

기상 데이터를 활용한 딥러닝 기반
미세먼지 농도 **예측** & 장소 **추천** 서비스

미 세 , 美 世

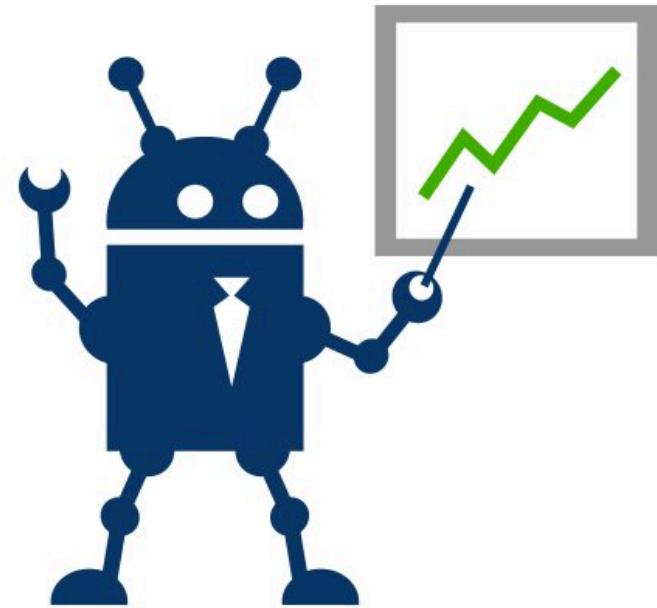
K P C

금융 빅데이터 분석가 양성과정
김민지, 김정태, 박의수, 신민경

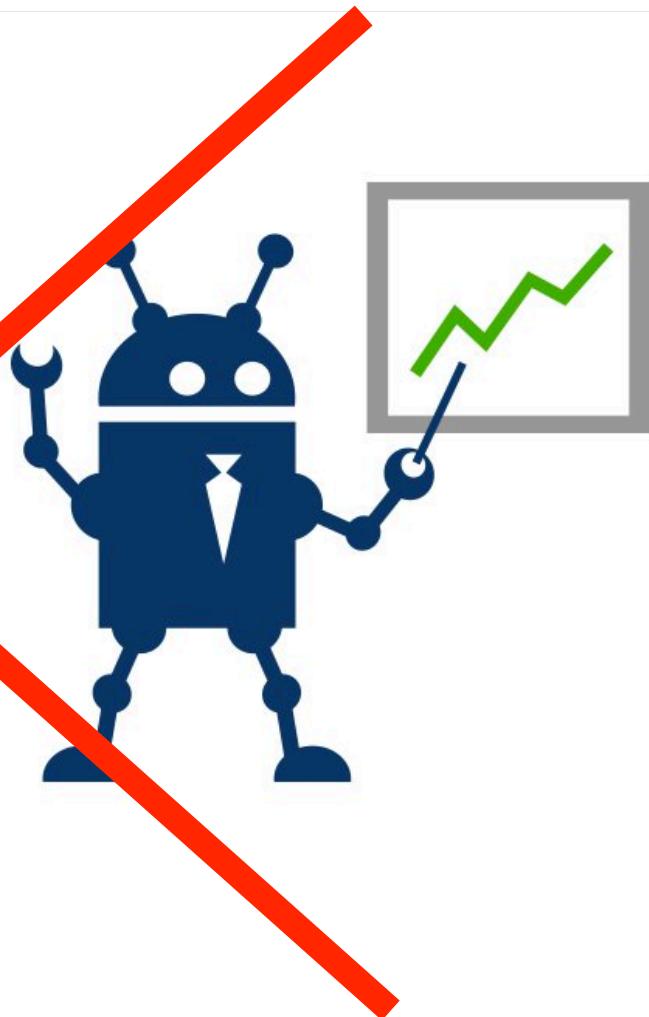
목 차

- 주제 선정 배경
- 프로젝트 개요
- 딥러닝 모델
- 개발 과정
 1. 데이터 수집, 전처리
 2. 분석 모델 구축, 테스트, 파라미터 튜닝
 3. 데이터베이스 설계 및 연동
 4. 시각화 및 웹 서비스 구현
- 한계점 및 보완점
- 기대효과 및 활용 방안

주제 선정 배경 (1)



주제 선정 배경 (1)



주제 선정 배경 (2)



“금융?”

주제 선정 배경 (2)



“일상!”

주제 선정 배경 (3)



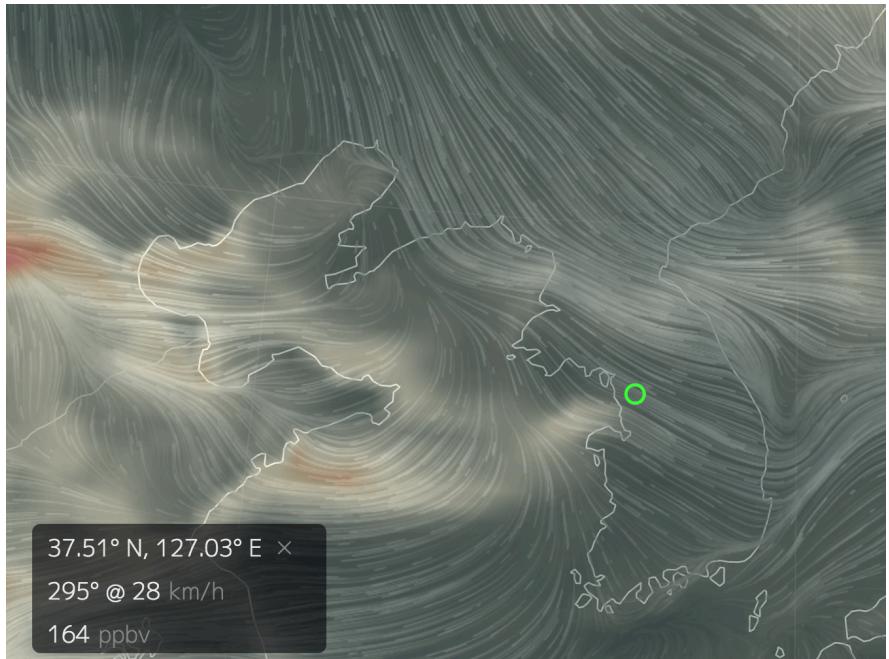
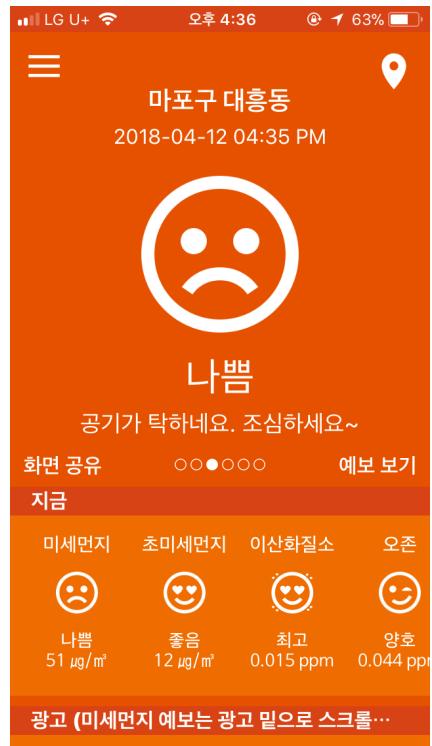
“이상”

