Matching Complex Documents Using Tolerance Rough Set Model

Nguyen T.B.[†] **Ho T.B.**[†] **Pham C.**[‡] **Kawasaki S.**[†]
[†]School of Knowledge Science
Japan Advanced Institute of Science and Technology
{thaibinh,bao,skawasa}@jaist.ac.jp
[‡]Cisco System, Inc., San Jose, California, USA
chpham@cisco.com

Abstract

Matching documents is to find similar ones of a given document from an existing document database. It is an important problem in practice as well as in research. This paper presents a method for matching document applying to identify duplicate errors in network systems. The database we used is CISCO network errors database (CNED), a collection of documents that describe error cases occurred in network systems. Each document is a complex document composed of several parts in different types: plain text, source code, dump code, debug, etc. Our experiments are carried out for CNED but the method can be extended to many other kinds of complex documents.

Keywords: Document matching, complex document, information retrieval, tolerance rough set model.

1 Introduction

Document matching has always been an important application and an attractive research topic. Documents can be divided into two types: plain text documents and complex documents. A plain text document contains only plain text, while a complex document contains not only plain text but also several parts in different forms, e.g., symbols, formulas, source code, XML, etc.

One real life example of complex documents is the collection of documents about network errors, called the CISCO network error database (CNED), provided by CISCO. Each document in the CNED contains information about a customer's network error such as: *Network topology*, *Configuration*, *Analysis information*, *source code*, etc. The information (i.e., data) contains network domain knowledge and is in various types: natural language text, machine generated code, source code, etc.

Other examples about complex documents can be raised are hospital records or company product error records [1].

In the example of the CNED above, when an error occurs in the network of a customer, the technical staffs have to create this error's document and try to find the solution for this error. An algorithm for matching error's documents would help the technical staff identify the errors occurred in the past which have the same root cause with the newly occurred error. It makes more quickly for the the technical staffs to find the root cause of the error.

There are many studies related to document matching and information retrieval from text documents. In [2], the authors presented a method for finding similar text files in large documents repositories. The method is based on chunking the byte stream to find unique signatures that may be shared in multiple files. In [3], the authors introduced a Bayesian framework for text case matching for Case-based Reasoning. In [4], Zhao et al. introduced a method for text mining using class association rules. However, those methods either are limited to their specific data or only concentrate on matching algorithm after representing the documents as feature vectors.

The objective of our research is to propose a general solution for the matching complex documents in the CNED. The problem can be defined as following: "*given a document of a newly occurred network error (the query document), find in the CNED the documents of errors that have the same root cause with the newly occurred error*". From here, we call the documents of the same root cause errors, as *duplicate documents*.

In this paper, we analyze the data and propose a general solution for the problem. Then we show a partial solution for the problem which matches two documents by matching only their natural language parts. In this partial solution,

from each natural language part, only 6-8 keywords were extracted. With this small number of keywords, current available similarity measures [5] often yield zero-values and the matching accuracy is decreased. To overcome this limitation, we employed the Tolerance rough set model (TRSM) [6] for representing text. TRSM, an extension of rough set theory [7], represents a text using not only its keywords but other keywords that have the relationship in context with its keywords. Therefore, the set of keywords representing a text is enriched, resulting in the increase of matching accuracy.

The rest of this paper is organized as follows: In section 2, we show more detail about the CNED and its documents. In section 3 we present a general solution for the matching problem. Section 4 shows our partial solution and its result. Finally, in section 5 we summarize the research and show our future work.

2 CISCO network error database

The CISCO network error database (CNED) is a collection of more than million documents about customers' network errors. Each document contains information about the error such as: network topology, network configuration, error description, analysis information, etc. The updated information via email such as source codes, dump codes is also included in the record.

Commonly, each record is organized by some components as follows:

- **Topology**: shows how hosts are connected
- **Error description**: briefly describe the error such as the symptoms, the circumstance of the network when the error occurred, etc.
- **Configuration**: show the configuration of the network such as network protocols, encapsulations, interfaces, etc.
- **Analysis information**: the results of executing commands.

An example of a document in CNED is shown in Figure 1.

