지능형 IoT서비스를 위한 기계학습 기반 동작 인식 기술

최 대 웅 · 조 현 중

고려대학교 컴퓨터정보학과

Ⅰ. 서 론

초기 웹(Web) 페이지의 형태는 단순히 텍스트를 표시해 주는 것에서 시작되었다. 그 후 텍스트에 의미를 부여해 주는 시맨틱 웹(Semantic Web)을 통해 Web 2.0, 소셜 네트워킹 웹(Social Networking Web) 시대를 맞게 된다. 하지만 센싱 기술, 컴퓨팅 자원 그리고 웹 어플리케이션 등의 빠른 발전으로 기존의 Web 2.0은 Web 3.0, 즉 유비쿼터스 컴퓨팅 웹(Ubiquitous computing web)으로 자연스럽게 진화되고 있다. 기존의 네트워크는 컴퓨터-컴퓨터 혹은 사람-컴퓨터 간에 상호작용이 형성되지만, 유비쿼터스 시대에서는 기존의 컴퓨터의 자리에 사물(혹은 모든 것)로 대체될 수 있다. 따라서 사람의 주변 환경에 있는 모든 것들이 무선으로 네트워크를 생성하고, 네트워크 안에서 서로 간에 데이터를 생성, 저장, 교환, 합성 등의 기능을 수행할 수 있게 된다. 사물인터넷(IoT)의 개념도 이것과 같이 시작되는데, 블루투스(Bluetooth)와 RFID(Radio Frequency Identification) 그리고 영상 센서 등을 이용한 무선 센싱 네트워크 기술들은 사물인터넷 형성을 위한 가장 중요한 요소로 꼽힌다[1].

사물인터넷은 사람의 환경 속으로 자연스럽게 또는 보이지 않는 곳에서 주변 상황을 지능적으로 인지하고 제어하게 하는 것이 근래의 정의다[1]. 또한 최근 센서의 발달은 스마트워치와 같은 소형 디바이스에 탑재될 수 있을 정도로 소형화되고 있을 뿐만 아니라, 배터리에 기반을 둔 다양한 웨어러블(Wearable) 컴퓨팅 장치에서 장시간 사용을 위해 저전력으로 구동될 수 있도록 하는 추세이다. 이러한 추세에 따라 최근 Strategy Analytics의 자료에 의하면 2020년에는 전 세계 사물 인터넷, 웨어러블 디바이스 및 각종 스마트 기기의 수가 약 330억 개에 다다르고, 이는 한 사람에 연결된 스마트 기기의 수가 현재보다 최소 2배에서 4.3배까지 증가하는 수치라고 예측하고 있다.

사물인터넷의 영역이 확장될 것으로 예상됨에 따라, 네트워크에 속한 사람과 사물 간에 생성될 미가공 데이터(raw data)의 수는 기하급수적으로 증가할 것이다. 지능형 IoT 서비스를 위해서는 사물인터넷이 자동적으로 사람의 의도와 주변 상황을 정확히 파악할 필요가 있다. 또한 사람-사물 간 네트워크 안의 수많은 미가공 데이터들 사이에서 중요한 의미를 파악하는 일은 실시간으로 처리되어야 한다. IoT 어플리케이션의 실시간 처리능력과 정확한 의미정보 파악은 사용자에게 해당 서비스가 어떻게 진행되고 있는지 또 이 다음에 무엇을 할 수 있는 지 등을 안내해 주는 피드백을 줄 수 있다. 이는 곧 해당 IoT 어플리케이션의 신뢰성과 연결되고, 더 나아가 지능형 IoT 서비스로 도약할 수 있는 기반을 다질 수 있을 것이다.

스마트폰이나 태블릿 PC와 같은 터치스크린 기반 디바이스의 출현은 사람들로 하여금 직관적인 입력 인터페이스의 중요성을 알려주기에 충분했다. 하지만, 사물인터넷의 목적과 각종 디바이스의 발전방향에 맞게 소형화되고 있는 스마트 기기들은 입력 방법으로 더 이상 터치 인터페이스만을 사용할 수는 없을 것이다. 특히, 대표적인 웨어러블 디바이스인 스마트워치는 너무 작은 스크린 크기로 인해 기존의 터치 입력 기술 외에 동작 인식(Gesture recognition) 기술이 많이 적용되고 있다. 그 중에서도 2015년에 구글은 Soli 프로젝트[2]를 통하여 미래의 동작 인식기술을 선보였는데, 이 기술은 무선 주파수를 이용하여 소형 레이더를 구성하고, 손가락들이 레이더 안에 들어왔을 때 신호를 주고받음으로써 손의 움직임을 파악하고, 손가락들 간의 미세한 동작으로 볼륨을 제어하거나, 시간을 조절하는 기능을 선보였다. 이는 시계에 적용될 수 있을 만큼 작은 컴퓨팅 장치, 즉 웨어러블 디바이스뿐만 아니라 사물인터넷에 까지도 확장이 가능한 형태를 암시했고, 사람과 사물간의 상호작용에 있어서 보다 실현가능성을 높여준 사례라 볼 수 있었다.

한편, 최근 기계학습 분야에서는 기존의 기계학습 알고리즘의 일반화 성능을 개선하기 위한 여러 연구가 진행되어

왔었다[3]~[6]. 먼저, 기계학습 모델 중 하나인 의사결정나무(Decision Tree)를 기반으로 구성된 Decision Forest의 연구들이 다양하게 진행되어져 왔었다[3],[4]. 다른 방면으로는, 인공신경망(Neural Network)의 은닉층(hidden layer)을 더 깊고 많이 구성하여 어려운 문제도 효율적으로 풀 수 있는 학습방법의 돌파구(breaktrough)가 제안되었고[5],[6], 이를 바탕으로 CNN(Convolutional Neural Network)[7], RNN(Recurrent Neural Network)[8]과 같은 딥러닝(Deep Learning)의 시대를 열게 해주었다. 놀랍게도 이러한 최신 기계학습 모델들이 특정 문제에서는 사람의 인지능력보다 더 뛰어난 성능을 보여주기도 했는데, 그 중에서도 Decision Forest류의 모델들은 최신 기계학습 모델 중에서도 대체적으로 우수한 성능을 보여주었다[9]. 따라서 센싱, 특징 추출 및 의미추론 등을 포함한 모든 영역에서 정확하고 빠르게 예측이 가능한 최신 고급 기계학습 알고리즘들은 지능형 IoT 서비스의 중추적 역할을 담당할 수 있을 것으로 여겨진다.