주제 선정 배경 (3)



“현실”

주제 선정 배경 (4)



“미세미세”

“미세빅”

“nullschool”

주제 선정 배경 (4)

미세미세 어플의 희망고문 | 자유게시판

원 1:1

요즘 미세미세 단하루도 맞는날 없었어요
일주일내내 12시전후로 좋음이라 스케줄 다 잡았는데 단하루도 맞지않네요
차라리 비슷하기라도 하지
매우나쁨과 좋은은 너무 비현실이라 ㅜㅜ
내일도 계속 황사일까요? ㅜㅜ

카페앱간편설치하기

댓글 1 | 등록순 | 조회수 170 | 좋아요 0

맞아요ㅋㅋ요새 미세미세 예보 완전 뜯이에요ㅠ—ㅠ 옛날엔 좀 맞았는데

서울시 수치 공유

- 동작, 관악 [H](#)
- 성북, 강북, 도봉 [H](#)
- 광진, 성동, 동대문 [H](#)
- 송파, 강동 [H](#)
- 마포, 은평, 서대문 [H](#)
- 구로, 금천, 영등포 [H](#)
- 강서, 양천 [H](#)
- 강남, 서초 [H](#)
- 노원, 중랑 [H](#)
- 종로, 중구, 용산 [H](#)

경기도 수치 공유

- 인천광역시 [H](#)
- 양평, 가평 [H](#)
- 수원(영통, 팔달, 장안, 권

178539 종곡동32

178436 종곡동 43

178258 종곡동 38 [1]

178136 금호동

177767 종곡동 시작 41

177475 어린이대공원 6~11 [1]

177442 종곡동 12

177437 금호동 [1]

177331 종곡동 42-50 [1]