Following are some properties of the documents that make the matching process difficult:

- The documents contain network domain knowledge that is not easy to exploit and represent in computer

```
1. Topology
==========

      Generator              UnitUnderTest              Reflector
                                (UUT)
   +------------+        +-----------------+        +-------------+
   |            |    |   |                 |        |             |
   |    agsk1 e1|----|---|e4   agsk2    s3|--------|s0  agsm3    |
   |            |    |   |                 |        |             |
   +------------+        +-----------------+        +-------------+


2. Summary of failed testcases
============================

      clns_static, clns_igrp, and clns_isis switching tests fail with
      102-8.4 version of images. Output of 'show clns neighbors' doesn't
      include the complete information when Serial interface with
      fr-ietf enacpaulation is configured in the switching path.

      Images: gs3-k.102-8.4, igs-bpx-l.102-8.4

      The following testcase is presented here:
switching/clns_static clns_static.0130201121241 mec N/A noencap slow mci_s 1 fr-ietf slow
Ethernet1 Ethernet4 Serial3 Serial0

      Generator <-------------> UUT <-------------> Reflector
              e1   arpa      e4     s3  fr-ietf    s0

3. Configuration
================
agsk1
=====
agsk1(config)#clns routing
agsk1(config)#clns net 47.0004.004D.0055.0000.0C00.0000.00
agsk1(config)#!
agsk1(config)#interface Ethernet1
agsk1(config-if)#clns enable

agsk2
=====
agsk2(config)#clns routing
agsk2(config)#clns net 47.0004.004D.0055.0000.0C00.0001.00
agsk2(config)#!
agsk2(config)#interface Ethernet4
agsk2(config-if)#clns enable
agsk2(config-if)#! ring-speed 16
agsk2(config-if)#no clns route-cache

4. Analysis
===========
show inter Ethernet1
Ethernet1 is up, line protocol is up
  Hardware is MCI Ethernet, address is 0000.0c02.cb98 (bia 0000.0c02.cb98)
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 4:00:00
  Last input 0:00:02, output 0:01:56, output hang never
  Last clearing of "show interface" counters never
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2 packets input, 178 bytes, 0 no buffer
     Received 1 broadcasts, 0 runts, 0 giants
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 input packets with dribble condition detected
     2 packets output, 195 bytes, 0 underruns
     0 output errors, 0 collisions, 3 interface resets, 0 restarts
```

Figure 1: Example of a document

- Information in the documents is in various types: topology is in the form of a graph, error description is in natural language, configuration and analysis information are machine generated codes, etc.
- The organizations of the documents (number of components, order of the components) are different and not well-formatted relying on the people who created it

The documents in the CNED are already divided into groups. Documents in each group are those of the same root cause errors. Each document belongs to only one group.

3 Methodology

The general method contains two main steps:

- **Data preparation**: extracting information

from the original document to fill in a pre-determined template

- **Document matching**: matching two filled template

The framework is shown in Figure 2

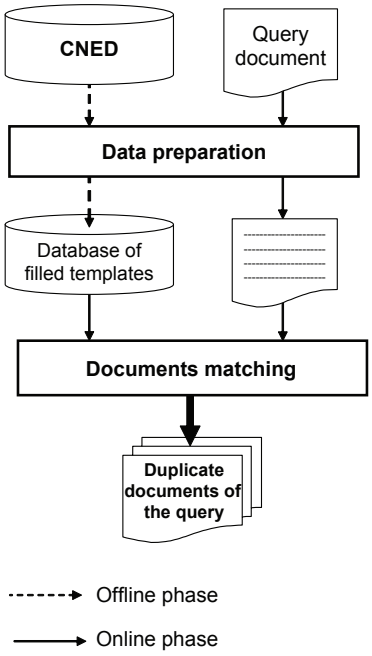

Figure 2: Framework of the method

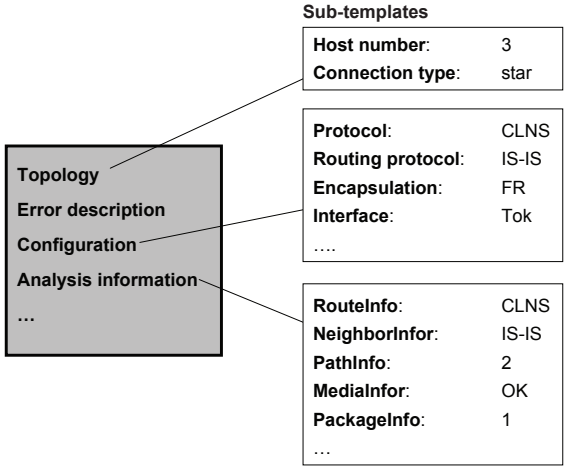

Figure 3: Template consisting sub-templates

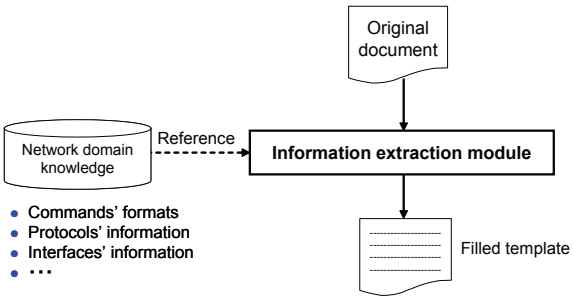

Figure 4: Information Extraction module

3.1 Data preparation

Data preparation is to extract information from each document to fill in the pre-determined template.

The template contains several slots, such as *Topology*, *Error description*, *Configuration*, *Analysis information*, etc. The value for the Error description slot is simply plain text, while those of other slots (e.g.Topology, Configuration, Analysis information) are represented using sub-templates with their own slots. The structure of the template with sub-templates is shown in Figure 3.

There are some researches related to information extracting in natural language text or from websites [8, 9, 10, 11]. Those methods use Natural language processing techniques to extract entities and their mentions from natural language text as well as HTML tags analysis techniques to extract information from web tables. However, in our data, each document contains not only plain text but also various type of information including network domain knowledge. Therefore, to extract information from the documents, it is necessary to create a Domain knowledge base for referencing. The Domain knowledge base should include information so that the Information Extraction module can use to know how to extract information from the documents. These information includes Commands' syntaxes, Formats of the commands executing results, Interface information, etc. The Information Extraction module is shown in Figure 4.

The Data preparation for the documents in CNED is implemented in an offline phase to create a database of filled templates. The Data preparation for the new document is executed in an online phase to fill in its template. The document matching phase in the next section will compare the new document with all the documents in the database by comparing corresponding information in the templates.

3.2 Documents matching

The Document matching phase computes the distance between the new document with all documents in the CNED and ranks the documents in the CNED based on such distances.

The distance between the new document and each document in the CNED is calculated through the distance between the values of corresponding slots in the two templates. For the simplicity to understand, we assume that there are only 4 components in each document: Topology, Error description, Configuration, Analysis information. The distance between two documents d_i, d_j can be written as in equation 1.

$$D(d_1, d_2) = \alpha_1 TP + \alpha_2 ED + \alpha_3 CF + \alpha_4 AI \quad (1)$$

TP, ED, CF, AI: are the distances between the two corresponding Topologies, Error descriptions, Configurations Analysis informations, respectively.

The error description is in natural language, so ED is calculated matching plain text documents.

TP, CF, AI, in turn, are computed through the distance between the values in the corresponding sub-slot of the sub-templates.

$$\begin{aligned} TP &= \beta_1 TP_1 + \beta_2 TP_2 + \beta_3 TP_3 + \cdots \\ CF &= \gamma_1 CF_1 + \gamma_2 CF_2 + \gamma_3 CF_3 + \cdots \\ AI &= \delta_1 AI_1 + \delta_2 AI_2 + \delta_3 AI_3 + \cdots \end{aligned} \quad (2)$$

The values of $\alpha_i, \beta_i, \gamma_i, \delta_i$ will be learned by supervised machine learning techniques using sample documents in the CNED. The distance metric should make the documents of the same groups close, and separate documents of different groups.

