applied sciences

MDPI

Article

Research on a Web System Data-Filling Method Based on Optical Character Recognition and Multi-Text Similarity

Hailu Su, Ruiqing Kang * and Yunli Fan

School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China; 14747759472@163.com (H.S.); fanyunli0928@163.com (Y.F.)
* Correspondence: krq@ustb.edu.cn

Abstract: In the development of web systems, data uploading is a relatively important function. The traditional method of uploading data is to manually fill out forms, but when the data to be uploaded mostly exist in the form of form images, and the form content contains a lot of similar field information and irrelevant edge information, using traditional methods is not only time-consuming and labor-intensive, but also prone to errors. This requires a technology that can automatically fill in complex form images. OCR is an optical character recognition technology that can convert images into digitized text data using computer vision methods. However, using this technology alone cannot complete the tasks of extracting relevant data and filling corresponding fields. To address this issue, this article proposes a method that combines OCR technology and Levenshtein multi-text similarity. This method can effectively solve the problem of data filling after parsing complex form images, and the application results of this method in web systems show that the filling accuracy for complex form images can reach over 90%.

Keywords: OCR; Levenshtein; web system; data filling

check for updates

Citation: Su, H.; Kang, R.; Fan, Y. Research on a Web System Data-Filling Method Based on Optical Character Recognition and Multi-Text Similarity. *Appl. Sci.* **2024**, *14*, 1034. https://doi.org/10.3390/app14031034

Academic Editor: José Salvador Sánchez Garreta

Received: 8 December 2023
Revised: 23 January 2024
Accepted: 24 January 2024
Published: 25 January 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The development of web systems refers to the use of various technologies and tools to create and build web-based applications or systems. These applications typically run in web browsers and communicate with servers through the HTTP protocol [1]. Nowadays, with the emergence of various new technologies and tools, as well as the iterative updates of various frameworks and libraries, the development of web systems has become more flexible and efficient.

The function of data uploading is a task requirement that is involved in web system development. In web systems, there are many ways to upload data, including form uploading, the remote file transfer protocol (FTP), remote interface upload (API upload), and so on [2]. Among them, uploading through forms is a common web upload method that uses form controls to collect corresponding data and send them to the server in the form of key value pairs to complete the upload operation [3]. There are also various ways to fill in data when collecting corresponding data in the control of the form. The most basic filling method is for users to manually fill in each field in the form sequentially, which is relatively inefficient and accurate [4]. In addition, another common filling method is to complete data filling through data communication between different systems. By accessing and reading data from the database, the corresponding fields are automatically filled in. This filling method significantly improves efficiency, but manual filling cannot be avoided when creating database tables in the previous system [5]. In recent years, deep learning technology has been widely applied in the field of OCR. Among them, convolutional neural networks (CNNs) have achieved great results in handwritten document retrieval. Through the word recognition method of Monte Carlo dropout CNNs, the recognition accuracy in the scenarios of querying by example and querying by string has reached a level

superior to existing methods [6]. In addition, an end-to-end trainable hybrid CNN-RNN architecture has been proposed to solve the problem of building powerful text recognition systems for Urdu and other cursive languages [7]. At the same time, combining CRNN, LSTM, and CTC to construct a set of methods has shown good results in searching and deciphering handwritten text, and can be used on relatively simple machines [8]. These achievements provide new ideas and methods for the development of OCR technology, making it more widely applied. In addition, handwritten form recognition technology has also been relatively complete, and there are many achievements in automatic data filling. Common application scenarios include handwritten case forms in the health and medical field [9], medical insurance reimbursement application forms [10], etc. Users can use handwritten form recognition technology to automatically fill the patient's handwritten content into the corresponding form fields, accelerating the medical service process; there are also exam answer sheets and student evaluation forms in the field of education and training [11]. Users can also use handwriting recognition technology to convert student handwritten content into machine-readable text and automatically fill in the corresponding form fields to improve the efficiency and accuracy of data entry. However, in the examples in the above-mentioned fields, the framework for data collection forms is often fixed, and the fields are also known in advance, which cannot be well filled in for multi-source data. In response to the reality that data content in certain industries often exists in the form of images and the data form frameworks are not consistent, this paper proposes a new data filling method. By combining advanced OCR technology and multiple text similarity algorithms, it can achieve automatic parsing and filling of complex form images from different frameworks in web systems, and the final data filling accuracy can reach over 90%.

2. Related Technologies and Research

2.1. OCR Recognition Technology

OCR refers to the process in which electronic devices examine printed characters on paper, determine their shape by detecting dark and bright patterns, and then translate the shape into computer text using character recognition methods [12]. The recognition process of OCR mainly includes several steps, as shown in Figure 1 [13].

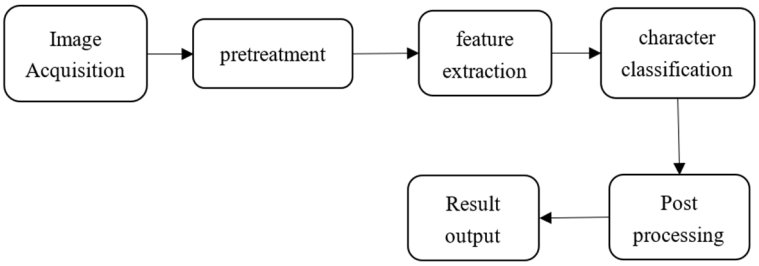

Figure 1. OCR recognition process diagram.

In some special cases, the obtained images may have problems such as angular tilt, unclear images, noise, or information loss [14], so before performing character recognition, it is necessary to pre-process the image to improve the accuracy of subsequent recognition. Common pre-processing operations include geometric transformation, image grayscale, binarization, denoising, etc. [15]. Image grayscale refers to transforming the original image from three channels to a single channel, converting the original color information into a single brightness information, in order to reduce the influence of irrelevant information in pixels. The weighted average method is the most commonly used grayscale method [16]. The following is the formula for the weighted average method:

$$Gray = 0.299R + 0.587G + 0.114B \quad (1)$$

R, G, and B represent the values of the three channels, and the weights are determined based on the sensitivity of the human eye to different colors. The calculated grayscale image will be more in line with the visual perception of the human eye [17]. After converting the image into a grayscale image, some threshold segmentation methods can be used for binarization processing, converting the grayscale image into a binary image with only black and white values. The purpose of doing so is also to further highlight the contours and edges of characters, facilitating subsequent character recognition [18].

