Ministry of Science and Higher Education of the Republic of Kazakhstan

SDU University

Ospan Smagul

Item matching based on image and text analysis

THESIS

Presented in Partial Fulfilment for the

Master of Technical Sciences Degree in Computer Science

(degree code: 7M06102)

Department of Computer Science

Faculty of Engineering and Natural Sciences

Supervisor: **Associate Professor, PhD Bogdanchikov Andrey**

Kaskelen 2024

SDU University
Faculty of Engineering and Natural Sciences
Department of Computer Science

Dean of Faculty

Associate Professor

PhD Ahmedov R.

_____

« _____ »_____2024

Topic of the thesis:

Item matching based on image and text analysis

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science" SDU University

Head of Department _____ Assistant Professor, PhD Mukash Zh.

Academic Supervisor ___ Associate Professor, PhD Bogdanchikov A.

Master student _____ Ospan Smagul

Kaskelen 2024

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Ospan Smagul

June, 2024

Acknowledgements

Thanks to the government of Republic of Kazakhstan for providing a grant and free education. Thanks to my supervisor for giving a great deal of knowledge. And for the fact that he professionally supervised the writing of the thesis, for his patience and for making my thesis understandable. Thanks for correcting me when I made mistakes and for motivating when there was no time. thank you for doing your job honestly. Thanks to my group mates for being with me on this path. And for my family for believing in me.

Dedication

This thesis is dedicated to my friends and family.

I also want to to dedicate this thesis to all the others who have supported me. Their contributions have been important in making this project a reality.

Abstract

In recent years, the e-commerce sector has experienced exponential growth as more retailers and brands have opened online sites to reach a worldwide consumer base. The rise in online marketplaces and comparison sites, where consumers may evaluate and buy goods from different vendors or companies, is another effect of the digital transformation. But this has also resulted in a proliferation of sellers and products, making it challenging for buyers to identify the ideal items and for vendors to connect with their intended clientele. Product matching has emerged as a critical problem for efficient decision-making in the retail and supply chain management sectors in response to these challenges. Purpose of the product matching is to match identical or nearly same products across various sources across, such as different e-commerce websites, based on features and their attributes. It can help retailers to find in demand products and boost sales. On the other hand, customers can take benefit from technologies by finding products they needed, compare across different aggregators, and make decisions to purchase. However, product matching is a challenging task. In most cases same products have different names, descriptions and image cards. With its recent breakthroughs in object detection and image categorization, deep learning has come a long way. A convolutional neural network may identify identical products based on the input picture and text, which might be a label or tag. In this work, three pre-trained deep convolutional models: MobileNet-V2, VGG-19, and ResNet-50, are implemented to find the most identical products based on text and image data. The best model for similarity detection based on text and image has been found using a variety of performance assessment techniques, including cosine similarity, Levenshtein distance, and custom metric score. The study concludes that the Mobilenet model is the best suitable model for handling these laborious tasks and supporting the development of a digital strategy plan.

Аңдатпа

Қазіргі кезде электрондық коммерция индустриясы экспоненциалды түрде өсіп жатқанынды көріп тұрмыз. Цифрлық трансформация сонымен қатар тұтынушылар әртүрлі бизнестен тауарларды салыстырып, сатып ала алатын онлайн сайттар санының артуына әкелді. Бұл сонымен қатар сатушылар мен тауарлар санының артуына әкеліп жатыр. Осы мәселені шешу үшін тауарларды сәйкестендіру тізбегін басқаруда және бөлшек сауда салаларында тиімді шешім қабылдау үшін алгоритм жазуға міндет болды. Тауарларды сәйкестендіру белгілі бір атрибуттарға немесе мүмкіндіктерге негізделген әртүрлі электрондық коммерция веб-сайттары сияқты әртүрлі көздер бойынша ұқсас немесе бірдей өнімдерді сәйкестендіру процесін білдіреді. Бұл саудагерлер сұранысқа ие өнімдерді анықтауға, олардың қорын басқаруды жақсартуға және сатуды арттыруға көмектеседі. Сонымен қатар тұтынушыларға іздеген өнімдерін табуға, бағаларды салыстыруға және сатып алу туралы негізделген шешім қабылдауға көмектеседі. Алайда, тауарларды сәйкестендіру күрделі мәселе болып табылады. Басты себептері өнімдердің әртүрлі атаулары, сипаттамалары және атрибуттары болуы мүмкін немесе әртүрлі санаттар мен ішкі санаттарда тізімделуі мүмкін. Объектіні анықтау және кескінді санаттаудағы соңғы жетістіктерімен терең оқыту ұзақ жолдан өтті. Конволюциялық нейрондық желіні пайдаланып, кіріс кескіні мен мәтінге негізделген ұқсас элементтерді сәйкестендіру болуы мүмкін. Бұл зерттеуде мәтіндік және кескін деректерін пайдалана отырып, қай кескіндердің ең ұқсас екенін анықтау үшін алдын ала дайындалған үш терең конволюционды модель (MobileNet-V2, VGG-19 және ResNet-50) пайдаланылады. Мәтін мен кескінге негізделген ұқсастықты анықтаудың ең жақсы үлгісі косинус ұқсастығы, Левенштейн қашықтығы және теңшелетін метрикалық ұпайды қоса алғанда, өнімділікті бағалаудың әртүрлі әдістерін қолдану арқылы табылды. Зерттеу нәтижесінде Mobilenet моделі осы ауыр міндеттерді шешуге ең қолайлы методі болып табылатын деген қорытындыға келдік.

Аннотация

В последние годы рынок электронной коммерции растет в геометрической прогрессии. Цифровая трансформация также привела к увеличению количества онлайн-торговых площадок и сайтов аггрегаторов, где клиенты могут сравнивать и приобретать продукты из различных магазинов. Однако это также привело к увеличению количества продуктов и продавцов. Чтобы решить эти проблемы, подбор продуктов стал важнейшей задачей для эффективного принятия решений в сфере управления цепочками поставок и розничной торговли. Сопоставление продуктов — это процесс сопоставления похожих или идентичных продуктов из различных источников, таких как различные веб-сайты электронной коммерции, на основе определенных атрибутов или функций. Это помогает ритейлерам и брендам определять продукты, пользующиеся спросом, улучшать управление запасами и увеличивать продажи. Это также помогает клиентам находить нужные продукты, сравнивать цены и принимать обоснованные решения о покупке. Однако сопоставление продуктов — сложная и трудная задача, поскольку продукты могут иметь разные названия, описания и атрибуты и могут быть перечислены в различных категориях и подкатегориях. Благодаря недавним прорывам в области компьютерного зрения и категоризации изображений глубокое обучение шагнуло далеко вперед. Используя сверточную нейронную сеть, можно идентифицировать сходные элементы на основе входного изображения и текста, который может быть меткой или тегом. В этом исследовании используются три предварительно обученные глубокие сверточные модели (MobileNet-V2, VGG-19 и ResNet-50), чтобы определить, какие изображения наиболее похожи, используя текстовые и графические данные. Лучшая модель для обнаружения сходства на основе текста и изображения была найдена с использованием различных методов оценки производительности, включая косинусное сходство, расстояние Левенштейна и оценку пользовательской метрики. В исследовании делается вывод, что модель Mobilenet является наиболее подходящей моделью для решения этих трудоемких задач и поддержки разработки плана цифровой стратегии.

Abbreviations

ML — Machine Learning

NLP — Natural Languages Processing

CV — Computer Vision

VGG — Visual Geometry Group

ResNet — Residual Neural Network

BERT — Bidirectional Encoder Representations from Transformers

TFIDF — Term Frequency-Inverse Document Frequency

CNN — Convolutional neural network

KNN — K-Nearest Neighbors

tf — term frequency

idf — inverse document frequency

UPC — Universal Product Code

GTIN — Global Trade Item Number

Table of Contents

Declaration i

Acknowledgements ii

Dedication iii

List of Abbreviations vii

1 Introduction 1

1.1 Motivation . . . 2

1.2 Literature review . . . 3

1.3 Related works . . . 6

1.4 Statement of Results . . . 8

2 Background 10

2.1 Natural Language Processing . . . 10

2.2 Computer Vision . . . 17

2.3 Product matching . . . 20

3 Methodology 24

3.1 Data Preprocessing . . . 24

3.2 Matching methods analysis . . . 26

3.2.1 VGG-19 . . . 26

3.2.2 MobileNet-V2 . . . 28

3.2.3 ResNet-50 . . . 30

4 Results **32**
4.1 Quantity of data 32
4.2 Main results of work 34
4.2.1 Results 35

5 Conclusion **39**

Bibliography **41**

A Preprocessing **44**

B Parsing image urls **45**

C Preprocessing images **46**

D Evaluation **47**

E Web platform backend **48**

F Web platform frontend **52**

Chapter 1

Introduction

In recent years, too many people and industries have become interested in the application of AI to their needs. Customers need to find necessary products from multiple aggregators, and year by year it becomes more challenging due to the large amount of offers. On the other hand, industries need to be competitive among all similar business offers. They want to analyze competitors by comparing their prices, stock availability, descriptions, or even reviews from customers. Also, big companies have started to implement dynamic price changes, which mostly rely on competitors' prices. Examples of such companies are Amazon, eBay, Wildberries, Ozon, and Kaspi. Companies like Technodom, Sulpak, and Mechta have the same products in stock, so it will be beneficial for them too to compare their own products with competitors' stock. So there has become a need for an accurate product matching algorithm. Currently, matching algorithms are based on either image comparison or textual comparison by description and name of the item. Those methods are not really effective in comparison with the hybrid method of both textual and image data comparison. We started by comparing various methods of text and image analysis of products, and then we used analysis from both methods to measure accuracy. An example of a product card is shown in figure 1.1 This thesis investigates cutting edge item matching methods. Our proposal is to assess and contrast various methods by utilizing both public matching benchmarks and up-to-date industrial data. We also offer suggestions on how various approaches may be applied to problems with product matching.

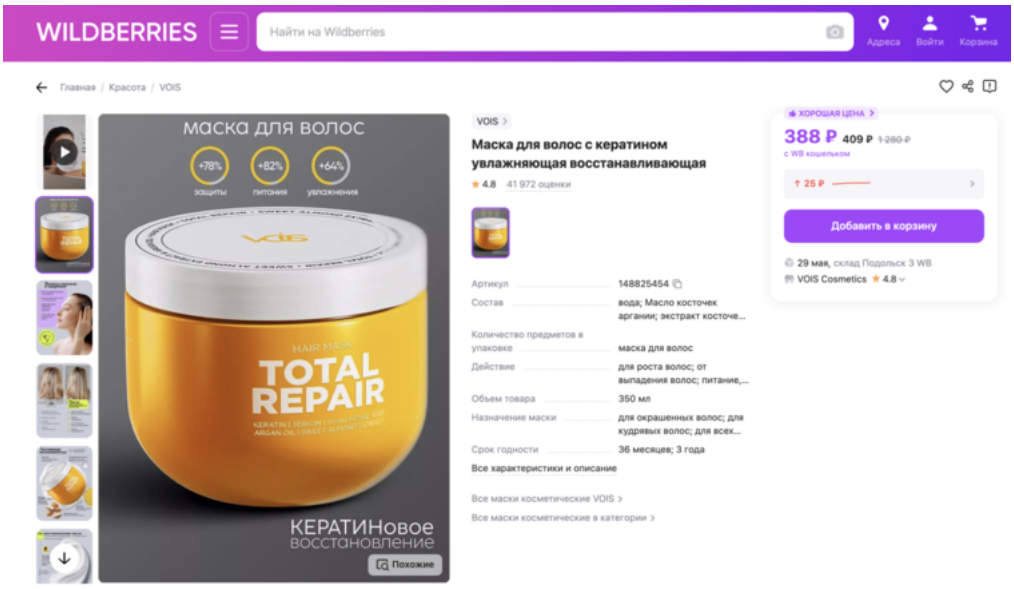

Figure 1.1: Image card sample.

1.1 Motivation

The quest for efficient product matching has been the guiding principle behind a lot of AI research powered by machine learning. It is clear that various machine learning techniques have been tried out and compared with respect to product matching yet none is the best algorithm for it. So as to enable them make smart choices and outcompete their business rivals, businesses have to select the most appropriate machine learning algorithm for product matching. One might find it hard to pick out which machine learning algorithm would work best from a range of them as these are software and so their efficiency would vary according to their characteristics, they were designed to solve and data sets they operate with. Furthermore, relative examinations measuring product similarity or dissimilarity by means of embeddings can be useful. The results of this study should help businesses make decisions by offering insightful information about the suitability of various machine learning algorithms when paired with specific embedding techniques. In addition to offering suggestions for further research in this area, the study will add to the body of information already available on machine learning algorithms for product matching.

