공학석사학위논문

공유 정보와 다구치 방법을 이용한 신경회로망에서의 입력특징 선택

Input Feature Selection for Neural Networks using Mutual Information and Taguchi Method

1999년 2월

서울대학교 대학원

전기공학부

곽 노 준

요약문

신경회로망에서의 특징 선택은 분류에 사용하는 학습 알고리즘에 상관없이 학습집단의 분류성능에 매우 중요한 요소이다. 신경회로망의 분류문제에서 분류에 상관이 없거나 중복된 특징들이 존재하므로 이들을 제거하여 중요한 특징들만 선택해 학습시킴으로써 보다 나은 학습성능을 가져올 수 있다. 본 논문에서는 신경회로망의 여러개의 입력 특징들 중에서 분류 과정에 중요하게 작용하는 입력 특징을 선택하는 신경회로망에서의 입력특징 선택 문제를 다루었다.

본 논문에서 제안하는 신경회로망을 위한 입력 특징 알고리즘은 크게 두 개로 첫째는 공유정보를 이용하는 MIFS 알고리즘의 성능을 향상시킨 알고리즘이고 둘째는 다구치 방법(Taguchi method)을 이용한 알고리즘이다.

개선된 MIFS 방법은 기존의 MIFS 방법의 중복된 특징을 제거할 수 있는 장점은 살리고 단점은 없애 일반적인 문제에 대해 좋은 성능을 내도록 설계되었다. 이 알고리즘을 여러 입력 선택 문제에 적용해 본 결과 기존의 MIFS 방법보다 일반적으로 우수한 성능을 낸다는 것을 확인할 수 있었다.

다구치 방법을 이용하는 입력 특징 선택 알고리즘은 공유 정보를 이용하는 알고리즘과는 별도로 어떻게 하면 신경회로망의 학습 횟수를 줄이면서 좋은 입력 특징을 선택할 수 있는가 하는 문제에 대한 하나의 해답으로서 제시되었다. 여러 가지 입력 특징 선택 문제에 이 방법을 적용해 본 결과 기존의 방법보다 좋은 성능을 얻을 수 있었다.

주요어 : 신경회로망, 입력특징 선택, 다구치 방법, 공유정보, 직교 행렬

학번 : 9 7 4 2 0 - 5 0 6

목 차

그림 목차

표 목차

1

서론

최근 공학에서는 인간이 지니고 있는 여러 가지 능력을 다양한 분야에 응용하려는 연구가 활발히 진행되고 있다. 이러한 인간의 능력 중에서 특히 효율적인 패턴인식(pattern recognition) 방법을 컴퓨터에 적용하기 위해 인간 두뇌 세포 구조를 공학적으로 모델링한 신경회로망(neural network)이 주목을 받고 있다. 주어진 패턴인식 문제를 해결하기 위해 신경회로망은 인간이 가지고 있는 학습 능력을 모방한 알고리즘을 사용한다. 신경회로망은 이 알고리즘을 통해 입력 패턴(input pattern)과 출력 패턴(output pattern) 사이의 관계를 나타낼 수 있다. 또한 신경회로망은 기존의 학습(learning)에 사용하지 않았던 새로운 입력 패턴에 대해 비교적 올바르다고 생각되는 출력을 내는데 이 때문에 신경회로망은 기존의 순차적 프로그래밍(sequential programming)에서는 해결할 수 없는 복잡하고 다양한 문제에 활발히 적용되고 있다.

신경회로망을 학습시키는 일반적인 방법은 다층 인식자 (multi-layer perceptron ; 이하 MLP) 를 오차 역전파(error back-progation; 이하 BP) 방법을 이

용해 입력 데이터와 출력 데이터를 서로 사상 (mapping) 시키는 것이다. 이렇게 학습된 신경망을 가지고 학습에 사용되지 않은 데이터로 확인인함으로써 문제를 잘 해결했는지의 여부를 검증한다. 하지만 이러한 문제 해결 접근 방법은 몇 가지 한계가 있다. 첫째, 학습에 사용되는 규칙에 따라서 신경망의 구조가 결정되지 않는다. 둘째, 입력 벡터 공간의 차원이 커질 경우 많은 데이터를 필요로 한다. 이런 경우 학습 전에 미리 입력 벡터 중의 일부를 선택해서 새로운 입력벡터를 구성하는 것이 더 효과적일 수 있다. 마지막으로 만약 만족할 만한 결과가 도출되지 않았을 경우 신경회로망의 특성상 더 이상의 분석이 어려워진다.

이러한 문제를 극복하기 위한 하나의 방안으로 입력벡터의 차원을 줄여 보다 빠르고 효율적인 학습을 할 수 있도록 하는 방법을 생각할 수 있다. 추출된 특징들로부터 출력에 큰 영향을 미치는 특징들만을 선택해 새로운 입력 벡터를 구성하려는 시도가 많이 이루어지고 있으며 이를 특징 추출 문제와는 별도로 특징 선택 (feature selection) 문제라 하고 최근에 이에 관한 연구가 data mining 문제와 연관되어 신경회로망을 연구하는 사람들로부터 새로운 연구 분야로 주목받기 시작했다.

패턴 분류 문제에서 신경회로망을 학습시키려면 각각의 입력 패턴과 그에 따른 출력 패턴이 있어야 한다. 여기서 입력 패턴은 N-차원 특징 벡터(feature vector)로 그야말로 그 개체의 특징을 나타내고 있어야 한다. 그러나 실제 패턴 분류 문제에서 어떤 특징(feature)들을 모아 입력 벡터(input vector)로 사용할지는 적용되는 문제에 따라 다르고 또 어떤 특징을 찾아냈다고 하더라도 과연 찾아낸 특징이 패턴을 분류하는 데 어느 정도의 기여를 하는 지 알 수가 없다. 어떤 특징이 필요한지 잘 모르기 때문에 실제로는 패턴 분류를 잘 하기 위하여

필요하다고 생각되는 특징들을 입력으로 모두 사용하게 된다. 이렇게 되면 신경회로망의 구조가 매우 복잡해지며 경우에 따라서는 신경회로망의 분류 성능을 떨어뜨릴 수 있다. 또한 실시간으로 특징을 찾아내고 처리해야 하는 문제에서는 계산량을 줄이는 것이 대단히 중요하므로 요구되는 성능을 주면서 필요한 특징만을 사용하는 신경회로망을 구현하는 것이 필요하다. 따라서 패턴 분류 문제에서 각각의 특징의 중요성을 측정하고 특징들을 조합해 보다 나은 성능을 갖는 새로운 특징 벡터를 얻는 체계적인 방법의 연구가 절실히 요구된다. 여러 가지 입력 특징들 중에서 패턴 분류에 중요한 역할을 하는 특징(salient feature)들을 골라내 이들로 적은 차원의 입력 벡터를 구성할 수 있다면 이를 사용해 좋은 분류 성능을 얻을 수 있을 것이다.

본 연구에서는 신경회로망에서 보다 좋은 특징 선택 방법을 제시하고자 한다. 신경회로망에서의 특징 선택은 분류에 사용하는 학습 알고리즘에 상관없이 학습집단의 분류성능에 매우 중요한 요소이다. 여러 개의 특징을 이용해 집단을 분류하고자 할 때 분류에 상관이 없거나(irrelavant) 중복된(redundant) 특징들이 존재하므로 이들을 제거하여 분류에 중요한 영향을 미치는 특징들만 추출해 학습시킴으로써 보다 나은 학습성능을 가져올 수 있다.

이 분야의 연구 중 가장 오래되고 또 가장 많이 알려진 방법으로 PCA (principal component analysis) [1]가 있는데 이는 여러 특징들을 선형 변환해서 가장 중요한 특징들을 새로 만드는 방법이다. 이 방법의 가장 큰 단점은 변환(transform)에 대해서 불변(invariant)이 아니라는 점이다. 따라서 입력 특징들을 선형적으로 스케일링하는 것만으로도 PCA 결과는 바뀌게 된다. PCA 이외에는 현재까지 이 분야가 연구의 중요성에도 불구하고 문제의 복잡성으로 인

해 학문적으로 커다란 성과 없이 미지의 영역으로 남아 있었다.

최근 이 문제가 여러 사람에 의해 본격적으로 다루어지기 시작했고 그 결과 여러 가지 방법이 제시되었다 [2]–[8]. 그 중 대표적인 방법으로 결정 나무구조(decision tree)를 이용해 중요한 특징을 차례대로 찾아내는 방법들 [6]–[8] 이 있고 이는 요즘 들어 관심을 끌고 있는 database mining 문제 [8], [9] 와도 밀접한 관련성이 있다. 이 결정 나무구조를 이용하는 알고리즘을 보완해 Setiono 등은 신경회로망에 입력되는 여러 특징들을 하나하나 빼 가며 반복학습시켜 분류에 중요한 특징을 찾는 방법을 제시했다 [6]. 이는 기본적으로 모든 특징들의 조합에 대해서 반복적으로 신경회로망을 학습해야 하므로 특징수가 많아지면 학습에 걸리는 시간이 굉장히 오래 걸리는 단점이 있다. Agrawal등이 data mining을 위해 제시한 CDP 알고리즘 역시 결정 나무구조를 이용하는 알고리즘으로 입출력간의 공유정보(mutual information) [13]를 이용한다 [8]. 하지만 이 알고리즘의 목적은 여러 입력들 중에서 중요한 특징들만을 뽑아내는 것이 아니라 입출력간의 관계를 나타내는 규칙을 찾아내는데 있기 때문에 이 방법을 신경회로망에 그대로 적용하기에는 약간의 무리가 따른다. 입출력간의 공유정보를 이용해 입력 특징을 선택하는 알고리즘으로 MIFS가 있으며 이 방법을 제시한 논문에서 Battiti는 공유정보를 이용하는 것이 입력특징 선택 문제에 적합한 방법임을 보였다 [7].

본 논문은 Battiti가 제시한 MIFS 알고리즘을 기초로 하여 좀 더 성능이 향상된 알고리즘을 제시한다. 이에 더불어 본 연구에서는 다구치가 복잡한 시스템에서의 견실 설계를 위해 제안한 다구치 방법(Taguchi method) [10] [11]을 신경회로망에서의 입력특징 선택 문제에 적용하고 여러 가지 실험을 통해서 새롭게

제시된 알고리즘의 성능을 분석하고 검증해 본다.

본 논문의 구성은 다음과 같다. 제 2장에서는 앞에서 간단히 언급한 기존의 특징 선택 알고리즘들을 살펴보고 각 알고리즘의 장단점에 대해 논한다. 제 3장에서는 공유정보와 다구치 방법에 대해 좀 더 자세하게 알아본다. 제 4장과 5장에서는 이를 이용해 새로운 입력특징 선택 알고리즘을 제시하고 제안된 알고리즘의 성능을 여러 가지 모의 실험을 통해 다른 알고리즘들과 비교 분석한여 마지막으로 제 6장에서는 여기서 제시한 알고리즘이 신경회로망에서의 입력특징 선택 문제에 매우 적합한 방법임을 보임으로써 결론을 맺는다.

2

기존의 입력 특징 선택 방법

신경회로망에서 여러 입력 특징들 중에서 가장 중요한 몇 개의 입력 특징만을 선택하여 최고의 성능을 얻으려고 한다면 입력 특징들로부터 만들 수 있는 모든 입력특징 벡터 조합을 각각 학습시키고 그들 중 최고의 성능을 내는 것을 입력 벡터로 삼으면 된다. 하지만 이 방법은 입력벡터가 N 차원일 때 $O(N^N)$번의 신경회로망 학습이 요구되므로 입력벡터의 차원이 커지면 최적의 입력벡터를 찾는다는 것은 거의 불가능해진다 [1] .

일반적으로 이 문제를 해결하는 방법으로 다음의 두가지 방법을 생각할 수 있다. 첫째 하향적 방법 (top-down) 으로서 먼저 전체 입력특징들로 신경회로망을 학습시키고 하나씩의 특징을 뺀 N개의 신경회로망을 학습시켜 이 중에서 분류 성능이 가장 좋은 것을 선택한 후 다시 이 $N-1$개의 특징들 중 하나씩을 빼서 학습하고 계속 이 과정을 반복해 나가는 것이다. 둘째로 이 방법과는 반대로 먼저 가장 중요하다고 생각되는 특징을 선택한 후 여기에 다른 특징들을 중요하다

[1] $\sum_{i=1}^{N} {}_N C_i \approx O(N^N)$

고 생각되는 순서대로 더해 나가는 방법이다. 이는 상향적 방법(bottom-up)이라 할 수 있겠다.

다음에 제시된 기존의 입력특징 선택 방법들은 모두 이 두 가지 방법 중 하나를 택하고 있다.

2.1 Neural-Network Feature Selector

Setiono 등에 의해 제시된 이 알고리즘은 하향적 방법에 해당하는 것으로 전체 입력 특징들을 모두 사용하여 신경회로망을 학습한 후 분류에 별로 영향을 못 미친다고 생각되는 특징들을 순서대로 빼 가면서 학습시키는 방법이다 [6]. 따라서 이 방법은 총 입력수가 N개일 때 $\Sigma_{n=1}^{N} n$ 즉, $O(N^2)$ 회의 학습이 요구된다. 일반적으로 신경회로망의 학습에는 오랜 시간이 걸리므로 신경회로망을 $O(N^2)$회 학습 학습해야 하는 이 방법 역시 학습 속도가 관건이 된다. 따라서 이 알고리즘에서는 학습 속도 향상을 위해 기존의 BP대신에 준뉴톤(quasi-Newton) 방법의 일종인 BFGS (Broyden-Fletcher-Shanno-Goldfrab) 방법을 학습에 이용한다. 또 작은 연결가중치(connection weight)를 갖는 특징이 출력에 영향을 덜 주게 하기 위해서 목적함수에 페널티(penalty) 함수를 더한다.

이 알고리즘을 간략히 기술하면 다음과 같다.

- 알고리즘

  1. 입력 특징 집합을 전체 입력 특징들로 놓는다.

  2. 입력특징 집합에 속해 있는 전체 특징들로 신경망을 학습시킨다.

3. 전체 입력 특징 집합에서 1개씩의 특징을 뺀 각각의 신경망을 학습시키고 각 신경망의 학습 성취를 구한다.
4. 분류율이 높은 신경망순으로 배열한다.
5. 입력 특징 집합에서 분류율이 제일 낮은 특징을 제외한 새로운 입력 특징 집합을 구성한다.
6. 알고리즘을 끝내는 조건에 맞지 않으면 2번으로 돌아간다.

이 알고리즘의 가장 큰 문제점은 중요하지 않다고 판단되는 특징을 차례로 제외하는 과정에서 네트웍을 계속 학습시켜야 하므로 입력 특징을 선택하는 데 시간이 오래 걸린다는 것이다. 또 작은 연결 가중치를 0 근처의 더 작은 값으로 만들어주기 위해 목적함수에 더하는 페널티 함수의 사용으로 학습에 바라지 않는 오차가 발생하며 특징을 선택해 나갈수록 이 오차가 무시못할 정도로 쌓이게 된다. 따라서 이 방법은 실생활에서 부딪히는 입력 공간의 차원이 10개 이상인 문제의 입력 특징 선택을 위해서는 적용하기가 힘든 것으로 보인다.

2.2 CDP (Classifier with Dynamic Pruning)

이 절에서 설명할 Agrawal 등의 CDP와 다음 절에서 설명할 Battiti의 MIFS 알고리즘은 둘 다 공유 정보를 이용해 입력 특징을 선택하는 방법이다. 공유 정보에 대해서는 다음 장에 자세히 설명하기로 하고 여기서는 이들의 대략적인 알고리즘을 살펴보기로 한다.

CDP는 Agrawal등이 data mining의 목적으로 제시한 알고리즘으로서 결정나무구조를 이용하여 상향식으로 입력 특징을 선택한다 [8]. Data mining이란

주어진 많은 양의 데이터에서 의사결정 혹은 특수한 목적의 일을 처리하는 데 있어 유용한 정보를 찾아서 데이터 간에 일정한 규칙을 만들고자 하는 것으로 최근에 컴퓨터의 정보 처리 속도 및 데이터 양이 엄청나게 증가하면서 빠르게 부각되고 있는 분야이다.

기본적으로 CDP는 data mining을 목적으로 하기 때문에 선택된 입력 특징들로부터 출력과의 관계를 맺어주는 일정한 규칙을 찾아낸다. 따라서 입력벡터와 출력벡터를 사상시켜준다는 점에서는 신경회로망과 유사하지만 입출력간의 정확한 관계를 분석하기 어려운 신경회로망의 특징상 이러한 정확한 입출력간의 관계를 알아내는 작업을 입력 선택 과정에서 수행하는 것은 불필요하다. 따라서 CDP의 결과를 그대로 신경회로망의 입력으로 사용하는 데는 어느 정도의 무리가 따른다. 이 알고리즘을 기술하기 전에 우선 스트링(string)을 입력 특징과 그 특징이 갖는 값의 쌍이라 정의하자. 예를 들면 (나이, 20), (나이, 24), ... , (성별, 남), (성별, 여), ... 와 같이 표현되는 것이다. 이제 이 알고리즘을 간략히 표현한 것이 다음이고 그림 2.1는 이를 블록 다이어그램으로 나타낸 것이다.

- 알고리즘

    1. Generate & Measure : 입력 패턴의 각 특징마다 모든 스트링을 만들고 각 스트링 별로 몇 개가 해당하는지 센다.
    2. Combine : 연속변수를 갖는 특징은 출력과의 공유정보가 최대가 되도록 비슷한 값을 갖는 것끼리 묶어서 하나의 분할 (partition)을 만든다. 이 때 공유정보를 구하기 쉽게 하기 위해 전체 공간을 두개의 분할로 나눈다.