177202 광장동 초미세 50

프로젝트 개요

프로젝트명

> 기상 데이터를 활용한 딥러닝 기반 미세먼지 농도 **예측** & 장소 **추천** 서비스

프로젝트 기간

> 2018.03.15~2018.04.11

프로젝트 목적

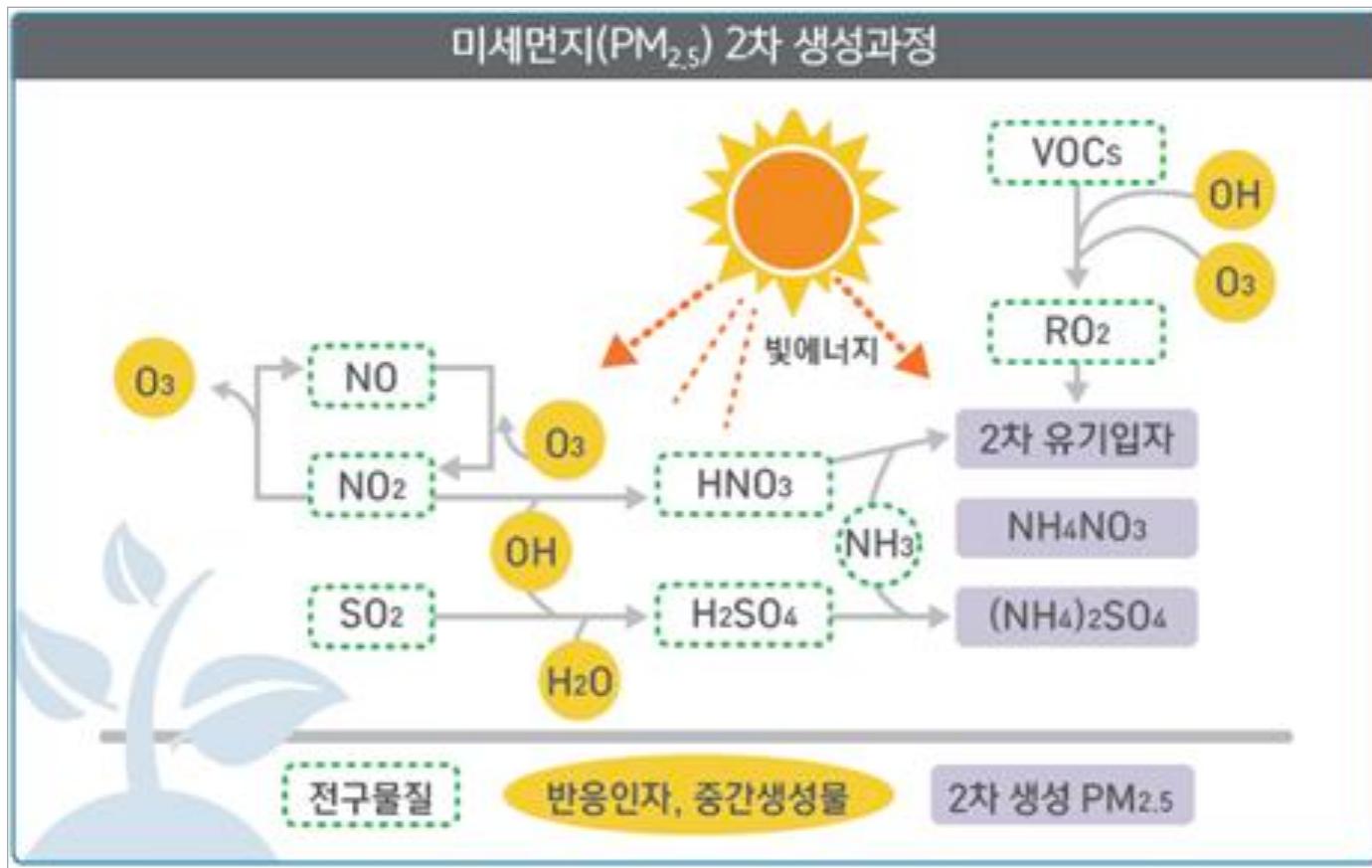
> 예측 성능 **상향**

> 예보 기반 추천 서비스 구현

활용 데이터 - 대기 데이터

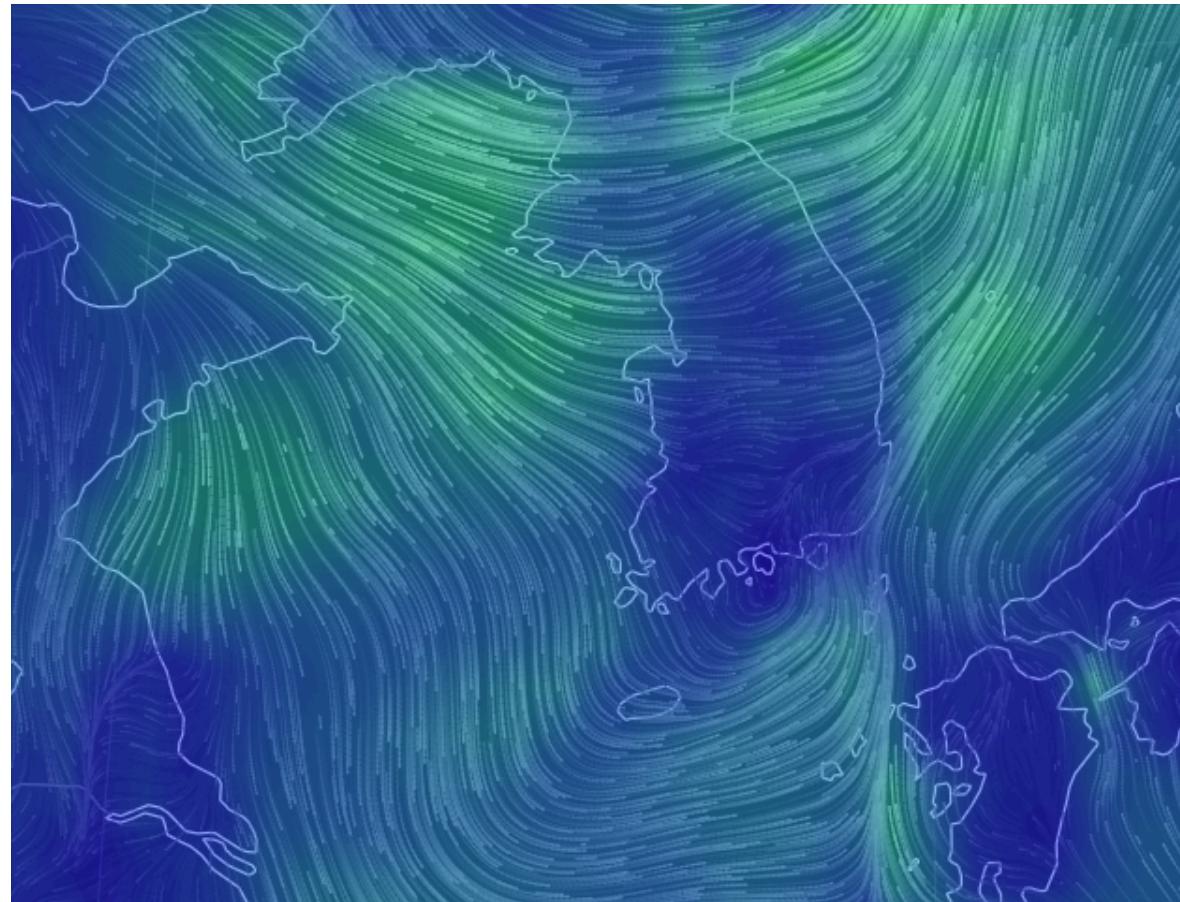


활용 데이터 - 대기 데이터



“미세먼지, 초미세먼지 / 이산화질소(NO_2), 오존(O_3), 일산화탄소(CO), 아황산가스(SO_2)”

활용 데이터 - 기상 데이터



활용 데이터 - 기상 데이터



'남서풍'에 '대기정체' 겹쳐 … '최악의 미세먼지' 2018.03.26.

이번 미세먼지는 남서풍을 타고 중국에서 유입된 것으로 '대기정체' 현상까지 발생하면서 최악의 상황에 달했다. 대기정체로 국내... 미세먼지의...



CBC뉴스 | NEWS&ISSUE

[오늘 미세먼지] 강수 세정효과…전국 '좋음'



강수량(mm), 풍향(°), 풍속(m/s), 기온(°C)

분석 시도

RamdomForest 모델 학습

```
clf = RandomForestClassifier(max_depth=5, random_state=0)
clf.fit(train_X, train_Y)

RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                      max_depth=5, max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                      oob_score=False, random_state=0, verbose=0, warm_start=False)

clf.score(test_X, test_Y)

0.0997229916897507
```

