TSHD: Topic Segmentation Based on Headings Detection (Case Study: Resumes)

Abstract

Many unstructured documents contain segments with specific topics. Extracting these segments and identifying their topics helps to access the required information directly. This can improve the quality of many NLP applications such as information extraction, information retrieval, summarization, and question answering. Resumes (CVs) are unstructured documents that have diverse formats. They contain various segments such as personal information, experience, and education. Manually processing resumes to find the most suitable candidates for a particular job is a difficult task. Due to the increased amount of data, it has become very necessary to manipulate resumes by computer to save time and effort. This research presents a new algorithm named TSHD for topic segmentation based on headings detection. We apply the algorithm to extract resume segments and identify their topics. The proposed TSHD algorithm is accurate and addresses many weaknesses in previous studies. Evaluation results show a very high F1 score (about 96%) and a very low segmentation error (about 2%). The algorithm can be easily adapted to deal with other textual domains that contain headings in their segments.

1. Introduction

Most people create resumes containing information to describe and highlight everything they have ever done, in order to get a suitable job. Big companies receive a large number of resumes every day, which makes choosing the right candidate, for a specific job from a large number of applicants, very hard. It requires very often an expensive process, significant time, and human effort to do this task manually by employees.

Some companies have created electronic forms to be filled out by the applicant [
1]. But this solution requires some effort by the applicant, in addition to the problem of incorrect filling of the required data. Hence, there is still a need to develop new methods for processing resume documents and extracting their important information automatically. This can be done by applying new methods and techniques, such as those found in the fields of natural language processing (NLP), text mining, and machine learning.

Resumes are unstructured textual documents [
1], that follow different templates and formats. They include segments, such as Personal Information, Experience, Education, Skills, and others. The number and order of segments vary from one document to another.

The process of extracting segments with specific topics from the text is called segmentation or topic segmentation [
2].

The highly efficient extraction of segments from unstructured resumes and structuring them in an appropriate way is an important challenge. It has a great impact on improving the performance and accuracy of information extraction from resumes. In addition to allowing direct access to the required information from specific segments, and reducing the search time, it plays an important role in improving the quality of many resume-processing applications. Topic segmentation of resumes also facilitates the exploration and analysis of resumes information, and the comparison of necessary information between different resumes.

The importance of topic segmentation is highlighted in improving the accuracy of extracting the required information from specific segments instead of searching within the entire document [2]. For example: to extract the university from which the resume owner graduated, searching within the entire resume for a university may lead to identifying the university in which he works, or a university journal in which he published his research. Thus, searching within a specific segment with a known topic contributes to obtaining the required information with greater accuracy.

The research questions for this study are as follows:(i)How to exploit segment headings to extract document segments efficiently?(ii)Can segment headings be used to identify the segment topic? (iii)Can we structure the extracted segments with their topics in a suitable format?

This research aims to present a new algorithm for extracting and structuring document segments and identifying their topics with high efficiency. The algorithm has been applied in the resume domain. Additionally, many problems in previous studies have been addressed by the proposed algorithm.

The main contribution of this paper is the proposition of a new NLP algorithm named TSHD to extract document segments and identify their topics based on headings detection. The proposed algorithm is not affected by the difference in document templates, segment order, and font style. It can also be used to improve the quality of various NLP applications such as information extraction based on the extracted segments.

The rest of this paper is organized as follows: Section 2 gives an overview of related work. Section 3 presents the proposed algorithm. Evaluation results and discussions are given in Section 4. Finally, Section 5 provides conclusions and future work.

2. Related Work

For topic segmentation, many methods have been developed over the past years. Some methods deal with domain-independent documents, and others deal with domain-specific documents such as news articles, Wikipedia pages, novels, and resumes.

TextTiling algorithm introduced by Hearst in [3] makes use of lexical frequency distributions to determine the similarity and correlation between adjacent text blocks using the vector space model and cosine similarity. The C99 algorithm introduced by Choi in [4] uses a ranking scheme and the cosine similarity measure in formulating a similarity matrix of all sentence pairs. Then, it determines the location of topic boundaries by clustering. The U00 algorithm introduced by Utiyama and Isahara in [5] is a statistical method that finds the maximum-probability segmentation of a given text using dynamic programming. This method is more accurate than the C99 algorithm.