본 논문에서는 지능형 IoT 서비스를 위한 동작 인식 기술의 연구 동향과 기계학습 알고리즘 중에 하나인 Decision Forest의 자세한 내용을 소개한다. 먼저, 2장에서는 그 간의 동작 인식 기술이 어떻게 연구되어져 왔는지를 살펴볼 것이다. 그리고 3장에서는 의사결정나무에 대한 한계와 Decision Forest의 일반적인 개념 및 대표적인 학습방법과 특징을 소개하고, 본 논문을 결론지을 것이다.

Ⅱ. 동작 인식(Gesture Recognition) 기술 동향

사람-컴퓨터 또는 사람-사물 간의 상호작용을 위한 동작 인식에 관한 연구는 끊임없이 소개되고 있지만, 입력 및 제어를 위해서는 터치 인터페이스가 여전히 좋은 성능을 갖고 있다. 하지만 터치 인터페이스가 직관적이고 빠른 피드백을 줄 수 있더라도, 기기와의 직접적인 접촉이 필수적이기 때문에 사물인터넷 네트워크 안 사람-사물간의 상호작용에서는 동작인식 방법이 스크린 크기에 제한되지 않고, 조금 더 광범위하게 활용될 수 있는 잠재력을 가지고 있다. 특히 구글의 Soli 프로젝트나 구글 글래스(Google Glass)[10]와 같은 최신 미래지향형 웨어러블 디바이스들이 보여주었듯이 사람이 착용할 수 있을 만큼 소형화되는 스마트 디바이스에는 단순 터치 입력을 넘어 동작 인식이 주요 인터페이스로 활용되는 것을 보여주고 있다.

동작 인식은 손이나 팔과 관련된 동작이 특히 많이 연구되지만, 신체 구조 중 발[11], 머리나 얼굴[12] 또는 몸 전체[13]를 이용한 연구들까지 다양하게 연구되어지고 있다[14]. 본 논문에서는 그 중에서도 가장 보편적인 동작 인식 방법인 손과 팔에 대한 연구에 대해 소개한다. 손과 팔에 대한 연구는 크게 데이터 글로브나 손목에 착용하는 웨어러블 디바이스 방법과 카메라를 이용한 컴퓨터 비전방법으로 나누어 볼 수 있다.

먼저, 웨어러블 디바이스의 일종으로 사람이 컴퓨팅 장치와 여러 개의 선으로 연결된 장갑을 착용하고, 손 끝점 혹은 손의 전체 모양을 알 수 있는 다양한 방법들이 제안되어져 왔었다[14],[15]. 예를 들면 Chen 등은 손가락 끝점에 전자석(electromagnet) 형태의 마커들을 달고, 각각의 전류 주파수를 다르게 하여 자기장 센싱이 가능한 소형 중앙 장치를 통해 열 손가락의 세밀한 추적이 가능하도록 하였다[15]. 이러한 장갑형태 외에도, Thalmic Lab에서 만든 Myo 센서[16]는 팔목에 보호대와 같이 착용하는 형태의 근전도(electromyography) 측정 센서로서 다양한 손의 동작들을 근육의 수축, 이완 시 전위변화를 통해 판단한다. 이 센서와 함께 스마트워치나 가상현실 헤드셋 등의 다른 디바이스와의 상호작용을 위한 손 동작 인식 연구에 대한 내용도 소개되어져 오고 있다[17].

컴퓨터 비전을 이용한 동작 인식은 일반 RGB 카메라나 적외선 카메라를 이용하게 된다. 더 나아가 적외선을 응용하여 깊이(depth) 값을 알 수 있는 Microsoft Kinect v2, Leap Motion, Asus Xtion 등의 깊이 센싱 카메라들이 보편화되면서 다양한 동작 인식 어플리케이션들이 문자 입력과 컨텐츠 제어 등에 사용되고 있다[14]. 컴퓨터비전 방식의 동작 인식은 데이터 글로브를 이용하는 것보다 사람이 무언가를 입거나, 직접적으로 장착하지 않을 수 있다는 장점이 있지만, 그에 반해 치명적인 단점을 갖고 있다. 먼저 빛의 변화에 따라 성능에 큰 영향을 미칠 수 있다는 것인데, 이 경우는 기존의 RGB 카메라 대신 적외선 카메라를 이용하게 되면서 어느 정도 해결할 수 있었다. 하지만 특정 영역의 어떤 물체에 의한 가림 현상(Occlusion), 즉 카메라에 의해 보이지 않는 영역은 추적이 불가능하다는 것이 여전히 큰 문제로 남아있었다. 이를 개

선하고자 여러 대의 카메라를 이용한 다중 스테레오 카메라 방법이 제안되어져 왔지만[18], 특정 시스템의 가림 현상을 줄이고자 매번 여러 대의 카메라를 사용하는 것은 비용도 많이 들 뿐만 아니라, 그만큼의 컴퓨팅 자원도 따라주어야만 했다. 하지만 최근 Sinha 등은 최신 기계학습 알고리즘을 이용해 단일 깊이 영상 카메라만을 가지고 한 손에 대해 가림 현상이 발생했을지라도, 손의 다른 특징들을 이용해 보이지 않는 손가락의 관절 각도를 추정하여 전체 손 모양이 추적 가능하다는 것을 보여주었다[19]. 이는 최신 기계학습 기술들이 가림 현상을 극복할 수 있다는 가능성을 보여준 좋은 사례로 보인다.

