International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 10-No.4 (2024) pp. 1361-1371
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Integrating Self-Attention Mechanisms For Contextually Relevant Information In Product Management

Pavan GUNDA[1]*, Thirupathi Rao KOMATI[2]

[1]Department of Computer Science and Engineering, GITAM School of Technology, GITAM, Deemed-to-be University, Visakhapatnam-530045, AP, India
* **Corresponding Author Email:** gsraghupavan9@gmail.com - **ORCID:** 0009-0008-6163-6789

[2] Department of Computer Science and Engineering, GITAM School of Technology, GITAM, Deemed-to-be University, Visakhapatnam-530045, AP, India
Email: tkomati@gitam.edu- **ORCID:** 0000-0002-0907-6824

Article Info:

DOI: 10.22399/ijcesen.651
Received : 19 November 2024
Accepted : 23 November 2024

Keywords:

Product Management,
NLP,
Transformer Architecture,
Translation Services,
Product Quality Enhancement.

Abstract:

GPT-Product is an innovative AI solution that aims to transform product management and development by using sophisticated natural language processing (NLP) abilities. Building on Transformer architecture, frameworks like as BERT, GPT, and T5 have greatly enhanced AI applications, thereby allowing more efficient chatbots, and translation services, content generating tools, and so on. GPT-Product utilises the advanced GPT-3.5 architecture to provide full solutions for market evaluation, interpretation of client input, and automated content development. This enhances decision-making processes. This product utilises the self-attention mechanism of the Transformer model to provide precise and contextually appropriate information, enabling effective management of the product lifetime. GPT-Product uses deep learning to optimise processes, decrease time-to-market, and enhance product quality. It positions itself as an essential tool for firms striving to maintain competitiveness in a rapidly changing industry.

1. Introduction

Due to the sudden change of the customers while interacting with the E-commerce platforms, the proposed model has designed a product GPT to analyze customer behavior, preferences, and past interactions to suggest personalized product recommendations, features, or configurations tailored to individual users or target audiences. In the existing approach, the model is designed using naïve Bayesian approach to describe the products based on the input prompt. In this approach, it treats the product descriptions as a set of text documents. It preprocesses the text using techniques like tokenization, stop-word removal, and TF-IDF. It uses the Multinomial Naive Bayes to classify the product descriptions based on the frequency of words. In general, all the traditional approaches assume that the description of the vectors are independent on the features associated with the product. These models also suffer from analyzing the customer behaviour patterns and understanding the context and dependencies between the words of the input prompt. So, to solve this problem the latest LLM uses the self attention mechanisms in the transformers while generating the product description.

1.1. Multi Head Self Attention: By utilizing the self-attention mechanism, the model is able to identify the relative significance of words in a sequence. For a set of queries Q, keys K, and values V:Attention(Q, K, V) = softmax ($QK^T d_k$) V
Where:

- $Q \in R^{nxd_k}$
- $V \in R^{nxd_k}$
- $K \in R^{nxd_k}$
- d_k is the dimension of the key queries.

Multi-head attention allows the model to attend to data from many representation subspaces at different locations at the same time:
MultiHead(Q, K, V) = Concat ($head_1$, …, $head_h$) W^O
Where each head is computed as:

$Head_i = Attention(QW_i^Q , KW_i^K , VW_i^V)$

Example Paragraph:

"Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black). Category. Televisions/Smart Televisions. Price: ₹8,999.

Link: https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"

Step-by-Step Calculation

Tokenization and Embedding

First, the input paragraph needs to be tokenized and embedded into vectors. For simplicity, let's assume each token (word) in the sentence is represented by a vector. Model have:

- Query matrix (Q)
- Key matrix (K)
- Value matrix (V)

Assume these matrices are derived from the embedding of the tokens. Let's define simplified matrices for illustration:

- **Q (Query Matrix)**: Q=[0.20.50.10.40.60.2]
- **K (Key Matrix)**: K=[0.30.40.10.50.50.3]
- **V (Value Matrix)**: V=[0.20.10.40.3]**2. Compute Dot Product (QKT)**

Calculate the dot product of Q and K^T:

$$QK^T = \begin{bmatrix} (0.2*0.3 + 0.5*0.4 + 0.1*0.1) & (0.2*0.5 + 0.5*0.5 + 0.1*0.3) \\ (0.4*0.3 + 0.6*0.4 + 0.2*0.1) & (0.4*0.5 + 0.6*0.5 + 0.2*0.3) \end{bmatrix}$$

$$QK^T = \begin{bmatrix} 0.20 + 0.20 + 0.01 & 0.10 + 0.25 + 0.03 \\ 0.12 + 0.24 + 0.02 & 0.20 + 0.30 + 0.06 \end{bmatrix}$$

$$QK^T = \begin{bmatrix} 0.41 & 0.38 \\ 0.38 & 0.56 \end{bmatrix}$$

Table 1. *Existing Approaches Comparative Analysis*

Author	Algorithm	Merits	Demerits	Accuracy
[1]	PLM	This process undergoes in NL form.	Only works on single cultural backgrounds for validation.	
[2]	LI-RADS	While extracting the features the accuracy has increased.	Process was complex for real-data.	85%
[3]	LI-RADS	Process of extraction is accurate and efficient.	Only worked on news.	
[4]	LLM	Overcome with stability issues.	Time-complexity.	89%

Compute the Weighted Sum of the Values

Finally, compute the weighted sum of the value vectors:

Output=Attention Scores×V

$$\text{Output} = \begin{bmatrix} 0.41 & 0.38 \\ 0.38 & 0.56 \end{bmatrix} \text{x} \begin{bmatrix} 0.2 & 0.1 \\ 0.4 & 0.3 \end{bmatrix} = \begin{bmatrix} 0.314 & 0.244 \\ 0.340 & 0.292 \end{bmatrix}$$

The final output matrix represents the contextually enhanced representations of the input tokens, which captures the relationships and dependencies between them using the self-attention mechanism.

