# A Novel Method for Word Segmentation and Spell Correction in e-Commerce Search Engines

Melis Öztürk Umut, Muhammed Bera Kaya and Mustafa Keskin
*Hepsiburada, Turkey*

Keywords: Word Segmentation, Spell Correction, e-Commerce, Natural Language Processing, Search Engines, Information Retrieval.

Abstract: E-commerce search engines face a common problem where users write multi-word queries as a single, concatenated word, such as ”blackshoe” instead of ”black shoe.” This issue complicates search algorithms, leading to poor user experience and lower conversion rates. Our observations from historical search data of an e-commerce platform confirm that these incorrectly concatenated terms are a significant challenge, indicating a need for improved detection and correction methods. This study aims to develop a novel method to accurately segment and correct these terms. Our approach is based on dictionary and statistical algorithms, using a custom-built dictionary and edit distance-based structures to quickly match and correct erroneous or concatenated words. The algorithm's parameters, including search frequency thresholds, maximum edit distance, and prefix length, were extensively tested with different combinations to find the optimal settings for both spell correction and word segmentation. While this method was specifically designed for a particular e-commerce application's dataset, it proposes a generalizable approach for other e-commerce platforms. The paper details the dataset preparation, the proposed methodology, and the performance metrics obtained.

## 1 INTRODUCTION

Search engines are a fundamental feature of e-commerce platforms. The ability for users to quickly and accurately find the products they are looking for directly impacts both customer satisfaction and the platform's success. A common problem encountered in search engines is the concatenation of words that should be written separately. For example, a user might search for ”blackshoe” instead of ”black shoe.” This frequent occurrence makes it difficult for search algorithms to produce accurate results and for users to access the products they need. Fast typing habits and the auto-complete features of mobile devices are contributing factors to these incorrectly merged terms. Observations from historical search data of an e-commerce application show that incorrectly concatenated terms represent a significant portion of search queries and that the existing system is open to improvement in detecting this situation. Therefore, this study aims to develop a new method to correctly separate concatenated words and correct spelling errors. This research is based on dictionary and statistical algorithms. The proposed algorithm can quickly match and correct erroneous or concatenated words using an edit distance-based dictionary structure. These features make it an attractive option for the e-commerce domain, where real-time performance is crucial. The variables used in the algorithm were tested with different combinations of values to determine the optimal settings for both spell correction and word segmentation. Although the study was specifically designed for the dataset of the application from which historical search data was obtained, it proposes a generalizable method for other e-commerce platforms. The following sections of the paper will detail the dataset preparation, the methodology, and the performance metrics obtained.

## 2 LITERATURE REVIEW

Spelling errors in e-commerce platforms have a direct impact on finding the desired product and on user experience. In studies for spell correction and word segmentation, dictionary-based methods, language model approaches, and hybrid models are used. In dictionary-based methods, metrics like Levenshtein distance and Jaccard similarity are used to correct spelling mistakes (Garbe, 2021). SymSpell is fre-



Öztürk Umut, M., Kaya, M. B. and Keskin, M.
A Novel Method for Word Segmentation and Spell Correction in e-Commerce Search Engines.
DOI: 10.5220/0014286500004848

In *Proceedings of the 2nd International Conference on Advances in Electrical, Electronics, Energy, and Computer Sciences (ICEEECS 2025)*, pages 14-19
ISBN: 978-989-758-783-2

