A Machine-Learning Based Approach for Extracting Logical Structure of a Styled Document

Tae-young Kim[1], Suntae Kim[1*], Sangchul Choi[1], Jeong-Ah Kim[2], Jae-Young Choi[3], Jong-Won Ko[3], Jee-Huong Lee[3], Youngwha Cho[3]
[1]Dept. of Software Engineering, CAIIT, Chonbuk National University
567 Baekje-daero, deokjin-gu, Jeonju-si, Jeollabuk-do 54896, Republic of Korea
[2]Department of Computer Education, Catholic Kwandong University, Beomil-ro 579 beon-gil, Kangneung-Si, Kangwon-Do, Republic of Korea
[3]College of Information and Communication Engineering, SungKyunKwan University,
2066 Seobu-Ro, Jangan-Gu, Suwon, Gyeonggi-Do, Republic of Korea
[e-mail: {*rlaxodud1200, stkim*, 114477aa*}*@jbnu.ac.kr, clara@cku.ac.kr*, {*jaeychoi, jwko0820, john, choyh2285*}*@skku.edu*]
*Corresponding author: Suntae Kim

Received October 10, 2016; revised December 13, 2016; accepted January 16, 2017; published February 28, 2017

Abstract

A styled document is a document that contains diverse decorating functions such as different font, colors, tables and images generally authored in a word processor (e.g., MS-WORD, Open Office). Compared to a plain-text document, a styled document enables a human to easily recognize a logical structure such as section, subsection and contents of a document. However, it is difficult for a computer to recognize the structure if a writer does not explicitly specify a type of an element by using the styling functions of a word processor. It is one of the obstacles to enhance document version management systems because they currently manage the document with a file as a unit, not the document elements as a management unit. This paper proposes a machine learning based approach to analyzing the logical structure of a styled document composing of sections, subsections and contents. We first suggest a feature vector for characterizing document elements from a styled document, composing of eight features such as font size, indentation and period, each of which is a frequently discovered item in a styled document. Then, we trained machine learning classifiers such as Random Forest and Support Vector Machine using the suggested feature vector. The trained classifiers are used to automatically identify logical structure of a styled document. Our experiment obtained 92.78% of precision and 94.02% of recall for analyzing the logical structure of 50 styled documents.

***Keywords*:** Logical Structure Analysis, Machine Learning, Feature Vector, Document Management System

A preliminary version of this paper was presented at APIC-IST 2016, and was selected as an outstanding paper.

https://doi.org/10.3837/tiis.2017.02.023 **ISSN : 1976-7277**

1. Introduction

A styled document uses diverse styling features for decorating different elements of a document such as sections, subsections and contents. Based on the consistent use of a style for each document element, a reader can recognize a logical structure of a document at a glance, while a computer cannot currently understand the structure if a writer does not explicitly denote a specific type of an element by using a function of a word processor program[1](e.g., ***Heading 1***,*Heading 2* in the *Style* menu of MS-WORD). However, a writer tends not to use the styling functions to designate a type of a document element during writing a document, which is an issue for a computer to automatically recognize the logical structure of a document. According to our survey (see **Table 1**), 56.7% (50.88% + 5.85%) of 171 documents are not properly authored by using the styling functions, and even 80%(76% + 4%) of 75 informal documents do not have appropriate style information in the documents, which indicates that more than half of documents are hard for a computer to automatically recognize the logical structure of a document, and it is harder for informal documents.

There has been several research to address the above issue. These researches can be summarized in two streams in terms of their approach. One is to define diverse rules to map physical elements (e.g., layout, bounding box, font attributes) from a document into logical structural elements(e.g., sections and authors)(see [2, 3, 4]). While these approaches can extract logical entities in an intuitive manner by using the rules, the rules should be updated accordingly whenever the document structure is changed. Another research stream is to apply machine learning algorithms to data extracted from physical elements of a document, and train the machine learning algorithms to automatically identify the logical elements (see [5, 1, 6]). However, these approaches only focus on document images, not styled documents authored in word processor programs.

Table 1. Statistics of styled and non-styled documents searched by Google with the keyword Report, SW Requirements Specifications and Research Plan for Formal Docs, and Meeting Minutes, and Agenda for Informal Docs

	Styled	**Non-Styled**	**Partially Styled**	**Num. of Docs**
Formal Docs	59(61.46%)	30(31.25%)	7(7.29%)	96
Informal Docs	15(20%)	57(76%)	3(4%)	75
All Docs	74(43.2%)	87(50.88%)	10(5.85%)	171

approaches can extract logical entities in an intuitive manner by using the rules, the rules should be updated accordingly whenever the document structure is changed. Another research stream is to apply machine learning algorithms to data extracted from physical elements of a document, and train the machine learning algorithms to automatically identify the logical elements (see [5, 1, 6]). However, these approaches only focus on document images, not styled documents authored in word processor programs.

