Novel Ontologies-based Optical Character Recognition-error Correction Cooperating with Graph Component Extraction

Sarunya Kanjanawattana
Graduate School, Shibaura Institute of Technology, Tokyo, Japan
nb14503@shibaura-it.ac.jp

Masaomi Kimura
Information Science and Engineering, Shibaura Institute of Technology, Tokyo, Japan
masaomi@sic.shibaura-it.ac.jp

Abstract

A graph is a data representation visually presenting significant information in the academic literature. Extracting graph information clearly contributes to readers, who are interested in graph information interpretation, because we can obtain significant information presenting in the graph. A typical tool used to transform image-based characters to computer editable characters is optical character recognition (OCR). Unfortunately, OCR cannot guarantee perfect results, because it is sensitive to noise and input quality. This becomes a serious problem because misrecognition provides misunderstanding information to readers and causes misleading communication. In this study, we present a novel method for OCR-error correction based on bar graphs using semantics, such as ontologies and dependency parsing. Moreover, we used a graph component extraction proposed in our previous study to omit irrelevant parts from graph components. It was applied to clean and prepare input data for this OCR-error correction. The main objectives of this paper are to extract significant information from the graph using OCR and to correct OCR errors using semantics. As a result, our method provided remarkable performance with the highest accuracies and F-measures. Moreover, we examined that our input data contained less of noise because of an efficiency of our graph component extraction. Based on the evidence, we conclude that our solution to the OCR problem achieves the objectives.

Keywords: OCR-error correction, post-processing, dependency parsing, ontology, graph-component extraction.

1. Introduction

Nowadays, with the advent of digital optical scanners, a digital document has been required on account of ease of use and search. Over several years, a great effort has been devoted to the study of image-based information extraction. Images, especially graph images, typically contain much expedient information. For example, authors usually use graphs to present their experimental results, including measurement data for clear explanations. Graphs graphically provide data summarization presenting essential information that is simply interpreted by acquiring small descriptive details. Thus, an automatic system extractable latent information from the graphs provides many contributions to society for disclosing explicit and implicit knowledge. To obtain a primary interpretation, initial graph components analyzed are axis descriptions (i.e., X- and Y-titles) and a legend. OCR is an approving solution used for acquiring them as a digital format of character letters.

OCR is extensively used in several applications, such as medical article citation database MEDLINE (Lasko & Hauser, 2000) and text extraction from image and video frames (Chen, Odobez & Bourlard, 2004). In regard to academics, countless documents have been converted from paper-based to digitized information using OCR. However, OCR cannot guarantee accurate outputs. Generally, a quality of OCR outputs is fairly decreased, if OCR inputs have various defects, e.g., poor printing quality, small image resolution, specific language requirement, and image noises. These are main causes of misrecognition that produce OCR errors. For example, a word “BED” may be incorrectly recognized as “8ED”. Regarding the OCR errors, there have been two types of word errors that may be found in our study, non-word and real-word errors (Tong & Evans, 1996).

A non-word error occurs when OCR recognizes a source-text as a string that invalidly corresponds to any vocabulary item in the dictionary. A real-word error occurs when OCR applies to the source-text and provides incorrect output strings which coincidently match to an item in the dictionary. For example, if OCR renders the source-text “Today is hot” as “Toolav is not”, then “Toolav” is a non-word error, and “not” is a real-word error. To mitigate these errors, there has been a great deal of study focusing on addressing them and proposing methods based on practical techniques, such as semantic utilization (Jobbins et al., 1996) and statistical similarity measurement (Nagata, 1998). To our knowledge, using the semantics is a reliable solution to alleviate the OCR errors, because it analyzes not only the words themselves but also the context of corresponding sentences. Our OCR-error correction method utilizes a concept of semantics, including ontologies and natural language processing (NLP) to identify and correct the errors.

However, OCR is unsuitable to directly apply to graph images, because there are irrelevant parts in the graphs, which do not necessitate for the primary interpretation, such as parts of bars and some numeric data. They may cause recognition noise (e.g., special characters and number); hence, they should be eliminated in advance by our graph component extraction for improving a quality of OCR results (Huang, Tan & Leow, 2005), (Kataria et al., 2008).

The input of this study is a collection of bar graphs which contains at least axis descriptions (i.e., X- and Y-titles), and optionally, a legend. We here highlight only the bar graphs because its characteristics are dominant and easy to comprehend by both human and machine. Moreover, we gather related contents of documents for creating our ontology, such as image captions and cited paragraphs.

In this study, we propose a novel method which is a combination of a graph component extraction and an OCR-error correction. Note that the graph component extraction has already been proposed in our previous study (Kanjanawattana & Kimura, 2016), which aims to separate irrelevant parts from graph components. We focus on only three basic components, i.e., an X-title, a Y-title, and a legend; thus we determine other parts of graphs as irrelevant parts that should be omitted beforehand to reduce noise by the graph component extraction. To improve the OCR results, we also present the method of OCR-error correction in this study which is a post-processing system to analyze the results and correct errors based on ontologies, NLP and edit distance. We designed and created our ontology supporting dependency parsing of English context, including word categories queried from DBpedia. The crucial objectives of this study are to extract information from graph components and to improve the quality of OCR results recognized from the extracted graph components. Our system considerably contributes benefit to society, particularly in regard to academics, by acquiring implicit knowledge in the graphs. Moreover, it can be adapted to many applications, such as image search engine.

The remainder of the paper is organized into six sections as follows: Section 2 reviews previous studies. We introduce our method in Section 3. Section 4 presents experiments and their results. In Section 5, we discuss the results and reveal significant findings discovered by this study. Section 6 concludes and suggests future work.

2. Related works

In this section, we review existing works related to this study. There are two minor sections as follows:

2.1 Image segmentation

Image segmentation is currently an active research area with several unsolvable problems. This technique can be used to capture and separate dominant objects from image backgrounds. Basically, it deals with many kinds of images, such as outdoor scenes (Alvarez, LeCun & Lopez, 2012) (Cheng et al., 2012) and medical images (Hiran & Doshi, 2013). In academics, a graph image used to summarize and analyze essential information is another target image for this active field. Bar graphs are our main target in this study. We attempted to separate the basic components to prepare the inputs of OCR-error correction. However, to achieve graph segmentation is difficult for traditional techniques (such as image processing), because positions of graph components, especially a legend, are unfixed. A dramatic study addressing this difficulty has been presented by (Kataria et al., 2008). They aimed to automatically extract elements (e.g., axis labels, legends, and data points) from within a two-dimensional graph and mitigate a problem of overlapping text and data points. They performed an image profiling to detect global features in order to identify coordinate axes. Moreover, they applied an extended K-median to isolate and detect the data points from a curve. However, they confronted a problem when trying to extract a legend. That can be solved by performing a connected component analysis to identify individual letters before applying OCR. The other interesting study is proposed in (Huang & Leow, 2005) whose main targets were to associate recognition results of textual and graphical information in scientific graphs. They individually recognized text and graphical regions of the graph images and then combined their results to achieve a full understanding. However, they encountered OCR errors that were solved by manual correction. Although these previous studies proposed effective methods to extract graph components, it did not identify types of individual components. In fact, each component carries essential information, but its role certainly differs. For example, the X- and Y-titles evince a relationship of the graph. The legend provides particular information regarding data described as data labels. Clearly, to identify the type to each component is surely important for graph interpretation. Our graph component extraction can achieve this obstacle. Moreover, we not only extracted graph components using the OCR technique but also tackled an OCR error problem by correcting errors based on our methods.

