http://dx.doi.org/10.3745/KIPSTD.2011.18D.5.339

주성분 분석의 K 평균 알고리즘을 통한 XML 문서 군집화 기법

김 우 생†

요 약

최근 들어 인터넷에서 많이 사용되는 XML 문서들을 효율적으로 접근, 질의, 저장하는 방법들이 연구된다. 본 논문은 XML 문서들을 효율적으로 군집화 하는 새로운 기법을 제안한다. XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현하여 특징 벡터 공간상의 벡터로 나타내고 주성분 분석을 통한 k 평균 알고리즘 기법을 사용하여 군집화를 시도한다. 실험 결과를 통하여 제안하는 기법이 좋은 결과를 얻을 수 있음을 보였다.

키워드 : XML 군집화, k 평균 알고리즘, 주성분 분석

XML Document Clustering Technique by K-means algorithm through PCA

Woosaeng Kim†

ABSTRACT

Recently, researches are studied in developing efficient techniques for accessing, querying, and storing XML documents which are frequently used in the Internet. In this paper, we propose a new method to cluster XML documents efficiently. We use a K-means algorithm with a Principal Component Analysis(PCA) to cluster XML documents after they are represented by vectors in the feature vector space by transferring them as names and levels of the elements of the corresponding trees. The experiment shows that our proposed method has a good result.

Keywords : XML Clustering, K-means algorithm, PCA

1. 서 론

최근 들어 인터넷에서 데이터 교환과 정보 관리에 사용되는 XML 문서들을 효율적으로 접근, 질의, 저장하는 방법들이 연구된다. XML 문서들에 대한 군집화는 유사한 문서들의 그룹을 만들어 검색과 처리, 체계적인 문서 관리와 문서 저장, 문서 보안 등을 용이하게 해준다.

본 논문은 XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현하여 XML 문서를 특징 벡터 공간 상의 벡터로 표현한다. 이러한 특징 벡터 공간 상의 XML 문서들을 군집화하기 위해, 주성분 분석으로 적절한 시드점을 적용한 k 평균 기법을 사용하였다. 실험 결과를 통하여 제안하는 방법이 기존 K 평균 기법보다 더 좋은 결과를 얻음을 보였다.

본 논문의 구성은 다음과 같다. 2장은 관련 연구이고, 3장은 XML 문서를 문서-원소이름 레벨 행렬로 표현하여 k 평균 알고리즘으로 군집화하는 방법에 대하여 설명한다. 4장은 실제 데이터로 실험을 하여 제안한 방법이 효율적인지를 검증하고, 5장에서 결론을 낸다.

2. 관련 연구

2.1 XML 군집화

최근에 XML 문서들에 대한 다양한 군집화 기법들이 연구되고 있다. XML 문서간의 공통 구조의 존재 여부를 0,1의 비트에 기반한 군집화[1,2], XML 문서의 빈발 경로나 대표 경로 등에 기반한 군집화[3,4,5], XML 문서의 구조에 푸리에 함수를 적용해 n 차원 벡터들에 기반한 군집화[6] 방법 등이 연구되고 있다.

※ 본 연구는 2011년도 광운대 교내연구비로 지원됨.
† 종신회원 : 광운대학교 컴퓨터공학부 교수
논문접수 : 2011년 6월 28일
수 정 일 : 1차 2011년 8월 12일
심사완료 : 2011년 8월 13일

2.2 주성분 분석(PCA)

주성분 분석은 고차원 입력 벡터를 저차원의 벡터로 표현하여 몇 개의 주성분 값으로 나타내는 다변량 데이터 처리 방법 중의 하나이다[7,8]. n차원의 벡터 $x=[x_1x_2...x_n]^T$가 있을 때 식 (1)과 식(2)를 적용해 나온 평균벡터와 공분산 행렬을 통해 고유벡터를 구한 뒤에 대응되는 고유값의 크기에 따라 고유벡터를 정렬하여 새로운 행렬 A를 만든다.

$$m_x = \frac{1}{M}\sum_{k=1}^{M} x_k \tag{1}$$

$$C_x = \frac{1}{M}\sum_{k=1}^{M} x_k x_k^T - m_k m_k^T \tag{2}$$

이 새로운 행렬 A를 변환 행렬로 사용해 식 (3)과 같이 벡터 x를 벡터 y로 변환하면, y의 열에 있는 새 변수들 $y_1y_2...y_n$은 비상관성을 가지며 단조 감소 분산 순서로 배열되어 분산 값이 큰 주성분들로 차원을 줄일 수 있다.