In order to handle the issue, we proposes a machine learning based approach to analyzing the logical structure of a styled document composing of sections, subsections and contents. It is composed of two phases: Training and Evaluating. In the training phase, we proposes a feature vector for characterizing document elements from a styled document, composing of eight features such as font size, indentation and periods, each of which is a frequently discovered item in styled documents. Then, we trained four machine learning classifiers

Random Forest[7], *Support Vector Machine*[8], *K-NN*[9] and *Naive Bayes*[10] using the suggested feature vector. In the evaluation phase, we analyze the logical structure of a styled document with the trained classifiers. Our experiment for the suggested approach obtained 92.78% of precision and 94.02% of recall in identifying the logical structure of the styled documents. The rest of the paper is structured as follows. Section 2 presents related work on analyzing the logical structure of diverse types of documentsSection 3 provides a background on a styled document authored in a word processor, and presents our suggested approach for identifying the logical structure of a styled document using machine learning based classifiers. Section 5 shows the experiment result for our approach. , and Section 6 concludes the paper.

2. Related Work

This section presents previous research on analyzing the logical structure of a document images or word processor files. These researches can be divided into two research streams: rule based approaches and machine learning based approaches. Kim et al. [2] suggested a rule based approach that maps physical elements (e.g., layout, bounding box, font attributes) from formal document images (e.g., an article or a proceeding paper) into logical structural elements such as title, author, affiliation and so forth. First, they extracted diverse level of geometric information (e.g., layout in the high level, characters in character level) using OCR (Optical Character Recognition). Then, they defined a set of mapping rules based on the physical entities to recognize the logical entities of the document images.

Similar to this research, Niyogi and Srihari[3] identified the logical structure of the newspaper such as a story and a photo block based on the physical structure including a document, a page and a block. They defined rules to extract the logical structure from the physical entities. Rauf *et al.*[4] tried to identify the logical elements of software requirements document such as use case name, actor name and scenarios from different requirements specification templates. They defined a set of rules that map physical entities of different requirements templates into logical entities of a requirements specification.

The aforementioned rule based approaches have a similar shortcoming that the rules are tightly coupled with a documents domains such as a requirement specification, a research or new papers. Thus, if the document structure is changed, the rules should be updated accordingly, while our approach is independent from domains because we extracted common features of documents from diverse domains.

As another approach, various researches applied machine learning algorithms to identifying the logical structure of a document. Kan *et al.*[5] also tried to identify the logical entities of formal document images(e.g., research papers). To identify the logical structure of a document, they separated the physical entities into a logical structure(LS) feature including a layout and text, a generic section(GS) feature composed of text headers with labels for training. They then built two models from the two features using the Conditional Random Field(CRF)[16]. The two models were merged to recognize the logical entities of the document better.

Mohemad *et al.*[1] identified document structure from structured PDF files using Machine learning clustering algorithms. They used the geometric information (i.e., x-y coordinates in a page) of the PDF elements and measured distance between elements using the coordinates. Then, they clustered the elements using Agglomerative hierarchical clustering algorithm[17]. Mao et al.[6] also suggested a machine learning based approach to recognize a logical entities in multi-style document images. They suggested five features such as font size, percentage of digit, percentage of capital letters, percentage of italic letters and percentage of bold letters. Then, they applied the Support Vector Machine algorithm to the data to identify the logical entities.

Similar to Kim, Kalapfl *et al.* [18, 19, 20] also suggested the machine learning based approach to support a digital library. In addition to the geometric information, they computed content-similarity between entities in a PDF document to identify its logical structure [18]. After this, Kalapfl and Kern[19] enhanced this research to automatically recognize the contextual information (e.g., journal abstract, email and address of authors, a title of a paper) to store them in a digital library. They defined six categories of feature vectors such as Language Model Features, Layout Features and so forth, and classified them by using Conditional Random Field [21]. Based on two researches, they automatically built the RDF (Research Description Framework) [22] storage to support semantic query of the digital information [20].

The suggested machine learning based approaches can be applicable only to the specific application domain. Mohemad *et al.*'s approach can be used only for identifying logical entities of a single page document image. Also, Kalapfl *et al.*'s approach can be applied into recognizing the logical entities from the PDF documents. However, a document is generally composed of many pages and contents can be spread throughout more than two pages, and written by using diversed word processors. In this situation, their approach cannot guarantee a good performance. Compared to these approach, since our approach is not based on a document image, but based on a word processor file, the entities throughout pages do not affect to the precision of the approach.

3. Background: Styled Documents and Its Logical Structure

There are diverse elements of a styled document such as a title, sections, subsections, tables and figures in a document. The structure can vary depending on the purpose of the document. However, most of the documents including reports, papers and meeting minutes tend to have a general structure mainly composing of the section, subsection and contents. While a section is a major logical division of a document, a subsection is a sub-logical division of a section. Also, a content indicates contents of a subsection or section generally consisting of sentences, paragraphs, tables and figures.

There are diverse elements of a styled document such as a title, sections, subsections, tables and figures in a document. The structure can vary depending on the purpose of the document. However, most of the documents including reports, papers and meeting minutes tend to have a general structure mainly composing of the section, subsection and contents. While a section is a major logical division of a document, a subsection is a sub-logical division of a section. Also, a content indicates contents of a subsection or section generally consisting of sentences, paragraphs, tables and figures. The logical structure of a document indicates the hierarchical structure composing of sections, subsection and contents to effectively deliver intension of a document [23].

