논문 2021-16-03

Hand-crafted 특징 및 머신 러닝 기반의 은하 이미지 분류 기법 개발

(Development of Galaxy Image Classification Based on Hand-crafted Features and Machine Learning)

오 윤 주, 정 희 철*
(Yoonju Oh, Heechul Jung)

Abstract : In this paper, we develop a galaxy image classification method based on hand-crafted features and machine learning techniques. Additionally, we provide an empirical analysis to reveal which combination of the techniques is effective for galaxy image classification. To achieve this, we developed a framework which consists of four modules such as preprocessing, feature extraction, feature post-processing, and classification. Finally, we found that the best technique for galaxy image classification is a method to use a median filter, ORB vector features and a voting classifier based on RBF SVM, random forest and logistic regression. The final method is efficient so we believe that it is applicable to embedded environments.

Keywords : Galaxy image classification, Machine learning, Hand-crafted feature, Image processing, Support vector machine, Logistic regression, Random forest, ORB feature

Ⅰ. 서 론

고가의 천체 망원경 장비의 가격이 낮아지면서 일반인들도 손쉽게 장비를 구매하여 은하, 행성 등의 관측을 하고 있다. 최근에는 스마트 망원경 (smart telescope)이 출시가 되었고, 망원경이 태블릿이나 스마트폰과 연결되어 은하 혹은 행성 이미지를 바로 전송받아 실시간으로 확인할 수 있게 되었다 [1]. 따라서, 모바일 장치에서 은하 분류를 해주거나, 더 나아가 직접 수집한 은하 이미지를 통해 학습을 할 수 있는 서비스가 가능해지게 되었다. 일반적으로 서버로 이미지를 전송해서 학습할 수도 있지만 이는 외부로 유출될 가능성이 있으므로 사생활이 침해될 수 있다. 또한 이미지가 네트워크망을 통해 전송이 되어야하기 때문에 전송 속도가 느린 경우 지연이 생기거나 외부 통신이 차단된 경우에는 서버로 전송하여 처리하는 것이 불가능하다. 본 연구에서는 이러한 서비스를 위한 hand-crafted 특징 및 머신러닝 기반의 은하 이미지 분류 기법을 개발하고자 한다. 그림 1은 우리가 목표로 하고 있는 총 10가지 클래스의 은하 이미지 종류를 보여준다. 그림 1에서 (h), (i), (j)는 나선은하 (spiral)의 형태를 보여주고 있다. 나선은하는 중심에서 원반으로 뻗은 나선구조를 가지고 있으며, 나선팔 (spiral arm)의 형태에 따라 loose, medium, tight로 명명한다. (e), (f)는 은하팽대부 (bulge)에 따라 구분이 되는데, 은하팽대부는 은하 중앙에 존재하는 별들의 군집을 의미한다. 그 모양에 따라 rounded, boxy로 구분할 수 있다. (a), (e), (f), (g), (h), (i), (j)는 원반은하 (disk)로서 납작하고 둥근 원반형의 모양을 띠고 있는 은하이다.

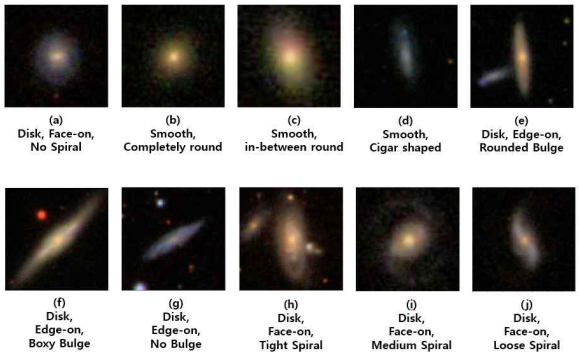

그림 1. 10개 클래스의 은하 이미지
Fig. 1. Galaxy images of ten classes

이러한 은하 이미지 분류의 경우 과거 사람이 디자인한 특징 (hand-crafted feature) 기반의 기술부터 최근의 딥러닝 (deep learning)을 통한 기법들까지 제안되어왔고, 최근 많은 발전을 이루어 왔다 [2-5]. 이러한 딥러닝 알고리즘의 경우 GPU와 같은 고사양의 하드웨어를 요구하기 때문에 높은 비용을 요구한다. 또한, hand-crafted 기법 및 머신러닝 기법은 전처리, 특징추출, 특징가공 모두 도메인 지식 (domain knowledge) 기반으로 수행하며, 이들은 물리적인 의미를 포함하고 있어 딥러닝에 비해 결과의 해석이 용이하다.

*Corresponding Author (heechul@knu.ac.kr)
Received: Nov. 30, 2020, Revised: Jan. 7, 2021, Accepted: Jan. 29, 2021.
Y.J. Oh: Kyungpook National University (M.S.)
H.C. Jung: Kyungpook National University (Assoc. Prof.)
※ 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1C1C1007423).

ⓒ IEMEK J. Embed. Sys. Appl. 2021 Feb. 16(1) 17-27
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2021.16.1.17

본 연구에서는 여러 hand-crafted 기법 및 머신러닝 기법을 이용하여 은하 이미지 분류에 대한 최적의 방법을 도출하였다. 먼저, 은하 이미지 프레임워크을 개발하여 베이스라인 기술들을 각 모듈별로 설정하고, 각 조합들의 인식률과 학습 속도를 측정하였다. 이를 기반으로 모델 탐색을 하여 최적의 모델을 얻을 수 있었다.

우리가 사용한 학습데이터는 총 25,753장인데, 빠른 모델 탐색을 위해 900장 이미지의 서브 샘플만 이용하여 탐색을 수행하였다. 결과적으로 단 900장의 서브 샘플로 탐색한 결과가 전체 학습데이터를 이용했을 때도 대체적으로 일관된 결과를 보여 계산시간을 단축하는 효과가 있었다.

이러한 최적화된 모델뿐만 아니라, 본 연구에서는 다양한 실험 결과를 제공하는데, 이를 통해 추후 연구자들이 새로운 hand-crafted 기반의 은하 이미지 분류 기술을 개발할 때 뿐만 아니라 딥러닝 기반의 은하 이미지 분류 기술을 개발할 경우에도 좋은 도메인 지식을 제공할 수 있을 것이다.

본 논문은 다음과 같이 구성된다. II장에서 기존 은하 분류에 관한 연구를 분석하고, III장에서 우리가 개발한 은하 이미지 분류 프레임워크에 대해서 소개한다. IV장에서는 모델 탐색 방법론을 제안하여 최적의 모델을 탐색하고, V장에서는 실험 환경 및 결과에 대해서 언급한다. 마지막으로 VI장에서 모델 성능에 대한 토의를 하고, VII장에서 본 연구에 대한 결론을 맺는다.

Ⅱ. 기존 연구

일반적으로, 은하 이미지 분류 기술은 크게 두 가지 카테고리로 구분할 수 있다. 먼저, 전형적인 hand-crafted 특징 및 머신 러닝 기법 등을 활용하여 은하 이미지 분류를 수행할 수 있고, 최근에는 딥러닝 기법을 통해 은하 이미지 분류를 하려는 시도들이 있다. 이 두 가지 접근 방식은 속도 및 인식률 측면에서 각기 다른 장단점을 갖는데, hand-crafted 특징 및 머신 러닝 기반의 기법은 일반적으로 학습 및 추론 속도가 빠르며, 딥러닝의 경우에는 고사양의 하드웨어를 요구한다. 일반적으로 인식률 측면에서는 딥러닝 알고리즘이 우위에 있지만, 사용하려는 시스템의 사양 및 상황을 고려하여 알고리즘 선택을 하여야 한다 [6, 7].

