

2021 한이음 공모전 개 발 보 고 서

2021. 10. 17

프로젝트명	국문	필기 인식을 이용하는 악보 작성 App
	영문	App to write sheet music using handwriting recognition
작 품 명	Music Note	

요 약 본

작품 정보		
프로젝트명	국문	필기 인식을 이용하는 악보 작성 App
	영문	App to write sheet music using handwriting recognition
작품명	Music Note	
작품 소개	Deep Learning을 이용하여 사용자가 오선지에 작성한 악보를 실시간으로 인식해 온전한 악보가 담긴 이미지, 파일로 만들어주는 iPad 기반 앱	
작품 구성도		
작품의 개발배경 및 필요성	<p>악보 작성 과정에서 PC 환경이나 <Notation Pad>와 같은 버튼을 이용하여 음표를 하나씩 추가하는 비효율적인 작업 방식의 앱에 비해, 필기 인식을 이용한 악보 작성 앱은 손쉬운 입력 방식을 이용하여 작업 효율을 크게 높여주고 있습니다. <Staff Pad>라는 이름의 아이패드용 앱이 있으나, 11만원이라는 비합리적인 금액에 비해, 실사용에서 인식률이 매우 낮고 반응속도가 떨어져 제한이 많습니다. 이러한 부분을 개선하여 합리적인 가격으로 제공했을 때, 그 가치가 매우 높을 것으로 예상됩니다.</p>	
작품의 기대효과 및 활용분야	<p>합리적인 금액으로 구매 가능한 앱이므로, 조금 더 많은 사용자들에게 부담없는 가격으로 효율적인 작곡 환경을 제공합니다.</p>	
작품의 특징점	<p>합리적인 가격, 실사용이 가능한 인식률, 반응속도, 직관적인 인터페이스</p>	
작품 기능	<ul style="list-style-type: none"> · 소셜 로그인을 기반으로 한 계정 관리 · 악보 데이터 저장, 관리 · 악보 작성을 위한 필기 인식 인터페이스 · 사용자의 필기를 딥러닝 모델을 이용하여 기호로 변환 · 변환된 기호 이미지를 악보에 출력 · 출력된 악보 재생 	

본 문

I. 작품 개요

※ 평가항목 : 기획력 (필요성, 차별성)

1. 작품 소개

1) Music Note

- Music Note는 사용자가 손글씨로 악보를 작성할 수 있는 아이패드 앱
- 사용자가 스마트 펜을 사용해 작성한 악보를 완성된 악보 이미지로 변환함

2) 기획 의도

- 버튼 입력 방식의 앱은 음악 기호를 하나씩 입력하는 특성상, 작업 소요시간이 길어짐
- 상용으로 존재하는 필기 인식 기반의 앱은 높은 가격으로 사용자에게 금전적인 부담을 줌
- 기존 앱의 낮은 인식률과 느린 반응 속도에 대한 개선하는 것을 목표로 함

3) 작품 내용

- 딥러닝 모델을 통해 사용자가 입력한 음악 기호를 식별하고 완성된 이미지 형태로 출력함
- 아이패드 앱의 PencilKit 캔버스를 이용하여 필기 인식 인터페이스를 구성함
- 데이터베이스 서버를 연동하여 작성한 악보나 계정 데이터를 관리함

2. 작품의 개발 배경 및 필요성

1) 방식의 효율성

- PC 환경이나 <Notation Pad> 와 같은 버튼을 이용하여 음표를 하나씩 추가하는 비효율적인 작업 방식의 앱에 비해, 필기 인식을 이용한 악보 작성 앱은 손쉬운 입력 방식을 이용하여 작업 효율을 크게 높여주고 있음

2) 가격의 효율성

- <Staff Pad> 라는 이름의 아이패드용 앱이 있으나, 11만원이라는 비합리적인 가격이 책정되어 있으며, 사용자들은 대체재가 없기 때문에 불가피하게 해당 앱을 구매하게 됨
- 높은 가격에 비하여 낮은 인식률과 반응속도는 실사용에 큰 제한 사항이 되고 이에 대한 개선이 필요함