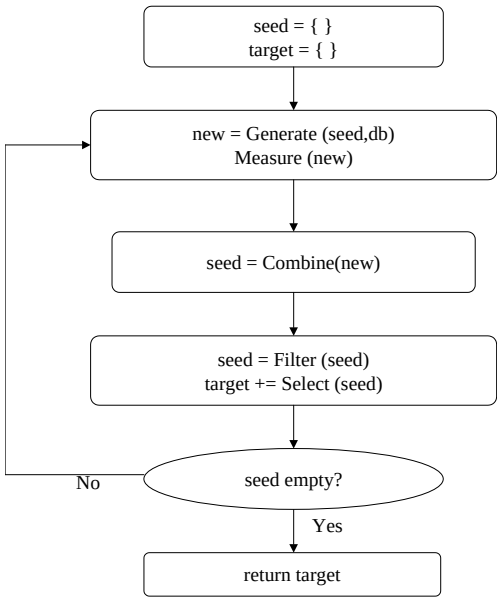


그림 2.1: CDP 알고리즘

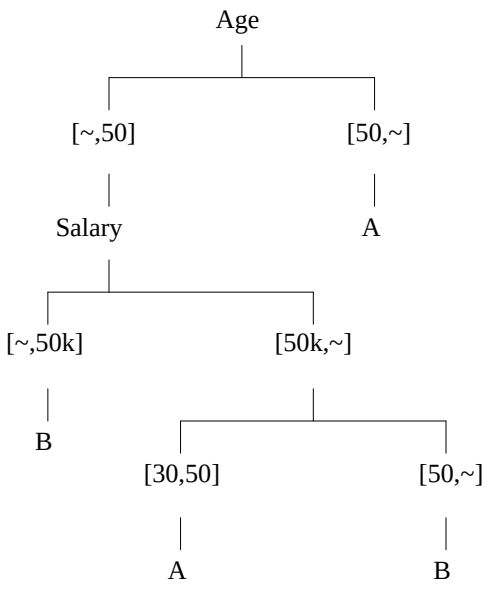


그림 2.2: CDP 결과의 예

3. Filter : 각각의 특징으로부터 얻은 스트링 중 공유 정보가 가장 큰 것만 남긴다.

4. Select : 필터링을 거친 스트링을 목적집합에 포함시킨다.

이 알고리즘을 사용해서 얻어지는 결정 나무구조의 예를 그림 2.2 보였다.

이 알고리즘의 문제점은 먼저 모든 패턴과 모든 입력 특징에 대해서 스트링을 만들어야 하므로 메모리가 부족할 수 있다는 점이다[2]. 또 여러개의 입력 특징들의 조합으로 출력이 결정되어질 경우 나무 구조의 깊이(depth)가 길어져 성능이 떨어진다. 이 알고리즘을 신경회로망에 적용하기 부적합한 가장 큰 이유는

[2]스트링 수 = pattern 수 × 특징수

그림 2.2에서 보는 바와 같이 같은 특징이 여러 번 반복해서 나타날 수 있다는 점이다. 따라서 이 방법처럼 입출력의 관계를 입력특징 선택 단계에서 정확하게 구할 필요 없이 분류에 중요하다고 생각되는 특징들만을 선택한 후 신경회로망을 이용해 학습시킨다면 더 좋은 성능을 얻을 수 있을 것이다.

2.3 MIFS (Mutual Information Feature Selector)

이 알고리즘도 앞 절에서 설명한 CDP와 마찬가지로 공유정보를 사용하고 입력 특징을 하나씩 더해 나가면서 입력 벡터를 구성하는 상향식 방법의 일종이다. [7]. 하지만 이 알고리즘은 CDP와는 달리 출력에 중요하게 작용한다고 생각되는 입력을 선택하고 선택된 순서대로 나열한다는 점에서 본 논문에서 연구하고자 하는 내용에 좀 더 가까운 알고리듬이다.

Battiti의 방법은 각각의 입력 특징들과 출력간의 공유정보를 구하고 중복된 입력 특징을 제거하기 위해 입력 특징들간의 공유정보를 이용해서 순서대로 입력 특징을 선택한다. 여기서 입력 특징들 간의 공유정보가 크다는 것은 입력 특징들이 가지고 있는 정보가 정보공학적 입장에서 중복이 많이 되어 있다는 것(redundancy)을 의미한다. 따라서 이러한 중복성을 제거하기 위해 이 알고리즘에서는 입력 특징들간의 공유정보를 활용한다. 알고리즘을 살펴보기 전에 몇 가지를 가정해 보자. 먼저 전체 입력 특징이 N개라고 가정하고 각각의 입력들을 $f_1, f_2, \ldots, f_N$, 출력을 C라 하자. 또 X와 Y라는 변수가 있을 때 이들의 공유정보를 $I(X;Y)$로 표현하기로 하자. 이제 MIFS 알고리즘은 다음과 같다.

- 알고리즘

1. (초기화)

   $F \longleftarrow$ "초기의 N개의 입력 특징",

   $S \longleftarrow$ "빈 집합 (empty set) " 으로 놓는다.

2. (각 입력 특징과 출력과의 공유 정보 계산)

   $\forall f \in F$ 에 대해 $I(C; f)$ 계산

3. (첫번째 특징 선택)

   모든 특징들 중에서 출력과의 공유정보($I(C; f)$)가 가장 큰 특징을 찾는다.

   $F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

4. (특징 선택)

   목적하는 수의 입력을 선택하기까지 다음을 반복한다.

   (a) (이미 선택된 특징과의 공유 정보 계산)

   $\forall (f, s)$, $f \in F, s \in S$ 에 대해 $I(f; s)$ 계산

   (b) (다음번 특징을 선택)

   $\forall f \in F$ 중에서 $I(C; f) - \beta \Sigma_{s \in S} I(f; s)$를 최대화하는 f를 찾고

   $F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

5. 선택된 결과인 S를 입력 특징선택의 결과로 내놓는다.

여기서 β는 입력 특징들 상호간에 가지고 있는 중복성을 줄여주는 변수로 $\beta = 0$일 때는 입력 특징들 간의 공유 정보는 하나도 고려하지 않고 단순히 입력 특징들과 출력과의 공유 정보가 높은 순서대로 입력 특징을 선택하게 된다. β값이 커질수록 입력 특징들 간의 공유 정보를 입력 선택에 많이 반영하게 되고

이로서 중복성은 줄어들게 된다. 하지만 β값이 너무 커지면 입력과 출력 사이의 관계는 거의 무시되고 입력들끼리 얼마나 많은 정보를 공유하는지만 고려하는 꼴이 되므로 알고리즘이 잘 동작하지 않게 된다.

MIFS 방법은 처음으로 본격적으로 공유 정보라는 개념을 신경회로망의 입력 선택 문제에 이용하였다는데서 의의가 크다. 하지만 이 방법은 비선형성이 강한 XOR문제 등에서는 가장 중요한 순서대로 입력 특징을 나열하지 못하는 단점이 있다. 제 4장에서는 MIFS 방법의 이러한 단점을 보완한 신경회로망에서의 새로운 입력 특징 선택 알고리즘을 제시한다.

3

공유 정보와 다구치 방법

이 장에서는 본 논문에서 제시하는 알고리즘이 이용할 공유 정보와 다구치 방법에 대해 자세히 살펴보기로 한다.

3.1 공유 정보 (Mutual Information)

3.1.1 공유 정보 (Mutual Information)

공유 정보가 무엇인지를 정의하기 전에 먼저 이를 정의하기 위해 필요한 엔트로피(entropy), 조건부 엔트로피(conditional entropy), 상대 엔트로피(relative entropy) 등에 대해서 알아보자.

- **엔트로피(Entropy)**

정보이론(information theory)에서 다루는 엔트로피라는 개념은 확률 변수의 불확정성의 정도를 표현하는 척도(measure)로 해석할 수 있다. 이산 확률 변

수(discrete random variable) X가 $\mathcal{X}$개의 변위(alphabet)값을 가질 수 있으며 이것의 확률분포함수가 $p(x) = \Pr\{X = x\}$, $x \in \mathcal{X}$ 라고 하자. 이 때 확률 변수 X의 엔트로피 $H(X)$는 다음과 같이 정의된다.

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{3.1}$$

여기서 log는 2를 밑으로 하고 엔트로피의 단위는 bits이다. 예를 들면 앞뒤가 나올 확률이 같은 동전을 던지는 사건이 갖는 엔트로피는 1bit이 된다.

위의 정의를 다른 각도에서 살펴보면 확률변수 X의 엔트로피는 확률분포함수가 $p(x)$일 때 $\log \frac{1}{p(x)}$의 기대값(E_p)으로 볼 수 있다. 따라서 엔트로피 $H(X)$를

$$H(X) = E_p \log \frac{1}{p(x)} \tag{3.2}$$

와 같이 정의할 수도 있다. 이 정의는 열역학에서의 엔트로피의 정의와 연관성을 갖는다.

다음은 엔트로피가 확률분포함수에 따라 어떻게 달라지는지를 보여주는 예이다.

예제 3.1 *확률 변수 X가 다음과 같은 분포를 갖는다고 하자.*

$$X = \begin{cases} 1 & \textit{with probability } p \\ 0 & \textit{with probability } 1-p \end{cases} \tag{3.3}$$

여기서 엔트로피를 계산하면 다음과 같다.

$$H(X) = -p \log p - (1-p) \log(1-p) \triangleq H(p) \tag{3.4}$$

그림 3.1는 p에 따른 $H(X)$를 나타낸 것이다. 이 그림을 보면 엔트로피는 $p = \frac{1}{2}$ 일 때 최대가 됨을 알 수 있다. 또 p가 0 혹은 1일 때는 엔트로피가 0임을 볼 수

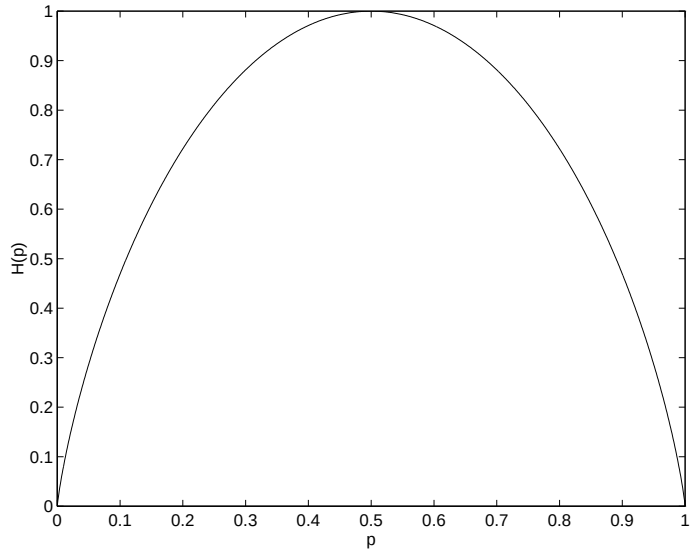


그림 3.1: p에 따른 $H(p)$

있다. 만약 p가 0이나 1이라면 확률 변수 X는 더 이상 변수가 아니므로 불확정성이 0가 되고 $p = \frac{1}{2}$라면 X가 0이나 1일 확률이 같으므로 어떤 것이 나올지를 예상하기가 가장 힘들어진다는 점에서 3.1의 엔트로피의 정의는 상식과 맞는다.

• 조인트 엔트로피(Joint Entropy)와 조건부 엔트로피(Conditional Entropy)

위에서는 하나의 확률 변수에 대한 엔트로피를 정의했다. 여기서는 엔트로피 개념을 여러개의 확률 변수의 쌍에 대해서까지 확장해 보자. 확률 변수 X, Y가 있고 이들의 조인트 확률분포함수가 $p(x, y)$라 할 때 X, Y의 조인트 엔트로피

$H(X,Y)$는 다음과 같이 정의된다.

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\ \log p(x,y). \tag{3.5}$$

이것을 다시 표현하면 단일 변수일 때의 엔트로피처럼

$$H(X,Y) = -E\ \log p(X,Y) \tag{3.6}$$

가 된다. 조인트 엔트로피는 확률 변수 X,Y가 가지는 전체 불확실성을 의미한다.

조건부 엔트로피는 어떤 조건이 주어졌을 때의 엔트로피의 기대값으로 정의되며 다음과 같다.

$$\begin{aligned} H(Y|X) &= \sum_{x\in\mathcal{X}} p(x)\, H(Y|X=x) && (3.7)\\ &= -\sum_{x\in\mathcal{X}} p(x) \sum_{y\in\mathcal{Y}} p(y|x)\ \log p(y|x) && (3.8)\\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\ \log p(y|x) && (3.9)\\ &= -E_{p(x,y)} \log p(Y|X). && (3.10) \end{aligned}$$

조건부 엔트로피와 조인트 엔트로피 사이에는 다음과 같은 관계가 성립한다[13].

$$\begin{aligned} H(X,Y) &= H(X) + H(Y|X) \\ &= H(Y) + H(X|Y) && (3.11) \end{aligned}$$

이것을 chain-rule이라 하며 식이 의미하는 바는 X와 Y 전체가 가지는 엔트로피의 총 량은 X가 가지는 엔트로피와 X가 주어졌을 때 Y에 남아 있는 엔트로피의 합이 된다는 것이다.

다음은 엔트로피가 가지는 몇 가지 중요한 성질이다.

- 엔트로피의 성질

1. $H(X) > 0$
2. $H(X|Y) \leq H(X)$

   등호는 X와 Y가 독립일 경우 성립한다.
3. $H(X_1, X_2, \ldots, X_n) \leq \sum_{i=1}^{n} H(X_i)$

   등호는 X_i들이 모두 독립일 경우 성립한다.
4. $H(X) \leq \log|\mathcal{X}|$

   등호는 X가 균등하게(uniformly) 분포할 때만 성립한다.

   (여기서 $|\cdot|$는 $\cdot$의 cardinality를 나타낸다.)
5. $H(p)$는 p에 대해 볼록함수(concative)이다.

• 상대 엔트로피(Relative Entropy)와 공유정보(Mutual Information)

상대 엔트로피는 두 분포의 차이를 나타내는 척도이다. 다시 말해서 정확한 확률 분포 p와 그것의 추정인 q의 상대 엔트로피 $D(p \parallel q)$ 는 p를 q라고 가정함으로써 생기는 오차를 표시하는 척도가 된다. 두 분포 $p(x), q(x)$ 사이의 상대 엔트로피는 다음과 같이 정의된다.

$$\begin{aligned} D(p \parallel q) &= \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \\ &= E_p \log \frac{p(X)}{q(X)} \end{aligned} \tag{3.12}$$

이 식을 보면 알 수 있듯 상대 엔트로피는 p와 q가 차이가 많이 나면 날수록 커지며 p와 q가 같을 때 0이 된다.

확률 변수들 간의 공유정보는 이름 그대로 두 확률 변수가 얼마나 많은 양의 정보를 공유하고 있는가하는 것을 재는 척도로서 다른 변수가 무엇인지를 알았을 때 확률 변수가 가지는 불확정성의 줄어듦을 나타내는 것이라고도 해석할 수 있다. 확률 변수들 간의 공유정보는 다음과 같이 정의된다.

$$\begin{aligned} I(X;Y) &= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\,\log\frac{p(x,y)}{p(x)p(y)} && (3.13)\\ &= D(\,p(x,y)\parallel p(x)p(y)\,) && (3.14)\\ &= E_{p(x,y)}\log\frac{p(X,Y)}{p(X)p(Y)}. && (3.15) \end{aligned}$$

위 식에서 보듯이 공유정보는 확률 변수 X와 Y가 서로 독립이라고 가정했을 때와 실제와의 상대 엔트로피라고 해석할 수도 있다. 공유정보와 엔트로피는 다음과 같은 중요한 관계를 갖고 이를 도식화한 것이 그림 3.2이다 [13].

$$I(X;Y) = H(X) - H(X|Y) \tag{3.16}$$

$$I(X;Y) = H(Y) - H(Y|X) \tag{3.17}$$

$$I(X:Y) = H(X) + H(Y) - H(X,Y) \tag{3.18}$$

$$I(X;Y) = I(Y;X) \tag{3.19}$$

$$I(X;X) = H(X) \tag{3.20}$$

이상으로 공유정보에 대한 설명을 끝마치겠다. 다음으로 주어진 샘플들의 히스토그램으로부터 공유정보를 추정하는 것이 실제의 정확한 공유정보와 얼마나 차이가 있는지 살펴보기로 한다.

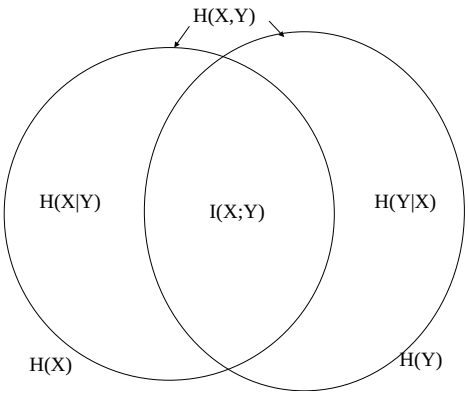


그림 3.2: 엔트로피와 공유정보와의 관계

3.1.2 공유 정보의 추정 (Estimation of Mutual Information)

공유 정보를 계산하기 위해서는 공유정보를 구하고자 하는 각각의 확률 변수들의 확률 분포를 알아야 한다. 그러나 입력 특징 선택 과정에 공유 정보를 사용하고자 할 경우 각각의 확률 분포가 주어지지 않으므로 주어진 입력 벡터와 출력 벡터 쌍의 샘플들로부터 공유 정보를 추정할 수밖에 없다. 따라서 우리는 먼저 이렇게 샘플들로부터 얻어진 공유 정보가 실제 공유 정보와 어느 정도의 오차를 나타내는지 알아보아야 한다.

먼저 N개의 입출력 쌍이 있으며 f는 입력 특징을 c는 출력값을 나타낸다고 하자. 또 그것들의 히스토그램으로부터 추정된 확률 분포 함수가 각각 $P(c), P(f), P(c,f)$라 하자 [1]. 마지막으로 K_f, K_c는 각각 입력과 출력의 분할 (partition) 의 수라고 가정하자 [2]. 이렇게 하면 실제의 정확한 입출력간의 공유

[1] 입력공간을 몇 개의 분할로 나누고 N개의 데이터 중에 나뉘어진 분할에 해당하는 것을 헤아림으로써 $P(c), P(f), P(c,f)$를 구할 수 있다 ($P_c = n_c/N, P_f = n_f/N, P_{cf} = n_{cf}/N$).

[2] 따라서, K_c는 분류하는 클래스의 수가 된다.

정보 $\overline{I}(c,f)$ 와 추정된 공유정보 $I(c,f)$의 오차는 다음과 같이 주어진다 [7].

$$\Delta I = I - \overline{I} \approx \frac{1}{2N}\left(\sum_{c,f}\frac{(\delta n_{cf})^2}{n_{cf}} - \sum_{c}\frac{(\delta n_c)^2}{n_c} - \sum_{f}\frac{(\delta n_f)^2}{n_f}\right) \tag{3.21}$$

여기서 δn은 n의 평균값에서의 변동이다($\delta n = n - \overline{n}$). ΔI는 c와 f에 대해서 $\overline{n}_{cf}/\overline{n}_c\overline{n}_f$ 가 그리 크게 변하지 않는다면 상대 변동값(δn)의 2차 근사화를 사용함으로써 오차를 작게 할 수 있다. 일반적으로 히스토그램에서의 상대변동값(δn)은 평균값의 제곱근에 비례하므로 다음과 같은 근사를 얻을 수 있다.

$$\Delta I \approx \frac{1}{2N}(K_cK_f - K_c - K_f) \tag{3.22}$$

따라서 일반적인 경우에 K_c와 K_f는 모두 2보다 크므로 추정된 공유정보 I는 원래의 공유정보 $\overline{I}$보다 크게 된다 ($K_cK_f - K_c - K_f > 0$). 또 이 오차는 f를 얼마나 많은 분할로 나누는가에 따라서만 달라진다(일반적으로 K_c는 클래스의 수이므로 고정된 값이다). 그러므로 입력특징 f를 몇 개의 공간으로 나누어서 공유정보를 구할지를 잘 정해야 한다. 만약 어떤 특징의 확률 분포가 복잡한 구조를 갖는 경우 K_f를 작게 잡으면 이러한 복잡한 확률 분포 구조를 상당히 단순화시키게 되고 측정된 공유정보값을 줄이게 된다. 또 반대로 K_f를 크게 잡으면 (3.22)에 의해서 원래의 공유정보와의 오차가 커지게 된다. 실제적으로 신경회로망의 입력 특징의 공유정보를 구할 때는 K_f를 10개 정도 놓고 공간을 같은 크기로 나누는 것이 바람직하다[7].

