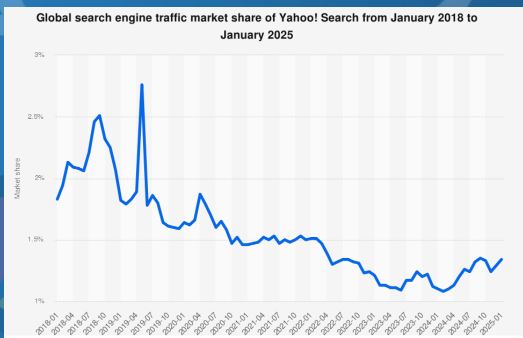
University of Stuttgart
Institute of Industrial Automation
and Software Engineering

Enhancing LLM Analysis for Numerical Data with Algorithmic Feature Extraction

Presenter: Wenjie Xie

Supervisor: Yuchen Xia M. Sc.

Examiner: Prof. Dr. Ing. Michael Weyrich



- Numerical Data Visualization

- Analysis:
  The chart illustrates a **steady downward of market share of Yahoo! Search**, with a sharp decline around 2019–2020, followed by a slow recovery after 2023.

Parameter	Equation
Mean	$T1 = \frac{\sum_{n=1}^{N} s(t)}{N}$
Standard deviation	$T2 = \sqrt{\frac{\sum_{n=1}^{N}(s(t) - T1)^2}{N-1}}$
Variance	$T3 = \left(\frac{\sum_{n=1}^{N}\sqrt{\lvert s(t)\rvert}}{N}\right)^2$
RMS	$T4 = \sqrt{\frac{\sum_{n=1}^{N}(s(t))^2}{N}}$
Absolute maximum	$T5 = \max \lvert s(t)\rvert$
Coefficient of skewness	$T6 = \frac{\sum_{n=1}^{N}(s(t) - T1)^3}{(N-1)(T2)^3}$
Kurtosis	$T7 = \frac{\sum_{n=1}^{N}(s(t) - T1)^4}{(N-1)(T2)^4}$

- Algorithmic Features

0 Agenda

- Motivation
- Basics
- Conceptual Design
- Evaluation and Analysis
- Summary and Outlook

1 Motivation

- Failure Case
- Challenge and Idea

Motivation

Failure Case

- Two Forms of Data Representation

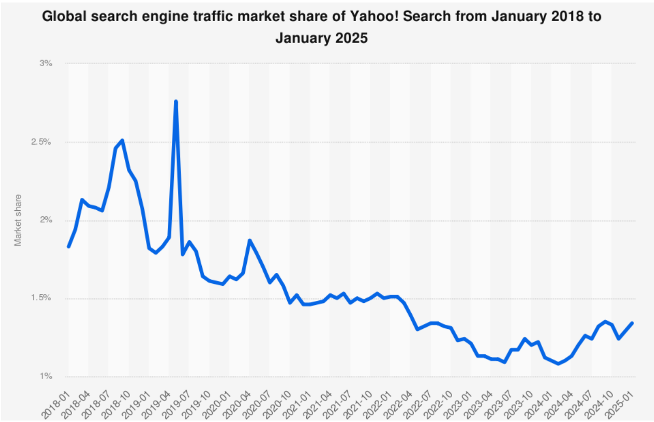
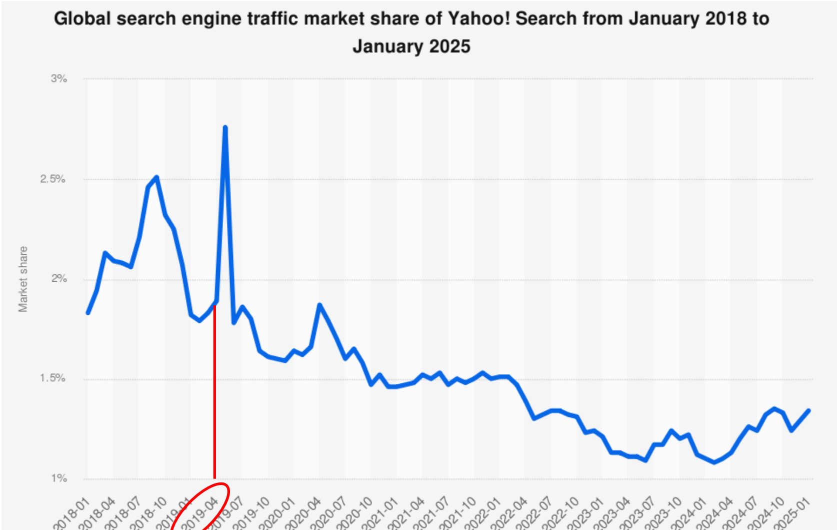


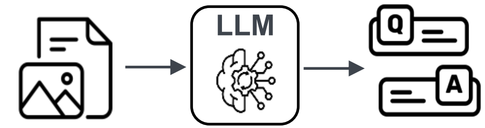
Chart Image

Jan-18	1.83	Aug-19	1.8	Mar-21	1.48
Feb-18	1.94	Sep-19	1.64	Apr-21	1.52
Mar-18	2.13	Oct-19	1.61	May-21	1.5
Apr-18	2.09	Nov-19	1.6	Jun-21	1.53
May-18	2.08	Dec-19	1.59	Jul-21	1.47
Jun-18	2.06	Jan-20	1.64	Aug-21	1.5
Jul-18	2.21	Feb-20	1.62	Sep-21	1.48
Aug-18	2.46	Mar-20	1.66	Oct-21	1.5
Sep-18	2.51	Apr-20	1.87	Nov-21	1.53
Oct-18	2.32	May-20	1.79	Dec-21	1.5
Nov-18	2.25	Jun-20	1.7	Jan-22	1.51
Dec-18	2.07	Jul-20	1.6	Feb-22	1.51
Jan-19	1.82	Aug-20	1.65	Mar-22	1.47
Feb-19	1.79	Sep-20	1.58	Apr-22	1.39
Mar-19	1.83	Oct-20	1.47	May-22	1.3
Apr-19	1.89	Nov-20	1.52	Jun-22	1.32
May-19	2.76	Dec-20	1.46	Jul-22	1.34
Jun-19	1.78	Jan-21	1.46	Aug-22	1.34
Jul-19	1.86	Feb-21	1.47	Sep-22	1.32

CSV

Data Format	Strengths	Limitations
Image	• Intuitive insight • Rich visual context	• No exact values • Depends on image quality
CSV	• Precise numerical representation • Statistical reliability	• Less semantic • Lacks visual cues





Jan-18	1.83	Aug-19	1.8	Mar-21	1.48
Feb-18	1.94	Sep-19	1.64	Apr-21	1.52
Mar-18	2.13	Oct-19	1.61	May-21	1.5
Apr-18	2.09	Nov-19	1.6	Jun-21	1.53
May-18	2.08	Dec-19	1.59	Jul-21	1.47
Jun-18	2.06	Jan-20	1.64	Aug-21	1.5
Jul-18	2.21	Feb-20	1.62	Sep-21	1.48
Aug-18	2.46	Mar-20	1.66	Oct-21	1.5
Sep-18	2.51	Apr-20	1.87	Nov-21	1.53
Oct-18	2.32	May-20	1.79	Dec-21	1.5
Nov-18	2.25	Jun-20	1.7	Jan-22	1.51
Dec-18	2.07	Jul-20	1.6	Feb-22	1.51
Jan-19	1.82	Aug-20	1.65	Mar-22	1.47
Feb-19	1.79	Sep-20	1.58	Apr-22	1.39
Mar-19	1.83	Oct-20	1.47	May-22	1.3
Apr-19	1.89	Nov-20	1.52	Jun-22	1.32
May-19	2.76	Dec-20	1.46	Jul-22	1.34
Jun-19	1.78	Jan-21	1.46	Aug-22	1.34
Jul-19	1.86	Feb-21	1.47	Sep-22	1.32

csv_yahoo-search-market-share-w...
电子表格

How was the overall trend of Yahoo! Search engine market share?

Summary of Yahoo! Search Engine Market Share (2018–2025):

- **2018–2019**: Yahoo's market share fluctuated between approximately **1.9% and 2.7%**, reaching its **highest point in April 2019**, with several earlier increases starting in mid-2018.
- **2020–2021**: The share hovered around **1.7% to 1.9%**, with some minor dips and recoveries, suggesting a phase of relative stability.
- **2022–2023**: A noticeable decline followed, with the market share dropping to its **lowest point of about 1.1% in mid-2023**.
- **2024–early 2025**: A mild rebound occurred, and by January 2025, the share had risen back to roughly **1.6%**.

Overall Trend:

Although Yahoo! Search experienced occasional short-term gains, the overall trend from 2018 to 2025 shows a **gradual downward movement**, with the final market share ending lower than where it began.