To sum up, current retailers are more and more interested in applying deep learning and machine learning algorithms to keep customers returning by fore-

casting product demand, coming up with personalized offers and more. It is one of the useful steps before employing any of these activities is product matching, which can be employed for both cleansing and enriching product data.

1.2 Literature review

Numerous studies and approaches to related problems have been made in the field of Product Similarity Matching. Problems with product matching have been addressed with both supervised and unsupervised learning approaches. In [1], four different classifiers are trained for product matching: Random Forest, Support Vector Machines (SVM), Naive Bayes, and Logistic Regression. The Random Forest classifier yielded the best results, according to the study. Logistic regression is also used in [2] to match offers with product specifications. Consequently, there was at least 85 percent precision for 37 categories of electrical products.

SVM and decision trees are employed in [3] to detect duplicates. The study only looked at text similarity, and it revealed that whether or not the SVM technique performed better than the unlearned baseline it was compared to relied on the data. The SVM model worked better when the test data and training data were comparable. Additionally, the outcome demonstrated that SVM outperformed decision trees. In [4], SVM is also employed to distinguish between representations and offers pertaining to the same product. The outcome demonstrated that, based on the product category, the F-measure ranged from 45 percent to 55 percent.

A novel semi-supervised clustering method for product matching was introduced by Martinek [5]. It combines a sizable unlabeled data set with a minimally labelled data set. This approach shows promise in increasing clustering accuracy while lessening the resource-intensive labelling procedure necessary for fully supervised learning models. Supervised approaches need the use of big labelled datasets, which makes them scalable even if they are frequently more accurate. The author's suggestion of semi-supervised learning is a potential remedy, but further research is needed to determine its true efficacy and generalizability. Comprehensive data sets that accurately represent the varied and dynamic nature of e-commerce inventories are also required. While the semi-supervised

clustering method for product matching proposed by Martinek et al. is an innovative approach to working with the huge and often inaccurate data sets typical of e-commerce, but it comes with its fair share of challenges. One of the key disadvantages is its dependence on the quality of the small amount of labeled data it uses, which can have a significant impact on the overall training outcome: if this data is not representative or of high quality, the model may not perform effectively. Additionally, these methods can be computationally intensive, which may limit their applicability in smaller or less resource-rich settings. There is also a risk of overfitting, especially due to the constraints used to control the clustering process. These constraints, while useful, require careful calibration to avoid fitting too closely to specific features of the training data, potentially reducing the model's ability to generalize to new, unseen data. Despite these challenges, this approach remains attractive due to its potential to reduce the need for extensive manual labeling while maintaining machine learning capabilities for product matching.

On the other hand, researchers from Greece proposed a new unsupervised algorithm for product name matching based on morphological analysis, which effectively addresses some of the shortcomings of previous methods that relied heavily on Internet search engines [6]. This algorithm works in two stages: it first processes product names to create combinations and permutations of words, and then uses these combinations to calculate match scores. Although this method greatly improves efficiency and scalability compared to methods that rely on external data sources, it is not without limitations. In particular, this approach may have problems with highly varied and creatively written product names that do not follow predictable patterns, and there may be potential missing matches when product descriptions vary significantly. Moreover, using fixed combinations of words can lead to missing deeper semantic similarities that may link products described in significantly different terms, creating problems in environments with very heterogeneous data. Additionally, the computational cost, although reduced, can still be significant for datasets with extremely large numbers of products, where even minor inefficiencies increase.

Researchers from University of Mannheim focused into the problem of matching

products using advanced deep learning techniques, particularly recurrent neural network (RNN)-based models and BERT (Bidirectional Encoder Representations from Transformers) [7]. The main purpose was to discern subtle nuances in product data. These methods are particularly good at dealing with the high variability and complexity inherent in textual product descriptions, which often include varied terminology and inconsistent formatting. The strength of RNN and BERT lies in their ability to generate contextual representations of text by enhancing the ability of model to understand and match semantically similar but lexically different product lists. Compared to traditional machine learning techniques the deep learning approaches demonstrate superior performance in learning directly from raw data. In the same time traditional machine learning models often rely on hand-crafted features and simpler statistical models. Traditional methods like decision trees or linear classifiers typically require extensive feature engineering to handle the text variability in product matching. In contrast, deep learning models automate feature extraction. Learning intricate patterns in large datasets that would be unfeasible to model manually. But this comes at the cost of increased computational demand and opacity. Deep learning models like BERT require substantial computational resources, which often require powerful hardware like GPUs for training. Moreover, black box nature makes them less interpretable than their traditional counterparts, which can be a significant drawback in applications where understanding model decisions is crucial. Thus, while deep learning offers substantial advancements in accuracy and learning capacity, it requires considerable resource investment and presents challenges in transparency and interpretability.

Existing strategies have been frequently employed in tackling the problems surrounding the domain. This is in contrast to traditional machine learning models which are less successful when it comes to product matching. These models often rely on extensive feature engineering to handle textual changes in product descriptions. By integrating these traditional models with deep learning approaches, it is possible to combine the strengths of both: the interpretability and simplicity of classical machine learning models as well as the deep feature extraction capabilities of neural networks. Martinek et al. demonstrated how semi-supervised

learning reduces time spent labeling while retaining accuracy with only a little amount of supervision. In addition, there have been other studies that show that morphological analysis can be used successfully without having any need for additional data sets and help enhance scalability or efficiency in various domains.Supervised methods also showed their limits. However, these methods may struggle with highly varied product titles and could miss deeper semantic links between products.

In light of these findings and the existing gaps in the literature, the power of convolutional neural networks for product matching is significant. These models are known for their robustness in extracting hierarchical features from images and can complement text-based methods by providing a multi-modal approach that potentially improves the accuracy and robustness of product matching systems. By integrating modern CNN and RNN architectures, research can address some of the limitations noted in text-only methods, such as problems with semantic understanding and processing noisy data. Moreover, an ensemble method combining these models can further enhance the system's ability to generalize across different data sets. Overall, the study provides a comprehensive solution and contribution to the product matching problem, which enhances the capabilities of deep learning models in e-commerce with a focus on practicality and scalability, and promotes the integration of image and text data into product similarity problem.

1.3 Related works

In the domain of eCommerce product data analysis and comparison, several approaches have been explored to extract, represent, and match product attributes across retailers' platforms. This section reviews pertinent works addressing named entity extraction, feature representation, and similarity assessment in the context of eCommerce product data.

WalmartLabs has developed a system that extracts attributes from product titles in large eCommerce catalogs like Walmart's. Short texts often do not have any syntax, which makes it difficult when trying to understand them. By using

conditional random fields and structured perceptron models and also combining them with a hand-picked normalization technique authors have proposed an interesting method of solving this problem. The main goal of extracting brand attributes is to illustrate this method. This research points out the importance of systems that extract attributes for better organization and comparison of products on retail websites, thereby comparing the suggested approach against other methodologies [8].

Rosie Hood's paper adds to this discourse by talking about how important it is to have features when analyzing product data, including human language sounds or images among other things. They understand that transforming product data in text form into what computers can work on requires application of techniques associated with Natural Language Processing. This paper thus emphasizes on application of word embedding models towards fulfilling this objective thereby making it easier for us to have them electronically without necessarily storing them in terms that can be understood manually. Furthermore, it discusses computer vision focusing on how convolutional neural networks extract feature vectors from product images 1.2, thus facilitating reliable product comparison and classification using visual qualities [9].

The authors in the next work follow their suggestion that in the subsequent paper they come up with an all-round technique for matching products that are founded on a number of reliable attributes of the products comprising title, description, images as well as price. Some of the components in this system include title similarity measure, image similarity evaluation, attribute extraction and price outlier detection. Main focus there is on finding core characteristics, like brand, state, color, and model number, while measuring differences between values for each attribute contained in different product listings. Adding that this research shows how to identify those prices which are far from the average thus increasing chances that people manage to find what they want instead of another thing online on various online markets [10].

These sources collectively contribute to the advancement of methodologies for analyzing and comparing product data in eCommerce settings. From the extraction of fine-grained attributes to the representation of textual and visual features

and the assessment of product similarity across various dimensions, these works offer valuable insights and techniques essential for enhancing product data organization, matching, and comparison across diverse retail platforms.

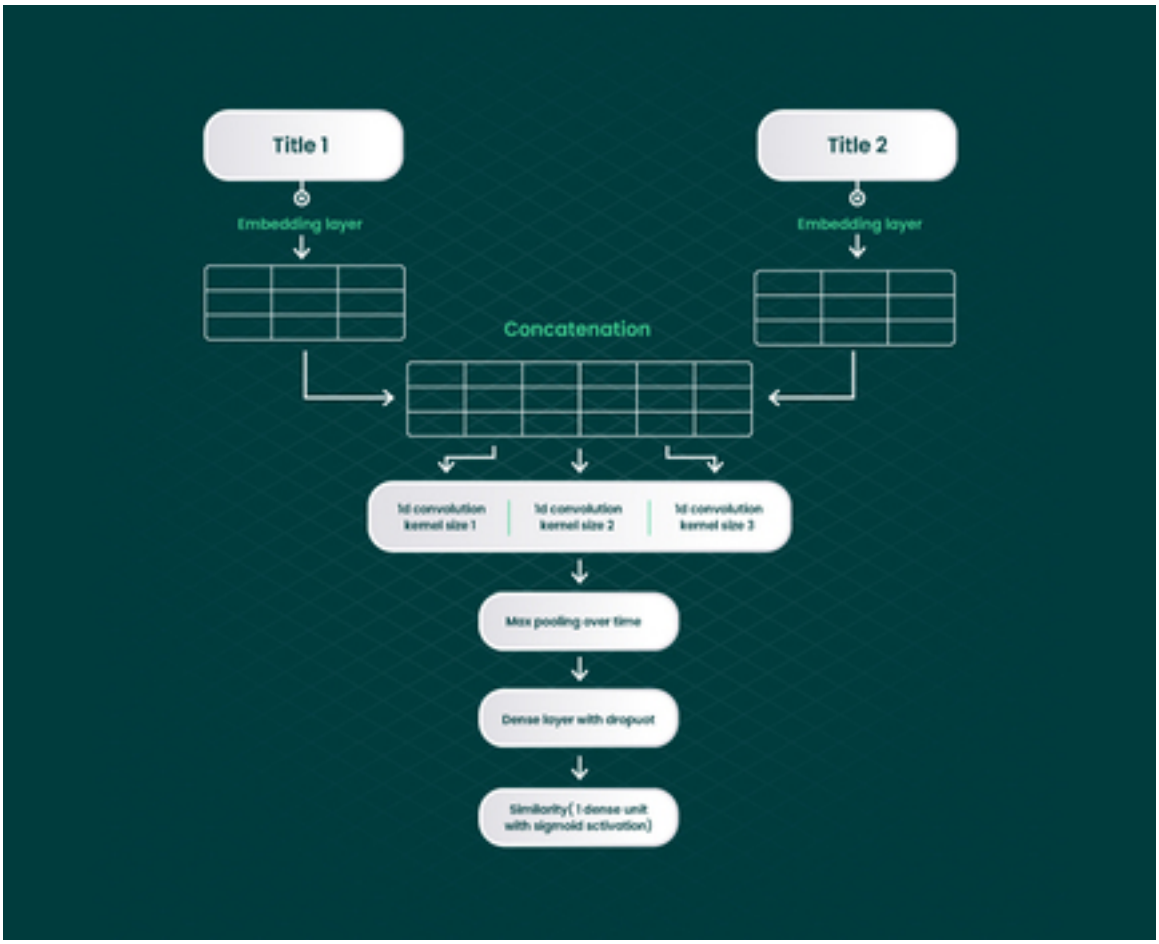

Figure 1.2: Image based comparison algorithm

1.4 Statement of Results

In chapter 4 we will show our results. We will consider which method efficiently in item matching. In figure 1.1 given product description sample, so this is our important data. The present study delved into the domain of product comparison, specifically examining the performance of machine learning approaches. Our findings emphasize the existing gaps in the capabilities of machine learning techniques and underscore the need for further advancements in the form of deep learning models. The purpose of this Thesis is to address the challenge of product matching approaches. Specifically, our focus lies in developing a deep learning

based approach for accurately match same products from different sources. While acknowledging the limitations of existing methods, we consider them as a foundational step for future investigations in this domain.

