# Natural Language Processing and Information Retrieval (CSEN1076)

*Abdelraheman Khaled Ali Asran Ahmed*

*Question Answering System for Product Recommendation*

*German University in Cairo*

# Contents

# 1 Introduction and Motivation

In our increasingly digital world, online shopping has become an integral part of our daily lives. Customers can now access a wide range of products from across the world with just a few clicks, making it simpler than ever to get exactly what they're looking for. But this wealth of options also brings with it a new set of challenges.

As users navigate e-commerce platforms like Amazon, they are often faced with a daunting task: deciding which product to purchase. With countless options available, it can be overwhelming to sift through product listings, compare features, and ultimately make a decision. This is where the concept of product recommendation comes into play.

Imagine you're in the market for a new laptop accessory, such as a mouse or a laptop stand. You may have specific requirements in mind, such as compatibility with your device or a certain price range. However, finding the perfect product that meets all your criteria can be like finding a needle in a haystack.

Many e-commerce platforms have implemented AI-driven conversational assistants, such as Alexa and AliMe, to address this challenge to help users navigate the online shopping experience. These assistants aim to provide personalized recommendations and answer user queries, guiding them towards products that best suit their needs.

One fundamental aspect of these conversational assistants is Product Question Answering (PQA), which involves automatically answering user-generated questions about specific products using natural language processing techniques. Unlike traditional question answering systems, PQA focuses on subjective opinions and preferences, making it a unique and challenging problem to solve.

This project aims to develop a Question Answering System for Product Recommendation, leveraging natural language processing (NLP) techniques. The system will analyze product descriptions, reviews, and other relevant data to provide personalized recommendations to users navigating e-commerce platforms like Amazon. The project is structured into three main phases: understanding the problem domain and conducting a literature review, building a neural network model, and fine-tuning the model with pre-trained models. The project's ultimate goal is to enhance the online shopping experience for users by simplifying the decision-making process and improving customer satisfaction.

# 2 Literature Review

Product Question Answering (PQA) is a specialized form of question answering that focuses on providing answers to queries related to products in the context of e-commerce and online shopping platforms. PQA is essential in the e-commerce space because it gives customers individualized and pertinent product information, which improves their shopping experience. Customers always have many questions and concerns about features, compatibility, user experiences, and other topics as they browse through huge selections of products and choices. PQA systems let users make informed decisions by automatically producing answers to questions submitted by users.

PQA systems can be categorized into several types based on the methodology employed to generate answers. These include:

1. **Opinion-based PQA**: Focuses on providing yes-no type answers to subjective questions by aggregating opinions from user-generated content such as product reviews.
2. **Extraction-based PQA**: Aims to extract specific text spans from supporting documents, such as product reviews or specifications, to serve as answers to user queries.
3. **Retrieval-based PQA**: Involves selecting the most relevant supporting documents from a candidate pool to answer user queries, typically by ranking documents based on their relevance to the query.
4. **Generation-based PQA**: Generates natural language responses to user queries based on the information available in supporting documents, often leveraging techniques such as sequence-to-sequence models.

Each type of PQA system has its own strengths and limitations, and the choice of approach depends on factors such as the nature of the questions, the availability of data, and the desired user experience.

## 2.1 Opinion-driven Product Query Resolution (PQR)

Opinion-driven PQR investigations concentrate on addressing binary inquiries prevalent within e-commerce platforms, where users frequently seek subjective viewpoints on products to guide their purchase choices. The primary aim of opinion-driven PQR is to deliver binary responses (Yes or No) to user-raised queries, typically derived from subjective expressions found in ancillary materials such as product reviews.

### 2.1.1 Datasets and Evaluation Mechanisms

Opinion-driven PQR systems frequently undergo evaluation using datasets like the Amazon Product Dataset, housing millions of answered queries and product reviews spanning diverse categories. Evaluation metrics such as Acc@k are commonly adopted to gauge performance, prioritizing the top $k$ queries based on prediction confidence.

### 2.1.2 Approaches

In their pioneering work, McAuley and Yang (2016) introduced Mixtures of Opinions for Question Answering (Moqa), a system built upon the Mixtures of Experts (MoEs) model. Moqa treats each review as an "expert" providing a binary prediction, either "Yes" or "No", in response to a given product-related question. The confidence of each review's prediction is weighted based on its relevance to the question, allowing Moqa to aggregate opinions effectively.

Subsequent advancements by Wan and McAuley (2016) and Yu and Lam (2018b) further refined Moqa to address the nuances of ambiguity and subjectivity inherent in product-related questions. Wan and McAuley's improvements focused on enhancing Moqa's ability to handle uncertain or ambiguous answers, ensuring robust performance across diverse question types. Meanwhile, Yu and Lam's contributions involved incorporating aspect-specific embeddings into Moqa, enabling the model to capture nuanced relationships between questions and reviews. These enhancements underscored the ongoing efforts to optimize opinion-based PQA systems for real-world applications.

In parallel, recent studies by Fan et al. (2019) and Zhang et al. (2019) have explored the potential of neural network architectures and pre-trained language models in advancing opinion-based PQA. By leveraging neural networks, such as BiLSTM, and pre-trained language models like BERT, these approaches aim to learn more sophisticated feature representations from the available data. By capturing the intricate interplay between questions and reviews, neural network-based models have demonstrated superior performance compared to traditional methods, showcasing their potential to revolutionize opinion-based PQA.

A notable innovation in opinion-based PQA comes from Rozen et al. (2021) with their introduction of Similarity-Based Answer Prediction (SimBA). SimBA introduces a novel approach to leveraging existing data by exploiting similarities in resolved questions about similar products. By identifying relevant answers from past interactions, SimBA offers a promising avenue for improving the accuracy and efficiency of opinion-based PQA systems. This innovative method highlights the importance of harnessing the wealth of available data to enhance the performance of PQA systems in real-world scenarios.

### 2.1.3 Limitations

Opinion-driven PQR strategies offer a straightforward approach to tackling a substantial portion of product-related queries using relatively uncomplicated methodologies. However, these strategies may lack intricate query-specific details, focusing primarily on presenting the overall opinion polarity without deeper insights.

## 2.2 Extraction-based PQA

Extraction-based Product Question Answering (PQA) operates akin to traditional extraction-based QA methods, aiming to extract specific spans of text from supporting documents as answers to product-related questions. The objective is to identify a sequence of tokens within a given document that correctly addresses the question at hand.

### 2.2.1 Datasets and Evaluation Mechanisms

Several datasets have been curated specifically for extraction-based PQA research, each with its unique characteristics and evaluation protocols. Xu et al. (2019) introduced the ReviewRC dataset, constructed from SemEval-2016 Task5 reviews, while Gupta et al. (2019) developed the AmazonQA dataset, derived from the Amazon dataset, distinguishing between answerable and unanswerable questions based on review content. Bjerva et al. (2020) contributed the SubjQA dataset, focusing on the relationship between subjectivity and PQA across various domains. Evaluation metrics such as Exact Match (EM) and F1 scores are commonly employed to assess model performance.