$$y = A(x - m_x) \tag{3}$$

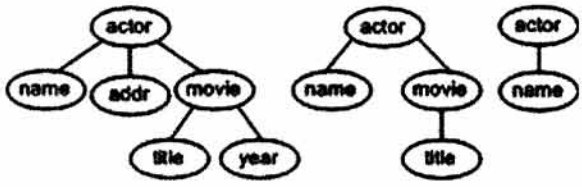
3. 문서-원소이름 레벨 행렬과 군집화 기법

본 논문은 사전에 군집의 숫자는 주어진다고 가정하며, 같은 DTD에서 생성된 문서들은 유사한 문서들로 간주하고 군집화를 시도한다. XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현하면 (그림 2)의 (a), (b), (c)는 모두 (그림 1)의 Actor DTD에 의해 생성되는 유사한 XML 문서들이다. 예를 들어, (그림 2(a))는 트리의 첫 레벨에 있는 원소 actor, 두 번째 레벨에 있는 원소들인 name, addr, movie, 그리고 세 번째 레벨에 있는 원소들인 title, year로 표현된다. XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현할 때 같은 DTD에 의해 생성되는 문서들은 같은 레벨의 같은 원소 이름들을 많이 공유하게 된다. 예를 들어, Actor DTD에 의해 생성되는 (그림 2(a))와 (그림 2(b))의 XML 문서들은 첫 번째 레벨의 actor, 두 번째 레벨의 name, movie, 세 번째 레벨의 title을 공유한다.

XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현하기 위해 문서-원소이름 레벨 행렬을 사용한다. 문서-원소이름 레벨 행렬에서 세로 축은 XML 문서들이고, 가로 축은 XML 문서들에 포함된 모든 원소 이름들의 알파벳 순서이며, 각 셀은 XML 문서에 포함된 원소 이름에 대응하는 레벨을 나타낸다. 따라서 (그림 2)의 (a), (b), (c)를 문서-원소이름 레벨 행렬로 표현하면 <표 1>과 같다.

```
Actor
 <!ELEMENT actor (name, addr?, movie*)>
 <!ELEMENT name (#PCDATA)>
 <!ELEMENT addr (#PCDATA)>
 <!ELEMENT movie (title, year?)>
 <!ELEMENT title (#PCDATA)>
 <!ELEMENT year (#PCDATA)>
```

(그림 1) Actor DTD



(a) (b) (c)

(그림 2) Actor DTD에 대응하는 트리들

〈표 1〉 문서-원소이름 레벨 행렬

	act	adr	mv	nm	ttl	yr
A1	1	2	2	2	3	3
A2	1	-	2	2	3	-
A3	1	-	-	2	-	-

XML 문서들을 문서-원소이름 레벨 행렬로 표현할 때 XML 문서는 특징 벡터 공간상의 특징 벡터 $x=[x_1x_2...x_n]^T$로 표현될 수 있다. 왜냐하면 문서-원소이름 레벨 행렬에서 원소 이름들을 특징 벡터 공간상의 기준 축들로 하고 대응하는 레벨들을 특징 값들로 간주 하면, XML 문서는 특징 벡터 공간 상의 벡터로 표현된다. 예를 들어, <표 1>의 A1 문서는 (1,2,2,2,3,3)의 벡터로, A2 문서는 (1,-,2,2,3, -)의 벡터로 표현된다. 따라서 본 논문에서는 이러한 특징 벡터 공간 상의 XML 문서들을 k 평균 알고리즘을 사용하여 군집화하는 방법을 제안한다. 시드점이라고 불리는 초기의 k개 패턴들은 무작위로 선택될 수도 있지만, 군집 구조에 관한 어떤 지식을 사용해 적절한 시드점들을 구할 수 있다면 더 나은 성능을 얻을 수 있다[7,8]. Book과 Club의 DTD가 (그림 3)과 같고, Actor, Book, Club의 DTD에 의해 생성된 8개의 XML 문서들과 이 문서들에 포함되는 10개의 원소 이름들로 구성된 문서-원소이름 레벨 행렬이 <표 2>와 같을 때, (A1, A2, A3), (B1, B2), (C1, C2, C3)의 각 그룹 문서들은 특징 벡터 공간상에서 서로 다른 군집을 형성함을 알 수 있다. 이처럼 각 그룹이 서로 다른 군집을 형성할 때 가장 적절한 시드점들은 각 그룹에서 한 개씩의 시드점 즉, 한 개씩의 문서를 찾는 것이다. 주어진 XML 문서들이 표2와 같을 경우 예를 들어, A2, B1, C3 문서를 3개의 시드점으로 선택하면 무작위 시드점으로 k 평균 알고리즘을 수행하는 것보다 좋은 결과를 얻을 수 있다.

