Will TuringBots Replace Human SDE || DS

Ding Ma

Executive Summary

This study evaluates whether AI-powered coding assistant can, to what extent, replace human developers by analyzing GitHub repositories' data.

Key Insights:

1. Trends in Programming Languages: Python leads in usage, especially for AI and DS projects, while the usage in C++ and Java declined.
2. Big Tech, existing (e.g. Google, Microsoft) and rising Techs (OpenAI), are main contributors to open-source projects incorporating high impacting community-driven libraries like PyTorch and Tensorflow.
3. Significant similarities exist between

Source Data Overview

Data Source: GitHub Archive (~1.36 TiB) in Google Cloud Storage with 5 folders.

Sub-Folder	Description	Use Case
Commits	Metadata about commits: author, date, SHA, parent commits, messages.	Analyze developer activity and trends.
Contents	File contents from repositories.	Analyze code and documentation.
Files	Metadata about repository files: file paths, modes, blob IDs.	Understand file structure and data flow.
Languages	Aggregated usage of programming languages by code size.	Track programming language trends.
Licenses	Licensing information for repositories.	Examine license distribution and correlations.

Methodology and Analysis Approach Overview

1. **Data Preparation**:
    - Extract data from cloud storage (e.g., GCS).
    - Clean missing values, remove duplicates, and standardize formats.
    - Perform exploratory analysis (EDA) to identify patterns, trends, and anomalies.
2. **Analysis (OLAP)**:
    - Aggregate data by key dimensions like time, category, or geography.
    - Use Spark SQL for efficient querying, enabling slicing, dicing, and filtering large datasets.
    - Perform trend analysis, correlation studies, and seasonality checks to derive actionable insights.
    - Group data to compute metrics (e.g., average ratings, top categories) and create summary statistics.
3. **Tools**:
    - **PySpark in GCP Dataproc**: Enables distributed data processing at scale.
    - **Visualization Libraries**: Create plots, dashboards, and reports to effectively communicate findings.

Combines robust processing with multidimensional analysis and visualization for clear, actionable insights.

Data Clean-up & Filtering - 1

Process:

- Eliminated irrelevant and duplicate records.
- Focused on well-populated, recent, and accurate data.
- Convert time_sec into Year-Month-Date Format
- Handled deeply nested data and identified core variables for analysis.
    a. Exploded nested and complex data structures into individual rows
    b. Select relevant columns and sub columns
- Filtered Data into correct time frame

Challenges: Dealing with data noise, size, and nested structures.

- The 2 figures show the schema before and after clean-up

```
start_date = "2007-10-19"
end_date = "2023-12-31"
```

```
root
 |-- commit: string (nullable = true)
 |-- tree: string (nullable = true)
 |-- name: string (nullable = true)
 |-- email: string (nullable = true)
 |-- author_date_seconds: long (nullable = true)
 |-- subject: string (nullable = true)
 |-- message: string (nullable = true)
 |-- repo_name: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- author_date: date (nullable = true)
```

```
root
 |-- commit: string (nullable = true)
 |-- tree: string (nullable = true)
 |-- parent: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- author: struct (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- email: string (nullable = true)
 |    |-- time_sec: long (nullable = true)
 |    |-- tz_offset: long (nullable = true)
 |    |-- date: struct (nullable = true)
 |    |    |-- seconds: long (nullable = true)
 |    |    |-- nanos: long (nullable = true)
 |-- committer: struct (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- email: string (nullable = true)
 |    |-- time_sec: long (nullable = true)
 |    |-- tz_offset: long (nullable = true)
 |    |-- date: struct (nullable = true)
 |    |    |-- seconds: long (nullable = true)
 |    |    |-- nanos: long (nullable = true)
 |-- subject: string (nullable = true)
 |-- message: string (nullable = true)
 |-- trailer: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- key: string (nullable = true)
 |    |    |-- value: string (nullable = true)
 |    |    |-- email: string (nullable = true)
 |-- difference: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- old_mode: long (nullable = true)
 |    |    |-- new_mode: long (nullable = true)
 |    |    |-- old_path: string (nullable = true)
 |    |    |-- new_path: string (nullable = true)
 |    |    |-- old_sha1: string (nullable = true)
 |    |    |-- new_sha1: string (nullable = true)
 |    |    |-- old_repo: string (nullable = true)
 |    |    |-- new_repo: string (nullable = true)
 |-- difference_truncated: boolean (nullable = true)
 |-- repo_name: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- encoding: string (nullable = true)
```

Exploratory Data Analysis (EDA) - Commits

Key Metrics:

1. **Top Author by Commits**:
   - Most active contributor based on total commits.
2. **Top Author by Repositories**:
   - Contributor with the broadest project involvement.
3. **Date with Most Commits**:
   - Peak activity date across all repositories.

