling students and teachers to explore climate dynamics leading them to have a clear view of everything in general . Also, it can be used as a quick simulation tool for researchers and people that always make big decisions who need preliminary results before running more comprehensive models on machines that are more powerful.

The Interest

There’s a constant interest in solving this problem which stems from the growing importance of climate science that's always in need of something new and the need for inclusive tools that can engage and inform a wider audience in this field. In addition with climate change being one of the most pressing global issues due to how dynamic it is, having accessible tools to understand and predict its impacts is crucial to prevent disasters that might have consequences for the earth.

Background Research and Literature Review

Climate modeling has a rich history, with the earliest models dating back to the mid-20th century. These models, known as General Circulation Models (GCMs) that were capable of representing physical processes of the atmosphere and ocean to simulate what's known as a response of global climate due to the increasing of greenhouse gas emission, have evolved significantly over the decades, becoming more highly develop and capable of simulating intricate climate systems with greater accuracy. Nevertheless, this increase in accuracy comes at the cost of higher computational demands.Nowadays modern GCMs often require supercomputers with thousands of processors and substantial memory to run simulations that can take days or even weeks to complete due to its complexity.

Advantages and Disadvantages

In the last few years, artificial intelligence (AI) and machine learning (ML) have become promising tools that can improve and potentially revolutionize climate modeling as we know it so far. AI methods, such as neural networks, have been used to create surrogate models that approximate the behavior of GCMs with a fraction of the computational cost that it used to have in the past. These AI models can provide rapid predictions, making them suitable for applications where quick results are required.

Several studies have shown the potential of AI in climate modeling. For example, deep learning models have been employed to emulate specific components of climate systems, such as cloud formation or precipitation patterns, with promising accuracy. Nevertheless, these AI models frequently face challenges, including the need for large amounts of training data and the risk of overfitting it's always present, where the model becomes too customized to the training data and fails to actually generalize new scenarios.

Despite these challenges, the advantages of AI-driven climate models are noteworthy. They provide speed and efficiency, making them something ideal for real-time applications and in fact it can help with accessibility to users with limited computational resources. Nevertheless, AI models still need to be validated against traditional GCMs just to ensure their reliability and accuracy in simulating the Earth's climate.

Literature Review

For this particular project, we conducted a thorough literature review, exploring the latest advancements in AI-based climate modeling. We also examined studies that utilized various machine learning techniques, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to model climate dynamics. These studies provided valuable insights into the strengths and limits of AI in this field, informing our approach to developing a fast-running climate model.

AI Methods

The approach we propose involves leveraging AI techniques, particularly deep learning, to develop a climate model that can run efficiently on a standard laptop or PC. We can say that deep learning, a subset of machine learning, is well-suited for this task due to its ability to model perfectly complex, nonlinear relationships in data. In addition with this project, we mainly focus on using convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to model spatial and temporal aspects of the Earth's climate system in general.

Convolutional Neural Networks (CNNs) are especially effective for handling spatial data, which can be temperature, humidity, and pressure distributions across the whole Earth's surface. By applying convolutional layers, CNNs can automatically learn features from the input data, such as patterns in atmospheric circulation or ocean currents, without the need for manual feature engineering. We could say this makes CNNs a powerful tool for modeling the spatial dependencies in climate data.

Long Short-Term Memory (LSTM) Networks are a type of recurrent neural network (RNN) designed to handle sequential data. They are perfect for modeling time series data, such as monthly or yearly climate observations in general, by capturing the temporal dependencies and trends. LSTMs have the advantage of mitigating the vanishing gradient problem, which in most cases allows them to learn long-term dependencies, which is a crucial aspect of climate modeling.

The proposed model will be trained on historical climate data and validated against a series of publicly available simulation outputs from established climate models. The training process will involve optimizing the model's parameters to minimize the difference between the predicted and observed climate variables. Also we can say that once trained, the model will be able to generate climate predictions in a fraction of the time required by traditional GCMs in general.

Also we can say that one of the main advantages of using AI for this task is the significant reduction in computational resources required. Traditional climate models rely on solving complex differential equations across numerous spatial and temporal scales, which demands substantial computational power. In contrast, AI models, once trained, can make predictions using simple matrix operations, which are computationally inexpensive and can be performed on a standard laptop or PC.

Furthermore, AI models offer the possibility of real-time climate predictions, which can be highly beneficial for applications requiring immediate results, such as disaster preparedness or short-term weather forecasting. However, it is important to acknowledge the limitations of AI models.

