Product Attribute Extraction and Product Listing Analysis from E-commerce Websites

Rishi Dey Chowdhury, Arghya Sarkar, Mrinmoy Banik, Prisha Reddy Bobbili

Indian Statistical Institute, Kolkata

Abstract

Product attributes, such as brand name, size, weight, or dimension, are critical in e-commerce as they help customers find and select the right product for their needs. However, obtaining, adding, and maintaining these values is extremely labour intensive, especially on larger sites. Therefore we ventured to use 1. SOTA transformer based models like Google's BERT & Facebook's RoBERT, to make Product Attribute Extraction (PAE) and 2. Train question-answering model to predict brand name based on product description. For PAE, we used Bi-LSTM model to create chunkings of proper nouns from description and formulated relationship with the associated non-noun phrases. Using this Parts-of-speech tagging, we identified clusters and based on these we came up with an unsupervised technique to relate a product's attribute name with its attribute value. For brand detection task we modified a question-answering transformer model (DistilBERT) to answer the question 'what is the brand name of this product?' based on the descriptions. After finetuning the model we achieved a desirable accuracy. The code for this work is available at here.

Introduction

Product attribute extraction refers to the process of automatically identifying and extracting specific attributes or features of a product from textual data, such as product descriptions, reviews, or specifications. These attributes can include various characteristics of a product, such as color, size, brand, material, weight, dimensions, and more. Product listing analysis involves analyzing various elements of a product listing, including the title, description, images, attributes, pricing, and other relevant information; the purpose of which is to assess the quality, completeness, and effectiveness of product listings and identify areas for improvement.

In the world of online shopping, consumers heavily rely on product information provided by e-commerce websites. Accurate and detailed attribute extraction helps in improving the search experience, filtering options, and product comparisons. It enables customers to find products that match their specific requirements and make informed purchasing decisions. It also makes way for better categorization and classification of products by helping in creating structured product catalogs, enabling efficient inventory management. At the same time, it can provide valuable insights into product landscape for the businesses. Analyzing attributes across a range of products can help identify trends, market gaps, and opportunities for product differentiation.

Background

Evolution of PAE

In the late 1990s and early 2000s, as e-commerce gained popularity, companies started building online platforms to sell products. Initially, product data was often manually entered by vendors, resulting in inconsistencies and lack of structure. Thus, a need for automated methods to extract and organize product attributes arose.

In the early stages, rule-based systems were developed to extract product attributes. These systems used predefined rules and patterns to identify and extract specific information from product descriptions. However, they had limited flexibility and struggled with handling variations and new patterns.

With the advancement of machine learning techniques, researchers and companies started exploring the use of algorithms to automatically extract product attributes. Supervised learning algorithms, such as Support Vector Machines (SVM) and Random Forests, were used to train models on labeled data to identify attributes. NER (Named Entity Recognition), a subtask of NLP. techniques have been widely used for product attribute extraction, leveraging annotated datasets and models trained on large corpora.

In recent years, there has been a growing interest in leveraging ontologies and knowledge graphs for

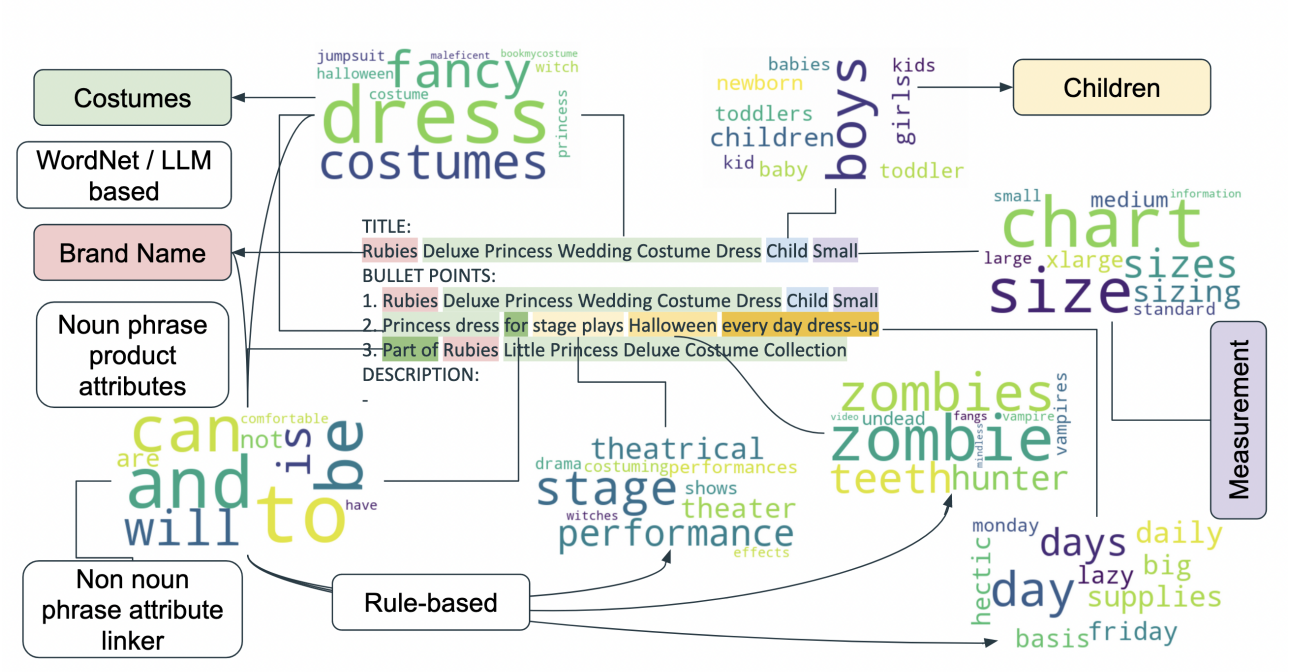

Figure 1: *Detection of all possible product attribute name - value pairs unsupervisedly and supervised brand name detection from product title, bullet points and descriptions listed on Amazon India website using our method.*

product attribute extraction. These approaches utilize structured knowledge representations to define product attributes and their relationships, enabling more accurate and comprehensive extraction.

Related Work

Rule - based Extraction This involves defining specific rules or patterns to identify and extract relevant attributes from product listings. These rules are typically based on the structure, format, and textual patterns observed in the product data.
Some of the key aspects of rule - based extraction are pattern matching, key-word based extraction, contextual rules, rule combination and prioritization, rule refinement and iterative improvement.
Rule-based extraction heavily relies on the defined rules, and any changes in the structure or format of the product listings may require updates to the rules. It may not handle variations or exceptions well without explicit rules for each case. Additionally, rule-based extraction may struggle with noisy or unstructured data.This proved to be a drawback of this method.

Bootstrapping This involves an iterative process of training and refining machine learning models using a limited amount of labeled data and then using these models to automatically label more data for further model improvement.
Some of the key aspects of bootstrapping are initial seed data, model training, model application, human verification, expansion of labeled data, model retraining, and repetition.

Noun Phrase Clustering Noun phrase clustering helps identify and group similar attributes within product listings, providing a structured representation of the information. It enables the extraction of relevant attributes by identifying clusters of related noun phrases and assigning attribute labels to those clusters. This approach helps in organizing and categorizing the product attributes, making it easier to analyze and understand the product listings in e-commerce websites.
Some of the key aspects of noun phrase clustering are noun phrase extraction, similarity calculation, clustering algorithm, cluster labeling, cluster refinement, attribute extraction.

N - gram Analysis This involves the extraction and analysis of consecutive sequences of N words from the product listings. It analysis helps identify patterns and recurring sequences of words within product listings and thus enables the discovery of attribute-related phrases.
The choice of the N value depends on the specific context and the desired level of detail in attribute extraction. Higher values of N capture longer sequences of words but may result in sparser data, while lower values of N capture shorter sequences but may miss out on more complex attribute relationships. The optimal value of N is usually obtained by experimenting and analysis of the

corresponding dataset.
Some of the key aspects of N gram analysis are N-gram extraction, frequency analysis, attribute identification, N-gram filtering, attribute extraction.