이상으로서 확률 분포를 미리 알지 못하는 경우에 샘플들로부터 공유정보를 추정할 때 생기는 오차에 대해서 살펴봤다.

본 논문은 이상과 같이 정의된 공유정보를 신경회로망의 입력특징 선택 알고리즘에 이용한다. 즉 어떤 입력 특징이 출력과의 공유정보가 높을수록 그 입력특징은 출력에 대한 정보를 많이 가지고 있다고 생각할 수 있으므로 중요한 특징으로 판단할 수 있다. 반대로 출력과의 공유 정보가 작은 특징들은 출력과 거의 무관하다(independent)고 판단할 수 있으므로 그 특징들을 입력 특징 벡터에서 제외해도 신경회로망의 학습 결과에는 크게 영향이 없을 것이라 생각할 수 있다. 또 어떤 두 입력 특징간에 공유정보가 크다는 것은 그것들이 가지고 있는 정보중에 상당량이 중복된다(redundant)는 것을 의미하고 최적의 입력특징 벡터를 구성하기 위해서는 이러한 중복된 특징들을 제외시켜야 할 것이다. 이상의 공유정보를 이용하여 제 4장에서는 신경회로망에서의 새로운 입력특징 선택 알고리즘을 제시한다.

3.2 다구치 방법 (Taguchi Method)

다구치 방법이란 1950년대 초에 일본의 다구치 겐이찌 박사가 제품 개발 과정에서의 효율성을 향상시키기 위해 고안한 방법으로 1980년대에 들어 미국과 유럽에서 각광을 받았다. 다구치 방법의 가장 대표적인 특징은 직교 행렬(orthogonal array : 이하 OA)을 사용하는 것이다. [10], [11].

3.2.1 직교 행렬(Orthogonal Array)

다구치의 이 방법은 생산과정에서 제품의 성능을 좌우하는 중요한 가변 요소를 실험을 바탕으로 찾아내고 이 요소들에 적절한 값을 부여하는 방법이다. 이 방법에서는 먼저 제품의 성능에 작용하는 몇 가지 요소가 선택되고 이 요소들의

값을 체계적으로 몇 단계로 바꾸어가면서 실험한 결과를 바탕으로 이들 중 제품의 성능을 가장 좋게 하는 요소들의 조합을 찾아낸다.

예를 들어 실험 결과 v가 그 값을 조정할 수 있는 몇 가지 변수 $u_1, u_2, \cdots, u_n$의 함수로 표현될 수 있다고 하자. 그렇다면 이것을 $v = f(u_1, \cdots, u_n)$이라 쓸 수 있다. 이러한 변수 $u_1, \cdots, u_n$을 제어요소(control factor)라고 하고 우리의 목적이 이러한 조정요소들을 잘 조정하여 결과 v가 목표하는 바에 가깝게 하는 것이라 하자. 이것은 각각의 조정요소들을 독립적으로 바꾸어 가면서 실험을 하고 그들 중 최적값을 택하거나 혹은 여러개의 조정요소를 동시에 몇 개씩 조정하여 그 결과를 체계적으로 분석하는 방법이 있을 수 있다. 다구치의 직교 행렬은 이렇게 여러 조정요소를 동시에 조정해서 그 결과를 분석에 용이하게 이용할 수 있게 하는 하나의 체계적인 방법이다.

먼저 조정을 해야하는 여러개의 변수가 있을 때 몇 번의 실험만을 거쳐 이들의 준최적(sub-optimal)해를 찾아내기 위해서 과연 어떻게 실험을 해야 하는지에 대해서 생각해 보자. 우선 가장 기본적인 생각으로 임의로 여러 변수들의 조합을 생성한 후 이 변수들의 조합에 대해 실험을 반복하는 방법(build-test-fix)을 생각할 수 있다. 이 방법은 가장 원시적인 방법으로 실험을 아주 많이 한다 해도 최적의 해를 찾을 가능성이 적다. 두번째로 생각할 수 있는 것으로 모든 경우의 변수들의 조합에 대해서 실험을 반복해서 그것들 중 최적해를 찾는 방법이 있다(full factorial). 이 방법을 쓰면 언젠가는 최적해를 찾을 수 있지만 예를 들어 변수들의 갯수가 N개이고 각각의 변수 i에 대해서 l_i개의 단계가 있다고 할 때 $\Pi_{i=1}^{N} l_i$번의 실험이 필요하므로 N이 커질 경우에는 사용이 거의 불가능하다. 따라서 적은 수의 실험만으로 최적의 해를 찾을 수 있는 방법이 필요하고 이를 만

표 3.1: 직교 행렬(OA)의 예

Run	A	B	C	D	E	F	G
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

족시키는 것이 다구치의 직교 행렬이다. 직교 행렬은 실험이 수행될 순서를 정해준다.

직교 행렬에서 직교라는 것은 여러 변수들의 다양한 조합들 중에서 어떤 특정한 하나의 변수의 중요도가 다른 변수들보다 더 혹은 덜하지 않도록 한다는 것을 의미한다. 또 직교라는 것은 각각의 변수가 결과에 미치는 영향을 다른 변수들이 결과에 미치는 영향에 수학적으로 아무 관계없이 독립적으로 계산할 수 있다는 것을 의미한다. 표 3.1은 보정되어야할 변수의 갯수가 7개이고 그 변수들이 변하는 단계(level)가 각각 2개일 경우에의 직교 행렬의 예이다. 이 행렬의 각 행은 7개의 변수들의 단계를 지정하고 이 행들에서 지정하는 대로 8회의 실험을 통해 준최적해를 찾아낸다. 이 경우에 full factorial 방법을 사용할 경우에 $2^7 = 128$번의 실험을 해야 한다. 첫번째 실험에서 모든 변수는 단계 1로 고정된다. 이후의 실험들은 행렬에서 지정하는대로 변수마다 단계를 정하고 8번의 실험까지 수행한다.

표 3.2: 직교 행렬(OA)의 직교성(orthogonality)

Run	A	B	C	D	E	F	G
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

표 3.2는 이 행렬의 직교성을 표현한다. 각 열에는 단계 1과 단계 2가 공통적으로 4번씩 들어있다. 또 A의 단계 1에 대해서 B의 단계 1과 단계 2가 공통적으로 2개씩 들어 있음을 볼 수 있다. 실제로 모든 변수의 쌍에 대해서 이러한 균형이 적용된다. 따라서 이 행렬을 직교 행렬이라 하는 것이다.

일반적으로 이러한 직교 행렬은 모든 단계의 변수들의 조합에 대해서 모두 구성가능하며 일반적으로 많이 쓰이는 직교 행렬을 $La(b^c)$와 같이 표현한다. 여기서 a는 실험의 횟수를 나타내고 b는 각 변수의 변하는 단계의 수를 c는 변수의 갯수를 각각 나타낸다. 대표적인 직교행렬은 표 3.3 – 3.5에 나타냈다.

이러한 직교 행렬에 의해 수행된 실험 결과의 평균과 분산 등을 분석함으로써 목표하는 준최적해를 찾아낼 수 있다. 다음에는 이러한 분석작업 중 평균분석법 (analysis of means : 이하 ANOM)에 대해 알아본다.

표 3.3: $L4(2^3)$ 직교 행렬

Run	1	2	3
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

표 3.4: $L9(3^4)$ 직교 행렬

Run	1	2	3	4
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

표 3.5: $L12(2^{11})$ 직교 행렬

Run	1	2	3	4	5	6	7	8	9	10	11
1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	2	2	2	2	2	2
3	1	1	2	2	2	1	1	1	2	2	2
4	1	2	1	2	2	1	2	2	1	1	2
5	1	2	2	1	2	2	1	2	1	2	1
6	1	2	2	2	1	2	2	1	2	1	1
7	2	1	2	2	1	1	2	2	1	2	1
8	2	1	2	1	2	2	2	1	1	1	2
9	2	1	1	2	2	2	1	2	2	1	1
10	2	2	2	1	1	1	1	2	2	1	2
11	2	2	1	2	1	2	1	1	1	2	2
12	2	2	1	1	2	1	2	1	2	2	1

3.2.2 평균분석법(Analysis of Means)

일반적으로 직교 행렬에 의해 수행된 실험 결과를 가지고 어떤 변수의 최적값 혹은 변수의 중요도 등을 판단하는 방법으로 평균분석법이 있다. 직교 행렬의 직교성 때문에 변수가 가질 수 있는 단계들은 모두 같은 만큼 실험되고 또 각 단계에 따라 다른 변수들의 영향도 모두 비슷하다. 따라서 어떤 변수의 각 단계에 해당하는 실험 결과값의 평균을 서로 비교함으로써 그 변수의 최적치나 중요도를 판단할 수 있다. 다음의 예를 보자.

예제 3.2 *$y = 3x_1 + 5x_2 - 2x_3$의 함수가 있다고 하자. 변수 x_1, x_2, x_3가 모두 {1,2} 2가지 단계만 가지고 있다고 할 때 이것들의 평균분석을 표 3.6에 나타냈다.*

표 3.6: 평균분석의 예($L4$ 사용)

Run	x_1	x_2	x_3	결과 y
1	1	1	1	$y_1 = 6$
2	1	2	2	$y_2 = 9$
3	2	1	2	$y_3 = 7$
4	2	2	1	$y_4 = 14$

	단계 1	단계 2	기울기
x_1	$(y_1 + y_2)/2 = 7.5$	$(y_3 + y_4)/2 = 10.5$	3
x_2	$(y_1 + y_3)/2 = 6.5$	$(y_2 + y_4)/2 = 11.5$	5
x_3	$(y_1 + y_4)/2 = 10$	$(y_2 + y_3)/2 = 8$	-2

표 3.6을 보면 평균 분석을 통해 x_1, x_2, x_3 각각에 대한 기울기가 모두 정확

하게 구해짐을 알 수 있다. 일반적으로 위의 예제와 같이 결과값이 각 변수들에 대하여 모두 선형일 경우에 직교 행렬로 실험한 결과를 평균분석법으로 분석하면 다른 변수들과의 작용으로 인한 오차가 없다. 하지만 일상적인 실험에서 이러한 선형성은 기대할 수 없고 따라서 일반적으로 다구치의 직교 행렬은 변수들과 결과값과의 관계가 어느정도 선형이라 판단할 수 있을 때, 또 변수들 사이에 상호 작용(interaction)이 별로 크지 않을 때 비교적 정확한 분석이 가능해진다. 이 조건을 좀 더 완화하여 변수들과 결과값과의 선형성에 대하여 말할 수 없을 때나 변수들 사에에 상호 작용이 작지 않을 때에도 만일 다른 변수들이 고정되어 있을 때 어떤 변수의 각 단계에 따른 결과값들의 차이로 계산되는 기울기가 언제나 같은 방향인 단조함수의 형태라면 다구치의 직교 행렬을 이용하여 어느 정도 정확한 분석이 가능해진다는 것이 일반적으로 알려져 있다 [11].

이상으로 다구치의 직교 행렬과 평균분석법을 알아보았다. 다음으로 이 다구치 방법을 신경회로망에 적용한 예를 살펴보고 입력 특징 선택 문제에 이 다구치 방법을 어떻게 이용할 수 있는지에 대해서 알아본다.

3.2.3 신경회로망에서 다구치 방법의 적용

다구치 방법을 신경회로망 분야에 적용한 예는 최근까지 그리 많지 않다. 그 이유는 아마도 다구치 방법이 제품의 생산과정에서의 일련의 실험을 체계적으로 하고자 하는 제품생산 분야를 위해 고안되었기 때문일 것이다.

최근에 다구치 방법을 신경회로망에 적용한 예로는 Peterson 등이 오차의 범위 내에서 신경회로망의 구조를 간단히 하려는 목적으로 사용한 것이 있다[12].

여기에서 Peterson 등은 신경회로망의 입력으로 들어가는 입력 데이터가 자체적으로 가지고 있는 잡음(noise) 성분을 줄이기 위해 다구치의 방법을 이용했다.

본 논문에서는 다구치 방법을 신경회로망의 입력 특징 선택 문제에 적용해 보려고 한다. 입력 특징 선택 문제는 각각의 입력 특징을 입력 벡터에 포함시킬지의 여부를 결정하는 문제이므로 N차원 이산 벡터 공간에서의 탐색문제로 해석할 수 있다. 따라서 각각의 입력특징들을 조정 변수로 보고 그 변수를 포함하는 입력특징 집합을 구성하는지의 여부에 따라 그 변수의 단계를 2단계로 나눈다. 즉 어떤 특징을 포함하면 그 변수의 단계를 1로 놓고 포함하지 않으면 그 변수를 2로 놓는다. 이렇게 하면 입력 특징 선택 문제는 이 벡터 공간에서의 탐색문제가 된다. 각각의 입력 특징을 다구치 직교 행렬의 각 열에 대응시킨 후 직교 행렬의 각 행에 대하여 신경회로망을 학습하여 이 결과를 평균분석하면 각 입력 특징별로 그 입력 특징을 포함하여 실험했을 때의 성능(분류율,오차 등)과 포함하지 않았을 때의 성능의 차, 즉 기울기를 얻을 수 있다. 기본적으로 어떤 특징의 기울기가 크다는 것은 그 특징이 분류에 중요하게 작용했다는 것을 의미하고 기울기가 작다는 것에서 그 특징이 분류에 별로 중요하지 작용하지 않는다는 것을 알 수 있다. 이러한 기본 아이디어를 바탕으로 제 5장에서는 다구치 방법을 이용하는 신경회로망에서의 입력특징 선택 방법을 제시한다.

4

새로운 입력 특징 선택 방법 I : 개선된 MIFS

4.1 MIFS의 문제점

이 장과 다음장에서는 신경회로망에서의 새로운 입력 특징 선택 방법을 제시한다. 먼저 이 장에서는 제 2장에서 설명한 Battiti의 MIFS 방법 [7] 을 좀 더 개선한 알고리즘을 제시하고 여러 실험을 통하여 성능을 분석해 본다.

앞 장에서 살펴본 것과 같이 어떤 두 확률변수 간의 공유 정보는 그것들이 가지고 있는 공통된 정보의 양을 나타낸다. 신경회로망에서 하는 일이 입력과 출력 사이를 사상시키는 일이라고 할 때 어떤 입력특징이 출력에 대한 정보를 많이 가지고 있는가 하는 문제는 입력과 출력 사이의 공유 정보를 계산해 봄으로써 풀릴 수 있다. 따라서 N개의 입력 특징들을 각각 $f_1, f_2, \cdots, f_N$이라 하고 출력 특징을 C라 할 때 만약 입출력간의 정확한 공유 정보가 구해질 수 있다면 신경

회로망의 입력특징 선택 문제는 다음과 같이 해결될 수 있다. 먼저 각각의 입력 특징과 출력간의 공유정보 $I(f_i;C)$를 구한 후 이것들 중 최고값을 갖는 특징을 선택한다. 이 특징을 선택된 입력특징의 집합 S에 포함시킨다. 다음으로 이 특징에 다른 특징들을 하나씩 포함한 $N-1$개의 입력특징 벡터들과 출력간의 공유정보 $I(S,f_i;C)$를 구한 후 이들 중 최고값을 나타내는 특징을 S에 포함시킨다. 다시 나머지 특징들과 선택된 특징들이 출력과 나타내는 공유정보 $I(S,f_i;C)$를 구해 최고값을 S에 포함시킨다. 이러한 방법을 계속하면 신경회로망에서의 입력특징 선택 방법은 해결된다. 이 이상적인 알고리즘은 다음과 같다.

- 이상적인 알고리즘

  1. (초기화)

     $F \longleftarrow$ "초기의 N개의 입력 특징",

     $S \longleftarrow$ "빈 집합 (empty set) " 으로 놓는다.

  2. (각 입력 특징과 출력과의 공유 정보 계산)

     $\forall f \in F$에 대해 $I(C;f)$ 계산

  3. (첫번째 특징 선택)

     모든 특징들 중에서 출력과의 공유정보($I(C;f)$)가 가장 큰 특징을 찾는다.

     $F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

  4. (특징 선택)

     목적하는 수의 입력을 선택하기까지 다음을 반복한다.

     (a) (출력과의 공유 정보 계산)

$\forall f$, $f \in F$에 대하여 $I(S, f; C)$ 계산

(b) (다음번 특징을 선택)

$\forall f$, $f \in F$중 출력과의 공유정보 $I(S, f; C)$가 가장 큰 f를 찾고

$F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

5. 선택된 결과인 S를 입력 특징선택의 결과로 내놓는다.

이 방법을 구현하기 위해서는 여러 입력특징들과 출력간의 조인트 공유정보 $I(f_1, f_2, \cdots, f_N; C)$를 구해야 한다. 입력 특징들의 히스토그램을 얻기 위해 각각의 특징들을 n_i개의 공간으로 나누고 출력 특징이 K_c개의 클래스로 이루어져 있다고 할 때 $I(f_1, f_2, \cdots, f_N; C)$를 구하기 위해서는 전체 공간을 $K_c \times \Pi_{i=1}^{N} n_i$개의 분할로 나누어야 한다. 이것은 입력특징이 5개 정도인 가장 간단한 경우에조차 입력 특징들을 각각 10개의 공간으로 나눌 경우 $K_c \times 10^5$의 메모리를 필요로 한다. 따라서 일반적인 경우에 조인트 공유정보 $I(f_1, f_2, \cdots, f_N; C)$를 구한다는 것은 불가능하다.

이를 극복하기 위해 제안된 알고리즘이 2.3절에서 간략히 소개한 Battiti의 MIFS이고 이는 조인트 공유정보 $I(f_1, f_2, \cdots, f_N; C)$를 구하는 대신 입력특징들 상호간의 공유정보 $I(f_i; f_j)$를 이용한다. 좀 더 자세하게 말하자면, 다음번 선택되는 입력특징을 정할 때 $I(f; C) - \beta \Sigma_{s \in S} I(f; s)$를 최대화하는 f를 찾는다. 이는 출력과의 공유 정보가 크고 이미 선택된 입력특징과의 공유정보가 적은 특징을 찾으려는 것이다. 이것을 그림으로 설명하면 다음과 같다.