Chapter 2

Background

2.1 Natural Language Processing

Since its start in the 1950s, NLP research has concentrated on a wide range of tasks, including text summarization, information extraction, machine translation, question-answering, information retrieval, topic modelling, and, most recently, opinion mining. Syntax was the topic of extensive research in the early days of NLP. In addition to being inspired by the prevalent idea of syntax-driven processing, which gained implicit or explicit support from academics in the field, this emphasis on syntax was prompted by the obvious requirement for syntactic processing in language interpretation.

Although the relevance of semantics in NLP was clear from the start, the research community first concentrated on solving syntax because machine learning techniques were more directly applicable in that domain. However, some researchers understood the importance of semantics and thought it was a more difficult challenge, or they thought a semantically-driven strategy would be more successful. Using semantic categories and semantic case frames, for instance, groups led by Masterman and Ceccato investigated semantic pattern matching. Ceccato, in particular, used semantic networks to represent knowledge and world information to improve linguistic semantics.

As stressed by Minsky in 1968, other works recognized the requirement of incor-

porating outside knowledge to interpret and react to linguistic input. In Schank's 1975 paper, the function of semantics was clearly emphasised, with a focus on general-purpose semantics that encompassed case structures for representation and semantically driven processing. These changes signaled an increasing understanding of the value of semantics and the application of semantic knowledge in furthering NLP research.

First-order logic (FOL), a logical system composed of axioms and inference rules, has subsequently become one of the most widely used representation systems in Natural language processing. Predicates with complex relational structures and supporting syntax, semantics, and, to a lesser extent, pragmatic expressions, are successfully formalized using FOL. The correct placement of symbol groups is determined by syntax, assuring their well-formedness. Well-formed expressions' meanings are defined by semantics. In contrast, pragmatics uses contextual information to strengthen linkages between diverse semantic components, which is critical for tasks such as word meaning disambiguation.

However, monotonicity is a problem for logic-based methods. The number of entailed sentences increases as more knowledge is given to the knowledge base, which contradicts a basic feature of human reasoning—the ability to freely and flexible change of opinion. Solutions like linear logic and default logic have been suggested as ways to solve this problem. Raymond Reiter developed default logic, which seeks to formalize presumptions such "all birds fly". However, problems occur when statements that are generally true but deviate from the general norms, such as "penguins do not fly," are formalised using default logic.

The production rule, which Chomsky established in 1956, is another frequently used model for defining natural language. There is a working memory that stores ongoing memory claims in a production rule system. These production rules are stored in a volatile working memory. Each production rule is written as "IF (conditions) THEN (actions)", with each rule consisting of a set of conditions and a subsequent set of actions.

A production rule system's core operation is a three-step cycle: "recognise", "resolve conflict" and "act". This cycle will continue until there are no more

rules relevant to working memory. During the "recognise" stage, the system identifies the rules whose antecedent conditions are met by the working memory that is now in use. The conflict set is the name given to these rules. The system chooses an appropriate subset of rules from the conflict set to be applied during the "resolve conflict" stage. The system conducts the chosen actions, which may include changing the working memory, in the final "act" stage.

The modularity of production rule systems is one of their benefits. The system can be managed and maintained more easily by grouping the production rules into modules.

Each rule in a production rule system runs independently of the others, making it simple to add and remove rules. The modular architecture of production rule systems lends them flexibility and adaptability. Production rule systems are characterised by a basic control structure and easily comprehensible rules for people. This is because the terminology used to encode the rules is meant to be understandable to people, and the rules are often derived from expert behaviour or knowledge.

However, as the size of the production rule system expands, scalability may become a problem. A system with thousands of rules requires a lot of maintenance work to manage. It gets increasingly difficult to ensure the appropriate operation and coordination of rules as the number of rules rises. As a result, maintaining a complex production rule system can be time- and resource-consuming.

The Ontology Web Language (OWL), which McGuinness and Van Harmelen introduced in 2004 [11], is another noteworthy NLP approach. The Resource Description Framework (RDF) is an XML-based vocabulary that OWL enhances to offer a more complete set of capabilities for ontology representation. Classes, class properties, relationships between classes, limitations on those relationships and properties may all be defined with its help.

RDF supports the subject-predicate-object model when stating claims about resources. Semantic consistency and ontology categorization are guaranteed by RDF-based reasoning engines. Particularly OWL focuses on precisely describing static structures, making it appropriate for representing declarative information.

OWL has restrictions, though.

First off, OWL demands rigorous definitions, which makes it unsuitable for describing information with varying degrees of confidence. It works better to convey factual or objective knowledge. It is also challenging for OWL to express knowledge that is temporally dependent, making it challenging to capture data that is dynamic or has temporal dependencies.

These restrictions show that although OWL is an effective tool for capturing some forms of information, it may not always be the best option, especially when dealing with subjective or temporal aspects.

Network-based methods are in fact frequently employed in NLP, with Bayesian networks (sometimes referred to as belief networks) serving as a notable illustration. A framework for expressing joint probability distributions over connected hypotheses is provided by Bayesian networks, which were first described by Pearl in 1985. To depict the causal links between variables in a Bayesian network, arcs in a directed acyclic graph (DAG) are utilised.

Subjective levels of confidence can be represented using Bayesian networks. To determine the likelihood of events, they expressly take prior knowledge into account and pool the available evidence. To compute a belief network's joint distribution, one must comprehend the conditional probability Pr(P|parents(P)) for each variable P. However, figuring out these probabilities for each variable can be difficult, particularly for activities involving a lot of data processing. The statistical tables for such networks might be difficult to maintain and update.

Additionally, the expressiveness of Bayesian networks is constrained, comparable to that of propositional logic. Semantic networks are hence frequently chosen in NLP research. Semantic networks give you more options for expressing and capturing intricate connections between concepts and entities. They offer a graphic representation that makes knowledge and meaning easier to understand and more richly represented.

Developed by Sowa in 1987 [12], a semantic network is a graphical representation that shows knowledge as interconnected nodes and arcs. There are many

different types of semantic networks, including assertional and definitional networks.

The "IsA" interactions between concepts and their subtypes are the main focus of definitional networks. These connections create generalizations that enable all of the subtypes of a supertype to inherit the properties specified for the supertype. The information is often taken as true in definitional networks.

On the other hand, assertional networks are used to make claims, and the data they contain is thought to be partially true. In assertional networks, contingent truth is more often determined by common sense than by default logic. In assertional networks, the propositions are supported by sufficient arguments, where the arguments support the proposition.

Semantic networks give knowledge an organized and illustrative representation, making it easier to identify relationships and to reason about the data they include.

Semantic networks originated with research by Simmons and Quillian [13] in the early 1960s. Marvin Minsky developed this idea further in his Society of Mind hypothesis in the late 1980s. According to Minsky, varied agents working together to do various cognitive tasks including remembering, comparing, generalizing, analogizing, and anticipating lead to the development of human intelligence. The artificial intelligence community was enthused by this idea of human cognition, which inspired the development of practical knowledge bases for NLP tasks.

A number of well-known initiatives, such as Cyc, WordNet, Thought-Treasure, and the Open Mind Common Sense project, were inspired by Minsky's notion. Doug Lenat created Cyc, a logic-based database of common sense information. WordNet is a global database of word senses that was developed by Christiane Fellbaum [14]. A system for interpreting stories was created by Erik Mueller and is called Thought-Treasure [15]. Rather than being manually generated by professional technologists, the Open Mind Common Sense project is a second-generation common-sense database where information is given by online volunteers and presented in natural language.

As part of the Open Mind Common Sense project, a significant amount of common sense information was acquired and is currently being used to a range of natural language processing tasks, such as opinion mining, story telling, textual emotion sensing, and casual conversation interpretation. By incorporating and utilizing common sense information in natural language processing tasks, these efforts have advanced NLP research.

Their technological specialties. It is true that the pace of progress in NLP has not always kept up with the quick changes in the digital landscape, even if there have been substantial achievements in NLP research, such as the creation of numerous models and techniques as previously described.

Natural language understanding's intricacy is one cause of this discrepancy. Due to the complexity and ambiguity of language, it is difficult for robots to effectively understand and produce writing that is human-like. In addition, when languages change over time, new slang, idioms, and cultural references are added, making NLP jobs much more challenging.

The enormous quantity of data and computing power needed for developing and deploying complex NLP models is another aspect. It gets more and harder to comprehend and interpret all the information accessible as the number of digital stuff expands exponentially.

It is crucial to remember that NLP research has advanced significantly in recent years. Machine translation, sentiment analysis, question answering, and text synthesis have all seen significant improvements thanks to the development of deep learning techniques, which were made possible by technology improvements and the availability of enormous datasets.

Additionally, combining NLP with other cutting-edge technologies like reinforcement learning, graph neural networks, and knowledge graphs has the potential to overcome some issues and advance the field of NLP research.

It is anticipated that NLP research will continue to grow at an accelerated rate as we observe the ongoing technological evolution and the rising demand for natural language processing capabilities. The future of human-computer interaction

and intelligent systems depends on researchers and practitioners actively investigating novel ideas and methodologies to enhance language understanding and generation.

It is true that a significant trend in the area is the move away from word-based approaches toward a more consistent exploitation of semantics in NLP systems. Although words are the basic building blocks of language, they frequently lack the depth and context necessary for proper meaning representation and deeper understanding.

NLP systems can capture a deeper grasp of language by focusing on multi-word expressions and the accompanying semantics and sentac. Idioms, collocations, and named entities are examples of multi-word formulations that have distinct meanings and nuances that are difficult for single words to adequately convey. NLP systems can comprehend language more complexly by taking into account the compositional character of language, in which words are combined to create new meanings.

Sentic and semantic integration is essential for decision-making and common sense in NLP. information the world, its people, events, and relationships, as well as being able to use logic and judgment to base judgments on this information, are all components of common sense knowledge. Sentics, which includes emotive information, gives language an additional layer of meaning by allowing systems to understand subtle emotional cues, individual viewpoints, and sentiment analysis.

Deep learning models, knowledge graphs, and other improvements in semantic representations have given NLP systems new ways to capture and use semantic data more efficiently. These methods make it possible to describe links, ideas, and hierarchies, which improves comprehension and inference abilities.

NLP research strives to advance the precision, contextual comprehension, and reasoning capacities of NLP systems along the "Semantics Curve," ultimately enabling more natural and intelligent interactions between humans and machines. Figure 2.1 illustrates how NLP research is said to have developed over three distinct periods or curves.

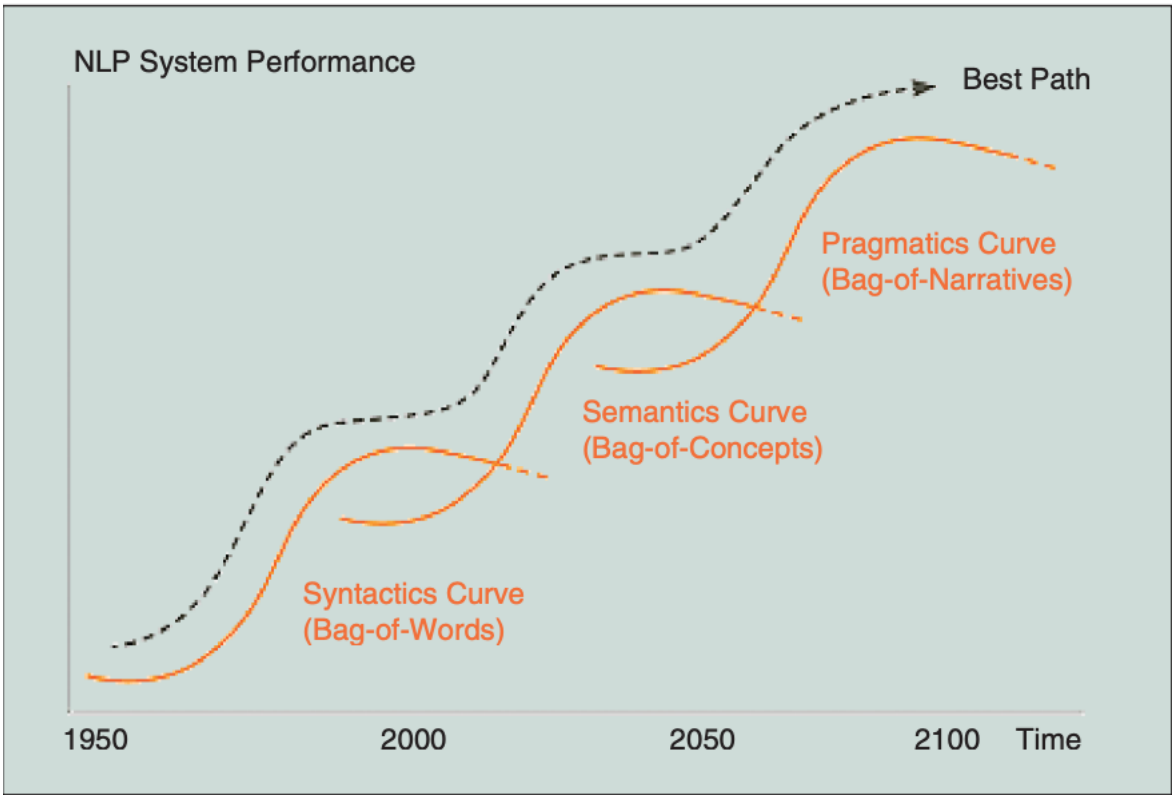

Figure 2.1: Envisioned evolution of NLP research through three different eras or curves.

