Modeling and Forecasting U.S. CO_2 Emissions[*]

Integrating ARIMA and LSTM Approaches to Capture Seasonal and Temporal Dynamics

Alex Wang Jiacan Sun Kevin You Ziheng Zhong

June 22, 2025

This paper analyzes monthly U.S. CO_2 emissions from 1973 to 2025 using linear regression enhanced by ARIMA and LSTM models to capture seasonal and temporal dependencies. We identify key drivers of emissions, such as transportation petroleum and renewable energy production, and illustrate significant forecasting improvements from integrating autoregressive methods. Results underscore the importance of accounting for time-series structures and suggest hybrid models provide more accurate predictions essential for informed environmental policy-making.

Table of contents

[*] Code and data supporting this analysis is available at: Link to repository.

1 Introduction

Climate change has become a critical global challenge, driven largely by rising carbon dioxide (CO_2) emissions from human activities. The Intergovernmental Panel on Climate Change (IPCC) warns that without immediate and deep emission reductions, the world is on track to exceed the Paris Agreement's 1.5°C warming threshold, underscoring an urgent need to curb CO_2 from all sectors (Intergovernmental Panel on Climate Change (IPCC) 2022). In this context, the United States – historically the single largest national source of CO_2 – plays a pivotal role. Cumulatively, the U.S. has emitted roughly 25% of all fossil-fuel CO_2 since the industrial era began (Ritchie 2019), and it remains one of the top annual emitters today. Understanding the trajectory of U.S. emissions, especially from major sectors like transportation, industry, and energy generation, is therefore of global importance for climate change mitigation.

The motivation for this research is grounded in climate and energy policy relevance. The chosen sectors – transportation, industrial production, and power generation – are the dominant sources of U.S. greenhouse gas emissions (Keerthana et al. 2023). For example, in 2019 the transportation sector accounted for about 29% of U.S. CO_2-equivalent emissions, followed by electricity generation (25%) and industrial processes (23%) (Keerthana et al. 2023). These activities collectively drive the bulk of national emissions, meaning any meaningful climate strategy must address each of them. Analyzing historical patterns in these sectors can reveal how past economic growth, technological changes, and policies (such as vehicle efficiency standards or power plant regulations) have impacted emissions. Moreover, forecasting future emissions is crucial for gauging progress toward sustainability goals. The U.S. has set ambitious targets under the Paris Agreement – pledging a 50–52% reduction in greenhouse gases by 2030 (from 2005 levels) and net-zero emissions by 2050 (Keerthana et al. 2023) – which heightens the real-world significance of this study. By projecting emissions trajectories, we can assess whether current trends align with these climate goals or if additional policy interventions may be required.

This paper will analyze monthly data from the U.S. Energy Information Administration (U.S. Energy Information Administration 2025), focusing on trends in energy consumption and production across major sectors. Our response variable is total CO_2 emissions, modeled as a function of six key predictors: transportation petroleum consumption, fossil fuel production, renewable energy generation, and energy consumption by the commercial, industrial, and residential sectors. Given the temporal nature of the data, we employ an ARIMA model and Long Short-Term Memory (LSTM) model, which will be discussed in more detail later. These two modeling approach allows us to assess both the cross-sectional and time-dependent dynamics influencing emissions, also enabling us to do forecasting .

The paper is structured as follows: Section 3 describes our data sources, preprocessing steps, and modeling approaches including ARIMA and LSTM techniques; Section 4 presents the results of our models along with diagnostic checks and forecast evaluations; Section 5 discusses implications, limitations, and potential future improvements based on our findings.

2 Literature Review

2.1 Traditional Statistical Models

Time series models like ARIMA and its seasonal version, SARIMA, have been frequently used to forecast CO emissions because they effectively handle seasonal and linear patterns. For instance, ARIMA models have successfully forecasted emission trends in countries like China and South Africa, clearly showing the steady increase in emissions and helping inform policy decisions Tian et al. (2025). However, these models often struggle with unexpected changes, such as sudden policy shifts or economic shocks, since they assume the data follow predictable, linear patterns. Therefore, ARIMA alone might not fully capture complex real-world emission trends Tian et al. (2025).

2.2 Machine Learning and Deep Learning Models

To overcome the limitations of traditional models, researchers have increasingly used deep learning methods, particularly LSTM and GRU neural networks. These approaches excel at capturing complex patterns and long-term relationships in emission data. A recent study forecasting India's CO emissions demonstrated that an LSTM model performed better than traditional ARIMA and other statistical models, achieving significantly lower prediction errors Kumari and Singh (2023). Similarly, a comparative study found deep learning models (like LSTM and GRU) consistently outperform statistical methods, especially when emissions data show irregular fluctuations or complex trends Ajala et al. (2025). These studies suggest that deep learning can greatly enhance forecasting accuracy.

2.3 Hybrid Models

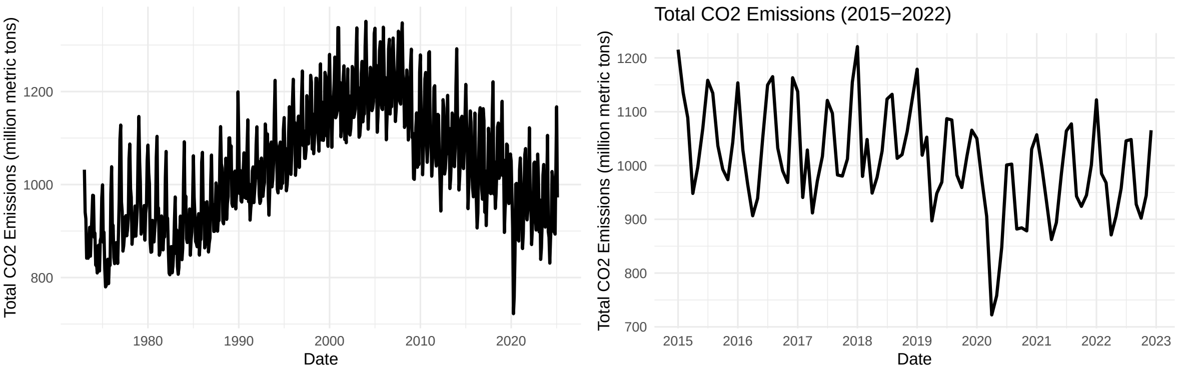
Recognizing the benefits of both traditional and deep learning methods, hybrid models have become popular. These models typically use ARIMA to capture straightforward seasonal and linear trends, while neural networks address more complex patterns. Zhang (2003) initially demonstrated the value of combining ARIMA and neural networks, showing improved accuracy over either model alone Zhang (2003). Recently, hybrid models using ARIMA and LSTM have successfully forecasted China's CO emissions, outperforming standalone models by capturing both regular seasonal cycles and unpredictable fluctuations Wen et al. (2023). Our paper adopts this hybrid approach, leveraging the strengths of ARIMA for simple patterns and LSTM for complex ones, building upon these prior successes to provide accurate forecasts of U.S. CO emissions.