quently preferred in e-commerce applications because of its low response times (Garbe, 2019). Wang and Zhang (2021) state in their study that while SymSpell has high real-time performance, its lack of contextual awareness can lead to incorrect corrections (Wang and Zhang, 2021). These shortcomings also create difficulties in separating concatenated words. Therefore, it has been suggested that dictionary-based methods alone are not sufficient and should be used in conjunction with other models. In Language Model (LM) based approaches, models such as BERT, DistilBERT, and T5 are used to analyze the context of a word and correct errors more accurately (Dutta and Pande, 2024a). Dutta et al. (2024) have shown that BART and T5 models increase the F1 score in spell checking by 4% (Dutta and Pande, 2024a). However, LM-based models have high computational costs and response times, making them difficult to use in real-time systems. For this reason, more optimized models like DistilBERT are used (Kakkar and Pande, 2023). SymSpell offers a speed advantage but does not consider contextual meaning, so hybrid systems that combine it with Transformer-based models are recommended. Guo et al. (2024) have shown that a method where candidate words from SymSpell are ranked by a language model increases accuracy (Guo and et al., 2024). Similarly, Dutta et al. (2024) increased the accuracy rate by using a re-ranking method specifically for separating concatenated words in e-commerce searches (Dutta and Pande, 2024b). In Turkish, separating concatenated words presents additional difficulties because the language is agglutinative and morphologically rich. Incorrect segmentation can lead to meaning loss and erroneous suggestions. In the context of e-commerce, examples of brands, products, and models that should not be segmented make the process particularly challenging. These difficulties have been addressed using language models and rule-based systems (Uzun, 2022). Phonetic analysis and multi-approach models have the potential to improve the process of separating concatenated words (Behrooznia et al., 2024). E-commerce platforms face various challenges in using spell checkers and word segmenters. The reasons for these challenges are brand names, product names, model names, and users' use of natural language (Pande, 2022). Developing customized models is crucial for improving the search experience and increasing conversion rates.

# 3 DATASET

During the dataset preparation phase, data obtained from various tables within an e-commerce platform were used. This data includes the search terms users have entered into the search engine and the frequency of these terms. The search terms and their search frequencies were determined using a dataset from the last 6 months. From this data, terms that had been searched for at least 10 times in the last 6 months were taken, and meaningless terms were cleaned. A dataset containing approximately 10 million search terms and their frequencies was created. This dataset is specific to the e-commerce platform where the study was conducted and is not publicly accessible.

## 3.1 Data Cleaning

Search terms with a frequency below 10 were removed from the dataset. Symbols, emojis, and extra spaces within the search terms were removed, and characters were converted to lowercase. Terms consisting only of symbols or numbers, and terms containing no characters, were removed from the dataset. The words that make up each search term were sorted alphabetically. As a result of this sorting, search terms with the same sorted order but different search frequencies, such as "blue tshirt" and "tshirt blue," had the low-frequency one removed.

## 3.2 Dictionary Creation

The dictionary used in dictionary-based algorithms is of great importance. To create the dictionary, the search terms were separated into their unigrams and bigrams. After creating unigrams and bigrams for each search term, the search frequency of the term was divided by the number of words in the term. The resulting value corresponds to the search frequency of the unigram in the term. If the same unigram appears multiple times in the dataset, the values from each search term are summed to get a final value. For example, if "black frame" was searched 100 times and "brown frame" was searched 70 times, the calculated frequency for "frame" would be 85. The same process was repeated for bigrams. After obtaining unigrams and bigrams, three different dictionaries were created. The first dictionary, dictionary_wa, was created from search terms and their frequencies without separating them into unigrams and bigrams. The second dictionary, dictionary_ou, was created from only unigrams. The third and final dictionary, dictionary_ub, was created from both unigrams and bigrams.

# 4 METHODOLOGY

In this study, the success of Compound Search and Word Segmentation methods in separating concatenated words in search terms was examined. To obtain the best-performing result, tests were conducted with various combinations of variables. The method used to separate concatenated words was also expected to be successful in spell correction. Subsequently, as seen in Figure 1, the method that gave the best result was tested for how well it could separate concatenated words.

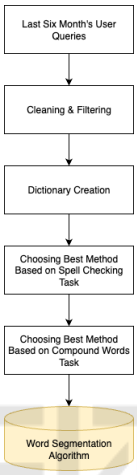


Figure 1: Overview of the compound word segmentation pipeline. The pipeline begins with the collection of the last six months' user-generated search queries. These queries undergo preprocessing steps, including cleaning meaningless words and filtering less searched queries, to ensure data quality. First, the performance of various parameters is evaluated on the spelling correction task, and subsequently, the best-performing method is tested on the word segmentation task with different parameter settings.