Fig. 1 shows the Development Plan Document composing of one section, two subsections and its contents. Each document element constitutes a logical hierarchy of the document. A writer tends to decorate each document elements consistently with the same font and its size, indentation, and bold or italic and so on. This makes a human to easily understand the logical structure of the document. Inconsistent use of each element's decoration makes one to feel awkward and even doubt the logical structure of the document. To facilitate it, many of word processors provide a styling function to be able to indicate the type of a document element with a combination of format decorations. It also helps a computer recognize the type of a document element, and enables the computer to manage document in more detail, rather than a file as a management unit.

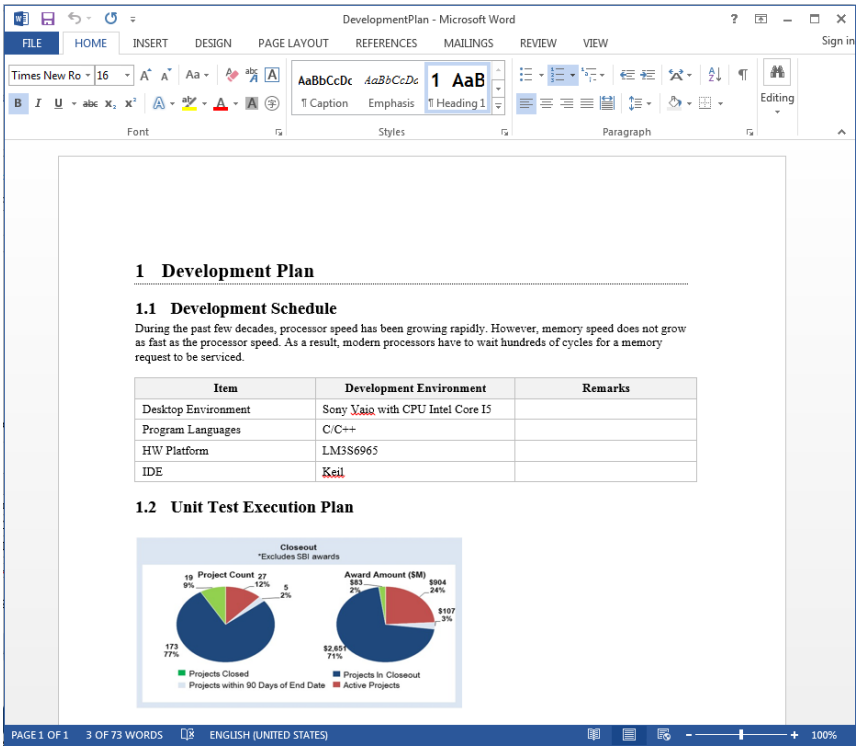


Fig. 1. Logical Structure of Styled Documents edited in the MS-WORD word processor

However, many documents are not authored by using the styling function as shown in **Table 1**. With a human's cognition, it is very hard to recognize use of the styling functions, because their appearance is almost similar. For a computer, however, there is no clue to figure out the type of document elements, unless a writer indicates a specific type of document elements using styling functions of a word processor(e.g., ***Heading 1***,*Heading 2* in the *Style* menu of MS-WORD). Thus, the objective of this paper proposes an approach for a computer to automatically recognize the type of a document element from a styled document not authored by using the styling function.

4. Extracting Logical Structure from Styled Documents

This section presents our suggested approach for identifying the logical structure of a styled document using machine learning based classifiers. In order to make a computer to automatically recognize types of document elements, we suggest a machine learning based approach as shown in **Fig. 2**. It consists of two phases: the *training* and *evaluation* phases. In the training phase, our approach obtains tagged styled-documents with document elements such as section, subsection and contents, extracts eight features, and trains classifiers to build a model for recognizing document elements. In the evaluation phase, different set of styled documents with tagging are used for evaluation. Then, we extract the same features with those of the training phase, and classify the document elements using the best accurate classifier from the training phase. From the following, we describe each phase and containing steps in more detail.