Pethe et al. in [6] presented a method for text segmentation in novels. They used a hybrid approach combining neural inference and regular expression-based rule matching to recognize chapter title headers in books and achieved an F1 score of 0.77 on this task. They presented cut-based and neural methods for chapter segmentation and achieved a low F1 score of 0.453.

Many studies proposed methods for topic identification, to identify the topics included in documents, or identify the topics of predefined segments. Topic modeling is one of the most popular ways of topic identification, which deals with domain-independent documents. Several models have been developed for topic identification using topic modeling such as latent Dirichlet allocation (LDA) [7] and structured topic model (STM) [8]. TopicTiling introduced by Riedl and Biemann in [9] is a modification of the TextTiling algorithm and makes use of LDA for topic modeling.

In the resume processing domain, many studies have several objectives for resume processing such as recruitment systems [

10, 11], information extraction [12], resume classification [13], and resume ranking [14]. Most studies extracted information from the whole document, which negatively affected the accuracy of results because of the inaccessibility of the required information directly. Because of that, new studies are more interested in topic segmentation to improve the accuracy of many applications such as those previously mentioned.

In the following, we review studies that presented methods for resume segmentation, explain their methods, and clarify their weaknesses.

Kessler et al. in [
15] indicated during their research, which represents an automatic recruitment system, that segmentation of text is a real issue, so they choose to use an existing tool called wvWare to split the paragraphs of the MS Word document. Then, they apply a corrective process in order to assign a correct label to each segment using Support Vector Machines (SVM). The main disadvantage of this system is the time cost resulting from the corrective process.

Sanyal et al. in [
16] applied a simple segmentation stage. They use a database or a data dictionary to hold the keywords or headings they find common in most of resumes. When a new resume is taken, a parser searches for the keywords and extracts all the data between them.

Yu et al. in [
17] applied some machine learning techniques to extract information from resumes in two stages. In the first stage, a resume is segmented into consecutive blocks attached with labels indicating the information types (general information segments) using Hidden Markov Models (HMM). Then in the second stage, some detailed information is extracted from the general information segments using Support Vector Machine (SVM). They evaluated the cascaded hybrid model with 1,200 Chinese resumes. In general information extraction, HMM achieved precision = 75.95% and recall = 75.89%, and SVM achieved precision = 80.95% and recall = 72.87%. This research supposes that using HMM to extract segments depends on the occurrence of general information in a fixed order. This hypothesis is not appropriate because the segment order often differs from one resume to another, and leads to the propagation of error to the second stage.

Reza and Zaman [
18] proposed a way to extract information from resumes by converting them from PDF to HTML using a Python library called urllib, then reverse engineering the HTML files to HTML code using a library called beautiful soap. An HTML code carries information about fonts like font size and font style. From this information, they try to detect segments. But this is not always appropriate because the font in resumes does not follow a specific standard. Therefore, some already existing resume headings were used by the segmentation process to enhance segmentation results. The weakness of this research is that the domain was kept restricted to the resumes of only engineering students and the amount of sample data versus the amount of test data was relatively small (to train and test the system only 50 resumes have been used) and their system achieved an accuracy of 80%–85%. In addition, resumes with some varied layout designs are not handled by their method.

Naive Bayes, C4.5, Random Forest, stacking, and conditional random field (CRF) are tested to classify resume lines into one of two classes, topic boundary, and nontopic boundary [
19]. The corpus was portioned into three datasets, training set (259 CVs), developer set (65 CVs), and test set (109 CVs). CRF achieved the best results compared to the other classifiers (precision = 80%, recall = 50%, and F1 score = 62%). But they do not address the variation of heading labels of the same segment between several resumes. Thus, the inability to extract similar segments between resumes causes the inability to access information from known topic segments directly.

Gunaseelan et al. in [
20] proposed a way to extract segments from resumes using supervised machine learning by training and testing several classifiers to predict whether a text line in a resume is a heading or not. After segmentation, they identify the topic only for the skills segment based on approximate string matching algorithm fuzzy-matching. XGBoost classifier outperformed the other classifiers used. It

achieved precision = 91.4%, recall = 89.8%, and F1 score = 90.1%. They have excluded the resumes containing text in some formats like lists and tables because it causes certain errors.