3 Method

3.1 Data

The dataset is from the U.S. Energy Information Administration's (EIA) Monthly Energy Review, which issues detailed energy statistics, including production, consumption, trade, and emissions metrics. The Monthly Energy Review is released on the last workday of each month. It contains preliminary data that are subject to revision in subsequent releases and historical data that may be updated when source publications are revised. For this project, data were extracted for the period ranging from January 1973 to February 2025, with a total of 626 records at a monthly frequency. The EIA collects these data through standardized survey forms completed by energy industry participants, following review protocols approved every three years to ensure consistency and quality across reporting entities. The dataset (U.S. Energy Information Administration 2025) is sourced from the the U.S. Energy Information Administration's (EIA) Monthly Energy Review that issues detailed energy statistics, including production, consumption, trade, and emissions metrics. The Monthly Energy Review is released on the last workday of each month and contains preliminary data that are subject to revision in subsequent releases, as well as historical data that may be updated when source publications are revised. For this project, data were extracted for the period ranging from January 1973 to February 2025, with a total of 626 records at a monthly frequency. The EIA collects these data through standardized survey forms completed by energy industry participants, following review protocols approved every three years to ensure consistency and quality across reporting entities.

The cleaned dataset is constructed by concatenating monthly U.S. data from different categories. It includes energy production, sectoral energy use, transportation fuel use, and CO_2 emissions. Emissions from other sources are aggregated by date, and the total emissions across all sectors are calculated. Additional statistics on petroleum consumption, fossil fuel and renewable energy output, and residential, commercial, and industrial energy consumption are



(a) Total CO2 Emissions Over Time

(b) Total CO2 Emissions (2015–2020)

Figure 1: Data Overview

merged after standardized Year and Month variables have been extracted. All datasets are joined in year and month.

The resulting dataset captures key indicators such as total CO_2 emissions (million metric tons), transportation petroleum consumption (million barrels per day), total fossil fuel and renewable energy output (quadrillion Btu), and sectoral energy consumption (trillion Btu). Units adhere to the conventions of the Monthly Energy Review, with energy metrics predominantly expressed in U.S. customary units—such as Btu for energy and barrels for petroleum volumes—and emissions in metric units.

From plot 1, we can say that total CO_2 emissions have a clear upward trend from the early 1970s through 2005 and then a decay from 2005 to the present. The sharp dip around 2020 likely reflects the economic slowdown during the COVID-19 pandemic. During this period, widespread lockdowns resulted in a significant reduction in transportation, industrial activity, and fossil fuel consumption, which contributed to the observed decline in emissions.

The ACF plot shows that the total CO_2 emission is non-stationary since the lag-1 autocorrelation is significant and the ACF decays very slowly. This confirms the presence of a non-stationary mean in the series.

Key variables included in the cleaned dataset are total CO_2 emissions (million metric tons), transportation petroleum consumption (million barrels per day), total fossil fuels production and total renewable energy production (quadrillion British thermal units [Btu]), and sectoral energy consumption in the commercial, industrial, and residential sectors (trillion Btu). Units adhere to the conventions of the Monthly Energy Review, with energy metrics predominantly expressed in U.S. customary units—such as Btu for energy and barrels for petroleum volumes—and emissions in metric units.

From plot 1, we can say that from the early 1970s through 2005, total CO_2 emissions has a clear upward trend, and then a decay from 2005 to present. The sharp dip around 2020 likely

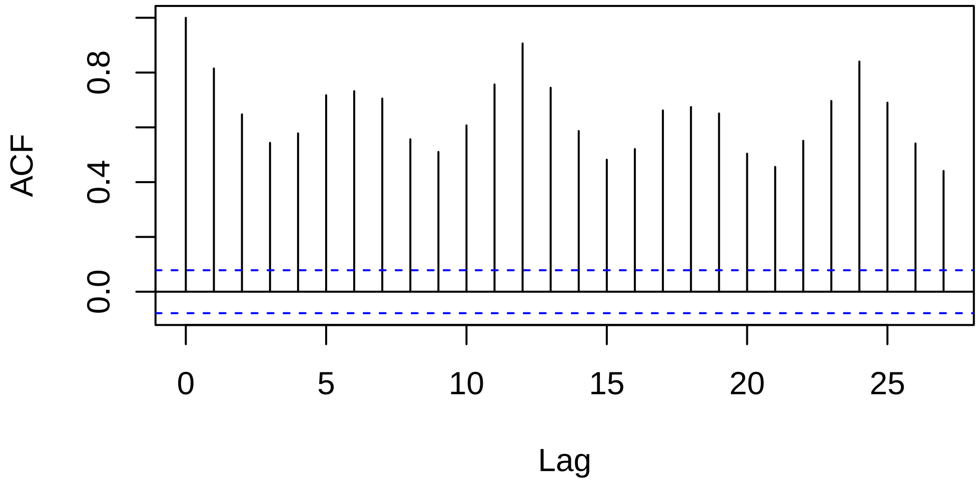


Figure 2: ACF of Total CO2 Emissions

reflects the economic slowdown during the COVID-19 pandemic, everyone is in quarantine, hence there is less traffic and fossil fuel usage, which could be the reason of the dip.

By observing the ACF plot, we can conclude that the total CO_2 emission is non-stationary since the lag-1 autocorrelation is essentially 1.0, and the ACF decays very slowly, confirming the presence of non-stationary mean in the series.

3.2 Models

To model the relationship between total CO_2 emissions and the set of covariates we picked:

y_t - Total CO_2 emissions

z_{t1} - Transportarion Petroleum Consumption

z_{t2} - Total Fossil Fuels Production

z_{t3} - Total Renewable Energy Production

z_{t4} - Commercial_Consumption

z_{t5} - Industrial_Consumption

z_{t6} - Residential_Consumption

while accounting for serial dependence in the errors, we followed the four-step procedure for regression with auto-correlated errors:

First, we fit an ordinary regression of y_t on $z_{t1}, \cdots, z_{t6}$ (acting as if the errors are uncorrelated).

$$y_t = \sum_{j=1}^{6} \beta_j z_{tj} + x_t$$

From this fit we can retain the residuals by

$$\hat{x}_t = y_t - \sum_{j=1}^{r} \beta_j z_{tj}$$

After fitting the initial linear regression model, we examined the residuals to check for autocorrelation using autocorrelation (ACF) and partial autocorrelation (PACF) plots, as well as the Box-Ljung test. These diagnostics showed strong and persistent correlation patterns, indicating that the residuals were not independent over time.