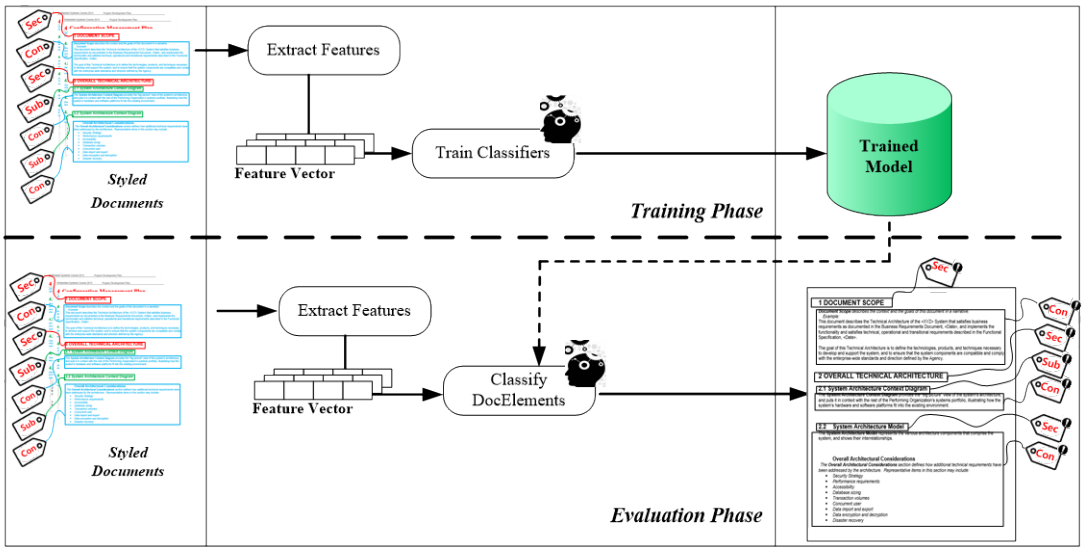


Fig. 2. An Approach for Extracting Logical Structure from Styled Documents

4.1 Extracting a Feature Vector

As the first step to apply machine learning techniques, the feature vector should be defined[11]. Thus, we gathered 171 documents (see **Table 1**) composed of reports and research papers, and inspected the characteristics of the styled document elements. Based on the inspection, we defined the feature vector composing of eight features as shown in **Table 2**. The feature vector is built for each element separated by CRLF (Carriage Return Line Feed).

Table 2. A Feature Vector for Analyzing the Structure of a Styled Document

Feature	Type	Description
FontSize	Real Number	The font size (FS) of a element (e) of a document (d) is computed with, $\frac{FS_e - MIN\_FS_d}{MAX\_FS_d - MIN\_FS_d}$ where MIN_FS_d and MAX_FS_d indicate minimum and maximum font sizes of the document d respectively.
Indentation	Boolean	Check if there is an indentation composing of a tab or more than one blank.
HasPeriod	Boolean	Check if the element ends with a period or not.
Multiple Sentences	Boolean	Check if the element is composed of multiple sentences ending with periods.
Aligment	Factor	Alignment type of each element, it can be one of the *Left, Right* ,*Center* and *Both*.
IsBold	Boolean	Check if the element has a **bold** attribute.
ItemMarkers	Boolean	Check if the element starts with item markers such as '1.', '1.1', • and so forth.
IsUnderLined	Boolean	Check if the element has a underline attribute.

In the **Table 2**, the *FontSize* feature is normalized, as the font size of each element is relatively

different compared to other elements in a document. For example, the font size of a section is generally greater than that of a subsection. Also, the font size of a subsection is also normally greater than that of contents. The *HasPeriod* feature checks if the elements ends with a period, reflecting that most of the titles of sections and subsections do not make a complete sentence ending with a period. Similar to this feature, the *MultipleSentences* feature is intended to stress the characteristic of a content, which are generally compose of multiple sentences. In case of sections or subsections, they do not generally contain multiple sentences. It should be noted that we ignored tables and images in defining the feature vector, because those are not shown in the titles of sections and subsections. They are not generally used in making titles of sections and subsections. Therefore, we do not consider them when defining a feature vector.

Fig. 3 illustrates the feature vectors extracted from a styled document. We put the number in front of each element to facilitate explanation. For the first feature *FontSize*, (1) is 1, (2) and (5) are 0.86, (3) is 0.41 by normalizing them. However, the table (4) and the figure (6) do not measure the font size, so that it become 0. For the second feature *Indentation*, (4) and (5) have 'true' because they contain an indentation respectively. As (3) is composed of multiple sentences, the *Multiple Sentence* feature is set to be 'true'. For the *ItemMarker* feature, the section (1) and the subsections (3) and (5) has the item markers. The last column is the *Label* to support training of the classifiers.

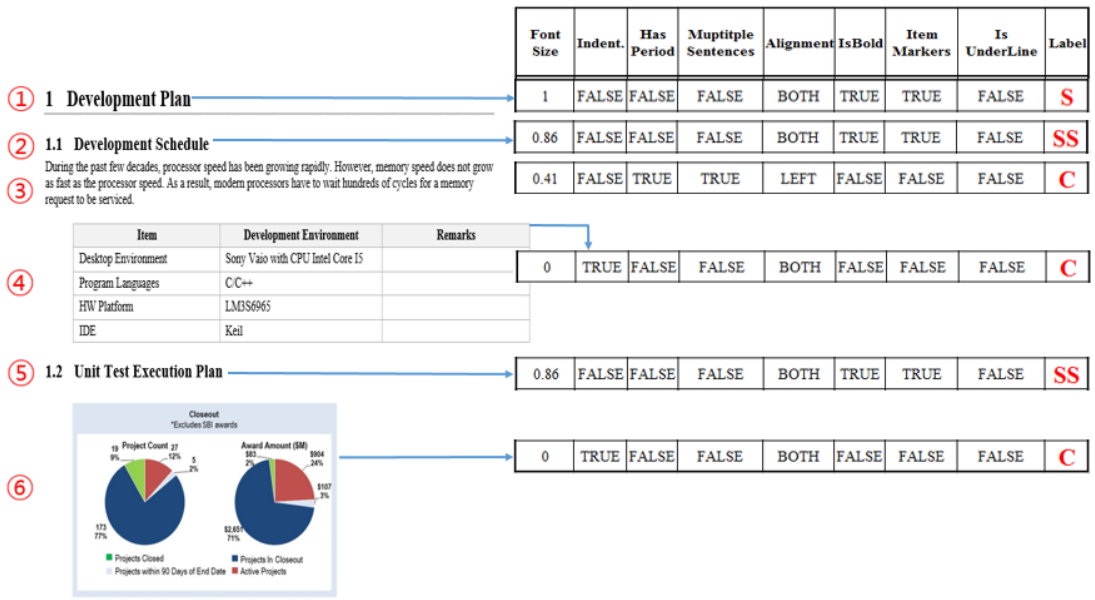


Fig. 3. Examples of Extracting Feature Vectors from Elements of a Styled Document