3. 작품의 특징 및 장점

1) 방식의 효율성 개선

- 버튼을 이용하여 음표를 하나씩 추가하는 방식 대신 사용자가 손글씨를 이용해 악보를 작성하는 방식을 도입하여 악보 작성 방식의 효율성을 개선함

2) 가격의 효율성 개선

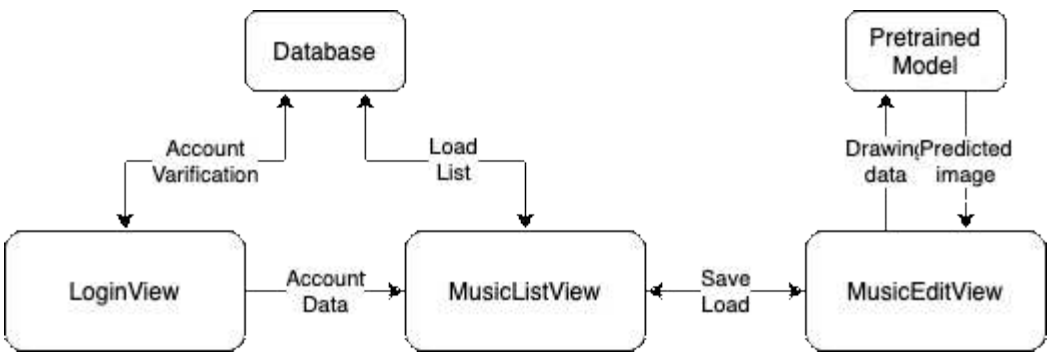
- 기존에 존재하는 App의 낮은 인식률과 반응속도를 적절한 딥러닝 모델 사용을 통해 성능을 향상시켜 사용자가 App을 구매하였을 때 만족감을 느끼게 함

II. 작품 내용

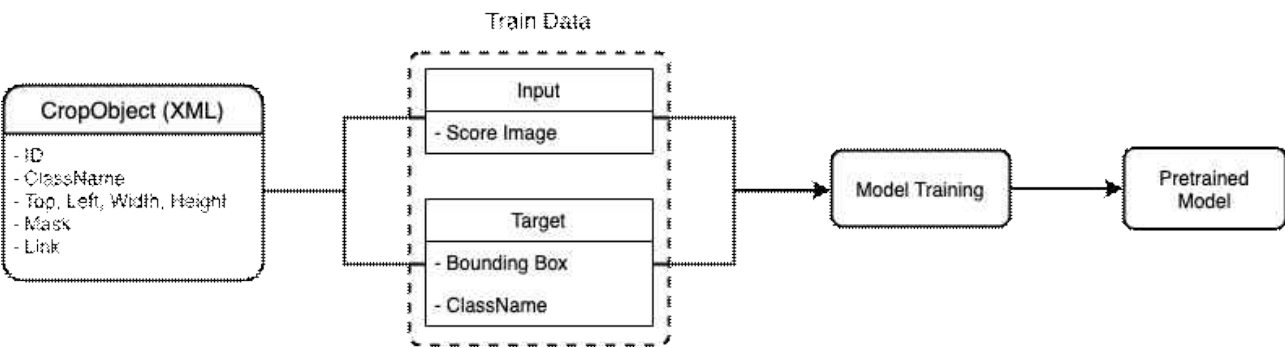
※ 평가항목 : 기술력 (기능구체성, 난이도, 완성도)

1. 작품 구성도

1) 앱 서비스 설계도



2) 딥러닝 모델 설계도

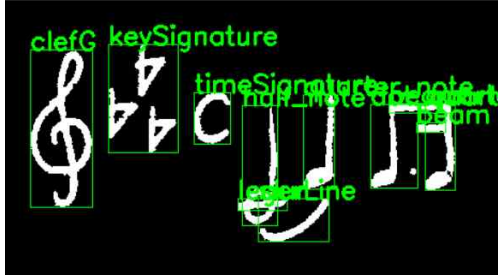


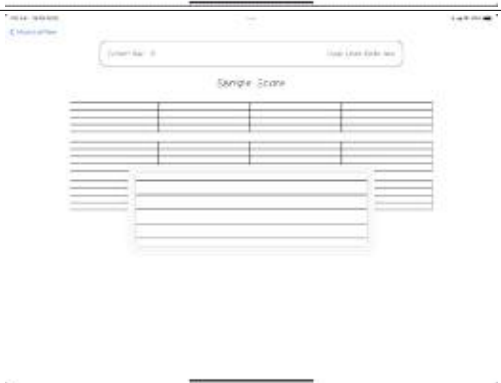


2. 작품 기능

1) 전체 기능 목록

구분	기능	설명	현재진척도(%)
S/W	필기 인식	딥러닝 모델을 통한 필기 인식 모델 구현하고, 변환된 기호 이미지를 악보에 출력	9/30
	계정기반 악보 관리	소셜 로그인을 기반으로 한 계정 및 악보 데이터를 저장하고 관리	9/15
	필기 인식 인터페이스	악보 작성을 위한 필기 인식 인터페이스 구현	100
	악보 재생	작성된 악보를 악기 소리를 이용하여 재생	10/10

