Quantitative Trading Report

1. Abstract

This research report is primarily based on chapters six to thirteen of "Trading Systems and Methods." It utilizes trend-following strategies, momentum strategies, MACD strategies, Bollinger Bands strategies, and incorporates price-time relationships. It also incorporates trend changes, introduces LSTM long-term signals, Transformer short-term signals, and combines them with smoothing techniques and voting mechanisms. By effectively combining multiple trading strategies, it constructs complex trading indicators as entry and exit signals for the RB1012 rebar futures contract on the F.CN.SHF exchange. The data frequency used is daily, and the strategy primarily employs a fixed risk-per-trade percentage to manage capital.

2. Trading Strategy Approach
2.1 Indicator Meanings and Calculation Process:

1. **定义信号**：

$$\text{Signal}_{\text{SMA}}(t) = \begin{cases} 1 & \text{if } \text{SMA}_{\text{short}}(t) > \text{SMA}_{\text{long}}(t) \\ -1 & \text{if } \text{SMA}_{\text{short}}(t) < \text{SMA}_{\text{long}}(t) \\ 0 & \text{otherwise} \end{cases}$$

其中：

$$\text{SMA}_{\text{short}}(t) = \frac{1}{N_{\text{short}}} \sum_{i=0}^{N_{\text{short}}-1} P(t-i)$$

$$\text{SMA}_{\text{long}}(t) = \frac{1}{N_{\text{long}}} \sum_{i=0}^{N_{\text{long}}-1} P(t-i)$$

$$\text{Signal}_{\text{RSI}}(t) = \begin{cases} 1 & \text{if RSI}(t) < 30 \\ -1 & \text{if RSI}(t) > 70 \\ 0 & \text{otherwise} \end{cases}$$

其中：

$$\text{RSI}(t) = 100 - \frac{100}{1 + \frac{\text{EMA}_{\text{gain}}(t)}{\text{EMA}_{\text{loss}}(t)}}$$

$\text{EMA}_{\text{gain}}(t)$ 和 $\text{EMA}_{\text{loss}}(t)$ 是价格上涨和下跌的指数移动平均值。

$$\text{Signal}_{\text{MACD}}(t) = \begin{cases} 1 & \text{if MACD}(t) > \text{Signal Line}(t) \\ -1 & \text{if MACD}(t) < \text{Signal Line}(t) \\ 0 & \text{otherwise} \end{cases}$$

其中：

$$\text{MACD}(t) = \text{EMA}_{\text{short}}(t) - \text{EMA}_{\text{long}}(t)$$

$$\text{Signal Line}(t) = \text{EMA}_{\text{signal}}(\text{MACD}(t))$$

其中：

$$\text{Upper Band}(t) = \text{SMA}(t) + k \cdot \sigma(t)$$

$$\text{Lower Band}(t) = \text{SMA}(t) - k \cdot \sigma(t)$$

$\sigma(t)$ 是价格的标准差，k 是常数。

$$Signal_{combined}(t) = \frac{1}{4}(Signal_{SMA}(t) + Signal_{RSI}(t) + Signal_{MACD}(t) + Signal_{Bollinger}(t)) + S_{LSTM} + S_{Transformer}$$

平滑信号：

$$\text{Smoothed Signal}_{\text{combined}}(t) = \frac{1}{N} \sum_{i=0}^{N-1} \text{Signal}_{\text{combined}}(t-i)$$

最终交易信号：

$$\text{Final Signal}(t) = \begin{cases} 1 & \text{if Smoothed Signal}_{\text{combined}}(t) > 0 \\ -1 & \text{if Smoothed Signal}_{\text{combined}}(t) < 0 \\ 0 & \text{if Smoothed Signal}_{\text{combined}}(t) = 0 \end{cases}$$

这个公式表示了结合多个策略信号并经过平滑处理后的最终交易信号，用以决策买入（1）、卖出（-1）或保持（0）头寸。

2.2 Main Strategy Concept

This code implements a technical indicator-based trading strategy. The main concept is to generate trading signals using multiple indicators such as Simple Moving Average (SMA), Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Bollinger Bands, and introduce LSTM long-term signals and Transformer short-term signals. These signals are then integrated using a voting mechanism to form a composite signal. The strategy smooths the signals to reduce noise and determines the positions (buy, sell, or hold) based on the generated composite signal. It also calculates the strategy's returns and compares them with the market returns.

