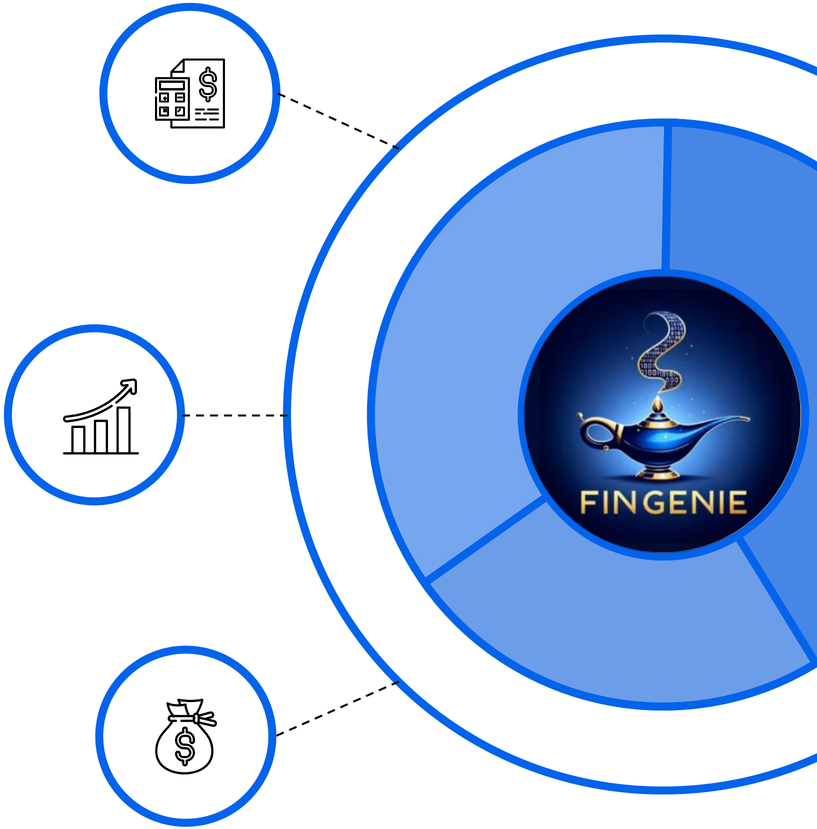
PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

FIN GENIE: A Gen AI-Powered Financial Reports Tool

Simplifying report summaries to facilitate informed financial decisions



Mitchell E. Daniels, Jr. School of Business

Our Team

Soham Agarwal
Purdue MS-BAIM

Durga Dash
Purdue MS-BAIM

Anto Frederic
Purdue MS-BAIM

Chaitanya Varma
Purdue MS-BAIM

Bheeshma Ramaraju
Purdue MS-BAIM

"The essence of a triumphant tale lies not in its plot, but in the hearts and minds of those who bring it to life"

Agenda

FINGENIE

01
Business problem and benefits
Why does it really matter?

02
Approach
“80% of the work comes from the approach (the 20%)”
Durga

03
Phase 1 and Phase 2
Diving deeper into the sprints and phases

04
Product Demo
The fun part!

05
Results and Learnings
Our success stories and setbacks-turned-lessons

06
Future Scope
A lot of ideas we would love to implement

Business Problem

The Issue

Financial investors need additional information to make decisions. Thus additional sources

The deeper issue

SEC filings provide detailed audited reports. Reading 100 page documents isn't easy!

Existing processes...

Are Time intensive

Sorting through SEC data for additional information required considerable time and resources to gain insights.

Have High Opportunity costs

The opportunity cost is high in the fast-paced finance industry, making traditional analysis methods inefficient.

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

FINGENIE

Business Benefits

Expected hours saved

At least 50% increase in efficiency

Existing Methods		Our Method
Days	→	Hours

Increased efficiency

Automation reduces time spent on report generation, to give more time for investors to analyze information.

Cost Reduction

Less manual analysis means more hours saved and better investment decisions.

Scalability

Ability to handle increased data volumes efficiently as the number of documents grows.

User engagement

Interactive AI assistance enhances user experience and allows two-way engagement with the financial data.