From the data:

- **Time period:** 2018–2025
- **Starting share (Jan 2018):** 1.83%
- **Ending share (most recent value):** 1.34%
- **Average share:** ~1.55%
- **Overall trend:** The market share of **Yahoo! Search has decreased** over time.

In summary, Yahoo's global search engine market share shows a **gradual decline** between 2018 and 2025.

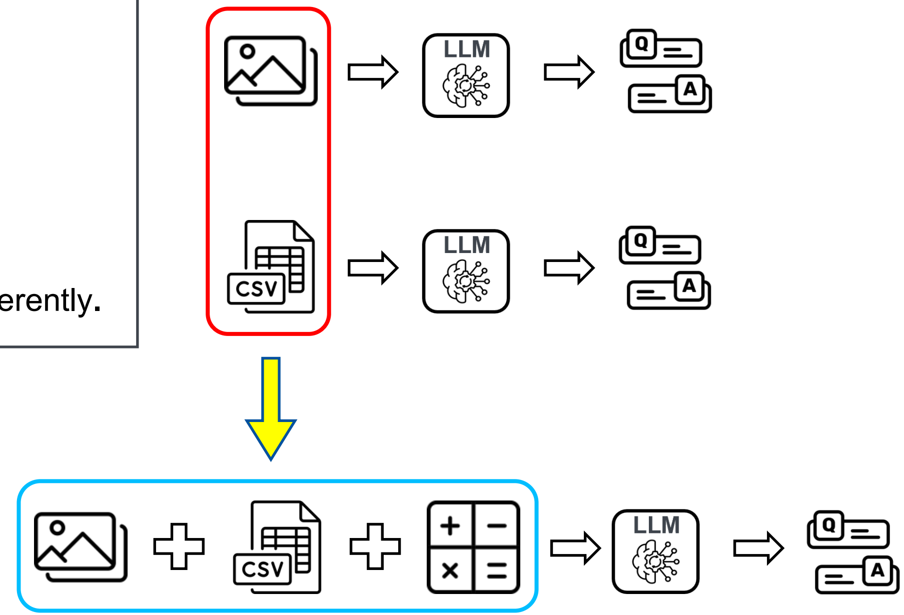
Semantically insightful but imprecise

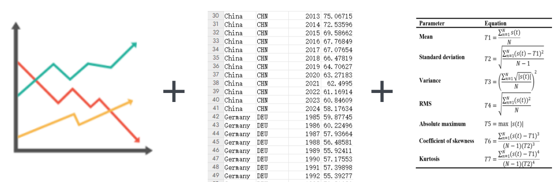
Precise but not semantically insightful

How to make them complement each other?

Parameter	Equation
Mean	$T1 = \frac{\sum_{n=1}^{N} s(t)}{N}$
Standard deviation	$T2 = \sqrt{\frac{\sum_{n=1}^{N}(s(t) - T1)^2}{N-1}}$
Variance	$T3 = \left(\frac{\sum_{n=1}^{N}\sqrt{\lvert s(t)\rvert}}{N}\right)^2$
RMS	$T4 = \sqrt{\frac{\sum_{n=1}^{N}(s(t))^2}{N}}$
Absolute maximum	$T5 = \max \lvert s(t)\rvert$
Coefficient of skewness	$T6 = \frac{\sum_{n=1}^{N}(s(t) - T1)^3}{(N-1)(T2)^3}$
Kurtosis	$T7 = \frac{\sum_{n=1}^{N}(s(t) - T1)^4}{(N-1)(T2)^4}$

CSVs
vo coherently.





💡 **Idea** — What if we …

1. Combine both images and CSVs
2. Add mathematical algorithm

? Accuracy↑

Improvement by how much?

2 Basics

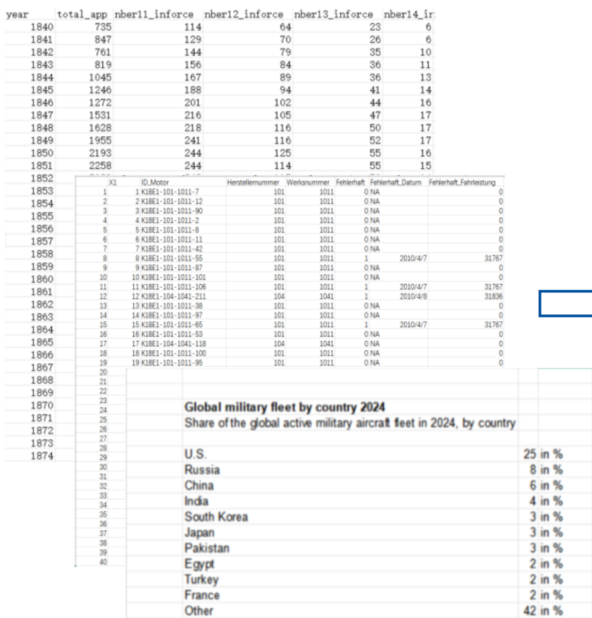
- Chart Image
- Time Series Data
- Mathematical Methods
- Multimodal LLM

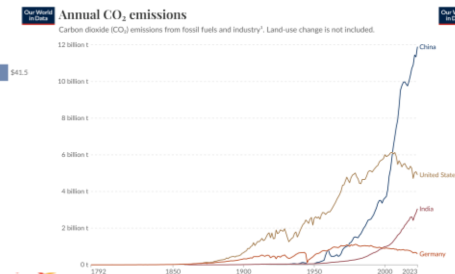
Basics

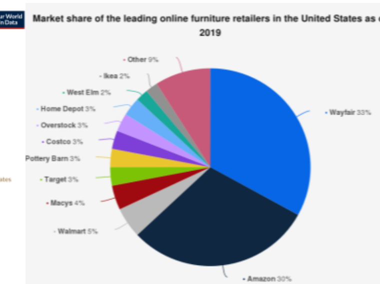
Chart Image —— Data Visualization

Data Visualization:

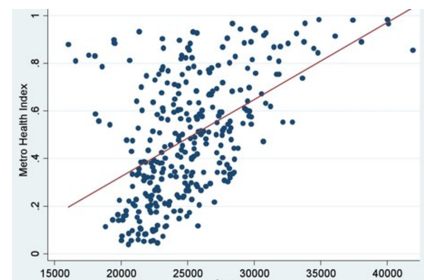
Abstract data → Visual chart image

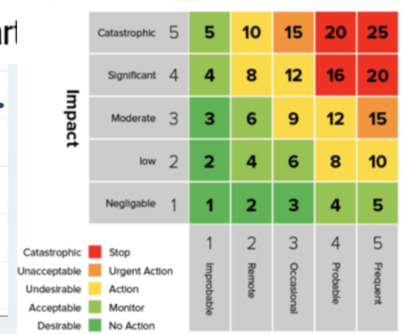






- Bar Chart (comparison)
- Line Chart (trend)
- Pie Chart (proportion)





- Scatter Plot (correlation)
- Heat Map (distribution)

Basics

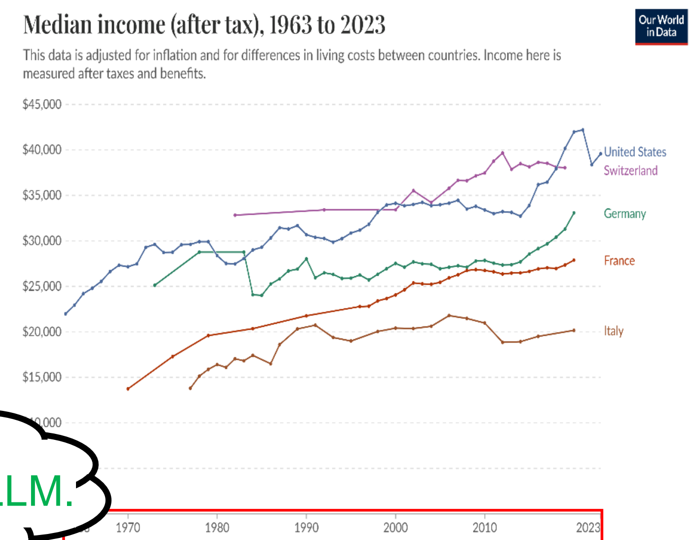
Time Series Data

- Definition: Time series data = data with a **time dimension**.

  Line chart + time axis → time series visualization

30	China	CHN	2013	75.06715
31	China	CHN	2014	72.53596
32	China	CHN	2015	69.58662
33	China	CHN	2016	67.76849
34	China	CHN	2017	67.07654
35	China	CHN	2018	66.47819
36	China	CHN	2019	64.70627
37	China	CHN	2020	63.27183
38	China	CHN	2021	62.4995
39	China	CHN	2022	61.16914
40	China	CHN	2023	60.84609
41	China	CHN	2024	58.17634
42	Germany	DEU	1985	59.87745
43	Germany	DEU	1986	60.22496
44	Germany	DEU	1987	57.93664
45	Germany	DEU	1988	56.48581
46	Germany	DEU	1989	[illegible]
47	Germany	DEU	1990	[illegible]
48	Germany	DEU	[illegible]	[illegible]
49	Germany	DEU	[illegible]	[illegible]
50	Germany	DEU	[illegible]	[illegible]

Visualization



Directly readable by LLM.