2.2 Computer Vision

There are two goals for computer vision. From a biological research standpoint, computer vision aims to create computational models of the human visual system. From an engineering standpoint, computer vision aims to develop independent systems that can do certain tasks that the human visual system can (and frequently surpasses) in many scenarios. In vision research, one of the most common tasks is to extract 3D and temporal information from time-varying 2D data, such as that obtained by one or more television cameras. Understanding such dynamic circumstances is involved in a broader sense.

It is clear that the two goals are interconnected. The characteristics and features of the human visual system serve as a common source of inspiration for engineers developing computer vision systems. Conversely, computer vision algorithms offer insights into the workings of the human visual system.

The field of computer vision is usually recognised as having its start in Larry

Roberts. In his 1960 PhD thesis from MIT, he investigated the possibility of extracting 3D geometrical information from 2D perspective images of blocks (polyhedra) [16]. After this study, a number of AI researchers examined computer vision within the blocks world paradigm at MIT and other locales.

In due course, scientists concluded that real-world images required attention. As a result, a significant amount of research was needed for "low-level" vision tasks like edge recognition and segmentation. A major turning point was the paradigm created by David Marr (cir. 1978) at MIT, who used a bottom-up approach to scene understanding [17].

Binocular stereo is utilised to construct a 2.5 D drawing of the scene after 2D photographs are processed using low-level algorithms to create the "primal sketch" (directed edge segments, etc.). In closing, as seen in figure 2.2 The objects in the scene are represented in 3D models by high-level (structural analysis, a priori knowledge) methods. This is perhaps the most significant study on computer vision that has ever been conducted. "No one can drive us out of the paradigm created for us by Marr," a number of scholars declared.

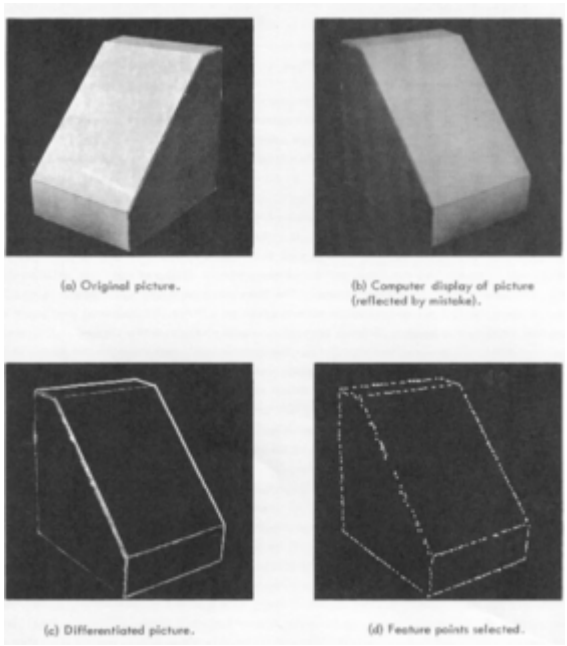

Figure 2.2: 3D structures of objects with all the hidden lines removed.

Nonetheless, a number of computer vision experts have lately called for a more top-down and varied approach after realising some of the drawbacks of Marr's

paradigm. To put it briefly, Marr's programme is extremely difficult to execute, but more significantly, most computer vision applications do not need the acquisition of complete 3D object models. In computer vision-based autonomous vehicle navigation, for example, it may be necessary to ascertain just if an object is approaching or receding from your automobile, rather than its exact three-dimensional motion. The "Purposive Vision," as this new paradigm is sometimes called, proposes that algorithms should be goal-driven and in many cases may even be qualitative [18]. Yiannis Aloimonos of the University of Maryland is among the principal proponents of this new paradigm.

The subject of computer vision research is well known for its difficulties. There are very few scientific issues that have been satisfactorily resolved. Computer vision technologies pale in contrast to the human visual system, which is just too efficient for many tasks (such as face identification). This is among the main reasons why this problem exists. People can recognise faces in a variety of settings with different lighting, perspectives, emotions, etc. For the most part, we have little trouble recognising a friend from a photo taken years ago. Moreover, it appears that we have an infinite capacity to learn faces for recognition at a later time. Building an autonomous system that works as well as this one appears to be unattainable.

The use of image similarity technology is now more advantageous than ever. The majority of common software can now effectively perform picture similarity algorithms due to the growing use of high-tech computers. There are numerous industrial uses for this technology, including supply chain management, manufacturing, retail, healthcare, and security. Businesses seeking to incorporate this technology into their operations for high economic value now have more options thanks to its recent strong growth and demand.

Open-world learning and active vision, object component connection, multi-modal detection, pixel-level detection, etc. are some of the challenges that must be overcome for effective integration of future image similarity algorithms to yield important results and major breakthroughs in recent years.

2.3 Product matching

In the retail industry, it's not uncommon for product references to offer scant information. This is most commonly seen in different print and digital media (flyers, banner ads, etc.) where the products that are highlighted are often not accompanied by a unique name or model number, universal product code (UPC) or global trade item number (GTIN). In the 1970s, UPC was first created as a North American standard for supermarkets [19]. GTIN was developed subsequently and was intended to serve as a global standard for supply chain logistics and product identification. A numbering superset of UPC is called GTIN [20].

For many stakeholders, being able to recognize products from lists or adverts holds enormous promise. From the perspective of the customer, products in niche markets can be more easily located due to the simpler identification of products from listings. It also enables systems to assemble products from several suppliers and provide reliable information, saving clients time and money. Through the provision of more comprehensive product listings, pertinent advertising, improved product suggestions, and other services, businesses may use this data to improve the consumer experience. When products from various sources can be easily recognized, even when they are vague, it is to the immediate advantage of consumers, advertisers, and product aggregators. A fuller examination of these advantages can be found in [21].

There are several types of product identification challenges. Certain product attributes may not have values, or the information may be insufficient. If certain values are not consistently recorded in the appropriate fields within a structured product record, such as when the product's brand is stated in its name but not in the brand field, the product data may become contaminated. Sometimes product listings are just a list of features; for example, "Apple iPhone 15 Pro Max 5G 8GB RAM 256GB"; there may be no division or obvious labelling of the product's name, brand, price, or other details.

In addition to the product data itself, retailers frequently classify their products into disparate, incompatible taxonomies, which presents further issues. A Chromebook—is it a laptop? Should apparel that is intended for both men and

women be categorized together or kept apart? Diverse merchants often make judgements that are characterised by imprecise or inconsistent classifications and distinctions, making it difficult to extract information from the meta-structure containing the product data. The difficulties that contemporary systems encounter are covered in more detail in [21]. The domain's primary areas of focus are the comparison and conversion of unstructured to structured records, the determination of the attributes that most strongly suggest a true match (as well as the common errors among these attributes), the maintenance of scalable methodologies as dataset sizes increase, and the handling of all possible data errors, incompleteness, and other less-than-perfect situations [22].

Since 2018, a large number of new publications have been published in the topic of product matching. While previously published studies were spaced further apart to allow for reference and building upon past work, new publications on essentially identical approaches have begun to surface only months apart [23]. The increasing amount of overlapping work in the field has made direct comparisons between more recently published methodologies and approaches less common.

There are several factors contributing to the increased interest. Human access to data has increased throughout time, more or less steadily, since the development of computers. This is true for both publicly available academic data and data that businesses collect and record. The abundance of useful and instructive data combined with the creation of big data processing techniques and tools has greatly increased the accessibility of product matching for both businesses and academics.

Consumers' inclination to purchase online is increasing in tandem with the expansion of data and data accessibility [24]. While traditional brick and mortar stores like Walmart are growing their online presence, Amazon continues to grow every year. As consumer spending shifts more and more into the virtual economy, businesses are investing more money in improving their online platforms. Product matching's ability to improve everything from product sites to product suggestions is making it more and more valuable [20].

Proposed solutions, like many contemporary issues, are derived from a wide range of scientific fields. Product matching is no different, utilizing several ad-

ditional well-established domains: Record Linkage, Natural Language Processing (NLP), Computer Vision (CV), Scraping, Classification.

Blocking is crucial for computationally effective classification approaches. Over the past 50 years, numerous classification techniques have been suggested and tested. The typical pair-wise classification approach does not handle transitive closure concerns caused by many conflicting matches during linkage. This challenge is solved by more recent approaches, such as group classification or rephrasing the problem to make advantage of clustering methods [19].

Text parsing and comprehension are the main goals of natural language processing, or NLP. Numerous subproblems that come up during product matching can be solved with NLP approaches. As demonstrated in, converting unstructured data into structured data is a typical use case. In a similar vein, resolves conflicting category or attribute names across product listings by using a linkage task. Textual attributes have been represented and features for classifiers have been created using NLP techniques such as Term Frequency - Inverse Document Frequency (TF-IDF) [24].

Computer Vision (CV) is a transformative technology in the realm of product matching, automating the process of identifying and comparing items through visual analysis. Utilizing algorithms, CV examines images and videos to execute detection, classification, and matching tasks. Its application is particularly prevalent in retail and e-commerce for inventory management, ensuring the accuracy and contemporaneity of product listings. Moreover, CV enhances customer experiences by suggesting visually similar products, thus improving satisfaction and engagement. The continuous advancement in CV technologies promises increased precision and efficiency, cementing its role as a critical tool in modern retail operations.

The technique of gathering data from websites is known as scraping. Search engine and e-commerce site data can be used to create and link databases for additional study. A cutting-edge web scraping framework called DEXTER is designed to extract product records. For example, earlier studies frequently employed their own datasets for linkage theory analysis. For their functions to be effective, prod-

uct aggregation services such as Google Products and Bing Product Catalogue depend on comparable web searches [22].

Traditional machine learning approaches are widely used in academics, as well as industry [22]. Deep learning now outperforms classic machine learning models with sufficient data. Recent studies demonstrate that deep learning algorithms provide cutting-edge performance. Mudgal et al. [25] provide an excellent overview of the trade-offs involved in designing deep learning models.

Publications in this topic are contributed to by a variety of parties with different goals. Businesses have used techniques created in this field to implement large-scale, practical goods [24]. Several firms have published papers that progress the subject. Companies may collaborate with academic institutions to provide access to data for practitioners. Academia advances cutting-edge techniques while simultaneously making them accessible. Magellan [26] is a prime example of a user-friendly and adaptable architecture for linking records.

It is reasonable to expect data to anticipate the future in a field driven by data. The age of deep learning is, in many respects, just getting started. Although it has proven useful in many fields, there is still room for innovation and creativity in its application. Future product linking systems will certainly integrate previously unusable data. Deep learning has revolutionized the use of picture data, but this is only the beginning. Future systems will presumably use consumer activity data, including interests, purchases, and demographics. As technology advances, corporations will increasingly adopt these approaches. Technical factors, such as using scalable frameworks and accurately evaluating performance in a changing product landscape, will be given priority.

There is a need to compare existing systems' performance across various data sets. Recent attention in this field has resulted in diverse approaches that have yet to be evaluated or merged. Identifying the advantages and disadvantages of different approaches remains a key topic. The usefulness of using individual product linking techniques on industrial data is investigated in this thesis.

Chapter 3

Methodology

3.1 Data Preprocessing

Preprocessing is done on the available raw data once it has been collected from a reliable source. It is necessary to ensure that the deep learning model produces the best results.

In this chapter, we present our approach for a product matching both using natural language processing and computer vision. The data used to evaluate our method was taken from open source task with items from wildberries and yandex market [27]. Initially, two json files were given. Each size was 4.5 gb for yandex market items and 2.6 gb for wildberries items respectively. Json file with yandex market items contains 1472566 rows and 19 columns, while json with wildberries items contains 1079155 rows and 22 columns. Also author provided csv file with matched Ids. This table was used to match the same item from both jsons. After joining all 3 data, 14702 matched pairs found. In wildberries table image urls were invalid, so image link parser was written. The code is located in Appendix B. After that kept only image urls and titles from both marketplaces 3.1.

