순서 바이어스 최소화에 의한 안정적 클러스터링 구축에 관한 연구

이 계 성†

요 약

데이터 마이닝, 또는 기계학습을 위한 무감독 학습 알고리즘인 개념적 클러스터링을 이용하여 계층적 구조를 유도해낼 때 자료를 처리하는 순서에 따라 서로 다른 결과에 도달하는 양상을 보인다. 이 순서 바이어스 문제를 해결하는 방안으로 먼저 주어진 자료 세트에 분류를 시행하여 초기 분류를 형성한다. 이 분류를 통해 최종 분류의 가능한 클래스 수를 예측하고 이 정보에 기반하여 자료 분석과 중심 정렬을 통해 자료 처리 순서를 새로이 설정한다. 재배열된 자료 세트에 ITERATE 분류 과정을 적용해 새로운 분류를 생성한다. 본 논문에서 이 과정을 반복하여 안정적이고 최적의 분류 점수를 갖도록 하는 알고리즘 REIT를 제안하였다. 이 알고리즘을 여러 자료 세트에 적용하고 순서 바이어스의 영향을 최소화하는지 여부를 실험을 통해 비교 분석하였다.

A Study on the Construction of Stable Clustering by Minimizing the Order Bias

Gye-Sung Lee†

ABSTRACT

When a hierarchical structure is derived from data set for data mining and machine learning, using a conceptual clustering algorithm, one of the unsupervised learning paradigms, it is not unusual to have a different set of outcomes with respect to the order of processing data objects. To overcome this problem, the first classification process is proceeded to construct an initial partition. The partition is expected to imply the possible range in the number of final classes. We apply center sorting to the data objects in the classes of the partition for new data ordering and build a new partition using ITERATE clustering procedure. We developed an algorithm, REIT that leads to the final partition with stable and best partition score. A number of experiments were performed to show the minimization of order bias effects using the algorithm.

1. 서 론

정보기술의 발달로 지식의 수집과 저장이 용이해졌고 따라서 자료의 다양성과 양적 증가로 인해 컴퓨터에 기반한 자료분석 기법이 매우 중요해 졌고 자료 속에 내재되어있는 개념, 또는 지식의 발견이나, 관계, 경향 등을 파악하는 것이 중요한 연구과제로 등장하여 활발한 연구가 진행되어왔다. 특히, 최근에는 수동적인 자료분석 방법에서 탈피하여 다량의 데이터 베이스에 저장되어 있는 자료로부터 상관규칙(association rules)을 찾아내는 데이터 마이닝[1], 또는 지식발견(knowledge discovery)[2]이라는 분야가 새로이 등장하면서 많은 연구가 진행 중에 있다. 자료의 양이 수십만 단위

※ 이 논문은 1997년 한국학술진흥재단 학술연구조성비에 의하여 지원되었음.
† 정 회 원 : 단국대학교 전산학과 교수
논문접수 : 1999년 2월 22일, 심사완료 : 1999년 5월 3일

에 이르고 수백 개의 필드로 구성되어있는 자료 집단을 자료분석가나 특정 통계기법을 사용하여 자료를 요약 정리하는 수준으로는 이를 해결할 수 없게되어 기계학습 기법을 이용한 방법이 다양하게 연구되고 있다. 더욱이 실세계의 자료가 모호하게 표현될 뿐 아니라 연속적인 수치자료의 처리 및 불완전한 자료의 처리에 대한 요구로 인해 자료분석을 더욱 어렵게 만든다. 이 문제를 더 어렵게 하는 것이 바로 평가 바이어스 문제이다. 특히, 개념적 클러스터링(conceptual clustering)과 같은 무감독 학습 기법에서는 클래스를 분류하는 기준으로 평가함수(evaluation function)를 사용하는데, 이 평가함수를 설정하는데 있어 중요한 요소가 바이어스의 설정이다. 즉, 어떤 바이어스가 포함되어 있느냐에 따라 해당 바이어스에 편중된 결과가 생성된다. 또 다른 중요한 바이어스 문제로 자료입력 순서에 따라 상이한 결과를 생성해 내는 순서바이어스 문제가 있다. 순서바이어스는 입력되는 자료객체의 순서에 따라 다른 결과를 생산해내는 문제를 야기한다. 본 연구에서는 개념적 클러스터링이 갖고 있는 이 순서바이어스 문제에 대해 연구 고찰하고 기존의 해결방안을 조사해 본 후, 이들이 갖는 문제점과 한계점을 지적하고, 개선된 알고리즘을 고안하여 순서 바이어스에 의한 영향을 최소화하는 방안을 제시하고자 한다. 또 고안된 알고리즘의 성능을 평가하기 위해 기존의 방법과 비교 분석하기로 한다. 성능 평가를 위해 기계학습 분야에서 자주 사용되는 2개의 대표적 자료 집단을 이용하고 실제 응용으로 사용되는 지질학적 자료를 활용하기로 한다.

본 논문의 구성은 다음과 같다. 2장에서 개념적 클러스터링에 대해 소개하고 이것이 갖는 순서 바이어스 문제와 기존의 해결 방안 및 그 한계를 설명한다. 3장에서는 기존의 해결방안의 문제를 개선한 REIT알고리즘을 소개한다. 4장에서는 자료 집단에 대한 소개와 불완전 정보 처리에 대한 방법을 설명한다. 5장에서는 실험결과를 기존의 방법과 비교하여 설명하고 6장에서는 결론과 논의를 요약한다.

2. 개념적 클러스터링의 순서바이어스 문제

본 장에서는 개념적 클러스터링에 대한 소개와 이것이 갖는 순서 바이어스 문제를 설명하고, 이 순서 바이어스와 관련하여 본 연구에서 논의되는 2개의 시스템, COBWEB[6]과 ITERATE[5]에 대해 설명하기로 한다. 동시에 이 시스템들이 추구하는 해결방안에 대해서도 알아본다.