Table 1: Model 1 Summary

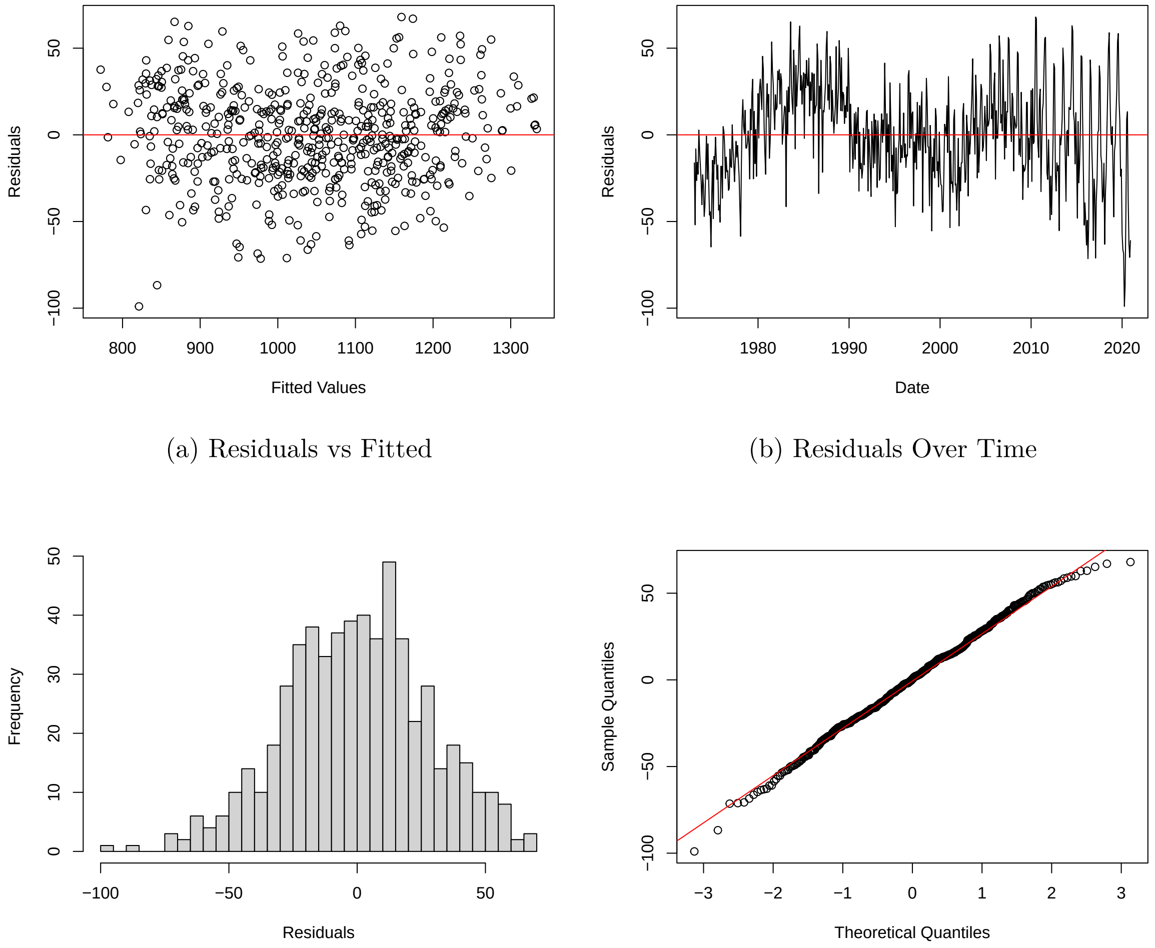
Variable	Estimate	Std. Error	t value	Pr(>|t|)	Significance
(Intercept)	294.0642	21.0351	13.9797	0.0000	***
Transportation	1.2353	0.2103	5.8735	0.0000	***
Fossil Fuel Prod.	-38.0689	4.3954	-8.6610	0.0000	***
Renewable Energy Prod.	-256.3311	26.5226	-9.6646	0.0000	***
Commercial	0.5023	0.0133	37.8640	0.0000	***
Industrial	0.1639	0.0081	20.1718	0.0000	***
Residential	0.0015	0.0058	0.2667	0.7898	

In our case, our model is

$$y_t = 294.1 + 1.235 z_{t1} - 38.07 z_{t2} - 256.3 z_{t3} + 0.5023 z_{t4} + 0.1639 z_{t5} + 0.001546 z_{t6} + x_t$$

In the scatter plot of residuals against the fitted values (Figure 3a), there is no clear funnel or curvature, which suggests that the variance of the errors is roughly constant across the range of predicted CO_2 emissions.

However, when we plot the residuals in time order, a pronounced seasonal “saw tooth” pattern emerges-residuals swing positive and negative in a regular annual pattern (Figure 3b). This



(a) Residuals vs Fitted

(b) Residuals Over Time

(c) Histogram of Residuals

(d) Normal Q-Q Plot

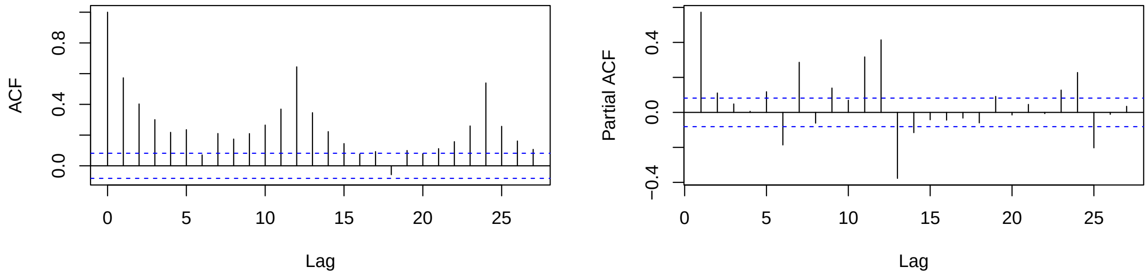
Figure 3: Model 1 Diagnostic Plots

indicates that our purely cross-sectional predictors haven't captured the strong yearly cycle in emissions, and so that seasonality remains in the error term.

The histogram of residuals (Figure 3c) is roughly symmetric around zero and appears approximately bell-shaped, albeit with mildly heavier tails than a perfect normal distribution.

The Q-Q plot (Figure 3d) confirms this: most points lie very close to the 45° line, indicating near-normality, but the extreme ends (both lower and upper tails) deviate slightly. In practice, these small departures from Gaussianity are unlikely to invalidate inference, especially once we explicitly model the autocorrelation and seasonality.

Then we examined the sample ACF and PACF of $\hat{x}_t$ to identify candidate ARMA(p,q) structures. Potential orders p and q were chosen by the usual cut-off and tail-off patterns in these plots. Finally we applied `forecast::auto.arima()` to select an appropriate ARIMA(p,d,q) model



(a) ACF of Residuals

(b) PACF of Residuals

Figure 4: Model 1 ACF & PACF

Figure 4a shows the autocorrelation of residuals from our linear model. We observe significant spikes at multiple lags, especially early ones, which exceed the blue dashed significance bands. This means the residuals are not independent over time, and there is clear serial correlation remaining in the model. This suggests that our linear model has not fully captured the temporal structure of the data.

Figure 4b shows partial autocorrelations of the residuals. We see large spikes at the first few lags, indicating that past values have a direct influence on the current residuals, even after accounting for previous lags. This supports the idea that an autoregressive structure may still be present and needs to be modeled, possibly with an ARIMA or time series component.

Table 2: Box-Ljung Test on Model Residuals

	Test	Statistic	df	p.value
X-squared	Box-Ljung	967.69	20	<0.001

The Ljung-Box test (Table 2) checks whether residuals are independently distributed. Here, the test statistic is 967.69 with 20 degrees of freedom, and the p-value is less than 0.001. This result strongly rejects the null hypothesis of no autocorrelation, confirming that the residuals still contain time-dependent patterns and the current model is insufficient to capture them.

