A Work Project, presented as part of the requirements for the Award of a Master's degree in Business Analytics from the Nova School of Business and Economics.

Product Entity Matching in a Multi-Domain Landscape

A NLP Approach

José Pedro Pereira da Silva

Work project carried out under the supervision of:

Qiwei Han, Ph.D.

20/12/2023

Group Part

Abstract

This paper explores entity matching and its vital role in e-commerce to track products across different domains. Focusing on five diverse approaches, we evaluate their performance based on the precision to recall trade-off. Each model is examined theoretically, emphasizing its advantages and limitations. We also discuss potential improvements to enhance their applicability in this field. Our research aims to provide deeper insights into the effectiveness of various entity matching strategies and their performance under distinct priorities, while considering model architecture and room for improvement.

Keywords

entity matching, record linkage, entity resolution, data linkage, e-commerce, machine learning

Acknowledgements

We would like to express our appreciation to our advisor, Qiwei Han, for his guidance and feedback. Additionally, we would like to thank Maximilian Kaiser, from Grips, for providing us the dataset and supporting us throughout the project.

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 22209).

Table of Contents

1. Introduction

E-commerce, the process of purchasing and selling products and services through the internet, is continuously transforming the retail industry. Its growth has been more noticeable in recent years, with a report from UNCTAD (2021) indicating that e-commerce's share of all retail sales increased from 16% to 19% in 2020. The industry is expanding due to technological advancements and changing customer behavior, resulting in an increasingly digital shopping experience. The move to online shopping has been sped up by worldwide events like the COVID-19 outbreak, which has boosted e-commerce to become a major power in the retail sector. The worldwide revenue in the e-commerce market is projected to reach about 3.1 trillion USD in 2023 (Statista 2023), with an expected 2.5 billion users by 2028.

The growth in usage and availability of data has made e-commerce an important arena of intelligence, thereby leading to a competitive advantage for those able to navigate it. Bungee Tech (2023) aptly notes, "In our omnichannel world... a quick search on our phone lets us quickly compare prices for pretty much any product we want to buy". This highlights the essential role of technology in streamlining product comparison and matching for retailers.

The global trade item number (GTIN), a standardized unique identifier of products, should simplify their grouping on various e-commerce platforms. However, its effectiveness is hindered by inconsistent global adoption, which presents significant obstacles in applying it to harmonize product data online, so other solutions should be considered.

Entity matching, as defined by Elmagarmid, Ipeirotis, and Verykios (2007), is “the process of identifying groups of records that relate to the same real-world entity”. In e-commerce, entity matching is essential for sorting and analyzing product data in a timely manner. Proper management of this process can give companies a competitive advantage by enabling efficient data integration. This skill is vital for successfully navigating modern data environments and

achieving business triumph in data-driven settings.

Although using entity matching in product data is a substantial use case, its reach extends well beyond that. It is important in healthcare, for instance, to ensure patient records are kept consistent across various healthcare providers. Similarly, in finance, it is crucial for combining customer details throughout various branches and services, improving customer relationship management, and assisting in detecting fraudulent activities. These examples demonstrate the extensive usefulness of entity matching in managing and interpreting large amounts of data across various sectors.

In particular, matching products online is a big challenge as it is difficult to determine the level of similarity needed to consider two products a match (Wang et al. 2011). Figure 1 shows that assessing the similarity between two product descriptions does not always determine whether they actually match. This inconsistency arises because minor differences can substantially distinguish products, while at other times, the absence of standardized descriptions can lead to notably divergent descriptions for products that are actual matches.

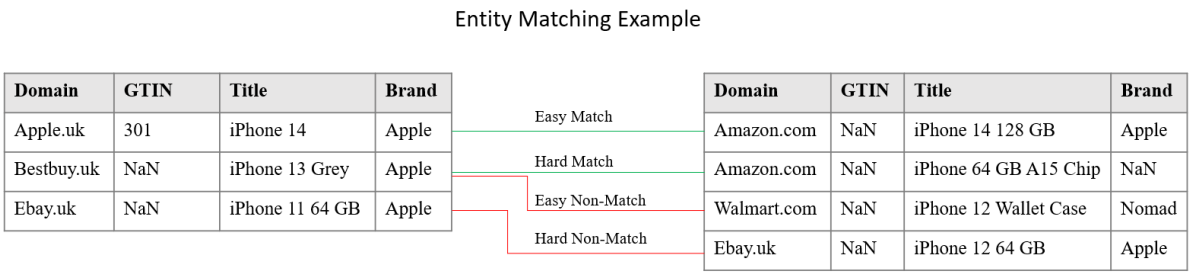


Figure 1: Entity matching example.

Since entity matching is a classification problem, precision and recall are two of the most important evaluation metrics. Precision measures the correctness of identified matches, while recall assesses the system's ability to capture actual matches. Balancing these two aspects is essential: a strict approach will miss a portion of actual matches, while a lenient method will

more often match different products. As such, this trade-off is integral to the effectiveness of entity matching systems.

Considering this, we define two research questions below. The first one refers to the trade-off between precision and recall, which represents a major discussion in this field as stated previously. To address it, we conduct a quantitative assessment of results. For the second one, we wish to understand how different model architectures relate to the challenge of entity matching, by using a more theoretical approach.

RQ1: How does model selection impact precision to recall trade-off and overall performance?
RQ2: How do different models / techniques fit into the field of entity matching?

To answer them, we first explore contributions to entity matching and how they relate to our approach. Following, we introduce the dataset, our models, and the methodology for comparing and evaluating them. Then, we compare the results of the given models, and how these affect the precision and recall metrics. Last, each individual contribution focuses on in-depth exploration of one model, and how it applies to product entity matching.

2. Related Works

Entity matching is an old field of study dating all the way back to 1946, when Halbert L. Dunn (1946) introduced the issue of record linkage and the importance of obtaining information of patients. Since then, the field has expanded into various areas and adopted numerous names, such as entity matching, entity resolution and data linkage (Barlaug and Gulla 2021). During the 1980s, the emergence of relational database management system (RDBMS) revolutionized data management and became the preferred way to store records (Novak and Hoksza 2010). Although these relationship-based databases ensure data integrity, combining data across systems becomes an issue. Without a common identifier across RDBMSs, the need for entity matching arises. The boom of the internet started another phase of entity matching, trying to combine data from different actors in the search for increased intelligence. Although these challenges have grown rapidly since the idea of entity matching first rose, there have also been many contributions to solve them, both technically and theoretically.

Considering the exponential growth of data in e-commerce and the quadratic nature of product entity matching, many consider blocking to be an essential part of entity matching rather than a preprocessing technique (Galhotra et al. 2021; Zeakis et al. 2023). The following subsections explore these contributions in two parts: contributions to blocking and contributions to matching. Last, we discuss which effects they have on our approach.

2.1 Blocking

A major constraint for everyone trying to perform entity matching is computational power. In theory, all possible combinations would have to be analyzed, which quickly amounts to an unmanageable volume. For example, a dataset of 50 000 entries would then require the computation of almost 1.25 billion pairs. Therefore, when performing entity matching, it’s

crucial to employ a technique known as blocking, which is the process of splitting data into blocks and only consider pairs within these blocks for matching (Papadakis et al. 2016). By doing so, blocking significantly reduces the computational load and makes entity matching a feasible task even with large datasets (Vidhya and Geetha 2019).

The main goal of blocking is to maintain all true matches while removing as many non-matches as possible. Consequently, when evaluating blocking effectiveness, we usually measure the reduction of comparison pairs in combination with recall cost. By recall cost, we are referring to the understanding that blocking recall sets the upper limit of overall model recall. The first contributions to blocking were rule-based or token-based, meaning they divided the data into blocks using a specific criterion or a shared characteristic between similar entities (Bilenko, Kamath, and Mooney 2006). The criteria can be based on various attributes like categories or any other feature that can indicate potential matches. While such approaches are effective when data is structured, it is far more challenging when data is dirty or unstructured. These methods are highly dependent on finding an effective rule or shared token, which is an issue with the heterogenous data online (Papadakis and Palpanas 2016).

To address these challenges, newer blocking frameworks seek to avoid solely relying on finding rules or shared attributes. In recent years, two major advancements influence the effectiveness of blocking: deep learning and natural language processing (NLP).

Deep learning seeks to leverage complex neural network architectures to discern subtle patterns and relationships within large datasets. This approach enables the identification of potential blocks based on learned representations of the data, rather than predefined rules or tokens. AutoBlock by Zhang et al. (2020) and exploration studies by Thirumuruganathan et al. (2021) demonstrate the effectiveness and future potential of deep learning approaches for blocking. Moreover, their study shows that such techniques outperform traditional ones significantly

when the data is dirty or unstructured.

More recent NLP models, based on transformer architectures, are able to understand context and semantics (Bojanowski et al. 2017). As a result, blocking approaches can be based on a deeper semantic understanding rather than rules or attributes. Nin et al. (2007) first demonstrated the positive impact and potential of semantic blocking compared to more traditional methods, which recently has become more accessible and effective through transformer-based NLP models. A more recent extensive survey of pre-trained embeddings for entity matching by Zeakis et al. (2023) found that Sentence-BERT models perform the best for blocking by reserving high recall across benchmark datasets. Note that these advancements are not exclusive, and modern blocking techniques typically use the combination of transformer-based NLP models and deep learning to achieve the best results.

In total, the blocking stage represents a trade-off between model time complexity and performance, as it generally leads to some actual matches being disregarded prior to matching. Recent studies found that modern blocking methods provide a significant increase in matching efficiency and precision with a cost of marginally reduced recall (Papadakis et al. 2020; Li et al. 2020). At the same time, the effectiveness of blocking is highly dependent on the nature of the data in question and the ability to apply adequate blocking techniques.

2.2 Matching

After the initial idea by Dunn (1946), Newcombe et al. (1959) presented the probabilistic basis for modern matching theory. Since then, many contributions have been presented and, one after the other, outperformed the “latest” techniques. The following section seeks to explain some of the more recent major breakthroughs, rather than exploring all advancements made throughout the years.

Advancements in NLP models have not only improved blocking, but also significantly enriched the matching stage. The modern models offer sophisticated semantic and contextual word representations, which far surpass traditional embedding techniques (Mikolov et al. 2013; Bojanowski et al. 2017; Pennington, Socher, and Manning 2014). Given the prevalence of textual features in entity matching, the advancements in NLP have positively influenced the evolution and effectiveness of matching approaches. The introduction of transformer architectures by Vaswani et al. (2017) marks another big step for NLP models, in which they were found to increase both effectiveness and time efficiency. For entity matching, these architectures have been proven to significantly outperform other state-of-the-art entity matching models (Brunner and Stockinger 2020).

Another crucial aspect of employing NLP models in matching is the capability to incorporate pre-trained models, such as BERT (Hugging Face 2023). These are trained on extensive and diverse datasets, enabling them to capture a broad range of linguistic patterns and intricacies. Moreover, their utility extends beyond mere plug-and-play applications as they are inherently adaptable through transfer learning. This process involves tailoring the model to suit specific contexts or data, thereby optimizing its performance for the targeted task. Such customization through transfer learning has been shown to significantly boost the performance of entity matching, as it enables the model to learn and adapt to patterns unique to the specific data (Brunner and Stockinger 2020). This implies that such approaches can be exploited without the need for training NLP models from scratch, which is a very comprehensive and time-consuming task.

Another notable area of advancements in matching models is within deep learning and neural networks, which in recent years have laid the foundation for major progress in numerous fields (Suk 2017). In entity matching, DeepER (Ebraheem et al. 2018) and DeepMatcher (Mudgal et al. 2018) were two of the first successful approaches employing deep learning. However, as

mentioned previously, the introduction of transformer-based NLP models outperformed these existing frameworks (Brunner and Stockinger 2020). More recent applications of deep learning, such as Ditto (Li et al. 2020), incorporate the progress of NLP models in their approach, leading to better results than previous standalone deep learning or NLP models (Li et al. 2021).

Finally, other contributions to matching are the self-supervised and semi-supervised approaches. Due to the growing challenge of obtaining precise labeled data, it is getting increasingly difficult to train sophisticated supervised models on relevant data. Self-supervised learning addresses this challenge by generating training signals and using them as labels in a supervised model. Semi-supervised, on the other hand, can be used when it is possible to extract a small portion of labeled data, which is very common with the GTIN in e-commerce. This method leverages a small amount of labeled data along with a larger pool of unlabeled data and allows the model to learn from a small set of known matches while also capitalizing on the broader context provided by the larger, unlabeled dataset. Ge et al. (2023) found that their self-supervised framework CollaborER far outperformed unsupervised models and almost equaled the performance of supervised methods across eight representative and widely used ER benchmarks.

2.3 Implications

The challenge of matching entities in e-commerce can be divided into three brackets: matching products across different actors, matching products within the same actor, matching products both within the same actor and across different actors. While most previous work has focused on the first two, this paper explores both multi-domain and intra-domain matching. A recent benchmark study for this kind of product entity matching is the WDC Training Dataset for Large-Scale Product Matching (Primpeli, Peeters, and Bizer 2019). The study found

conclusions similar to the works presented above, with NLP embeddings and deep neural network-based approaches outperforming traditional methods. Therefore, we considered the work above as highly relevant, even though our challenge is slightly different.

From the advancements and contributions presented above, we can draw several learnings to help guide our paper's approach to matching products within and across retailers. First, the identification of effective blocking features in our dataset is paramount. Features were explored not only in the scope of predicting matches, but also as a blocking attribute. Although blocking tends to decrease recall, it is indispensable for the efficiency of training models and predicting matches.

Additionally, all models should exploit the transformer architecture of modern NLP. Constrained by time to train and resources, we pivot towards utilizing pre-trained NLP models instead of training them by ourselves. Moreover, we can adapt these pre-trained models to our specific data using transfer learning. Such methods have performed similar to others which were trained from scratch (He, Girshick, and Dollar 2019), and will allow us to focus on other parts of the research. Based on the research by Zeakis et al. (2023), we should also give particular consideration to Sentence-BERT models as these have proven most effective in both blocking and matching.

Last, considering the effectiveness of supervised models compared with unsupervised or self-supervised models, we cannot disregard their potential effectiveness in our use-case as well. This implies that the data must support a robust construction of such models. Consequently, we need to ensure that the final data has an adequate number of precise target values. More about how we adapt the initial data to ensure such quality can be found in Section 3.2.

3. Data Understanding

The data used in our research was provided by Grips, a German transaction intelligence company. It aims to “illuminate blind spots and create a comprehensive map of online commerce for retailers and brands” (Grips 2023). The dataset was generated through keyword-based web scraping, which involves a focused approach where specific keywords guide the data extraction process. For each one, the scraper performs a search on Google and visits the first “n” results, from which it extracts the product information using schema.org microdata before proceeding to the next keyword on the list. This method is particularly effective for collecting targeted information from vast web resources. However, it tends to compromise data quality because standardized scraping across various domains can pose a challenge in accurately identifying and retrieving the correct data. Once the web scraping was completed, Grips applied a category prediction model which adds a category attribute to the data. Additionally, an image URL was added as a final feature, linking each product to an image.

The final dataset provided is extensive, comprising over 7.2 million rows and 13 distinct columns. These columns contain important e-commerce metrics: keyword (search term used), domain (website address), GTIN, uniform resource locator (URL), product title, brand, stock keeping unit (SKU), category, country code (location identifier), currency (medium of exchange for pricing), price, price converted to USD, and image URL (link to the product image). A comprehensive data dictionary detailing these attributes is available in Appendix 1.

3.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial step in machine learning, which consists of examining the main characteristics of a dataset, often using statistical methods and data visualization. It is employed to discover patterns, spot anomalies, or check assumptions with

the help of summary statistics and graphical representations. Our analysis is conducted column by column with two main goals: understanding the values in each one and searching for trends and useful features for both phases of entity matching.

Some columns are incomplete, with six of them exhibiting null values to varying extents. A negligible amount of missing data is observed in four of those, whereas the GTIN and brand columns have 3.1 million and 1.8 million null values, respectively. Considering the purpose of product entity matching, our EDA focuses exclusively on instances with a non-null GTIN. This approach is strategic, as a sufficient number of identification labels are available to support supervised learning models and validate unsupervised ones. As will be explained below, there are many inaccuracies in this dataset, and as such it will undergo significant filtering in the data cleaning phase, detailed in Section 3.2. This realization led us to decide that only exploring the entire raw dataset could yield insights of limited scope. Therefore, a similar analysis is performed on the curated dataset, to extract findings which could be more pertinent to the modeling stage of this project. This dual approach to EDA ensures a comprehensive understanding of the dataset's structure and contents, which is particularly important to define data preparation procedures, while also enabling us to come across important patterns for the downstream development of suitable models. Additional figures related to frequency distributions and other characteristics are available in the Appendix, in Section 9.