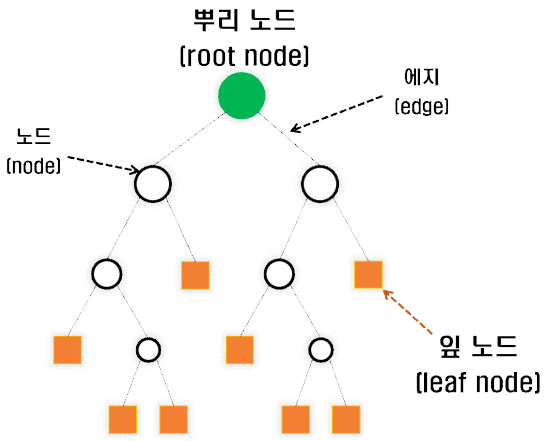
또, 최근 몇 개의 논문에서는 기존의 동작 인식 방법에 Decision Forest를 적용하는 방법을 소개하고 있다[13],[20],[21]. 그 중에서도 Microsoft 키넥트 v2는 Decision Forest의 가장 실용적인 어플리케이션 사례로 꼽힌다. 키넥트는 칼라와 깊이영상을 동시에 획득할 수 있는 디바이스로, 그것의 주요 특징 중 하나로는 키넥트 앞에 있는 여러 사람의 골격구조를 실시간으로 인식, 추적하여 게임 및 다양한 동작 인식 어플리케이션에 응용할 수 있다는 것이었다. 이 때 골격구조를 인식하고 추적하기 위하여 Decision Forest를 사용하였고, 좋은 성능을 보여주고 있다[13]. 따라서 저 비용, 저 전력으로 구동되는 소형 컴퓨팅 장치에서 동작 인식을 위한 최신 기계학습 알고리즘들의 활용은 앞으로도 중요하게 다뤄질 것으로 전망된다.

Ⅲ. Decision Forest

이번 장에서는 Decision Forest에 대해 알아보기에 앞서, 그에 기반이 되는 의사결정나무(Decision Tree)에 대한 간략한 설명과 함께 의사결정나무의 특성 및 한계점은 어떤 것이 있었고, 그것을 개선하기 위해 Decision Forest가 어떤 식으로 발전해 왔는지를 소개한다.

3-1 의사결정나무(Decision Tree)

컴퓨터 공학에서의 자료 구조에는 트리(Tree)라는 중요한 개념이 있다. 의사결정나무는 이름에서도 알 수 있듯이, 트리를 기본 구조로 사용하게 되는데, [그림 1]은 트리의 구조



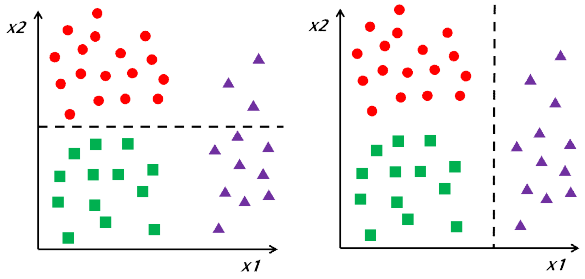
[그림 1] 이진 트리(Binary tree)의 기본 구조. 각각의 원은 노드(node)를 나타내고, 노드와 노드를 연결해주는 에지(edge)로 구성되어 있다.

를 잘 표현해주고 있다. 먼저 트리는 동그라미로 표시된 노드(node)와 노드와 노드를 이어주는 에지(edge)로 구성이 된다. 각 노드는 그것의 자식 노드(child node)를 가질 수 있는데, 자식노드를 최대 2개만 가질 수 있는 트리를 이진트리(binary tree)라고 한다. 이진트리는 이해하기 쉬울 뿐만 아니라, 다른 형태의 트리를 동등한 이진 트리로 표현 가능하기 때문에[4], 본 논문은 이진 트리만을 이용하여 설명한다. 이진 트리에서 특정 노드의 자식노드가 2개일 때, 왼쪽에 있는 것을 왼쪽 자식 노드(left child node)라 하고, 오른쪽에 있는 것을 오른쪽 자식 노드(right child node)라고 한다. 또한 트리의 맨 위에 있는 노드를 뿌리 노드(root node)라 하고, 더 이상 자식 노드를 갖지 않는 노드를 잎 노드(leaf node)라고 한다.

각 노드에서는 특정 데이터가 들어왔을 때 그것을 왼쪽이나 오른쪽 자식노드로 보낼 수 있는 질문을 포함하고 있는데, 데이터를 왼쪽과 오른쪽 자식노드로 분류하는 질문을 특징공간의 차원 수에 따라 다르지만, 기계학습에서는 결정초평면(Decision hyperplane)이라고 부른다. 이러한 결정초평면들은 의사결정나무의 학습(learning)을 통해 정해지게 된다. 학습의 목적은 데이터들의 부류(class)가 최대한 같은 것들끼리 분류가 되도록, 즉 데이터의 동질성이 높도록 각 노드의 결정초평면을 만드는 것이다. 이때 왼쪽과 오른쪽 자식노드로 분류된 데이터들의 동질성을 측정해주기 위해 불

순도(Impurity)의 개념이 사용된다. 불순도는 여러 방법에 의해 정의할 수 있는데[22], 모든 부류들이 특정 노드에 있을 확률이 같을 경우에 가장 큰 값을 갖는다. 반대로 특정 노드에서 하나의 부류에 대한 데이터들만 있을 경우에는 불순도가 가장 낮은 경우로, 그 때 데이터들을 분류하는 결정초평면은 해당 노드의 분류기로 선택되기에 가장 좋은 형태이다. 본 논문에서는 설명의 편의를 위해 간단한 예를 들어 설명한다. [그림 2]는 의사결정나무를 학습시키기 위한 3개 부류에 대한 훈련 데이터 집합들의 2차원 분포도를 보여주고 있는데, 같은 부류에 대해서는 같은 모양을 나타내고 있다. 각 데이터들이 2차원 특징공간(x1, x2)을 갖는다고 가정했을 때, [그림 2] (a)와 [그림 2] (b)는 데이터를 서로 다르게 분류하고 있다. 이때 [그림 2] (a)보다는 [그림 2] (b)의 형태로 분류를 했을 때, 불순도의 정의에 의해 오른쪽의 데이터가 완전히 분리가 되므로 동질의 데이터들로 잘 구별되어진 것을 알 수가 있고, 불순도가 가장 낮다. 따라서 학습 시에 각 노드는 [그림 2] (b)와 같이 왼쪽 노드와 오른쪽 노드로 동질의 데이터가 더 잘 분류될 수 있는, 즉 불순도를 최소화하는 결정초평면을 선택하게 된다.