2.1 순서 바이어스 문제

기계학습 기법의 하나인 개념적 클러스터링은 무감독(unsupervised) 학습 기법의 일종이다. 자료 객체의 나열로부터 중요한 개념, 또는 자료집단의 구조를 유도해 내는 알고리즘이다. 이에 적용되는 집단 분석(cluster analysis) 또는 수치적 분류(numerical taxonomy)는 일반적으로 자료 객체간의 유사성을 계산해 유사한 자료 객체들을 카테고리 또는 클래스로 모으는 작업으로 알려져 있다. 각 자료 객체는 수치로 표시한 속성들의 나열로 표현되고, 클러스터링 작업은 다차원 측정공간에서 분포도에 따른 구조분석을 통해 그 자료 집단에 내포된 구조를 추출해내는 것으로 볼 수 있다. 그 구조형태는 계층적이거나 분할적(계층적이 아닌 일차원의) 그룹의 형태를 갖는다.

수치적 클러스터링 기법들은 여러 응용에서 성공적으로 사용되어 왔으나, 자료 객체의 표현이 명칭(nominal)적인 값을 포함한 경우에는 쉽게 적용되지 않았다. 수치적 기법의 사용 제약은 명칭을 포함하는 자료를 처리하는 개념적 방식(수치적 분류에 대응되는 용어)으로 발전하면서 그 제약이 해결되기 시작했다. 수치적 방식은 유사성과 차이점을 다차원 공간에서 거리 측정(예, Manhattan, Euclidean, Mahalanobis 방식)[10]을 통해 파악된다. 반면 개념적 방식에서는 자료 값이 차지하는 비중에 따라 결정되는 확률 값으로 측정된다. CLUSTER/2[7]는 그룹 내에서의 공통 속성값, 그룹간에는 다른 속성값, 그리고 그룹을 기술하는 표현식의 간단 명료성을 클러스터링 작업의 기준함수(criterion function)로 사용한다. UNIMEM[8]은 그룹내의 유사성에 국한하는 Hamming distance를 이용해 클래스 트리를 만든다. AUTOCLASS[9]와 COBWEB[6]은 각 그룹을 자료 객체들의 속성에 대한 확률분포로 정의한다.

계층적 클러스터링 방법에 기반을 둔 개념형성 시스템에는 순서바이어스에 대한 논의가 많이 있어 왔다[3, 4,5]. 개념적 클러스터링을 이용한 이 시스템은 일련의 자료를 입력으로 받아들여 클래스 구조로 나누고 각 클래스를 구성하는 특성들로 각 클래스를 정의한다. 개념형성 방법은 일반적으로 학습과정으로 실명할 수 있고 학습과정은 점진적(incremental)이어야한다는 특성을 갖는다. 클러스터링이 점진적이라는 것은 인간의

기억 자원을 인지 과학적인 입장에서 볼 때 자연스러운 형태이고 자료분석 관점에서 보더라도 계속 변화하고 발전하는 자료의 본질적 특성상 매우 필요한 방식임에 틀림이 없다. 그러나 이것으로 인해 발생하는 중요한 문제점으로 순서바이어스 영향(order bias effect)이 있다. 학습과 개념형성 관점에서의 클러스터링은 점진적 클러스터링 방법의 전형적인 예이므로 순서 바이어스가 방식 자체에 존재하는 것으로 볼 수 있다. 반면 자료 분석적인 관점에서의 클러스터링은 순차적인 입력을 전제로 할 필요는 없으므로 방식에서 오는 순서바이어스는 없앨 수 있으나 자료의 적정한 배열 여부에 따라 순서바이어스의 영향을 받게된다. 우선 전자에 해당하는 COBWEB의 순서 바이어스 문제를 살펴보고, 계속해서 후자에 해당하는 ITERATE 시스템에 대해 알아보기로 한다.

2.2 COBWEB의 순서 바이어스 문제

COBWEB은 원래 인간의 개념형성과 인지과정을 모델로 하여 개발되었다. COBWEB은 자료 객체의 속성-값 쌍을 입력으로 받아들이고 출력으로 개념트리를 생성시킨다. 트리의 노드는 하나의 개념을 나타내고 위에서 아래로 내려 갈수록 일반적 개념으로부터 보다 상세한 개념으로 이어진다. 끝 노드들은 하나의 자료 객체로 구성된 노드가 된다. 이것은 점진적 학습기로 자료 객체가 기존의 트리에 입력되면 개념노드의 변화를 가져오게 되고 경우에 따라서는 전체적 구조를 변경시킬 수도 있다.

COBWEB에 사용된 카테고리 유틸리티(CU)는 Gluck과 Corter[11]에 의해 처음 사용되었는데 카테고리 유틸리티의 기대값은 $P(A_i = V_{ij}|C_k)^2$로 정의되고, 이것은 주어진 자료 객체가 클래스 C_k에 속한다고 가정할 때 A_i가 V_{ij} 값을 갖게될 확률을 나타내고 그 의미하는 바는 자료 객체의 해당 클래스와의 확률적 결합도를 나타낸다고 말할 수 있다. 어떤 클래스 C_k에 대해 CU는 다음과 같이 정의된다.

$$CU_k = P(C_k)\sum_i\sum_j P(A_i = V_{ij}|C_k)^2 - P(A_i = V_{ij})^2$$

이것은 속성값 쌍이 클래스 k에 대해 올바르게 추측될 수 있는 속성 값들의 기대값 $(P(C_k)\sum\sum P(A_i = V_{ij}|C_k)^2)$과 클래스에 대한 정보가 없을 때의 기대값 $(P(C_k)\sum\sum P(A_i = V_{ij})^2)$과의 차이에 해당하는 증가분을 뜻한다. Fisher는 그의 논문([6])에서 이 카테고리 유틸리티의 역할은 클래스 내부의 유사성(similarity)과 클래스간 상호 차이점(dissimilarity)을 극대화하는 타협점을 찾는 것이라고 설명하였다. 이와 같은 평가함수를 바탕으로 그는 클래스 트리를 구축하는 점진적 알고리즘을 개발하였다.