2) S/W 주요 기능

기능	설명	작품실물사진
필기 인식	딥러닝 모델을 통한 필기 인식 모델 구현하고, 변환된 기호 이미지를 악보에 출력	
계정기반 악보 관리	소셜 로그인을 기반으로 한 계정 로그인	
	악보 데이터를 저장하고 관리	
필기 인식 인터페이스	악보 작성을 위한 필기 인식 인터페이스 구현	

3) H/W 주요 기능

기능/부품	설명	작품실물사진

3. 주요 적용 기술

1) CNN

- CNN은 이미지 처리에서 이미지의 공간 정보를 유지한 상태로 학습이 가능한 모델임
- 기본적으로 Region Feature를 뽑아내는 Convolution Layer, Feature Dimension을 줄이기 위한 Pooling Layer, 그리고 최종적인 분류를 위한 FC-Layer (Fully Connected Layer, 일반적인 MLP 구조)로 이루어짐

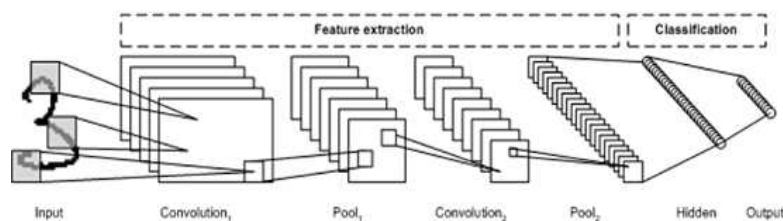


그림 1: CNN 구조

- Convolution Layer는 Receptive Field(Filter, Kernel)를 정의해 입력 층의 이미지의 Feature를 추출하는 역할을 하며 Receptive Filed가 지정한 간격으로 순회를 하면서 Convolution 연산을 통해 Feature Map을 만들어냄

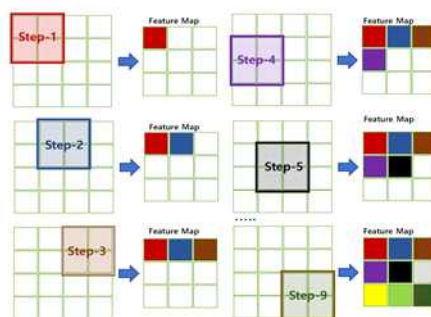


그림 2: Convolution 연산

- Pooling Layer는 Data의 크기를 줄이거나 특정 데이터를 강조하는 용도로 사용, Training 속도를 향상시키기 위해 Dimension을 줄이는 개념이며 Average

Pooling, Max Pooling 등이 존재



그림 3: Pooling Example

- Fully-Connected-Layer는 일반적인 MLP 구조와 동일하며, 이전 Pooling Layer에서 나온 Feature를 Flatten 시켜 MLP의 Input으로 놓고 학습을 진행함

2) Faster-RCNN

- Faster-RCNN은 Object Detection에 사용되는 RCNN 및 Fast-RCNN을 개선하여 만들어진 모델이며, 사물이 무엇인지 분류하는 것만이 아닌 사물의 위치까지 찾아주는 2-stage detector 모델임
- Faster-RCNN은 기존에 예상되는 지역마다 CNN을 적용하는 방식, Selective Search를 사용하는 방식에서 RPN(Region Proposal Network)를 도입하여 속도를 더욱 향상시킨 모델임

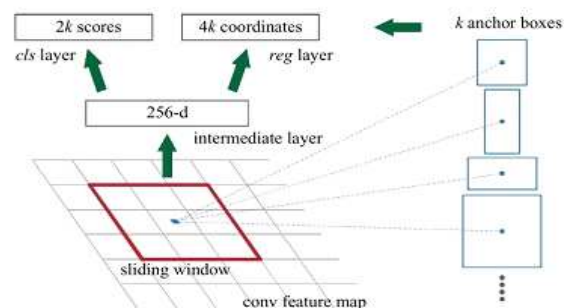


그림 4: RPN 구조

- 이미지가 Convolution Layers를 통과하여 나온 Feature Map을 바탕으로 RPN에서는 사물의 위치를 표시하는 Bounding Box와 해당 Box가 사물일 확률을 출력하고, 최종적으로 분류기에서 사물의 Class와 박스의 위치를 출력함