2.2 OCR-error correction

A great deal of effort has developed many approaches to correct the OCR errors over several years. Nagata (Nagata, 1998) emphasized his work to correct misrecognized characters using character shape similarity and statistical language model. He attempted to challenge to Japanese whose sentences did not include word delimiters (e.g., space). However, we noticed that this previous study cannot correct such items as acronyms and transliterated foreign words because they often show in English (such as ISO and SONY) that cannot recognize by OCR included by Japanese language package. It differs from our method because ours can correct words universally as long as they appear in the source document.

Semantic-based techniques (e.g., context-based analysis and ontology) are proper solutions addressing the OCR problem. Wick et al. (Wick, Ross & Learned-Miller, 2007) realized that conventional systems identified low-confidence outputs that were insufficient to correct misrecognition errors. They used topic models automatically detecting the semantic context of scanned documents and specified the word frequency to correct the errors. However, a limitation of topic models is high training time required, because users must classify documents to acquire their corresponding topics prior applying OCR. An interesting method related to correct OCR errors is also described in (Bassil & Alwani, 2012). They developed a context-based method based on Google’s online spelling suggestion to correct the OCR errors. They avoided using an offline dictionary because a huge volume of terms needed to gather in a source computer, which consumed a lot of resources. Google is a massive online database containing a large collection of word sequences. It is suitable to be a data source of correcting word suggestion. However, this technique is limited to use via online that need to concern about network availability and efficiency, e.g., speed and bandwidth.

Recent studies addressing the problem of OCR errors tend to use ontology and semantics. Jobbins et al. (Jobbins, Evett & Sherkat, 1996) developed a system of automatic semantic-relation identification between words in Roget’s Thesaurus. This knowledge source contains explicit links between words and related vocabulary items for each part of speech, unlike an ordinary dictionary. Their method depended on Relation algorithm that located semantic relations between words and calculated a relatedness score of each word. However, this technique possibly encountered a difficulty, if dealing with words in a sentence. They may obtain a real-word error in a same category or cross reference. To solve this problem, not only word categories but also sentence dependencies should be used, because each word in the sentence definitely contains at least one dependency linking to some other words in the same sentence. Zhuang et al. (Zhuang & Zhu, 2005) introduced an OCR post-processing method based on multiple forms of knowledge, for example, language knowledge and candidate distance information given by the OCR engine. They focused on Chinese characters. A similarity between this existing study and our study is to find candidates depended on similarity distances. However, this previous study was limited to long sentences containing many dependencies, because it used n-gram supportable contiguous sequence of n items from given sentences.

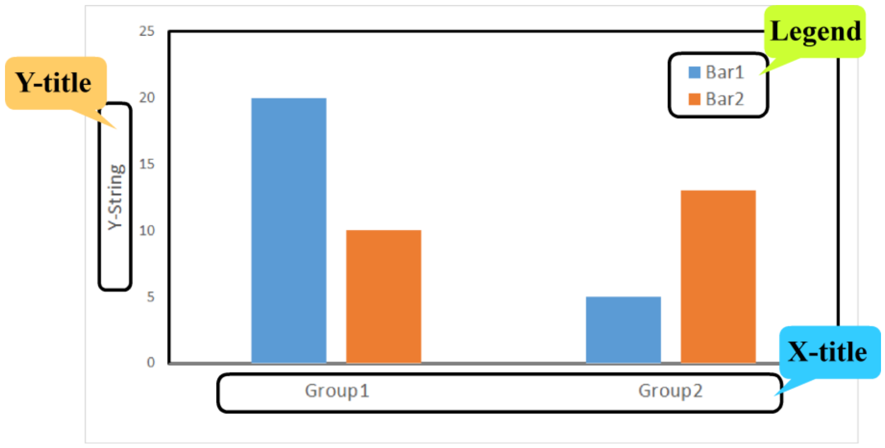


Figure 1. Illustration of graph components.

3. Methodology

In this study, we utilize a pre-processing and suggest a post-processing methods to achieve a difficulty of OCR errors. We introduce the new post-processing method of OCR-error correction for bar graphs utilizing several techniques, i.e., ontologies, NLP, and edit distance. Moreover, owing to reducing the presence of irrelevant parts, we used the pre-processing method of graph-component extraction proposed in (Kanjanawattana & Kimura, 2016) to detect and extract the basic graph components, such as a legend and axis descriptions, including omitting irrelevant parts.

We divide our methodology into two major modules as follows:

3.1 Graph-component extraction

The main task of this module is to separate the components (Figure 1) into individual images. As described in our previous study, the method was designed based on a generality of the graph structure. There were two distinct parts in this module: axis description extraction and legend extraction. For axis description extraction, we used a horizontal and vertical partitioning process to get X- and Y-titles respectively. Moreover, we utilized a pixel projection to investigate locations of peaks of each horizontal profile in order to omit irrelevant parts from the partitioned components. Note that the height of peak denotes how many pixels containing significant information exist. The location of the first peak was discarded to obtain a cleaned X-title because the first peak represents an internal part of a bar or data measurement. Likewise, we kept the first peak and omitted the rest to obtain a cleaned Y-title. Regarding legend extraction, there were five steps that used to extract the legend: data preprocessing, data transformation, clustering process, discrete Fourier transformation (DFT) process, and classification process. The main purpose of our previous study was to estimate a suitable Epsilon for DBSCAN to identify a legend position in graphs. Data preprocessing was used to eliminate unrelated areas, such as an axis title region and a left-and-bottom region. Data transformation was used to rescale the inputs to a smaller size using average subsampling, including transforming to the datasets of XY-coordinate corresponding to the locations of the existed pixels. Clustering process is the main part of our previous study. We used DBSCAN that typically requires two parameters: MinPts and Epsilon. To estimate a suitable Epsilon for each graph image, we utilized an idea of region density. In each quarter, we obtained an area with the highest density. Afterward, we investigated a candidate point of each area, which had the furthest distance from the center of the same area but had the smallest distance measured from a neighbor area. Finally, we calculated distances between selected candidate points and chose the shortest one; moreover, we divided the shortest distance by the image width to obtain the Epsilon. Then, we cropped the images scaled back to original sizes, corresponding to the clustering results. Next, the cropped images were applied by DFT that used to reveal image characteristics represented in the frequency domain. According to data construction for classification, the characteristics of each quarter of the DFT image provided similar information; thus, we selected only a single quarter as input for classification. Finally, classification process was utilized to classify images with a legend. We applied support vector machines (SVM) to the data obtained from DFT process using radial basis function (RBF) kernel. In the end of this module, we obtained cleaned graph components, i.e., axis descriptions and legends.