이렇게 학습이 완료된 의사결정나무는 테스팅(testing) 과정을 통해서 성능을 확인하게 된다. 학습에 사용했던 훈련 데이터 집합과는 다른 테스트 데이터 집합이 사용되는데, 학습 시 보지 못했던 데이터에 대해서 성능을 측정하게 되



(a) 결정 초평면 1 (b) 결정 초평면 2

[그림 2] 2차원 특징공간(x1, x2)를 갖는 데이터들의 분포도. 서로 다른 모양과 색으로 표시된 3개의 부류가 (a)와 (b)의 결정 초평면에 의해 분류된 모습을 보이고 있다. 불순도의 정의에 의해 (a)보다는 (b)가 동질의 데이터로 잘 분류하고 있다.

므로 이것을 기계학습의 일반화(generalization) 성능이라고 한다.

학습이 완료된 의사결정나무는 모든 노드가 데이터를 어떻게 분류할 지에 대한 결정초평면이 만들어져 있는데, 이때 의사결정나무가 잘 학습이 되었는 지를 확인하기 위한 테스팅(testing) 과정을 거쳐야만 한다. 테스팅에서는 훈련 데이터와는 구별되는 테스트 데이터들을 뿌리노드에서부터 시작해서 각 노드를 거쳐 어떤 잎 노드에 도착하는 지를 통해 결과를 도출해내면 된다. 이때 결과로 도출된 값, 즉 의사결정나무가 예측한 값과 실제 테스트 데이터의 값의 비교를 통해 예측이 잘 되었는지를 측정하게 된다.

3-2 의사결정나무의 한계와 Decision Forest

기존의 의사결정 나무는 좋은 일반화 성능으로 분석을 위한 좋은 도구로서 많이 사용되어져 왔었다[23],[24]. 하지만 의사결정 나무의 일반화 성능을 높이기 위해 나무의 깊이(depth)를 깊게 할수록 오히려 해당 훈련 데이터 집합에 과적합(overfitting)되어 결과적으로는 일반화 능력이 떨어지는 단점을 갖고 있었다. 이 문제를 개선하기 위해 뿌리 노드부터 성능에 큰 도움이 되지 않는 노드들을 제거하는 가지치기(Pruning)[25]와 같은 기법이 제안되었지만, 추가적인 시간 및 컴퓨팅 자원을 투자해야할 뿐만 아니라, 기존 모델의 성능 한계를 뛰어넘는 것처럼 보이지는 않았다.

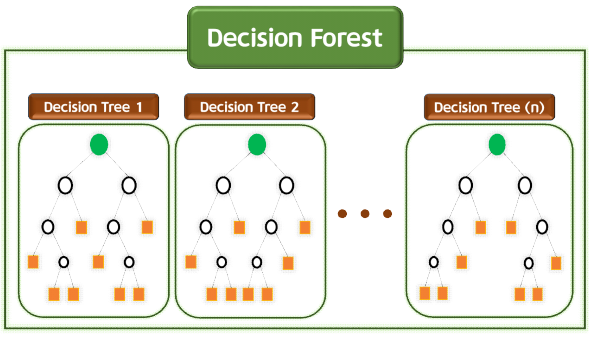
한편, 기계학습 분야에서는 의사결정나무뿐만 아니라, 인공신경망(Neural Net), SVM(Support Vector Machine)과 같은 단일 기계학습 알고리즘들의 일반화 성능을 개선하고자 앙상블 개념이 제안됐었다[3],[22]. 앙상블은 특정 문제를 해결하기 위해 하나의 기계학습 모델이 아닌 서로 다른 여러 개의 기계학습 모델들로 해를 찾는 방법이다. 앙상블 내의 각 기계학습 모델들은 서로 다른 구조를 나타낼수록, 즉 모델들의 다양성(diversity)이 보장될수록 단일 기계학습 모델보다 좋은 일반화 능력을 갖는다는 것이 증명되어 왔다[26].

Decision Forest는 의사결정나무(Decision Tree)의 앙상블 개념으로써 1990년 초기에 처음 소개되고, 90년대 중반부터 관심을 받기 시작했다[27],[28]. 기존의 의사결정나무는 단일 모델로서 사용될 때에 불안정성이라는 큰 단점이자 특징을 갖고 있었다. 불안정성이란 학습에 사용되는 데이터 집합에

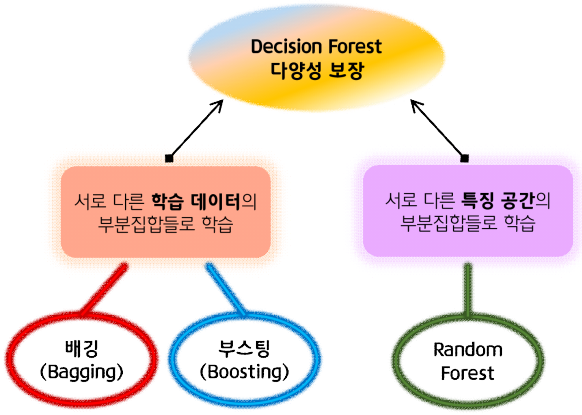
약간의 변화가 생기면 나무의 구조가 크게 변하는 것을 말하는데, 이러한 불안정성은 단일 모델로만 사용했을 시에는 단점으로 작용하였지만, [그림 3]과 같이 서로 다른 여러 개의 의사결정나무로 구성된 앙상블 모델에서 다양성을 보장해주는 하나의 방법이 될 수 있었다. 이러한 생각에서 의사결정나무가 Decision Forest로 발전될 수 있었고, 각 의사결정나무의 불안정성이 앙상블 모델에서의 다양성을 보장해주면서 단일 의사결정나무보다 더 좋은 일반화 성능을 보여준다는 것은 여러 논문에서 증명되어 왔었다[3],[4],[22],[26].

Decision Forest는 해결하고자 하는 문제 도메인과 어떤 특징을 갖는 데이터를 사용하느냐에 따라 분류(Classification)와 회귀(Regression)문제 외에도 다양한 어플리케이션에 적용될 수 있었는데, 그 중에서도 분류(Classification)문제에 대한 성능은 Decision Forest가 유명해지는 데에 큰 이바지를 했다[4]. 다음 절에서는 Decision Forest를 사용하기 위한 학습 및 테스팅 방법을 소개할 것이다.