In this paper, we present a new algorithm for topic segmentation of documents with high efficiency. The algorithm has been applied in the resume domain. Compared with the related works, the proposed algorithm addresses several weaknesses in previous studies since it deals with resumes of people with different specialties and is not affected by the difference in resume templates, segment order, and font style; moreover, it applies topic identification to identify segment topics. The algorithm can be easily adapted to deal with other textual domains that contain headings for their segments.

3. TSHD Algorithm

TSHD stands for topic segmentation based on headings detection. Its main goal is extracting document segments and identifying their topics based on headings detection. TSHD uses the python NLTK package [
21] to implement some NLP tasks. We will start by reviewing (in Figure 1) the general architecture of the algorithm, followed by an explanation of each of its stages in detail. At the end of this section, an illustrative example will be presented.

The proposed algorithm consists of the following three main stages:(i)The first stage: preprocessing Preprocessing is the first stage of the algorithm, in which raw data (resumes) is preprocessed to prepare the data for the next stage. This stage consists of a series of six sequential steps (more details in Section 3.1).(ii)The second stage: headings detection At this stage, the locations of segment headings are determined, and similar segment heading labels are unified. This is done by doing two consecutive linear scans: cue phrases scan and then cue words scan (See Section 3.2).(iii)The third stage: segmentation Segmentation is the last stage of the algorithm, where segment headings and their contents are extracted and structured as JSON pairs.

3.1. Preprocessing

At this stage, raw data is preprocessed and refined, in order to be prepared for the next stage. The steps of this stage are shown in Figure 2.

3.1.1. Data Transformation

It is the process of converting text documents (resumes) from their various formats to text format, in order to be processed by computer. For example, converting docx to txt using the python package “docxpy” [
22].

3.1.2. Lines Tokenization

It is the process of dividing the text into a set of lines and storing them in a list. At this stage, we will treat the document as a list of lines.

3.1.3. Data Cleaning

It is the process of removing useless and misleading data, such as punctuation, bulleted and numbered lists, multiple spaces, and blank lines.

3.1.4. Normalization

It is the process of converting the entire text letters into a uniform case (lowercase), in order to standardize the differences between letter cases.

3.1.5. Lines Enumeration

In this step, an enumerate (denoted as *E*) is created from the list of lines resulting from the previous steps. It records pairs of (index, line) for all resume lines.

3.1.6. Lines Refinement

It is the process of removing nonheading lines from the enumerate *E* to create a refined enumerate of potential headings while keeping line indexes as they are in *E*. This aims to eliminate long lines from *E* because they are less likely to be potential headings.

Resumes include many long lines within their segment content, which may contain segment-heading keywords. Let us look at the following section of a resume: With over 6 years of *experience* in application developing for international companies. *Experience*. Employment at Maryland University. Teaching Assistant in the Department of Computer Science from 1/1/2017

Notice the keyword "experience," which is a normal word in the first line, but indicates also a heading of the experience segment. This word may cause a problem if the first line was indicated as the heading of the experience segment. To solve the problem, all lines with a word count higher than a predefined threshold denoted in this paper as **Th** (maximum length of potential headings) are removed from *E*. By experimentation and testing, the most appropriate **Th** should be chosen wisely (will be discussed later in Section 4.3). Segmentation results can be relatively improved, by applying more than one scan to the resume and increasing the threshold value in each scan. But this has a negative impact on computational and time processing costs.

The output of preprocessing stage is a refined enumerate, which contains a set of potential headings with their indexes after it has been cleaned, normalized, and refined. Then it passes as input to the next stage (Headings Detection Stage).

3.2. Headings Detection

At this stage, segment headings and their indexes are detected based on cue words/cue phrases. Then, similar heading labels are unified in order to identify the common segment topics between different resumes. Researchers in [
23] show that there are some words and phrases indicating changes in the topic, and can be considered as indicators of text segmentation to a set of segments, each with one topic. They are called cue words/cue phrases. In this paper, cue word refers to a keyword that is more likely to appear in segment headings. In the same way, a cue phrase refers to a phrase that is also more likely to appear in segment headings. As shown in Figure 3, the headings detection stage is divided into two consecutive linear scans.(i)First scan (cue phrases scan): segment headings are detected based on whether they contain a cue phrase.(ii)Second scan (cue words scan): segment headings are detected based on whether they contain a cue word.