The Timeline

Dates: January 12th 2024 – April 2nd 2024

January

WIREFRAMING

Designing the process flow

February

PHASE 1

Experimenting with existing tools

Feb + March

PHASE 2

Creating our own mechanisms using code

March

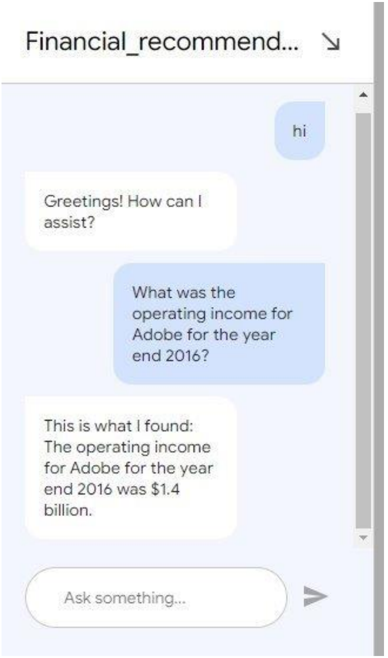
BENCHMARKING

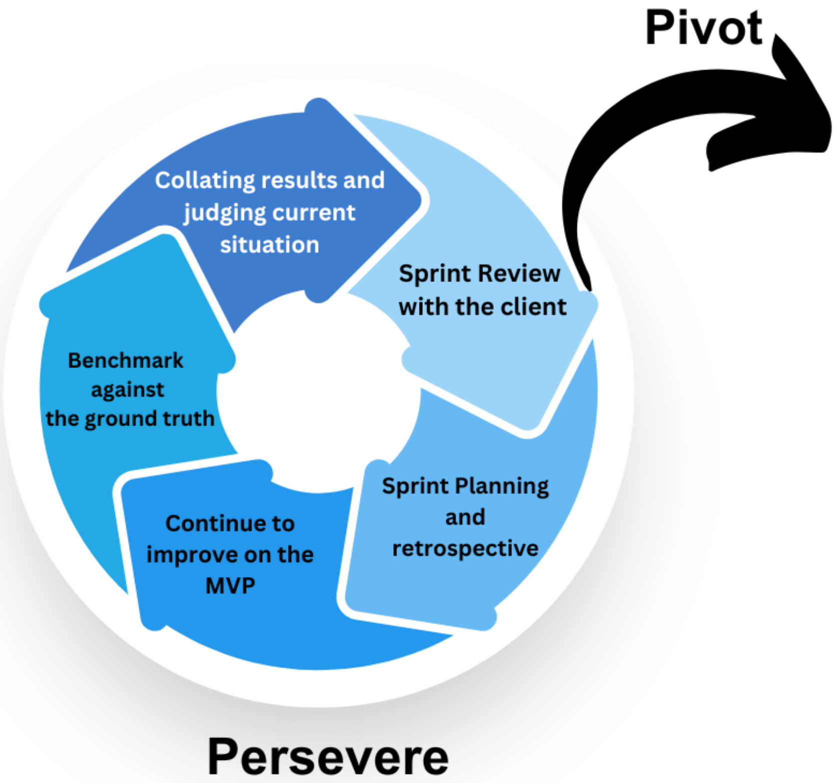
Benchmarking the results to assess accuracy

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Agile Approach

1) Older model





2) Newer model

Phase 1 Implementations

Step One -

Scraping 10k documents from the SEC website.

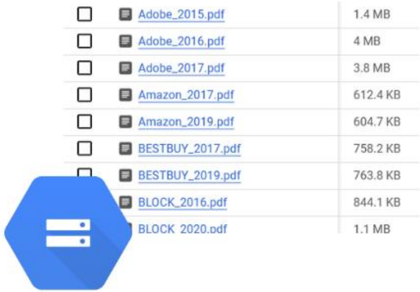

UNITED STATES
SECURITIES AND EXCHANGE COMMISSION
Washington, D.C. 20549
FORM 10-K
☒ ANNUAL REPORT PURSUANT TO SECTION 13 OR 15(d) OF THE SECURITIES EXCHANGE ACT OF 1934
For the fiscal year ended December 31, 2008
OR
☐ TRANSITION REPORT PURSUANT TO SECTION 13 OR 15(d) OF THE SECURITIES EXCHANGE ACT OF 1934
For the transition period from to
Commission File No. 1-2217
The Coca-Cola Company
(Exact name of Registrant as specified in its charter)
DELAWARE (State or other jurisdiction of incorporation or organization)
58-0628465 (IRS Employer Identification No.)
One Coca-Cola Plaza Atlanta, Georgia (Address of principal executive offices)
30313 (Zip Code)
Registrant's telephone number, including area code: (404) 676-2121
Securities registered pursuant to Section 12(b) of the Act:
Title of each class: COMMON STOCK, $0.25 PAR VALUE
Name of each exchange on which registered: NEW YORK STOCK EXCHANGE

Step Three -

Using the large language models on GCP via the Search and Conversation, and Dialogflow plaform.. Powered by Vertex AI.

Step Two -

Storing the documents in the cloud to ensure scalability.