Supervised Text-based Feature Extraction This involves training a machine learning model to extract relevant features or attributes from the textual data in a supervised manner.
In a deeper context, supervised text feature extraction allows for the automatic extraction of relevant attributes from product listings, leveraging labeled training data to train a model that learns the patterns and relationships between textual features and attribute information.
Some of the key aspects of supervised text-based feature extraction are data annotation, feature engineering, model training, label prediction, post processing and evaluation.
The effectiveness of supervised text feature extraction heavily relies on the quality and representativeness of the annotated training data. The availability of a diverse and accurately labeled dataset is crucial for training a reliable and robust model.

Unsupervised Text-based Feature Extraction Unlike supervised methods that rely on labeled training data, unsupervised approaches aim to automatically extract relevant features or attributes from textual data without the need for explicit annotations.
Some of the key aspects of unsupervised text-based feature extraction are text processing, vectorization, dimentionality reduction, clustering, cluster analysis, and attribute extraction.
Basically, unsupervised text feature extraction offers a data-driven approach to discovering meaningful patterns and attributes in product listings without the need for explicit annotations. It enables the exploration and identification of latent attributes within the data, allowing for a more comprehensive understanding of the products and their descriptions.
However, it's important to note that the interpretation and validation of the extracted attributes may require human expert review and domain knowledge to ensure their relevance and accuracy.

Our Methodology

Word Embedding Generation

Word embedding is a powerful approach that transforms text data into a numerical representation, enabling machines to understand and analyze natural language. By capturing semantic and syntactic relationships between words, word embedding facilitates various NLP tasks such as language translation, sentiment analysis, document classification, and information retrieval.

Word embedding techniques can be categorized into conventional, distributional, and contextual word embedding models. Conventional models (BOW, n-gram, TF-IDF) deals with frequencies of words or n-grams in documents and tries to find importance of words or phrases on the documents. Major limitation of these models are that sentence's contextual meanings are not analysed so it cannot handle unseen words.

In the distributional representation model, the context in which a word is used determines its meaning in a sentence. Distributional models predict semantic similarity based on the similarity of observable contexts. A distributional model represents a word or phrase in context, but a Vector Space Model (VSM) represents meaning in a high-dimensional space [1]. VSM suffers due to the curse of dimensionality resulting from a relatively sparse vector space with a larger dataset.

The conventional and distributional representation approaches learn static word embedding. After training, each word representation is identified. The semantic meaning of the word polysemy can vary depending on the context. Understanding the actual context is required for most downstream tasks in natural language processing. For example, "apple" is a fruit but usually refers to a firm in technical articles. The vectors of words in the contextualized word embedding can be modified according to the input contexts utilizing neural language models.

Among contextual representation models BERT is one of the most popular model. In 2019 by researchers at Google AI Language introduced BERT (Bidirectional Encoder Representations from Transformers)[2]. BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. Through training BERT uses Masked LM and Next Sentence Prediction to minimize the combined loss function of the two strategies.

Recently, SOTA performance is achieved through Sentence-Transformers which trains BERT with contrastive learning on pair-wise similar sentences (sometimes also containing negative sentences). The model learns to map similar sentences together either by minimizing Cosine Distance or by Triplet Loss. These models are able to generate high quality sentence or word level embeddings.

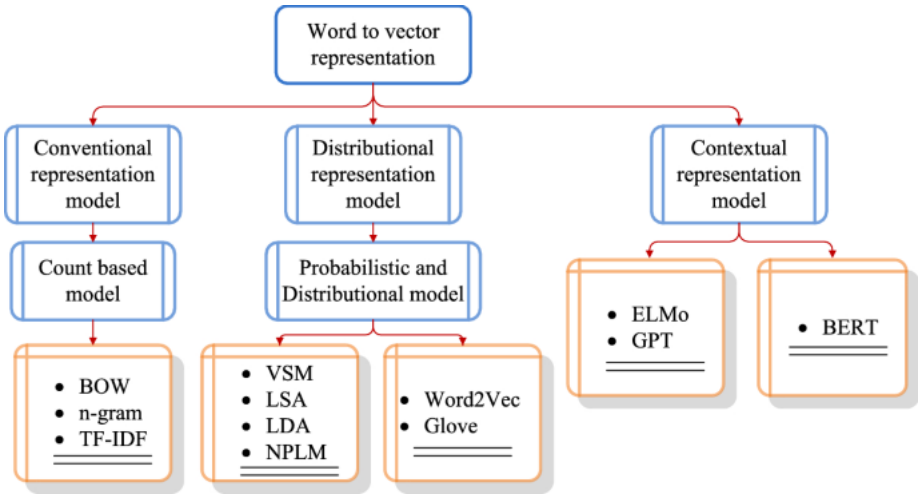

Figure 2: *Various word embedding models. Source: Springer*

Masked Language Model

Masked-language modeling (MLM), a unique training approach that gave BERT unparalleled success in NLP. Before feeding the input tokens to BERT we replace 15% of tokens by a [mask] token. So we're actually inputting an incomplete sentence and asking BERT to complete it for us. Although BERT does not know meaning of the sentence, but it does know that given linguistic patterns, and the context of these words, and from this it can give the most likely answer. MLMs have proven to be highly effective in a wide range of natural language processing tasks, including text generation, machine translation, sentiment analysis, and question answering. We use them to fine-tune on our product dataset to improve feature representation of the tokens that appear in product datasets and better captures the contextual dependence.

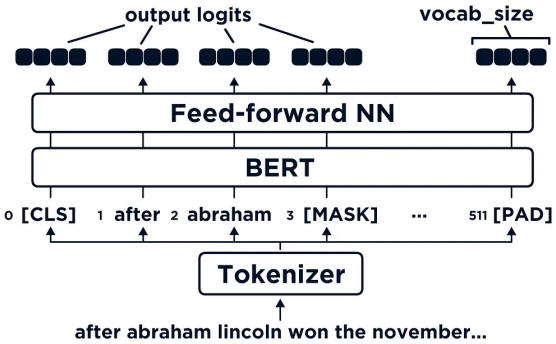

Figure 3: *Masked language model. In this image, before passing our tokens into BERT - we have masked the lincoln token, replacing it with [MASK]. Source: Towardsdatascience*

Chunking

Chunking is the method of breaking a sentence into disjoint parts based on some property. With our primary focus is extracting tuples of the form $(attribute, value)$. It is observed that in product titles, descriptions and bullet points, Noun Phrases form the values of various attributes and identifying these noun phrases gives us candidate values for attributes. But one major drawback that still remains is to relate these values with appropriate attributes. This is where, we try to analyze the relationship of noun phrases with surrounding non noun phrases i.e. specially verb phrases or cluster of similar noun phrases to come up with appropriate attributes.

