project title*

project subtitle

a Jiacan Sun Kevin You Ziheng Zhong

June 20, 2025

Table of contents

```
#| include: false
#| warning: false
#| message: false

# load library
library(tidyverse)
library(lubridate)
library(gridExtra)
```

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

```r
library(forecast)
library(ggplot2)
library(knitr)
library(readr)
library(dplyr)
library(astsa)
library(here)

# load data


# load model
```

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 XXX, 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 2 TBD, Section 3 TBD, Section 4 TBD.

2 Method

2.1 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.

```r
#| label: fig-summary-1
#| echo: false
#| warning: false
#| message: false
#| layout-ncol: 2
#| fig-cap: "Data Overview"
#| fig-subcap: ["Total CO2 Emissions Over Time", "Total CO2 Emissions (2015-2020)"]

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")
```

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 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 CO2 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 CO2 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.

```
#| label: fig-summary-2
#| echo: false
#| warning: false
#| message: false
#| fig-cap: "ACF of Total CO2 Emissions"

# 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 = "")
```

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 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.

2.2 Model

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

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

2.2.1 Model 1

```
#| label: tbl-model-summary
#| tbl-cap: "Model 1 Summary"
#| echo: false
#| warning: false
#| message: false

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)
```

In our case, our model is

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

```
#| label: fig-model1-fittings
#| echo: false
#| warning: false
#| message: false
#| fig-cap: "Model 1 Diagnostic Plots"
#| fig-subcap: ["Residuals vs Fitted", "Residuals Over Time", "Histogram of Residuals", "Norr
#| layout-nrow: 2
#| layout-ncol: 2
```

```r
#| fig-height: 5
#| fig-width: 6

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")
```

In the scatter plot of residuals against the fitted values, 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. This 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 is roughly symmetric around zero and appears approximately bell-shaped, albeit with mildly heavier tails than a perfect normal distribution.

The Q-Q plot 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.

Second, 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

```
#| label: fig-acf-model1
#| fig-cap: "ACF of Residuals for Model 1"
#| echo: false
#| warning: false
#| message: false

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

?@fig-acf-model1 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.

```
#| label: fig-pacf-model1
#| fig-cap: "PACF of Residuals for Model 1"
#| echo: false
#| warning: false
#| message: false

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

?@fig-pacf-model1 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.

```
#| label: tbl-model1-box
#| echo: false
#| warning: false
#| message: false
#| tbl-cap: "Box-Ljung Test on Model Residuals"

# 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)
```

The Box-Ljung test (**?@tbl-model1-box**) 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.

```
#| label: tbl-model1-auto-arima
#| echo: false
#| warning: false
#| message: false
#| tbl-cap: "ARIMA Model Summary on OLS Residuals"

# 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)
```

The `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.

```
#| label: tbl-arima-coef
#| echo: false
#| warning: false
#| message: false
#| tbl-cap: "ARIMACoefficients and Standard Errors"

# 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)
```

```
#| label: fig-model1-combine-fit-actual
#| echo: false
#| warning: false
#| message: false
#| fig-cap: "Model Fittings"
#| fig-subcap: ["Combined Model Fit","Linear Model Fit vs Actual"]
#| layout-ncol: 2
#| fig-height: 5

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")
```

?@fig-model1-combine-fit-actual 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. **?@fig-model1-combine-fit-actual-1** shows the combined model (linear + ARIMA), which closely follows the seasonal and long-term trends, especially in the mid and later periods. **?@fig-model1-combine-fit-actual-2** displays the linear model alone, which captures the overall trend but misses the recurring seasonal spikes and dips.

```
#| label: tbl-model-comparison
#| echo: false
#| warning: false
#| message: false
#| tbl-cap: "Comparison of Linear Model vs. Linear + ARIMA Model"

# 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)
```

```
#| label: fig-rss-rmse-comparison
#| echo: false
#| warning: false
#| message: false
#| fig-cap: "RSS and RMSE Comparison of Linear vs. Linear + ARIMA Models"
#| fig-height: 4
#| fig-width: 6

# 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")
```

?@fig-rss-rmse-comparison 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.

2.2.2 Model 2

```python
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




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