After image pre-processing, feature extraction and character classification are performed. Features are the key information used to recognize text, and each different text can be distinguished from other texts through features [19]. Character classification is the process of passing the extracted features to the classifier, allowing the trained classifier to recognize the given features as corresponding text [20]. In recent years, most scholars in the field of computer vision have used the CRNN algorithm to solve this problem [21]. The network structure of the CRNN algorithm is shown in Figure 2 [22].

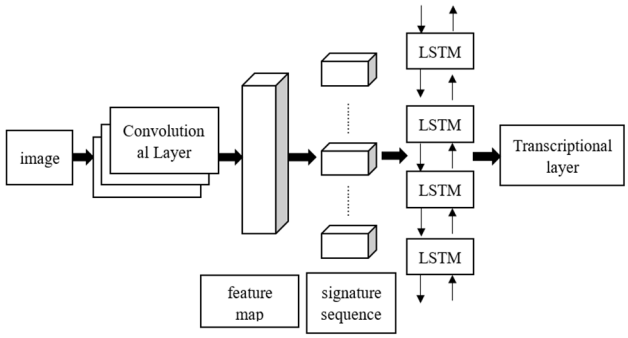

Figure 2. CRNN algorithm structure diagram.

The network structure consists of three parts: a convolutional layer, a recurrent layer, and a transcriptional layer [23]. The function of the convolutional layer is to extract features from the input image, and the extracted feature sequence is input into the loop layer. The loop layer can predict the labels of the feature sequence, and finally, through the transcription layer, integrate the results of the predicted feature sequence labels. By predicting each time step, the sequence label with the highest probability of occurrence is obtained, which is converted into the final recognition result [24]. For various parameters in the network structure, they generally need to be adjusted and optimized according to specific problems and datasets. For example, in Xinyu Fu's article "CRNN: A Joint Neural Network for Redundancy Detection" [25], he configured global training parameters to achieve better results, including setting the filters to 400, the hidden size to 400, the window size to 20, the pooling window to two, and the stride to one, as well as setting the learning rate to 0.01, training steps to 1000, and the optimizer to Adam. For the system in this article, OCR recognition is not the focus of the research, so the online recognition method of Baidu Zhiyun is adopted. Here, only a simple introduction to the network structure is provided, and specific parameter configurations are not required.

The final post-processing mainly involves further processing and optimizing the classification results to improve accuracy, eliminate errors, and provide more reliable and usable recognition results [26]. The specific processing can include text correction, semantic parsing, error correction, and so on.

2.2. Field Matching Technology

In practical work, it is common to have pre-defined form templates. For example, in common scenarios such as opening a bank account, insurance claims [27], and school exams, the same format of forms is often used to collect information. The field positions and sizes of these forms are usually fixed, so it is possible to define the form template in advance and directly match and recognize the fields based on the template. The specific field matching steps are shown in the Figure 3:

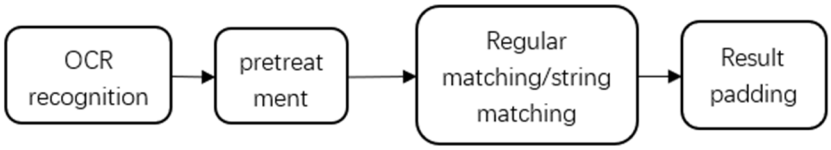

Figure 3. Field matching step diagram.

After experiencing previous OCR image recognition, a series of actionable text data will be obtained, which contains useful information required for the task, as well as a large number of irrelevant characters such as spaces and line breaks. Therefore, it is necessary to pre-process the text data first, including removing spaces, line breaks, and formatting specific formatted data. In the pre-processed data, the position information of relevant fields will also be retained, which refers to the relative position and layout rules of the fields in the form. This can help us to better understand the structure and context of the form. For tasks with pre-defined form templates, this can greatly simplify the difficulty of field matching. We can directly narrow the matching range based on the pre-defined field position, and for pre-defined forms, their corresponding fields are also very similar, so field matching can be performed directly on a small scale based on the field names defined in the form template, and filtering can be carried out using simple regular expressions or string matching [28]. However, this approach also has its limitations. For cases where the form framework is not fixed, the usefulness of field position information may not be as obvious, as the field positions may vary greatly across different forms. In addition, this method cannot accurately fill in some fields with different names but similar meanings. Therefore, in this article, we introduce a multiple similarity algorithm to compare from the perspectives of key and value in order to further improve the accuracy of field filling.

2.3. Levenshtein Editing Distance

The editing distance obtained using the Levenshtein algorithm is an indicator used to measure the degree of difference between two strings [29]. The definition of this distance is the minimum number of operations required to convert the source string to the target string, including insertion, deletion, and replacement. Using dynamic programming, each string is sequentially compared, and the algorithm has a time complexity of O (mn) and a spatial complexity of O (mn), where m and n represent the lengths of the source string (S) and the target string (T), respectively [30]. The calculation formula for the editing distance, D (S, T), is as follows:

$$D_{ij} = \begin{cases} 0, & i = 0 \; and \; j = 0 \\ min \begin{cases} D_{i-1,j-1} + W_a \\ D_{i-1,j} + W_b \\ D_{i,j-1} + W_c \end{cases} & , \; otherwise \end{cases} \quad (2)$$

In the above equation, $D_{ij} = D(S_0 \ldots S_i, T_0 \ldots T_j)$, $0 \leq i \leq m$, and $0 \leq j \leq n$, where $S_0 \ldots S_i$ is the source string and $T_0 \ldots T_j$ is the target string. The values W_a ,W_b ,W_c in the equation represent three different operations (delete, insert, and replace), each corresponding to an operand. In general experimental studies, researchers set the number of deletion and replacement operations to 1 and the number of replacement operations to 2 [31], so D_{ij} refers to the minimum number of edits from the source string (S) to the target string (T) in order to calculate the similarity between S and T.