A. Standard Transformation

Designed for sequence-to-sequence applications like machine translation, the standard Transformer neural network architecture was introduced in 2017. The encoder is responsible for handling the input sequence, while the decoder is responsible for creating the output sequence based on its encoder-decoder architecture [1-10]. The self-attention mechanism of the Transformer is the main novelty as it lets the model concurrently evaluate the relevance of many words in a sentence instead of consecutively. Faster training and better management of long-range dependencies made possible by this parallel processing capacity surpass that of conventional models such as RNNs. Every encoder and decoder layer comprises position-wise feed-forward networks and multi-head self-attention systems. Positional encoding is employed to keep the sequence's order since the model does not grasp it intrinsically. Flexible and efficient, the architecture has established new NLP standards, enabling BERT and GPT. The Transformer's capacity to handle enormous amounts of data and grasp complicated text linkages has made it a key AI tool.

B. BERT

Google released BERT, which stands for "Bidirectional Encoder Representations from Transformers," in 2018 as a pre-trained language model. Traditional models only handle text in one way, but BERT gets information from both sides at the same time, which makes it bidirectional.

By looking at the words that come before and after a word, this helps BERT understand its full meaning, which makes it much better at many NLP jobs[11].

C. GPT

Transformer is at the heart of GPT, and self-attention processes are used to find word relationships [12,13]. Later versions, such as GPT-2 and GPT-3, are much bigger and can do a lot more. For example, GPT-3 has 175 billion factors. This scale lets it write very complex text and do many NLP tasks without needing training just for those tasks. GPT is a key piece of AI because it can create natural-sounding text that fits the situation. It has led to improvements in chats, content creation, and automatic text generation.

D. PoBERTa

PoBERTa, which stands for "Portuguese RoBERTa," is a version of the RoBERTa model that works with the Portuguese language. RoBERTa, which builds on BERT, improves BERT's pre-training by using more data, longer training times, and changes like getting rid of the next sentence forecast goal. This gets these benefits, but it has only been taught on big Portuguese text samples. This specialisation helps to do better on many NLP challenges in Portuguese, like mood evaluation, classification of texts, and named object recognition[14].

E. DistilBERT

DistilBERT, a simpler, quicker, and more efficient BERT model, retains most of its performance. Knowledge distillation is the process of teaching a smaller model to behave like a larger model that has already been trained. In this case, BERT is the larger model that was already taught. By cutting down on the number of factors by about 40%, DistilBERT makes computations easier and faster, and it also uses less memory. Despite these decreases, BERT maintains 97% of its language knowledge[15,16]. Figure 1 is the different Approaches of Transformers.

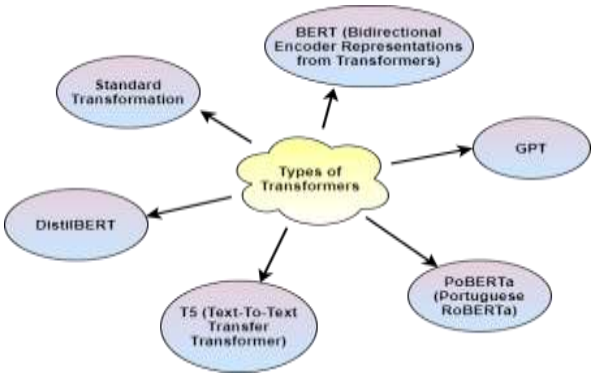

Figure 1: *Different Approaches of Transformers*

2. Literature Survey

Yang Yin et al [1] has developed a PLM for the detection of GPT. Retrieving pertinent semantic information and precisely describing implicit cultural semantics present difficulties. Employ the GPT-3.5 to examine and map semantic aspects, thereby optimising two design idea generators, addressing these. These generators help to automatically retrieve and represent culturally semantic information in plain language form. Validation reveal that this approach produces design conceptions that precisely reflect cultural knowledge.

Kyowon Gu et al [2] has introduced a LI-RADS for the determination of GPT in different sectors. The GPT-3.5 paradigm for product design analysis and mapping of cultural semantics. Data collecting starts the process as cultural semantic knowledge is obtained from many sources. After that, this data is cleaned to find significant symbols and attributes fit for cultural settings. By means of fine-tuning to grasp these cultural semantics, the GPT-3.5 model is able to create design conceptions reflecting the cultural knowledge. Two design idea generators are developed: one stresses natural language description and the other symbolic semantic information expression. [3] These generators translate abstract cultural themes into specific design ideas. Iterative testing and improvement where produced design ideas are assessed and refined depending on comments are also part of the approach. The final result is a collection of culturally enhanced product design ideas meant to motivate designers. Table 1 is existing approaches comparative analysis.

Tinghui Ouyang et al [4] has implemented a LLM related methodology for the text identification. The part on the methodology of the paper offers a thorough way for assessing the stability of a sentiment analysis system derived from ChatGPT. Two main elements of this are model resilience and operational stability. Operational stability is mostly concerned with the consistency and dependability of the system under many circumstances. This include looking at the system's capacity to withstand disturbances and handling of various inputs.

