# Upoma: A Dynamic Online Price Comparison Tool for Bangladeshi E-commerce Websites

Ashraful Alam, Atqiya Abida Anjum, Fahmid Shafat Tasin, Mizanur Rahman Reyad,
Sadia Afrin Sinthee, Nahid Hossain
*Department of Computer Science And Engineering*
*United International University, Dhaka, Bangladesh*
aamorroid66@gmail.com, atqiyanabila01@gmail.com, fahmidtasin@gmail.com, reyad.p220@gmail.com,
s.sadiaafrin.s@gmail.com, nahid@cse.uiu.ac.bd

***Abstract*—The e-commerce industry is rapidly emerging all over the world. E-commerce sites provide the independence to choose and buy things over the internet. However, different websites put different prices on the exact same product, and this leads people to spend extra money on products which could be bought with less. Although there are several e-commerce sites available in Bangladesh, no price comparison tool yet developed. Thus, we have designed and developed a price comparison tool for Bangladeshi websites named 'Upoma (Comparison)' which supports comparison even if the titles and prices of products are written English, Bangla, and Transliterated Bangla. We have fetched over 90,53,015 product details from various websites to enrich our result based on the user's query. In order to make our system dynamic and to keep pace with real-time changes occurring on the sites, our database is automatically updated in every 12 hours. Our system displays the result with 93.06% accuracy according to the user's query.**

***Index Terms*—Price Comparison, E-commerce, Web Mining, Natural Language Processing.**

## I. Introduction

Nowadays, e-commerce websites have been prevalent and growing up in an unprecedented manner. As per the report of Statista 2019, the e-commerce market has collected 3.53 trillion US dollars [1], which is suggestive of the proliferation of e-commerce websites around the world. As per the stats above, it can be derived that a massive number of people rely on buying various from e-commerce websites rather the physical stores. E-commerce website holders are deliberately putting prices on their websites derailed from actual rates leveraging on people's demand. Therefore, Consumers unbeknownst of the actual cost of different products beforehand are buying products by squandering money. Thus people are being deceived, paying more money than necessary to buy a product. Synthesizing the fact above, the importance of the price comparison tool is beyond gainsaying. However, there is a substantial amount of researches, browser Add-ons have been proposed in foreign languages and freely available in many countries including India. With that being said, two notable price comparison tool which has been introduced in India, namely www.mysmartprice.com and the other one is www.compareraja.in, similar to these two Indian websites, pricemama.com.bd does serve the purpose of a price comparison tool in our country. However, this website does not provide efficient results and does not support Bangla language searches such as our price comparison tool does. This motivates us to develop a price comparison tool for Bangladesh and for Bangla language. Our Solution aims for a collaborative platform for allowing a consumer to evaluate the price and make the purchase decision more manageable according to their budget.

A considerable amount of congruent works but in foreign languages had been deployed earlier. However, Likening to our proposed method, J. Nakash et al. had proposed a solution that uses an inverted indexing technique to compare the products found on different websites In 2015 [2]. Another work similar to our method was published In 2019 by S. Mehak et al. They designed a tool for price comparison which uses scrapping scripts written with a python library and improvise the storage for scrapped data [3]. In 2018, K.Pradeep et al. formulated a pattern analysis recommender system by analyzing buying patterns using data mining technique [4]. In 2016, R. shah et al. established a website with Django framework and MongoDB for comparing price using web crawling and also used request and BeautifulSoup4 library for web scrapping [5]. In 2015, C. Zheng et al. established a way to use Beautiful Soup library for retrieval of information from webpage [6] . A. Horch et al. proposed a method of automated identification and extraction of product price data from arbitrary e-shop websites in this same year [7]. In 2014, Y. Ming-Hsiung et al. came up with a method to find the lowest price and notify the user for better satisfaction using web mining techniques [8].