Cue words scan alone is not enough because some segment headings do not contain cue words that can be used to identify these segments. Examples: personal data, professional background, and others. In addition, there are common and frequently used cue phrases. Example: work experience and educational qualification.

Furthermore, the point of applying two consecutive scans and applying cue phrases scan first—rather than searching for cue words/cue phrases in one scan—is because cue phrases scan results are more reliable. Therefore, we made its results unmodifiable by the second scan. Then followed by cue words scan to identify the remaining segment headings.

3.2.1. Headings Detection Main Tasks

Heading detection involves many tasks applied during the first and second scans for cue word/cue phrase detection.(i)Word Tokenization The process of splitting text into tokens, and storing them in a list. At this step, each line will be tokenized into a list of words using the "word tokenizer" provided by the NLTK package.(ii)Stemming In order to standardize the differences that may appear in some headings, "PorterStemmer" provided by NLTK is used to extract the roots of words, e.g., Skills word is returned to its root "skill"; education and educational words are returned to their root "educ," etc.(iii)N-gram tagging At this stage, within the first scan "cue phrases scan," an n-gram for $n = 2$, called 2-gram or bigram, is applied to generate all pairs of adjacent words, in order to match and identify segment headings that contain a cue phrase.(iv)Headings unification Heading labels of the same segment may differ from one resume to another, e.g., Education, Academia, Academic Qualifications, Academic Background, Educational Qualification, Educational Background, and Academic Credentials. All these labels refer to one segment heading, which is Education. The lack of standardization of these labels leads to the inability to explore similar segments between multiple resumes, and hence the inability to extract information from specific segments, rather than searching within the entire document. At this task, topic identification of segments is done by unifying similar segment headings. Headings representing segments about the same topic are grouped together in "unified heading" groups (personal_info, experience, education, skills, certifications, languages, awards, interest, summary, goal, military_service, additional_info, and others). For the previous example, all labels are assigned to the unified heading "education." The unified heading and its location are added to a Headings Dictionary when any of its synonyms are detected during scanning. The Headings Dictionary consists of {key : value} pairs and stores the unified headings and their locations {unified_heading: line_index}. Also, for headings that rarely appear or have few synonyms—which is useless to unify their synonyms–they are referred to as "others," such as references, projects, and membership, and add their root as a heading to the Headings Dictionary. Tables 1 and 2 show the most commonly used cue words/cue phrases, collected from various sources. Their synonyms are grouped and unified. The roots of words are stored for the purposes that were previously explained.

The algorithm can deal with other textual domains that contain headings for their segments such as scientific research publications, reports, online articles, and memorandums. This can be done by adapting Tables 1 and 2 with cue phrases and cue words belonging to another domain.

3.2.2. Dealing with Special Cases

In the following, it is shown a number of special cases that are likely to appear in various resumes and the way to deal with them.(i)Subheadings Segment contents may include some subheadings that may have the same cue word/cue phrase found in the segment heading, which should not be identified as a segment heading. To illustrate, look at the following section of a resume: Skills Programming Skills: Python-Java-C++ Software Skills: Linux-Windows Notice that the cue word "skills" appears three times. The first occurrence represents the segment heading, while the next two occurrences represent subheadings within the segment content. This is resolved, by recording the first occurrence of cue words/cue phrases and making it unmodifiable if it appears again later. (ii)Recurrence of cue words/cue phrases or their synonyms Cue words/cue phrases that identify a segment heading or one of their synonyms may recur within the segment content. The example is as follows: Experience Employment at Maryland University Teaching Assistant in the Department of Computer Science from 1/1/2017 Notice that the cue word "employment," which is one of the synonyms that identify the heading of the experience segment, appears within the content of the experience segment, assuming that the line containing it—a short line—is not excluded in the lines refinement process. This is resolved, by taking advantage of unifying similar headings, recording the first occurrence of cue words/cue phrases, and making them unmodifiable if any of their synonyms recur later.(iii)Composite segments Some resumes may include segments with close topics, e.g., Certifications, Awards, Honors, and Achievements. Some resume owners may group them into one composite segment such as "Certifications and Awards," "Awards/Achievements," or "Honors, Certifications, and Awards." Such cue words should not be considered as synonyms of a single segment heading and unifying them because they may appear as independent segments. The following section of a resume represents a composite segment: Certifications and Awards.