3.2 OCR-error correction

This module is a core of this paper. On the subject of OCR-error correction, we utilize ontologies to solve OCR problems. Moreover, we integrate an edit distance and NLP to our correction system, because we realize how useful of sentence's context to predict unknown or misspelling vocabulary items based on a word suggestion from the edit distance. We create our ontology supporting results of parsed sentences, i.e., part of speech (POS) tags and sentence dependencies, as well as Named-entity recognition (NER) queried from DBpedia.

For this module, we divide procedures into three steps.

3.2.1 Candidate selection

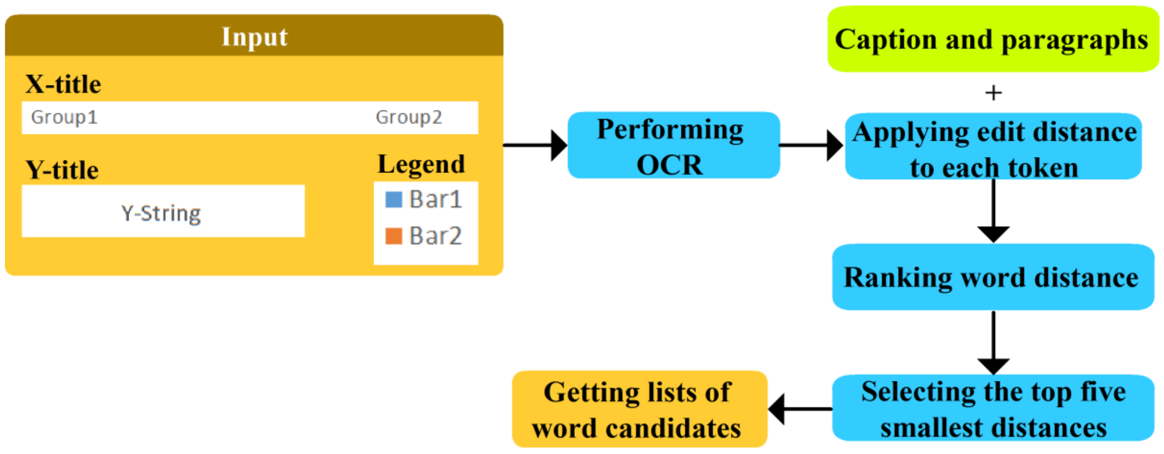


Figure 2. Steps of candidate selection.

The input of this module is the cleaned graph components acquired from the previous module. We apply OCR to them and obtain OCR results. We utilize the edit distance technique to measure word distances and rank them in ascending score. Then, we selected the top five words as candidates to be used to replace incorrect OCR results (Figure 2). We collect only five words because this quantity is reasonable for utilization and resource management. For example, a graph image contains a word "well" at its X-title that is incorrectly rendered as "woll". Our system can select top five candidates ordered by ascending distance scores as follows: welt, will, wall, well, and with. Obviously, if the number of candidates is too small (e.g., one or three), we definitely miss a correct word “well”, which also appears on the list. Moreover, a high quantity of candidates causes unnecessary processes. Their distances are calculated between tokens from OCR results and the other from corresponding caption or paragraphs. Note that the distance inversely varies to a similarity. A higher distance represents a smaller similarity and vice versa. The selected top five candidates for each OCR token are stored into the list of candidates in ascending order of distance scores. Finally, we acquire the OCR results, including their lists of candidates.

3.2.2 Ontology design and creation

To fully support our OCR-error correction, we need to create our own ontology following the design, illustrated in Figure 3, which includes four entities (i.e., Word, TagCategory, PartOfSpeech, and PartTypeCategory classes) and several object properties (e.g., belong_to, has_type, and depend_on). The Word entity represents every individual token from captions and cited paragraphs of the images used in this study. The TagCategory gathers category names or NER attached to each token, such as person name, location, and animal, by querying DBpedia via its SPARQL endpoint. Furthermore, we use Stanford Named Entity Recognizer (Stanford NER) to identify the category of the tokens organized into seven categories, i.e., Location, Person, Organization, Money, Percent, Date, and Time. The PartOfSpeech collects POS tagging of each token. For this entity, the total number of individuals is fixed at 36 instances, whose names are from Penn treebank nodes, such as CC, VB, and NNP. The PartTypeCategory represents groups of POS taggings. For example, a singular proper noun indicates NNP belonging to the Noun group. Regarding properties in our ontology, we design several properties, which states relations among entities. The same_as represents the relations of at least two synonymous tokens that are stored in the Word entity. For example, “Japan” and “Nihon” are synonyms that refer to the same concept. Our ontology covers the synonyms expressing the same concept. Moreover, the depend_on property is a crucial property, because it presents dependency relationships between paired tokens parsed from sentences in the captions and the paragraphs. The number of sub-properties of depend_on relations is fixed at 67 properties representing typed dependencies, such as conj, dep, and nsubj.

To prepare individuals for our ontology, entire sentences included in the captions and paragraphs are tokenized into tokens. Afterward, we utilize a dependency parser (Stanford parser) to analyze the sentences in order to obtain their dependencies, POS tags, and NER classes. As mentioned above, NER classes are obtained by the parser and the SPARQL query processed in DBpedia. All prepared data are gathered as instances of our ontology.