In this paper, we have proposed a robust price comparison tool for Bangladeshi e-commerce websites and for Bangla language. We have used the Scrapy framework to build spiders to crawl through the websites and fetch or scrape information posted on the site [9]. Fetched data is stored in the CSV database. We vectorize titles and use the Cosine Similarity algorithm to find the similarity between the search text and dataset text [10]. We have used the Flask framework for integration between python and website [11]. We have designed User Interface for a user-friendly interaction while searching for query and for showing results appertaining to correspondent query.

This paper is organized as follows: In section 2, we describe the proposed system and step by step explanations of our work and algorithm. The paper illustrates the experimental result and performance analysis in section 3, while section 4 encompasses the paper with limitation of our system and plan for future work.

## II. PROPOSED METHOD

In this section, we have demonstrated different aspects and the procedure of implementation of our price comparison tool. Algorithm 1 demonstrates the pseudocode for the creation of the database where Algorithm 2 represents the pseudocode for the language processing and comparison methods.

**Algorithm 1** Dataset creation

```
1: for each E-commerce site do
2:     for each category in the site do
3:         for each product in the category do
4:             ProductDetails→Name,Price,link,Image
5:             Use CSS selector to find ProductDetails
6:             remove garbage data
7:             CSV dataset ← ProductDetails
8:         end for
9:     end for
10: end for
```

### *A. Dataset Creation*

Dataset is the core element of our price comparison tool. Thus, we have used web crawling approach to collect necessary data from different websites in a fast and efficient way [12]. We have built different spiders for different websites using Python's powerful Scrapy framework to collect information about products [13].After that, we store the extracted data in an individual CSV file under the column named "Image", "Product name", "Product price", "Offer", and "Website link" for every website [14]. Once the crawler has completed it traversing through the whole website, it starts to scrape off the correspondents to a particular product discarding superfluous information. Since websites tend to perform changes in products after a specific time, our system has been equipped with dynamicity to keep pace with the changes. We have introduced an automated script that runs in every 12 hours and renews the data in our database.

Table I: Product count in the database.

| E-Commerce Websites | Category | Total Product |
|---|---|---|
| Pickaboo.com | 5 | 1267 |
| Othoba.com | 10 | 31875 |
| Bagdoom.com | 36 | 25838 |
| Sindabad.com | 10 | 18031 |
| Phoneshopbd.com | 6 | 985 |
| Phonesellbd.com | 6 | 1053 |
| Daraz.com.bd | 12 | 8959595 |
| Banglashoppers.com | 8 | 1985 |
| Jadroo.com | 11 | 12386 |

These websites are selected for scraping data based on factors such as diversity of products, popularity and trust, technical support, and scalability. On this note, some of the leading websites have been eliminated such as chaldal.com as Chaldal.com focuses only on grocery items. Another notable website namely bdshop.com cannot be included as the website only provides hardware equipment. We have also ruled out Evaly.com and Daraz.com for being third party platforms for selling products.
The spiders are designed in such a way if an e-commerce website authority does remove any category, product or make changes in any products under specific categories, spiders will not be halted and will remain unaffected despite the changes. The aforementioned process is repeated until all websites are traversed, and relevant data is stored. The pseudocode of the method is shown in Algorithm 1. We have collected over 90,53,015 data by scraping off the websites. A summary of it is shown in Table 1.

### *B. Transliteration*

A user can put any product name in the search bar in both Bangla and English language. We have implemented indic-transliteration to perform transliteration on every query. Transliteration is a process that Romanizes every letter present in the query. Firstly, indic-transliteration checks for the language. If the input is in Bangla, the query is transliterated into English and vice-versa with the concept of double metaphone method. [15].

### *C. Processing of Bag of Words & Vectorization*

NLP algorithms are capable of vector-based representation rather than the symbolic representation of words. For this reason, we have implemented a text modeling algorithm, "Bag of Words". Bag of words approach preprocess the raw data by turning all the words that exist in the database into a bag of words, including the words presented in queries and paired with their word count per database.

We have generated a vector representation of the database using TF-IDF. It has been used to depict the significance of word to its corresponding single document in digital libraries, around 83 percent of text-based recommendation [16].