여러 연구를 통해서 COBWEB은 자료 입력순서에 의존한다는 현상이 밝혀졌고 자료 객체의 입력 순서가 구체적으로 클래스 트리 구축에 미치는 영향이 밝혀지기도 했다[3,4,5]. 특히 [3]에서는 어떤 자료입력순서는 분류 트리를 형성할 때 자료 객체가 적절하지 못한 노드에 속하는 경우가 발생하였고 결국 의미를 부여할 수 없는 노드의 형성으로 발전할 가능성이 있다는 것을 보이기도 하였다. 이들은 COBWEB 알고리즘에서 활용한 분리 및 병합 연산자를 확대하여 승진(promote) 연산자를 고안하였다. 그러나 근본적으로 순서에 의존하는 방법들은 모든 경우의 수를 탐색하는 소진 탐색을 통해 최적의 해를 찾기 전까지는 그 해가 제한적일 수밖에 없다.

이 순서문제를 해결하기 위한 방안의 하나로 반복적인 최적화를 통해 안정적인 클러스터링을 구현하고자 하는 시도가 있다[4]. 이 방법의 기본 전략은 최초의 클러스터링을 계산적으로 빠르게(계산비용을 최소화하여) 임시 클러스터링을 구축하고, 이를 반복적으로 최적화시켜 나간다. 이를 위해서는 최초의 클러스터링 구축은 계산적으로 값싼 방법을 선택하고 반복적 최적화를 위해 다양한 탐색 제어 방법을 고안(예로, 재배치 연산자, redistribution operator)하고 활용한다. 그러나 이 방식은 순서 바이어스를 직접적으로 해결하기보다는 최적화를 통해 간접적으로 순서 바이어스 문제를 해소하는 방안으로 볼 수 있다.

2.3 ITERATE의 문제 해결 방안

또 다른 시도로 ITERATE 시스템[5]을 들 수 있다. 자료 분석을 통해 가장 상이한 자료들을 순서적으로 나열 한 후 이 순서대로 자료 객체들을 공급할 경우 그렇지 않은 경우에 비해 좀더 개선된 분류 트리가 형성된다는 점을 주장하였다. ITERATE 알고리즘은 COBWEB의 제어 구조를 수정하여 지엽적인 문제해결 수준에서 벗어나 이 순서바이어스 문제에 대해 보다 전역적인 관점에서 해결하려는 시도를 하였다. 객체를

초기의 분류된 그룹에 대해 반복적 재배치를 수행하는데, 재배치는 정의된 기준에 가장 합당한 것(예로, 카테고리 유틸리티를 최대로 만드는 것)에 배정하는 방식을 취한다. 이 방식은 수치적 분류를 위해 채용한 클러스터링 알고리즘에서 사용한 제어구조와 유사하다. 근본적으로 이 알고리즘도 객체를 반복적으로 재배치하는데 기준은 근접성 측도(proximity measure)에 기반을 둔다. 이 반복적 재배치는 특정 전역 기준(예로, 평균 제곱 에러의 최소화)을 만족할 때까지 지속된다.

ITERATE 알고리즘은 다음과 같이 요약된다 :

① 비유사도(dissimilarity)에 따른 자료 객체의 순서화
② COBWEB[6] 알고리즘에 기반하여 개념 트리의 생성
③ 트리 구조에서 자료의 초기 베이스 레벨의 추출
④ 카테고리 일치도[6]에 따른 그룹간 자료 객체의 반복적 재배치
⑤ 재배치하여 변경되는 자료 객체가 더 이상 없을 때까지 반복적 재배치를 계속한다.

ITERATE의 가장 큰 목적 중 하나가 바로 순서에 의한 분류 결과의 불안정성을 배제하는데 있다. ITERATE는 COBWEB의 수정된 버전을 사용하여 초기 분류 트리를 생성하므로 순서에 의존하는 문제를 자동적으로 갖게된다. 이 문제를 해결하기 위해 위의 알고리즘에서 첫 번째 단계의 과정이 포함되어진다. 이 과정을 수용하게 된 이유의 하나는 자료 분석이라는 목적에 있어서는 수집된 자료 전체가 미리 알려져 있는 경우가 대부분이므로 분류 트리 구성이 굳이 점진적일 필요가 없다는 것이다. ITERATE 알고리즘은 이 점진적 속성을 포기하는 대신 자료를 비유사도 중심으로 배열하여 균형 있는 트리 구성과 안정적 분류를 유도해 내도록 노력한다.

ITERATE에서 시도하는 최적의 자료 입력 순서는 자료 객체를 무작위로 선택하고 반복적으로 다음 자료 객체를 선택하는데 선택기준은 이전 자료 객체들과 비교하여 최대로 상이한 객체를 선택하게된다. 최적의 순차적 배열을 통해 전처리가 끝나면 클러스터링 단계로 넘어가고 클러스터링 구조를 최적화 하는 단계로 이어진다. 이것은 비록 ITERATE가 순서 바이어스에 의한 영향을 최소화하려는 시도를 하였으나 효과 면에서 COBWEB보다 어느 정도 개선된 효과를 보고 있으나 순서 바이어스에 의한 문제를 완전히 해결하지 못하였고 비유사도에 따라 배열된 자료 입력 순서가 순서 바이어스의 영향을 최소화하였다고 단정하기에는 많은 문제점을 갖고 있다. 그것은 본 연구의 실험을 통해 나타났듯이 자료의 전처리 과정에서 비유사도에 따라 배열하였다 하더라도 최상의 결과에 도달하지 못하는 경우가 종종 발생하였고 최적의 결과에 도달하는데 요구되는 반복 회수가 많은 경우도 자주 발견되었다. 이와 같은 문제는 근본적으로 자료에 의한 분석만으로, 또한 순서화 배열을 통한 것만 가지고 순서 바이어스 문제를 해결하는 데에는 한계가 있다는 것을 증명하는 결과인 셈이다. 또한 순서화하는데 필요한 계산적 비용이 추가되어 효율적이지 못하다는 단점도 아울러 갖게된다.