3.1.1 GTIN

This variable serves as a cornerstone for product identification within our dataset. It is a sequence of numbers uniquely identifying a product on a global scale, ensuring standardization and efficiency in international commerce. In our case, it serves as a way to create binary target values for comparison pairs.

Within the scope of collected data, there are approximately 2.2 million unique GTINs among

the 4.1 million entries. This could suggest a positive degree of product variety, but it also implies that a vast majority of these GTINs are associated with a single instance in the dataset, as evidenced by the frequency distribution in Figure 2.

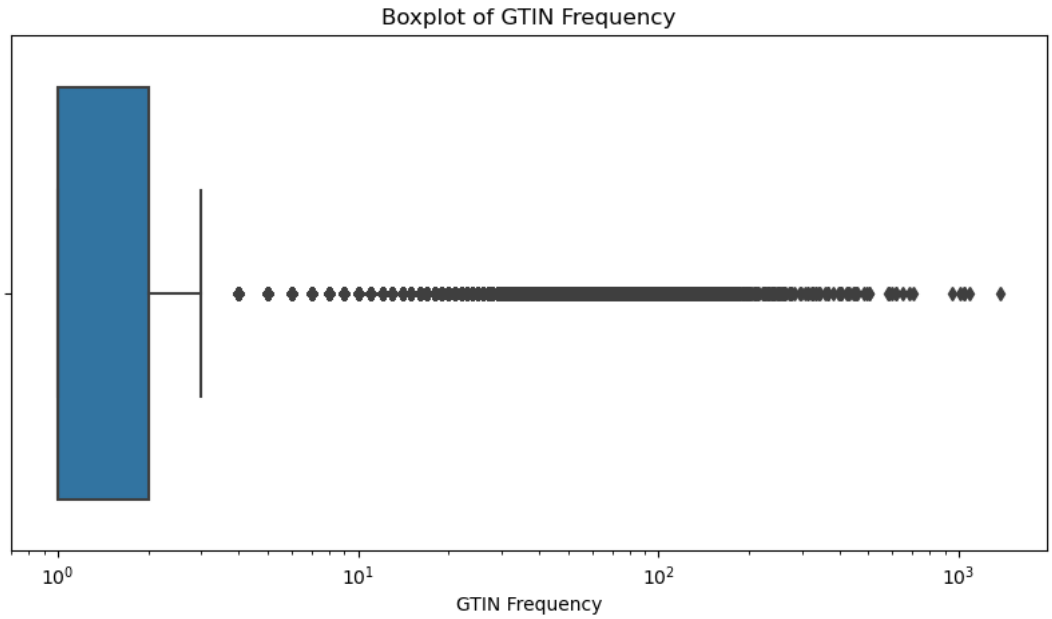


Figure 2: Boxplot of GTIN frequency with logarithmic scale on horizontal axis.

During the data cleaning process, a decision was made to retain only GTINs whose count is above a certain threshold. This filtering criterion is predicated on the notion that labels with few occurrences contribute minimally to the overarching goal of matching products and were found to be often associated with incorrect entries. By doing so, we enhance the robustness of our data, which is advantageous for pattern recognition and match identification. Figure 3, similar to the one mentioned above, supports this approach by highlighting the frequency of GTINs kept in the data.

Therefore, analyzing this column provided an understanding of product diversity, as well as valuable guidance on the data curation methodology by underscoring the importance of repeat occurrences for meaningful pattern detection in entity resolution.

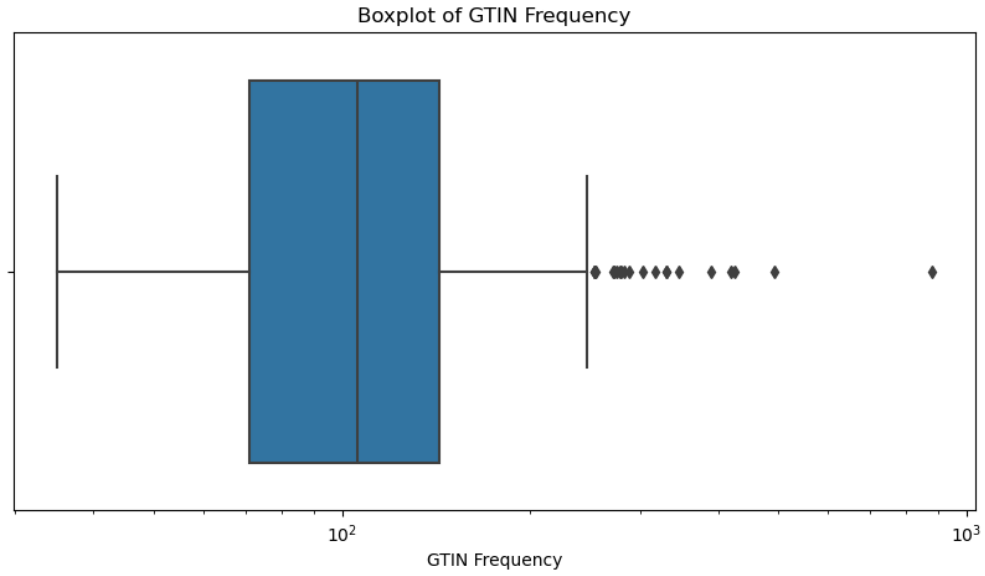


Figure 3: Boxplot of GTIN frequency with logarithmic scale on horizontal axis (clean dataset).

3.1.2 Keyword

This column captures the search terms used on Google for the discovery of each product record. These keywords are expected to be closely associated with the product, providing insight into how consumers are finding items online.

In the original dataset, our analysis shows that the most frequent keywords are often tied to specific domains, and do not provide much added value since there is another column referring to this information. On the curated data, the keyword landscape changes significantly: no domain entries remain, and the majority of keywords appear only once or twice. Despite the low frequency of individual keywords, they tend to be similar when related to a common product, with slight differences due to variations in search behavior or phrasing.

This observation creates an interesting dynamic where each GTIN is associated with a wide array of keywords, but almost every keyword is uniquely linked to a single GTIN. Such characteristic could imply that keywords have potential to serve as an attribute in the modeling process to identify product matches. However, their variable nature casts some uncertainty on

their definitive value as a predictive feature within the models. Therefore, the utility of keywords will be assessed on a model-by-model basis. For supervised models, an empirical approach evaluates the impact of keywords on models' performance, determining whether they contribute with significant predictive power.

3.1.3 Domain

The domain column contains the web address where each product is listed for sale. A surprising number of unique values is present at more than 3.1 million. We quickly detected the existence of many incorrect entries, characterized by numeric data and singular occurrences. These anomalies were filtered to ensure that only valid domains are retained for further steps.

Post-cleaning, the dataset is refined to include only legitimate domains, each represented by multiple instances. Statistically, each product appears on average in approximately four distinct domains, which is beneficial for the primary objective of this project: to match products across various domains.

Regarding distribution, the average domain lists around 20 different products, albeit with a wide range. Some domains feature more than 100 distinct products, while others have just one. This variance must be taken into consideration when partitioning the data into train, validation, and test sets, as detailed in Section 5.1. However, the domain will not be used as a predictive feature within the models, given our intention to identify product matches irrespective of it.

The exploration of this column has highlighted the necessity for thorough data filtering, and secondly, it has validated the assumption that products are listed across multiple domains, a key quality indicator for the suitability of our dataset for this research.

3.1.4 URL

This variable represents a specific web address for each product entry, which is more detailed than domain. Each URL is invariably linked to a unique product, ensuring a one-to-one correspondence between a web address and a product entry. However, this uniqueness also limits the URL's utility for our goal. Since matching products across different domains is a main objective, and a change in domain results in a different URL, this column loses its relevance as a feature for entity resolution. This is corroborated by the fact that each product is associated with an average of approximately four different URLs, which aligns with the previously discussed average number of domains per product.

Consequently, the URL column will not be used in the modeling phase since it does not provide the cross-domain predictive value necessary for the identification of product matches.

3.1.5 Product Title

In the context of this dataset, a product title is the name given to a product as it appears on an e-commerce website. This title typically contains the brand, model, and sometimes specific attributes such as color or size. It is anticipated to be a pivotal textual feature for matching products across different domains due to its descriptive nature.

The data reveals a need for considerable cleaning of the product title column. Undesirable characters, such as repeated apostrophes and dashes, as well as superfluous spacing, must be removed to standardize the titles and prepare them for analysis. Moreover, the presence of some typographical errors requires a sophisticated approach to text processing, potentially involving a complex language model that transcends basic word-level embeddings.

There are about 1 million unique product titles out of 4.1 million total records. This level of repetition indicates that while there is some overlap, it is not so extensive that entity resolution

becomes trivial. Our analysis also uncovered that each GTIN is associated with an average of about four product titles. Given a similar value for domains per GTIN stated above, it is inferred, and subsequently confirmed by manual checking, that product titles are consistent within a domain. Additionally, when compared across domains, this variable still exhibits high similarity in cases where it is not identical.

With the exploration of this column, we found an important necessity for data curation and a consistent similarity between product titles for a single entity, even across domains, underscoring its potential as a valuable feature for entity resolution. In comparison to keyword, the product title is more informative because it directly relates to the product's identity.

3.1.6 Brand

Brand refers to the name identifying one company's products as distinct from those of others. It acts as an identifier for a group of products, so it may be of value for our goal.

This feature presents an immediate challenge: almost 1 million entries do not have an associated brand, accounting for almost a quarter of the dataset. Additionally, there are more than 80,000 unique brand values, which suggests data inaccuracies rather than a genuine representation of brand diversity, verified when looking at specific entries. As such, comprehensive data cleaning is imperative to minimize the number of null values and ensure that each entry correctly represents an actual brand.

After data curation, only real brands are retained, and many null values have been filled successfully. The ratio of brands per GTIN is approximately one, mostly aligning with expectations. On average, brands are associated with 20 different products, which is positive as they are not overwhelmingly diverse.

Since the product title does not always include a brand name, this column becomes a valuable

supplementary feature for product matching. It provides an additional layer of information that can be important when other features are insufficient for accurate results. It could also be considered as a blocking attribute, but as explained in Section 6.1.1, the employed strategy performs this to a great extent, and brand would not add value to this phase.

Overall, this analysis allowed us to understand a requirement for thorough data cleaning, since brand is one of the least accurate and complete columns. However, it is emphasized that once data quality is enhanced, the brand feature should indeed be incorporated into our models. Its inclusion is expected to contribute positively to the robustness of matching algorithms.

3.1.7 SKU

The SKU is a distinctive sequence of alphanumeric characters used by retailers for internal inventory management and tracking. Unlike GTIN, there is no universally adopted standard for SKUs, making them unique to each retailer.

Despite having few null entries and being typically associated with a singular product, the usefulness of this feature in predictive modeling is limited. The retailer-specific nature of SKUs means they primarily serve to connect entries referring to the same product within a retailer, rather than across different ones. Such relationship is further elucidated by the observation that each product is on average linked to around four distinct SKU values. As would be expected, this mirrors the earlier insight regarding the average number of domains per product, reinforcing our previous takeaway.

As a result, while SKUs are integral to individual retailer operations, their variability and domain-specificity render them less valuable as a feature for models aimed at matching products in our broader, multi-domain context.

3.1.8 Category

The category column pertains to a systematic classification of products into specific groups based on their characteristics or functions. It is organized in several hierarchical levels that offer increasing detail about the product's nature, usually ranging from three to five in total.

Upon examining the filtered dataset, it becomes evident that all categories are well-represented, containing numerous entries that span a diverse range of products. A key observation is that all records under a specific GTIN share the same category, reflecting a high degree of consistency in product classification.

The reliable structure and uniformity of the category data make it an excellent candidate for blocking. Since there are no cross-category matches, recall will remain unaffected and many comparison pairs can be quickly disregarded as non-matches, increasing computational efficiency. Nevertheless, in cases where it is not used for blocking, this attribute might still provide value as a text feature, since it should increase similarity for records referring to the same product.

In conclusion, category emerges as a valuable asset for our project. It not only serves as an effective tool for blocking but also has potential as a feature in certain models. This dual utility emphasizes its importance in enhancing the efficiency and performance of our models.

3.1.9 Country Code

A country code is a standardized combination of letters representing the country where each product is being sold, essential for geographical categorization and analysis. Our dataset primarily contains products sold in the United States, as indicated by the predominance of US country codes.

Despite its significance for geographic-based analysis, this variable does not offer substantial

value for our objective of matching records related to the same product. The rationale is that products are often available in multiple countries, and there is no other clear relationship between a country code and the likelihood of two products referring to a single entity.

3.1.10 Currency

This column refers to the type of money used in each product's listed price. Given the international nature of e-commerce, the currency is pivotal for understanding financial aspects of our dataset, especially in terms of standardizing price information across diverse markets.

Mirroring our findings from the country code analysis, a vast majority of records in this dataset list prices in USD. This is an expected outcome, considering the predominance of US-based entries. The primary utility of this variable lies in its role during data cleaning, aiding in the conversion of all prices to a single currency, ensuring consistency and comparability.

Even though we observe that most GTINs have their associated records listed in a single currency, it does not appear to be a useful feature for entity resolution. As was explained for country code, the currency is more indicative of the context in which the product is sold rather than the entity itself. Products can be listed in different currencies across various domains, reducing their effectiveness as a distinguishing feature for product matching.

3.1.11 Price

Price represents the monetary value assigned to each product by its vendor. It is a dynamic numeric attribute, varying across different products and retailers.

Our overall analysis of price data reveals a large standard deviation relative to the mean, indicating either a wide distribution or the presence of extreme outliers. Figure 4 confirms the latter, showing numerous large outliers and a skewed distribution.

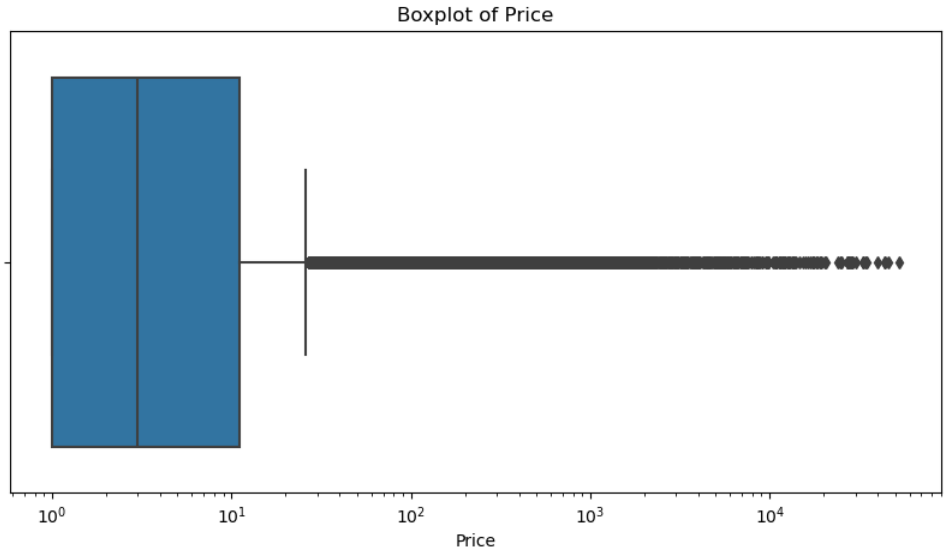


Figure 4: Boxplot of price with logarithmic scale on horizontal axis.

We noticed the existence of certain GTINs that exhibit unexpectedly high coefficients of variation (ratio of standard deviation to mean). Closer inspection of these cases often reveals incorrect price entries, for instance, the recurrence of $15,000 for much cheaper products. Such inaccuracies are targeted for correction during the data cleaning process.

After cleaning and preprocessing, the price column more closely approximates a standard Gaussian distribution. Within each GTIN, the standard deviation of price is notably small, as illustrated in Figure 5. This homogeneity implies that price could be a valuable predictive feature in entity matching models, as products with the same GTIN are expected to have equal or similar prices.

As a side note, there is another column for prices converted to USD. However, this conversion was found to have inaccuracies in certain currencies. To address this, we developed our own conversion method, explained in Section 3.2.

In conclusion, despite the initial need for data cleaning and manual price conversions, this variable holds significant potential as a numeric feature in entity matching models, because of its ability to indicate similarities between records referencing the same product.