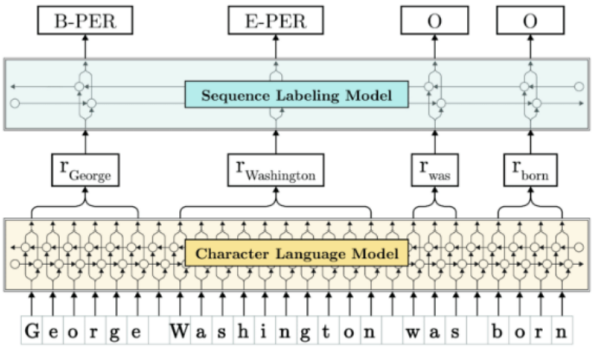

Figure 4: *Illustration of character-based sequence-to-sequence tagging using BI-LSTM.*

The Chunker used for the purpose is a BiLSTM (Bidirectional Long Short Term Memory) + CRF (Conditional Random Field) which is priorly trained on labelled data. The LSTM uses the following equations to model the input and output parametrized by

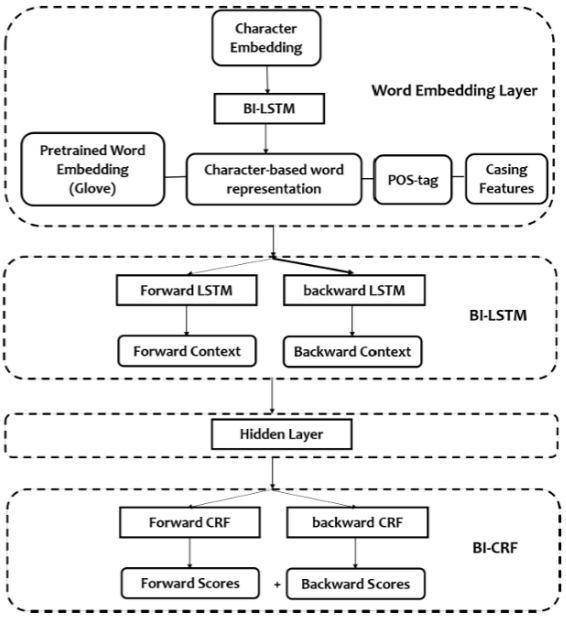

Figure 5: *Architecture of the Sentence Chunker Network [3]*

the weight W.

$$
\begin{aligned}
i_t &= \sigma(W_i[x_t; h_{t-1}] + b_i) \\
f_t &= \sigma(W_f[x_t; h_{t-1}] + b_f) \\
o_t &= \sigma(W_o[x_t; h_{t-1}] + b_o) \\
\tilde{c} &= tanh(W_c[x_t; h_{t-1}] + b_c) \\
c_t &= f_t c_{t-1} + i_t \tilde{c_t}) \\
h_t &= o_t tanh(c_t)
\end{aligned}
$$

In our case, since we use a Bi-LSTM, we gather the context from both sides $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$ before passing it to a the Linear and CRF Layer.

Now, that we have captured the contextual information in each word and have generated hidden features. We use this features for the purpose of token-level classification for weather we are at the beginning, inside or outside of a phrase and what type of phrase it is. This is modelled through a CRF which is a variation Markov Random Field where all the clique potentials ϕ_c, $1 \leq c \leq C$ are conditioned on input features. Given the labels $y = (y_1, \ldots, y_T)$ and Bi-LSTM hidden features $x = (x_1, \ldots, x_T)$, the CRF models the distribution as follows,

$$p(y|x) = \frac{1}{Z(x)} \prod_c \phi_c(y, x)$$

where Z(x) is a normalization function

$$Z(x) = \sum_y \prod_c \phi_c(y_c, x)$$

As the hidden layer on top of the BI-LSTM produces the score matrix P for a given sequence, the CRF layer learns only the transition probability of the output labels, A. The input features of observed tokens $\mathbf{x}$ in the clique $\phi_c(x, y)$ is the score matrix P_{i,y_i} learned by the hidden layer. For the given sequence of predictions $y = \{y_1, , y_T\}$, the probability score including the start and end tag, y_0 and $y_T + 1$ introduced in the inference algorithm is defined as

$$S(x, y) = \sum_{i=0}^{T} A(y_i, y_{i+1}) + \sum_{i=1}^{T} P_{i,y_i}$$

The probability for the sequence y is given by

$$p(y|x) = \frac{e^{S(x,y)}}{\sum_{y' \in \mathbf{y}} e^{S(x,y')}}$$

During training, maximum likelihood of the probability of correct sequences in the training are maximized. The final output tag sequence is decided based on the maximum score given by

$$y^* = \underset{y' \in \mathbf{y}}{argmax} S(x, y')$$

.

Using a pretrained English chunker model, details in the Implementation section, we convert:

- Product Title $\longrightarrow \{(p_1, l_1), \ldots (p_1, l_1)\}$
- Product Bullet Points $\longrightarrow \{(p_1, l_1), \ldots (p_1, l_1)\}$
- Product Description $\longrightarrow \{(p_1, l_1), \ldots (p_1, l_1)\}$

where p_i is the phrase and l_i is the corresponding label i.e. noun phrase and non noun phrase.

For clustering over non noun phrases, we used a rule-based strategy to simply consider the noun phrases following the non noun phrase to be the attribute value for the attribute

Topic Modelling

After chunking all the product titles, bullet points and descriptions, we are left with a set of noun phrases, which are possible values of various attributes for a product and non-noun phrases. Now, to consolidate and decipher the attribute of these candidate values we perform topic modelling for each product type and when the products under a product type are only a few, we resort to using more products from other categories to help identify semantically similar clusters. After identifying the clusters, for each product we get output of the form $\{(C_i, A_i)\}_{i=1}^{N}$, where N is the number of noun phrase appearing in the product details and the A_i is an attribute value corresponding to the attribute which represents the semantically similar noun phrases in the cluster C_i.

To capture semantic meaning of all the candidate attribute values i.e. noun phrases, we used 3 pretrained models to generate word embeddings for the attribute values:

- BERT: Trained using MLM on Large English Corpus
- Fine-tuned BERT: BERT is further trained on the product dataset
- Sentence Transformer: Trained on positive and negative pair of sentences to bring sentences with semantically close meaning closer.

To perform any sort of clustering on the large word embeddings is computation-intensive. Moreover, they contain redundancy and hence we resort to first reducing the embedding dimension of the words. Although Principle Component Analysis (PCA) can be used for this, we use Uniform Manifold Approximation and Projection (UMAP) to perform dimensionality reduction for its better local structure preserving property compared to PCA.

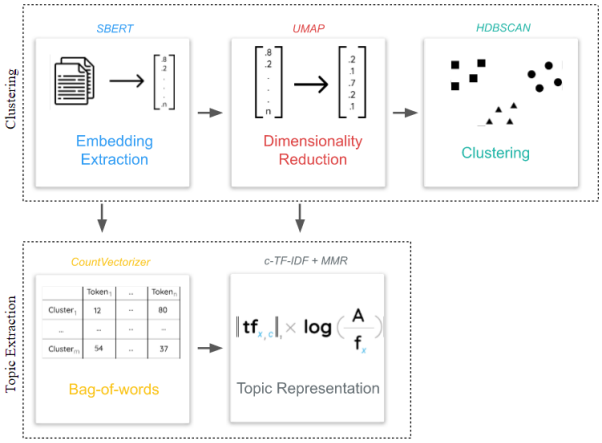

Figure 6: *Topic Modelling Pipeline*

After having reduced our embeddings, we leverage a density-based clustering technique, HDBSCAN. It can find clusters of different shapes and has the nice feature of identifying outliers where possible. As a result, we do not force documents into a cluster where they might not belong. This improves the resulting topic representation.

From the pool of all words, we generate a Bag-of-words representation and count the number of occurrence of each word in each cluster to come up with topic representation. By using a bag-of-words representation, no assumption is made concerning the structure of the clusters. Moreover, the bag-of-words representation is L_1-normalized to account for clusters that have different sizes.

From the generated bag-of-words representation, we want to know what makes one cluster different from another (Which words are typical for cluster 1 and not so much for all other clusters?). To solve this, we need to modify TF-IDF such that it considers topics (i.e., clusters) instead of documents.