3. 안정적 클러스터링을 위한 REIT 알고리즘

위에서 언급한 문제점과 한계를 개선하기 위해 REIT 알고리즘을 고안하였다. REIT 알고리즘은 ITERATE [5]에 근거를 두고 있고 ITERATE는 기본적으로 COBWEB 시스템으로부터 유도되었다고 할 수 있다. 이 세 가지 모두 COBWEB의 카테고리 유틸리티[11]를 이용하여 분류하고 있다. 그러나 각각의 차이점은 제어 구조에 있어 각기 다른 제어 루틴을 수용하고 있어 순서 바이어스의 영향을 최소화하도록 노력한다. REIT 알고리즘은 ITERATE 알고리즘에서 사용하는 비유사도를 중심으로 하는 자료 객체의 순서화 과정을 제거하여 입력 자료에 대한 순서를 무시하고 분류 트리를 통해 베이스 레벨의 초기 프로토타입을 결정한 후 카테고리 일치도[6]에 따른 재배치 연산을 통해 초기 분류 그룹을 구축한다. 초기 분류 결과는 순서에 의한 영향을 받은 결과라고 단정해도 된다. 먼저 자료 순서에 대한 영향을 포함하고 분류를 시도하였으므로 COBWEB의 분류 트리로부터 유도된 베이스레벨의 프로토타입 클래스나 그로부터 생성된 분류 그룹 모두 순서에 의한 영향을 받을 수밖에 없는 결과이다. 그러나 초기 분류 그룹의 각 클래스는 중요한 정보를 제공해 준다.

먼저 클래스 수는 다음 자료 순서화에 중요한 정보를 제공한다고 할 수 있다. 아무리 순서에 의한 영향을 받는다고 하더라도 산출된 클래스 수에 있어서는, 최종 결과로 진행하는데 있어 급격한 변화를 갖지 않는다고 가정할 수 있다. ITERATE는 최적의 배열이라고 정의한 자료의 순서화를 통해 COBWEB보다 안정적인 결론에 도달하였다고 주장하였다[5]. 그러나 이 자료 배

결에는 자료 객체 각각의 속성 최대로 상이한 배열을 추구하고 있으므로 이와 같은 배열은 각 자료 객체가 하나의 클래스를 형성할 때 그 의미가 있는 것이다. 따라서 클래스 정보에 대한 것이 전혀 없는 상태에서는 이와 같은 순서화가 최적의 효과를 나타낸다고 단정할 수 없다. 초기 분류를 통해 자료에 대한 정보와 클래스에 대한 정보를 수집한 후 이를 근거로 자료를 재배열한다면, 예상된 클래스 수에 따른 상이한 자료 배열이 이뤄질 수 있고 이를 통해 최적의 결과에 수렴할 수 있다는 것이 본 연구의 핵심이라 할 수 있다.

초기 분류에서 얻어진 각 클래스에 포함된 자료 객체들은 카테고리 유틸리티에 근거한 분류이기 때문에 클래스 내의 객체간에는 상호 유사성이 높게 구성되어 있으며, 클래스 상호간에 있는 객체들 간에는 비유사도가 높게 설정되어 있다고 할 수 있다. 초기 분류에서 얻어진 자료 객체들은 순서에 의해 재한 받지 않는 자료 순서로 변환하기 위해 먼저 각 클래스의 중심이 되는 자료 객체를 선정한다. 클래스의 중심은 클래스를 대표할 수 있는 가장 보편적인 객체로 선정된다. 중심이 되는 객체는 속성 A_i에 대해 값 V_{ij}를 가질 때, $\sum_i P(A_i=V_{ij})$값을 최대로 하는 객체가 선정될 것이다. 클래스간의 순서가 전체 결과에 미치는 영향에 대해서는 본 연구에서 시행하지 않았다. 임의적으로 클래스 순서가 결정되고 각 클래스 별로 반복하면서 중심이 되는 중심 객체를 선택하여 자료 객체 순서를 결정한다. 이 중심 정렬 알고리즘을 요약하면 다음과 같다. (그림 1) : 여기서 C = {C_1, C_2, C_3, ... }는 클래스를 나타내고 O_{kc}는 클래스 C_k의 자료 객체 c를 나타낸다. A_i는 속성을 나타내고 속성값은 객체 l의 값 V_{il}로 표시하기로 한다.

```
for i=1,2, ...
   OCi = {Oij | 클래스 i, 자료객체 j}
center=maxcenter=0
while (∃Okl in any OCk)
{  for Ck, k=1,2, ...
   {   for Okl, l=1,2, ...
       {   center = ∑iP(Ai=Vil)
           if (center > maxcenter)
               Okc = Okl
       }
       output Okc
       OCk = OCk    {Okc}
   }
}
```