분석 시도

RandomForest 모델 학습

```
clf = RandomForestClassifier(max_depth=5, random_state=0)
clf.fit(train_X, train_Y)

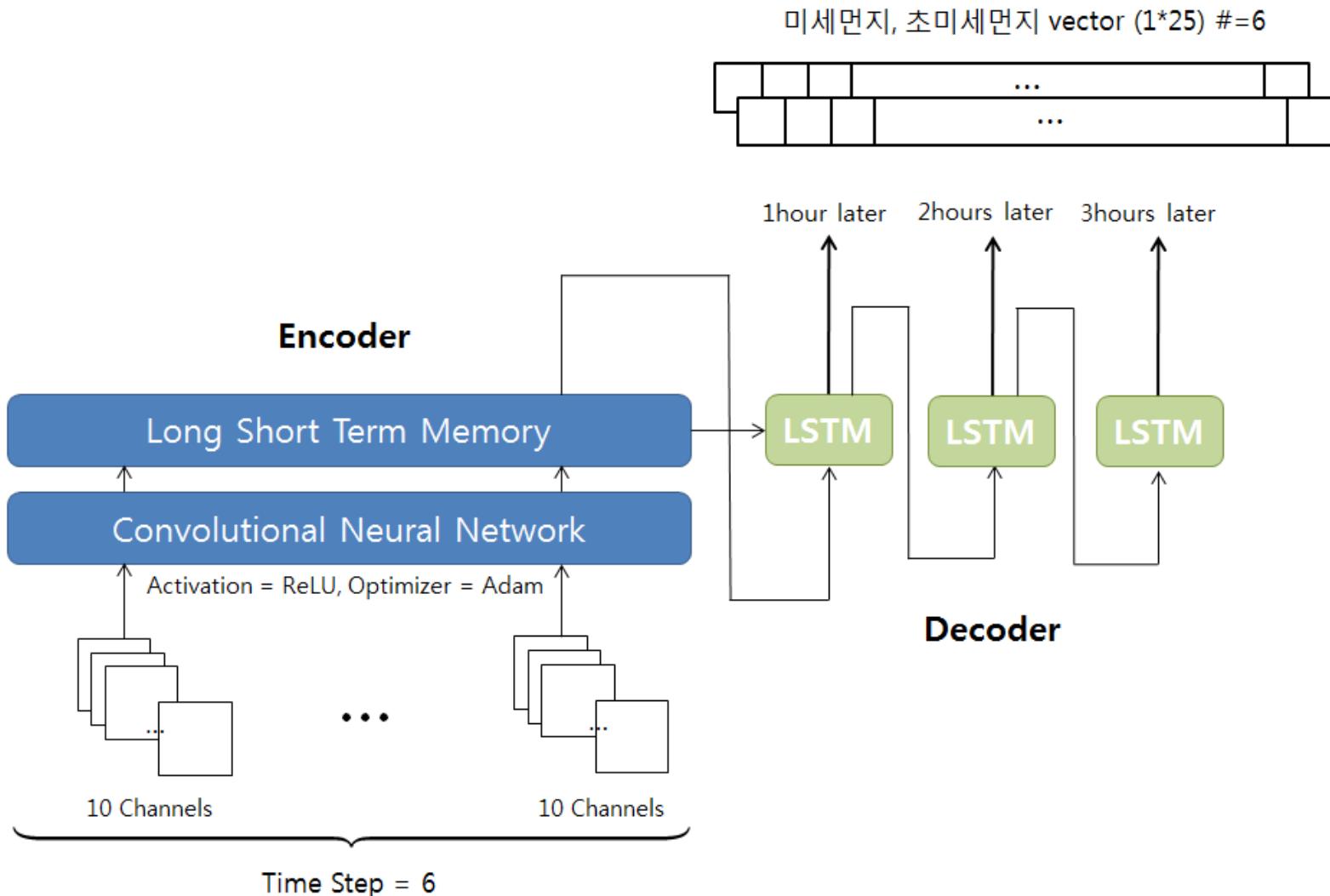
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                      max_depth=5, max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                      oob_score=False, random_state=0, verbose=0, warm_start=False)
```

```
clf.score(test_X, test_Y)
```

```
0.0997229916897507
```

실패

딥러닝 모델 설계



데이터 Reshaping

近墨者黑

(근묵자흑)

“묵을 가까이 하면 점어진다.”

데이터 Reshaping

近墨者黑

(근묵자흑)

‘옆 동네 공기가 안좋으면 **우리 동네** 공기도 안좋아진다!’

데이터 Reshaping

近墨者黑

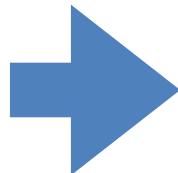
(근묵자흑)

‘옆 동네 공기가 안좋으면 **우리 동네** 공기도 안좋아진다!’

ex) 옆 동네 : 중국 > 우리 동네 : 한국

옆 동네 : 중구 > 우리 동네 : 종로구

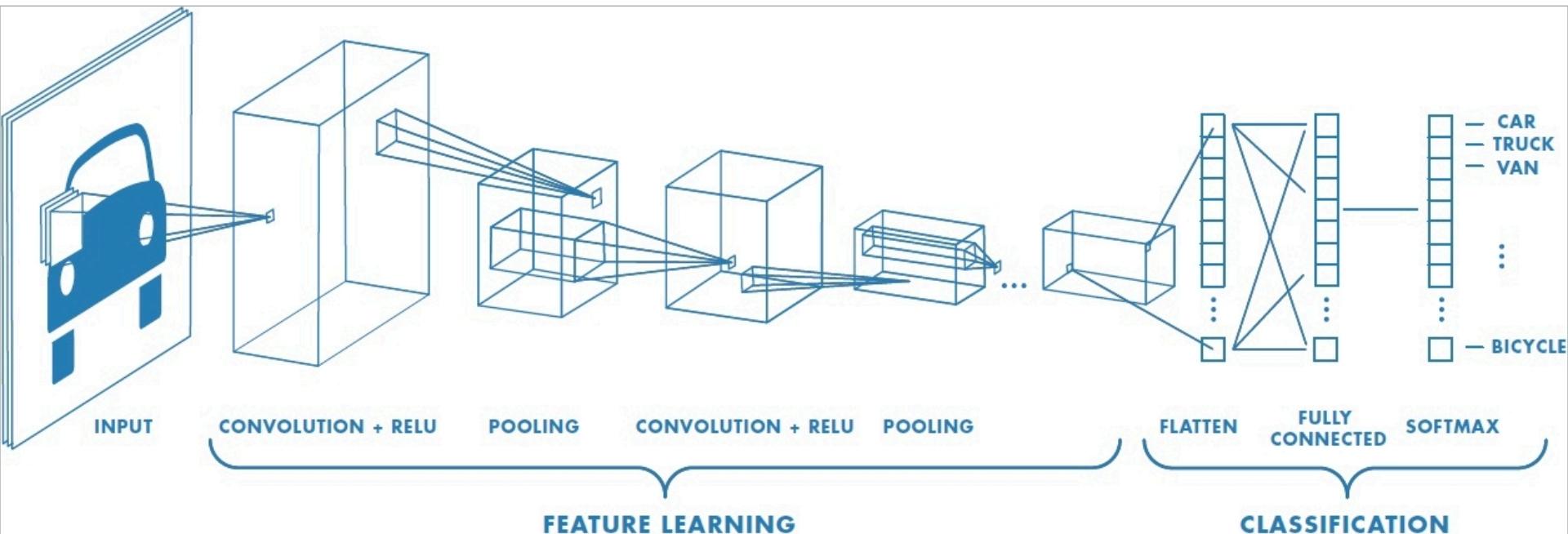
데이터 Reshaping



0	은평구	0	강북구	도봉구	0
0	서대문구	종로구	성북구	노원구	0
강서구	마포구	중구	동대문구	종량구	0
양천구	영등포구	용산구	성동구	광진구	강동구
구로구	동작구	서초구	강남구	송파구	0
금천구	관악구	0	0	0	0