2.4. Similarity Calculation Method

After obtaining the edited distance result, the similarity calculation between the two strings can be performed. The traditional similarity calculation formula is as follows [32]:

$$sim = 1 - \frac{ld}{m+n} \tag{3}$$

ld represents the editing distance between two strings, and m and n are the lengths of the two strings. However, this formula cannot handle the problem of inverted strings, and this calculation formula does not consider the existence of common substrings, which does not have universal applicability. Scholars have proposed the following similarity calculation formula [33]:

$$sim(S,T) = \frac{lcs}{lcs + ld + \frac{p}{lm+p}} \tag{4}$$

In the above equation, S is the source string, T is the target string, lcs is the length of the longest common substring, lm is the length of the S string, and p is the starting position of the common substring. After introducing the longest common substring, the problem of inverted strings can be solved to some extent when calculating similarity, and more accurate judgments can be made when editing distances that are equal. And the introduction, p/(lm+p), is also aimed at making further distinctions when ld and lcs are equal; that is, the higher the starting position of the common substring, the greater the impact on similarity. At present, the most commonly used method for calculating the length of LCS is to use dynamic programming [34], and the specific formula is as follows:

$$dp(i,j) = \begin{cases} 0 & i = 0 \; or \; j = 0 \\ dp(i-1, j-1) + 1 & S[i-1] = T[j-1] \\ 0 & S[i-1] \neq T[j-1] \end{cases} \tag{5}$$

In the above equation, S and T represent two strings, and i and j represent the characters at the corresponding positions of the strings (S). If either of the two strings has a length of 0, it indicates that there is no common substring. Therefore, dp(0,j) = dp(i,0) = 0; if the rest is calculated according to the formula, the final result is the length of LCS we need.

3. Filling Method Based on OCR and Text Similarity

On the basis of the existing web system development framework, by integrating OCR technology and text similarity algorithms, as well as improving the relevant field similarity comparison process according to the actual functional needs of the system, we can achieve our proposed goal of automatically filling images containing data content into corresponding web pages for different data form frameworks. As shown in the Figure 4 below, it is the overall structure of the system's functions.

In the front-end construction of the system, the most commonly used method is to use the combination of Vue and Element to build the page [35]. Therefore, we choose the el-upload component in the Element component to implement the function of uploading image files. It can convert images into binary data and send it to the backend through the Axios plugin. The most commonly used construction method in the backend is to use the springboot framework [36]. When an image is sent to the backend springboot server through a post request, the backend will parse the binary data in the request body and convert the binary data into usable byte arrays according to the HTTP protocol. At this point, the image is successfully uploaded to the backend server for subsequent recognition processing.

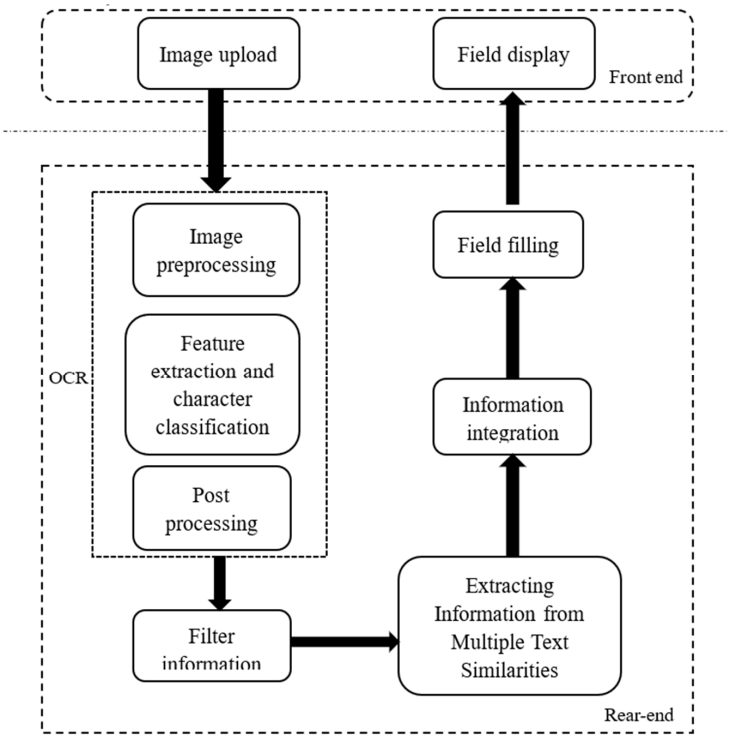

Figure 4. Functional structure diagram.

After the image is converted into a byte array in the backend, the next OCR recognition operation can be performed. In the OCR recognition process of this system, the relevant interfaces of Baidu Zhiyun are used [37]. The following Figure 5 shows the recognition process.

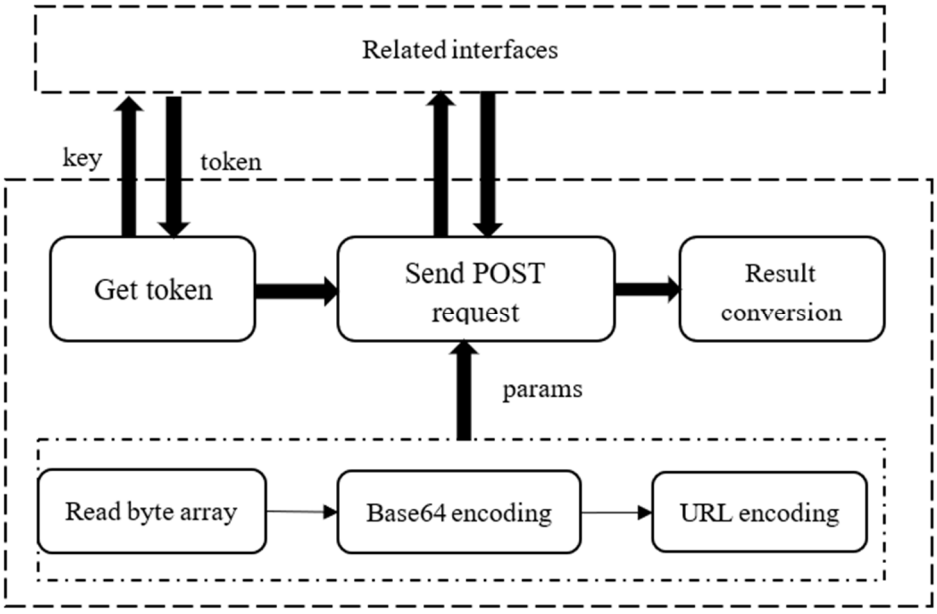

Figure 5. Identification process diagram.

