

1. MERN 스택이란?
2. Docker 설정
3. NODE 개발환경 설정
4. REACT 개발환경 설정
5. EXPRESS 개발환경 설정
6. MONGO DB 개발환경 설정
7. 개발 시작
8. 실행 순서
9. 코드 설명

코드상세 Github

주소: [https://github.com/ChoHyunGook/CRUD\\_PROJECT](https://github.com/ChoHyunGook/CRUD_PROJECT)



## 1. MERN 스택이란?

M= MongoDB, E=Express.js, R= React.js, N= Node.js 를 사용하여 웹사이트를 개발하는 것을 말한다.

Server에는 Node.js, Express.js, MongoDB 가 있으며 Client(Browser)인 React.js 가 있고 서로 Requests 와 Responses 를 주고받는다. 이 안에서의 Data 는 JSON Format 이라고 한다.

Container 기반 가상화 플랫폼인 Docker 위에 쌓아가는 구조로 Node.js, React.js, Express.js, MongoDB 의 순으로 쌓아간다.

**1.1 Stack(스택)이란?** :스택(stack)은 제한적으로 접근할 수 있는 나열 구조이다. 그 접근 방법은 언제나 목록의 끝에서만 일어난다. 끝먼저내기 목록(Pushdown list)이라고도 한다. 스택은 한 쪽 끝에서만 자료를 넣거나 뺄 수 있는 선형 구조(LIFO - Last In First Out)으로 되어 있다. 자료를 넣는 것을 '밀어넣는다' 하여 푸쉬(push)라고 하고 반대로 넣어둔 자료를 꺼내는 것을 팝(pop)이라고 하는데, 이때 꺼내지는 자료는 가장 최근에 푸쉬한 자료부터 나오게 된다. 이처럼 나중에 넣은 값이 먼저 나오는 것을 LIFO 구조라고 한다. -위키백과-

**1.2 Docker :** Docker 는 컨테이너 기반의 오픈소스 가상화 플랫폼이다. 가상 머신처럼 독립된 실행환경을 만들어주는 것으로, 운영체제를 설치한 것과 유사한 효과를 낼 수 있지만, 실제 운영체제를 설치하지 않기 때문에 설치 용량이 적고 실행 속도 또한 빠르다.

**1.3 MongoDB :** MongoDB 는 C++로 작성된 문서지향적 데이터베이스이며 NoSQL DB 이다 Not Only SQL 의 약자로 기존 DBMS 가 가지고 있는 특성도 가지고 있으며, JavaScript 의 JSON 과 같은 동적 스키마형 Documents 들을 선호하는 특징이 있다. 이러한 것들을 MongoDB 에선 BSON 이라고 한다.

**1.4 Express.js :** Express.js 는 Node.js 를 위한 Web FrameWork 이며, 빠르고 개방적이며 간결한 표준 서버 FrameWork 이며, JavaScript 언어로 작성된다.

**1.5 React.js :** React.js 는 사용자의 Interface 를 만들기 위한 자바스크립트의 Library 이다. 스스로의 상태를 관리하는 Encapsulated 된 Component 를 기반으로 되어있으며, JavaScript 로 작성된다.

**1.6 Node.js :** Node.js 는 JavaScript Engine 에 기반한 Server Side Platform 이며, 내장 HTTP Server Library 를 포함하고 있어서 별도의 소프트웨어가 없이 동작할 수 있다. MERN 스택의 제일 밑 부분에 Server에서 JavaScript 가 작동되게 하는 Runtime Environment Platform 이다. Node.js 의 특징 중 하나는 카카오톡에서 쓰는 것을 예로 들어 Non-Blocking 으로 Requests 한 후 Responses 를 기다리지 않고 다른 API 를 Requests 할 수 있으며, 이전에 Requests 했던 API 의 Responses 가 오면 Event Loop 가 확인하여 처리해준다.

## 2. Docker 설정

Window 기반으로 Docker 를 Install 하는 방법이다.

해당 사이트 <https://docs.docker.com/desktop/windows/install/>

에서 Docker Desktop for Windows 를 클릭하여 exe 파일을 다운 받아준다. 해당 파일을 실행하여 설치를 진행해주는지 Configuration에서 Install required Windows components for WSL2 와 Add shortcut to desktop 둘다 체크를 하고 설치를 진행한다. 설치가 완료된 후 정상적으로 Install 이 되었는지 확인하기 위해 Windows PowerShell 또는 cmd 에서 Docker 버전을 확인하는 명령어 docker -version 을 적은후 버전을 확인한다.

### 3. MERN 스택의 Node.js 설정

Window 기반으로 Node.js 를 Install 하는 방법이다.

해당사이트 <https://nodejs.org/ko/> 에서 각 버전에 따른 exe 파일을 다운로드하여 설치를 진행한다. 그리고 Node.js 의 경우 환경변수를 설정해주어야하는데 윈도우+R 키 또는 시작프로그램에서 실행창을 열어 sysdm.cpl ,3 을 입력해 준다. 그럼 시스템 속성에서 환경변수를 클릭한 후 시스템변수에 새로만들기를 클릭하여 변수이름으로는 NODE\_HOME 을 입력하고 값에는 설치되어있는 경로 C:\Program Files\nodejs 를 입력해 준다. 그리고 환경변수 및 시스템속성에 확인, 적용 후 정상적으로 node.js 가 설치되어 있는지 확인하기 위해 Windows PowerShell 또는 cmd 에서 버전을 확인하는 명령어 node -version 을 적은후 다운로드했던 버전이 맞는지 확인한다.

### 4. React.js 설정

React 를 사용하기 위해 VsCode 를 해당사이트 <https://code.visualstudio.com/> 에서 exe 파일을 받아 설치하여준다. 설치 진행 중 추가작업 선택에서 체크는 모두 하고 설치를 진행한다. 그리고 진행에 필요한 확장프로그램들을 설치해 준다.

### 5. Express.js 설정

Express 를 설치하기 위하여 해당 Express 를 설치할 폴더에서 Windows PowerShell 또는 cmd 를 실행하여준다. Express 를 설치할 폴더에서 이름을 지정해주는 Linux 명령어 mkdir '폴더명' 을 실행한후 만들어진 폴더 안에서 명령어 npx express-generator 를 작성한 후 실행하면 express.js 기본설정이 완료된다.

### 6. MongoDB 설정

MongoDB 를 설치하기위해 해당사이트 <https://www.mongodb.com/try/download/compass> 에서 원하는

버전에 맞는 파일을 받아준다. Zip 파일로 받았을때 압축을 푼 후 압축을 푼 폴더 내 MongoDBCompass.exe 파일을 실행하여 준다. 해당 프로그램에 설치 후 Docker 에 MongoDB 를 올려주기 위하여 터미널에 다음 명령어들을 입력한다. Docker pull mongo, docker run --name mongodb\_docker -e MONGO\_INITDB\_ROOT\_USERNAME=root -e MONGO\_INITDB\_ROOT\_PASSWORD=root -d -p 27017:27017 mongo 를 차례대로 입력해주면 Docker 에 mongodb\_docker 라는 container 가 추가되어 있는 것을 확인할 수 있다.

```
PS C:\Users\bitcamp\soccer\soccer-express> mongosh "mongodb://localhost:27017" --username "root" --password "root"
Current Mongosh Log ID: 624d2a62cde72cb3bba4bb3f
Connecting to:          mongodb://localhost:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Using MongodB:          5.0.6
Using Mongosh:          1.1.7

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

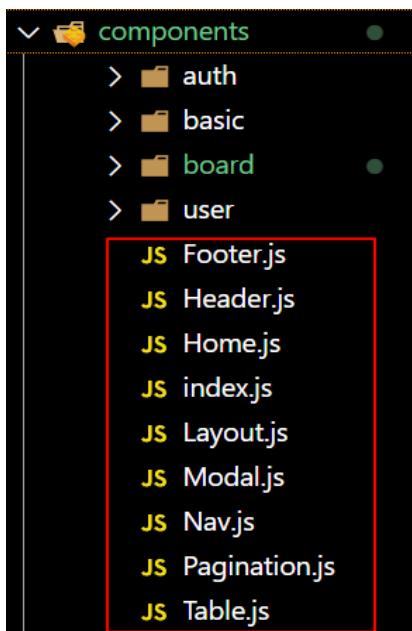
-----
The server generated these startup warnings when booting:
2022-04-06T05:51:25.756+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2022-04-06T05:51:26.359+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest setting it to 'never'
-----

test> show dbs
admin   213 kB
config  36.9 kB
local   73.7 kB
test> use soccer_db
switched to db soccer_db
soccer_db> db
Browserslist: caniuse-lite is outdated. Please run:
npx browserslist@latest --update-db

Why you should do it regularly:
https://github.com/browserslist/browserslist#browsers-data-updating
soccer_db> show dbs
admin   213 kB
config  36.9 kB
local   73.7 kB
soccer_db> db.user.insert({ 'username': 'test', 'password': '1', 'name': '테스트', 'telephone': '010-1234' })
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '_id': ObjectId("624d2ba4ef7fa158d10a2d1f") }
}
soccer_db> show dbs
admin   213 kB
config  73.7 kB
local   73.7 kB
soccer_db  8.19 kB
soccer_db> show collections
user
soccer_db> |
```

### 7. 개발 시작

MERN 스택의 Node.js 는 앞서 설치를 하였고 그다음 순서인 React 를 시작한다. 프로젝트를 생성 한 후 화면이 직접적으로 보이는 components 폴더와 Router, Middleware,Saga 가 있는 리덕스인 modules 폴더를 생성해주고 기존에 있던 pages 는 Next 로 프레임워크로 사용한다. 첫 화면을 구성하기 위해 기본 아래 구성을 만들어준다.



위 사진과 같이 auth 폴더를 생성한 후 로그인에 필요한 login.js 와 회원가입에 필요한 Register.js 를 생성해준다.



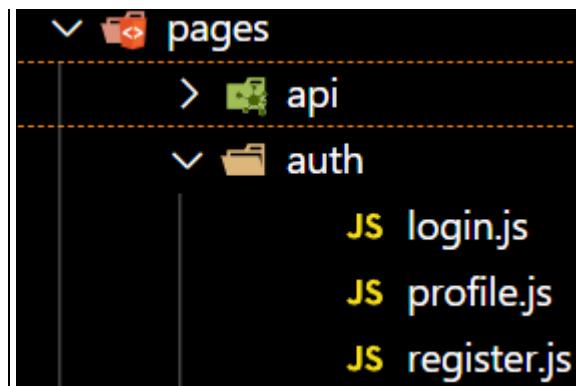
그리고 user 폴더를 생성해 프로필을 볼 수 있는 Profile.js 를 생성해준다. 화면을 구성하는 코딩을 한 후 아래와 같이 index.js 에 경로를 지정한다

```

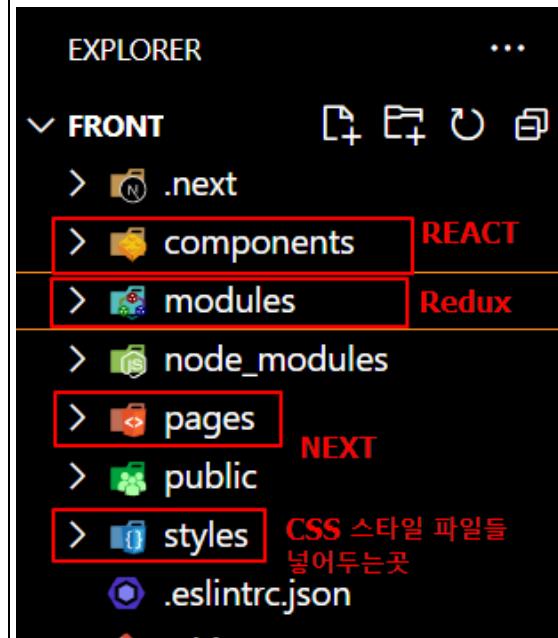
1 export * from './Home'
2 export * from './Header'
3 export * from './Nav'
4 export * from './Table'
5 export * from './Pagination'
6 export * from './Footer'
7 export * from './Modal'
8 export * from './Layout'
9 export * from './auth/Login'
10 export * from './auth/Register'
11 export * from './user/Profile'
12 export * from './board/Write'
13 export * from './board/Update'
14 export * from './board/Remove'
15 export * from './board/List'

```

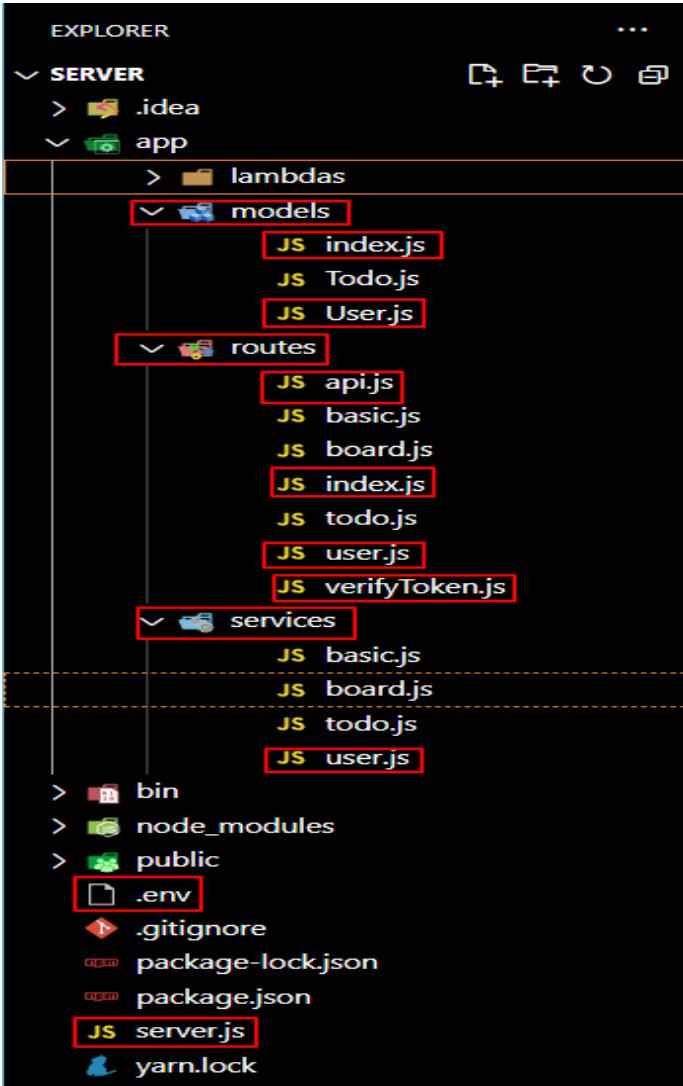
Layout.js 그리고 pages 폴더에 index.js 안에 화면 구성을 완료시킨다. 그리고 \_app.js 에 코딩을 한 후 components 폴더처럼 auth 폴더에도 똑같이 login.js, profile.js, register.js 를 동일하게 만들어준다.



그리고 modules 폴더에도 auth, user 폴더를 생성하여 똑같이 3 개를 동일하게 생성해 준다.



앞서 말했듯 components 엔 화면을 구성하는 코딩이 필요하며 modules 에는 각 컴포넌트의 데이터 전달의 상태를 관리하기 위한 상태관리 도구인 리덕스를 바닐라 스크립트로 자신이 직접 컨트롤하기 쉽게 구현하며 pages 는 React 의 프레임워크인 Next 를 코딩 해 준다.



그리고 MERN의 다음인 Server인 Express로 넘어간다. 해당 Express에 기본으로 되어있는 app.js를 server.js로 만들어주고 경로를 지정해준다. Express에 필요한 models, routes, services 폴더들을 생성하고 위와 같이 만들어준다.

.env로 들어가 해당 코드들을 입력하여 MongoDB와 REACT를 연결하기 위해 아래와 같이 입력해 준다.

```
1 .env
2 PORT=5000
3 MONGO_URI=mongodb://root:root@localhost:27017/soccerdb?authSource=admin&authMechanism=SCRAM-SHA-1
4 JWT_SECRET=jwtSecret
5 ORIGIN=http://localhost:3000
```

여기서 로그인에서 필요한 Token은 PostMan을 사용하여 <https://www.postman.com/>에서 다운을 받아 사용한다.

## 8. 실행순서

MERN의 실행순서는 Stack 구조로써 반대로 Docker를 실행하며 뭉고 DB를 켜고 EXPRESS를 실행시키고 REACT를 실행하는 순서이다.

## 9. 코드설명

코드의 설명을 조금 더 하면 React 부분인 component 폴더의 경우 onSubmit과 onChange와 같은 이벤트가 발생하는 기능을 사용하여 값을 넘겨주는 무상태(props)로 만들어 주며, 아래의 그림과 같이 callback 함수를 사용하여준다.

```
useEffect(() => {
  console.log('모듈에 저장된 로그인값: ' + JSON.stringify(loginUser))
  if (loginUser === null) {
    setUserUrls({
      subTitles: [
        | '회원가입', '로그인'
      ],
      urls: ["/auth/register", "/auth/login"]
    })
  }
})
```

다음으로 Next 부분의 pages 폴더의 경우 무상태를 State로 만들어주는 코딩을 해야하는데 이곳에서도 동일하게 onSubmit과 onChange의 이벤트 발생을 사용하여 아래와 같이 lambda를 사용하여 준다.

```
const onChange = e => {
  e.preventDefault()
  const {name, value} = e.target
  setUser({
    ...user,
    [name]: value
  })
}
```

그다음 modules에서는 서버인 express로 넘기는 역할을 하며 비동기 객체인 axios를 아래와 같이 post를 사용하여 값을 이동시킨다.

```
const SERVER = 'http://127.0.0.1:5000'

const loginAPI = payload => axios.post(
  `${SERVER}/user/login`,
  payload,
  {headers}
```

Modules의 index.js에는 rootReducer와 rootReducer 부분을 잘 확인해야하며, 아래 사진과같이 해당부분들을 잘 확인해 주어야한다.

```
const rootReducer = combineReducers({
  index: (state = {}, action) => { // 데이터가 콘솔에만 찍
    switch (action.type) {
      case HYDRATE:
        console.log("HYDRATE", action);
        return { ...state, ...action.payload };
      default:
        return state;
    }
  },
  login,
  register,
});

export function* rootSaga() {
  yield all([counterSaga(), registerSaga(), loginSaga()]);
}

export default rootReducer;
```

그리고 express로 넘어가 server.js의 엔트리 포인트를 잡아주는 use를 사용하여 주고 mongoose라는 뭉고 db의 연결하는 라이브러리를 아래와 같이 코딩하여 준다. Mongoose의 설치는 npm i mongoose 을 터미널에 실행하여준다.

```
async function startServer() {
  const app = express();
  const {mongoURI, port, jwtSecret} = applyDotenv();
  app.use(express.static('public'));
  app.use(express.urlencoded({extended: true}));
  app.use(express.json());
  const _passport = applyPassport(passport, jwtSecret);
  app.use(_passport.initialize());
  app.use("/", index);
  app.use("/api", api);
  app.use("/basic", basic);
  app.use("/board", board);
  app.use("/todo", _passport.authenticate('jwt', {
    session: false
  }));
  app.use("/user", user);
  app.use(morgan('dev'))
  db
    .mongoose
    .connect(mongoURI, {
      useNewUrlParser: true,
      useUnifiedTopology: true
    });
}
```

Routes 풀더의 경우 req, res,next 를 받는 middleware 함수를 사용한다.

```
app.use(function (_req, res, next) {
  res.header(
    "Access-Control-Allow-Headers",
    "x-access-token, Origin, Content-Type, Accept"
  );
  next();
});
```

Service는 확인시켜주는 코드를 console로 확인하고 핸들링하는 코드를 넣어준다.

```
join(req, res) {
  new User(req.body).save(function (err) {
    if (err) {
      res
        .status(500)
        .send({message: err});
      console.log('회원가입 실패');
      return;
    } else {
      res
        .status(200)
        .json({ok: 'ok'});
    }
})
```

Models는 DB에 저장될 스키마와 패스워드와 같은 암호화가 필요한 기능에 대한 코드인 bcrypt를 사용하여준다.

```
const userSchema = mongoose.Schema({
  userId: {type: String, maxlength: 10, unique: 1},
  password: String,
  email: {type: String, trim: true, unique: 1},
  name: String,
  phone: {type: String, maxlength: 15},
  image: String,
  birth: String,
  address: String,
  token: String
}, {timestamps: true})
```

```
userSchema.pre("save", function (next) {
  let user = this
  const saltRounds = 10
  //model 인의 paswword가 변환될때만 암호화
  //if (user.isModified("password")) {
    bcrypt.genSalt(saltRounds, function (err, salt) {
      if (err) return next(err);
      bcrypt.hash(user.password, salt, function (err, hash) {
        if (err) return next(err);
        user.password = hash;
        next();
      });
    });
  //}
})
```

## Python Deep Learning을 활용한 표절, 작사, 작곡프로그램 개발

프로젝트 포지션

PM / PO : 조현국 (Pytorch, Transformer, KOGPT-3) - 작사 프로그램

PL : 권혜민 (Pytorch, MuseGAN) - 표절, 작곡 프로그램

PA : 서성민 (React, TypeScript) - 프론트(화면)

<목차>

- 1.개발 동기
- 2.현 시장분석
- 3.서비스대상자
- 4.사용기술
- 5.데이터베이스
6. 각 프로그램에 따른 개발진행방향
7. 개발진행 중 변경 된 히스토리

### CRP 주소

-도메인 :

<http://crp.kr.s3-website.ap-northeast-2.amazonaws.com>

### -CRP GitHub :

[https://github.com/ChoHyunGook/CRP\\_GPT3-Lyric-Program](https://github.com/ChoHyunGook/CRP_GPT3-Lyric-Program)

### 1. 개발동기

#### 1.1 표절과 저작권 침해

표절은 다른 사람의 저작권으로부터 동의를 받지 않고 내용을 자신의 것처럼 이용하는 행위를 말한다.

-한국 행정학회

“표절을 고의적으로나 또는 의도하지 않았다고 해도 출처를 명확하게 밝히지 않은 채, 타인의 지적재산을 임의로 사용하는 것으로 정의한다.”라고 정의했다.

-문화체육관광부

표절이란 저작권자의 허락 없이 그 저작물을 복제, 공연, 공중송신인 방송, 전송, 전시, 배포 등의 행위를 하는 것으로 공정이용에 해당하지 않는 것을 말한다.

-논문

표절에 대한 판단은 원저작자의 소송이 있어야지만 저작권법적인 측면에서 표절에 대한 판단이 시작된다. 음악저작물의 표절 판단 기준과 표절 관련 소송에 대한 법원의 판례의 기준으로는 크게 창작성, 의거성, 실질적 유사성을 말한다.



CRP (Create Road Program)

출처: 위키백과, 음악저작물 표절 기준에 관한 고찰 : 대중음악을 중심으로(<https://www.koreascience.or.kr/article/JAKO201415640234203.pdf>)

## 1.2 음악 저작권에 대한 표절을 인공지능으로 풀어내는 프로그램을 개발하게 된 이유

현대 사회에 음악 저작권에 대한 표절논란이 많다.

음악에서 표절의 기준은 악보기준 4마디가 동일할 시 표절 즉 저작권 침해라는 이슈가 발생이 되며,

실제 음악저작물 표절판례를 기준으로 음악의 기본요소들 중 박자, 템포, 화성, 화음을 이용하여 악보 이미지 파일을 등록하면 데이터베이스에 저장되어 있는 악보들과 비교분석하여 등록한 악보가 표절인지 판별해주는 프로그램이다.

이 프로그램의 주요 요지는 이미지화 된 악보의 음악적  
기본요소들을 모두 데이터화 시킬 뿐더러 표절에 대한 판별을  
**Python**의 인공지능을 활용하여 사용자가 조금 더 편하게 사용이  
가능하게 만드는 것이다. 사용자의 경우 원하는것을 챗봇을 통하여  
대화가 가능하며 표절된 정보들을 쉽게 알수 있다.

### 1.3 악보분석이 필요한 이유

현대 사회에서 작곡가들 또는 피아노 연주자들이 코드를 직접 수기로 작성하여 확인하며 연주하려는 악보를 연주하려 할 때 필요한 악보의 구성요소들을 해당 프로그램에서 원하는 것들을 채점을 통하여 빠르게 알아볼 수 있다.

#### 1.4 잘 사용하는 그램이 필요한 이유

요즘 현대 사회에선 인공지능으로 이미지처리와 자연어 처리 등이 많은 곳에서 사용되고 있는데 이러한 것을 음악에서의 자연어처리가 사용될 수 있는 작사를 도입하여 해당 음악 즉 악보에서 사용되는 음악의 종류에 따라 편리하게 작사를 하게 하려 할 것이다.

## 1.5 작곡프로그램이 필요한 이유

해당 기존 작곡프로그램들이 많이 나와있지만 사이트 내에서 작곡가들이 전문적으로 사용하고 있는 작곡프로그램을 만들어 우리 사이트 만의 독보적인 작곡프로그램을 알아보려한다.

## 1.6 사업적 출면(기대효과/활성화/촉진전략)

기준에 자신들만의 멜로디와 코드가 적힌 악보 ‘리드시트’를 바탕으로 하는 스포티파이와 달리 간편하게 악보를 이미지파일만으로도 쉽고 간편한 사용을 할 수 있도록 하며,

곡을 발표하는 가수, 음악전공을 하고 있는 대학생, 작곡가 및 엔터테인먼트에서도 사용할 수 있도록 서비스를 제공하여 사업적 측면에서도 뛰어난 아이템이다.

추후 학술지와 논문 그리고 프로그램 개발에 쓰이는 소프트웨어 소스코드 유사도검사같은 표질과 저작권에 연관되어있는 모든 것들을 단 하나의 프로그램으로 컨트롤 할 수 있는 확장성까지 있다.

## 1.8 악보의 구성요소

Nocturne in E-flat Major, Op.9 No.2

Chopin

The musical score consists of two staves. The top staff is in 12/8 time with a key signature of one flat. The bottom staff is in common time with a key signature of one flat. Various performance markings are highlighted with colored boxes and dashed lines connecting them to specific notes or measures. Korean annotations include:  
 - 조표 (Top Staff): 박자표 (Beat Pattern), 불임줄 (Unison Line), 이음줄 (Connected Notes), 꾸밈음 (Grace Notes), 임시표 (Temporary Sharp/Clef).  
 - 높은음자리표 (Top Staff): 빠르기말 (Andante) (♩ = 132), espress. dolce.  
 - 낮은음자리표 (Bottom Staff): 나티냄말 (Nati Nammal), 음표 (Note), 셈여림 (Counting Rest), 세로줄 (Vertical Line).  
 - 물음표 (Bottom Staff): 세로줄 (Vertical Line), 세로줄 (Vertical Line).  
 - 기타 (Other): 뒷줄 (Back Note), 마다 (Mada).

< 쇼팽의 Nocturne in E-flat Major, Op.9 No.2에 사용된 기호, 숫자, 문자들 >

#### ■ 안보이 이해를 돋기워하 표

오선죽

알보의 제일 기초중의 기초인 선이다

세로줄

마디를 나누는 기본 선이다

경세로죽

박자 또는 조표가 변경될 때 구분짓는 선이다

자바에서 객체의 초기화와 같은 개념이다

도돌이표

제한을 두여 일정 마디를 반복하게 하는 선이다

프로그래밍 언어에서 쓰이는 Loop 중 for와 같은 개념이다.

끝세로줄

해당 악보를 끝내는 선이다

## 보표

작은보표: 오선줄 하나를 칭하는 표이다.

큰보표: 오선줄 두개가 묶여있는 표이다.

모음보표: 대합창곡에 쓰이며 오선줄 4개묶음의 표이다.

## 묶음표

보표를 묶을 때 사용하는 표이다.

## 음자리표

높은음자리표: 첫 음의 높이를 나타내기 위한 자리표이다.

낮은음자리표: 높은음자리표 음의 높이 기준으로 F(라)를 도로 잡는 자리표이다.

## 박자표

일정한 박자를 단위로 나누어 분수로 표현해놓은 것이다.

## 조표

샵 (#) : 음을 높이는기호이다.

플렛(b) : 음을 낮추는 기호이다.

제자리표: 샵 또는 플렛으로 변화된 음을 되돌릴때 사용하는 기호이다.

## 임시표

조표들이 음표앞에 붙어 임시적으로 음을 높이거나 낮추는것을 임시표라한다.임시표가 들어있는 마디안에서만 흐력이 있다.

## 음표

흔히 말하는 게이름이라고도하며 음의 길이와 높낮이를 표현하는 표이다.

## 쉼표

음을 내지않고 쉬는 길이를 표현하는 표이다.

## 점 음표, 점 쉼표

음표와 쉼표 옆에 점이 찍혀있는 것이며, 점이 찍혀있는 음표와 쉼표의 길이의 반을 더 가지게 되어 2박을 가진 음표 또는 쉼표일 경우 3박을 가지게 된다.

## 덧줄

한 오선줄위에 쓸수있는 음은 총 11개의 음을 표현할수있으며, 그이상의 음을 표현하기 위해 오선줄의 밑과 위에 줄을 긋는것이다.

덧줄 수의 제한은 없지만 대체적으로 한눈에 알아보기 쉽도록 5개이내로 사용하며, 그이상의 경우 옥타브기호를 사용한다.

## 붙임줄

높이가 같은 두음을 연결한 것으로 한음으로 연주하게 하는 줄이다.

## 이음줄

다른 음을 이어서 연결하여 부드럽게 연주하게 하는 줄이다.

## 화음

두개이상의 음이 동시에 소리내는것을 화음이라 하며, 음표2개가 붙어있다.

## 화성

코드라고도 하며 영어의 속어와 문법과 같이 일정한 법칙에 따라 약속된 음들을 코드화 시켜놓은것이다.

## 악상의 종류

나타냄말 : 악곡의 표현을 나타내는 표시이다.

대표적인 종류로 **Amabile** 사랑스럽게, **Brilante** 화려하게, **Cantabile** 노래하듯이, **Dolce** 부드럽게 등의 종류가 있다.