4.2 Training Classifiers

The next step after defining the feature vector is to train classifiers with the given training data set. We select four representative classifiers: Random Forest[7], Support Vector Machine(SVM)[8], K-Nearest Neighbor(KNN)[9] and Naive Bayes(NB) algorithms[10]. To implement each classifier, we applied the Weka library 3.8[12] with following parameters:

1. Random Forest: the weka.classifiers.trees.RandomForest class with numTrees = 100
2. SVM: the weka.classifiers.functions.LibSVM class with cost = 1.0, degree = 3, kernal type = radial basis function: exp (-gamma* |u-v|^2)

3. KNN: the weka.classifiers.lszy.IBk class with KNN = 3
4. Navie Bayes: the weka.classifiers.bayes.NaieBayes class with no parameters

In order to measure the distance between two feature vectors, we apply the Euclidean distance function after converting Boolean TRUE into 1, and FALSE into 0. In Section 3, we apply the classification algorithms, and find the best appropriate algorithms for analyzing the logical structure of styled documents.

5. Experiments

This section describes the results of the experiment that is designed to evaluate our approach. The two-fold purpose of this evaluation is to validate the feature vector suggested in the previous section, and to find out the best performance classifier among Random Forest, SVM, KNN and Naive Bayes algorithms with measuring precision and recall of the classifiers. For this experiment, we collected 50 documents from Google and carried out the experiment first. Then, we make a balanced data set composing of the similar number of sections and subsection with that of contents, and find the best classifier again.

Table 3 shows the data set for training and evaluation of our approach. We collected 50 documents from Google with the keyword Report, Software Requirements Specifications, and Research Plan, Meeting Minutes and Agenda, and extracted document elements. The Unbalanced Data Set indicates the original data set extracted from the documents, while the Balanced Data Set denotes the data set which makes the elements to be a similar number of elements with others. To make the balanced data set, we randomly sampled the designated number of elements from the original elements of unbalanced data set with replacement. For example, the 5,655 subsections are sampled from the 2,845 subsections of unbalanced data to make 8,500 subsections.

Table 3. Documents and Document Elements for this Experiment

	Unbalanced Data Set	**Balanced Data Set**
# of Documents	50	50
# of Sections	750	8,500 (+ sampled 7,750)
# of Subsections	2,845	8,500 (+ sampled 5,655)
# of Contents	8,535	8,535

5.1 Validation of the feature vector

As a first step experiment, we manually put labels including a section, a subsection and a content with different colors into each element. Then, we parsed the documents written in MS-Word by using Docx4J[13], which is a library for accessing the elements of MS-Word file, and analyzed the document in terms of the feature vector. We analyzed which attributes mostly have an influence on characterizing types of an element by using Weka[12]. To obtain the ranked attributes, we applied information gain [14] to the training set. **Table 4** shows the result of the analysis, showing that IsBold, FontSize and HasPeriod are the most influential elements in turn for recognizing types of an element in the styled document. Additionally, the result shows that indentation does not affect the document element identification, while underline has more influence on the identification rather than indentation.distibution of the feature vector, we can understand that the feature vectors can characterize the elements of a styled document in an appropriate way.

In addition, we investigated the data set in terms of frequency of the values, and selected t he top three feature attributes: IsBold, FontSize and HasPeriod. **Table 5** shows the result of t he investigation. The data set obtained from the unbalanced data set. As shown in the table, 8 1% of the section has a large font size (0.67-1.0 in the normalized measure), while 86% (52% +34%) of subsections are written in the medium and small size of a font respectively. For the bold attribute, 91% of sections and 84% subsections tend to have the bold attribute, while85% of the contents do not have it. This indicates that the feature IsBold can clearly separate secti ons and subsections from contents. That of the contents tends to use smaller size of fonts, indi cating that subsections and contents are not clearly distinguished from each other by using the font size, but it helps one to distinguish sections. Based on the HasPeriod feature, we can rec ognize that 100% of sections and subsections do not have a period at the end of the sentence, while contents sometimes have a period. The reason that the contents always do not have a pe riod is that many cases in contents do not make complete sentences. Based on the data distrib ution, we can understand that the feature vector may characterize the elements of a styled doc ument in an appropriate way.