### 2.2.2 Approaches

Methodologically, researchers have explored diverse approaches to address the challenges inherent in extraction-based PQA. Xu et al. (2019) utilized pretraining objectives like masked language modeling and next-sentence prediction to enhance BERT's performance on both general MRC and e-commerce review datasets. In contrast, Gupta et al. (2019) integrated information retrieval techniques to filter irrelevant reviews and built an answerability classifier to handle unanswerable questions, employing the R-Net model for span-based QA. Bjerva et al. (2020) introduced a subjectivity-aware QA model, leveraging multi-task learning to jointly tackle extraction-based PQA and subjectivity classification.

### 2.2.3 Limitations

Despite the potential of extraction-based PQA to provide precise answers, its practicality in real-world applications may be limited due to its less user-friendly nature and potential loss of additional contextual information, as discussed by McAuley and Yang (2016) and Deng et al. (2022). Consequently, the focus on extraction-based PQA in recent years has been relatively limited compared to other PQA paradigms.

## 2.3 Retrieval-based PQA

Retrieval-based Product Question Answering (PQA) approaches frame the task as a sentence selection problem, aiming to retrieve the most suitable answer from a pool of candidate sentences to effectively address the given question.

### 2.3.1 Datasets and Evaluation Mechanisms

The absence of ground-truth question-review (QR) pairs poses a challenge for retrieval-based PQA evaluation. While efforts have been made to annotate additional QR pairs into existing datasets, such as the Amazon dataset, original datasets can still be employed for evaluation. Standard ranking metrics like mean average precision (MAP), mean reciprocal rank (MRR), and normalized discounted cumulative gain (NDCG) are commonly used to assess model performance.

### 2.3.2 Approaches

Initial efforts, like the SuperAgent chatbot by Cui et al. (2017), employed multiple ranking modules to select answers from various data sources within product pages. Subsequent work by Kulkarni et al. (2019) introduced a pipeline system involving question classification and ensemble matching models for answer ranking. Recent approaches tend to focus on end-to-end models trained on limited sources, addressing the need for extensive annotated data.

To mitigate the low-resource challenge, transfer learning frameworks have been proposed, leveraging shared knowledge from large-scale datasets like Quora and MultiNLI. Mittal et al. (2021) introduced a distillation-based training algorithm using QA pairs retrieved by syntactic matching systems. Additionally, distant supervision paradigms and multi-task deep learning methods have been explored to train models using both user-generated QA data and manually labeled QR pairs.

Efforts to improve interpretability include identifying important keywords within questions and associating relevant words from QA pairs. Pre-trained

language models like BERT have been employed to obtain weak supervision signals from community QA pairs for measuring relevance between questions and heterogeneous information.

### 2.3.3 Limitations

Retrieval-based approaches offer complete and informative sentences as answers but may lack precision in addressing specific questions due to the general nature of supporting documents like reviews, which are not tailored for answering individual queries.

## 2.4 Generation-based PQA

Generation-based Product Question Answering (PQA) draws inspiration from sequence-to-sequence (Seq2seq) models applied in other natural language generation tasks. These models aim to automatically generate natural language sentences as answers to product-related questions.

### 2.4.1 Datasets Evaluation Mechanisms

The Amazon dataset and JD dataset are commonly used for generative PQA. The JD dataset, originating from one of China's largest e-commerce platforms, encompasses a vast array of products and categories, each associated with QA pairs, reviews, and product attributes. Evaluation of generation-based methods involves both automatic metrics like ROUGE, BLEU, Embedding-based Similarity, BertScore, and BleuRT, as well as human evaluation protocols assessing aspects such as fluency, consistency, and helpfulness.

### 2.4.2 Approaches

Generation-based PQA typically involves retrieving relevant documents as a preprocessing step before model development. To address noise in retrieved documents, approaches like Wasserstein distance-based adversarial learning and attention-based weighting strategies have been employed. Furthermore, considering the subjective nature of many product-related questions, some methods incorporate opinion mining into answer generation. Cross-passage hierarchical memory networks and heterogeneous graph neural networks are proposed to leverage information from diverse resources for answer generation. Additionally, efforts have been made to tackle issues like safe answer generation and benchmarking PQA over semi-structured data.

### 2.4.3 Limitations

Generation-based methods offer natural answers tailored to specific questions but face challenges like hallucination, factual inconsistency, and a lack of robust automatic evaluation protocols.

## 2.5 Conclusion

In conclusion, each type of PQA system has its strengths and limitations, and the choice of approach depends on factors such as the nature of the questions, data availability, and desired user experience. Future research efforts should aim to address the challenges inherent in each approach while further enhancing the accuracy and efficiency of PQA systems in real-world scenarios.

# 3 Dataset Overview

## 3.1 Introduction to the Dataset

The dataset that will be used for our analysis is "Amazon's 500 Bestsellers in Laptop Gear 2024". This carefully selected dataset offers a comprehensive view of the current trends and preferences within the laptop accessory market on Amazon. It provides insights into a wide array of products, ranging from essential electronic peripherals to novelty decals, reflecting the diverse interests and demands of consumers in the digital marketplace.

## 3.2 Source and Size of the Dataset

The dataset originates from Amazon and comprises a collection of the top 500 best-selling laptop accessories. It is expertly curated to capture the pulse of the market, offering a comprehensive snapshot of products resonating with customers on the platform.

## 3.3 Main Features and Variables

- **Title**: This feature captures the essence of each product, providing a concise description of its main attributes and functionalities.
- **Brand**: Identifies the producer or manufacturer of the product, which is essential for brand impact analysis and understanding market dynamics.
- **Description**: Offers detailed narratives about each product, providing rich textual data ripe for analysis. This feature enables deeper insights into product features, specifications, and unique selling points.
- **Price/Currency**: Indicates the currency in which the price is listed, providing essential information for market analysis and comparison across different regions or platforms.
- **Price/Value**: Provides economic data points that are crucial for market and sales analyses. Understanding price trends and variations can help identify pricing strategies and consumer preferences.
- **Stars**: Reflects customer satisfaction and perceptions of product quality. The star rating system, commonly used on e-commerce platforms like Amazon, offers valuable insights into consumer sentiment and product performance.
- **ReviewsCount**: Measures the level of engagement and popularity among consumers. A higher number of reviews typically indicates greater customer interest and interaction with the product.

## 3.4 Missing Data Analysis

### 3.4.1 Analysis Findings

An initial analysis of the dataset was conducted to gain insights and identify potential limitations. This analysis included examining missing values across various columns, which can provide valuable cues regarding data quality and completeness. The plot in Figure 1 illustrates that approximately 245 entries lack descriptions, potentially impacting the comprehensiveness of our analysis as detailed narratives provide valuable insights into product features, functionality, and consumer perceptions. Additionally, there are 30 missing values each in the 'price/currency' and 'price/value' columns, suggesting potential inconsistencies or errors in recording price information. Furthermore, around 241 entries are missing values for both 'stars' and 'reviewsCount' columns, indicating that these products may not have received any reviews yet.