셈여림표: 음의 세기를 나타내는 표이다.

대표적으로 **pianissimo** 매우여리게, **mezzopiano** 조금여리게, **forte** 세게가 있다.

연주 중 셈여림을 변화시키는 기호

대표적으로 **crease**(크레센도) 점점세게, **decrease**(데크레센도) 점점약하게, **Accent**(악센트)로 그 음만 특히 세게, 스타카토 같은 짧게 끊어서 연주하는 것 등이 있다.

빠르기말: 악곡의 속도를 조절하는 것이다.

대표적으로 **Adagio** 아주느리게, **Andante** 느리게, **Allegro** 빠르게, **Vivace** 아주빠르게 가 있다.

연주중 빠르기를 변화시키는 기호

대표적으로 **accel**(아첼레란도) 점점빠르게, **rit**(리타르단도) 점점느리게, **atempo**(아템포) 원래빠르기로 등이 있다.

## 2. 현 시장분석(조사)

### 2.1 사례 (표절논란 이슈들)

#### 2.1.1 Sky 캐슬 드라마 **We All Lie**

출처:[http://star.ohmynews.com/NWS\\_Web/OhmyStar/at\\_pg.aspx?CNTN\\_CD=A0002508487](http://star.ohmynews.com/NWS_Web/OhmyStar/at_pg.aspx?CNTN_CD=A0002508487)

#### 2.1.2 다비치

사랑과 전쟁: 이토 유나의 pureyes의 후렴구와 이 곡의 후렴구가 거의 비슷하다. 더불어 오송의 지못미라는 노래와 표절의혹이 있기도 하다.

8282: 이 노래의 도입부가 MIKA의 Happy Ending의 도입부와 비슷하다는 의혹이 있다.

### 2.1.3 로이킴

봄봄봄: 어쿠스틱 레인의 love is canon이라는 곡과 멜로디가 흡사하다

출처:

<https://namu.wiki/w/%ED%91%9C%EC%A0%88/%EC%9D%98%ED%98%B9>

### 2.1.4 너에게 쓰는편지

표 4. 이 사건의 각 대비 부분의 화성 진행

\* C장조 변형 후 비교

|      | 1소절 |    | 2소절  |           | 3소절 |    | 4소절 |   |
|------|-----|----|------|-----------|-----|----|-----|---|
| 원고   | G   | C  | C    | Am        | F   | Dm | Em  |   |
| 피고   | G   | C  | Bdim | C/E       | F   | C  | Em  |   |
| 동일여부 | 동일  | 동일 | 유사   |           | 동일  |    | 동일  |   |
|      | 5소절 |    | 6소절  |           | 7소절 |    | 8소절 |   |
| 원고   | F   | G  | C    | G         | Am  | F  | Ab  | F |
| 피고   | Dm  | G  | C    | Gm/<br>Bb | A7  | Dm | F   | G |
| 동일여부 | 유사  | 동일 | 동일   | 유사        | 유사  | 유사 | 유사  |   |

자료: 수원지방법원 제6민사부, 2006. 10. 20. 판결, 사건 2006가합8583  
손해배상(기)

--"너에게쓰는편지" 표절사건의 표 -

그룹 '더더'가 가창한 'It's you'와 'MC몽'과 '린'이 함께 부른 '너에 게 쓰는 편지'의 곡 후렴구 8소절을 그대로 표절하거나 일부 변형하여 사용하였다고 소송을 제기하였다.

출처:

<https://www.koreascience.or.kr/article/JAKO201415640234203.pdf>

## 2.2 악보분석

악보의 상세 분석이 되는 프로그램은 현재 시장조사로 없으며 연주하는 사람들과 작곡하는 사람들이 직접 수기로 화성과 분석을 하고 있다.

## 2.3 작사 프로그램

현재 NLP로 작곡프로그램을 개발한 포자랩스가 있으나, 인공지능으로 작사서비스를 하고 있는 곳은 크게 없는 것으로 확인된다.

## 2.4 작곡프로그램

대표적으로 Chrome Music Lap 과 같이 사이트 내에서 다양한 기능으로 작곡을 할 수 있게 구현해놓은 프로그램들이 있다.

링크: <https://musiclab.chromeexperiments.com/>

## 2.5 참고자료 (논문, 문헌)

### 2.5.1 스포티파이

스포티파이의 표절 방지 기술은 멜로디와 코드가 적힌 악보인 '리드 시트'를 바탕으로 표절 여부를 판별한다. 표절 여부를 확인하고 싶은 리드 시트와 데이터베이스에 담긴 노래들의 시트를 비교한다.

출처: <http://www.aitimes.com/news/articleView.html?idxno=134588>

### 2.5.2 음악저작물 표절 기준에 관한 고찰 : 대중음악을 중심으로

표절에 대한 사건 중 표절이라 판결난 예시들을 들어 표절 판단기준을 잡게 한다.

-미래창조과학부(MSIP)의 재원으로 한국연구재단의 지원을 받은 논문

출처: <https://www.koreascience.or.kr/article/JAKO201415640234203.pdf>

### 2.5.3 opencv를 이용한 악보인식 티스토리

악보인식을 위한 알고리즘들을 1~12로 나누어 기재하였다.

출처: <https://hackids.tistory.com/120>

### 2.5.4 Hierarchical ART2 알고리즘을 이용한 다른 논문

Hierarchical ART2 알고리즘을 이용하여 악보인식을 한 논문들이다.

출처1: <https://scienceon.kisti.re.kr/commons/util/originalView.do?cn=CFKO200816263465677&oCn=NPAP08297839&bt=CFKO&journal=NPRO00293702>

출처2: <https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiView.kci?sereArticleSearchBean.artId=ART001388285>

## 2.5.5 영상처리와 딥러닝을 이용한 악보코드 변환프로그램

해당 논문에서는 opencv 와 MIDI파일을 사용하여 악곡 연주도 가능하게끔 사용하였으며, 해당 논문을 이용하여 분석에서 쓰이는 악곡연주를 개발한다.

출처:<https://www.koreascience.or.kr/article/JAKO202111037332612.pdf>

## 2.5.7 악보연주기능에 대한 파이썬 알고리즘

Music21 라이브러리를 사용한 코드를 분석해보았다.

출처:

<https://inspiringpeople.github.io/data%20analysis/midi-music-data-extraction-using-music21/>

## 2.5.8 영상처리를 통한 악보인식 및 재생

C++,opencv2.4.8버전을 사용한 발표자료이다.

출처:[http://117.16.143.51/Class/2014/14CD1/%EC%98%81%EC%83%81%EC%B2%98%EB%A6%AC%EB%A5%BC%20%ED%86%BC%ED%95%9C%20%EC%95%85%EB%B3%BC%4%EC%9D%BC%8B%9D%20%EB%BC%8F%20%EC%9E%AC%EC%83%9D\\_%EC%BC%9C%EC%A2%85%EB%BC%9C%ED%91%9C.pdf](http://117.16.143.51/Class/2014/14CD1/%EC%98%81%EC%83%81%EC%B2%98%EB%A6%AC%EB%A5%BC%20%ED%86%BC%ED%95%9C%20%EC%95%85%EB%B3%BC%4%EC%9D%BC%8B%9D%20%EB%BC%8F%20%EC%9E%AC%EC%83%9D_%EC%BC%9C%EC%A2%85%EB%BC%9C%ED%91%9C.pdf)

## 2.5.9 자바스크립트에 음악을 입히는 작업분석

현업에서 쓰이는 작곡프로그램을 자바스크립트랑 연동시켜주는 영상이다.

출처: <https://www.youtube.com/watch?v=ahTs3efFzFI>

## 2.5.10 Transfomer 논문

링크: <https://youtu.be/AA621UofTUA>

## 2.5.11 GPT-3

Few-Shot,One-Shot,Zero-Shot Learning 에 대한 블로그이며, 장점과 단점이 기재되어있다.

출처:<https://blog.naver.com/PostView.nhn?blogId=dh0985&logNo=222321560055>

## 3. 서비스 대상자

각종 음악분야 엔터테인먼트, 작곡가, 대학생, 교수, 연구원

## 4. 사용하는 기술

-개발언어: PYTHON(3.8 ver), JAVASCRIPT(es6 ver)

-사용환경 및 STACK: PYTORCH(1.8.0), GAN, YOLO, TRANSFORMER(4.20.1), GPT-3, Google Colab(pro), MUSEGAN

-사용하는 프로그램:

VsCode(1.66.2 ver), Node.js(16.13.0 ver), MongoDB(5.0.6 ver), React.js(18.0.0 ver), Pycharm(ver), , MariaDB(ver)

-UI/UX Design 협업 Tool : JustInMind

## 5. DATABASE(MariaDB)

딥러닝 모델이 무게가 무겁다보니 AWS 및 개발환경의 한계로 디스크에서 돌리기로 하였으며, numpy로 바꾼 데이터를 MariaDB를 활용하여 사용한다.

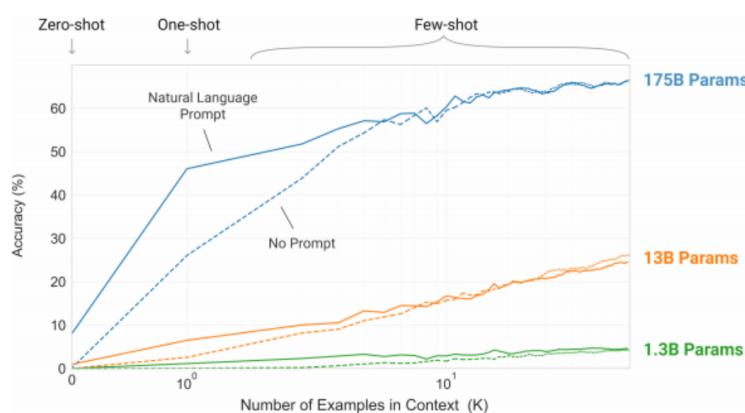
## 6. 각 프로그램 개발 진행방향

### 6.1 작사프로그램 -조현국

<작사프로그램>

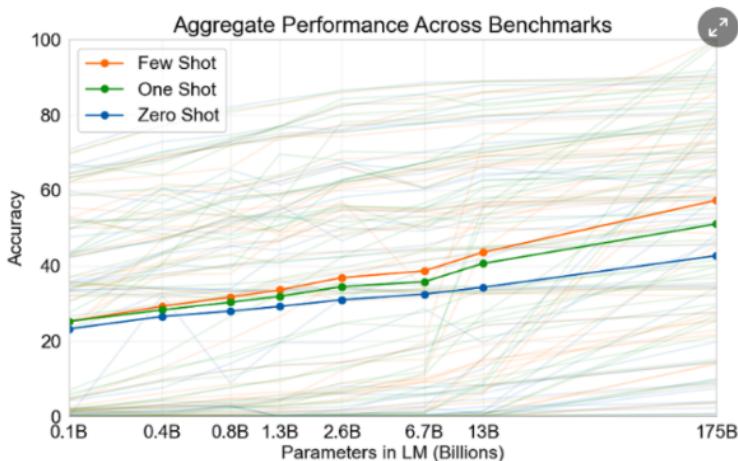
사용하는 모델: **KOGPT-3**

KOGPT-3를 사용한 이유는 기존 GPT-2의 800만개의 데이터셋과 15억개 파라미터를 뛰어넘어 3000억개의 데이터셋과 1750개의 파라미터를 가져 Fine-Tuning에서 필요로 하는 많은 데이터셋과 그 데이터셋을 학습시키는 시간 대신 몇 개의 Sample Data만 주어져도 좋은 성능을 낼수 있는 Few-Shot-Learning이 있으며, 한가지 샘플로 학습을 시키는 One-Shot-Learning, 한가지 데이터 즉 샘플을 자연어로 설명하여도 학습을 하는 Zero-Shot-Learning 등 성능 면에서 뛰어나다.



하지만 KoGPT-3의 단점도 당연히 있다. 많은 데이터를 Pre-Train이 되어있다보니 확실히 거대한 PLM으로 개발하는데 높은 환경을 필요로 하고 인터넷에서 떠도는 데이터셋들로 학습하다보니 성차별, 인종차별 발언도 포함이 되어있으며, GPT-3은 범용적인 다양한 문제를 잘 풀어나갈 뿐 문제들의 최적화 되어있는 모델들에 비해 성능은 떨어진다. 그리고 GPT-3의 기반인

Transformer 구조에 기억이라는 개념이 존재하지 않아 현재의 문맥을 입력으로 주어야 그에 따른 출력을 얻는다.



## 6.2 표절 프로그램 -권혜민

사용되는 모델: **GAN**

이미지 인식에 사용되는 알고리즘: **YOLO**

<표절기능>

등록한 악보이미지의 표절판독과 표절되는 부분을 표절이 아니게끔 편곡을 해주고 음악이 연주되게 한다. 이때 사용되는것들은 딥러닝 GAN과 midi로 전환되게 해주는 Music21 라이브러리가 사용된다.

## 6.3 작곡(편곡) 프로그램 -권혜민

사용되는모델: **MuseGAN**

midi 파일로 되어있는 데이터셋을 통해 또 다른 피아노를 생성시키고 피아노, 기타, 드럼, 베이스등 다른 롤을 추가시켜서 생성할 수 있다 생성되어진 파일은 midi 파일이다

## 6.4 프론트 - **TypeScript** 서성민

사용된 기술스택 : **JavaScript, TypeScript, React, Next.js, Redux, Redux-Saga, HTML, CSS, AWS**

CRP팀 프로젝트의 총 화면을 제작하였고 세부 화면으로는 표절, 분석, 작사 기능에 관한 파일업로드 기능을 구현하였고, HTML구조로 짜여져 있는 mp3플레이어에 대한 OpenSource를 가지고 React구조로 코드화하여 구현하였다. 또한 Tone이라는 Web Audio Framework를 이용하여 Piano를 화면에서 연주할 수 있게 제작하였다.

## 7. 개발진행 중 변경 된 히스토리

**7.1** 표절에 관한 프로그램 개발을 종점으로 해당 프로젝트가 시작되었다. 마지막 프로젝트에 대한 큰 구상만 그려놓은 상태로 파이썬, 자바, 자바스크립트, 리눅스로 인원 구성을 하여 해당 프로젝트가 시작되었다.

**7.2** 악보 표절에 관한 알고리즘과 구성요소들을 알아보며 정보들을 알아보고 수업을 진행하면서 인원들을 각각 다른 언어로 배운 하다보니 기능의 추가가 필요하여 악보분석창이 추가되었다. 구상 중인 것들이 어떠한 기술이 필요한지 알지 못하여 일단 각 version별로 악보가 읽어내는것을 기입하였다. 초기에 준비하고 만든 발표 자료의 링크이다.

링크:

<https://docs.google.com/document/d/17Gsh8NZaLnFXm9T4KEqcwb5Pt7NI69hTT6nw0A5j3Q/edit>

**7.3** 강사님의 피드백으로 작곡프로그램 기능 개발을 알아보다가 Java Spring으로 백단을 구성하지 않아도 된다는 점을 알게 되었으며, 다른 반의 파이썬 발표를 듣고 난 후 3명이 모두 파이썬 기능을 하자는 말이 나와 서버를 django로 연결하며, melon과 shazam과 같은 음성검색 기능을 추가하기로 하고 해당 서비스 중인 것에서 차별점을 두기 위하여 검색된 노래의 장르 또는 음률이 비슷한 노래를 추천해주는 추천 알고리즘을 넣기로 했다.

**7.4** 각 기능과 협업 tool인 JustInMind로 화면을 발표하고 피드백을 받은 챗봇이라는 피드백을 반영하여 챗봇 기능을 추가하기로 하였다. 그리고 다른반 파이썬 발표 때 사용한 GPU사양과 사용 갯수를 듣고 현실적으로 표절과 분석 프로그램에 쓰일 악보를 머신러닝하는데 한계점을 느끼게 되었고 해당하는 논문들을 읽다보니 기존에 모으려던 DATABASE 악보 전체가 아닌 2줄~3줄로 줄여 현실적인 개발환경을 만들기로 하였으며 라이브러리를 opencv와 yolo를 사용한다는것을 알게 되었다.

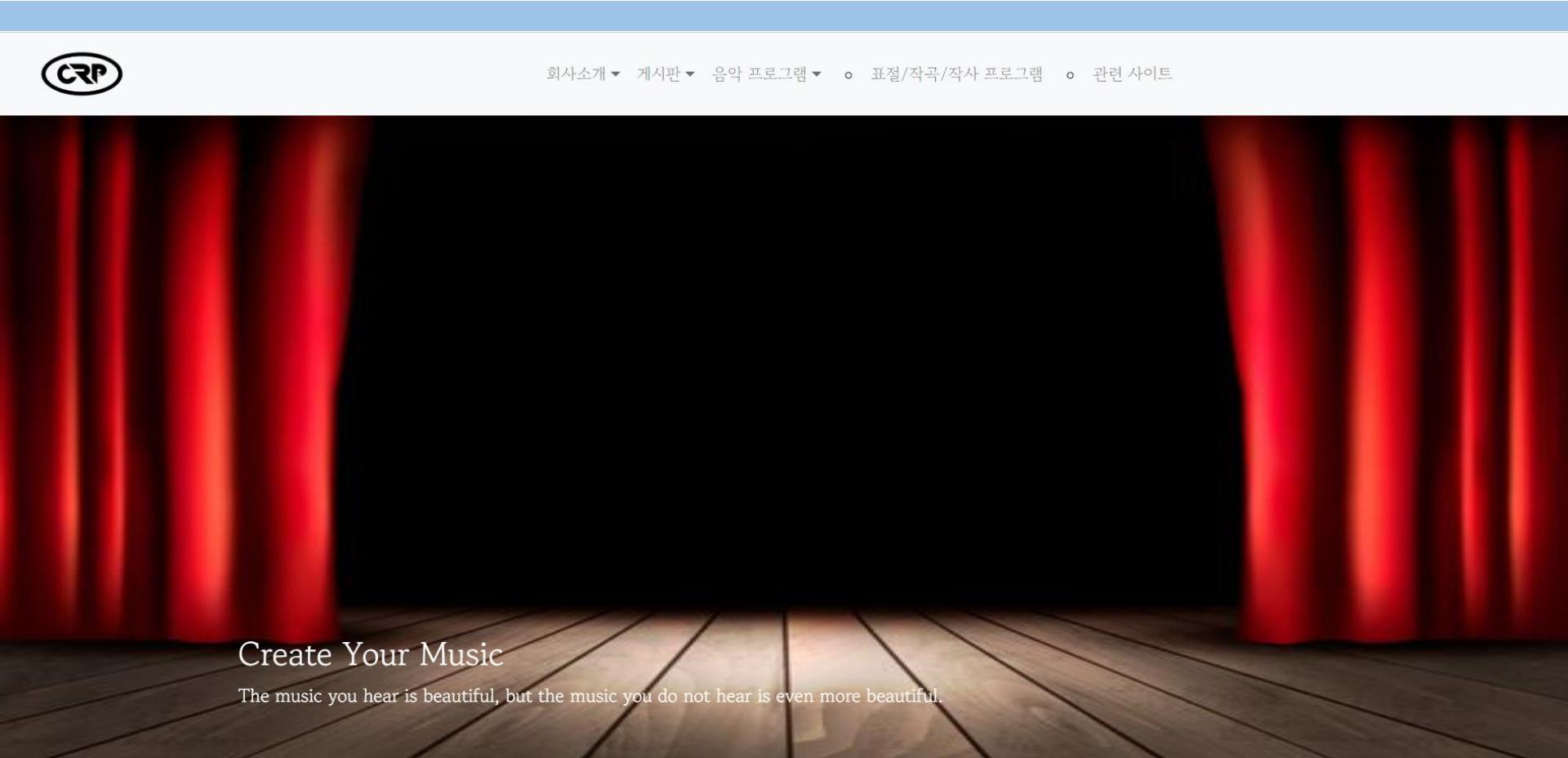
**7.5** 표절과 악보분석을 다른 기능으로 나누어 조현국, 권혜민이 각각 진행하려고 준비하고 있었으나 표절과 악보분석에 공통되는 이미지추출과 챗봇이라는 기능을 다시 나누어 각 페이지마다 사용할 수 있는 챗봇기능과 표절 분석에서 사용하려는 보이스봇에 대한 기능은 조현국이 맡아서 진행하기로 하고, 이미지를 추출하고 알고리즘을 적용하는 표절과 악보분석은 권혜민이 맡기로 했다.

**7.6** 처음 이 프로그램의 이름을 CopyRightProgram으로 표절을 기반으로 만든 제목을 지었으나 작곡, 음원검색, 챗봇등 다른 기능들이 추가로 진행되면서 음악의 모든것을 통용하는 프로그램 이름으로 변경을 회의를 하여 CreateRoadProgram으로 수정하기로 하였다. 도메인을 crp.kr로 지정한 후이고 결제까지 해놓은 상태라 CRP안에서 여러가지 뜻을 찾다가 CreateRoad와 CantusRegnum(라틴어로 노래왕국) 중 회의 결과 CreateRoad로 결정하였다.

**7.7** 6월 28일 멘토님과 첫만남에서 가닥이 잡혔다. 악보 업로드 후 midi파일로 변환하는 Music21을 들었으며 각 악보마다 numpy배열로 불러온 데이터를 코사인유사도를 돌려 표절인지 판별하기로 하였으며 음악 추천검색은 MP3파일을 넘파이로 라벨링하여 기존 MP3파일에 라벨을 붙여 검색이 가능하도록 이야기 했으며 작곡프로그램에서 넘어온 결과값을 GAN을 사용하여 편곡을 해보기로 하였다.

**7.8** 7월 11일 강사님과의 회의와 12일 멘토님과의 두번째 회의로 남은 시간안에 현실적인 마지막 프로젝트 완성을 위해 세부적인 내용들과 빼야할 것들을 정리하였다. 읽기 조금더 쉬운 동요악보로 모두 통일 하기로 했으며 **numpy**배열로 넘어오는 작곡프로그램의 피아노 코드를 정했으며 **GAN**이 활용되어 돌아가는 것을 동영상 촬영을 하여 발표하기로 했으며 보이스봇에게 물어볼 품을 정하였다. 작곡프로그램이 있어 작사프로그램도 추가하기로하여 **GPT-3**모델로 구현하기로 하고 슬픔과 기쁨의 감정표현에 따라 동요 가사가 만들어지게 하기로 하였다. 수업이 끝나고 프로젝트 기간이 한달이 남았는데 음악 검색프로그램까지 진행하기가 어려울 것 같아 빼게 되었다.

**7.9** 생각보다 파이썬 수업이 지체되어 표절에서 쓸 보이스봇 및 챗봇은 조현국의 개인프로젝트로 하기로 하며, 작사프로그램을 구현 후 **GPT-3** 논문 및 취직 준비를 하기로 하였다. **AWS**에 딥러닝 모델들을 돌리기 힘들어 디스크 내에서 돌리기로 한 후 영상을 찍어 발표를 하기로 하였다. **server**에서 받아 화면에 표출해야해서 **FastAPI**로 중간 서버를 사용하기로 하였으며, 프로젝트의 마무리단계로 진입하였다.



Create Your Music

The music you hear is beautiful, but the music you do not hear is even more beautiful.

# 딥러닝을 활용한 작곡 작사 표절 프로그램

TEAM: CRP  
조현국, 권혜민, 서성민

# 목차

1. 프로젝트 개요
2. 팀 구성 및 역할
3. 프로젝트 수행 절차
4. 개발 언어 및 툴
5. 화면 - 프론트엔드
6. 표절 프로그램
7. 작곡 프로그램
8. 작사 프로그램

# | 프로젝트 개요



음원 표절 논란이 많은 현대시대에

파이썬 딥러닝 기술로

“음원표절을 잡아내는 **표절프로그램**이 있으면 어떨까?”

“그렇다면 **작곡과 작사** 또한 할 수 있지 않을까?”

“또 그렇다면! **하나의 웹사이트** 내에서 표절, 작곡, 작사  
음악에 모든 것들을 담아 낼 수는 없을까?”

이 세가지의 궁금점으로 시작한 프로젝트입니다.

# | 팀 구성 및 역할

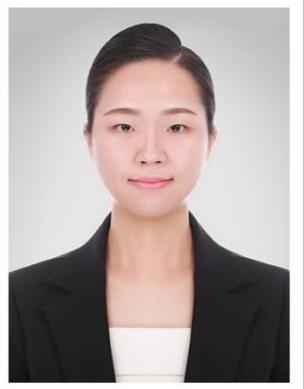


PM/PO

조 현국

Lyric Programmer

Python / NLP / Pytorch



PL

권 혜민

Composition Programmer

Python / GAN / Pytorch



PA

서 성민

Front Programmer

JavaScript / TypeScript / REACT

# | 프로젝트 수행 절차

| 구분             | 기간                      | 활동  | 비고                           |
|----------------|-------------------------|---|------------------------------|
| 사전 기획          | ▶ 4/25(월)~5/2(월)        | ▶ 프로젝트 기획 및 주제 선정<br>▶ 기획안 작성               | ▶ 아이디어 선정                    |
| 데이터 수집         | ▶ 5/3(화)~5/10(화)        | ▶ 필요 악보, mp3파일 수집<br>▶ 외부 데이터 수집            | ▶ 필요한 DB 수집                  |
| 기본화면 구성        | ▶ 5/10(수)~5/26(목)       | ▶ 화면 기본 구성을 REACT로 구현                       |                              |
| 파이썬 모델 찾기 및 학습 | ▶ 05/27(월)~7/24(일)      | ▶ 파이썬 모델 구현을 위한 학습                          | ▶ 팀별 중간보고 실시<br>화면 TYPE으로 전환 |
| 프로젝트 서비스 구축    | ▶ 7/25(월)~8/8(월)        | ▶ AWS 및 학습된 모델 정리<br>▶ 프로젝트 발표영상 준비 및 서류 준비 | ▶ 최적화, 오류 수정                 |
| 총 개발기간         | ▶ 4/25(월)~8/8(월)(총 15주) | -   | -                            |

# | 개발언어 및 라이브러리



JavaScript



python



TypeScript



PyTorch



TensorFlow



NumPy



pandas



matplotlib



OpenCV



React



scikit-learn



Redux

# |화면 (FrontEnd) – Next.js + TypeScript

- TypeScript와 Next.js를 베이스로 화면을 구현함.
- React 기반으로 음악 컨텐츠를 중심으로 필요한 기능과 화면을 구현함.