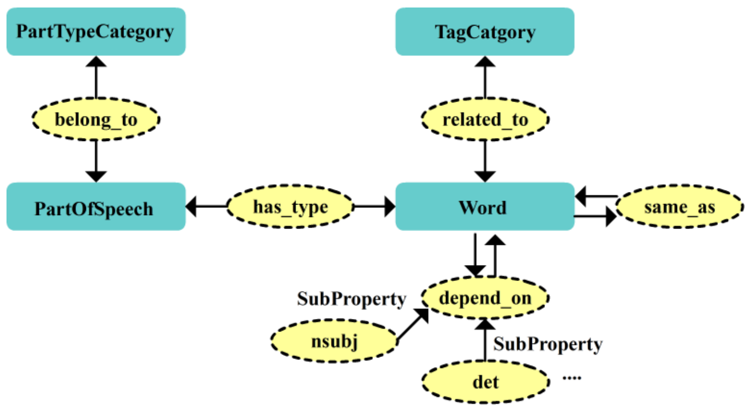


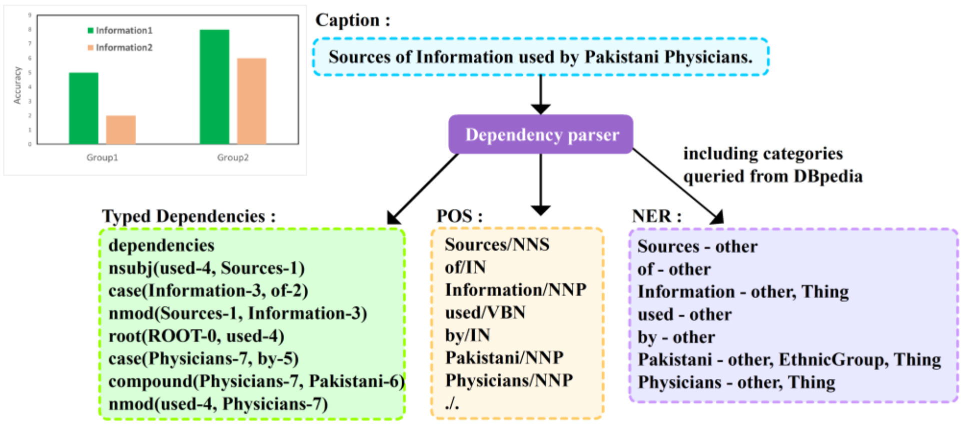
Figure 3. Demonstration of our ontology structure describing entities, properties, and relations.

3.2.3 Error correction

The main purpose of this step is to correct the OCR errors using the ontology and the lists of candidates from the previous steps. The basic idea is that the tokens from graph components should appear in corresponding captions or cited paragraphs because authors generally explain information based on the graphs in their documents.

Initially, we begin to create a dependency dictionary, called DepDic that records the chain dependencies of the tokens. This dictionary is created, if at least one OCR token identically matches to the first candidate in its own list, and the token is used as a head of dependency chains. As an example shown in Figure 4, we obtain a word "Information" from the graph's legend. Suppose that it can be found in its caption, and OCR provides a correct recognition. A parser provides results as typed dependencies based on a caption, including POS tags and NERs. We acquire a dependency chain of "Information" that includes "Sources," "of," "used," "Physicians," "Pakistani," and "by." This chain is recorded into DepDic.

To cover all possible situations for correcting OCR errors, we divided our method into four major conditions, as presented in Figure 5.



```
dependencies
nsubj(used-4, Sources-1)
case(Information-3, of-2)
nmod(Sources-1, Information-3)
root(ROOT-0, used-4)
case(Physicians-7, by-5)
compound(Physicians-7, Pakistani-6)
nmod(used-4, Physicians-7)
```

```
Sources/NNS
of/IN
Information/NNP
used/VBN
by/IN
Pakistani/NNP
Physicians/NNP
./.
```

```
Sources - other
of - other
Information - other, Thing
used - other
by - other
Pakistani - other, EthnicGroup, Thing
Physicians - other, Thing
```

Figure 4. Example of grammar dependency parsing, including POS tags and typed dependencies, and NER classes of each token.

For the first condition, we focus on eliminating recognition noises from our inputs, such as special characters and numbers. This condition is used to filter unused characters. We consider number characters as recognition noises because our main targets in this study are the axis descriptions and the legend, which generally describe in alphabet rather than numeric characters. Moreover, we ignore escape characters (e.g., /, <, and *), because they are reserved characters of SPARQL.

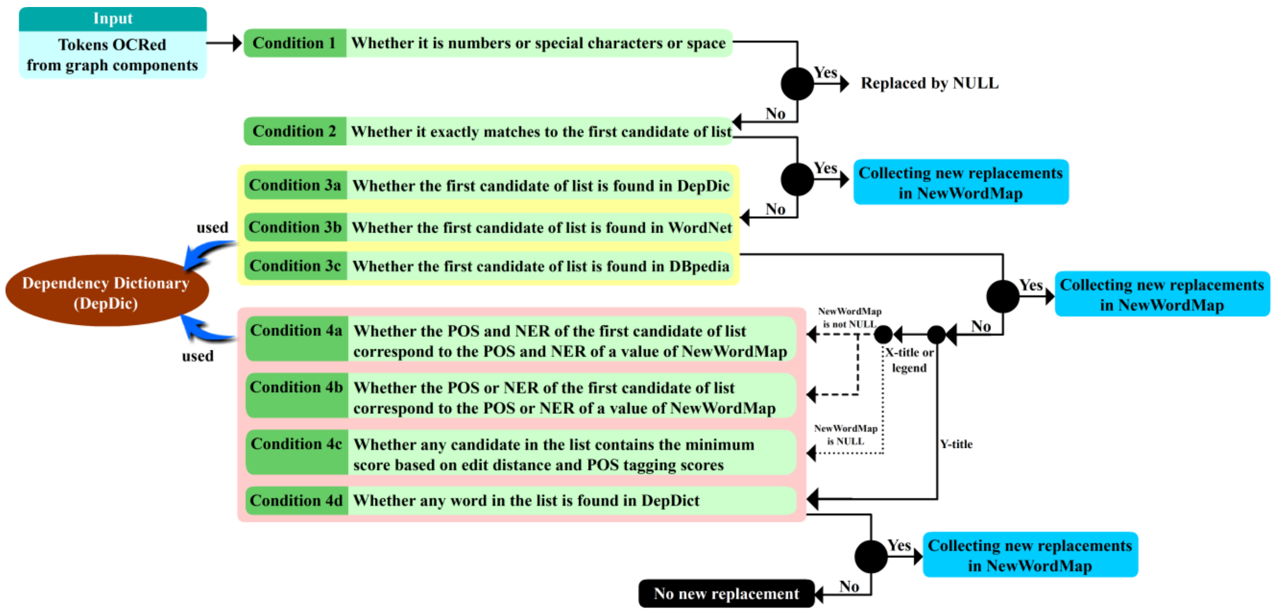

Input
Tokens OCRed from graph components
Condition 1
Whether it is numbers or special characters or space
Yes
Replaced by NULL
No
Condition 2
Whether it exactly matches to the first candidate of list
Yes
Collecting new replacements in NewWordMap
No
Condition 3a
Whether the first candidate of list is found in DepDic
Condition 3b
Whether the first candidate of list is found in WordNet
Condition 3c
Whether the first candidate of list is found in DBpedia
used
Dependency Dictionary (DepDic)
used
Yes
Collecting new replacements in NewWordMap
No
NewWordMap is not NULL
X-title or legend
Condition 4a
Whether the POS and NER of the first candidate of list correspond to the POS and NER of a value of NewWordMap
Condition 4b
Whether the POS or NER of the first candidate of list correspond to the POS or NER of a value of NewWordMap
NewWordMap is NULL
Y-title
Condition 4c
Whether any candidate in the list contains the minimum score based on edit distance and POS tagging scores
Condition 4d
Whether any word in the list is found in DepDict
Yes
Collecting new replacements in NewWordMap
No
No new replacement

Figure 5. Demonstration of OCR-error correction covering possible conditions to filter and correct errors.