## 4.1 Variables

To find the best-performing method, various variables were tested in different combinations. The variables used in this context are as follows:

- Search Frequency Threshold (SFT)
- Maximum Edit Distance (MED)
- Prefix Length (PL)
- Unigram Search Frequency Threshold (USFT)
- Used Dictionary (UD)

The Search Frequency Threshold variable was created to prevent search terms with a low search frequency, which are more likely to have spelling errors, from being included in the dictionary. The values tested for this variable were 200 and 250. The Maximum Edit Distance variable refers to the maximum edit distance between a search term and a suggested word (Garbe, 2021) (Garbe, 2019). The values tested for this variable were 3 and 4. The Prefix Length variable indicates the length of word prefixes used for spell checking (mammothb, 2024). The values tested for this variable were 7, 9, and 10. The Unigram Search Frequency Threshold variable is a threshold value used to prevent the creation of single-word dictionary elements that may be spelling errors or meaningless, among the unigrams created from the dataset. Higher values were tested for this variable compared to the Search Frequency Threshold. The values tested were 300 and 400. The Used Dictionary variable represents the three different dictionaries created using the methods described in the Dataset section. Dictionary_wa represents the dictionary created without separating search terms into unigrams and bigrams, dictionary_ou represents the dictionary created from only unigrams, and dictionary_ub represents the dictionary created from both unigrams and bigrams.

## 4.2 Statistical Methods

To find the best-performing method, two different methods for separating concatenated words were tested using the created variables.

### 4.2.1 Compound Search

In this study, an automatic correction algorithm was applied that can detect and correct spelling errors and word merging/splitting errors in multi-word phrases. The method used combines dictionary-based search, edit distance calculation, and statistical language model approaches. The algorithm's workflow consists of the following technical steps:

- Tokenization and Preprocessing: The input text is first separated into words. In this step, numerical expressions and abbreviations (all-uppercase terms) are detected and excluded from the correction process.
- Generation of Word Correction Candidates: For each word, possible corrections from the dictionary are determined using the Levenshtein edit distance. Candidates with the lowest edit distance and highest frequency are selected as potential corrections.
- Word Merging (Combination) Analysis: If a merged form of two consecutive words, e.g., "ap" + "ple" → "apple", exists in the dictionary, its edit distance and frequency are compared with the sum of the individually corrected forms. If the total error cost (edit distance + frequency-based

score) of the merged form is lower, the two words are corrected as a single word.

- Word Splitting Analysis: Words not found in the dictionary or with a high edit distance are split into two at all possible points. Correction candidates are generated for each part, and the total of the bigram frequency and edit distance created by these two words is evaluated. If the split form has a higher probability than the original word, the word is split into two.
- Scoring and Selection of Candidates: For each correction candidate, a score is calculated based on the edit distance and word/bigram frequency. This score includes both an accuracy (edit distance) and a language model probability (frequency) component. The candidate with the highest score is selected as the final correction.
- Merging Results and Output Generation: All correction decisions are combined to create the corrected version of the original phrase. If necessary, the letter case of the original text is preserved. The final output is presented as a list of suggested corrections.

This methodology holistically addresses both independent word-based spelling errors and errors caused by word merging and splitting using a statistical language model and an edit distance-based approach. Thus, complex spelling errors in multi-word phrases can be detected and corrected with high accuracy.

#### 4.2.2 Word Segmentation

For the text segmentation step, the algorithm developed by Garbe (2019) was used to separate user inputs written with missing spaces into meaningful words (Garbe, 2019). This method, unlike classic dynamic programming approaches, offers a non-recursive and linear time complexity (O(n)) structure. The algorithm progresses along the input text up to a certain maximum word length, evaluating possible splits at each position and selecting the highest probability segmentation using log-probability scores based on word frequencies. Existing spaces are also taken into account during the segmentation process to determine the most suitable split points. In this way, the method can perform word segmentation effectively and quickly, especially in noisy or space-less texts.

### 4.3 Evaluation

The method to be used for separating concatenated words was decided by comparing the performance of the variable combinations that gave the best results in spell checking. The test set used to compare performance metrics contains 2978 examples of test search terms and their correctly spelled forms. Precision (P), Recall (R), F1, and Accuracy (A) were used as performance metrics, and the results were calculated for each variable combination. To calculate the compared metrics, True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) predictions were found. The meanings of the terms TP, TN, FP, and FN in the context of the spell checking study are given in Table 1.

