Cell Detection and Verification Model for Automated Cell Counting (Term-Project) Artificial Intelligence, Fall 2024

김 진[01]

[1]부산대학교 의생명융합공학부

202255644@pusan.ac.kr

요 약

본 프로젝트는 세포 카운팅 작업의 자동화를 목표로 OpenCV와 ResNet18을 결합한 시스템을 설계하였다. OpenCV의 findContour를 통해 세포 후보 영역을 탐지하고, ResNet18 CNN 모델로 실제 세포 여부를 판별하는 방식으로 작동하며, 테스트 결과 **99%의 정확도**를 기록하였다. 데이터 증강 기법을 활용해 모델의 학습 데이터 다양성을 확보하고, 세포 분류의 정확성을 향상시켰다. 본 시스템은 실험 생산성을 높이고 작업의 신뢰성을 강화할 수 있는 가능성을 보여주었다. 다만, 데이터셋 부족, 다양한 환경에서의 검증 미비, OpenCV 탐지 단계의 한계는 향후 개선이 필요한 부분으로 남아 있다. 이 시스템은 의생명공학 실험 및 연구에서 실질적인 효율성을 제공할 수 있는 유용한 도구로 활용될 가능성이 있다.

1. 서론(Introduction)

1.1 연구 배경 및 필요성

부산대학교 의생명융합공학부의 전공 필수 과목인 의생명공학실험에서는 실험 과정 중 세포의 총 개수를 정확히 계산하는 작업이 빈번히 요구된다. 이를 위해 일반적으로 헤모사이토미터에 세포를 로딩한 후, 연구자가 세포를 일일이 눈으로 세어 기록하는 방식을 사용하고 있다. 그러나 이 과정은 수작업으로 이루어지기 때문에 시간과 노력이 많이 들며, 실험자에 따라 결과의 일관성이 떨어질 가능성이 있다. 또한, 세포 수가 많거나 반복적인 작업이 요구되는 상황에서는 실수로 인한 부정확성이 발생할 우려가 크다. 이러한 문제는 실험의 효율성을 저하시킬 뿐만 아니라 연구 결과의 신뢰성에도 부정적인 영향을 미칠 수 있다. 따라서 세포를 자동으로 탐지하고 개수를 계산할 수 있는 시스템을 개발하는 것은 연구자들에게 시간 절약과 높은 정확도를 제공하여 실험 생산성을 크게 향상시킬 수 있는 중요한 방법이다.

1.2 문제 정의 및 연구 목표

현재 OpenCV 기반의 객체 탐지 모델은 세포 탐지 작업에 널리 활용될 수 있는 방법 중 하나로 알려져 있다. 이러한 규칙 기반 모델은 특정 패턴을 인식하여 세포를 탐지하지만, 세포와 유사한 형상을 가진 노이즈를 세포로 잘못 탐지하는 경우가 많다. 예를 들어, 이미지 상의 동그란 먼지 입자나 배경의 불규칙적인 형상이 세포로 인식되는 사례가 빈번히 발생하며, 이는 세포 탐지의 정확도를 크게 떨어뜨리는 한계를 초래한다. 특히, 세포가 불규칙적으로 분포되어 있거나 이미지 품질이 낮은 경우 이러한 문제는 더욱 두드러지게 나타난다.

이 프로젝트의 주요 목표는 이러한 기존 방법의 한계를 극복하고, 세포 탐색과 검증 과정을 결합한 자동화된 세포 카운팅 시스템을 개발하는 것이다. 이를 위해 OpenCV를 사용하여 세포가 있을 가능성이 높은 영역을 탐지하고, 딥러닝 기반의 분류 모델(ResNet)을 추가적으로 적용하여 탐지된 영역이 실제로 세포인지 여부를 검증하는 과정을 도입하였다. 이 검증 단계를 통해 OpenCV의 탐지 결과를 보완하고, 세포와 노이즈를 효과적으로 구분할 수 있다.

궁극적으로 본 연구는 정확하고 효율적인 세포 카운팅 시스템을 구현함으로써 실험자의 작업 부담을 줄이고, 연구 결과의 신뢰성을 높이는 데 기여하는 것을 목표로 한다. 이러한 시스템은 기존의 수작업 세포 카운팅 과정을 자동화하고, 의생명공학 실험 및 관련 연구에서 데이터 수집의 정확도와 생산성을 동시에 향상시킬 것으로 기대된다.