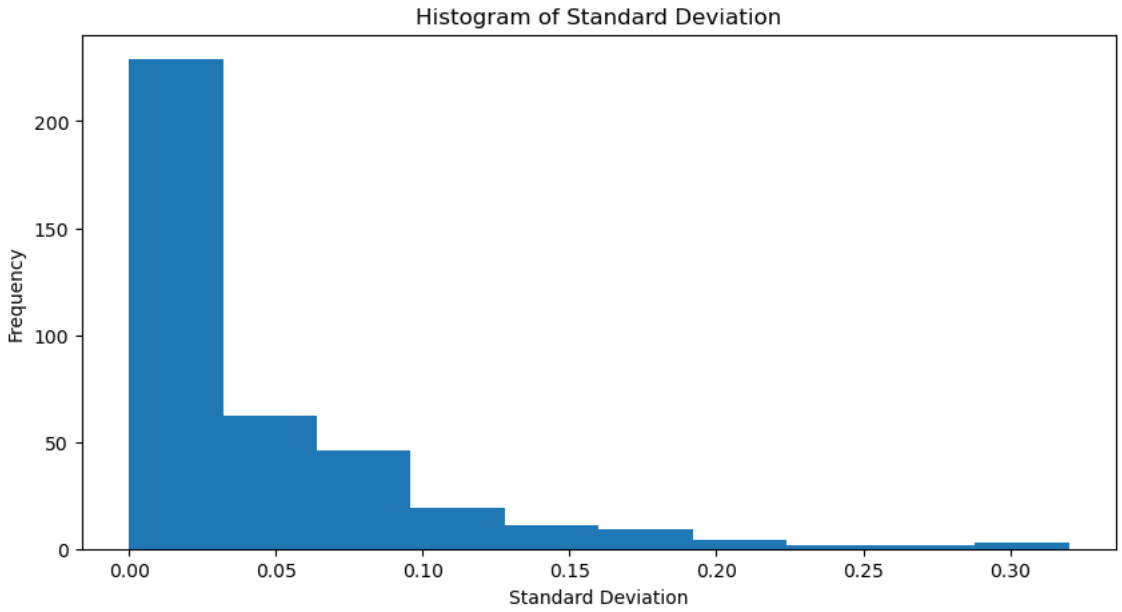


Figure 5: Histogram of price standard deviation (clean dataset).

3.1.12 Image URL

This variable pertains to the web address linking to an image of the product. A visual representation provides different information from textual or numeric columns.

Upon exploration, it was observed that while there is a considerable amount of null values, the total number of unique image URLs is lower than expected, suggesting potential data issues. A deeper investigation reveals that most GTINs are connected to a single image URL, regardless of the domain. This pattern is counterintuitive, as one would expect different domains to use distinct images for a product, or at least from a distinct URL.

As a result, the image URL often functions as a unique identifier for products. If it were to be used as a predictive feature in entity matching models, it could significantly simplify the matching process, almost to the point of making it trivial. This would be acceptable if such trends were legitimate, but as explained that is not the case.

Although this variable could allow us to create image similarity models to be used in

conjunction with others, we conclude that it is not useful because of clear unauthentic patterns. Nevertheless, if correct entries were available, their potential should not be understated.

3.1.13 Feature Relationships

For this part of our exploration, we strategically chose to focus on strictly filtered data. This decision was driven by the need to ascertain correspondence and validate important assumptions. Utilizing the original dataset would likely have skewed our analysis, since it contains numerous inaccuracies and false patterns mentioned previously.

We observed that keywords occur under a single category, confirming a certain degree of specificity in the former variable. This observation aligns with our hypothesis that keywords might contribute meaningfully to the models. However, drawing a definitive conclusion about their overall potential contribution remains challenging.

Our analysis of product titles revealed that each title is typically associated with a single domain. This finding underscores our earlier observation about variations in product titles across different domains. Despite these changes, product titles maintain a high degree of similarity when related to a specific product, reinforcing their potential in the matching process. In terms of pricing, the near-zero standard deviation within product titles aligns with our expectations and supports the idea of price being a valuable numeric attribute.

Finally, looking into image URLs demonstrated an average association with nearly three domains. This finding verifies our assessment that image data may not be entirely legitimate. Given this irregularity and the strong correlation of image URLs with specific GTINs, we reaffirm that they should not be used as a predictive feature in models.

3.1.14 Feature Summary

FEATURE	KEY TAKEAWAYS	USAGE
GTIN	Filtering needed to remove inaccuracies. Can be used to label actual matches.	Generating Labels
KEYWORD	Inconclusive relevance as a predictive feature.	Blocking & Matching
DOMAIN	Irrelevant for identification of matches. Useful to create task-specific set splits.	Train, Validation, Test Split
URL	Only applies within domain.	None
PRODUCT TITLE	Need for comprehensive text cleaning. Consistent within domain, similar across. Important predictive feature.	Blocking & Matching
BRAND	Large portion of null values. Supplementary where product title lacks brand.	Blocking & Matching
SKU	Same as URL.	None
CATEGORY	Well-structured and accurate. Distinct for each product. Mostly valuable for blocking.	Blocking & Matching
COUNTRY CODE	Irrelevant for task.	None
CURRENCY	Useful for price conversion. Irrelevant as a predictive feature.	Data Preparation
PRICE	Need to deal with outliers and inaccuracies. Low standard deviation for each product. Important for matching.	Matching
IMAGE URL	Unexpectedly identical across domains. Cannot be used as predictive feature.	None

Table 1: Feature summary.

3.2 Data Preparation

To prepare the data for modeling, we applied several cleaning and preprocessing steps. First, preliminary cleaning involved the removal of null GTIN values and the elimination of duplicate records. This phase aimed to establish a baseline of data consistency and reliability.

Then, a focused approach to data filtering was adopted. Records were filtered to retain only GTINs appearing more than 100 times and domains with more than 5 occurrences. This was done because less common GTINs had frequent data inaccuracies. Filtering ensured the dataset was concentrated on commonly occurring items and domains, and thereby more likely to present good data quality.

Another critical aspect of this process was standardizing price data. Given the existence of multiple currencies, all prices were converted to USD using average monthly exchange rates. Following this, a logarithmic transformation was applied, reducing the impact of significant outliers. Standardization of the log-transformed prices further scaled the data, making it suitable for predictive modeling. We also dropped several columns that were no longer needed.

For textual data, the process of cleaning and standardizing included multiple steps. Product titles and brands were stripped of extraneous characters, reduced to lowercase, and underwent space normalization. This aimed to keep only alphanumeric characters and minimize irrelevant text differences, which often pose a challenge in natural language processing. Moreover, a brand extraction function was employed to improve this attribute’s quality. Incorrect brand entries were replaced with null values by using a frequency filter, and subsequently the product title was analyzed in an attempt to find brand names which can fill null records.

To end this process, a new removal of duplicate records was required after dropping irrelevant columns earlier. An examination of the resulting dataset confirmed its readiness for the

modeling phase, as records are now thoroughly cleaned and standardized. The data preparation in our project exemplifies the comprehensive work required to structure and clean online product data, as summarized in Figure 6. Hence, it is an important area of focus in every entity matching project, as dirty or unstructured data may significantly hurt model performance.

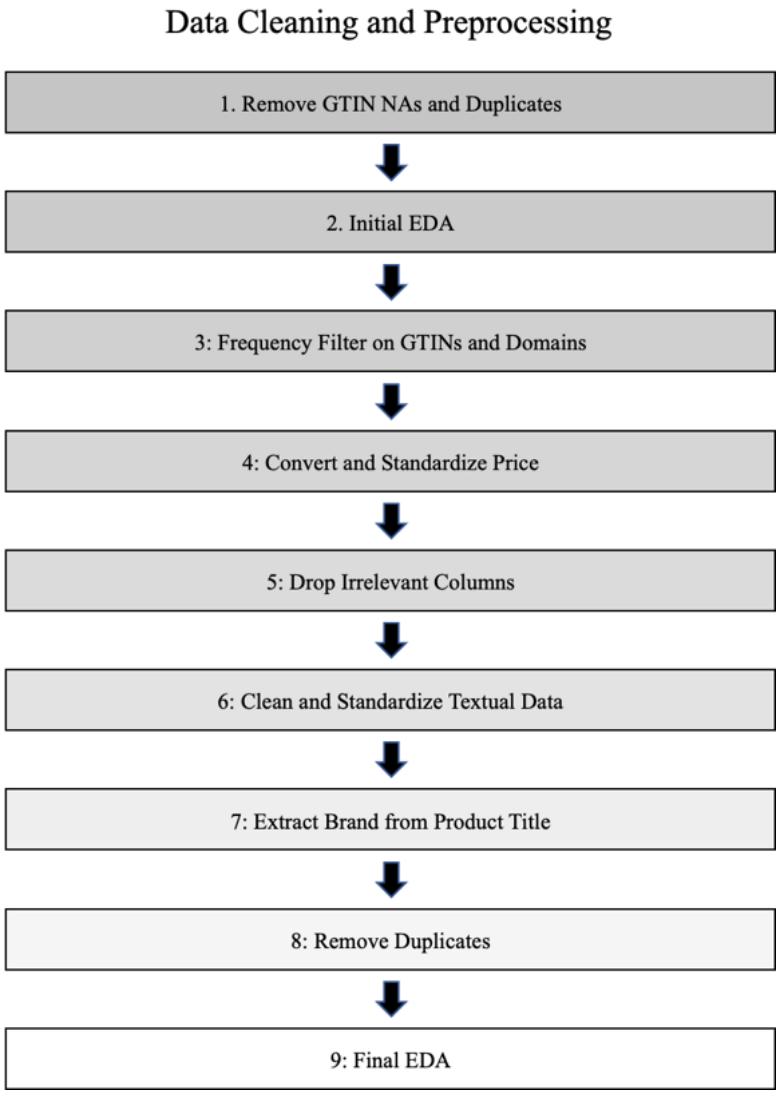


Figure 6: Data cleaning and preprocessing.

4. Modeling

In this study, we explored five different types of models for entity matching, whose flow is illustrated in Figure 7. These were chosen based on related works and our understanding of which model combinations contribute the most to addressing the research questions. First, the performance of deep learning models makes such approaches indispensable. Therefore, we chose two deep learning models: Siamese recurrent network (SRN) and graph neural network (GNN). Second, considering the advancements and success of NLP models in entity matching (Brunner and Stockinger 2020), we included a pre-trained NLP classifier and evaluated its performance with and without transfer learning. With the first three models functioning as black box models, it was important for the final two approaches to offer greater transparency. Thus, we adopted a density-based clustering method and a community detection algorithm, both grouping products by feature similarity. They are also unsupervised, which is an advantage considering the issue of collecting accurate GTIN data online. The following section gives a quick overview of each model, and a more detailed explanation is presented in our theoretical analysis of individual models.

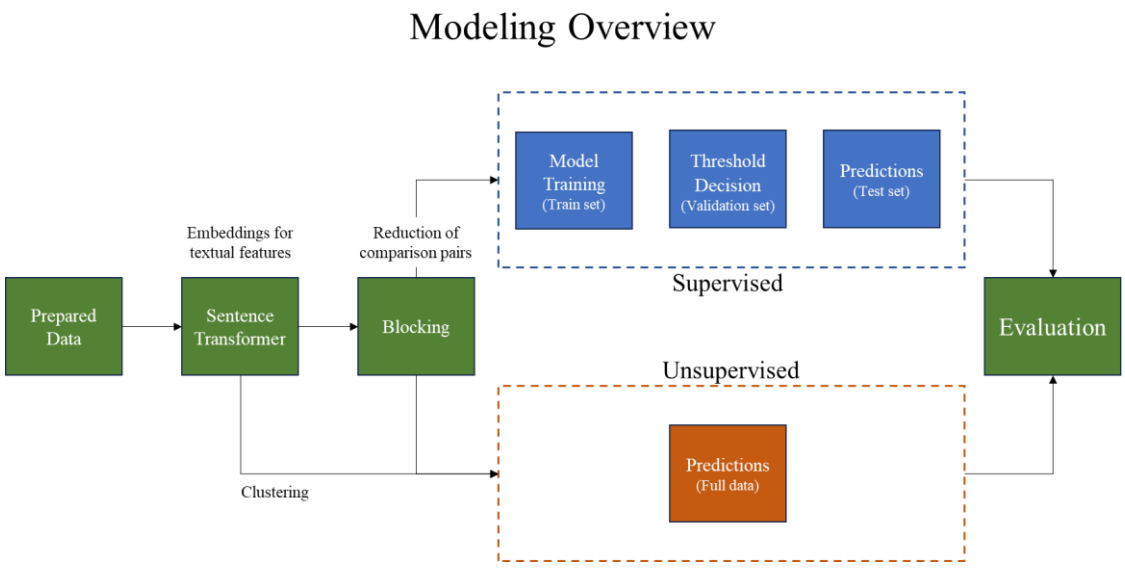


Figure 7: Modeling overview.

4.1 Model Descriptions

4.1.1 Siamese Recurrent Network

Common to all our approaches was an iterative experimentation process of feature combinations, from which the best performing group was selected. Here, chosen textual attributes (keyword, product title and brand) are vectorized using the GTR-T5 model, as explained in Section 4.2, and transformed through principal component analysis (PCA) for dimensionality reduction. Comparison pairs are then created based on category blocking and a cosine similarity threshold for text embeddings, ensuring that only entries with sufficient textual resemblance are included in training. This is a combination of NLP guided and token-based blocking, reducing the number of comparisons while retaining almost all matches.

For matching, a SRN receives vectors that combine lower-dimensional text embeddings with price data, aligning with the importance of price identified during the EDA. The final step involves calculating the Euclidean distance and passing this distance through a dense layer with a sigmoid activation function to get outputs between 0 and 1. This method leverages both textual and numeric data for effective results and uses a suitable architecture for our goal.

4.1.2 Graph Neural Network

The GNN applies the same feature transformation as the SRN. Identical blocking is also employed to minimize comparison pairs and increase model efficiency. For the matching stage, GraphSAGE (Graph Sample and Aggregation) (Hamilton, Ying, and Leskovec 2017)is chosen, leveraging the power of graph-based representations to capture relationships within the data. Records become nodes with assigned features, and labels are allocated to each edge (comparison) in the following way: 1 if they are the same product and 0 if not. For each node in the graph, the model learns a low-dimensional vector representation that encodes

information from its local neighborhood. By iteratively sampling and aggregating information from neighboring nodes, it produces rich and informative embeddings that capture the relational context. The model's final output ranges from 0 to 1, with larger values indicating a higher likelihood of a match.

4.1.3 NLP Classifier

The NLP classifier applies some pre-matching steps from the models above. First, it creates embeddings with the GTR-T5 model and employs equal blocking and generation of possible pairs. However, in contrast to the previous models, it does not use embeddings in the matching phase. Instead, when forming pairs, it takes the whole sentence that would normally be used to create embeddings. Then, in the matching phase, a cross-encoder is applied to classify the pairs as either match or non-match. This also excludes price as a feature since the cross-encoder only works with textual patterns. Multiple transfer learning approaches were tested, further explained in Section 6.1.3. The specific output varies, but common for all is that higher values refer to more confident matches.

4.1.4 Density-Based Clustering

Our clustering approach to product entity matching consists of three main steps. First, it transforms textual features into embeddings using the GTR-T5 model. Due to the clustering algorithm relying on density and distance, this model employs different textual features than previous ones. The chosen textual features align with EDA findings to best represent similarity: product title, brand and partial category. As the strongest attribute, product title was repeated twice to increase its representation. Additionally, we added partial category as an input instead of using it as a blocking feature since its effectiveness was limited for this model, as detailed in Section 6.1.4. Last, the standardized price was introduced with a balancing weight to ensure increased representation in distance computations.

The second stage uses the HDBSCAN algorithm (Campello, Moulavi, and Sander 2013) with cosine distance to predict clusters. This algorithm was chosen based on the combination of best early results and ability to handle varying cluster densities, a characteristic which is both present in our data and common for product records. HDBSCAN dynamically adjusts to the data's density distribution, enabling it to identify clusters of different shapes and sizes. We add a final step which allows for consolidation of clusters into more cohesive groupings, as well as allocation of entities previously determined as noise. The final output is a cluster label for each entry, representing a product group. As a consequence, for comparisons with other models, we incorporated a custom function that categorizes all entities within a single cluster as matches, while treating those across different clusters as non-matches.

4.1.5 Community Detection

In the community detection approach, a combination of text embeddings and price information was utilized to group products. The embeddings were generated using the GTR-T5 model and PCA reduction, using identical features to the SRN. Then, a graph was created where each record is a node and edges are added where embedding similarities and price proximity surpass a certain threshold. The community detection was executed based on this graph via the label propagation algorithm in NetworkX (Cordasco 2010), forming product communities. This method integrated both textual and numeric features for comprehensive community formation.

4.2 Introduction to Sentence Transformers

All models use the GTR-T5 sentence transformer (Ni et al. 2021) to vectorize textual features due to several reasons. First, empirical evidence for this model has shown high performance compared to other state-of-the-art approaches. Studies by Zeakis et al. (2023) on NLP models' performance in both blocking and matching suggest GTR-T5 excels over models like BERT in tasks demanding deep semantic comprehension. Second, when testing different transformers,

we also found it to outperform others. For our data, different domains often structure product titles in varied ways. The GTR-T5 is adept at handling these variations, ensuring that semantically similar entities are represented closely even when their textual descriptions differ. Third, the sentence transformer architecture ensures that it also represents feature relationships. Instead of vectorizing features individually, the transformer receives multiple concatenated attributes at once, as shown in Figure 8. The combined text input is then processed and transformed into a vector with 768 components.