### *D. Algorithm for similarity checking*

We have contrived an exact match function from the scratch to extract the precise product what the user has searched for. Exact products are presented in a descending manner. Thus, the product with a lower price ranks first in that order. To find the congruence between user input and the data present in the database, we have constructed the cosine similarity algorithm by using the Sci-kit learn library. Cosine similarity distinguishes between two or more documents on account of the orientation (angle), not the magnitude [17]. The similarity between two documents is calculated by the cosine of the angle of two vectors. Mathematically measurement of cosine angle of the two vectors projected in multidimensional space is considered as distance measurement of two documents.

Cosine Similarity focuses on the common words and the occurrence frequency of common words between common words. In addition, Cosine similarity overcomes the flaw of zero matches between two documents irrespective of their sizes. For instance, if a and b are two vectors then cosine similarity uses the following rule:

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}} - - - - - - - - - (1)$$

Where , $\vec{a}\cdot\vec{b} = \sum_1^n a_i b_i = a_1b_1 + a_2b_2 + \cdots + a_nb_n$ is the dot product of the two vectors.
If the cosine value is 0, then the angle between a and b vector, two vectors are at 90 degrees to each other and share no similarity. Thus it can be depicted that two vectors are similar when the cosine of the angle of two vectors is smaller.

**Algorithm 2** Product search

```
1:  InSent ← Product to be searched
2:  T ← Minimal similarity threshold
3:  DetectLanguage($InSent$)
4:  if detected language of the input is Bengali then
5:      Transliterate InSent to English as TSent
6:  else
7:      Transliterate InSent to Bengali as TSent
8:  end if
9:  for each website in our data set do
10:     generate bag of words with TF-IDF
11:     for all products do
12:         PN ← product name
13:         if PN matches with InSent then
14:             put product details in ExactList
15:         end if
16:         if PN matches with TSent then
17:             put product details in ExactList
18:         end if
19:         if cosine similarity($PN, InSent$) >=T then
20:             put product details in SimilarList
21:         end if
22:         if cosine similarity($PN, TSent$) >=T then
23:             put product details in SimilarList
24:         end if
25:     end for
26: end for
27: Sort ExactList according to price in ascending order
28: Sort SimilarList according to similarity in descending
    order and price in ascending order
```

On account of cosine similarity, the user can see similar products which are corresponding to their queries. However, there may be too many similar products on different e-commerce websites similar to that of the user's choice. As the cosine similarity algorithm is performed on every bag (multiset) of words of each website and sorts similar products by their rate of similarity, it can create a long list with repeated or less similar products. Which will certainly increase the difficulty to find the right product from the prolonged list of products. For this reason, we have introduced a variable to control the length of the list, namely, the similarity threshold. The similarity threshold controls how many similar products a user wants to see in the resultant list. It is controlled with a numbered slider bar that dictates the minimum percentage of similarity needed for a product to be able to appear in the result. This value can vary from 0-99 %. We have excluded 100 % match as it is done in a separate algorithm that can detect exact Matched products. Similar results are based upon the query of the user and displayed in an ascending manner.

The result of exact match and cosine similar match are shown in the different tables for convenience.