1. Hand-crafted 특징 및 머신러닝 기반의 은하 이미지 분류

2004년에는 국소 가중 회귀 (locally weighted regression)와 배깅 (bagging) 으로 앙상블한 기법이 제안되었다 [8]. 은하 이미지는 데이터 증강 (data augmentation)을 위해 회전하고 중앙에 배치하고 자르도록 하였고, PCA (principal component analysis)을 사용하여 데이터의 차원을 줄이고 이미지에서 관련 정보를 추출했다. 2007년에 제안된 기법은 SVM (Support Vector Machines), 랜덤 포레스트, Naïve Bayes의 세가지 기계학습 기반의 알고리즘이었다 [9]. 2016년에는 NMF (non-negative matrix factorization) 기법을 사용한 기법이 제안되었으며, 이 또한 총 3개의 은하 종류에 대해 인식을 수행하였고, 92.76%의 인식률을 달성하였다 [6]. 기존의 hand-crafted 특징 및 머신러닝 기반의 연구와 본 연구의 차이는 인식 가능한 카테고리의 수에 있다. 기존에는 3종류의 은하에 대해 인식이 가능했지만, 우리는 10개의 은하를 인식하는 것을 목표로 한다.

2. 딥러닝 기반의 은하 이미지 분류

최근에는 은하 이미지 분류에 딥러닝 기술도 활용되고 있다. 2016년에 제안된 기술은 SDSS (Sloan Digital Sky Survey) 및 CFHTLenS (Canada-France-Hawaii Telescope Lensing Survey)의 데이터를 사용하였고, 심층 컨볼루션 신경망 (Deep Convolutional Neural Network) 모델을 사용하는 별-은하 분류 프레임워크를 제시한다 [10]. 그 이후에 더 깊은 8개의 레이어를 가진 심층 컨볼루션 신경망 아키텍처가 제안되었다 [11]. 최근에는 더욱 깊은 레이어를 갖는 모델들을 활용한 은하 분류 기법들이 제안되었으며, 90% 이상의 고성능을 보이고 있다 [12, 7].

Ⅲ. 은하 이미지 분류 프레임워크

본 연구에서는 각 모듈간의 조합 및 분석 실험을 위해 은하 이미지 분류를 위한 프레임워크을 개발하였다. 프레임워크은 전처리 모듈, 특징 추출 모듈, 특징 가공, 분류 모듈 네 가지로 구성되어있으며, 이는 표 1에 나와있다. 표 1의 전처리 모듈과 특징 가공 모듈에는 기능을 수행하지 않는 none 요소가 포함되어있다.

1. 전처리 모듈

특징 추출 알고리즘을 수행하기 전에 데이터 전처리 (data preprocessing)를 통해서 특징 추출을 더 잘 할 수 있도록 해준다. 이미지를 블러 (blur) 해주거나 엣지 검출

표 1. 은하 이미지 분류 프레임워크
Table 1. Framework of galaxy image classification

Preprocessin Module (7)	Feature Extration Module (2)	Feature Post-processing Module (7)	Classification Module (7)
Median Filter	ORB (scalar)	Standard Scaler	Random Forest
Gaussian filter	ORB (vector)	Min Max Scaler	Logistic Regression
Histogram Equalization		Robust Scaler	RBF SVM
CLAHE		Normalizer	Voting Classifier
Edge Detection (Sobel x)		PCA	Adaboost
Edge Detection (Sobel y)		Outlier	Extremely Randomized Trees
None		None	K-nearest neighbor

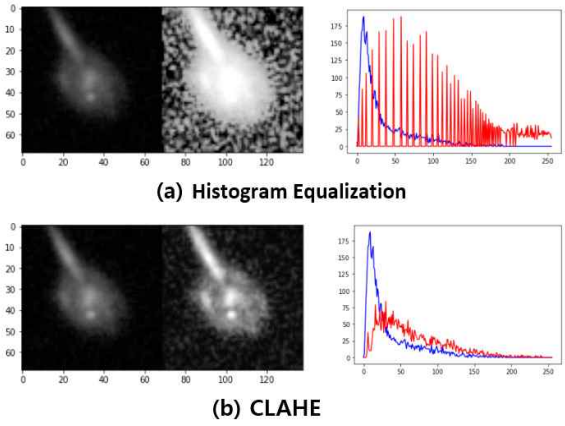

그림 2. 히스토그램 평활화 및 CLAHE
Fig. 2. Histogram equalization and CLAHE

(edge detection)을 해주는 등의 이미지 처리 (image processing)를 할 수 있다. 또한, 히스토그램 (histogram)을 이용하여 이미지의 명암을 조절해주는 표현 방식도 있다.

1.1 미디언 필터 (median filter, M/F)

미디언 필터는 주어진 마스크 (mask) 영역의 값들을 크기 순서대로 정렬한 후 중앙값을 선택하는 필터이다 [13]. 미디언 필터를 사용하면 가장 큰 값과 가장 작은 값들은 제외시켜준다. 그래서 미디언 필터는 이미지를 정확한 값으로 표현하지 못해서 블러 될 수 있지만 임의의 흑백 픽셀을 가진 소금-후추 노이즈 (salt and pepper noise)를 줄이는 필터로서 효과적이다.

은하 이미지의 주변 별들이 소금-후추 노이즈 역할을 할 것이라고 예상하고 이를 제거해주기위해 7×7 마스크로 된 미디언 필터를 사용해보았다.

1.2 가우시안 필터 (Gaussian filter, G/F)

일반적으로 노이즈를 제거할 때 사용하는 필터 중에 대표적인 필터는 가우시안 필터이다 [13]. 미디언 필터와 동일하게 가우시안 노이즈로서 주변 별들을 생각해 볼 수 있었고, 이들을 제거하기위해 7×7 마스크로 된 가우시안 필터를 사용해보았다.

1.3 히스토그램 평활화 (histogram equalization, HE)

히스토그램 평활화는 명암 값의 분포가 한쪽으로 치우치거나 균일하지 못한 이미지를 일정한 분포를 가진 히스토그램으로 생성하는 것이다 [13]. 따라서 평활화를 수행한 히스토그램은 보다 균일한 분포를 가지게 되어, 한 곳에 집중되어 있는 명암 값을 펼쳐서 고르게 분포하도록 하는 역할을 한다.

대비 제한 적응 히스토그램 평활화 (CLAHE, Contrast-Limited Adaptive Histogram Equalization)는 히스토그램 평활화의 한계를 극복하는 방법으로써 contrast limit라는 값을 적용하여 한 영역이 이 값을 넘어가는 경우에는 다른 영역에 균일하게 배분되도록 한다 [14].

표 2. 수직(x) 마스크
Table 2. Vertical mask

-1	0	1
-2	0	2
-1	0	1

표 3. 수평(y) 마스크
Table 3. Horizontal mask

1	2	1
0	0	0
-1	-2	-1

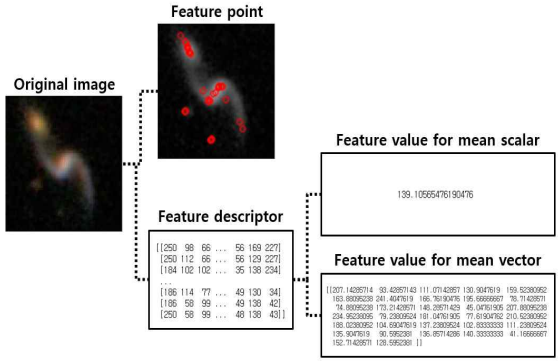

그림 3. ORB 특징 추출
Fig. 3. ORB feature extraction

대부분 은하를 촬영한 이미지는 전체적으로 어둡다. 그래서 우리는 히스토그램 평활화 및 CLAHE를 적용했다. 히스토그램 평활화를 하면 약간 밝은 부분도 명도가 극대화되어서 특징추출에 큰 도움이 될 것이라고 추측하였다.