NEXT.js + TypeScript

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판에 대한 Component와 Pages 부분

```
import { Article } from '@/modules/types'
import React from 'react'

type Props = {
    onChange : (e: React.FormEvent<HTMLInputElement> | any ) => void
    onSubmit : (e: React.FormEvent<HTMLFormElement> ) => void
}

const AddBoard: React.FC<Props> = ({onChange, onSubmit}) => {
    const date = new Date();
    const parseDate = date.toDateString()
```

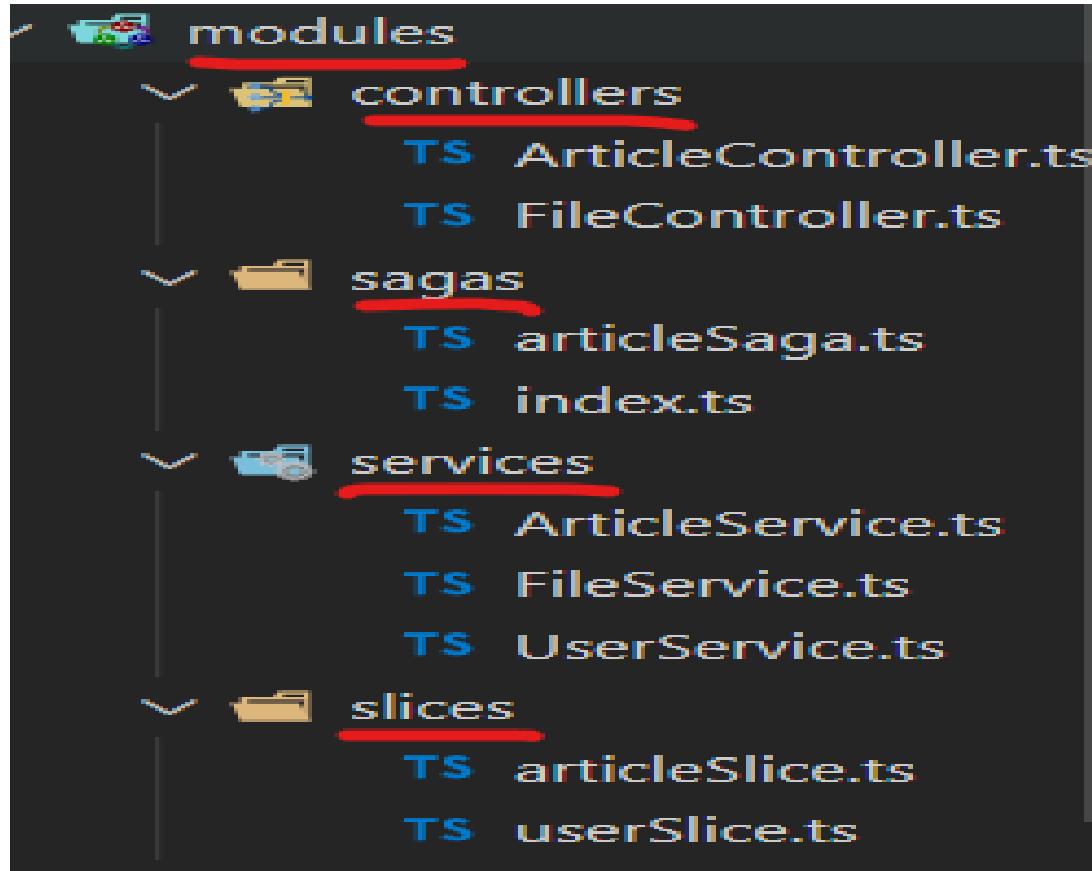
```
import { Article } from "@/modules/types";
import React, { useState } from "react";
import Image from "next/image";
import { musicData } from "@/modules/types";
export interface Props {
    datas: Article[];
    onDeleteClick: any;
}

const AllBoardList: React.FC<Props> = ({ datas, onDeleteClick }: Props) => {
    const [text, setText] = useState([
        {
            id: 1,
            title: "CRP",
            content: "Music is my life",
        },
        {
            id: 2,
            title: "CRP Team",
            content: "Enjoy your Life!",
        },
    ]);
}
```

# |화면 (FrontEnd) – TypeScript + Next (component)

게시판

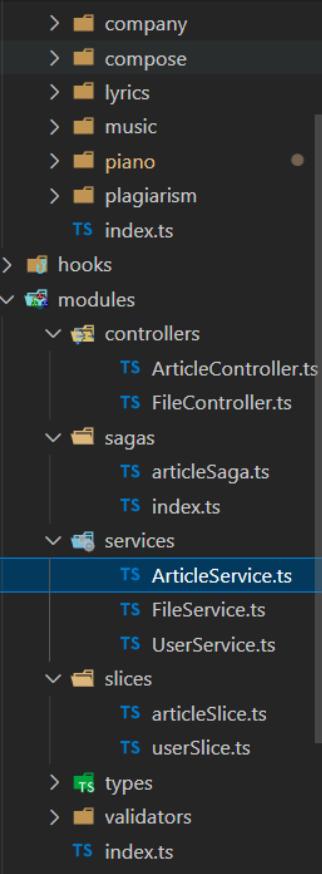
- Redux 부분 구조 및 내부 구조



# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux 부분 내부 코드 과정 1



```
1 import { Article, ArticleState } from "@/modules/types";
2 import { createSlice, PayloadAction } from "@reduxjs/toolkit";
3
4 export class ArticleService {
5   public createArticleSlice() {
6     const initialState: ArticleState = {
7       data: {
8         id: 1,
9         title: "",
10        content: "",
11        open: "",
12        picture: null,
13        writtenDate: "",
14        pictureName: ""
15      },
16      status: "loading",
17      error: null,
18    };
19    return {
20      name: "articleSlice",
21      initialState,
22      reducers: {
23        writeArticle: (state: any, action: PayloadAction<Article>) => {
24          alert(`게시글 작성 액션 요청`);
25          console.log(action);
26          state.data = action.payload;
27          state.status = "loading";
28          console.log(
29            `게시글 작성 성공 - 리듀서 ${JSON.stringify(state.data)}`
30          );
31        }
32      }
33    };
34  }
35}
```

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux 부분 내부 코드 과정 2

The screenshot shows a code editor with a dark theme. On the left is a file tree for the 'CRP-FRONT' project. The 'slices' folder contains two files: 'articleSlice.ts' (selected) and 'userSlice.ts'. The main pane displays the content of 'articleSlice.ts'.

```
modules > slices > TS articleSlice.ts > ...
1 import { Article, ArticleState } from "@/modules/types";
2 import { createSlice, PayloadAction } from "@reduxjs/toolkit";
3 import { ArticleService } from "../services/ArticleService";
4
5 const articleService = new ArticleService();
6 const ArticleSlice = createSlice(articleService.createArticleSlice());
7
8 export const {
9   writeArticle,
10  writeArticleSuccess,
11  writeArticleFailure,
12  fetchArticles,
13  fetchArticleSuccess,
14  removeArticle,
15  fetchMyArticle,
16  writeComment,
17 } = ArticleSlice.actions;
18 const { reducer, actions } = ArticleSlice;
19 export const ArticleActions = actions;
20 export default reducer;
21
```

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux부분 내부 코드 과정 3

```
1 import {FileController} from "./controllers/FileController"
2 import { ArticleController } from "./controllers/ArticleController";
3 import {FileService} from "./services/FileService"
4 import {ArticleService} from "./services/ArticleService"
5 import articleSaga from "./sagas"
6 import FileValidator from "./validators";
7
8 export { FileValidator, FileService, FileController, ArticleController, ArticleService, articleSaga };
```

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux 부분 내부 코드 과정 4

```
const rootReducer = (
  state: ReturnType<typeof combinedReducer>,
  action: AnyAction
) => {
  if(action.payload === HYDRATE) { // action.type => action.payload 07-14
    return{
      ...state, // use previous state
      ...action.payload // apply delta from hydration
    }
  } else {
    return combinedReducer(state,action)
  }
}
const makeStore = () =>{
  const store =
  configureStore({
    reducer:{ rootReducer },
    middleware: (getDefaultMiddleware) =>
    getDefaultMiddleware({serializableCheck: false})
    //직렬화 문제 발생 시 {serializableCheck: false} 파라미터로 전달
    .prepend(sagaMiddleware)
    .concat(logger),
    devTools : isDev
  });
  sagaMiddleware.run(rootSaga)
```

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux 부분 내부 코드 과정 5

The screenshot shows a code editor with the following file structure:

- modules > sagas > **articleSaga.ts** (highlighted with a red underline)
- modules > controllers > ArticleController.ts
- modules > controllers > FileController.ts
- modules > sagas > index.ts
- modules > services > ArticleService.ts
- modules > services > FileService.ts
- modules > services > UserService.ts
- slices > articleSlice.ts
- slices > userSlice.ts
- types > index.ts

The content of **articleSaga.ts** is as follows:

```
1 import { ArticleController } from "@/modules/controllers/ArticleController";
2 import { call, put, takeEvery, takeLatest } from "redux-saga/effects";
3 import { ArticleActions } from "../slices/articleSlice";
4 import { Article } from "../types";

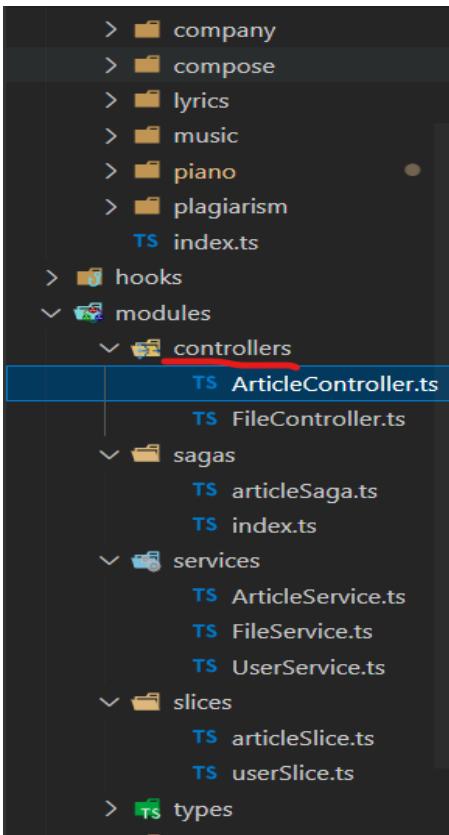
5
6 //get Saga
7 function* writeArticleSaga(action: { payload: Article }) {
8   const { writeArticleSuccess, writeArticleFailure } = ArticleActions;
9   const param = action.payload;
10  const articleController = new ArticleController();
11  try {
12    yield call(articleController.writeArticle, param);
13    yield put(writeArticleSuccess());
14  } catch (error) {
15    yield put(writeArticleFailure());
16  }
17}

18
19 function* fetchMyArticleSaga(action: { payload: any }) {
20   const { fetchMyArticleSuccess, fetchMyArticleFailure } = ArticleActions;
21   try {
22     const response: Article = yield call(action.payload);
23     yield put(fetchMyArticleSuccess(response));
24   } catch (error) {
25     yield put(fetchMyArticleFailure());
26   }
27}
```

# |화면 (FrontEnd) – TypeScript + Next (component)

## 게시판

- 게시판 관련 Redux 부분 내부 코드 과정 6



```
1 < import { Article } from "@/modules/types";
2 import axios, { AxiosResponse } from "axios";
3 import {HOST_4000} from "@/components/common/Path"
4
5 const headers = {
6   "Content-Type": "application/json",
7   Authorization: "JWT fefege...",
8 }
9 export class ArticleController {
10
11
12   async writeArticle(writeData: Article) : Promise<any> {
13     try {
14       await axios.post(`${HOST_4000}/Article`, writeData, {headers})
15     } catch (err) {
16       return err;
17     }
18   }
19
20   async removeArticle (id: any ) : Promise<any> {
21     try{
22       await axios.delete(` ${HOST_4000}/Article/${id}` , {data : id} )
23     } catch (err) {
24       return(err);
25     }
26   }
27 }
```

# |화면 (FrontEnd) – TypeScript + Next (component)



## ■ 게시판 화면 구현

나만의 게시글

이미지도 업로드하고 게시글도 작성해보세요

게시글 등록하기

Mon Aug 08 2022

제목 입력

파일 선택

선택된 파일 없음

게시글 작성란

UPLOAD

# CRP 게시판 #

CRP

Music is my life

삭제

CRP Team

Enjoy your Life!

삭제

Copyright © CRP WebSite 2022

개발자들 사이트 소개 관련사이트

# |화면 (FrontEnd) – TypeScript + Next (component)

## FileUpload

- component에 상태를 만들어 주어 파일 업로드에 관한 기능 구현

```
const PlUpload: React.FC<Props> = ({ onSubmit, onSubmit1 }: Props) => {  
  const [uploadFormError, setUploadFormError] = useState<string>("");  
  
  const handleFileUpload = async (element: HTMLInputElement) => {  
    const file = element.files;  
  
    if (!file) {  
      return;  
    }  
  
    const validFileSize = await validator.validateFileSize(file[0].size);  
    const validFileType = await validator.validateFileType(  
      FileService.getFileExtension(file[0].name)  
    );  
  
    if (!validFileSize.isValid) {  
      setUploadFormError(validFileSize.errorMessage);  
      return;  
    }  
  
    if (!validFileType.isValid) {  
      setUploadFormError(validFileType.errorMessage);  
      return;  
    }  
  
    if (uploadFormError && validFileSize.isValid) {  
      setUploadFormError("");  
    }  
  
    const fileController = new FileController(file[0]);  
    const fileUploadResponse = await fileController.uploadFile();  
  };  
};
```

# |화면 (FrontEnd) – TypeScript + Next (pages)

## FileUpload

- 파일업로드에 관한 페이지 구현

```
const onSubmitFile = async (e: React.FormEvent<HTMLFormElement>) => {
  e.preventDefault();
  window.location.href = `${HOST_3000}/plagiarism/plagiarism`;
};

const fixonSubmitFile =  async (e: React.FormEvent<HTMLFormElement>) => {
  e.preventDefault()
}

const PlUploadPage: NextPage = () => {

  useEffect(()=> {
  },[])

  return (
    <Plupload onSubmit = {onSubmitFile} onsubmit1={fixonSubmitFile}>
  )
}
export default PlUploadPage
```

# I화면 (FrontEnd) – 파일 업로드 화면 구현 과정

## FileUpload

- 파일 업로드 구현 화면 1



회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트

원본용 약보를 업로드하세요

파일 선택 선택된 파일 없음

약보 등록

표절여부를 확인하고 싶은 약보를 업로드하세요

파일 선택 선택된 파일 없음

약보 등록



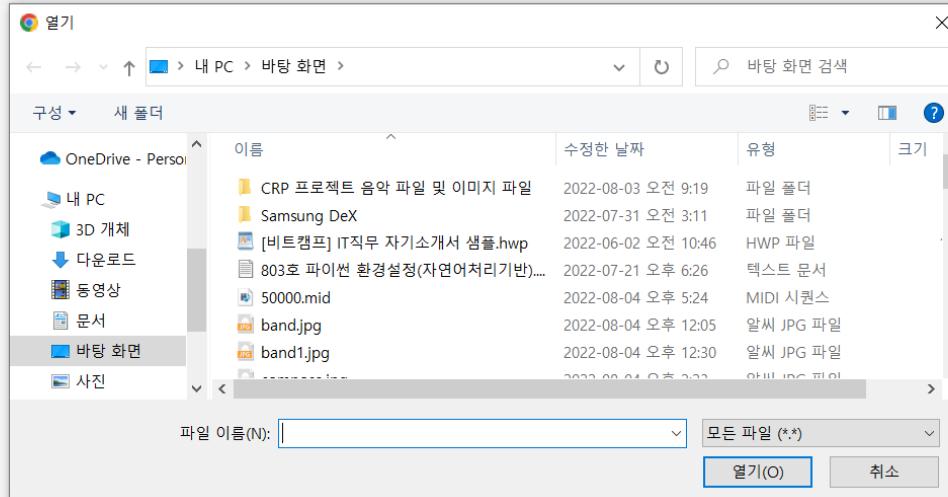
# I화면 (FrontEnd) – 파일 업로드 화면 구현 과정

## FileUpload

- 파일 업로드 구현 화면 2



회사소개 ▾ 게시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트



표절여부를 확인하고 싶은 악보를 업로드하세요

파일 선택 선택된 파일 없음

악보 등록

# I화면 (FrontEnd) – 파일 업로드 화면 구현 과정

## FileUpload

- 파일 업로드 구현 화면 3



회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트

원본용 약보를 업로드하세요

파일 선택 선택된 파일 없음

약보 등록

표절여부를 확인하고 싶은 약보를 업로드하세요

파일 선택 school.jpg

약보 등록



# |화면 (FrontEnd) – 파일 업로드 화면 구현 과정

## FileUpload

- 파일 업로드 구현 화면 4



회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트

Score

Score

학교 종 이 땡땡땡



표절판독하기

# |화면 (FrontEnd) – 파일 업로드 화면 구현 과정

## FileUpload

- 파일 업로드 구현 화면 6

Score



Score

학교 종이 땡땡땡



표절판독하기

검사 결과 : 표절

하이라이트 된 부분을 제외한 마디가 학교  
종이 땡땡땡 악보와 유사합니다.

# |화면 (FrontEnd) – React

## Piano

- Tone이라는 web audio Framework를 사용하여 피아노를 구현함.

```
1 import './Piano.module.css';
2 import css from "styled-jsx/css"
3
4 import { Button } from 'react-bootstrap';
5 import React, {useEffect, useState} from 'react';
6 import axios from 'axios';
7
8 import {
9   playC4,
10  playDb4,
11  playD4,
12  playEb4,
13  playE4,
14  playF4,
15  playGb4,
16  playG4,
17  playAb4,
18  playA4,
19  playBb4,
20  playB4,
21  playC5,
22  playDb5,
23  playD5,
24  playEb5,
25  playE5,
26  playF5,
27  playGb5,
28  playG5,
29  playAb5,
30  playA5,
31  playBb5,
32  playB5,
33  playC6,
34  PlayNote
35 } from "../Tone.js"
```

```
function Piano(){

  const [saveNote, setSaveNote] = useState()

  const tonePianoApi = async(saveNote) => {
    try {
      console.log(`API 진입`)
      const response = await axios.post(`${HOST_3000}`, saveNote, {headers})
    } catch (err) {
      return err;
    }
  }

  useEffect(() => {
    window.addEventListener('keydown', PlayNote)
  }, [])
}
```

# |화면 (FrontEnd) – React + JavaScript

## Tone

- Tone.Synth 는 단일 오실레이터와 ADSR 엔벨로프가 있는 기본 신디사이저

```
import * as Tone from "tone"

✓ const playC4 = (keycode) =>{
  const synth = new Tone.Synth().toDestination();
  synth.triggerAttackRelease("C4", "8n");
  return keycode
}
```

```
const PlayNote = (event) =>{
  if(event.keyCode === 65){
    let a = localStorage.getItem("note")
    let b = playC4("A");
    let c = a + ',' + b
    //alert('최종 저장된 값: '+ c)
    localStorage.setItem("note", c)
  }
}
```

# |화면 (FrontEnd) – 결과

## Tone - Piano

- 피아노 구현 화면

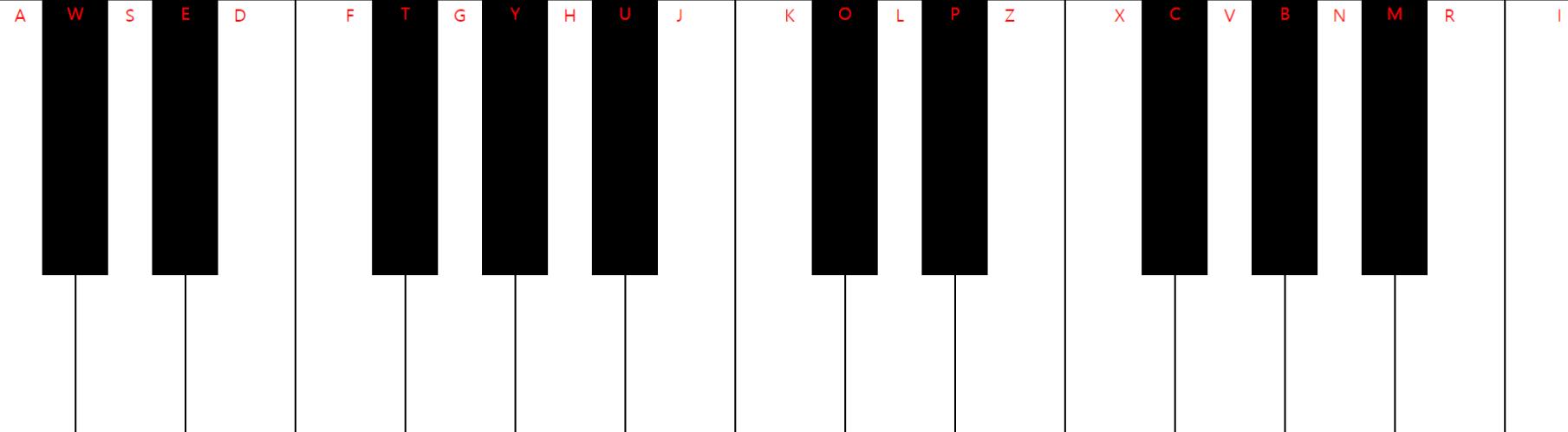


회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트

Enjoy Playing Piano!

녹음시작

녹음 끝



# I화면 (FrontEnd) – 결과

## Tone - Piano

- 피아노 keynote 저장된 화면

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The title bar includes tabs for Elements, Console, Sources, Network, Performance, Memory, Application, Security, Lighthouse, Recorder, Performance insights, Redux, and more. A status bar at the bottom indicates 3 warnings and 1 issue.

The main console area displays the following content:

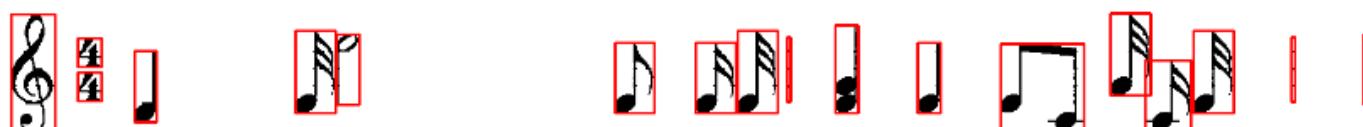
- \* Tone.js v14.7.77 \*
- Global.js?8405:67
- audio-context-constructor.js?123c:11
- test-audio-scheduled...s-support.js?199c:4
- constant-source-node...ructor.js?cf4e:42
- Piano.js?5a72:50
- Piano.js?5a72:249

Below the logs, a variable expansion for 'item' is shown:

```
▼{item: Array(12)} ⓘ
  ►item: (12) ['G', 'G', 'H', 'H', 'G', 'G', 'D', 'G', 'G', 'D', 'D', 'S']
```

A red highlight is applied to the array elements from index 0 to 11, specifically the sequence: 'G', 'G', 'H', 'H', 'G', 'G', 'D', 'G', 'G', 'D', 'D', 'S'.

# | 표절 프로그램



1. Cutted 1

```
[ \meter<"4/4"> d1/4 e1/32 e2/2 e1/8 e1/16 e1/32 {e1/4,g1/4} e1/4 e1/8 c1/8 g1/32 c1/16 e1/32 ]
```

# | 표절 프로그램

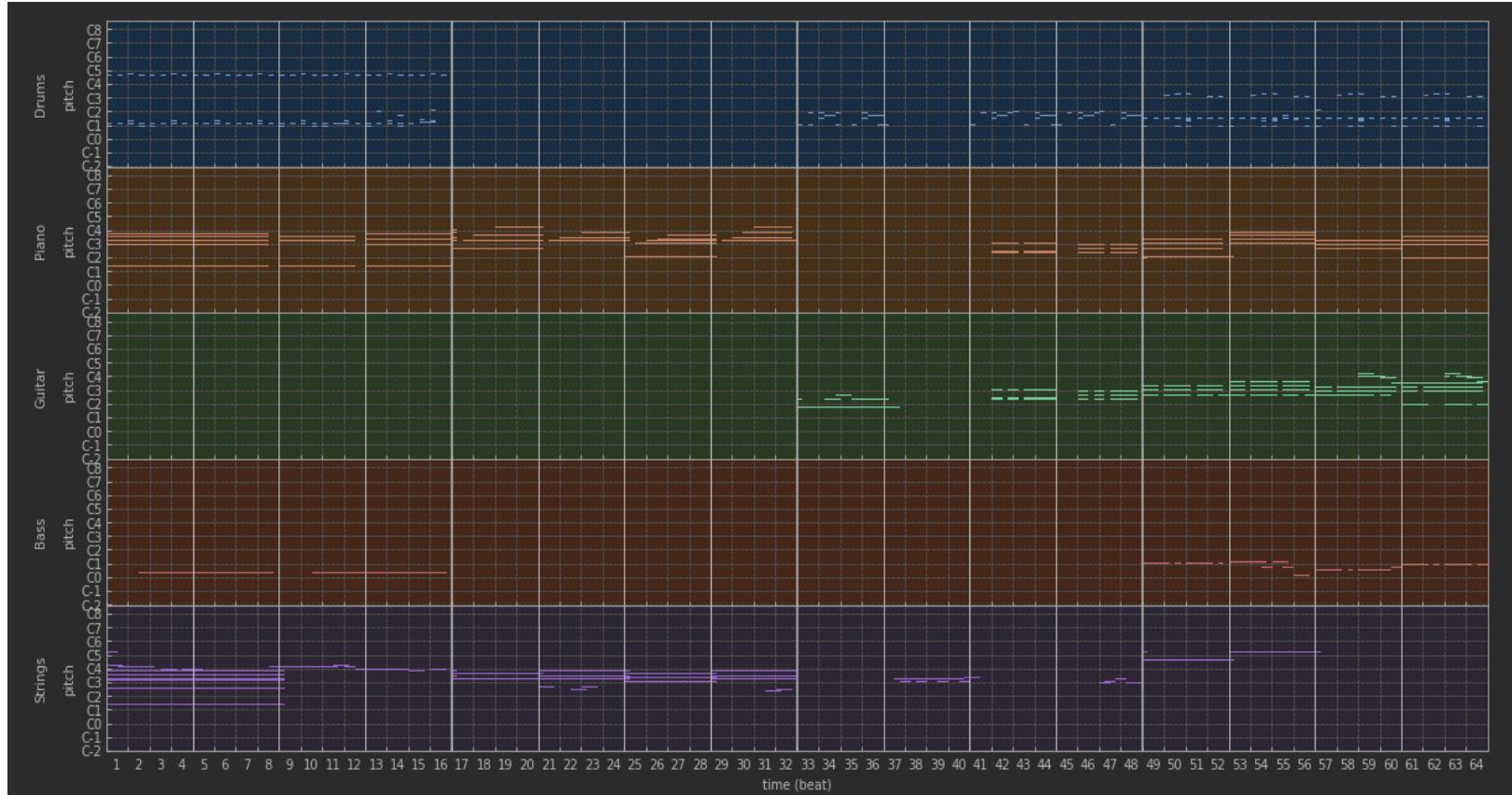
## Image

- 악보인식 하는 코드

```
1 import numpy as np
2 import pickle
3 from numpy.linalg import norm
4 from skimage.filters import *
5 from skimage.color import rgb2gray
6 import matplotlib.pyplot as plt
7 import cv2 as cv
8 import os
9 import time
10
11
12 def binarize(img, method):
13     if method == 'skimage_local':
14         return img >= threshold_local(img, 31, offset=3)
15     if method == 'skimage_sauvola':
16         return img >= threshold_sauvola(img, 31)
17     if method == 'cv_adaptive':
18         return cv.adaptiveThreshold(img, 255, cv.ADAPTIVE_THRESH_GAUSSIAN_C, cv.THRESH_BINARY_INV, 41, 10)
19
20
21 def get_staff_corners(img, contour): #this function doens't work well, it needs a different implementation
22     image_corners = np.array([[0, 0], [img.shape[1], 0], [0, img.shape[0]], [img.shape[1], img.shape[0]]])
23     staff_corners = np.zeros((4,2))
24
25     staff_corners[0] = max(contour, key=lambda point:norm(point[0] - image_corners[3]))
26     staff_corners[1] = max(contour, key=lambda point:norm(point[0] - image_corners[2]))
27     staff_corners[2] = max(contour, key=lambda point:norm(point[0] - image_corners[1]))
28     staff_corners[3] = max(contour, key=lambda point:norm(point[0] - image_corners[0]))
29
30     return staff_corners
```

# | 작곡 프로그램

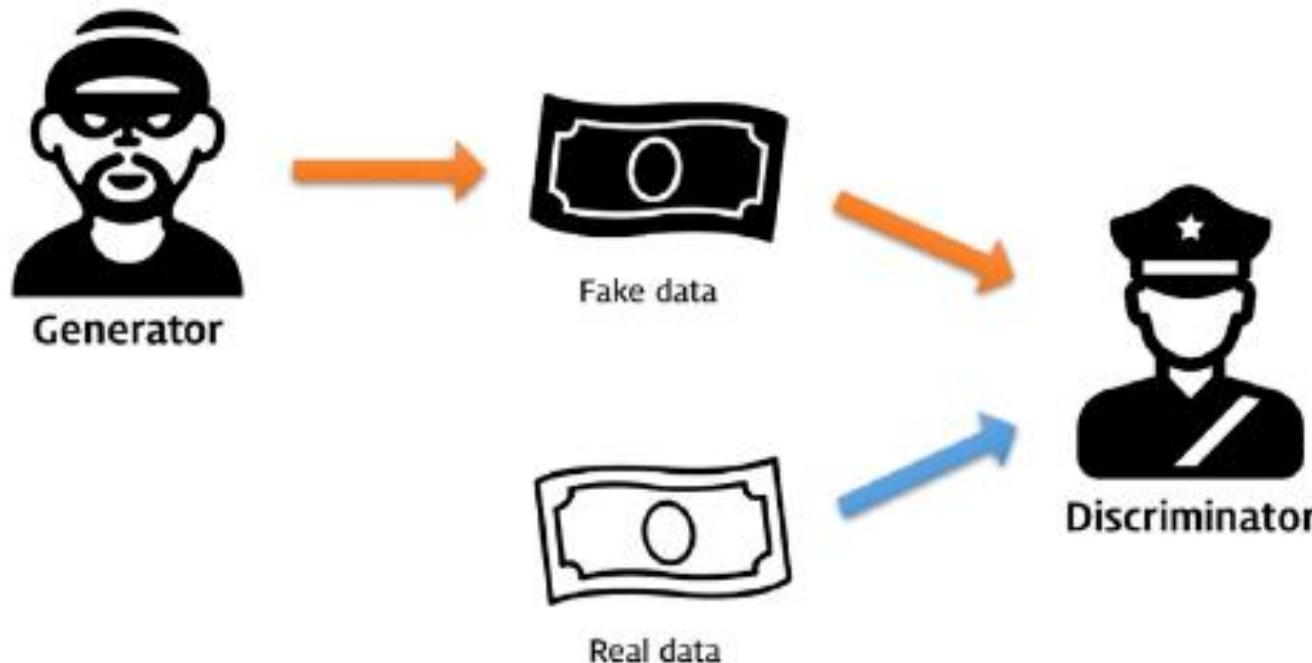
## Muse-GAN Data Set



5개의 세션으로 이루어진 Lakh Pianoroll Dataset 을 사용하였습니다.