2. 방법론(Methodology)

2.1 데이터셋 구성 및 전처리

본 연구에서는 실험실에서 직접 촬영한 헤모사이토미터에 로딩된 세포의 현미경 이미지를 기반으로 데이터셋을 구성하였다. 데이터는 OpenCV 기반 탐지 결과를 활용하여 생성되었으며, 추가적으로 데이터 증강 기법을 적용하여 데이터의 다양성과 모델의 일반화 성능을 강화하였다.

2.1.1 세포 탐색 영역 정의

OpenCV 기반 객체 탐지 기술을 사용하여 세포가 존재할 가능성이 높은 영역을 추출하였다. 이 과정은 다음과 같은 단계로 구성되었다.

1) 그레이스케일 변환 및 이진화: 원본 이미지를 그레이스케일로 변환한 뒤, 임계값(Threshold)을 적용하여 이진화된 이미지를 생성하였다.

2) 컨투어 탐지: 이진화된 이미지에서 세포와 유사한 형상을 가진 영역을 탐지하였다.

3) 바운딩 박스 생성: 탐지된 컨투어의 중심점을 기준으로 고정된 크기(예: 24x24 픽셀)의 바운딩 박스를 설정하여 후보 영역을 잘라내었다.

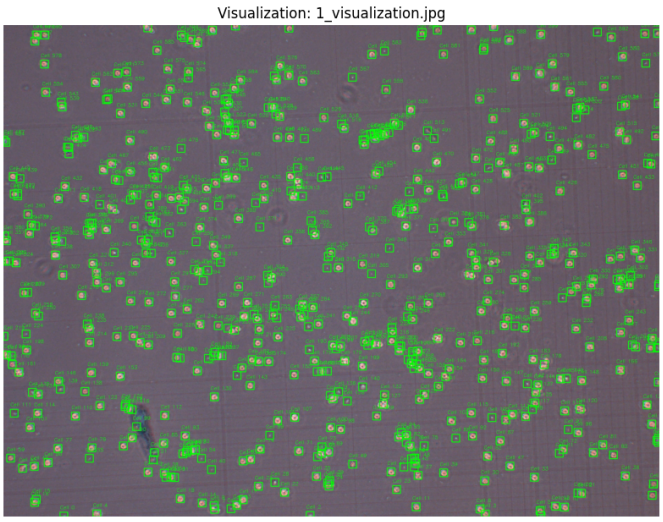

그림 1.

탐지된 모든 후보 영역은 독립적인 이미지로 저장되었으며, 딥러닝 모델 학습 및 테스트에 사용되었다.

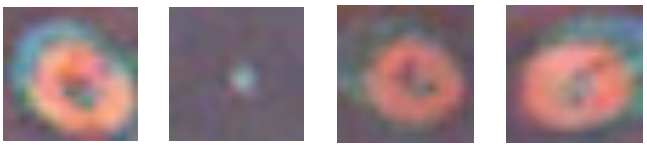

그림 2.

2.1.2 클래스 구성

데이터셋은 세포가 실제로 존재하는 영역과 세포와 유사한 노이즈 영역을 구분하기 위해 두 개의 클래스로 구성되었다.

1) Alive: 실제 세포가 존재하는 영역.

2) Misclassified: 세포가 아닌 노이즈를 포함한 영역.

모든 데이터는 실험자가 직접 라벨링하여 정확한 학습 데이터를 구성하였다. 라벨링 작업은 OpenCV를 통해 탐지된 각 세포 후보 영역을 하나씩 확인하며, 해당 영역이 실제 세포인지 또는 노이즈인지 수작업으로 지정하는 방식으로 진행되었다.

2.1.3 데이터 증강 및 정규화

탐지된 세포 데이터를 기반으로 데이터 증강(Data Augmentation)을 수행하여 데이터셋의 다양성을 확보하였다. 이 과정은 모델이 다양한 환경에서도 세포를 정확히 분류할 수 있도록 하기 위한 필수적인 과정으로, 다음과 같은 증강 기법이 적용되었다:

1) 좌우 반전: 세포를 좌우로 뒤집어 대칭적인 데이터를 확보.

2) 회전: -30도에서 +30도 범위 내 무작위로 회전.

3) 밝기 및 대비 조정: 이미지의 밝기와 대비를 ±20% 범위 내에서 변환.

4) 위치 변환 및 왜곡: 무작위로 위치를 이동하거나. Affine 변환을 통해 왜곡.