Time Series Data	CSV	Time-indexed Data Frame	NumPy Array
Data Structure	Text-based table	Pandas structure	Numerical array

Basics

Mathematical Methods

Why Combine Mathematical Methods?

- Quantifies raw numerical data
- Provides descriptive semantical me

! Conclusion:
Mathematical methods bridge the gap
numerical values and **semantic unde**

	Feature	Description	Python Packages (other than numpy, pandas)
	Max, Min, Range	Captures amplitude variation.	(custom logic)
Time-Domain	Peaks and Valleys	Local maxima and minima counts and	scipy / (custom logic)

Parameter	Equation
	$\frac{\sum_{n=1}^{N} s(t)}{N}$
	$\sqrt{\frac{\sum_{n=1}^{N}(s(t)-T1)^2}{N-1}}$
	$\left(\frac{\sum_{n=1}^{N}\sqrt{\lvert s(t)\rvert}}{N}\right)^2$
	$\sqrt{\frac{\sum_{n=1}^{N}(s(t))^2}{N}}$
	$\max \lvert s(t)\rvert$

[1]:
```python
import pandas as pd
import numpy as np
from scipy.stats import skew, kurtosis
from scipy.signal import find_peaks
import matplotlib.pyplot as plt

# === Load CSV File ===
file_path = '/drive/csv_yahoo-search-market-share-worldwide-2018-202
df = pd.read_csv(file_path)

# Display CSV preview
df.head()
```

```
Matplotlib is building the font cache; this may take a moment.
```

[1]:

	Date	Global search engine traffic market share of Yahoo! Search
0	2018-01	1.83
1	2018-02	1.94
2	2018-03	2.13
3	2018-04	2.09
4	2018-05	2.08

[2]:
```python
# === Feature Extraction ===
value_col = df.select_dtypes(include=[np.number]).columns[0]
data = df[value_col].values

# Time-Domain Features
max_val = np.max(data)
min_val = np.min(data)
range_val = max_val - min_val

peaks, _ = find_peaks(data)
valleys, _ = find_peaks(-data)
num_peaks = len(peaks)
num_valleys = len(valleys)

trend_coef = np.polyfit(np.arange(len(data)), data, deg=1)[0]
volatility = pd.Series(data).rolling(window=3, center=True).std().mean()
skewness = skew(data)
kurt = kurtosis(data)

# Statistical Features
mean_val = np.mean(data)
std_val = np.std(data)
iqr_val = np.percentile(data, 75) - np.percentile(data, 25)

# Print Results
print("===== Time-Domain Features =====")
print(f"Max: {max_val:.4f}")
print(f"Min: {min_val:.4f}")
print(f"Range: {range_val:.4f}")
print(f"Number of Peaks: {num_peaks}")
print(f"Number of Valleys: {num_valleys}")
print(f"Trend (slope): {trend_coef:.6f}")
print(f"Volatility (avg rolling std): {volatility:.6f}")
print(f"Skewness: {skewness:.6f}")
print(f"Kurtosis: {kurt:.6f}")
print("\n===== Statistical Features =====")
print(f"Mean: {mean_val:.4f}")
print(f"Std Dev: {std_val:.4f}")
print(f"IQR: {iqr_val:.4f}")
```

```
===== Time-Domain Features =====
Max: 2.7600
Min: 1.0800
Range: 1.6800
Number of Peaks: 19
Number of Valleys: 19
Trend (slope): -0.012723
Volatility (avg rolling std): 0.069175
Skewness: 1.042926
Kurtosis: 0.833659

===== Statistical Features =====
Mean: 1.5480
Std Dev: 0.3614
IQR: 0.5200
```

- Raw CSV data
- Algorithm-Based Mathematical Analysis
- Extracted Numerical Results

How to connect with LLM?

Basics

Mathematical Methods

Connection with LLM

Jan-18	1.83	Aug-19	1.8	Mar-21	1.48
Feb-18	1.94	Sep-19	1.64	Apr-21	1.52
Mar-18	2.13	Oct-19	1.61	May-21	1.5
Apr-18	2.09	Nov-19	1.6	Jun-21	1.53
May-18	2.08	Dec-19	1.59	Jul-21	1.47
Jun-18	2.06	Jan-20	1.64	Aug-21	1.5
Jul-18	2.21	Feb-20	1.62	Sep-21	1.48
Aug-18	2.46	Mar-20	1.66	Oct-21	1.5
Sep-18	2.51	Apr-20	1.87	Nov-21	1.53
Oct-18	2.32	May-20	1.79	Dec-21	1.5
Nov-18	2.25	Jun-20	1.7	Jan-22	1.51
Dec-18	2.07	Jul-20	1.6	Feb-22	1.51
Jan-19	1.82	Aug-20	1.65	Mar-22	1.47
Feb-19	1.79	Sep-20	1.58	Apr-22	1.39
Mar-19	1.83	Oct-20	1.47	May-22	1.3
Apr-19	1.89	Nov-20	1.52	Jun-22	1.32
May-19	2.76	Dec-20	1.46	Jul-22	1.34
Jun-19	1.78	Jan-21	1.46	Aug-22	1.34
Jul-19	1.86	Feb-21	1.47	Sep-22	1.32

CSV

What are the time-domain features and statistical features of this CSV data? And how much are they?

Question

python

复制代码

```python
import pandas as pd
import numpy as np
from scipy.stats import linregress

# Load and clean data
data = pd.read_csv("csv_yahoo-search-market-share-worldwide-2018-2025.csv")
data.columns = ["Date", "MarketShare"]
data["MarketShare"] = data["MarketShare"].astype(float)

# --- Time-domain features ---
x = np.arange(len(data))
y = data["MarketShare"].values

# Trend analysis
slope, intercept, r_value, p_value, std_err = linregress(x, y)
autocorr = data["MarketShare"].autocorr(lag=1)
volatility_ratio = data["MarketShare"].std() / data["MarketShare"].mean()

# --- Statistical features ---
features = {
    "Mean": np.mean(y),
    "Median": np.median(y),
    "Standard Deviation": np.std(y),
    "Minimum": np.min(y),
    "Maximum": np.max(y),
    "Range": np.max(y) - np.min(y),
    "25th Percentile": np.percentile(y, 25),
    "75th Percentile": np.percentile(y, 75),
    "Trend Slope": slope,
    "Trend Intercept": intercept,
    "Autocorrelation (lag=1)": autocorr,
    "Volatility Ratio (std/mean)": volatility_ratio,
    "Data Length (N)": len(y)
}

features_df = pd.DataFrame(list(features.items()), columns=["Feature", "Value"])
print(features_df)
```

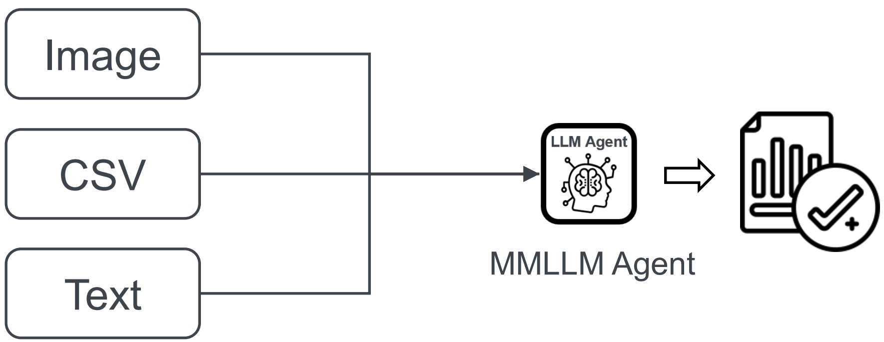
Math Result

```
===== Time-Domain Features =====
Max: 2.7600
Min: 1.0800
Range: 1.6800
Number of Peaks: 19
Number of Valleys: 19
Trend (slope): -0.012723
Volatility (avg rolling std): 0.091100
Skewness: 1.042926
Kurtosis: 0.833659

===== Statistical Features =====
Mean: 1.5480
Std Dev: 0.3614
IQR: 0.5200
```

Basics

Multimodal LLM

- Key: MMLLM can **integrate** info from **different types of data modalities**
  - **Visual** modality (Chart image) provides **semantic** cues.
  - **Numerical** modality (CSV) enables **exact** numerical computation.
  - **Textual** modality (Question) defines **reasoning** intent. (User Intention)
- From Manual to Autonomous Interpretation



- The Multimodal LLM combines **Visual**, **Numerical**, and **Textual** modalities with **Mathematical Methods**, achieving autonomous chart analysis.