그림 4.1에서 f_s는 이미 선택된 입력 특징, f_i는 아직 선택되어지지 않은 입력특징 C는 출력이라고 하자. 그러면 위에서 설명한 이상적인 입력특징 선택 알고리즘은 그림 4.2에 나타낸 것처럼 빗금친 부분 즉 2,3,4 부분을 최대화하는 입

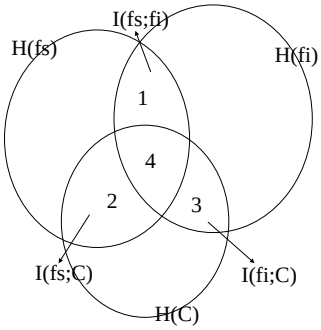


그림 4.1: 신경회로망의 입출력 간의 관계

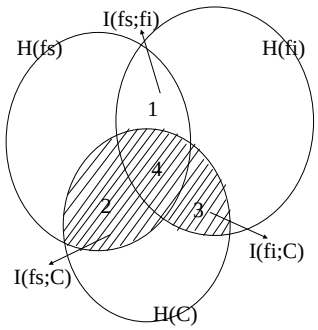


그림 4.2: 이상적인 알고리즘

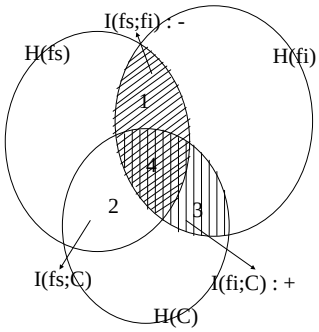


그림 4.3: MIFS 알고리즘

력 f_i를 선택하는 것이다. 여기에서 $I(f_s;C)$ 부분(2와 4)은 선택되지 않은 모든 입력 f_i들과 f_s의 출력과의 조인트 공유정보 $I(f_s,f_i;C)$를 계산하는 데 공통부분이 되므로 이것은 결과적으로 부분 3을 최대화하는 f_i를 찾는 문제가 된다. 반면에 MIFS방법은 $I(f_i;C)-\beta I(F_s;C)$를 최대화하는 것이기 때문에 β를 1로 했을 때 그림 4.3에서 보는 것처럼 이상적인 알고리즘에서 부분 1이 추가로 더 빠지게 된다. 따라서 만약 어떠한 특징이 이미 선택되어진 특징 f_s와 공유하고 있

는 정보가 크다면 부분 1이 늘어날 가능성이 그만큼 더 커지고 f_s가 출력에 대해 가지고 있지 않은 정보 즉, 남아있는 출력과의 공유정보인 부분 3이 크더라도 부분 1의 효과 때문에 입력 특징으로 선택되어지지 않을 가능성이 많아진다. 이는 다음의 예에서 확인할 수 있다.

예제 4.1 *확률 변수 X, Y가 [0,1]사이에 균등하게 분포되어 있고, 입력 특징은 $X, X-Y, Y^2$ 3개이며 출력 Z는 다음과 같이 주어진다고 하자.*

$$Z = \begin{cases} 0 & if \;\; X + 0.2Y < 0.6 \\ 1 & if \;\; X + 0.2Y \geq 0.6 \end{cases}$$

이러한 샘플 데이터를 만들고 공유정보를 구하기 위해 각각의 입력특징 공간을 10개로 나누었을 때 각 특징들의 출력에 대한 공유정보와 상호간의 공유정보는 표 4.1와 같고, β를 1로 놓았을 때 선택되는 특징의 순서는 $X, Y^2, X-Y$의 순이다.

표 4.1: MIFS 방법이 잘못된 선택을 하는 예

(a) 출력과의 공유정보($I(f_i; Z)$)

X	$X-Y$	Y^2
0.845882	0.262094	0.01704

(b) 상호간의 공유정보($I(f_i; f_j)$)

	X	$X-Y$	Y^2
X	–	0.616807	0.061037
$X-Y$	0.616807	–	0.562441
Y^2	0.061037	0.562441	–

(c) MIFS 사용시 $I(f_i; Z) - I(f_i; f_s)$)

$X-Y$	$I(X-Y; Z) - I(X-Y; X) = -0.353713$
Y^2	$I(Y^2; Z) - I(Y^2; X) = -0.043997$

예제 4.1에서 $\beta = 1$인 MIFS방법으로 입력 특징 선택을 하면 가장 먼저 출력과의 공유 정보가 가장 큰 X가 선택되고 다음으로 표 4.1 (c) 에서 보듯 분류를 할 때 더 중요하게 작용하는 $X - Y$보다는 Y^2을 선택한다 [1]. 이것은 X와 $X - Y$사이의 공유정보가 커서 그림 4.1의 부분 4가 커짐으로써 발생되는 현상이다. 이러한 예를 볼 때 MIFS 알고리즘은 중복된 특징을 제거하는 역할은 잘 수행하지만 중복된 특징을 제거하는 것이 언제나 좋은 성능을 내는 것은 아니다. 이러한 MIFS의 단점을 보완하기 위해서 다음 절에서는 그림 4.1의 부분 4의 크기에 거의 영향을 받지 않으므로 이상적인 알고리즘과 가까운 새로운 입력특징 선택 알고리즘을 제시한다.

4.2 개선된 알고리즘

MIFS 알고리즘은 제 2.3절에서 설명한 것과 같이 다음에 추가될 입력 특징을 선택할 때 기존의 입력특징과의 공유정보를 이용한다. 이는 비슷한 성질을 가지는 특징이 이미 선택되었을 때 중복된 특징이 다시 선택되어 학습에 초래되는 낭비를 막을 수는 있지만 앞 절에서 설명한 것과 같은 부작용이 나타날 수 있다. 이것은 MIFS가 이상적인 알고리즘에 그림 4.1의 부분 4를 뺀 것을 최대화하려는 것이 원인이다. 따라서 일반적으로 부분 3이 무시할 수 없을 정도로 크더라고 부분 4가 다른 특징들보다 상대적으로 더 크다면 그 특징은 MIFS방법으로는 선택되지 않는다. 이러한 단점은 입력특징 선택 알고리즘에 부분 4가 영향을 못 미치

[1] X와 $X - Y$의 선형조합으로 Y를 찾아낼 수 있다. 즉, X와 $X - Y$만 있으면 정확하게 출력 Z를 분류할 수 있으므로 X가 주어졌을 때 Y^2보다는 $X - Y$가 출력 Z에 대한 남아있는 정보를 더 많이 가지고 있다.

도록 해서 부분 3의 크기로만 입력특징을 선택하도록 한다면 해결이 가능하다.

이상적인 알고리즘에서 최대화하려는 그림 4.2의 빗금친 부분(2+3+4)은 다음과 같이 표현될 수 있다.

$$I(f_s, f_i; C) = I(f_s; C) + I(f_i; C|f_s) \tag{4.1}$$

여기서 $I(X;Y|Z)$는 Z가 주어졌을 때 X와 Y 사이에 남아있는 공유정보 즉 조건부 공유정보이다. 따라서 이 식에서 $I(f_s;C)$는 부분 2와 4를 $I(f_i;C|f_s)$는 부분 3을 각각 나타낸다. 이상적인 입력 특징 선택 과정에서 선택되지 않은 특징에 대해서 $I(f_s;C)$는 언제나 공통적으로 포함되므로 이는 부분 3 즉, $I(f_i;C|f_s)$을 최대화하는 특징을 찾는 것이 된다. 하지만 일반적으로 $I(f_i;C|f_s)$을 계산하려면 입력공간을 수많은 공간으로 분할해야 하므로 실질적으로 이것은 거의 불가능하다 [2]. 따라서 본 논문에서는 비교적 구하기 쉬운 $I(f_s;f_i)$와 $I(f_i;C)$를 이용해 $I(f_i;C|f_s)$를 근사화하고자 한다. 현실적으로 $I(f_s;f_i)$와 $I(f_i;C)$만을 이용해 정확한 $I(f_i;C|f_s)$를 구하는 것은 불가능하지만 $I(f_i;C|f_s)$은 다음과 같이 표현될 수 있다.

$$I(f_i;C|f_s) = I(f_i;C) - \{I(f_s;f_i) - I(f_s;f_i|C)\} \tag{4.2}$$

위 식에서 $I(f_s;f_i)$는 부분 1과 4를 $I(f_s;f_i|C)$는 부분 1을 각각 나타내므로 $I(f_s;f_i) - I(f_s;f_i|C)$는 부분 4를 나타낸다. 이 식에서 $I(f_s;f_i|C)$에 주목해보자. 이것이 의미하는 바는 출력 C가 주어졌을 때 이미 선택된 입력 f_s와 또 다른 입력 f_i간의 공유정보이고 만약 f_s의 정보량 $H(f_s)$와 출력 C를 이미 알고 있다

[2] $I(f_i;C|f_s)$를 구할 때 나누어야 하는 분할의 수는 조인트 엔트로피 $H(f_i, f_s, C)$를 구할 때의 분할의 수와 같다. 이 과정의 어려움은 앞에서 설명되었다.

고 할 때의 f_s에 남아 있는 정보의 양 $H(f_s|C)$의 비율이 f_s와 f_i간의 공유정보 $I(f_s;f_i)$와 C가 주어졌을 때의 f_s와 f_i간의 공유정보 $I(f_s;f_i|C)$의 비율이 같다고 한다면 즉 다음의 식이 성립할 때

$$\frac{H(f_s|C)}{H(f_s)} = \frac{I(f_s;f_i|C)}{I(f_s;f_i)} \tag{4.3}$$

$I(f_s;f_i|C)$는 다음과 같이 쓸 수 있다.

$$I(f_s;f_i|C) = \frac{H(f_s|C)}{H(f_s)} I(f_s;f_i) \tag{4.4}$$

이것과 (4.2)를 함께 이용하면

$$\begin{aligned} I(f_i;C|f_s) &= I(f_i;C) - \{I(f_s;f_i) - I(f_s;f_i|C)\} \\ &= I(f_i;C) - (1 - \frac{H(f_s|C)}{H(f_s)}) I(f_s;f_i) \\ &= I(f_i;C) - \frac{I(f_s;C)}{H(f_s)} I(f_s;f_i) \end{aligned} \tag{4.5}$$

를 얻을 수 있다. (4.3)의 조건은 정보량이 그림 4.1의 각 부분에 골고루 퍼져 있을 때 성립하며 실제 예제 4.1에 대해서 (4.3)의 비율을 구해보면 표 4.2에 보인 것과 같은 결과를 얻었다.

이 결과를 보면 조건이 큰 오차 없이 잘 들어맞는다는 것을 확인할 수 있다. 따라서 일반적인 경우에 (4.5)을 쓸 수 있다. 이상과 같은 아이디어를 바탕으로 본 논문에서 제안하는 새로운 입력특징 알고리즘은 다음과 같다.

- 제안하는 알고리즘 (보완된 MIFS)

  1. (초기화)

     $F \longleftarrow$ "초기의 N개의 입력 특징",

     $S \longleftarrow$ "빈 집합 (empty set) " 으로 놓는다.

표 4.2: 예제 4.1에 대한 (4.3)의 타당성

(a) $H(f_s;C)$와 $H(f_s)$의 비

$H(X)$	3.31814
$H(X\|Z)$	2.47226
$H(X\|Z)/H(X)$	0.745

(b) $I(f_s;f_i\|C)$와 $I(f_s;f_i)$의 비

$I(X-Y;X)$	0.616807	$I(Y^2;X)$	0.061037
$I(X-Y;X\|Z)$	0.437918	$I(Y^2;X\|Z)$	0.049123
$I(X-Y;X\|Z)/I(X-Y;X)$	0.709	$I(Y^2;X)/I(Y^2;X\|Z)$	0.805

2. (각 입력 특징과 출력과의 공유 정보 계산)

   $\forall f \in F$에 대해 $I(C;f)$ 계산

3. (첫번째 특징 선택)

   모든 특징들 중에서 출력과의 공유정보($I(C;f)$)가 가장 큰 특징을 찾는다.

   $F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

4. (특징 선택)

   목적하는 수의 입력을 선택하기까지 다음을 반복한다.

   (a) (출력과의 공유 정보 계산)

   $\forall f$, $f \in F$에 대하여 $I(f;C)$ 계산

   (b) (이미 선택된 특징의 엔트로피 및 공유정보 계산)

   $\forall s$, $s \in S$에 대하여 $H(s), I(s;C)$ 계산

(c) (이미 선택된 특징과의 공유 정보 계산)

$\forall(f,s)$, $f \in F$, $s \in S$에 대하여 $I(f;s)$ 계산

(d) (다음번 특징을 선택)

$\forall f$, $f \in F$ 에 대해

$I(f;C) - \beta \sum_{s \in S} \frac{I(s;C)}{H(s)} I(f;s)$ 을 최대로 하는 f를 찾고

$F \longleftarrow F \backslash \{f\}$, $S \longleftarrow \{f\}$ 으로 놓는다.

5. 선택된 결과인 S를 입력 특징선택의 결과로 내놓는다.

여기서 β는 기존의 MIFS에서와 마찬가지로 알고리즘에 유연성을 주기 위한 것으로 β를 0으로 놓았을 때는 단순히 출력과의 공유정보의 크기순으로 입력을 선택하는 알고리즘이 된다. 또 β값이 커질수록 비슷한 성질을 가진 특징들을 제외시키는 확률이 그만큼 더 큰 알고리즘이 된다. 일반적으로 (4.5)에서 보는 바와 같이 β는 1정도로 놓으면 된다. 하지만 제 3.1절에서 언급한 것처럼 일반적으로 샘플로부터 공유정보를 추정할 때 추정된 공유정보는 원래의 공유정보보다 크게 계산되는 경향이 있으므로 출력과의 공유정보와 입력들 서로간의 공유정보 모두가 원래보다 커진다. 이미 선택된 입력의 수가 많지 않을 때 이러한 공유정보의 계산상의 오차는 서로를 상쇄하는 효과를 가지지만 이미 선택된 입력의 수가 많아지면 즉, 위의 알고리즘에서 $|S|$가 커지게 되면 입력 선택을 위한 목적함수를 계산하는 과정에서 이러한 오차가 쌓이므로 그 효과가 크게 작용할 수 있다. 따라서 이 알고리즘은 이미 선택된 입력의 수가 많아지면 이러한 오차를 줄이기 위해 β값을 1보다 작은 값으로 놓을 필요가 있다. 그러므로 이미 선택된 입력의 수가 많아질 때 β값을 적응적으로 줄여주는 것이 필요할 수도 있다. 하지만 이 오차는 (3.22)처럼 샘플의 수에 반비례하고 그 값도 그리 크지 않

기 때문에 앞으로 설명될 실험에 이 방법은 사용되지 않았다.

새롭게 제시된 알고리즘은 기존의 MIFS의 장점인 여러개의 공통된 (redundant) 특징들이 있을 때 이것들을 잘 제거하는 성질은 가지면서도 MIFS와는 달리 예제 4.1와 같은 공통되지만 그 남아있는 출력과의 공유정보가 큰 특징들도 잘 선택하는 좋은 성질을 갖는다.

다음절에서는 제안된 알고리즘을 여러 문제에 적용해 성능을 검증해 본다.

4.3 실험 결과

• 간단한 문제

예제 4.1의 문제를 MIFS와 제안된 알고리즘으로 여러 β 값에 대해 입력특징 선택 작업을 수행한 결과는 표 4.3에 보였다. 입력 데이터는 1000개의 패턴으로 구성되어 있고 각각의 값들은 [0,1] 사이의 값으로 정규화했다. 또 입력 특징들을 각각 10개씩의 분할로 나누어 엔트로피 및 공유정보를 계산했다..

실험 결과를 보면 Battit가 제안한 것처럼 MIFS에 β값을 $0.5 \sim 1$을 사용할 경우 [7] 특징 선택이 잘 안 되었음을 볼 수 있다. 이것은 이 문제의 특성상 첫번째 특징으로 출력과의 공유정보가 가장 큰 X가 선택된 후 두번째 특징을 선택하는 과정에서 표 4.1(c) 에 나타낸 것처럼 X와의 공유정보가 너무 많이 빠짐으로써 일어난 결과이다. 새로 제안된 알고리즘을 사용할 경우 제안된 대로 $\beta = 1$일 경우나 다른 모든 β값에 대해서 모두 제대로된 입력 특징 선택을 했다.

표 4.3: 예제 4.1에 대한 MIFS와 제안된 알고리즘의 비교 (F1 = X, F2 = $X-Y$, F3 = Y^2)

(a) MIFS를 사용한 경우

β	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
첫번째 선택	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1
두번째 선택	F2	F2	F2	F2	F2	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3
세번째 선택	F3	F3	F3	F3	F3	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2

(b) 새로 제안된 알고리즘을 사용한 경우

β	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
첫번째 선택	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1	F1
두번째 선택	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2	F2
세번째 선택	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3	F3

• IBM 데이터

이 데이터는 Agrawal 등이 제 2.2절에서 설명한 데이터 mining을 위한 알고리즘인 CDP의 성능을 검증하기 위해 의해 처음으로 구성했고 [8], Setiono등도 자신들의 입력특징 선택 알고리즘을 검증하기 위해 사용했다 [6]. 이 데이터의 모든 패턴들은 표 4.4에 있는 9개의 특징들의 조합으로 이루어지며 출력은 표 4.5에 있는 함수들을 통해 계산된다.

본 논문에서는 실험을 위해 각각의 함수에 대해 1000개의 입출력 패턴을 구성했고 9개의 입력들은 모두 [0,1] 사이의 값으로 정규화 했다. 입력 특징들의 엔트로피와 공유정보 등을 구하기 위해 각각의 입력 특징공간을 10개로 분할한 후 실험했다. 편의를 위해 표 4.5의 3개의 함수로 출력을 분류한 데이터 집합을 각각

표 4.4: IBM 데이터의 입력 특징

Attribute	Description	Value
salary	salary	uniformly distributed from 20,000 to 150,000.
commission	commission	if salary $\geq$75,000 $\longrightarrow$ commission = 0
		else uniformly distributed from 10,000 to 75,000.
age	age	uniformly distributed from 20 to 80.
elevel	education level	unoformly distributed from [0,1, ... ,4].
car	make of the car	unoformly distributed from [1,2, ... ,20].
zipcode	zipcode of the town	uniformly chosen from 9 available zipcodes.
hlevel	value of the house	unoformly distributed from $0.5k10000$ to $1.5k1000000$
		where $k \in \{0, \ldots, 9\}$ depends on zipcode.
hyears	years house owned	unoformly distributed from [1,2, ... ,30].
loan	total amount of lown	unoformly distributed from 1 to 500000.

IBM1, IBM2, IBM3 라 하고 표 4.4의 9개 입력 특징들은 각각 $F1, F2, \ldots, F9$ 라 하겠다.