The second condition is whether the OCR result finds an exact match in its own list. Our system examines the similarity between the OCR result and the first word of its list whose distance is minimal. Typically, the paired words are identical, if the distance score is equal to zero. If this condition is satisfied, a replacement is unnecessary due to the accurate OCR result obtained. Afterward, we collect the result to a mapping list, namely NewWordMap used to store the OCR results and their new replacements.

The third condition is whether OCR provides a correct recognition that matches nothing in its own list. As aforementioned, our basic idea is that the descriptions of graph components should correspondingly appear in either caption or paragraphs. Unfortunately, the descriptions possibly appear nowhere in the document. Under this situation, we obtain the list of candidates with high distances, which has a low chance to find a match from the list. This condition handles the problem of missing matching tokens by analyzing three minor conditions.

Condition 3a is whether the first candidate of the list has been found in DepDic. Clearly, the first candidate of the list contains the highest chance of matching because of the lowest distance score provided. Moreover, if the candidate is discovered in DepDic, its chance to be selected as a new replacement should be increased. Therefore, if this minor condition is satisfied, our method suggests the first candidate of the list as a new replacement, because not only the smallest distance score is provided, but it also appears in the same chain of dependency. Condition 3b is processed to check whether the OCR result is actually existed by querying WordNet. Condition 3c has a procedure similar to Condition 3b, but it differs in using DBpedia instead of WordNet. If these two minor conditions are satisfied, we receive no null values returned from their SPARQL endpoints, and the new replacement is not needed. Regards an order of Condition 3, we normally apply these conditions following by this order, Condition 3a, 3b, and 3c respectively. However, if the distance score is over than a threshold, the order of the condition is switched to the following order, i.e., 3b, 3c, and 3a.

We reorder the conditions because we need to prevent errors resulting from the length of OCR result that is lower than a threshold, especially for two or three characters. The short-length words generally represent as prepositions (e.g., “in,” “on,” or “at”), conjunctions (e.g., “so” or “as”) and abbreviations (e.g., POS and NLP). Every sentence regularly includes at least one preposition or conjunction, since the short-length words are ordinarily stored in DepDic. Based on this explanation, they may be assigned as an incorrect replacement accidentally. For example, we obtain a word “is” from OCR, and the first candidate of its list is “on” already recorded in DepDic. A distance score of them is only two, but their appearances are totally different. Following the original order, the word "is" is wrongly assigned by the word "on" as a new replacement. In order to reduce a chance to encounter this situation, a rearrangement of condition orders has been suggested. According to the new condition order, we obtain a correct replacement as the word "is" itself, because it has been found in WordNet.

The last condition is Condition 4 separately processed based on types of components. Initially, our system checks on NewWordMap list. If the list is not null, Condition 4a and 4b are processed. Otherwise, Condition 4c is operated. A basic idea of this condition is that X-title and legend should be described by a corresponding category. For example, a financial bar graph contains an X-title, a Y-title, and a legend. The X-title describes product names, which are in the same category (e.g., apple, orange, or banana).

The NewWordMap is available if there is at least one word stored. Condition 4a is whether the POS and NER of the first candidate of the list corresponded to both of a value stored in NewWordMap. To satisfy this condition, we check the POS tag and NER of the first candidate of the list and the POS tag and NER of the word stored in the NewWordMap. If their POS and NER are consistent, we obtain a new replacement. Condition 4b is similar to Condition 4a. If either POS tag or NER has been matched, we also flexibly accept the first candidate of the list as the new replacement. Condition 4c is operated if the NewWordMap is unavailable or null. We cannot find any comparison from the list; thus, we introduce another solution that utilizes only the list of candidates. This condition is whether any candidate of the list contains the minimum score which sums up from both the edit distance score and a POS tagging score. Regards the POS tagging score, we assign a score to each POS tag depended on the priorities of word replacement selection based on our experience. The tagging scores are assigned as following: noun (score = 0), adjective (score = 1), verb (score = 2), article (score = 3), adverb (score = 4), preposition (score = 5), conjunction (score = 6), interjection (score = 7), others (score = 8) and number (score = 9). Noun provides the highest chance to appear at the X-title or the legend; since its score should be minimum. We select the replacement assigned by the smallest score, which basically comes from a summation of the smallest distance score and Noun tagging score.

Y-title is described as a sentence or a noun phrase that is different from the X-title or the legend. Tokens from Y-title connect to other tokens by their dependencies; therefore using DepDic should be an appropriate option for selecting the most similar word in the list as a new replacement. Condition 4d is whether any word in the list appeared in DepDic. Every candidate in the list is iteratively explored in the list of DepDic until a match is retrieved and is selected as the replacement.

Otherwise, if we cannot obtain any new replacement from the above conditions, the OCR tokens are used as their own replacements.

4. Experiments and results

We conducted experiments to evaluate our method. We divided them into four tests as presented in Table 1. To evaluate the first module, we compared results obtained from a tradition method and the first module of our method. The traditional method, namely image partition method, extracted the X- and Y-titles by image partitioning similar to our method, but an idea to extract the legend was different. It extracted the legend by cropping all possible areas where located the legend, such as the top and right side of the image, including irrelevant or relevant parts. A comparison between Experiment 1 and 2 revealed the significant experimental results expressing the performance of the different graph component extraction methods. To evaluate the second module, we observed the results from Experiment 1 and 3 to compare the performance between the edit distance technique and our OCR-error correction. The performance of our study was represented in Experiment 4, which was a combination of module 1 and 2.

Several performance rates (i.e., accuracy, precision, recall, and F-measure) were evaluated in this study. The accuracy is a statistical measurement to identify how well a method tests correctly. A higher accuracy rate represents to the consistency of predicted values which are same as given values. The precision is the measurement of given data to present how many outputs are positively classified. The recall is to define how well the outputs cover the positives. F-measure is an averaged combination of precision and recall. Noise ratio is an evaluated measurement to identify how much recognition noises are produced by the system, such as numbers and special characters. The overall measurement results are summarized in Figure 6 and 7.

Table 1. Settings of our experiments.