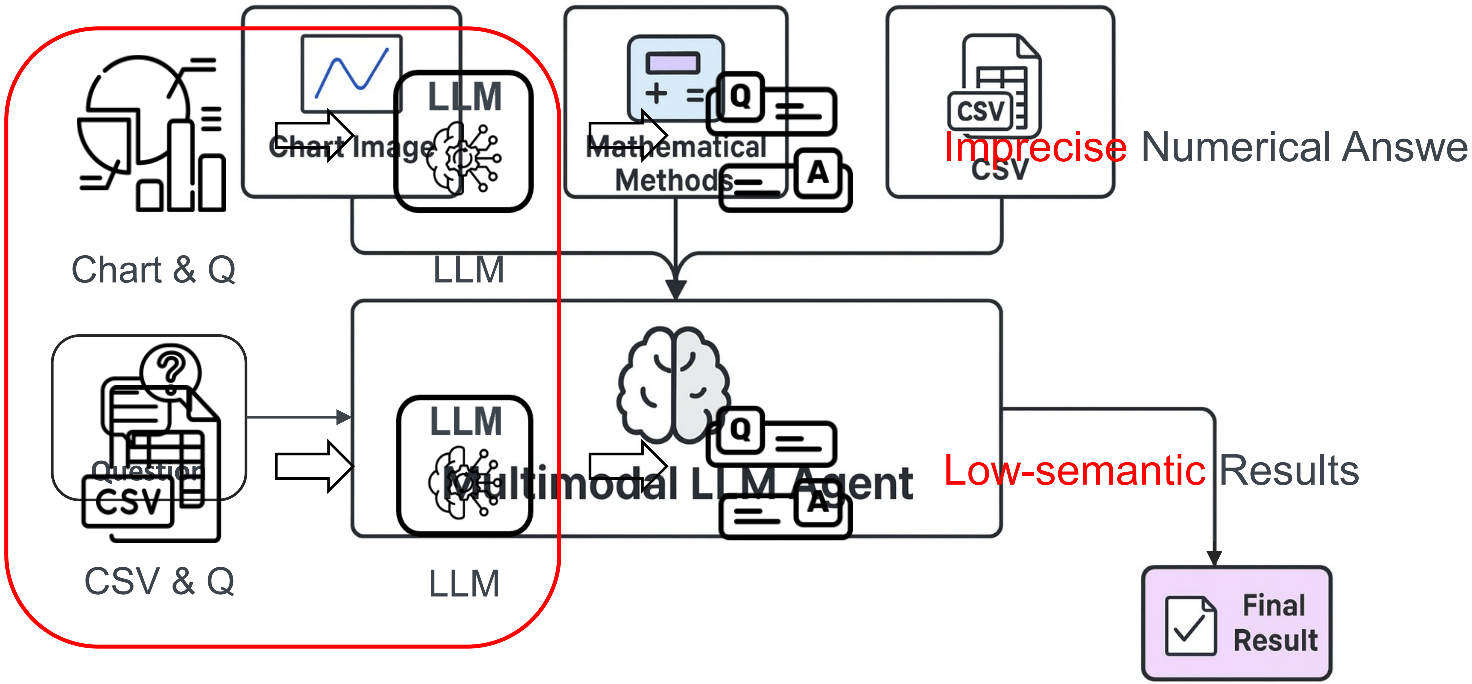
3 Conceptual Design

- System Overview
- System Design

Conceptual Design

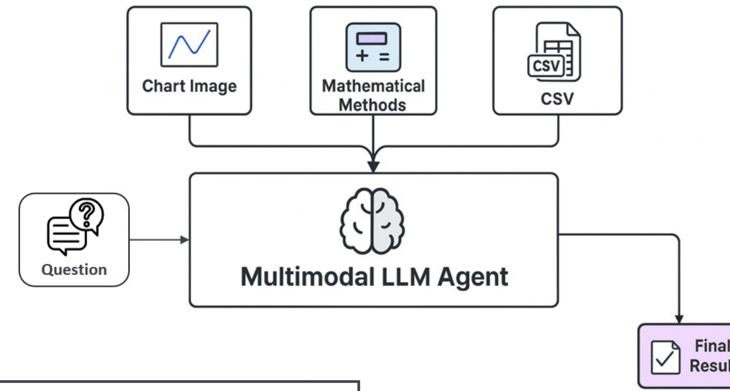
System Overview

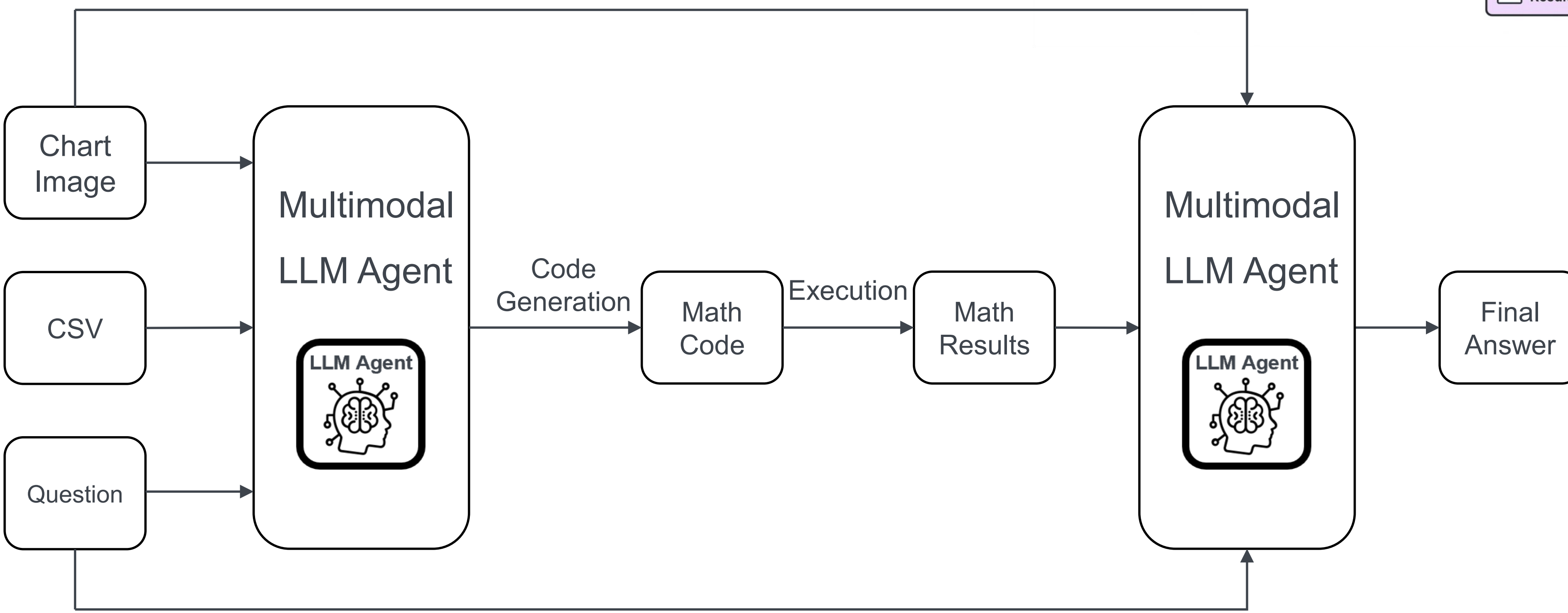
- Single-modality analysis
- Multi-modalities analysis



Conceptual Design

System Overview



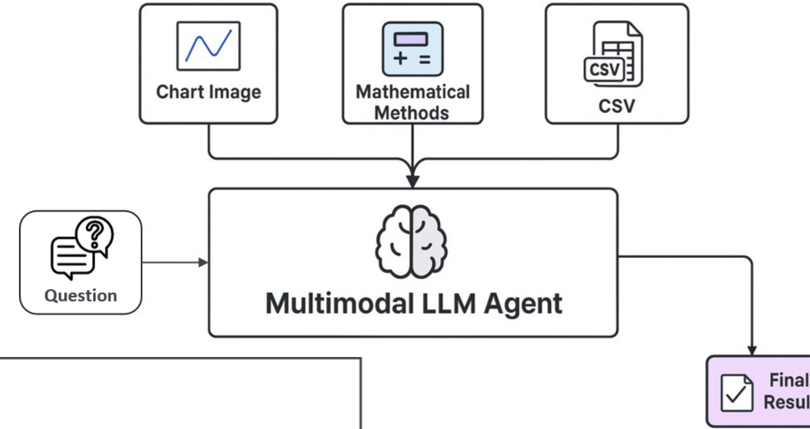


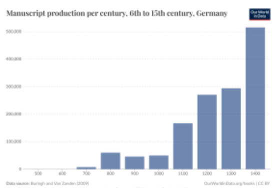
4 Experiment

- Multimodal Data Input
- LLM-Based Analysis
- Feature Planning
- Code Generation & Execution
- Raw Extracted Results
- Answer Synthesis