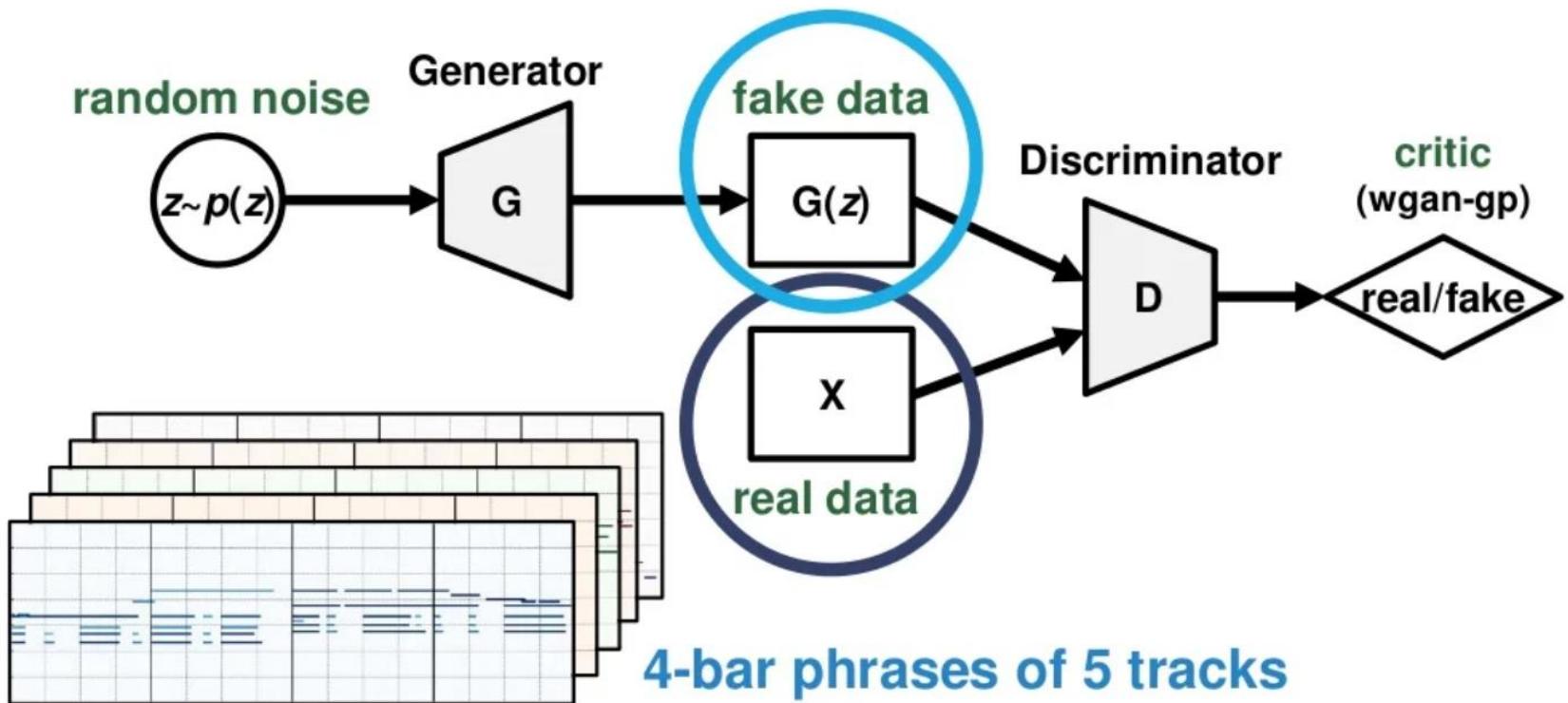
# |작곡 프로그램

Muse-GAN



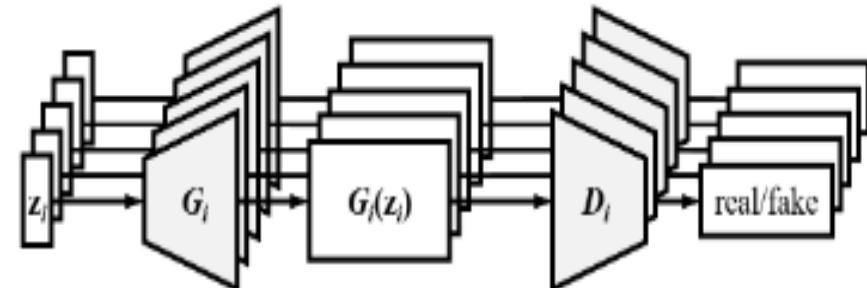
# |작곡 프로그램

Muse-GAN



# |작곡프로그램 – (Muse GAN)

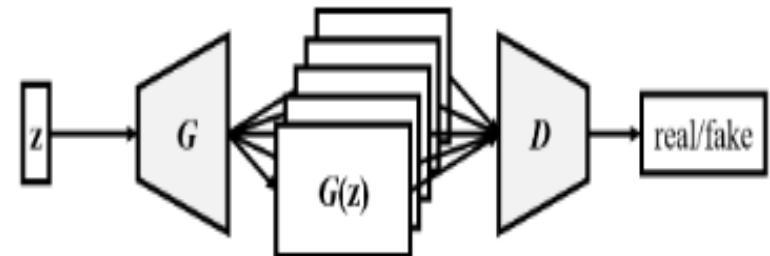
Jamming model



jamming 모델

랜덤 벡터  $z_i$  (에서 자체 트랙의 음악  
을 생성합니다.  
이러한 생성기는 서로 다른 판별기  
로부터 역전파를 수신합니다.

Composer model

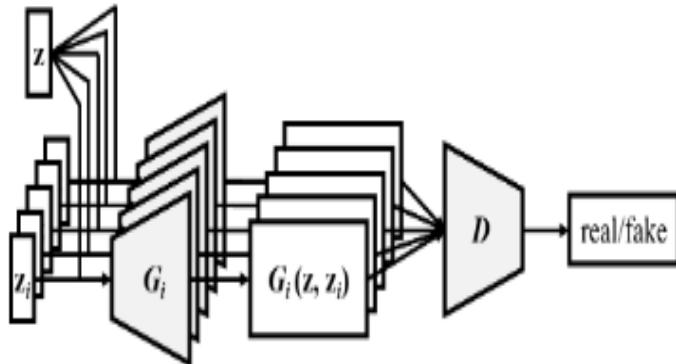


Composer 모델

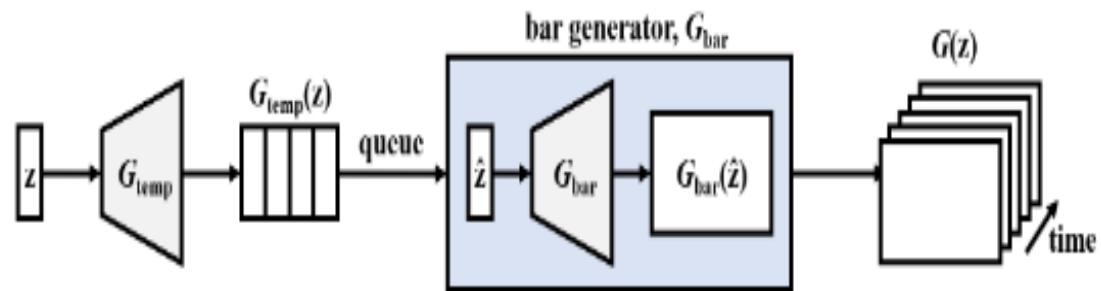
이 모델은 하나의 랜덤 벡터  $z$  와  
M 트랙을 집합적으로 검사하여  
입력 음악이 진짜인지 가짜인지  
구분하는 하나의 판별기가 필요합니다.

# |작곡프로그램 – (Muse GAN)

Hybrid model



Generation from scratch

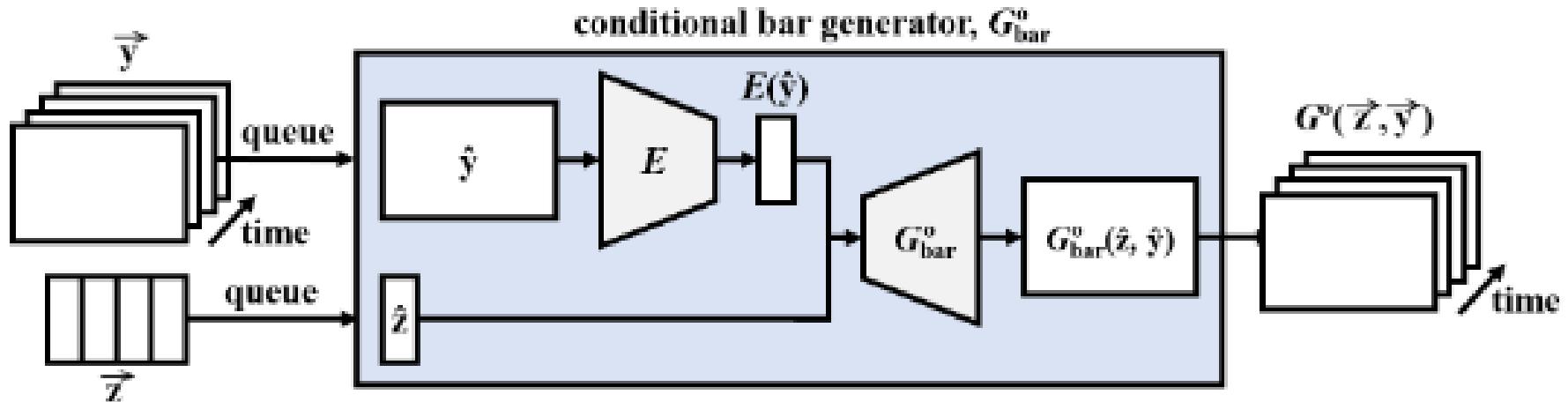


Hybrid 모델은 jamming 과 Composer의 개념을 결합한 M 개의 생성기를 필요로 하며 각각은 트랙 간 랜덤 벡터  $z$  및 트랙 내 랜덤 벡터  $z_i$ 를 입력으로 사용 합니다.  
M 트랙을 집합적으로 평가하기 위해 하나의 판별자만 사용합니다.

Generator 는  $G_{temp}$  ,  $G_{bar}$  라는 하위 네트워크로 구성되어 있습니다  $G_{temp}$ 은 시간 정보를 전달할것 같은 부분 벡터의 시퀀스에 노이즈 벡터를 매핑하고  $G_{bar}$ 에서는 피아노 롤을 순차적으로 생성하는 데 사용 합니다.

# |작곡프로그램 – (Muse GAN)

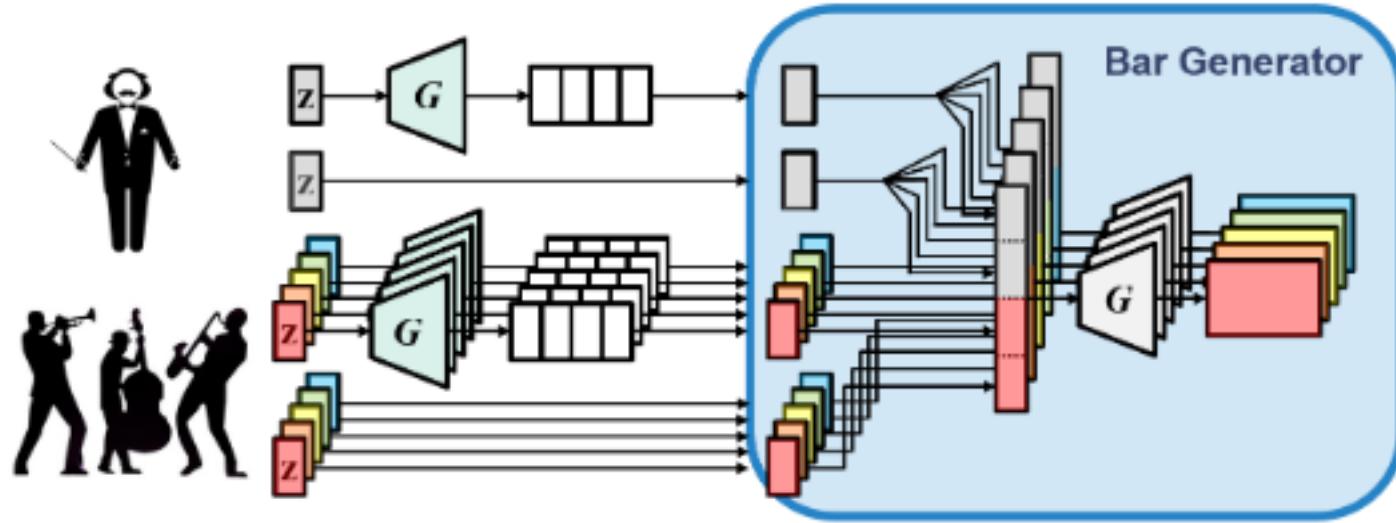
Track-conditional generation



Track generation

$G^o$ 는 트랙과 임의 노이즈를 입력으로 받는  $G^o$  bar를 사용하여 막대를 차례로 생성합니다. 고차원 조건으로 이러한 조건부 생성을 달성하기 위해 추가 인코더  $E$ 가 조건을  $z$  공간에 매핑하도록 훈련됩니다. 인코더는 주어진 트랙에서 피처 대신 트랙간의 피처를 추출 합니다.

# |작곡프로그램 – (Muse GAN)

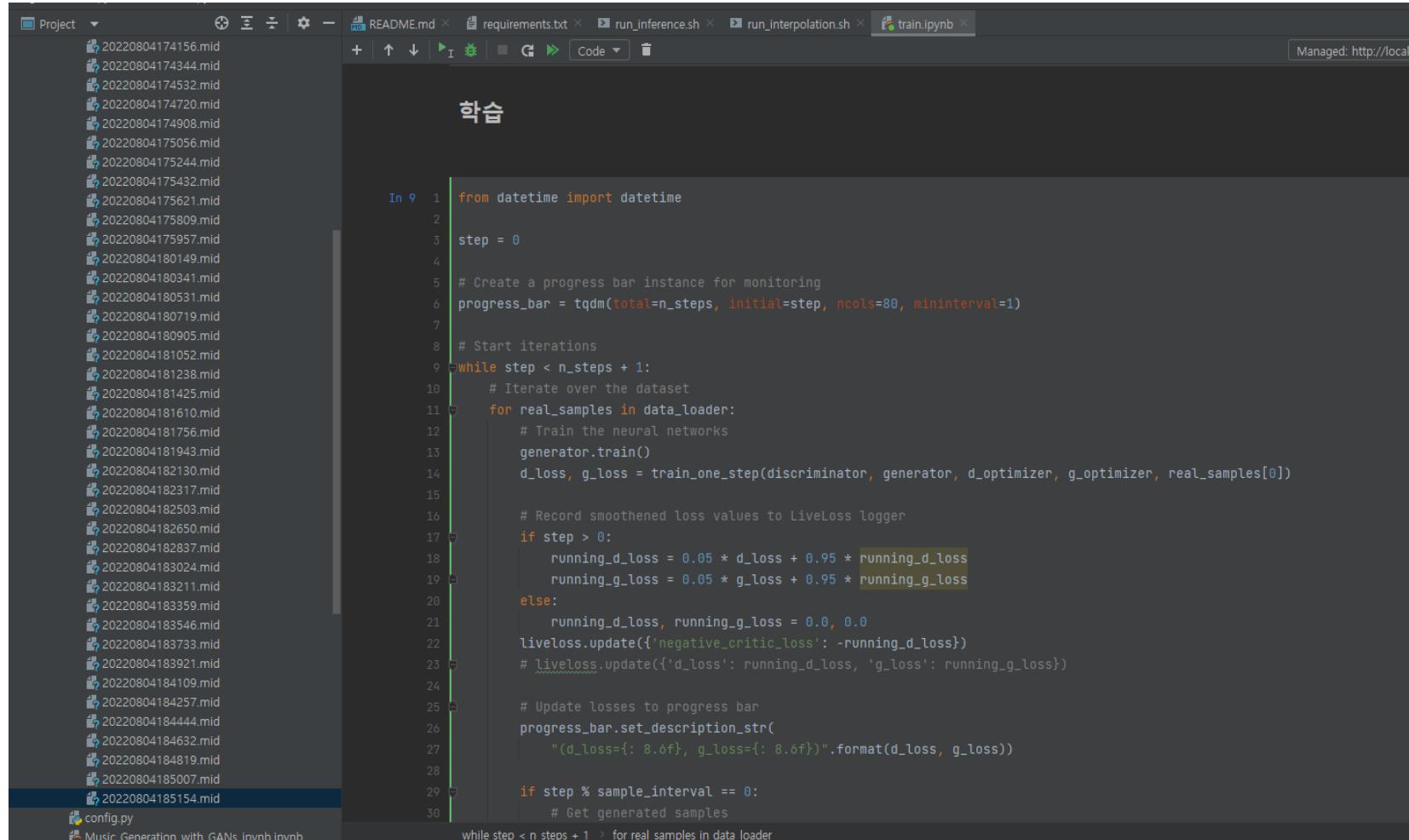


Muse GAN

Muse GAN은 4가지 유형의 임의 벡터를 입력합니다. 트랙에 대해 공유  $G_{temp}$  및 개인  $G_{temp}$ 는 각각 벡터  $z_t$   $z_{i,t}$ 를 입력으로 사용합니다. 각각은 트랙 간 및 트랙 내 시간 정보를 각각 포함하는 벡터를 출력합니다. 출력은 벡터  $z$  및  $z_i$ 와 함께 연결되어 Bar Generator에 제공됩니다. 그 후 순차적으로 피아노 롤을 생성하는  $G_{bar}$ 를 통해 다섯 가지 세션의 롤이 생성됩니다.

# |작곡 프로그램

## 학습 결과



Project ▾    README.md    requirements.txt    run\_inference.sh    run\_interpolation.sh    train.ipynb

Managed: http://local

### 학습

```
In 9 1 from datetime import datetime
2
3 step = 0
4
5 # Create a progress bar instance for monitoring
6 progress_bar = tqdm(total=n_steps, initial=step, ncols=80, mininterval=1)
7
8 # Start iterations
9 while step < n_steps + 1:
10     # Iterate over the dataset
11     for real_samples in data_loader:
12         # Train the neural networks
13         generator.train()
14         d_loss, g_loss = train_one_step(discriminator, generator, d_optimizer, g_optimizer, real_samples[0])
15
16         # Record smoothed loss values to LiveLoss logger
17         if step > 0:
18             running_d_loss = 0.05 * d_loss + 0.95 * running_d_loss
19             running_g_loss = 0.05 * g_loss + 0.95 * running_g_loss
20         else:
21             running_d_loss, running_g_loss = 0.0, 0.0
22         liveLoss.update({'negative_critic_loss': -running_d_loss})
23         # liveLoss.update({'d_loss': running_d_loss, 'g_loss': running_g_loss})
24
25         # Update losses to progress bar
26         progress_bar.set_description_str(
27             "(d_loss=: 8.6f, g_loss=: 8.6f)".format(d_loss, g_loss))
28
29         if step % sample_interval == 0:
30             # Get generated samples
```

while step < n\_steps + 1 → for real\_samples in data\_loader

20220804174156.mid  
20220804174344.mid  
20220804174532.mid  
20220804174720.mid  
20220804174908.mid  
20220804175056.mid  
20220804175244.mid  
20220804175432.mid  
20220804175621.mid  
20220804175809.mid  
20220804175957.mid  
20220804180149.mid  
20220804180341.mid  
20220804180531.mid  
20220804180719.mid  
20220804180905.mid  
20220804181052.mid  
20220804181238.mid  
20220804181425.mid  
20220804181610.mid  
20220804181756.mid  
20220804181943.mid  
20220804182130.mid  
20220804182317.mid  
20220804182503.mid  
20220804182650.mid  
20220804182837.mid  
20220804183024.mid  
20220804183211.mid  
20220804183359.mid  
20220804183546.mid  
20220804183733.mid  
20220804183921.mid  
20220804184109.mid  
20220804184257.mid  
20220804184444.mid  
20220804184632.mid  
20220804184819.mid  
20220804185007.mid  
20220804185154.mid

config.py  
Music\_Generation\_with\_GANs.ipynb.ipynb

# |작곡 프로그램

Vision

## ■ Muse-GAN



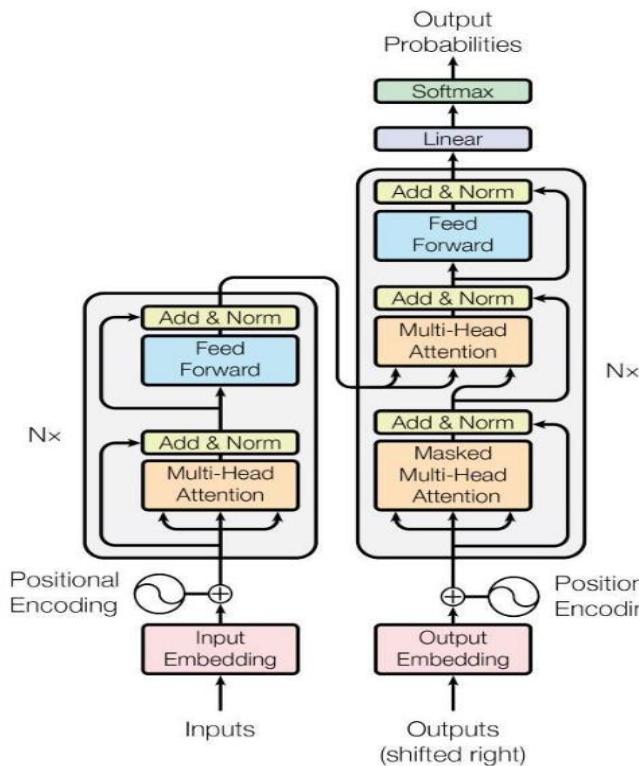
회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트



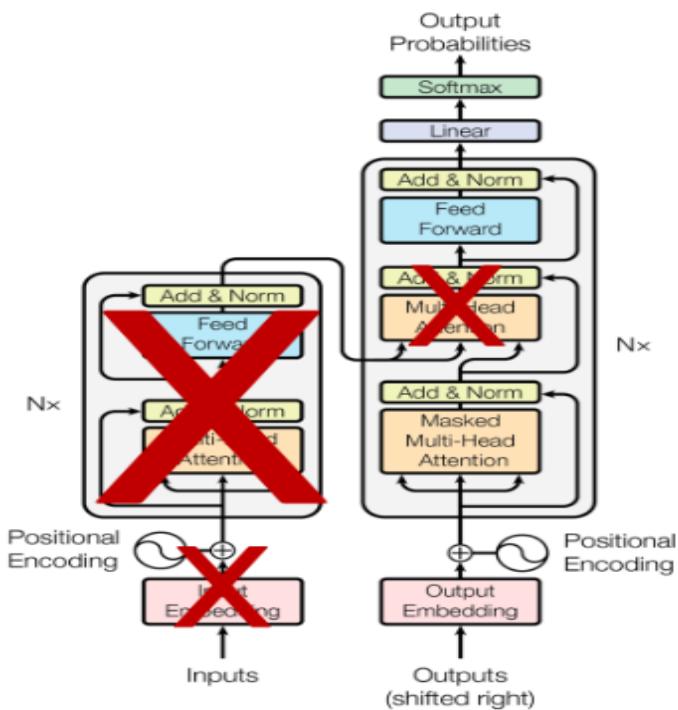
# |작사 프로그램 (NLP – KOGPT-3 PLM)

PLM

- 다음 단어 맞추기에 적합한 GPT를 선택



<Transformer>

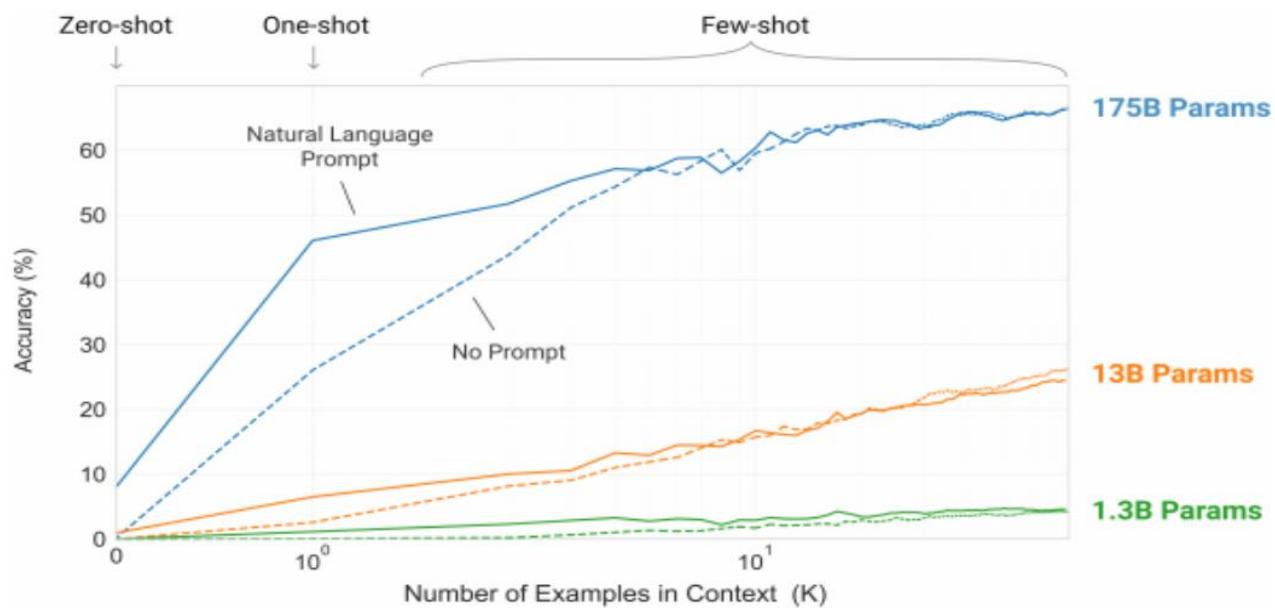


<GPT>

# |작사 프로그램 (NLP – KOGPT-3)

## 장점

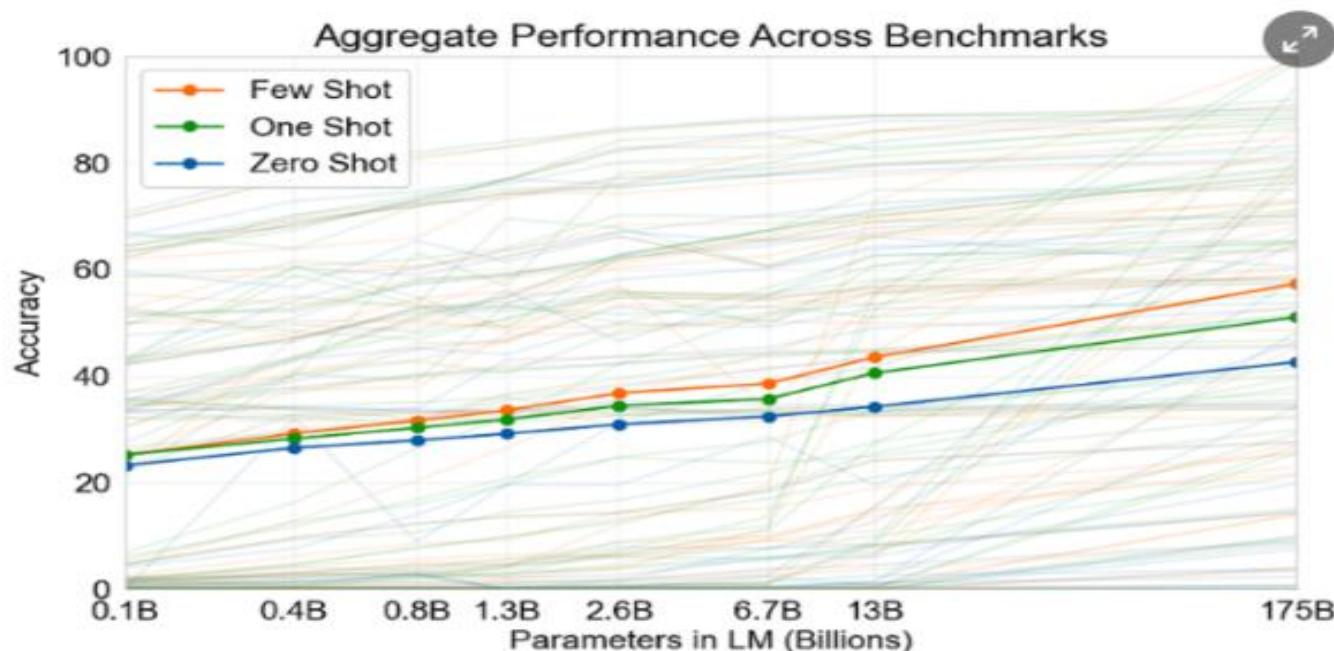
- GPT2 = 800만개 데이터 셋 15억개 파라미터
- GPT3 = 3000억개 데이터셋 1750억개 파라미터
- Fine-Tuning에서 필요한 많은 데이터와 그 많은 데이터들을 학습시킬 시간이 필요
- 하지만 Few-Shot-Learning은 몇 개의 샘플만 던져도 좋은 성능을 냈.(그래프)



# |작사 프로그램 (NLP – KOGPT-3)

## 단점

- 성차별, 인종차별 발언도 포함되어 있음
- Transformer기반 기억개념 없음, 현재의 문맥을 입력해 주어야 그에 맞는 OUTPUT 발생
- 범용적인 다양한 문제만 잘 풀어나감(그래프)
- 데이터양과 비례하는 모델
- Google Colab(Pro) 사용(개발환경의 한계)



# |작사 프로그램 (CRAWLING-CODE)

## CRAWLING

- 동요 가사가 올려져 있는 블로그에서 크롤링 진행

```
1 import requests
2 from bs4 import BeautifulSoup
3 import pandas as pd
4
5 url = 'https://m.cafe.daum.net/kidchoir/5Hf2/7?listURI=%2Fkidchoir%2F5Hf2'
6
7 response = requests.get(url)
8
9 if response.status_code == 200:
10     html = response.text
11     soup = BeautifulSoup(html, 'html.parser')
12     title = soup.select_one('#mArticle > div.view_info')
13     data=pd.DataFrame(title)
14     data.to_csv('동요가사.csv', encoding='utf-8')
15 else :
16     print(response.status_code)
```