Table 3: ARIMA Model Summary on OLS Residuals

Metric	Value
Sigma^2	468.2533
Log Likelihood	-2581.0715
AIC	5178.1431
AICc	5178.3975
BIC	5212.9780
ME	-0.0920
RMSE	21.4884
MAE	16.7951
MPE	93.5415
MAPE	256.5873
MASE	0.8007
ACF1	0.0280

To address issues mentioned above, we applied an **ARIMA model** to the residuals from our linear regression. We used the `auto.arima()` function to select the best-fitting ARIMA model for the residuals. More details and results are presented in Section 4

We also explored a machine learning approach by using a **Long Short-Term Memory (LSTM)** neural network. LSTM models are particularly good at handling sequential data because they can effectively capture long-term patterns without needing the data to be stationary. To better fits the need of this model, Year and Month are converted and combined into a single column in datetime format. Key features related to CO_2 emissions are then selected. This includes energy consumption and production metrics. Moreover, we applied MinMax normalization to both the features and the target variable to manage scale differences to ensure the LSTM performs optimally.

We then created input sequences using a sliding window approach, where each sequence includes 24 months of historical data to predict the CO_2 emissions of the next month. After preparing the data, we split it into three sets: training, validation, and testing. Specifically, data before 2021 was used for training, data from 2021 served as the validation set, and data from 2022 onward was used for testing and prediction. To ensure consistent and reproducible results, we set a random seed before training the model. Our final model included two LSTM layers with dropout layers added to reduce overfitting, followed by a dense layer that provided the final CO_2 emission predictions. Additionally, we applied an early stopping mechanism during training, which stops the process if improvements in validation loss become negligible over consecutive epochs. This further helps prevent overfitting and improves the reliability of the model. Full evaluation of results are provided in Section 4

4 Result

4.1 Model 1

Continuing with Method section, our `auto.arima()` call picked an ARIMA(5,1,2) model on the OLS residuals $\hat{x}_t$. Concretely, letting $\nabla\hat{x}_t = \hat{x}_t - \hat{x}_{t-1}$, the fitted model is

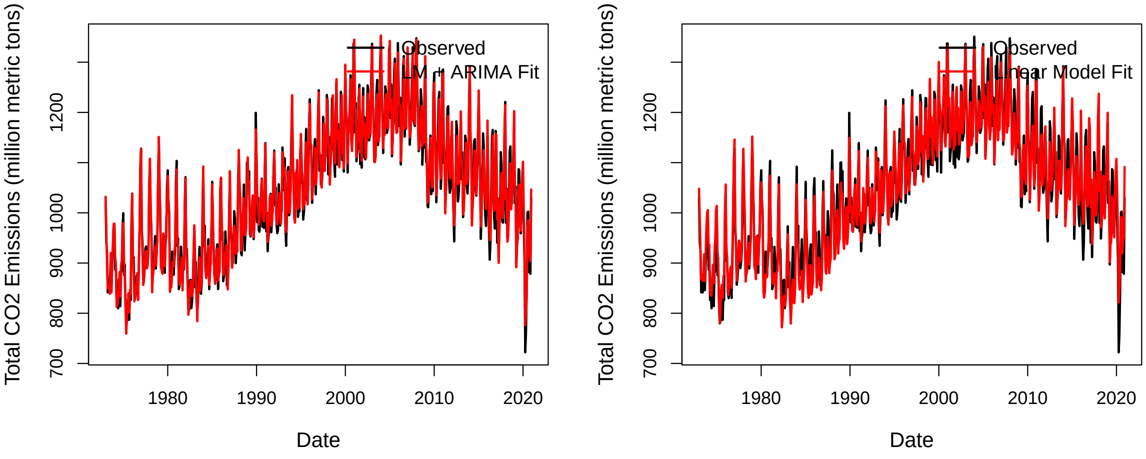
$$(1-0.2594B+0.3250B^2+0.0259B^3-0.1480B^4+0.1365B^5)(1-B)\hat{x}_t = (1-0.2066B-0.6959B^2)w_t$$

where B is the backshift operator and w_t is white noise with estimated $\sigma^2(w_t) = 468.3$. AIC of 5178.1 confirms this is the most parsimonious high-order fit `auto.arima` found, and the small training-set ACF1 (≈ 0.028) suggests remaining serial dependence is minimal after differencing and fitting these parameters.

Table 4: ARIMACoefficients and Standard Errors

	Term	Estimate	Std_Error
ar1	ar1	-0.2594	0.0618
ar2	ar2	0.3250	0.0494
ar3	ar3	0.0259	0.0466
ar4	ar4	-0.1480	0.0456
ar5	ar5	0.1365	0.0465
ma1	ma1	-0.2066	0.0473
ma2	ma2	-0.6959	0.0418

Figure 5 compares the performance of two models in capturing U.S. CO_2 emissions over time. In both plots, the black line represents the observed emissions, while the red line shows the model predictions. Figure 5a shows the combined model (linear + ARIMA), which closely follows the seasonal and long-term trends, especially in the mid and later periods. Figure 5b displays the linear model alone, which captures the overall trend but misses the recurring seasonal spikes and dips.



(a) Combined Model Fit

(b) Linear Model Fit vs Actual

Figure 5: Model Fittings

Table 5: Comparison of Linear Model vs. Linear + ARIMA Model

Model	RSS	RMSE
Linear Model	451898.4	28.0097
Linear + ARIMA Model	265967.9	21.4884

