Using CHRONOBERT Time Series Forecasting to Utilize Pairs Trading Strategies

This project investigates whether CHRONOBERT can enhance financial time series forecasting for pairs trading by leveraging its chronological training structure.

Caleb Johnson
Casey Hackett
Edgard Cuadra
Shanshan Gong

What is Pairs Trading?

Pairs trading involves identifying two correlated assets and the spread between their prices. When the spread diverges significantly from its historical average, traders buy the underperformer and sell the outperformer.

Key Concepts:

- Mean Reversion: The spread between two assets is expected to return to its historical average.
- Statistical Arbitrage: Trades are based on statistical patterns in prices, not on company earnings, management, or industry outlook.
- Market Neutrality: Long and short positions are balanced to reduce exposure to overall market movements.

Why CHRONOBERT?

What is BERT?

BERT is a model that helps machines understand context by analyzing the surrounding words in both directions, enabling a deeper grasp of meaning in language.

What Makes CHRONOBERT Different?

CHRONOBERT is an advanced adaptation of BERT that is specifically designed for time-sensitive applications. It maintains awareness of the chronological order and timing of events, which is critical for accurate predictions in fields like financial forecasting.

The Methods

BERT (Bidirectional Encoder Representations from Transformers):

- BERT is a language model made by Google that reads text and understands it, kind of like humans do.
- Reads words in both directions – from left to right and right to left – to understand context better.

ChronoBERT:

- Special version of BERT made for time based data, like medical records or documents that change over time.
- Difference from BERT:
    - It adds a timestamp to each piece of text so it knows when things happen.
    - It uses that time information to understand how language or meaning changes over time.

Traditional:

- Uses statistical models like OLS, linear regression, or autoregressive models (AR, ARIMA) to analyze numerical or time series data.
- These models often assume stable relationships and patterns, and may not handle language or unstructured text data well.

Hypotheses

1. CHRONOBERT’s out-of-sample performance will align more closely with its cross-validated results than models without chronological training, indicating lower lookahead bias.

2. Trading strategies based on CHRONOBERT will deliver more stable risk-adjusted returns across unseen market regimes than those using BERT or traditional methods.

Methodology: Overview

1. Find interesting ticker pairs

2. Collect adjusted close and returns for training period (2016-2018) and test period (2019)

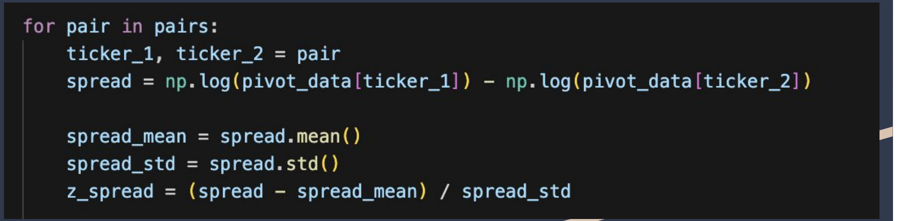
3. Compute log-normal Z-score spread

4. Use CHRONOBERT, BERT, and a traditional method to train on the data from 2016-2018 and predict the spread for 2019.

5. Create portfolios based on predicted spreads

6. Evaluate and compare out-of-sample performance for 2019 using MSE, R^2, and cumulative returns.

Methodology: Data Collection



```python
for pair in pairs:
    ticker_1, ticker_2 = pair
    spread = np.log(pivot_data[ticker_1]) - np.log(pivot_data[ticker_2])

    spread_mean = spread.mean()
    spread_std = spread.std()
    z_spread = (spread - spread_mean) / spread_std
```

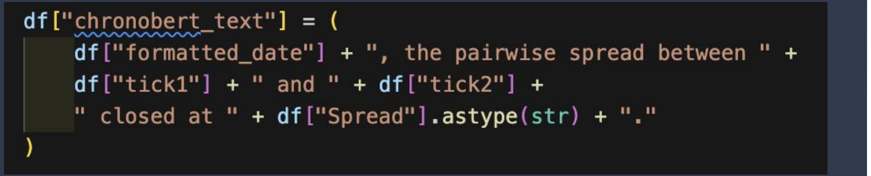
Basic Dataset:

- Spreads_weekly_large.csv
  - Collect Adj Close data from finance
  - Calculate Spreads
  - Calculate Ticker Pair returns
- Spreads_testing.csv
  - Repeat steps above for out-of-sample period

Produce Dataset:

- CHRONOBERT_spreads_weekly.csv
- Traditional_Spreads_weekly_Return.csv
- bert_spread.csv

Methodology: CHRONOBERT



```
df["chronobert_text"] = (
    df["formatted_date"] + ", the pairwise spread between " +
    df["tick1"] + " and " + df["tick2"] +
    " closed at " + df["Spread"].astype(str) + "."
)
```

1. Input Construction

- Spread turned to interpretable text format
- e.g. “On April 21st, 2020, the spread between MSFT and TSLA was .5643”