Experiment	Method of graph component extraction	Method of OCR-error correction
1	Image partition method	Edit distance
2	Our first module	Edit distance
3	Image partition method	Our second module
4	Our first module	Our second module

As Experiment 1, it was a combination of the image partition method and edit distance. It was said to be a fundamental idea to acquire the information from graphs and correcting OCR results. As the results, all performance rates were presented the lowest values, except the noise ratio. The noise ratio was up to 29.48% that was the maximum ratio comparing to other experiments. However, after we examining the noise ratio from Experiment 2 which was a combination between our graph component extraction and the edit distance, we realized that our first module could efficiently handle the noises of irrelevant parts better than the image partition method because the noise ratio obviously presented a lower rate, 19%. Moreover, the accuracy and F-measure were increased to 57.28% and 50.54% respectively, whereas the performance rates of Experiment 1 were only 46.98% and 39.77%. Experiment 3 was a combination of the image partition method and our OCR-error correction. All performance rates were dramatically improved comparing to the first experiment. The accuracy was up to 80.75%, and the F-measure reached to 82.28%. For Experiment 4, we combined our first and second modules proposed by this study. The performance was better than others. We obtained the highest accuracy rates, 84.23%, and F-measure, 86.02%, including less of recognition noises.

We obtained the number of errors 249 tokens from total 1579 tokens in Experiment 4. We analytically observed causes of errors separated into three types: missing token error, real-word error, and suggestion error. The missing token error presents the number of tokens unable to extract from the graph. The real-word error represents the error of misrecognition but accidently matches to a vocabulary item in a dictionary. The suggestion error means the error from our system suggesting an incorrect result. To realize a portion of errors, the percentages of each error proportioned to the total number of errors were presented as follows: 27.71% for the missing error, 37.75% for the real-word error, and 34.54% for the suggestion error. Clearly, among the errors obtained during the experiment, the real-word error and the suggestion error needed to be concerned and mitigated.

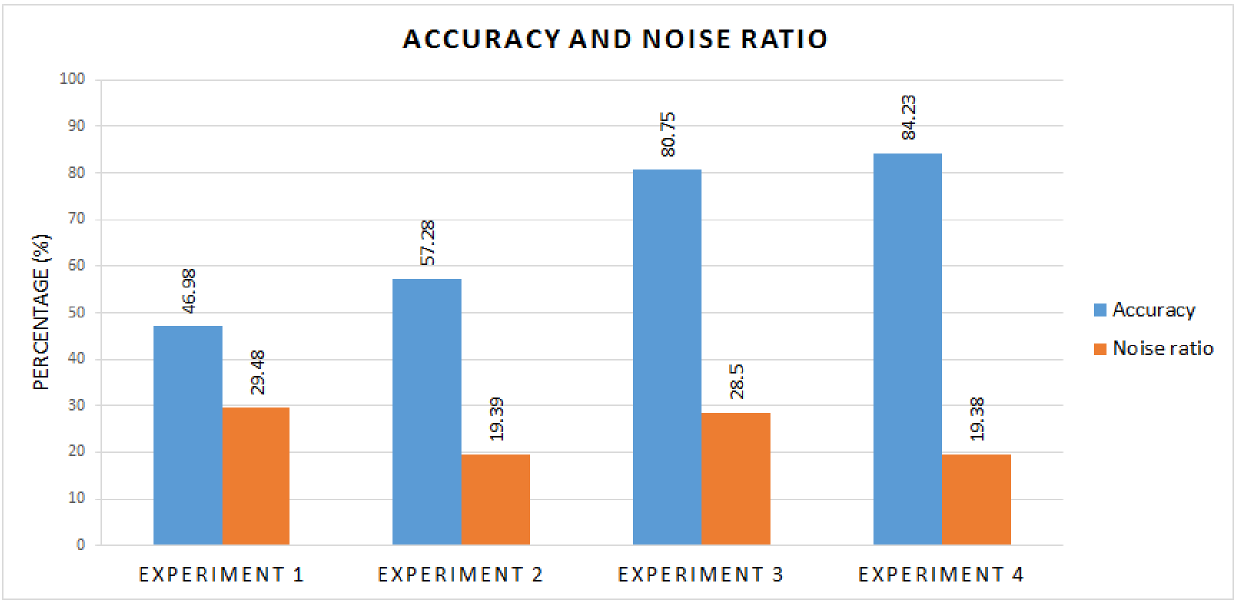


Figure 6. Illustration of accuracies and noise ratios of all experiments.

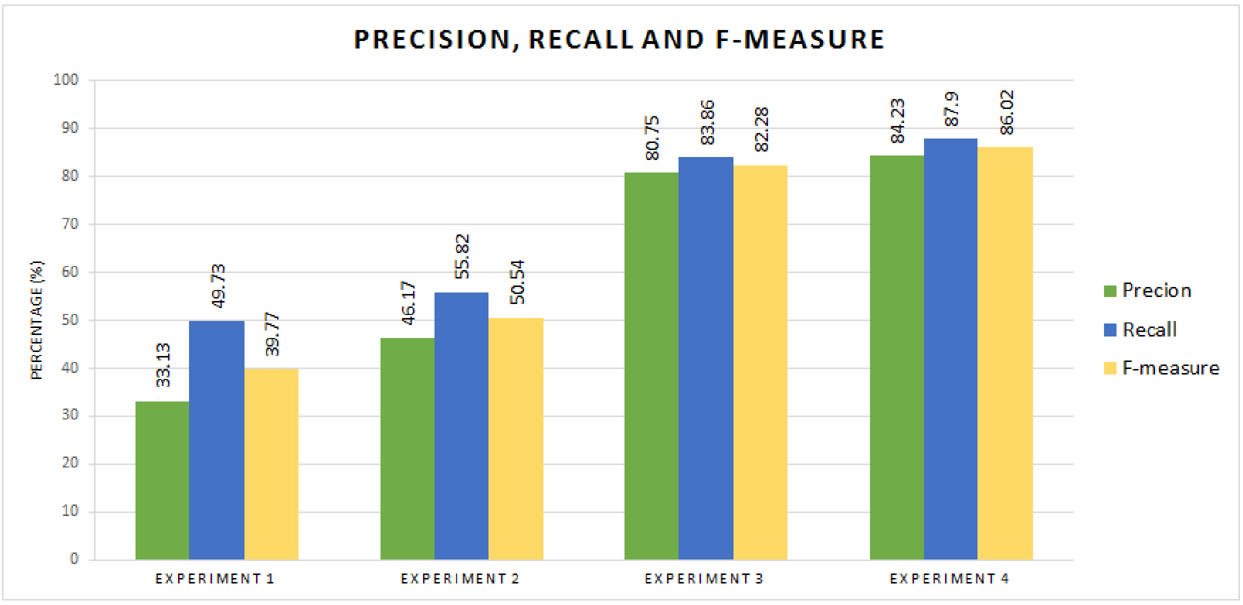


Figure 7. Illustration of precision, recall and F-measure of all experiments.