The following steps provide a detailed description of our method. The accompanying flowchart provides a concise overview of our proposed approach. On figure 3.2 clearly shown preprocessing steps of text data.

Certainly, preprocessing of data is an essential step before training a model.

```
df.iloc[:, :4].sample(10)
```

	wildb_title	wildb_img	ydx_title	ydx_img
771	Ключ комбинированный 6 мм HELFER	https://basket-01.wbbasket.ru/vol109/part10946/10946865/images/big/1.webp	Ключ комбинированный 6 мм HELFER	https://avatars.mds.yandex.net/get-mpic/5253116/img_id2728756479759938495.jpeg/9
10584	Монтировка усиленный 400 мм (Cr-V)	https://basket-01.wbbasket.ru/vol87/part8735/8735728/images/big/1.webp	Монтировка усиленная 400 мм (Cr-V)	https://avatars.mds.yandex.net/get-mpic/4012413/img_id2346073148156477077.jpeg/9
8819	Скатерть-самобранка. Русские сказки	https://basket-01.wbbasket.ru/vol65/part6582/6582406/images/big/1.webp	Скатерть-самобранка	https://avatars.mds.yandex.net/get-mpic/5177644/img_id7587600403988777639.jpeg/9
11740	Пазлы 4 в 1. ВОЕННЫЙ ТРАНСПОРТ.	https://basket-01.wbbasket.ru/vol97/part9790/9790608/images/big/1.webp	"Пазлы ""Военный транспорт"" 4 в 1"	https://avatars.mds.yandex.net/get-mpic/5220903/img_id6652277171312232175.jpeg/9
353	Крем для лица укрепляющий с коллагеном BLACK INTENSIVE ANTI-WRINKLE COLLAGEN CREAM, 70гр.	https://basket-01.wbbasket.ru/vol104/part10467/10467626/images/big/1.webp	HANIxHANI Black Intensive Anti-Wrinkle Collagen Cream Крем для лица укрепляющий с коллагеном, 70 г	https://avatars.mds.yandex.net/get-mpic/4544069/img_id2437844145046009818.jpeg/9
5337	Ракетка для настольного тенниса Softspin Plus, FL (CONC) / теннисная ракетка / для игры в теннис	https://basket-02.wbbasket.ru/vol181/part18178/18178030/images/big/1.webp	Ракетка для настольного тенниса Butterfly Softspin Plus	https://avatars.mds.yandex.net/get-mpic/4034173/img_id3929662791487360664.jpeg/9
3414	Сумка АВТОМОБИЛИСТА SKYWAY 47*24*16см Рис. Девушка	https://basket-01.wbbasket.ru/vol135/part13573/13573798/images/big/1.webp	Сумка автомобилиста SKYWAY 47*24*16см Рис. Авто	https://avatars.mds.yandex.net/get-mpic/5272194/img_id7676098629886808533.jpeg/9
6258	Туалетная вода LANCOME Hypnose Homme мужская 75 мл	https://basket-02.wbbasket.ru/vol212/part21230/21230521/images/big/1.webp	Туалетная вода Lancome Hypnose Homme, 75 мл	https://avatars.mds.yandex.net/get-mpic/2008455/img_id2418427105188614156.jpeg/9
5506	Люстра подвесная Divinare CONO 2015/17 SP-3 3x40Вт E14	https://basket-02.wbbasket.ru/vol188/part18866/18866939/images/big/1.webp	Светильник Divinare Cono 2015/17 SP-3, E14, 120 Вт, кол-во ламп: 3 шт., цвет арматуры: латунный, цвет плафона: латунь	https://avatars.mds.yandex.net/get-mpic/4415357/img_id4619703173381304841.jpeg/9
2589	Сетевое зарядное устройство Ginzzu GA-3010UB	https://basket-01.wbbasket.ru/vol126/part12671/12671756/images/big/1.webp	Сетевая зарядка Ginzzu GA-3010UB, черный	https://avatars.mds.yandex.net/get-mpic/5146791/img_id8327681915342332455.jpeg/9

Figure 3.1: Final dataset.

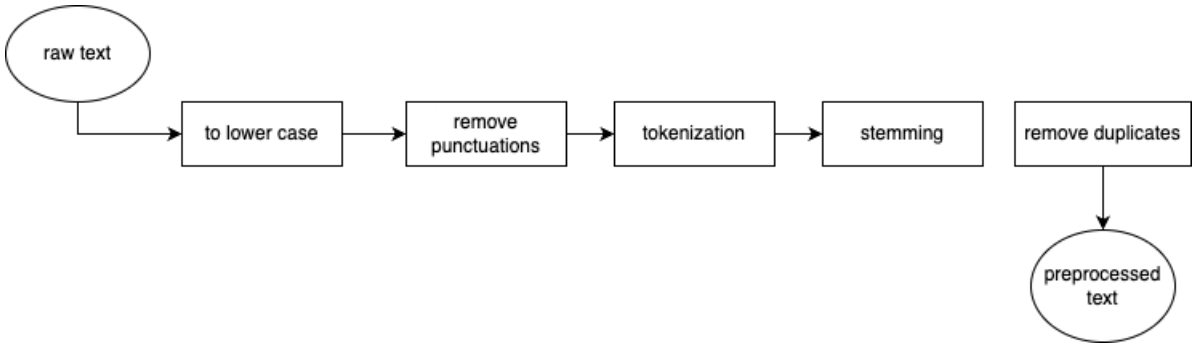

Figure 3.2: Preprocessing Steps.

Here are the different stages of text preprocessing typically performed to get ready the data for training:

1. To lowercase: all letters to same register.

2. Cleaning and normalization: This stage involves removing unwanted characters, such as punctuation marks or special symbols, and converting the text to a consistent format.

3. Tokenization: The text is divided into individual tokens, such as words or subwords. This process helps break down the text into meaningful units.

4. Stemming: Stemming involves reducing words to their base form by removing prefixes or suffixes. It helps consolidate variations of words and improve generalization.

5. Text normalization: This step involves applying additional techniques such

as removing excessive whitespace, handling repeated characters, or converting accented characters to their base form.

In terms of images, the first step in this strategy is to resize the images. Using the Tensorflow library, each image has been shrunk to 224 by 224 so that the deep learning model can train more quickly on smaller images. Depending on the quantity and size of pixels, the neural network model must learn a lot while working with a larger input image. Therefore, scaling images is thought to be one of the most effective data pre-processing methods for lowering the training overhead. The photos are enhanced and pre-processed to lessen model bias and increase their generalizability before the image embedding is extracted. Following this stage, an array of images is created and fitted to the models in order to extract the image embedding.

These preprocessing stages help clean and transform the text data into a format that is more suitable for feature extraction and training machine learning models.

3.2 Matching methods analysis

3.2.1 VGG-19

To retrieve the image's style and content representation, centre layer of the model is used. Low-level features like edges and textures are represented by the excitation response of the first few layers of the network, which starts at the input layer. The final few levels, which represent more complex elements like wheels or eyes, reflect the number of layers that decreases. We are utilising a pre-trained image classification network, the VGG-19 network structure. To define content and stylistic representations from photographs, these intermediary layers are required. Here is an illustration of the model's structure, which consists of a custom model 's convolutional neural network composed by a series of convolutional layers and maximum pooling layers and is more direct than the initial half of VGG-19 model 3.3 .

There are two maximum pooling layers and five convolutional layers in this model. Each convolutional layer is followed by a ReLU activation function. Cus-

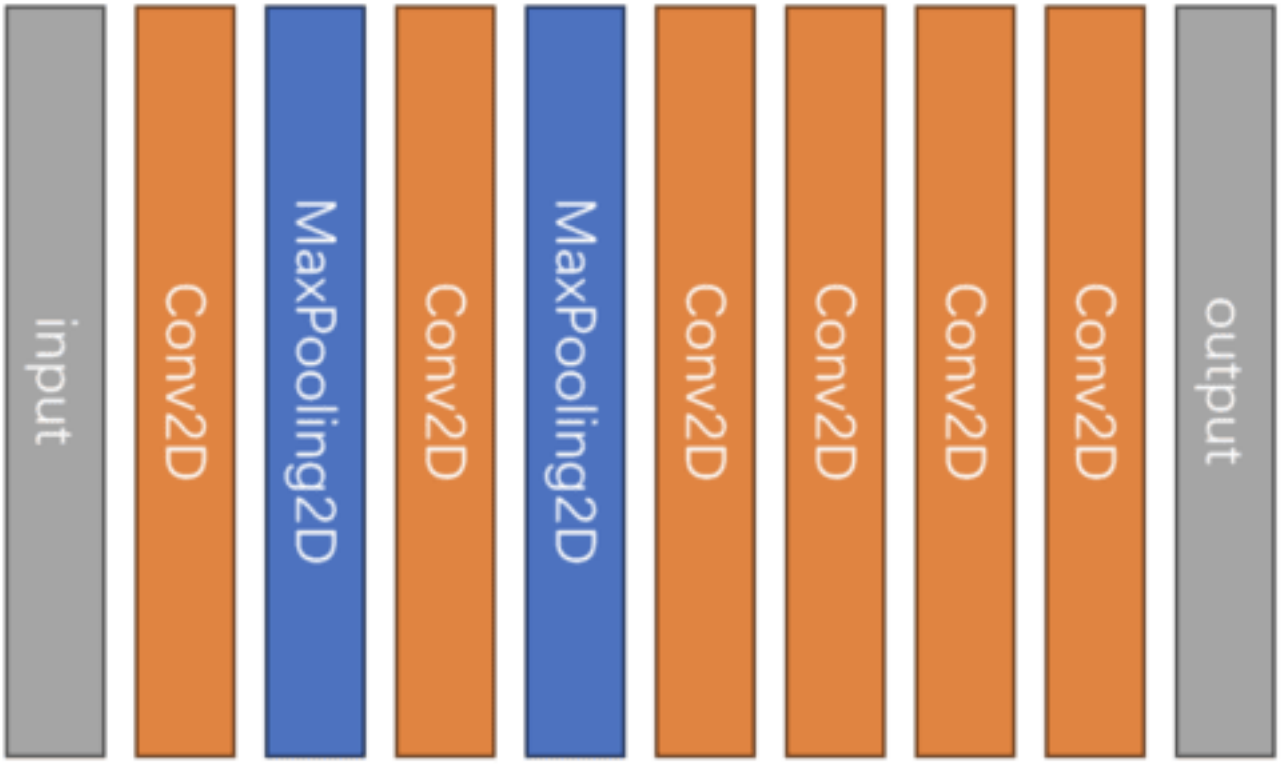

Figure 3.3: VGG-19 model structure.

tom models are designed to extract feature representations from input photos. A new model that we have defined produces feature representations at various levels. To get these feature representations, you have to extract the model's convolutional layer outputs. Feature representations of the image are considered as model outputs at various levels. For these feature representations, high-dimensional tensors are used with each channel representing a different feature. Although feature representations consist of more semantic characteristics at higher levels, they contain more low-level features at lower levels. We can obtain features on different levels extracting them with different numbers of convolution and maximum pooling layers.

Three of the 19 layers are fully connected, while 16 of the layers are convolutional. It goes by the name convolutional neural network, and uses as its database ImageNet, a model trained on roughly one million photos. ImageNet can classify images into over a thousand categories, for instance, tools, gadgets, vehicles, planes, bikes, dogs, numbers etc. The measurement could accept at most 224 x 224 pixel images and it is fairly good in photo category. Using this model with

pretrained weights has led to the discovery of transfer learning notion. As the name suggests, transfer learning is the process of moving information from one model to another. Because you do not have to start from scratch every time a model has to be trained, it saves a significant amount of time and computing resources. It entails building another model off of an existing one. Many of them offer pre-trained models, which may be used as a starting point to transfer the results to a new model for increased accuracy and reduced processing requirements. For instance, by utilising a trained model that can recognise animals based on their eyes, you may train a new model to recognise animals based on their other parts of the body. The architecture of VGG-19 can be seen in 3.4.