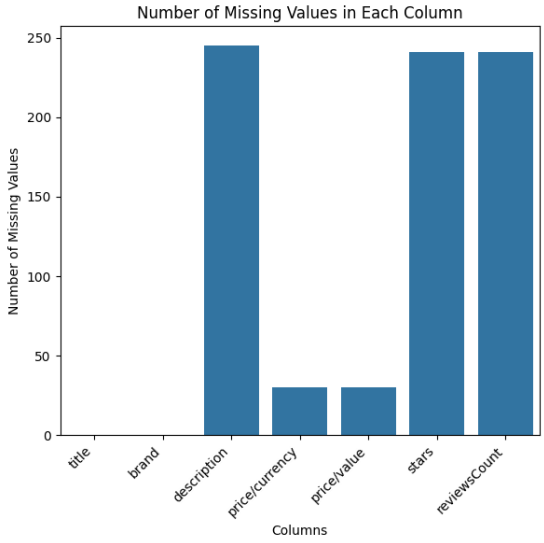



Figure 1: Number of Missing Values

To facilitate further analysis of the dataset, missing values were addressed by substituting them with appropriate placeholders. For the entries lacking descriptions, the placeholder "No description Available" was used. Similarly, missing values in the 'stars' and 'reviewsCount' columns were replaced with zeros. As only 30 rows were missing price information, these entries were dropped from the dataset

### 3.4.2 Limitations

- **Biased Reviews Data:** The missing values in the 'stars' and 'reviewsCount' columns suggest potential biases in the review data. Products without reviews may have different characteristics or may not have been as popular as those with reviews, leading to skewed perceptions of product satisfaction and popularity.
- **Incomplete Product Information:** The absence of descriptions for a significant number of entries limits the depth of our understanding of product features and attributes. This could introduce bias if certain types of products are more likely to have missing descriptions, affecting analyses that rely on detailed product information.
- **Price Data Reliability:** The missing values in the price-related columns raise concerns about the reliability of pricing data. Incomplete or inconsistent pricing information could lead to inaccurate analyses of pricing trends or comparisons between products.
- **Quality of Data Collection**: Overall, the presence of missing values across multiple columns indicates potential issues in data collection or recording processes. These inconsistencies may introduce biases and adversely impact the performance of models relying on this data.

## 3.5 Numeric Columns Analysis

### 3.5.1 Exploring Distributions of Numeric Columns

For the analysis of the numeric columns, we will begin by examining the trends within these columns. This exploration aims to uncover the distribution and fluctuations within the numerical data, shedding light on potential patterns. We will then explore the correlation between these numeric columns to understand the relationships and dependencies among different variables. This analysis will provide valuable insights into the underlying patterns and associations within the dataset, facilitating a deeper understanding of the factors influencing product features, pricing, customer satisfaction, and other key metrics.

- **Price/Value Distribution:** The histogram analysis reveals that the prices of laptop accessories vary widely, ranging from $2.99 to $575.00. The distribution is right-skewed, with a larger number of products having lower prices. Common price points cluster around values below $10.00, indicating that lower-priced accessories are more prevalent in the dataset. However, higher-priced accessories are also represented but less frequently.
- **Stars Distribution:** The stars column, representing customer satisfaction ratings, shows a diverse range of ratings from 0.0 to 5.0. The most

common rating is 0.0, showing that many products had missing values in the stars column. However, there are also numerous highly rated products with ratings above 4.0, indicating a mix of satisfaction levels among customers.

- **ReviewsCount Distribution:** The reviewsCount column, indicating the number of reviews received, exhibits a wide range of counts from 0 to 42821. Like the stars column, the most common count is 0, showing that many products had missing values in the reviews column. However, there are products with varying levels of review engagement, with some receiving a significant number of reviews. Overall, while some products have high engagement levels, others lack reviews, potentially impacting the reliability of customer feedback.

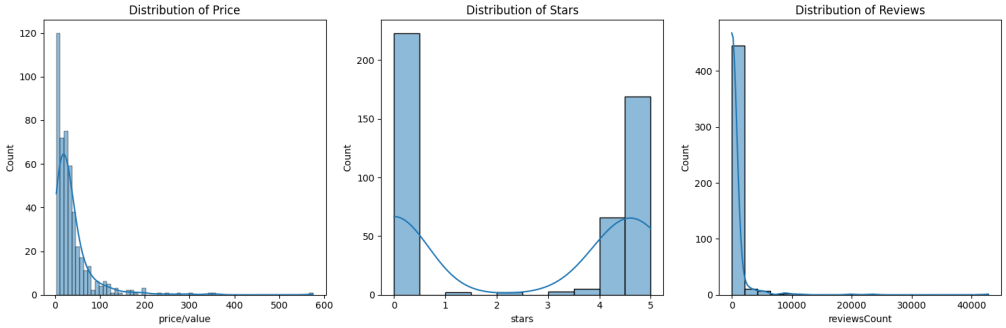



Figure 2: Numeric Columns Distributions

### 3.5.2 Correlation Analysis

The correlation analysis among the numeric columns in the dataset suggests several insights. Across all pairs of columns (price/value vs. stars, price/value vs. reviews count, stars vs. reviews count), there is more or less no correlation. This indicates that higher prices don't always equate to higher satisfaction, pricing has little impact on customer engagement measured by review counts, and there's no strong relationship between customer satisfaction ratings and the number of reviews.

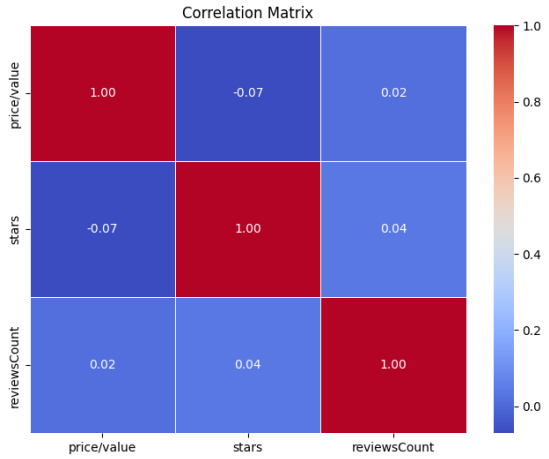



Figure 3: Correlation Matrix

### 3.5.3 Limitations

The presence of numerous products with zero ratings and reviews may skew perceptions of overall product satisfaction and popularity. Additionally, the right-skewed distribution of prices and review counts indicates potential biases towards lower-priced and less-reviewed products, impacting the generalizability of insights derived from the dataset. The observed correlations, or lack thereof, might stem from the dataset's narrow scope, which only includes the top 500 best-selling products. This selection bias could lead to weaker or negligible correlations since these popular products generally have lower prices and higher customer satisfaction ratings.

## 3.6 Textual Analysis of Product Titles, Brands, and Descriptions

This section explores the textual aspects of the dataset, focusing on the 'Title', 'Brand', and 'Description' columns. The goal is to uncover insights into the characteristics, features, and brand associations of the listed laptop accessories.

### 3.6.1 Brand Analysis

The dataset comprises 327 unique brands, reflecting a diverse range of options available in the laptop accessory market. Among these, the most frequent brands include "Generic" for unbranded products, along with "LOVEVOOK" each appearing 11 times in the dataset.