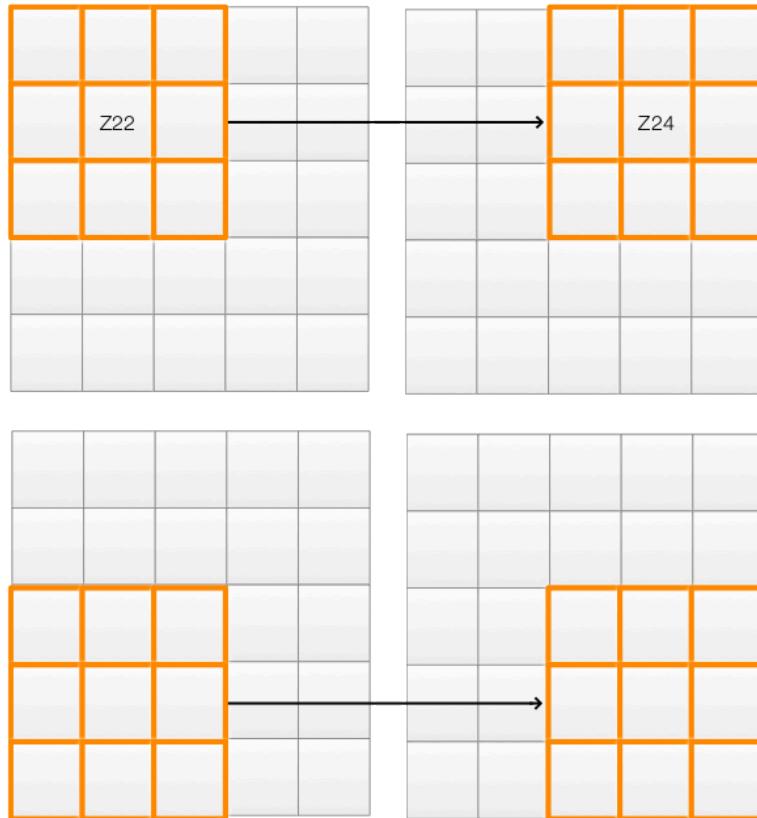
“근접 지역 정보 반영”

딥러닝 모델 - Convolutional Neural Network



“Image Recognition에 특화된 CNN”

딥러닝 모델 - Convolutional Neural Network



0	은평구	0	강북구	도봉구	0
0	서대문구	종로구	성북구	노원구	0
강서구	마포구	중구	동대문구	중랑구	0
양천구	영등포구	용산구	성동구	광진구	강동구
구로구	동작구	서초구	강남구	송파구	0
금천구	관악구	0	0	0	0

“filter를 통해 근접한 지역정보를 함께 학습하도록!”

딥러닝 모델 - Convolutional Neural Network

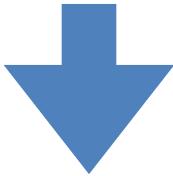
0	은평구	0	강북구	도봉구	0
0	서대문구	종로구	성북구	노원구	0
강서구	마포구	중구	동대문구	중랑구	0
양천구	영등포구	용산구	성동구	광진구	강동구
구로구	동작구	서초구	강남구	송파구	0
금천구	관악구	0	0	0	0

0	은평구	0	강북구	도봉구	0
0	서대문구	종로구	성북구	노원구	0
강서구	마포구	중구	동대문구	중랑구	0
양천구	영등포구	용산구	성동구	광진구	강동구
구로구	동작구	서초구	강남구	송파구	0
금천구	관악구	0	0	0	0

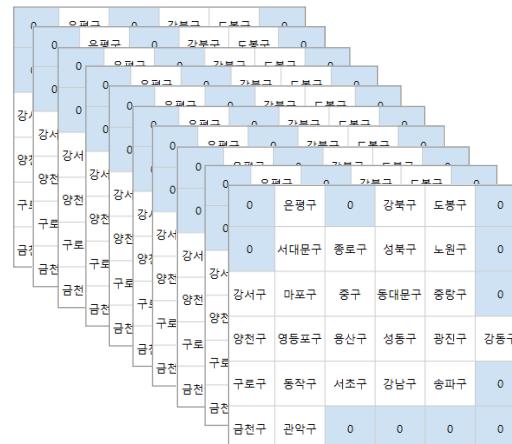
0	은평구	0	강북구	도봉구	0
0	서대문구	종로구	성북구	노원구	0
강서구	마포구	중구	동대문구	중랑구	0
양천구	영등포구	용산구	성동구	광진구	강동구
구로구	동작구	서초구	강남구	송파구	0
금천구	관악구	0	0	0	0

‘기온’

‘풍향’

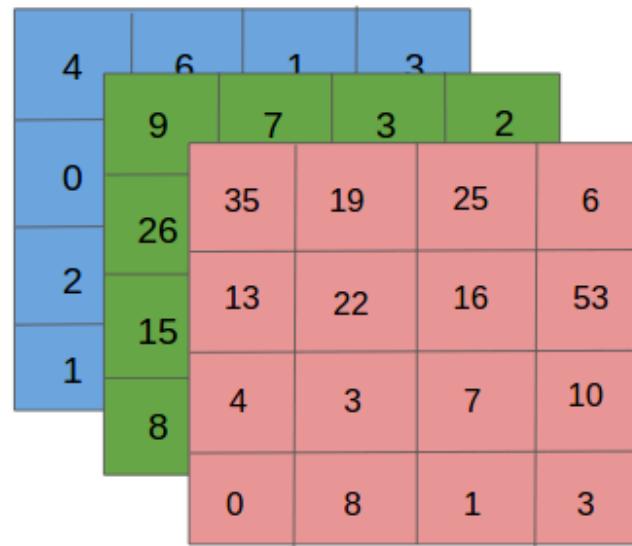


‘오존’



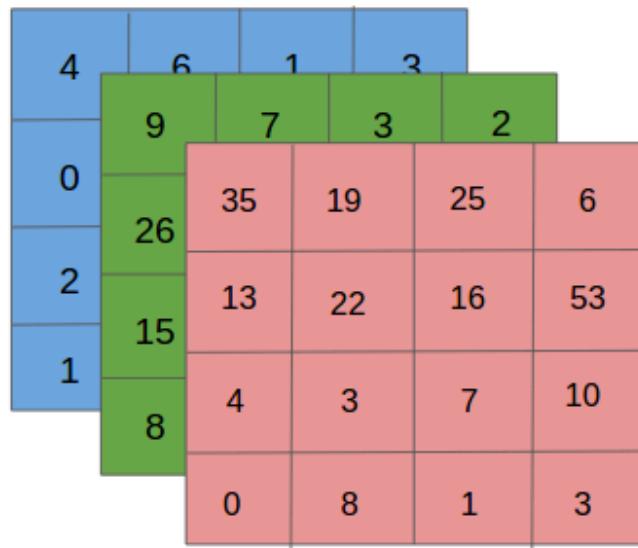
“10 Channels”