Tong Guo et al [5] has chosen a LLM based technology for the detection of text in different scenario. This method uses iterative improvement to make the information better over time, which makes the model more accurate. Effective Using minimum labelling to find and fix mistakes helps to focus on lowering the manual annotation time, therefore drastically minimising the time needed as opposed to conventional techniques. Daniele Grandi et al [6] has developed a LLM based methodology. And despite more computing time, the strategy included a chain-of-thought method to improve the reasoning capacity of the models. The findings underlined the need of model-specific tailoring and indicated that while chain-of-thought approaches enhanced explainability, they also required major computing resources. This technique combines human insights and AI to

optimise sustainable product design material choices.

Samuel N. Kirshner et al [7] has selected a LLM methodology for the prediction of text. The approach presented in the given PDF focuses on assessing ChatGPT's decision-making & recommendation biases in many experimental settings. Firas Wahsheh et al [8] has worked using AI to predict the text. The steps outlined in the paper involve creating and improving the GPT-4 model using a large dataset that includes product details, user questions, and buying habits from e-commerce sites.

2.1. Research Gaps Identified:

- A more detailed framework for human-AI feedback loops and knowledge transfer mechanisms could improve the integration of expert insights into the models, particularly for complex, evolving industries like sustainable product design.
- Research could explore the impact of data diversity and bias on the model's ability to accurately capture and represent cultural semantics. Studies could analyze how incomplete or biased datasets might skew the AI's understanding of certain cultures and propose solutions to mitigate these issues.

3. Proposed methodology

3. 1. Transformers

Transformers is a type of neural network topology that came out in 2017 and is mostly used for positions related to NLP. When it comes to producing predictions or creating text, they make use of a technique known as self-attention, which enables the model to evaluate the significance of various words inside a phrase. Transformers are able to analyse all of the words in a phrase concurrently, in contrast to standard sequential models such as RNNs. This allows for quicker training and improved management of long-range dependencies. Encoders and decoders are the components that make up this design. The encoder is responsible for processing the sequence that is input, while the decoder is responsible for producing the sequence that is output.

3.2. GPT2LMHeadModel

GPT2LMHeadModel is a Hugging Face Transformers library implementation that works with OpenAI's GPT-2 model for NLP. This Transformer model generates human-like text. The OpenAI-developed text generator generates coherent and contextually appropriate text. This uses the Transformer architecture's decoder. Instead of the encoder-decoder Transformer design, GPT-2 employs simply the decoder. These features make it perfect for autoregressive tasks, in which the model creates text by guessing the next word in a string based on the words that came before it. GPT-2, like previous Transformer models, employs self-attention to prioritise input tokens while creating the next token. Before learning particular tasks, GPT-2 is pre-trained on a vast corpus of text data to get a general knowledge of language. Hugging Face Transformers' GPT2LMHeadModel class simplifies language modelling using the GPT-2 model.

Vocabulary (Simplified for Example):

- {"Samsung": 0, "80": 1, "cm": 2, "(32": 3, "Inches)": 4, "Wondertainment": 5, "Series": 6, "HD": 7, "Ready": 8, "LED": 9, "Smart": 10, "TV": 11, "IJA32T4340BKXXL": 12, "(Glossy": 13, "Black)": 14, "Category": 15, "Televisions": 16, "/Smart": 17, "Televisions.": 18, "Price:": 19, "₹8,999.": 20, "Link:": 21, "[Amazon]": 22}

Input Sequence: "Samsung 80 cm (32 Inches) Wondertainment"

Token IDs: Using the vocabulary, "Samsung 80 cm (32 Inches) Wondertainment" is tokenized as [0, 1, 2, 3, 4, 5].

GPT-2 Model Components

Embeddings

- **Embedding Matrix (E):** Suppose our embedding matrix is E = [[0.1, 0.2], [0.3, 0.4], [0.5, 0.6], [0.7, 0.8], [0.9, 1.0], [1.1, 1.2], [1.3, 1.4], [1.5, 1.6], [1.7, 1.8], [1.9, 2.0], [2.1, 2.2], [2.3, 2.4], [2.5, 2.6], [2.7, 2.8], [2.9, 3.0], [3.1, 3.2], [3.3, 3.4], [3.5, 3.6], [3.7, 3.8], [3.9, 4.0], [4.1, 4.2], [4.3, 4.4], [4.5, 4.6], [4.7, 4.8], [4.9, 5.0]]
- **Embedded Input:**
  - For token "Samsung" (index 0): [0.1, 0.2]
  - For token "80" (index 1): [0.3, 0.4]
  - Continue similarly for other tokens.

Transformer Block (Single Layer)

Self-Attention Mechanism

1. **Query, Key, and Value Matrices (Weights):**
   - Suppose $W_Q = W_K = W_V$ =[[1,0],[0,1]]
2. **Compute Queries (Q), Keys (K), and Values (V):**
   - For token "Samsung":
     - Query: Q=[0.1,0.2]× W_Q =[0.1,0.2]
     - Key: K=[0.1,0.2]× W_K =[0.1,0.2]
     - Value: V=[0.1,0.2]× W_V =[0.1,0.2]
   - Repeat for other tokens.
3. **Compute Attention Scores:**
   - For simplicity, let's compute attention score for the token "Samsung":

- Scores=$\frac{QK^T}{\sqrt{H}}$
- Given H=, compute QK^T:
  - [0.1,0.2]·[0.1,0.2]^T=[0.01+0.04]=0.05
- Scaled Scores: $\frac{0.5}{\sqrt{2}}$=0.05/1.414≈0.035