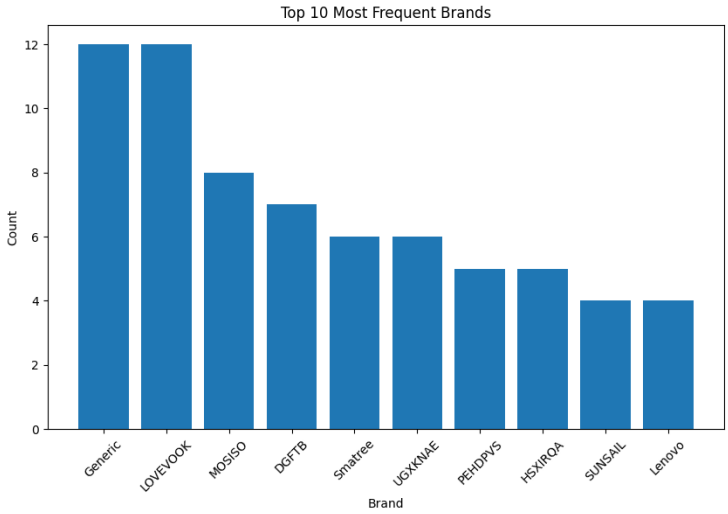



Figure 4: Top 10 Brands

The presence of "Generic" as a frequent brand may indicate a segment of unbranded or generic products. This may suggest a set of budget-friendly options or products sourced from various manufacturers without a distinct brand identity. However, it's unclear how these products compare to branded alternatives in terms of quality. Additionally, recommending products labeled as "Generic" may introduce ambiguity for users relying on the model's recommendations, as they may lack the assurance or familiarity associated with recognized brands. This ambiguity could affect user trust and satisfaction with the recommendation system, highlighting the importance of addressing brand diversity and clarity in product recommendations.

### 3.6.2 Title and Description Analysis

In this section, the focus is on exploring the title and description columns of the dataset, providing valuable insights into the characteristics and features of laptop accessories.

- **Basic summary statistics:** For titles, we observed a total word count of 13,107, with an average word length of approximately 5.06 characters. The character count for titles is 77,731, indicating concise descriptions. However, the relatively low sentence count of 15 suggests that titles are typically short phrases rather than complete sentences, which is expected given their purpose of providing concise product identifiers.

  On the other hand, the description column exhibits more extensive text content, with a significantly higher word count of 33,050. Despite the

higher word count, the average word length is slightly lower at approximately 4.78 characters, indicating a slightly more varied vocabulary compared to titles. The character count for descriptions is notably higher at 186,068, reflecting the additional detail and information provided in this column. Furthermore, the higher sentence count of 992 suggests that descriptions contain more complete sentences and detailed information about the products.

- **Word count distribution:** This section explores the distribution of word counts across titles and descriptions, shedding light on the common lengths and patterns observed in these textual elements. For titles, there is a prevalence of moderate-length titles, typically ranging from 16 to 29 words. This suggests that concise yet descriptive titles are common among laptop accessories listed on the platform. Longer titles, exceeding 29 words, are less frequent, indicating a preference for brevity in conveying product information.

  In contrast, descriptions exhibit a skew towards shorter lengths, with the majority falling within the range of 2 to 16 words. Notably, many of these concise descriptions correspond to products that lacked detailed descriptions and were replaced with the placeholder "No description available." This introduces a potential bias in the analysis, as a significant portion of products in the dataset lack genuine descriptions. Consequently, the true distribution of description lengths may be skewed, affecting the overall understanding of how product information is presented. Additionally, there is a notable decline in frequency for descriptions with word counts exceeding 16, indicating that longer descriptions are less prevalent.

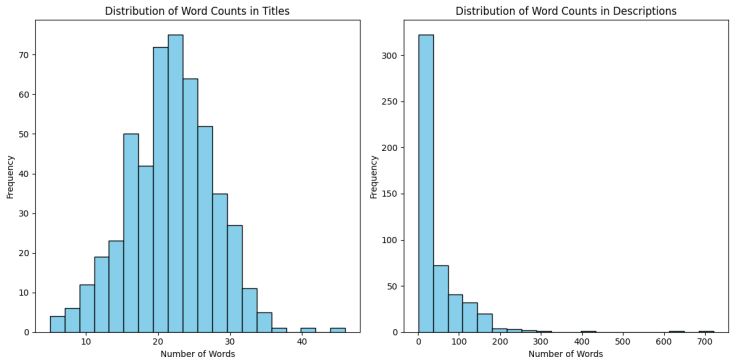
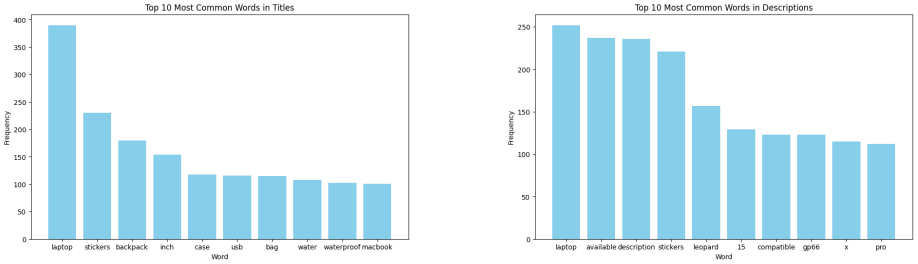



Figure 5: Word Count Distribution for Titles and Descriptions

- **Vocabulary Analysis:** The title vocabulary consists of 2837 unique words. The top 10 most frequent words include terms like "laptop", "stickers", "backpack", "inch", and "case", which are indicative of common laptop accessories. The high frequency of words related to specific products

such as "macbook", "waterproof", and "usb" suggests popular categories within the dataset.

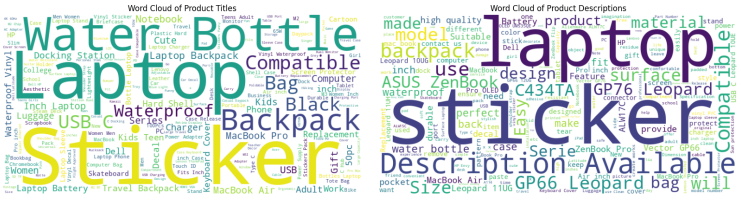
The description vocabulary is larger, comprising 5540 unique words. The top 10 most frequent words include terms like "laptop", "available", "description", "stickers", and "leopard". Interestingly, the word "available" appears frequently, possibly due to its use as a placeholder in descriptions. Other common words like "compatible", "pro", and specific product names like "gp66" and "15" reflect attributes and specifications mentioned in the descriptions




(a) Top 10 words in titles

(b) Top 10 words in descriptions

Figure 6: Comparison of top 10 words in titles and descriptions




(a) Title Word Cloud

(b) Description Word Cloud

Figure 7: Word Clouds of Titles and Descriptions

The most frequent words in both titles and descriptions are related to product features, types, and specifications commonly associated with laptop accessories. Certain words like "available" in descriptions may not contribute much semantic meaning but are included due to their use as placeholders. The vocabulary size in descriptions is larger than that in titles, indicating a greater diversity of language used to describe products and their attributes.

# 4 Basic Model Development and Evaluation

## 4.1 Introduction

In this section, we embark on the development and evaluation of a neural network model tailored for predicting the prices of laptop accessories based solely on their titles and descriptions. We have chosen to utilize only the title and description columns from the dataset, omitting other potentially relevant features. This decision was made to focus on the columns most directly related to the task at hand and to streamline our approach, particularly considering that we are not leveraging any pre-trained models.