After obtaining the byte array of the image, the first step is to convert it to a Base64 encoded string, as each character encoded with Base64 is an ASCII character, which can be easily transmitted in various communication protocols. Afterwards, the encoding method in Java.net was used to encode the string into a URL and concatenate it into a POST request parameter, completing the configuration of the params parameter. In order to ensure the

security and timeliness of use, we also need to obtain a valid token from the cloud platform before we can use the OCR function [38]. This requires us to first pass the API to the cloud_Key and Secret_Key. After both the token and params have been obtained, we can request the cloud again and return the desired recognition result. The intelligent cloud also undergoes pre-processing, feature extraction, character classification, post-processing, and other operations in this process [39], and it also utilizes the network structure of CRNN in the intermediate feature extraction and character recognition stages.

After obtaining the information in the image, it is necessary to filter it because not all the information is what we need. For this system, as it is filling in web forms, we only need the data related to form filling in the image. These types of data are observed to exist in the form of "name: content", so we can use them for segmentation. The following Figure 6 is a flowchart of a segmentation method.

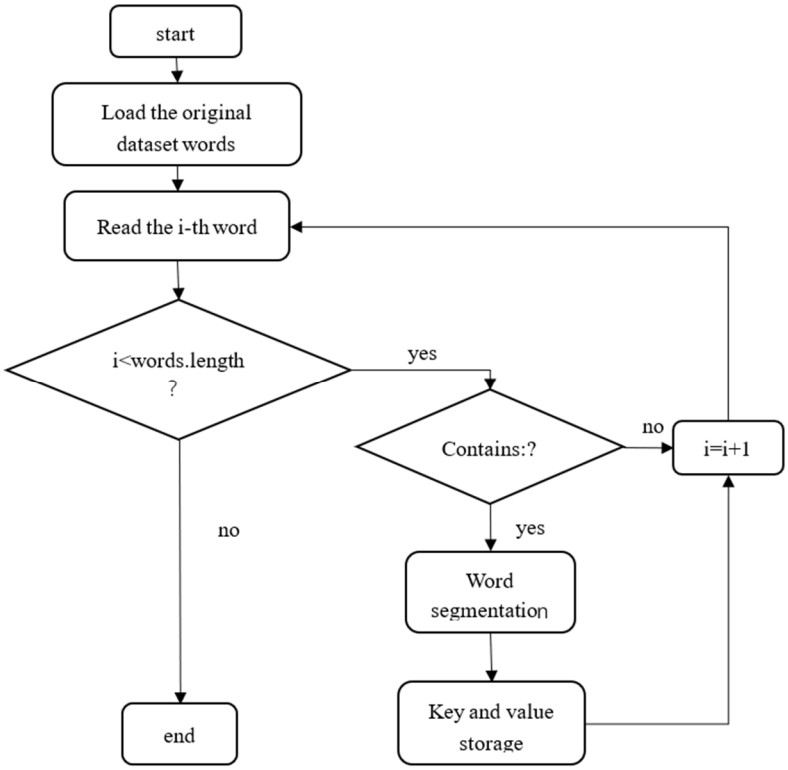

Figure 6. Segmentation process diagram.

We use a HashMap to store the "key" and "value" values for the filtered data. However, before conducting formal text similarity comparison, it is necessary to establish the data information of form fields. We also use a HashMap to store it, and store the key value as the field name. "Value" stores the corresponding field's data table number, which corresponds to the standard answer of previous fields. At this point, the image filtering data OCR will be obtained before entering the similarity comparison_Map and field data f_Map.

For the determination of similarity in Chinese short texts, methods have always been improved, from SOW/BOW statistical frequency [40] to n-gram sliding windows [41], from topic models [42] to deep learning [43]. The evolution of methods is to meet the similarity situation in different situations. However, for the field matching problem in this system, due to its own problems such as a short text word count, concise semantics, and small training data, it is not suitable to use deep learning for similarity judgment. Therefore, we have returned to the most naive way of judging similarity based on editing distance and common strings. In response to this task, we proposed the concept of importance, which divides fields into importance levels to ensure the accuracy of calculations, as shown in the following Figure 7.

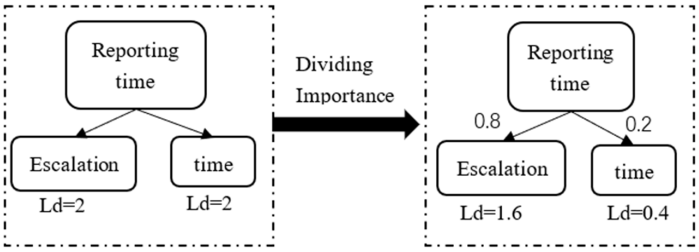

Figure 7. Importance division diagram.

In the above figure, it can be seen that if the demonstration field "reporting time" is calculated according to the usual editing distance formula, the distance between the "reporting" and "time" fields is equal, which does not meet the expected results. When we assign an importance of 0.8 to "time" and 0.2 to "reporting", recalculating the distance will make a difference. The result obtained is that "time" is more important, which means that when removing or adding variables with higher importance, the editing distance will also be larger. For the storage of importance, we also use a nested structure of HashMap, where the key stores the corresponding field and the value stores the importance-related content of the field.

After obtaining the relevant data above, you can enter the similarity comparison stage to extract the required information content from the front-end form. The specific process is shown in the following Figure 8.

In the figure, we obtained the result by double comparing the "key" and "value" values. The map is the final desired result, and after some packaging and integration operations, the data can be sent to the front-end for filling. The evaluation criteria for the two sims in the figure are also determined through multiple experiments. At this point, all image filling methods based on OCR and text similarity have been introduced clearly.