Moreover, we investigated the number of missing tokens for Experiment 3 and 4. Note that the total number of tokens, which should be able extracted by OCR, was 1165. In Experiment 3, the missing tokens were 151 tokens or 13% of total tokens missing. Meanwhile, in Experiment 4, we obtained a missing tokens only 69 tokens (6.92%). Apparently, the missing tokens were decreased, if we applied our data to the graph component extraction.

Figure 8 presents accuracy rates of all conditions proposed in our OCR-error correction. Condition 1 used to detect and omit the recognition noises provided the 100% correction. Due to our effective graph component extraction, the accuracy rate of Condition 2 reached 99.15%. Condition 3 presented a reasonable accuracy rate about 81.11%; therefore, using ontologies to investigate a meaning of a word was also appropriate. However, the lowest accuracy was appeared at Condition 4, 29.47%.

Furthermore, we statistically calculated the significant difference between results from Experiment 2 and 4 by McNemar's test. This tool uses for statistically testing on paired nominal data to examine a significant change from two sets of data obtained before and after treatments. In our case, the before treatment referred to Experiment 2 using the edit distance, and the after treatment was our OCR-error correction in Experiment 4. Note that we ignored results corrected by Condition 1 to stable the statistical data because the edit distance cannot handle the noises. We calculated a two-tailed probability value (*P* value), which used to determine to accept or reject a null hypothesis. The small *P* value represents a significant difference between two sets of data. The two-tailed P value is less than 0.0001 that means the results from both experiments are considered to be extremely statistically significant.

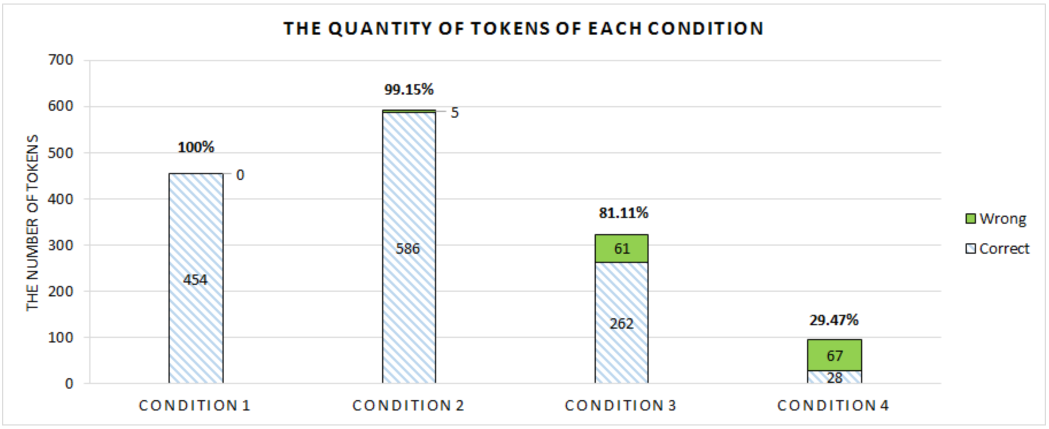


Figure 8. The number of tokens, including accuracy rates of each condition.

5. Discussion

We proposed a new method of OCR-error correction based on bar graph images using semantics. We improved our idea by using a graph component extraction proposed in our previous study to reduce irrelevant parts from extracted components before applying to OCR. We aimed to advance the quality of OCR results and proposed an effective method to accurately detect and extract graph components. We conducted four tests to evaluate the performance of our method (Table 1). We calculated several performance rates, i.e., accuracy, precision, recall, F-measure, and noise ratio. The experiment representing our method was Experiment 4.

As presented in Figure 6 and 7, Experiment 1 provided the lowest performance rates, including the highest noise ratio; thus, a combination of fundamental methods (i.e., the image partition method and edit distance) is inappropriate to solve this study's problems. For Experiment 2, we observed that the noise ratio reduced around 10% from the first experiment. Moreover, the accuracy and F-measure had 10% increased. This positive situation happened, because we changed the image partition method to our first module. Obviously, the performances are improved, if the noiseless data are used.

Typically, during a recognition process of OCR, it analyzes many objects inside images that sometimes are not character strings. They may lead OCR misunderstanding because they interfere the recognition process and mainly cause errors; hence, our data cleaned by our graph component extraction definitely enhance the performance of the system.

Experiment 3 provided high accuracy and F-measure: 80.75% and 82.28% respectively. To compare between Experiment 1 and 3, the results showed that our OCR-error correct dominantly affects the performance of the system, because the accuracy rate and F-measure of Experiment 3 were much greater than Experiment 1 about 33.77% and 42.51% respectively.

The performance of our graph component extraction supports the system to reduce noises around 10% as resulted in Experiment 1 and 2. Moreover, according to the different performance of Experiment 1 and 3, it is dramatically increased to around 38% averagely. This evidence contributes the quality of our second module or OCR-error correction that is much better than the edit distance in a case of suggesting the correct tokens because ours can handle a limitation of edit distance.

As mentioned above, the basic idea of edit distance is to use two tokens compared to measure their distance. The comparable tokens are selected from OCR results and the captions or paragraphs. In a case of edit distance providing a correct suggestion, the tokens from graphs are required to have a match to tokens in the captions or paragraphs. However, the tokens may not be mentioned in any contents of documents because of two reasons. First, they are general words that should be known by readers. Thus, it is unnecessary to explain them. Second, they may not directly relate to their topics or studies. Based on this story, the edit distance cannot guarantee the correct suggestion, particularly in the situation of no matched tokens in the captions and paragraphs. On the other hand, our study provided the good quality of OCR-error suggestion which utilized the ontologies to check the meaning of tokens; therefore, it properly mitigates the shortcoming of edit distance. We also obtain a correct suggestion, even if we cannot find any match in the captions or paragraphs.

As the results of Experiment 4, we acquired high accuracy and F-measure: 84.23% and 86.02% respectively; in addition, the noise ratio was decreased comparing to Experiment 1, 19.38%. The accuracy and F-measure of our method were significantly higher than the first experiment about 37.25%and 46.25% respectively. The main purpose of this experiment is to prove the performance of the combination of our methods, i.e., our graph component extraction and OCR-error correction. An important implication of these findings is that a cooperation of our graph component extraction and OCR-error correction are supportive each other because the performance also improves comparing to other experiments.

We endeavored to compare the results from Experiment 4 to a state-of-art study. Zhuang et al. (Zhuang & Zhu, 2005) presented a remarkable study to correct OCR errors based on semantics similar to ours. Their results reported that, after we compared between their method and a basic method, an error reduction was 29%, and the accuracy was 83.73%. Likewise, we analyzed the error reduction comparing between our method and the edit distance. Our error reduction was about 27%, and our accuracy was slightly higher than the previous study, 84.23%. Eventually, the idea to use the semantics to mitigate the OCR problem was agreeable, because, to compare to non-semantic methods, the higher performance were presented from our and the previous studies; furthermore, the results were corresponding.