```python
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()
```

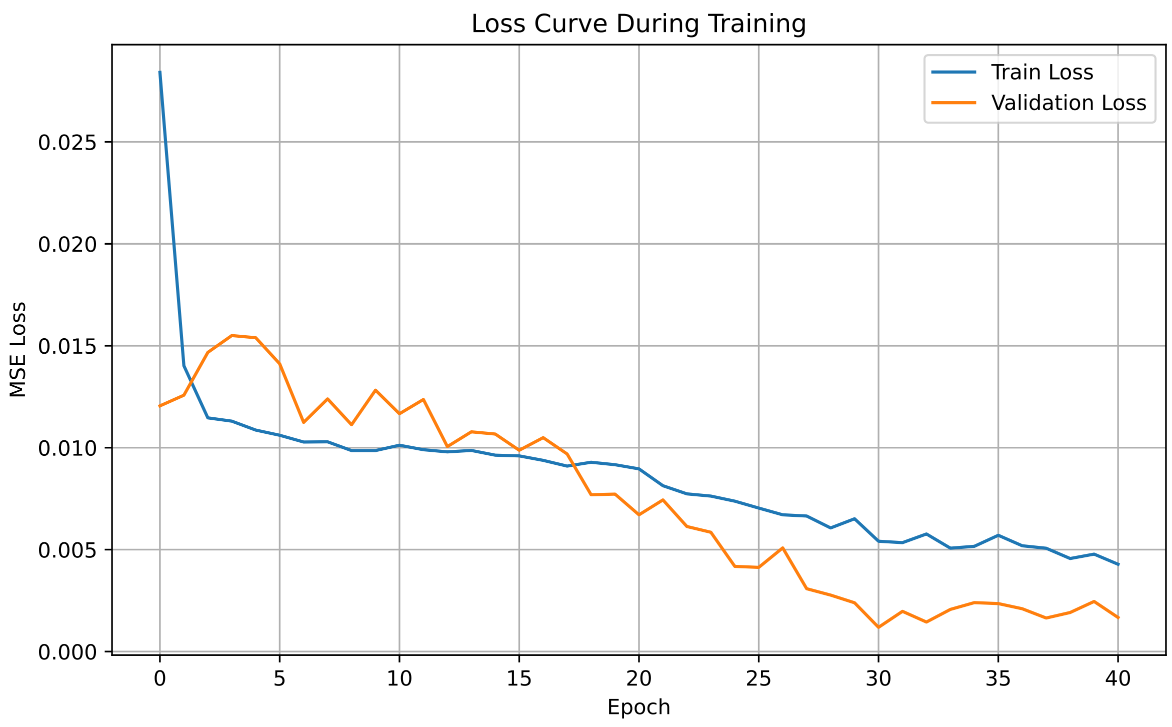
```python
# normalize the data
feature_scaler = MinMaxScaler()
target_scaler = MinMaxScaler()

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

```python
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]
```

```python
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)
])
```

```python
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
)
```

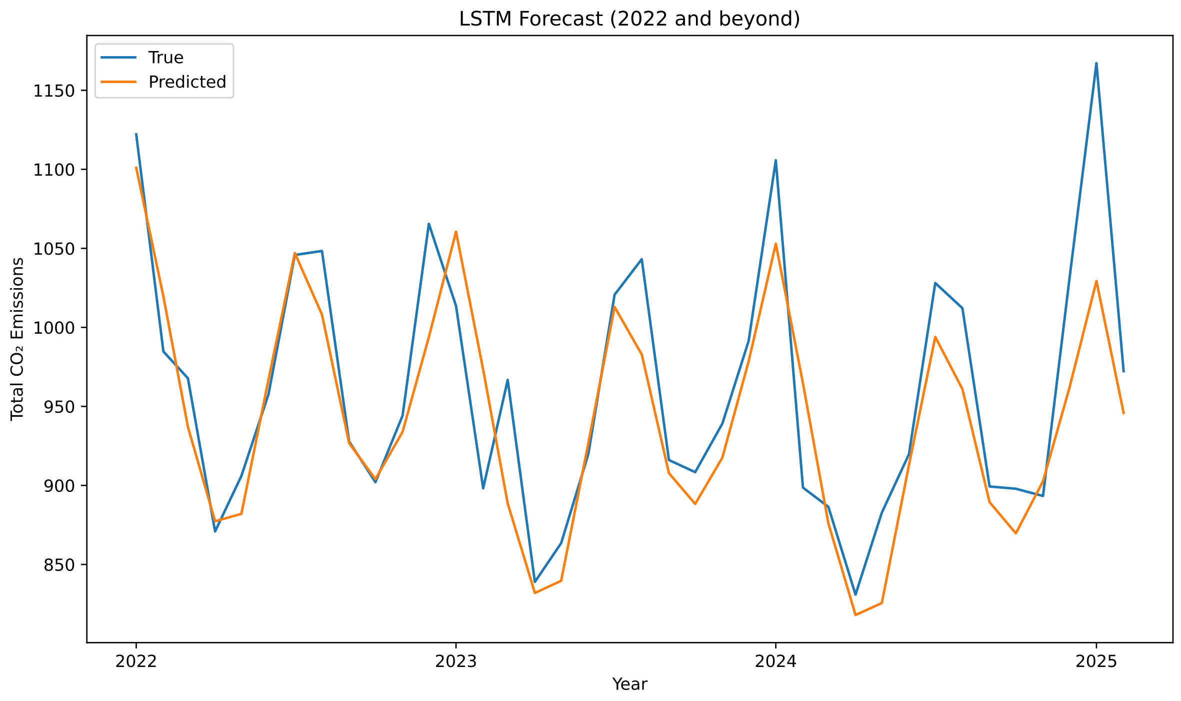
```python
# loss function
plt.figure(figsize=(8, 5))
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title("Loss Curve During Training")
plt.xlabel("Epoch")
plt.ylabel("MSE Loss")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()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



```
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)
```

```
Train MSE: 2329.53, RMSE: 48.27
Valid MSE: 469.45, RMSE: 21.67
Test MSE: 1828.54, RMSE: 42.76
```

```
# 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()
plt.show()
```



3 Result

4 Discussion

5 Appendix

5.1 Project Code

References

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.

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

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