IBM1 데이터

IBM1 데이터는 표 4.4에서 보듯 age와 salary 즉, $F1$과 $F3$가 출력을 결정한다. 따라서 입력 특징 선택 알고리즘이 이들 두 특징들을 첫번째와 두번째의 특징으로 선택한다면 이 데이타에 대해 입력 특징 선택 알고리즘이 잘 적용된다고 할 수 있다.

먼저 입력 특징들과 출력간의 공유정보와 상호간의 공유정보를 보면 그림 4.4의 (a),(b),(c)와 같다. 그림 4.4(a)는 입력 특징들과 출력간의 공유정보를

표 4.5: IBM 데이터의 분류 함수(classification functions)

Function 1

Group A: $((\text{age} < 40) \ \wedge \ (50000 \leq \text{salary} \leq 100000)) \vee$

$((40 \leq \text{age} < 60) \ \wedge (75000 \leq \text{salary} \leq 125000)) \vee$

$((\text{age} \geq 60) \ \wedge (25000 \leq \text{salary} \leq 75000))$.

Group B: Otherwise.

Function 2

Group A: $((\text{age} < 40) \wedge$

$(((\text{elevel} \in [0 \ldots 2] \ ? \ (25000 \leq \text{salary} \leq 75000)) : (50000 \leq \text{salary} \leq 100000))) \vee$

$((40 \leq \text{age} < 60) \wedge$

$(((\text{elevel} \in [1 \ldots 3] \ ? \ (50000 \leq \text{salary} \leq 100000)) : (75000 \leq \text{salary} \leq 125000))) \vee$

$((\text{age} \geq 60) \wedge$

$(((\text{elevel} \in [2 \ldots 4] \ ? \ (50000 \leq \text{salary} \leq 100000)) : (25000 \leq \text{salary} \leq 75000)))$.

Group B: Otherwise.

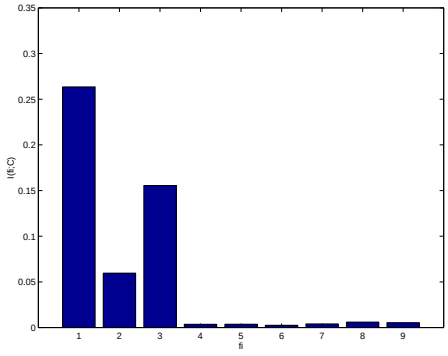
Function 3

Group A: disposable > 0, where

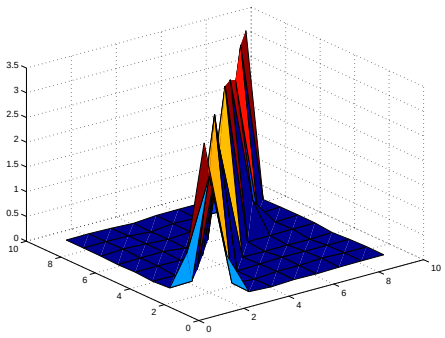
disposable $= (0.67 \times (\text{salary} + \text{commission}) - 5000 \times \text{elevel} - 0.2 \times \text{loan} - 10000)$.

Group B: Otherwise.

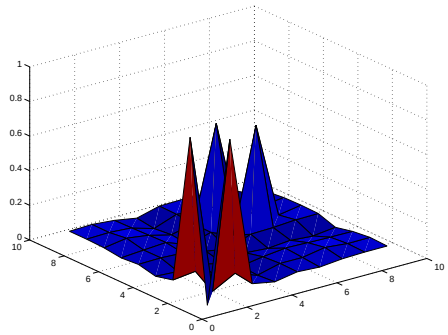
(b)는 입력특징 F_i와 F_j간의 공유정보를 나타내고, i와 j가 같을 때의 공유정보는 특징 F_i의 엔트로피가 되어 상대적으로 값이 커지며 이것은 본 알고리즘의 계산상 의미가 없으므로 (c)는 이를 모두 0으로 놓았을 때의 입력특징 상호간의 공유정보를 나타낸 것이다. 그림 4.4의 (d)는 MIFS 알고리즘에서 $\beta = 0$일 때의 입력 특징이 선택된 순서를 나타내고 이는 출력과의 공유정보만을 이용하므로 결과적으로 그림 4.4(a)를 크기 순으로 배열한 것이 된다. 이 그래프의 높이



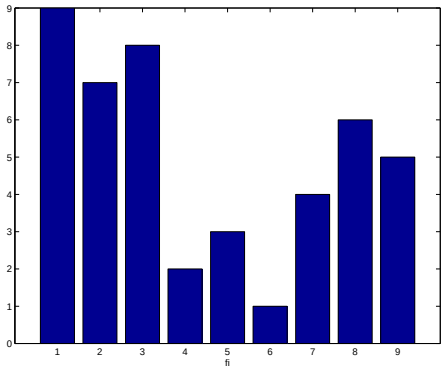
(a) 출력과의 공유정보



(b) 입력 상호간의 공유정보



(c) 입력 상호간의 공유정보(대각성분 = 0)

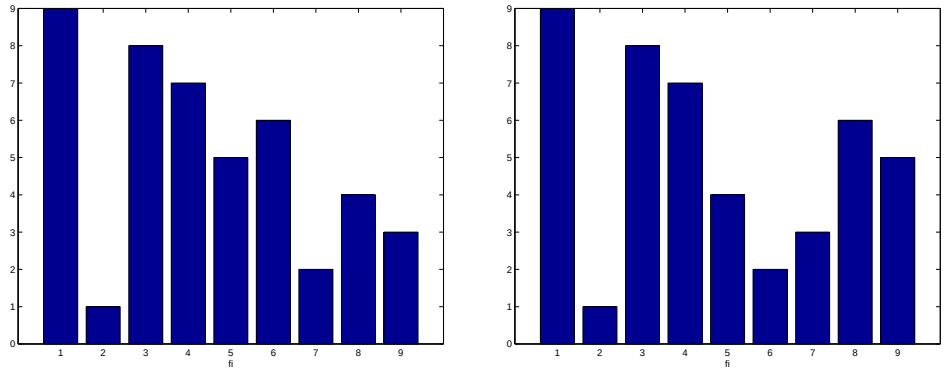


(d) 선택된 순서(출력과의 공유정보만 이용 : $\beta = 0$)

그림 4.4: IBM1 데이터에 대한 공유정보

가 큰 특징일수록 먼저 선택된다.

이 데이터에 대해 MIFS방법과 새로 제안된 방법을 이용해 입력 특징 선택을 수행한 후 얻은 결과는 그림 4.5와 같고 이는 입력 특징이 선택된 순서를 나타낸다. 두 방법 모두 $\beta = 1$로 놓았다. 그림에서 보듯 두 방법 모두 $F1$과 $F3$를 각각 첫번째와 두번째 특징으로 선택하므로 이 문제에 대해서 두 방법이 모두 정확한 특징을 찾아낸다고 할 수 있다.



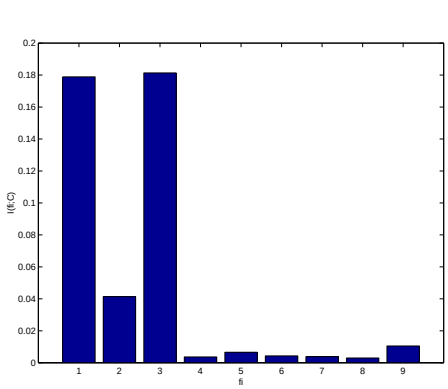
(a) MIFS ($\beta = 1$)　　(b) 제안된 방법 ($\beta = 1$)

그림 4.5: IBM1 데이터에 MIFS와 제안된 방법의 선택 순서

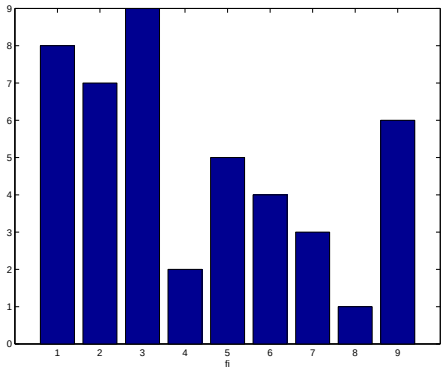
IBM2 데이터

표 4.5를 보면 이 데이터의 출력은 age($F3$), elevel($F4$), salary($F1$)에 의해서 결정된다. 따라서 입력특징 선택 알고리즘이 이 세가지 특징을 차례대로 선택한다면 좋은 결과를 내는 입력 선택 알고리즘이라고 할 수 있다. 이 데이터에 대해 입력특징들의 출력과의 공유정보를 계산한 것이 그림 4.6(a)이다. IBM2 데이터의 입력 특징들의 분포는 표 4.4에 있는 것처럼 모든 IBM 데이터에 대해 공통이므로 입력 특징들끼리의 공유정보는 IBM1에 대해서와 그리 크게 차이나지 않는다. 따라서 입력특징끼리의 공유정보에 대한 그림은 나타내지 않았다 (그림 4.4(b)(c) 참조). 그림 4.6(b)는 출력과의 공유정보만을 가지고 입력특징을 선택한 결과이고 그림 4.7(a), (b)는 각각 MIFS와 제안된 알고리즘으로 입력 특징선택을 수행한 결과로 입력들이 선택된 순서를 보여준다.

MIFS나 제안된 방법의 $\beta = 0$일 때 입력 선택 순서인 그림 4.6(b)를 보면 $F1$과 $F3$는 첫번째와 두번째의 입력으로 선택되었지만 $F4$는 9개의 특징들 중 마지막으로 선택됨을 볼 수 있다. 또 제외되어야 할 특징인 $F2$와 출력과의 공

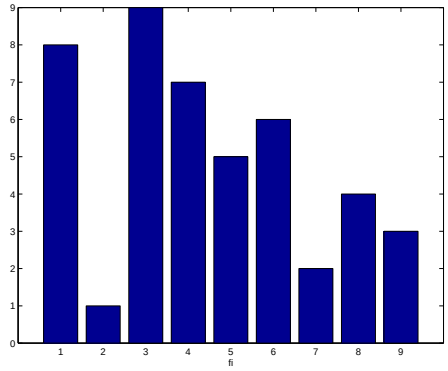


(a) 출력과의 공유정보

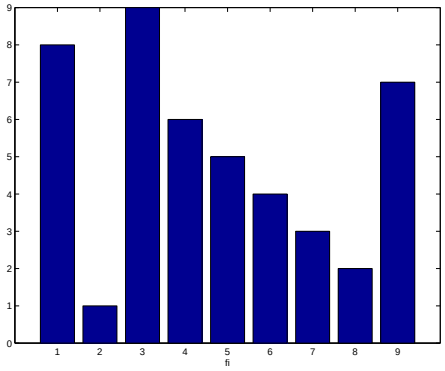


(b) 선택된 순서(출력과의 공유정보만 이용 : $\beta = 0$)

그림 4.6: IBM2 데이터에 대한 공유정보



(a) MIFS ($\beta = 1$)



(b) 제안된 방법 ($\beta = 1$)

그림 4.7: IBM2 데이터에 MIFS와 제안된 방법의 선택 순서

유정보가 다른 것들에 비해 너무 크다는 것도 확인할 수 있다. 이것은 입력들의 출력과의 공유정보만을 가지고는 좋은 입력 특징 선택을 할 수 없다는 것을 단적으로 보여준다. 그러나 MIFS방법이나 제안된 방법에서 $\beta = 1$을 사용할 때 $F4$가 각각 세번째와 네번째의 중요한 특징으로 선택되었다. 이는 특징 선택에서 출력과의 공유정보와 이미 선택된 다른 특징들과의 공유정보를 동시에 이용

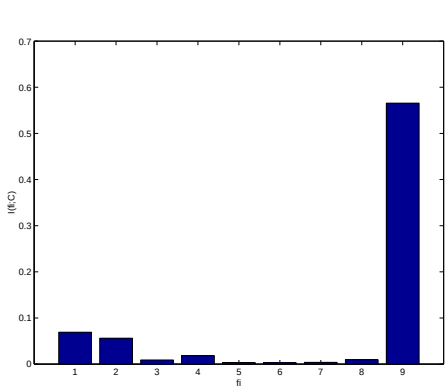
할 때 보다 나은 입력 특징 선택 알고리즘이 된다는 것을 보여준다. 그림 4.7를 보면 특징 $F4$를 기존의 MIFS는 세번째 특징으로 제안된 알고리즘은 네번째 중요한 특징으로 각각 선택함을 볼 수 있다. 따라서 이 결과만으로 볼 때는 기존의 MIFS가 제안된 알고리즘보다 성능이 더 우수하다고 볼 수도 있다. 하지만 그림 4.6(a)를 보면 $F1, F2, F3$의 출력과의 공유정보에 비해서 나머지 것들의 출력과의 공유정보가 너무 작고 그중에서는 $F9$가 공유정보가 가장 크게 계산되었음을 볼 수 있다. 이는 제 3.1.2절에서 설명한 공유정보 계산과정에서의 오차의 범위내라고 볼 수 있고 따라서 $F9$의 출력과의 공유정보가 그림보다 조금만 작게 계산되었다고 가정할 때 제안된 알고리즘에서 $F4$가 먼저 선택될 가능성이 그만큼 더 커진다고 할 수 있다 [3]. 실제로 제안된 알고리즘도 β가 1.2 이상이 되면 $F4$를 세번째로 중요한 특징으로 선택함을 실험을 통해 확인할 수 있었다.

IBM3 데이터

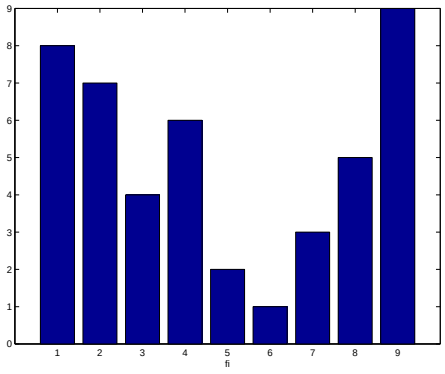
이 데이터의 출력은 표 4.5에서 보는 바와 같이 입력 salary($F1$), commission($F2$), elevel($F4$), loan($F9$)에 의해 결정된다. 따라서 좋은 입력 특징 선택 알고리즘은 이들을 중요한 특징으로 선택해야 한다. 그림 4.8는 입력들과 출력간의 공유정보와 이들만을 이용해 특징을 선택한 순서이고, 그림 4.9(a),(b)는 각각 MIFS와 새롭게 제시된 알고리즘을 썼을 때의 선택된 입력의 순서이다.

그림 4.9(a) 를 보면 기존의 MIFS 방법은 출력을 결정하는 네가지 입력특징 $F1, F2, F3, F9$ 중 세가지 특징은 중요한 특징으로 선택하나 $F2$는 9가지 특징

[3] 실제로 식 3.22 를 구해보면 $K_c = 2, K_f = 10, N = 1000$ 이므로 $\Delta I \approx \frac{1}{2000}(20 - 2 - 10) = 0.004$ 이다. $F9$의 출력과의 공유정보가 0.0105 로 계산되었고 이는 추정되는 오차와 비슷한 order이다.

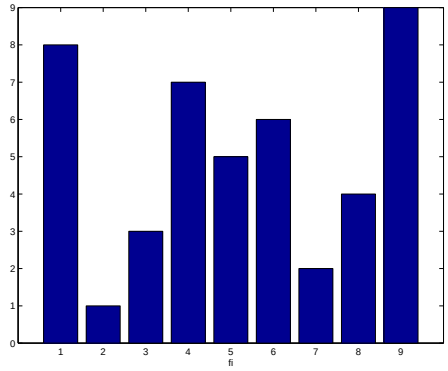


(a) 출력과의 공유정보

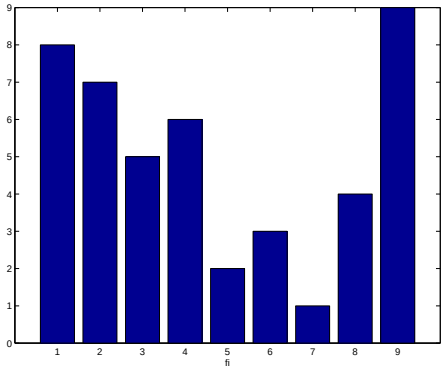


(b) 선택된 순서(출력과의 공유정보만 이용 : $\beta = 0$)

그림 4.8: IBM3 데이터에 대한 공유정보



(a) MIFS ($\beta = 1$)



(b) 제안된 방법 ($\beta = 1$)

그림 4.9: IBM3 데이터에 MIFS와 제안된 방법의 선택 순서

중 가장 중요하지 않은 특징이라고 판단한다. 이는 그림 4.4 (c)에서 보듯 $F2$가 $F1$과의 공유정보가 너무 커서 $F1$이 이미 선택된 후에는 $F2$가 중요한 특징으로 선택되지 않기 때문이다. 이것은 예제 4.1에서와 비슷한 현상이다. 반면에 새롭게 제안된 알고리즘을 사용할 경우에는 그림 4.9 (b)에서 보듯 중요한 네개의 특징인 $F1, F2, F4, F9$을 순서대로 잘 선택함을 볼 수 있다. 이 문제에 대해 MIFS

방법의 성능이 나쁜 것은 앞의 IBM2 문제에 대해 제안된 알고리즘의 성능이 좀 떨어지는 것과는 달리 단순히 계산상의 오차라고 하기에는 무리가 있다 [4]. 실제로 β값을 변화시키면서 선택한 입력특징의 순서를 표 4.6에 MIFS와 제안된 알고리즘에 대해 보였다.

표에서 MIFS방법은 β값이 0.1만 되어도 $F2$를 전체중에 두번째로 중요하지 않은 특징으로 판단함을 볼 수 있다. 반면에 제안된 알고리즘은 실험한 모든 β값에 대해 모두 정확하게 4개의 특징을 선택하고 있다.

• Sonar 문제

이 문제는 수중 음파 탐지기가 잠수함과 같은 구형체의 금속에 반사되어 오는 신호와 잠수함과 비슷한 바닷속 바위에 반사되어 나오는 신호를 분류하는 문제로 Battiti가 이 문제를 PCA와 MIFS를 비교하는데 사용하였다 [7]. 본 논문에서는 MIFS와 제안된 방법의 성능을 비교하는데 이 예를 사용하겠다.

이 데이타의 입력 특징은 60개이고 [0,1]사이의 값으로 정규화하여 사용했고 출력은 mine/rock 중 하나로 각각 하나씩의 노드를 가져 (0,1) 혹은 (1,0)의 값을 가지도록 했다. 전체 데이터는 208개의 샘플로 이루어져 있다. 입력특징의 공유정보와 엔트로피 등을 구하기 위한 분할은 각 입력 특징별로 10개로 나누었다. 이 문제는 앞의 IBM 예제와는 달리 정확히 어떤 특징이 중요한지를 미리 알지 못하기 때문에 60개의 특징중 상위 5% –20% 정도인 3∼12개의 특징을 기존의 MIFS와 제안된 방법으로 선택한 후 이것들을 신경회로망으로 학습한 결과로 성능을 비교했다. 신경회로망의 구조는 모두 Battiti가 한 것처럼 은닉층이

[4] 그림 4.8 (a)에서 $F2$의 출력에 대한 공유정보는 0.0561로 공유정보 계산상의 오차라고 할 수 있는 0.004의 10배가 넘기 때문이다.