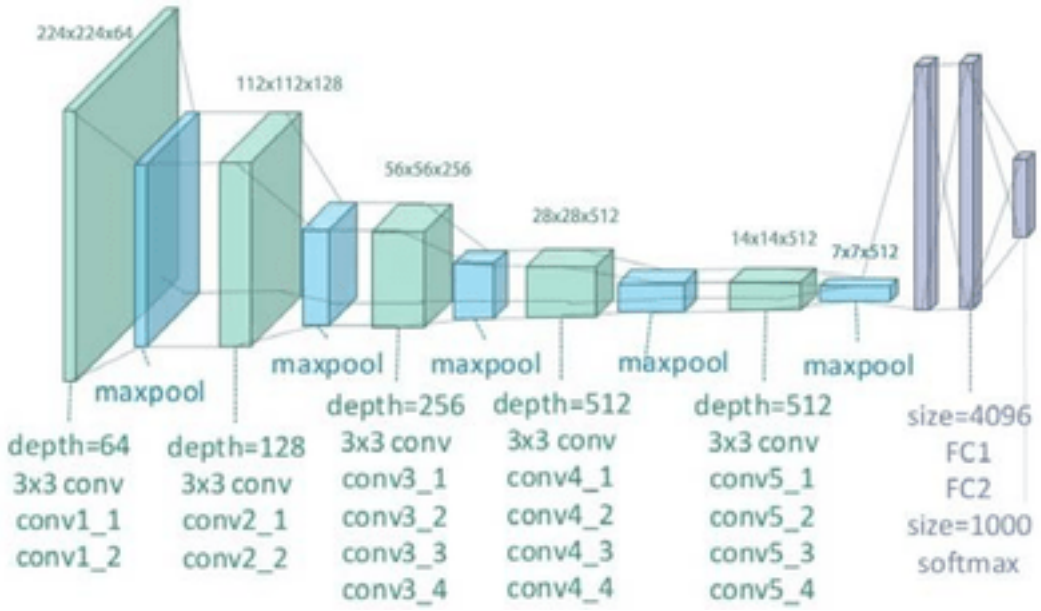

Figure 3.4: VGG-19 architecture.

3.2.2 MobileNet-V2

The primary building component of MobileNet V2 now looks like this 3.5, however it still uses depthwise separable convolutions.

This time, the block contains three convolutional layers. The final two are the ones we are already familiar with: a 1x1 pointwise convolution layer comes after a depthwise convolution that filters the inputs. But today, this 1x1 layer has a new role. The number of channels in V1 was either doubled or kept constant by the pointwise convolution. However, it decreases the quantity of channels in V2. This

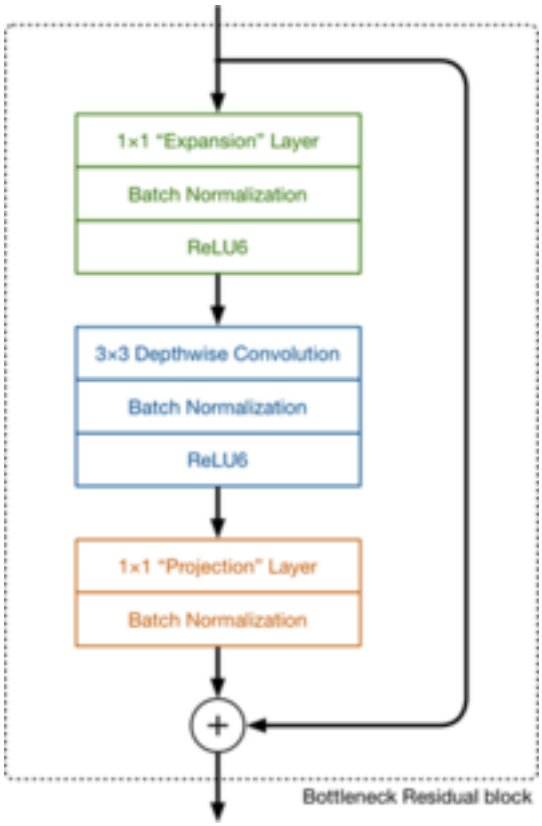

Figure 3.5: Main building blocks of MobileNet-V2.

layer is currently called the projection layer since it projects data with a large number of dimensions (channels) into a tensor with a significantly lower number of dimensions. For example, the depthwise layer operating on a tensor with 144 channels may be reduced to just 24 channels by the projection layer. This kind of layer is also referred to as a bottleneck layer since it reduces the amount of data that travels over the network. The new kid on the block is the initial layer. This convolution is also 1x1. Its objective is to increase the data's channel count prior to depthwise convolution. As a result, this expansion layer essentially performs the reverse of the projection layer, always having more output channels than input channels.

MobileNet in practical applications is an effective and portable CNN architecture. To develop a lighter model, MobileNet essentially uses depth-resolvable convolutions in place of the standard convolutions utilised in prior designs. The MobileNet width multiplier and resolution gain are two new global meta-parameters that allow model writers to trade off speed and compact size for latency or accuracy. To build MobileNet, deep separable convolutional layers are used. Point

convolution and depth convolution make up each depth separable convolution layer. If depth and point convolutions are counted independently, MobileNet has 28 layers. By optimising the width factor meta-parameters, we may minimise the 4.2 million parameters that constitute a typical MobileNet. The architecture of MobileNet-V2 is shown in 3.6.

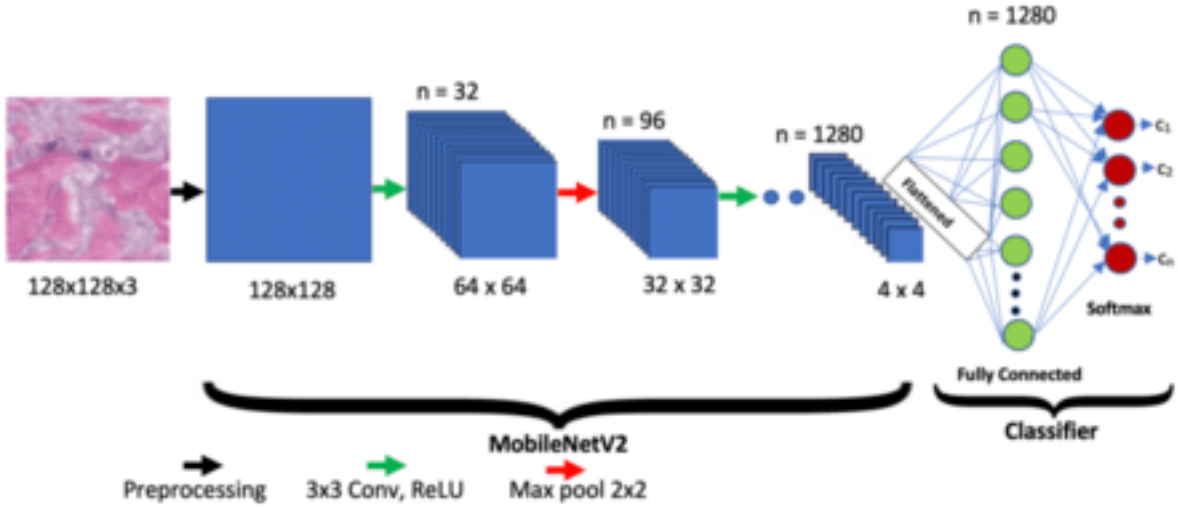

Figure 3.6: MobileNet-V2 architecture.

3.2.3 ResNet-50

Resnet50 is a well known Resnet architecture acknowledged for its 50 layers and happens to be amongst the most popular ones. In 2015, it achieved the greatest performance on the ImageNet dataset. Connected by residual connections are a number of convolutional layers in each of these sixteen residual blocks which make up ResNet50. It's crucial to remember that the architecture includes fully linked layers, pooling layers, and a softmax output layer for classification. An image of size 224x224x3 is fed into the ResNet50's input layer. The image's RGB colour channels are represented by the number 3.

A well-liked neural network that addresses the issue of training deep neural networks is called ResNet, or residual network. Neural networks with more than 140 layers are now very simple to train thanks to the development of ResNet. Because the gradients were inversely connected, they disappeared after every layer before ResNet. This indicates that performance achieves saturation after a given

number of layers. ResNet uses identity connectivity, a method that deals with the problem of vanishing gradients, to achieve this. The architecture of ResNet-50 is shown in 3.7.

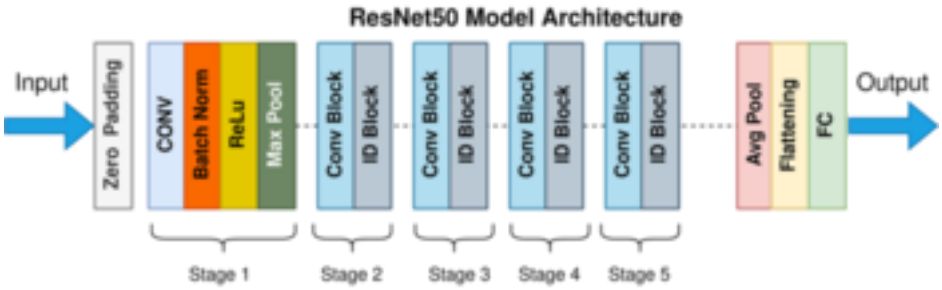

Figure 3.7: ResNet-50 architecture.

Chapter 4

Results

4.1 Quantity of data

The data will be represented in some vector or numeric format. At first we wait for some words in data, but for machine learning analysis, the words will be represented in vector or digit format. Images were resized, shrunked to 224 by 224 and scaled so that the deep learning model can train more quickly.

The data have about 14.000 rows with matched item names and matched image urls from both sources. Although the data was from open sourced document, we had to parse links to product images from wildberries site, because initial urls were out of date and invalid. The code is located in Appendix B. Titles our main source of information, which is very beneficial in matching same products. Images also our main source of information, because rarely aggregators use different images on the same item.

In the preprocessing stage, several essential steps are taken to prepare the text data for further analysis. Firstly, special characters are removed from both the title and body of the text. This helps eliminate any unwanted symbols or punctuation that may hinder the accuracy of subsequent processing steps. Next, repeated words were removed, because they can be noise for our model. To ensure consistency and avoid issues with word matching, all characters in the text are converted to lowercase. Lastly, the words are stemmed, which involves reducing

them to their base or dictionary form. This step helps capture the core meaning of the words and reduces variations, such as different tenses or plural forms, to enhance the accuracy of subsequent analysis tasks. By performing these preprocessing steps, the text data is cleansed and standardized, ready for various natural language processing techniques and machine learning algorithms. The code is located in Appendix A. Preprocessed titles can be seen in figure 4.1

```
df[['wildb_title', 'wildb_title_preprocessed', 'ydx_title', 'ydx_title_preprocessed']].sample(10)
```

	wildb_title	wildb_title_preprocessed	ydx_title	ydx_title_preprocessed
9099	Меловая магнитная доска "Сердце" для рисования на холодильник + мел / детская грифельная черная	мелов магнитн доск сердц для рисован на холодильник мел детск грифельн черн	"Магнитная меловая доска Doski4you ""Сердце"" для рисования на холодильник, комплект / детская грифельная мел"	магнитн мелов доск doski4you сердц для рисован на холодильник комплект детск грифельн мел
903	Чехол LYAMBDA REGUL для Samsung Galaxy A30s/A50/A50s (LA06-RG-A50-BK) Black	чехол lyambda regul для samsung galaxy a30sa50a50s la06rga50bk black	Чехол LYAMBDA REGUL для Samsung Galaxy A30s/A50/A50s (LA06-RG-A50-BK) Black	чехол lyambda regul для samsung galaxy a30sa50a50s la06rga50bk black
6145	Массажное масло Chicco Baby Moments 200 мл	массажн масл chicco baby moments 200 мл	Chicco Масло массажное Baby Moments, 200 мл	chicco масл массажн baby moments 200 мл
1517	Папка-планшет пластиковая ErichKrause Diamond Original, A4, черный (в пакете по 4 шт.)	папкапланшет пластиков erichkrause diamond original a4 черн в пакет по 4 шт	ErichKrause Портфель пластиковый Diamond Original A4 черный	erichkrause портфел пластиков diamond original a4 черн
9546	Набор инструментов БОХУМ BG078-1214, 78 шт.	набор инструмент бохум bg0781214 78 шт	Набор инструментов BERGER Бохум BG078-1214, 78 предм.	набор инструмент berger бохум bg0781214 78 предм
7219	Углошлифовальная машина УШМ-180/1800М, 180мм, 1800 Вт	углошлифовальн машин ушм1801800м 180мм 1800 вт	УШМ Интерскол УШМ-180/1800М, 1800 Вт, 180 мм	ушм интерскол ушм1801800м 1800 вт 180 мм
709	Рамка для выключателей и розеток двойная стекло Эра Elegance голубой 14-5102-28	рамк для выключател и розеток двойн стекл эр elegance голуб 14510228	ЭРА 14-5102-28 ЭРА Рамка на 2 поста, стекло, Эра Elegance, голубой+бел (5/50/1200)	эр 14510228 рамк на 2 пост стекл elegance голубойбел 5501200
11581	Аэрограф AGS7/0.2	аэрограф ags702	Аэрограф Fubag AGS7/0.2	аэрограф fubag ags702
6985	LA ROCHE-POSAY / HYDRAPHASE UV INTENSE LEGERE Интенсивный увлажняющий флюид для лица SPF20, 50 мл.	la rocheposay hydraphase uv intense legere интенсивн увлажня флюид для лиц spf20 50 мл	La Roche-Posay Hydraphase UV Intense Legere Интенсивный увлажняющий флюид с защитой от UV для лица, шеи и области декольте, 50 мл	la rocheposay hydraphase uv intense legere интенсивн увлажня флюид с защит от для лиц ше и област деколь 50 мл
8160	Замок навесной Ч/50.	замок навесн ч50	"ЗАМОК НАВЕСНОЙ ""PARK"" Ч/50 (1/12/120)"	замок навесн park ч50 112120

Figure 4.1: Preprocessed titles.