International Computer Drivers License (ICDL) ACM ICPC 2019 Gold medal. In this case, an error will occur if these cue words are identified as different segment headings, e.g., “Certifications” is identified as a segment heading and “Awards” as a new segment heading. To avoid this problem, the index of the segment heading is checked to assure it is not already added to the Headings Dictionary.

3.2.3. Headings Detection Algorithm

The refined enumerate generated by the previous stage is passed as input to this stage. At this stage, the Headings Dictionary is passed to the first scan and then to the second scan in order to identify headings according to the linear scan order.

(1) First scan: “cue phrases scan.” In this scan, the algorithm identifies the location of headings that contain cue phrases, unifies similar heading labels, and adds the unified headings with their locations to the Headings Dictionary (See. Algorithm 1 for pseudocode).

(1) **Data**
(2) *D*: Headings Dictionary
(3) **Input**
(4) RE: Refined Enumerate
(5) **Output**
(6) *D* is passed to Algorithm 2
(7) **for all** (lineIndex, line) ∈ RE **do**
(8) words tokenization of line
(9) create bigrams_list of words
(10) **for all** ($word_i$, $word_{i+1}$) ∈ bigrams_list **do**
(11) stemming of ($word_i$, $word_{i+1}$)
(12) match them to CuePhrases_table
(13) **if** matches **then**
(14) get unifiedHeading
(15) record lineIndex
(16) **if** unifiedHeading ∉ *D* **then**
(17) add {unifiedHeading:lineIndex} to *D*
(18) **end if**
(19) **else**
(20) continue
(21) **end if**
(22) **end for**
(23) **end for**

Algorithm 1
Cue phrases scan.

(2) Second scan: “cue words scan.” In this scan, the algorithm identifies the location of remaining headings that contain cue words, unifies similar heading labels, and adds the unified headings with their locations to the Headings Dictionary (See. Algorithm 2 for pseudocode).

(1) **Input**
(2) RE: Refined Enumerate

```
(3)     D: Headings Dictionary (the output of Algorithm 1)
(4)  Output
(5)     D after adding the unified headings with their locations
(6)  for all (lineIndex, line) ∈ RE do
(7)     words tokenization of line
(8)     for all word ∈ words_list do
(9)         stemming of word
(10)        match it to CueWords_table
(11)        if matches then
(12)            get unifiedHeading
(13)            record lineIndex
(14)            if unifiedHeading ≠ others then
(15)                if unifiedHeading ∉D and lineIndex ∉D then
(16)                    add {unifiedHeading: lineIndex} to D
(17)                end if
(18)            else
(19)                if stem(word) ∉D and lineIndex ∉D then
(20)                    add {stem(word): lineIndex} to D
(21)                end if
(22)            end if
(23)        else
(24)            continue
(25)        end if
(26)    end for
(27) end for
```

Algorithm 2
Cue words scan.

At the end of this stage, the Headings Dictionary items are resorted in ascending order according to heading locations and then passed to the next stage “segmentation stage.”

3.3. Segmentation

This is the last stage of the algorithm, where segment headings and their contents are extracted, based on the Headings Dictionary resulting from the previous stage, and the raw lines of resume “line tokens” (before any modifications are made to them), as shown in Figure 4.

In the following, the method of extracting segment headings and their contents from resumes is presented. Then, how to implement this stage of the algorithm is explained.

3.3.1. Extraction of the First Segment of Resumes

Many resumes include at the beginning a segment that includes at least the name of the resume owner and may include his specialty or a summary of the resume. This part of the resume does not usually have a heading. As in this example. John Smith Software Engineer With over 6 years of experience in application developing for international companies.

The first segment of the resume is extracted by extracting the part between the beginning of the resume and the first segment heading. Then, it is added to the personal information segment.

3.3.2. Extraction of Segment Headings and Their Contents

This is done by associating unified headings in Headings Dictionary with their contents in line tokens, depending on the locations of these headings that were detected in the previous stage, then extracting the parts between them. Figure 5 shows the method of extracting segment headings and their contents.

At the end of this stage, the unified segment headings and their contents are stored and structured as JSON pairs.

3.4. An Illustrative Example

Table 3 presents an illustrative example of a resume and shows the output of each stage of the TSHD algorithm.
Table 3
An illustrative example shows the output of each stage of the TSHD algorithm.