Conceptual Design

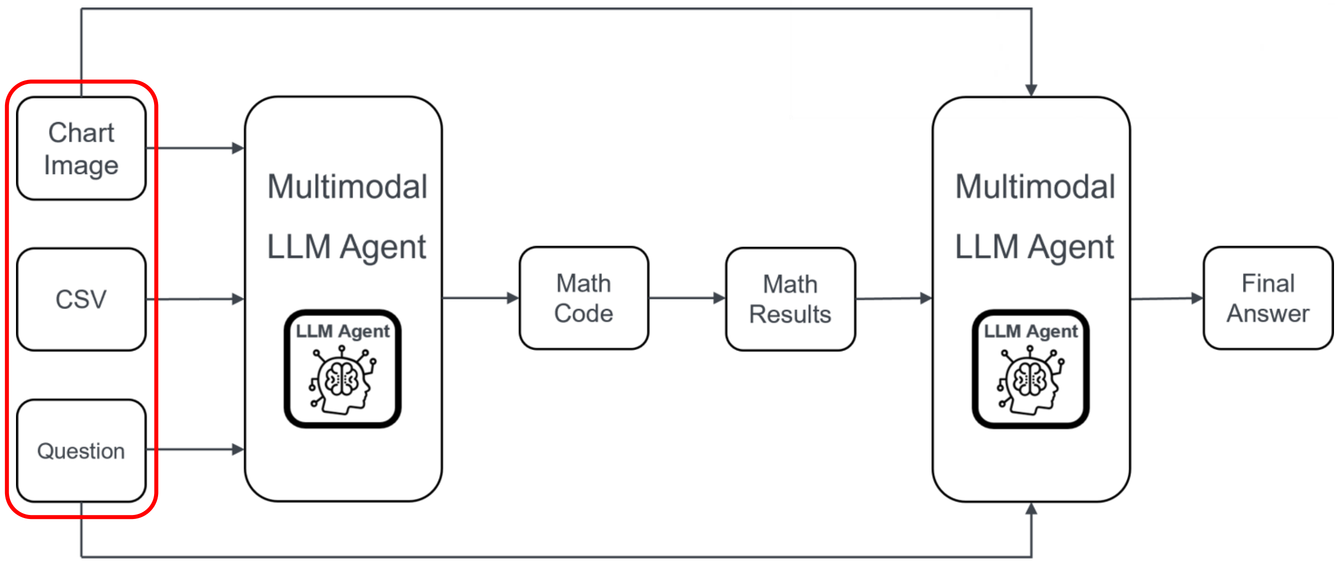
Multimodal Data Input





Entity	Code	Year	Manuscript
Germany	DEU	500	0
Germany	DEU	600	0
Germany	DEU	700	7503
Germany	DEU	800	59771
Germany	DEU	900	45703
Germany	DEU	1000	49548
Germany	DEU	1100	166876
Germany	DEU	1200	270392
Germany	DEU	1300	293814
Germany	DEU	1400	515116

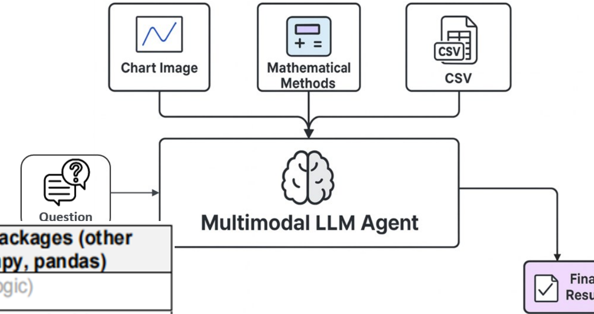
What overall trend can be observed in manuscript production from the 8th to the 15th century?

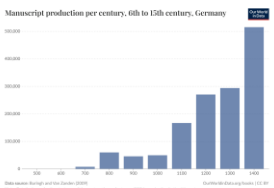



```
Stage 1: Image CSV Question Analysis
------------------------------
📁 Image path: "C:\Users\xwjbuer\Desktop\Test\D15\D15-manuscript-production-century2.png"
💬 Analysis question: What overall trend can be observed in manuscript production from the 8th century to the 15th century?
📊 CSV path: "C:\Users\xwjbuer\Desktop\Test\D15\manuscript-production-century.filtered\manuscript-production-century.csv"
```

Conceptual Design

LLM-Based Analysis

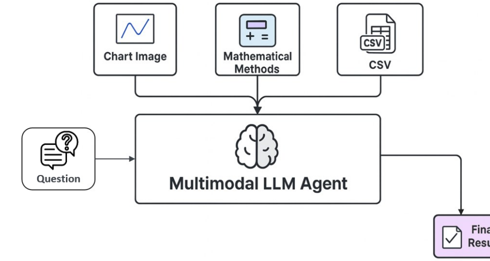




Entity	Code	Year	Manuscript
Germany	DEU	500	0
Germany	DEU	600	0
Germany	DEU	700	7503
Germany	DEU	800	59771
Germany	DEU	900	45703
Germany	DEU	1000	49548
Germany	DEU	1100	166876
Germany	DEU	1200	270392
Germany	DEU	1300	293814
Germany	DEU	1400	515116

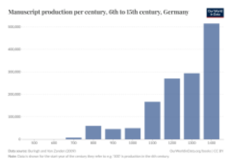
What overall trend can be observed in manuscript production from the 8th to the 15th century?

	Feature	Description	Python Packages (other than numpy, pandas)
	Max, Min, Range	Captures amplitude variation.	(custom logic)
Time-Domain Features	Peaks and Valleys	Local maxima and minima counts and positions.	scipy / (custom logic)
	Rise Time and Fall Time	Duration for rising or falling between thresholds.	(custom logic)
	Trend	Long-term movement via moving averages or regression.	(custom logic)
	Volatility	Variation over a moving window.	(custom logic)
	Skewness	Asymmetry of value distribution.	scipy / (custom logic)
	Kurtosis	Sharpness of the distribution peak.	scipy / (custom logic)
	Edge Detection	Identifies abrupt changes in values.	scipy / (custom logic)
	Sliding Window Features	Rolling metrics (mean, std, max, min, etc.) for local dynamics.	scipy / (custom logic)
	Zero-Crossing Rate (ZCR)	Times the signal crosses zero.	(custom logic)
Frequency-Domain Features	Spectral Power	Energy across frequency components.	scipy / (custom logic)
	Dominant Frequency	Frequency with the highest energy.	scipy / (custom logic)
	Frequency Bandwidth	Frequency range around the dominant frequency.	scipy / (custom logic)
Statistical Features	Mean	Average pattern of the series.	(custom logic)
	Standard Deviation	Magnitude of fluctuation around the statistical mean.	(custom logic)
	Interquartile Range (IQR)	Range between the 25th and 75th percentiles.	(custom logic)
	Repeating Patterns	Detects cyclic or recurring structures.	statsmodels / (custom logic)
	Sample Entropy	Measure of randomness or irregularity.	antropy, nolds / (custom logic)



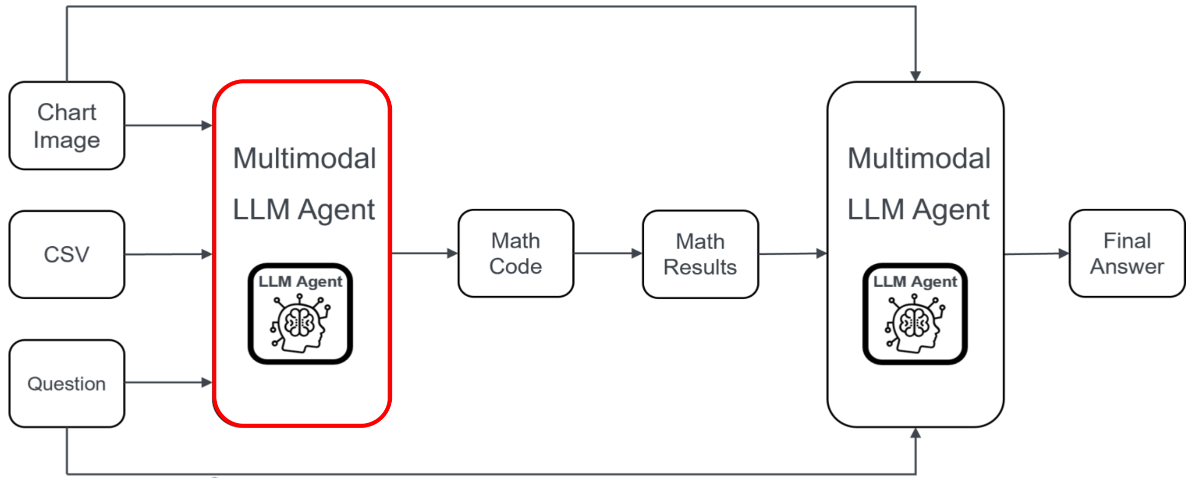
Conceptual Design

LLM-Based Reasoning



Entity	Code	Year	Manuscript
Germany	DEU	500	0
Germany	DEU	600	0
Germany	DEU	700	7503
Germany	DEU	800	59771
Germany	DEU	900	45703
Germany	DEU	1000	49548
Germany	DEU	1100	166876
Germany	DEU	1200	270392
Germany	DEU	1300	293814
Germany	DEU	1400	515116

What overall trend can be observed in manuscript production from the 8th to the 15th century?