The primary objective of this section is to construct and assess the performance of a neural network model, devoid of pre-trained representations, to infer the prices of laptop accessories. Leveraging the textual information encapsulated within these two columns, we aim to harness the power of natural language processing techniques and deep learning architectures to capture nuanced relationships between product attributes and pricing dynamics.

In the following subsections, we will explore the step-by-step process involved in developing the model:

- We begin by discussing the data preprocessing steps, specifically applied to the title and description columns.
- Following this, we outline the neural network architecture's components and their relevance to the prediction task.
- Next, we detail the training procedure, covering dataset partitioning, hyperparameter selection, and model optimization.
- Finally, we evaluate the model's effectiveness in predicting prices.

This section represents the culmination of efforts to transform raw textual data into actionable insights for users navigating laptop accessory purchases. Through careful experimentation and analysis, we aim to refine and enhance our model in subsequent sections.

## 4.2 Data Preprocessing

In this subsection, we detail the preprocessing steps undertaken to refine the textual data from the title and description columns of the dataset. Each preprocessing step is carefully chosen and justified to enhance the quality of the input data for our neural network model.

- **Cleaning and Tokenization of Text Data**

  We begin by cleaning the text data to remove any special characters or noise that may hinder model performance. This step ensures that the input data is standardized and free from unnecessary artifacts. Tokenization follows the cleaning process, where we split the text into individual tokens or words. By breaking down the text into its constituent elements, we facilitate further analysis and processing. Cleaning and tokenization are essential steps to prepare the textual data for analysis.

- **Lemmatization of Tokens**

  Following tokenization, we apply lemmatization to normalize the tokens, reducing them to their base or dictionary form. This step ensures consistency in the representation of words with similar meanings. This helps reduce the dimensionality of the feature space while preserving semantic meaning.

  Despite the potential computational overhead associated with lemmatization compared to stemming, its benefits in terms of precision and context preservation are deemed essential for our task. Moreover, considering the relatively small size of our dataset (470 products only), the computational overhead of lemmatization is acceptable.

  By opting for lemmatization, we aim to enhance the effectiveness of our neural network model, particularly in the absence of pre-trained representations, and facilitate more accurate predictions of laptop accessory prices.

- **Concatenation of Title and Description Sequences**

  Once the title and description tokens are lemmatized, we concatenate them to create unified sequences representing each product. This concatenation allows the model to consider both the product title and its description when making predictions. By combining the title and description sequences, we provide the model with a comprehensive representation of each product. This holistic view enables the model to capture relevant information from both sources and make more informed predictions about product prices.

- **Text Data Encoding and Sequence Padding**
  After tokenizing the text data, we convert the individual tokens (words) into numerical representations through word encoding. Each unique word in the vocabulary is assigned a unique integer index, effectively encoding the textual data into a numerical format that the neural network can process. By representing words numerically, we enable the model to perform mathematical operations on the input data, facilitating the learning of complex patterns and relationships.

  After that, we pad the token sequences to a fixed length of 150 tokens to ensure uniform input dimensions for the neural network model. Sequences longer than 150 tokens are truncated from the end, while shorter sequences are padded with zeros at the end. By padding sequences to a fixed length, we facilitate efficient batch processing during training while accommodating variability in the length of product titles and descriptions.

  While the actual maximum length of the title plus description in the dataset reached approximately 700 tokens, choosing this as our fixed sequence length would have led to significant padding for the majority of sequences. This excessive padding could potentially introduce uniformity in the input representations, as many sequences would consist mostly of zeros after padding. Such uniformity might hinder the model's ability to distinguish between different inputs and learn meaningful patterns from the data. That's why the choice of a maximum sequence length of 150 tokens strikes a balance between capturing essential information from the text data and preserving representation diversity.

Each preprocessing step is guided by the overarching goal of enhancing the quality and relevance of the input data for our neural network model. We carefully consider these steps to optimize the model's performance in predicting laptop accessory prices.

## 4.3 Model Architecture

In this subsection, we elucidate the neural network architecture employed for predicting laptop accessory prices based on their titles and descriptions. Each component and parameter choice is carefully justified to align with the task at hand and the characteristics of our dataset.

### 4.3.1 Description of the Neural Network Architecture:

Our chosen architecture comprises a bidirectional Long Short-Term Memory (LSTM) network, a type of recurrent neural network (RNN) known for its ability to capture sequential dependencies in data. The bidirectional aspect allows the model to consider both past and future contexts when making predictions, enabling it to capture nuanced relationships within the input sequences.

#### 4.3.2 Explanation of Layers:

- **Input Layer:**
  The input layer receives the concatenated title and description sequences as input. Each sequence is represented as a sequence of tokens, with the length determined by the maximum sequence length chosen during preprocessing.

- **Embedding Layer:**
  The embedding layer converts the input token sequences into dense vector representations. This transformation is crucial for the model to learn meaningful representations of words within the context of the prediction task. The embedding dimension, set to 100 in our case, determines the dimensionality of the dense vector space.

  The embedding dimension of 100 is chosen to strike a balance between capturing sufficient semantic information from the input tokens and avoiding overfitting, especially given the relatively small size of our dataset vocabulary (6775 words).

- **LSTM Layer:**
  The LSTM layer consists of bidirectional LSTM units that process the embedded token sequences. By leveraging the memory cells and gating mechanisms inherent to LSTM units, the model can effectively capture long-range dependencies in the input data.

  The bidirectional LSTM architecture is chosen to leverage the sequential nature of the input data and capture both past and future contexts in the product titles and descriptions. This enables the model to learn from the entire input sequence and make informed predictions about product prices.

  The number of LSTM units, set to 128 in our model, determines the capacity of the network to capture complex patterns in the textual data.The choice of 128 LSTM units balances model complexity with computational efficiency.

  While a larger number of units could potentially capture more intricate patterns in the data, it may also lead to overfitting, especially in the absence of a large training dataset. Conversely, a smaller number of units may limit the model's capacity to capture complex relationships. Therefore, 128 units are deemed suitable for our task, providing an adequate balance between model capacity and generalization.

- **Dense Layer:**
  The dense layer serves as the intermediate layer between the LSTM output and the final output layer. It applies a rectified linear unit (ReLU) activation function to introduce non-linearity into the network, facilitating the learning of complex relationships between features extracted by

the LSTM layer. ReLU was chosen for its simplicity and effectiveness in mitigating the vanishing gradient problem.

Again we have decided to utilize 64 units in the dense layer to strike a balance between model complexity and computational efficiency and to mitigate the risk of overfitting and promote better generalization to unseen data, ensuring that the model remains robust and efficient during training and inference.

- **Output Layer:**
  The output layer consists of a single neuron, responsible for predicting the price of the laptop accessory. Since price prediction is a regression task, the output layer does not employ an activation function, allowing the model to output continuous values representing predicted prices.

## 4.4 Model Training

In this subsection, we outline the process of training our neural network model for predicting laptop accessory prices. We detail the dataset splitting strategy, training process parameters, and the monitoring of training progress and performance metrics on the validation set.

### 4.4.1 Splitting the Dataset:

The dataset is split into three subsets: training, validation, and testing sets using a ratio of 80:10:10, respectively.This ratio is chosen to ensure a sufficient amount of data for training the model while still providing ample validation and testing samples to evaluate its performance accurately.