딥러닝 모델 - Convolutional Neural Network



“R, G, B Channel”

딥러닝 모델 - Convolutional Neural Network

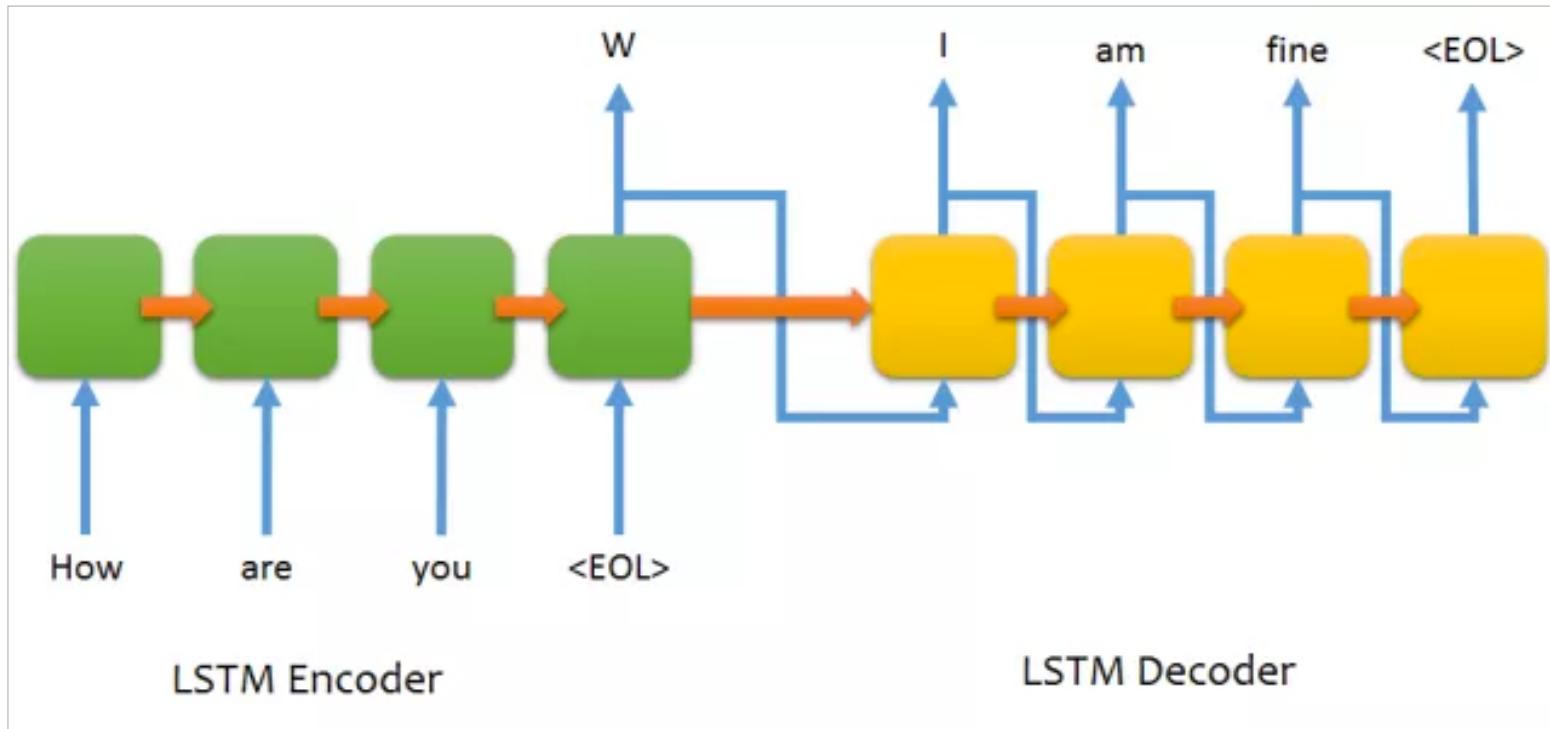


-

0	은평구	0	강북구	도봉구	0			
0	은평구	0	강북구	도봉구	0			
0	0	은평구	0	강북구	도봉구			
강서구	0	0	은평구	0	강북구	도봉구	0	
양천구	0	0	0	서대문구	종로구	성북구	노원구	0
구로구	0	0	0	0	0	0	0	0
금천구	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

“10 Channels”

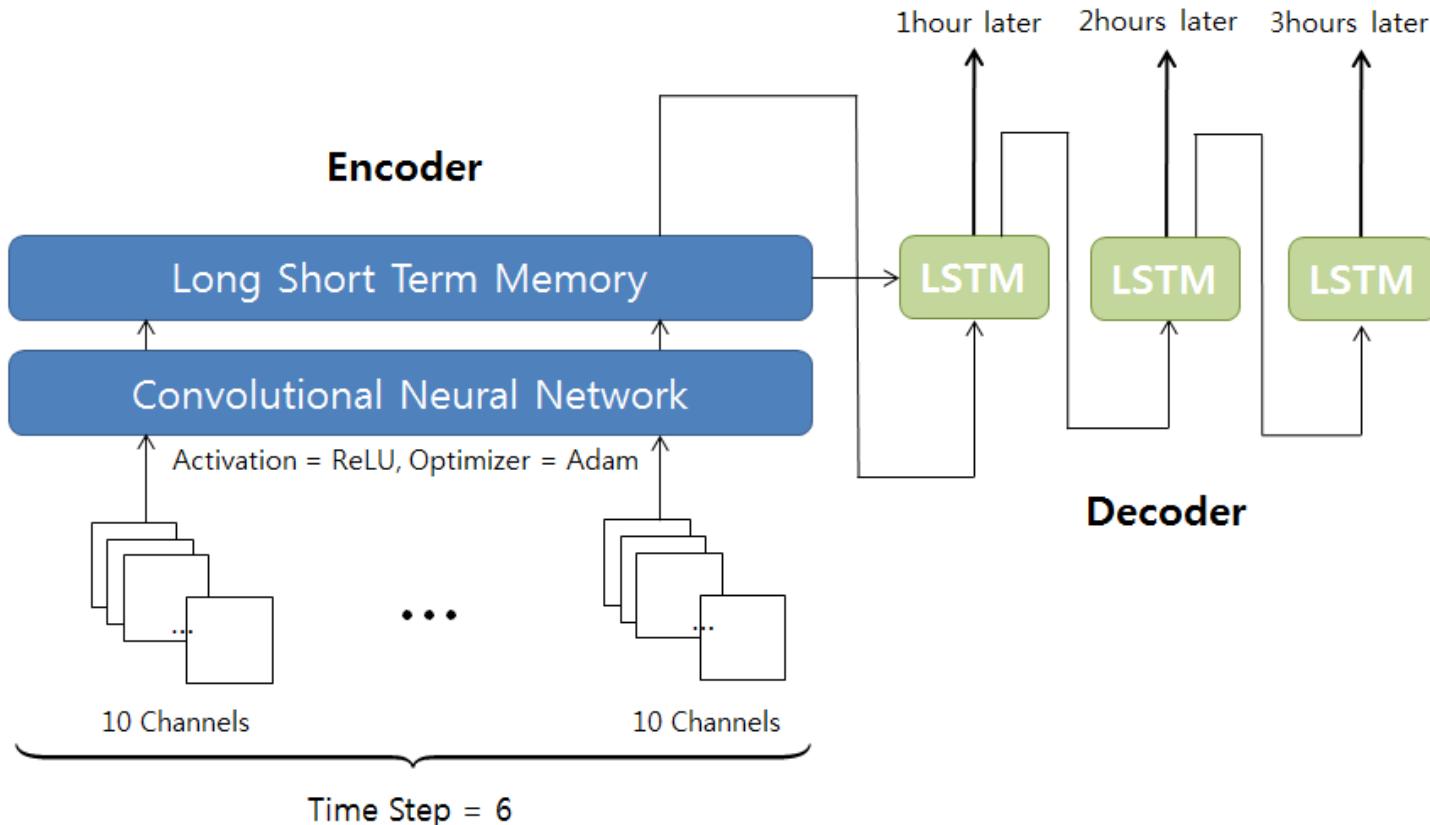
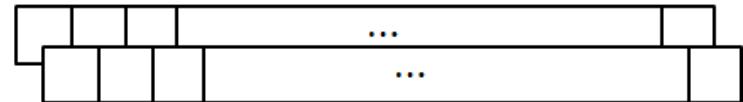
딥러닝 모델 - Long Short Term Memory