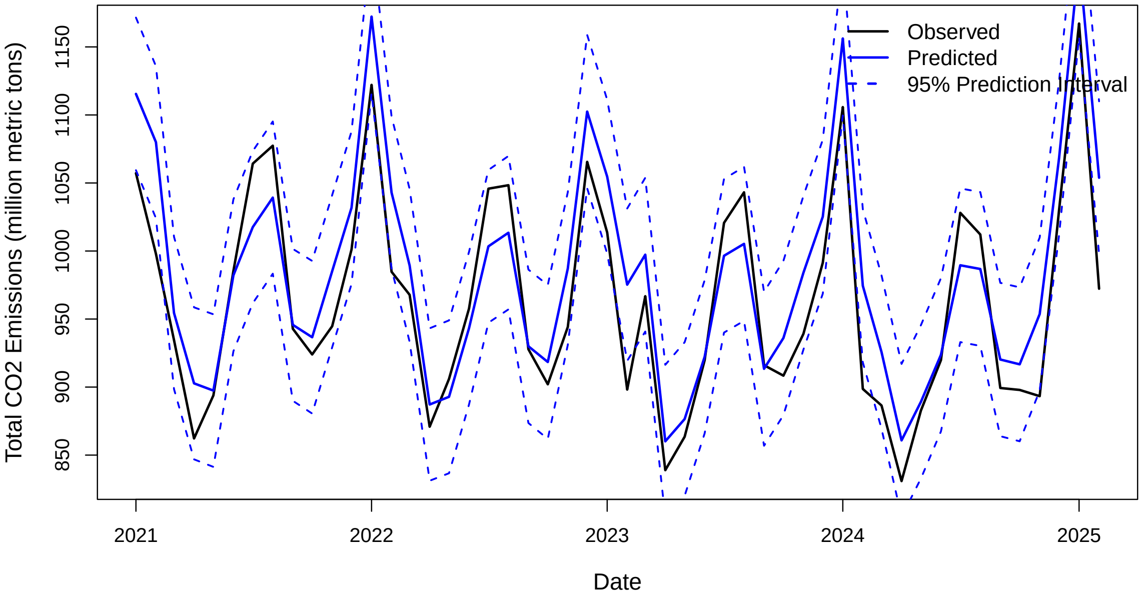


Figure 6: RSS and RMSE Comparison of Linear vs. Linear + ARIMA Models

Figure 6 shows how well the combined Linear + ARIMA model predicts CO_2 emissions for the years 2021 to 2025. The black line represents the actual observed emissions, while the solid blue line shows the model's predicted values, and the dashed blue lines indicate the 95% prediction interval. The predicted values follow the seasonal ups and downs of the observed data quite closely, staying mostly within the prediction bounds. This suggests that the model is effective at capturing both the trend and seasonal patterns in the data.

4.2 Model 2

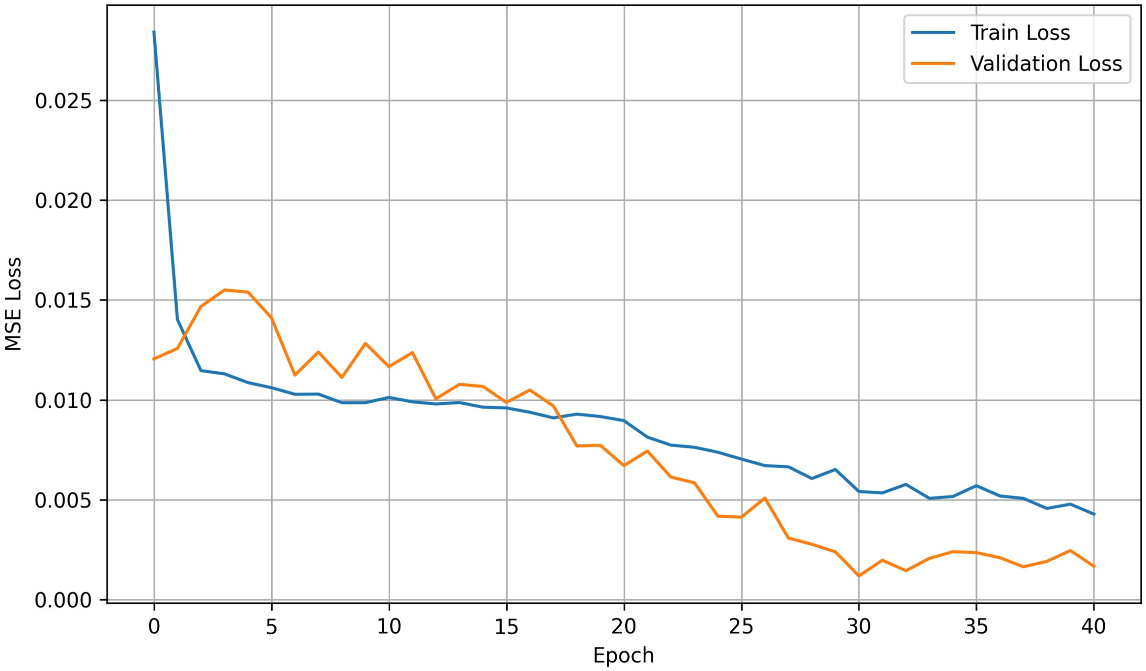


Figure 7: Loss Curve

Figure 7 shows how the model's error changes over time during training. Specifically, it plots the Mean Squared Error (MSE) for both the training and validation data across 40 epochs. At the beginning, both training and validation losses are relatively high, especially the training loss. As training progresses, both losses gradually decrease. This means the model is learning and improving its accuracy over time. Around the middle of the training, the validation loss starts to follow the same trend as the training loss and eventually becomes even lower. This is a good sign, showing that the model is not overfitting and is able to generalize well to unseen data. Overall, the loss curve shows that the model training was successful and well-regularized. The training process is automatically stopped after 40 epochs due to the early stopping mechanism. This happens when further improvement in validation loss becomes negligible.

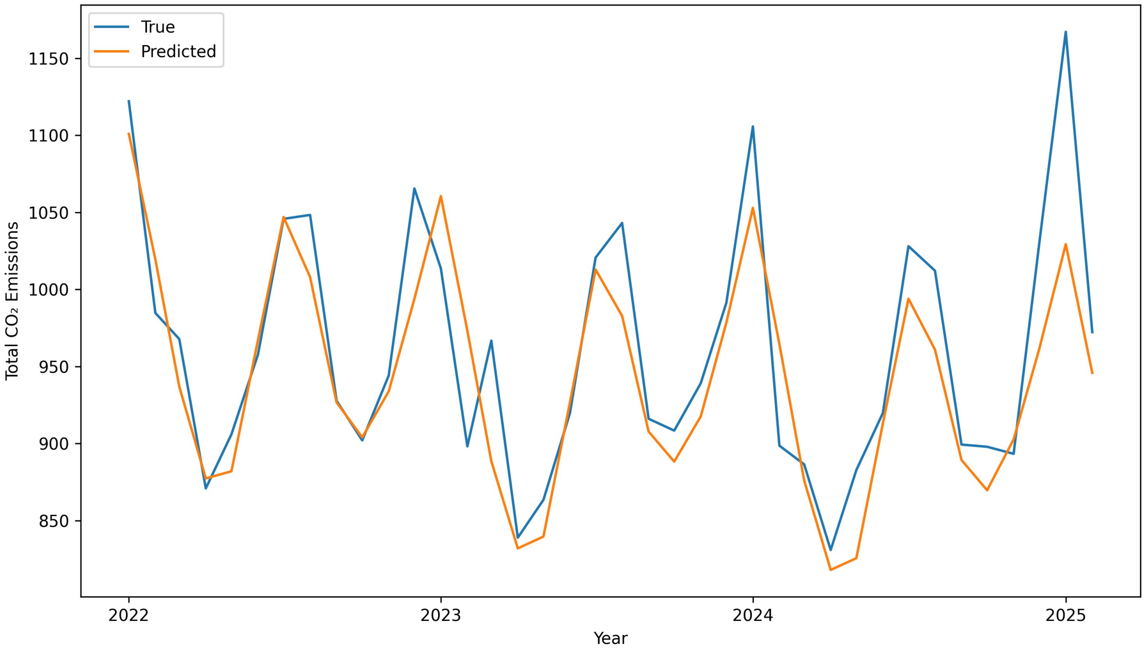


Figure 8: Test Forecast

Figure 8 compares the model's predicted CO_2 emissions (in orange) with the actual true emissions (in blue) for the years after 2022. The two lines follow very similar patterns, showing that the model has learned the seasonal trends in the data. Although the predicted values are not exactly the same as the true values, they are close and show the same ups and downs. This suggests the model is able to capture important patterns over time, like seasonal cycles. The small differences between the predicted and actual values are expected and acceptable. In general, the forecast shows that the LSTM model performs well in predicting future CO_2 emissions.