The training set is used to train the model, the validation set is employed to tune hyperparameters and monitor performance during training, and the testing set is utilized to evaluate the final performance of the trained model on unseen data. The dataset splitting ensures that the model's performance is evaluated on independent data, thereby providing a reliable estimate of its generalization ability.

#### 4.4.2 Training Process:

- **Batch Size:**
  Set to 32, the batch size determines the number of samples processed before updating the model's weights. A batch size of 32 strikes a balance between computational efficiency and training stability.

- **Epochs:**
  We train the model for 10 epochs, where an epoch represents one pass through the entire training dataset. This allows the model to iteratively refine its weights and improve performance over time to capture complex patterns in the data.

- **Optimizer Settings:**
  The Adam optimizer is employed with default settings, including a learning rate of 0.001. Adam is chosen for its effectiveness in training deep neural networks and adaptive adjustment of learning rates based on the gradient magnitudes of model parameters.

- **Loss Function:**
  We utilize the mean squared error (MSE) loss function for training our model. MSE measures the average squared difference between the predicted prices and the actual prices of laptop accessories in the training dataset. This loss function is well-suited for regression tasks and provides a smooth and differentiable objective for model optimization.

During training, we monitor the model's performance on the validation set to assess its generalization ability and prevent overfitting. Performance metrics such as mean squared error (MSE) and mean absolute error (MAE) are computed on the validation set after each epoch. By tracking these metrics, we can identify potential issues such as overfitting or poor convergence and adjust hyperparameters accordingly to optimize model performance.

#### 4.4.3 Training Progress Analysis

Training Progress Analysis provides insights into the performance of the neural network model during training. We analyze the trends of training loss (MSE) and mean absolute error (MAE) as well as validation loss (MSE) and mean absolute error (MAE) over the course of training epochs.

- **Training Loss and Mean Absolute Error (MAE):**
  The training loss (MSE) decreases steadily over the epochs, signifying the model's learning process and enhancement in predicting laptop accessory prices. Similarly, the mean absolute error (MAE) on the training set decreases, indicating that the model's predictions are converging towards the actual prices with training progression.

- **Validation Loss and Mean Absolute Error (MAE):**
  Initially, the validation loss (MSE) and mean absolute error (MAE) exhibit a decreasing trend, reflecting the model's ability to generalize well to unseen data. However, beyond a certain number of epochs, both metrics stabilize or show slight increases, suggesting potential overfitting to the training data. Despite this, the final validation loss (1486.41) and MAE (27.09) remain relatively low, indicating reasonable performance of the model on the validation set.

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 3941.1536 | 2769.0745 |
| 2 | 3058.5154 | 1872.2606 |
| 3 | 2730.6440 | 1620.5831 |
| 4 | 2676.9148 | 1635.8971 |
| 5 | 2665.6924 | 1654.4270 |
| 6 | 2671.7400 | 1688.1238 |
| 7 | 2662.6802 | 1660.9471 |
| 8 | 2666.4351 | 1626.2347 |
| 9 | 2611.1729 | 1580.2756 |
| 10 | 2437.9158 | 1486.4126 |

Table 1: Training and Validation Loss

## 4.5 Evaluation Results

In this subsection, we present the performance metrics obtained on the testing dataset and visualize the model's performance through residual plots. Performance metrics on the testing dataset provide insights into the model's effectiveness in predicting laptop accessory prices.

- **Test Loss:**
  The mean squared error (MSE) on the testing dataset is calculated to be 2644.96. This metric represents the average squared difference between the predicted prices and the actual prices of laptop accessories in the testing set.
- **Test MAE:**
  The mean absolute error (MAE) on the testing dataset is 27.31. This metric represents the average absolute difference between the predicted prices and the actual prices of laptop accessories in the testing set.

Figure 8 displays the residual plot generated from the model's predictions on the testing dataset. The x-axis represents the predicted prices, while the y-axis represents the residuals, which are the differences between the predicted

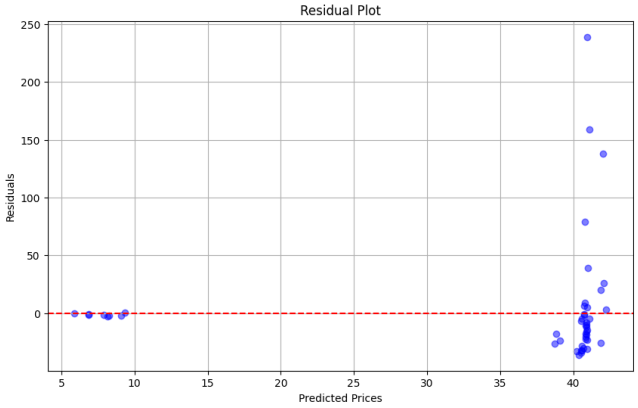



Figure 8: Residual Plot

prices and the actual prices. Residual analysis provides insights into the model's prediction errors by examining the differences between predicted and actual prices.

## 4.6 Discussion

### 4.6.1 Interpretation of Results

The evaluation results provide valuable insights into the effectiveness of our model for predicting product prices. The mean squared error (MSE) and mean absolute error (MAE) on the testing. However, it's essential to compare these results with our expectations and the characteristics of the price distribution in the dataset.

The price/value column statistics reveal a wide range of prices, with a mean of approximately $38.34 and a standard deviation of $50.89. The median price is $24.99, with prices ranging from $2.99 to $575.00. Considering this distribution, our model's test MAE of approximately $ 27.31 suggests that, on average, our predictions deviate by this amount from the actual prices this deviation is relatively significant. However, it aligns with our expectations, given the inherent variability and complexity of pricing in the laptop accessory market.

### 4.6.2 Limitations and Potential Areas for Improvement

However, it's crucial to acknowledge the limitations of our approach and identify potential areas for improvement. The relatively high MAE indicates room for enhancement in the model's predictive accuracy. Additionally, the absence of pre-trained embeddings and the simplicity of the model architecture may limit

its ability to capture intricate relationships in the data.

Furthermore, the presence of missing values and the small size of the dataset that doesn't capture a wide range of products with different natures could have impacted the model's performance. In future iterations, addressing these limitations by incorporating more extensive and cleaner datasets, exploring advanced model architectures, fine-tuning hyperparameters, and integrating pre-trained language models, such as BERT could lead to improved performance and generalization.

### 4.6.3 Conclusion

In conclusion, while our model's performance may not be optimal, considering the constraints and challenges inherent in our approach, it represents a promising starting point for further refinement and exploration. By iteratively improving the model and addressing its limitations, we aim to develop a more robust and accurate question-answering system that empowers users to navigate the dynamic landscape of laptop accessories with confidence and ease.

# 5 Pretrained Model Fine-tuning and Evaluation

## 5.1 Introduction

In this section, we detail the development and evaluation of a pretrained model for predicting the prices of laptop accessories, with a focus on fine-tuning. Building upon the foundation laid in previous milestones, we extend our approach by incorporating a pretrained model, specifically BERT, and fine-tuning it to suit our task. This section provides an overview of the methodology employed, including

- preprocessing steps
- model architecture
- training procedures

The objectives of this section includes

- resenting training and evaluation results.
- analyzing the differences between the pretrained model approach and the simple model approach
- discussing the findings and implications of our work.