표 4.6: β값에 따른 IBM3 입력 특징 선택 순서

(a) MIFS 사용시

β	선택된 순서								
0.0	9	1	2	4	8	3	7	5	6
0.1	9	1	4	8	3	6	5	2	7
0.2	9	1	4	6	8	5	3	7	2
0.5	9	1	4	6	5	8	3	7	2
0.7	9	1	4	6	5	8	3	7	2
1.0	9	1	4	6	5	8	3	7	2
1.2	9	1	4	6	5	8	3	7	2
1.5	9	1	4	6	5	8	3	7	2

(b) 제안된 알고리즘 사용시

β	선택된 순서								
0.0	9	1	2	4	8	3	7	5	6
0.1	9	1	2	4	8	3	7	5	6
0.2	9	1	2	4	8	3	7	5	6
0.5	9	1	2	4	8	3	6	5	7
0.7	9	1	2	4	8	3	6	5	7
1.0	9	1	2	4	3	8	6	5	7
1.2	9	1	2	4	3	8	6	5	7
1.5	9	1	2	4	3	6	8	5	7

표 4.7: Sonar 데이터에 대한 입력 특징 선택 알고리즘의 분류율 (%)

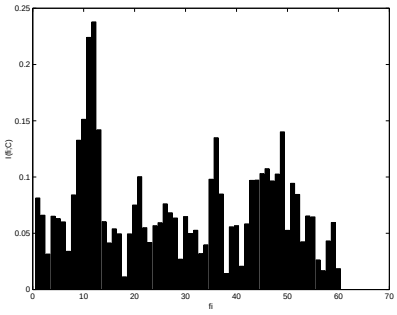
	MIFS	제안된 방법
상위 3개 선택	60.10	71.15
상위 4개 선택	63.46	73.56
상위 6개 선택	67.31	81.73
상위 9개 선택	71.15	87.02
상위 12개 선택	91.83	94.23
전체 (60개)	100	

하나이고 은닉층의 노드수는 3개, 출력 노드는 2개인 ($N-3-2$) 구조를 사용했고 학습에는 일반적인 BP 알고리즘을 이용했다. BP 알고리즘에서 모멘텀은 0.9로 학습률 (learning rate)은 0.01로 놓았고 30,000회 학습했다.

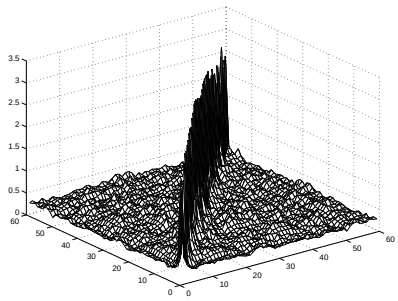
그림 4.10(a),(b),(c),(d)는 각각 입력특징들과 출력과의 공유정보, 입력 특징들간의 공유정보, 대각성분을 없앴을 때의 입력 특징들간의 공유정보, 입력과 출력간 공유정보의 크기 순서를 나타낸다. 그림 4.10의 (e)는 출력과의 공유정보가 가장 큰 12번째 특징과 다른 특징들간의 공유정보를 보여준다. 그림 4.11는 β를 1로 놓았을 때 MIFS와 제안된 방법을 이용해 입력 특징을 선택한 순서를 나타낸다.

표 4.7은 MIFS와 제안된 알고리즘으로 선택된 상위 3, 4, 6, 9, 12개를 위에서 설명한 신경망으로 학습한 결과이고 그림 4.12 와 4.13는 각각 상위 4개와 6개를 선택해서 학습한 학습 곡선이다. 모든 경우에 결과는 3번씩 실험한 결과의 평균이다.

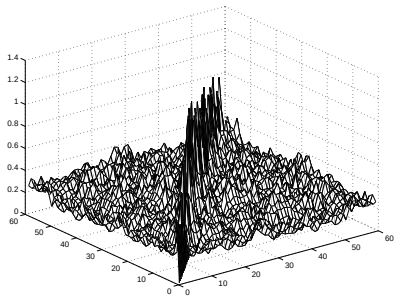
결과를 보면 Sonar 문제에 대해서 기존의 MIFS보다 제안된 입력 특징 선택



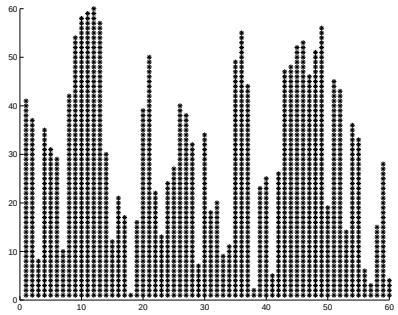
(a) 출력과의 공유정보



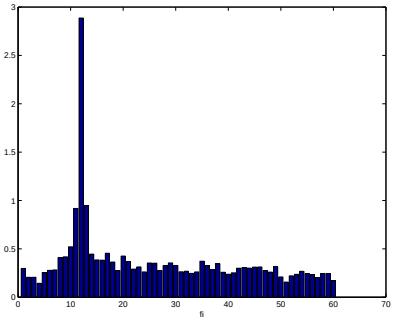
(b) 입력 상호간의 공유정보



(c) 입력 상호간의 공유정보(대각성분 = 0)

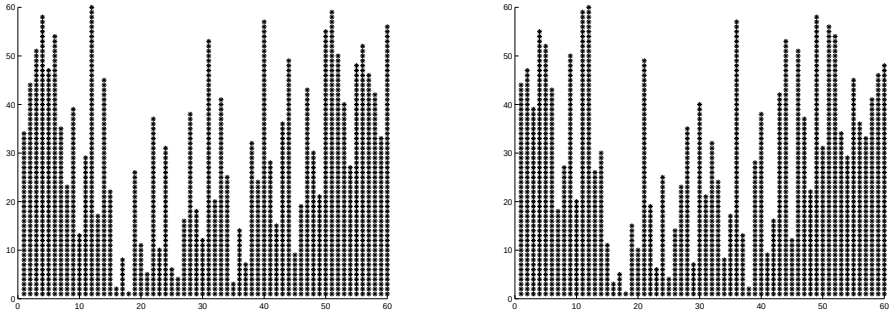


(d) 선택된 순서(출력과의 공유정보만 이용 : $\beta = 0$)



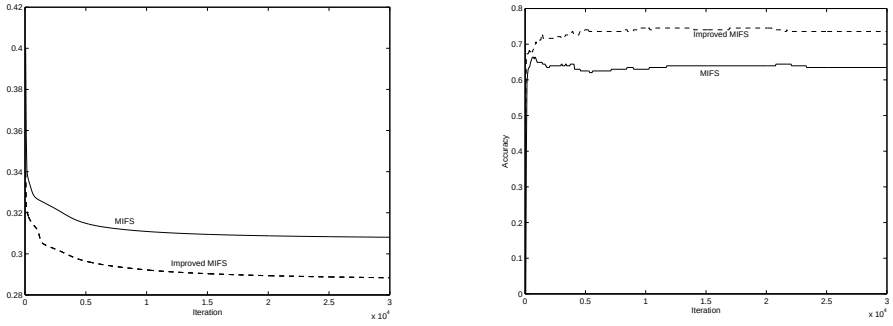
(e) 입력 12와 다른 입력들 간의 공유정보

그림 4.10: Sonar 데이터에 대한 공유정보



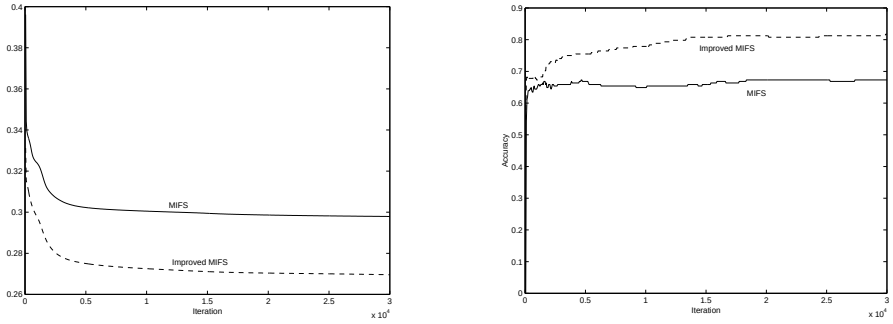
(a) MIFS ($\beta = 1$) (b) 제안된 알고리즘 ($\beta = 1$)

그림 4.11: Sonar 데이터에 대한 입력 특징 선택 순서



(a) 학습 곡선 (b) 분류율

그림 4.12: 상위 4개의 특징으로 학습한 학습 결과



(a) 학습 곡선 (b) 분류율

그림 4.13: 상위 6개의 특징으로 학습한 학습 결과

방법을 쓸 경우 3~9 까지의 입력을 선택할 경우 모두 약 10%이상의 분류율의 향상을 가져왔고 12개의 입력을 선택할 경우에도 제안된 방법의 분류 성능이 더 뛰어남을 볼 수 있다. 제안된 방법으로 12개까지의 입력을 선택해 신경회로망을 학습한 결과를 보면 약 95%정도의 분류율을 얻는다. 이는 일반적으로 입력이 많은 문제에서 본 논문에서 제안한 알고리즘이 입력을 선택하는 데 유용하게 쓰일 수 있음을 입증한다.

• 기타 문제

아래에 사용한 데이터들은 모두 UCI machine learing database에서 가져왔고 인터넷을 통해 구할 수 있다 [5].

Diabetes 문제

이 데이터는 모두 768개의 패턴으로 구성되어 있으며 여러 특징들로부터 당뇨에 걸렸는지의 여부를 판단하는 문제이다. 입력 특징은 나이, 임신횟수, 혈당량 등으로 모두 8개이며 출력은 당뇨 환자 여부의 두가지 종류로 각각 500개 268개씩의 데이터로 구성되어 있다.

이 데이터에 대해서 기존의 MIFS방법과 제안된 방법을 이용해 상위 3~7 개씩 입력 특징을 선택한 후 학습한 결과는 표 4.8이다. 학습에 사용한 은닉층의 노드수는 5개이고 학습률은 0.01, 모멘텀은 0.9로 놓았다. 학습은 15,000회까지 반복하였다. 표를 보면 이 문제에 대해서 MIFS 알고리즘이나 제안된 알고리즘 모두 비슷한 성능을 내는 것을 볼 수 있다. 이렇게 두 알고리즘의 우열을 판단하

[5] http://www.cs.toronto.edu/~delve/ 참조

표 4.8: Diabetes 데이터에 대한 입력 특징 선택 알고리즘의 분류율 (%)

	MIFS	제안된 방법
상위 3개 선택	66.67	67.06
상위 5개 선택	70.07	71.48
상위 7개 선택	72.53	72.53
전체 (8개)	73.87	

기 어려워지는 이유는 전체 입력이 8개로 비교적 그 수가 적고 전체를 이용했을 때의 분류율도 75% 정도로 비교적 낮기 때문일 것이다.

Glass 문제

이 데이터는 214개의 패턴으로 구성되어 있으며 유리의 종류를 구분하는 문제이다. 입력특징의 총 수는 9개로 각각 Fe,Ca 등의 무기물의 함유량을 나타내고 출력은 6개의 유리 종류 중 하나로 되어 있다.

이 데이터에 대해서 기존의 MIFS방법과 제안된 방법을 이용해 상위 4~8 개의 특징을 선택했고 이들로 학습한 결과는 표 4.9이다. 학습에 사용한 은닉층의 노드수는 10개이고 학습률은 0.01, 모멘텀은 0.9로 놓았다. 학습은 15,000회까지 반복하였다.

표를 보면 4개의 입력을 선택했을 때 분류율이 기존의 방법은 46.72%인데 제안된 알고리즘은 65.42%로 약 20%의 분류율의 차이가 난다. 이는 기존의 알고리즘으로 4개의 입력을 선택했을 때는 분류에 아주 중요하게 작용하는 입력특징이 선택되지 않았으나 제안된 알고리즘으로는 이 입력특징을 선택했다고 판단할 수 있다. 6개와 8개를 선택했을 경우에는 두 알고리즘 사이에 분류율의 차

표 4.9: Glass 데이터에 대한 입력 특징 선택 알고리즘의 분류율 (%)

	MIFS	제안된 방법
상위 4개 선택	46.72	64.02
상위 6개 선택	65.42	63.62
상위 8개 선택	77.01	78.97
전체 (9개)	82.18	

이가 거의 없다. 이는 총 입력의 수가 9개로 6개와 8개를 선택했을 때는 어느정도 중요한 특징은 모두 포함되었기 때문일 것이다.

Card 문제

이 데이터는 모두 690개의 패턴으로 구성되어 있으며 신용카드의 발급여부를 구분하는 문제이다. 입력특징의 총 수는 15개로 이 입력들은 연속적인 것들과 이산적인 것들이 섞여 있다. 출력은 신용카드의 발급 여부로 각각 307, 383개씩의 패턴으로 구성되어 있다.

이 데이터에 대해서 기존의 MIFS방법과 제안된 방법을 이용해 상위 4~10

표 4.10: Card 데이터에 대한 입력 특징 선택 알고리즘의 분류율 (%)

	MIFS	제안된 방법
상위 4개 선택	78.84	84.20
상위 6개 선택	81.74	84.35
상위 8개 선택	83.48	87.97
상위 10개 선택	90.14	90.72
전체 (15개)	99.13	

개의 특징을 선택했고 이들로 학습한 결과는 표 4.10이다. 학습에 사용한 은닉층의 노드수는 10개이고 학습률은 0.01, 모멘텀은 0.9로 놓았다. 학습은 15,000회까지 반복하였다.

5

새로운 입력 특징 선택 방법 II : 다구치 방법의 적용

앞 장에서 제시한 알고리즘을 포함하여 공유정보를 입력특징 선택에 이용하는 알고리즘은 일반적으로 언제나 출력과의 공유정보가 가장 큰 특징을 가장 중요한 특징으로 선택하게 된다. 이러한 방법은 대부분의 경우에 별 문제가 없지만 만약 첫번째 특징을 잘못 선택한다면 성능이 떨어지는 특징 집합을 결과로 낼 가능성이 있다. 따라서 공유정보를 이용하는 것에 더불어 다른 방법을 동시에 사용한다면 이런 문제를 해결할 수 있을 것이다. 본 장에서는 제 3장에서 설명한 다구치 방법을 이용해 입력 특징을 선택하는 알고리즘을 제시한다.

5.1 공유정보를 이용하는 알고리즘들의 문제점

제 4장에서 제시한 성능이 향상된 MIFS방법이나 기존의 MIFS는 모두 공유 정

보를 이용함으로써 시간이 오래 걸리는 학습과정 없이 분류에 중요한 입력을 선택한다는 점에서 신경망을 반복적으로 많이 학습해야 하는 제 2.1절에서 설명한 Setiono류의 알고리즘에 대한 잇점을 갖는다. 하지만 이러한 방법은 언제나 제일 첫번째 선택되는 특징은 출력과의 공유정보가 가장 큰 특징이 된다. 이러한 성질은 앞의 IBM 문제처럼 다른 특징들에 비해 출력과의 공유정보가 월등히 큰 경우에는 별 문제가 없겠지만 가장 큰 출력과의 공유정보가 다른 것들보다 그리 크지 않은 경우에는 특징 선택 결과가 공유정보를 계산하는 과정에서의 오차에 의해서 크게 영향을 받을 수 있다. 이것은 한마디로 첫단추를 잘못 끼운것에 비유될 수 있다. 다음과 같은 간단한 예를 보자.

예제 5.1 *확률 변수 X, Y가 각각 [-1,1]사이에 균일하게 분포되어 있고 입력 특징은 X, Y, X^2Y 3개이며 출력 Z는 다음과 같이 주어진다고 하자.*

$$Z = \begin{cases} 0 & if \;\; XY < 0 \\ 1 & if \;\; XY \geq 0 \end{cases}$$

이것은 그림 5.1과 같이 XY 공간을 2개의 공간으로 나누는 XOR문제로 신경회로망의 성능을 분석하는데 주로 많이 사용되어 왔다. 입력특징이 X, Y 외에 X^2Y가 사용되었을 때 공유정보를 이용하는 각 알고리즘의 입력특징 선택 순서는 표 5.1에서 볼 수 있듯 언제나 X, Y보다는 X^2Y를 먼저 선택한다.

위에서 본 예와 같은 현상은 두 개 이상의 특징들의 조합으로 출력에 영향을 미치는 문제에 공유정보를 이용하는 특징선택 알고리즘을 사용할 경우 일반적으로 나타나는 현상으로 공유정보만을 가지고 특징 선택을 할 경우에 이러한 문제를 해결하기 어렵다. 다음 절에서는 이러한 문제를 해결할 수 있는 다구치 방법을 이용한 입력 특징 선택 알고리즘을 제시한다.

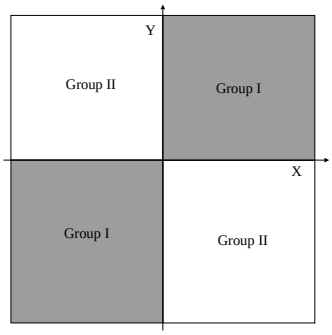


그림 5.1: XOR 문제

표 5.1: XOR 문제에 공유정보를 이용하는 알고리즘 사용시의 특징 선택 순서

입력특징	X	Y	X^2Y
출력과의 공유정보	0.00886869	0.00733066	0.0111475
MIFS($\beta = 1$)	3	2	1
개선된 MIFS($\beta = 1$)	3	2	1

5.2 다구치 입력특징 선택 알고리즘

일반적으로 신경회로망에서의 입력 특징을 선택하려면 공유 정보를 이용하는 방법 외에는 신경회로망을 반복적으로 학습하는 방법을 생각할 수 있다. 하지만 이러한 방법은 제 2.1절에서 설명한 Setiono등의 방법 [6] 처럼 언제나 신경회로망을 학습하는데 걸리는 시간이 걸림돌이 된다. 입력특징 선택을 위한 학습시간을 줄이는 방법은 크게 학습 자체에 걸리는 시간을 줄이는 방법과 학습의 횟수를 방법을 줄이는 방법이 있으며 [6]에서 사용한 준뉴톤(quasi-Newton) 방법의 일종인 BFGS(Broyden-Flecher-Shanno-Goldfrab)방법은 전자에 해당한다.