4 A partial solution

We investigated thoroughly the database and realized that among the components of the document, the error descriptions of the same root cause errors often share some keywords related to the error. Beside, they have following features: 1) Compact and the contents are highly reliable 2) In natural language

As a preliminary solution, we did an experiment using only the error descriptions in matching two documents. Then, the similarity of two document given in (1) becomes the similarity of the two Error descriptions:

$$D(d_1, d_2) = ED \quad (3)$$

The framework for this partial solution is shown in Figure 5.

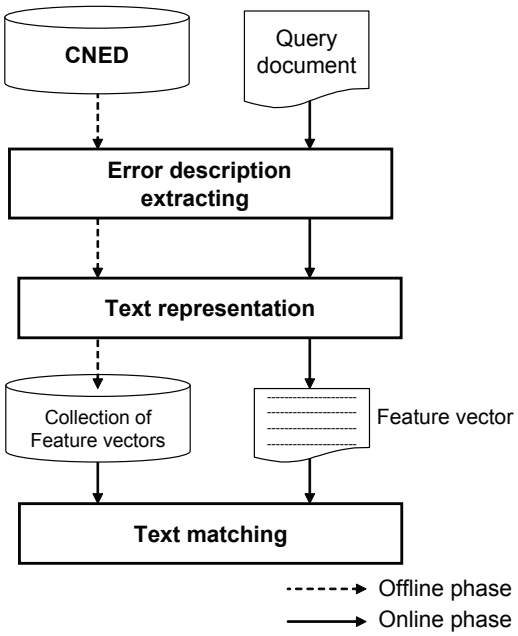

Figure 5: The framework of the partial solution

As shown in Figure 5, the Error description extraction phase for the documents in CNED is implemented in an offline process to create a database of Error descriptions. The Error description extraction for the new document is executed in an online process.

4.1 Extracting error description

The Error descriptions often begin with some signs such as: “Error description”, “Error summary”, etc. Therefore, the *Error description module* simply searchs for these signs and extract the Error descriptions.

4.2 Representing text

4.2.1 Traditional bag-of-words method

Traditionally, the bag-of-words method is used for presenting natural language documents. A document is represented by a vector of keywords extracted from the document, with associated weights with respect to the importance of the keywords in the document and within the whole document collection.

Each Error description is a natural language document, and in this section, we call it a document. We denote

$E = \{e_1, e_2, ..., e_M\}$: the set of all documents (Error descriptions)

$T = \{t_1, t_2, ..., t_N\}$: the set of keywords extracted from the documents in E.

The weight of a term in a document is often determined using the so-called TF.IDF in which the weight of a term is determined by two factor: how often the keywords t_i occurs in the document e_j and how often it occurs in the whole documents collection, as in equation 4 [12]:

$$w_{ij} = f_{e_j}(t_i) \log \frac{M}{f_E(t_i)} \quad (4)$$

$f_{e_j}(t_i)$: number of occurrences of t_i in e_j
$f_E(t_i)$: total number of occurrences of t_i in E.

With the bag-of-words methods, we could extract only 6-8 keywords from each document. With this few keywords, the common measure similarities often yield zero-value, and the matching quality is not so good. Table 1 is an example of 5 documents with there keywords.

No.	***Keywords characterizing documents***
e_1	clns, isis, route, across, link, frame_relay, encapsulation
e_2	learn_path, net, isis
e_3	iso_igrp, updates, frame_relay, ietf, encapsulation
e_4	frame_relay, ietf, atm-dxi, ad-ietf, encapsulation, clns
e_5	clns, route, learn_path, iso_igrp

Table 1: Keywords of first 5 documents

To overcome this limitation, we used the Tolerance rough set model (TRSM) [6], a model for representing text, which consider the relationship in context between keywords. In this model, a document e_j is represented not only by its keywords, but also other keywords which have the relationship with its keywords. With this extension, the number of keywords for representing e_j is enlarged, resulting in the increase of matching performance.

The Error description extracting and Text representing for the documents in CNED are done offline to make a collection of feature vectors. That process for the new document is done online to make its feature vector.

4.2.2 Tolerance rough set model

As mentioned in 4.2.1, the TRSM represents e_j by not only keywords in e_j but other keywords which have relationship with its keywords. The relationship used here is the co-occurrence relation. It means that there exists a relationship between t_i and t_j if the number of co-occurrence of t_i and t_j is larger than a threshold.