(그림 1) 중심정렬

위의 중심 정렬 알고리즘은 이미 분류된 초기 분류의 클래스를 중심으로 자료 객체를 다시 정렬하게되고 이어 분류 트리 생성, 초기 클래스 집단 구성, 반복적 재배치 연산을 마치고 새로운 분류에 대해 분류 점수(클래스의 평균 카테고리 유틸리티 - $\sum_k$ 카테고리 유틸리티(C_k)/(클래스 수))의 변화를 조사한다. 개선된 값을 갖게되면 분류과정을 종료하고 그렇지 않다면 중심정렬부터 전 과정을 다시 반복한다. REIT 알고리즘을 다시 정돈하면 (그림 2)와 같이 요약할 수 있다.

```
0 PU = 0 : 분류점수
1 카테고리 유틸리티를 이용하여 계층적 분류 트리
  생성
2 트리로부터 대표할 수 있는 노드를 선택하여 초기
  클래스 집단을 구성
3 카테고리 일치도[6]에 따른 클래스간 자료 객체의
  반복적 재배치
4 재배치하여 변경되는 자료 객체가 더 이상 없을
  때까지 3을 반복
5 분류된 그룹에 대해 PU를 계산
6 PU에 대한 개선이 없다면 종료
7 중심 정렬 알고리즘 적용하여 자료 객체 정렬
8 1로 이동하여 계속 실행
```

(그림 2) REIT 알고리즘

ITERATE에서 취하는 전처리 과정은 $O(n^2)$의 계산비용을 요구하게 된다. 중심 정렬을 이용한 REIT는 ITERATE에서 취한 전처리 과정을 거치지 않으므로 이와 같은 계산을 절약할 수 있다. 그러나 REIT에서는 초기 분류를 생성하기 위해 분류 트리 생성, 재배치 연산이 추가적으로 요구되지만 이 계산 비용은 $O(n\log n)$을 넘지 않는다[4]. 중심정렬에서는 개별 객체간 비교가 아니라 $P(A_i=V_{ij})$를 계산하므로 radix 정렬을 이용하면 $O(n)$으로 계산이 될 수 있다. REIT가 모든 경우에 최적의 해로 항상 수렴한다는 것을 증명할 수는 없으나 실험을 통해 ITERATE 보다 안정적 클러스터링 결과를 보다 빠르게 산출할 수 있다는 것을 보여준다.

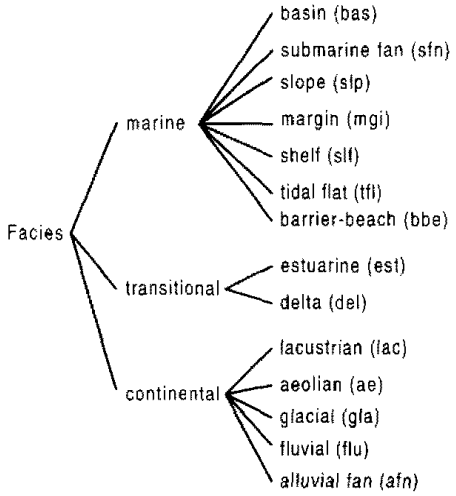
4. 자료 집단 및 불완전 정보 처리

실험을 통해 REIT가 분류를 얼마나 효율적으로 수행하는가와 자료 순서가 미치는 영향을 조사하기로 한다. 다음과 같은 3개의 자료 집단(<표 1>)에 대해

REIT를 적용하였다 : ① 콩(soybean)의 질병 분류 ② 붓꽃(iris) 분류 ③ 퇴적층 분류(facies). 처음 두 개의 자료 집단은 UCI 자료 창고[13]에서 수집하였고, 마지막 자료는 지질학적 분석 자료로 [12]로부터 수집한 자료이다. 세 번째 자료는 실제 지질학 영역에서 전문가에 의해 수집된 자료로 본 연구의 실용적인 관점에서 제안된 알고리즘의 평가를 위해 사용하기로 한다.

〈표 1〉 자료 세트

자료 세트	알려진 분류	자료수	속성수
soybean	4가지 질병 : D1, D2, D3, D4	47	35
iris	setosa virginica versicolor	150	4
facies	14개의 facies 집단	188	16



(그림 3) Facies 구조

Facies라는 지질특성은 14개의 facies 구조((그림 3))를 갖고 이를 구분하는 속성은 16개로 이뤄져 있다. 이들로부터 만들어진 규칙이 144개가되므로 facies를 분석하여 분류하는 작업이 얼마나 어려운 일인지를 짐작할 수 있다. 다중 결론을 갖는 규칙은 개별 결론으로 나눠 별개의 자료 객체로 나눈다. 이렇게 나눴을 때 총 188개의 자료 객체가 얻어졌다. Facies 규칙들을 REIT 알고리즘에 맞는 자료형태로 전환했을 때 약 60~70% 정도가 값을 갖지 않은 자료형태를 취한다. 이같이 체계적 불완전 정보의 처리를 위해 카테고리 유틸리티를 수정한다. 우선 카테고리 유틸리티를 측정하기 위한 수식에서 $P(A_i = V_{ij})$와 $P(A_i = V_{ij}|C_k)$를 다음과 같이 재정의한다.