먼저, Decision Forest의 다양성을 보장해주기 위한 다양한 학습방법이 제안되어져 왔다[29],[30]. [그림 4]에서 볼 수 있듯이, 본 논문에서는 대표적으로 많이 인용되고 사용된 3가지 학습 방법을 소개하는데, 각각은 배깅(Bagging)[29],[30], 부스팅(Boosting)[27] 그리고 Random Forest[3],[4]가 이에 해당한다. 3가지 학습방법들은 다양성을 보장해주는 방법에 따라 크게 두 가지로 분류할 수 있는데, 첫 번째는 각 의사결정나무마다 훈련 데이터 집합을 다르게 구성하여 학습하는 것이고 3-3절에서는 이 방법에 해당하는 배깅과 부스팅 방법에 대해 설명한다. 다른 하나로는 각 의사결정나무가 전체 훈



[그림 3] 여러 개의 서로 다른 의사결정나무들로 구성된 Decision Forest



[그림 4] Decision Forest의 다양성을 보장하기 위한 대표적인 3가지 학습(learning)방법의 구분. Decision Forest 내의 의사결정나무들을 학습데이터(배깅과 부스팅) 또는 특징 공간(Random Forest)을 서로 다르게 하여 학습시킨다.

련 데이터를 이용하여 학습하지만, 특징 벡터의 서로 다른 부분집합들을 이용하여 학습하게 되는 방법이다. 3-4절에서 설명하는 Random Forest가 이에 해당하며, 이에 대한 자세한 설명은 다음 절에서부터 차례대로 살펴본다.

3-3 배깅(Bagging)과 부스팅(Boosting)

앙상블 생성의 기본적인 방법들 중 하나인 배깅[29],[30]은 훈련데이터 집합을 다양하게 구성해주는 재 샘플링(resampling) 기법으로, 간단하면서도 효과적인 방법으로 알려져 있다. 배깅에서는 Decision Forest의 다양성을 보장하기 위해 각 의사결정나무를 전체 학습 데이터 집합의 서로 다른 부분집합으로 학습하게 한다. 즉, 임의의 의사결정나무에 대해서 전체 훈련데이터 집합에서 임의의 개수로 데이터를 추출하여 훈련데이터 부분집합을 구성하고, 이 부분집합만으로 해당 의사결정나무를 학습시킨다. 그리고 그 다음 의사결정나무에 대해서는 다시 기존에 선택된 데이터를 포함한 전체 훈련데이터 집합 내에서 다시 임의의 개수로 데이터를 추출하여 또 다른 부분집합을 구성하고, 이 부분집합으로 학습시킨다. 이 과정이 Decision Forest 내의 모든 의사결정나무에 반복된다. 이때 의사결정나무는 불안정성을 갖고 있기 때문에, 결국 모든 의사결정나무들은 서로 다른 구조를 갖게 되고, 이

것이 Decision Forest의 다양성을 확보하게 해준다.

부스팅[27]은 배깅과 같이 앙상블 생성을 위한 대표적인 재샘플링 기법 중 하나로 Decision Forest에도 적용되어져 왔다. 배깅에서는 Decision Forest 내의 각 의사결정나무들이 독립적으로 구성되어졌다면, 부스팅에서는 조금 더 개선된 형태로 특정 의사결정나무가 이전의 의사결정나무에 연관성을 가지면서 구성된다. 부스팅에서도 하나의 의사결정나무를 학습시키기 위해 훈련 데이터 집합에서 임의로 추출된 부분집합을 사용하게 되는데, 이때 한번 선택된 데이터는 다시 선택될 수 없도록 한다는 점에서 배깅과 다르다. 또, 각 데이터는 이전에 만들어진 의사결정나무에 의해 가중치 값을 부여받게 되는데, 이전의 의사결정나무가 잘 분류한 데이터보다 잘못 분류한 데이터에 더 높은 값을 매긴다. 즉, 가중치가 높은 데이터를 더 중요하게 취급하고, 가중치가 큰 학습 데이터를 잘못 분류하면 더 큰 오류를 범한 효과를 낸다. 그 다음에 만들어진 의사결정나무 분류기는 높은 가중치를 갖는 데이터에 대한 오류를 최소화하도록 학습이 된다.

단순히 훈련 데이터 집합을 부분적으로 추출하여 학습하는 간단함에도 불구하고, 배깅과 부스팅은 전체 훈련 데이터 집합을 이용하는 단일 의사결정나무보다 대체로 더 좋은 일반화 성능을 보여왔다[29]. 이러한 이유는 Decision Forest내의 다양성에 의해 각 의사결정나무들이 서로 다른 다양한 결과를 도출할 수 있고, 그 안에서 최적의 결과를 선택하기 때문이었다. 다시 말하면, 단일 의사결정나무의 불안정성을 위의 방법들이 보다 안정적인 결과를 도출해내도록 개선할 수 있었다는 것을 보여준다.

3-4 Random Forest

위의 방법들은 Decision Forest 내의 각 의사결정나무들을 다른 훈련 데이터들로 학습이 되게 해주는 방법을 취했다. 한편, 대체적으로 좋은 성능을 보였음에도, 배깅과 같은 경우, 효율성에 문제점을 있음을 지적하는 의견도 있었다[4]. 첫 번째 문제로는 배깅을 이용할 때에 전체 학습 데이터를 이용되지 않아 데이터가 낭비될 수 있다는 것이다. 배깅의 경우, 각 의사결정나무를 위한 훈련 데이터의 부분집합을 추출할 때, 한번 선택된 데이터를 다시 포함하는 대치(replacement)를 허용하기 때문에 특정 데이터가 반복적으로 선택되거나 또는 그 데이터가 단 한 번도 학습에 사용되지 않을 수 있었다. 또한, 배깅과 부스팅은 의사결정나무 학습 시에 모든 특징 공간을 이용하게 되는데, 이러한 방법들은 데이터의 특징공간의 차원 수가 매우 높은 경우, 예를 들면 영상처리에서 이미지의 각 픽셀화소수를 특징공간으로 갖는 경우, 각 의사결정나무의 모든 노드에 대해서 전체 특징공간으로 결정초평면을 정하는 일은 실용적이지 못하다.