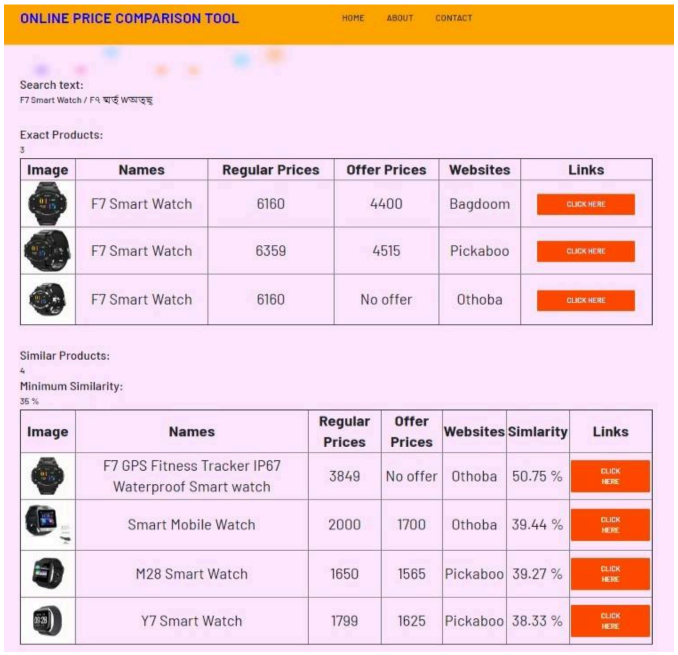
### *E. Input Constraint Removal*

Cosine Similarity and exact match function do find the exact match or similar product contingent to the query. For instance, if user input is in Bangla such as "নকিয়া ৩৩১০" transliteration converts it into "Nokia 3310" and shows both exact and similar result if such data is present in the database. In another case, If the user shuffles the words of their query, it does not affect on our system producing the result. For instance, if a user inserts a query "3310 Nokia" instead of "Nokia 3310", our system will bring the correct result if the product details are available in our database.

## III. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

In this section, we have thoroughly illustrated the experimental results and performance analysis. The experiment was done Google Colab [18] and in a high-performance server computer with 32GB RAM, 6GB Graphics Card, and Intel Xenon Processor. Python 3.7 was the core language and we use some libraries from Scikit-learn [19]. Our system shows results according to the user's query and user-defined threshold.

According to Figure 1, the user's query text is "F7 smart-watch", and the transliterated query is shown next to it. Our system shows three products found similar to the user's choice with Image, regular price, offer price, and a corresponding website link. If a user wants to buy a product of his choice, the link given in our site will redirect to the corresponding product page. If there is no difference between regular price and offer price, the result comes with a "No offer" tag beside the product. In this figure, the first table shows the exact product which the user has searched for. The 2nd table refers to all of the products that are similar to the user's search. The 2nd table is displaying the results, which are 35% or higher similar to the user search. Hereby, if a user increases the similarity threshold, there will be less product manifestation in the 2nd table. In the other case, If a user increments the similarity threshold to 99%, the 2nd table will not appear with similar products. We have tested our system by putting automated search queries and adjusting the similarity threshold



Search text:
F7 Smart Watch / F৭ স্মার্ট ওয়াচ

Exact Products:
3

| Image | Names | Regular Prices | Offer Prices | Websites | Links |
|---|---|---|---|---|---|
| | F7 Smart Watch | 6160 | 4400 | Bagdoom | CLICK HERE |
| | F7 Smart Watch | 6359 | 4515 | Pickaboo | CLICK HERE |
| | F7 Smart Watch | 6160 | No offer | Othoba | CLICK HERE |

Similar Products:
4
Minimum Similarity:
35 %

| Image | Names | Regular Prices | Offer Prices | Websites | Simlarity | Links |
|---|---|---|---|---|---|---|
| | F7 GPS Fitness Tracker IP67 Waterproof Smart watch | 3849 | No offer | Othoba | 50.75 % | CLICK HERE |
| | Smart Mobile Watch | 2000 | 1700 | Othoba | 39.44 % | CLICK HERE |
| | M28 Smart Watch | 1650 | 1565 | Pickaboo | 39.27 % | CLICK HERE |
| | Y7 Smart Watch | 1799 | 1625 | Pickaboo | 38.33 % | CLICK HERE |

Figure 1: Result appearance

in different values. In every search, we have found at least one exact result matched with the query. We have considered it as an error if the exact product does not appear on the result page despite the availability of that product in the database. After performing 1500 queries, we have observed that our system shows the result with 93.06% of reasonably good accuracy.