🔍 Selected Features:

1. category_max [Bar Chart Features]
   - 📝 Reason: Identifies the century with the highest manuscript production, indicating an increasing trend.
   - 🔥 Priority: high
   - 💡 Calculation: Find the century with the maximum bar height.

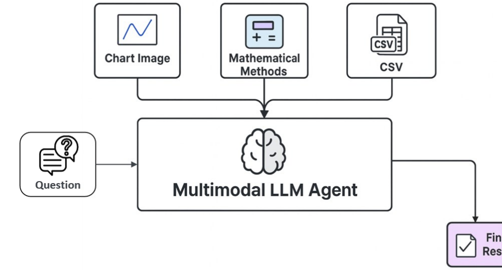
2. category_min [Bar Chart Features]
   - 📝 Reason: Identifies the century with the lowest manuscript production, supporting the trend analysis.
   - 🔥 Priority: medium
   - 💡 Calculation: Find the century with the minimum bar height.

3. difference_between_categories [Bar Chart Features]
   - 📝 Reason: Calculates the increase in manuscript production over the centuries.
   - 🔥 Priority: high
   - 💡 Calculation: Compute the difference between the maximum and minimum values.

LLM-Based Analysis Results

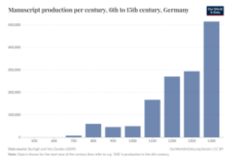
- Which feature?
- Why this feature?
- How to calculate?

```
🔍 Generated feature extraction code:
------------------------------------------------------------
# -*- coding: utf-8 -*-
import pandas as pd
import numpy as np
import json

def convert_to_serializable(obj):
    """Convert non-serializable objects to serializable formats."""
    if isinstance(obj, (np.integer, np.floating)):
        return obj.item()
    elif isinstance(obj, np.ndarray):
        return obj.tolist()
    elif isinstance(obj, pd.Timestamp):
        return obj.isoformat()
    else:
        return str(obj)

def detect_column_types(df):
    """Detect column types based on name patterns and data types."""
    time_columns = []
    value_columns = []
    category_columns = []

    for col in df.columns:
        col_lower = col.lower()
        if any(keyword in col_lower for keyword in ['year', 'date', 'time', 'day', 'month']):
            time_columns.append(col)
        elif pd.api.types.is_numeric_dtype(df[col]):
            value_columns.append(col)
        elif any(keyword in col_lower for keyword in ['category', 'entity', 'country', 'region']):
            category_columns.append(col)

    return time_columns, value_columns, category_columns

def extract_features_from_bar_chart(csv_file_path):
    """Extract selected features from a bar chart data."""
    # Read the CSV file
    df = pd.read_csv(csv_file_path)

    # Detect column types
    time_columns, value_columns, category_columns = detect_column_types(df)

    # Determine the x-axis and y-axis based on Stage A context
    # Stage A indicates a time span from 8th to 15th century, so we use 'Year' as x-axis
    # and 'Manuscript production per century' as y-axis
    x_axis = next((col for col in time_columns if 'year' in col.lower()), None)
    y_axis = next((col for col in value_columns if 'manuscript production' in col.lower()), None)

    if not x_axis or not y_axis:
        raise ValueError("Required columns for x-axis or y-axis not found.")

    # Extract features
    features = {}

    # Feature: category_max
    max_value = df[y_axis].max()
    max_century = df.loc[df[y_axis] == max_value, x_axis].iloc[0]
    features['category_max'] = {
        'century': convert_to_serializable(max_century),
        'value': convert_to_serializable(max_value)
    }

    # Feature: category_min
    min_value = df[y_axis].min()
    min_century = df.loc[df[y_axis] == min_value, x_axis].iloc[0]
    features['category_min'] = {
        'century': convert_to_serializable(min_century),
        'value': convert_to_serializable(min_value)
    }

    # Feature: category_min
    min_value = df[y_axis].min()
    min_century = df.loc[df[y_axis] == min_value, x_axis].iloc[0]
    features['category_min'] = {
        'century': convert_to_serializable(min_century),
        'value': convert_to_serializable(min_value)
    }

    # Feature: difference_between_categories
    difference = max_value - min_value
    features['difference_between_categories'] = convert_to_serializable(difference)

    return features

def main():
    # Define the CSV file path
    csv_file_path = r'C:\Users\xwjbuer\Desktop\Test\D15\manuscript-production-century.filtered\manuscript-production-century.csv'

    # Extract features
    features = extract_features_from_bar_chart(csv_file_path)

    # Print results as JSON
    print(json.dumps(features, indent=4))

if __name__ == "__main__":
    main()
------------------------------------------------------------
```



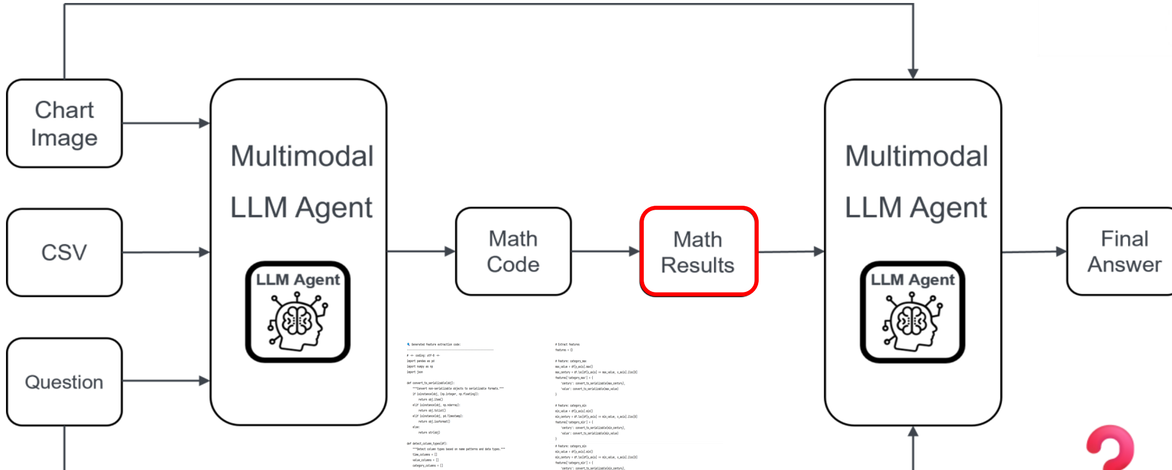
Conceptual Design

Raw Numerical Results



Entity	Code	Year	Manuscript
Germany	DEU	500	0
Germany	DEU	600	0
Germany	DEU	700	7503
Germany	DEU	800	59771
Germany	DEU	900	45703
Germany	DEU	1000	49548
Germany	DEU	1100	166876
Germany	DEU	1200	270392
Germany	DEU	1300	293814
Germany	DEU	1400	515116

What overall trend can be observed in manuscript production from the 8th to the 15th century?



```
⚡ Executing feature extraction...
============================================================
📊 Feature extraction results:
{
  "raw_output": "{\n    \"category_max\": {\n        \"century\": 1400,\n        \"value\": 515116\n    },\n
                 \"category_min\": {\n        \"century\": 500,\n        \"value\": 0\n    },\n
                 \"difference_between_categories\": 515116\n}"
}
============================================================
✅ Feature extraction completed!
```

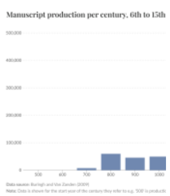
Precise but Non-semantic

❓ HOW

Precise and Semantic

Conce

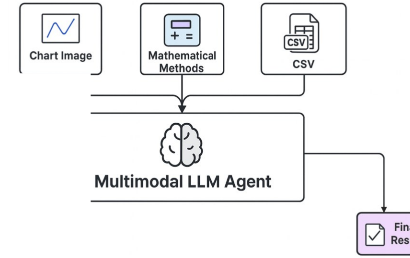
Answe



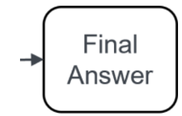
Entity	Code	Year
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	
Germany	DEU	