각 이미지에 대해 9개의 증강 이미지를 추가 생성하여, 데이터셋의 크기를 10배로 확장하였다. 모든 이미지는 딥러닝 모델의 입력에 적합하도록 64x64 픽셀로 리사이즈되었으며, RGB 값을 [0, 1] 범위로 정규화하였다.

2.2 모델 설계

본 프로젝트에서는 OpenCV를 활용해 초기 세포 탐지 작업을 수행한 후, 탐지된 세포 후보 영역이 실제 세포인지 노이즈인지 검증하기 위한 딥러닝 모델을 설계하였다. 이를 위해 ResNet18을 기반으로 한 딥러닝 모델을 구성하였으며, 학습 데이터는 직접 라벨링한 세포 이미지와 증강된 데이터셋을 사용하였다. 모델 설계는 다음과 같은 과정으로 진행되었다.

2.2.1 데이터셋 구성 및 데이터 로더

데이터셋은 직접 라벨링한 세포 후보 영역 이미지를 기반으로 구성되었다. 이를 학습 데이터와 테스트 데

이터로 분리하기 위해 scikit-learn의 train_test_split 함수를 사용하였다.

1) 학습 데이터: 데이터셋의 80%를 사용하여 모델을 학습.

2) 테스트 데이터: 데이터셋의 20%를 사용하여 학습되지 않은 데이터에 대한 모델의 일반화 성능을 평가.

훈련 및 테스트 데이터는 PyTorch의 DataLoader를 사용하여 배치 단위로 처리되며, 배치 크기는 32로 설정하였다. 이미지에 적용된 전처리와 증강(transform)은 모델이 다양한 상황에서도 세포를 잘 분류할 수 있도록 도움을 주었다.

2.2.2 모델 구조

본 연구에서 사용된 딥러닝 모델은 ResNet18(Residual Network)을 기반으로 설계되었다. ResNet은 잔차 학습(Residual Learning)을 통해 깊은 네트워크에서도 효과적으로 학습할 수 있는 구조를 제공한다.

1) Pretrained Model: ResNet18은 ImageNet 데이터셋으로 사전 학습된 가중치를 초기값으로 사용하였다.

2) Output Layer 수정: ImageNet의 1,000 클래스 분류용으로 설계된 ResNet18의 마지막 fully connected 레이어를 두 개의 클래스(alive, misclassified)에 맞게 수정하였다.

모델 정의는 PyTorch를 사용하여 구현되었으며, ResNet18의 구조는 간결하면서도 강력한 성능을 제공하여 본 연구에 적합하다.

2.2.3 학습 구성

모델 학습은 다음과 같은 구성으로 진행되었다:

1) 손실 함수: 분류 작업을 위해 크로스 엔트로피. 손실 함수(CrossEntropyLoss)를 사용하였다.

2) 최적화 알고리즘: Adam Optimizer를 사용하여 학습 속도와 안정성을 균형 있게 조정하였으며, 학습률은 0.001로 설정하였다.

3) 하드웨어 설정: GPU(CUDA)를 활용하여 학습 속도를 높였다.

2.2.4 학습 및 검증

모델 학습은 총 10 에포크(epoch) 동안 수행되었으며, 각 에포크에서 다음의 작업이 진행되었다:

1) Forward Pass: 입력 이미지를 모델에 전달하여. 예측값 계산.

2) Backward Pass: 손실 함수의 출력을 기반으로 역전파(Backpropagation)를 수행하여 가중치를 업데이트.

3) 정확도 계산: 각 에포크가 끝날 때마다 정확도를 계산하여 모델의 학습 상태를 모니터링.

학습 도중 계산된 평균 손실(Loss)은 모델의 수렴 상태를 확인하는 데 사용되었으며, 정확도(Accuracy)는 모델이 학습 데이터에 적합하게 학습되고 있음을 보여준다.

```
Epoch 1/10, Loss: 0.0946, Accuracy: 96.89%
Epoch 2/10, Loss: 0.0583, Accuracy: 97.89%
Epoch 3/10, Loss: 0.0543, Accuracy: 97.97%
Epoch 4/10, Loss: 0.0484, Accuracy: 98.24%
Epoch 5/10, Loss: 0.0398, Accuracy: 98.50%
Epoch 6/10, Loss: 0.0418, Accuracy: 98.47%
Epoch 7/10, Loss: 0.0387, Accuracy: 98.54%
Epoch 8/10, Loss: 0.0354, Accuracy: 98.58%
Epoch 9/10, Loss: 0.0340, Accuracy: 98.73%
Epoch 10/10, Loss: 0.0325, Accuracy: 98.75%
```

그림 3.