5 Discussion

5.1 Findings Conclusion

Our research used both linear regression and ARIMA models to analyze and forecast monthly U.S. CO_2 emissions based on dominant sectors of greenhouse gas emissions. Our results show that while a linear regression model does explain much of the variation in emissions, the ACF and PACF plots of the residuals from the model suggest there is serial correlation of the errors. The linear model clearly does not account for the temporal structure of the data. Therefore, incorporating the time series structure into our model using an ARIMA model significantly improves on this aspect. Residuals are more likely to be independent, and our forecasts are more accurate due to the model accounting for factors like autocorrelation and seasonality. We discuss these differences below.

5.2 Implications of Model

Our linear model identified 6 key predictors, 5 of which are statistically significant, that drive total CO_2 emissions. Of the statistically significant predictors, transportation petroleum consumption exhibits a strong positive relationship with emissions, which is consistent with the industry's carbon emission expectations. Furthermore, commercial and industrial energy consumption also have positive effects with significant p-values, with coefficients that align with their respective industry expectations. These results validate the inclusion of these sectors as relevant predictors for our model.

However, fossil fuel production and renewable energy production both show large and statistically significant negative relationships with total emissions. The coefficient for renewable energy is approximately -256, which supports the idea that renewable energy greatly reduces emissions. The negative coefficient for fossil fuel production, which is approximately -38, seems counterintuitive, as it is generally understood that fossil fuel production directly releases many CO_2 emissions into the atmosphere. Assuming a linear model, the only explanations could be that increased fossil fuel production results in increased exports of further CO_2 emitting products, or the actual process of producing fossil fuels uses renewable energy, both of which are unlikely. Residential energy consumption exhibits a small, positive, but statistically insignificant relationship with emissions. This suggests that residential consumption has limited marginal contribution in total emissions, or that it could offset seasonal dynamics.

5.3 Other Implications

Our results demonstrate the importance of incorporating temporal factors into environmental forecasting models, especially for time series data. Only relying on cross-sectional, linear regressors in our model fails to account for the strong autocorrelation and seasonality persistent in emissions data. Our results suggest that the hybrid approaches, where temporal factors are captured in the ARIMA model, are better suited for forecasting CO_2 emissions.

Previous literature has also analyzed the importance of combining different models and exploring nonlinear relationships on emissions, such as using the Stochastic Impacts by Regression on Population, Affluence, and Technology (STIRPAT) model (Fan et al. 2006). Here, environmental impact is taken to be a combination of a product of IPAT factors (population, affluence, and technology), while also incorporating a stochastic element that allows for a more flexible regression.

Furthermore, while not directly modelling emissions, energy demand has shown to be well-modelled using a mixture of ARIMA, SARIMA, GARCH, and ARMAX (ARMA with exogenous inputs) frameworks (Suganthi and Samuel 2012). Several of the identified regressors are strongly correlated with total energy demand, as basic economic frameworks suggest that the demand and supply for energy should be very similar. Most energy consumption, especially from fossil fuels, directly results in CO_2 emissions, so the factors that affect energy demand may also carry over to emissions. As these models seem to accurately capture real-world effects in their forecasts, it could prove to be useful to incorporate these ideas into future studies.

In practical terms, understanding the factors that go into CO_2 emissions and energy production is relevant to modern-day governmental policy. Accurate forecasts can help policymakers evaluate the expected environmental impact of legislation relating to urgent issues such as fuel efficiency, corporate emissions, and the 'carbon emissions market', which is gaining traction as governments worldwide are increasing their efforts to reduce climate change. Forecasts that correctly account for seasonal, environmental, and other policy shocks across time can directly lead to better carbon pricing models that are responsive to those shocks. Therefore, as emissions forecasting becomes increasingly important for meeting emissions and environmental targets, analyzing more flexible models which better account for temporal factors will be essential for effective policy design and the future of the environment.

5.4 Limitations & Improvements

Our current model still has a few limitations. Firstly, the model is based on historical linear patterns and cannot fully account for the shocks caused by governmental policy, nonlinear effects, or other exogenous shocks like the pandemic or potential wars. Next, while ARIMA models do capture serial dependence, it assumes stationarity after enough differencing and does not account for conditional heteroskedasticity, which a GARCH-style model would. Finally, our analysis combines emissions across all U.S. regions and sectors, which could potentially misrepresent other industry-specific patterns as a result.

As seen in the previous studies, our models could incorporate seasonal ARIMA (SARIMA) models to better capture seasonality, giving us a better understanding of the seasonal patterns, which could better address these problems in the future. Furthermore, generalized autoregressive conditional heteroskedasticity (GARCH) models to better capture conditional volatility in CO_2 emissions could help explain the effects of unpredictable shocks. Alternatively, the STIRPAT model, which is also commonly used, could be helpful for environmental forecasting. Finally, potentially exploring other machine learning techniques in addition to the ones above could further improve and optimize CO_2 emissions models.

Our integrated linear and ARIMA modeling approach gives us a strong foundation for forecasting U.S. CO_2 emissions. We have demonstrated the value of accounting for the temporal factors in environmental data. Our model gives forecasts without excessive model complexity and avoids overfitting, which are both risks of using the previously mentioned models. Our analysis gives us a great baseline for future work, potentially using more advanced models. It also highlights the need for continued improvements and progress in environmental forecasting to achieve accurate, data-driven climate policy.