## 5.2 Model Selection

We chose the BERT (Bidirectional Encoder Representations from Transformers) model as the cornerstone of our price prediction framework for laptop accessories. Several factors guided our decision :

- **Contextual Understanding:** BERT excels at understanding words within the context they're used, allowing it to grasp the nuanced relationships between product descriptions and titles. This contextual understanding is crucial for accurately predicting prices based on textual inputs.
- **Pretrained Knowledge:** Leveraging BERT's pretrained knowledge, acquired from vast amounts of text data, provides a solid foundation for our price prediction task. By building upon this wealth of linguistic information, BERT can effectively capture the semantic nuances inherent in product descriptions and titles.

- **Fine-Tuning Capability:** BERT's architecture allows for fine-tuning on specific downstream tasks, making it adaptable to our price prediction domain. Through fine-tuning, we can adjust BERT's parameters to better align with the characteristics of our dataset, thereby improving its predictive performance.
- **Proven Performance:** BERT has demonstrated state-of-the-art performance across a range of natural language processing tasks, showcasing its effectiveness in understanding and processing textual data. This track record instills confidence in BERT's ability to accurately predict prices for laptop accessories.
- **Prediction Focus:** BERT fits our needs perfectly because we're focused on making predictions. We don't need a model that generates text; instead, we need one that understands and learns from the data to make accurate predictions. BERT, being an encoder model without a decoder, aligns well with our prediction-focused task.

In the subsequent sections, we will go through the details of fine-tuning BERT for our specific domain and present the outcomes of our experiments.

## 5.3 Preprocessing

In this section, we outline the preprocessing steps undertaken to prepare the input data for training our price prediction model. The preprocessing is divided into two parts: handling the title and description text data, and encoding the brand information.

### 5.3.1 Title and Description Processing

The title and description text data undergo several preprocessing steps to facilitate effective utilization by the model. The main difference in the preprocessing stage compared to the previous section lies in the utilization of the BERT tokenizer for most of the processing steps:

1. **Concatenation:** Similar to the previous section, the title and description of each product are concatenated into a single text string to provide comprehensive input to the model.
2. **Tokenization:** Instead of manually tokenizing the text, we employ the BERT tokenizer to tokenize the concatenated text string. This ensures consistent tokenization using the same vocabulary as the pretrained BERT model, capturing fine-grained linguistic features.

3. **Padding and Truncation:** The tokenized sequences are padded or truncated to ensure uniform length across all input samples using the BERT tokenizer's built-in functionality. The tokenizer adds special tokens to sequences shorter than a specified length, while truncation removes excess tokens from sequences longer than the specified length. This ensures consistency in input size, which is essential for batch processing.

4. **Token Encoding:** The tokenized sequences are encoded using the BERT tokenizer, generating input features consisting of input IDs and attention masks. Input IDs represent the numerical IDs assigned to each token, while attention masks indicate which tokens are actual data and which are padding tokens.

### 5.3.2 Brand Encoding

The categorical brand labels are encoded using label encoding. Each unique brand is assigned a numerical label, enabling the model to process brand information numerically.

By preprocessing the title, description, and brand information in this manner, we prepare the input data in a format suitable for consumption by the model.

## 5.4 Model Architecture

In this section, we explore the architecture of our price prediction model, detailing how we utilized BERT (Bidirectional Encoder Representations from Transformers) and integrated the input data into its framework. We also provide a concise overview of the BERT architecture for context.

### 5.4.1 BERT Architecture

BERT is a transformer-based model architecture that has revolutionized natural language processing tasks. Its architecture consists of multiple layers of bidirectional transformer encoders. Each encoder layer employs self-attention mechanisms to capture contextual information from both left and right contexts of each word in a sequence. The transformer architecture facilitates parallel processing of input sequences, enabling efficient training and inference.

Key components of the BERT architecture include:

- **Input Embeddings:** BERT initially receives input sequences as token embeddings, segment embeddings, and positional embeddings. Token embeddings represent the semantic meaning of individual tokens, segment embeddings distinguish between different segments of input sequences (e.g., sentences), and positional embeddings encode the positional information of tokens within sequences.
- **Transformer Encoder Layers:** BERT comprises multiple transformer encoder layers stacked on top of each other. Each encoder layer consists of self-attention mechanisms followed by feedforward neural networks. The self-attention mechanism allows the model to attend to relevant tokens in the input sequence, capturing dependencies between words and contextual information.
- **Final Pooling Layer:** BERT typically employs a pooling strategy, where the output representations of the special [CLS] token from the last encoder layer are aggregated to generate a fixed-size representation of the input sequence. This pooled representation is then used for downstream tasks such as classification or regression.

### 5.4.2 Utilizing BERT for Price Prediction

In our price prediction model, we harness the power of BERT by fine-tuning a pretrained BERT model on our dataset. The architecture of our model includes the following components:

- **BERT Pretrained Model Initialization:** We initialize our model with a pretrained BERT model, specifically 'bert-base-uncased'. This pretrained BERT model has been trained on a vast corpus of text data and learned rich semantic representations, making it well-suited for various natural language processing tasks.

- **Fine-Tuning Layers Addition:**
  - On top of the pretrained BERT model, we add additional layers specifically tailored to our price prediction task. These layers typically include dense neural network layers that capture higher-level features learned from the BERT representations.
  - In our architecture, we introduce a dropout layer with a dropout rate of 0.1 to mitigate overfitting.
  - Following the dropout layer, we employ a fully connected layer (fc1) with 512 units, responsible for combining BERT's output with the brand embedding. The input dimension of this layer is the sum of the BERT output dimension (768) and the dimension of the brand embedding.
  - Additionally, we include another fully connected layer (fc2) with 1 unit for regression, outputting the predicted price.
- **Integration of Input Data:** The input data, consisting of concatenated title and description sequences along with encoded brand information, is fed into the BERT model during both training and inference phases. To accommodate the brand information, we augment BERT's output with a brand embedding. This combined representation is then passed through the additional layers for further processing and price prediction.

This approach allows us to leverage the strengths of BERT in understanding natural language while tailoring the model to our specific task requirements.

## 5.5 Training Phase

In the training phase of our price prediction model, we employed specific parameters and strategies to optimize model performance while considering computational limitations. Let's discuss each parameter and strategy:

- **Batch Size (batch_size = 16):** We chose a relatively small batch size of 16 due to limited RAM resources. A smaller batch size allows us to fit more data into memory during training, reducing the risk of memory overflow. However, smaller batch sizes may lead to slower convergence and less stable training compared to larger batch sizes. Nonetheless, given our hardware constraints, a batch size of 16 was deemed appropriate.
- **Learning Rate (learning_rate = 2e-5):** The learning rate determines the step size at which the model parameters are updated during optimization. We opted for a learning rate of 2e-5, which is commonly used in fine-tuning pretrained transformer models like BERT. This learning rate strikes a balance between fast convergence and stability during training.