그림 2는 각 기법을 적용한 결과를 보여준다. 시각적으로 봤을 때 그림 2. (a) 보다 (b) 가 더 좋아보이며, (a)에서 전체적인 부분에 균일화를 적용하여서 원본 이미지보다 더 넓은 영역이 밝게 표현되어졌기 때문에 고유한 특성이 오히려 사라졌다고 볼 수 있다.

1.4 소벨 엣지 검출 (Sobel edge detection)

소벨 엣지 검출은 1차 미분을 기반으로 한 대표적인 엣지 검출 방법이다 [15]. 은하의 경우 전체적인 모양 (shape)에 대한 정보가 은하 인식을 하는데 도움을 줄 것으로 추측된다. 이러한 모양 정보를 추출하기 위해서는 은하의 바운더리 부분을 검출할 수 있는 알고리즘이 필요하다. 표 2, 3에 나타나있는 소벨 엣지 검출 알고리즘이 이러한 역할을 해줄 수 있을 것이라고 생각하여, 전처리 모듈에 추가하였다.

2. 특징 추출 모듈

우리는 ORB [16] 특징을 이용하여 은하 이미지 분류를 수행하였다. ORB 특징은 이미지에서 FAST [17]를 이용하여 특징을 검출 (keypoint detection) 하는 부분과 BRIEF [18]를 이용하여 특징 기술자 (feature descriptor)를 계산하는 두 단계로 구성되어 있다. 그림 3과 같이 이미지에 따라 검출되는 특징의 개수는 매번 변할 수 있으며 (variable

length), 다음과 같은 방법을 이용하여 이를 고정된 길이(fixed length)를 갖는 특징으로 변환을 수행한다.

2.1 스칼라 특징 추출

우리는 스칼라 특징으로 하나의 이미지에서 추출된 특징들의 평균을 이용한다. 특징 추출 모듈에 이미지 한 장이 입력되면 N개의 ORB 특징이 추출된다. 앞에서 언급한 것과 같이 N은 이미지 마다 다를 수 있다.

스칼라 특징 추출은 아래 수식과 같다. f는 ORB 특징이고, N개의 특징점과 d차원 (e.g. $d=32$) 으로 구성되어있다. $f_{i,j}$는 i번째 특징 벡터 f_i의 j번째 요소 값을 의미한다.

$$f_s = \frac{1}{N}\frac{1}{d}\sum_{i=1}^{N}\sum_{j=1}^{d} f_{i,j}. \quad (1)$$

즉, f_s는 하나의 이미지를 대표할 수 있는 특징으로 이용될 수 있고, 우리는 이를 스칼라 특징이라 정의한다. 위의 수식을 통해 이미지 한 장을 하나의 차원으로 매핑하는 것이 가능해진다.

2.2 벡터 특징 추출

스칼라 기반의 특징 차원이 낮아 계산에는 효율적이나 과도하게 원래 정보를 잃어버릴 수 있으므로, 우리는 다음의 방법도 사용한다. 이미지 한 장에는 특징 기술자의 정보를 담은 벡터들이 특징점 개수만큼 존재한다. 이 특징들의 차원은 살리면서 평균을 구해주면 위의 스칼라 특징 추출 방법보다 특징 정보가 더 보존될 수 있게 될 것이라고 추측하여 아래와 같이 평균 특징 벡터를 구현하였다.

이미지 한 장에 대한 벡터 특징 f_v는 다음과 같이 정의된다.

$$f_v = \frac{1}{N}\sum_{i=1}^{N} f_i. \quad (2)$$

즉, f_v는 이미지 한 장에서 추출된 모든 특징 벡터 f_i의 평균값이다. 스칼라 특징과는 다르게 원래 특징 벡터의 차원을 유지한다.

3. 특징 가공 모듈

특징 가공 모듈에서는 특징 추출된 이미지들의 값을 정규화하거나 차원 축소 (dimension reduction), 혹은 이상치 (outlier) 데이터를 제거하는 등의 기능을 수행한다.

3.1 피처 스케일링 (feature scaling : STD scaler, min max scaler, robust scaler, normalizer)

scaler는 모든 데이터에 선형 변환을 적용하여 전체 데이터의 분포를 일정 범위로 모으는 역할을 한다. 우리는 scikit-learn에서 제공하고 있는 4가지 종류의 scaler를 사용하였으며, 각 scaler에 대한 설명은 다음 표 4와 같다 [19].

표 4. 피처 스케일링의 종류
Table 4.Type of feature scaling

type	function
standard scaler	평균을 0, 표준편차를 1이 되도록 변환. 표준화를 통해 변환될 피처는 원래 값에서 피처의 평균을 뺀 값을 피처의 표준 편차로 나눈 값으로 계산.
min max scaler	최대값을 1, 최소값을 0이 되도록 변환. 새로운 데이터는 원래 값에서 피처 의 최소값을 뺀 값을 피처의 최대값과 최소값의 차이로 나눈 값으로 변환.
robust scaler	중앙값을 0, IQR (interquantile range)을 1이 되도록 변환. 데이터들을 3 quartile에서 1 quartile을 제거한 IQR로 스케일링을 하는 기능을 수행.
normalizer	유클리드 거리가 1이 되도록 데이터를 조정. 새로운 데이터는 원래값에서 세 개의 피처의 크기를 합한 값으로 나눠줌.

3.2 PCA (principal component analysis)

우리는 차원 축소를 위해 분포된 데이터의 주성분을 찾는 방법인 PCA를 사용한다 [20]. 본 연구에서 사용한 은하 이미지는 69×69 차원이며, 이 중 불필요한 차원이 존재할 것이라고 생각하였다. 즉, 은하 부분을 제외한 백그라운드 (background) 부분은 은하 분류에 큰 도움이 되지 않을 것이다. 따라서 차원 축소를 통해 이러한 불필요한 차원을 제거할 수 있을 것으로 생각하고 특징 가공 모듈에 추가하였다.

3.3 이상치

이상치는 데이터 상의 다른 값들의 분포와 비교했을 때 비정상적으로 떨어져있는 관측치이다. 사람이 어노테이션 (annotation) 작업을 하다보면 잘못해서 레이블이 다르게 분류될 수도 있기 때문에 주어진 데이터에 이상치가 있을 거라고 생각하였다. 그래서 이상치를 제거하기위해 다양한 방법이 있지만 이 논문에서는 isolation forest를 사용하여 고차원 데이터 셋의 이상치를 탐색하고, 찾아낸 이상치를 데이터 셋에서 제거하는 방식을 사용하여 실험해보았다 [21].

4. 분류기 모듈

특징 가공된 이미지들을 학습시키기위해 분류기를 선택해야한다. 가장 많이 사용하는 분류기부터 성능이 가장 좋다고 알려진 분류기까지 다양하게 사용하여 테스트 해보았다.

4.1 랜덤 포레스트 (random forest, RF)

랜덤 포레스트는 다수의 결정 트리들이 학습하는 앙상블 방법이다 [22]. 학습과 판별을 빠르게 처리하고, 학습 데이터의 노이즈에도 강하다. 하지만 학습 데이터의 개수가 적으면 과적합이 발생하기도 한다.