4. **Apply Softmax to Get Attention Weights:**
   - Softmax of [0.035] (since there is only one score for simplicity):
     - Softmax(0.035)=$\frac{e^{0.035}}{e^{0.035}}$ =1
5. **Compute Attention Output:**
   - Weighted sum of the values using attention weights:
     - Attention Output=Weight×V =1×[0.1,0.2]=[0.1,0.2]

Feed-Forward Network (FFN)

- **FFN Output:**
  - Assume FFN is an identity function for simplicity:
    - FFN Output=[0.1,0.2]

Output Layer

- **Project Hidden States to Vocabulary Size:**
  - Assume W_O is the transpose of the embedding matrix E:
    - $W_O=E^T$
    - Output: [0.1,0.2]×W_O

Final Output Calculation:

- Let's say after projection, we get logits for each token:
  - Logits: [0.11,0.25,0.39] (as previously detailed)
- **Apply Softmax to Get Probabilities:**
  - Exponentials: $e^{0.11}$≈1.116, $e^{0.25}$≈1.284, $e^{0.39}$≈1.477
  - Sum of exponentials: 1.116+1.284+1.477=3.877
  - Softmax Probabilities: [0.288,0.331,0.381]

Final Prediction:

- The token with the highest probability is "cm" (index 2), so the model predicts "cm" as the next word.

3.3 GPT2Tokenizer

Tokenizing text in preparation for it to be processed by the GPT-2 model is the primary purpose of the GPT2Tokenizer, which is an essential component of the Hugging Face Transformers library. The act of transforming raw text into a series of tokens that the model can comprehend and modify is referred to as tokenization. The GPT2Tokenizer employs a subword tokenization approach known as Byte Pair Encoding (BPE), which divides word into subwords or letters depending on the frequency with which they appear in the training corpus. The tokenizer is able to handle a broad vocabulary, including uncommon and unheard-of terms, since it is able to break them down into subwords that are more popular. In addition to converting text into tokens, the tokenizer is also responsible for the inclusion of specific tokens that are needed by the model. One example of this is the end-of-sequence token as an example.

Input Sentence

"Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black). Category. Televisions Smart Televisions. Price: {8,999}.
Link:https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"

Tokenization

The tokenizer converts the text into token IDs based on its vocabulary. For example:

- "Samsung" -> 1234
- "80" -> 5678
- "cm" -> 910
- "(32" -> 1112
- "Inches)" -> 1314
- "Wondertainment" -> 1516
- "Series" -> 1718
- "HD" -> 1920
- "Ready" -> 2122
- "LED" -> 2324
- "Smart" -> 2526
- "TV" -> 2728
- "IJA32T4340BKXXL" -> 2930
- "(Glossy" -> 3132
- "Black)" -> 3334
- "(Black)." -> 3536
- "Category." -> 3738
- "Televisions" -> 3940
- "Smart" -> 4142
- "Televisions." -> 4344
- "Price:" -> 4546
- "{8,999}" -> 4748
- "Link:" -> 4950
- "https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/" -> 5152

So, the token IDs might look like:
[1234, 5678, 910, 1112, 1314, 1516, 1718, 1920, 2122, 2324, 2526, 2728, 2930, 3132, 3334, 3536, 3738, 3940, 4142, 4344, 4546, 4748, 4950, 5152]

Detokenization: To convert the token IDs back to readable text, the process is reversed:
Token IDs: [1234, 5678, 910, 1112, 1314, 1516, 1718, 1920, 2122, 2324, 2526, 2728, 2930, 3132, 3334, 3536, 3738, 3940, 4142, 4344, 4546, 4748, 4950, 5152]
Detokenized Text: "Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black).

Category. Televisions Smart Televisions. Price: {8,999}.
Link:https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"

3.4. GPT2Config

GPT2config specifies the architecture and many values controlling the behaviour of the model. N_layer, which defines model depth, and n_head, which regulates parallel attention methods in each layer, are key GPT2Config characteristics. The setting determines the hidden layer size (n_embd), which affects embedding dimensionality and model capacity. In addition, GPT2Config allows you to select the vocabulary size (vocab_size), which is the number of unique tokens that the model can manage, along with the maximum sequence length (n_positions), which is the length of input sequences that the model can process. GPT2Config also controls other critical settings such dropout rates for regularisation, which by random deactivating a percentage of the neurons assist avoid overfitting during training. Also included in the design is the activation function that the model uses, which is usually the GEU (Gaussian Error Linear Unit).

3.5. Fine Tuning of GPT-2

These metrics include loss values, accuracy, along with complexity. Training rate scheduling, gradient cutting, and early termination are also used to improve stability and efficiency. Fine-tuning can be implemented in a variety of applications, including sentiment analysis, query answering, and text generation, by furnishing the model with pertinent datasets. Train GPT-2 on a discussion dataset to fine-tune it for a conversational agent. A validation or test set evaluates the model's performance and generalisation after fine-tuning. After fine-tuning, the model may be stored and used for inference, using fresh inputs to produce or forecast text. Fine-tuning GPT-2 lets it use its language comprehension and generating skills while adjusting to its position. Fine-tuning is a strong way to customise GPT-2 to perform effectively on specialised tasks with less data than training a model and the process is shown in figure 2.