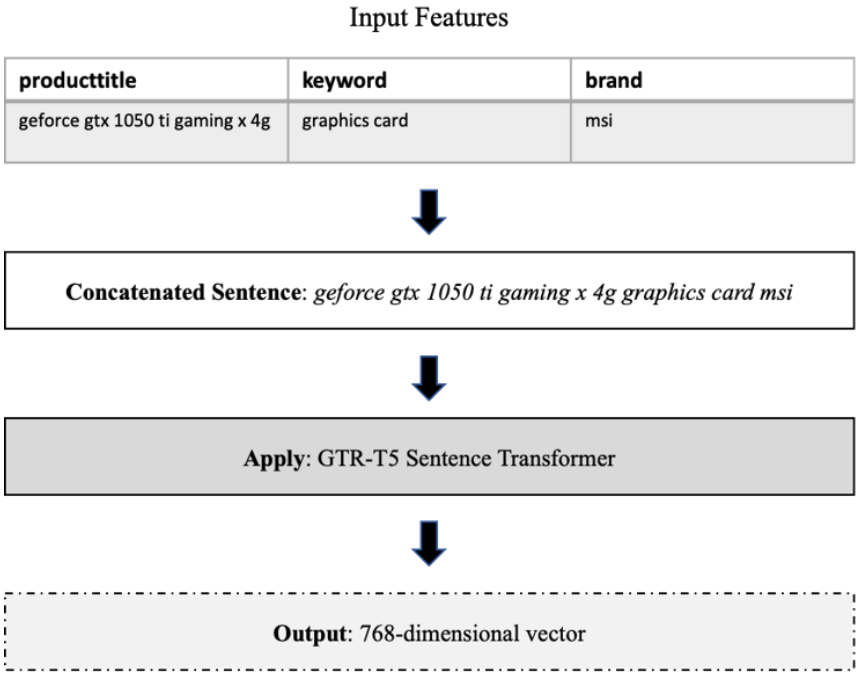


Figure 8: GTR-T5 transformation of input features.

The GTR-T5 base-model is an application of the GTR framework, whose architecture is based on T5 (Text-to-Text Transfer Transformer), a versatile encoder-decoder model designed to consolidate various NLP tasks into a singular framework. Rather than developing a novel model, it capitalizes on established methodologies. The T5 follows a similar structural design to BERT and this approach is predicated on the notion that while BERT operates primarily as an encoder, an encoder-decoder configuration can also yield effective outcomes, extending its utility to tasks beyond the scope of an encoder, such as text generation (Wolfe 2023).

Even though the GTR-T5 is one of the more complex computational methods, the time efficiency could be sacrificed for an increase in performance in our use case. Considering that entity matching entails processing a vast number of comparison pairs, the matching phase in various models often demands more computational resources than the sentence transformer. Consequently, prioritizing performance over efficiency is a worthwhile compromise in this case, especially considering the critical role of embeddings in the process.

The model also allows for transfer learning, meaning we could adapt it to better understand our specific data patterns. After exploring some less computationally extensive transfer learning techniques, these did not yield any improvements in representing differences, and we therefore chose not to pursue it further and rather focused on improving blocking and matching techniques. However, applying heavier transfer learning without server limitations might leave room for improvement in the future.

4.3 Server Limitations

When addressing the constraints related to computational resources in our project, it is essential to acknowledge the limitations imposed by the hardware infrastructure. The dedicated server is equipped with an Intel(R) Xeon(R) Platinum 8358 CPU, which features 30 logical processors and employs hyper-threading technology across 15 physical cores, each operating at 2.6 GHz. Additionally, it boasts an expansive memory capacity, with the potential to support up to 236 GB of RAM. Despite this powerful setup, the server's availability and performance were significantly hampered due to shared usage among multiple groups of students. This forced us to settle for simpler models, even though we would have liked to explore more computational expensive approaches. Such limitations underscore the importance of efficient resource management and the potential need for scalable or alternative computational solutions to accommodate concurrent demands, particularly for demanding tasks like entity matching.

5. Methodology

In this section, we outline our methodology for comparing entity matching models, each designed to address the complexities in matching entities across domains. The primary focus of our comparison lies in understanding the trade-off between precision and recall, and how each model performs to account for this trade-off and desired results.

5.1 Test Set

A test set is a crucial component in the development and evaluation of machine learning models. It serves as an independent dataset separated from the training data, used to assess how well a model generalizes to new, unseen instances. In our case, a simple train and test split on comparison pairs leads to all the same individual records being present in both training and testing. To solve this, we strategically split records from the original dataset, dedicating one specific domain to be assigned as test data. Then, while the training set contains all comparisons of records from other domains, the test set is composed by two types of pairs: both entries from the test set or one from each. As such, at least one new entry is always present in test comparisons, minimizing the issue of information leakage. Additionally, this is what would happen with unseen product records, since new instances would require comparisons between themselves, and to all other records already established. For reference, a validation set is created in a similar way.

For unsupervised approaches, we decided not to include a test set based on two reasons. First, hyperparameter selection is based on EDA findings, so evaluation on the whole dataset is valid and can be considered as the model's ability to generalize. Second, their performance suffers from significant variations depending on the chosen test domain. If ignored, it could provide an inaccurate representation of the model's performance for better or worse.

In conclusion, while it would be ideal to have the exact same evaluation data for all models, we believe our approach still allows for valid comparisons of performance. Supervised models, which have less variation of results for different test domains, can follow standard training, validation, and testing steps. Unsupervised models, if dealt with carefully and avoiding undesirable data leakage, can be evaluated on the whole dataset.

5.2 Evaluation Metric

In entity matching, the discernment between accuracy, precision, and recall is critical, particularly in the face of data imbalance, typical in these datasets. With non-matches far outnumbering matches, a high accuracy rate might result from correctly identifying non-matches, but this does not necessarily reflect effectiveness in detecting actual matches. Therefore, accuracy is rarely a suitable indicator of performance in entity matching.

Precision in this context is vital due to the negative impact of false positives: incorrectly matching unrelated products can disrupt subsequent operations, such as pricing and inventory management. While false negatives are also undesirable, their immediate impact is generally weaker as they are associated with less comprehensive match identification rather than erroneous connections. Moreover, it should be easier to account for shortcomings in recall than precision, for example, multiplying numeric results by a factor equal to the inverse of recall. However, this comes with other issues that are out of scope for this paper.

For the blocking stage in specific, recall needs to be prioritized, since it sets the upper limit for the overall metric. Higher recall ensures the retention of more actual matches for subsequent steps. Consequently, low recall in blocking inevitably limits the effectiveness of any model, regardless of its matching performance.

When evaluating models, we pivot around the precision-recall trade-off and address data

imbalance. Decision thresholds are essential to interpret results and determined based on the validation set, then applied to the test set. These are calculated through optimization of formulas with different focuses:

Precision-focused, where we prioritize precision by doubling its weight, suitable for scenarios where the cost of misclassification is high for positive instances.

$$\max\left(\frac{(2 \times precision) + recall}{3}\right)$$

Formula 1: Precision-focused formula.

Balanced, where we try to minimize the absolute difference between precision and recall. This method locates the equilibrium between precision and recall, appropriate for scenarios where both false positives and false negatives are equally undesirable.

$$\min(|precision - recall|)$$

Formula 2: Balanced formula.

Recall-focused, where we do the opposite, doubling recall's weight. When identifying all positive instances is more relevant than correctness, this is a better approach.

$$\max\left(\frac{precision + (2 \times recall)}{3}\right)$$

Formula 3: Recall-focused formula.

Our approach adheres to standard procedures to minimize information leakage from the test set, by defining these thresholds on validation set predictions. Adjusting them helps in understanding each model's trade-off between precision and recall, ensuring a thorough analysis that aligns with our first research question.

6. Performance Analysis

In the next subsections, we explore the results of each model individually and then compare them, ensuring an understanding of separate results before an important parallel analysis.

6.1 Individual Results

6.1.1 SRN

This model demonstrated remarkable performance and both phases are major contributors. Starting with blocking, its effects on reducing comparison pairs while maintaining actual matches is notable. It employs a token-based method on category and a threshold for cosine similarity of text embeddings. Figures 9 and 10 showcase the evolution of recall according to a varying threshold and resulting candidate set size for the training set. A final threshold of 0.5 was defined via interpretation of these figures along with an objective of maintaining over 99% recall for training and validation sets. Using this value, the evolution of remaining pairs and matches across the undertaken steps are presented in Table 2. Simply blocking on category already provides great reduction in comparison pairs: more than 95% are disregarded straight away in all sets. Additionally, text similarity filtering adds to this decrease in a variable amount across sets, but always maintaining high recall, which stays above 99%.

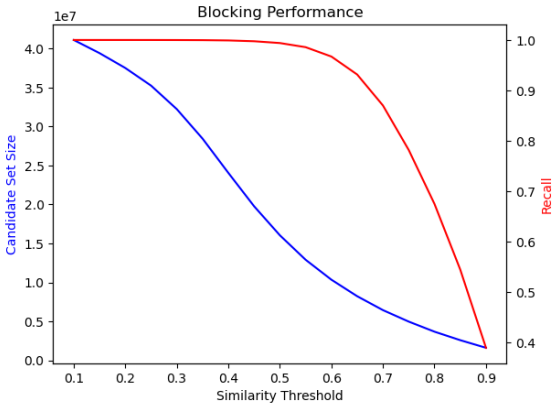


Figure 9: Line chart of candidate set size and recall over similarity threshold for blocking.

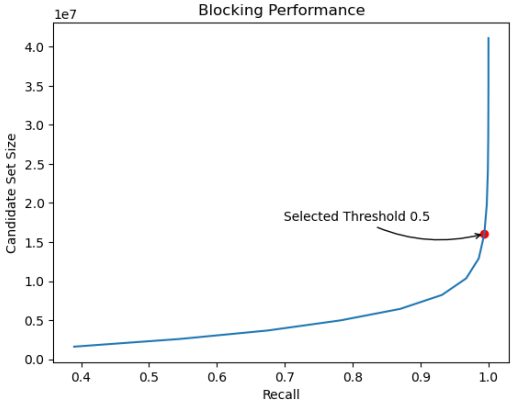


Figure 10: Line chart of candidate set size over recall for blocking.

Set	Measure	No Blocking	Category	Category and Text Similarities	Recall
Train	Comparison Pairs	931 328 061	45 194 555	16 045 519	99.4%
	Actual Matches	3 495 222	3 495 222	3 474 017	
Validation	Comparison Pairs	45 204 660	1 408 479	237 600	99.0%
	Actual Matches	110 024	110 024	108 873	
Test	Comparison Pairs	143 134 485	6 548 812	2 764 085	99.2%
	Actual Matches	418 699	418 699	415 412	

Table 2: Blocking results.

Regarding matching, Figure 11 shows the validation set evolution of accuracy, precision, and recall, when decision thresholds vary between 0 and 1. As expected, recall starts close to 1 and drops continuously while the opposite behavior is observed for precision. With an intersection around 90%, we restate this model's great performance in predicting matches. While precision, balanced and recall focused thresholds are chosen on validation set results, we still believe there is some value in showing a similar representation for the test set, presented in Figure 12. This is intended to show the model's potential if improvements were made to threshold decision methods. We are aware this would not be a best practice for implementation purposes, but it adds information in this research context.

This comprehensive approach, including the blocking results and analysis of matching performance at varying thresholds, provided a thorough understanding of the model's behavior. The SRN model emerges as one of the best-performing methods, showcasing its robustness and adaptability. For other model results in Section 6.1 and comparisons in Section 6.2, we only assess matching performance, since our analysis is focused on the matching phase's trade-off between precision and recall. Additionally, this blocking phase achieves over 99% recall, so overall results would still be nearly identical.

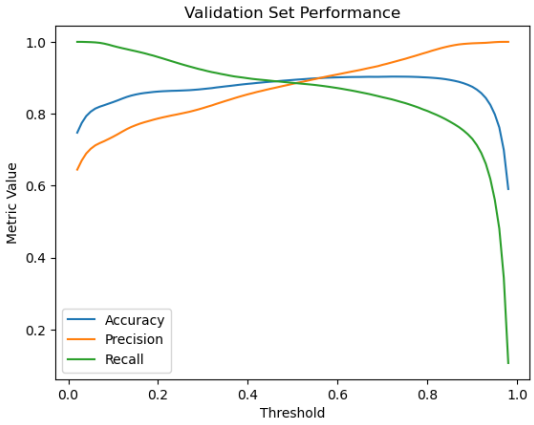


Figure 11: SRN matching performance on validation set.

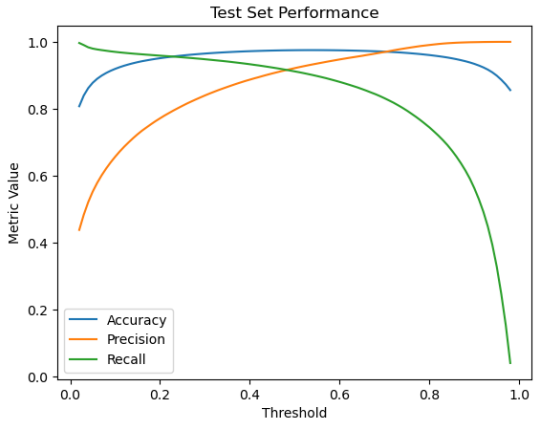


Figure 12: SRN matching performance on test set.

6.1.2 GNN

The blocking phase results are identical to the SRN model. For matching, GraphSAGE was the best performing GNN, generalizing better for new data. It is quite efficient, enabling usage of a large input during training. As executed in the other supervised models, thresholds were selected based on performance on the validation set.

Metric	Precision-Focused	Balanced	Recall-Focused
Precision	95.5%	88.5%	75.4%
Recall	75.8%	89.2%	93.2%

Table 3: GNN matching results.

As shown in Table 3, this model achieves 88.5% precision and 89.2% recall for the balanced approach. For precision or recall focuses, the trade-off is quite severe. Figures 13 and 14 illustrate this within a more comprehensive evolution of results. Overall performance is good even with a low number of epochs in training. The results reflect this model's ability to capture patterns within the existing constraints, gaining knowledge of the data by propagating information through the graph, and ending as one of the best performing models.

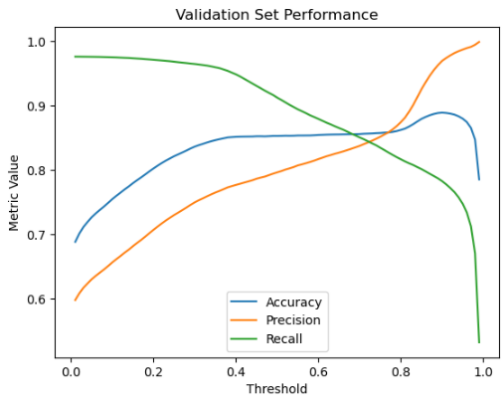


Figure 13: GNN matching performance on validation set.

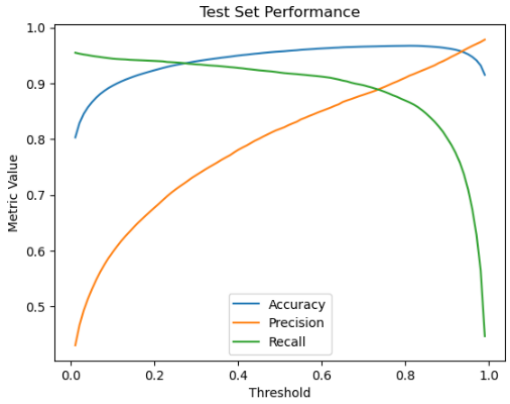


Figure 14: GNN matching performance on test set.

6.1.3 NLP Classifier

After going through the same blocking as the previous models, four different approaches were applied to the cross-encoder: none, light, medium and heavy transfer learning. As more comprehensive methods of transfer learning could lead to overfitting, all techniques were evaluated to ensure an appropriate selection. As shown in Table 4, performance is positively correlated with the amount of transfer learning, indicating that it is an essential part of applying pre-trained models to the task of entity matching. Although a time-consuming step, it achieved a notable increase in both precision and recall.

Model	Metric	Precision-Focused	Balanced	Recall-Focused
Imported model	Precision	45.1%	42.2%	28.7%
	Recall	52.8%	58.1%	76.3%
Light transfer learning	Precision	78.8%	63.4%	38.5%
	Recall	50.8%	69.9%	88.4%
Medium transfer learning	Precision	83.1%	75.0%	65.1%
	Recall	61.0%	72.3%	80.1%
Heavy transfer learning	Precision	91.3%	90.4%	84.7%
	Recall	71.9%	74.4%	80.4%

Table 4: NLP classifier matching results.