Table 1: Used Metrics and Their Meanings.

| Terms | Description |
|---|---|
| TP | A spelling mistake was actually present, and it was detected and corrected correctly. |
| TN | No spelling mistake was actually present, and it was detected and not corrected (correctly). |
| FN | A spelling mistake was actually present, but it was not detected and was corrected incorrectly. |
| FP | A spelling mistake was actually present, and it was detected and corrected incorrectly OR a spelling mistake was not actually present, but it was detected and corrected incorrectly. |

In terms of spell checking, various variables were evaluated for the mentioned method using P, R, F1, and A metrics. In Table 2, only the best results from the used variables are shared. The best-performing results were obtained with SFT=200, MED=3, PL=7, and USFT=300. As seen in the table, most of the best results came from dictionary_ub. The method that gave the best results in spell checking was then tested for how well it could separate concatenated terms. For this, 102 examples of search terms that should have been written separately but were written concatenated were identified, which were noticed due to various problematic cases and were frequently misspelled by users. In addition, 41 correctly written search terms (either separated or concatenated) were included. This way, a dataset of 143 terms was created, and this dataset was used on the method that gave the best results from the spell checking evaluation. In this way, it was tested how accurately the method would separate concatenated words. The results of this evaluation are shared in Table 3. When the metrics are examined, it is seen that the best performance was obtained when the Word Segmentation method was used.

On the method that gave the best result for separating concatenated words, distance_sum and log_prob_sum values were calculated. Of these values, distance_sum indicates the number of charac-

Table 2: Spell Checking Results.

| Variables | | | | | Metrics | | | |
|---|---|---|---|---|---|---|---|---|
| SFT | MED | PL | USFT | UD | P | R | F1 | A |
| 200.0 | 3 | 7 | 300 | dictionary_ub | 47.65% | 52.53% | 49.97% | 69.34% |
| 200.0 | 3 | 9 | 400 | dictionary_wa | 47.39% | 52.55% | 49.84% | 69.31% |
| 200.0 | 3 | 10 | 300 | dictionary_ub | 47.65% | 52.53% | 49.97% | 69.34% |
| 200.0 | 4 | 7 | 300 | dictionary_ub | 40.98% | 56.84% | 47.62% | 65.95% |
| 200.0 | 4 | 9 | 300 | dictionary_ub | 40.98% | 56.84% | 47.62% | 65.95% |
| 200.0 | 4 | 10 | 300 | dictionary_ub | 40.98% | 56.84% | 47.62% | 65.95% |
| 250.0 | 3 | 7 | 300 | dictionary_ub | 47.78% | 52.01% | 49.81% | 69.41% |
| 250.0 | 3 | 9 | 300 | dictionary_ub | 47.78% | 52.01% | 49.81% | 69.41% |
| 250.0 | 3 | 10 | 300 | dictionary_ub | 47.78% | 52.01% | 49.81% | 69.41% |
| 250.0 | 4 | 7 | 300 | dictionary_ub | 40.77% | 56.21% | 47.26% | 65.75% |
| 250.0 | 4 | 9 | 300 | dictionary_ub | 40.77% | 56.21% | 47.26% | 65.75% |
| 250.0 | 4 | 10 | 300 | dictionary_ub | 40.77% | 56.21% | 47.26% | 65.75% |

Table 3: Concatenated Word Separation Results.

| Method | P | R | F1 | A |
|---|---|---|---|---|
| Compound Search | 43.69% | 92.85% | 59.42% | 49.64% |
| Word Segmentation | 46.28% | 96.55% | 62.56% | 52.48% |

ters that differ between the function's input and its prediction, while log_prob_sum gives the sum of the logarithmic probabilities of the word formation. In the study, the best results were obtained with distance_sum = 2 and log_prob_sum = -15. The evaluations made according to these values are presented in Table 4. When the P, R, F1, and A values used during the comparison were examined on this new, acceptable test data, the results were observed to be higher. Performance results obtained using various threshold values are shared in Table 4. One of the outcomes of the study was that the dictionary-based method used was insufficient in capturing contextual meaning. In this approach, semantic details, as in language models, were not successfully captured. For example, the abbreviation for the word doctor, "dr," could not be corrected as expected by the dictionary-based approach. The same situation exists with the example of "e-book" for "electronic book."