# |작사 프로그램 (CRAWLING-OUTPUT)

## CRAWLING

### ■ 총 157곡의 동요가사

▣ 동요악보모음.csv

1 ,제목,가사  
2 0,BINGO,옆집에 사는 개이름 빙고라지요 / B I N G O / B I N G O / B I N G O / 빙고 개 이름  
3 1,Edelweiss 에델바이스,"Edelweiss, Edelweiss, Every morning you greet me. Small and white, clean and bright, You look happy to meet me. Blossom o  
4 2,Ten Little Indians,"One little, two little, Three little Indians. Four little, five little, Six little Indians. Seven little, eight little,Nine  
5 3,TV 유키원 하나 둘 셋,친구야 손 잡고 하나 둘 셋 엄마도 아빠도 하나 둘 셋 우리는 언제나 하나 둘 셋 정말 정말 좋아요 하나 둘 셋 풍선 타고 놓동동 구름 타  
6 4,가게놀이,가게놀이 할 사람 모두모여라 가게놀이 할 사람 모두모여라 사과 한개 주세요 1000원입니다. 귤 하나 주세요 그것도 1000원이죠 그럼 모두다 얼마입니  
7 5,가위바위보(쎄쎄쎄),아침 바람 찬 바람에 출고 가는 저 기러기 우리 선생 계실 적에 염서 한 장 써 주세요 한 장 말고 두 장이요 두 장 말고 세 장이요 구리구  
8 6,강아지,우리집 강아지는 복슬 강아지 어머니가 빨래 가면 멍멍멍 풀랑풀랑 따라가며 멍멍멍 우리집 강아지는 예쁜 강아지 학교 갔다 돌아오면 멍멍멍 꼬리치고  
9 7,개구리,개굴개굴 개구리 노래를 한다 아들 손자 며느리 다 모여서 밤새도록 하여도 듣는 이 없네 듣는 사람 없어도 날이 밝도록 개굴개굴 개구리 노래를 한다 개  
10 8,개구리,엄마 개구리가 노래 부른다 짹짹 짹쨍쨍쨍 짹쨍쨍쨍 짹짹 짹짹 이야기야요 이야기야요 이야기야야야 이야기야야야 이야기야야야 아기 개구리가 노래 부른다 짹짹 짹쨍쨍쨍 짹  
11 9,겨울나무,나무야 나무야 겨울 나무야 눈 쌓인 응달에 외로이 서서 아무도 찾지 않는 추운 겨울을 바람 따라 휘파람만 불고 있느냐  
12 10,겨울바람,손이 시려워 (꽁) 발이 시려워 (꽁) 겨울 바람 때문에 (꽁꽁꽁) 손이 꽁꽁꽁 (꽁) 발이 꽁꽁꽁 (꽁) 겨울 바람 때문에 (꽁꽁꽁) 어디서 이 바람은 시  
13 11,고기 잡이,고기를 잡으려 바다로 갈까나. 고기를 잡으려 강으로 갈까나. 이 병에 가득히 넣어 가지고서 라라라라 라라라라 온다나. 선생님 모시고 가고 싶지마는  
14 12,고드름,고드름 고드름 수정 고드름 고드름 따다가 발을 엮어서 각시방 영창에 달아 놓아요  
15 13,고향의 봄,1. 나의 살던 고향은 꽃피는 산골 복숭아꽃 살구꽃 아기진달래 울긋불긋 꽃 대궐 차리인 동네 그 속에서 놀던 때가 그립습니다 2. 꽃동네 새 동네 나  
16 14,곰 세 마리,곰 세 마리가 한 집에 있어 아빠곰 엄마곰 애기곰 아빠곰은 뚱뚱해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쌰으쌰 잘한다  
17 15,과꽃,올해도 과꽃이 피었습니다 꽃밭 가득 예쁘게 피었습니다 누나는 과꽃을 좋아했지요 꽃이 피면 꽃밭에서 아주살았죠 과꽃 예쁜 꽃을 들여다 보면 꽃 속에  
18 16,과수원 길,동구 박 과수원길 아카시아 꽃이 활짝 꽂네 하얀 꽃 이파리 눈송이처럼 날리네 향긋한 꽃냄새가 실바람타고 솔솔 둘이서 말이 없네 얼굴 마주 보며 생  
19 17,귀여운 꼬마,귀여운 꼬마가 닭장에 가서 암탉을 잡으려다 놀쳤다네 닭장 밖에 있던 배고픈 여우 옮겨니 하면서 물고 갔다네 꼬꼬댁 암탉 소리를 쳤네 꼬꼬댁 암  
20 18,그대로 멈춰라,즐겁게 춤을 추다가 그대로 멈춰라 즐겁게 춤을 추다가 그대로 멈춰라 눈도 감지 말고 웃지도 말고 옮지도 말고 움직이지 마 즐겁게 춤을 추다가  
21 19,기차길옆,기찻길 옆 오막살이 아기 아기 잘도 잔다 칙폭 칙칙폭폭 칙칙폭폭 칙칙폭폭 기차소리 요란해도 아기 아기 잘도 잔다 기찻길 옆 옥수수밭 옥수수는 잘도  
22 20,기차를 타고,1. 기차타고 신나게 달려가보자 높은 산도 지나고 넓은 들도 지나고 푸른산을 지날때엔 산새를 찾고 넓은 바다 지날때엔 물새와 놀고 설레임을 가득  
23 21,기찻길 옆,기찻길 옆 오막살이 아기 아기 잘도 잔다 칙 폭 칙 칙 폭 폭 칙 칙 폭 폭 칙 칙 폭 폭 기차소리 요란해도 아기 아기 잘도 잔다  
24 22,"깊은 계곡, 깊은 계곡 광산 마을 동굴 집에","늙은 아빠 어여쁜 딸이 사랑으로 살았네 오 내사랑 오 내사랑 나의 귀여운 클레멘타인 너는 영영 가버리고 나만 훌  
25 23,깡깡총체조,손을 높이 손을 높이 쭉쭉쭉쭉쭉쭉 뻗어 봐요 발을 쿵쿵쿵쿵쿵쿵굴려 봐요 영덩이를 실룩 실룩살룩 이쪽 저쪽 실룩살룩 뱅글뱅글 능  
26 24,꼬까신,개나리 노오란 꽃 그늘 아래 가지런히 놓여 있는 꼬까신 하나 아기는 살짝 신벗어 놓고 맨발로 한들 한들 나들이 갔나 가지런히 가다리는 꼬까신 하나  
27 25,꼬마눈사람,한겨울에 밀짚모자 꼬마 눈사람 눈썹이 우습구나 코도 비둘고 거울을 보여줄까 꼬마 눈사람  
28 26,꼬마자동차 봉봉,봉봉봉 아주 작은 자동차 꼬마 자동차가 나왔다 봉봉봉 꽂향기 를 맡으면 힘이 솟는 꼬마 자동차 엄마 찾아 모험 찾아 나서는 세계 여행 우리  
29 27,꼬부랑 할머니,꼬부랑 할머니가 꼬부랑 고갯길을 꼬부랑 꼬부랑 넘어가고 있네 꼬부랑 꼬부랑 꼬부랑 꼬부랑 고개는 열두 고개 고개를 고개를 넘어간다  
30 28,꼼꼼약속해,너하고 나는 친구 되어서 사이좋게 지내자 새끼손가락 고리 걸고 꼼꼼 약속해  
31 29,꽃밭에서,아빠하고 나하고 만든 꽃밭에 채송화도 봉송아도 한창입니다 아빠가 매어놓은 새끼줄 따라 나팔꽃도 어울리게 피었습니다

# |작사 프로그램 (NLP – KOGPT-3)

## Code

- Few-shot-Learning 실행
- Temperature의 값은 양수여야 한다.

```
1 import torch
2 from transformers import AutoTokenizer, AutoModelForCausalLM
3
4 tokenizer = AutoTokenizer.from_pretrained(
5     'kakaobrain/kogpt', revision='KoGPT6B-ryan1.5b-float16', # or float32 version: revision=KoGPT6B-ryan1.5b
6     bos_token='[BOS]', eos_token='[EOS]', unk_token='[UNK]', pad_token='[PAD]', mask_token='[MASK]'
7 )
8 model = AutoModelForCausalLM.from_pretrained(
9     'kakaobrain/kogpt', revision='KoGPT6B-ryan1.5b-float16', # or float32 version: revision=KoGPT6B-ryan1.5b
10    pad_token_id=tokenizer.eos_token_id,
11    torch_dtype='auto', low_cpu_mem_usage=True
12 ),to(device='cuda', non_blocking=True)
13
14 _ = model.eval()
15
16
17 #prompt: 입력, 최대 2048 tokens (1500 단어)
18 #completion: 생성된 출력, 최대 2048 tokens (1500 단어)
19 #tokens: 말뭉치 조각 - 단어와 비슷한, 많은 토큰이 공백으로 시작함 " hello"
20 #prompt 끝부분에 공백을 놔두면 안됨.
21 prompt = ''
22 동요 노래 가사
23
24 1.파란 하늘 파란 하늘 풀이 드리운 푸른 언덕에 아기 염소 여럿이 물을 들고 놀아요 해처럼 밝은 얼굴로
25
26 2.학교 층이 땡땡땡 어서모이자 선생님이 우리를 기다리신다
27
28 3.산토끼 토끼야 어디를 가느냐 깡충깡충뛰면서 어디를 가느냐
29
30 4.였다 었다 비행기 날아라 날아라 놀이 놀이 날아라 우리 비행기
31
32 5.새신을 신고 뛰어보자 팔짝 머리가 하늘까지 달겠네
33
34 6.곰 세 마리가 한 집에 있어 아빠곰 엄마곰 애기곰 출출해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
35
36 7.***
37
38 with torch.no_grad():
39     tokens = tokenizer.encode(prompt, return_tensors='pt'),to(device='cuda', non_blocking=True)
40     gen_tokens = model.generate(tokens, do_sample=True, temperature=0.8, max_length=350)
41     generated = tokenizer.batch_decode(gen_tokens)[0]
42
43 print(generated)
```

# |작사 프로그램 (Temperature=0.2)

0.2

## ■ 도돌이표 느낌이다...

동요 노래 가사

- 1.파란 하늘 파란 하늘 꿈이 드리운 푸른 언덕에 아기 염소 여럿이 물을 놀고 놀아요 해처럼 밝은 얼굴로
- 2.학교 종이 팽팽팽 어서모이자 선생님이 우리를 기다리신다
- 3.산토끼 토끼야 어디를 가느냐 깡충깡충 뛰면서 어디를 가느냐
- 4.떴다 떴다 비행기 날마라 날마라 놀이 놀이 날마라 우리 비행기
- 5.새신을 신고 뛰어보자 팔짝 머리가 하늘까지 닿겠네
- 6.곰 세 마리가 한 집에 있어 아빠곰 엄마곰 애기곰 아빠곰은 뚱뚱해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 7.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 8.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 9.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 10.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 11.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 12.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 13.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 14.아빠 곰은 뚱뚱해 엄마 곰은 날씬해 애기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 15.아빠

# |작사 프로그램 (Temperature=0.4)

0.4

■ 0.2와 크게 다르지 않다...

동요 노래 가사

- 1.파란 하늘 파란 하늘 꿈이 드리운 푸른 언덕에 아기 염소 여럿이 물을 놀아요 해처럼 밝은 얼굴로
- 2.학교 종이 땅땡땡 어서모이자 선생님이 무리를 기다리신다
- 3.산토끼 토끼야 머리를 가느냐 깡총깡총 뛰면서 머리를 가느냐
- 4.떴다 떴다 비행기 날아라 날아라 높이 높이 날아라 우리 비행기
- 5.새신을 신고 뛰어보자 팔짝 머리가 하늘까지 달겠네
- 6.곰 세 마리가 한 집에 있어 아빠곰 엄마곰 애기곰 아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 7.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 8.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 9.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 10.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 11.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 12.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 13.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 14.아빠곰은 뚫뚫해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 15.아빠

# |작사 프로그램 (Temperature=0.6)

0.6

- 뭔가 내용이 달라지기 시작하다가 또 똑같다...

## 동요 노래 가사

- 1.파란 하늘 파란 하늘 품이 드리운 푸른 언덕에 마기 엽소 여럿이 물을 들고 놀아요 해처럼 밝은 얼굴로
- 2.학교 졸미 팽팽팽 어서모이자 선생님이 우리를 기다리신다
- 3.산토끼 토끼야 어디를 가느냐 깔충깔충 위면서 어디를 가느냐
- 4.떴다 떴다 비행기 날아라 날아라 놀이 놀이 날아라 우리 비행기
- 5.새신을 신고 뛰어보자 팔짝 머리가 하늘까지 달겠네
- 6.곰 세 마리가 한 집에 있어 아빠곰 엄마곰 얘기곰 아빠곰은 뚱뚱해 엄마곰은 날씬해 얘기곰은 너무 귀여워 으쓱으쓱 잘한다
- 7.아빠 곰은 뚱뚱해 엄마 곰도 뚱뚱해 얘기 곰은 너무 귀여워 으쓱으쓱 잘한다
- 8.율나라에도 별나라에도 잘 수가 있단다.
- 9.나는 나는 자라서 무엇이 될까
- 10.나는 나는 자라서 무엇이 될까
- 11.나는 나는 자라서 무엇이 될까
- 12.나는 나는 자라서 무엇이 될까
- 13.나는 나는 자라서 무엇이 될까
- 14.나는 나는 자라서 무엇이 될까
- 15.나는 나는 자라서 무엇이 될까
- 16.나는 나는 자라서 무엇이 될까
- 17.나는 나는 자라서 무엇이 될까
- 18.나는 나는 자라서 무엇이 될까
- 19.나는 나는 자라서 무엇이 될까
- 20.나는 나는 자라서 무엇이 될까

# |작사 프로그램 (Temperature=1.0)

1.0

- 시를 쓰는 것 같다...

제 1 부

[얼마야]

제 1 장 아빠 곰은 뚤뚫해

제 2 장 아빠 곰은 날씬해

제 1 부

제 1 장 아빠 곰은 뚤뚫해

얼마야

"어, 이게 뭐야? 웬 벌레가?"

"벌레가 아니라 나비예요."

"무슨 소리?"

"나비가 아니라 잠자리란 말이요."

"어허, 이런 바보 같으니."

"아빠, 난 나비가 아니라 잠자리란 말이거든."

# |작사 프로그램 (Temperature=0.8)

## 최종 Out-Put

- Temperature의 조절 끝에 0.8로 지정

동요 노래 가사

- 1.파란 하늘 파란 하늘 꿈이 드리운 푸른 언덕에 아기 염소 여럿이 풀을 뜯고 놀아요 해처럼 밝은 얼굴로
- 2.학교 종이 맹맹맹 어서모이자 선생님이 우리를 기다리신다
- 3.산토끼 토끼야 어디를 가느냐 강총강총 위면서 어디를 가느냐
- 4.떴다 였다 비행기 날마라 날마라 높이 높이 날마라 우리 비행기
- 5.새신을 신고 뛰어보자 팔짝 머리가 하늘까지 닿겠네
- 6.곰 세 마리가 한 집에 있어 아빠곰 엄마곰 애기곰은 풍뚱해 엄마곰은 날씬해 애기곰은 너무 귀여워 으쓱으쓱 잘한다
- 7.구름을 잡아라 넷물을 따라 가면 내가 놀던 물이 흘러 어디로 가는지 알 수 있지
- 8.햇님이 방긋 웃으면 나는야 햇님이 얼굴
- 9.비 오는 날 무산 속 얼굴 마주하고 너랑 나랑 둘이 비를 맞네
- 10.새나라의 어린이는 일찍 일어납니다 학교 가는 길에 달그락 달그락
- 11.눈 오는 날 눈 싸움을 해요 동무들과 모여서 눈 싸움을 해요 눈 싸움을 해요
- 12.누나 손을 잡고서 노래를 불러요 동동동동 손을 잡고서
- 13.산에 올라요 산에 올라요 푸른 하늘 푸른 한울 푸른 언덕에
- 14.oi 다음에 커서 어른 되면 나도 어른 되면
- 15.저 건너 푸른 언덕에 아기 염소 여럿이 풀을 뜯고 놀아요 해처럼 밝은 얼굴로

# |작사 프로그램

NLP

## ■ KOGPT-3 사용



회사소개 ▾ 계시판 ▾ 음악 프로그램 ▾ ○ 표절/작곡/작사 프로그램 ○ 관련 사이트

### 작사 키워드

동요

작사하기

동요 노래 가사 :

1.파란 하늘 파란 하늘 꿈이 드리운 푸른  
언덕에 아기 염소 여럿이 풀을 뜯고 놀아  
요 해처럼 밝은 얼굴로

2.학교 종이 땅땡땡 어서모이자 선생님이  
우리를 기다리신다

3.산토끼 토끼야 어디를 가느냐 깡총깡총  
뛰면서 어디를 가느냐

감사합니다

# Q & A

CRP 산트자트

스프린트 PO : 서성민(Next 개발자)

스프린트기간: 5/23 ~ 6/10

스프린트 목표

- 1. 화면 1 차 완성
- 2. 데이터베이스 스키마 완성

스프린트 상세:

1주차 ( 5/23 ~ 5/27 )

<서성민>

- ~~전체 화면 관리~~
- ~~Home (로그인 전) 화면~~
- ~~Home (로그인 후) 화면~~

<권혜민>

- ~~음원 검색 화면 컴포넌트 작성~~
- ~~음원 검색 중 화면 컴포넌트 작성~~
- ~~음원 검색 확인 화면 컴포넌트 작성~~
- ~~사용자 (회원가입, 로그인) 화면 컴포넌트 작성~~

<조현국>

- 악보 분석 등록 컴포넌트 작성
- 악보 분석 확인 컴포넌트 작성
- 회사 → 개발자들 컴포넌트 작성

2주차 ( 5/30 ~ 6/3 )

<서성민>

- ~~타임스크립트 공부~~
- ~~자바스크립트 공부~~
- ~~화면 발표 완료~~

<권혜민>

- ~~[X] 음성인식 기술 STT, TTS 알고리즘 찾아보기~~

관련 링크: <https://penguin-story.tistory.com/1>

<조현국>

- OpenCV 를 이용한 악보인식 티스토리 분석 후 정리하기

관련 링크: <https://hackids.tistory.com/120>

## 2 주차 회의 내용

- 챗봇을 음성 챗봇인 휴먼 AI로 대체, 질문지 제공
- 프론트에서 웹 장고 제거 (서버리스)
- 음악 30초 미리 듣기
- 화면 시나리오 구성
- 튜토리얼 각 화면마다 구성
- 문의하기 크롤링, 게시판 하드 코딩

## 최종 파트 분담

조현국 : NLP / KOBERT / KOGPT

권혜민 : Muse GAN / VIT / CV

서성민 : 프론트 화면 구성 / 작곡 (tone.js)

## 3 주차 회의 내용

### DB 수집 계획

개인 데이터는 각각 들고 있기

1. 이미지 악보 (디지털 악보) : 가요 악보 5 개, 동요 악보 5 개 (한 손)
2. mp3 파일: 인기 가요 5 개
3. 휴먼 AI : GIF 파일로 은아 사진 캐릭터화
4. 음성 챗봇 데이터 : AI 허브에서 음성 챗봇용 데이터 수집

3 주차 ( 6/6 ~ 6/10 )

<서성민>

- ~~타임스크립트 공부 및 화면 마감 개획 짜놓기~~
- ~~공통요소 (게시판) 코드 분석하기~~
- ~~Javascript로 짜여진 팀 프로젝트 템플릿 부분 경량화 및 Typescript로 녹이는 거 고민하기~~

#### <권혜민>

- ~~퍼신러닝/러닝 개념 배우기~~
- ~~지도학습 / 비지도학습 / 준지도학습 개념 공부~~

#### <조현국>

- RNN 개념 공부하기
- 손실함수와 활성화함수에 대해 공부

스프린트 PO : 권혜민 (Python 개발자)

스프린트기간: 6/13 ~ 7/17

스프린트 목표

3. 화면 1 차 완성
4. 데이터베이스 스키마 완성

스프린트 상세:

4주차 ( 6/13 ~ 6/17 )

#### <서성민>

- ~~javascript에 음악 이식하는 작업 분석~~  
관련 링크  
<https://www.youtube.com/watch?v=ah7s3efFzFI>
- ~~악보 이미지 데이터 수집할 사이트 찾아보기~~  
[https://imslp.org/wiki/Main\\_Page](https://imslp.org/wiki/Main_Page)

#### <권혜민>

- ~~신경망, 생물학적 뉴런/인공뉴런~~
- ~~활성화 함수 / 손실함수~~

#### <조현국>

- 전이학습(Fine-Tuning) 공부

5주차 ( 6/20 ~ 6/24 )

#### <서성민>

- ~~악보 데이터 수집~~
- ~~tone.js framework 연구하기~~  
관련 링크  
<https://tonejs.github.io/>

#### <권혜민>

- ~~경사 하강법 최적화 알고리즘~~
- ~~역전파와 자동미분~~
- ~~화률적 경사하강법~~
- ~~미니배치 경사하강법~~

#### <조현국>

- PLM의 종류와 트랜스포머가 사용되게 된 역사 공부
- 주변 단어를 예측하는 WordEmbedding에 대한 공부(word2vec)
- context에 따른 Embedding에 대해 공부(ELMo)

6주차 ( 6/27 ~ 7/1 )

#### <서성민>

- ~~Tonejs ( Web Audio Framework ) 활용하여 react로 피아노 화면 구현~~
- ~~Javascript + React로 화면 구현~~
- ~~타임스크립트를 활용한 팀 프로젝트 화면 스타일 연구.~~

#### <권혜민>

- ~~일반화 / 드롭아웃~~
- ~~드롭아웃 동작방식 / 역동작방식~~
- ~~데이터 증강 / 얼리스타핑 / 배치 정규화~~

#### <조현국>

- 기계 번역 모델 Seq2seq에 대해 공부
- 입력 문장이 길어도 소실을 줄이는 Attention에 대해 공부

7주차 ( 7/4 ~ 7/8 )

#### <서성민>

- ~~Javascript를 Typescript로 전환하여 화면 구현~~
- ~~tone-piano에 note(음) json 형태로 저장할 수 있는 방법 연구~~

#### <권혜민>

- ~~텐서플로 데이터 흐름 그래프~~
- ~~정적 그래프 변수 / tf.layers 기반 모델 정의~~
- ~~자동미분 - 손실과 옵티마이저~~

#### <조현국>

- Attention 연산 기반 트랜스포머의 내부 encoding/decoding 공부(multi head attention)
- Multi head Attention의 기본 Quert의 유사한 Key를 찾아 value를 얻어내는 과정 공부
- 디코더의 대표모델 GPT 연구

8주차 ( 7/11 ~ 7/15 )

#### <서성민>

- ~~Redux를 활용하여 계시판 구현~~
- ~~팀 프로젝트 스타일 마무리~~

#### <권혜민>

- ~~이미지 분류를 위한 신경망/합성곱 신경망 공부~~
- ~~Google Colab을 통한 전이학습과 파인튜닝 예제하기~~
- ~~감정 분류를 위한 신경망 공부~~

#### <조현국>

- 트랜스포머 Tokenizer 활용하여 문장 토큰화 / 단어 토큰화
- 트랜스포머 Tokenizer 활용하여 단어 임베딩 해보기
- 인코더 대표모델 BERT 연구

스프린트 PO : 조현국(Python 개발자)

스프린트기간: 7/18 ~ 8/9

#### 스프린트 목표

5. 화면 1차 완성
6. 데이터베이스 스키마 완성

스프린트 상세:

9주차 ( 7/18 ~ 7/22 )

#### <서성민>

- ~~FileUpload 가능 구현~~
- ~~redux를 활용하여 만든 게시판과 서버 연결하기 위해 연구중. (openapi)~~
- ~~화면에서 파일 업로드한 이미지 openapi(server)로 전달하기 위해 연구중.~~

#### <권혜민>

- ~~오토인코더 공부하기~~
- ~~이미지 시가화 하기 / 전처리~~
- ~~선형 오토인코더 구축하고 학습시키기~~
- ~~Music 21을 이용해 midi 파일 npy로 변환하는 코드 작성~~

#### <조현국>

- 트랜스포머 논문 분석하기
- 개인프로젝트 감성분석 챗봇 모델 돌려보기

10주차 ( 7/25 ~ 7/29 )

#### <서성민>

- ~~FileUpload 기능 구현~~
- ~~redux 를 활용하여 만든 게시판과 서버 연결하기 위해 연구중. (openapi)~~
- ~~화면에서 파일 업로드한 이미지 openapi (server) 로 전달하기 위해 연구중.~~

#### <권혜민>

- ~~합성곱 오토인코더 예제 구현하기~~
- ~~Muse GAN~~에서 ~~pretrained models~~ 가져오기
- ~~생성자 구현하기 / 구분자 구현하기~~
- ~~YOLO~~로 객체 감지 / 탐지기 테스트
- ~~DarkNet~~ 프레임워크 사용하기

#### <조현국>

- 트랜스포머 논문 및 KOBERT, KOGPT 코드 분석하기

#### <조현국>

- 트랜스포머 논문 분석하기
- KOGPT-2를 이용한 가사 분석하기
- 분석된 가사 전이학습, 파인튜닝하기
- 프로젝트 발표자료 PPT 준비하기
- 이력서 및 개발보고서 준비하기

#### 12주차 ( 8/8 ~ 8/9 )

#### <서성민>

- FileUpload 기능 구현
- redux 를 활용하여 만든 게시판과 서버 연결하기 위해 연구중. (openapi)
- 화면에서 파일 업로드한 이미지 openapi (server) 로 전달하기 위해 연구중.

#### <권혜민>

- Muse GAN 구현 기능 점검하기
- YOLO 논문 분석 정리하기

#### <조현국>

- 작사하기 기능 최종 점검
- Transfomer 논문 분석 정리하기(이력서에 추가)

#### 11주차 ( 8/1 ~ 8/5 )

#### <서성민>

- FileUpload 기능 구현
- redux 를 활용하여 만든 게시판과 서버 연결하기 위해 연구중. (openapi)
- 화면에서 파일 업로드한 이미지 openapi (server) 로 전달하기 위해 연구중.

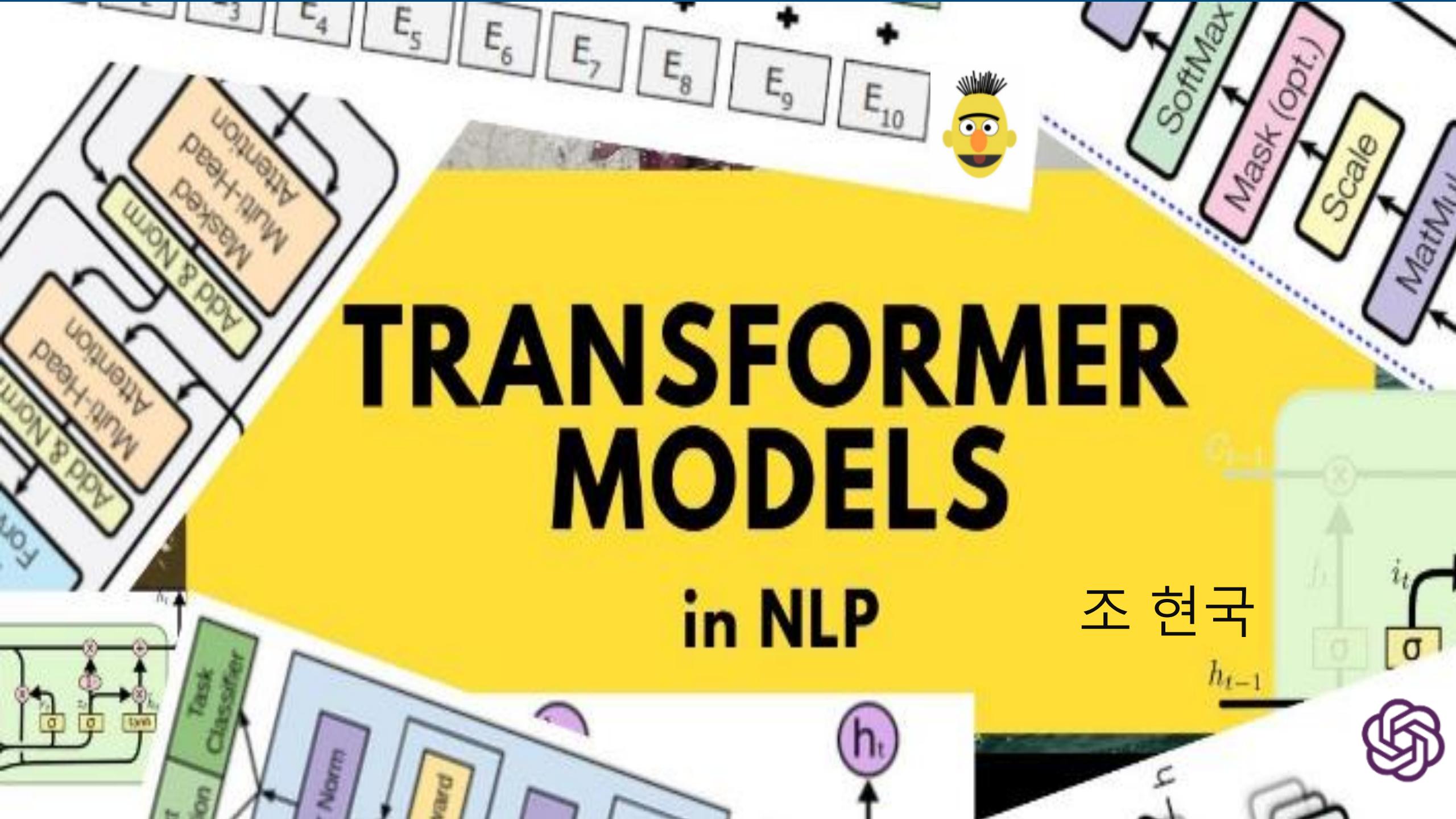
#### <권혜민>

- Muse GAN pytorch로 구현하기
- 데이터 데이터 받아서 구현하기
- Self-attention 동작원리 공부하기
- ViT 논문 공부 / 리뷰 하기

# TRANSFORMER MODELS

in NLP

조현국



# 목차

## Contents

#1, Introduce PLMs

#2, NLP 기법

#3, Attention

#4, TRANSFOMER

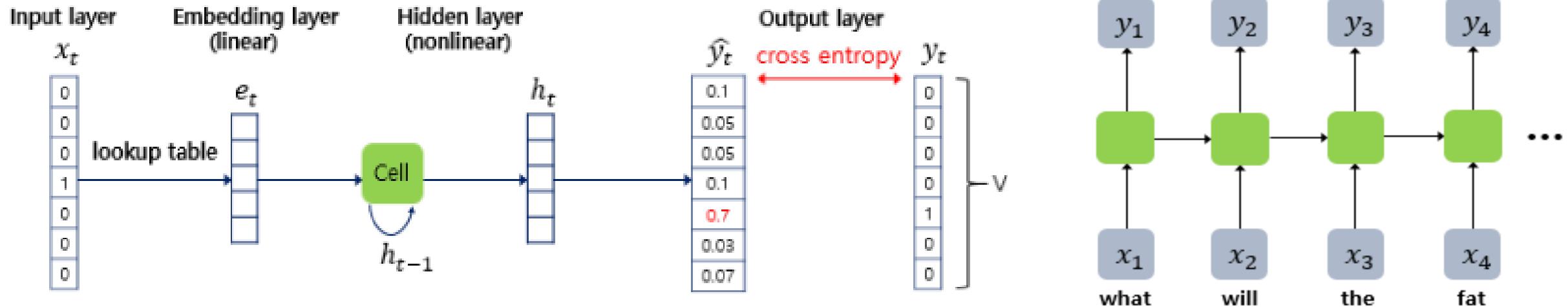


Part 1,

# Introduce PLMs



# RNN



- Recurrent Neural Network Language Model의 약자
- 시점(time step)의 개념이 도입
- 출력층 => SoftMax 함수
- 손실함수 => Cross-Entropy