Step Four -

Designing a front end chatbot and integrating this with our hyperparameter tuned LLMs

Wireframed Pipeline – Phase 1

Data Extraction

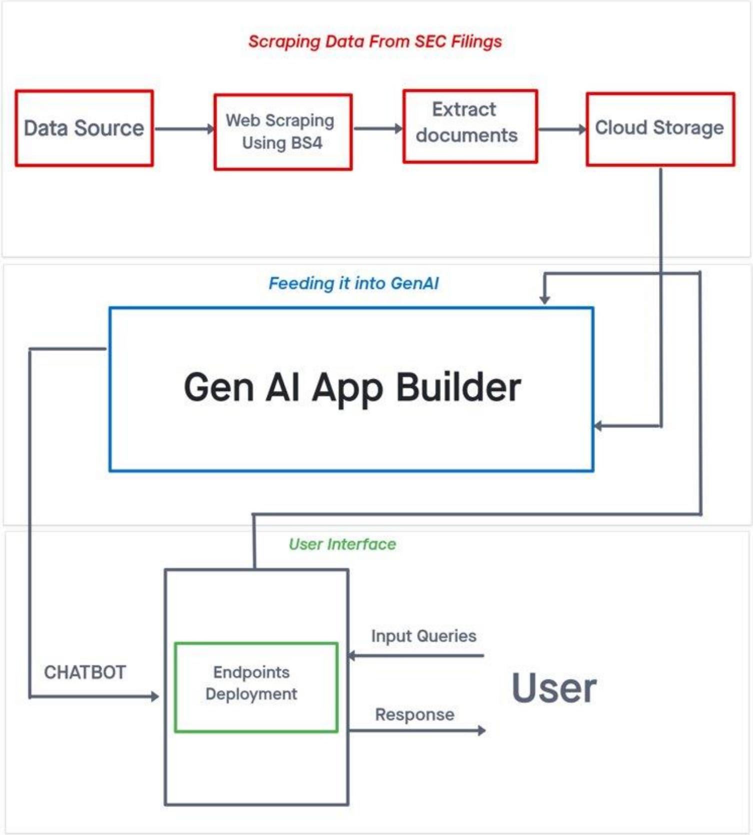
A combination of multiple extraction techniques to get accurate data.

Data Interpretation

Google's cloud service "Search and Conversation" allows for an 'out-of-box' solution.

Data Delivery

Allows the user to have conversations with the data.



Phase 2 Implementations

Objective

- More control over responses generated.
- Ability to tailor responses to the financial domain
- Make a solution that shows an improvement over the Search & Conversation cloud service.

Additional Step 1 -

- Preprocessing SEC filings in batches with techniques like tokenization and lemmatization and chunking for context.
- vector stores are created using these chunks for efficient retrieval.

*Web Scraping and cloud storage methods were the same as in phase 1.

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Phase 2 Implementations

FINGENIE

Additional Step 2 -

- API's are called within a code environment. Prompting techniques, Query rephrasing, Chunk size strategies, few-shot prompting and many other techniques were tried to give us superior results.
- Models used: 1) Gecko embeddings + TextBison LLM model 2) Text Bison model.

Additional Step 3 -

- The top matching chunks are extracted to provide context in the prompt. The techniques applied above help for a more accurate retrieval of data.
- LLM generates the financial information and sends it back to the investor.

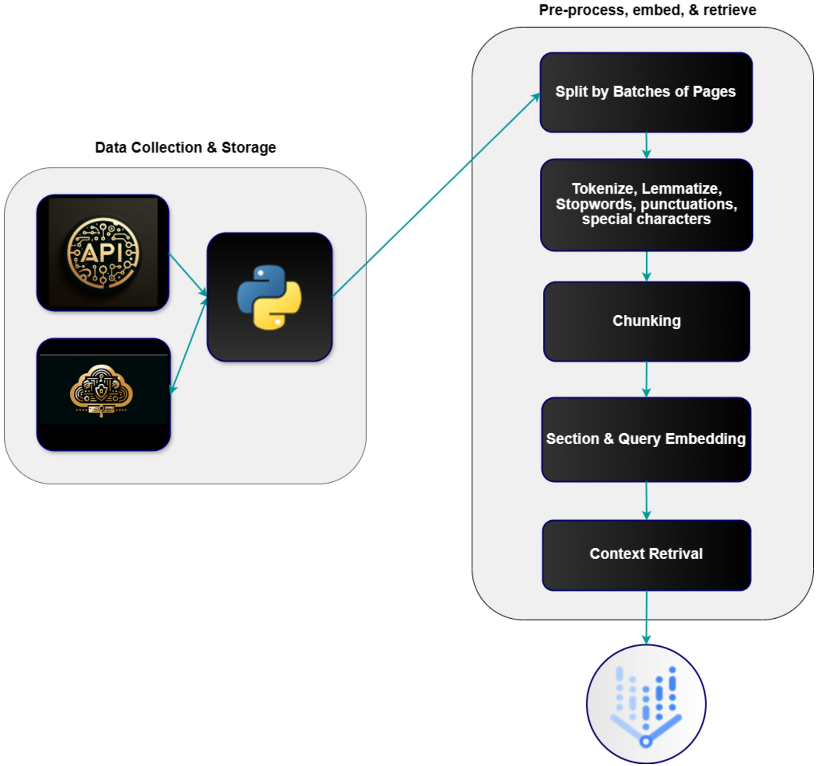
Wireframed Pipeline – Phase 2

Preprocessing

Building a framework to allow model to fetch relevant information.