4.2 로지스틱 회귀 (logistic regression, LR)

로지스틱 회귀는 회귀를 사용하여 데이터가 어떤 범주에

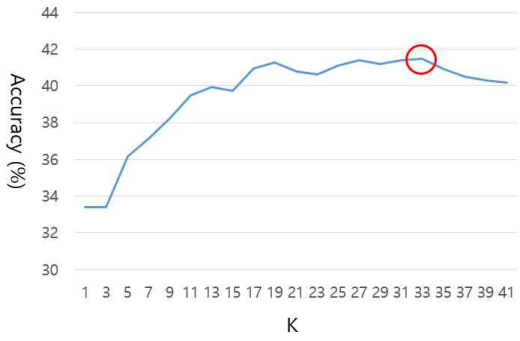

그림 4. K에 따른 성능 비교
Fig. 4. Accuracy comparison according to K

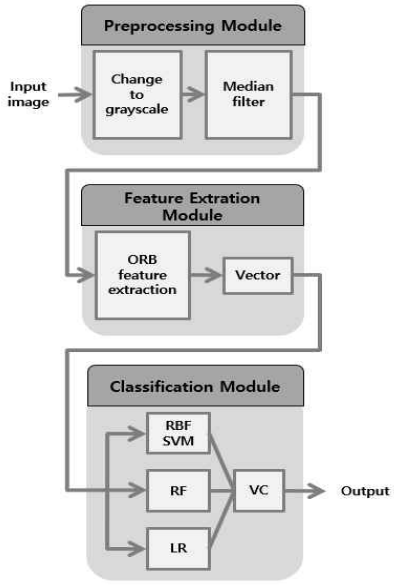

그림 5. 탐색 결과 찾은 알고리즘 순서도
Fig. 5. Best algorithm flowchart

속할 확률을 예측하고 그 확률에 따라 가능성이 더 높은 범주에 속하는 것으로 분류해주는 지도학습 방법이다 [23].

4.3 RBF SVM (RBF support vector machine)

SVM은 마진 (margin)이 최대화할 수 있는 결정 경계 (decision boundary)를 찾는 방법이다 [24]. 머신러닝 알고리즘이지만 딥러닝과 비슷한 성능을 내고 가볍게 이루어져있다. 그 중에서도 가장 많이 사용되는 RBF SVM (radial basis function support vector machine)은 한 분류의 데이터가 다른 분류 데이터들 사이에 섞여있으면 분류하기 어려워지는데 이를 해결하는 방법이다. cost는 얼마나 많은 데이터 샘플이 다른 클래스에 놓이는 것을 허용하는지를 결정하고, gamma는 하나의 데이터 샘플이 영향력을 행사하는 거리를 결정한다. 이때, 두 값이 모두 커질수록 알고리즘 복잡도는 증가하게 되므로 분류가 어려운 데이터에 적합한 알고리즘이다.

4.4 투표 기반 분류기 (voting classifier, VC)

랜덤포레스트, 로지스틱 회귀, RBF SVM를 투표 기반 분류기에 넣고 평균을 내어 확률이 가장 높은 클래스를 예측하는 간접투표 (soft voting)을 하게 된다.

4.5 아다부스트 (adaboost, AB)

결정 트리 분류기 (decision tree classifier)를 사용하여 아다부스팅을 한다. 아다부스팅은 가중치를 부여하는 방식으로 약 분류기 (weak classifier)를 강 분류기 (strong classifier)로 생성하는 방법이다 [25].

4.6 익스트림 랜덤 트리 (extremely randomized trees, ERT)

랜덤 포레스트 분류기는 결정 트리를 개별로 사용하는 결합 방법으로서, 데이터 특징차원을 무작위로 감소시켜 일부만 선택하여 사용한다. 이때 tree를 더 무작위하게 만들기 위해 이 무작위성을 극단적으로 증가시킨 것이 익스트림 랜덤 트리이다 [26]. 익스트림 랜덤 트리가 랜덤 포레스트 분류기보다 더 폭넓은 시각으로 특성들을 평가하기 때문에 성능향상의 원인이 될 수 있을 것이라고 예측했다.

4.7 K-최근접 이웃 (k-nearest neighbor, KNN)

가장 가까운 K개의 데이터를 보고 주어진 데이터를 분류하는 K-최근접 이웃을 사용하여 설계해보았다 [27]. 그림 4와 같이 K가 33일 때 가장 최고의 성능이 나타난 것으로 보아 주어진 데이터 주변의 33개 데이터의 다수결 분류가 가장 정확하다는 것을 알 수 있다.

Ⅳ. 최적 모델 탐색

1. 모델 탐색 방법론

각 모듈의 총 조합 개수는 무수히 많다. 우리는 모든 조합을 실험하지 않고, 다음의 방법론으로 최적의 조합을 찾아낸다.

1. 특징 추출 (2가지)과 분류기 (7가지) 먼저 실험해본다. (2*7=14가지) [표 4, 표 5]
2. 스칼라 특징 추출과 벡터 특징 추출 중에서 성능이 좋은 것을 선택한다. → 벡터 특징 추출 (7가지)
3. 아래 두 과정을 독립적으로 수행한다.
   3.1. 전처리 (7가지)와 벡터 특징 추출 모듈을 학습시킨다. (7*7=49가지) [표 6]
   3.2. 벡터 특징 추출 모듈과 특징 가공 (7가지)을 학습시킨다. (7*7=49가지) [표 7]
4. 3번에서 평가한 것들 중에서 성능이 가장 좋은 것을 선택한다.

2. 선택된 최종 모델

최종 모델은 그림 5와 같이 입력된 이미지를 그레이스케일 (grayscale)로 변경해주고, 미디언 필터를 사용한다. 미디

언 필터는 주변의 별을 소금-후추 노이즈로 간주하여 이를 제거해주는 역할을 한다. 그리고 ORB 특징을 사용하여 벡터 특징 추출을 한다. 이 특징 추출기는 SURF [28]와 SIFT [29]보다 빠르고 SURF 보다 정확하다고 알려져있다. 최종적으로 추출된 특징을 RBF SVM, 랜덤 포레스트, 로지스틱 회귀를 활용한 투표 기반 분류기에 넣고 결과를 도출한다.

V. 실 험

1. 실험 환경

우리의 실험 셋팅은 다음과 같다. 먼저 그림 1과 같이 총 9개의 클래스로 구분된 은하 이미지를 분류하기 위한 이미지 데이터를 사용한 데이터 셋은 Galaxy 10 [30] 이다. Galaxy 10은 10개의 클래스로 분리된 25,753장의 69×69 픽셀을 가진 은하 이미지로 구성되어있다.

제안된 모델 탐색을 위해 우리는 900장의 이미지를 랜덤으로 선택하였으며, 각 클래스의 균일성을 위해 1개의 클래스는 17장의 데이터만을 가지고 있어서 이를 제거하였고, 나머지 9개의 클래스로부터 각 100장의 이미지를 획득하였다. 이 중 90장은 검증 셋 (validation set)으로 사용되며, 나머지 810장은 훈련 셋 (training set)으로 학습하였다. 이러한 과정을 총 100번 반복하여 검증 셋에 대한 인식률의 평균과 표준편차를 구하였다. 표 4에서부터 표 9까지 제시된 시간은 특징 추출과 100번의 학습 및 테스트를 포함한다. mean acc (%)는 100번 학습 (fit)하고 예측 (predict)한 점수 (score)를 평균한 정확도 (accuracy)를 의미한다. ±이후의 값은 표준편차 (standard deviation)로 정확도의 허용 범위를 나타낸다. time(s)는 데이터를 로딩한 시간을 제외하고 특징추출과 분류기로 평가한 시간을 나타내며, 초 단위로 측정하였다.

최종적으로 제안 기법을 통해 찾은 최적 모델의 성능을 평가하기 위해 모든 데이터 셋 (25,753장) 의 90%를 훈련 셋으로 사용하였으며, 나머지 10%는 테스트 셋 (test set)으로 사용하였다.

실험 환경에서 사용된 시스템 스펙은 2-core Intel (R) Xeon (R) CPU@ 2.30GHz이고, RAM은 13G 이다. 머신러닝 알고리즘은 scikit-learn의 버전 0.22.2, 특징 추출 알고리즘은 openCV의 버전 4.1.2.30을 사용하였다.