Model architecture is another important characteristic of the cross-encoder. Due to computational resource limitations stated in Section 4.3, we were unable to do transfer learning on more complex pre-trained models. As a consequence, these results should be interpreted with consideration for a greater potential of more computationally expensive approaches.

Figures 15 and 16 illustrate the performance on validation and test sets for different thresholds. An issue when defining the thresholds on the validation set could arise from poor alignment of predictions. In this case, validation intersection is at a higher threshold than test intersection, which also creates unexpected patterns when analyzing performance for different focuses. Referring back to Table 4, precision can be higher than recall even when the latter is prioritized.

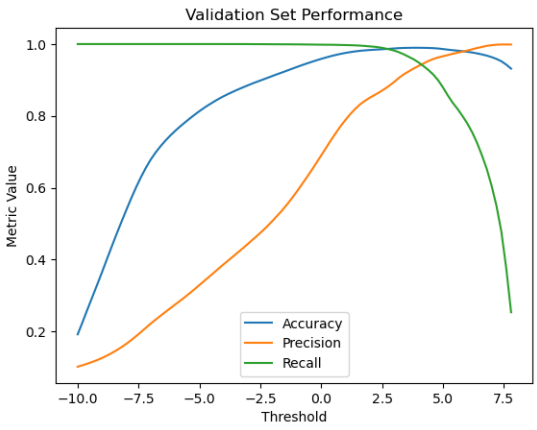


Figure 15: NLP classifier matching performance on validation set.

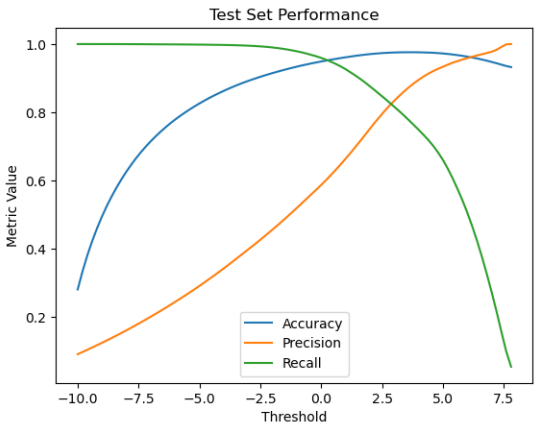


Figure 16: NLP classifier matching performance on test set.

6.1.4 Density-Based Clustering

In the clustering model, multiple options were considered and tested. Since thresholds cannot be used for this model, different techniques provide similar effects to the decision process for each focus. To analyze their individual performance, represented in Table 5, an iterative process was added to isolate each one. It is important to note that this is not tuning the model, but rather seeing how the introduction of different steps impacts results.

The first technique tested is blocking versus non-blocking. For clustering, the idea of blocking is slightly different from supervised models since it is not performing pairwise comparisons, so only a token-based method on category is employed. Incorporating blocking by category before predicting clusters could enhance the model's ability to distinguish between nearly identical products by reducing the number of records in each iteration. When evaluated, blocking achieved slightly better precision but much worse recall, as presented in Table 5. This happens because the model predicts additional unnecessary clusters.

A second differentiator for the clustering model is its hyperparameters. Although it is not tuned on a training set, we assessed how higher or lower configurations influenced the clustering stage. These initial experiments were not conducted on the same data used for testing, but rather a representative dataset which allowed us to understand model behavior without compromising the integrity of test data. Setting low cluster sizes, compared to EDA indications, resulted in the model generating too many clusters and thereby having a high precision and low recall. Contrarily, high values lead to more data being labeled as noise, again yielding similar results.

A third option is adding a post-processing step where noise and clusters with high similarity are handled. The idea of adding this technique as a last stage of the model pipeline is to ensure similar products are not split into multiple clusters, as well as finding suitable groups for non-allocated entities. Consequently, recall can be increased with a cost in precision. However, it must be implemented in a conservative way to avoid extreme precision decrease.

As a conclusion, this model's strength lies in its ability to achieve a high precision. While recall is lower, the emphasis on avoiding false positives indicates a conservative approach in making positive predictions. Techniques that could increase recall have limited utility when trying to retain acceptable precision.

Technique	Precision	Recall
Non-blocking	97.5%	62.4%
Blocking	98.6%	54.7%
High parameters	97.2%	58.2%
Low parameters	96.8%	57.5%
Noise handling	92.8%	66.3%
Merging clusters	82.8%	73.2%
Conservative noise handling and merging clusters	82.1%	76.5%
Full noise handling and merging clusters	17.1%	89.8%

Table 5: Clustering model results.

6.1.5 Community Detection

The community detection model demonstrates a precision of 60.4% and a recall of 74.5% when pursuing a balanced approach. The results shown in Table 6, while worse than other models, were anticipated given the model's lower complexity. It was designed to explore a novel solution, and with it comes some sacrifice in precision and recall. This trade-off is a natural consequence of the community detection algorithm, which inherently focuses on broader groupings of products rather than detailed pairwise classification.

Metric	Precision-Focused	Balanced	Recall-Focused
Precision	70.8%	60.4%	44.1%
Recall	47.0%	74.5%	86.2%

Table 6: Community detection model results.

A noticeable aspect of the community detection model is the severe trade-off between precision and recall. For research purposes, we decided to present performance across several possible thresholds, illustrated in Figure 17, to represent this model’s potential. The thresholds based on EDA findings may not align perfectly with results, which would be ideal but hard to achieve.

In summary, while the community detection model might not match the precision and recall rates of our other, more complex models, it still offers insights into their natural groupings.

Figure 17: Community detection model performance.

6.2 Model Comparisons

When comparing our models' performances, we adopt a sequential approach, focusing on one aspect at a time for a clearer understanding of performance under different objectives.

	Model	SRN	GNN	NLP Classifier	Clustering	Community Detection
Precision-Focused	Precision	99.5%	95.5%	91.3%	97.5%	70.8%
	Recall	70.7%	75.8%	71.9%	62.4%	47.0%
	F1-score	82.7%	84.5%	80.4%	76.1%	56.5%
Balanced	Precision	92.5%	88.5%	90.4%	82.1%	60.4%
	Recall	90.9%	89.2%	74.4%	76.5%	74.5%
	F1-score	91.7%	88.8%	81.6%	79.2%	66.7%
Recall-Focused	Precision	71.2%	75.4%	84.7%	82.1%	44.1%
	Recall	96.5%	93.2%	80.4%	76.5%	86.2%
	F1-score	81.9%	83.4%	82.5%	79.2%	58.3%

Table 7: All model results for different focuses.

6.2.1 Precision-Focused

Standout models for precision-focused results are the SRN and GNN, which achieve remarkable precision while maintaining reasonable levels of recall. Although F1-score is higher for the latter, a closer analysis of individual results demonstrates how the SRN has a better trade-off. In this case, its recall decreases further because the increase in precision is taken to a nearly perfect level. Then, the NLP classifier and clustering model present similar potential, but once again hidden by different metric values. While clustering achieves impressive precision, its recall cost is proportionally higher. Again, assessing F1-score alone might provide an incorrect perception of their performance when precision is prioritized. Finally, the community detection approach is characterized by significantly weaker precision and sometimes recall than other models. This model comes out as a less suitable choice, particularly in this specific analysis.

Our analysis reveals that 3 out of 5 models can be almost flawless in predicted matches, yet their ability to maintain high recall varies. Depending on a project's specific needs, any of these approaches could be a suitable choice when precision is of the utmost importance, and selection should hinge on the willingness to compromise recall, as well as possible requirements for model transparency or resource limitations. In particular, it should be reinforced that the SRN and GNN are objectively the highest performing models.

6.2.2 Balanced

The balanced focus results demonstrate how the adaptability of each model varies. The SRN and GNN perform best, showcasing their ability to simultaneously achieve notable precision and recall. In comparison, our two unsupervised approaches display a bigger drop-off in precision than increase in recall. This outcome embodies their challenge with recall: it is hard to effectively increase it without a relatively larger decrease in precision. The NLP classifier

does not change much from the precision focus, as its threshold on the validation set does not perfectly align with test results. As stated in Section 6.1.3, this characteristic should be remembered when evaluating model performance.

Regarding F1-score, it accurately represents model performance for this specific focus and three clear groupings appear. First, our deep learning approaches score significantly higher than remaining models, proving their effectiveness in entity matching with a low number of misclassified pairs. The second group includes the clustering model and NLP classifier, whose performance is worse, but good enough to confirm them as suitable approaches for entity matching in a balanced focus environment. Last, the community detection model performs significantly worse than the other models, suggesting that it is not suitable for product entity matching.

Overall, our models behave quite differently when it comes to a balanced focus. While the deep learning techniques manage to maintain high precision when giving similar importance to recall, the unsupervised models and NLP classifier struggle. This assessment with equal metric importance provides an objectively better understanding of each model's performance, even though most companies give preference to precision, as explained earlier.

6.2.3 Recall-Focused

For this approach, the SRN and GNN models stand out again, demonstrating their ability to identify a wide range of actual matches while maintaining decent precision. As a result of recall prioritization, these models include more false positives than in precision-focused scenarios, but their trade-off remains impressive. The community detection model emulates a similar trend, having higher recall than precision. However, its results are subpar and display a lack of ability to predict matches, despite having high recall. In contrast, the NLP classifier and clustering approach follow an unexpected trend. Both have a lower recall measure than

precision, and for the latter there is no change compared to its balanced results. These models failed in switching their focus to finding true matches, which can be justified by issues with defining decision thresholds on the validation set, as detailed in Section 6.1.3.

Overall, only the SRN and GNN manage to prioritize recall effectively, without an extreme cost in precision. In regard to community detection, performance is in general underwhelming, and other approaches simply do not follow the expected trend and reach a subpar measure for recall. Similarly to some results for a precision focus, F1-scores can once again be misleading, by not differentiating which models actually achieve remarkable recall from those who fail to prioritize this metric.

This method is beneficial in scenarios where missing a true match is more impactful than incorrectly identifying one. The choice between our models would be restricted to SRN and GNN, depending on the desirable trade-off in precision and objective for recall.

6.3 Conclusion

To sum up the analysis of model performance, it is essential to reflect on our approach to evaluating results across different focuses. Specifically, it is instrumental in addressing our first research question concerning the trade-off between precision and recall. This examination provides valuable insights into how different models perform under varying criteria, underscoring their compromise in precision and recall.

Across all results, models could clearly be organized into three groups. Particularly, the SRN and GNN exhibited notable trade-offs between precision and recall and achieved the best performance for a balance in these metrics. We regard them as most suitable for applications that require a more demanding approach to entity matching.

On a secondary tier, the NLP classifier and clustering models displayed similar potential. While

not attaining the same level of performance as previous ones, these still offer substantial value. Moreover, the clustering model stands out due to its interpretability and transparency, which are advantageous characteristics in certain applications.

An interesting aspect of our analysis was the exploration of community detection as a more unorthodox approach. While this model proved to be less effective than the others in predicting matches, its inclusion in our study provided a broader perspective on the range of methods that can be applied to entity matching.

In conclusion, our findings emphasize a crucial aspect of entity matching: the pursuit of perfect precision or recall invariably entails a significant cost to the other measure. Therefore, most use cases would benefit from adopting a middle-ground approach, tailored to their specific objectives and requirements. This balanced strategy allows for a more realistic and practical implementation of entity matching models, acknowledging their trade-offs and limitations.

7. Summary

This paper has explored several approaches to entity matching and analyzed the precision and recall trade-off for each one. We started with introducing the challenges of online product data and cross-domain matching. Then, we examined recent contributions to both blocking and matching in order to establish a deeper understanding of different techniques and their characteristics. Their implications for our work were detailed and used in subsequent sections as a theoretical basis for practical developments.

Regarding the dataset, it was compiled using a keyword scraper designed to capture a wide array of product records. EDA conducted on this dataset uncovered several key findings: keyword, product title, brand, category, and price emerged as important features for modeling, while domain information proved to be useful for set splits. Furthermore, exploring the data revealed a necessity for rigorous data cleaning, as it contained numerous inaccuracies and non-standardized entries, which are common issues when working with scraped data.

Several models were chosen to explore different approaches to product entity matching: a SRN, a GNN, an NLP classifier, a density-based clustering model, and a community detection algorithm. The rationale behind this diverse range was to capture advantageous characteristics of each model, thereby providing a more comprehensive overview of possibilities. It also allowed for a richer analysis and comparison of model performance, shedding light on their potential and relation to precision and recall objectives.

Our methodology was carefully structured to assess model performance under various conditions. For models allowing such an approach, we employed a split into train, validation and test sets, following standard procedures to evaluate their predictive capabilities. Considering the goal of matching products across domains, this feature was used to divide the

data. Additionally, we applied three methods for calculation of decision thresholds with distinct priorities: precision, balance, and recall. Each provided a good understanding of the models' ability to perform under specific constraints and was instrumental in allowing us to address our first research question about the precision and recall trade-off.

Our findings point to the superiority of deep learning models in the context of product entity matching. Such methods consistently outperformed others, affirming their status as the leading approach in this field. Additionally, the use of NLP classifiers and clustering techniques proved to still be effective. Notably, the transparency of clustering is an advantage, offering clear insights into the matching process. On the other hand, while community detection emerged as an intriguing model, its performance was found to be comparatively lower. Across all results, our best-performing models demonstrated high effectiveness with only slight variations in their trade-offs. This suggests that while certain models excel in specific aspects, top-tier models maintain a consistent level of performance across different evaluative criteria.

The following individual components do not further explore model performance, but rather go in-depth to explain their construction, advantages, limitations, and potential improvem

C. A Natural Language Processing Classifier Approach

C.1 Introduction

"In recent years we have seen new methods based upon deep learning techniques for natural language processing emerge" (Barlaug and Gulla 2021). This evolution from rule-based to statistical models, and now to deep learning, has significantly enhanced the field of NLP, particularly in entity matching. Deep learning has introduced models with the ability to understand and generate human-like text, a crucial advancement for tackling complex entity matching challenges.

Building on this progression, cross-encoders represent a further step in the advancement of NLP. Systems like Ditto demonstrate the power of pre-trained Transformer-based language models in sequence-pair classification, offering a more contextualized understanding of language. "We present Ditto, a novel EM solution based on pre-trained Transformer-based language models...which have been shown to generate highly contextualized embeddings" (Y. Li et al. 2020). Cross-encoders extend this paradigm by analyzing text pairs in conjunction, allowing for an intricate understanding of their interrelation, and marking them as an effective solution for the subtlety demands of entity matching in modern applications like e-commerce.

This emphasis on cross-encoders within the NLP landscape highlights their potential as a powerful tool for accurately matching entities. Their ability to process text pairs jointly positions them as a valuable advancement, building upon the deep learning techniques that have already revolutionized the field of NLP. Exploring the capabilities of cross-encoders further could yield significant benefits in enhancing accuracy and efficiency in the context of entity matching.

C.2 Model Architecture

This model architecture can be divided into three main steps as illustrated in Figure C.1.

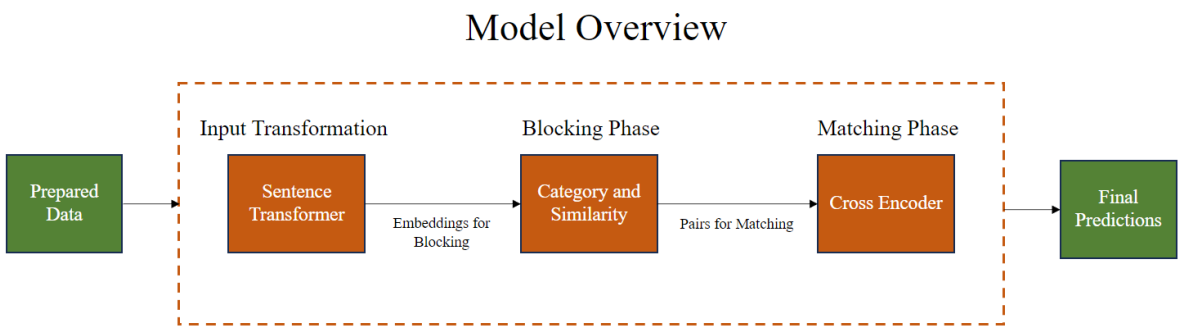


Figure C.1: NLP classifier model overview.

The model utilizes the GTR-T5 framework for transforming textual features, primarily chosen for its proficiency in complex language interpretation, vital in the early stages of entity matching as further explained in Section 4.2. It refines raw inputs: keywords, product titles, and brands into a format ready for deeper analysis. However, it's important to note that the model doesn't use the embeddings generated by GTR-T5 for comparison. Instead, it creates its own embeddings when comparing text pairs, ensuring a more tailored approach to entity matching. These newly generated embeddings capture both semantic richness and syntactic precision, making them crucial for accurately representing product descriptions.