When we apply TF-IDF on a set of documents, we actually compare the importance of words between documents. Here we instead treat all documents in a single category (e.g., a cluster) as a single document and then apply TF-IDF. This give importance scores for words within a cluster. The more important words are within a cluster, the more it is representative of that topic. In other words, if we extract the most important words per cluster, we get descriptions of topics! This model is called class-based TF-IDF (c-TF-IDF). For a term x within class c, c-TF-IDF is,

$$W_{x,c} = ||tf_{x,c}|| \cdot \log(1 + \frac{A}{f_x})$$

Where $tf_{x,c}$ is frequency of word x in class c, A is average no of word per class, f_x is frequency of word x across all classes.

Based on this weighted c-TF-IDF words representing each topic, we may generate topic label based on Large Language Models or WordNet Synsets.

Brand Name Extraction

We first used adaptive Skip-gram (AdaGram) to encode the combined product description (Title+Description+Bullet points) into a vector space. The original Skip-gram model [4] is formulated as a set of grouped word prediction tasks. Each task consists of prediction of a word v given a word w using correspondingly their output and input representations

$$p(v|w,\theta) = \frac{\exp(in_w^t out_v)}{\sum_{v'=1}^{V} \exp(in_w^t out_{v'})},$$

Where $\theta = \{in, out\}$ is global parameter consisting of input representation (one-odd encoding) and output representation. We multiply all such probabilities to get a likelihood function and we maximize it wrt θ. But in adaptive skip-gram for each word we use multiple input representation to capture the semantics of all possible word meanings. To identify the

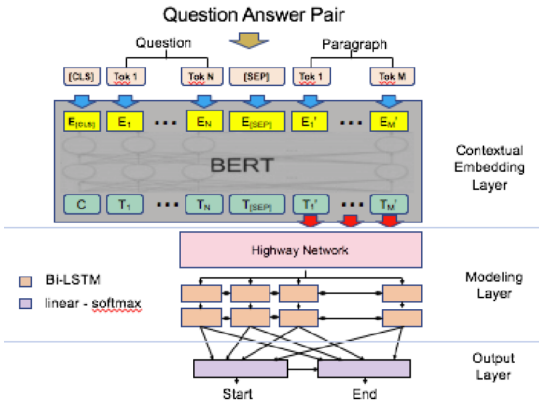

Figure 7: *BERT based question answering model to predict start and end tokens of the brand name*

specific meaning a word is being used for we use a

Bayesian nonparametric approach. For this reason we introduce latent variable z that encodes the index of active meaning which takes the prior distribution

$$p(z = k|w, \beta) = \beta_{wk} \prod_{r=1}^{k-1} (1 - \beta_{wr})$$

$$\beta_{wk}|\alpha \sim Beta(1, \alpha), k = 1, \ldots$$

Combining all parts together we may write the AdaGram model as follows:

$$p(Y, Z, \beta|X, \alpha, \theta) = \prod_{w=1}^{V} \prod_{k=1}^{\infty} p(\beta_{wk}|\alpha) \prod_{i=1}^{N} p(z_i|x_i, \beta) \prod_{j=1}^{C} p(y_{ij}|z_i, x_i, \theta),$$

where $Z = \{z_i\}_{i=1}^{N}$ is a set of senses for all the words. Using this unsupervised technique we find a word embedding for the data.

This question answering BERT model depicted in figure 7 is trained to answer the question 'What is the brand name of the product?' and works in the following way: maps sentences to tokens and tokens to embeddings, then the BERT Encoder transformer injects all the contexts that it has learned during it's fine tuning to create the final sequence embeddings. These embeddings can then be passed through two-layer Highway Network to transform each hidden vector T_i. The transformation is applied twice and each time with distinct learnable parameters. ,which means we apply the above transformation twice, each time using distinct learnable parameters. By introducing the Highway Network we hope to refine the BERT embedded representation with the gating mechanism and enable to the LSTM structure with bigger step transition depth to optimize the training procedures.

$$x_p = ReLU(W_p T_i + b_p)$$
$$x_{gate} = \sigma(W_{gate} T_i + b_{gate})$$
$$x_{highway} = x_{gate} \odot x_p + (1 - x_{gate}) \odot T_i$$

We predict the answer start and end probability distributions independently with two heads, each performing a linear down-projection followed by softmax.

Transfer learning

Transfer learning is a common practice in the literature when dealing with models that require extensive self-supervised training, such as word embedding and other similar techniques. These models are typically trained on large-scale unlabeled datasets using unsupervised or supervised learning approaches.

Word embedding models, like Word2Vec, GloVe, and BERT, typically demand extensive self-supervised training on large-scale unlabeled datasets to learn meaningful representations of words. This training involves predicting word contexts or co-occurrences to capture semantic relationships between words. However, training these models from scratch on specific datasets can be resource-intensive and time-consuming. For these reason these approach is almost unusable for our study.

To overcome this challenge, transfer learning is commonly employed in the literature. Researchers utilize pre-trained word embedding models, which have been trained on massive corporations, and fine-tune them on their target datasets. By leveraging the knowledge and representations acquired during the initial training, transfer learning enables the adaptation of these models to specific tasks and domains. This approach reduces the need for extensive training from scratch, accelerates process, and allows for effective utilization of limited labeled data. As a result, transfer learning and fine-tuning of pre-trained word embedding models have become a usual practice in the literature, empowering researchers to achieve better performance and efficiency in various natural language processing applications. In our case we take various pre-trained model like BERT and DistilBERT as base pre-trained models and fine tune them on our product datasets.

Dataset Overview

In the context of this report, we collected various datasets to support our analysis and findings. These datasets were collected mainly through web scraping and consisted primarily of unstructured data, specifically textual format. The 2 prominently used data sets are described below.

Dataset 1

Title : Amazon ML Challenge 2023
Source : Kaggle (Web-scraping)
Data Description :
The dataset comprises of 6 columns - Product ID, Product Title, Bullet Points mentioned for the Product, Description of the Product, Product Type ID, Length of the Product.
The important variables among the 6 are Product Title, Product type ID, Bullet points mentioned for the Product and Description of the Product.
The size of this data is 2.2 million.

Product titles are short, to the point, but do not provide all the features or uses of the product.That is where bullet points and description shine in listing features and uses of the product. Product type ID is the differentiating factor.

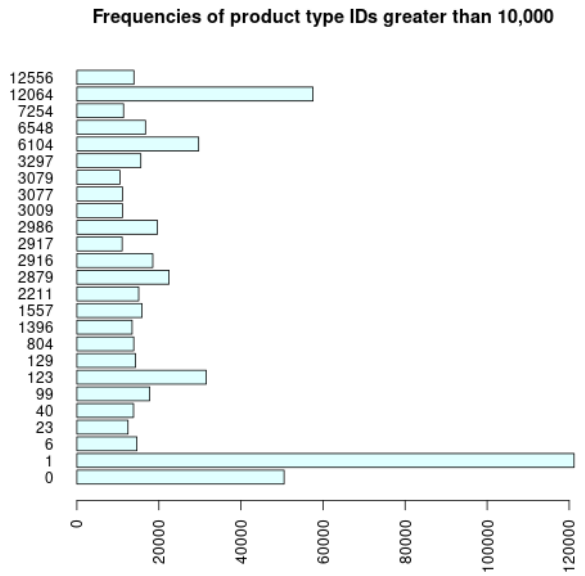

Figure 8

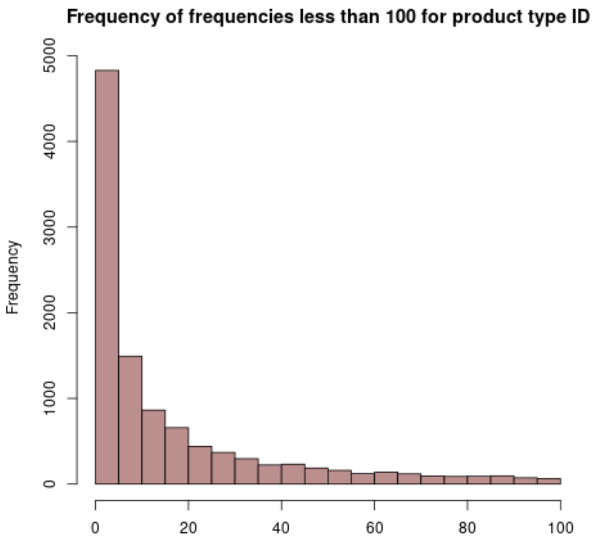

Figure 9

Access and availability : Public

Figure 8 visualises the barplot of all the type IDs that have a frequency of over 10000. There are 25 product types, that have atleast 10000 products within.