Table 4. Ranking of Feature Vector Attributes

Ranking	Feature	Score
1	*FontSize*	0.51468
2	*isBold*	0.31289
3	*hasPeriod*	0.14248
4	*MultipleSentences*	0.08224
5	*IsNum*	0.07641
6	*isUnderLined*	0.01652
7	*Indentation*	0.00899
8	*Alignment*	0.0

Table 5. Distribution of Feature Vector Attributes

Feature	Value	Section(750)	SubSection(2,845)	Contents(1352)
Font Size	0.67 - 1.0	607 (81%)	398 (14%)	0 (0%)
	0.1 - 0.33	30 (4%)	1,480 (52%)	6,999 (82%)
	0.34 - 0.66	113 (15%)	967 (34%)	1,536 (18%)
isBold	`True`	682 (91%)	2,390 (84%)	1,280 (15%)
	`False`	68 (9%)	455 (16%)	7255 (85%)
Has Period	`True`	0(0%)	0 (0%)	4438 (52%)
	`False`	750(100%)	2845(100%)	4097(48%)

5. 2 Validating Classifiers

The second step experiment is to find out the best classifier with measuring precision, recall and F-measure of each classifier. We applied the following equations [15]:

$$\text{Precision} = \frac{|(Actual_e \cap Predicted_e)|}{|Predicted_e|} \quad (1)$$

$$\text{Recall} = \frac{|(Actual_e \cap Predicted_e)|}{|Actual_{id}|} \quad (2)$$

$$\text{F} - \text{Measure} = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \quad (3)$$

, where Predicted_e is the set of the document elements predicted by our approach, and Actual_e is the set of the actual document elements tagged manually for this experiment.

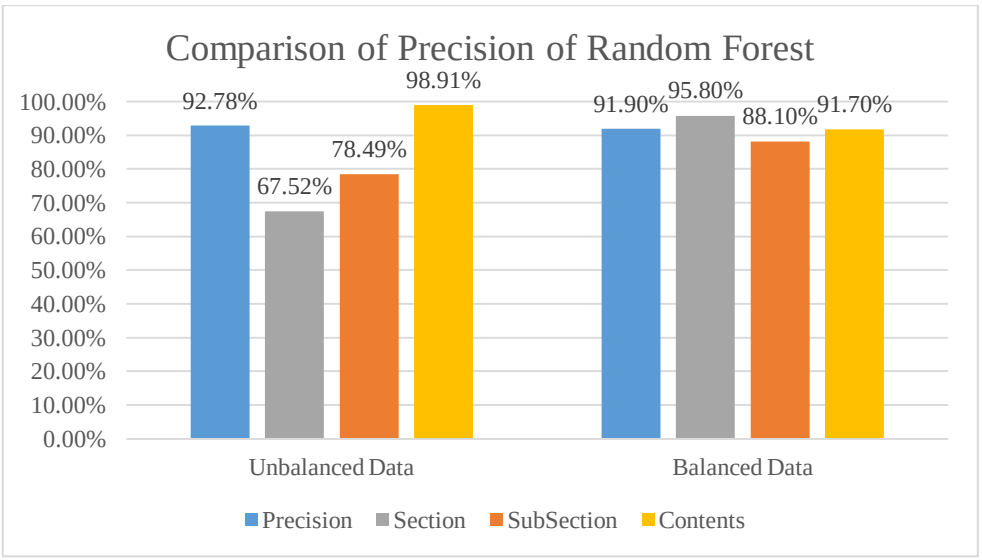
Table 6 shows the result of measuring precision and recall of the classifiers with the given data set. According to the result, the Random Forest classifier obtained the highest precision 92.78%, while the *SVM* classifier produces the lowest precision *88.65%*. *KNN* and *NB* got similar precisions respectively. For the recall and F-measure measures, the Random Forest algorithm also got the highest values.

Table 6. Precision, Recall and F-Measure of Machine Learning classifiers

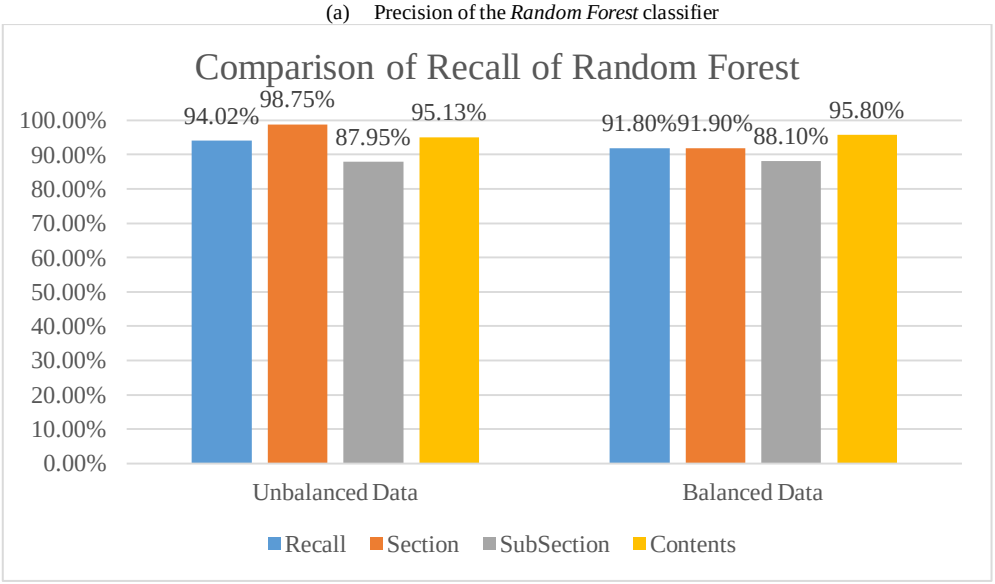
	Random Forest	SVM	KNN	NB
Precision	92.78%	*88.65%*	91.98%	91.46%
Section	67.52%	43.41%	67.52%	57.66%
SubSection	78.49%	80.00%	87.80%	76.19%
Contents	98.91%	96.39%	94.83%	99.14%
Recall	94.02%	89.84%	93.21%	92.68%
Section	98.75%	98.75%	98.75%	98.75%
SubSection	87.95%	61.24%	72.64%	83.39%
Contents	95.13%	95.88%	97.60%	94.46%
F-Measure	93.39 %	89.24%	92.59%	92.07%
Section	80.20%	60.31%	80.20%	72.81%
SubSection	82.95%	69.37%	79.50%	79.63%
Contents	96.98%	96.13%	96.20%	96.74%