# PLM(Pretrained Language Model)

## PLMs

(Pretrained  
Language Models)

BERT

GPT-3

BART

RoBERTa

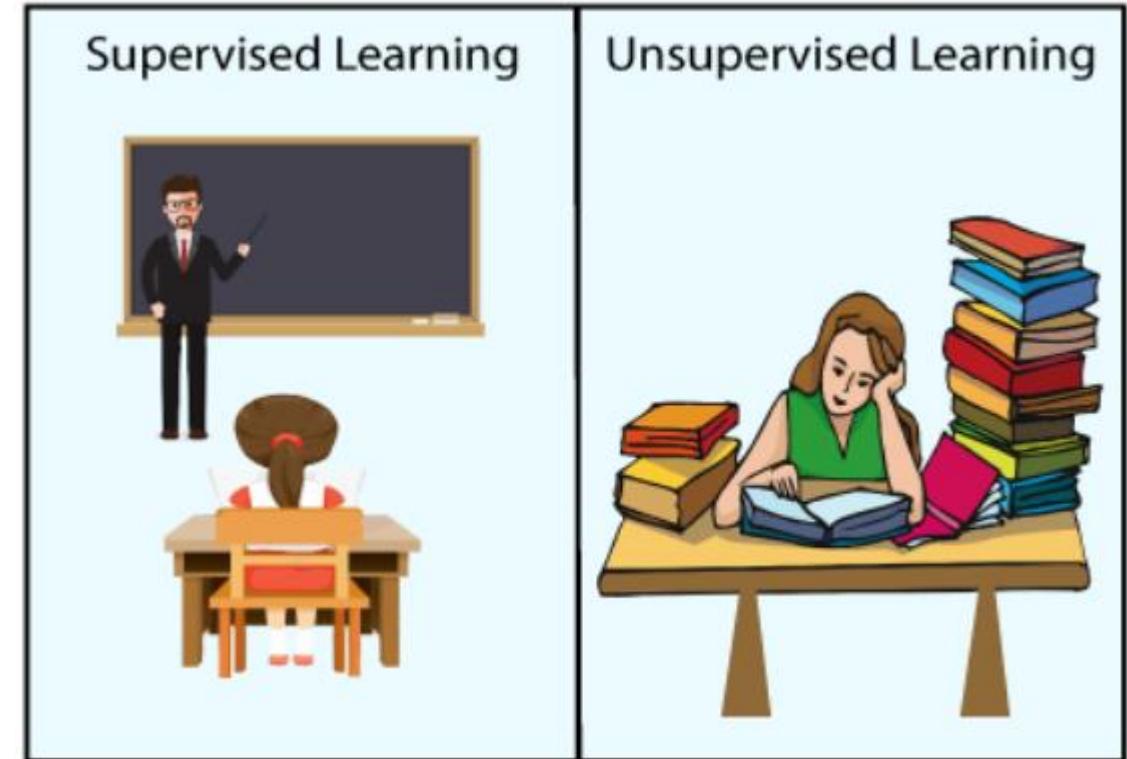
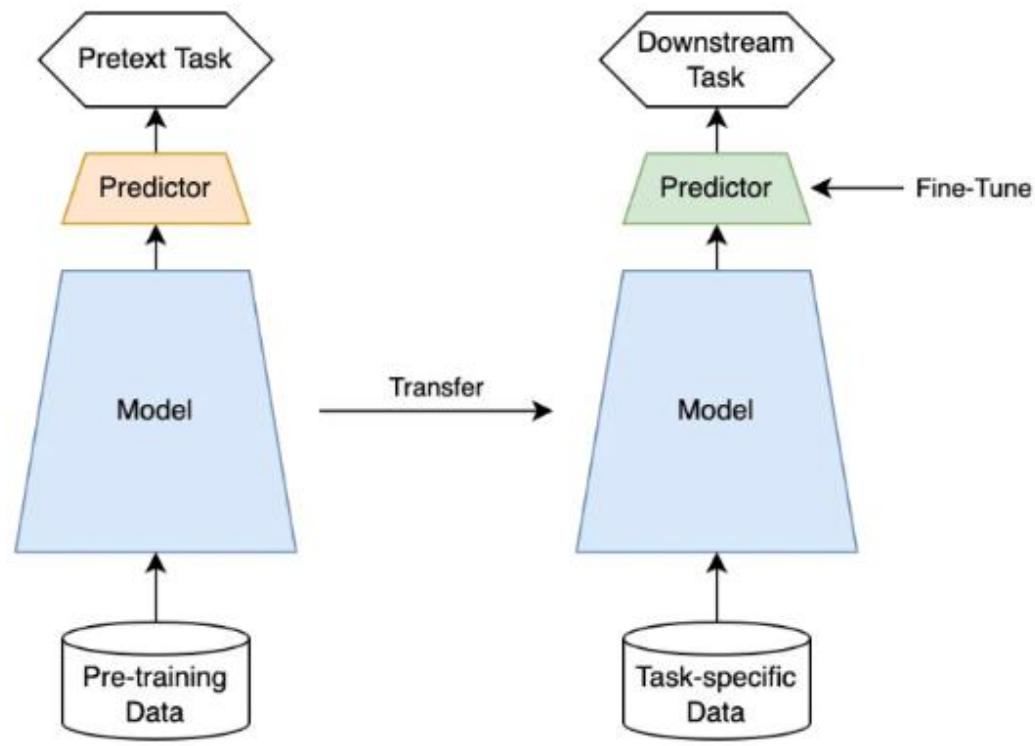
...

- 대규모의 데이터를 사전학습  
(Pretrain)한 모델

-Fine-Tuning을 하여 보다 적은 양  
의 데이터로 높은 성능을 얻을 수  
있다.

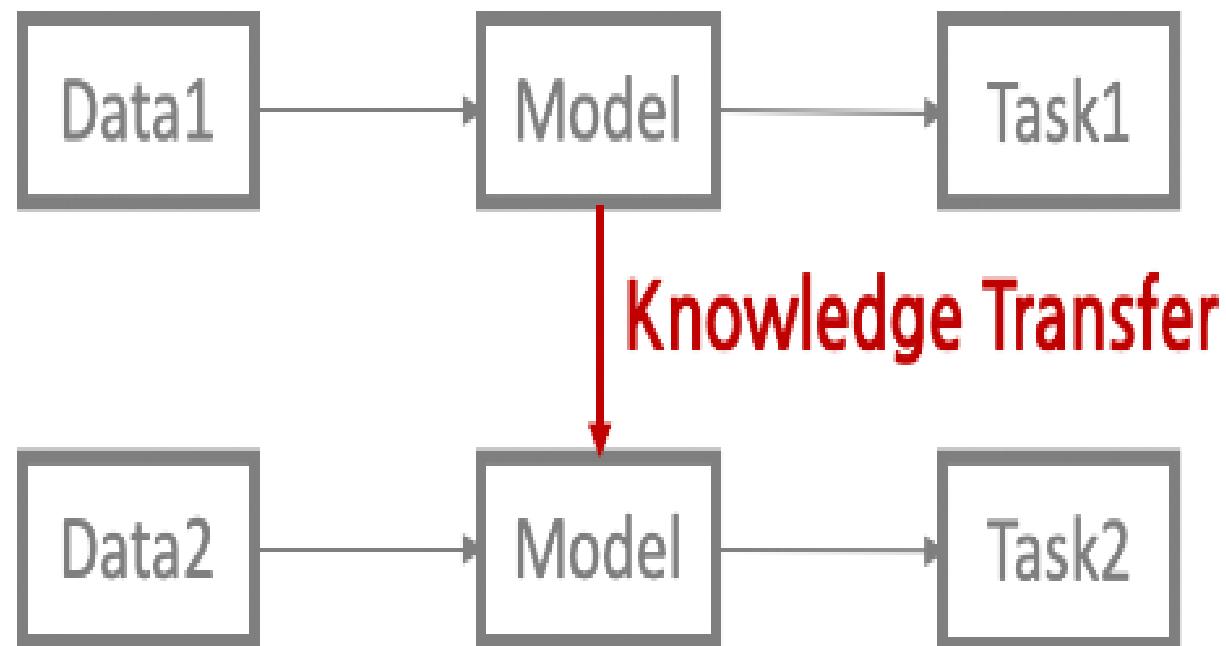
-대표적인 모델 GPT, BERT

# Self-Supervised Learning



데이터 내부의 구조(Inner Structure)를 활용하여 Label(정답)이 있는 것처럼 학습  
샘플의 일부 정보를 활용해 나머지 정보를 Label (정답)로 삼아 예측하도록 학습  
이 과정에서 나온 모델을 다른 작업(Task)에 Transfer-Learning해 성능을 극대화

# Transfer-Learning



- 특정 Task1을 학습한 모델을 다른 Task2 수행에 재사용

- 모델의 학습 속도가 빨라짐

- 학습 방식

## 1. Fine-Tuning

Downstream Task Data 전체사용  
Downstream Data에 맞게 모델 전체 UPDATE

## 2. Prompt-Tuning

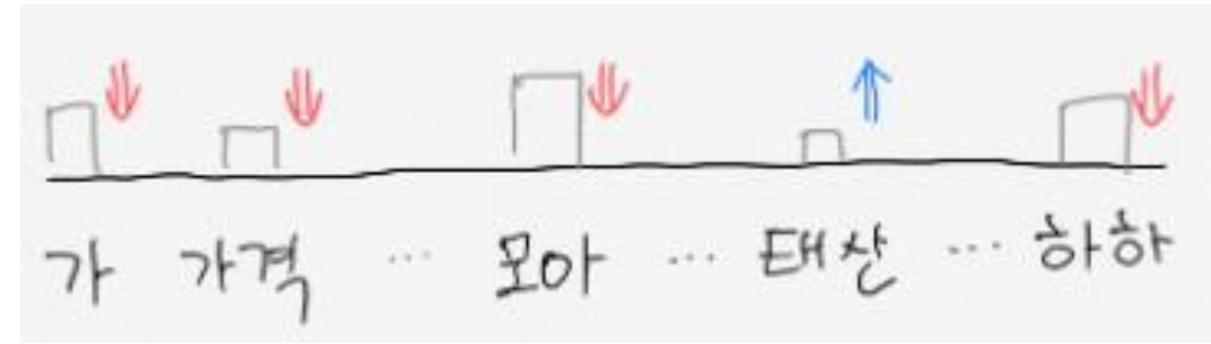
Downstream Task Data 전체사용  
Downstream Data에 맞게 모델 일부 UPDATE

## 3. In-context Learning (GPT-3)

Downstream Task Data 일부사용  
모델 UPDATE 없음

# UPSTEAM TASK(Pretrain - GPT)

티끌 모아



< 다음 단어 맞추기 >

문맥을 이해해 정답을 맞히는 모델

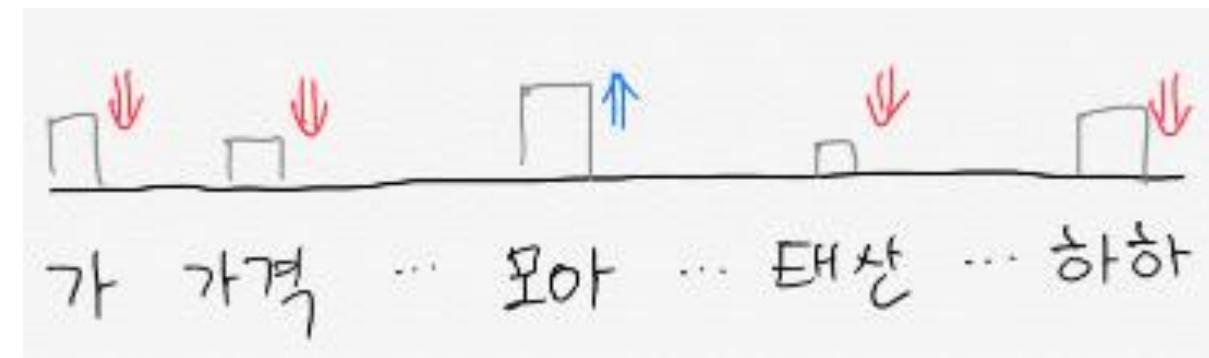
다음 단어의 정답인 태산이라는 단어의 확률을 높이고 나머지 단어들의 확률을 낮춘다.

# UPSTEAM TASK(Prettrain - BERT)

티끌



태산



< 빈칸 채우기 >

Masked Language Model(마스크 언어 모델)

빈칸에 들어갈 단어인 ‘모아’라는 단어의 확률을 높이고 나머지 단어들의 확률을 낮춘다.

# DOWNSTREAM TASK(Fine-Tuning)

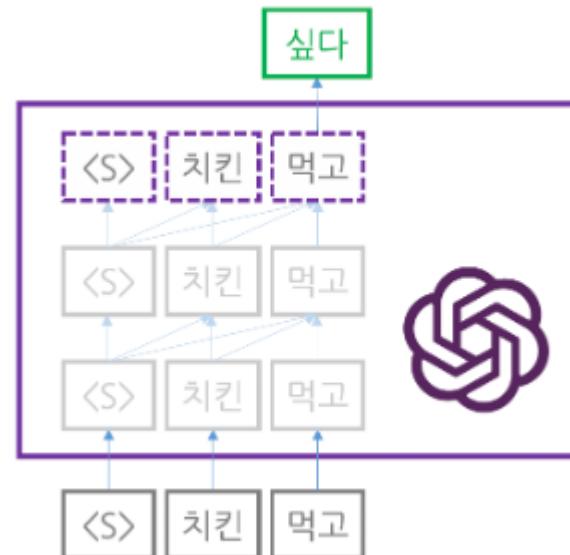
< 문서 분류 (BERT) >



< 개체명 인식 (BERT) >



< 문장 생성 (GPT) >



< 자연어 추론 (BERT) >



< 질의 응답 (BERT) >



Downstream Task의 본질은 분류(classification)

입력 받은 자연어가 어떤 범주에 해당하는지 확률 형태로 반환

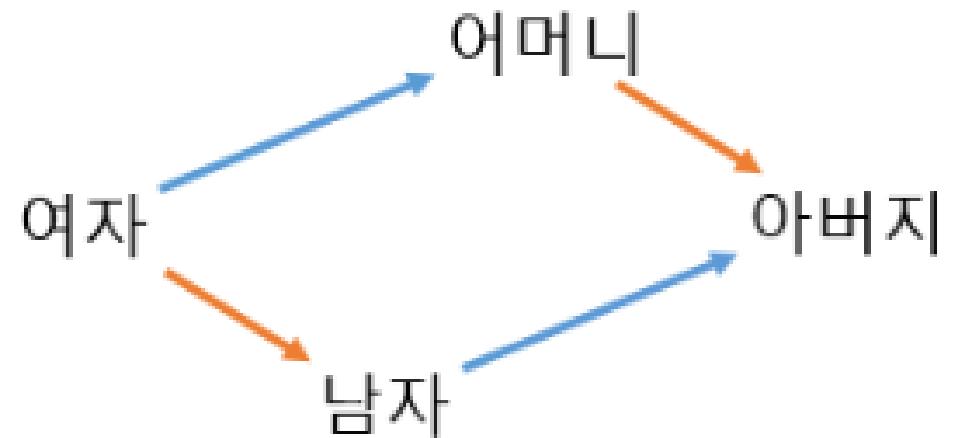
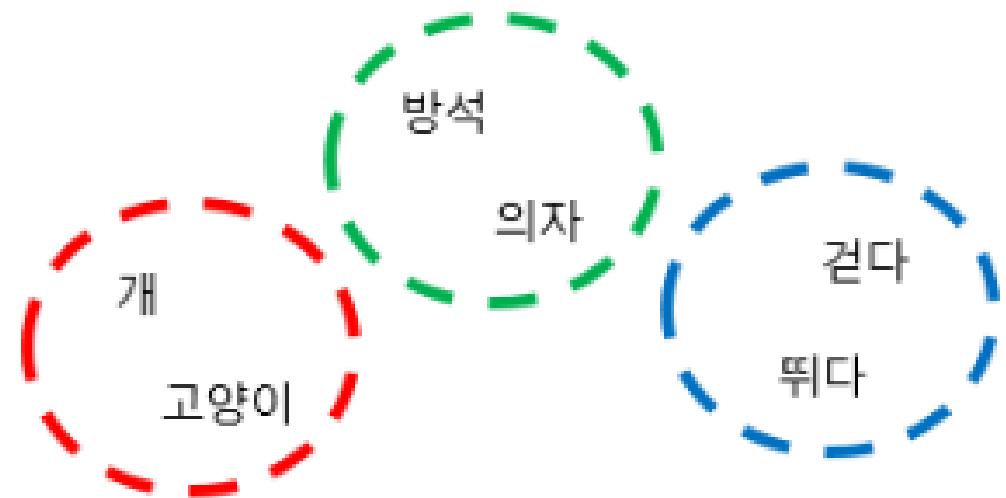


Part 2

# NLP 기법

# Word-Embedding

“유사 의미의 벡터는 가까운 공간에 존재” “단어를 벡터화 하여 산술 연산”



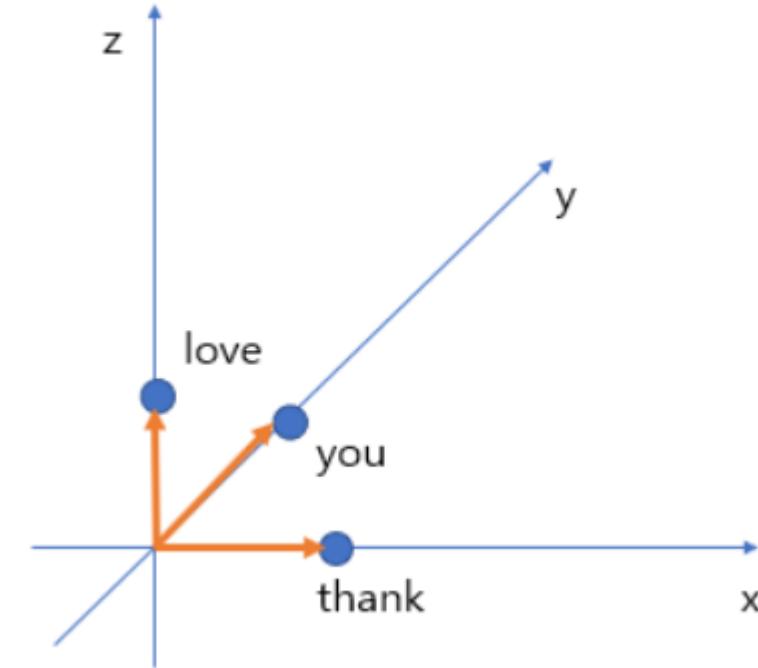
Word를 Dense Vector로 표현하는 방법(주변 단어)

Word2Vec에서 사용

유사도 검사 시 사용

# One-Hot-Encoding

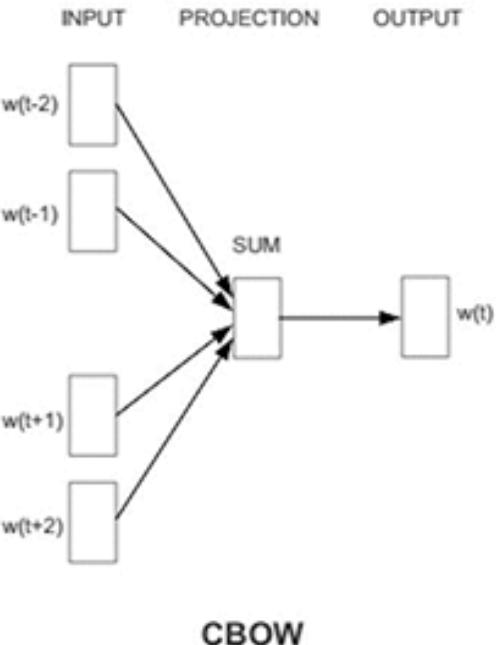
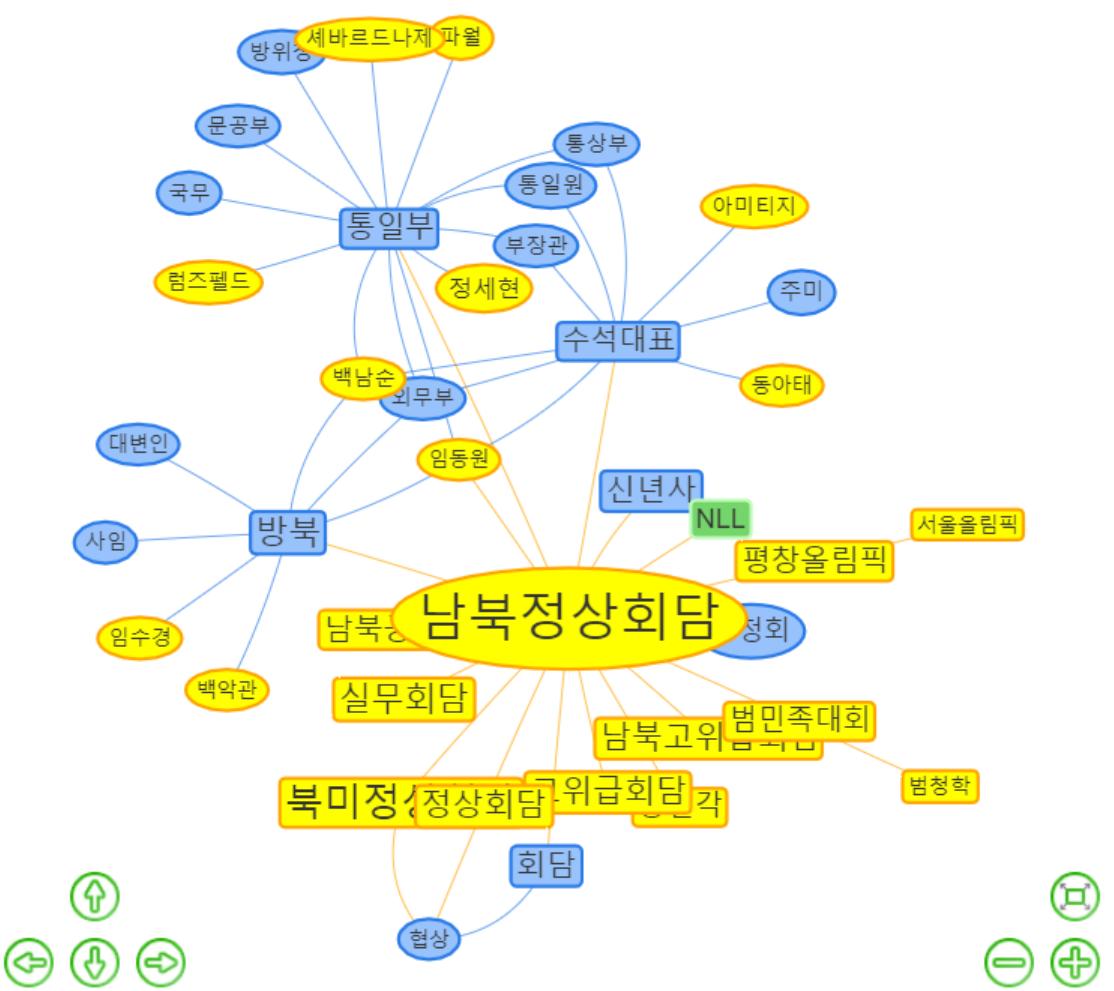
| 단어      | 단어 인덱스 | 원-핫 벡터                |
|---------|--------|-----------------------|
| you     | 0      | [1, 0, 0, 0, 0, 0, 0] |
| say     | 1      | [0, 1, 0, 0, 0, 0, 0] |
| goodbye | 2      | [0, 0, 1, 0, 0, 0, 0] |
| and     | 3      | [0, 0, 0, 1, 0, 0, 0] |
| I       | 4      | [0, 0, 0, 0, 1, 0, 0] |
| say     | 5      | [0, 0, 0, 0, 0, 1, 0] |
| hello   | 6      | [0, 0, 0, 0, 0, 0, 1] |



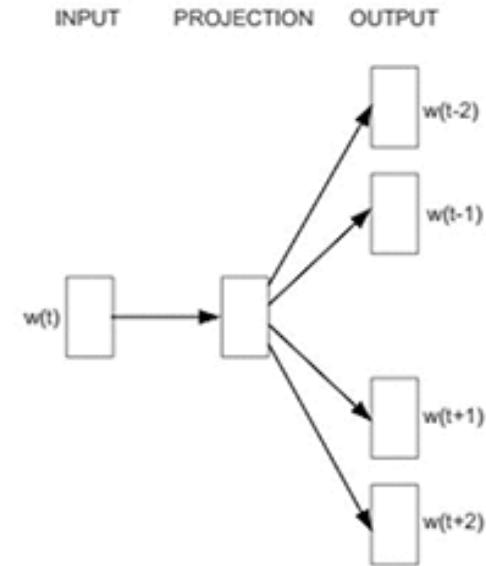
표현하고 싶은 단어의 인덱스에 1의 값을 부여하고 그 외 인덱스에 0을 부여하는 방식

부여된 벡터들을 **one-hot-vector**라 한다.

# Word2Vec



## CBOW



## Skip-gram

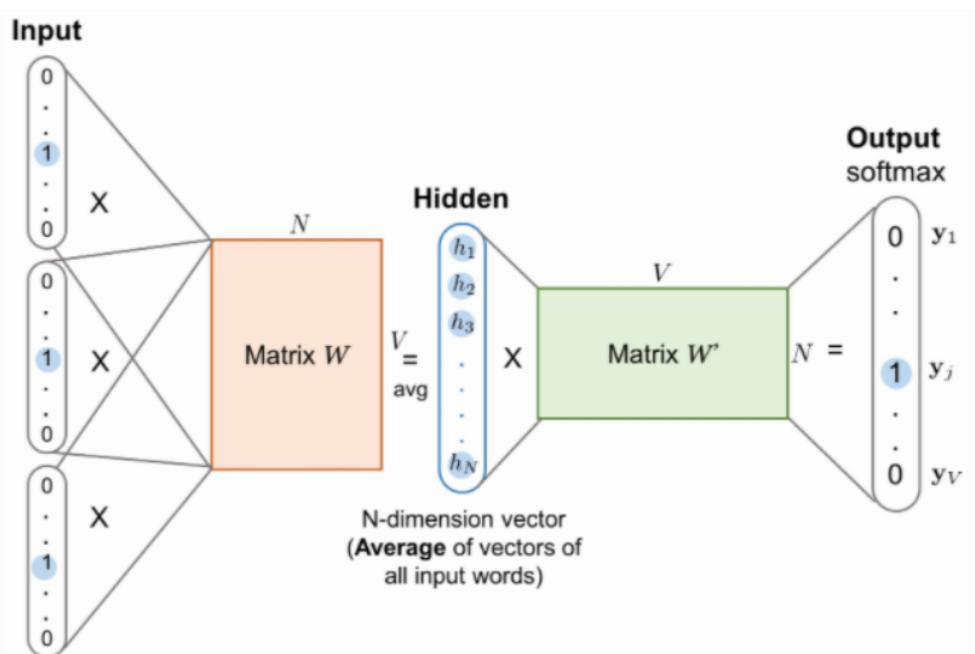
통계 기반 기법으로 단어 빈도수를 기반하여 벡터로 표현

# NNLM(Neural Net Language Model) 기반 대량의 문서를 벡터공간에 고수준의 의미를 가진 벡터를 가지도록 하는 모델

CBOW 방식, Skip-gram 방식이 있다.