In the technical process, the model adeptly converts concatenated sentence strings into these specialized embeddings. These embeddings generated by GTR-T5, are stored for use in the subsequent blocking phase. This approach highlights a significant aspect of our methodology: transforming diverse, unstructured data into vector formats specifically designed for comparative analysis. This step is fundamental, bridging the gap between raw textual data and an analytically robust format, paving the way for focused and efficient entity comparison in the blocking phase.

After generating initial embeddings, our model transitions into the blocking phase, designed to efficiently manage the volume of pairwise comparisons in large datasets. This is essential for controlling computational load and ensuring accuracy in data-rich environments. In this stage, product comparisons are strategically narrowed down, focusing on pairs more likely to be true matches.

Products are first grouped by category, facilitating comparisons between similar items and increasing the likelihood of identifying genuine matches. A similarity threshold is then applied to the GTR-T5 embeddings, balancing inclusivity and exclusivity to effectively identify potential matches. Pairs that meet this threshold are processed further, forming textual tuples—comprising keywords, product title, and brand—and assigned labels based on their GTINs, with 1 indicating matches and 0 for non-matches.

By reducing the number of comparisons, the blocking phase significantly cuts down on computational demands, enhancing the model's efficiency as detailed in Section 6.1.1. This focused approach not only streamlines the entity matching process but also prepares the data for the subsequent cross-encoder phase, which performs a deeper analysis on this refined dataset. Thus, the blocking phase plays a critical role in the model, demonstrating how targeted preprocessing and advanced NLP techniques can improve the efficiency of machine learning models in entity matching.

In the final phase of our model, the cross-encoder is employed, building upon the dataset refined in the blocking phase. This shift to a cross-encoder framework allows for a comprehensive analysis of text pairs, essential for precise entity matching. By jointly examining text pairs, the cross-encoder unravels the intricate contextual connections between them, crucial in determining whether different textual features point to the same product.

The strength of the cross-encoder in our setup is amplified by our use of transfer learning.

Drawing from a pre-trained model, we finetune it to suit our specific entity matching needs. This approach leverages the extensive linguistic knowledge embedded in the pre-trained model, significantly reducing the necessity for large-scale, domain-specific training. This method not only saves resources but also brings a depth of understanding to the model that would be challenging to achieve from scratch. Our decision thresholds, integral to the model's accuracy, are determined based on the model's performance on the validation set, aligning with the procedures outlined in Section 5.2.

In the testing phase, the cross-encoder carefully evaluates each pair passing the blocking stage. It assesses similarity scores derived from learned embeddings, which reflect a refined understanding of text relationships honed during the transfer learning process. The application of predetermined thresholds to these scores is a critical step, dividing the pairs into matches and non-matches. This evaluation phase is pivotal, directly impacting the effectiveness and reliability of the entity matching process.

The incorporation of the cross-encoder, particularly enhanced by transfer learning, marks a significant advancement in our approach to entity matching. Its capability to conduct a joint analysis of text pairs, enriched by the subtle understanding acquired through transfer learning, introduces a solid level of validity and depth to the process. Hence, the cross-encoder not only improves our approach but also signifies a deeper comprehension of complex product data relationships.

To conclude our model's architecture, the testing and evaluation phase focused primarily on TinyBERT due to hardware constraints. This phase tested the cross-encoder's ability to analyze text pairs for entity matching, comparing similarity scores with thresholds from our validation set. Such a step was crucial to assess both the accuracy and efficiency of the model, especially given our limited computational resources. This evaluation ensures the model's relevance and

applicability in real-world scenarios, despite the challenges presented by hardware limitations.

C.3 Advantages

This model significantly benefits from incorporating a cross-encoder architecture paired with pre-trained models like TinyBERT, MiniBERT, and RoBERTa variants. These models, having honed their linguistic capabilities on comprehensive datasets such as Wikipedia and BooksCorpus (Nils Reimers 2023), provide our system with an in-depth understanding of language intricacies crucial for differentiating subtle variances in product descriptions—a key aspect of entity matching.

The use of these pre-trained models streamlines our model's deployment, equipping it with robust linguistic skills essential for effective entity matching. This strategy bypasses the need for labor-intensive data gathering and initial training, leading to significant savings in time and resources. Additionally, their adaptability allows for efficient text interpretation, requiring only minimal additional training to tailor them to our dataset’s unique requirements.

The adaptability of pre-trained models is essential in our framework, offering immense value in scenarios where data and requirements are in constant change. This characteristic is especially vital in dynamic sectors like e-commerce, where rapid adaptation to evolving product data or consumer trends is crucial. Integrating these models into our cross-encoder setup not only streamlines efficiency but also exemplifies the real-world applicability of advanced NLP in complex tasks, such as entity matching, where quick, accurate decision-making is key.

In addition, the cross-encoder's architectural versatility was a focal point during development, allowing us to tailor the model to diverse operational contexts. We tested various transfer learning configurations trying to find a balance between training size, processing speed, and

computational load to suit our server's capacity. This flexibility is a significant asset, enabling the model to perform in various settings – from those with limited computational resources to environments where more extensive processing capabilities are available. This ability to customize the transfer learning ensures that the cross-encoder is capable of efficiently handling a wide array of entity matching tasks, making it a highly adaptable solution in the field of NLP models.

Addressing hardware limitations was pivotal in our model development. While larger models like RoBERTa showed promise in initial tests, hardware constraints limited their deployment. This also leaves room for potential improvements and an even more impactful NLP classifier in entity matching.

The ease of implementing the cross-encoder stands out as another significant advantage. Despite its sophistication, the integration process was streamlined, thanks to user-friendly NLP tools and libraries such as SentenceTransformers and Hugging Face's Transformers. These resources, complete with comprehensive documentation and community support, simplified the model's incorporation into our framework. This accessibility is crucial, making advanced NLP techniques attainable for a broader range of users and applications, beyond just e-commerce and data management.

Evaluating the cross-encoder's performance in entity matching requires a detailed view. Its proficiency in analyzing text pairs is offset by its processing efficiency. The model's thorough analysis, beneficial for understanding complex textual relationships, can lead to longer processing times. This aspect situates the cross-encoder as a moderate performer in our comparisons. Its effectiveness in scenarios with time constraints needs careful consideration, as rapid processing may be compromised.

In summary, the cross-encoder's straightforward implementation is paired with a balanced

performance in entity matching tasks. Its detailed text pair analysis brings depth to the process but affects speed. These traits underscore its role in our model, exemplifying advanced NLP's utility in complex tasks like e-commerce. The cross-encoder's application highlights the importance of aligning NLP tools with specific task requirements to optimize entity matching outcomes.

C.4 Limitations

The cross-encoder's architecture, while robust and accurate, introduces a significant challenge in terms of computational demand, particularly during the prediction phase. Although proving to be an advantage in some specific cases, it can also become a limitation in the model's processing time. The cross-encoder, designed to analyze text pairs jointly for in-depth understanding, necessitates considerable computational resources for each pair of texts processed.

This computational intensity becomes a critical factor when dealing with extensive datasets common in entity matching applications, such as those in e-commerce. The requirement to process a large number of product pairs means that the model's efficiency can quickly bottleneck, impacting scenarios where speed is paramount. The intensive computational demand poses challenges not only in terms of processing speed but also in scalability and accessibility, especially for users with limited computational resources.

As we consider the implications of this computational heaviness, it is essential to note its impact on the overall feasibility of the model, particularly in environments where quick response times are integral to the application's success.

Continuing from the discussion on computational demands, another significant limitation of our cross-encoder model is its reliance on effective blocking for efficiency, particularly when

dealing with dirty or unstructured data.

The scalability of this approach also poses a significant challenge, particularly with very large datasets or in real-time applications. As previously mentioned, the computationally intensive nature when using more complex NLP's limits its scalability in rapid processing environments. This poses to be a big problem in scenarios where large-scale online content tends to have a scale up demand capability.

The blocking phase is designed to reduce the computational load by filtering out unlikely matches before they reach the cross-encoder. However, this step's effectiveness is heavily contingent on the quality and structure of the input data. In scenarios where the data is dirty or unstructured, the blocking mechanism may struggle to accurately filter pairs for the cross-encoder. This issue becomes pronounced when the data lacks clear features or is inconsistent, making it challenging to form meaningful sentences or identify relevant tokens for effective comparison.

The cross-encoder's reliance on clean, structured data becomes evident in the blocking phase, where dirty or unstructured data can significantly hinder the model's performance. Poor data quality can lead to an overwhelming number of comparisons for the cross-encoder, increasing computational demands and potentially degrading the system's efficiency. Moreover, it may result in false negatives and positives, thereby diminishing the accuracy of entity matching.

Effective blocking is thus critical, forming a link in the model's performance chain where each step directly influences the subsequent one. Inefficiencies in the blocking phase can have a cascading effect, potentially compromising the outcome of the cross-encoder's analysis. To mitigate this, the blocking mechanism must be refined, and the input data must be meticulously prepared to align with the requirements of entity matching.

Dirty data is not only a limitation for blocking. In real-world scenarios, where good quality data is often not available, the cross-encoder's matching performance may also suffer. The absence of clean, feature-rich data can impede the model's ability to discern valuable insights, resulting in suboptimal entity matching. This underlines the importance of robust data preparation practices and a comprehensive data pipeline to maintain the integrity of the input data.

By addressing these challenges head-on, refining the blocking phase, and committing to rigorous data preparation, the model's efficiency and accuracy can be substantially improved. This focused approach will ensure that the model can be effectively utilized across a variety of real-world applications, fully harnessing its potential in the field of entity matching.

Transitioning from the challenges related to blocking efficiency and data quality, another challenge with the cross-encoder model: its transparency, or rather, the lack thereof. This aspect, while often overshadowed by more technical challenges, is crucial in understanding and improving the model's functionality.

The cross-encoder's complexity can obscure the reasoning behind its predictions, a common challenge among advanced deep learning systems. This black box nature is particularly problematic in entity matching, where reasons for specific pairings are as important as the matches themselves. Such insight is essential for finetuning, mitigating biases, and enhancing interpretability.

As we conclude this chapter, we recognize that the model's performance and flexibility are counterbalanced by its computational intensity, reliance on data quality, and opacity. These challenges, while not undermining the model's value, underscore the necessity for ongoing improvement to fulfill the model's potential and meet user expectations.

C.5 Further Improvements

Exploring more advanced pre-trained models in the cross-encoder framework offers significant potential for enhancing entity matching. More advanced models like larger BERT variants or RoBERTa, known for their deep linguistic analysis capabilities, could substantially improve the model's performance. However, as previously mentioned, our venture into these sophisticated architectures was limited by computational constraints. Initial tests on a small dataset segment showed encouraging results, suggesting that with adequate computational resources, these advanced models could greatly improve the model’s performance.

This exploration highlights a critical challenge: the balance between pursuing advanced NLP technologies and the practicality imposed by computational limitations. Addressing this gap is essential for broader application and accessibility of these advanced NLP tools, particularly for smaller entities or individual researchers.

In the field of Natural Language Processing, cross-encoders and bi-encoders represent two distinct approaches to encoding and analyzing text. Our model performs a joint analysis of text pairs, creating a single embedding that captures the relationship between texts, while bi-encoders encode texts independently, efficient for large dataset comparisons, as seen in Figures C.2 and C.3.

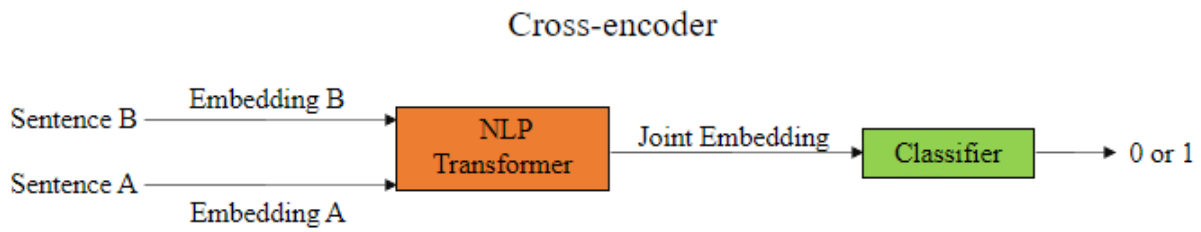


Figure C.2: Cross-encoder overview.

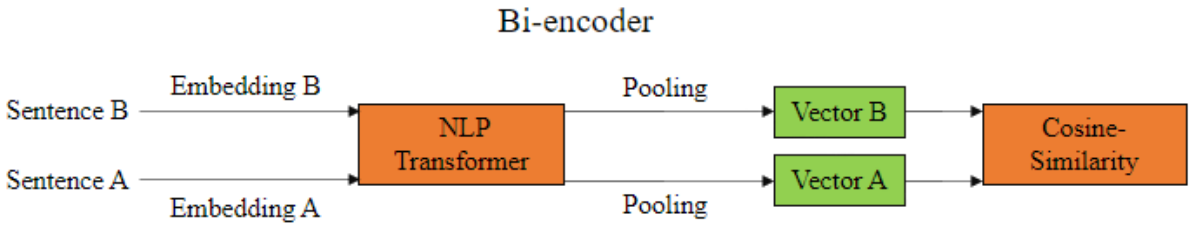


Figure C.3: Bi-encoder overview.

As shows in the image, bi-encoders architecture features two separate encoders that independently process different inputs, such as sentences or paragraphs. Typically, transformer based, these encoders convert inputs into fixed-length vector representations in a shared embedding space. The core of bi-encoders lies in their ability to independently process inputs. Given its characteristics, it is tendentially a better tool in dealing with a large amount of data. A study suggests that "As the encoded sentences can be cached and reused, bi-encoding is much more efficient, and the outputs of a bi-encoder can be used off-the-shelf as sentence embeddings for downstream tasks." (Liu and Jiao 2022)

Given the characteristics of each model, a hybrid approach, combining the strengths of both architectures, offers a compelling direction for future research. Initially, a bi-encoder could quickly iterate through vast text pairs, isolating likely matches. Then, a cross-encoder could conduct detailed analysis on this subset, applying its deep contextual understanding and more complex architecture when evaluating less comparisons.

This system could provide bi-encoder scalability for initial filtering, followed by the detailed analysis of cross-encoders. This integration could significantly enhance entity matching, merging computational efficiency with analytical depth. Further research into seamlessly combining these models could yield a versatile tool for various NLP applications.

In practical scenarios, like e-commerce product catalog management or customer service query categorization, this model could be revolutionary. For instance, in e-commerce, a bi-encoder might efficiently group products, with the cross-encoder refining product matches within these groups, enhancing efficiency and user experience.

However, technical challenges in integration exist. Aligning the bi-encoder's independent data encoding with the cross-encoder's joint analysis may require innovative methods, possibly through an intermediary layer or algorithm that harmonizes the outputs and inputs of each stage, thus optimizing the system's overall functionality.

C.6 Conclusion

The exploration of the cross-encoder model within the sphere of entity matching has been quite revealing, offering significant insights into its advanced capabilities in the field of natural language processing. By incorporating pre-trained models, the cross-encoder has effectively demonstrated its proficiency in parsing and interpreting complex textual information. This particular ability is of paramount importance, especially when it comes to the precise and accurate identification of product matches, thereby underscoring the considerable value of the model in tasks that demand a high level of detailed analysis.

From the initial model architecture to the intricate blocking phase, we've seen the cross-encoder's design tailored specifically for entity matching. Despite the computational intensity, the model's strategic approach to data processing sets the stage for its ability to identify genuine

matches, reaffirming its place as a specialized tool in NLP applications.

The cross-encoder model showcases its strengths with advanced pre-trained models and an adaptable architecture, enhancing efficiency for complex text analysis in entity matching. This prowess, especially useful in e-commerce, enables sophisticated processing of text pairs, demonstrating its practical applicability.

However, this model also faces challenges. Its capacity for detailed analysis, while beneficial, demands significant computational resources, particularly in handling extensive datasets. This requirement poses a contrast: the model’s depth of analysis, an advantage for performance, is counterbalanced by practical limitations in computational intensity. Such a juxtaposition between analytical depth and resource demands highlights the need for a balanced application of the model, aligning its strengths with operational constraints.

The exploration of future improvements paints a promising picture for the cross-encoder model, especially with the intriguing possibility of integrating it with bi-encoder architectures. Such a combination could leverage the cross-encoder's contextual understanding and the bi-encoder's swift data processing, potentially revolutionizing entity matching. This blend could offer a solution that efficiently navigates large datasets while maintaining the analytical depth required for accurate matching.

Expanding on this, the cross-encoder's journey reflects a dynamic landscape in NLP, where continuous innovation is key. Its potential synergies with other models suggest an exciting future, where it could offer even more sophisticated solutions for complex challenges in entity matching. This evolving role of the cross-encoder, driven by advancements and collaborations, reinforces its significance in advancing the capabilities of NLP applications.