3. Strategy Performance

3.1 Indicator Construction

Basic Indicator Construction:

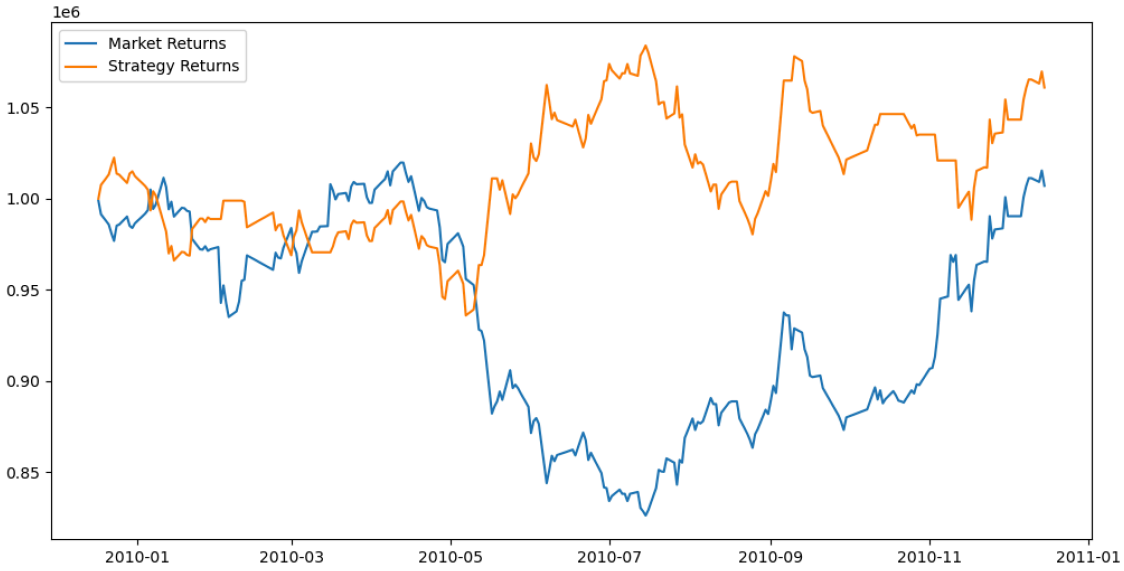
```
def SMA(data, window):
        return data['CLOSE'].rolling(window=window).mean()
data['SMA_short'] = SMA(data, 20)
data['SMA_long'] = SMA(data, 50)


def RSI(data, period=14):
        delta = data['CLOSE'].diff() gain = (delta.where(delta > 0,
0)).rolling(window=period).mean()
        loss = (-delta.where(delta < 0, 0)).rolling(window=period).mean()
        RS = gain / loss
```

```
        return 100 - (100 / (1 + RS))


def MACD(data, short_period=12, long_period=26, signal_period=9):
        short_ema = data['CLOSE'].ewm(span=short_period, adjust=False).mean()
        long_ema = data['CLOSE'].ewm(span=long_period, adjust=False).mean()
        macd = short_ema - long_ema
        signal = macd.ewm(span=signal_period, adjust=False).mean()
        return macd, signal


def BollingerBands(data, period=20, std_dev=2):
        sma = data['CLOSE'].rolling(window=period).mean()
        std =    data['CLOSE'].rolling(window=period).std()
        upper_band = sma + (std_dev * std)
        lower_band = sma - (std_dev * std)
        return upper_band, lower_band



data['Average_Signal'] = (data['Signal_SMA'] + data['Signal_RSI'] + data['Signal_MACD'] +
data['Signal_Bollinger']) / 4



class LSTMModel(nn.Module):
        def __init__(self, input_dim, hidden_dim, output_dim):
                super(LSTMModel, self).__init__()
                self.lstm = nn.LSTM(input_dim, hidden_dim, batch_first=True)
                self.fc = nn.Linear(hidden_dim, output_dim)
                self.sigmoid = nn.Sigmoid()

        def forward(self, x):
                _, (hn, _) = self.lstm(x)
                out = self.fc(hn[-1])
                out = self.sigmoid(out)
                return out



class TransformerModel(nn.Module):
        def __init__(self, input_dim, d_model, nhead, num_layers, output_dim,
dim_feedforward=2048, dropout=0.1):
                super(TransformerModel, self).__init__()
                self.transformer = nn.Transformer(d_model, nhead, num_layers, num_layers,
dim_feedforward, dropout)
                self.fc = nn.Linear(d_model, output_dim)
                self.sigmoid = nn.Sigmoid()
```