# CBOW

You ? goodbye and I say hello.  
윈도우크기 = 1



중심 단어  
The fat cat sat on the mat

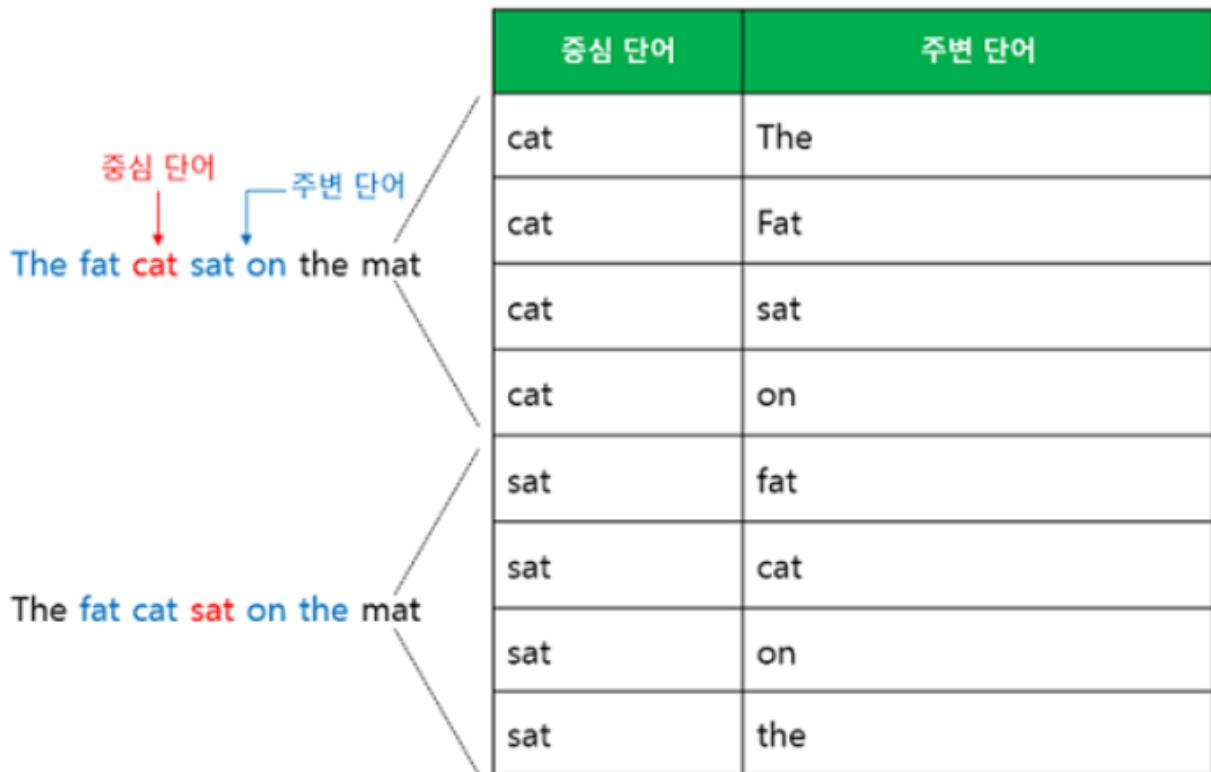
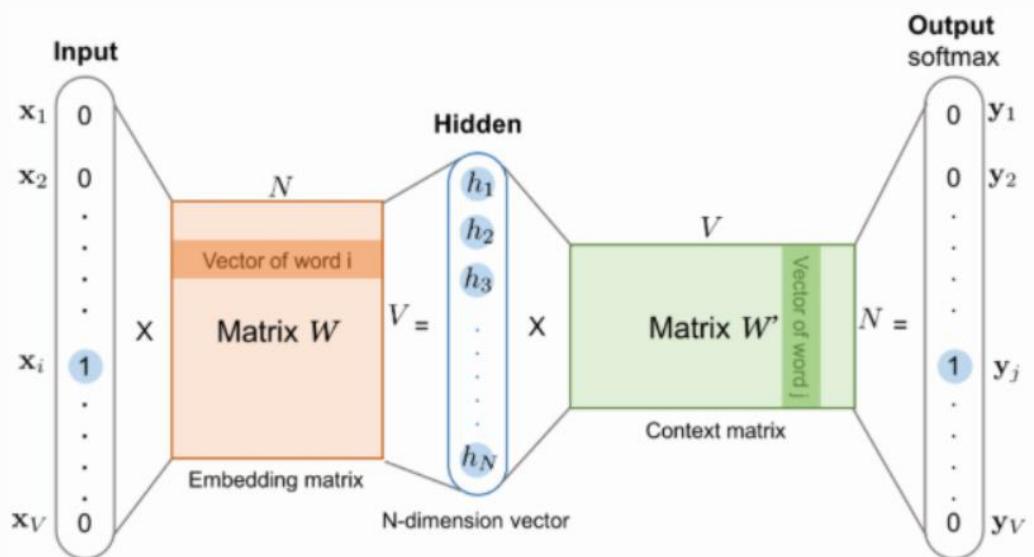
주변 단어  
The fat cat sat on the mat

| 중심 단어              | 주변 단어  |
|--------------------|--|
| [1, 0, 0, 0, 0, 0] | [0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0]   |
| [0, 1, 0, 0, 0, 0] | [1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0]                           |
| [0, 0, 1, 0, 0, 0] | [1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 1, 0, 0]    |
| [0, 0, 0, 1, 0, 0] | [0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0] |
| [0, 0, 0, 0, 1, 0] | [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1] |
| [0, 0, 0, 0, 0, 1] | [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]   |

주변단어를 기준으로 중심단어를 예측하는 모델

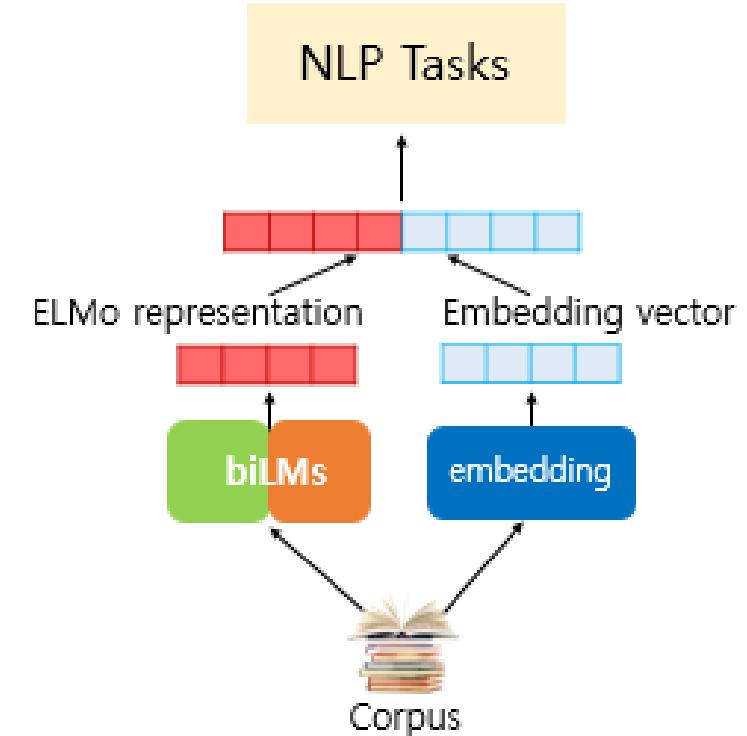
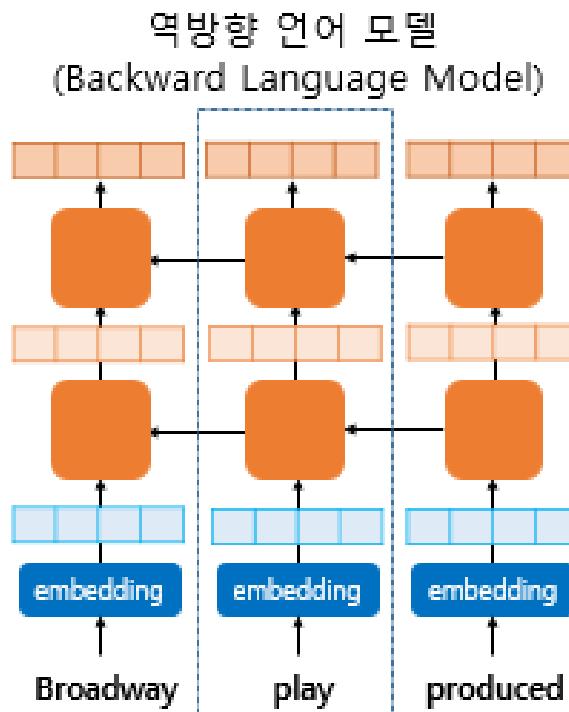
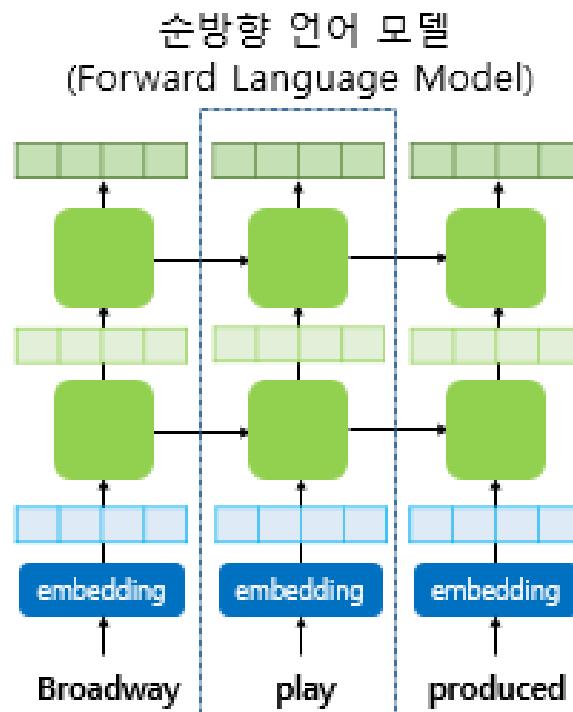
# Skip-gram

say and I say hello.



중심단어를 기준으로 주변단어를 예측하는 모델

# ELMo



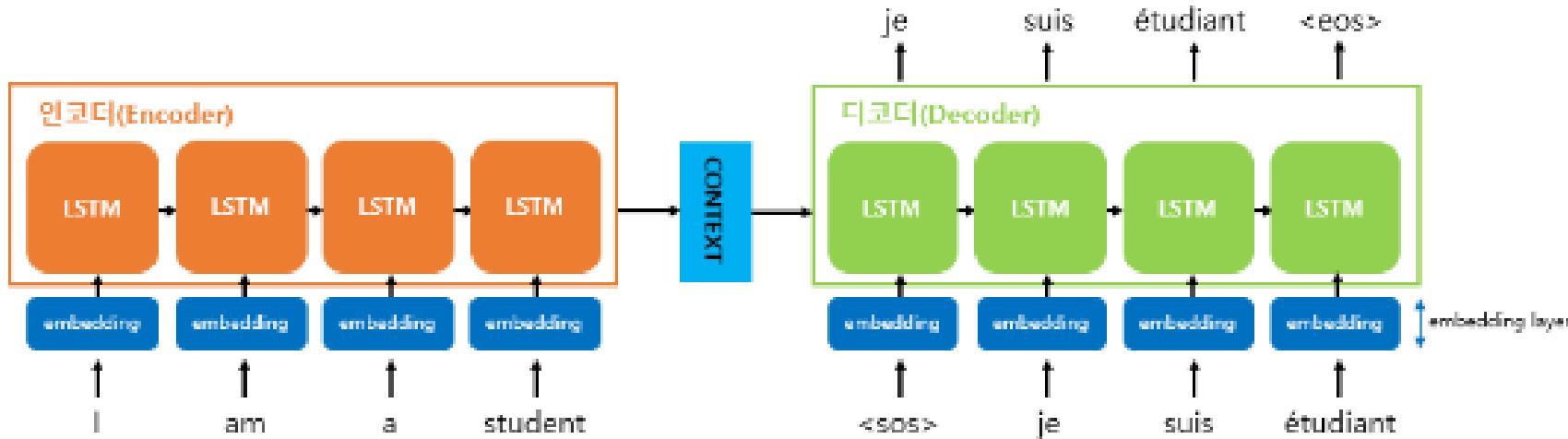
Pre-trained Language Model이며, Word-Embedding 시 문맥을 고려하여 성능을 극대화

Word2Vec이 단어 중심이였다면, ELMo는 문맥 즉, 문장을 고려한다.

# PLM과 Word2Vec, ELMo의 차이점

- Word2Vec과 ELMo의 방식
  - Feature를 사전 학습하는 방식  
Embeding Layer에 가중치(weight)만 사전학습  
(Fine-Tuning을 할 수 없다.)
  - 추후 PLM방식
- Model Parameter 자체를 사전 학습하는 방식  
(Fine-Tuning이 가능하다.)

# Seq2seq



Encoder 와 Decoder라는 두개의 모듈로 나뉜다.

## Encoder

입력 문장의 모든 단어들을 순차적으로 입력 받은 뒤에 모든 단어들을 압축하여 하나의 Vector로 만든다.  
(0) Vector를 context vector라 부른다.)

- Attention(Key) => Hidden state

## Decoder

Encoder에서 압축된 context vector를 받아 번역된 단어를 하나씩 순차적으로 출력한다.

- Attention(Query) => Hidden State

# Seq2seq의 문제점

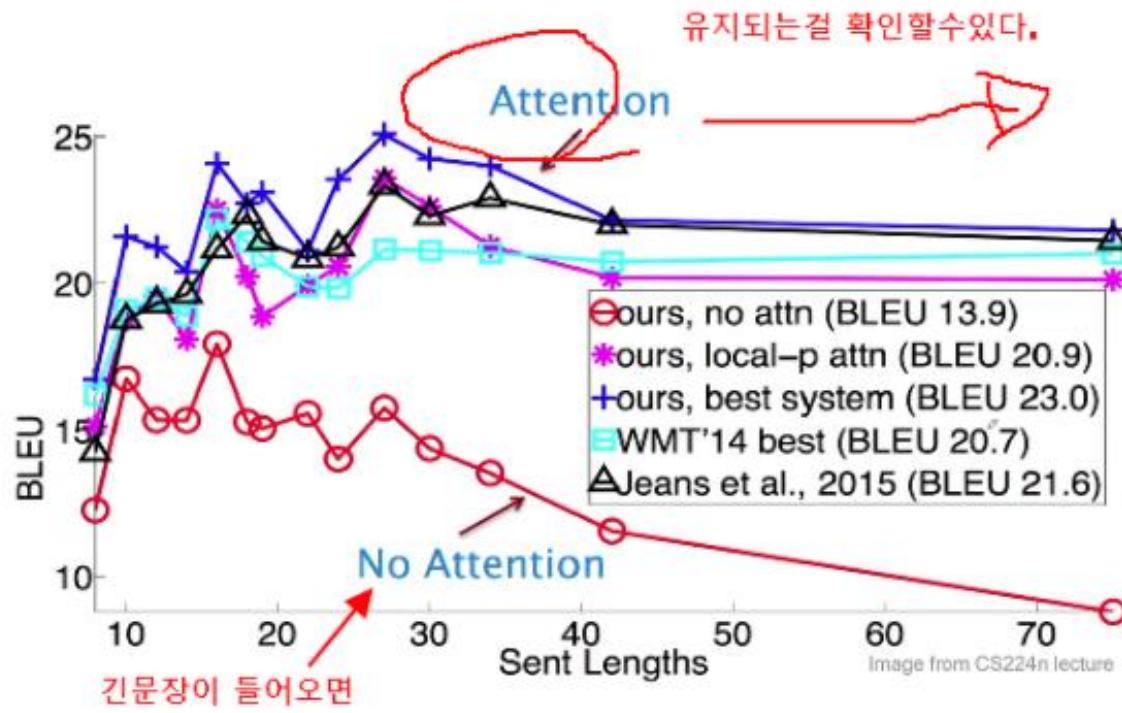
- 하나의 고정된 **context vector**에 모든 정보를 압축  
    ⇒ 정보 손실 발생
  - RNN의 고질적인 문제  
        => Vanishing Gradient(기울기 소실) 발생
  - 즉, 번역 및 입력한 문장의 길이가 길어질 수록  
        번역 품질(성능)이 떨어지는 현상이 발생

Part 3

# Attention



# Attention Mechanism

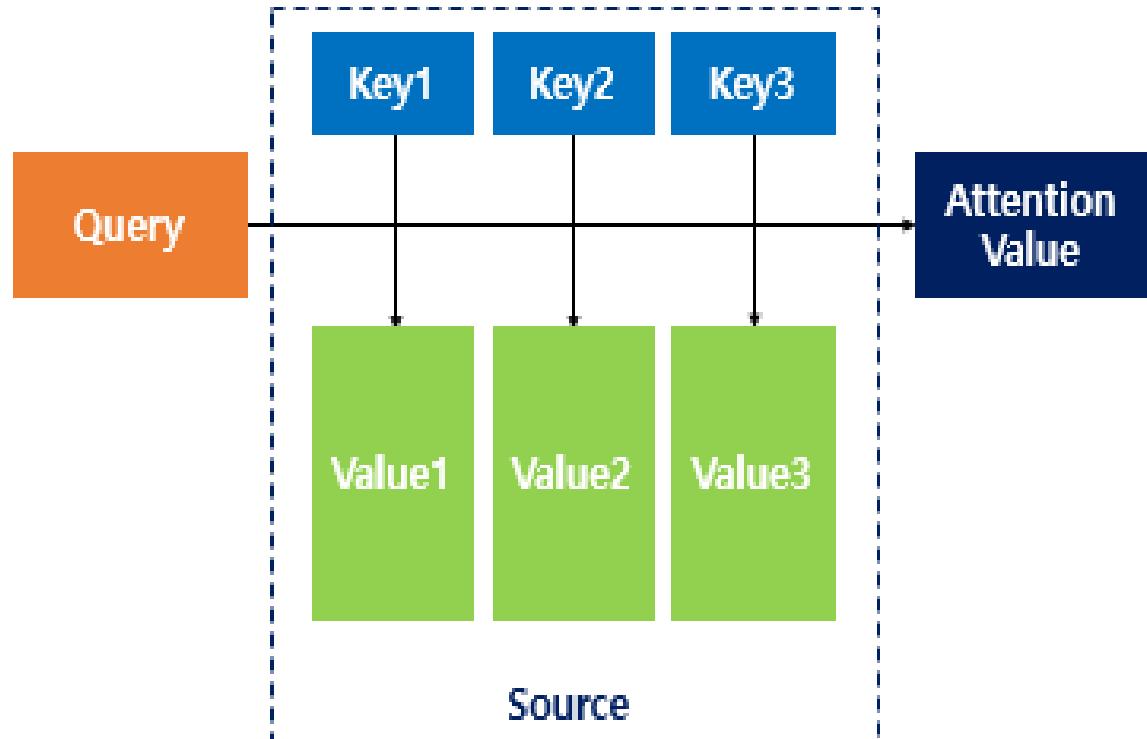


Seq2seq의 문제점을 보완하고 긴 문장의 정확도를 보정하는 기법

Decoder에서 출력할 단어를 예측하는 time-step마다 Encoder에서 입력 받은 시퀀스를 다시 참고한다.

(이때 모든 단어를 참고하지 않고 예측 단어와 관련이 있는 입력단어를 치중해서 본다.)

# Attention 입력



Attention 함수의 입력은 Query, Key, Value이다.

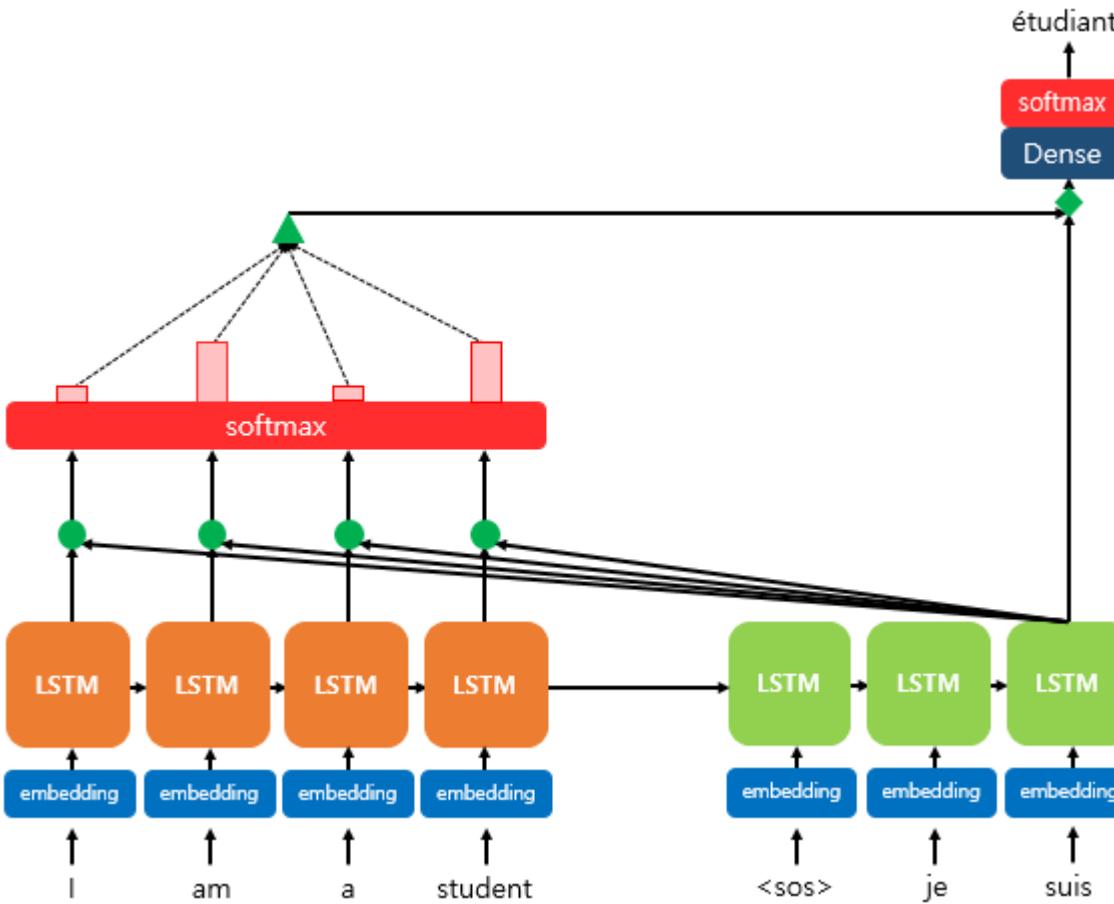
$$\text{Attention}(Q, K, V) = \text{Attention Value}$$

$Q = t$  시점의 Decoder cell의 은닉상태  
 $K, V =$  모든 시점의 Encoder Cell의 은닉상태들

-- 진행 --

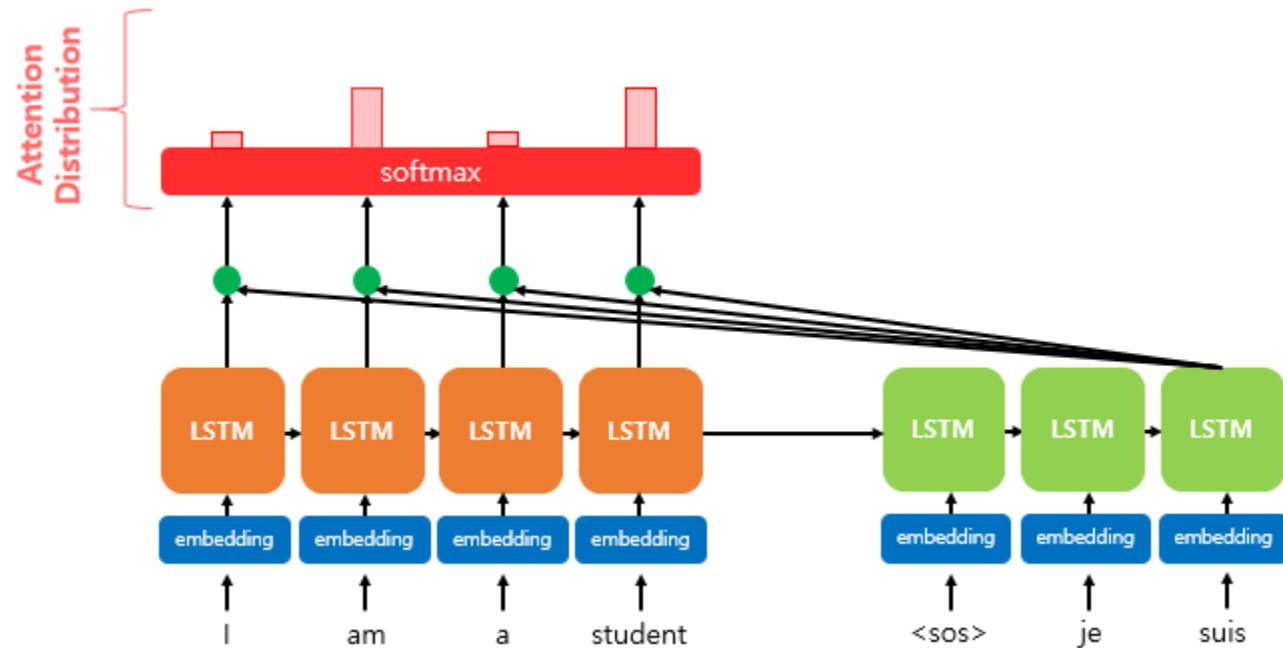
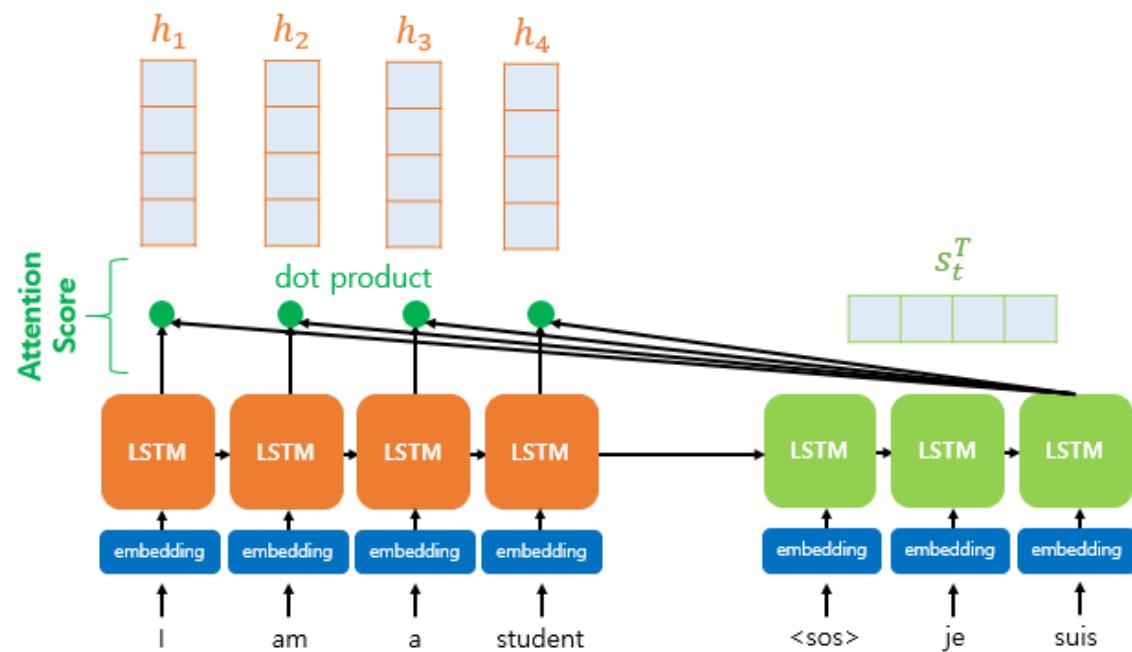
1. Q에 대해 모든 K와의 유사도를 구한다.
2. 해당 유사도를 K와 매핑 되어있는 각각의 V에 반영한다.
3. 유사도가 반영된 값을 모두 더해 Return한다.

# Dot-Product Attention



Attention의 종류 중 이해가 쉬운 Dot-Product Attention으로 설명  
Decoder의 세번째 LSTM에서 연결된 선이 바로 Attention이다

# Dot-Product Attention



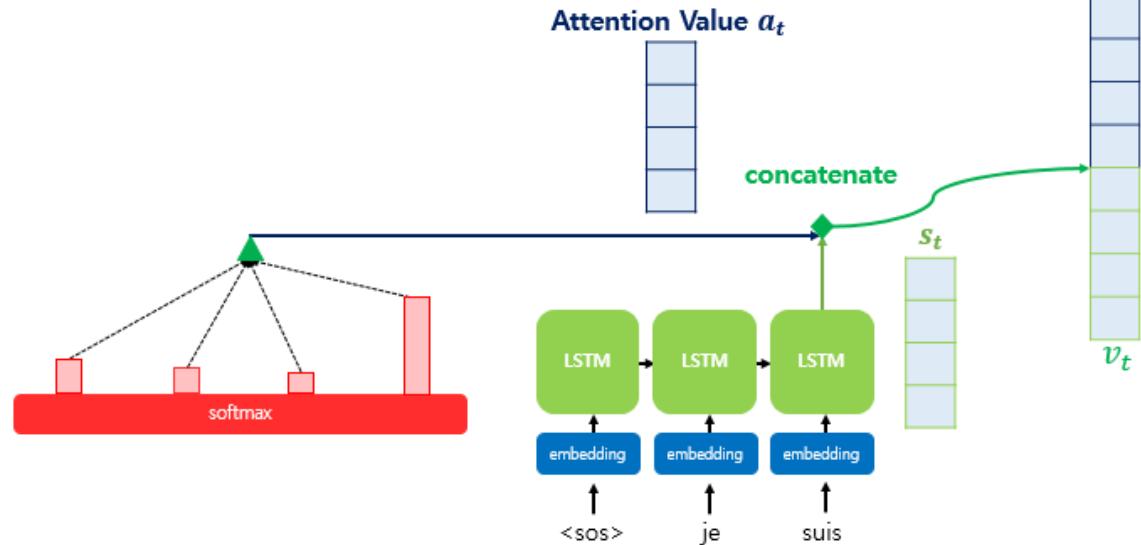
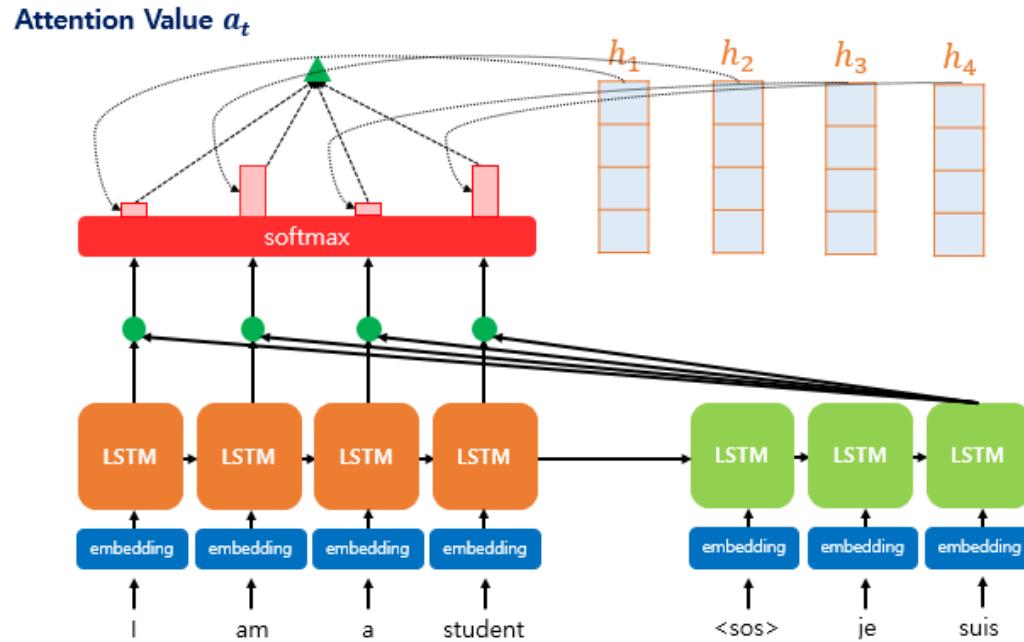
## 1. 왼쪽 그림

Attention Score를 구하는 것 현재 Decoder시점  $t$ 에서 단어를 예측하기 위해 Encoder의 모든 은닉상태 각각 Decoder의 현시점의 은닉상태  $s_t$ 와 얼마나 유사한지를 판단하는 스코어 값이다.