4. Results and Discussion

This section presents an evaluation and a discussion concerning TSHD algorithm results after applying it to a real set of resumes. Then it shows how effective the TSHD is by comparing it with the most prominent previous studies.

4.1. Dataset Description

In order to test the proposed algorithm, 105 resumes (containing 4733 lines after deleting the blank lines) written in English were collected from random websites. They belong to people with different specialties. These resumes are unstructured text documents and follow different templates. Their segments vary in number and order. Also, the style, size, and color of the font do not follow any standards.

4.2. Evaluation Metrics

There are two types of metrics used in evaluating segmentation results; classification-based and segmentation-based metrics (explained in detail in the following).

4.2.1. Classification-Based Metrics

Segmentation can be considered as a binary classification task by classifying all resume lines into the following two classes:(i)Topic boundary: this class represents a line separating two segments with two different topics. In this study, it is the segment heading.(ii)Nontopic boundary: this class represents a line within the segment content.

Therefore, the most important classification metrics for binary classification are [
24]as follows: True positive (TP): number of segment headings that are correctly identified by the algorithm. False negative (FN): number of segment headings that are incorrectly not identified by the algorithm. $r \quad ision$ 2 False positive (FP): number of lines that are incorrectly identified as segment headings by the algorithm.

$$1_{score} \quad \frac{2 \times \quad r \quad ision \times}{r \quad ision},$$

$$A \quad ur \quad y$$

$$1_{\text{score}} \quad \frac{2 \times \quad r \quad \text{ision} \times}{r \quad \text{ision}},$$

$$A \quad \text{ur} \quad y \qquad\qquad 3$$

3 True negative (TN): number of lines that are not segment headings and correctly identified by the algorithm.

Classification metrics are strict in their decision because they do not consider how close the identified topic boundary is to the actual topic boundary for the false results.

4.2.2. Segmentation-Based Metrics

(i)P_k For the segmentation task, P_k gives the probabilistic error metric in segmentation. It indicates the probability of points (lines) being identified in the wrong segments. It tests whether or not lines are separated by segment breaks [

×

25. D. Beeferman, A. Berger, and J. Lafferty, "Statistical models for text segmentation," *Machine Learning*, vol. 34, no. 1/3, pp. 177–210, 1999.
View at: Publisher Site | Google Scholar

See in References

25]. P_k values range from zero to one where smaller values are better. The value of P_k is calculated by passing a constant-sized window of k words across the text. In each step, both ends of the window are tested for hypothesized segments, to determine if they are actually separated by a segment break. The value of k can be chosen randomly, but in general, P_k is evaluated by fixing k to be half of the average reference segment length.(ii)WindowDiff (WD) Researchers propose the WindowDiff metric, a simple modification to the P_k metric, to avoid several problems [

×

26. L. Pevzner and M. A. Hearst, "A critique and improvement of an evaluation metric for text segmentation," *Computational Linguistics*, vol. 28, no. 1, pp. 19–36, 2002.
View at: Publisher Site | Google Scholar

See in References

26]. WindowDiff moves a fixed-sized window across the text and penalizes the algorithm whenever the number of boundaries within the window does not match the true number of boundaries for that window of text. WD values range from zero to one (smaller is better).

4.3. Evaluation Results

Classification and segmentation metrics are applied to evaluate TSHD results. The algorithm is implemented and tested for several values of threshold **Th** (maximum length of lines), which are indicated in the lines refinement stage (See. Section 3.1.6). Tested thresholds are 1, 2, 3, 4, 5, 6, 7, and all lines without refinement.

Table 4 and Figure 6 show classification metrics results to evaluate TSHD for several thresholds.

Notice that values of F1 score and accuracy are very high for thresholds 2, 3, and 4. TSHD achieves the best results when **Th** = 3 words, with F1 score = 0.964, and accuracy = 0.992.

Segmentation metrics P_k and WD, are used to evaluate the algorithm for several thresholds. They calculate the segmentation probabilistic error. P_k is obtained by fixing k to be half of the average

reference segment length. WD metric is calculated for the most common window widths: 2, 3, and 4. Results are shown in Table 5 and Figure 7.

Notice that all segmentation metrics achieve a very low segmentation error rate for thresholds 2, 3, and 4. TSHD achieves the best results when **Th** = 3 words. At this threshold $P_k = 0.022$, and WD values equals (0.016, 0.024, and 0.032) for Window widths (2, 3, 4), respectively.