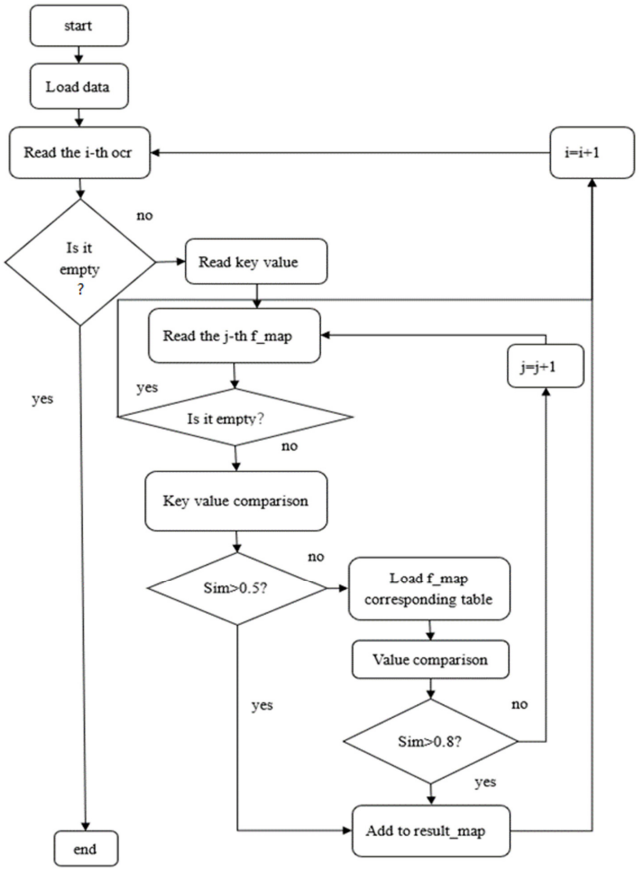

Figure 8. Similarity comparison chart.

4. Experimental Results and Analysis

To verify the effectiveness of the method proposed in this article, we conducted tests on 80 self-made images that roughly meet the upload requirements. Forty of the images were used to train the two required sim values for comparison in the system. The remaining forty images were used to test and determine the final adjusted system filling accuracy. Each image contains multiple similar fields and irrelevant edge information, which is more in line with the actual complex situation. The approximate content of the image is shown in Figure 9 below:

As shown in the above figure, the form contains multiple sets of similar information, such as "reporting location", "reporting department", "reporting name", "detection name", "detection method", etc. In addition, it also contains multiple sets of unrelated information, such as the form name, description, and warning signs. Due to the fact that the submitted form is designed by different departments in different regions, there may be some additions or deletions in the content of the submitted form. Some may have the same requirements, but their names may also be different. This requires our similarity algorithm to distinguish them.

The information content in self-made images includes useful information and irrelevant information. The useful information is further divided into similar field information and dissimilar field information. The following Figures 10 and 11 shows the distribution of the ratio of similar field information to useful information and the ratio of irrelevant information to overall information in the image.

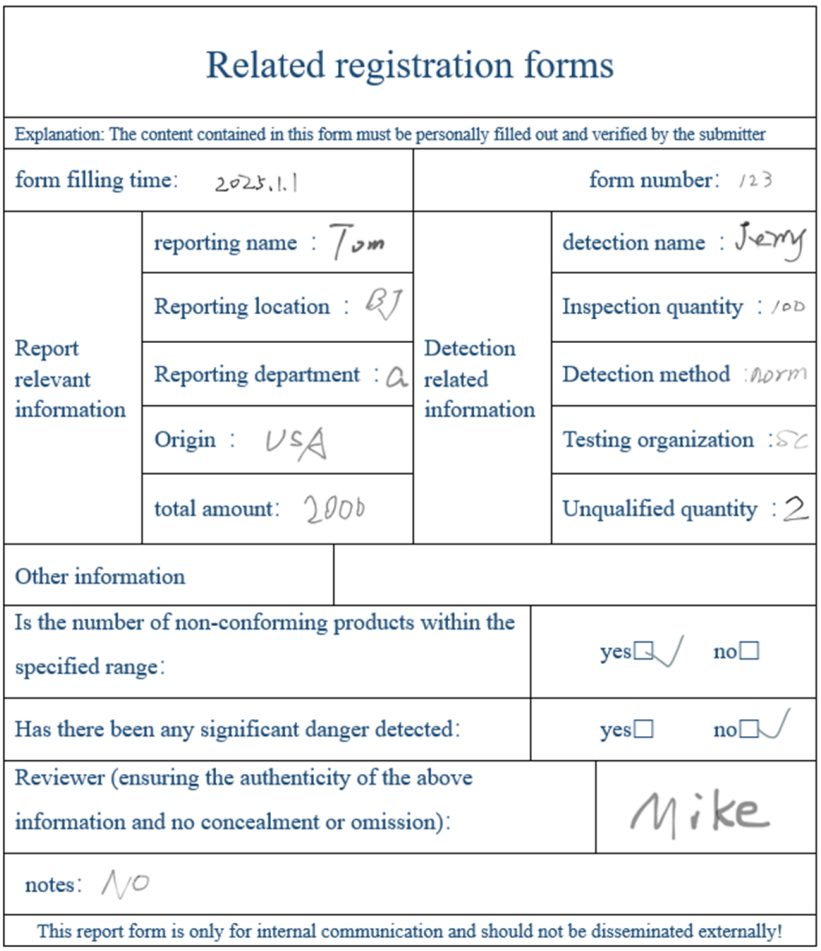

Figure 9. Example figure.

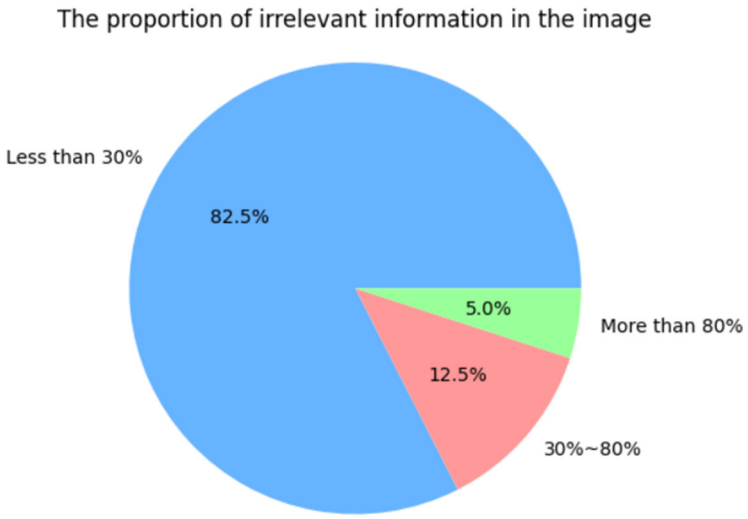

Figure 10. The proportion of irrelevant information in the image.