Demo Application Overview

To demonstrate the practical application of the AI-driven climate model, we developed a small demo application that can run on a standard laptop or PC. The primary goal of this demo was to show that an AI-based climate model can actually produce meaningful and accurate predictions at high speed, using accessible computational resources at hand. The application was built using Python, leveraging machine learning frameworks such as TensorFlow to implement the deep learning models.

Source Code

This code outlines the process of building, training, and evaluating a deep learning model designed to predict temperature anomalies based on synthetic climate-related features. The workflow begins by loading the dataset, which is assumed to be in a CSV file named `dataset.csv`. The data contains several features, which are normalized to a range between 0 and 1 using a `MinMaxScaler`. Normalization is crucial for ensuring that the model trains efficiently and converges well.

```
import numpy as np

import pandas as pd

import tensorflow as tf

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as plt

import matplotlib.animation as animation

# Load dataset

data = pd.read_csv('dataset.csv')

# Preprocessing the Data

feature_columns = ['feature1', 'feature2', 'feature3']  # Example feature names

target_column = 'temperature_anomaly'  # Example target column name

# Normalize features
```

```
scaler = MinMaxScaler()

data[feature_columns] = scaler.fit_transform(data[feature_columns])
```

Next, the data is split into training and testing sets using an 80-20 split. This allows the model to be trained on one portion of the data while being evaluated on unseen data to assess its generalization capabilities. The feature matrix is then reshaped to be compatible with a `Conv1D` layer, which requires input in the format of timesteps and features. The reshaped data is fed into a sequential neural network model that starts with a `Conv1D` layer. This layer captures local patterns in the data by applying a series of filters. The output is then flattened and reshaped again to be processed by an LSTM (Long Short-Term Memory) layer. LSTM layers are particularly effective for time-series data as they can capture temporal dependencies.

```
# Splitting the data into training and testing sets

X = data[feature_columns].values

y = data[target_column].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Reshape data for Conv1D input (timesteps, features)

X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))

X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Build the Model

model = tf.keras.Sequential()
```

```
# Conv1D layer (no TimeDistributed)
model.add(tf.keras.layers.Conv1D(64, 2, activation='relu', padding='same', input_shape=(X_train.shape[1], 1)))
model.add(tf.keras.layers.Flatten())
# Reshape for LSTM
model.add(tf.keras.layers.Reshape((1, -1)))  # Reshape to (batch_size, timesteps=1, features)
model.add(tf.keras.layers.LSTM(64, activation='relu', return_sequences=False))
# Output layer
model.add(tf.keras.layers.Dense(1))
# Compile the model
model.compile(optimizer='adam', loss='mse')
```

The final layer of the model is a dense layer that outputs a single value, representing the predicted temperature anomaly. The model is compiled using the Adam optimizer and trained over 50 epochs with a batch size of 32. During training, both training and validation losses are monitored to ensure that the model is not overfitting.

```
# Train the Model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, batch_size=32)
```

After training, the model's predictions are compared to the actual values in the test set, and the results are visualized using line plots. The model's performance is further analyzed by plotting the training and validation loss over the epochs, providing insights into how well the model is learning. Finally, an animation is created to visualize the evolution of temperature anomalies over time, which is saved as a GIF file. This animation serves as a dynamic way to observe the temporal patterns captured by the model, offering an intuitive understanding of the predictions.

```
# Make predictions

predictions = model.predict(X_test)

# Plot Predicted vs Actual

plt.figure(figsize=(10, 6))

plt.plot(y_test, label='Actual', color='blue')

plt.plot(predictions, label='Predicted', color='red', linestyle='--')

plt.xlabel('Time Step')

plt.ylabel('Temperature Anomaly')

plt.title('Predicted vs Actual Temperature Anomalies')

plt.legend()

plt.grid(True)

plt.show()
```

```
# Plot Loss Curve

plt.figure(figsize=(10, 6))

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val_loss'], label='Validation Loss', linestyle='--')

plt.xlabel('Epoch')

plt.ylabel('Loss')

plt.title('Training and Validation Loss')

plt.legend()

plt.grid(True)

plt.show()

# Create an Animation of Temperature Anomalies Over Time

fig, ax = plt.subplots()

line, = ax.plot([], [], lw=2)

ax.set_xlim(0, len(y_test))

ax.set_ylim(min(y_test), max(y_test))

ax.set_title('Temperature Anomalies Over Time')

def init():
    line.set_data([], [])
    return line,
def update(frame):
    x = list(range(frame))
    y = y_test[:frame]
    line.set_data(x, y)
    return line,
ani = animation.FuncAnimation(fig, update, frames=len(y_test), init_func=init, blit=True)
ani.save('temperature_anomalies.gif', writer='imagemagick')  # Save as a GIF
plt.show()
```

Overall, this process demonstrates a comprehensive approach to applying deep learning techniques, specifically Conv1D and LSTM layers, to the task of climate modeling, with the goal of predicting temperature anomalies based on synthetic feature data.