What overall tren
observed in manu
production from t
the 15th century?

```
⚡ Executing feature extraction.
================================
📊 Feature extraction results:
{
 "raw_output": "{\n    \"categ
                      \"categ
                      \"differ
}
================================
✅ Feature extraction completed!
```




```python
def synthesize_answer(self, extracted_features):  1 usage
    """
    Stage 3: Use extracted features to generate final answer
    """
    print("\n🎯 Stage 3: Synthesizing final answer...")
    print("=" * 60)

    # Build prompt with extracted features and original question
    prompt = f"""
    Based on the following extracted features and the original question, provide a precise answer.

    Original Question: {self.experiment_log['inputs']['question']}

    Chart Type: {self.chart_type}

    Extracted Features:
    {json.dumps(extracted_features, indent=2)}

    Instructions:
    - Use the features directly to answer the question.
    - If the features contain multiple values, focus on the most relevant ones.
    - Be specific and include numerical values where available.
    - If the question involves comparison, highlight the differences.
    - Keep the answer clear and to the point.
    - If the features don't fully answer the question, explain what's missing.

    Answer:
    """
```




```
===================
century to the 15th century
ts peak in the 15th century
the 6th century. This
over the centuries.
===================
```

5 Evaluation & Analysis

- Test Case
- Experiment Example
- Experiment Results
- Evaluation & Analysis

Evaluation & Analysis

Test Case

D1
D4
D7
D10
D13
D16
D19
D22
D25
D28

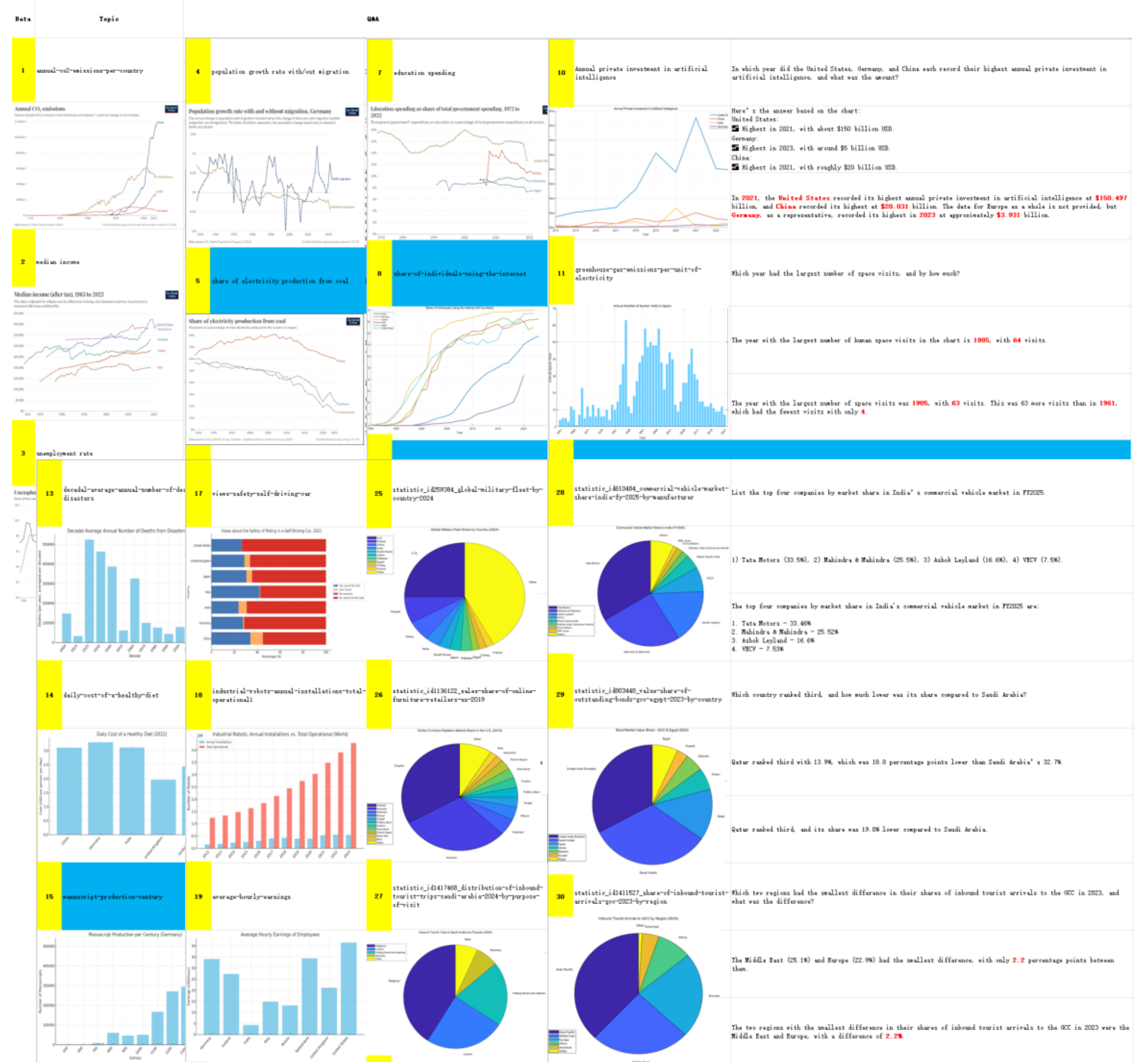
Line

D2
D5
D8
D11
D14
D17
D20
D23
D26
D29

Bar

D3
D6
D9
D12
D15
D18
D21
D24
D27
D30

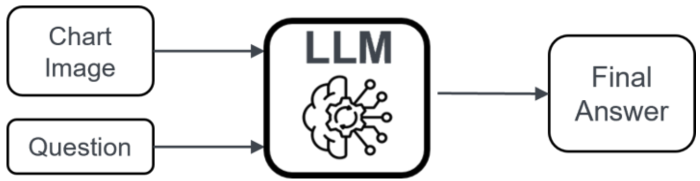
Pie



Evaluation & Analysis

Ablation Experiment

Baseline:



Only LLM
No CSV Input

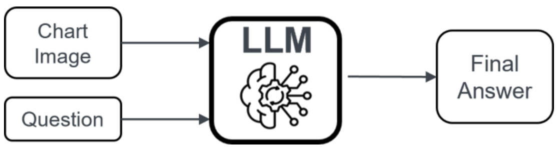
GPT-Analyzer:

Ablation: Without Mathematical Algorithm

Evaluation & Analysis

Test Case Example

Baseline:



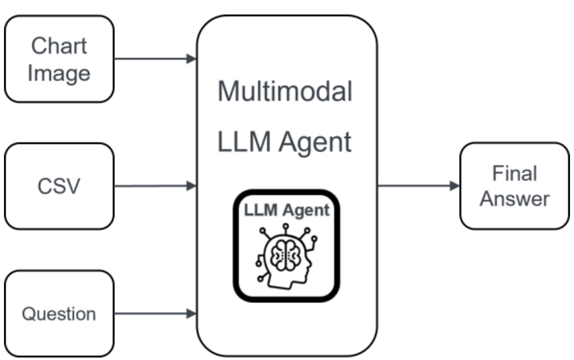
☑ Image
☒ CSV
☒ Math

☑ Image
☑ CSV
☒ Math

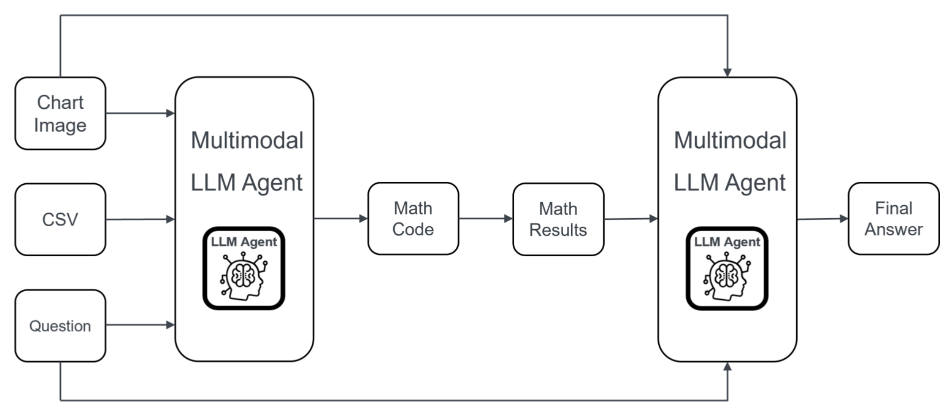
☑ Image
☑ CSV
☑ Math

Question / System Type	What **overall trend** can be observed in manuscript production from the 8th century to the 15th century?
Baseline	Manuscript production generally increased over the centuries, with **slow growth** before the 11th century, followed by **rapid expansion** from the 12th to the 15th century.
Ablation Experiment	Manuscript production in Germany shows a strong upward trend from the 8th to the 15th century — starting at a very low level in the early centuries and **increasing steadily**, with the most rapid growth after the 11th century, reaching its peak in the 15th century at **over 500,000** manuscripts.
GPT-Analyzer	From the 8th century to the 15th century, manuscript production shows a significant upward trend. Starting from **7,503** manuscripts in the **8th** century, production increased to **515,116** manuscripts by the **15th** century. The most notable increases occurred between the 11th and 12th centuries (an **increase of 103,516** manuscripts) and between the 13th and 14th centuries (an **increase of 221,302** manuscripts). Overall, there is a clear and **substantial growth** in manuscript production over these centuries.