이것을 개선하기 위해 전체 학습 데이터를 이용하지만, 특징공간을 나누어 학습하는 Random Forest 방법이 제안되었다[3],[4]. 훈련 데이터 집합을 재 샘플링하는 것과 비슷하게, Decision Forest 내의 의사결정나무들의 각 노드에서는 전체 특징공간을 임의의 개수만큼 선택한 특징공간의 부분집합으로 훈련 데이터를 최적으로 나누게 된다. 이 때 추출될 특징공간의 개수, 즉 부분집합의 크기는 사용자에 의해 조절될 수 있고, 이러한 과정을 의사결정나무에 randomness를 부여한다고 한다. 부분집합의 크기는 1부터 전체 특징 공간의 개수까지 임의로 설정될 수 있는데, 크기가 1인 경우에는 의사결정나무들의 각 노드가 한 개의 특징공간으로 훈련 데이터를 분류하게 된다. 반대로, 부분집합의 크기가 전체 특징 공간의 개수와 같다면 그 Decision Forest에는 randomness가 전혀 주입되지 않은 방법이고, 그것은 단지 구조가 동일한 의사결정나무가 여러 개가 있을 뿐이다. 따라서 Random Forest의 경우는 의사결정나무의 각 노드가 학습에 사용할 특징공간을 서로 다르게 해줌으로써 다양성을 보장해주는 방법이다.

최근 제안된 Decision Forest의 학습방법 중에서도 Random Forest는 배깅과 부스팅보다도 대체적으로 가장 좋은 일반화 성능을 보여주면서[26], 가장 인기가 많고 이것이 제안된 이후에는 Decision Forest를 Random Forest, Randomized Forest, Randomized Decision Forest로 불리기도 한다. 이번 절에서 설명한 특징 공간을 서로 다르게 구성하는 방법 외에도, Decision Forest에 randomness를 부여하기 위한 다양한 방법들이 최근에까지 연구되어지고 있다[3],[4],[31],[32]. Extremely Randomized trees[31]는 Decision Forest 내 의사결정나무들의 각 노드에서 특징공간의 부분집합 개수뿐만 아니라, 데이터를 분류하는 결정초평면 모델과 그와 관련된 임계값까지도 임의로 선택하는 방법을 제안했다. 또, Rotation forest[32]는 각 의

사결정나무들을 전체 훈련 데이터를 이용하여 학습시키지만, 특징공간을 회전시키는 방법으로 각 의사결정나무들 구성 간에 차이를 주어 다양성을 보장해주는 방법을 제안했다.

3-5 테스팅(Testing)

Decision Forest들은 학습 데이터와 구별되는 테스트 데이터 집합을 가지고 해당 모델의 성능을 테스트하게 된다. 기존의 의사결정나무의 테스트방법과 같이 각 테스트 데이터가 Decision Forest 내의 모든 의사결정나무의 뿌리노드(root node)에서 시작해서 각 노드의 test function을 거쳐 왼쪽이나 오른쪽 자식노드로 옮겨가고, 다시 자식 노드의 test function을 거쳐 그것의 자식노드로 옮겨가는 과정을 잎 노드(leaf node)에 도달할 때까지 반복한다. 이 과정을 통해 각 의사결정나무의 잎 노드에 도달하게 되면 그 결과들을 하나의 해로 합쳐서 최종 결과를 출력하도록 한다.

3-6 Decision Forest의 결합

Decision Forest 내의 모든 의사결정나무가 학습이 완료되면 그것들의 결과를 하나의 해로 결정하는 과정이 필요하다. Decision Forest가 해결하고자 하는 문제, 사용된 학습 알고리즘, 데이터의 특성에 따라 잎 노드들의 해를 합치는 방법은 여러 방법이 존재한다[4],[22]. 대표적으로 다수 투표(majority voting)와 가중 다수 투표(weighted majority voting), 그리고 평균방법이 있다. 투표방법은 해 집단에 대해서 잎 노드들의 결과를 가지고 투표를 해서 가장 많은 표를 받은 특정 해를 Decision Forest의 결과로 선택하는 방법이다. 가중 다수 투표방법은 부스팅으로 학습된 경우에 사용되는 앙상블 결합방법으로 Decision Forest내의 의사결정나무 중에 신뢰도가 높은 분류기에 비중을 더 주는 방법이다. 또, 평균방법은 해 집단에서 잎 노드들의 결과 평균값과 가장 유사한 해를 결과로 선택하는 방법이다. 이 때 같은 훈련 데이터들을 가지고 같은 방법으로 학습된 서로 다른 Decision Forest들일지라도 해를 합치는 방법에 따라 다른 결과를 내기도 한다[4]. 그러므로 결합방법을 적용했을 때의 데이터 분포를 참고하여 풀어내고자 하는 문제와 데이터의 특성에 맞는 방법을 선택하는 것이 필요하다.

3-7 특징

SVM류의 알고리즘이나 기계학습 알고리즘들의 일반적인 앙상블을 위한 부스팅과 같은 알고리즘들은 좋은 성능을 보였음에도, 이 기술들은 다중 부류 문제(Multiple class problem)로는 확장되기에 어려움이 있었다[33],[34]. 하지만, Decision Forest는 클래스 수에 상관없이 일관되게 좋은 성능을 보여줬을 뿐만 아니라[35],[36], 고차원의 특징벡터를 갖는 이진 분류 (Binary classification)에서도 다른 알고리즘들보다 좋은 일반화 능력을 보여줬었다[4],[37].

더 나아가, Decision forest가 유명해진 이유는 분류(Classification)나 회귀(Regression) 문제와 같은 지도 학습(Supervised Learning)뿐만 아니라, 비지도 학습(Unsupervised Learning)이나 준지도 학습(Semi-supervised Learning)에서도 뛰어난 성능을 보여주었기 때문이다[4]. 최근 연구에서는 Decision Forest를 포함한 179개의 최신 기계학습 알고리즘들의 성능을 비교하는 연구를 진행했었다[9]. 그 논문에서는 모델들의 성능비교가 문제의 도메인과 데이터의 특성의 영향을 받을 수는 있지만, 많은 경우에 Decision Forest가 대체적으로 가장 좋은 성능을 낸다는 것을 보여주었다. 이 밖에도, Decision Forest를 인기를 뒷받침하는 근거가 되는 특징들은 여러 가지가 있었다.