Figure 9 visualsises the frequency of the frequencies. So there are about 4800 products types such that at most 5 products are of one specific type. There is a drastic descent from the first class frequency to the second, and thus we must understand that most of the product types offer very less variety in terms of the products.

Figure 10 shows the hierarchy observed in the product types for Electronics Category of Amazon.

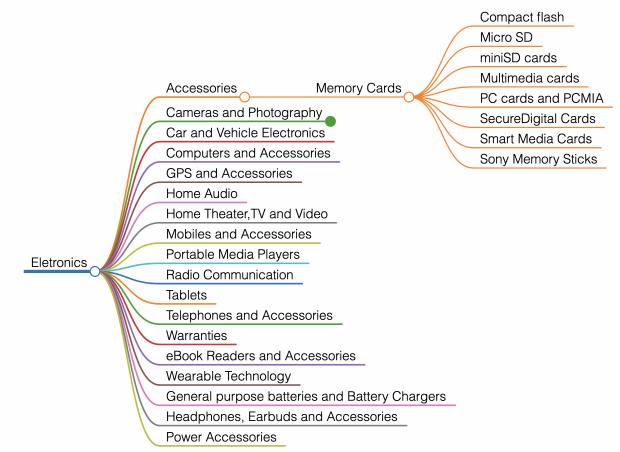

Figure 10: *The hierarchy of the product types as observed on Amazon for Electronics Category*

Dataset 2

Title : Amazon ML Challenge 2021 HackerEarth

Source : Kaggle (Web-scraping)

Data Description :
The dataset comprises of 5 columns - Product Title, Description of the Product, Bullet Points mentioned for the Product, Product Brand, Browse Node ID for the Product
All the variables are considered important in this scenario.
The size of this data is 2.6 million.

Access and availability : Public

Experimental results

Implementation

Product attribute extraction Product titles, bullet points and descriptions are written in a very different style and varies from dealer to dealer as well as product to product, even though the information conveyed by each are quite understandable by humans.

To make the data easier to process through any chunker model, some data cleaning was done which included replacing various instances of 'inches', 'inch', 'in', etc to 'Inch', replacing special characters and adding spaces. Stop words, etc were removed as that would interfere with the chunking performance. Product dataset tend to use commas, vertical bar, etc a lot those were replaced with appropriate words.

After basic data cleaning, Flair sequence-to-sequence tagger [5] English Chunker was used to perform chunking as described in the methodology. The model was trained on CoNLL-2000. We labeled noun phrases of all sentence and identified associated non-noun phrases. The performance on sentences that were written naturally the performance was near perfect. For product titles it failed some times to detect different concepts like dimensions and name of product. But on an average the performance was decent.

With the chunked product sentences about the product. Topic modelling over noun phrases and non noun phrases was performed on BERTopic with vanilla DistilRoberta[6], Fine-tuned DistilRoberta and sentence transformer all-MiniLM-L6-v2. The sentence transformer model trained on semantic similarity task outperformed as the semnatic similarity captured varying words with similar semantic meaning and the Fine-tuned model performed better than the vanilla one. Each cluster clearly revealed clustered attributes capturing the same semantic sense. Non noun phrase clustering on the other hand revealed similar use cases, properties relating to noun phrases from products in that category.

A combination of both the clustering techniques provided an unsupervised product attribute extraction technique.

Brand extraction First we merged Title, Description and bullet points to create a combined description. After tokenization we use AdaGram algorithm to find an embedding for this text.

Then we used this embedding to fine-tune a question-answering transformer model DistilBERT. DistilBERT [7] is a modified version of BERT which Knowledge distillation technique to reduce 40% of parameter while retaining 97% of the performance. We fine-tuned it to answer the question 'What is the brand name of this product?' from the product description. We selected 90% of the data randomly as training data and fine-tuned DistilBERT with it. Then we measured the accuracy by applying the model on rest of the data.

But this fine-tuned model suffered from data bias and only performed well when the brand name was at the beginning of the context and consistently failed miserably when it was anywhere else. This was expected as our synthetic contexts had title at the beginning merged with description and bullet points at the end and most product titles started with the brand names. So we further fine tuned this model this time using the context as description+title+bullet points. Now it performs fairly well even when the brand name is in the middle as can be seem from table 1 below comparing the mean f1 score of the 3 models in different contexts. Mean f1 score is calculated as :-

$$
\begin{aligned}
Precision &= \frac{TP}{TP + FP} \\
Recall &= \frac{TP}{TP + FN} \\
F1\ score &= \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \\
&= \frac{2 \times Precision \times Recall}{Precision + Recall} \\
&= \frac{2 \times TP}{2TP + FP + FN}
\end{aligned}
$$

Examples

Unsupervised PAE We have presented our unsupervised attribute extraction technique in figure 16 for the product attribute topics identified from the products with type id 1002 which refers to products in the category: Toys & Games → Dress Up & Pretend Play → Costumes. For the ease of interpretation only top 20 topic representations are shown. It is evident that the method successfully extracted candidate attribute names i.e. Topic 0, . . . , Topic 321, since we detected 321 topics in total. Manual human evaluation of the topic representation indeed shows semantically similar attributes grouped together. This can be also observed from 13.

Above we discussed about the attribute value clusters obtained for noun phrase clustering, a similar clustering over the non noun phrases provides us with a very different and interesting set of attributes which sheds lights on attributes like *'what this product is used for?', 'what this product is made of?', 'who this product is meant for?', 'what are some instructions need to be followed for the use of product?'*, etc. A similar topic representation over non noun phrases is shown in figure 12.

All the noun phrases are visualized in the embedding space generated out of embeddings from the sentence transformer model in figure 11. We can see how the various clusters with similar context and meaning of the product attributes are mapped closer.

Combining all these, we are able to extract a near-perfect product graph representing all the attribute names and values. It is shown in figure 1 and few application of our method is shown in table 3.

Brand Name Extraction Table 2 lists some examples of how the finetuned distilBERT perform against some real examples. It shows the one trained on title+desc+bulletpts performs poorly in all but the last case, though outperforms the vanilla distilBERT by a significant margin. The last model on the other hand annihilates both of them nearly in all category. Only in case of bullet points we hit it's limitations as