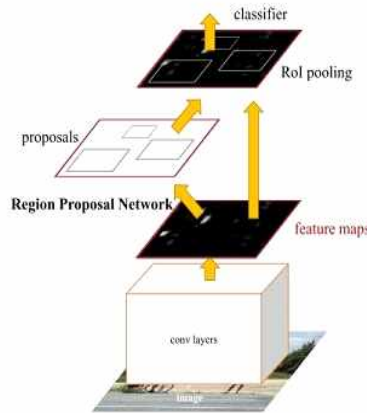


그림 5: Faster-RCNN

3) YOLOv5

- YOLO(You Only Look Once)는 Faster-RCNN과 같은 Object detection에서 사용하는 모델이며, Faster-RCNN과 다르게 1-stage detector 모델임
- YOLO는 이미지 전체에 대해서 하나의 신경망이 한 번의 계산만으로 bounding box와 클래스 확률을 예측하기 때문에 탐지 속도가 빠르고 주변 정보까지 학습하며 이미지 전체를 처리하기 때문에 background error가 작음
- Input images를 $S \times S$ grid로 나누고 각각의 grid cell은 B개의 bounding box와 그 bounding box에 대한 confidence score를 예측함

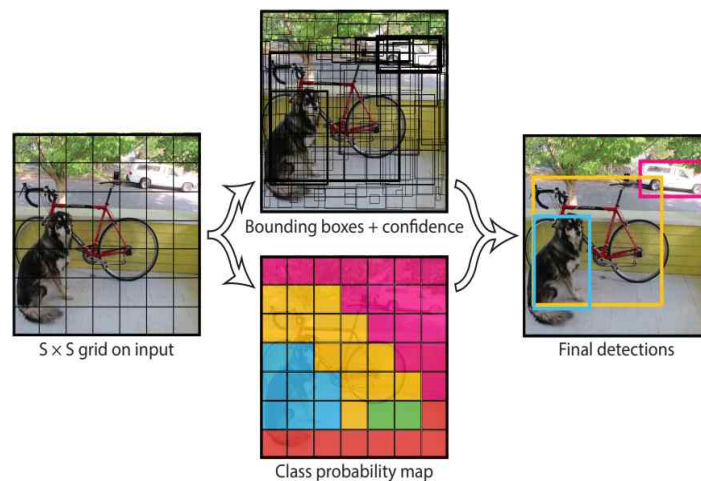


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

그림 6: YOLOv5

4) SwiftUI

- SwiftUI는 모든 애플 플랫폼에서 사용자 인터페이스를 만들 수 있게 해주는 새로운 개발 패러다임으로, 하나의 도구 및 API로 모든 애플 기기의 사용자 인터페이스를 구현할 수 있도록 함
- SwiftUI는 선언적 구문을 사용함으로써, 기능 명시가 수월하며, 기존 StoryBoard 대비 가독성이 향상되고 유지 관리가 용이함
- SwiftUI의 PencilKit을 이용하여 사용자의 터치 입력이나, 애플 펜슬 입력을 인식하고, 입력된 데이터를 출력하는데 사용함



그림 7: SwiftUI Pencilkit

4. 작품 개발 환경

구분		상세내용
S/W 개발환경	OS	macOSX Big Sur 11.5.1
	개발환경(IDE)	VSCode(Web Application), Xcode(IPad Application)
	개발도구	Django, SwiftUI, cocoapods
	개발언어	HTML, CSS, JavaScript, SwiftUI
	기타사항	웹, 앱 기반 서비스 개발환경
S/W 개발환경	OS	Windows 10
	개발환경(IDE)	PyCharm
	개발도구	Pytorch, Keras
	개발언어	Python
	기타사항	모델 구축 S/W 개발환경
프로젝트 관리환경	형상관리	GitHub
	의사소통관리	Zoom, Webex
	기타사항	2주에 한번씩 오프라인 회의를 거쳐 형상 관리와 의사소통을 진행함

5. 기타 사항 [본문에서 표현되지 못한 작품의 가치(Value)] 및 제작 노력

- XML 형태로 되어있는 데이터 파일을 학습 가능한 형태의 이미지 데이터와 그에 해당하는 Label 및 Bounding Box 좌표로 변환하는 과정에 많은 노력을 가함