2. 모델 탐색 실험 결과

2.1 스칼라/벡터 성능 비교 (IV. 1. 2번 과정)

우리는 스칼라/벡터 ORB를 비교하기 위한 실험을 진행하였으며 그 결과는 표 4, 표 5와 같다. 표 5과 같이 스칼라 ORB와 8개의 분류기로 각각 실험해보았을 때, KNN이 21.74%로 가장 인식률이 높았고, 표 6와 같이 벡터 ORB로 평가했을 때는 46%로 투표기반분류기가 가장 높았다.

K-최근접 이웃은 연산이 가장 짧은 시간이 소요되었다. 또한 RBF SVM은 성능도 상위권이면서 테스트 시간이 매우 빠른 것을 확인 할 수 있다. 반대로 아다부스트는 시간이 가장 오래 걸리면서 성능은 가장 낮았다.

표 5. 스칼라 ORB와 분류기별 성능
Table 5. Performance by scalar ORB and classifiers

	mean acc (%)	time (s)
ORB(scalar)+LR	13.73±3.20	2.25
ORB(scalar)+RF	16.06±3.55	41.29
ORB(scalar)+ERT	16.23±3.67	22.90
ORB(scalar)+VC	17.42±3.64	80.96
ORB(scalar)+AB	20.10±4.13	97.65
ORB(scalar)+ RBF SVM	20.14±3.95	9.11
ORB(scalar)+KNN	**21.74±3.61**	**1.56**

표 6. 벡터 ORB와 분류기별 성능
Table 6. Performance by vector ORB and classifiers

	mean acc (%)	time (s)
ORB(vector)+AB	32.39±5.20	105.01
ORB(vector)+KNN	41.49±5.27	**1.96**
ORB(vector)+LR	43.46±4.77	6.71
ORB(vector)+RF	43.56±5.34	42.11
ORB(vector)+ERT	44.24±5.44	22.27
ORB(vector)+ RBF SVM	45.88±4.29	7.46
ORB(vector)+VC	**46.00±5.24**	78.75

2.2 전처리 및 특징 가공 성능 비교 (IV. 1. 3번 과정)

표 7은 전처리 모듈의 변화와 특징 가공 모듈의 변화에 따른 모델 탐색 실험에 대한 결과를 보여준다. 먼저, 히스토그램 평활화와 CLAHE로 밝은 이미지에서 특징점을 구하고, 특징 기술자를 구할 때는 구한 특징점과 히스토그램 평활화 또는 CLAHE로 처리한 이미지를 계산하는 방식을 사용하였다. 이미지를 밝게 하기 때문에 특징점과 특징 기술자를 구할 때 이를 적용하지 않았을 때에 비해 성능이 향상될 것이라고 예상했다. 하지만 히스토그램 평활화와 CLAHE 모두가 특징 벡터에서 오히려 성능이 낮게 나왔다. 또한 CLAHE가 히스토그램 평활화보다 성능이 더 많이 감소되었다. 큰 별과 작은 별들에서 어두운 부분의 밝기가 밝아지면서 이들이 크게 확대되어져서 특징 추출에 영향을 준 것으로 추측해볼 수 있다.

미디언 필터와 가우시안 필터를 사용하였을 때 이미지에 블러 현상이 발생하여 특징추출을 하면 더 성능이 낮아질 것으로 예측하였다. 그러나 실험결과는 반대로 특징을 더 잘 찾을 수 있게 되었다. 미디언 필터는 이미지의 소금-후추 노이즈를 제거하는데 효과적이다. 그리고 가우시안 필터는 이미지의 가우스 노이즈를 제거하는데 효과적이다. 은하 이미지에 나오는 큰 별 이외의 주변의 작은 별들은 이런 노이즈와 같은 기능을 한다. 그래서 가우시안 필터와 미디언 필터를 사용했을 때 이 노이즈들을 제거해주는 역할을 한 것이다. 가우시안 필터가 미디언 필터보다는 성능이 낮다. 이것은 별들의 분포가 소금-후추 노이즈에 더 가깝다는 것을 알 수 있다.

표 7. 모델 탐색한 실험 결과 (900장 이용, 파란색 글자는 최적 모델을 의미)
Table 7. Experimental results of the model search

preprocessing module	HE		CLAHE		G/F		M/F		Sobel x		Sobel y	
unit / classifier	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB(vector) +AB	25.32 ±4.58	106.68	21.48 ±4.24	107.73	32.90 ±4.75	101.92	31.17 ±4.33	101.14	29.40 ±4.79	103.04	28.16 ±4.61	103.21
ORB(vector) +KNN	26.99 ±4.08	**3.24**	22.84 ±4.38	**3.81**	42.84 ±5.05	**2.05**	43.87 ±5.15	**1.99**	34.72 ±4.93	**2.32**	32.74 ±4.26	**2.32**
ORB(vector) +LR	28.92 ±3.71	7.69	**27.90 ±4.05**	7.75	44.71 ±5.30	7.32	45.79 ±4.59	7.71	35.61 ±4.84	8.57	35.82 ±4.75	8.44
ORB(vector) +RF	29.47 ±4.72	49.06	23.62 ±4.69	45.76	45.53 ±5.10	41.03	47.44 ±4.04	40.69	38.67 ±5.21	42.35	35.53 ±4.66	42.00
ORB(vector) +ERT	28.42 ±4.34	24.35	23.49 ±4.38	25.12	45.70 ±4.16	22.26	47.23 ±4.98	22.03	38.28 ±4.93	23.99	34.74 ±4.79	23.99
ORB(vector) +RBF SVM	**30.84 ±4.62**	9.53	27.14 ±4.72	11.04	**48.22 ±5.50**	7.75	47.58 ±4.74	7.40	39.42 ±5.02	7.08	37.32 ±5.31	7.29
ORB(vector) +VC	29.91 ±4.10	84.75	27.84 ±4.31	89.82	47.76 ±4.91	78.73	**50.42 ±4.68**	77.30	**39.47 ±4.64**	76.03	**38.26 ±4.85**	76.91
post-processing module	STD scaler		Min Max Scaler		Robust Scaler		Normalizer		PCA		outlier	
unit / classifier	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB(vector) +AB	32.39 ±5.20	104.94	32.39 ±5.20	104.62	32.39 ±5.2	102.88	29.82 ±4.71	106.80	21.77 ±7.29	52.37	32.43 ±4.98	126.45
ORB(vector) +KNN	40.16 ±4.60	**2.02**	40.31 ±4.44	**2.00**	40.51 ±4.71	**2.61**	39.13 ±4.96	**1.91**	26.22 ±8.88	**2.27**	41.54 ±5.29	**21.21**
ORB(vector) +LR	43.03 ±4.84	4.99	44.00 ±4.68	3.54	43.23 ±4.97	7.41	30.87 ±5.76	2.47	24.88 ±8.96	4.08	43.23 ±4.62	24.64
ORB(vector) +RF	43.38 ±5.11	42.28	43.69 ±5.09	45.48	43.71 ±5.40	41.54	41.57 ±5.58	42.56	24.29 ±7.81	29.64	43.81 ±4.98	60.28
ORB(vector) +ERT	43.83 ±5.25	22.48	43.40 ±5.15	22.39	43.64 ±5.34	24.07	**42.23 ±4.98**	22.55	23.51 ±7.86	24.08	42.97 ±5.05	40.55
ORB(vector) +RBF SVM	19.07 ±4.06	10.85	8.26 ±4.88	10.99	12.26 ±4.58	10.51	8.26 ±5.02	10.98	**26.79 ±9.17**	6.74	**45.87 ±4.35**	25.48
ORB(vector) +VC	**44.66 ±5.15**	92.74	**45.96 ±5.10**	91.49	**45.06 ±5.25**	87.73	41.88 ±5.17	90.73	26.19 ±8.93	47.38	45.61 ±5.03	96.82

소벨 엣지 검출은 노이즈까지 엣지로 검출할 정도로 밝기에 민감하기 때문에 실험 결과로 베이스라인에 비해 성능이 하락하였다. 엣지를 검출하면 ORB가 특징을 잘 추출할 수 있을 것이라고 예상한 것과 달리 오히려 방해를 하고 있다는 것을 알 수 있다.