Steps:

- Extracted and aggregated commit logs.
- Ranked authors by commits and repository contributions.
- Analyzed commit timestamps for high-activity dates.

```
+-----------+------+
|author_date| count|
+-----------+------+
| 2017-05-23|387824|
| 2017-10-23|384453|
| 2003-02-04|324689|
| 2016-09-20|208416|
| 2016-09-19|203750|
+-----------+------+
```

```
+-----------------+------------+----------+
|name             |commit_count|repo_count|
+-----------------+------------+----------+
|shenzhouzd       |1188430     |1188430   |
|dependabot[bot]  |622184      |622184    |
|Duane F. King    |597940      |597940    |
|Marge Spiderworthy|495786     |495786    |
|Curt Clifton     |384915      |384915    |
+-----------------+------------+----------+
```

Exploratory Data Analysis (EDA) - Contents

- Content Max and Min size

```
+----------+--------+
|  max_size|min_size|
+----------+--------+
|5328264172|       1|
+----------+--------+
```

- Average size order by binary (true/false)

```
+------+------------+------------------+
|binary|binary_count|          avg_size|
+------+------------+------------------+
|  true|    52225197|228811.04712621763|
| false|   228966776|26671.877369793598|
+------+------------+------------------+
```

Exploratory Data Analysis (EDA) - Files

- Files and unique files per repository

repo_name	file_count	unique_file_ids
cdnjs/cdnjs	14239969	14239969
extend1994/cdnjs	5452435	5452435
sufuf3/cdnjs	5452435	5452435
joeyparrish/cdnjs	5440022	5440022
jonobr1/cdnjs	4925081	4925081

- File count per mode

mode	count
33188	2161371558
33261	142534168
40960	5224300
57344	291379
33252	1902
33204	1279
33152	144
33277	94
33279	56
33206	43
33272	10
33216	8
33260	3
33256	1

Exploratory Data Analysis (EDA) - Languages

- Languages & Repot Count/Size Relationships
  - This to some extent, show the popularity of languages measured by repo_count

```
+----------+----------+------------------+----------+---------+
|name      |repo_count|avg_bytes         |max_bytes |min_bytes|
+----------+----------+------------------+----------+---------+
|JavaScript|1099966   |1127059.5834780347|5395746994|0        |
|CSS       |807826    |187000.89174030052|340273471 |0        |
|HTML      |777433    |453047.07289502764|4508833233|1        |
|Shell     |640886    |48444.899197361156|219153905 |0        |
|Python    |550905    |343711.07882665796|715484989 |0        |
+----------+----------+------------------+----------+---------+
only showing top 5 rows
```

Exploratory Data Analysis (EDA) - Licenses

- Total number of licenses

```
+--------------+
|total_licenses|
+--------------+
|       3325634|
+--------------+
```

- Number of licenses associated with agencies

```
+------------+-------+
|license     |count  |
+------------+-------+
|mit         |1696489|
|apache-2.0  |495134 |
|gpl-2.0     |341505 |
|gpl-3.0     |340407 |
|bsd-3-clause|150701 |
+------------+-------+
```