While most of the classifiers got high precision for contents such as 98.91% of *Random Forest* and 99.14% of *Naïve Bayes*, they did not show high precision for classifying sections compared to that of contents. Even, the precision of *SVM* for sections got 43.41% of precision for the unbalanced data set. Thus, we randomly sampled a designated number of elements from the original elements of unbalanced data set, and made the balanced data as shown in **Table 3**. Given the balanced data set, we classified the data set with all classifiers again.

Fig. 4 shows comparison between precision/recall measure of evaluation with unbalanced and balanced data sets for the Random Forest algorithm, which has the best performance. According to the figure, about 0.88% of precision was decreased. However, when it comes to each element's precision, the precision of classification of sections was improved from 67.52% into 95.80% (+28.28%), and also 78.49% of precision for subsections was improved to 88.10% (+9.61%). Average precision for three elements was increased by about 10.23% from 81.64% ((67.52% + 78.49% + 95.91%)/3) to 91.87((95.8% + 88.1% + 91.7%)/3). Although the precision was increased, the recall of the balanced data was almost similar (i.e., 91.90% → 91.80%). Based on the balanced data set, we could obtain the feasible result for our approach.



(a) Precision of the *Random Forest* classifier



(b) Recall of the Random Forest classifier

Fig. 4. Precision and Recall of the Random Forest classifier with Unbalanced/Balanced Data Set

6. Conclusions

In this paper, we presented a machine learning based approach to analyzing the logical structure of styled documents composing of sections, subsections and contents. We showed a feature vector for analyzing the structure of a styled document composing of 8 attributes, and presented four classifiers to identify the elements. From our experiment, the Random Forest classifiers has the highest precision 92.78% and recall 94.02%. As future work, we have a plan to develop the document configuration management system for software Research & Development that can automatically identify document elements at run-time. Based on this research, the system can compare differences between two versions of a document, and analyze the semantic difference between two document elements.

Acknowledgement

This research was supported by the Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (NRF- 2014M3C4A7030503).

References

[1] R. Mohemad, A.R. Hamdan, Z.A. Othman, and N.M.M. Noor, "Automatic Document Structure Analysis of Structured PDF Files," *International Journal of New Computer Architectures and their Applications (IJNCAA)*, vol. 1, no. 2, pp. 404-411, August, 2011. Article (CrossRef Link).

[2] J. Kim, D. X. Le, and G.R. Thoma, "Automated labeling in document images," in *Proc. of SPIE Conference on Document Recognition and Retrieval VIII*, vol. 4307, pp. 111-122, January, 2001. Article (CrossRef Link).

[3] D. Niyogi and S.N. Srihari, "Knowledge-based derivation of document logical structure," in *Proc. of International Conference on Document Analysis and Recognition*, pp. 472-475, August 14 - 15, 1995. Article (CrossRef Link).

[4] R. Rauf, M. Antkiewicz, and K. Czarnecki, "Logical structure extraction from software requirements documents," in *Proc. of 19th IEEE International Requirements Engineering Conference*, pp. 101-110, August 29, 2011. Article (CrossRef Link).

[5] Kan, Min-Yen, Luong, Minh-Thang, "Logical Structure Recovery in Scholarly Articles with Rich Document Features," *International Journal of Digital Library Systems*, vol. 1, no. 4, pp. 1-23, October, 2010. Article (CrossRef Link).

[6] S. Mao, Z. Xu, T. Tjahjadi, and G. R. Thoma, "Logical Entity Recognition in Multi-Style Document Page Images," in *Proc. of 18th International Conference on Pattern Recognition*, pp. 876-879, August 20-24, 2006. Article (CrossRef Link).

[7] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, October, 2001.

[8] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, July, 1995. Article (CrossRef Link).

[9] David W. Aha, Dennis F. Kibler, Marc K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, pp. 37–66, January, 1991. Article (CrossRef Link).

[10] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd Edition, Prentice Hall, New Jersey, 2003.

[11] C. Bishop, *Pattern recognition and machine learning*, Springer, Berlin, 2006. Article (CrossRef Link).

[12] Weka Home Page. (Available at http://www.cs.waikato.ac.nz/ml/weka/).