Chunking, embedding and retrieval

Helping the LLM understand how the document is structured.



PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Wireframed Pipeline – Phase 2

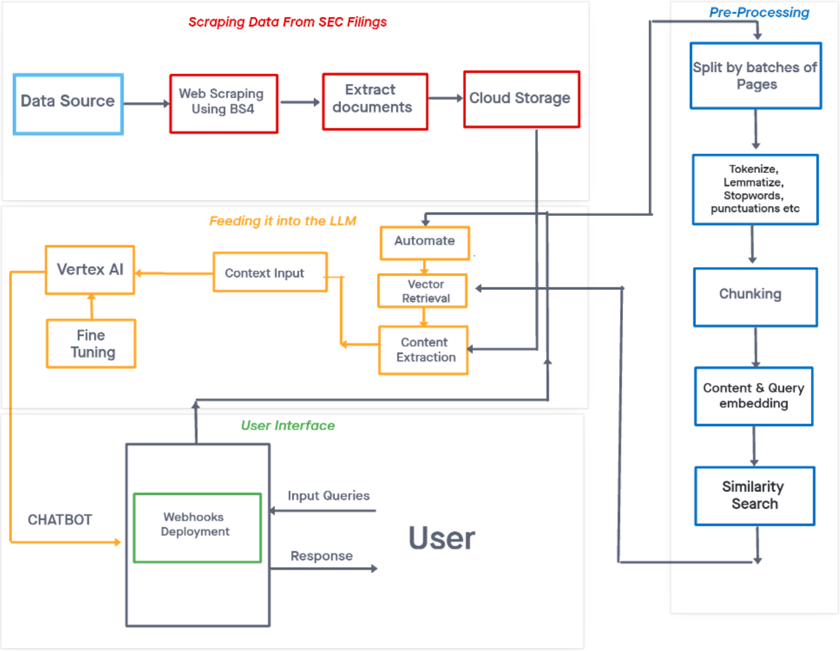
FINGENIE

Cloud first

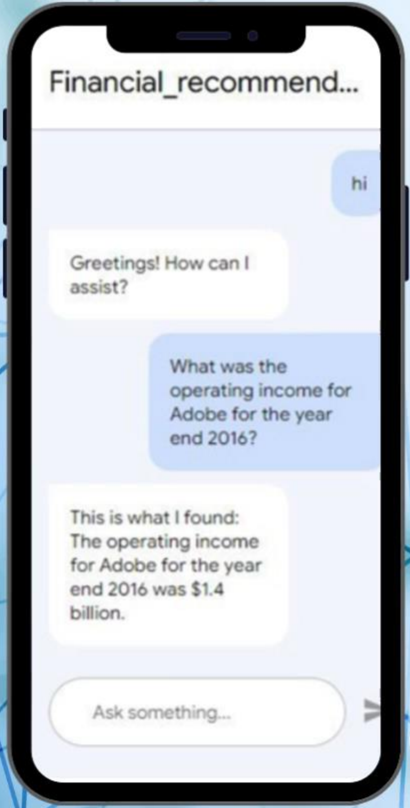
Using Google Cloud Platform (GCP) to stitch together technical components.

Feeding the LLM and hyperparameter tuning

Using the latest practices in GenAI development to enhance responses delivered in the finance domain.