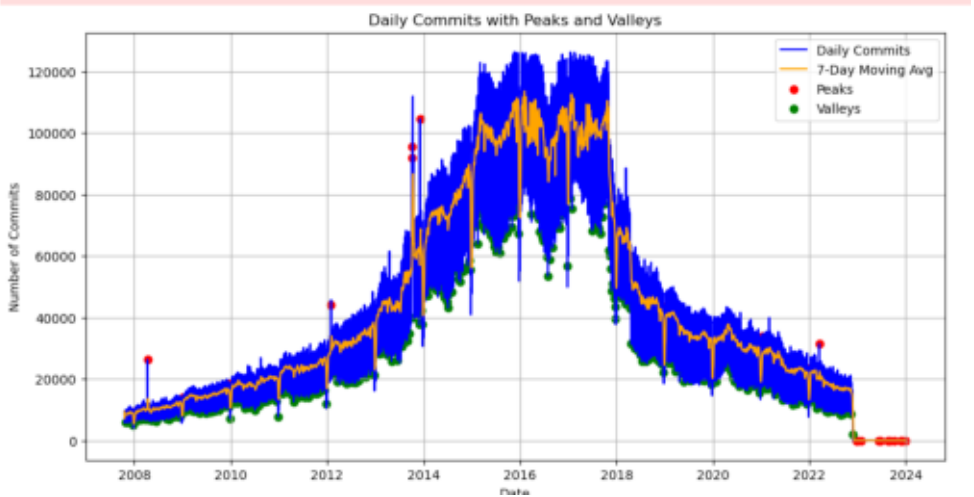
Timeline Analysis - Commit Evolution

Process

1. Convert commit dates to `datetime` format.
2. Compute average daily commit numbers.
3. Exclude outliers (beyond 1.5 IQR).
4. Identify peaks and valleys.
5. Plot timeline for trend visualization.

Findings

- **Peak Activity**: Commit numbers peaked in **2017**.
- **Decline**: Gradual decline in commits since 2017.
- **Seasonality**:
    - Evidence of seasonal changes.
    - **Valleys** observed at the end of each year.



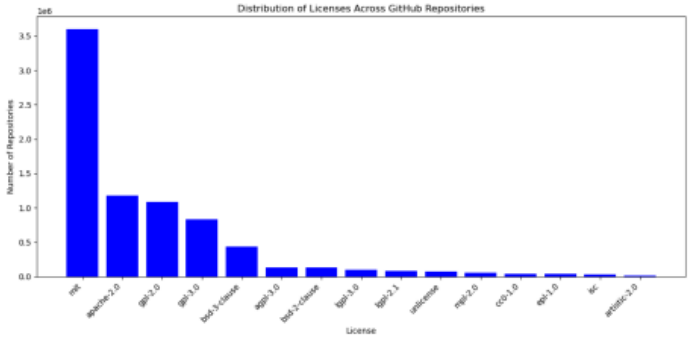
Evolution of Programming Language Popularity

- **C Language**: Historically the most dominant programming language.
- **Current Trends**: Gradual decline in popularity as other languages gain traction.
- **Rising Competitors**:
  - **Python**: Dominates in data science, machine learning, and web development.
  - **JavaScript**: Leads in web development and full-stack applications.

Conclusion: C's dominance is diminishing as modern languages cater to evolving technological demands.

License Analysis

- MIT is Associated with the most license measured by Repo_count



license	language_name	total_bytes
agpl-3.0	JavaScript	15794439063
agpl-3.0	Python	13947621322
agpl-3.0	PHP	13588429815
agpl-3.0	Java	9069228035
agpl-3.0	HTML	7554705783
agpl-3.0	C++	5751133734
agpl-3.0	C	3907708584
agpl-3.0	DM	3622970504
agpl-3.0	CSS	2888001637
agpl-3.0	Ruby	2288620396
agpl-3.0	Perl	1530883080
agpl-3.0	Go	1029848879
agpl-3.0	Jupyter Notebook	847658519
agpl-3.0	C#	672460439
agpl-3.0	CoffeeScript	544906857
agpl-3.0	PLSQL	468656318
agpl-3.0	Lua	413882838
agpl-3.0	Smarty	369882600
agpl-3.0	TypeScript	346205394
agpl-3.0	Gettext Catalog	292231695

- Javascript and Python have significant association with License in total bytes

Most Popular Technology & Repositories Analysis - 1

- Most Popular Repositories by total_bytes

repo_name	total_bytes
aashish24/cdnjs	5580890143
mgoldsborough/cdnjs	5579157705
stdlib-js/www	5554088641
nareshs435/cdnjs	5538002104
dhowe/cdnjs	5521841528

- Fastest Growing repository

repo_name	max_growth
dperezde/little-p...	39574885091040
zhiyisun/linux	30119615920065
HarveyHunt/linux	30116595516210
elp/iwlwifi	29719397850576
sunny256/linux	29694985884945
hannes/linux	29693850495405
djbw/linux	29691488896800
oldzhu/linux	29557106009730
tescande/linux-nf...	29418462793290
Broadcom/cygnus-l...	29417129180265