To analyze the errors obtained from Experiment 4, there were three types of errors occurred during the experiments, i.e., the missing error, the real-word error, and the suggestion error. As the results of Experiment 4, the highest proportion of errors was the real-word error. This error happens when the OCR incorrectly recognizes tokens, but it has been accidently found in ontologies. The possible solution is to use a specific ontology. The ontologies currently used in this study are general ontologies related to English vocabularies. Since, an opportunity to get incorrect suggestion should be reduced if we use the specific ontology rather than the general ones because the suggested results relate to a domain of the graph. For example, if a graph relates to biology; thus the biology ontology should be utilized. The next error that we should concern is the suggestion error. It occurs, if the recognized token is incorrect and not found in the ontologies; hence, our system suggests the most similar token which is mostly incorrect, because there is not an identical token appearing in the captions and the paragraphs. One possible solution to this problem is to extend the content of documents to increase a probability to find an identical token, such as using whole contents of the document, not limited to only the captions or corresponding paragraphs. However, it is time-consuming, because there are a lot of tokens that need to measure the distances, including querying to ontologies. For the missing error, it happens because of the limitation of OCR and a mistake of partitioning process of our graph component extraction. OCR sometimes cannot recognize any characters, if the font size is too small or too big; since OCR returns a null value, which causes a missing token. Furthermore, we partitioned images by a constant; thus, it is possible to cut them at wrong positions. For example, we have a graph containing a two-sentence Y-title. The system may mistake to cut at the middle of the title. We retrieve an incomplete title that causes missing token errors.

To deeply analyze the results of each condition as presented in Figure 8, Condition 1 offered the 100% correction. It means that our system has a high capability to detect and omit the recognition noises. For Condition 2, we obtained the high accuracy because of the benefit of our graph component extraction. It extracts the relevant components, which help to enhance the OCR performance to correctly recognize character strings. The reasonable rate is presented in Condition 3. This condition is proved that a viewpoint to use grammar dependencies and ontologies is acceptable. However, most errors occurred in this condition are the real-word error. The final condition is Condition 4 suggesting the lowest accuracy rate comparing to among conditions. The suggestion errors have been mostly found in this condition. The causes and solutions of these errors have been described above.

As the statistical evidence, we concluded that the difference between our OCR-error correction and the edit distance was the significant difference due to a small P value obtained.

6. Conclusion

In this study, we proposed the method of OCR-error correction based on semantics, including the graph component extraction. The objectives of this study were to extract their significant information by using OCR and to correct OCR results by utilizing the ontologies and dependency parsing.

From the experiments that have been carried out, it is possible to conclude that our methods achieve the objectives of this study. As the experimental results, our method presented the highest performance rates greater than other methods, and the noise ratio was small. Simultaneously, the fundamental method showed the highest noise ratio and the lowest performance rates, because many irrelevant objects were included in the input data, which interfered the recognition process. Furthermore, the edit distance suggested a correction based on strings appearing in the captions and paragraphs; therefore, suggestion errors certainly occurred, if identical tokens were missing from them. The OCR-error correction introduced in this study contained four conditions to cover all possible situations that might occur during the correction processing. It has clearly shown that our method can perfectly handle and omit the recognition noises, such as numbers and special characters. Moreover, due to our effective graph component extraction, the cleaned components omitted irrelevant parts were acquired. Indeed, with this data, OCR provided accurate recognitions. Using the semantics to correct OCR errors is also appropriate because it mitigates the problems of missing identical tokens in the captions or paragraphs.

The next stage of our research will be moved to graph-content information extraction, such as a height of a bar and a tendency of a line. We will design a new ontology to support extractable graph information and to utilize other ontologies in order to reveal latent information.

References

Lasko, T. A., & Hauser, S. E. (2000). Approximate string matching algorithms for limited-vocabulary OCR output correction. In Photonics West 2001-Electronic Imaging. International Society for Optics and Photonics, 232-240.

Chen, D., Odobez, J. M., & Bourlard, H. (2004). Text detection and recognition in images and video frames. Pattern recognition, 37(3), 595-608.

Jobbins, A., Raza, G., Evett, L., & Sherkat, N. (1996). Postprocessing for ocr: Correcting errors using semantic relations. In LEDAR. Language Engineering for Document Analysis and Recognition, AISB 1996 Workshop, Sussex, England.

Nagata, M. (1998, August). Japanese OCR error correction using character shape similarity and statistical language model. In Proceedings of the 17th international conference on Computational linguistics-Volume 2, 922-928.

Kataria, S., Browuer, W., Mitra, P., & Giles, C. L. (2008). Automatic Extraction of Data Points and Text Blocks from 2-Dimensional Plots in Digital Documents. In AAAI, 8, 1169-1174.

Huang, W., Tan, C. L., & Leow, W. K. (2005). Associating text and graphics for scientific chart understanding. In Eighth International Conference on Document Analysis and Recognition (ICDAR'05). IEEE, 580-584.

Tong, X., & Evans, D. A. (1996). A statistical approach to automatic OCR error correction in context. In Proceedings of the fourth workshop on very large corpora, 88-100.

Wick, M., Ross, M., & Learned-Miller, E. (2007). Context-sensitive error correction: Using topic models to improve OCR. In Ninth International Conference on Document Analysis and Recognition (ICDAR 2007). IEEE, 2, 1168-1172.

Bassil, Y., & Alwani, M. (2012). Ocr post-processing error correction algorithm using google online spelling suggestion. arXiv preprint arXiv:1204.0191.

Zhuang, L., & Zhu, X. (2005). An OCR post-processing approach based on multi-knowledge. In International Conference on Knowledge-Based and Intelligent Information and Engineering Systems. Springer Berlin Heidelberg, 346-352.

Kanjanawattana, S., & Kimura, M. (2016). A proposal of a method to extract and identify components in bar graphs. International Journal of Computer Theory and Engineering (IJCTE). Forthcoming 2016.

Alvarez, J. M., Gevers, T., LeCun, Y., & Lopez, A. M. (2012, October). Road scene segmentation from a single image. In European Conference on Computer Visio. Springer Berlin Heidelberg, 376-389.

Cheng, C., Koschan, A., Chen, C. H., Page, D. L., & Abidi, M. A. (2012). Outdoor scene image segmentation based on background recognition and perceptual organization. IEEE transactions on image processing, 21(3), 1007-1019.

Hiran, K. K., & Doshi, R. (2013). An artificial neural network approach for brain tumor detection using digital image segmentation. Brain, 2(5).