6 Appendix

6.1 Project Code

6.1.1 R

```r
#### R Code Part ####

# load library
library(tidyverse)
library(lubridate)
library(gridExtra)
library(forecast)
library(ggplot2)
library(knitr)
library(readr)
library(dplyr)
library(astsa)
library(here)

# load data
data <- read.csv("../Data/cleaned_data.csv")


data <- data %>%
  mutate(
    Date = as.Date(paste(Year, Month, "1", sep = "-"), format = "%Y-%m-%d")
  )

ggplot(data, aes(x = Date, y = Total_CO2_Emissions)) +
  geom_line(size = 1) +
  labs(
    x     = "Date",
    y     = "Total CO2 Emissions (million metric tons)"
  ) +
  theme_minimal()


data_zoom <- data %>%
  filter(Date >= as.Date("2015-01-01") & Date <= as.Date("2022-12-31"))
```

```
ggplot(data_zoom, aes(x = Date, y = Total_CO2_Emissions)) +
  geom_line(size = 1) +
  labs(
    title = "Total CO2 Emissions (2015-2022)",
    x     = "Date",
    y     = "Total CO2 Emissions (million metric tons)"
  ) +
  theme_minimal() +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y")

# Convert to time series first
co2_ts <- ts(data$Total_CO2_Emissions)

# Generate ACF object without plotting
acf_obj <- acf(co2_ts, plot = FALSE)

# Manually plot it without a title
plot(acf_obj, main = "")

data_train <- data[data$Year <= 2020, ]
data_test  <- data[data$Year > 2020, ]

model1 <- lm(
  Total_CO2_Emissions ~ .
    - Date - Year - Month,
  data = data_train
)

model_summary <- summary(model1)

# Convert coefficients to a data frame
coef_table <- as.data.frame(coef(model_summary))

# Add row names as a column
coef_table$Variable <- rownames(coef_table)
rownames(coef_table) <- NULL

# Reorder
coef_table <- coef_table[, c("Variable", "Estimate", "Std. Error", "t value", "Pr(>|t|)")]

# Shorten variable names using a named vector
var_names <- c(
  "(Intercept)" = "(Intercept)",
  "Transportarion.Petroleum.Consumption" = "Transportation",
  "Total.Fossil.Fuels.Production" = "Fossil Fuel Prod.",
  "Total.Renewable.Energy.Production" = "Renewable Energy Prod.",
  "Commercial_Consumption" = "Commercial",
  "Industrial_Consumption" = "Industrial",
  "Residential_Consumption" = "Residential"
)

# Apply shortened names
coef_table$Variable <- var_names[coef_table$Variable]

# Add significance codes (optional, mimicking base R behavior)
coef_table$Significance <- cut(coef_table$`Pr(>|t|)`,
                               breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, Inf),
                               labels = c("***", "**", "*", ".", ""))

# Display the table
knitr::kable(coef_table, digits = 4)

plot(model1$fitted.values, resid(model1),
     xlab = "Fitted Values",
     ylab = "Residuals")
abline(h = 0, col = "red")

plot(data_train$Date, resid(model1),
     type = "l",
     xlab = "Date",
     ylab = "Residuals")
abline(h = 0, col = "red")

hist(resid(model1),
     breaks = 30,
     main = NULL,
     xlab = "Residuals")

# Residuals satisfied identical normal distribution
qqnorm(resid(model1), main = NULL)
qqline(resid(model1), col = "red")

acf(resid(model1), main = "")
```

```r
pacf(resid(model1), main = "")

# ljung-box test shows that the residuals are auto-correlated
ljung_result <- Box.test(resid(model1), lag = 20, type = "Ljung-Box")

# Create table from the output
box_test_table <- data.frame(
  `Test`        = "Box-Ljung",
  `Statistic`   = round(ljung_result$statistic, 2),
  `df`          = ljung_result$parameter,
  `p-value`     = format.pval(ljung_result$p.value, digits = 3, eps = .001)
)

# Display the table
knitr::kable(box_test_table)

# auto fitted arima
resid_ts <- ts(resid(model1))
resid_arima <- forecast::auto.arima(resid_ts)

# Extract summary metrics
arima_summary <- summary(resid_arima)

# Get training accuracy metrics separately
train_metrics <- accuracy(resid_arima)

# Build a combined table
arima_metrics <- data.frame(
  Metric = c("Sigma²", "Log Likelihood", "AIC", "AICc", "BIC",
             "ME", "RMSE", "MAE", "MPE", "MAPE", "MASE", "ACF1"),
  Value = c(
    arima_summary$sigma2,
    arima_summary$loglik,
    arima_summary$aic,
    arima_summary$aicc,
    arima_summary$bic,
    train_metrics[1, "ME"],
    train_metrics[1, "RMSE"],
    train_metrics[1, "MAE"],
    train_metrics[1, "MPE"],
    train_metrics[1, "MAPE"],
    train_metrics[1, "MASE"],
    train_metrics[1, "ACF1"]
  )
)

# Display the table
knitr::kable(arima_metrics, digits = 4)

# Fit ARIMA model
resid_ts <- ts(resid(model1))
resid_arima <- forecast::auto.arima(resid_ts)

# Extract coefficients and standard errors
coef_vals <- coef(resid_arima)
se_vals <- sqrt(diag(resid_arima$var.coef))

# Create table
arima_coef_table <- data.frame(
  Term = names(coef_vals),
  Estimate = round(coef_vals, 4),
  Std_Error = round(se_vals, 4)
)

# Display
knitr::kable(arima_coef_table)

y_hat_lm <- fitted(model1)

resid_hat_arima <- fitted(resid_arima)
y_hat_combined <- y_hat_lm + resid_hat_arima

plot(data_train$Date, data_train$Total_CO2_Emissions, type = "l", col = "black", lwd = 2,
     ylab = "Total CO2 Emissions (million metric tons)",
     xlab = "Date",
     cex.lab = 1.2, cex.main = 1.4)

lines(data_train$Date, y_hat_combined, col = "red", lwd = 2)

legend("topright",
       legend = c("Observed", "LM + ARIMA Fit"),
       col = c("black", "red"),
       lty = 1,
       lwd = 2,
       cex = 1.1,
       bty = "n")



y_true <- data_train$Total_CO2_Emissions
y_lm   <- fitted(model1)

plot(data_train$Date, y_true, type = "l", col = "black", lwd = 2,
     ylab = "Total CO2 Emissions (million metric tons)",
     xlab = "Date",
     cex.lab = 1.2, cex.main = 1.4)

lines(data_train$Date, y_lm, col = "red", lwd = 2)

legend("topright",
       legend = c("Observed", "Linear Model Fit"),
       col = c("black", "red"),
       lty = 1,
       lwd = 2,
       cex = 1.1,
       bty = "n")

# Compute RSS and RMSE if not already done
rss_lm <- sum((y_true - y_lm)^2)
rss_combined <- sum((y_true - y_hat_combined)^2)

rmse_lm <- sqrt(mean((y_true - y_lm)^2))
rmse_combined <- sqrt(mean((y_true - y_hat_combined)^2))

# Create a comparison table
model_comparison <- data.frame(
  Model = c("Linear Model", "Linear + ARIMA Model"),
  RSS   = c(rss_lm, rss_combined),
  RMSE  = c(rmse_lm, rmse_combined)
)

# Display the table
knitr::kable(model_comparison, digits = 4)

# Predict data adter 2020
pred_result <- predict(model1, newdata = data_test, interval = "prediction", level = 0.95)
```

```
y_test_true <- data_test$Total_CO2_Emissions
y_test_pred <- pred_result[, "fit"]
y_test_lower <- pred_result[, "lwr"]
y_test_upper <- pred_result[, "upr"]

plot(data_test$Date, y_test_true, type = "l", col = "black", lwd = 2,
     ylab = "Total CO2 Emissions (million metric tons)",
     xlab = "Date",
     cex.lab = 1.2, cex.main = 1.4)

lines(data_test$Date, y_test_pred, col = "blue", lwd = 2)
lines(data_test$Date, y_test_lower, col = "blue", lty = 2, lwd = 1.5)
lines(data_test$Date, y_test_upper, col = "blue", lty = 2, lwd = 1.5)

legend("topright",
       legend = c("Observed", "Predicted", "95% Prediction Interval"),
       col = c("black", "blue", "blue"),
       lty = c(1, 1, 2),
       lwd = 2,
       cex = 1.1,
       bty = "n")
```