표 6에서 가장 좋은 성능을 보인 투표기반 분류기, 벡터 ORB 특징값 저장 알고리즘과 같이 학습시켜보았을 때, 가장 우수했던 scaler는 min max scaler였다. standard scaler는 44.66%, min max scaler는 45.96%, robust scaler는 45.06%, normalizer는 41.88%가 나왔다. 최대값을 1, 최소값을 0으로 하여 그 사이에 데이터를 분포하게 하는 min max scaler는 이상치가 있으면 평균과 분산에 영향을 미쳐서 변환된 값이 매우 좁은 범위로 압축될 수 있다. 하지만, 현재 데이터 셋은 이상치가 없으므로 가장 이상적인 scaler의 방법이라고 할 수 있다. 주목할 만한 점은 RBF SVM이 standard scaler, min max scaler, robust scaler 또는 normalizer를 사용했을 때 매우 많이 성능이 낮아진 것을 볼 수 있다.

고차원의 데이터가 넓은 공간에 있어서 데이터 양이 희박하기 때문에 차원을 축소하면 성능이 올라갈 것이라고 예상하고, 먼저 PCA를 사용해보았다. 성능이 PCA를 사용하지 않았을 때에 비해 낮아졌다. 차원을 축소하면서 정보 손실이 발생한 것으로 추측해볼 수 있다.

Isolation forest를 사용하여 고차원의 데이터 셋의 이상치를 찾아보았다. 이상치는 3~5개 사이로 제거가 되었다. 성능은 이상치를 제거하기 전에 비해 낮게 나왔다. 이상치 제거가 오히려 학습할 수 있는 데이터 수를 줄여서 성능이 낮아진 것을 알 수 있다. 따라서 모든 데이터는 정상치에 근접한 데이터 셋이라는 것을 알 수 있다.

최고 성능을 보인 모듈은 미디언 필터를 적용한 이미지에 벡터 ORB로 특징 추출한 다음 투표 기반 분류기에서 학습시킨 것이다. 50.42%의 정확도와 77.30초의 시간이 소요되었

표 8. 모델 탐색한 실험 결과 (25,753장 이용, 파란색 글자는 최적 모델을 의미)
Table 8. Experimental results of the model search

preprocessing module	HE		CLAHE		G/F		M/F		Sobel x		Sobel y	
unit / classifier	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB(vector) +AB	36.97 ±1.18	2388.02	39.02 ±0.97	2779.90	53.93 ±1.74	1879.40	53.66 ±1.96	1820.26	35.17 ±1.54	2484.14	35.29 ±1.95	2516.21
ORB(vector) +KNN	44.24 ±1.11	**231.93**	39.95 ±0.94	346.35	63.19 ±0.87	**338.84**	64.12 ±0.88	**311.85**	46.15 ±1.01	552.51	45.87 ±1.01	412.70
ORB(vector) +LR	47.75 ±0.95	313.99	45.76 ±1.02	**271.89**	62.25 ±0.88	402.99	63.59 ±0.97	419.20	47.36 ±0.99	**317.37**	47.56 ±0.99	**262.24**
ORB(vector) +RF	48.83 ±0.87	1373.54	45.31 ±0.97	1376.34	66.50 ±0.84	1169.68	67.57 ±0.75	1163.49	49.83 ±1.07	1366.59	49.70 ±1.08	1348.78
ORB(vector) +ERT	48.61 ±0.94	442.69	44.36 ±1.06	460.11	66.22 ±0.88	382.65	67.43 ±0.71	381.05	49.51 ±1.06	457.07	49.41 ±1.05	414.98
ORB(vector) +RBF SVM	48.92 ±0.94	4335.75	46.35 ±1.03	4625.86	66.56 ±0.86	3060.32	67.03 ±0.99	2972.62	49.69 ±1.07	3862.75	49.30 ±0.97	3850.77
ORB(vector) +VC	**49.72 ±0.93**	22097.48	**47.64 ±1.03**	23668.40	**67.00 ±0.79**	14252.05	**67.75 ±0.88**	14254.67	**50.72 ±1.06**	20243.79	**50.39 ±1.04**	19633.71
post-processing module	**STD scaler**		**Min Max Scaler**		**Robust Scaler**		**Normalizer**		**PCA**		**outlier**	
unit / classifier	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB(vector) +AB	49.70 ±4.12	1887.02	49.70 ±4.12	2178.36	49.70 ±4.12	2120.88	50.26 ±3.16	2569.62	37.18 ±2.07	570.29	48.65 ±4.52	2103.58
ORB(vector) +KNN	61.88 ±0.98	314.52	62.03 ±0.90	378.51	61.61 ±1.00	381.06	61.48 ±0.95	284.40	52.85 ±1.68	**46.27**	61.53 ±0.92	**364.83**
ORB(vector) +LR	60.82 ±0.95	**207.00**	60.44 ±0.99	**238.33**	60.81 ±0.95	**271.69**	56.03 ±1.05	**167.85**	49.32 ±1.30	74.27	60.70 ±0.94	376.88
ORB(vector) +RF	**65.18 ±0.93**	1123.29	**65.23 ±0.97**	1268.49	**65.17 ±1.03**	1216.44	**64.90 ±1.00**	1300.94	50.65 ±1.86	395.00	65.09 ±1.00	1300.29
ORB(vector) +ERT	64.92 ±0.98	352.38	64.87 ±0.90	389.37	64.84 ±0.95	384.49	64.54 ±0.99	377.26	50.57 ±1.82	246.68	64.87 ±1.02	480.91
ORB(vector) +RBF SVM	58.44 ±1.01	3497.66	40.64 ±1.90	3990.81	56.93 ±1.01	4016.35	31.45 ±1.03	4782.46	**53.45 ±1.49**	2320.79	65.63 ±0.85	2980.31
ORB(vector) +VC	63.85 ±0.97	19963.35	63.83 ±1.02	24617.01	63.76 ±0.98	19950.82	62.37 ±1.12	23213.97	53.32 ±1.60	11302.18	**66.19 ±0.91**	14856.65

고 이를 통해서 미디언 필터의 중요성을 확인할 수 있었다. 주변의 작은 별들을 제거하므로 인해 특징추출에 불필요한 부분을 미리 걸러낼 수 있다. 모든 데이터 전처리 모듈과 특징 가공 모듈에서 시간이 가장 빠르게 소요된 분류기는 K-최근접 이웃이다. 이는 상위 3번째로 인식률이 좋은 것으로 나타났다.

3. 전체 데이터셋에 적용 실험 결과

모델 탐색으로 찾은 분류에 적합한 모델 조합은 25,753장의 이미지로 재실험하여 표 8의 결과와 같이 최종성능을 찾았다. 최적의 모델 조합인 투표기반 분류기와 미디언필터가 성능이 가장 높은 것을 볼 수 있다. 모델 탐색을 25,753장의 데이터를 이용하여 할 경우 약 90시간이 걸린다. 900장의 데이터를 이용하여 모델 탐색에 걸리는 시간은 1시간 이내이다. 따라서 우리가 제안한 모델 탐색 기법을 사용하면 약 1/90의 시간으로 단축시킬 수 있다.