For each keyword t_i, define its tolerance rough set $R(t_i)$ as the set of keywords that have number of co-occurrence with this keyword is large than a threshold θ, including itself.

$$R(t_i) = \{t_j \in T | f_E(t_i, t_j) \geq \theta\} \cup t_i \quad (5)$$

$f_E(t_i, t_j)$: total number of co-occurrences of t_i and t_j in E.

Then each document e_j in the set E will be represented by its keywords and the keywords in the tolerance rough set of these keywords. In this case, the set that represents the e_j is called its upper-approximation, and denoted by $U(e_j)$.

$$U(e_j) = \bigcup_{t_k \in e_j} R(t_k) \quad (6)$$

The weigh of keyword t_i in e_j is extended into the following formula:

$$w_{ij} = \begin{cases} (1 + \log(f_{e_j}(t_i))) \log \frac{M}{f_E(t_i)} & \text{if } t_i \in e_j \\ min_{t_h \in e_j} w_{hj} \frac{\log \frac{M}{f_E(t_i)}}{1 + \log \frac{M}{f_E(t_i)}} & \text{if } t_i \in U \backslash d_j \\ 0 & \text{if } t_i \notin U \end{cases} \quad (7)$$

in which, $U = U(e_j)$.

This way of weighting ensures that each term of $U(e_j)$ but not appears in e_j has a weight smaller than any term appearing in e_j. And the keywords that not appears in the $U(e_j)$ are weighted 0.

Table 2 shows an example of 5 documents and there upper-approximations.

4.3 Matching text

Many similarity measures between documents can be used in TRSM matching algorithm. We used two common coefficients of Dice, and Cosine [13, 5] for our TRSM matching algorithm to calculate the similarity between pairs of documents e_i and e_j . For example, the Dice coefficient is:

$$S_D(e_i, e_j) = \frac{2 \sum_{k=1}^{N} (w_{ki} \times w_{kj})}{\sum_{k=1}^{N} (w_{ki}^2 + w_{kj}^2)} \quad (8)$$

No.	***Keywords***	***U(d_i)***
e_1	$t_{180},\ t_{44},\ t_{32},\ t_{113},\ t_{108},\ t_{61}, t_{12}$	$t_{180}, t_{44}, t_{32}, t_{113}, t_{108}, t_{61}, t_{12}, t_5, t_7, t_9, t_{22}$
e_2	t_{168}, t_5, t_{113}	$t_{168}, t_5, t_{113}, t_{12}$
e_3	$t_{87}, t_{91}, t_{32}, t_{19}, t_{61}$	$t_{87}, t_{91}, t_{32}, t_{19}, t_{61}$
e_4	$t_{91}, t_{57}, t_{32}, t_{19}, t_{108}, t_{61}$	$t_{91}, t_{57}, t_{32}, t_{19}, t_{108}, t_{61}$
e_5	$t_{87}, t_5, t_{108}, t_{12}$	$t_{87}, t_5, t_{108}, t_{12}, t_7, t_9, t_{11}, t_{37}$

Table 2: Approximations of first 5 documents

It is worth to note that most similarity measures for documents [13, 5] yields a large number of zero values when documents are represented by only a few terms as many of them may have no terms in common.

Similarity measures above will be applied to the upper-approximation $U(e_j)$. Two main advantages of using upper approximation are: i) To reduce the number of zero-valued coefficients by considering documents themselves together with the related terms in tolerance classes, and ii) The upper approximations formed by tolerance classes make it possible to relate documents that may have few terms in common with the query.

4.4 Experiment and Evaluation

4.4.1 Experimental design

The purpose of our experiment is to estimate the matching performance. We did an experiment on a set of 1500 documents. Given a query document, we measured the distances from this query document to all documents in the documents set. The documents in the database will be sort ascendingly based on the distances between them and the query document. If in 5 first ranked documents there exists a duplicate document of the query, we say that the query hit by *5 First Ranked*. If in 3 first ranked documents there exists a duplicate document of the query, we say that the query hit by *3 First Ranked* If the first ranked document is a duplicate document of the query document, we say that the query hit by *First Ranked*. We estimated the matching performance using *5 first ranked rate* (5 FRR), 3 First Ranked rate (3 FRR) and *First Ranked rate* (FRR).