$P(A_i = V_{ij})$은 $\frac{A_i = V_{ij}\text{를 갖는 } object \text{ 수}}{\text{관측된 } A_i\text{를 갖는 } object \text{ 수}}$로 재정의 된다. 관측된 자료에 대해 처리하기 위해 salience 요소[6]를 정의하는데 이것은 속성에 대한 값이 특성 클래스에 대해 관측될 확률을 나타내게 한다. 이를 고려한 카테고리 유틸리티는 아래와 같이 정의된다 :

$$P(c_k)\sum_i P(\text{관측된 } A_i|C_k)\sum_j P(A_i = V_{ij}|C_k)^2 - \sum_i P(\text{관측된 } A_i)\sum_j P(A_i = V_{ij})^2.$$

여기서

$$P(\text{관측된 } A_i|C_k) = \frac{A_i\text{에대해 관측값을 갖는 } C_k\text{의 } object \text{ 수}}{C_k\text{의 } object \text{ 수}}$$

5. 실험 결과

5.1 실험내용

순서 바이어스에 의한 영향을 평가하기 위해 기존의 ITERATE의 결과와 REIT의 결과를 비교 분석하기로 한다. 자료 객체 입력 순서에 따라 다른 결과를 생성하는 것이 자주 발생하므로 자료의 입력 순서를 결정하기 위해 최선, 최악, 무작위 정렬의 3가지 경우로 구분하여 분석하기로 한다. 최선의 경우는 ITERATE에서 시도하는 비유사도에 의한 자료 객체 정렬을 의미하는 것이고 최악의 경우는 최선에 반대되는 경우로 가장 유사한 자료 객체를 연속적으로 배열하는 방법이다. 무작위 경우는 주어진 자료 세트로부터 자료 객체를 무작위로 추출하여 배열하는 방식이다. 최선과 최악의 경우 많은 자료 객체가 동일한 비 유사도 값을 갖는 경우가 종종 있어 한가지 배열로 정렬되는 것이 아니다. 따라서 최선과 최악으로 정렬하기 이전에 무작위로 배열하여 서로 다른 최선과 최악의 배열을 갖도록 하였다. 각 경우 20회의 실험을 거쳐 그것이 산출하는 결과를 요약하여 ITERATE와 중심 정렬을 사용하는 REIT의 결과를 비교 분석한다.

5.2 soybean

일반적으로 예상되는 soybean에 대한 분류는 4개의 클래스 레이블(D1, D2, D3, D4로 표시)로 이미 잘 알

나게 되었으므로 4개의 그룹으로 나뉘지는 것을 예상할 수 있다. 그러나 실제로 생성되는 클래스는 두 가지 경우로 나타난다. 4개의 클래스 경우와 3개의 클래스 경우로 나뉘진다(<표 2>). 3개의 클래스로 나뉘지는 이유는 D3와 D4에 속하는 자료 개체들이 서로 유사하기 때문이다.

〈표 2〉 soybean 분류 클래스

분 류	클 래 스	분류점수
4개 클래스	C1(D1:10), C2(D2:10), C3(D3:10), C4(D4:17)	1.389
3개 클래스	C1(D1:10), C2(D2:10), C3(D3:10,D4:17)	1.468

〈표 3〉 재배치 회수

	재배치 회수				평균	최적 수렴회수	중심정렬 재배치평균
	1	2	3	4			
최선	14	5	1	0	1.35	4	1.95
최악	0	0	0	20	4.0	20	1.0
무작위	11	9	0	0	1.45	13	1.25

최악의 경우에 대해 실험한 결과 모두 4번의 재배치 과정을 거쳐 3개의 클래스로 나뉘졌다(<표 3>). 4개의 클래스로 나뉘진 결과의 분류 점수는 3개의 클래스를 갖는 분류의 분류 점수에 비해 적으나 결과론적으로 D3과 D4가 혼합되어 있어 동일한 객체로 이뤄진 클래스가 아닌 것으로 나타났다. COBWEB 계열의 분류 시스템은 모두 카테고리 유틸리티에 의한 평가함수를 사용하고 이것의 목적은 분류 전체에 걸쳐 평균 예측력을 최대로 하는데 목적이 있다. 이 기준에 따라 가장 높은 카테고리 유틸리티를 갖는 분류가 최적의 해로 성의하였다. 두 가지 클래스 분류에 대한 상세한 평가는 [5]에 자세히 설명되어 있다. 최적에 이르는 회수는 최악의 정렬의 경우가 가장 높았다. 물론 최적의 결론에 도달하는데 드는 비용이 4회의 재배치 과정을 겪은 후에 일어난 결과이다. 또한 각 경우에 중심정렬을 적용한 결과 60회 모두 최고의 분류점수를 갖는 분류가 일어났다.

재배치가 적용된 회수(<표 3>)를 살펴보면 최선이나 무작위 경우에는 큰 차이를 볼 수 없었으나(1.35 대 1.45) 최악의 경우는 매우 높은 수치(4.0)를 나타냈다. 만일 중심정렬에서 생성되는 순서로 자료 객체가 배열된다면 평균 1.25회의 더 빠른 재배치로 최적의 결과에 도달할 수 있었을 것이다. 따라서 ITERATE에서 사용하는 최선의 정렬보다 좀더 안정적이면서 효율을 높일 수 있는 자료 객체 순서정렬이 있다는 것을 지적해 주는 경우라 할 수 있다.

〈표 4〉 iris 분류

분 류	클 래 스	분류점수	회수
분류 I	C1(seto:50) C2(vers:35, virg:2) C3(vers:15, virg:48)	0.460	10
분류 II	C1(seto:50) C2(vers:25) C3(vers:25, virg:50)	0.446	2
분류 III	C1(seto:50) C2(vers:2, virg:46) C2(vers:48, virg:4)	0.459	5
분류 IV	C1(seto:50) C2(vers:20, virg:48) C3(vers:30, virg:2)	0.455	3
분류 V	C1(seto:50, vers:5) C2(vers:45, virg:50)	0.442	최악
분류 VI	C1(seto:50) C2(vers:25) C3(vers:23, virg:4) C4(vers:2, virg:46)	0.396	최선