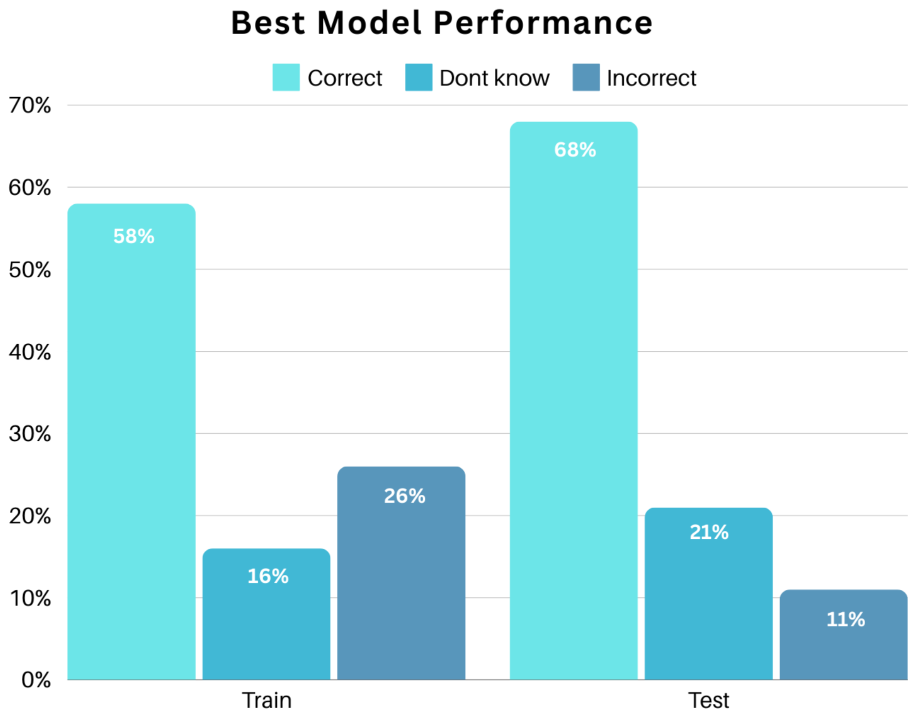


Product Demo



PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Results

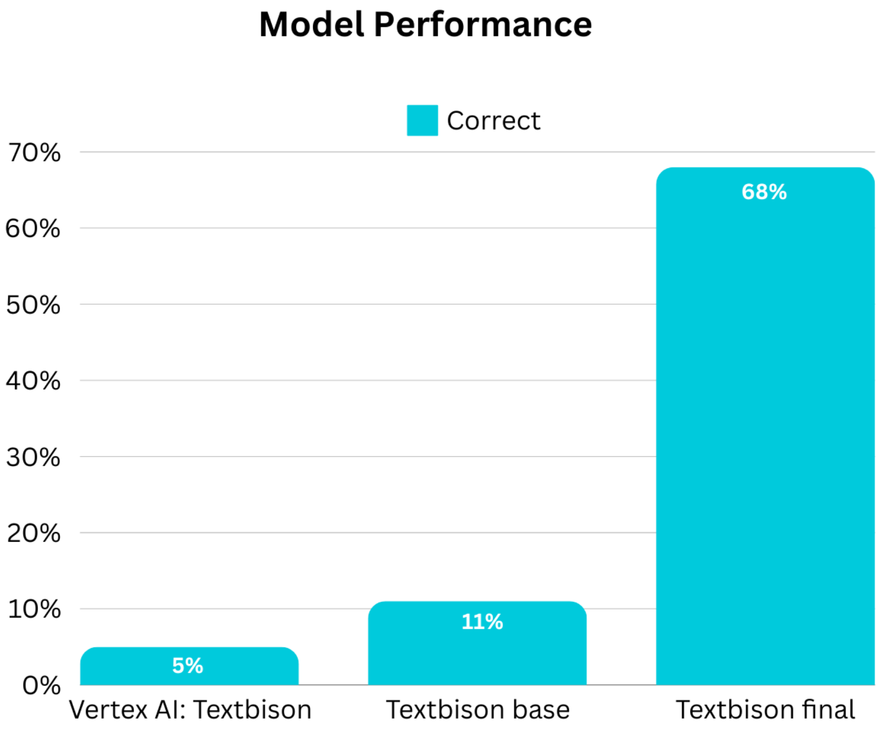


Test vs Train

- Our model demonstrated comparable performance across both training and testing phases
- Noticeable decrease in the percentage of incorrect outcomes alongside these improvements
- Indicates a robust and reliable performance

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Results



Model Performances

- The transition from Vertex AI to the TextBison base model resulted in a **2X** performance boost
- Followed by a significant **6X** enhancement when moving from the TextBison base to the TextBison final model

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

FINGENIE

Learnings

What worked?	What didn't work?
• Query Rephrasing. • Page Batches/ Chunk Size. • Data cleaning. • Increased context window. • One Shot prompting.	• Overlap of content in chunking. • Tokenizing and increasing prompt similarity. • Using the ReAct framework. • Fine-tuning the model on example questions. • Cascading model Bert+Bison did not yield significant change.