## IV. CONCLUSION AND FUTURE WORK

The paper illustrated a comprehensive price comparison tool based on Bangladesh's perspective. We have suggested the lowest price for the same product based on the similarity between the user's query and products stored in the database. We have built a first-ever price comparison tool that can process queries in Bangla language. Although we have reached our goal, the system is not beyond limitation. We have used generic indic-transliteration to process data in Bangla. As there is less English alphabet than the Bangla alphabet, indic-transliteration sometimes fails to Romanize the whole query properly. For this reason, our next strategy is to develop a transliteration algorithm to conform to our system. In the future, we are planning to arrange a manifestation of consumer reviews and ratings to help consumers to identify the right product to cover their needs according to their budget. Moreover, we will use Neural Networks to generate different predictions and suggestions based on the user's query.

## References

[1] Esther Shaulova and Lodovica Biagi. ecommerce report 2019.

[2] Jawahire Nakash, Shaikh Anas, Siddiqi Muzammil Ahmad, Ansari Mohd. Azam, and Tabrez Khan. Real time product analysis using data mining. *International Journal of Advanced Research in Computer Engineering Technology*, Mar 2015.

[3] Shakra Mehak, Rabia Zafar, Sharaz Aslam, and Sohail Masood Bhatti. Exploiting filtering approach with web scrapping for smart online shopping penny wise a wise tool for online shopping. In *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–5. IEEE, 2019.

[4] Kali Pradeep, I Bhagyasri, and P Praneetha. E-commerce with backbone of data mining. *International Journal of Engineering and Technical Research*, 2:460, 10 2018.

[5] Riya Shah, Karishma Pathan, Anand Masurkar, and Shweta Rewatkar. Comparison of e-commerce products using web mining.

[6] Chunmei Zheng, Guomei He, and Zuojie Peng. A study of web information extraction technology based on beautiful soup. *JCP*, 10(6):381–387, 2015.

[7] Andrea Horch, Holger Kett, and Anette Weisbecker. Mining e-commerce data from e-shop websites. In *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 2, pages 153–160. IEEE, 2015.

[8] Ying Ming-Hsiung and Hsu Yeh-Yen. A commodity search system for online shopping based on ontology and web mining. In *IET Conference Proceedings*. The Institution of Engineering & Technology, 2014.

[9] Ahmad Tasnim Siddiqui and Sultan Aljahdali. Web mining techniques in e-commerce applications. *arXiv preprint arXiv:1311.7388*, 2013.

[10] Dwijen Rudrapal, Amitava Das, and Baby Bhattacharya. Measuring semantic similarity for bengali tweets using wordnet. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 537–544, 2015.

[11] Fankar Armash Aslam, Hawa Nabeel Mohammed, Jummal Musab Mohd, Murade Aaraf Gulamgaus, and PS Lok. Efficient way of web development using python and flask. *International Journal of Advanced Research in Computer Science*, 6(2), 2015.

[12] Eloisa Vargiu and Mirko Urru. Exploiting web scraping in a collaborative filtering-based approach to web advertising. *Artif. Intell. Research*, 2(1):44–54, 2013.

[13] Amna Shifia Nisafani, Rully Agus Hendrawan, and Arif Wibisono. Eliciting data from website using scrapy: An example. *SEMNASTE-KNOMEDIA ONLINE*, 5(1):2–1, 2017.

[14] Joydeep Bhattacharjee. Working with data. In *Practical Machine Learning with Rust*, pages 141–186. Springer, 2020.

[15] N. UzZaman and M. Khan. A double metaphone encoding for bangla and its application in spelling checker. In *2005 International Conference on Natural Language Processing and Knowledge Engineering*, pages 705–710, Oct 2005.

[16] Joeran Beel, Stefan Langer, Marcel Genzmehr, Bela Gipp, Corinna Breitinger, and Andreas Nürnberger. Research paper recommender system evaluation: a quantitative literature survey. In *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*, pages 15–22, 2013.

[17] Jun Ye. Cosine similarity measures for intuitionistic fuzzy sets and their applications. *Mathematical and computer modelling*, 53(1-2):91–97, 2011.

[18] Tiago Pessoa, Raul Medeiros, Thiago Nepomuceno, Gui-Bin Bian, V.H.C. Albuquerque, and Pedro Pedrosa Filho. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, PP:1–1, 10 2018.

[19] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.