Most Popular Technology & Repositories Analysis - 2

- Technology keywords : ["pytorch", "transformers", "openai"]

technology	count
pytorch	153
openai	58
transformers	155

- ‘Big Tech’ contribution to AI and repository growth
  - OpenAI contributed Significantly more to other big techs

big_tech_association	count
Unknown	66
Meta	2
Google	2
Microsoft	2
Apple	1
OpenAI	61

- Noticeable Increase in AI related technology started from 2017

year	frequency
2007	1
2008	1
2009	2
2010	4
2011	2
2012	6
2013	8
2014	11
2015	3
2016	4
2017	5
2018	2
2019	1
2020	4
2021	1
2022	3
2017	17
2018	22
2019	23
2020	29

Most Popular Technology & Repositories Analysis - 3

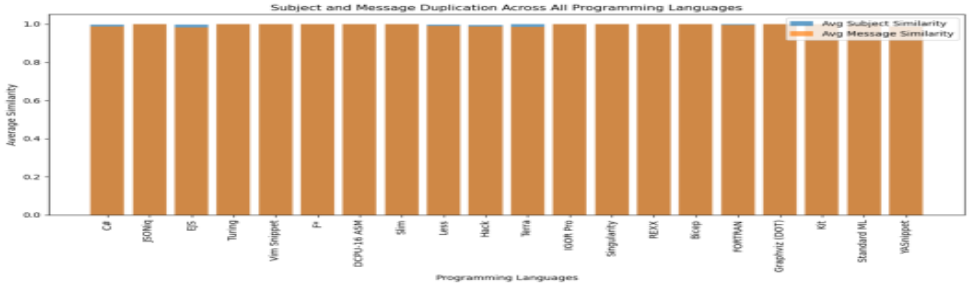
- Most Frequent Reason to commit to a repository
  - Minor modifications (e.g. update, merge) are still the most common

subject	count
Update README.md	5243487
	3681681
Translation update done using Pootle.	2670842
Initial commit	2250000
drew a picture :art:	1619859
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net-2.6	1501475
Merge remote-tracking branch 'origin/master'	1176095
update	1025494
Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/tiwai/sound-2.6	998573
Merge branch 'x86-fixes-for-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/tip/linux-2.6-tip	980628
Merge branch 'master' of master.kernel.org:/pub/scm/linux/kernel/git/davem/net-2.6	882516
merge	840233
Merge branch 'master' of git://git.kernel.org/pub/scm/linux/kernel/git/linville/wireless-2.6	797148
Launchpad automatic translations update.	782685
Merge	762042
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net	742819
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/sparc-2.6	731703
Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/dtor/input	707763
Merge branch 'upstream-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/jgarzik/libata-dev	699254
[maven-release-plugin] prepare for next development iteration	683795

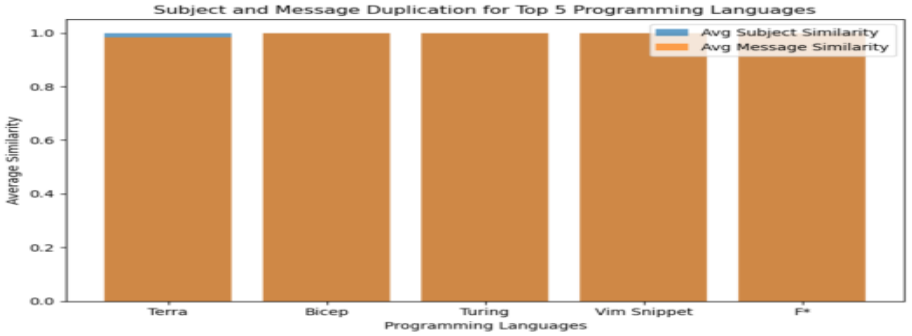
Subject & Message Uniqueness Analysis - 1

- Distinct Rows: Few are distinct

```
Total Rows: 3218419786
Distinct Subjects: 169309119
Distinct Messages: 179875673
Subject Uniqueness Ratio: 5.26%
Message Uniqueness Ratio: 5.59%
```