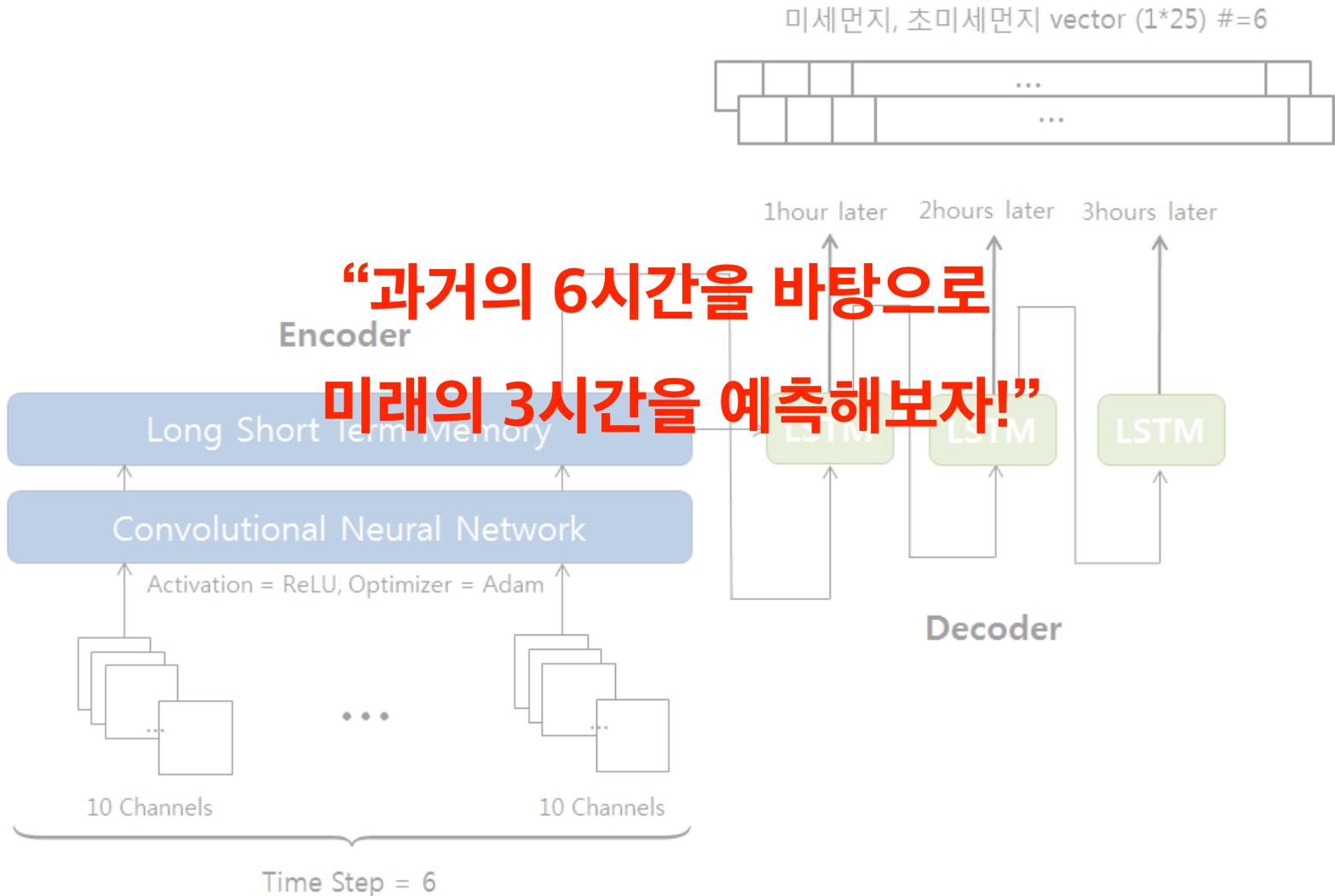
“**미래**의 대기는 **과거**와 **현재**의 대기 상태에 영향을 받는다!”