다구치의 직교 행렬은 원래 실험의 횟수를 줄여보려는 생각에서 출발한 것

으로 신경회로망에서의 입력 특징을 선택하는 문제에서도 이 방법을 쓰는 것을 고려해볼 수 있다. 다구치의 직교 행렬은 제 3.2절에서 설명한 것처럼 여러 단계로 변할 수 있는 조정 변수(control variable)들을 체계적으로 바꾸어 가면서 준최적해를 찾아내는 방법이다. 신경회로망의 입력 선택 문제에 다구치의 직교 행렬을 이용하고자 한다면 먼저 변수들의 단계를 어떻게 정할까 하는 문제에 대해 생각해 봐야 한다. 이 문제에서 변수는 각각의 입력 특징들로 잡고 변수들을 입력벡터에 포함할지 안할지의 여부를 그 변수의 단계로 보자. 이렇게 되면 각각의 변수에 대해 2개씩의 단계가 생기며 이 문제의 최적해는 틀림없이 이런 단계들의 조합들 중 하나가 될 것이다.

직교 행렬의 이용방법을 이렇게 정하고 나서 또 하나 생각해야 할 것은 과연 이 문제에 대해 다구치의 직교 행렬이 얼마나 정확한 해를 줄 것인가 하는 문제이다. 다구치의 직교 행렬이 최적해를 찾으려면 실험 결과가 각 변수들의 선형조합(linear combination) 형태로 나타나야 하나 일상적인 실험에서 이러한 선형성은 기대할 수 없고 일반적으로 다구치의 직교 행렬은 변수들과 결과값과의 관계가 어느정도 선형이라 판단할 수 있을 때 또 변수들 사이에 상호 작용(interaction)이 별로 크지 않을 때 비교적 정확한 분석이 가능해진다. 이 조건을 좀 더 완화하여 변수들과 결과값과의 선형성에 대하여 말할 수 없을 때나 변수들 사에에 상호 작용이 작지 않을 때에도 만일 변수들의 각 단계가 다른 변수들이 고정되어 있을 때 언제나 같은 방향의 기울기를 갖는 단조함수의 형태라면 다구치의 직교 행렬을 이용하여 어느 정도 정확한 분석이 가능해진다는 것이 일반적으로 알려져 있다 [11]. 신경회로망의 학습에서 일반적으로 출력에 중요한 특징이라면 그 특징을 사용했을 때와 사용하지 않았을 때의 학습 성능은 그

특징을 사용했을 때 우수하다. 따라서 신경회로망의 입력 특징 선택 문제는 결과값이 입력들의 사용여부에 관한 단조함수꼴이라고 할 수 있다. 그러므로 직교행렬을 이 문제에 이용한다면 많지 않은 학습 횟수로 어느 정도 좋은 성능을 얻을 수 있을 것이다.

다구치 방법을 이용하는 입력특징 선택 알고리즘을 개략적으로 설명하면 다음과 같다. 먼저 다구치의 직교 행렬을 만들고 직교 행렬의 각 열에 입력 특징들을 하나하나 대응시킨다. 직교 행렬의 각 행을 한 번의 신경회로망의 학습으로 보고 이 때 신경회로망의 입력벡터로 그 행의 열 중에서 '1'(혹은 '2')에 해당하는 입력 특징들을 모아서 학습시킨다. 이렇게 학습된 결과를 저장한 다음 모든 신경회로망의 학습이 끝난 후 특정한 입력 특징을 입력 벡터에 포함시켰을 때와 그렇지 않을때의 평균정확도를 비교해서 그 차이가 큰 순서로 나열하면 입력 특징을 선택할 수 있다. 일반적으로 이 방법을 이용한다면 N개의 입력이 있을 때 $2^{\lceil \log_2 N+1 \rceil}$번의 학습이 필요하므로 N에 비례하는 $O(N)$번 정도의 실험이 필요하다. (여기서 $\lceil \cdot \rceil$는 $\cdot$보다 크거나 같은 최소의 정수이다.) 따라서 위와 같은 방법으로 입력을 선택한다면 비교적 적은 실험으로 좋은 입력 특징을 선택할 수 있을 것이다. 표 5.2은 입력 특징이 3개인 간단한 문제에서 다구치의 직교 행렬을 이용하는 방법을 간략히 보인 것이다.

여기서 $F1$을 이용했을 때의 평균분류율은 (학습 1의 분류율 + 학습 2의 분류율) / 2로 $F1$을 이용하지 않을 때의 평균분류율은 (학습 3의 분류율 + 학습 4의 분류율) / 2로 각각 계산하여 이것들의 차인 평균증가율을 구한다. 나머지 특징들 $F2, F3$에 대해서도 같은 방법으로 평균증가율을 구할 수 있고 평균증가율이 다른것들보다 큰 특징을 중요한 특징이라고 판단할 수 있다.

표 5.2: 입력 특징 선택 문제에 직교행렬을 이용하는 방법($L4$)

(a) 직교 행렬 ($L4$)

Run	1	2	3
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

(b) 입력 특징 선택 (O : 사용, – : 사용 않음)

신경망 학습	$F1$	$F2$	$F3$	분류율(%)
1	O	O	O	90
2	O	–	–	30
3	–	O	–	40
4	–	–	O	60
평균분류율(사용시) (%)	60	65	75	–
평균분류율(불용시) (%)	50	45	35	–
평균 증가율(%)	10	20	40	–
선택 순서	3	2	1	–

표 5.2과 같은 비교적 작은 행렬을 이용할 때는 언제나 첫번째 실험은 모든 특징을 포함하는 것이 되고 나머지들은 총 입력 특징수의 반 이하의 특징을 이용해 신경망을 학습하므로 평균분석법을 이용했을 때 오차가 커질 가능성이 많다. 따라서 이렇게 입력의 수가 별로 많지 않을 때는 표 5.2의 예에서 이용한 직교 행렬의 성분이 '1'인 특징을 포함하는 학습 외에 '2'에 해당하는 특징을 포함하는 입력벡터를 구성해 학습시킴으로써 좀 더 나은 특징 선택을 할 수 있다.

또 특징의 수가 너무 많을 때에는 먼저 공유 정보를 이용하는 알고리즘을 이용해 어느정도 특징수를 줄여 놓은 후 이 방법을 이용한다면 보다 나은 성능을 얻을 수 있을 것이다.

다음은 본 논문에서 제시하는 다구치 방법을 이용하여 입력 특징을 선택하는 알고리즘이다.

- 다구치 입력 선택 알고리즘

  1. (필터링)

     입력 특징의 갯수가 너무 많으면 선택하고자 하는 특징 갯수의 2배수 정도를 앞 장에서 제시한 개선된 MIFS등으로 미리 선택한다.

  2. (직교 행렬 구함)

     필터링된 특징갯수에 맞는 직교 행렬을 찾는다.

  3. $n = 1, 2$에 대해 다음을 반복한다.

     (a) (신경회로망의 입력 특징 벡터 구성) 직교 행렬의 성분 n에 해당하는 입력 특징을 포함하는 특징 벡터를 만든다.

     (b) (신경회로망 학습)

     각각의 행에 대해 신경회로망을 학습하고 분류율을 저장한다.

     (c) (평균 분석법)

     평균분석법을 이용해 각각의 특징에 대한 분류율의 증가분을 계산한다.

  4. (입력 특징 선택)

     각각의 특징에 대해서 $n = 1$과 $n = 2$ 일 때 분류율의 증가분을 서로 더한 후 이의 크기 순서대로 입력 특징을 선택한다.

다음 절에는 위의 알고리즘을 이용해 다양한 데이터에 대해 입력 특징을 선택한 결과를 보였다.

5.3 실험 결과

용어를 간단히 하기 위해 앞으로의 모든 경우에 '1'에 해당하는 입력특징을 포함하는 벡터 만들경우 '방법 I' '2'를 이용할 경우는 '방법 II'라 하겠다. 학습에는 모두 일반적인 BP알고리즘을 이용했고 은닉층은 1개를 썼다. 모든 입력은 [0,1] 사이의 값으로 정규화해 사용했다.

• 간단한 문제

앞의 예제 5.1 에 대해 다구치 입력 특징 선택 알고리즘을 사용한 결과는 표 5.3와 같다. 여기서 은닉층의 노드수는 3개를 사용했고, 학습률은 0.02를 모멘

표 5.3: XOR 문제에 다구치 입력특징 선택 알고리즘 사용시의 특징 선택 순서

	학습	X	Y	X^2Y	분류율(%)
방법 I	1	O	O	O	89.0
	2	O	–	–	0.0
	3	–	O	–	0.0
	4	–	–	O	0.0
	slope	44.5	44.5	44.5	–
방법 II	1	–	–	–	0.0
	2	–	O	O	1.8
	3	O	–	O	82.4
	4	O	O	–	91.7
	slope	86.15	5.55	-3.75	–
선택순서		1	2	3	–

텀은 0.9를 각각 썼고 10,000회 학습했다. 표 5.3에서 다구치 입력특징 순서는 X, Y, X^2Y순으로 이는 방법 I, II에서 얻은 기울기의 합의 크기순으로 선택한 것이다.

이 문제는 예제 5.1에서 보았듯이 공유 정보를 이용하는 알고리즘인 MIFS나 개선된 MIFS알고리즘으로는 정확한 입력 특징을 찾을 수 없지만 다구치 알고리즘을 이용할 경우 문제를 잘 해결함을 볼 수 있다.

• IBM 데이터

이 곳에서는 앞 장에서 다루었던 IBM 데이터에 다구치 방법을 이용해 입력 특징을 선택하고자 한다. IBM 데이터의 전체 입력 특징 수가 9개로 비교적 작으므로 앞절에서 제시된 알고리즘의 첫번째 단계인 필터링은 거치지 않았다. 즉 모든 특징들을 다구치의 직교 행렬의 각 열에 대응시키고 이것을 학습한 결과로 입력특징 선택 알고리즘을 수행했다. 학습에는 일반적인 BP알고리즘이 사용되었고 은닉층의 노드 수는 10개를 이용했다. 학습률은 0.01을 모멘텀은 0.9를 사용했고 10,000회까지 반복학습시켰다. 특징 선택의 신뢰도를 높이기 위해 방법 I과 II를 모두 사용한 뒤 평균하여 입력 특징을 선택했다.

IBM1 데이터

표 5.4 는 IBM1 데이터에 대한 다구치 입력 특징 선택 과정을 보여주며 그림 5.2는 다구치 입력 특징 선택 순서이다. 이를 보면 표 4.5에서 출력에 중요한 영향을 미치는 특징 $F1$,과 $F3$가 각각 첫번째와 두번째의 입력 특징으로 선택되었음을 볼 수 있다. 또 표 5.4에서 기울기를 보면 $F1$과 $F3$의 기울기가 각각 40.9%,

표 5.4: IBM1 데이터에 다구치 입력특징 선택 알고리즘 사용시의 특징 선택 과정 ($L12$ 이용)

Run	F1	F2	F3	F4	F5	F6	F7	F8	F9	–	–	방법 I	방법 II
1	1	1	1	1	1	1	1	1	1	1	1	99.8	0.0
2	1	1	1	1	1	2	2	2	2	2	2	98.3	74.6
3	1	1	2	2	2	1	1	1	2	2	2	69.5	65.2
4	1	2	1	2	2	1	2	2	1	1	2	98.0	65.8
5	1	2	2	1	2	2	1	2	1	2	1	71.2	76.1
6	1	2	2	2	1	2	2	1	2	1	1	70.4	78.8
7	2	1	2	2	1	1	2	2	1	2	1	71.3	99.0
8	2	1	2	1	2	2	2	1	1	1	2	67.4	98.8
9	2	1	1	2	2	2	1	2	2	1	1	75.2	78.0
10	2	2	2	1	1	1	1	2	2	1	2	76.0	98.5
11	2	2	1	2	1	2	1	1	1	2	2	65.0	70.9
12	2	2	1	1	2	1	2	1	2	2	1	63.4	71.8
slope	40.9	13.9	38.1	10.7	11.6	17.9	14.6	8.6	12.7	–	–	–	–
선택순서	1	5	2	8	7	3	4	9	6	–	–	–	–

38.1%로 다른 것들에 비해 월등히 큰것을 볼 수 있다. 이로써 IBM1 데이터에 대한 사전정보가 전혀 없더라도 $F1$과 $F3$가 다른 특징들에 비해서 훨씬 중요하다는 것을 추측할 수 있다.

IBM2 데이터

표 5.5와 그림 5.3는 다구치 입력선택 방법을 IBM2 데이터에 적용했을 때의 특징 선택 결과이다. 이 결과를 보면 중요한 입력 특징인 $F1, F3, F4$ 중에 $F1$과

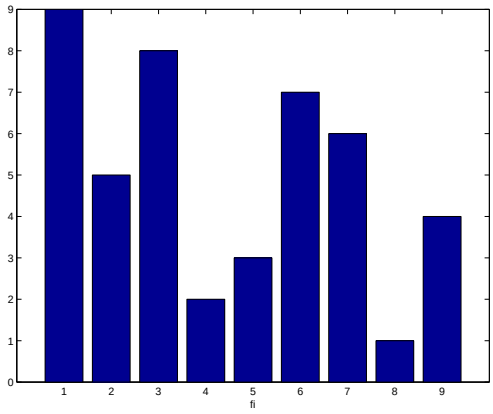


그림 5.2: 다구치 방법의 입력특징 선택 순서 - IBM1

$F3$는 각각 첫번째와 두번째로 선택했지만 $F4$는 중요한 특징으로 선택하지 않는 것을 확인할 수 있다. 이런 현상은 그림 4.6 (b) 출력과의 공유 정보만을 이용했을 때 입력 특징 선택 순서와 유사한 면을 보여준다. 이런 오동작을 막기 위해서는 앞 절에 제시한 알고리즘처럼 일단 필터링을 거친 다음 다구치 방법으로 입력 특징을 선택해야 할 것으로 보인다. 이 문제에 대해서 다구치 방법이 정확한 특징선택은 하지 못하였지만 특징 $F1, F3$의 기울기가 다른 것들보다 훨씬 큼을 볼 때 어느정도의 성능은 낸다고 할 수 있다.

IBM3 데이터

이 문제에 대한 특징 선택 과정은 표 5.6에 보였고 특징 선택 결과는 그림 5.4에 있다. 이 문제에 대해서 다구치 입력 특징 선택 방법은 중요한 특징인 $F1, F2, F4, F9$을 정확히 차례대로 선택함을 볼 수 있다.

표 5.5: IBM2 데이터에 다구치 입력특징 선택 알고리즘 사용시의 특징 선택 과정 ($L12$ 이용)

Run	F1	F2	F3	F4	F5	F6	F7	F8	F9	–	–	방법 I	방법 II
1	1	1	1	1	1	1	1	1	1	1	1	99.7	0.0
2	1	1	1	1	1	2	2	2	2	2	2	98.3	75.4
3	1	1	2	2	2	1	1	1	2	2	2	81.1	60.5
4	1	2	1	2	2	1	2	2	1	1	2	94.0	77.7
5	1	2	2	1	2	2	1	2	1	2	1	70.4	84.7
6	1	2	2	2	1	2	2	1	2	1	1	72.1	92.4
7	2	1	2	2	1	1	2	2	1	2	1	68.8	97.3
8	2	1	2	1	2	2	2	1	1	1	2	70.1	89.7
9	2	1	1	2	2	2	1	2	2	1	1	74.1	92.7
10	2	2	2	1	1	1	1	2	2	1	2	68.4	97.0
11	2	2	1	2	1	2	1	1	1	2	2	79.1	79.4
12	2	2	1	1	2	1	2	1	2	2	1	65.8	73.8
slope	38.1	21.9	33.7	13.8	11.4	20.2	15.9	20.4	14.1	–	–	–	–
선택순서	1	3	2	8	9	5	6	4	7	–	–	–	–

표 5.6: IBM3 데이터에 다구치 입력특징 선택 알고리즘 사용시의 특징 선택 과정 ($L12$ 이용)

Run	F1	F2	F3	F4	F5	F6	F7	F8	F9	–	–	방법 I	방법 II
1	1	1	1	1	1	1	1	1	1	1	1	100	0.0
2	1	1	1	1	1	2	2	2	2	2	2	64.4	86.4
3	1	1	2	2	2	1	1	1	2	2	2	63.8	89.7
4	1	2	1	2	2	1	2	2	1	1	2	96.0	69.1
5	1	2	2	1	2	2	1	2	1	2	1	95.7	58.1
6	1	2	2	2	1	2	2	1	2	1	1	46.2	97.7
7	2	1	2	2	1	1	2	2	1	2	1	87.0	69.1
8	2	1	2	1	2	2	2	1	1	1	2	90.7	62.5
9	2	1	1	2	2	2	1	2	2	1	1	32.9	98.3
10	2	2	2	1	1	1	1	2	2	1	2	46.2	98.3
11	2	2	1	2	1	2	1	1	1	2	2	88.0	59.8
12	2	2	1	1	2	1	2	1	2	2	1	40.5	98.7
slope	27.7	16.9	9.2	17.2	12.8	8.9	13.4	13.0	85.6	–	–	–	–
선택순서	2	4	8	3	7	9	5	6	1	–	–	–	–

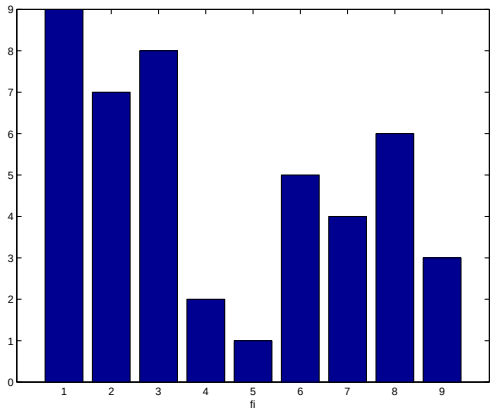


그림 5.3: 다구치 방법의 입력특징 선택 순서 - IBM2

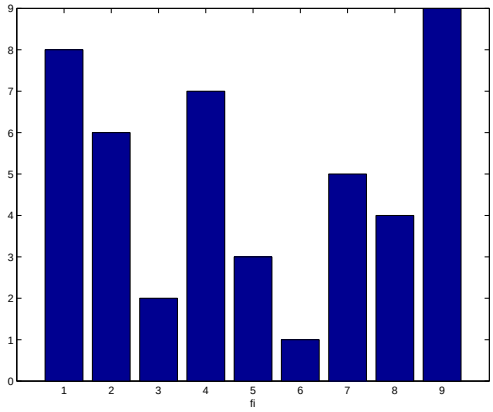


그림 5.4: 다구치 방법의 입력특징 선택 순서 - IBM3

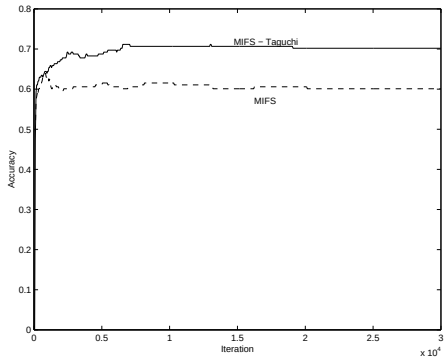
• **Sonar 문제**

여기서는 앞장에서 개선된 MIFS를 MIFS와 비교하기 위해 사용한 Sonar문제에 대해 다구치 방법을 이용해 입력을 선택해 본다. 이 데이터는 입력이 60개나 되므로 필터링 없는 다구치 방법을 그대로 쓰기에는 무리가 따른다. 따라서 이들 60개의 입력들 중 5.1절에서 제시한 알고리즘의 필터링 과정에 나와 있듯 다

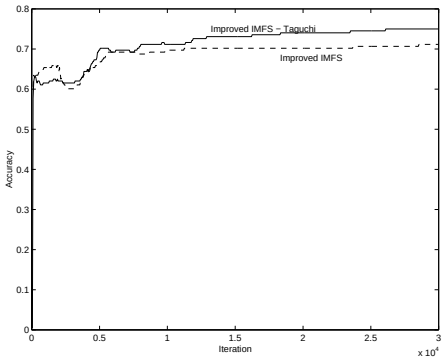
구치 방법으로 선택하고자 하는 특징수의 2배수를 공유정보를 이용하는 방법인 MIFS와 4장에서 제세된 개선된 MIFS방법을 이용해 미리 선택한 후 이것을 다구치의 직교 행렬에 적용하여 결과를 얻었다. 이렇게 선택된 상위 3, 4, 6, 9개를 이용한 분류율을 MIFS를 이용했을 때와 개선된 MIFS를 이용했을 때의 선택결과와 비교하였다. 특징 선택과정에는 표 5.7에 보인 것처럼 방법 I과 방법 II를 모두 사용했다. 3개와 4개의 입력을 선택할 때는 $L8$ 직교 행렬을 이용했고 6개의 입력을 선택할 때는 $L16$ 직교 행렬을 이용했다. 신경회로망의 구조는 제 4장에서처럼 은닉층의 수는 3개를 이용했고 모멘텀은 0.9 학습률은 0.01을 이용했다. 학습은 30,000회를 반복했다.