3.6. Count Vectorize

In NLP, count vectorization is a basic method that converts text data into numerical vectors for use as input for machine learning systems. Text documents are converted into token count matrices using this approach. The matrix is organised such that each row represents a document, and each column indicates a distinct token (word or phrase) from the total corpus. The value for every matrix cell shows the frequency of the related token in the relevant document. This method is appropriate for many analytical and predictive chores as it facilitates the quantification of the text data. Often the initial phase of text preparation, count vectorization opens the path for more complex text representations as word embeddings or

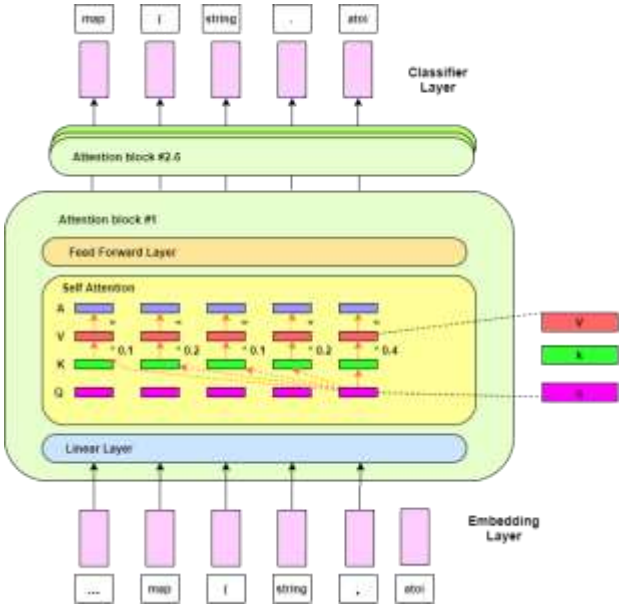

Figure 2: *Working of Fine Tuned GPT Model*

TF-IDF. Count Vectorization is accomplished using techniques from libraries such Scikit-learn most often.

Example Text Corpus

Let's use a simple text corpus consisting of three documents:

1. **Paragraph A**: "Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black). Category. Televisions Smart Televisions. Price: {8,999}.
Link:https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"
2. **Paragraph B**: "LG 80 cm (32 Inches) Smart TV with Built-in WiFi (Glossy Black). Best Price: {9,499}. Category. Televisions. Link: https://www.amazon.in/LG-Smart-TV-Glossy-Black/"
3. **Paragraph C**: "Sony Bravia 108 cm (43 Inches) 4K Ultra HD Smart TV KD-43X8000H (Black). Price: {24,999}. Best Category: Televisions Smart. Link: https://www.amazon.in/Sony-Bravia-4K-Ultra-HD-TV"

Step-by-Step Process

Step 1: Tokenization

First, we tokenize the text into individual words:

1. **Paragraph A**: "Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black). Category. Televisions Smart Televisions. Price: {8,999}.

Link:https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"

- Tokenized: ["Samsung", "80", "cm", "Inches", "Wondertainment", "Series", "HD", "Ready", "LED", "Smart", "TV", "IJA32T4340BKXXL", "Glossy", "Black", "Category", "Televisions", "Smart", "Price", "8,999", "Link"]

2. **Paragraph B**: "LG 80 cm (32 Inches) Smart TV with Built-in WiFi (Glossy Black). Best Price: {9,499}. Category. Televisions. Link: https://www.amazon.in/LG-Smart-TV-Glossy-Black/"

- Tokenized: ["LG", "80", "cm", "Inches", "Smart", "TV", "Built-in", "WiFi", "Glossy", "Black", "Best", "Price", "9,499", "Category", "Televisions", "Link"]

3. **Paragraph C**: "Sony Bravia 108 cm (43 Inches) 4K Ultra HD Smart TV KD-43X8000H (Black). Price: {24,999}. Best Category: Televisions Smart. Link: https://www.amazon.in/Sony-Bravia-4K-Ultra-HD-TV"

- Tokenized: ["Sony", "Bravia", "108", "cm", "Inches", "4K", "Ultra", "HD", "Smart", "TV", "KD-43X8000H", "Black", "Price", "24,999", "Best", "Category", "Televisions", "Smart", "Link"]

Step 2: Build Vocabulary

The next step is to build a vocabulary of unique words across all the paragraphs: Vocabulary: ["80", "cm", "Inches", "Smart", "TV", "Price", "Category", "Televisions", "Black", "Link", "Samsung", "Wondertainment", "Series", "HD", "Ready", "LED", "IJA32T4340BKXXL", "LG", "Built-in", "WiFi", "Best", "Glossy", "Sony", "Bravia", "108", "43", "4K", "Ultra", "KD-43X8000H", "24,999", "8,999", "9,499"]

Each word is assigned an index:

- "80" -> 0
- "cm" -> 1
- "Inches" -> 2
- "Smart" -> 3
- "TV" -> 4
- "Price" -> 5
- … and so on for the rest of the vocabulary.

Step 3: Create the Document-Term Matrix

Using this vocabulary, we transform each document into a vector of token counts. For example:

1. **Paragraph A** -> [1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1]
   - "80": 1 occurrence
   - "cm": 1 occurrence
   - "Inches": 1 occurrence
   - "Smart": 1 occurrence
   - "TV": 1 occurrence
   - "Price": 1 occurrence
   - … and so on.
2. **Paragraph B** -> [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]
   - Similar breakdown for Paragraph B.
3. **Paragraph C** -> [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0]
   - Similar breakdown for Paragraph C.