Ⅵ. 토 의

1. 혼동행렬 (confusion matrix)

혼동행렬은 모델의 성능을 평가하는 지표이다. 그림 6의 혼동행렬은 모델 탐색을 통해 찾은 최적의 모델을 보여주고 있다. 최적의 모델은 미디언 필터를 사용한 전처리 모듈과 ORB벡터 특징 기반의 특징 추출 모듈, 투표기반 분류기 모듈을 사용한 것이다. 레이블 0에서 예측값이 클래스 1과 8로 인식되고 있는 것을 볼 수 있다. 이 부분 때문에 성능이 많이 하락한 것으로 보인다. 그림 7의 (a)의 이미지들은 육안으로 보았을 때 통일성을 찾기가 힘들다. 그래서 학습을 할때 레이블 0은 정확하게 분류하지 못하여 학습이 잘 되지 않는 것으로 추측된다. (a)는 (b) 또는 (c)의 형태와 유사점을 가진 것들이 있다. 그러므로 (a)의 이미지를 테스트하면, 학습 정확도가 낮은 클래스 0보다 클래스 1이나 클래스 8이라는 결과를 가져온다.

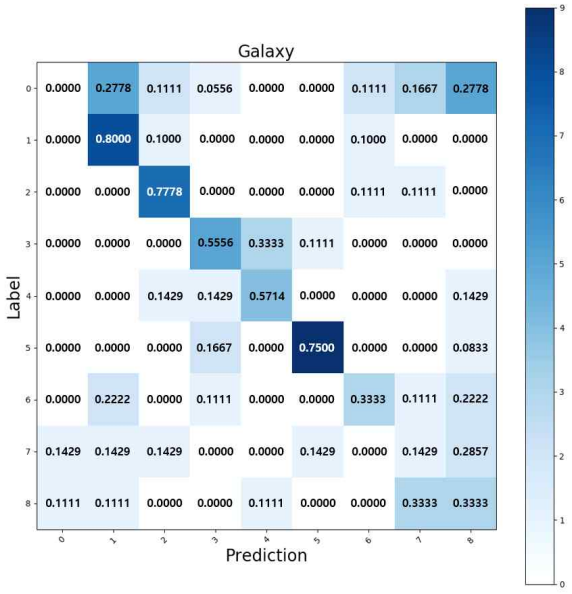

그림 6. 혼동 행렬
Fig. 6. Confusion Matrix

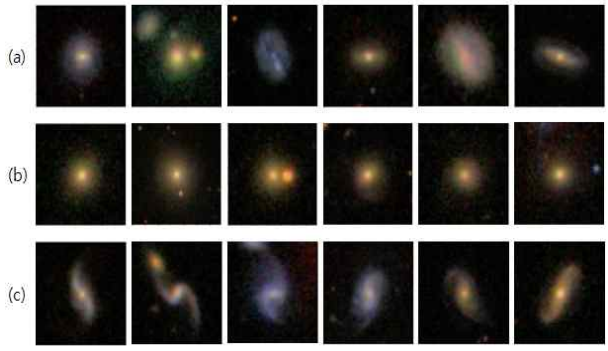

그림 7. (a) 클래스 0, (b) 클래스 1,
(c) 클래스 8의 임의의 이미지 6장
Fig. 7. 6 random images of
(a) class 0, (b) class 1, (c) class 8

2. 미디언 필터 실험

표 9은 미디언 필터를 3×3, 5×5, 7×7 크기의 마스크를 사용하여 실험한 결과이다. 각 필터에서 가장 좋은 성능을 나타내는 분류기는 투표기반 분류기이며, 필터 크기가 커질수록 대체적으로 더 분류를 잘 하는 것을 볼 수 있다. 우리는 가장 높은 성능을 가지는 7×7 미디언 필터를 사용하여 모델 탐색 및 전체 데이터셋 실험을 진행하였다.

3. 가우시안 필터 실험

표 10는 가우시안 필터를 3×3, 5×5, 7×7 크기의 마스크를 사용하여 실험한 결과이다. 필터의 크기 변화에 따라서 실험한 결과는 7×7에서 최고 정확도를 보였다. 그리고 속도 측면에서는 3×3, 5×5, 7×7 모두 비슷한 결과를 보였다. 따라서 우리는 모든 실험에 7×7 필터를 적용하였다.

표 9. 필터 크기에 따른 미디언 필터 성능 비교
Table 9. Performance of median filter

	M/F (3×3)		M/F (5×5)		M/F (7×7)	
	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB (vector) +AB	33.49± 5.64	103.94	32.21± 4.33	102.14	31.17± 4.33	101.14
ORB (vector) +KNN	42.80± 5.08	**1.89**	42.1 ±5.56	**1.94**	43.87 ±5.15	**1.99**
ORB (vector) +LR	45.77± 5.04	6.88	46.30± 5.11	7.35	45.79± 4.59	7.71
ORB (vector) +RF	44.64± 5.38	42.46	45.00± 5.17	41.24	47.44± 4.04	40.69
ORB (vector) +ERT	44.30± 5.27	22.03	45.14± 4.80	22.23	47.23± 4.98	22.03
ORB (vector) +RBF SVM	45.03± 5.44	7.29	47.07± 4.81	7.58	47.58± 4.74	7.40
ORB (vector) +VC	**47.74± 4.82**	77.36	**49.14± 4.87**	78.81	**50.42± 4.68**	77.30

표 10. 필터 크기에 따른 가우시안 필터 성능 비교
Table 10. Performance of Gaussian filter

	G/F (3×3)		G/F (5×5)		G/F (7×7)	
	mean acc (%)	time (s)	mean acc (%)	time (s)	mean acc (%)	time (s)
ORB (vector) +AB	32.20± 4.80	102.43	31.92± 4.62	100.04	32.90± 4.75	101.92
ORB (vector) +KNN	40.82± 4.53	**2.06**	41.16± 5.38	**2.16**	42.84± 5.05	**2.05**
ORB (vector) +LR	43.78± 5.26	7.22	44.49± 5.26	9.73	44.71± 5.30	7.32
ORB (vector) +RF	43.97± 4.73	42.53	45.14± 4.83	39.64	45.53± 5.10	41.03
ORB (vector) +ERT	43.27± 4.89	22.23	44.70± 5.11	23.44	45.70± 4.16	22.26
ORB (vector) +RBF SVM	45.41± 5.00	7.64	46.03± 5.57	6.86	**48.22± 5.50**	7.75
ORB (vector) +VC	**46.46± 5.28**	78.93	**47.19± 5.43**	76.90	47.76± 4.91	78.73

Ⅶ. 결 론

최근 천체 망원경의 보급화로 인해 일반인들의 수요가 증가하면서 모바일 장치와 연동하여 은하 혹은 행성 이미지를 자동으로 분류 및 학습할 수 있는 서비스를 제공받을 수 있게 되었다. 본 연구에서는 직관적으로 생각할 수 있는 hand-crafted 기반의 특징 및 머신러닝 기법을 사용하여 은하 이미지를 분류하였을 때 어떤 조합이 효과적인지에 대한 모델 탐색을 진행하였다. 이를 위해 우리는 4개의 모듈이 포함된 프레임워크을 개발하였고, 서브샘플 900장만을 이용

하여 각 모듈의 조합 중에 최적의 조합을 찾을 수 있는 방법론을 제시하였다. 결과적으로 가장 높은 성능을 나타낸 모듈의 조합은 미디언 필터를 사용한 전처리 모듈과 ORB 벡터 특징 기반의 특징 추출 모듈, 투표기반 분류기 모듈이며, 이는 900장의 학습데이터, 9개의 카테고리를 기반으로 50.42%의 정확도를 보였고, 25,753장 학습 데이터로 확장하여 학습시켰을 때는 10개의 카테고리에 대해 67.75%의 정확도를 달성하였다.