표 5.8의 각 열은 각각 MIFS 방법, 개선된 MIFS 방법, MIFS 필터링을 통한 다구치 방법, 개선된 MIFS 필터링을 통한 다구치 방법을 입력 특징 문제에 적용했을 때의 분류율을 나타낸다. 이 표를 보면 일반적으로 다구치 방법이 필터링에 사용한 알고리즘보다 분류 성능이 더 우수함을 볼 수 있다. 그림 5.5와 5.6는 각각 상위 3개와 4개 선택시의 각 알고리즘의 분류율 곡선을 보여준다.

표 5.8를 보면 모든 경우에 대해서 다구치 방법을 이용할 때가 필터링에 사용한 알고리즘만을 사용해서 입력 특징을 선택할 때보다 분류율이 좋아졌음을 볼 수 있다. 또 필터링에 기존의 MIFS 방법을 사용한 것과 4장에서 새롭게 제안된 알고리즘을 이용하는 방법을 비교하면 네가지 경우 모두에 대해서 4장에서 제시된 알고리즘을 이용하는 것이 분류율의 약 5%이상의 향상을 가져 왔다. 결과적으로 이 실험은 입력 특징이 다구치의 직교 행렬을 바로 이용하기에는 너무 많은 경우에 4장에서 제시된 알고리즘으로 필터링을 거친 후 다구치 방법을 이용하면 좋은 성능을 얻을 수 있다는 것을 보여준다.

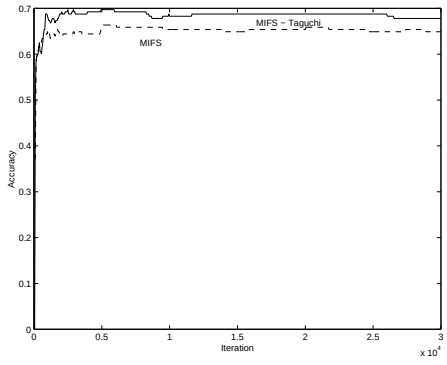


(a) MIFS와 다구치 방법

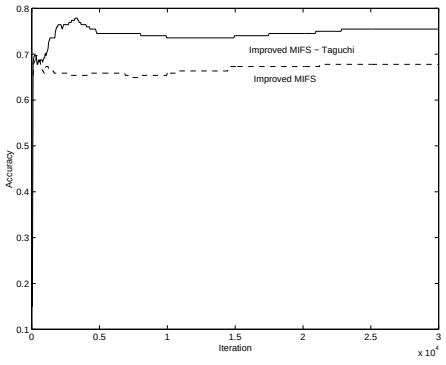


(b) 개선된 MIFS와 다구치 방법

그림 5.5: 상위 3개를 선택했을 때의 분류율 (Sonar 문제)



(a) MIFS와 다구치 방법



(b) 개선된 MIFS와 다구치 방법

그림 5.6: 상위 4개를 선택했을 때의 분류율 (Sonar 문제)

표 5.7: Sonar 문제에 다구치 입력특징 선택 알고리즘 사용시의 특징 선택 과정 (상위 6개 선택, 필터링 :개선된 MIFS, $L16$ 이용)

Run	F4	F5	F9	F11	F12	F21	F36	F44	F46	F49	F51	F52	–	–	–	방법 I	방법 II
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.8894	0.0
2	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	0.8798	0.6682
3	1	1	1	2	2	2	2	1	1	1	1	2	2	2	2	0.7981	0.8269
4	1	1	1	2	2	2	2	2	2	2	2	1	1	1	1	0.6827	0.8701
5	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2	0.8413	0.8125
6	1	2	2	1	1	2	2	2	2	1	1	2	2	1	1	0.6683	0.8076
7	1	2	2	2	2	1	1	1	1	2	2	2	2	1	1	0.6875	0.7980
8	1	2	2	2	2	1	1	2	2	1	1	1	1	2	2	0.6971	0.7788
9	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	0.8461	0.8509
10	2	1	2	1	2	1	2	2	1	2	1	2	1	2	1	0.7259	0.8317
11	2	1	2	2	1	2	1	1	2	1	2	2	1	2	1	0.8173	0.8701
12	2	1	2	2	1	2	1	2	1	2	1	1	2	1	2	0.7836	0.7307
13	2	2	1	1	2	2	1	1	2	2	1	1	2	2	1	0.8028	0.7980
14	2	2	1	1	2	2	1	2	1	1	2	2	1	1	2	0.7980	0.7788
15	2	2	1	2	1	1	2	1	2	2	1	2	1	1	2	0.7403	0.8076
16	2	2	1	2	1	1	2	2	1	1	2	1	2	2	1	0.7980	0.7980
선택순서	11	5	4	3	2	7	1	9	6	10	12	8	–	–	–	–	–

표 5.8: 다구치 방법과 MIFS의 비교 (Sonar 테이터) - 분류율(%)

	MIFS	개선된 MIFS	다구치 방법 (MIFS)	다구치 방법 (개선된 MIFS)
상위 3개	60.10	71.15	70.19	75.00
상위 4개	63.46	73.56	67.78	75.48
상위 6개	67.30	81.73	81.73	87.50
상위 9개	71.15	87.02	84.65	92.86
전체(60개)	100			

- **기타 문제**

Diabetes 문제

모두 8개의 입력으로 구성되어 있는 이 데이터에 대해서 다구치 방법을 사용해 상위 3~7 개의 입력 특징을 선택한 후 신경망으로 분류실험을 한 결과는 표 5.9에 보였다. 학습에 사용한 은닉층의 노드수는 5개이고 학습률은 0.01, 모멘텀은 0.9로 놓았다. 학습은 15,000회까지 반복하였다. 다구치 방법에 이용한 직교 행렬은 $L12$이다. 표에서 보면 알 수 있듯 이 데이터에 대해서 MIFS, 개선된 MIFS, 다구치 방법은 모두 비슷한 분류 성능을 보여준다.

Glass 문제

앞 장에서 설명한 이 데이터는 모두 9개의 입력 특징이 있으며 출력은 모두 6개의 종류이다. 이 데이터에 대해 9개 입력을 모두 사용하여 다구치 방법으로 상위 4~8 개의 입력 특징을 선택한 후 이들로 학습해서 얻은 분류율을 표 5.10에 보였다. 여기에 사용된 직교 행렬은 $L12$이고, 학습에 사용한 은닉층의 노드수는

표 5.9: Diabetes 데이터에 대한 입력 특징 선택 알고리즘의 분류율 (%)

	MIFS	개선된 MIFS	다구치 방법
상위 3개 선택	66.67	67.06	66.93
상위 5개 선택	70.07	71.48	69.79
상위 7개 선택	72.53	72.53	71.61
전체 (8개)		73.87	

표 5.10: Glass 데이터에 대한 다구치 특징 선택 알고리즘의 분류율 (%)

	MIFS	개선된 MIFS	다구치 방법
상위 4개 선택	46.72	64.02	62.07
상위 6개 선택	65.42	63.63	67.76
상위 8개 선택	77.01	78.97	82.71
전체 (9개)		82.18	

10개이며 학습률은 0.01, 모멘텀은 0.9로 놓았다. 학습은 15,000회까지 반복하였다.

이 데이터에 대해 다구치 방법을 사용했을 때 개선된 MIFS를 이용했을 때와 성능이 비슷한 것을 볼 수 있다.

Card 문제

15개의 입력 특징들로 이루어진 이 데이터에 대해서 필터링 없이 $L16$을 이용하는 다구치 방법으로 4~10 개의 입력특징을 선택하였다. 이들로 학습한 결과는 표 5.11에 MIFS와 4장에서 제시된 개선된 MIFS방법과 비교해서 나타냈다. 학습에 사용한 은닉층의 노드수는 10개이고 학습률은 0.01, 모멘텀은 0.9로 놓았

표 5.11: Card 데이터에 대한 다구치 특징 선택 알고리즘의 분류율 (%)

	MIFS	개선된 MIFS	다구치 방법
상위 4개 선택	78.84	84.20	85.02
상위 6개 선택	81.74	84.35	85.02
상위 8개 선택	83.48	87.97	90.01
상위 10개 선택	90.14	90.72	93.10
전체 (15개)		99.13	

다. 학습은 15,000회까지 반복하였다. 표를 보면 다구치 방법이 MIFS나 개선된 MIFS보다 약간 성능이 더 좋은 것을 볼 수 있다.

이상 여러 가지 문제에 다구치 방법을 적용해 입력 특징들을 선택한 결과 모든 입력 특징들을 다구치 직교 행렬의 각 열에 대응시켜 입력 특징 선택을 수행한 IBM 문제, diabetes, glass, card등의 문제에 대해서는 다구치 방법이 제 4장에서 제안된 개선된 MIFS 방법과 거의 비슷한 성능을 얻었다. 다구치 방법을 수행할 때 비록 적은 수이지만 신경회로망을 학습해야 하므로 공유정보를 이용하는 알고리즘들보다는 시간이 많이 걸린다. 따라서 다구치 방법은 Sonar 문제와 같이 입력 특징의 수가 20개 정도 이상인 경우에 일단 공유정보를 이용하는 알고리즘으로 필터링을 거친 특징들에 대해서 사용하는 것이 효과적일 것이다.

6

결론

신경회로망에서의 특징 선택은 분류에 사용하는 학습 알고리즘에 상관없이 학습집단의 분류성능에 매우 중요한 요소이다. 여러 개의 특징을 이용해 집단을 분류하고자 할 때 분류에 상관이 없거나(irrelavant) 중복된(redundant) 특징들이 존재하므로 이들을 제거하여 분류에 중요한 영향을 미치는 특징들만 추출해 학습시킴으로써 보다 나은 학습성능을 가져올 수 있다. 본 논문에서는 신경회로망의 여러개의 입력 특징들 중에서 분류 과정에 중요하게 작용하는 입력 특징을 선택하는 신경회로망에서의 입력특징 선택 문제를 다루었다.

이 문제에 대한 여러 가지 방법들 중 공유 정보를 이용하는 방법은 신경회로망을 반복학습시키지 않아도 되므로 학습에 걸리는 시간이 항상 걸림돌이었던 기존의 방법들에 대해 잇점을 갖는다. 본 논문에서는 이런 방법들 중 현재까지 많이 이용되어온 MIFS방법을 보완한 방법을 제 4장에서 제시했고 제 5장에서는 다구치의 직교 행렬을 이용해 비교적 적은 학습횟수로 출력에 중대한 영향을 미치는 입력 특징을 선택하는 방법을 제시하였다.

기존의 MIFS 방법은 입력들과 출력 사이의 공유정보 뿐만 아니라 이미 선택된 입력 특징들과 새롭게 선택될 입력 특징들 사이의 공유정보도 입력 선택에 이용한다는 점에서 중복된(redundant) 특징들을 제거할 수 있다는 강점을 가지나 중복된 특징이 출력에 대한 서로 다른 정보를 많이 가지고 있을 때 문제가 된다. 제 4장에서 제시된 개선된 MIFS 방법은 기존의 MIFS 방법의 중복된 특징을 제거할 수 있는 장점은 살리면서도 이러한 문제에 대해서도 좋은 성능을 내도록 설계되었다. 이 알고리즘을 여러 입력 선택 문제에 적용해 본 결과 기존의 MIFS 방법보다 일반적으로 우수한 성능을 낸다는 것을 확인할 수 있었다.

제 5장에서 제시된 다구치 방법을 이용하는 입력 특징 선택 알고리즘은 제 4장에서 공유 정보를 이용하는 알고리즘과는 별도로 어떻게 하면 신경회로망의 학습 횟수를 줄이면서 좋은 입력 특징을 선택할 수 있는가 하는 문제에 대한 하나의 해답으로서 제시되었다. N개의 입력 특징이 있을 때 다구치의 직교 행렬을 이용할 경우 $O(N)$번의 신경회로망 학습만으로 입력 특징 선택을 할 수 있고 여러 가지 입력 특징 선택 문제에 이 방법을 적용해 본 결과 기존의 방법보다 좋은 성능을 얻을 수 있었다.

다구치 입력 특징 선택 방법을 이용할 경우 기존의 신경회로망을 반복학습해야 하는 알고리즘들보다 입력 선택 과정의 시간을 줄여주기는 하나 $O(N)$회의 신경회로망 학습을 해야 하므로 N이 커지면 시간이 오래 걸릴 수 있다. 따라서 일반적으로 이 방법은 입력 특징의 갯수가 20개 이하일 때 모든 입력들을 직교 행렬의 각 열에 대응시켜 사용하는 것이 바람직하며 입력특징의 수가 20개정도 이상일 때는 일단 공유 정보를 이용하는 알고리즘과 같은 시간이 별로 안 걸리는 알고리즘으로 입력 특징들을 추려낸 후 이렇게 한번 걸러진 입

력 특징들을 다구치 방법을 이용해 입력 특징 선택을 하는 것이 효과적이다. 실제로 제 5.2.3에서 입력 특징의 수가 60개인 Sonar 문제에 이런 방법을 적용해 본 결과 MIFS나 개선된 MIFS만을 사용하는 것보다 좋은 성능을 얻을 수 있었다.

본 논문에서 제안된 개선된 MIFS 방법과 다구치 방법을 이용하는 입력 특징 선택 알고리즘은 여러 가지 입력들 중에서 중요한 입력을 선택해야 하는 입력 선택 문제에 대한 다양한 해결책 중 하나로 활발히 이용될 수 있을 것이다. 또 다구치 방법을 이용하는 입력 특징 선택 과정에서 신경회로망을 학습시키는 대신 공유 정보나 계산하는 데 시간이 덜 걸리는 다른 방법을 이용하는 알고리즘도 차후 생각해 볼 수 있을 것이다.

참고 문헌

[1] Joliffe, I.T., *Pricipal Component Analysis*, New York: Springer-Verlag, 1986.

[2] K.L Priddy et al., "Bayesian selection of important features for feedforward neural networks," *Neurocomputing*, vol. 5, no. 2 and 3, 1993.

[3] L.M. Belue and K.W. Bauer, "Methods of determining input features for multilayer perceptrons," *Neur. Comput.*, vol. 7, no. 2, 1995.

[4] Q. Li, and D.W. Tufts, "Principal feature classification," *IEEE Trans. Neural Networks*, vol. 8, no. 1, Jan. 1997.

[5] J.M. Steppe, K.W. Bauer Jr., and S.K. Rogers., "Integrated feature and architecture selection," *IEEE Trans. Neural Networks*, vol. 7, no 4, pp. 1007-1014, July 1996.

[6] R. Setiono and H. Liu, "Neural network feature selector," *IEEE Trans. Neural Networks*, vol. 8, no. 3, May 1997.

[7] Roberto Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Trans. Neural Networks*, vol. 5, no. 4, July 1994.

[8] R. Agrawal, T. Imielinski, and A. Swami, "Database mining: a performance perspective," *IEEE Trans. Know. and Data Eng.*, vol. 5, no. 6, Dec. 1993.

[9] T.M. Anwar, H.W. Beck, and S.B. Navathe, "Knowledge mining by imprecise querying: a classification based approach," *IEEE 8th Int. Conf. on Data Eng.*, Phoenix, Arizona, Feb. 1992.

[10] G. Taguchi, *Taguchi on Robust Technology Development*, ASME, New York, NY, 1993.

[11] W.Y. Fowlkes and C.M.Creveling, *Engineering Methods for Robust Product Design*, Addison-Wesley, New York, 1995.

[12] G.E. Peterson et al., "Using Taguchi's method of experimental design to control errors in layered perceptrons," *IEEE Trans. Neural Networks*, vol. 6, no. 4, July 1995.

[13] T.M. Cover, and J.A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.

Abstract

Feature Selection in neural networks is very important part, regardless of the learning algorithm which is used to train the network. Due to the existence of irrelevant and redundant attributes, by selecting only the relevant attributes of the data, higher predictive accuracy can be acquired. In this paper, the problem of feature selection, which is to select only the important features in classification procedure is dealt with.

For this purpose, I propose two algorithms. First is an algorithm which is the improved version of the MIFS which is proposed by Battiti based on the mutual information, and the second one is an algorithm using Taguchi method.

Improved MIFS algorithm is designed in order both to utilize the advantage of the MIFS which can exclude the redundant features and to suppress the undesirable side effects in feature selection procedure. The algorithm is applied to some of the problems widely tested by many researchers, and the performance of the proposed algorithm is better than that of MIFS in general.

The Taguchi feature selection algorithm is proposed as an answer to the question of how to find the good features with as small number of learning as possible apart from the algorithms using mutual information. Taguchi method performed well with the features filtered with mutual informative methods, especially when large number of features are available.

Keywords : neural network, feature selection, mutual information, Taguchi method, orthogonal array.

Student Number : 9 7 4 2 0 - 5 0 6