[13] Docx4j Enterprise Edition Homepage. (Available at http://www.docx4java.org/trac/docx4j).

[14] T. Mitchell. *Machine Learning*, The Mc-Graw-Hill, New York, 1997. Article (CrossRef Link).

[15] W.B. Frakes and R. Baeza-Yates, *Information Retrival : Data Structures and Algorithms*, Prentice-Hall, New Jersey, 1992. Article (CrossRef Link).

[16] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proc. of the Eighteenth International Conference on Machine Learning*, pp. 282-289, June 28 - July 1, 2001. Article (CrossRef Link).

[17] I.H. Witten, E. Frank, and M.A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, Third Edition, Morgan Kaufmann, Burlington, 2011. Article (CrossRef Link).

[18] S. Klampfl, M. Granitzer, K. Jack and R. Kern, "Unsupervised document structure analysis of digital scientific articles," *International Journal on Digital Libraries*, vol. 14, Issue 3-4, pp. 83-99, August, 2014. Article (CrossRef Link).

[19] S. Klampfl and R. Kern, "Machine Learning Techniques for Automatically Extracting Contextual Information from Scientific Publications," *Semantic Web Evaluation Challenges - Second SemWebEval Challenge at ESWC 2015*, pp. 105-116 , May 31 - June 4, 2015. Article (CrossRef Link).

[20] S. Klampfl and R. Kern, “Reconstructing the logical structure of a Scientific Publication Using Machine Learning,” in *Proc. of Semantic Web Challenges - Third SemWebEval Challenge at ESWC 2016*, pp. 255-268, May 29 - June 2, 2016.
[21] J. Lafferty , A. McCallum and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. of the Eighteenth International Conference on Machine Learning*, pp. 282-289, June 28 - July 1, 2001.
[22] Ora Lassila, Ralph R. Swick, *Resource Description Framework (RDF) Model and Syntax Specification*. 1999. (Available at https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/).
[23] L. Liu and M. Tamer Ozsu, *Encyclopedia of Database Systems*. Springer, Berlin, 2009.
Article (CrossRef Link)

Tae-young Kim is currently in the master course in Department of Software Engineering at Chonbuk National University. He received the B.S. degree in the Department of Software Engineering at Chonbuk National University in 2016. His research focuses on code patterns, source code mining, and machine learning.

Suntae Kim is an Associate Professor of the Department of Software Engineering at Chonbuk National University. He received his B.S. degree in computer science and engineering from Chung-Ang University in 2003, and the M.S. Degree and PH.D. Degree in computer science and engineering from Sogang University in 2007 and 2010. He worked in Software Craft Co. Ltd., as a senior consultant and engineer for financial enterprise systems during 2002?2004. Also, he developed Android based Smart TV middleware from 2009 to 2010. His research focuses on software architecture, design patterns, requirements engineering, and source code mining.

Sangchul Choi is currently in the master course in Department of Software Engineering at Chonbuk National University. He received the B.S. degree in the Department of Software Engineering at Chonbuk National University in 2016. His research focuses on software process, source code mining, and machine learning.

Jong-Won Ko received the B.S., M.S., and Ph.D. degrees from Dept. of Computer Engineering, KyungHee University, Korea, in 2002, 2004, and 2012. His research interests include MDA, Model Based Test, Software Traceability, and Machine Learning.

Jee-Hyong Lee received his B.S., M.S., and Ph.D. in computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1993, 1995, and 1999, respectively. From 2000 to 2002, he was an international fellow at SRI International, USA. He joined Sungkyunkwan University, Suwon, Korea, as a faculty member in 2002. His research interests include fuzzy theory and application, intelligent systems, and machine learning.

Jeong Ah Kim received Ph. D degree at ChungAng University. Since 1996, she has worked at Catholic Kwandong University as professor. She is the member of Korea Institute of Information Science and Engineering and the member of board of directors of Convergent Research Society. Her research areas are software product line engineering, software modeling, software process improvement, clinical decision support system.

Jae-Young Choi is a professor with the department of computer engineering, college of software at the Sungkyunkwan University, Korea. He received his B.S. degree in mathematics in 1995, and the M.S. and Ph.D. degrees in computer science from the Kyungwon University, Korea, in 1999 and 2004, respectively. From 2004 to the middle of 2006, he joined the Vision Laboratory at the University of California, Los Angeles, USA, as a postdoctoral researcher. He has also served as a BK21 research professor at Kyungwon University from 2006 to 2010. His research interests include computer vision, machine learning, ubiquitous computing, network management, software engineering and R&D strategies.

Young-Hwa Cho is currently a distinguished visiting professor in the college of software at the Sungkyunkwan University, Korea. He received his B.S. degree in statistics and the M.S. degree in computer science from Sungkyunkwan University in 1977 and 1990, respectively. In 1999, he completed his Ph.D. degree in computer science from Chungbuk University, Korea. He was with KISTI(Korea Institute of Science and Technology Information) as a President from 2001 to 2006. He has also served as a President at KISTEP(Korea Institute of S&T Evaluation and Planning) from 2007 to 2008. He joined Kyungwon University as a visiting professor from 2008 to 2010. His current research interests cover software engineering, data base, information communication technology, R&D strategies and so on.