- **Number of Epochs (num_epochs = 10):** Due to limited computational power, we restricted the number of epochs to 10. The number of epochs defines the number of times the entire training dataset is passed through the model during training. With a limited number of epochs, we aimed to achieve reasonable performance improvements while avoiding excessive training times. However, it's important to note that model convergence may not be fully achieved within this limited epoch range, potentially affecting the final performance.

Regarding our dataset split, we allocated 80% of the data for training, 10% for validation, and 10% for testing. This split allows us to evaluate the model's performance on unseen data while ensuring sufficient data for both training and validation.

## 5.6 Results

In this section, we present the results of our price prediction model. We evaluate the model's performance using the Mean Squared Error (MSE) metric, which measures the average squared difference between the predicted and actual prices.

We chose MSE as our performance metric due to its suitability for regression tasks and its ability to quantify the magnitude of prediction errors. By minimizing MSE during training, we aim to optimize the model's ability to make accurate price predictions.

### 5.6.1 Training Results

Here are the training results showing the loss values for each epoch and the corresponding validation loss:

| Epoch | Training Loss (MSE) | Validation Loss (MSE) |
|---|---|---|
| 1 | 4191.5744 | 4418.9487 |
| 2 | 4011.9904 | 4201.7373 |
| 3 | 3843.6064 | 4000.6934 |
| 4 | 3686.5248 | 3807.8892 |
| 5 | 3541.4464 | 3625.7041 |
| 6 | 3404.0112 | 3457.5764 |
| 7 | 3285.9200 | 3305.0376 |
| 8 | 3163.2736 | 3119.1428 |
| 9 | 3018.6064 | 2964.7173 |
| 10 | 2920.6848 | 2830.8445 |

Table 2: Training and Validation Loss (MSE)

### 5.6.2 Validation Residual Plot

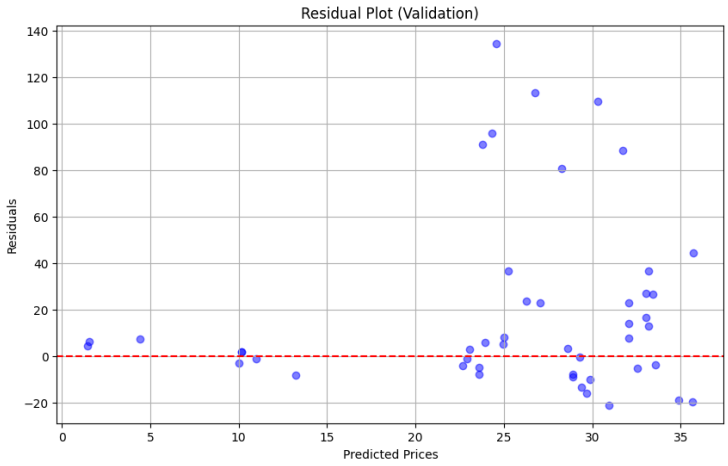



Figure 9: Validation Residual Plot

### 5.6.3 Testing Results

The test loss obtained from the testing dataset is 2758.7461

### 5.6.4 Test Residual Plot

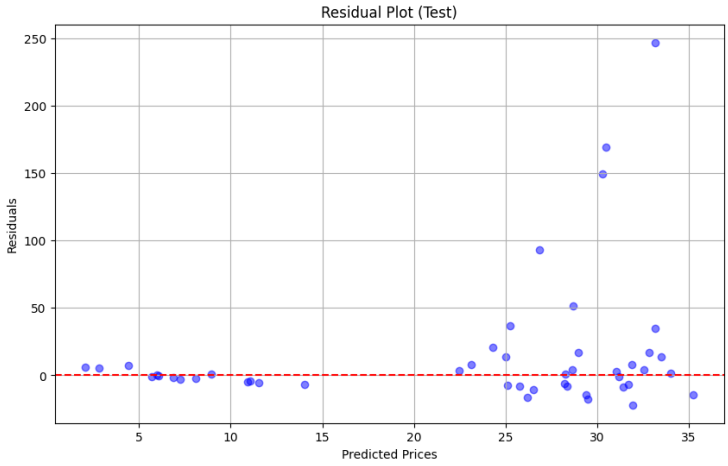



Figure 10: Test Residual Plot

These results provide insights into the performance of our price prediction model on both the training and testing datasets. While the training results demonstrate the model's learning progression and validation performance, the testing results offer an evaluation of the model's generalization capability on unseen data. Additionally, the residual plots visually depict the distribution of errors in the model's predictions, aiding in the assessment of model accuracy and reliability.

## 5.7 Model Comparison and Analysis

In this section, we compare the performance of the basic model developed in Milestone 2 with the pretrained model implemented in Milestone 3. We analyze the results of both models to understand the differences and implications.

### 5.7.1 Analysis

From the comparison, we observe that the basic model consistently outperforms the pretrained model in terms of training loss across all epochs. The basic model achieves lower training loss values, indicating better convergence and fitting to the training data. In contrast, the pretrained model exhibits higher training loss values, suggesting less effective learning during training.

The test loss values suggest that Milestone 2 achieved a slightly lower mean squared error (MSE) compared to Milestone 3. This indicates that the basic model performed marginally better in terms of predicting the prices of laptop accessories based on their title and description alone.

Furthermore, we examine the residual plots derived from the test predictions of both models. The residual plot of Milestone 2 reveals a concentration of predictions within relatively narrow intervals, notably between 5-10 and 35-45. Conversely, the residual plot of Milestone 3 depicts a broader array of predicted prices, spanning most of the price spectrum from 0 to 35. This discrepancy in the predicted price distribution implies that Milestone 3, leveraging the pretrained model and integrating brand information, generated predictions covering a wider range of potential price.

Overall, while Milestone 2 demonstrates competitive performance in terms of test loss, Milestone 3 offers a more comprehensive and nuanced approach to price prediction, capturing a wider spectrum of potential price ranges and leveraging brand information to enhance predictive accuracy.

### 5.7.2 Reasoning

The pretrained model utilizes a more complex architecture with the inclusion of a pretrained BERT encoder. While BERT is capable of capturing complex

patterns and semantic representations from text data, its generic nature may not always align perfectly with the specific task of price prediction.

The use of a pretrained model in Milestone 3 may have introduced additional complexities and nuances that the model struggled to adapt to, leading to slower convergence and higher training losses. Additionally, the architecture of the pretrained model, particularly its deeper layers and larger parameter space, may have posed challenges in training, especially given the limited computational resources available.

Moreover, the preprocessing differences between the two milestones could have influenced the training outcomes. In Milestone 3, the utilization of a BERT tokenizer and the inclusion of brand information introduced additional features and complexities to the input data, potentially affecting the model's learning dynamics and convergence behavior.

## 5.8 Conlusion

The differences in training results highlight the trade-offs and considerations involved in choosing between a basic model and a pretrained model for our task. While the pretrained model offers the advantage of leveraging prelearned representations and potentially capturing richer semantic information, it also poses challenges in training and requires careful tuning and optimization to achieve optimal performance compared to simpler models that are designed specifically to do a certain task like price prediction in our case.

Furthermore, the higher training losses observed in Milestone 3 highlight the importance of thorough experimentation and analysis when incorporating complex models into the workflow. Future iterations may benefit from exploring alternative architectures, optimization strategies, or preprocessing techniques to mitigate these challenges and enhance model performance.