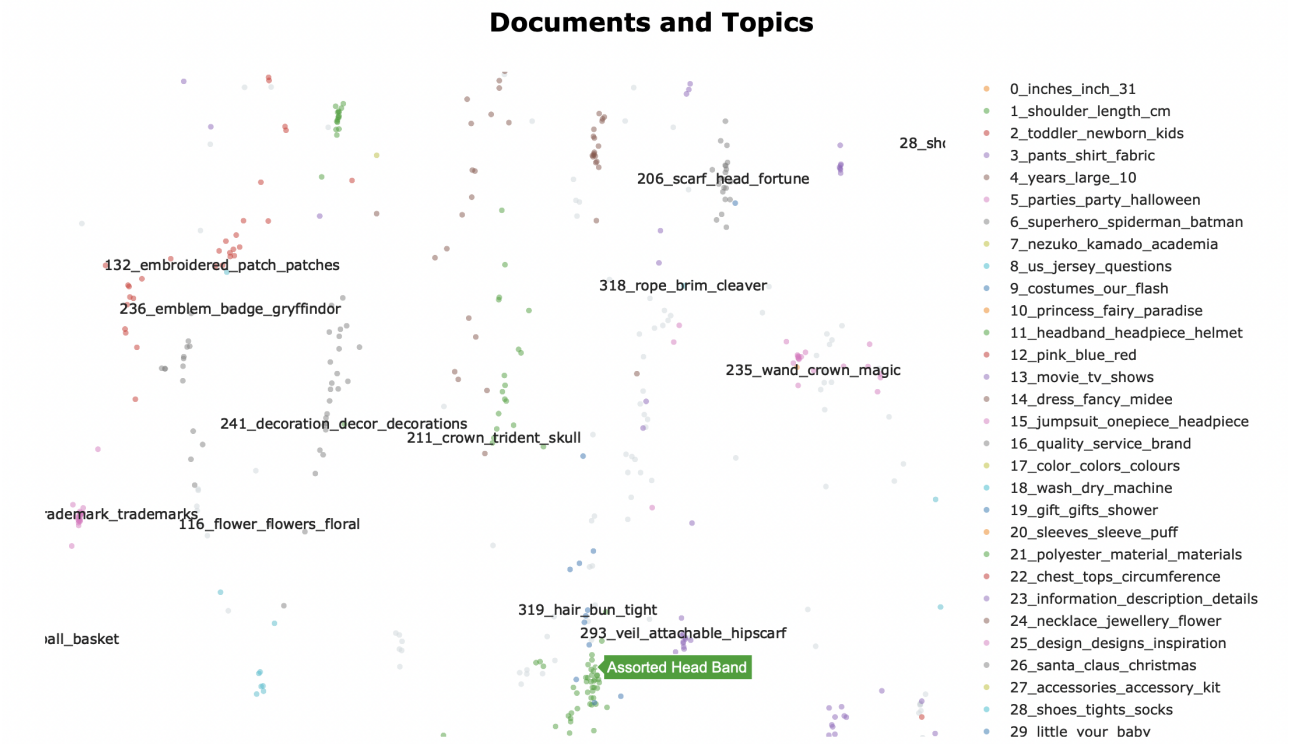

Figure 11: *All noun phrases visualized in the embedding space of the clustered products with type id* 1002*: Toys & Games → Dress Up & Pretend Play → Costumes. The product attribute Assorted Head Band of some product lies in the cluster 319_hair_bun_tight which can be thought analogous to an attribute name like hairstyle. So, the attribute name-value pair extracted from this product is (319_hair_bun_tight, 319_hair_bun_tight) equivalent to (hairstyle, Assorted Hair Band)*

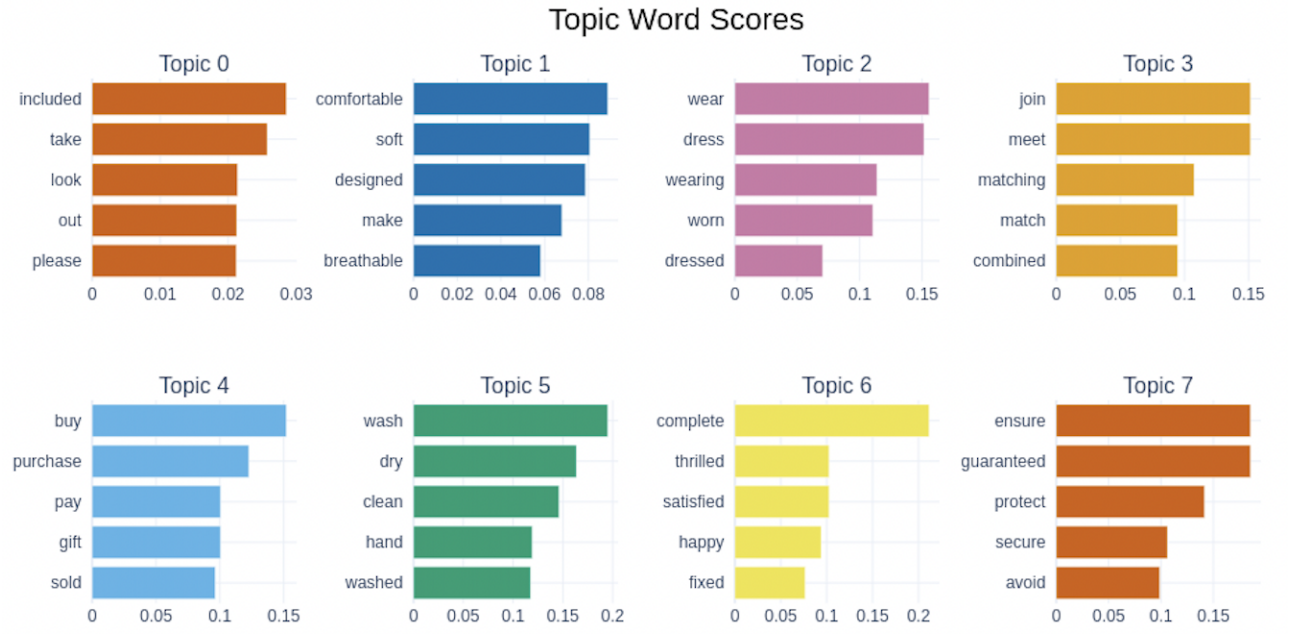

Figure 12: *Top* 10 *Topic Representations for various non noun phrase based attribute clusters detected in products with type id* 1002*: Toys & Games → Dress Up & Pretend Play → Costumes*

Table 1: *Comparison of model accuracies via Mean F1 scores of 3 models*

Product Attribute used for Inference	Pretained DistilBERT without Fine Tuning	Title+Desc+BullPts finetuned DistilBERT	Desc+Title+BullPts finetuned DistilBERT
Title	0.2517	0.8933	0.8833
Description	0.1480	0.1691	0.4067
Bullet Points	0.0250	0.0289	0.0745
Description + Title	0.2109	0.2767	0.8538
Bullet Points + Title	0.1877	0.1708	0.7367
Description + Bullet Points	0.0909	0.1167	0.4067
Description + Title + Bullet Points	0.1501	0.2251	0.8602

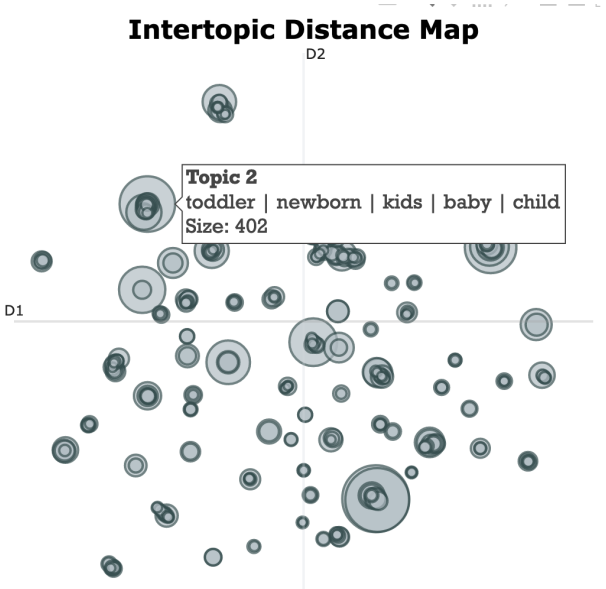

Figure 13: *The intertopic distance map shows similar attribute values and their clear differentiation from the others for the products with type id* 1002