5.3 iris

iris 자료 세트는 3개의 클래스 레이블이 있다 : setosa (seto), virginica(virg), versicolor(vers). 20번의 다른 무작위 정렬에 대해 ITERATE를 실행하였을 때 <표 4>와 같은 4가지 분류로 결과가 산출되었다. 마지막에 나타나 있는 분류 V와 VI은 무작위 경우에서 발견되지는 않았으나 최악의 정렬과 최선의 정렬을 실행하는 과정에서 각각 발견되었다. <표 4>로부터 확인할 수 있듯이 분류 점수가 가장 높은 분류 I이 최적의 해임을 알 수 있다. 최선의 정렬의 경우 17개의 경우가 분류 I로 수렴하였고 중심 정렬을 적용하는 경우 모두 분류 I을 산출하였다(<표 5>). 최악의 정렬인 경우 하나의 케이스만 빼놓고 모두 분류 IV를 산출하였는데 이는 같은 클래스 레이블을 갖는 자료 객체가 연속적으로 입력하는 경우로 트리 구조가 왜곡되어 나타날

뿐 아니라 분류 점수에 있어서도 낮은 값을 갖게 되는 경우이다. soybean 경우와 마찬가지로 비록 최선의 배열이라 하지만 85%만이 최적에 해를 산출한 반면 중심정렬을 이용한 경우 100%의 최적에 해로 귀결되었다. 최선의 배열 중 15%는 최적이 아닌 경우로 판명된 실험이라 할 수 있겠다.

<표 6>은 재배치의 반복 회수를 나타내는 표이다. 무작위의 경우에서 분류 I를 산출하지 않은 10개의 케이스에 대해 2번의 중심 정렬이 요구되며, 나머지 10개의 경우에는 중심정렬이 1회 수행된다. 중심정렬을 이용한 분류가 1회 수행되었다는 것은 초기 분류가 최적에 도달하였다는 것을 확인하는 것으로 더 이상의 분류가 불필요하다는 뜻이다. soybean의 경우와 마찬가지로 자료 객체만을 분석한 최선의 정렬보다 중심정렬을 이용한 분류가 최적의 안정적 결과에 도달하게 보장해 줄뿐만 아니라 중심 정렬을 이용할 경우 평균 2.25회의 재배치가 요구되므로 최선의 정렬이 갖는 2.65보다 더 빠르게 수렴함을 보여준다. 요약하면 자료의 비유사도에만 의존하여 생성된 정렬은 결코 최적의 결과 산출을 확신할 수 없을 뿐만 아니라 효율성에 있어서도 다소 떨어진다는 것을 보여주는 중요한 실험 결과가 된다.

〈표 5〉 클래스별 분류회수

	분 류	회 수
최 선	I	17
	III	1
	IV	1
	VI	1
최 악	IV	19
	V	1
무 작 위	<표 4>	
중심정렬	I	60

〈표 6〉 재배치 회수

	재배치 회수					평균	중심정렬 재배치평균
	1	2	3	4	5		
최악	0	0	1	19	0	3.95	2.4
최선	0	7	13	0	0	2.65	1.65
무작위	3	10	5	1	1	2.35	2.25

5.4 Facies

Facies 자료에 대한 예상된 결과는 고정된 것이 없다. 앞서 기술한대로 60~70%에 대한 자료가 값을 갖지 않은 자료 형태로 있어 앞에서 논의한 자료 세트처럼 명확하게 몇 개로 나열할 수 있는 분류는 없다. (그림 3)은 보통 지질학 교육용 자료에서 볼 수 있는 분류로 3가지 부류로 나눠지고 각각의 더 세분화된 계층구조로 분류 구조로 나타나는데 이것은 형성과정 관점에서 단순 분류한 것이기 때문에 특성에 따른 분류를 한다면 이처럼 명확하게 분류되지 않을 수 있다. 실제로 가장 높은 분류 점수 (1.089)를 갖는 분류가 <표 7>에 나타나 있다. 각 facies 명칭 옆에 facies의 수를 표시하고 있다.

〈표 7〉 Facies 분류

C1	aeo:3 lac:4 gla:4 flu:8 afn:3 del:22 bbe:2 shf:3 sfn:14 bas:4 est:9	C2	aeo:1 flu:2 del:7 bbe:4 tfl:8 shf:6 slp:14 sfn:8 bas:5	C3	aeo:2 lac:14 gla:4 flu:21 afn:6 del:3 shf:7 mgi:1 bas:3 est:4

<표 8>은 재배치회수와 분류 점수의 변화를 나타내는 표이다. 각 경우에 대해 3번의 중심정렬을 적용하여 재배치 회수와 분류점수의 변화를 조사하였고 이를 20번 시행하여 그 값을 평균해 나온 결과표이다. 중심정렬을 3회에 걸쳐 강제적으로 수행한 이유는 앞선 자료 세트처럼 뚜렷하게 수렴되는 결과가 없기 때문이다. 실제 알고리즘에서는 분류 점수가 감소하는 경우에 반복이 종료된다. 이 표를 통해서 확인할 수 있는 사항이 몇 가지 있는데 첫째로 재배치 회수의 변화에서 최악과 최선의 정렬의 경우에는 중심 정렬을 적용한 분류가 더 많은 재배치 회수를 수행하였으나 무작위의 경우 중심 정렬을 이용한 분류에서 재배치 회수가 줄어듦을 알 수 있다. 최악의 경우 비록 재배치 회수는 증가하였으나 분류점수는 중심정렬 1에서 증가하다가 다음 중심정렬 2에서 감소로 돌아섰다. 반면 최선의 경우에서는 초기 분류가 최소의 재배치 회수를 기록하였고 분류 점수도 높게 산출되었으나, 전체적으로 최악의 경우가 평균적으로 가장 높은 분류점수를 기록하였다. 무작위의 경우에서는 중심정렬 1단계에서 최소의 재배치 회수를 기록하였고 분류 점수도 최대를 기록하였다. 이 경우와 최악의 정렬을 이용한 분류에

서 중심 정렬이 중요한 역할을 담당하였다는 것을 보여준다.