In the future engineering steps, we employ the CountVectorizer to convert our text documents into a matrix of token counts. This process allows us to represent the textual data in a numerical form, enabling further analysis. Additionally, we utilize the tf-idf transformer to transform the count matrix into a normalized tf-idf representation by formula 4.1.1. This transformation helps capture the importance of each token in the document collection by taking into account both the term frequency (tf) by formula 4.1.2 and the inverse document frequency (idf) by formula 4.1.3. By incorporating these techniques, we enhance the effectiveness and depth of our text analysis and modeling approaches.

$$tf\ idf(t, d, D) = tf(t, D) * idf(t, D) \tag{4.1.1}$$

$$tf\ (t, d) = log(1 + freq(t, d)) \tag{4.1.2}$$

$$idf(t, D) = log(\frac{N}{count(d \in D : t \in d)}) \tag{4.1.3}$$

For the images, the first step in the preprocessing stage is to resize the images. Using the Tensorflow library, each image has been shrunk to 256 by 256 so that the deep learning model can train more quickly on smaller images. Depending on the quantity and size of pixels, the neural network model must learn a lot while working with a larger input image. Therefore, scaling images is thought to be one of the most effective data pre-processing methods for lowering the training overhead. The photos are enhanced and pre-processed to lessen model bias and increase their generalizability before the image embedding is extracted. Following this stage, an array of images is created and fitted to the models in order to extract the image embedding. The code is located in Appendix C. Product match by images from different sources can be seen in figure 4.2

Figure 4.2: The same images from different sources.

4.2 Main results of work

Deep learning techniques play a major role in both decision-making and behaviour analysis. To predict matching commodities, three deep learning models that have already been trained are utilised. One of these is transfer learning, which produces exact results by using a weight that has previously been trained. It starts by linearly organising the array, and then it assigns a score to each component of the image to analyse it. Essentially, it is adding extra relevance to every aspect of the picture, which will help with identification later on.

In this job, three deep learning based algorithms including Vgg-19, Mobilenet, and Resnet-50 are utilized. The superlative model having the most weighted custom metric can be employed. This metric is founded upon the principle that similar images should exhibit greater cosine similarity while the Levenshtein distance for text ought to be minimized. The weights are assigned to each metric so that, for example, cosine similarity is assigned 0.6 and the normalised Levenshtein distance is assigned 0.4. This is because the product image is the same everywhere in the world, but the text may vary depending on linguistic constraints. As a result, a weighted sum of both metrics is taken into consideration for model evaluation. The implementation of several libraries was facilitated by NumPy, plotly, scipy, sklearn, TensorFlow, Keras, OpenCV, tqdm, and others. The reason Google Collab should be employed when training these models is its availability of free GPU services while CNN is needed in this suggested process. The programming is done with Python language. The following specifications have been required to put this model into practice: google colab with 12GB RAM, 4 CPU Cores and GPU Nvidia P100.

Every implemented model must be evaluated using the normalised Levenshtein distance, custom metric, and cosine similarity. The problem at hand involves recommendation, specifically multiclass recommendation. To assess several pretrained deep learning models, test data is used to calculate cosine similarity, Levenshtein distance, and a proprietary metric. All measures are intended once each model has been trained on training data and evaluated on test data. The best model for suggesting related products is determined by calculating the custom metric's highest value. To see how the implemented models compare, bar charts are plotted. Because the dataset is big, it takes roughly 4 hours to train each model.

4.2.1 Results

Image embedding is retrieved using models like VGG-19, ResNet-50, and Mobilenet, which have been developed using the train data. Following the model's processing of each image, the list's image embeddings are concatenated. Moreover, these models predict the five most similar images based on the test image,

and their cosine similarities are computed in relation to the test image.

To determine the similarity between the image - Cosine Similarity, and for the text - Levenshtein Distance respectively is used. The custom metric is a total similarity score that has been computed using these two metrics. When it comes to similarity detection, the model with the greatest custom metric score will be considered the most efficient. The results of these metrics indicate the likelihood of this approach being used in actual e-commerce portal applications.

The average cosine distance for the Mobilenet Model is 0.88. The average cosine similarity score for VGG-19 is 0.75. Whereas the average cosine similarity for ResNet50 is 0.70. Afterwards, the text data is retrieved and their levenshtein distance is computed with regard to test dataset based on the suggested five images. The average normalised Levenshtein distance for the VGG-19 model is determined to be 0.42, while the normalised Levenshtein distance for the MobileNet and ResNet-50 models are respectively 0.48 and 0.51.

The best model is the one with the lowest normalised Levenshtein distance and the highest average cosine similarity. A custom metric, or total similarity score, has been calculated based on the Cosine Similarity and Levenshtein Distance to make this calculation considerably simpler. Figure 4.3 displays for each model Average Cosine, Levenshtein distance and the custom metric in relation to test dataset.

It has been determined through analysis of the numbers in the bar chart presented in Figure 4.3 that the custom metric for the MobileNet model is 0.74. Whereas the custom metric score for the ResNet-50 model is 0.68 and for the VGG-19 model is 0.70, all three metrics were obtained for this model by following the equivalent processes stated above. The MobileNet model has demonstrated the greatest results for similarity detection for test dataset.

We measured accuracy by finding the closest item and compare it with ground truth item. The code is located in Appendix D. In terms of accuracy MobileNet model showed the highest score - 0.82, while VGG-19 showed 0.77 accuracy and ResNet-50 0.75 accuracy respectively.

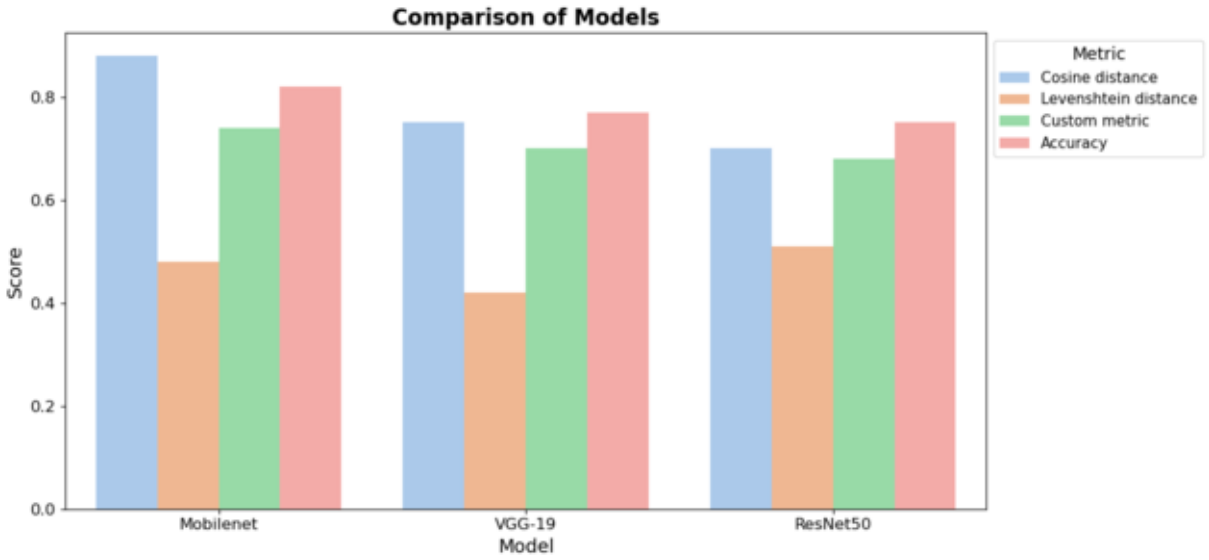

Figure 4.3: Different Models comparison: Cosine similarity vs Normalized Levenshtein distance vs Custom metric.

The results of the studies indicate that MobileNet Architecture is the most effective model for matching product similarities, outperforming other models like VGG-19 and ResNet-50. MobileNet is a lightweight convolutional-layered model that is extremely intricate and deep. Based on the aforementioned analysis, the mobilenet model performs exceptionally well in terms of cosine similarity and is marginally closer to other models in terms of Levenshtein distance output. The findings of all models are equivalent because they are all deep models that were trained using ImageNet data. Although the matched products predicted by these algorithms are nearly identical to the mobilenet model, the custom metrics value is lower, the Levenshtein distance is larger, the cosine similarity is smaller and the accuracy is the highest. Because the dataset is big, it takes over four hours to train each model. Result can be found in table 4.2.1

Table 1: Results metrics of models

	Cosine distance	Levenshtein distance	Custom metric	Accuracy
Mobilenet	0.88	0.48	0.74	0.82
VGG-19	0.75	0.42	0.70	0.77
ResNet50	0.70	0.51	0.68	0.75

Description: Comparison of various models based on different metrics including Cosine distance, Levenshtein distance, Custom metric and Accuracy.

Finally, in order to present the results of our work, a web platform was created.

Here, you can upload the URL of an item image and the title of the item 4.4. The algorithm will find the best match from the list of items provided earlier. For demonstration purposes, 1000 items were uploaded. The code is located in Appendix E and Appendix F.

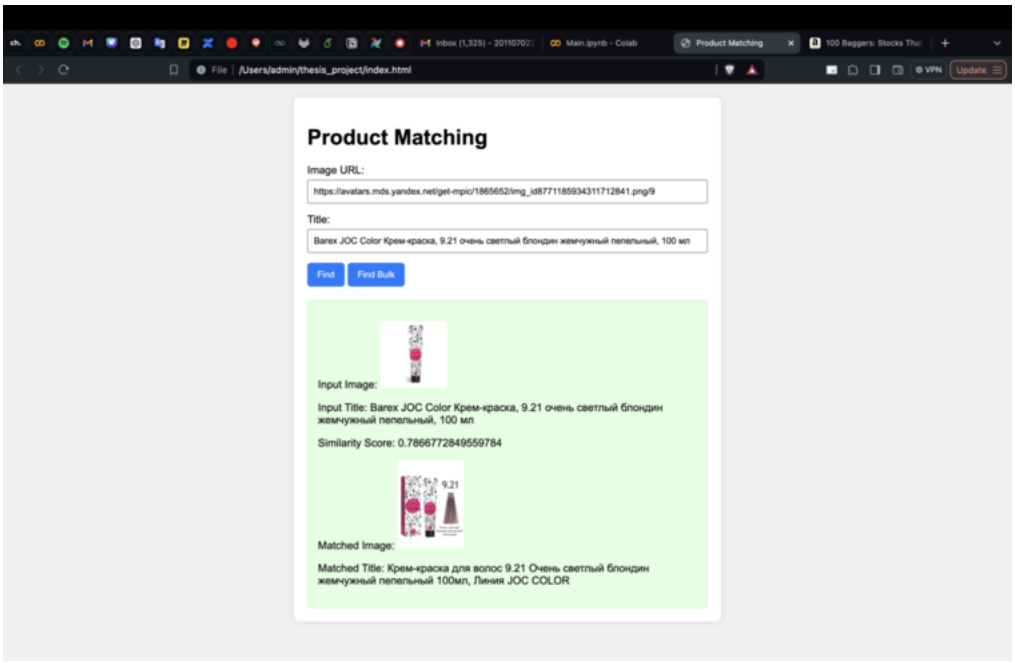

Figure 4.4: Web Platform

Chapter 5

Conclusion

Finding the products that match the best or predicting which ones are the most comparable are still difficult tasks. In this study, images and text embeddings, together with the labels corresponding to each image, are extracted using the first pre-trained deep convolutional models. After determining the Levenshtein distance and the model cosine similarity, a custom metric is calculated. Based on the text and image with the greatest custom metric score, average normalised Levenshtein distance, and average cosine similarity, the pretrained Mobilenet model may be trained to find most relevant items or comparable images.