3. 결과 및 분석

3.1 모델 성능 평가

본 연구에서 설계된 모델의 성능은 테스트 데이터셋을 기반으로 평가되었으며, 정밀도(Precision), 재현율(Recall), F1-Score와 같은 주요 분류 성능 지표를 사용하여 분석하였다. 테스트 데이터는 총 5,080개의 샘플로 구성되었으며, 두 개의 클래스(alive, misclassified)로 분류되었다. 모델의 성능 평가는 다음과 같다:

1) 클래스별 성능 분석

Alive 클래스: alive 클래스에서 모델은 1.00(100%)의 정밀도, 1.00(100%)의 재현율, 1.00(100%)의 F1-Score를 기록하였다. 이는 모델이 실제 세포를 매우 정확하게 탐지하고 분류했음을 나타낸다.

Misclassified 클래스: misclassified 클래스에서는 0.97(97%)의 정밀도, 0.97(97%)의 재현율, 0.97(97%)의 F1-Score를 기록하였다. 이는 노이즈와 세포를 구분하는 과정에서 일부 샘플이 잘못 분류되었음을 보여주지만, 여전히 높은 성능을 유지하고 있음을 나타낸다.

2) 전체 성능

정확도(Accuracy): 전체 테스트 데이터에서 99%의 정확도를 기록하였다. 이는 모델이 대부분의 샘플을 정확하게 분류했음을 의미한다.

Macro Average: alive와 misclassified 두 클래스의 평균 성능은 정밀도, 재현율, F1-Score에서 모두 0.98(98%)을 기록하였다. 이는 클래스 간 균형 잡힌 성능을 보여준다.

Weighted Average: 데이터 클래스의 비율을 반영한 가중 평균 성능은 정밀도, 재현율, F1-Score에서 모두 0.99(99%)를 기록하였다.

모델은 alive 클래스에서 매우 뛰어난 성능, 그리고 misclassified 클래스에서도 높은 성능을 기록하며 전반적으로 우수한 분류 성능을 보여주었다. 테스트 데이터에서 99%의 정확도를 기록한 결과는 세포 탐지와 노이즈 제거 작업에서 제안된 모델의 효과성을 입증한다. 그러나 misclassified 클래스에서 일부 오류가 발생하였으므로, 데이터셋의 다양성을 늘리거나 추가적인 검증 과정을 통해 성능을 더욱 향상시킬 여지가 있다.

	precision	recall	f1-score	support
alive	1.00	1.00	1.00	4391
misclassified	0.97	0.97	0.97	689
accuracy			0.99	5080
macro avg	0.98	0.98	0.98	5080
weighted avg	0.99	0.99	0.99	5080

그림 4.

3.2 모델의 실제 적용 결과

학습된 모델의 실제 성능을 평가하기 위해, 새로운 테스트 이미지를 사용하여 테스트를 수행하였다. 테스트 과정은 OpenCV를 활용한 초기 세포 탐지와, 학습된 딥러닝 모델(ResNet18)을 사용한 세포 검증 단계로 구성되었다. 이를 통해 이미지에서 탐지된 세포의 개수와 각 세포의 클래스(alive, misclassified)를 분류하였다.

3.2.1 테스트 이미지 전처리 및 세포 탐지

테스트 이미지에서 OpenCV를 사용하여 초기 세포 탐지를 수행하였다. 그레이스케일 변환 및 이진화를 통해 컨투어를 검출하고, 각 세포 후보 영역에 대해 고정된 크기(24x24 픽셀)의 바운딩 박스를 생성하였다. 이 과정을 통해 총 197개의 세포 후보 영역을 추출하였다. 각 후보 영역은 독립적인 이미지로 저장되었으며, 이를 딥러닝 모델의 입력 데이터로 사용하였다. 탐지된 세포의 결과는 그림으로 시각화되었으며, 시각화를 통해 세포 후보 영역이 적절히 탐지되었음을 확인할 수 있었다.