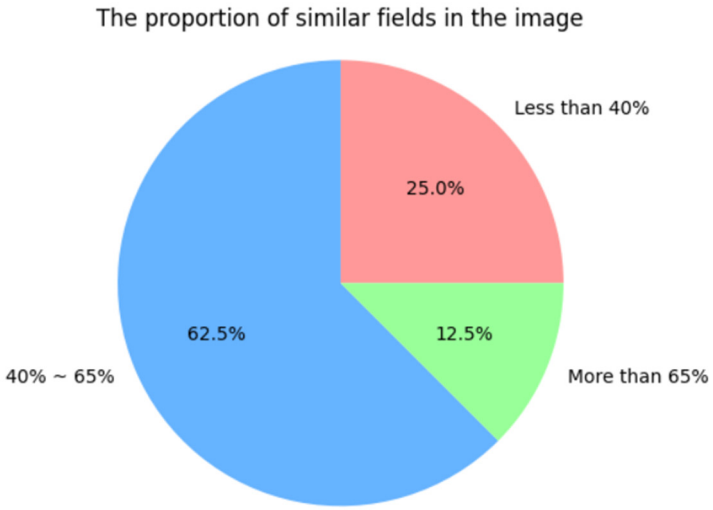

Figure 11. The proportion of similar fields in the image.

The evaluation criteria for sim mentioned above are the final results obtained through multiple experiments. For the evaluation indicators of results, we use the following accuracy formula [44]:

$$P = \frac{TP}{TP + FP} \tag{6}$$

For the accuracy of the final filling of an image, we divide the number of correctly filled fields (TP) by the number of fields it fills (TP + FP) [45]. The accuracy corresponding to a set sim standard is the average of all, imagesP_{avg}, as shown in the following formula:

$$P_{avg} = \frac{P_n + \ldots + P_1}{n} \tag{7}$$

We use the method of controlling variables to sequentially determine the two sims mentioned above. The results are shown in the following figure.

As shown in Figure 12, we can see that setting the first similarity judgment condition (sim1) to around 0.5 will result in the highest accuracy. When we fix the value of sim1 and change the value of sim2, it can be determined from Figure 13 that when the value of sim2 is around 0.8, the accuracy will reach its highest. Finally, we fixed the values of sim1 and sim2 and tested 40 images, resulting in the following figure:

As shown in the Figure 14, the filling accuracy of the vast majority of the 40 images tested reached over 90%. In actual work, staff can first upload relevant images for automatic field filling, and then manually check for content supplementation. This can greatly improve work efficiency and reduce the probability of manual errors. The low filling accuracy of individual images is due to the fact that most of the fields in the form of individual images are not filled in, which rarely happens in practical work. Even if it happens, it will be quickly screened out during manual inspection within the allowable error range. This result meets our expected content and also proves the usability of the method proposed in this paper.

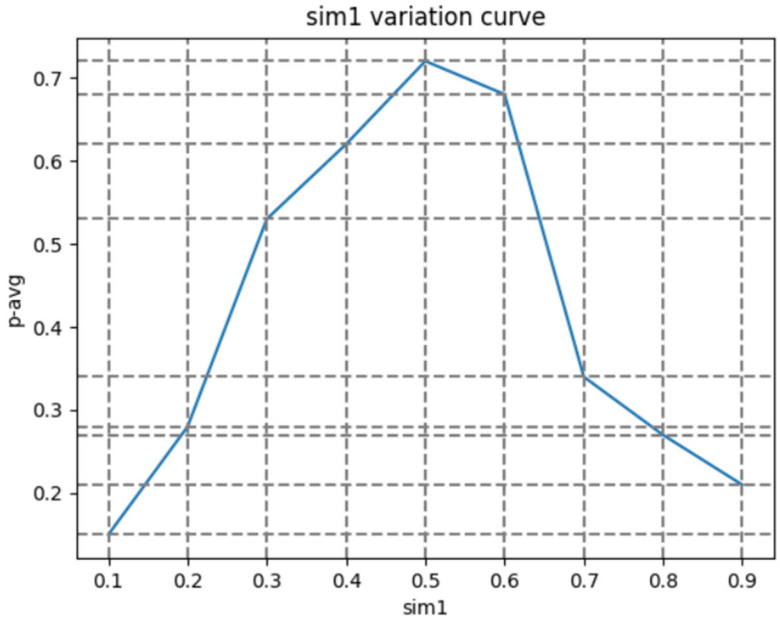

Figure 12. Sim1 variation curve.

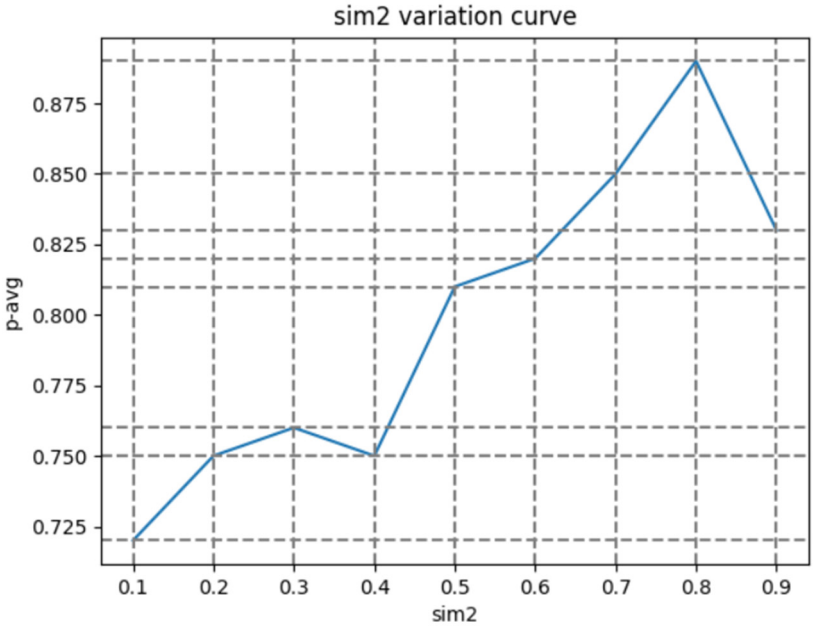

Figure 13. Sim2 variation curve.