4.3.1. Results Discussion

As shown in Figure 6, the higher the threshold value, the lower the precision value due to increasing the number of lines that are incorrectly identified as segment headings (the false positive). Because the longer the line is, the more likely it is to be a segmented content and the less likely it is to be a segment heading; moreover, the lower the threshold value, the lower the recall value because of increasing the number of segment headings that are incorrectly not identified (the false negative). It is due to the failure to identify segment headings whose length is greater than the specified threshold. Recall value also decreases when all lines are taken, as a result of not identifying segment headings if one of their synonyms has occurred inside a segmented content before because the algorithm determines the first occurrence of cue words/cue phrases.

As shown in Table 4, the accuracy metric is very high, which is close to 99%, because of the high value of true negative (number of lines that are not segment headings and correctly identified by the algorithm). As shown in Figure 7, the different values of window width in the WD metric reflect the same evaluation results, and increasing the window width does not give better results. Therefore, any value of window width can be used, which gave the best results when window width = 2.

The lines refinement stage has a significant positive effect in improving F1-score results from 74% to 96% (See. Figure 6) and reducing segmentation error from 19% to 2% (See. Figure 7); Consequently, all classification and segmentation metrics that have been applied show the effectiveness of the TSHD algorithm. Classification metrics show a very high F1 score (about 96%), and segmentation metrics show a very low segmentation error (about 2%).

4.3.2. Results Comparison

In the following, the results of the most prominent studies in resume segmentation are presented. Researchers have evaluated their methods after applying them to different data sets. It is not possible to compare their results accurately because the same data set is unavailable. However, the TSHD algorithm deals with various forms of resumes, and the data set that is used to evaluate the algorithm does not follow any standards and has been randomly collected from multiple sources. Thus, we will compare the evaluation results of the algorithm with the results of those studies, regardless of any potential considerations in the data set used by them.

Researchers in [
17] test HMM and SVM models to extract segments from resumes. On the other hand, researchers in [19] test several classifiers and get the highest results using conditional random field (CRF). Researchers in [20] apply the XG boost classifier in predicting headings. They evaluate their results using precision, recall, and F1 score. Table 6 shows their evaluation results in comparison with the results of the TSHD algorithm in this paper for $Th = 3$ words.

5. Conclusions and Future Work

In this paper, we presented a new algorithm named TSHD for extracting document segments and identifying their topics based on headings detection. We have applied the algorithm to extract resume segments, in order to facilitate selecting suitable candidates for a particular job post. The study focuses on improving the segmentation accuracy, due to its significant impact on improving many NLP applications. This helps in accessing the required information from specific segments

directly, instead of searching within the entire document. Also, the algorithm addresses many weaknesses in previous studies.

The proposed algorithm deals with various forms of unstructured resumes of candidates with different specializations. It extracts segments, identifies their topics, and structures them in JSON format. Then, the structured segments can be used for several purposes, including(i)Improving the quality of many applications that process and analyze resumes with different objectives(ii)Saving time and human effort by the direct access to the required information and the ability to compare it(iii)Identify the available information in resumes. For example, reviewing resumes of candidates who have publications and awards(iv)The ability to reformat resumes with different templates and print them using one standard template

The algorithm was evaluated using classification and segmentation metrics and tested for several values of threshold (maximum length of lines). The evaluation results show the effectiveness of the algorithm, which achieves the best results for threshold values 2, 3, and 4. It achieves a very high F1 score (about 96%), and a very low segmentation error (about 2%). The evaluation results were compared with the most prominent similar studies, which showed the clear superiority of the TSHD algorithm. The algorithm can be easily adapted to deal with other textual domains that contain headings for their segments. This can be done by using cue words and cue phrases belonging to that domain and choosing the appropriate threshold.

In future work, it would be interesting to test our algorithm on different domains such as research publications. Furthermore, it will be also interesting to try building the cue words and cue phrases tables automatically using machine learning or deep learning methods. We expect that the high-quality segmentation that is achieved by our algorithm can improve many NLP applications such as information extraction, information retrieval, and others. However, the proposed algorithm is not applicable to documents that do not contain headings for their segments.

Data Availability

All data included in this study are available from the first author upon reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Al-Baath University and Al-Wataniya Private University.