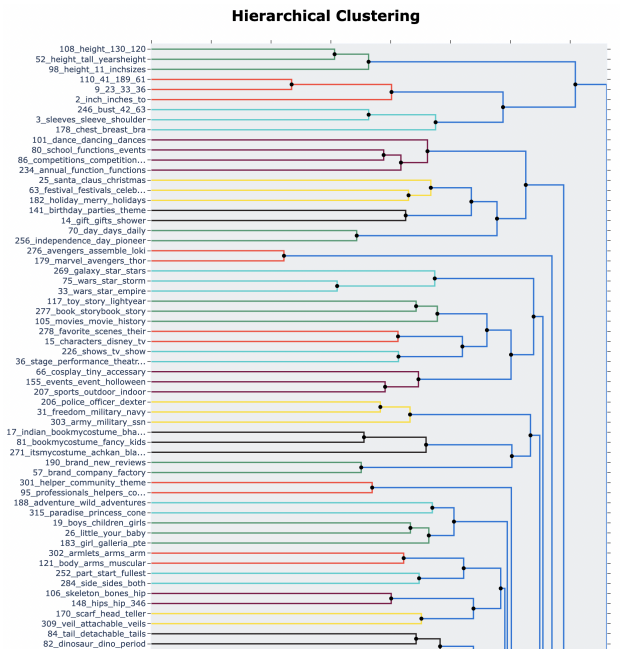

Figure 14: *The hierarchical tree created from topic modelling on the noun phrase products attribute shows appropriate grouping of similar semantic concepts*

being trained on primarily a positive dataset it can't confidently report the absence of brand name in the context.

Conclusion

In conclusion, this project focused on the task of product attribute extraction and brand name extraction from E-commerce data, specifically from Amazon. It is important to note that the technique used in this project is an unsupervised approach. Furthermore, it should be emphasized that there is currently no reliable open-sourced brand name extractor available for Amazon data.

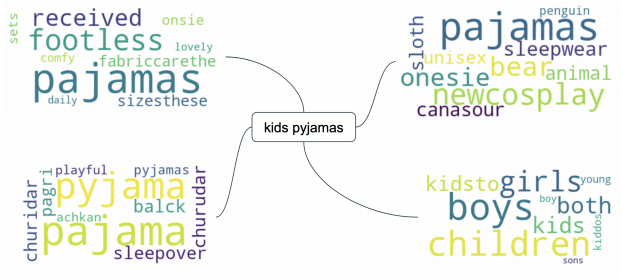

Figure 15: *The topics found can help us search the semantic phrase of product attributes. The above shows an example of the top matching clusters retrieved for the search query: kids pyjamas*

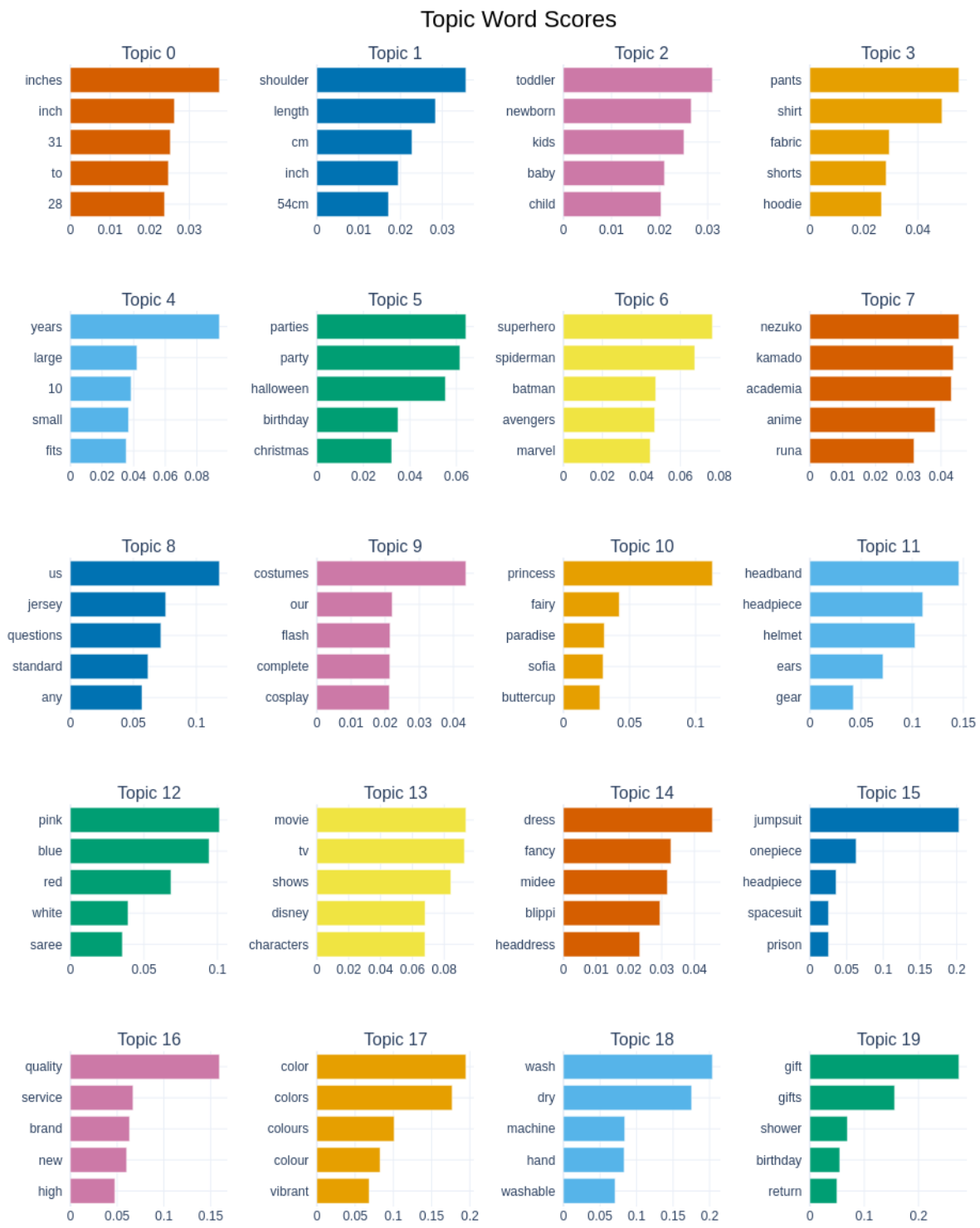

Figure 16: *Top* 20 *Topic Representations for various noun phrase based attribute clusters detected in products with type id* 1002*: Toys & Games → Dress Up & Pretend Play → Costumes*

The project's code is openly accessible, allowing for transparency and replication of the work. However, it is essential to acknowledge certain limitations. Firstly, when dealing with a large number of noun phrases or non-noun phrases, the extraction process might be slow. To address this issue, the project currently relies on rule-based linking of noun phrases with non-noun phrases. However, there is room for