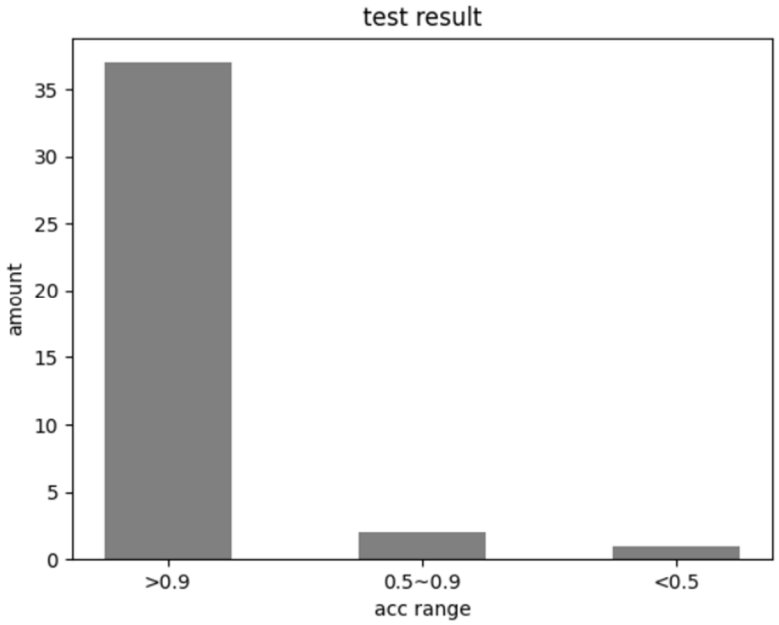

Figure 14. Test result.

5. Conclusions

This article focuses on the problem of data uploading and filling in web systems. Based on the existing advanced OCR technology and text similarity algorithms, combined and improved, the goal of filling fields in complex form images was effectively achieved. According to the test results, it was found that the accuracy of image recognition and filling in practical applications can reach over 90%. However, this method also has some limitations, as its proposal is based on practical engineering problems, so the field information considered is also related to this project. If fields need to be changed, they may need to be readjusted. In the future, further optimization can be based on this method, such as considering real-time updates of database tables corresponding to fields, in order to better adapt to different image forms.

Author Contributions: H.S., R.K. and Y.F. conceived and designed the system; H.S. implemented the entire system; H.S. and Y.F. debugged relevant parameters and drew data graphs; H.S. reviewed and edited the paper; R.K. revised the paper; Y.F. helped in writing the related works. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data provided in this study can be provided at the request of the corresponding author. Due to the fact that this article is based on an actual engineering project, and the data involved in the project needs to be kept confidential, the data has not been made public.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Molina-Ríos, J.; Pedreira-Souto, N. Comparison of development methodologies in web applications. *Inf. Softw. Technol.* **2020**, *119*, 106238. [CrossRef]
2. Xu, Y.; Cao, S. The Implementation of Large Video File Upload System Based on the HTML5 API and Ajax. In Proceedings of the 2015 Joint International Mechanical, Electronic and Information Technology Conference (JIMET-15), Chongqing, China, 18–20 December 2015; pp. 15–19.
3. Lestari, N.S.; Ramadi, G.D.; Mahardika, A.G. Web-Based Online Study Plan Card Application Design. *J. Phys. Conf. Ser.* **2021**, *1783*, 012046. [CrossRef]
4. Diaz, O.; Otaduy, I.; Puente, G. User-driven automation of web form filling. In Proceedings of the Web Engineering: 13th International Conference, ICWE 2013, Aalborg, Denmark, 8–12 July 2013; pp. 171–185.
5. Suryadi, A.; Balakrishnan, T.A. Website Based Patient Clinical Data Information Filling and Registration System. *Proc. Int. Conf. Nurs. Health Sci.* **2023**, *4*, 197–206. [CrossRef]