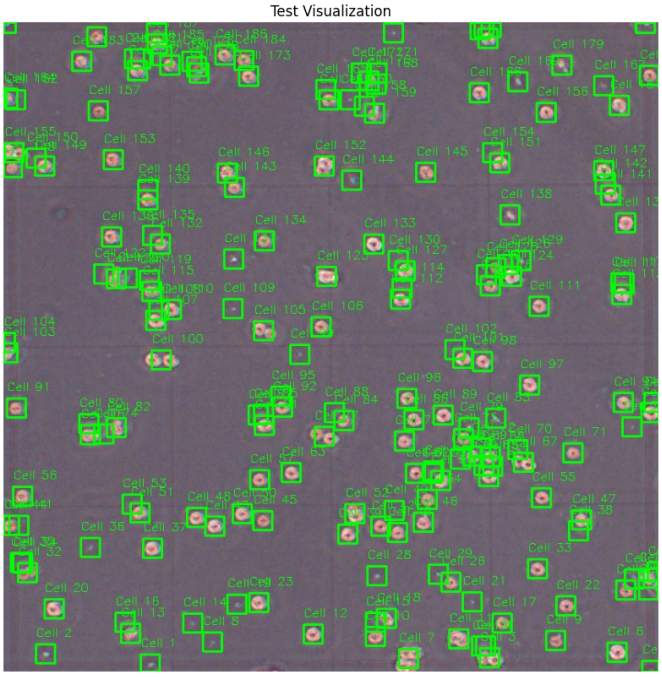

그림 5.

3.2.2 모델을 사용한 세포 수 검증

탐지된 148개의 세포 후보 영역은 학습된 모델에 입력되어 alive와 misclassified 두 클래스 중 하나로 분류되었다. 테스트 데이터는 배치 단위로 처리되었으며, 모델의 예측 결과는 다음과 같다

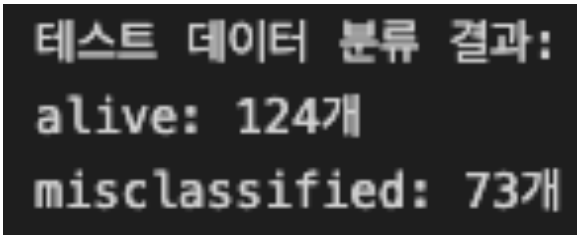

그림 6.

실제로 이미지에 포함된 세포의 수는 119개로 예측 결과와는 약간의 오차가 있지만, 이로써 모델이 OpenCV의 탐지 결과를 보완하고, 세포를 효과적으로 검증하는 데 기여했음을 확인할 수 있다.

4. 결론 및 제언

4.1 프로젝트 결론

본 프로젝트에서는 OpenCV 기반 객체 탐지와 ResNet18 기반 CNN 분류 모델을 결합하여 세포 카운팅 자동화 시스템을 구현하였다. 이 시스템은

OpenCV를 통해 초기 탐지된 후보 영역을 CNN 모델이 검증하여 세포의 개수를 판별하는 방식으로 작동하며, 테스트 결과 Accuracy 99%를 기록하였다. 실제 세포 수와 비교했을 때 오차가 매우 적은 수준으로 나타났으며, 시스템의 높은 정확성과 신뢰성을 입증하였다. 또한, 본 프로젝트는 기존 수작업 방식에 비해 시간 절약과 결과의 일관성을 제공함으로써, 실험 생산성을 크게 향상시킬 수 있는 가능성을 보여주었다.

4.2 프로젝트 한계

본 프로젝트는 세포 카운팅 자동화의 가능성을 확인했지만, 몇 가지 한계점이 존재하였다. 첫째, 데이터셋이 다양한 환경에서 수집되지 않아 조명 변화나 해상도 차이와 같은 실제 환경에서의 성능을 충분히 검증하지 못하였다. 둘째, OpenCV의 findContour를 활용한 객체 탐지 단계에서 세포와 유사한 노이즈를 불필요하게 탐지하여 CNN 모델의 연산 부담을 가중시키는 문제가 있었다. 마지막으로, 테스트 데이터가 훈련 데이터에서 증강된 데이터로 구성되어 있어 실제 환경에서의 일반화 성능을 판단하는 데 한계가 있었다.

4.3 향후 개선 방향

향후 프로젝트의 개선을 위해 데이터셋의 다양성을 확대하고, 모델의 성능을 최적화하는 노력이 필요하다. 다양한 조도와 배경 조건에서 촬영된 데이터를 추가로 수집하고, 데이터 증강 기법을 활용하여 모델의 일반화 성능을 강화해야 한다. 또한, OpenCV 기반 탐지 단계에서 발생하는 불필요한 노이즈 탐지를 줄이기 위해 후처리 알고리즘을 도입하거나 탐지 기법을 개선해야 한다. 마지막으로, CNN 모델의 경량화와 속도 최적화를 통해 실시간 처리 성능을 높임으로써, 실험 및 연구 환경에서 더욱 실질적으로 활용될 수 있는 시스템으로 발전시킬 필요가 있다.