The deep convolutional architecture and lightweight design of this model are important implementation factors. A dataset including about 14000 images with corresponding titles is used in this prediction and analysis study to identify comparable images. By identifying the most comparable products based on the image and text that contains the label or tags associated with the image, this study achieved its goal of detecting similar goods. Finally, in order to present the results of our work, a web platform was created. Under the hood, a pre-trained MobileNet model was used to find similar images.

The study also shed light on the pervasive challenge of working with limited computational capabilities, the obtained results are somewhat limited to their optimal results due to the limited amount of product images and their titles. Future research can incorporate a more sophisticated matching product system

with the vast amount of product image and labelled tag data, undoubtedly leading to improved accuracy and outcomes. More processing power will result in a faster training time for these models. E-commerce companies can save a great deal of time by classifying identical products with more accuracy. These methods can also be applied to recommendation systems, which assist customers in making the best decision.

Bibliography

[1] Petar Ristoski et al. A machine learning approach for product matching and categorization. Semantic web 9.5, 2018.

[2] Anitha Kannan et al. Matching unstructured product offers to structured product specifications. pages 404–412, 2011.

[3] Mikhail Bilenko and Raymond J Mooney. Adaptive duplicate detection using learnable string similarity measures. pages 39–48, 2003.

[4] Hanna Köpcke et al. Tailoring entity resolution for matching product offers. pages 545–550, 2012.

[5] Amir H. Gandomi Alicja Martinek†, Szymon Łukasik†. Text-based product matching - semi-supervised clustering approach. pages 5–6, 2024.

[6] Leonidas Akritidis. Effective unsupervised matching of product titles with k-combinations and permutations. pages 5–7, 2018.

[7] Christian Bizer Ralph Peeters. Integrating product data using deep learning. 2021.

[8] Ajinkya More. Attribute extraction from product titles in ecommerce. pages 5–6, 2016.

[9] Rosie Hood. Unravelling product matching in retail with ai. 2019.

[10] Ajinkya More. Product matching in ecommerce using deep learning. 2017.

[11] Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. W3C recommendation, 10(10):2004, 2004.

[12] John F Sowa. Principles of semantic networks: Explorations in the representation of knowledge. Morgan Kaufmann, 2014.

[13] Robert F Simmons et al. Semantic networks: Their computation and use for understanding English sentences. Department of Computer Sciences and Computer-Assisted Instruction Laboratory ..., 1972.

[14] Christiane Fellbaum, Musa Alkhalifa, W Black, Sabri Elkateb, Adam Pease, H Rodriguez, and PTJM Vossen. Introducing the arabic wordnet project. In Third Global Wordnet Conference, 2006.

[15] Erik T Mueller. Natural language processing with thought treasure, 1998.

[16] Y. Aloimonos. Special issue on purposive and qualitive active vision, cvgip b: Image understanding. 56, 1992.

[17] D. Marr. Vision: A computational investigation into the human representation and processing of visual information. 1982.

[18] L. Roberts. Machine perception of 3d solids. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 159–197. MIT Press, 1965.

[19] D. Savir and G. J. Laurer. The characteristics and decodability of the universal product code symbol. 1975.

[20] D. L. Brock. Integrating the electronic product code (epc) and the global trade item number (gtin). 2002.

[21] J. Jovanovic and E. Bagheri. Electronic commerce meets the semantic web. it professional. pages 56–65, 2016.

[22] R. Agrawal A. Kannan, I. E. Givoni and A. Fuxman. Matching unstructured product offers to structured product specifications. pages 404–412, 2011.

[23] U. Brunner and K. Stockinger. Entity matching with transformer architectures a step forward in data integration. 2020.

[24] T. S. H. Teo. Attitudes toward online shopping and the internet. Behaviour Information Technology, 2002.

[25] T. Rekatsinas A. Doan Y. Park G. Krishnan R. Deep E. Arcaute S. Mudgal, H. Li and V. Raghavendra. Deep learning for entity matching: A design space exploration. pages 19–34, 2018.

[26] P. S. G. C. A. Doan A. Ardalan J. R. Ballard H. Li F. Panahi H. Zhang J. Naughton S. Prasad G. Krishnan R. Deep P. Konda, S. Das and V. Raghavendra. Magellan: Toward building entity matching management systems over data science stacks. Proc. VLDB Endow., 9(13), 2016.

[27] Matching the same items from different marketplaces. 2021. URL https://docs.google.com/document/d/1ATQeY6ZnCcRsleUzjSEIbWWtI2CImalpdZ4LcKvuVJY/edit?pli=1#heading=h.hjw4cmcnpwgy.

Appendix A

Preprocessing

```
def preprocess_text(text):
    text = text.lower()

    text = re.sub(r'[^\w\s]', '', text)

    words = word_tokenize(text)

    stemmer = SnowballStemmer("russian")
    stemmed_words = [stemmer.stem(word) for word in words]

    unique_words = list(OrderedDict.fromkeys(stemmed_words))

    preprocessed_text = ' '.join(unique_words)

    return preprocessed_text
```

Appendix B

Parsing image urls

```
def get_img_link(item_id):
    item_id = str(item_id)
    vol = str(item_id)[:len(item_id)-5]
    part = str(item_id)[:len(item_id)-3]
    img = ''

    for i in range(1, 17):
        i = str(i)

        if int(i)<10:
            i = '0'+i

        url = f'https://basket-{i}.wbbasket.ru/vol{vol}/part{part}/{item_id}/images/big/1.webp'

        response = requests.get(
            url,
            headers=headers,
            timeout=10
        )

        if response.status_code == 200:
            img = url
            break

    if response.status_code != 200:
        img = response.status_code

    return img
```

Appendix C

Preprocessing images

```
def load_and_preprocess_image(image_url, model_name):
    size = (224, 224)
    response = requests.get(image_url)
    img = Image.open(BytesIO(response.content))
    img = img.convert('RGB')
    img = img.resize(size, Image.NEAREST)
    img_array = img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    if model_name == 'vgg19':
        return preprocess_input_vgg19(img_array)
    elif model_name == 'mobilenetv2':
        return preprocess_input_mobilenetv2(img_array)
    elif model_name == 'resnet50':
        return preprocess_input_resnet50(img_array)
```

Appendix D

Evaluation

```
def evaluate_accuracy(similarity_scores, initial_df):
    correct_matches = 0
    for idx, scores in enumerate(similarity_scores):
        max_sim_index = np.argmax(scores)
        correct_index = idx

        if max_sim_index == correct_index:
            correct_matches += 1

    accuracy = correct_matches / len(similarity_scores)
    return accuracy
```

Appendix E

Web platform backend

```
from fastapi import FastAPI, HTTPException
from fastapi.middleware.cors import CORSMiddleware
from pydantic import BaseModel
from typing import List, Dict
import pandas as pd
import numpy as np
from io import StringIO
from Levenshtein import distance as levenshtein
from sklearn.metrics.pairwise import cosine_similarity

import re
import nltk
nltk.download('stopwords')
nltk.download('punkt')
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from collections import OrderedDict
from nltk.stem.snowball import SnowballStemmer

import requests
from tensorflow.keras.preprocessing.image import img_to_array, load_img
from io import BytesIO
from PIL import Image
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input as
    preprocess_input_mobilenetv2

app = FastAPI()

```

```
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

class Item(BaseModel):
    image_url: str
    title: str

class DataFrameInput(BaseModel):
    data: List[Dict[str, str]]

df = pd.read_csv('wildb_dataset.csv')

wilbd_embeddings = np.load('wilbd_embeddings.npy')


def load_and_preprocess_image(image_url, model_name):
    size = (224, 224)
    response = requests.get(image_url)
    img = Image.open(BytesIO(response.content))
    img = img.convert('RGB')
    img = img.resize(size, Image.NEAREST)
    img_array = img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    if model_name == 'vgg19':
        return preprocess_input_vgg19(img_array)
    elif model_name == 'mobilenetv2':
        return preprocess_input_mobilenetv2(img_array)
    elif model_name == 'resnet50':
        return preprocess_input_resnet50(img_array)

def get_model_features(df, model, preprocess_function, col):
    model_features = []
    for img_path in df[col]:
        processed_img = load_and_preprocess_image(img_path,
   preprocess_function)
        features = model.predict(processed_img)
        features = features.flatten()
        model_features.append(features)
    return np.array(model_features)

def preprocess_text(text):
```

```
    text = text.lower()
    text = re.sub(r'[^\w\s]', '', text)
    words = word_tokenize(text)
    stemmer = SnowballStemmer("russian")
    stemmed_words = [stemmer.stem(word) for word in words]
    unique_words = list(OrderedDict.fromkeys(stemmed_words))
    preprocessed_text = ' '.join(unique_words)

    return preprocessed_text

@app.post("/calculate_similarity")
async def calculate_similarity(item: Item) -> Dict:
    global df
    global wilbd_embeddings

    input_image = item.image_url
    input_title = item.title
    df_to_compare = pd.DataFrame(data={'img': [input_image], 'title': [input_title]})
    df_to_compare['title2'] = df_to_compare.title.apply(preprocess_text)

    df = df.rename(columns={'wildb_title_preprocessed': 'title', 'wildb_img': 'img'})
    mobilenetv2_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
    ydx_img_features = get_model_features(df_to_compare, mobilenetv2_model, 'mobilenetv2', 'img')

    score_list = []
    for i in range(len(wilbd_embeddings)):
        score = cosine_similarity([wilbd_embeddings[i]], ydx_img_features)[0][0]
        score2 = levenshtein(df.title[i], df_to_compare.title2[0])/100

        total_score = 0.6*score+0.4*(1-score2)
        score_list.append(total_score)

    max_index = np.argmax(score_list)
    max_score = score_list[max_index]+0.3
    df = df.reset_index(drop=True)
    matched_title = df.wildb_title[max_index]
    matched_image = df.img[max_index]

    if max_score<0.7:
        matched_title = 'No match'
        matched_image = 'No match'
```

```
116
117     return {
118         "input_image": item.image_url,
119         "input_title": item.title,
120         "similarity_score": max_score,
121         "matched_image": matched_image,
122         "matched_title": matched_title
123     }
124
125 if __name__ == "__main__":
126     import uvicorn
127     uvicorn.run(app, host="127.0.0.1", port=8000)
```

Appendix F

Web platform frontend

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Product Matching</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 20px;
            background-color: #f0f0f0;
        }
        .container {
            max-width: 600px;
            margin: auto;
            padding: 20px;
            background-color: #fff;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            border-radius: 8px;
        }
        .form-group {
            margin-bottom: 15px;
        }
        label {
            display: block;
            margin-bottom: 5px;
        }
        input[type="text"], input[type="file"] {
            width: 100%;
```

```
            padding: 8px;
            box-sizing: border-box;
        }
        button {
            padding: 10px 15px;
            background-color: #007bff;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
        .result {
            margin-top: 20px;
            padding: 15px;
            background-color: #e0ffe0;
            border: 1px solid #b2ffb2;
            border-radius: 5px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Product Matching</h1>
        <div class="form-group">
            <label for="image-url">Image URL:</label>
            <input type="text" id="image-url" required>
        </div>
        <div class="form-group">
            <label for="title">Title:</label>
            <input type="text" id="title" required>
        </div>
        <button onclick="submitForm()">Find</button>
        <div id="result" class="result" style="display:none;"></div>
    </div>

    <script>
        async function submitForm() {
            const imageUrl = document.getElementById('image-url').value;
            const title = document.getElementById('title').value;

            const response = await fetch('http://127.0.0.1:8000/
   calculate_similarity', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json',
                    'Accept': 'application/json'
```

```
                },
                body: JSON.stringify({ image_url: imageUrl, title: title })
            });

            if (!response.ok) {
                const errorText = await response.text();
                alert(`Error: ${errorText}`);
                return;
            }

            const data = await response.json();
            document.getElementById('result').style.display = 'block';
            document.getElementById('result').innerHTML = `
                <p>Input Image: <img src="${data.input_image}" alt="Input Image" width="100"></p>
                <p>Input Title: ${data.input_title}</p>
                <p>Similarity Score: ${data.similarity_score}</p>
                <p>Matched Image: <img src="${data.matched_image}" alt="Matched Image" width="100"></p>
                <p>Matched Title: ${data.matched_title}</p>
            `;
        }
    </script>
</body>
</html>
```