첫 번째로 다수의 의사결정나무를 기반으로 구성된 Decision Forest는 테스팅 및 추론을 빠르게 진행할 수 있다. 의사결정나무는 단순히 각 노드의 질문에 맞는지 틀리는지에 대한 조건문을 통해 그것의 자식노드로 전달한다. Decision Forest의 기초가 되는 의사결정나무의 테스팅 및 예측알고리즘은 이러한 과정에서 낮은 컴퓨팅 자원을 요구하기 때문에, Decision Forest 또한 컴퓨팅 자원관점에서 매우 효율적이라고 알려진 바 있다[4].

이전 장에서 설명한 배깅과 Random Forest의 randomness의 개념은 상호 배타적(mutually exclusive)이지 않다. 이 말은 Decision Forest 내의 각 의사결정나무를 학습시킬 때에 배깅과 randomness가 동시에 적용될 수 있다는 것을 말한다. 실제로, Extremely Random Forest나 Random Forest와 관련된 연구에서는 이 두 방법을 같이 적용한 사례도 있고, 두 개념을 하나로 방법으로 묶어서 소개한 바도 있다[3],[4].

부스팅은 이전에 만들어진 의사결정나무와 다음에 만들어질 의사결정나무가 서로 연관성을 갖지만, 배깅과 Random Forest는 모델들 간에 독립적인 관계를 갖는다. 한편, 최근 멀

티코어(multi-core) 또는 매니코어(many-core) CPU의 저전력, 고성능 지향적 발전과 그래픽카드(GPU)의 성능발전은 독립적인 태스크들의 단순 반복계산을 실시간 가속화가 가능하도록 해주고 있다. 따라서 배깅과 Random Forest의 경우, 학습을 위한 병렬처리가 충분히 가능하다[4],[9],[38].

결론적으로 최신 컴퓨팅 자원의 발전과 저 전력에서 구동 가능하게 설계되는 최신 스마트 기기의 요구에 맞게 다양한 특성을 가진 Decision Forest는 지능형 IoT 서비스를 위한 최신 기계학습 알고리즘 중에 최선으로 꼽을 수 있다. 사물인터넷의 영역이 커짐에 따라 더 많은 데이터를 실시간으로 처리하고 응용할 수 있어야 하는데, Decision Forest의 우수성은 사물 인터넷의 자동적인 상황 인지 시스템뿐만 아니라, 사람-사물 간의 상호작용을 위한 동작 인식 분야에서도 충분히 좋은 성능을 낼 것으로 예상된다.

Ⅳ. 결 론

본 논문에서는 지능형 IoT 서비스를 위한 동작 인식 기술에 대한 개발 및 연구 동향을 살펴봤다. 다양한 센싱 디바이스들의 소형화 추세에 따라, 적용된 어플리케이션이 더욱 다양해지고, 그에 따른 사람과 사물간의 상호작용이 동작 기반으로 이루어질 수 있다. 또, 동작 인식을 위해 기계학습기반의 기존 알고리즘들을 살펴보았고, 최신 기계학습 알고리즘 중에 하나인 Decision Forest에 대해서 자세히 다루어보았다. 기계학습의 앙상블 개념인 Decision Forest는 근래의 고급 기계학습 알고리즘들과 비교하여 대체적으로 우수한 성능을 보여주었는데, 그러한 이유 중에 하나로 Decision Forest를 구성하는 각 의사결정나무들이 다양한 학습방법으로 기존의 단일 의사결정나무가 갖는 불안정성을 보완해주고, 다양성을 보장해주기 때문이다. 최근 연구들에 의하면 동작 인식을 위한 Decision Forest의 활용은 기존 알고리즘을 대체할 수 있을 뿐만 아니라, 더 좋은 성능을 기대할 수 있는 대안으로 충분한 조건을 갖고 있는 것으로 생각된다.

Ⅴ. 요 약

최근 RFID와 같은 무선 센싱 네트워크 기술과 객체 추적을 위한 센싱 디바이스 및 다양한 컴퓨팅 자원들이 빠르게 발전함에 따라, 기존 웹의 형태는 소셜 웹에서 유비쿼터스 컴퓨팅 웹으로 자연스럽게 진화되고 있다. 유비쿼터스 컴퓨팅 웹에서 사물인터넷(IoT)은 기존의 컴퓨터를 대체할 수 있는데, 이것은 곧 한 사람과 주변 사물들 간에 연결되는 네트워크가 확장되는 것과 동시에 네트워크 안에서 생성되는 데이터의 수가 기하급수적으로 증가되는 것을 의미한다. 따라서 보다 지능적인 IoT 서비스를 위해서는, 수많은 미가공 데이터들 사이에서 사람의 의도와 상황을 실시간으로 정확히 파악할 수 있어야 한다. 이때 사물과의 상호작용을 위한 동작 인식 기술(Gesture recognition)은 집적적인 접촉을 필요로 하지 않기 때문에, 미래의 사람-사물 간 상호작용에 응용될 수 있는 잠재력을 갖고 있다. 한편, 기계학습 분야의 최신 알고리즘들은 다양한 문제에서 사람의 인지능력을 종종 뛰어넘는 성능을 보이고 있는데, 그 중에서도 의사결정나무(Decision Tree)를 기반으로 한 Decision Forest는 분류(Classification)와 회귀(Regression)를 포함한 전 영역에 걸쳐 우월한 성능을 보이고 있다. 따라서 본 논문에서는 지능형 IoT 서비스를 위한 다양한 동작 인식 기술들을 알아보고, 동작 인식을 위한 Decision Forest의 기본 개념과 구현을 위한 학습, 테스팅에 대해 구체적으로 소개한다. 특히 대표적으로 사용되는 3가지 학습방법인 배깅(Bagging), 부스팅(Boosting) 그리고 Random Forest에 대해 소개하고, 이것들이 동작 인식을 위해 어떠한 특징을 갖는지 기존의 연구결과를 토대로 알아보았다.