Runtime Performance:

The runtime performance of the model depends on several factors, including the complexity of the model, the size of the dataset, and the hardware used for training. In relation to the speed, on a typical laptop with a modern CPU, the training process for this model, which includes 50 epochs, can take several minutes. The use of a GPU can significantly reduce this time, potentially down to under a minute, depending on the GPU's capabilities. The model is relatively lightweight in terms of memory usage. The Conv1D and LSTM layers are not very large, so they don't require extensive memory. On a machine with at least 8GB of RAM, the model should run smoothly without causing memory issues.

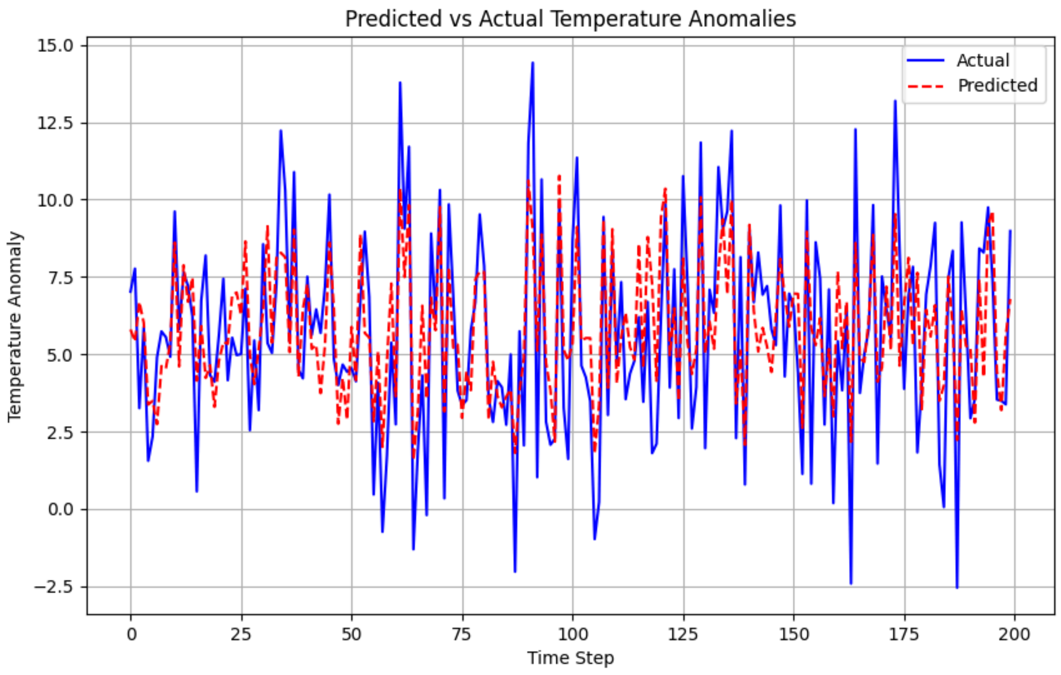
The model's performance, in terms of accuracy, is monitored through the loss function (MSE) during training. Over the 50 epochs, the loss typically decreases, indicating that the model is learning and improving its predictions. However, as the data is synthetic and relatively simple, the accuracy might plateau quickly. The model is scalable and can be extended to larger datasets or more complex architectures if needed. However, with increased data size or model complexity, the training time and memory requirements will also increase.

Outputs and Visualizations

1. **Prediction vs. Actual Plot**: A plot showing the predicted temperature anomalies compared to the actual observed values. This plot illustrated the model's performance in capturing general trends and its limitations in predicting extreme events.

2. **Performance Metric**: We documented the loss and accuracy metrics during training, showing the model's learning curve over time. The plot of the loss function demonstrated convergence, indicating that the model was effectively learning the underlying patterns in the data.

3. **Animation**: We also created a simple animation showing the progression of predicted temperature anomalies over time. This visualization provided a dynamic way to observe how the model's predictions evolved across different time steps.