## 2. 오른쪽 그림

활성화 함수인 Softmax함수를 통해 Attention 분포를 구한다.  
Encoder의 은닉상태에서의 Attention Weight의 크기를 직사각형으로 시각화 하였다.

# Dot-Product Attention



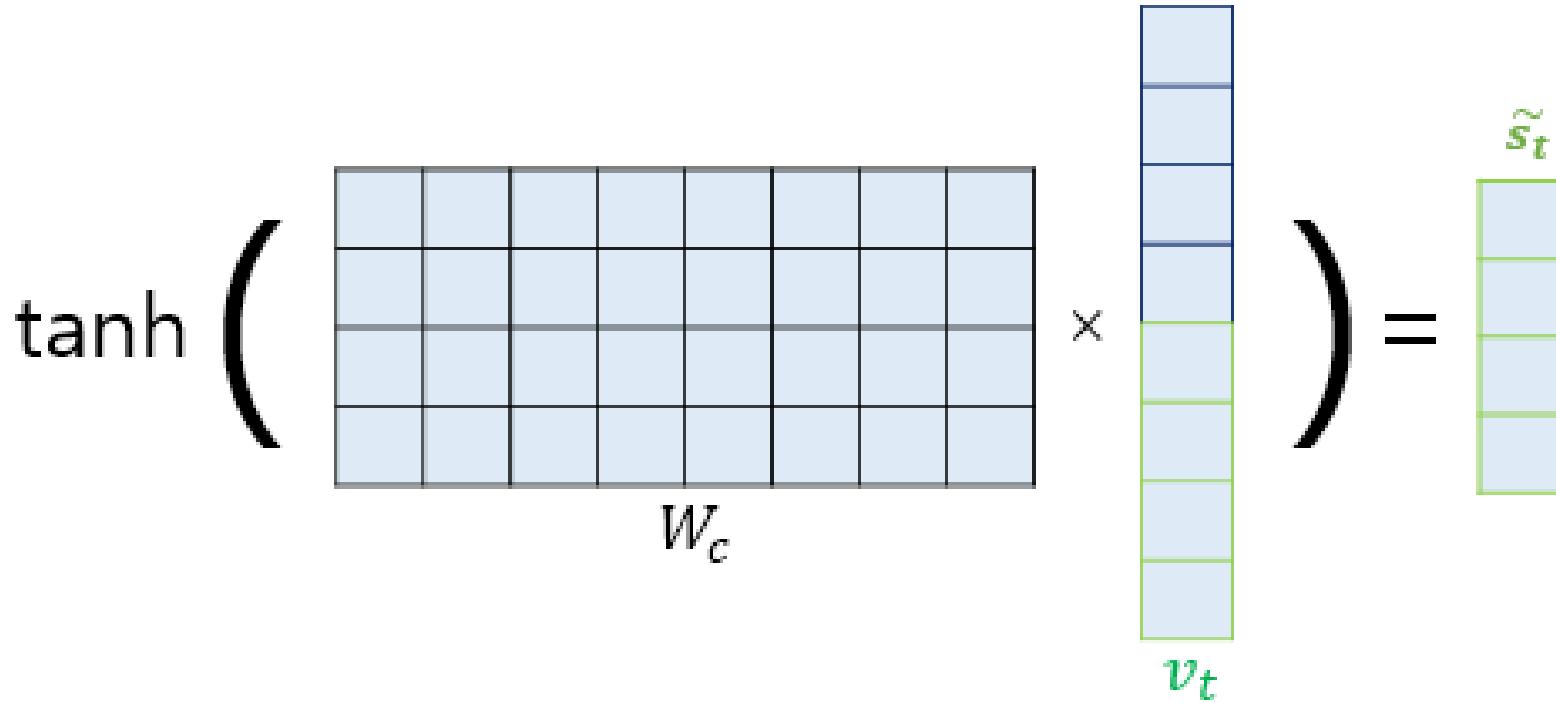
왼쪽 그림

각 Encoder의 Attention Weight와 은닉 상태를 합쳐 Attention값(at)을 구한다.

오른쪽그림

Attention 값과 Decoder의 t시점의 은닉상태를 연결한다.(Concatenate)  
이때  $v_t$ 를  $y^*$  예측 연산의 입력으로 사용하여 Encoder의 정보를 활용해  $y^*$ 를  
좀 더 잘 예측 할 수 있게 된다. 이것이 바로 Attention의 핵심이다.

# Dot-Product Attention



가중치 행렬과 곱한 후에 하이퍼볼릭탄젠트 함수를 지나 출력층 연산을 위한 새로운 Vector  $\sim s_t$ 를 가진다.  
Seq2seq의 출력층 입력이 t시점의 은닉상태인  $s_t$ 였으나, Attention에서는  $\sim s_t$ 가 되는 것이다.

# Attention의 다른 종류

| 이름                     | 스코어 함수   | Defined by                |
|------------------------|--|---------------------------|
| <i>dot</i>             | $score(s_t, h_i) = s_t^T h_i$                                      | Luong et al.<br>(2015)    |
| <i>scaled dot</i>      | $score(s_t, h_i) = \frac{s_t^T h_i}{\sqrt{n}}$                     | Vaswani et<br>al. (2017)  |
| <i>general</i>         | $score(s_t, h_i) = s_t^T W_a h_i$ // 단, $W_a$ 는 학습 가능한 가중치 행렬      | Luong et al.<br>(2015)    |
| <i>concat</i>          | $score(s_t, h_i) = W_a^T \tanh(W_b[s_t; h_i])$                     | Bahdanau<br>et al. (2015) |
| <i>location - base</i> | $\alpha_t = softmax(W_a s_t)$ // $\alpha_t$ 산출 시에 $s_t$ 만 사용하는 방법. | Luong et al.<br>(2015)    |

**st => Query**  
**hj => Keys**  
**Wa, Wb => Weight**

Part 4

# TRANSFOMER



Part 5

# BERT



Part 6

GPT-2



Part 7

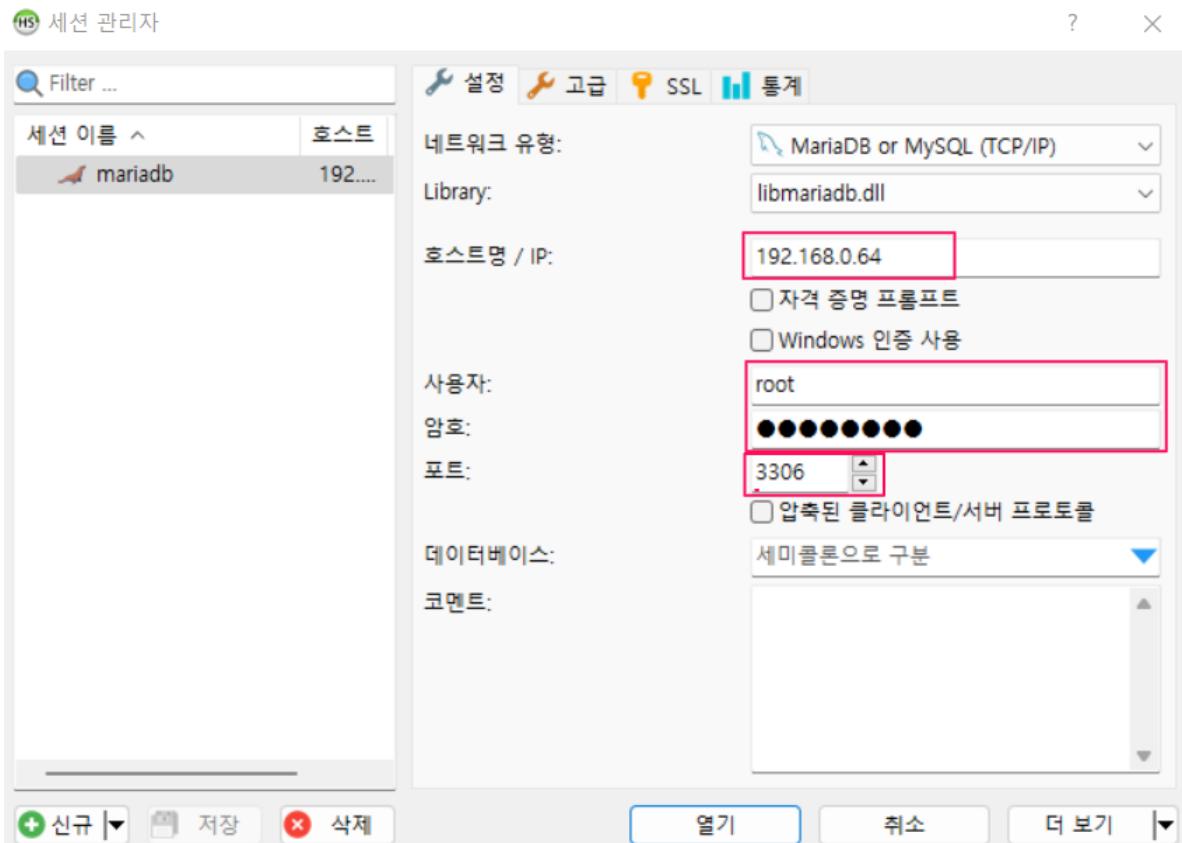
# GPT-3





감사합니다.

# 우분투 - mariadb 연결 with HeidiSQL



내 노트북에서 heidisQL 실행

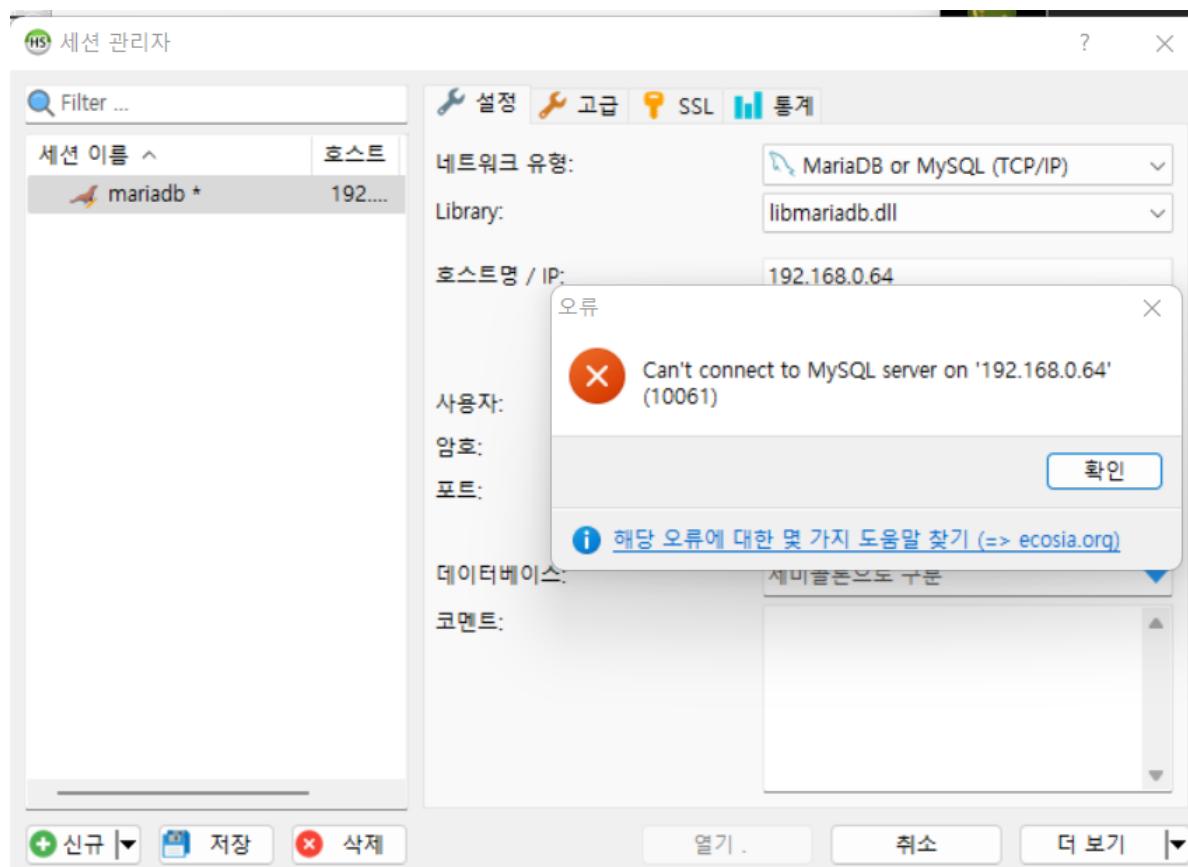
ip에 우분투 ip를 입력

사용자/ 암호에 본인이 설정한 이름과 암호를 입력한다.

포트는 기본 db 포트인 3306 이지만 커스텀 했다면 본인 커스텀 port를 입력한다.

---

정확히 입력했음에도 아래와 같은 에러가 난다면 원격 접속 허용을 위한 설정을 시작한다.



```
mysql> SELECT Host,User,plugin,authentication_string FROM mysql.user;
+-----+-----+-----+
| Host | User | plugin           | authentication_string |
+-----+-----+-----+
| localhost | root | mysql_native_password | *8024A6913C57E024BDFC6E813A57DFB924E6803A |
+-----+-----+-----+
2 rows in set (0.000 sec)
```

mysql에 들어가 `host,User,plugin,authentication_string FROM mysql.user;`  
명령어로 접속 가능 상태를 확인한다.

위 사진과 같이 나온다면 root에 대한 host가 자기 자신으로만 되어있는 상황이다.

mysql 상태에서

`GRANT ALL PRIVILEGES ON *.* TO '아이디'@'%' IDENTIFIED BY '본인 패스워드';`  
해당 쿼리문을 입력하여

root 행에 대해서 localhost 대신 '%'로 update하여 모든 ip에 대해서 접근이 가능하게 한다.

다음로 mysql 설정에서 bind-address에 해당하는 설정을 바꾸어 주어야 한다.

참고 블로그 마다 bind-address 가 있는 폴더가 다른으로 아래 명령어를 통해 해당 텍스트가 있는 문서를 찾는다.

```
sudo grep -ir 'bind-address' /etc/mysql/
```

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo grep -ir 'bind-address' /etc/mysql/  
/etc/mysql/mariadb.conf.d/50-server.cnf:bind-address = 127.0.0.1  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ vi /etc/mysql/mariadb.conf.d/50-server.cnf  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo mysql -u root -p  
Enter password:
```

그러면 이렇게 해당 텍스트가 있는 폴더 주소가 나온다.

```
/etc/mysql/mariadb.conf.d/50-server.cnf  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ vi /etc/mysql/mariadb.conf.d/50-server.cnf  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$
```

해당 파일 안의 설정을 바꾸기 위해 명령어를 입력하여 접속한다.

### vi 찾은 파일 주소

```
#  
# See the examples of server my.cnf files in /usr/share/mysql  
  
# this is read by the standalone daemon and embedded servers  
[server]  
  
# this is only for the mysqld standalone daemon  
[mysqld]  
  
#  
# * Basic Settings  
#  
user = mysql  
pid-file = /run/mysqld/mysqld.pid  
socket = /run/mysqld/mysqld.sock  
port = 3306 주석 해제  
basedir = /usr  
datadir = /var/lib/mysql  
tmpdir = /tmp  
lc-messages-dir = /usr/share/mysql  
#skip-external-locking  
  
# Instead of skip-networking the default is now to listen only on  
# localhost which is more compatible and is not less secure.  
  
bind-address = 127.0.0.1 주석 걸어주기  
  
#  
# * Fine Tuning  
#  
#key_buffer_size = 16M  
1 change; before #2 68 초 전
```

29,1 2%

bind-address와 위와 같이 되어 있다면 주석을 걸어 모든 ip 주소에서 접속할 수 있도록 해준다.

\*\*만약 수정 후에도 저장이 안되거나 읽기 모드로 나온다면 해당 파일에 수정 권한을 주어야 한다.

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo su  
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# chmod -R 777 /etc/mysql/mariadb.conf.d/50-server.cnf
```

chmod -R 777 filename

<https://88240.tistory.com/13> 추가 설명은 해당 블로그를 참고한다.

(혹시 에러가 난다면 sudo su 명령어로 관리자 모드로 접속하여 다시 해보자!)

위 과정을 모두 수행한 후 노트북에서 맨 위 과정을 해주면 접속 완료!

---

# 우분투 mariadb 설치

```
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# apt install mariadb-server ↓
```

```
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...↓
```

```
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# apt-get install mariadb-client↓
```

```
패키지 목록을 읽는 중입니다... 완료↓
```

**apt install mariadb-server**

**apt-get install mariadb-client**

먼저 마리아디비 서버와 클라이언트를 설치한다.

---

만약 위에 명령어를 넣었을때

```
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# sudo mysql_secure_installation↓
```

```
↓
```

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB!↓
```

**sudo mysql\_secure\_installation**

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ apt install mariadb-server↓
```

```
E: 잠금 파일 /var/lib/dpkg/lock-frontend 파일을 열 수 없습니다 - open (13: 허가 거부)↓
```

```
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?↓
```

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ^C↓
```

해당 에러가 난다면

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ^C↓
```

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo su↓
```

```
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# readvice↓
```

```
-----
```

**sudo su**

해당 명령어로 관리자 모드로 변환 후 명령어를 다시 입력해 준다.

---

```
↓  
Change the root password? [Y/n] y ↓
```

```
New password: ↓
```

```
Re-enter new password: ↓
```

```
Password updated successfully!↓
```

- root 비밀 번호 설정 질문  
변경을 원하면 y를 하고 새 비밀 번호를 설정해 준다

```
↓  
Remove anonymous users? [Y/n] y↓  
... Success!↓  
↓
```

- 익명 계정을 지우겠는가?

```
↓  
Disallow root login remotely? [Y/n] y↓  
... Success!↓  
↓
```

- root계정으로 원격에서 접속을 허용할 것 인가?  
원격 접속을 사용할 것이라면 n!!!!!!!!!!

```
↓  
Remove test database and access to it? [Y/n] y↓  
- Dropping test database...↓  
... Success!↓  
- Removing privileges on test database...↓  
... Success!↓
```

- test 데이터베이스를 삭제할 것인가

```
www-data@www-data:~$  
↓  
Reload privilege tables now? [Y/n] y↓  
... Success!↓  
↓
```

- 지금까지 설정한 내용을 즉시 반영하겠는가?

```
Thanks for using MariaDB!↓
root@readvice-HP-EliteDesk-800-G4-TWR:/home/readvice# sudo mysql -u root -p↓
Enter password: ↓
Welcome to the MariaDB monitor. Commands end with ; or \w\g.↓
Your MariaDB connection id is 44.
```

**sudo mysql -u root -p**

마리아 디비에 접속해 설정이 잘 완료 되었는지 확인한다.

```
MariaDB [(none)]> show databases;↓
+-----+↓
| Database |↓
+-----+↓
| information_schema |↓
| mysql |↓
| performance_schema |↓
+-----+↓
3 rows in set (0.000 sec)↓
↓
MariaDB [(none)]> select version();↓
+-----+↓
| version() |↓
+-----+↓
| 10.3.34-MariaDB-0ubuntu0.20.04.1 |↓
+-----+↓
1 row in set (0.000 sec)↓
↓
MariaDB [(none)]>←
```

# 우분투 원격 접속

우분투가 설정되어 있는 컴퓨터에서 터미널을 열어준다.

터미널 단축키 : ctrl + alt + t

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo apt-get install xrdp
[sudo] readvice 암호: 
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
```

**sudo apt-get install xrdp**

해당 명령어를 통해 원격 접속이 가능하도록 xrdp를 설치한다.

nano 가 아닌 vim을 사용하여 설정하기 위해 vim을 설치해 준다.

```
local.d/45-allow-colord.pkla
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo apt-get update
기존:1 http://security.ubuntu.com/ubuntu focal-security InRelease
기존:2 http://kr.archive.ubuntu.com/ubuntu focal InRelease
기존:3 http://kr.archive.ubuntu.com/ubuntu focal-updates InRelease
기존:4 http://kr.archive.ubuntu.com/ubuntu focal-backports InRelease
패키지 목록을 읽는 중입니다... 완료
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo apt-get install vim
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
```

**sudo apt-get update**

**sudo apt-get install vim**

```
Processing triggers for man-db (2.9.1-1) ...
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo vim /etc/polkit-1/localauthority/50-local.d/45-allow-colord.pkla
```

**sudo vim /etc/polkit-1/localauthority/50-local.d/45-allow-colord.pkla**

vim으로 해당 파일에 접속해

```
[Allow Colord all Users]
```

```
Identity=unix-user:*
```

```
Action=org.freedesktop.color-manager.create-device;org.freedesktop.color-manager.create-profile;org.freedesktop.color-manager.delete-device;org.freedesktop.color-manager.delete-profile;org.freedesktop.color-manager.modify-device;org.freedesktop.color-manager.modify-profile  
ResultAny=no  
ResultInactive=no  
ResultActive=yes
```

해당 명령어를 삽입 -> esc -> :wq!

```
local.d/45-allow-colord.pkla↓  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ifconfig↓
```

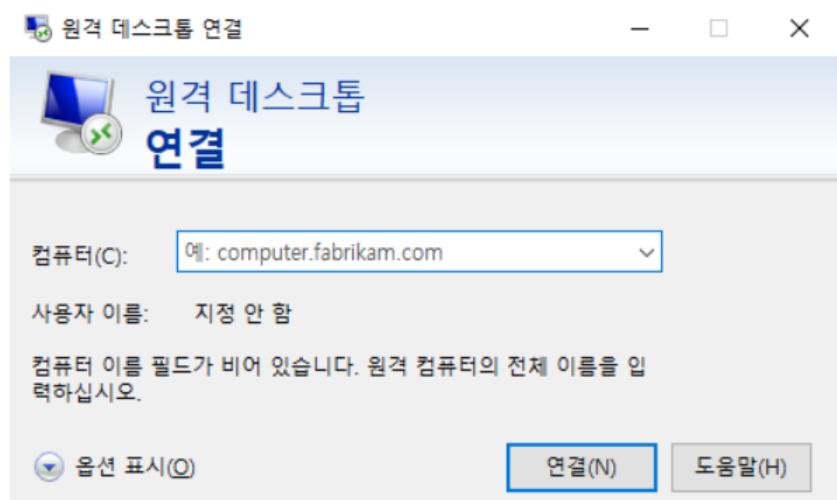
ifconfig 명령어를 통해 ip 주소를 확인한다.

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ifconfig↓  
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500↓  
    inet 192.168.0.64 netmask 255.255.255.0 broadcast 192.168.0.255↓  
        inet6 fe80::5f18:4a1d:291b:5b0 prefixlen 64 scopeid 0x20<link>↓  
            ether c8:d9:d2:23:07:a1 txqueuelen 1000 (Ethernet)↓  
                RX packets 345372 bytes 494852129 (494.8 MB)↓  
                RX errors 0 dropped 2865 overruns 0 frame 0↓  
                TX packets 57298 bytes 7844227 (7.8 MB)↓  
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0↓  
                device interrupt 16 memory 0xe5000000-e5020000 ↓  
↓  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536↓  
    inet 127.0.0.1 netmask 255.0.0.0↓  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>↓  
        loop txqueuelen 1000 (Local Loopback)↓  
            RX packets 3067 bytes 310426 (310.4 KB)↓  
            RX errors 0 dropped 0 overruns 0 frame 0↓  
            TX packets 3067 bytes 310426 (310.4 KB)↓  
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0↓  
↓
```

+) 만약 ifcomfig 명령어를 찾을 수 없다고 뜨면  
**sudo apt install net-tools**로 net-tools를 설치한다.

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ifconfig↓  
↓  
명령어 'ifconfig' 을(를) 찾을 수 없습니다. 그러나 다음을 통해 설치할 수 있습니다:↓  
↓  
sudo apt install net-tools↓  
↓  
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo apt install net-tools↓  
패키지 목록을 읽는 중입니다... 완료↓  
의존성 트리를 만드는 중입니다 ↓
```

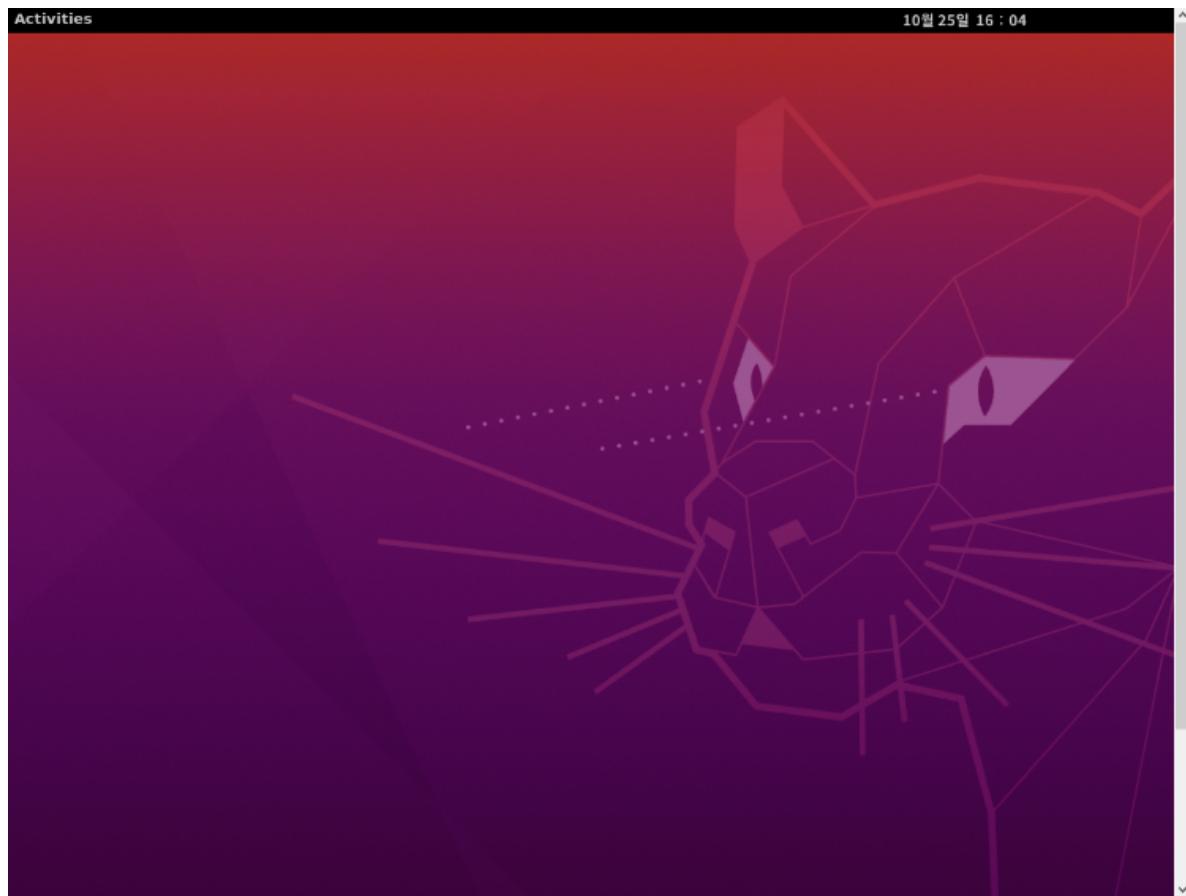
- IP주소 + 포트번호 입력  
위에서 포트 변경 안했다면 IP주소만 입력



내 노트북에서 원격 데스크톱을 실행 후  
위에서 얻은 ip 주소를 입력한다.



설정한 우분투 이름과 비밀번호를 입력하면



접속 완료!

---

원격 접속 시 화면이 검정색일 때는

```
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo vim /etc/xrdp/startwm.sh
```

**sudo vim /etc/xrdp/startwm.sh**

해당 명령어를 입력 후

```

GNU nano 2.9.3                               /etc/xrdp/startwm.sh

    test -z "${LC_ALL+x}" || export LC_ALL
    test -z "${LC_COLLATE+x}" || export LC_COLLATE
    test -z "${LC_CTYPE+x}" || export LC_CTYPE
    test -z "${LC_IDENTIFICATION+x}" || export LC_IDENTIFICATION
    test -z "${LC_MEASUREMENT+x}" || export LC_MEASUREMENT
    test -z "${LC_MESSAGES+x}" || export LC_MESSAGES
    test -z "${LC_MONETARY+x}" || export LC_MONETARY
    test -z "${LC_NAME+x}" || export LC_NAME
    test -z "${LC_NUMERIC+x}" || export LC_NUMERIC
    test -z "${LC_PAPER+x}" || export LC_PAPER
    test -z "${LC_TELEPHONE+x}" || export LC_TELEPHONE
    test -z "${LC_TIME+x}" || export LC_TIME
    test -z "${LOCPATH+x}" || export LOCPATH
fi

if test -r /etc/profile; then
    . /etc/profile
fi

unset DBUS_SESSION_BUS_ADDRESS
unset XDG_RUNTIME_DIR

test -x /etc/X11/Xsession && exec /etc/X11/Xsession
exec /bin/sh /etc/X11/Xsession

```

사진과 같은 부분에 아래 코드를 삽입 -> esc -> :wq!

```

unset DBUS_SESSION_BUS_ADDRESS
unset XDG_RUNTIME_DIR

```

```

readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ sudo vim /etc/xrdp/star
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ service xrdp restart
readvice@readvice-HP-EliteDesk-800-G4-TWR:~$ ^C

```

그리고 재실행 후 원격접속을 다시 시도한다.