참 고 문 헌

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions", *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645-1660, Sep. 2013.

[2] Google Soli Project, https://atap.google.com/soli/

[3] L. Rokach, "Decision forest: Twenty years of research", *Information Fusion*, vol. 27, pp. 111-125, Jan. 2016.

[4] A. Criminisi, J. Shotton, *Decision Forests for Computer Vision and Medical Image Analysis*, Springer, pp. 7-45, 2013.

[5] G. E. Hinton, S. Osindero, and Y. W. The, "A fast learning algorithm for deep belief nets", *Neural Computation*, vol. 17, no. 7, pp. 1527-1554, May. 2006.

[6] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks", *Advances in Neural Information Processing Systems*, vol. 19, pp. 153-160, 2007.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", *Advances in Neural Information Processing Systems*, pp. 1097-1105, 2012.

[8] J. Schmidhuber, "Deep learning in neural networks: An overview", *Neural Networks*, vol 61, pp. 85-117, Jan. 2015.

[9] M. Fernandez-Delgado, E. Gernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems", *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3133-3181, Oct. 2014.

[10] Google Glass FAQ, https://developers.google.com/glass/, Jan. 2014.

[11] K. Fukahori, D. Sakamoto, and T. Igarashi, "Exploring subtle foot plantar-based gestures with sock-placed pressure sensors", In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 3019-3028, Apr. 2015.

[12] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210-227, Apr. 2009.

[13] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, and A. Blake, "Efficient human pose estimation from single depth images", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2821-2840, Oct. 2013.

[14] S. Mitra, T. Acharya, "Gesture recognition: A survey", *IEEE Trans. Systems, Man, and Cybermetics, Part C (Applications and Reviews)*, vol. 37, no. 3, pp. 311-324, 2007.

[15] K. Y. Chen, S. Patel, and S. Keller, "Finexus: Tracking precise motions of multiple fingertips using magnetic sensing", In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 1504-1514, 2016.

[16] Tharmic labs의 Myo센서, https://www.myo.com, 2012.

[17] Y. Yang, S. Chae, J. Shim, and T. D. Han, "EMG sensor-based two-hand smart watch interaction", In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pp. 73-74, Nov. 2015.

[18] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 492-504, Mar. 2009.

[19] A. Sinha, C. Choi, and K. Ramani, "DeepHand: Robust hand pose estimation by completing a matrix imputed with deep features", In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4150-4158, 2016.

[20] N. C. Camgoz, A. A. Kindiroglu, and L. Akarun, "Gesture recognition using template based random forest classifiers", In *Workshop at the European Conference on Computer Vision*, pp. 579-594, Sep. 2014.

[21] X. Zhao, Z. Song, J. Guo, Y. Zhao, and F. Zheng, "Real-time hand gesture detection and recognition by random forest", In *Communications and Information Processing,* Springer Berlin Heidelberg, vol. 289, pp. 747-755, 2012.

[22] 오일석, "패턴인식", 교보문고, 2008.

[23] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, Chapman and Hall/ CRC press, 1984.

[24] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[25] J. Mingers, "An empirical comparison of pruning methods for decision tree induction", *Machine Learning*, vol. 4, no. 2, pp. 227-243, Nov. 1989.

[26] T. K. Ho, "The random subspace method for constructing decision forests", *IEEE Trans. Pattern Analysis and Ma-*

chine Intelligence, vol. 20, no. 8, pp. 832-844, Aug. 1998.

[27] Y. Freund, R. E. Schapire. "Experiments with a new boosting algorithm", In *ICML*, vol. 96, pp. 148-156, 1996.

[28] T. K. Ho, "Random decision forests", In *Proceedings of the Third International Conference on Document Analysis and Recognition*, vol. 1, pp. 278-282, Aug. 1995.

[29] L. Breiman, "Bagging predictors", *Machine Learning*, vol. 24, no. 2, pp. 123-140, Aug. 1996.

[30] L. Breiman, "Random forests", *Machine Learning*, vol. 45, no. 1, pp. 5-32, Oct. 2001.

[31] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees", *Machine Learning*, vol. 64, no. 1, pp. 3-42, Apr. 2006.

[32] J. J. Rodriguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: A new classifier ensemble method", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1619-1630, Oct. 2006.

[33] K. Crammer, Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines", *Journal of Machine Learning Research*, vol. 2, pp. 265-292, 2001.

[34] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 5, pp. 854-869, May. 2007.

[35] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, Jun. 2008.

[36] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, and R. Moore, "Real-time human pose recognition in parts from single depth images", *Communications of the ACM*, vol. 56, no. 1, pp. 116-124, Jan. 2013.

[37] R. Caruana, N. Karampatziakis, and A. Yessenalina, "An dmpirical evaluation of supervised learning in high dimensions", In *Proceedings of the 25th International Conference on Machine Learning*, pp. 96-103, Jul. 2008.

[38] L. Mitchell, T. M. Sloan, M. Mewissen, P. Ghazal, T. Forster, M. Piotrowski, and A. S. Trew, "A parallel random forest classifier for R", In *Proceedings of the Second International Workshop on Emerging Computational Methods for the Life Sciences*, pp. 1-6, 2011.

≡ 필자소개 ≡

최 대 웅

2013년 8월: 고려대학교 컴퓨터정보학과 (이학사)
2016년 2월: 고려대학교 컴퓨터정보학과 (이학석사)
2016년 3월～현재: 고려대학교 컴퓨터정보학과 (공학박사)
[주 관심분야] Human Computer Interaction, Text Entry

조 현 중

1996년 2월: 경북대학교 전자공학과 (공학사)
1998년 2월: 포항공과대학교 전자전기공학과 (공학석사)
2006년 9월: 버지니아공과대학교 컴퓨터공학과 (공학박사)
2006년 10월～2009년 2월: ETRI 수석 연구원
2009년 3월～현재: 고려대학교 컴퓨터정보학과 부교수
[주 관심분야] Embedded Real-Time Systems, Real-Time Scheduling, Synchronization for Single/Multiprocessors, Wireless Sensor Networks, Human-Computer Interaction