본 연구에서 제시한 접근 방식이나 실험 결과는 추후 다른 연구자의 은하 인식 기술 개발에 도움을 줄 수 있다. 하지만, 여전히 해결하지 못한 이슈가 남아있어 이를 후속연구로 진행할 예정이다. 첫째로, 개발된 hand-crafted 특징 기반의 알고리즘을 스마트 망원경과 같은 임베디드 시스템에 탑재할 계획이다. 또한, 해석이 용이하진 않지만, 절대적인 정확도의 향상을 위해 추후에는 딥러닝 모델을 도입할 예정이다. 딥러닝 모델을 설계할 때, 본 연구에서 밝혀진 사실을 활용할 수 있는데, 예를 들어, 본 연구에서 입증된 미디언 필터의 효과를 합성곱 신경망 (convolutional neural network) 에서도 모사할 수 있도록 필터 일부를 미디언 필터로 교체하는 식의 접근 방법을 도입할 수 있다. 개발된 딥러닝 모델의 처리 시간 절감을 위해 가지치기 기법 (pruning technique)과 같은 모델 경량화 기법 [31]을 활용할 계획이다.

References

[1] G.I. Redfern, "Telescopes," Astrophotography is Easy!, Springer, Cham, pp. 55-76, 2020.

[2] D. Rouan, L. Tasca, G. Soucail, O. Le Fèvre, "A Robust Morphological Classification of High-redshift Galaxies Using Support Vector Machines on Seeing Limited Images-I. Method Description," Astronomy & Astrophysics, Vol. 478, No. 3, pp. 971-980, 2008.

[3] M. Banerji, O. Lahav, C.J. Lintott, F.B. Abdalla, K. Schawinski, S. P. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar, A. Szalay, D. Thomas, J. Vandenberg, "Galaxy Zoo: Reproducing Galaxy Morphologies Via Machine Learning," Monthly Notices of the Royal Astronomical Society, Vol. 406, No. 1, pp. 342-353, 2010.

[4] X.P. Zhu, J.M. Dai, C.J. Bian, Y. Chen, S. Chen, C. Ho, "Galaxy Morphology Classification with Deep Convolutional Neural Networks," Astrophysics and Space Science, Vol. 364, No. 4, pp. 55, 2019.

[5] M. Jiménez, M.T. Torres, R. John, I. Triguero, "Galaxy Image Classification Based on Citizen Science Data: A Comparative Study," IEEE Access, Vol. 8, pp. 47232-47246, 2020.

[6] I. Selim, A.E. Keshk, B.M. El Shourbugy, "Galaxy Image Classification Using Non-negative Matrix Factorization," Int. J. Comput. Appl, Vol. 137, No. 5, pp. 4-8, 2016.

[7] D. Misra, S.N. Mohanty, M. Agarwal, S.K. Gupta, "Convoluted Cosmos: Classifying Galaxy Images Using Deep Learning," Data Management, Analytics and Innovation. Springer, Singapore, pp. 569-579 2020.

[8] J. De La Calleja, O. Fuentes, "Machine Learning and Image Analysis for Morphological Galaxy Classification," Monthly Notices of the Royal Astronomical Society, Vol. 349, No. 1, pp. 87-93, 2004.

[9] S. Kasivajhula, N. Raghavan, H. Shah, "Morphological Galaxy Classification Using Machine Learning," Monthly Notices of the Royal Astronomical Society, Vol. 8, pp. 1-8, 2007.

[10] E.J. Kim, R.J. Brunner, "Star-galaxy Classification Using Deep Convolutional Neural Networks," Monthly Notices of the Royal Astronomical Society, pp. stw2672, 2016.

[11] N.E.M. Khalifa, M.H.N. Taha, A.E. Hassanien, I. M. Selim, "Deep Galaxy: Classification of Galaxies Based on Deep Convolutional Neural Networks," arXiv preprint, Vol. 1709, No. 02245, 2017.

[12] X.P. Zhu, J.M. Dai, C.J. Bian, Y. Chen, S. Chen, C. Hu, "Galaxy Morphology Classification with Deep Convolutional Neural Networks," Astrophysics and Space Science, Vol. 364, No. 4, pp. 55, 2019.

[13] C.R. Gonzalez, R.E. Woods, S.L. Eddins, “Digital Image Processing Using MATLAB,” Pearson Education India, 2004.

[14] K. Zuiderveld, "Contrast Limited Adaptive Histogram Equalization," Graphics gems, pp. 474-485, 1994.

[15] I. Sobel"History and Definition of the Sobel Operator," Retrieved from the World Wide Web, Vol. 1505, 2014.

[16] E. Rublee, V. Rabaud, K. Konolige, "ORB: An Efficient Alternative to SIFT or SURF," 2011 International conference on computer vision. Ieee, 2011.

[17] E. Rosten, T. Drummond, "Machine Learning for High-speed Corner Detection," European conference on computer vision. Springer, Berlin, Heidelberg, 2006.

[18] M. Calonder, V. Lepetit, C. Strecha, P. Fua, "Brief: Binary Robust Independent Elementary Features," European conference on computer vision, Springer, Berlin, Heidelberg, 2010.

[19] F. Pedregosa, G. Varoquaux, A. Gramfor, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, "Machine Learning in Python," JMLR, Vol. 12, pp. 2825-2830, 2011

[20] H. Abdi, L.J. Williams, "Principal Component Analysis," Wiley interdisciplinary reviews: computational statistics, Vol. 2, No. 4, pp. 433-459, 2010.

[21] F.T. Liu, K.M. Ting, Z.H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, IEEE, 2008.
[22] L. Breiman, "Random Forests," Machine learning, Vol. 45, No. 1, pp. 5-32, 2001.
[23] D.R. Cox, "The Regression Analysis of Binary Sequences," Journal of the Royal Statistical Society: Series B (Methodological), Vol. 20, No. 2, pp. 215-232, 1958.
[24] C. Cortes, V. Vapnik, "Support-vector Networks," Machine Learning, Vol. 20, No. 3, pp. 273-297, 1995.
[25] Y. Freund, R.E. Schapire, "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting," Journal of computer and system sciences, Vol. 55, No. 1, pp. 119-139, 1997.
[26] P. Geurts, D. Ernst, L. Wehenkell, "Extremely Randomized Trees," Machine Learning, Vol. 63, No. 1, pp. 3-42, 2006.
[27] N.S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," The American Statistician, Vol. 46, No. 3, pp. 175-185, 1992.
[28] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, "Surf: Speeded up Robust Features," European conference on computer vision, Springer, Berlin, Heidelberg, 2006.
[29] D.G. Lowe, "Distinctive Image Features from Scale-invariant Keypoints," International journal of computer vision, Vol. 60, No. 2, pp. 91-110, 2004.
[30] https://astronn.readthedocs.io/en/latest/galaxy10.html
[31] S. Han, J. Pool, J. Tran, W.J. Dally, "Learning both Weights and Connections for Efficient Neural Network," Advances in neural information processing systems 28, pp. 1135-1143, 2015.

Yoonju Oh (오 윤 주)

2015 Electronic Engineering from Yeungnam University (B.S.)
2020~Artificial Intelligence form Kyungpook National University (M.S.)

Career:
2020 Kyungpook National University, M.S. Student
Field of Interests: Deep Learning & Computer Vision
Email: 0813oyj@knu.ac.kr

Heechul Jung (정 희 철)

2007 Internet from Sejong University (B.S.)
2010 Information and Communication Engineering from the Gwangju Institute of Science and Technology (M.S.)
2018 Electrical Engineering from the Korea Advanced Institute of Science and Technology (Ph.D.)
2019~Artificial Intelligence form Kyungpook National University (Assistant Professor)

Career:
2019 Hyundai Motor Company, Senior Researcher
2019 Kyungpook National University, Assistant Professor
Field of Interests: Deep Learning & Computer Vision
Email: heechul@knu.ac.kr