Additional Learnings

Importance of model selection

Developed a model leveraging BART Large CNN to condense 10K documents into structured summaries by sections

Balancing Brevity and Detail

Fine-tuning summary parameters underscores the importance of striking a balance between brevity and detail

Document Summarization

```
Summary of Section 2: We earn tradable credits in the operation of our business under various regulations rela
ission vehicles (“ZEVs”),  greenhouse gas, fuel economy and clean fuel. We sell these credits to other regulat
o can use the credits to comply with  regulatory requirements. Sales of these credits are recognized within au
atory credits in our consolidated financial statements included elsewhere in this Annual Report on Form 10-K.
==================================================
Summary of Section 3: We may be unable to meet our projected construction timelines, costs and production ramp
ries, or we may experience difficulties in generating and maintaining demand for products manufactured there.
ll depend on our ability to continue to expand our sales capabilities. If we experience production delays or i
recast demand, our business, financial condition and operating results may be harmed.
==================================================
Summary of Section 4: Tesla's future growth and success are dependent upon consumers’ demand for electric vehi
fically our vehicles. The prices and availability of raw materials such as lithium, nickel, cobalt and/or othe
e unstable, depending on market conditions and global demand.
==================================================
Summary of Section 5: The target demographics for our vehicles, particularly Model 3 and Model Y, are highly c
les of vehicles tend to be cyclical in many markets, which may expose us to further volatility. Increased comp
result in lower vehicle unit sales, price reductions, revenue shortfalls, loss of customers and loss of market
shandling of these products may cause disruption to the operation of our facilities.
==================================================
Summary of Section 6: As of December 31, 2022, we and our subsidiaries had outstanding $2.06 billion in aggreg
amount of debt. Our ability to make scheduled payments of the principal and interest on our indebtedness when
our future performance, which is subject to economic, financial, competitive and other factors beyond our cont
==================================================
Summary of Section 7: We are exposed to fluctuations in currency exchange rates. We may need to defend ourselv
ellectual property infringement claims, which may be time-consuming and expensive. There can be no assurance t
able to adequately identify and protect the portions of intellectual property that are strategic to our busine
e potential suits or other legal demands by our competitors.
==================================================
```

Additional Learnings

BERT Model...

PDF Embedding Processing

Read documents in batches, extract text per section, preprocess and generate semantic embeddings.

Vector Store Creation

A store (dataframe) of texts and their embeddings is created to facilitate similarity searches.

Context Retrieval

For a given question, the most relevant document sections are identified using embedding similarities.

Question Answering

With the context narrowed down, the question is answered using the fine-tuned BERT QA model

What new was discovered with this approach is that the context was retrieved very well without much pre-processing using the BERT model, but the model has troubled response generation which needs to be improved in the future scope of work. As part of this, research on how to tune context window, domain specific fine tuning, hugging face pipelines are being explored.

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Future Scope

FINGENIE

- A combination of models to accurately identify tabular data and avoid misinterpretation.
- Use pre-defined examples for model fine-tuning to enhance accuracy.
- Create a benchmarking framework for comprehensive financial data analysis
- Pair LLMs with mathematical libraries to improve accuracy in answering quantitative questions.
- Develop an embedding model on extensive financial data to enhance context comprehension and information extraction.

Thank you for your support!

Ashwin Mishra

Prof. Matthew Lanham

Purdue

Purdue University Mitchell E. Daniels, Jr. School of Business

FINGENIE

Thank You!

We will now take any questions.

Appendix

Appendix 1

UNITED STATES
SECURITIES AND EXCHANGE COMMISSION
Washington, D.C. 20549

FORM 10-K

☒ ANNUAL REPORT PURSUANT TO SECTION 13 OR 15(d) OF THE SECURITIES EXCHANGE ACT OF 1934

For the Fiscal Year Ended June 30, 2022

OR

☐ TRANSITION REPORT PURSUANT TO SECTION 13 OR 15(d) OF THE SECURITIES EXCHANGE ACT OF 1934

For the Transition Period From to

Commission File Number 001-37845

MICROSOFT CORPORATION

WASHINGTON (STATE OF INCORPORATION)	91-1144442 (I.R.S. ID)

ONE MICROSOFT WAY, REDMOND, WASHINGTON 98052-6399

(425) 882-8080

www.microsoft.com/investor

Securities registered pursuant to Section 12(b) of the Act:

Title of each class	Trading Symbol	Name of exchange on which registered
Common stock, $0.00000625 par value per share	MSFT	NASDAQ
3.125%Notes due 2028	MSFT	NASDAQ
2.625%Notes due 2033	MSFT	NASDAQ

Securities registered pursuant to Section 12(g) of the Act:

PURDUE UNIVERSITY | Mitchell E. Daniels, Jr. School of Business

Appendix 2

Appendix 3

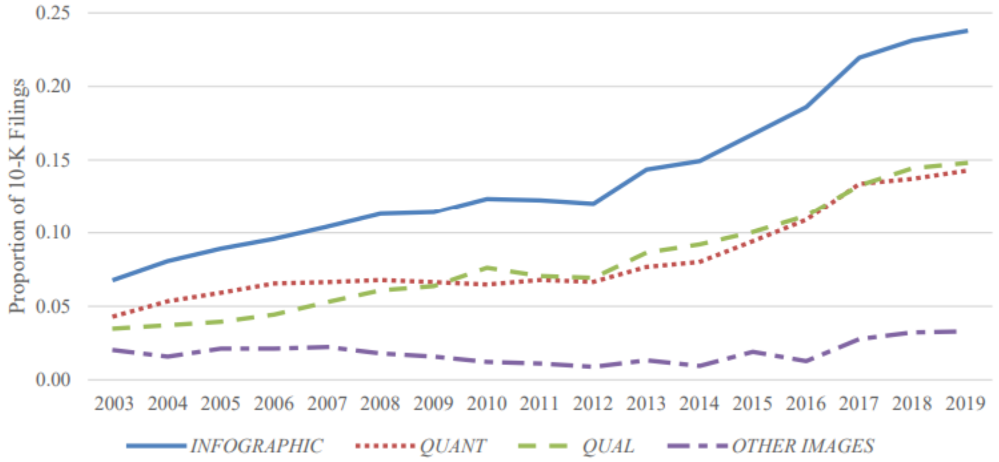


Fig. 3. Trend over time in the proportion of 10-K filings containing at least one infographic or other image. The sample consists of 47,906 10-K filings from 2003-2019. We define all variables in Appendix C and provide additional information on infographic types in Appendix A. The full-color version is available online.

Appendix 4

Gantt View · Filter · Arial · 10 · Baselines

	Task	Description	Assigned To	Status	Start Date	End Date
	for embedding (Tokenization, Removing stop words and punctuations, Lowercasing)					
7	Draft the steps required in embedding		AF Anto Frederic Henr	Complete	01/23/24	01/25/24
8	Work on Improving ideation documentation		SA Soham Agarwal			
9	Understanding 10K/10Q Documents		SB Sai Bheeshma Rar	Complete	01/23/24	01/30/24
10	Literature Review		SB Sai Bheeshma Rar	Complete	02/13/24	
11	GCP Interface		SB Sai Bheeshma Rar	Complete	01/22/24	01/31/24
12	Gen AI Builder Research		SB Sai Bheeshma Rar	Complete	02/09/24	
13	Taking KT for embeddings		CS Chaitanya Sanaboi	Complete	02/22/24	02/23/24
14	Continue the research and debugging of existing flow		CS Chaitanya Sanaboi	In Progress	02/24/24	02/29/24
15	**Web Scraping**					
16	Revisit web data analytics SEC filing task to understand and scrape data.		SA Soham Agarwal	Complete	01/17/24	04/02/24
17	Explore XBRL & SEC APIs on how to organize data for aggregation		DM Durga Madhab Das	Complete	01/17/24	01/23/24

Jan 14: S M T W T F S S M

Appendix 5

```
embeddings_model = TextEmbeddingModel.from_pretrained("textembedding-gecko@001")
file_text_model = TextGenerationModel.from_pretrained("text-bison@001")
text_model = TextGenerationModel.from_pretrained("text-bison-32k")
rephrase_model = TextGenerationModel.from_pretrained("text-unicorn@001")

chat_model = ChatModel.from_pretrained("chat-bison-32k")
chat = chat_model.start_chat(
          context="You are an expert financial analyst. You know how to read SEC filings such as 10-K
)

gemini_pro_model = GenerativeModel("gemini-1.0-pro")
```

Models Tested & Used

Appendix 6

```
rephrase_prompt=f"""Act as a financial analyst with expert knowledge of SEC 10-K documents.
Could you rephrase this question so that a financial novice can easily understand - {{USER_QUERY}}
USER_QUERY: ```{question}```\n
Please make sure to re-phrase the query such that it is 100% same in meaning to the original query
"""


user_query = rephrase_model.predict(rephrase_prompt, temperature=0, top_k=1, max_output_tokens=30
user_query = user_query.replace('\n', ' ')
user_query = add_fy_to_years(user_query)

lemmatized_query = []

for word in word_tokenize(user_query):
  lemmatized_query.append(lemmatizer.lemmatize(word))

user_query_proc = " ".join([word.lower() for word in lemmatized_query if word not in string.punct
```

Pre-Processing (User-Query)

Appendix 7 Pre-Processing (PDF Content)

```python
section_content = read_pages_in_batches(file_path_pdf, 6)
```

```python
def read_pages_in_batches(file_path_pdf, pages_per_batch=2):
    section_content = []

    with open(file_path_pdf, 'rb') as pdf_file:
        pdf_reader = PyPDF2.PdfReader(pdf_file)
        total_pages = len(pdf_reader.pages)
        #doc = fitz.open(pdf_file)
        #total_pages = len(doc)

        for page_num in range(0, total_pages, pages_per_batch):
            text = ""
            for i in range(pages_per_batch):
                if page_num + i < total_pages:
                    text += pdf_reader.pages[page_num + i].extract_text()
                    #text += doc.load_page(page_num + i).get_text()

            section_content.append(text)

    return section_content
```