Document-Term Matrix

In matrix form, this is represented as:

Document	80	Cm	Inches	Smart	TV	Price	Category	Television	Black
ParagraphA	1	1	1	1	1	1	1	2	1
ParagraphB	1	1	1	1	1	1	1	1	1
ParagraphC	0	1	1	1	1	1	1	1	1

3.7 Cosine Similarity

In text analysis and data retrieval, cosine similarity is a frequently used metric measuring the cosine of an angle across two vectors that are not zero in a space with multiple dimensions. Since this statistic captures the direction rather than the vector magnitude, it is especially helpful for evaluating the similarity across documents or text fragments. Any written data is shown as a series of vectors in a space with many dimensions, and each dimension is linked to a different word in the language. The cosine similarity score runs from -1 to 1, where 1 denotes that the vectors are exactly aligned that is, the papers are similar in terms of direction 0 denotes they are orthogonal that is, they share no terms and -1 implies they are diametrically opposing. Cosine similarity's main benefits are its capacity to convey an indicator of similarity free from document length, therefore emphasising only the word use overlap.

Applying to Text Data

To calculate cosine similarity for text, we first need to represent the text as vectors. We will use a

simplified example where each word is treated as a dimension.

Example Paragraphs:

1. **Paragraph A:** "Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV IJA32T4340BKXXL (Glossy Black) (Black). Category. Televisions Smart Televisions. Price: {8,999}. Link: https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Glossy-Black/"
2. **Paragraph B:** "Samsung 32 Inches Wondertainment Series HD Ready LED Smart TV Black. Category: Televisions. Price: 8,999. Link: https://www.amazon.in/Samsung-Wondertainment-UA32T4340BKXXL-Black/"

Let's create a simple term frequency (TF) vector representation for these paragraphs.

Assumptions for Vector Representation:

- Terms: ["Samsung", "32", "Inches", "Wondertainment", "Series", "HD", "Ready", "LED", "Smart", "TV", "Black", "Category", "Price", "8,999", "Link"]

Now, the vector representation of A and B would be the count of occurrence of above terms in each text but not considering the URL

- **Vector Representation for Paragraph A:** A=[1,1,1,1,1,1,1,1,1,1,2,1,1,1,1]
- **Vector Representation for Paragraph B:** B=[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]

Step 1: Calculate the Dot Product

$A \cdot B = 1\times1+1\times1+1\times1+1\times1+1\times1+1\times1+1\times1+1\times1+1\times1+1\times1+2\times1+1\times1+1\times1+1\times1+1\times1=16$

Step 2: Calculate the Magnitude of A and B

$\|A\| = \sqrt{1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+2^2+1^2+1^2+1^2+1^2} = \sqrt{18}$

$\|B\| = \sqrt{1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2} = \sqrt{15}$

Step 3: Calculate the Cosine Similarity

Cosine Similarity $= \frac{A \cdot B}{\|A\| x \|B\|} = \frac{16}{\sqrt{18} x \sqrt{15}} = \frac{16}{\sqrt{270}} \approx 0.973$

So the cosine similarity for these two paragraphs is approximately 0.973.

3.8. Pre-trained models for GPT

By providing strong, universal language models that may be modified for a broad range of particular tasks, models that have been trained for GPT represent a major development in natural language processing. These models OpenAI's GPT-2 and GPT-3 are trained on large datasets including varied online material, which helps them to produce coherent and contextually appropriate language throughout many uses. Learning to forecast the next word in a phrase, the pre-training process captures complex linguistic structures and patterns. GPT-2 offers adaptability in juggling performance with computing resources. With 175 billion parameters, GPT-3 dramatically expands the bounds of what is possible for language models, showcasing astounding powers in producing writing that is human-like, responding to queries, translating across languages, and even carrying out mathematical calculations. Hugging Face Transformers is one such platform that makes pre-trained models available and provides user-friendly interfaces for incorporating GPT models into various applications.

4. Results and Discussions

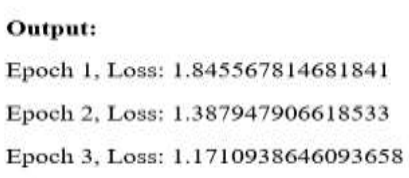

***Figure 3**: Loss Incurred by Proposed Methodology*

Figure 3 represents the losses incurred during the epochs. It offers an understandable graphic depiction of how the loss gradually drops with each training session, signifying successful learning. A full run through the training dataset is referred to as an epoch. Every training set sample is used precisely once to alter the model's parameters during each epoch. The model's initial forecasts were less accurate, as indicated by the loss, which starts at a greater value. As the learning rate of the model gets closer to an optimal point, the slower rate of reduction represents diminishing returns. Table 2 presents the comparison between the traditional and fine tuned GPT model. The

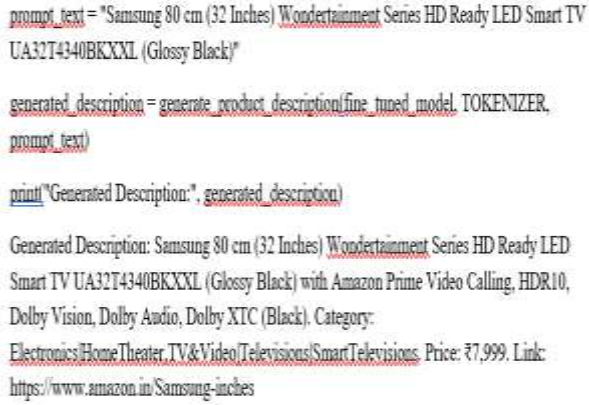

***Figure 4:** Generation of Product Description*

traditional approach is designed using the naïve Bayesian, which is efficient in processing the textual data but it lacks in identification of

categories and the appropriate place for finding the product. The proposed fine tune approach can easily identify the category of the product along with pricing and the location to purchase it.