2. Embedding Generation

- Chronobert turns this text into a 768 row vector

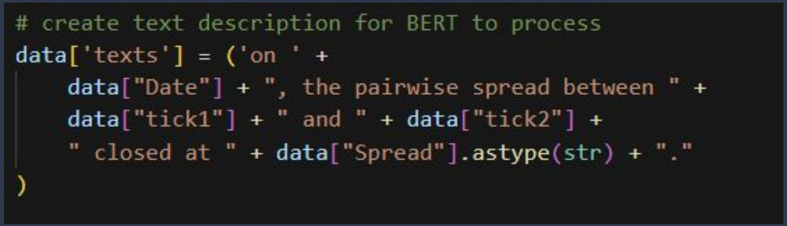
3. Pair Identity Encoding

- Ticker pairs are one-hot encoded
- Vector becomes 768+N dimensional

4. Ridge Regression Model
- Trained on the vectors from Chronobert and hot-encoding
- Predicts future spread values given embeddings and pair encoding

5. Forecasting
- Future inputs are formatted as:
- `"Based on recent values (average:{recent_avg:.4f}, Spread on December 30, 2018: {prev spread:.4f}), "`
- `f"the spread between {t1} and {t2} is projected to be 0.0000 on {date_str}."`

Methodology: BERT

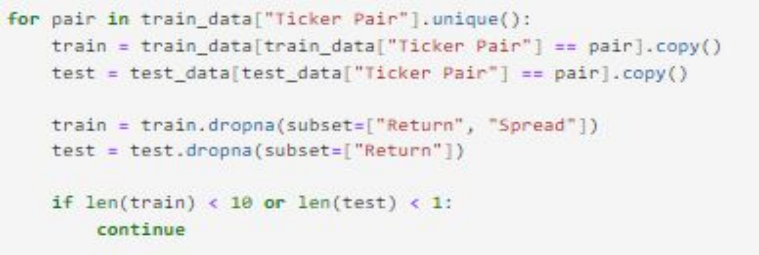


```
# create text description for BERT to process
data['texts'] = ('on ' +
    data["Date"] + ", the pairwise spread between " +
    data["tick1"] + " and " + data["tick2"] +
    " closed at " + data["Spread"].astype(str) + "."
)
```

Timestamp embedding

- BERT model flattens time into the text:
    - “On 2023-04-20, the pairwise spread between AAPL and MSFT closed at 0.52."
- Each row is treated as independent for sequential modeling.
- Transformer attention over time: Standard BERT only pays attention to the words within a single sentence.
- Encoding Progression: BERT doesn’t know how to track how the language evolves – it just learns patterns between text context and the spread.

Methodology: Traditional - OLS



```python
for pair in train_data["Ticker Pair"].unique():
    train = train_data[train_data["Ticker Pair"] == pair].copy()
    test = test_data[test_data["Ticker Pair"] == pair].copy()

    train = train.dropna(subset=["Return", "Spread"])
    test = test.dropna(subset=["Return"])

    if len(train) < 10 or len(test) < 1:
        continue
```

1. Input Construction
   - Uses raw numerical values instead of text
   - Input feature: Market Return (Return□)
   - Target variable: Spread□
2. Feature Preparation
   - No embeddings or text encoding
   - Ticker pairs are one-hot encoded
   - No temporal structure modeled beyond what return provides
3. Regression Model
   - Fits a simple Ridge Regression for each stock pair: Spread(t)=β0+β1 · Return(t)+εtTrained on 2016–2018 data
   - Predicts spread values for 2019
4. Forecasting
   - Predicts 2019 weekly spreads using current-period returns
   - Output column: Traditional_Spread
   - Results saved to: Traditional Spreads weekly Return.csv

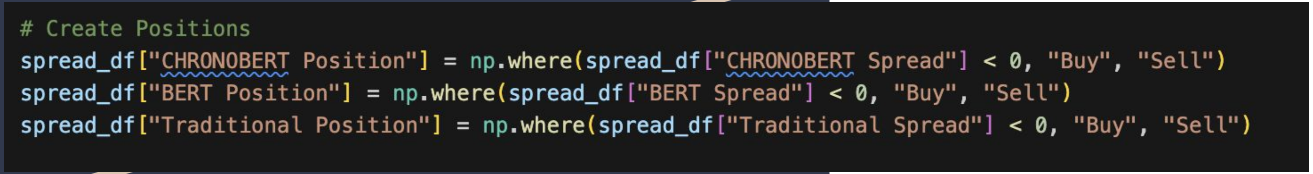
Methodology: Analysis

Analyzing Hypothesis 1:

- Calculate R^2 values
- Calculate MSE values
- Compare

Analyzing Hypothesis 2:

- Create Long/Short signals based on predicted spreads
- Create portfolios based on these signals
- Calculate portfolio returns
- Compare



```
# Create Positions
spread_df["CHRONOBERT Position"] = np.where(spread_df["CHRONOBERT Spread"] < 0, "Buy", "Sell")
spread_df["BERT Position"] = np.where(spread_df["BERT Spread"] < 0, "Buy", "Sell")
spread_df["Traditional Position"] = np.where(spread_df["Traditional Spread"] < 0, "Buy", "Sell")
```

Results

Check out the dashboard for an interactive view of the results as well as our final conclusions.

CHRONOPAIRS Dashboard