Table 4: Concatenated Word Separation Results After Threshold Values.

| Variables | | Metrics | | | | |
|---|---|---|---|---|---|---|
| distance_sum | log_prob_sum | Count | P | R | F1 | A |
| 2 | -15 | 43 | 86.84% | 94.28% | 90.41% | 83.72% |
| 1 | -15 | 36 | 87.87% | 96.66% | 92.06% | 86.11% |
| 2 | -14 | 38 | 85.71% | 93.75% | 89.55% | 81.57% |
| 1 | -14 | 31 | 86.66% | 96.29% | 91.22% | 83.87% |
| 2 | -13 | 37 | 85.29% | 93.54% | 89.23% | 81.08% |
| 1 | -13 | 31 | 86.66% | 96.29% | 91.22% | 83.87% |
| 3 | -15 | 33 | 78.57% | 94.28% | 85.71% | 76.59% |
| 3 | -14 | 41 | 78.94% | 93.75% | 85.71% | 75.60% |
| 3 | -13 | 40 | 78.37% | 93.54% | 85.29% | 75.00% |
| 1 | -12 | 31 | 86.66% | 96.29% | 91.22% | 83.87% |
| 1 | -11 | 31 | 86.66% | 96.29% | 91.22% | 83.87% |
| 1 | -10 | 31 | 86.66% | 96.29% | 91.22% | 83.87% |
| 2 | -12 | 37 | 85.29% | 93.54% | 89.23% | 81.08% |
| 2 | -11 | 37 | 85.29% | 93.54% | 89.23% | 81.08% |
| 2 | -10 | 37 | 85.29% | 93.54% | 89.23% | 81.08% |
| 3 | -12 | 40 | 78.37% | 93.54% | 85.29% | 75.00% |
| 3 | -11 | 40 | 78.37% | 93.54% | 85.29% | 75.00% |
| 3 | -10 | 40 | 78.37% | 93.54% | 85.29% | 75.00% |

# 5 CONCLUSION

One of the problems encountered in search engines is concatenated words that should be written separately. This study investigated how this situation can be solved using dictionary-based algorithms and compared the results for an e-commerce platform using various metrics. To get the best result, a prediction was first made on a test set containing examples of concatenated words using the variable combination that gave the best results in spell checking. The method that gave the best results for separating concatenated words was Word Segmentation. When the output values of the best method were used as threshold values, it was observed that the performance increased, but the number of examples predicted was significantly reduced. In future work, the dictionary used can be enriched with details such as product names, brand names, and product descriptions. The search frequencies of the elements in the dictionary can be calculated using different methods. It is thought that higher performance can be achieved in this way.

# ACKNOWLEDGEMENTS

This project was made possible by the individual contributions of each member of the recommendation team within Hepsiburada technology group. Also, this project would not have been possible if the technology group management of Hepsiburada had not supported and encouraged the recommendation team in innovation.

# REFERENCES

Behrooznia, A., Bedir, H., and Uzun, O. (2024). Statistical methods for turkish compound word segmentation. *Journal of Computational Linguistics*, 30(1):99–120.

Dutta, S. and Pande, R. (2024a). Improving spelling correction in e-commerce search using bart and t5. *Proceedings of the IEEE Conference on NLP*, 45(2):342–357.

Dutta, S. and Pande, R. (2024b). Ranking-based spell correction using neural networks. *E-Commerce and AI Applications*, 18(5):289–303.

Garbe, W. (2019). Fast word segmentation for noisy text. Blog post.

Garbe, W. (2021). Symspell: Symmetric delete spelling correction algorithm. GitHub.

Guo, L. and et al. (2024). Hybrid spelling correction models combining symspell and deep learning. *Journal of Artificial Intelligence Research*, 61(4):219–235.

Kakkar, P. and Pande, R. (2023). Weak supervision for typo correction in high-traffic search engines. *ACM Transactions on Information Systems*, 39(1):77–92.

mammothb (2024). symspellpy: Python port of symspell. GitHub.

Pande, R. (2022). Custom typo correction models for e-commerce. *Proceedings of the International Conference on E-Commerce AI*, 34(1):97–115.

Uzun, O. (2022). Phonetic analysis of turkish compounds for improved segmentation. *International Journal of Language Processing*, 19(2):45–67.

Wang, Y. and Zhang, X. (2021). Real-time spelling correction using symspell for e-commerce search. *Journal of Information Retrieval*, 24(3):189–205.