Table 2: *Results Generation by Traditional & Proposed Approach*

Sample Prompt	**Generated Description using Traditional Approach**	**Generated Description using Proposed Approach**
Mi 108 cm (43 inches) Full HD Android LED TV	Mi 108 cm (43 inches) Full HD Android LED TV with 4K resolution, 4G LTE, Wi-Fi 802.11 b/g/n, Bluetooth 4.0, USB Type-C, HDMI, USB Type-A, USB 3.0, SD card reader, and microSD card slot. The LG G6 is the first Android TV with 4K resolution, 4G LTE, Wi-Fi 802.11 b/g/n, Bluetooth 4.	Mi 108 cm (43 inches) Full HD Android LED TV 4C Category:Electronics|HomeTheater, TV&Video|Accessories|SmartTelevisions.Price: ₹1,499. Link: https://www.amazon.in/L42-inches-Android-LED-TV-Smart-Televisions-Black.html?ref=cm-amazon-Amazon-Amazon-Amazon-US
Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV UA32T4340BKXXL (Glossy Black)	Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV UA32T4340BKXXL(Glossy Black)Wondertainment Series HD Ready LED Smart TV UA32T4340BKXXL (Glossy Black) Wondertainment Series HD Ready LED Smart TV UA32T4340BKXXL (Glossy Black) Wondertainment Series HD Ready LED Smart TV UA32T4340BKXX	Samsung 80 cm (32 Inches) Wondertainment Series HD Ready LED Smart TV UA32T4340BKXXL (Glossy Black). Category: Electronics|HomeTheater, TV&Video|SmartTelevisions|SmartTelevisions. Price: ₹1,999.Link:https://www.amazon.in/Samsung-80-inches-Wondertainment-Smart-TV-UA-64-Inch-LED-Smart-

Table 3. *Metric Evaluation in Different Approaches*

Model Name	**Algorithm**	**Metric**	**Description**	**Value**
Traditional	Naïve Bayesian	Accuracy	In traditional approaches, the efficiency of the model is measured using the accuracy, which measures the TF-IDF values to identify the relevance between the input prompt and generated text.	24.9%
Proposed	Fine Tuned GPT	Perplexity	In LLM's the efficiency of the model is computed using the likelihood between the elements. Lower the value higher the confidence of the model in predicting the category of the product.	7.07

Table 3 is metric evaluation in different Approaches. A comprehensive and in-depth overview of a Samsung Smart TV is shown in Figure 4, which also highlights a number of important features. It starts with the most important details, such the name of the TV and the color variants that are offered, so that users can identify the particular model being described right away. The TV's attributes, including its screen resolution, display technology, and cutting-edge technologies like voice control and built-in apps, are then thoroughly described in the figure 5. It also has visual aids to assist users quickly understand important features. A URL or link to buy the TV takes viewers to the official web page or approved shops, while the TV's pricing is prominently shown.

The model was producing increasingly more accurate predictions on the training set, as seen by the constant increase in training accuracy. Training and validation measures showed steady progress, indicating that the model had good learning capabilities. Deep learning models often evolve from learning fundamental patterns to understanding more complicated associations with a rapid initial training phase preceding a slower, more subtle improvement phase. The robustness and dependability of the model were ensured, so making it appropriate for real-world applications, by the monitoring and analysis of key performance

measures. Because it offers a statistic that rises as the model's effectiveness does, inverse perplexity is helpful. Better performance is indicated by higher values, which facilitates the interpretation of trends and advancements. The number of epochs or any other indicator of the training process is represented by the X-Axis. Y-Axis: Depicts inverted confusion. The inverse perplexity will rise as the model develops and learns.

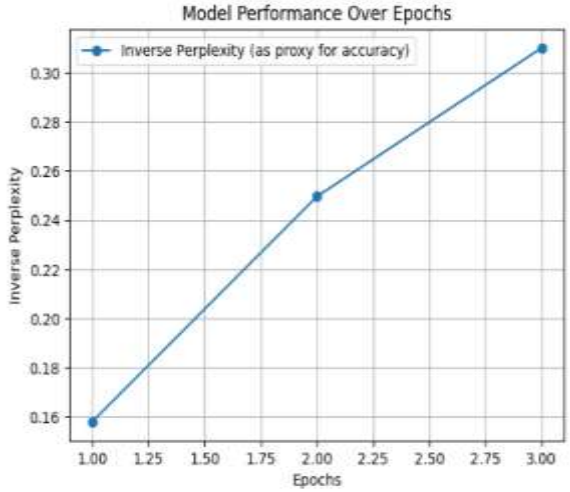

Figure 5: *Model Performance*

4. Conclusions

GPT-Product is a unique solution that revolutionises the product management space with its cutting-edge AI and NLP technology, providing unmatched capabilities. Its capacity to evaluate enormous volumes of data and provide insightful analysis guarantees that companies may make fast and correct judgements. Incorporating GPT-3.5's advanced language model improves user experience by offering premium content and deep knowledge of consumer feedback qualities essential for ongoing development and innovation. GPT-Product not only maximises resource use but also speeds the development cycle by automating mundane operations and providing actionable information, therefore producing better quality goods and more market agility. GPT-Product becomes more important as companies negotiate the complexity of contemporary marketplaces as it helps them to keep a competitive advantage by means of smart and effective product management practices. The subject discussed in the paper is interesting and a number of similar works were reported [17-20].