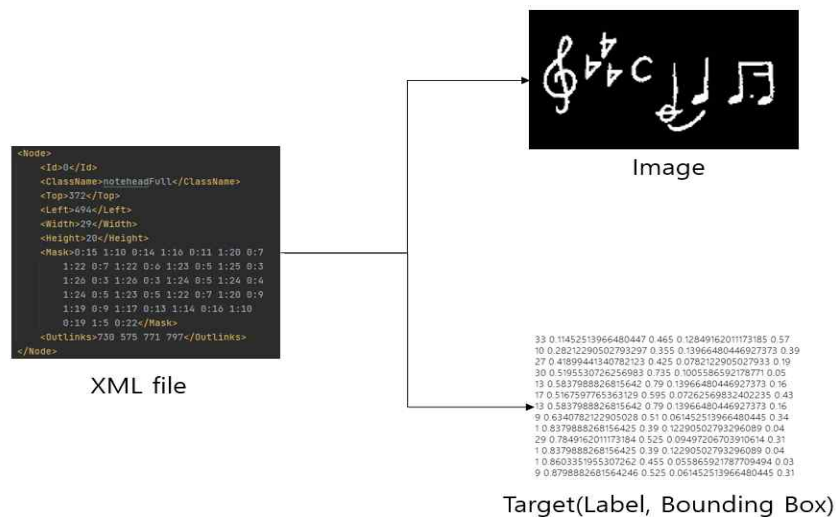


그림 8: Data Preprocessing

- 기존의 스토리보드 방식으로 작성된 많은 드로잉 기반 앱에 비해 비교적 자료가 적지만 최신의 방식으로 발전 가능성이 높은 SwiftUI 으로 서비스를 구현한다는 점에서 노력이 많이 필요함

III. 프로젝트 수행 내용

※ 평가항목 : 수행능력 (문제해결능력, 수행충실성)

1. 프로젝트 수행일정

프로젝트 기간 (ICT멘토링 사이트 기준)		2021.04.13. ~ 2021.11.30.											
구분	추진내용	프로젝트 기간											
		3월	4월	5월	6월	7월	8월	9월	10월	11월	12월		
계획	프로젝트 계획												
분석	관련 논문 조사, 기반 기술 정리												
설계	데이터 전처리												
개발	모델 학습												
	웹 버전												
	앱 버전												
테스트	아이패드를 이용한 테스트, 성능 향상												
종료	종료												

2. 프로젝트 추진 과정에서의 문제점 및 해결방안

1) 프로젝트 관리 측면

- 각자의 개발 범위가 명확하게 구분되어, 프로젝트 버전 관리, 통합 과정에서 별다른 문제가 없었음

2) 작품 개발 측면

2.1) 필기 인식 모델

- 딥러닝 모델 설계 과정에서 처음에는 1-layer CNN 모델 학습을 통해 사용자가 그린 음악 기호를 분류하려고 시도함
- 1-layer CNN 모델은 음악 기호 분류 성능은 준수하나 사용자가 작성한 음표의 높낮이를 알 수 없는 문제가 발생함
- Faster-RCNN 모델로 분류 및 그린 음악 기호의 위치를 모두 찾는 방법을 찾아냈으나, Swift에서 사용하는 Script로 바꾸는 방법을 현재 지원하지 않아서 Object Detection을 할 때 사용하는 다른 모델인 YOLOv5를 이용하여 이를 해결함

2.2) 앱 기반 서비스

- UIKit 과 SwiftUI 사이에서 부분적인 정보들로 인하여 초기에 필요한 자료들을 수집하는 것에 어려움이 있었음

- SwiftUI 와 UIKitDelegate lifecycle을 확실하게 선택한 이후에는 자료 수집이 수월해졌음

3. 프로젝트를 통해 배우거나 느낀 점

- 필기 인식 관련 이미지 처리에 사용되는 다양한 딥러닝 모델을 조사하고 배울 수 있는 기회가 되었음
- 모델 학습에 사용할 데이터셋을 구축 및 전처리 하는 과정의 중요성에 대해 깨닫게 됨
- 웹, 앱을 동시에 개발하여 유기적으로 돌아가는 시스템에 대한 이해도를 높일 수 있었음

IV. 작품의 기대효과 및 활용분야

※ 평가항목 : 기획력 (활용가능성)

1. 작품의 기대효과

- 기존 소프트웨어의 낮은 음악 기호 인식성, 반응속도 등의 기술적인 불만들을 해소하고 이용자들을 만족시킬 수 있음.
- 기존 소프트웨어의 비현실적인 가격에 대한 개선을 도모할 수 있음.
- 앱 기반의 환경과 소셜 로그인 기반 데이터베이스를 통해 이용자들이 어느 곳에서나 악보를 작성할 수 있는 환경을 제공할 수 있음.

2. 작품의 활용분야

- 클래식, 실용음악 등 음악 계열에 종사하는 사람들이 환경의 제약 없이 편하게 아이패드를 통하여 작곡을 할 수 있다.
- 필기 인식이라는 직관적인 방식을 통해 음악을 처음 접하는 사람들도 부담없는 금액으로 작곡을 경험할 수 있다.