Looking ahead, the cross-encoder's development is poised to be significantly influenced by

emerging trends in artificial intelligence and natural language processing. Advancements in areas like machine learning interpretability, real-time processing, and automated data cleaning could further refine the model's efficiency and application scope. This evolution, in tandem with ongoing research in NLP, promises to expand the cross-encoder's capabilities, making it more adaptable to the ever-increasing complexity and variety of data in our digital world.

8. References

Barlaug, Nils, and Jon Atle Gulla. 2021. “Neural Networks for Entity Matching: A Survey.” *ACM Trans. Knowl. Discov. Data* 15 (3). https://doi.org/10.1145/3442200.

Bilenko, Mikhail, Beena Kamath, and Raymond J Mooney. 2006. “Adaptive Blocking: Learning to Scale Up Record Linkage.” In *Sixth International Conference on Data Mining (ICDM’06)*, 87–96. https://doi.org/10.1109/ICDM.2006.13.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. “Enriching Word Vectors with Subword Information.” *Transactions of the Association for Computational Linguistics* 5 (June): 135–46. https://doi.org/10.1162/tacl_a_00051.

Brunner, Ursin, and Kurt Stockinger. 2020. “Entity Matching with Transformer Architectures - a Step Forward in Data Integration.” In *Proceedings of EDBT 2020*, 463–73. OpenProceedings. https://doi.org/10.5441/002/edbt.2020.58.

Bungee Tech. 2023. “ML Product Matching: A Key Factor to Retail Success.” Https://Bungeetech.Com/Product-Matching-Key-to-Success/. December 2, 2023.

Campello, Ricardo J G B, Davoud Moulavi, and Joerg Sander. 2013. “Density-Based Clustering Based on Hierarchical Density Estimates.” In *Advances in Knowledge Discovery and Data Mining*, edited by Jian Pei, Vincent S Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, 160–72. Berlin, Heidelberg: Springer Berlin Heidelberg.

Cordasco, G., & Gargano, L. 2010. “Community Detection via Semi-Synchronous Label Propagation Algorithms.” *Business Applications of Social Network Analysis (BASNA)*, December.

Dunn, Halbert L. 1946. “Record Linkage.” *American Journal of Public Health and the Nations Health* 36 (12): 1412–16. https://doi.org/10.2105/AJPH.36.12.1412.

Ebraheem, Muhammad, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2018. “Distributed Representations of Tuples for Entity Resolution.” *Proc. VLDB Endow.* 11 (11): 1454–67. https://doi.org/10.14778/3236187.3236198.

Elmagarmid, Ahmed K, Panagiotis G Ipeirotis, and Vassilios S Verykios. 2007. “Duplicate Record Detection: A Survey.” *IEEE Transactions on Knowledge and Data Engineering* 19 (1): 1–16. https://doi.org/10.1109/TKDE.2007.250581.

Galhotra, Sainyam, Donatella Firmani, Barna Saha, and Divesh Srivastava. 2021. “BEER: Blocking for Effective Entity Resolution.” In *Proceedings of the 2021 International Conference on Management of Data*, 2711–15. SIGMOD ’21. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3448016.3452747.

Ge, Congcong, Pengfei Wang, Lu Chen, Xiaoze Liu, Baihua Zheng, and Yunjun Gao. 2023. “CollaborEM: A Self-Supervised Entity Matching Framework Using Multi-Features Collaboration.” *IEEE Transactions on Knowledge and Data Engineering* 35 (12): 12139–52. https://doi.org/10.1109/TKDE.2021.3134806.

Grips. 2023. “About.” Https://Gripsintelligence.Com/About. December 18, 2023.

Hamilton, William, Rex Ying, and Jure Leskovec. 2017. “Inductive Representation Learning on Large Graphs.” https://cs.stanford.edu/people/jure/pubs/graphsage-nips17.pdf.

He, Kaiming, Ross Girshick, and Piotr Dollar. 2019. “Rethinking ImageNet Pre-Training.” In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 4917–26. https://doi.org/10.1109/ICCV.2019.00502.

Hugging Face. 2023. “BERT.” https://huggingface.co/Docs/Transformers/Model_doc/Bert. November 30, 2023.

Li, Bo-Han, Yi Liu, An-Man Zhang, Wen-Huan Wang, and Shuo Wan. 2020. “A Survey on Blocking Technology of Entity Resolution.” *Journal of Computer Science and Technology* 35 (4): 769–93. https://doi.org/10.1007/s11390-020-0350-4.

Li, Yuliang, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. “Deep Entity Matching with Pre-Trained Language Models.” *Proc. VLDB Endow.* 14 (1): 50–60. https://doi.org/10.14778/3421424.3421431.

Li, Yuliang, Jinfeng Li, Yoshihiko Suhara, Jin Wang, Wataru Hirota, and Wang-Chiew Tan. 2021. “Deep Entity Matching: Challenges and Opportunities.” *J. Data and Information Quality* 13 (1). https://doi.org/10.1145/3431816.

Liu, Fangyu, and YunLong Jiao. 2022. “Improving Unsupervised Sentence-Pair Comparison.” Https://www.amazon.science/Blog/Improving-Unsupervised-Sentence-Pair-Comparison. 2022.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. “Distributed Representations of Words and Phrases and Their Compositionality.” In *Advances in Neural Information Processing Systems*, edited by C J Burges, L Bottou, M Welling, Z Ghahramani, and K Q Weinberger. Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

Mudgal, Sidharth, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. “Deep Learning for Entity Matching: A Design Space Exploration.” In *Proceedings of the 2018 International Conference on Management of Data*, 19–34. SIGMOD ’18. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3183713.3196926.

Newcombe, H B, J M Kennedy, S J Axford, and A P James. 1959. “Automatic Linkage of Vital Records.” *Science* 130 (3381): 954–59. https://doi.org/10.1126/science.130.3381.954.

Ni, Jianmo, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Ábrego, Ji Ma, Vincent Zhao, et al. 2021. “Large Dual Encoders Are Generalizable Retrievers.”

Nils Reimers. 2023. “Sentence Transformers - Cross-Encoders.” https://huggingface.co/Cross-Encoder. December 18, 2023.

Nin, Jordi, Victor Muntes-Mulero, Norbert Martinez-Bazan, and Josep-L. Larriba-Pey. 2007. “On the Use of Semantic Blocking Techniques for Data Cleansing and Integration.” In *11th International Database Engineering and Applications Symposium (IDEAS 2007)*, 190–98. https://doi.org/10.1109/IDEAS.2007.4318104.

Novak, Jiri, and David Hoksza. 2010. “Parametrised Hausdorff Distance as a Non-Metric Similarity Model for Tandem Mass Spectrometry.” In *CEUR Workshop Proceedings*, 567:1–12.

Papadakis, George, and Themis Palpanas. 2016. “Blocking for Large-Scale Entity Resolution: Challenges, Algorithms, and Practical Examples.” In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, 1436–39. https://doi.org/10.1109/ICDE.2016.7498364.

Papadakis, George, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2020. “Blocking and Filtering Techniques for Entity Resolution: A Survey.” *ACM Comput. Surv.* 53 (2). https://doi.org/10.1145/3377455.

Papadakis, George, Jonathan Svirsky, Avigdor Gal, and Themis Palpanas. 2016. “Comparative Analysis of Approximate Blocking Techniques for Entity Resolution.” *Proc. VLDB Endow.* 9 (9): 684–95. https://doi.org/10.14778/2947618.2947624.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. “GloVe: Global Vectors for Word Representation.” In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, edited by Alessandro Moschitti,

Bo Pang, and Walter Daelemans, 1532–43. Doha, Qatar: Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1162.

Primpeli, Anna, Ralph Peeters, and Christian Bizer. 2019. “The WDC Training Dataset and Gold Standard for Large-Scale Product Matching.” In *WWW ’19: Companion Proceedings of The 2019 World Wide Web Conference*, 381–86. https://doi.org/10.1145/3308560.3316609.

Statista. 2023. “ECommerce - Worldwide: Statista Market Forecast.” https://www.statista.com/outlook/emo/ecommerce/worldwide.

Suk, Heung-Il. 2017. “Chapter 1 - An Introduction to Neural Networks and Deep Learning.” In *Deep Learning for Medical Image Analysis*, edited by S Kevin Zhou, Hayit Greenspan, and Dinggang Shen, 3–24. Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-810408-8.00002-X.

Thirumuruganathan, Saravanan, Han Li, Nan Tang, Mourad Ouzzani, Yash Govind, Derek Paulsen, Glenn Fung, and AnHai Doan. 2021. “Deep Learning for Blocking in Entity Matching: A Design Space Exploration.” *Proc. VLDB Endow.* 14 (11): 2459–72. https://doi.org/10.14778/3476249.3476294.

UNCTAD. 2021. “Global E-Commerce Jumps to $26.7 Trillion, COVID-19 Boosts Online Sales.” https://unctad.org/News/Global-e-Commerce-Jumps-267-Trillion-Covid-19-Boosts-Online-Sales. May 3, 2021.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. “Attention Is All You Need.” In *Advances in Neural Information Processing Systems*, edited by I Guyon, U Von Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett. Vol. 30. Curran Associates, Inc.

https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vidhya, K A, and T V Geetha. 2019. “Entity Resolution and Blocking: A Review.” In *2019 IEEE 9th International Conference on Advanced Computing (IACC)*, 133–40. https://doi.org/10.1109/IACC48062.2019.8971572.

Wang, Jiannan, Guoliang Li, Jeffrey Xu Yu, and Jianhua Feng. 2011. “Entity Matching: How Similar Is Similar.” *Proc. VLDB Endow.* 4 (10): 622–33. https://doi.org/10.14778/2021017.2021020.

Wolfe, Cameron. 2023. “T5: Text-to-Text Transformers.” March 27, 2023.

Zeakis, Alexandros, George Papadakis, Dimitrios Skoutas, and Manolis Koubarakis. 2023. “Pre-Trained Embeddings for Entity Resolution: An Experimental Analysis.” *Proc. VLDB Endow.* 16 (9): 2225–38. https://doi.org/10.14778/3598581.3598594.

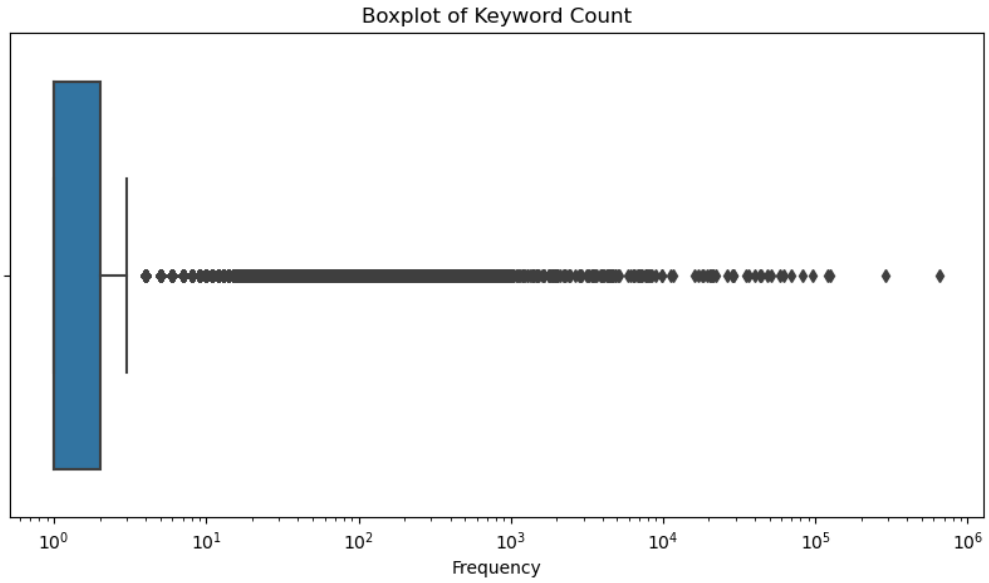
Zhang, Wei, Hao Wei, Bunyamin Sisman, Xin Luna Dong, Christos Faloutsos, and Davd Page. 2020. “AutoBlock: A Hands-off Blocking Framework for Entity Matching.” In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 744–52. WSDM ’20. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3336191.3371813.

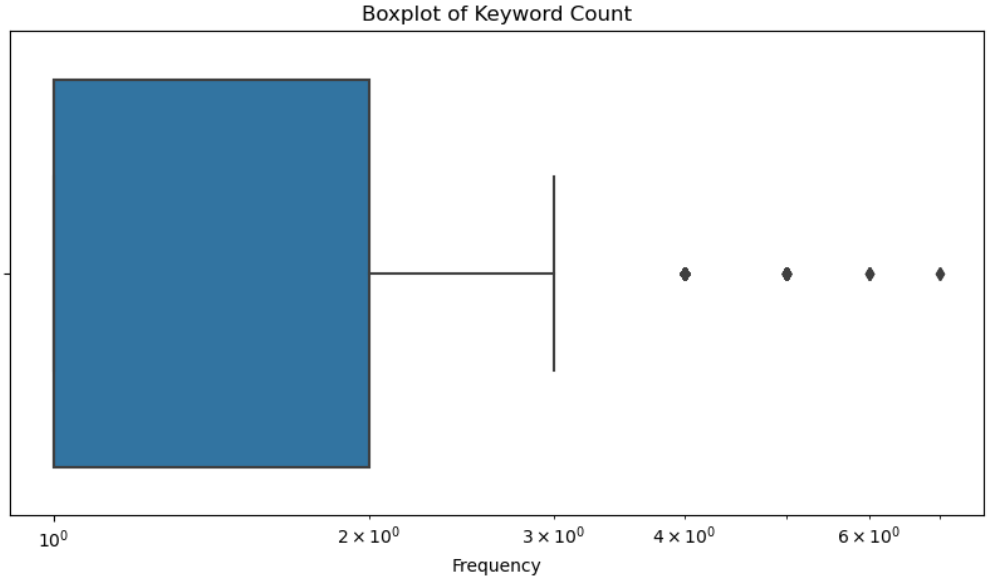
9. Appendix

9.1 Data Dictionary

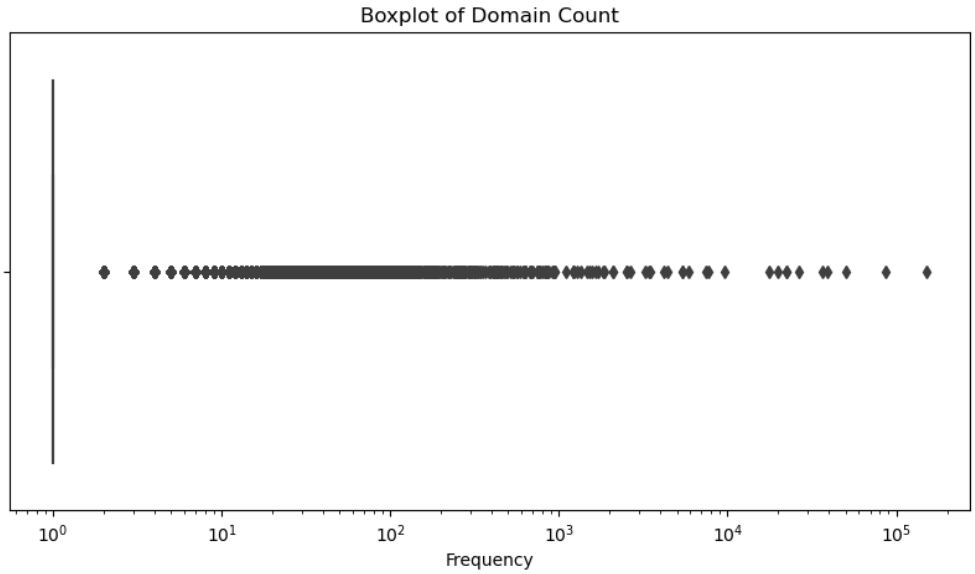
Feature	Description	Type	Examples
keyword	Keyword on google search	Text	Baseus
domain	Top level domain of the website	Text	amazon.co.uk
gtin	Number identifying a product - standardised across domains	Float	28887041.0
url	Full URL of the product	Text	https://goaxil.com/products/battery-10
producttitle	Title of the product as scraped from metadata	Text	Iphone 14 128 GB
brand	Brand of the product as scraped from metadata	Text	Duracell
sku	Alphanumeric code identifying a product within a retailer – not standardised across domains	Text	B09YH85SKV
category	Category of the product (prediction by Grips)	Text	Electronics > Communications
countrycode	Identifier of the country	Text	US
currency	Currency of the product price	Text	USD
maxprice	Price of the product	Float	8.49
maxpriceusd	Price of the product in USD	Float	17.504336
imageurl	URL to the image of the product	Text	https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcSInGwLtBnrpSICNMOc31qptezzP0EvhQPQx8FUrkOw-KbT0zTw