〈표 8〉 재배치 회수와 분류 점수

	재배치회수				평균 분류점수			
	ITER-ATE	중심 1	중심 2	중심 3	ITER-ATE	중심 1	중심 2	중심 3
최악	10.5	11.6	12.9	11.6	0.715	0.95	0.85	0.90
최선	10.2	11.6	11.8	11.5	0.883	0.86	0.87	0.68
무작위	13.5	11.9	12.4	13.2	0.853	0.92	0.84	0.83

Facies 자료 세트에 대한 분류에서 각 경우 20회씩 다른 자료 배열을 갖고 분류를 시도하였으나 같은 결과를 산출하는 경우는 없었다. 이것은 자료가 많은 부분에 걸쳐 값을 갖지 못하였기 때문이고 실제 지질학 분야는 배경 이론이 다른 분야와 달리 매우 모호하다고 할 수 있다. 전문가 사이에서도 용어나 지질학적 모델이 서로 다르게 정의되어 있을 정도로 모호하다고 할 수 있다. 따라서 이런 자료로부터 얻어지는 새로운 개념 역시 같은 문제를 겪게되는데 이를 보상하는 방법의 하나로 REIT와 같은 개념적 클러스터링을 이용하게된다[14]. 그리고 최종 결정은 지질학 분야의 전문가에 의해서 평가될 수 있을 것이다.

6. 결론 및 논의

본 연구에서는 자료의 처리 순서에 따라 클러스터링 결과가 달라지는 문제를 집중적으로 조사 분석하였고 COBWEB의 카테고리 유틸리티에 근거하여 순서 바이어스 문제를 해결하려한 ITERATE를 개선하여 보다 안정적 클러스터링이 되도록 시도하였다. ITERATE가 비유사도에 따른 자료 객체 정렬을 시도한 후 재배치 연산을 통해 분류를 시도하였으나 계산 비용이 비싼 이 정렬과정을 제거하고 대신 초기 분류에서 얻어진 그룹을 대상으로 중심 정렬을 시도하고 이를 통해 다시 분류를 시도하여 개선된 결과를 얻었다. ITERATE는 최선의 정렬을 통해 COBWEB과 같은 점진적 클러스터링 시스템의 순서 영향을 줄일 수 있었으나 근본적으로 간과한 것이 있다면 자료를 분석하여 순서화하였다 하더라도 예상되는 클래스의 수를 알지 못한 상황에서 자료간 비유사도만을 가지고 인위적으로 순서화하나의 최적의 안정적 결론에 도달하지 못할 수 있음을 보였다. REIT에서는 자료간의 비유사도에 의한 배열 외에도 클래스에 대한 초기 정보를 활용하여 자료 입력 순서를 결정하였기 때문에 순서 영향을 크게 줄일 수 있는 기회를 가진 것으로 분석된다. 개념의 정립이 모호한 facies 같은 경우에 있어서는 무감독 개념 학습 시스템이 매우 유용하게 사용될 수 있으므로 보다 안정적 결과에 도달하는데 REIT가 기여를 하였다고 할 수 있다.

참 고 문 헌

[1] Chen, M., J. Han, and P.S. Yu, "Data mining : An overview from database," IEEE Transaction on Knowledge and Data Engineering, 1997.

[2] Fayyad, U., "Data mining and knowledge discovery in databases," CACM, Vol.39, No.11, Nov., 1996.

[3] McKusick, K.B., Langley, P., "Constraints on tree structure in concept formation," Proc. 12th Int. Joint on AI, Sydney, Australia, pp.810-816, 1991.

[4] Fisher, D., "Iterative Optimization and Simplification of Hierarchical Clustering," Journal of AI Research, 4, pp.147-179, 1996.

[5] Biswas, G, Weinberg, J.B. and Fisher, H.D., "ITERATE : A conceptual clustering algorithm for data mining" IEEE Tr. on systems, man and cybernetics, Vol.28, part C No.2, May 1998.

[6] Fisher, D.H., "Knowledge acquisition via incremental conceptual clustering," Machine Learning 2, pp.139-172, 1987.

[7] Michalski, R.S., and Stepp, R.E., "Learning from observation : Conceptual clustering," Machine Learning: An AI approach, R.S. Michalski, J.G., et al.(Eds.), Morgan Kaufmann 1983.

[8] Lebowitz, M., "Experiments with incremental concept formation : UNIMEM," Machine Learning, 2, pp.103-138, 1987.

[9] Cheesman, P., et al., "AutoClass : A Baysian classification system," Proceedings of the Fifth International Conference on Machine Learning,

Ann Arbor, MI, 1988. pp.54-64.

[10] Jain, A.K. and Dubes, R.C., Algorithms for cluster analysis, Englewood Cliffs, NJ : Prentice Hall, 1988.

[11] Gluck, M. and Corter, J., "Information, uncertainty, and the utility of categories," Proceedings of the Seventh Annual Conference of the Cognitive Science Society, Irvine, CA, pp.283-287, 1988.

[12] Cheong, D.K., et al., "PLAYMAKER, a knowledge based expert system," Geobyte, Vol.7, No.6, pp. 28-41, 1992.

[13] UCI repository, http://www.ics.uci.edu/~mlearn

[14] Biswas, G and Lee, G, "Knowledge reorganization : A rule model scheme for efficient reasoning," Proc. 10th IEEE conf. on AI for Applications, San Antonio, TX, mar. 1994.

이 계 성

e-mail : gslee@cs.dankook.ac.kr

1980년 서강대학교 전자공학과 졸업(학사)

1982년 한국과학기술원 전산학과 졸업(석사)

1994년 Vanderbilt 대학 전산학과 졸업(박사)

1994년~1996년 대구대학교 전산정보학과 전임강사

1996년~현재 단국대학교 전산학과 조교수

관심분야 : 기계학습, 데이터마이닝, 에이전트