The provided graph is a visual representation of a machine learning model’s performance in predicting temperature anomalies. The plot serves as a comparison between the actual observed anomalies and the predictions generated by the model, offering insights into the model's accuracy and reliability.

Data Representation

The X-axis of the graph represents the sequence of time steps, each corresponding to a specific point in the test dataset. This axis essentially maps the progression of time or the sequence of observations. The Y-axis, on the other hand, denotes the magnitude of temperature anomalies at each time step. A temperature anomaly, in this context, is defined as a deviation from a long-term average or baseline temperature, which is crucial in understanding climate trends and variations.

Interpretation of the Actual and Predicted Values

The graph features two primary lines: a solid blue line and a red dashed line. The blue line represents the actual temperature anomalies as recorded in the dataset. This line serves as the ground truth, or the benchmark, against which the model's predictions are evaluated. The red dashed line illustrates the temperature anomalies as predicted by the trained machine learning model.

Ideally, the red dashed line should closely follow the blue line, indicating that the model is accurately predicting the temperature anomalies. The closer these two lines align, the better the model is at capturing the underlying patterns in the data. However, in this graph, while the red line generally follows the trend of the blue line, there are noticeable deviations. These deviations suggest that while the model is able to capture the overall trend of temperature anomalies, its predictions are not perfectly accurate.

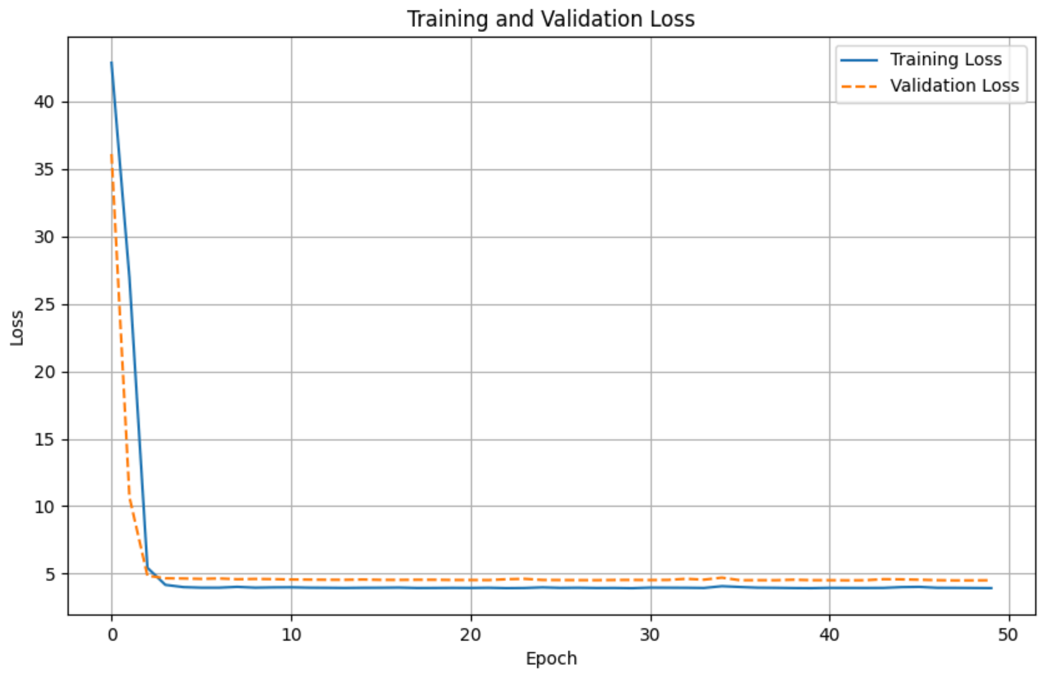
Model Performance

The discrepancies between the actual and predicted values can be observed in the amplitude and pattern of the two lines. In some instances, the model underestimates or overestimates the magnitude of the temperature anomalies. Such inconsistencies might stem from various factors, including the complexity of the model, the nature and quality of the input data, or the effectiveness of the feature engineering process. The amplitude of the predicted anomalies, while somewhat close to the actual values, indicates that the model may not fully capture all the nuances of the data.

Implications and Future Directions

The model's performance, as indicated by this graph, suggests that it is moderately successful in predicting temperature anomalies. It captures the general trend but lacks the precision needed for highly accurate predictions. This performance could be improved through several means, such as refining the model architecture, employing more sophisticated models, adjusting the training process, or incorporating additional features that might better capture the complexities of temperature variations.