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] Yin, Y. (2024, June 23). Cultural Product Design Concept Generation with Symbolic Semantic *Information ExpressionUsingGPT*. https://doi.org/10.21606/drs.2024.508

[2] Gu, K., Lee, J. H., Shin, J., Hwang, J. A., Min, J. H., Jeong, W. K., Lee, M. W., Song, K. D., & Bae, S. H. (2024). Using GPT-4 for LI-RADS feature extraction and categorization with multilingual free-text reports. *Liver International*, 44(7),1578–1587. https://doi.org/10.1111/liv.15891

[3] Bdoor, S. Y., & Habes, M. (2024). Use Chat GPT in Media Content Production Digital Newsrooms Perspective. In Studies in Big Data (Vol. 144, pp. 545–561). *Springer Science and Business Media Deutschland GmbH.* https://doi.org/10.1007/978-3-031-52280-2_34

[4] Ouyang, T., MaungMaung, A., Konishi, K., Seo, Y., & Echizen, I. (2024). Stability Analysis of ChatGPT-based Sentiment Analysis in AI Quality Assurance. http://arxiv.org/abs/2401.07441

[5] Guo, T. (2024). Improving Text Generation for Product Description by Label Error Correction. *TechRxiv.* https://doi.org/10.36227/techrxiv.171198068.89981260/v1

[6] Grandi, D., Jain, Y. P., Groom, A., Cramer, B., & McComb, C. (2024). Evaluating Large Language Models for Material Selection. http://arxiv.org/abs/2405.03695

[7] Kirshner, S. N. (2024). GPT and CLT: The impact of ChatGPT's level of abstraction on consumer recommendations. *Journal of Retailing and Consumer Services*, 76. https://doi.org/10.1016/j.jretconser.2023.103580

[8] Wahsheh, F. R., Moaiad, Y. al, Baker El-Ebiary, Y. A., Amir Fazamin Wan Hamzah, W. M., Yusoff, M. H., & Pandey, B. (2023). E-Commerce Product Retrieval Using Knowledge from GPT-4. *2023 International Conference on Computer Science and Emerging Technologies*, CSET 2023.

https://doi.org/10.1109/CSET58993.2023.10346860

[9] Roumeliotis, K. I., Tselikas, N. D., & Nasiopoulos, D. K. (2024). Precision-Driven Product Recommendation Software: Unsupervised Models, Evaluated by GPT-4 LLM for Enhanced Recommender Systems. *In Software* 3(1);62–80).MDPIAG. https://doi.org/10.3390/software3010004

[10] Li, L., Zhang, Y., & Chen, L. (2023). Prompt Distillation for Efficient LLM-based Recommendation. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. CIKM '23: *The 32nd ACM International Conference on Information and KnowledgeManagement.ACM.* https://doi.org/10.1145/3583780.3615017

[11] Shi, G., Deng, X., Luo, L., Xia, L., Bao, L., Ye, B., Du, F., Pan, S., & Li, Y. (2024). LLM-Powered Explanations: Unraveling Recommendations Through Subgraph Reasoning(Version2). *arXiv*. https://doi.org/10.48550/ARXIV.2406.15859

[12] Soviero, B., Kuhn, D., Salle, A., Moreira, V.P. (2024). ChatGPT Goes Shopping: LLMs Can Predict Relevance in eCommerce Search. In: Goharian, N., *et al.* Advances in Information Retrieval. ECIR 2024. *Lecture Notes in Computer Science, vol 14611. Springer, Cham.* https://doi.org/10.1007/978-3-031-56066-8_1

[13] Wang, H., & Na, T. (2023). Rethinking E-Commerce Search. *In ACM SIGIR Forum* 57(2);1–19). Association for Computing Machinery(ACM). https://doi.org/10.1145/3642979.3643007

[14] Li, Y., Ma, S., Wang, X., Huang, S., Jiang, C., Zheng, H.-T., Xie, P., Huang, F., & Jiang, Y. (2024). EcomGPT: Instruction-Tuning Large Language Models with Chain-of-Task Tasks for E-commerce. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38*(17), 18582-18590. https://doi.org/10.1609/aaai.v38i17.29820

[15] Sara, Fatima. (2024). Data-driven E-commerce: The Intersection of Sentiment Analysis, Causal Reasoning, and LLMs. 10.13140/RG.2.2.13655.48806.

[16]Ngo Tran, G.T., Le Dinh, T., Pham-Nguyen, C. (2024). BABot: A Framework for the LLM-Based Chatbot Supporting Business Analytics in e-Commerce. In: Nguyen, N.T., *et al.* Computational Collective Intelligence. ICCCI 2024. *Lecture Notes in Computer Science(),vol14810.Springer,Cham*. https://doi.org/10.1007/978-3-031-70816-9_15

[17]Guven, M. (2024). A Comprehensive Review of Large Language Models in Cyber Security. *International Journal of Computational and Experimental Science and Engineering*, 10(3);507-516. https://doi.org/10.22399/ijcesen.469

[18]Adem, A. İrem, Turhan, Çiğdem, & Sezen, A. (2024). Systematic Mapping Study on Natural Language Processing for Social Robots. *International Journal of Computational and Experimental Science and Engineering*, 10(4);560-567. https://doi.org/10.22399/ijcesen.341

[19]R. Dineshkumar, A. Ameelia Roseline, Tatiraju V. Rajani Kanth, J. Nirmaladevi, & G. Ravi. (2024). Adaptive Transformer-Based Multi-Modal Image Fusion for Real-Time Medical Diagnosis and Object Detection. *International Journal of Computational and Experimental Science and Engineering*, 10(4);890-897. https://doi.org/10.22399/ijcesen.562

[20]AY, S. (2024). The Use of Agile Models in Software Engineering: Emerging and Declining Themes. *International Journal of Computational and Experimental Science and Engineering*, 10(4);1242-1248. https://doi.org/10.22399/ijcesen.703