Table 2: *Some product descriptions and corresponding brand detections*

Product Description	Pretained DistilBERT without Fine Tuning	Title+ Desc + BullPts finetuned DistilBERT	Desc + Title + BullPts finetuned DistilBERT	Original Brand Name
Amazon Brand - Vedaka Premium Red Masoor Whole, 1 kg | Rich in Protein | No Cholesterol | No Additives	Amazon	vedaka	vedaka	Vedaka
Ant Esports GP100 Controller Joysticks for PC (Windows 7/8/8.1/10) / PS3 / Andriod/Steam Gaming Wired Gamepad	Ant Esports GP100 Controller Joysticks	ant sports	ant sports	Ant Sports
Illuminate your life with the LuminaGlo Smart Light Bulb from EnerGlow. This revolutionary bulb not only brightens up your space with a warm, inviting glow, but it also syncs seamlessly with your smartphone.	LuminaGlo Smart Light Bulb	energlow	energlow	EnerGlow
Experience the refreshing power of AquaVita's HydraFresh Facial Mist. Enriched with a blend of natural botanical extracts, this mist revitalizes your skin complexion.	HydraFresh	aquavita's	aquavita	Aquavita
3 idea Imagine Create Print Creality Ender 3 V2 20222 Upgrade Version 3D Printer FDM 3D printerSilent Motherboard Meanwell Power Supply Carborundum Glass Bed Color Display	Imagine Create Print Creality Ender	3 idea imagine create print	3 idea imagine create print	Creality
3D Pen with Adapter | 3D Pen for Kids | 3D Pen with 3 * 1.75MM PLA Filaments 10m Each | 3D Printing Pen Drawing Toy | 3D Printing Pen - Perfect for DIY and Crafting.			[CLS] token	Brand not mentioned
KONG Puppy toy is customized for a growing puppy's baby teeth, the unique, natural rubber formula is the most gentle within the KONG rubber toy line. Designed to meet the needs of a puppy's 28-baby teeth, it helps teach appropriate chewing behavior while offering enrichment and satisfying a younger pup's instinctual needs.	KONG Puppy	kong	kong	KONG
Slurrp Farm No Maida Millet Noodles | Not Fried, No MSG | Foxtail Millet and Little Millet Noodles Combo | Pack of 2-192g Each	Slurrp Farm No Maida Millet Noodles	slurrp farm	slurrp farm	slurrp Farm
Ever wonder what the moon looks liked up close? Now, with the MOVA 4.5 Inch Moon Rotating Globe, the majesty of space has never been so near. This otherworldly globe embodies the mysterious beauty of Earth's only natural satellite. The MOVA globe brings this haunting lunar object into the comfort of the home, school or workplace.	MOVA 4. 5 Inch Moon Rotating Globe	ever wonder what the moon looks liked up close? now, with the mova	mova	MOVA

Table 3: *Some product descriptions and unsupervised corresponding product attribute extractions*

Product Title	Product Attribute Value Pairs
TITLE: Suit Yourself Classic Tinkerbell Halloween Costume for Girls Medium Includes Wings	Brand Name: None; Costume: Classic Tinkerbell Halloween Costume; Who the product is for?: Girls; Measurement: Medium, What does the product include?: Wings
TITLE: Rubies Deluxe Princess Wedding Costume Dress Child Small; BULLET POINTS: 1. Rubies Deluxe Princess Wedding Costume Dress Child Small, 2. Princess dress for stage plays Halloween every day dress-up, 3. Part of Rubies Little Princess Deluxe Costume Collection; DESCRIPTION: -	Brand name: Rubies; Costume: Princess Wedding Costume Dress; Costume: Princess Dress; Who this product is for?: Children; Measurement: Small; What this product is for?: stage plays, Halloween, every day dress-up; What this product is a part of? Rubies Little Princess Deluxe Costume Collection
TITLE: Roma Costume 3pc Bavarian Beauty , Blue or White Small; BULLET POINTS: 1. Women ' s Costume; DESCRIPTION: Includes Top Suspender Skirt with Trim Detail and Hat	Brand Name: Roma, Number of Items: 3pc; Color: Blue or White; Measurement: Small; Who the product is for?: Women; What does the product include?: Top Suspender Skirt; What does the product comes with?: Trim Detail and Hat

improvement by training a better transformer-based relationship extractor in the future, which would capture contextual meaning more effectively.

The method employed in this project successfully extracts numerous attribute-value pairs. However, it is important to note that sometimes the attribute-topic cluster may be uninterpretable due to noise present in the data. This indicates the need for further refinement in order to achieve more accurate and meaningful results.

The current bottleneck in the process lies in the chunker, which is not perfectly fine-tuned on product datasets due to the lack of labels for chunking in such datasets. Fine-tuning the chunker on product datasets would enhance its performance and address this limitation.

Finally, it is worth mentioning that the project provides several examples of extracted attribute values, similar to the brand name extraction table. These examples demonstrate the effectiveness of the method in identifying and extracting relevant information from the E-commerce data.

In summary, although this project employs an unsupervised technique for product attribute and brand name extraction from Amazon data, there are limitations to consider, such as slow processing with a large number of noun phrases or non-noun phrases. However, with further improvements, such as a better transformer-based relationship extractor and fine-tuning the chunker on product datasets, it is possible to enhance the accuracy and interpretability of attribute-topic clusters.

References

[1] Katrin Erk. "Vector Space Models of Word Meaning and Phrase Meaning: A Survey". In: *Language and Linguistics Compass* 6.10 (2012), pp. 635–653. DOI: https://doi.org/10.1002/lnco.362. eprint: https://compass.onlinelibrary.wiley.com/doi/pdf/10.1002/lnco.362. URL: https://compass.onlinelibrary.wiley.com/doi/abs/10.1002/lnco.362.

[2] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[3] Rrubaa Panchendrarajan and Aravindh Amaresan. "Bidirectional LSTM-CRF for Named Entity Recognition". In: *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*. Hong Kong: Association for Computational Linguistics, Jan. 2018. URL: https://aclanthology.org/Y18-1061.

[4] Sergey Bartunov et al. "Breaking Sticks and Ambiguities with Adaptive Skip-gram". In: *CoRR* abs/1502.07257 (2015). arXiv: 1502.07257. URL: http://arxiv.org/abs/1502.07257.

[5] Guillaume Lample et al. "Neural Architectures for Named Entity Recognition". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 260–270. DOI: 10.18653/v1/N16-1030. URL: https://aclanthology.org/N16-1030.

[6] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *CoRR* abs/1907.11692 (2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692.

[7] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *NeurIPS EMC²Workshop*. 2019.

Appendix

Training Statistics

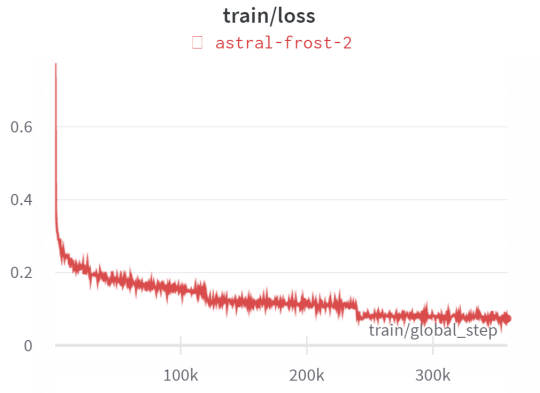

Figure 17: *Training Loss for Brand Name Extractor*

Figure 18: *Evaluation Loss for Brand Name Extractor*

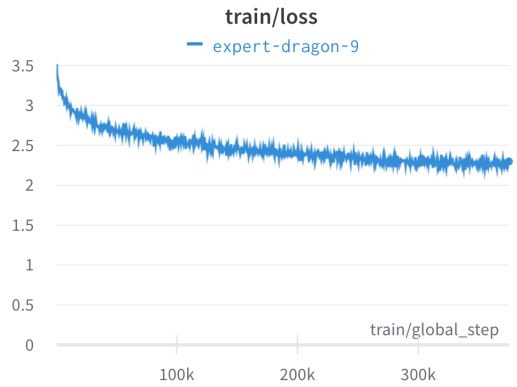

Figure 19: *Training Loss for Masked Language Model*

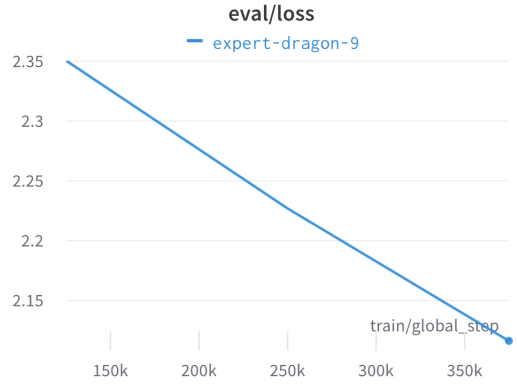

Figure 20: *Evaluation Loss for Masked Language Model*