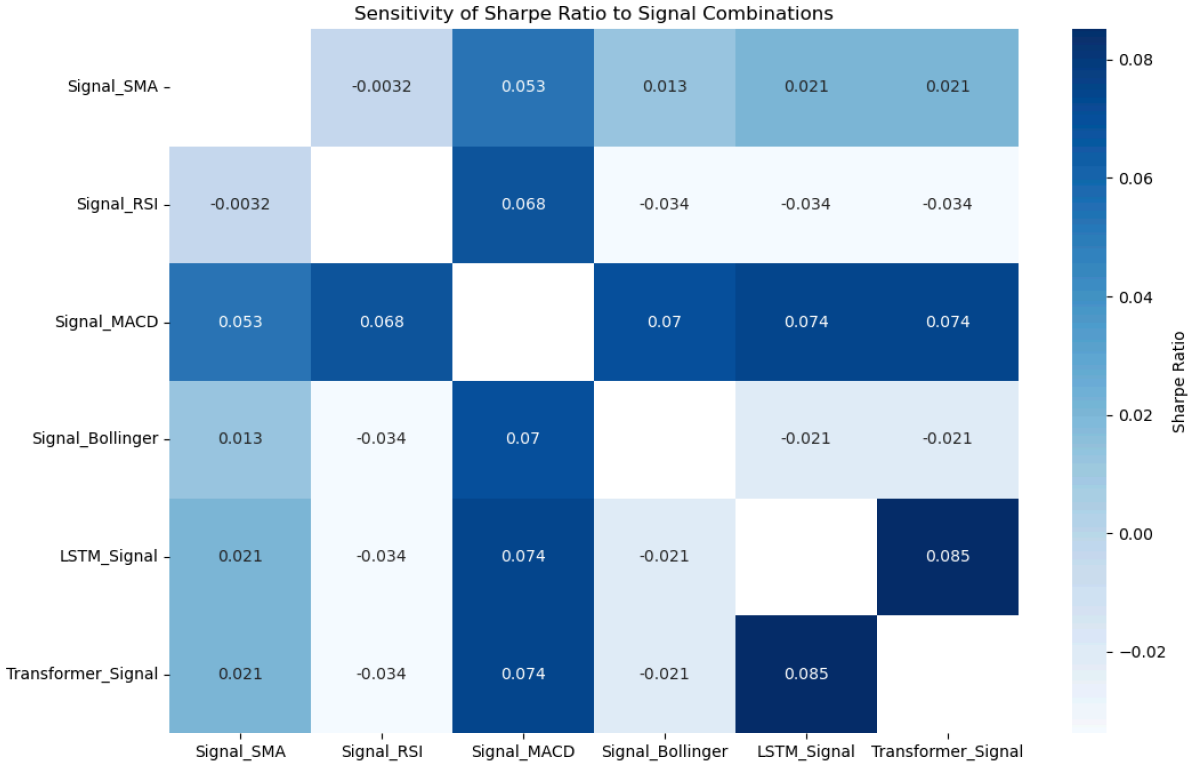
```
            self.embedding = nn.Linear(input_dim, d_model)

        def forward(self, x):
            x = self.embedding(x)
            x = x.permute(1, 0, 2)
            output = self.transformer(x, x)
            output = output.mean(dim=0)
            output = self.fc(output)
            output = self.sigmoid(output)
            return output


def predict(model, data_loader):
    model.eval()
    predictions = []
    with torch.no_grad():
        for X_batch, _ in data_loader:
            outputs = model(X_batch)
            predictions.append(outputs.numpy())
    return np.concatenate(predictions).flatten()


scaler = MinMaxScaler()
lstm_predictions_scaled = scaler.fit_transform(lstm_predictions.reshape(-1, 1)).flatten()
transformer_predictions_scaled = scaler.fit_transform(transformer_predictions.reshape(-1,
1)).flatten()
data['LSTM_Signal'] = 0
data.iloc[train_size+look_back:, data.columns.get_loc('LSTM_Signal')] =
lstm_predictions_scaled
data['Transformer_Signal'] = 0 data.iloc[train_size+look_back:,
data.columns.get_loc('Transformer_Signal')] = transformer_predictions_scaled


def updated_voting_signal(row):
    buy_votes = 0
    sell_votes = 0
    if row['Signal_SMA'] > 0:
        buy_votes += 1
    elif row['Signal_SMA'] < 0:
        sell_votes += 1

    if row['Signal_RSI'] > 0:
        buy_votes += 1
    elif row['Signal_RSI'] < 0:
        sell_votes += 1

    if row['Signal_MACD'] > 0:
        buy_votes += 1
    elif row['Signal_MACD'] < 0:
```

```
            sell_votes += 1

    if row['Signal_Bollinger'] > 0:
            buy_votes += 1
    elif row['Signal_Bollinger'] < 0:
            sell_votes += 1

    if row['LSTM_Signal'] > 0:
            buy_votes += 1
    elif row['LSTM_Signal'] < 0:
            sell_votes += 1

    if row['Transformer_Signal'] > 0:
            buy_votes += 1
    elif row['Transformer_Signal'] < 0:
    sell_votes += 1

    if buy_votes > sell_votes:
            return 1
    elif sell_votes > buy_votes:
            return -1
    else:
            return 0
```