6.1.2 Python

```
#### Python Code Part ####

# load library
import os
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
```

```
# load data
df = pd.read_csv("../Data/cleaned_data.csv")

df['Date'] = pd.to_datetime(df['Year'].astype(str) + '-' + df['Month'].astype(str) + '-01')
df = df.sort_values('Date')

features = [
    'Transportarion Petroleum Consumption',
    'Total Fossil Fuels Production',
    'Total Renewable Energy Production',
    'Commercial_Consumption',
    'Industrial_Consumption',
    'Residential_Consumption',
    'Month'
]
target = 'Total_CO2_Emissions'
data_all = df[features + [target, 'Date']].dropna()

# normalize the data
feature_scaler = MinMaxScaler()
target_scaler = MinMaxScaler()

data_scaled = data_all.copy()
data_scaled[features] = feature_scaler.fit_transform(data_all[features])
data_scaled[[target]] = target_scaler.fit_transform(data_all[[target]])

# Construct sliding window samples
def make_sequences(data, seq_len=12):
    X, y, dates = [], [], []
    for i in range(len(data) - seq_len):
        X.append(data.iloc[i:i+seq_len][features].values)
        y.append(data.iloc[i+seq_len][target])
        dates.append(data.iloc[i+seq_len]['Date'])
    return np.array(X), np.array(y), np.array(dates)

seq_len = 24
X_all, y_all, dates_all = make_sequences(data_scaled, seq_len)

# train validation test set split
train_end = pd.Timestamp("2020-12-31")
valid_end = pd.Timestamp("2021-12-31")
```

```
train_idx = dates_all <= train_end
valid_idx = (dates_all > train_end) & (dates_all <= valid_end)
test_idx = dates_all > valid_end

X_train, y_train = X_all[train_idx], y_all[train_idx]
X_valid, y_valid = X_all[valid_idx], y_all[valid_idx]
X_test, y_test = X_all[test_idx], y_all[test_idx]

def set_seed(seed=42):
    os.environ['PYTHONHASHSEED'] = str(seed)
    random.seed(seed)
    np.random.seed(seed)
    tf.random.set_seed(seed)
    tf.config.experimental.enable_op_determinism()

set_seed(42)

# Construct the model
model = Sequential([
    LSTM(128, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.3),
    LSTM(64),
    Dense(1)
])
model.compile(optimizer='adam', loss='mse')

early_stop = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

history = model.fit(
    X_train, y_train,
    validation_data=(X_valid, y_valid),
    epochs=100,
    batch_size=16,
    callbacks=[early_stop],
    verbose=0
)

# loss function
plt.figure(figsize=(8, 5))
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title("Loss Curve During Training")
plt.xlabel("Epoch")
plt.ylabel("MSE Loss")
plt.legend()
plt.grid(True)
plt.tight_layout()

os.makedirs("Charts", exist_ok=True)
plt.savefig("Charts/loss_curve.png", dpi=300)
plt.close()

def predict_and_inverse(X, y, scaler):
    y_pred = model.predict(X, verbose=0)
    y_pred_inv = scaler.inverse_transform(y_pred.reshape(-1, 1))
    y_true_inv = scaler.inverse_transform(y.reshape(-1, 1))
    return y_true_inv, y_pred_inv

y_train_inv, y_pred_train_inv = predict_and_inverse(X_train, y_train, target_scaler)
y_valid_inv, y_pred_valid_inv = predict_and_inverse(X_valid, y_valid, target_scaler)
y_test_inv,  y_pred_test_inv  = predict_and_inverse(X_test,  y_test,  target_scaler)

# MSE evaluation for both training validation and prediction set
def print_rmse(label, y_true, y_pred):
    mse = mean_squared_error(y_true, y_pred)
    rmse = np.sqrt(mse)
    print(f"{label} MSE: {mse:.2f}, RMSE: {rmse:.2f}")
    return rmse

train_rmse = print_rmse("Train", y_train_inv, y_pred_train_inv)
valid_rmse = print_rmse("Valid", y_valid_inv, y_pred_valid_inv)
test_rmse  = print_rmse("Test",  y_test_inv,  y_pred_test_inv)

# Prediction on Test dataset
plt.figure(figsize=(10, 6))
plt.plot(dates_all[test_idx], y_test_inv, label='True')
plt.plot(dates_all[test_idx], y_pred_test_inv, label='Predicted')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y'))
plt.gca().xaxis.set_major_locator(mdates.YearLocator())
plt.title('LSTM Forecast (2022 and beyond)')
plt.xlabel("Year")
plt.ylabel("Total CO  Emissions")
plt.legend()
plt.tight_layout()
```

```
os.makedirs("Charts", exist_ok=True)
plt.savefig("Charts/test_forecast.png", dpi=300)
plt.close()
```

References

Ajala, Adewole Adetoro, Oluwatosin Lawrence Adeoye, Olawale Moshood Salami, and Ayoola Yusuf Jimoh. 2025. "An Examination of Daily CO_2 Emissions Prediction Through a Comparative Analysis of Machine Learning, Deep Learning, and Statistical Models." *Environmental Science and Pollution Research* 32: 2510–35. https://doi.org/10.1007/s11356-024-35764-8.

Fan, Ying, Lan-Cui Liu, Gang Wu, and Yi-Ming Wei. 2006. "Analyzing Impact Factors of CO2 Emissions Using the STIRPAT Model." *Environmental Impact Assessment Review* 26 (4): 377–95. https://doi.org/https://doi.org/10.1016/j.eiar.2005.11.007.

Intergovernmental Panel on Climate Change (IPCC). 2022. "Climate Change 2022: Mitigation of Climate Change."

Keerthana, Krishnamurthy Baskar, Shih-Wei Wu, Mu-En Wu, and Thangavelu Kokulnathan. 2023. "The United States Energy Consumption and Carbon Dioxide Emissions: A Comprehensive Forecast Using a Regression Model." *Sustainability* 15 (10): 7932. https://doi.org/10.3390/su15107932.

Kumari, Surbhi, and Sunil Kumar Singh. 2023. "Machine Learning-Based Time Series Models for Effective CO_2 Emission Prediction in India." *Environmental Science and Pollution Research* 30: 116601–16. https://doi.org/10.1007/s11356-022-21723-8.

Ritchie, Hannah. 2019. "Who Has Contributed Most to Global CO2 Emissions?" *Our World in Data*.

Suganthi, L., and Anand A. Samuel. 2012. "Energy Models for Demand Forecasting—a Review." *Renewable and Sustainable Energy Reviews* 16 (2): 1223–40. https://doi.org/https://doi.org/10.1016/j.rser.2011.08.014.

Tian, Yaxin, Xiang Ren, Keke Li, and Xiangqian Li. 2025. "Carbon Dioxide Emission Forecast: A Review of Existing Models and Future Challenges." *Sustainability* 17 (4): 1471. https://doi.org/10.3390/su17041471.

U.S. Energy Information Administration. 2025. "Monthly Energy Review." U.S. Energy Information Administration; https://www.eia.gov/totalenergy/data/monthly/.

Wen, Tingxin, Yazhou Liu, Yun He Bai, and Haoyuan Liu. 2023. "Modeling and Forecasting CO_2 Emissions in China and Its Regions Using a Novel ARIMA-LSTM Model." *Heliyon* 9 (11): e21241. https://doi.org/10.1016/j.heliyon.2023.e21241.

Zhang, G. P. 2003. "Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model." *Neurocomputing* 50: 159–75. https://doi.org/10.1016/S0925-2312(01)00702-0.