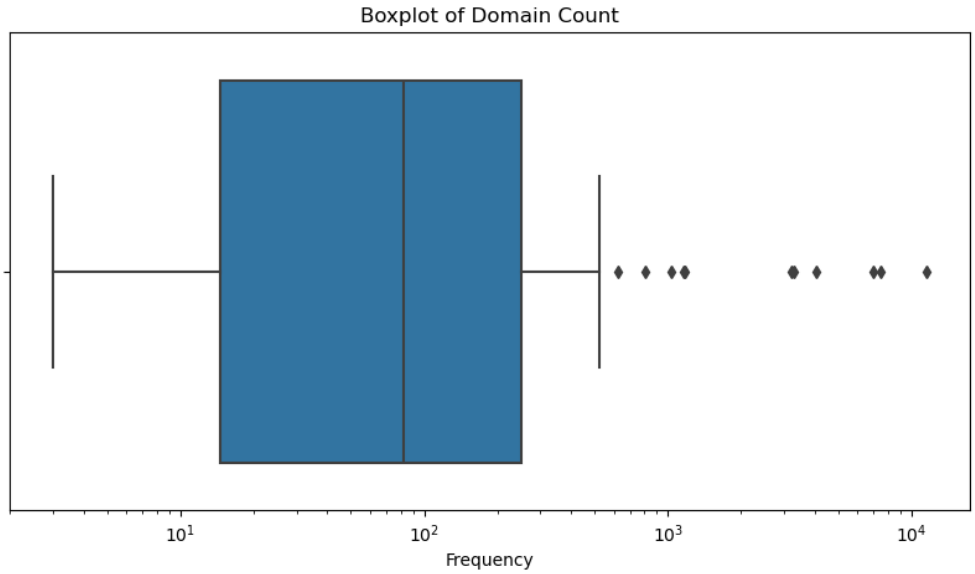
9.2 Keyword Count Before and After Cleaning



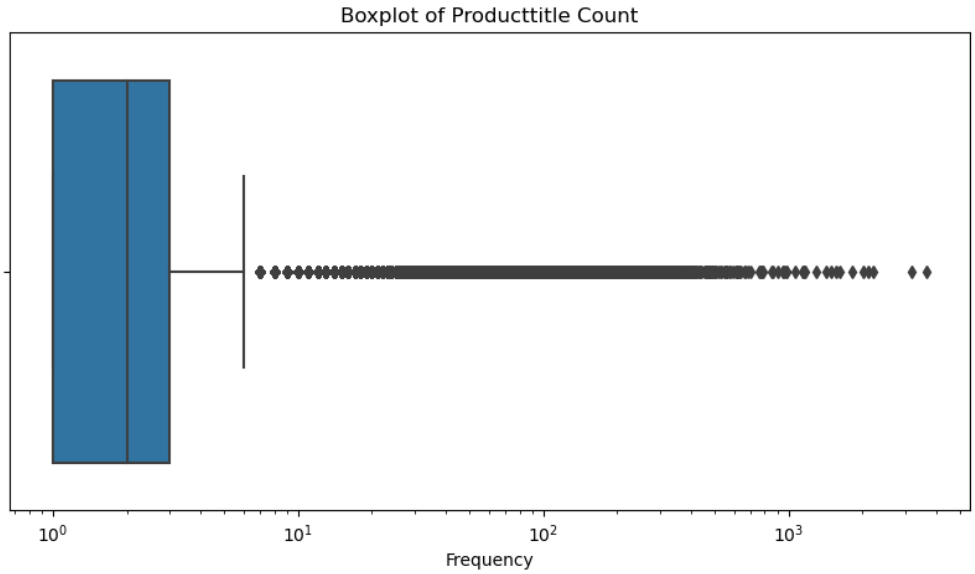


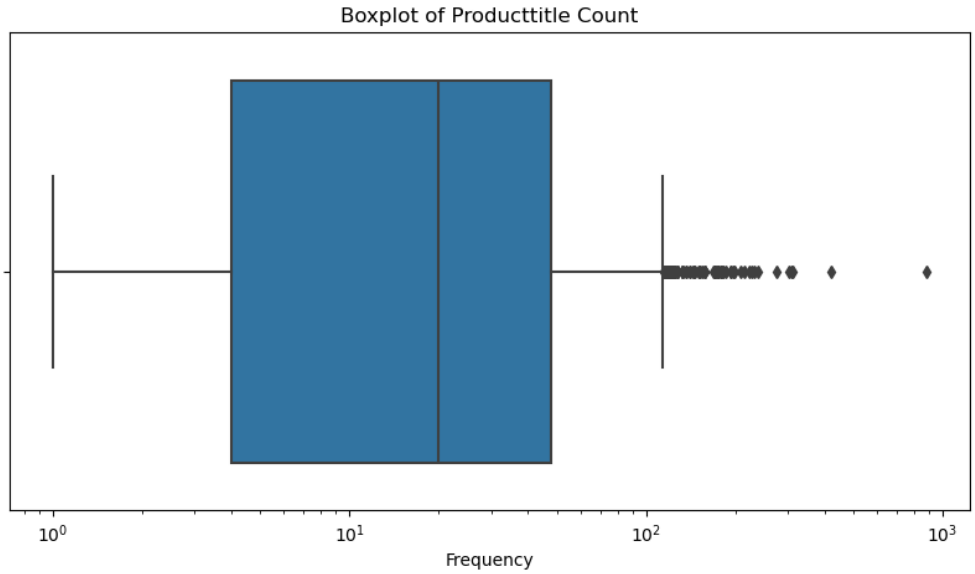
9.3 Domain Count Before and After Cleaning



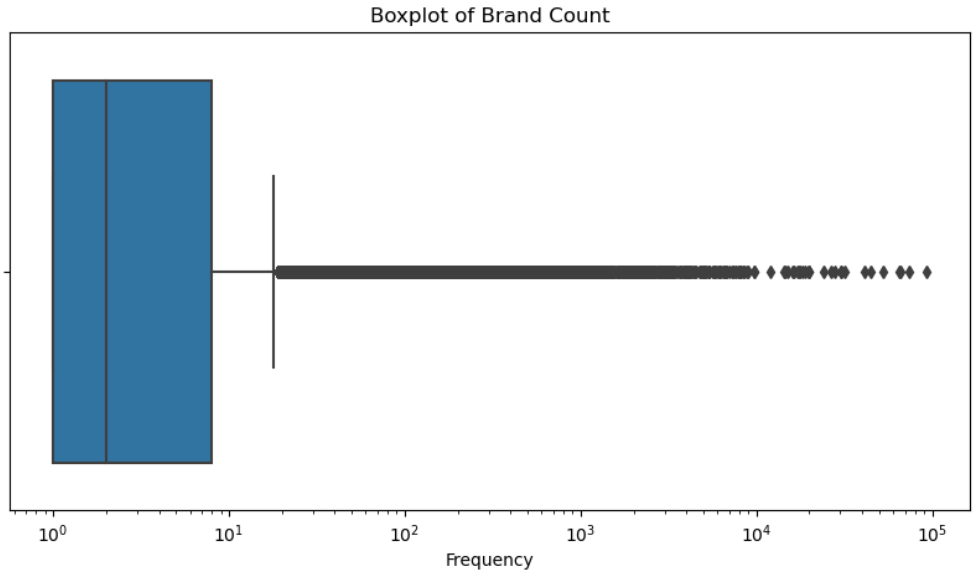


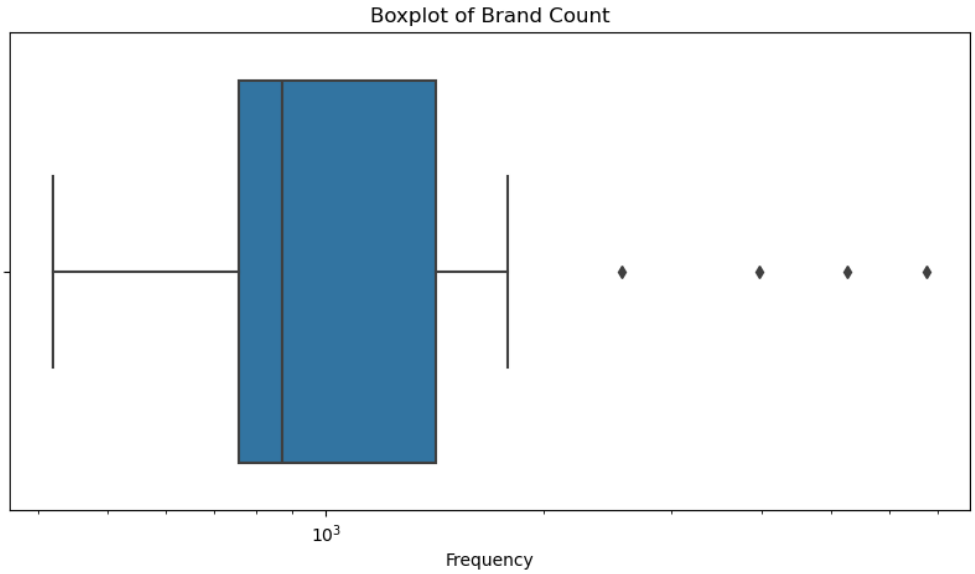
9.4 Product Title Count Before and After Cleaning



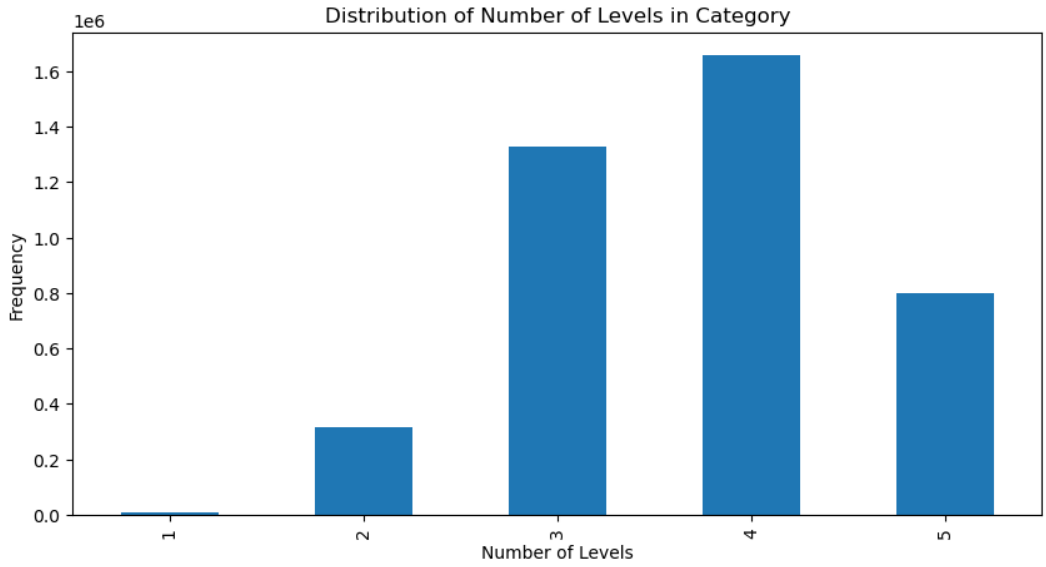


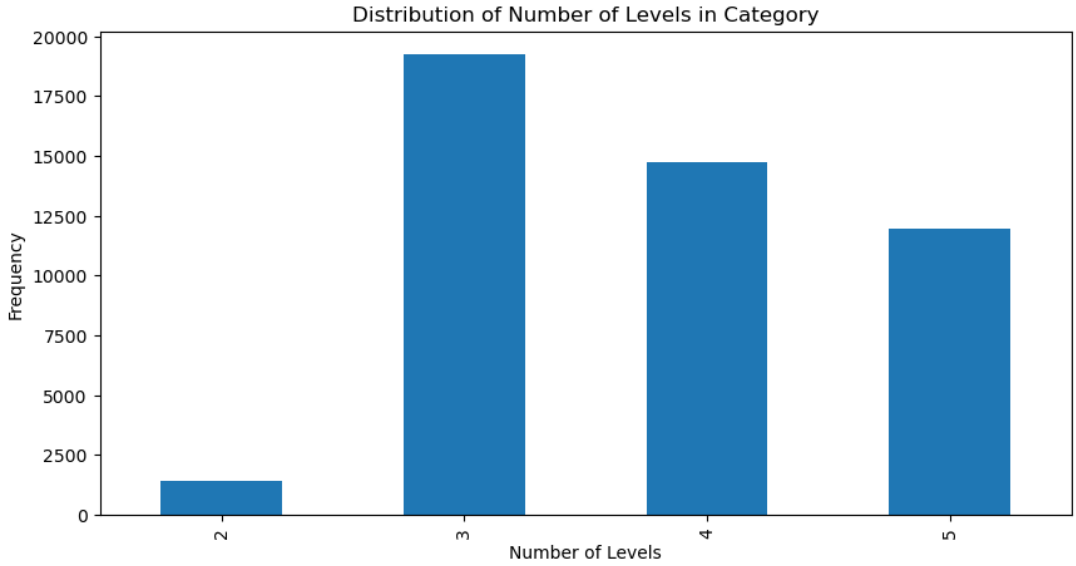
9.5 Brand Count Before and After Cleaning



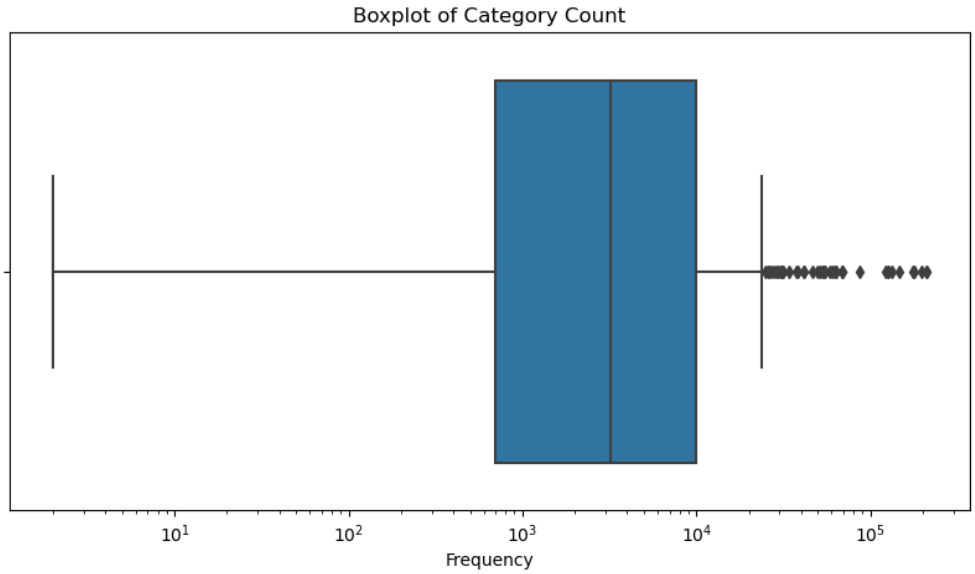


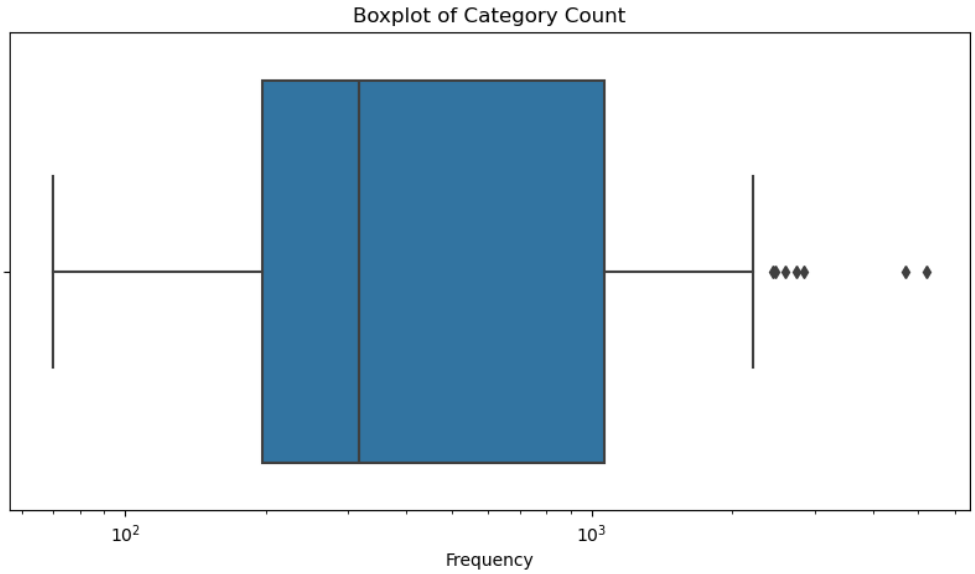
9.6 Distribution of Category Levels Before and After Cleaning





9.7 Category Count Before and After Cleaning





9.8 Price Distribution Before and After Cleaning