Appendix 8

```
# Tokenize and lemmatize the content in section_content
lemmatized_content = []
for section in section_content:
    lemmatized_section = []
    for word in word_tokenize(section):
        lemmatized_section.append(lemmatizer.lemmatize(word))
    lemmatized_content.append(" ".join(lemmatized_section))

preprocessed_sections = []

for content in lemmatized_content:
  proc_sections = [ word for word in content.lower().split() if word not in string.punctuatio
  preprocessed_sections.append(' '.join(proc_sections))
```

Appendix 9

Create Vector Store

```
text_lengths = [len(t) for t in preprocessed_sections]
size_overlap = int(np.median(text_lengths)//2)

CHUNK_SIZE = size_overlap
OVERLAP = 500
preprocessed_sections = preprocessed_sections[:len(preprocessed_sections)-1]

vector_store = create_vector_store(preprocessed_sections, CHUNK_SIZE, OVERLAP)
vector_store.to_pickle(file_path_pkl)
```

```
def create_vector_store(texts, chunk_size, overlap):
    vector_store = pd.DataFrame()

    # Split texts and filter out empty ones during splitting
    non_empty_texts = [text for text in itertools.chain.from_iterable(
        split_overlap(t, chunk_size, overlap) for t in texts if t.strip())]

    vector_store["texts"] = non_empty_texts

    # Create embeddings from those texts
    vector_store["embeddings"] = (
        vector_store["texts"].progress_apply(get_embeddings).apply(np.array)
    )

    return vector_store
```

Appendix 10

```python
# Separates seq into multiple chunks in the specified size with the specified overlap
def split_overlap(seq, size, overlap):
    if len(seq) <= size:
        return [seq]
    return ["".join(x) for x in zip(*[seq[i :: size - overlap] for i in range(size)])]
```

```python
@retry.Retry(timeout=300.0)
def get_embeddings(text):
    return embeddings_model.get_embeddings([text])[0].values
```

Appendix 11 Retrieving Context

```
def answer_question(user_query_proc, question, vector_store, num_docs=12, print_prompt=False):
    context = get_context(user_query_proc, vector_store, num_docs)
```

```
def get_context(question, vector_store, num_docs):
    # Tokenize the user query
    #query_tokens = word_tokenize(question)

    # Embed the search query
    #query_vector = np.array(get_embeddings("".join(query_tokens)))

    query_vector = np.array(get_embeddings(question))

    # Get similarity to all other vectors and sort, cut off at num_docs
    top_matched = (
        vector_store["embeddings"]
        .apply(get_similarity_fn(query_vector))
        .sort_values(ascending=False)[:num_docs]
        .index
    )
    top_matched_df = vector_store[vector_store.index.isin(top_matched)][["texts"]]

    # Return a string with the top matches
    context = " ".join(top_matched_df.texts.values)
    #context = top_matched_df.texts.values
    return context
```

Appendix 12

```
# Compute the cosine similarity of two vectors, wrap as returned function to make easier to use with
def get_similarity_fn(query_vector):
    def fn(row):
        return np.dot(row, query_vector) / (
            numpy.linalg.norm(row) * numpy.linalg.norm(query_vector)
        )

    return fn
```

Appendix 13 One-Shot Prompting for result

```
def answer_question(user_query_proc, question, vector_store, num_docs=12, print_prompt=False):
    context = get_context(user_query_proc, vector_store, num_docs)

    while len(context) < 100000:
      num_docs += 1
      context = get_context(user_query_proc, vector_store, num_docs)

    print(len(context))

    train_context = "Table of Contents ACTIVISION BLIZZARD, INC. AND SUBSIDIARIES CONSOLIDATED S
    train_question = "What is the FY2017 - FY2019 3 year average of capex as a % of revenue for
    train_answer = "To calculate the three-year average of capital expenditures (capex) as a per

    qa_prompt = f"""Input Example: Your mission is to answer questions based on a given context
    Context: ```{train_context}```
    Question: ***{train_question}***
    Before you give an answer, make sure it is only from information in the context. If the info
    Reply concisely.
    Answer: ```{train_answer}```\n
    Your mission is to answer questions based on a given context. Remember that before you give
    Context: ```{context}```
    Question: ***{question}***
    Before you give an answer, make sure it is only from information in the context. If the info
    Reply concisely.
    Answer: """

    if print_prompt:
        print(qa_prompt)
    result = text_model.predict(qa_prompt, temperature=0, top_k=3, max_output_tokens=400)
```