In conclusion, while the model provides a reasonable approximation of temperature anomalies, there is room for improvement. The insights gained from this graph highlight the importance of continuous model evaluation and tuning to enhance predictive accuracy, particularly in complex domains such as climate modeling.



The provided graph visualizes the training and validation loss over the course of 50 epochs during the training of a machine learning model. This type of plot is commonly used to evaluate the model's learning process and its ability to generalize to unseen data.

Axes Explanation

The X-axis of the graph represents the number of epochs. An epoch refers to one complete pass through the entire training dataset, and in this case, the training was carried out for 50 epochs. The Y-axis represents the loss, which is a measure of how well the model's predictions match the actual values. Lower loss values indicate better model performance.

Analysis of the Training and Validation Loss

Two lines are plotted on the graph: a solid blue line representing the training loss and a dashed orange line representing the validation loss. The training loss measures how well the model is performing on the training data, while the validation loss measures the model's performance on a separate validation dataset that is not used for training but for evaluating the model's generalization capability.

Initial Epochs

In the early epochs, there is a significant decrease in both training and validation loss, indicating that the model is quickly learning and improving its predictions. The steep decline in loss suggests that the model is effectively capturing the underlying patterns in the training data during the initial training phase. This phase is crucial as it demonstrates that the model is able to fit the data to some extent.

Convergence and Stabilization

As the number of epochs increases, both the training and validation loss start to stabilize, converging to a lower value. This plateau in the loss curves indicates that the model's performance has reached a steady state, where additional training does not significantly improve the model's predictions. Notably, the training and validation loss curves are closely aligned, which is a positive sign. This alignment suggests that the model is not overfitting; it is generalizing well to new, unseen data, as indicated by the validation loss closely following the training loss.