Ablation:



GPT-Analyzer:



Evaluation & Analysis

Experiment Results

Chart Nr.	Metric	Baseline	Ablation	GPT-Analyzer	Chart Type
1	Correctness	0	0	1	line
	Accuracy	0	0	1	line
2	Correctness	1	1	1	line
	Accuracy	0	0	1	line
3	Correctness	1	1	1	line
	Accuracy	0	0.5	1	line
4	Correctness	1	1	1	line
	Accuracy	0	1	1	line
5	Correctness	1	1	1	line
	Accuracy	0	0	1	line
6	Correctness	1	0	1	line
	Accuracy	1	0	1	line
7	Correctness	1	1	1	line
	Accuracy	0	0.5	1	line
8	Correctness	0	1	1	line
	Accuracy	0	0	1	line
9	Correctness	1	1	1	line
	Accuracy	0	0	1	line
10	Correctness	1	1	1	line
	Accuracy	0	0.5	1	line
11	Correctness	0	1	1	bar
	Accuracy	0	1	1	bar
12	Correctness	1	1	1	bar
	Accuracy	0	0.5	1	bar
13	Correctness	1	1	1	bar
	Accuracy	0	0.5	1	bar
14	Correctness	1	1	1	bar
	Accuracy	0.5	0.5	0.5	bar
15	Correctness	0	0	1	bar
	Accuracy	0	0	1	bar
16	Correctness	1	1	1	bar
	Accuracy	0	0.5	1	bar
17	Correctness	0	0	1	bar
	Accuracy	0	0	1	bar
18	Correctness	1	1	1	bar
	Accuracy	0	0	1	bar
19	Correctness	0	1	1	bar
	Accuracy	0	1	1	bar
20	Correctness	1	1	0	bar
	Accuracy	0	1	0	bar
21	Correctness	1	1	1	pie
	Accuracy	0	1	1	pie
22	Correctness	1	1	1	pie
	Accuracy	0	1	1	pie
23	Correctness	1	1	1	pie
	Accuracy	0	1	1	pie
24	Correctness	1	1	1	pie
	Accuracy	1	1	1	pie
25	Correctness	1	1	1	pie
	Accuracy	1	1	1	pie
26	Correctness	1	1	1	pie
	Accuracy	1	1	1	pie
27	Correctness	0	0	0	pie
	Accuracy	0	0	0	pie
28	Correctness	1	1	1	pie
	Accuracy	0	1	1	pie
29	Correctness	1	1	1	pie
	Accuracy	0	1	1	pie
30	Correctness	0	1	1	pie
	Accuracy	0	1	1	pie

Metrics:

- **Correctness** measures whether the answer correctly addresses the question.
- **Accuracy** measures precision of numerical values in the correct answer.

☑ Image
☒ CSV
☒ Math

☑ Image
☑ CSV
☒ Math

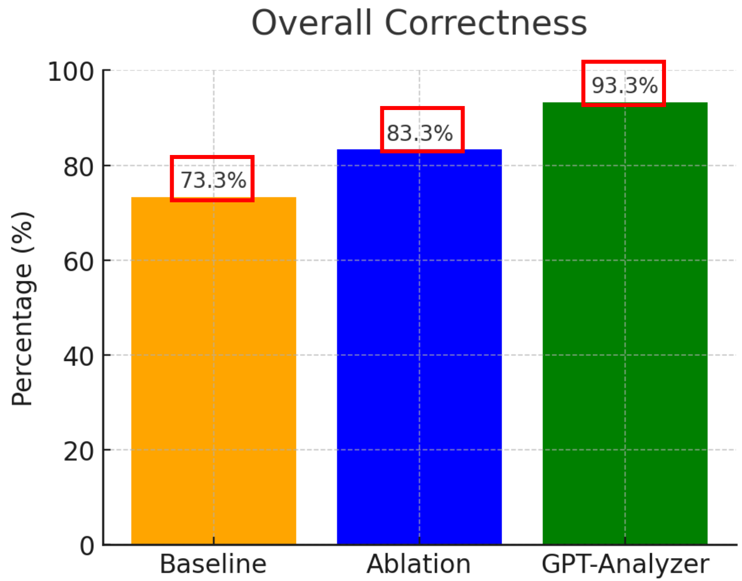
☑ Image
☑ CSV
☑ Math

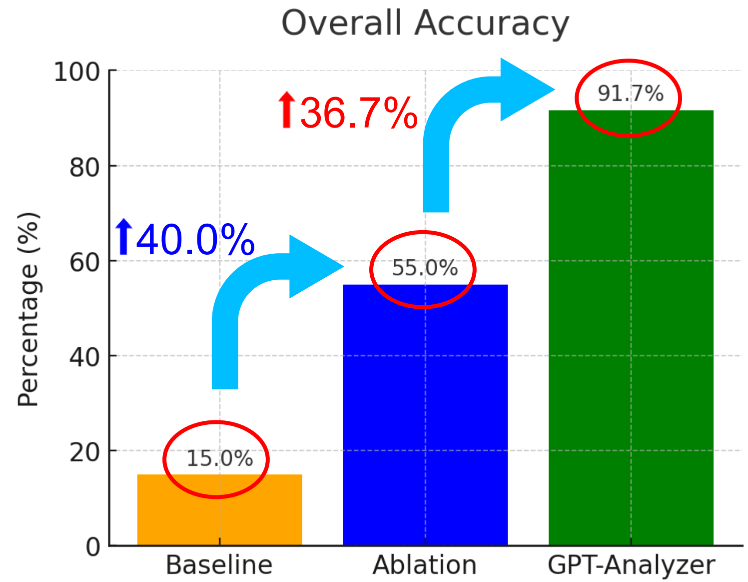
Score	Baseline	Ablation	GPT-Analyzer
Correctness	22/30	25/30	28/30
Accuracy	4.5/30	16.5/30	27.5/30

Evaluation & Analysis

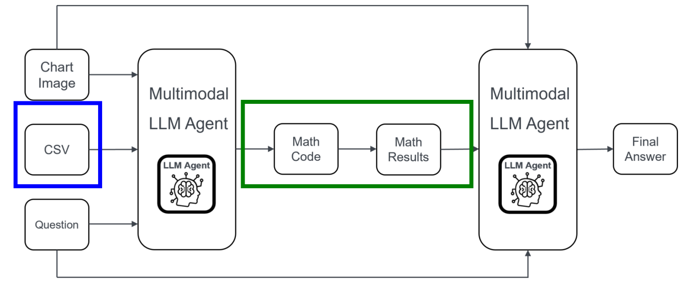
Evaluation & Analysis

- In conclusion, these results clearly show that combining **math code** with **LLM reasoning** significantly enhances LLM's analysis in handling numerical data.





Metric	Baseline	Ablation	GPT-Analyzer
Correctness	73.3%	83.3%	93.3%
Accuracy	15%	55.0% ⬆	91.7% ⬆⬆



6 Summary & Outlook

Summary and Outlook

Summary

- Designed a system to enhance LLM analysis for numerical data

    (Combines different **data multimodalities: image, textual, CSV**)

- Introduced **mathematical algorithm** to

    → Bridge raw numerical data with **semantic reasoning**

    → Convert plain CSV values into **interpretable features**

Limitation & Future Work

- Current **feature extraction** still relies on predefined algorithms
- Explore **autonomous agent frameworks** for self-improving analysis

Quelle

- Chang C, Wang W Y, Peng W C, et al. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters[J]. ACM Transactions on Intelligent Systems and Technology, 2025, 16(3): 1-20.
- Masry A, Long D X, Tan J Q, et al. Chartqa: A benchmark for question answering about charts with visual and logical reasoning[J]. arXiv preprint arXiv:2203.10244, 2022.
- Zhao T, Sun M. Enhancing LLM's interpretability for time series via multi-level aligned embeddings[J].
- Jia N, Yuan C, Wu Y, et al. Improving LLM Interpretability and Performance via Guided Embedding Refinement for Sequential Recommendation[J]. arXiv preprint arXiv:2504.11658, 2025. Jia N, Yuan C, Wu Y, et al. Improving LLM Interpretability and Performance via Guided Embedding Refinement for Sequential Recommendation[J]. arXiv preprint arXiv:2504.11658, 2025.
- https://www.statista.com/statistics/1219407/market-share-held-by-yahoo-worldwide/?srsltid=AfmBOoparVaxMSlJsHlSCMPqsWb-TV9ZDrfIJMNF4DgSsVp91I5NNE1-

University of Stuttgart
Institut of Industrial Automation
and Software Engineering

Thank you!

Wenjie Xie

e-mail st180778@stud.uni-stuttgart.de
phone +49 (0) 711 685-
fax +49 (0) 711 685-

University of Stuttgart