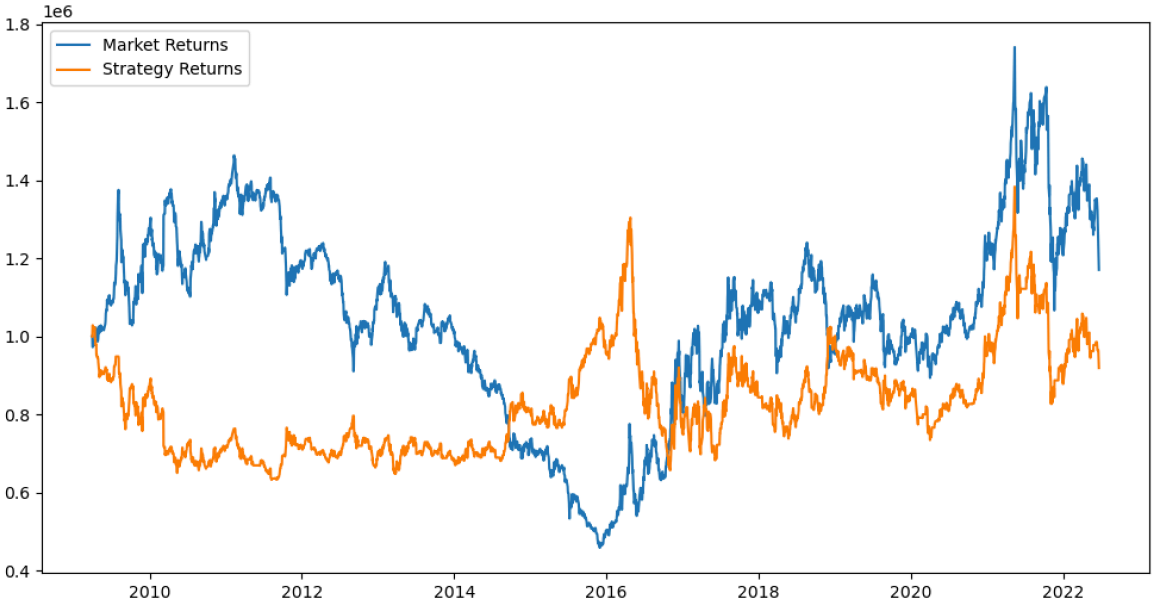
Indicator Visualization:

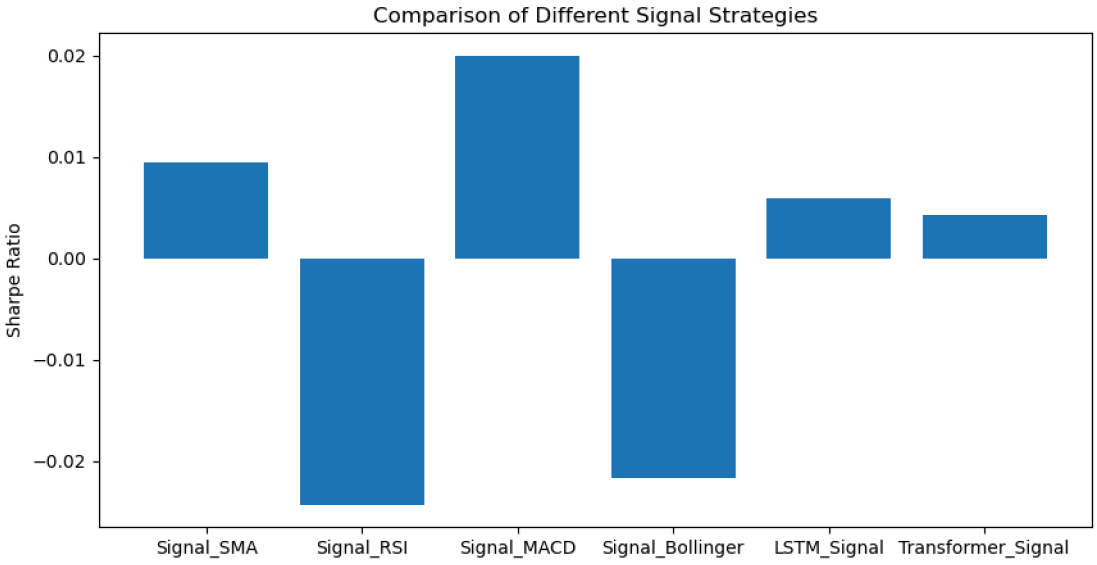


3.2 Parameter Sensitivity

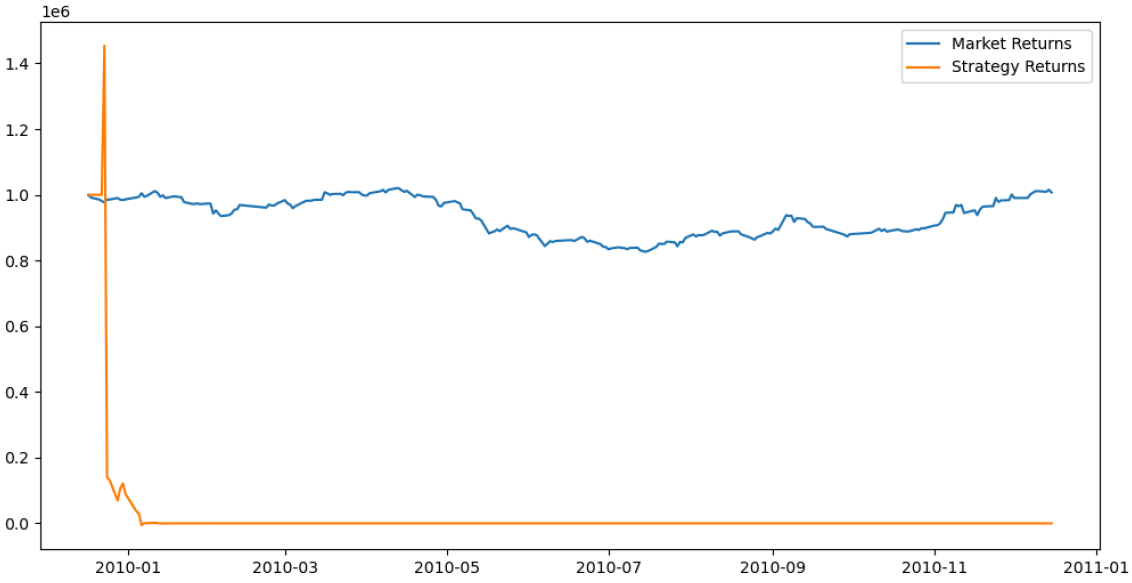


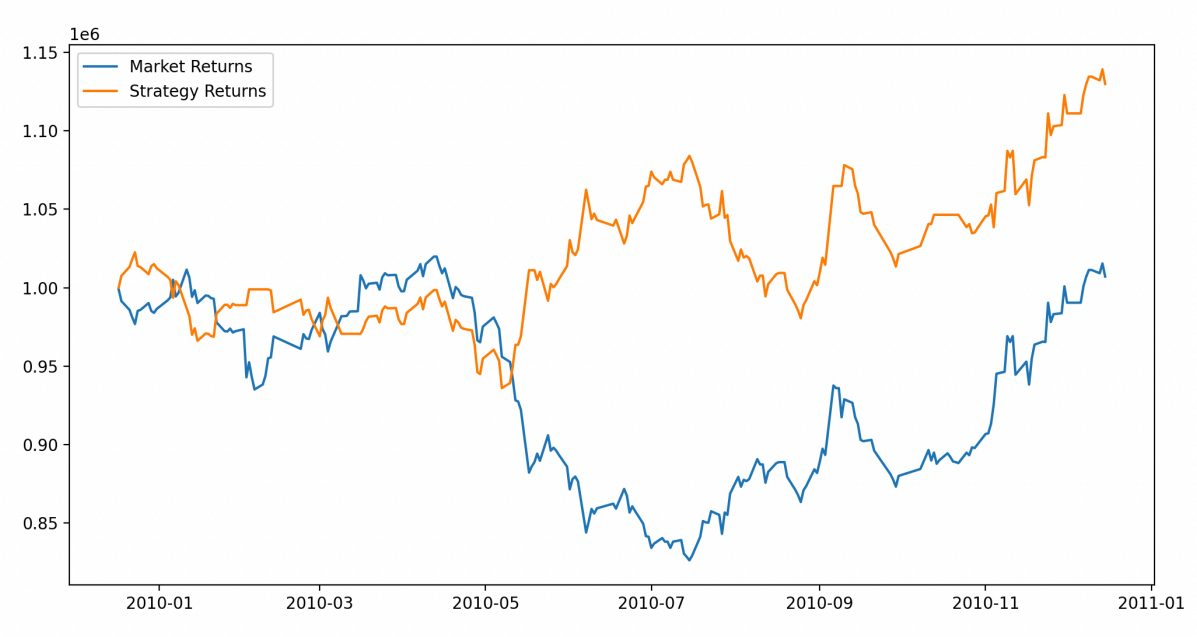
3.3 Multi-Asset Testing

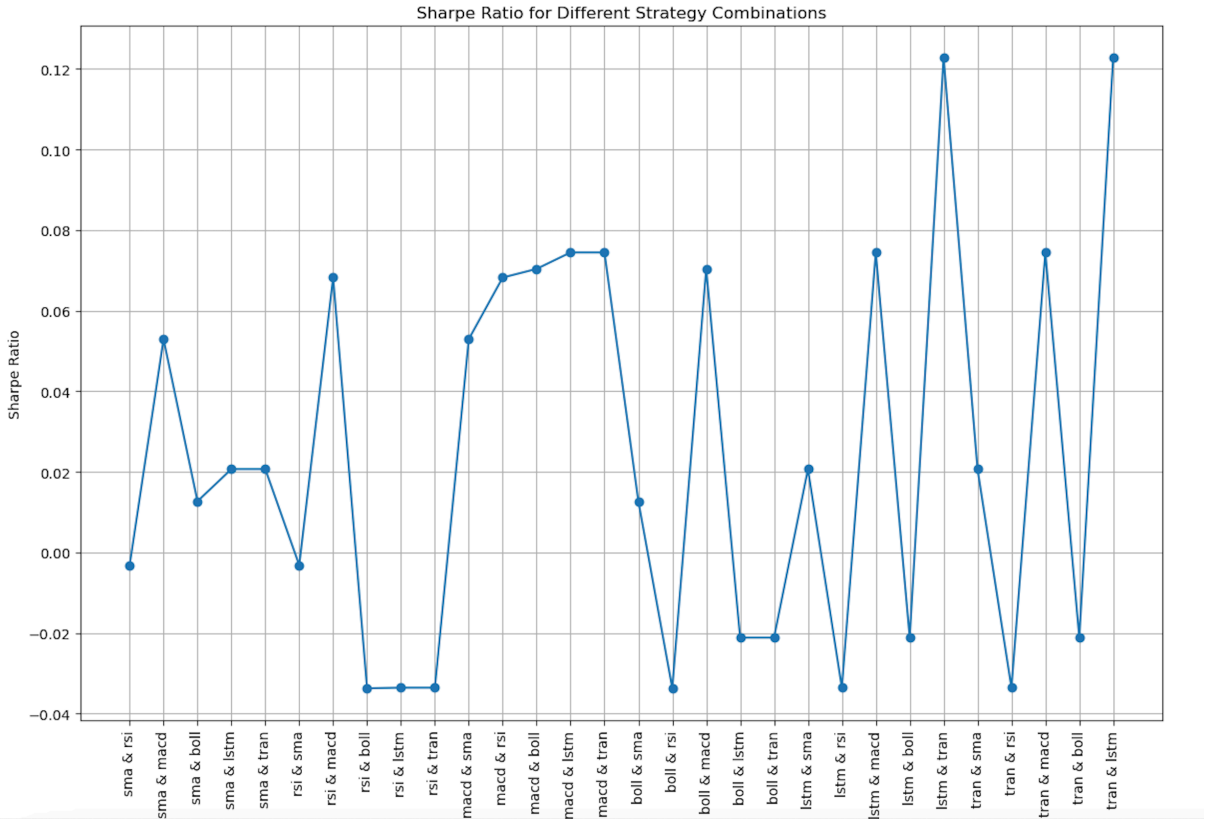




By combining signals from five futures contracts in the black metal sector (i.e., rebar, hot-rolled plate, coking coal, coke, and iron ore) using equal weighting, LSTM and Transformer models were trained. The models were trained for 1000 epochs and validated and compared on a test set. The purpose of this approach is to find the optimal parameter combination by integrating various technical indicator signals and using these combinations to validate the futures data.




