Data Linkage Discovery Applications

D

Richard S. Segall
Arkansas State University, USA

Shen Lu
University of South Florida, USA

INTRODUCTION

This chapter discusses the topic of linkage discovery for data and their applications. Since the publication of Segall and Lu (2014) which pertained to linkage discovery with glossaries many other investigators have performed research on applications of linkage discovery for big data under new conditions and criteria. This chapter enhances Segall & Lu (2014) by not only presenting a more coherent tabular summary of work of others presented in Segall & Lu (2014) and also again presented within this chapter, but also by including additional references that pertain to applications of linkage discovery not requiring a glossary framework.

BACKGROUND

This section discusses some of the terminology of text mining that is also used in linkage discovery that is web-based form of knowledge discovery. Latent Semantic Analysis (LSA) can be used to discover knowledge from text with a general mathematical learning method without knowing prior linguistic or perceptual similarity knowledge.

Latent Semantic Analysis (LSA) is a Natural Language Processing (NLP) technique that is based on similarity of words but not grammatical or syntactical structure and extracts knowledge through the similarity of individual words. The motivation of LSA in terms of psychology is that people learn knowledge only from similarity of individual words taken as units, not with knowledge of their syntactical or grammatical function.

Experimental result for linkage discovery for glossaries was shown in Lu et al. (2011) and Lu et al. (2012) and by other investigators that, by combining glossaries with the text, we can extract more meaningful words from the text and then link similar sections together.

Latent Semantic Analysis (LSA) can provide the meanings of the terms based on the context. However, one article cannot include all of the domain knowledge and the definition extracted from the context where the term appears in that article is not accurate. But, in glossaries, all of the terms are defined clearly. In Lu et al. (2011) and Lu et al. (2012), we manually put the definitions of the terms in glossaries to those words in an article and use those definitions to improve the accuracy of the background knowledge we can extract from the context. In this way, we can define meaningful words and use them to decide the theme of the corresponding sections.

Ferret (2002) presented a method, called TOPICOLL, for using collocations for topic segmentation and link detection. Figure 1 below illustrates the automation of the algorithm of Ferret (2002) for detecting topic shifts.

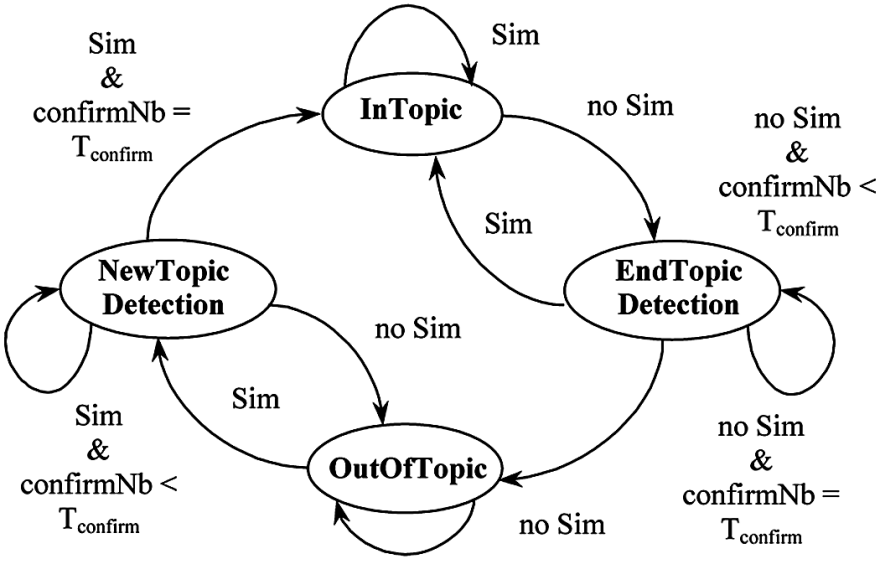
TOPICOLL Algorithm [Source: Ferret (2002)]

Parameters:

```
1.        State: {NewTopicDectec-
tion, InTopic, EndTopicDetection,
OutOfTopic}
2.        Sim: {True, False} // If
```

DOI: 10.4018/978-1-5225-2255-3.ch155

Figure 1. Automation for topic shift detection
[Ferret (2002)]




```
the context of the focus window and
the context of the current segment is
similar, Sim is True; otherwise, Sim
is False.
3.        ConfirmNb: integer // the
number of successive positions.
4.        Tconfirm: constant // the
threshold of successive positions.
Input: Document
Begin:
Foreach FocusWindow is Document
{
1.        if(State == NewTopicDetec-
tion && Sim == False && ConfirmNb <
ConfirmNb){
2.                State = OutOfTop-
ic;
3.                ConfirmNb = 0;
}
4.        if (state == NewTopicDe-
tection && Sim = True && ConfirmNb <
Tconfirm){
5.                State = NewTopic-
Detection;
6.                ConfirmNb++;
}
7.        if(State == NewTopicDe-
tection && Sim == True && ConfirmNb
≥Tconfirm){
8.                State = InTopic;
9.                ConfirmNb = 0;
}
10.       if(State == InTopic &&
Sim == True){
11.               State = InTopic;
}
12.       if(State == InTopic &&
Sim == False){
13.               State = EndTopic-
Detection;
14.               ConfirmNb = 0;
}
15.       if(State == EndTopicDe-
tection && Sim == False && ConfirmNb
< Tconfirm){
16.               State = EndTopic-
Detection;
17.               ConfirmNb++;
}
18.       if(State == EndTopicDe-
tection && Sim == True && ConfirmNb <
Tconfirm){
```