4.4.2 Experimental result

Bellow is the 5 FRR, 3 FRR and FRR with some value of θ using Dice similarity measure and Cosine similarity measure.

θ	5 FRR	3 FRR	FRR
1	81%	80%	78%
2	82%	80%	77%
3	86%	82%	76%
4	87%	84%	77%
5	85%	82%	79%
6	85%	83%	80%
7	84%	82%	80%

Table 3: Performance with Dice similarity measure

θ	5 FRR	3 FRR	FRR
1	83%	78%	72%
2	84%	80%	75%
3	87%	80%	73%
4	87%	82%	75%
5	86%	82%	78%
6	86%	82%	77%
7	88%	83%	79%

Table 4: Performance with Cosine similarity measure

From the *FRR* column, it can be realized that the error descriptions occupy about 70% information of the documents. Using only the error description, a considerable result is achieved.

However, since the number of keywords are limited, when the number of documents increase, say, 1 million, documents sharing a quite high number of keywords may not be likely limited to duplicate documents. Therefore, using only the error description is not efficient. An algorithm to extract all useful information in the documents is required.

5 Conclusion and Future work

We have introduced and analyzed a complex documents matching method for identifying du-

plicate error cases in the CISCO network. As a partial solution of the whole solution, we did an experiment using only the error description part to match two documents. Although the result is quite good, however, when the number of documents using for the experiment become larger, using only the error description is not good enough.

In the future we concentrate on extracting useful information from the documents and in defining a distance metric between two documents.

Acknowledgments

The authors would like to thank Mr. Pham N.K. for his valuable comments to improve this paper.

References

[1] C. H. Pham, F. Lin, N. Gupta, and K. Ma. Big gap from academic response to industry's demand for optimized engineering efficacy. In *The International Conference on Dependable Systems and Networks (DSN-2006)*, 2006.

[2] G. Forman, K. Eshghi, and S. Chiocchetti. Finding similar files in large document repositories. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 394–400, New York, NY, USA, 2005. ACM Press.

[3] P. Kontkanen, P. Myllymäki, T. Silander, and H. Tirri. On bayesian case matching. In *EWCBR '98: Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, pages 13–24, London, UK, 1998. Springer-Verlag.

[4] K. Zhao, B. Liu, J. Benkler, and W. Xiao. Opportunity map: identifying causes of failure - a deployed data mining system. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 892–901, New York, NY, USA, 2006. ACM Press.

[5] W. B. Frakes and R. A. Baeza-Yates, editors. *Information Retrieval: Data Structures & Algorithms*. Prentice-Hall, 1992.

[6] T. B. Ho, S. Kawasaki, and N. B. Nguyen. Documents clustering using tolerance rough set model and its application to information retrieval. In *Intelligent exploration of the web*, pages 181–196. Physica-Verlag, January 2003.

[7] Z. Pawlak. *Rough Sets - Theoretical Aspects of Reasoning about Data*. Kluwer Academic, Dordrecht, 1991.

[8] H. Cunningham. Information Extraction, Automatic. *Encyclopedia of Language and Linguistics, 2nd Edition*, 2005.

[9] D.Maynard, K. Bontcheva, and H. Cunningham. Towards a semantic extraction of named entities. In *Recent Advances in Natural Language Processing*, Bulgaria, 2003.

[10] W. W. Cohen and A. McCallum. Information extraction from the world wide web. In *KDD '03: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2003.

[11] B. Habegger and M. Quafafou. Building web information extraction tasks. In *WI '04: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 349–355, Washington, DC, USA, 2004. IEEE Computer Society.

[12] D. L. Lee, H. Chuang, and K. Seamons. Document ranking and the vector-space model. *IEEE Software*, 14(2):67–75, 1997.

[13] B. R. Boyce, C .T Meadow, and H. K. Donald. *Measurement in Information Science*. Academic Press, 1994.