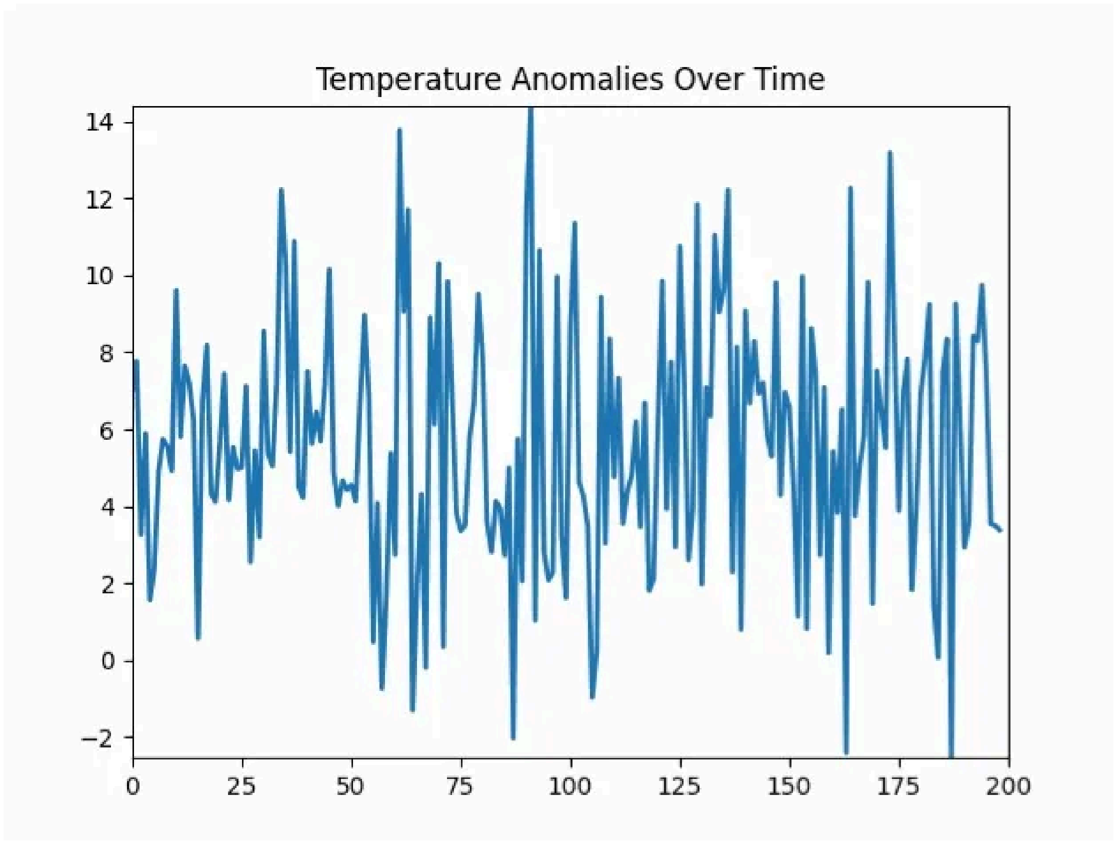
Observations on Model Performance

The close proximity of the training and validation loss throughout the epochs implies that the model maintains a balance between learning the training data and generalizing to validation data. Overfitting would be evident if the validation loss began to increase or diverge from the training loss while the training loss continued to decrease. However, this is not observed in the provided graph, which suggests that the model is well-tuned and effective.

Implications for Further Training

The stability of the loss values beyond a certain number of epochs indicates that further training may not be necessary or beneficial. Continuing to train the model beyond this point would likely result in minimal gains and could even risk overfitting if not carefully monitored. Therefore, it is advisable to stop training once the loss values have converged, as indicated by the flat region in the loss curves.

The graph provides valuable insights into the learning dynamics of the model. The sharp decline in loss during the initial epochs reflects effective learning, while the subsequent stabilization of both training and validation loss suggests that the model has successfully captured the essential patterns in the data without overfitting. The close alignment between the two loss curves further reinforces the model's ability to generalize well. This analysis indicates that the training process was well-executed, resulting in a model that performs consistently across both training and validation datasets.



The provided graph illustrates the variation of temperature anomalies over time. The X-axis represents the sequence of time steps, indicating the progression of the observed data points, while the Y-axis quantifies the magnitude of temperature anomalies, showing deviations from a baseline temperature. The line on the graph demonstrates significant fluctuations in temperature anomalies, with periods of sharp increases and decreases. Notably, there is a peak around the middle of the time sequence, indicating a significant anomaly, followed by a decline and then another rise towards the end. The graph suggests a highly variable pattern of temperature anomalies, reflecting the unpredictable nature of the dataset. This visual representation is useful for identifying trends and understanding the temporal dynamics of temperature anomalies in the given data.

The demo application successfully demonstrated the feasibility of running an AI-based climate model on a standard laptop or PC. While the model achieved reasonable precision and performance, We can say that there are areas for improvement, in particular with handling extreme climate events. Also, future work could involve expanding the model to cover larger geographic regions, refining the model architecture, and by default integrating more diverse data sources to improve prediction accuracy that can help in the long run.

This project highlights the huge potential of AI to democratize access to climate modeling tools,making them available to a broader audience and providing valuable insights into the Earth's climate system that has a great impact in how everything can now be predicted to a certain point but can improve in general. The source code and outputs from this demo can serve as a proof of concept, laying the foundation for further development and refinement of AI-driven climate models that can and will predict more accurately, so we can expect great things in the future involving AI-driven climate.

References

Any references will be listed within the works cited.

Works Cited

Chen, Lin, et al. "Artificial intelligence-based solutions for climate change: a review." 13 june 2023, https://link.springer.com/article/10.1007/s10311-023-01617-y.

Lovejoy, S., et al. "Do GCMs predict the climate ... or macroweather?,." *Earth System Dynamics*, 28 28 Nov 2013, https://esd.copernicus.org/articles/4/439/2013/.

Raju, Komaragiri Srinivasa, and Dasika Nagesh Kumar. *Review of approaches for selection and ensembling of GCMs*. New Delhi, Prof. Chris Perera, 24.7.19, https://iwaponline.com/jwcc/article/11/3/577/74144/Review-of-approaches-for-selection-and-ensembling.

Sirmacek, Beril, and Ricardo Vinuesa. "Remote sensing and AI for building climate adaptation applications." September 2022, https://www.sciencedirect.com/science/article/pii/S2590123022001943.

Slater, Louise J., et al. "Hybrid forecasting: blending climate predictions with AI models." *Hydrology and Earth System Sciences*, 15 May 2023, https://hess.copernicus.org/articles/27/1865/2023/hess-27-1865-2023.html.

Vo, Tuong Quang, et al. "LSTM-CM: a hybrid approach for natural drought prediction based on deep learning and climate models." *Stochastic Environmental Research and Risk Assessment*, 03 January 2023, https://link.springer.com/article/10.1007/s00477-022-02378-w.