- Language & Similarity
  - There is little difference between languages



Subject & Message Uniqueness Analysis - 2

Procedure:

- **Data Preparation:** Extracted programming languages with `explode` and filtered non-null repositories and languages.
- **Tokenization & Vectorization:** Tokenized `subject` and `message` fields; transformed tokens into numerical feature vectors using HashingTF.
- **Similarity Calculation:** Applied cosine similarity UDF to compute similarity between `subject` and `message` vectors.
- **Data Integration:** Filtered identical vectors, joined similarity scores with language data, and selected key columns.
- **Aggregation & Analysis:** Grouped by programming language, computed average similarity, and filtered top 5 languages: C, JavaScript, C++, PHP, HTML.
- **Visualization:** Converted results to Pandas, plotted a bar chart of average cosine similarity for top 5 languages.

Result:

- **Insight:** There is significant consistency in commit messages and subjects across programming languages.

Conclusion

AI Tools Improve Productivity but Don't Replace Humans: AI-powered solutions like PyTorch and TensorFlow excel at automating repetitive coding tasks (e.g., bug fixes, minor updates). However, complex problem-solving and innovation still require human judgment and creativity. Commit message analysis shows AI's impact on improving efficiency.

Programming Language Trends Reflect Industry Shifts: Python dominates AI and data science, supported by Big Tech (e.g., OpenAI, Microsoft) and community-driven libraries like PyTorch. Traditional languages like C are declining, while JavaScript and Python gain traction due to their adaptability in modern applications.

Commit Activity Reveals Key Technological Events: Commit peaks in 2017 align with AI advancements, while seasonal trends (e.g., year-end declines) suggest activity patterns influenced by organizational cycles.

Big Tech Drives AI Repository Growth: OpenAI and Microsoft play pivotal roles in accelerating adoption of cutting-edge AI technologies, contributing significantly to the most rapidly growing repositories.

Big Data Challenges in Analysis: Processing 1.36 TiB of nested GitHub data required extensive filtering, standardization, and optimization to handle noise and enable scalable insights, highlighting the complexity of analyzing open-source ecosystems.

Recommendations - 1

Leverage AI Tools for Repetitive Coding Tasks:

- Since the analysis shows a significant portion of commits relate to minor modifications (e.g., updates and bug fixes), organizations should utilize AI-powered tools like GitHub CoPilot and Amazon Code Whisperer to automate these tasks, improving efficiency and reducing manual effort.

Focus on Python and JavaScript for AI and Web Development:

- The analysis highlights Python's dominance in AI and data science projects and JavaScript's leadership in web development. Developers should prioritize adopting and enhancing their skills in these languages, which are associated with cutting-edge technologies and rapid repository growth.

Collaborate with Big Tech Open-Source Initiatives:

- Big Tech companies like OpenAI and Microsoft have significantly contributed to AI-related repositories, driving repository growth and adoption of technologies such as PyTorch and Transformers. Organizations should collaborate with or adopt tools from these open-source initiatives to stay aligned with industry-leading practices.

Monitor and Incorporate Emerging AI Technologies:

- Technologies like PyTorch, TensorFlow, and Transformer models have shown explosive growth in repositories since 2017. Companies should continuously monitor and integrate these emerging technologies into their projects to remain competitive in AI-driven applications.

Recommendations - 2

Enhance Productivity with Data-Driven Insights:

- Commit activity trends, such as the peak in 2017 and seasonal valleys at the end of each year, suggest opportunities to optimize development cycles. Organizations can schedule major releases or projects during peak activity periods and focus on AI-driven tools during slower cycles.

Streamline License Management:

- The MIT license is the most common across repositories, especially for Python and JavaScript projects. Organizations should consider using licenses like MIT for simplicity and alignment with community norms when releasing open-source projects.

Improve Commit Message and Subject Clarity:

- Consistency in commit messages and subjects across programming languages, as shown by cosine similarity analysis, indicates room for improvement in message clarity and uniqueness. Teams should adopt clear naming conventions and utilize AI tools to generate meaningful commit messages.

Utilize Distributed Systems for Data Management:

- The extensive size (~1.36 TiB) and complexity of GitHub data required significant cleaning and processing. Organizations managing large-scale repositories should use distributed computing frameworks like PySpark on GCP Dataproc for efficient data handling and analysis.