1e6
Market Returns
Strategy Returns
1.15
1.10
1.05
1.00
0.95
0.90
0.85
2010-01
2010-03
2010-05
2010-07
2010-09
2010-11
2011-01


Sharpe Ratio for Different Strategy Combinations
Sharpe Ratio
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
sma & rsi
sma & macd
sma & boll
sma & lstm
sma & tran
rsi & sma
rsi & macd
rsi & boll
rsi & lstm
rsi & tran
macd & sma
macd & rsi
macd & boll
macd & lstm
macd & tran
boll & sma
boll & rsi
boll & macd
boll & lstm
boll & tran
lstm & sma
lstm & rsi
lstm & macd
lstm & boll
lstm & tran
tran & sma
tran & rsi
tran & macd
tran & boll
tran & lstm

4. Conclusion

From the above chart, it can be observed that the strategy performs poorly when only using SMA, RSI, MACD, and Signal_Bollinger without incorporating LSTM and Transformer signals. However, after incorporating deep learning signals, both for a single futures contract like rebar and for multiple futures contracts, the strategy shows significantly improved performance, consistently outperforming the market returns. Additionally, the line chart depicting the strategy signal combinations and Sharpe ratio demonstrates that the combination of LSTM and Transformer signals shows the highest improvement in Sharpe ratio. This further validates the effectiveness of deep learning financial signals.