6. Daraee, F.; Mozaffari, S.; Razavi, S.M. Handwritten keyword spotting using deep neural networks and certainty prediction. *Comput. Electr. Eng.* **2021**, *92*, 107111. [CrossRef]
7. Jain, M.; Mathew, M.; Jawahar, C.V. Unconstrained OCR for Urdu Using Deep CNN-RNN Hybrid Networks. In Proceedings of the 2017 4th IAPR Asian Conference on Pattern Recognition (ACPR), Nanjing, China, 26–29 November 2017; pp. 747–752.
8. Semkovych, V.; Shymanskyi, V. Combining OCR Methods to Improve Handwritten Text Recognition with Low System Technical Requirements. In Proceedings of the The International Symposium on Computer Science, Digital Economy and Intelligent Systems, Wuhan, China, 11–13 November 2022; pp. 693–702.
9. Shaw, U.; Mamgai, R.; Malhotra, I. Medical Handwritten Prescription Recognition and Information Retrieval Using Neural Network. In Proceedings of the 2021 6th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 7–9 October 2021; pp. 46–50.
10. Aluga, D.; Nnyanzi, L.A.; King, N.; Okolie, E.A.; Raby, P. Effect of electronic prescribing compared to paper-based (handwritten) prescribing on primary medication adherence in an outpatient setting: A systematic review. *Appl. Clin. Inform.* **2021**, *12*, 845–855. [CrossRef]
11. Sanuvala, G.; Fatima, S.S. A Study of Automated Evaluation of Student's Examination Paper Using Machine Learning Techniques. In Proceedings of the 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 19–20 February 2021; pp. 1049–1054.
12. Thorat, C.; Bhat, A.; Sawant, P.; Bartakke, I.; Shirsath, S. A detailed review on text extraction using optical character recognition. *ICT Anal. Appl.* **2022**, *314*, 719–728.
13. Karthick, K.; Ravindrakumar, K.; Francis, R.; Ilankannan, S. Steps involved in text recognition and recent research in OCR; a study. *Int. J. Recent Technol. Eng.* **2019**, *8*, 2277–3878.
14. Kshetry, R.L. Image preprocessing and modified adaptive thresholding for improving OCR. *arXiv* **2021**, arXiv:2111.14075. [CrossRef]
15. Mursari, L.R.; Wibowo, A. The effectiveness of image preprocessing on digital handwritten scripts recognition with the implementation of OCR Tesseract. *Comput. Eng. Appl. J.* **2021**, *10*, 177–186. [CrossRef]
16. Ma, T.; Yue, M.; Yuan, C.; Yuan, H. File text recognition and management system based on tesseract-OCR. In Proceedings of the 2021 3rd International Conference on Applied Machine Learning (ICAML), Changsha, China, 23–25 July 2021; pp. 236–239.
17. Kamisetty, V.N.S.R.; Chidvilas, B.S.; Revathy, S.; Jeyanthi, P.; Anu, V.M.; Gladence, L.M. Digitization of Data from Invoice Using OCR. In Proceedings of the 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 29–31 March 2022; pp. 1–10.
18. Maliński, K.; Okarma, K. Analysis of Image Preprocessing and Binarization Methods for OCR-Based Detection and Classification of Electronic Integrated Circuit Labeling. *Electronics* **2023**, *12*, 2449. [CrossRef]
19. Nahar, K.M.; Alsmadi, I.; Al Mamlook, R.E.; Nasayreh, A.; Gharaibeh, H.; Almuflih, A.S.; Alasim, F. Recognition of Arabic Air-Written Letters: Machine Learning, Convolutional Neural Networks, and Optical Character Recognition (OCR) Techniques. *Sensors* **2023**, *23*, 9475. [CrossRef]
20. Yu, W.; Lu, N.; Qi, X.; Gong, P.; Xiao, R. PICK: Processing key information extraction from documents using improved graph learning-convolutional networks. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 4363–4370.
21. Biró, A.; Cuesta-Vargas, A.I.; Martín-Martín, J.; Szilágyi, L.; Szilágyi, S.M. Synthetized Multilanguage OCR Using CRNN and SVTR Models for Realtime Collaborative Tools. *Appl. Sci.* **2023**, *13*, 4419. [CrossRef]
22. He, Y. Research on Text Detection and Recognition Based on OCR Recognition Technology. In Proceedings of the 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, China, 27–29 September 2020; pp. 132–140.
23. Verma, P.; Foomani, G. Improvement in OCR Technologies in Postal Industry Using CNN-RNN Architecture: Literature Review. *Int. J. Mach. Learn. Comput.* **2022**, *12*, 154–163.
24. Idris, A.A.; Taha, D.B. Handwritten Text Recognition Using CRNN. In Proceedings of the 2022 8th International Conference on Contemporary Information Technology and Mathematics (ICCITM), Mosul, Iraq, 31 August–1 September 2022; pp. 329–334.
25. Fu, X.; Ch'ng, E.; Aickelin, U.; See, S. CRNN: A joint neural network for redundancy detection. In Proceedings of the 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, China, 29–31 May 2017; pp. 1–8.
26. Nguyen, T.T.H.; Jatowt, A.; Coustaty, M.; Doucet, A. Survey of post-OCR processing approaches. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–37. [CrossRef]
27. Kumar, P.; Revathy, S. An Automated Invoice Handling Method Using OCR. In Proceedings of the Data Intelligence and Cognitive Informatics: Proceedings of ICDICI 2020, Tirunelveli, India, 8–9 July 2020; pp. 243–254.
28. Jiju, A.; Tuscano, S.; Badgujar, C. OCR text extraction. *Int. J. Eng. Manag. Res.* **2021**, *11*, 83–86. [CrossRef]
29. Reid, M.; Zhong, V. LEWIS: Levenshtein editing for unsupervised text style transfer. *arXiv* **2021**, arXiv:2105.08206.
30. Da, C.; Wang, P.; Yao, C. Levenshtein OCR. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23 October 2022; pp. 322–338.
31. Rustamovna, A.U. Understanding the levenshtein distance equation for beginners. *Am. J. Eng. Technol.* **2021**, *3*, 134–139.

32. Wang, J.; Xu, W.; Yan, W.; Li, C. Text Similarity Calculation Method Based on Hybrid Model of LDA and TF-IDF. In Proceedings of the 2019 3rd International Conference on Computer Science and Artificial Intelligence, Normal, IL, USA, 6–8 December 2019; pp. 1–8.
33. Zang, R.; Sun, H.; Yang, F.; Feng, G.; Yin, L. Text similarity calculation method based on Levenshtein and TFRSF. *Comput. Mod.* **2018**, *4*, 84–89.
34. Amir, A.; Charalampopoulos, P.; Pissis, S.P.; Radoszewski, J. Dynamic and internal longest common substring. *Algorithmica* **2020**, *82*, 3707–3743. [CrossRef]
35. Irhansyah, T.; Nasution, M.I.P. Development Of Thesis Repository Application In The Faculty Of Science And Technology Use Implementation Of Vue. Js Framework. *J. Inf. Syst. Technol. Res.* **2023**, *2*, 66–77.
36. Zhang, F.; Sun, G.; Zheng, B.; Dong, L. Design and implementation of energy management system based on spring boot framework. *Information* **2021**, *12*, 457. [CrossRef]
37. Jiang, Y.; Dong, H.; El Saddik, A. Baidu Meizu deep learning competition: Arithmetic operation recognition using end-to-end learning OCR technologies. *IEEE Access* **2018**, *6*, 60128–60136. [CrossRef]
38. Fang, H.; Bao, M. Raw material form recognition based on Tesseract-OCR. In Proceedings of the 2021 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS), Shenyang, China, 10–11 December 2021; pp. 942–945.
39. Xu, Y.; Dai, P.; Li, Z.; Wang, H.; Cao, X. The Best Protection is Attack: Fooling Scene Text Recognition With Minimal Pixels. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1580–1595. [CrossRef]
40. Terra, E.L.; Clarke, C.L. Frequency Estimates for Statistical Word Similarity Measures. In Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, Edmonton, Canada, 27 May–1 June 2003; pp. 244–251.
41. Khreisat, L. A machine learning approach for Arabic text classification using N-gram frequency statistics. *J. Informetr.* **2009**, *3*, 72–77. [CrossRef]
42. Shao, M.; Qin, L. Text Similarity Computing Based on LDA Topic Model and Word Co-Occurrence. In Proceedings of the 2014 2nd International Conference on Software Engineering, Knowledge Engineering and Information Engineering (SEKEIE 2014), Singapore, 5–6 August 2014; pp. 199–203.
43. Li, Z.; Chen, H.; Chen, H. Biomedical text similarity evaluation using attention mechanism and Siamese neural network. *IEEE Access* **2021**, *9*, 105002–105011. [CrossRef]
44. Wen, X.; Jaxa-Rozen, M.; Trutnevyte, E. Accuracy indicators for evaluating retrospective performance of energy system models. *Appl. Energy* **2022**, *325*, 119906. [CrossRef]
45. Ji, M.; Zhang, X. A short text similarity calculation method combining semantic and headword attention mechanism. *Sci. Program.* **2022**, *2022*, 8252492. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.