〈표 2〉 문서-원소이름 레벨 행렬

	act	adr	aut	bk	clu	cln	nm	mv	nm	ph	ttl	yr
A1	1	2	-	-	-	-	-	2	2	-	3	3
A2	1	-	-	-	-	-	-	2	2	-	3	-
A3	1	2	-	-	-	-	-	-	-	-	-	-
B1	-	-	2	1	-	-	-	-	3	3	2	2
B2	-	-	2	1	-	-	-	-	3	-	2	2
C1	-	3	-	-	1	2	2	-	3	3	-	-
C2	-	-	-	-	1	2	2	-	3	3	-	-
C3	-	-	-	-	1	2	2	-	3	3	-	-

```
Book
  <!ELEMENT book (title, year, author*)>
  <!ELEMENT author (name, phone?)>
  <!ELEMENT title (#PCDATA)>

  <!ELEMENT year (#PCDATA)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT phone (#PCDATA)>
Club
  <!ELEMENT club (clubname, member+)>
  <!ELEMENT member (name, phone, addr?)>
  <!ELEMENT clubname (#PCDATA)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT phnone (#PCDATA)>
  <!ELEMENT addr (#PCDATA)>
```

(그림 3) Book과 Club의 DTD

본 논문에서는 이러한 시드점들을 찾기 위해서 주성분 분석 기법을 사용한다. 특징 벡터는 차원의 수만큼 존재하는 기준 축들로 표현되므로 결국 차원을 축소한다는 의미는, 중요한 기준 축들을 찾아 특징 벡터를 중요한 기준 축들로 다시 표현하는 것이다. 따라서 적절한 시드점들을 찾기 위해서 문서-원소이름 레벨 행렬에서 문서들을 기준 축들로, 원소 이름들을 특징 값들로 가정한 특징벡터 공간 상에서 주성분 분석 기법을 적용하여 중요한 기준 축들을 즉, 중요한 문서들을 시드점들로 찾는다. <표 2>의 문서 A1부터 C3까지의 8개의 기준 축들로 구성된 특징 벡터 공간 상의 12개의 벡터들에 대해 주성분 분석을 적용한 결과는 표3과 같다. <표 2>에서 '—'와 같이 해당 되는 원소 이름이 없는 경우는 트리의 레벨 값과 구분하기 위하여 음수 값인 -100을 사용하였다. <표 2>의 경우 Actor, Book, Club의 DTD에 의해 생성된 문서들의 군집화가 요구되므로 3개의 시드점들을 구해야 한다. <표 3>의 주성분 열(P1~P8)은 고유값이 적은 즉, 분산 성분이 줄어드는 순서로 정열 되어 있으므로, P1 열에서 가장 절대값이 큰 0.479 즉 C2 문서, 마찬가지로 P2 열에서 -0.609 즉 B1 문서, P3 열에서 -0.655 즉 A2 문서를 각각 시드점으로 선택한다. 주성분 분석을 통하여 A, B, C의 각 그룹에서 한 문서씩이 시드점으로 선택되었음을 알 수 있다. 이것은 각 그룹에서 하나씩의 문서가 선택될 때, 이 문서들이 모든 원소 이름들을 대부분 포함할 수 있기 때문이다.

〈표 3〉 〈표 2〉의 주성분분석결과

	P1	P2	P3	P4	P5	P6	P7	P8
A1	-0.339	0.33	-0.494	0.373	0.48	-0.293	0.277	0
A2	-0.237	0.23	-0.655	-0.433	-0.373	0.165	-0.329	0
A3	-0.061	0.406	0.107	0.505	-0.612	0.303	0.31	0
B1	-0.209	-0.609	-0.189	0.293	-0.443	-0.516	-0.04	0
B2	-0.312	-0.516	-0.169	0.199	0.216	0.72	0.053	0
C1	0.471	0.087	-0.201	0.533	0.092	0.078	-0.659	0
C2	0.479	-0.126	-0.325	-0.066	-0.046	0.046	0.375	0.71
C3	0.479	-0.126	-0.325	-0.066	-0.046	0.046	0.375	-0.71

4. 실험 및 평가

본 연구는 위스콘신 대학의 XML 데이터 뱅크에서 제공하는 데이터들을 사용하여 제안하는 방법의 효율성을 실험하였다. 이 데이터 뱅크는 bibliography, club, company profiles, stock quotes, department, personal information, movies, actor의 8개 DTD들을 제공하며 일부 DTD 들은 같은 원소 이름을 공유한다[9]. <표 4>는 형성되는 군집의 수를 다르게 했을 때 주성분 분석을 통해 적절한 시드점들을 찾는 비율과 k 평균 알고리즘의 수행 결과를 보여준다. 군집은 2개에서 7개까지의 6개의 서로 다른 크기의 집단으로 나누었으며, 각 집단에서는 10번씩의 실험을 통하여 평균 값을 구하였다. 예를 들어, 3개 군집으로 이루어진 집단은 8개의 DTD 중에서 3개의 DTD를 선택할 수 있는 모든 경우 수에서 임의의 10개 경우만으로 집단을 구해 사용했다. 또한 각 군집에는 각각의 DTD에 대해 생성된 10개씩의 유효한 XML 문서들을 생성해 사용하였다. 표 4에서 적절한 시드점들이란 모든 군집에서 한 문서씩을 시드점으로 선택하는 경우를 의미한다. 예를 들어, 4개의 군집에서 4개의 시드점들을 찾을 때 3개의 군집에서만 시드점들이 찾아진다면 적절한 시드점들을 찾는 비율은 75%가 된다. 표4에서 보듯이 군집의 수가 많은 집단일 수록 PCA를 통해 적절한 시드점들을 찾는 비율이 낮아짐을 알 수 있다. 즉 군집의 수가 많은 집단일 수록 일부 군집에서만 시드점들이 선택되는 경우가 발생한다. 이것은 군집 수가 많은 집단일 수록 다양한 DTD가 사용되어 같은 원소 이름들을 공유하는 XML 문서들이 많아 지기 때문에, 일부 군집에서만 문서들이 시드점들로 선택되더라도 집단에 존재하는 대부분의 원소 이름들을 포함할 수 있기 때문이다. 또한 제대로 된 군집을 찾을 군집 성공 비율은 주성분 분석 기법의 시드점들을 사용한 k 평균 알고리즘이 무작위 시드점들을 사용한 k 평균 알고리즘보다 두 배 정도의 좋은 결과를 얻음을 알 수 있다. 이것은 주성분 기법을 사용하면 적절한 시드점들을 찾을 수 있기 때문이다. 그러나 군집의 수가 많은 집단일 수록 군집 성공 비율이 낮아지는 것은 군집의 수가 많은 집단일 수록 적절한 시드점들을 찾는 비율도 낮아지기 때문이다.

〈표 4〉 군집수에 따른 실험 결과

군집수	적절한 시드점을 찾는 비율	주성분 군집 성공률	무작위 군집 성공률
2	90	100	60
3	90	90	40
4	90	85	40
5	90	76	40
6	86	72	40
7	82	60	27

5. 결 론

XML 문서를 대응하는 트리 구조의 원소들의 이름과 레벨로 표현할 때 같은 DTD에 의해 생성되는 문서들은 같은 레벨의 같은 원소 이름들을 많이 공유하게 된다. 따라서 본 연구에서는 모든 문서의 원소 이름들을 특징 벡터 공간상의 기준 축들로 하고 대응하는 레벨들을 특징 값들로 간주 하여, XML 문서들을 특징 벡터 공간 상의 벡터들로 표현하고 주성분 분석을 적용한 k 평균 알고리즘을 사용하여 군집화를 시도하였다. 실험을 통해 무작위 k 평균 알고리즘보다 주성분 분석으로 적절한 시드점들을 통한 k 평균 알고리즘으로 군집화하는 경우가 좋은 결과를 얻을 수 있었다. 추후 과제로는 주성분 분석을 통한 k 평균 알고리즘을 다양한 응용에 적용하는 연구가 필요하다.

참 고 문 헌

[1] Yoon, J., Raghavan, "BitCube: Clustering and Statistical Analysis for XML Documents", Journal of Intelligent Information Systems, Vol.17, 2001.

[2] 김우생, "비트벡터에 기반한 XML 군집화 기법", 대한전자공학회 논문지, 2010. 9.

[3] 이정원, 이기호, "유사성 기반 XML 문서 분석 기법", 정보과학회 논문지: 소프트웨어 및 응용 제 29권 제 5-6호, 2002. 6.

[4] 황정희, 류근호,"XML 문서의 공통 구조를 이용한 클러스터링 기법", 정보과학회논문지 D: 데이타베이스 제 32권 제 6호, 2005. 12.

[5] 황정희, "클러스터의 주요항목 가중치 기반 XML 문서 클러스터링", 한국정보처리학회 논문지 D- 데이터베이스, 2007.

[6] 이호석, "함수 변환과 FFT에 기반한 조정자가 없는 XML 문서 클러스터링 기법", 한국정보처리학회 논문지 D-, 2007.

[7] Gonzalez R.C., Woods R.E., "Digital Image Processing, Prentice-Hall, 2007.

[8] 오일석, 패턴인식, 교보문고, 2008.

[9 Niagara Query Engine, http://www.cs.wisc.edu/niagara/data.html

김 우 생

e-mail : kwsrain@kw.ac.kr

1985년 텍사스 주립대학 전산학과(학사)

1987년 미네소타 주립대학 전산학과(석사)

1991년 미네소타 주립대학 전산학과(박사)

1987년~1988년 현대전자 제우스 컴퓨터 과장

2001년 UC 버클리대학 교환 교수

1992년~현 재 광운대학교 컴퓨터공학부 교수

관심분야 : 데이터베이스, 멀티미디어 등