딥러닝 모델 - CNN + RNN

미세먼지, 초미세먼지 vector (1×25) #=6



딥러닝 모델 - CNN + RNN



개발 과정

- 데이터 수집
- 데이터 전처리
- 예측 모델 학습, 테스트, 파라미터 튜닝
- 데이터베이스 설계 및 연동
- 데이터 시각화 및 작업 스케줄링
- WEB 구현

개발 과정 - 데이터 수집 (1)

	강남구	강동구	강북구	강서구	관악구	광진구	구로구	금천구	노원구	도봉구	동대문구	동작구	마포구	서대문구
2015-01-01 01:00:00	314.4	347.1	308.2	268.9	348.7	323.9	311.9	307.7	266.6	299.2	336.8	281.4	304.8	314
2015-01-01 02:00:00	301.1	291.2	287.1	261	0.5	309.7	321.7	319	270.1	323.8	354.7	311.3	312.3	307.7
2015-01-01 03:00:00	352.9	307.9	304.2	277.3	9.4	324.8	353.3	328.6	265.6	331.4	0.2	303.3	306.3	313.4
2015-01-01 04:00:00	319.2	6.9	324.7	279.1	10.7	305.9	323.7	306.5	281.2	327.4	2.9	327.7	308.6	316.6
2015-01-01 05:00:00	300.1	114.6	290.6	243.5	12.4	309.6	319.9	334.7	278.1	321.8	347.9	299.7	311.9	308.8
2015-01-01 06:00:00	287.4	284.3	311.4	226.3	3.6	313.4	321.1	340.2	276.5	317.3	352.8	281	307.1	325.3
2015-01-01 07:00:00	16.1	348.3	302.3	243	346.9	311.2	330.7	329.7	284.2	104.7	348.6	299.8	305.8	321.2



기상자료개방포털 <https://data.kma.go.kr>
종관기상관측 데이터 2015년-2017년 수집

개발 과정 - 데이터 수집 (2)



```
for i, gu in enumerate(get_gu_list(), 1):
    if not os.path.isdir(os.path.join(dosi, gu)):
        os.mkdir(os.path.join(dosi, gu))
    click_gu(i)
    df = pd.DataFrame(columns=item_list)
    for date in generate_date(start, end):
        click_date(date)
        for item in item_list:
            item_select.send_keys(item)
            search.click()
            time.sleep(1)
            a = browser.find_elements_by_css_selector('#tablefix1 > tbody > tr')
            if len(a) < 100:
                continue
            df[item] = [browser.find_elements_by_css_selector('#tablefix1 > tbody > tr:nth-child({})'.format(i))]
```

“파이썬 Selenium 사용”

```
from urllib import request
from selenium import webdriver
from selenium.webdriver.support.ui import Select
```

날짜 (월-일:시)	PM ₁₀ ($\mu\text{g}/\text{m}^3$)		PM _{2.5} ($\mu\text{g}/\text{m}^3$)		오존 (ppm)		이산화질소 (ppm)		일산화탄소 (ppm)		아황산가스 (ppm)	
	1시간		1시간		1시간		1시간		1시간		1시간	
04-12:18	●	32	●	10	●	0.043	●	0.021	●	0.3	●	0.004
04-12:17	●	31	●	8	●	0.047	●	0.018	●	0.3	●	0.003
04-12:16	●	32	●	9	●	0.049	●	0.015	●	0.3	●	0.004
04-12:15	●	38	●	13	●	0.049	●	0.015	●	0.3	●	0.004

에어코리아 www.airkorea.or.kr
학습용 대기 데이터 웹 크롤링

개발 과정 - 데이터 수집 (3)



```
def get_weather_data(date, time, pageNo=1, numRows=100):  
  
    url = 'http://newsky2.kma.go.kr/service/SecndSrtpdFrcstInfoSe:  
key1 = 'EmaFGtVYwlVSmg%2Fl2aYzL3qkcZgsTt3QOTFQ%2Fwff34R3HGxHk'  
key2 = '7%2F0t7PviAWG7n7U8BDQbUh19DxoymlR4Y10rAscaNcBLxSYfVzcx:  
city_coord = dict(OrderedDict([('은평구',[59,127]), ('강북구', [60,127]),  
('서대문구', [59,126]), ('종로구', [60,127]), ('성북구', [58,127]), ('강서구', [58,127]), ('마포구', [59,127]), ('중구', [58,126]), ('양천구', [58,126]), ('영등포구', [59,126]), ('용산구', [58,125]), ('구로구', [58,125]), ('동작구', [59,125]), ('서초구', [58,124]), ('금천구', [58,124]), ('관악구', [59,124]))])  
city_coord = OrderedDict(sorted(city_coord.items()))  
temperature_list = []  
rain_list = []  
direction_list = []  
velocity_list = []  
  
for city in city_coord.keys():  
    metadata = {  
        "base_date":date,  
        "base_time":time,  
        "ftype":'ODAM',  
        "pageNo": str(pageNo),  
        "numOfRows": str(numOfRows),  
        "nx":(city_coord[city][0]),  
        "ny":(city_coord[city][1]),  
        "_type": "json"  
    }  
    response = requests.get(url+key2, params=metadata).json()  
    response = requests.get(url+key1, params=metadata).json()
```

“실시간 기상, 대기 데이터(json) 조회”

```
def get_air_data(pageNo=1, numRows=100):  
  
    url = 'http://openapi.airkorea.or.kr/openapi/services/re:  
key = 'EmaFGtVYwlVSmg%2Fl2aYzL3qkcZgsTt3QOTFQ%2Fwff34R3H0:  
city_list = []  
coValue_list = []  
no2Value_list = []  
o3Value_list = []  
pm10Value_list = []  
pm25Value_list = []  
so2Value_list = []  
  
metadata = {  
    "sidoName": "서울",  
    "searchCondition": "HOUR",  
    "pageNo": str(pageNo),  
    "numOfRows": str(numOfRows),  
    "_returnType": "json"  
}  
  
response = requests.get(url+key, params=metadata).json()  
target = response['list']
```

공공데이터포털 www.data.go.kr API 활용

개발 과정 - 데이터 수집 (4)

어디로 갈까

축제·행사

구석구석 여행지찾기

● 총 게시물 : 222건, 페이지 : 1/23

NEW



책의 해, 세계 책과 저작권의 날 2018

2018년 '세계 책의 날'
널리 알리고자 하는 행

서울 종로구

10 04.12

"파이썬 BeautifulSoup 사용"

```
def get_festival(url):
    festival = OrderedDict()
    with request.urlopen(url) as fp:
        source = fp.read()
    soup = BeautifulSoup(source, 'html.parser')

    festival['행사명'] = soup.find(id='content_title').string.strip()
    for li in soup.find('figcaption').find_all('li'):
        key = li.find('b').string.strip()
        value = '\n'.join([c.string.strip() for c in li.find('span').children])
        festival[key] = value

    for tag in soup.find('ul', class_='ptList').find_all('li'):
        em = tag.find('em')
        if not em:
            break
        key = em.string.strip()
        value = '\n'.join([c.string.strip() for c in tag.children if c.string])
        festival[key] = value

    return festival
```

개발 과정 - 데이터 수집 (5)



전국주요상권현황

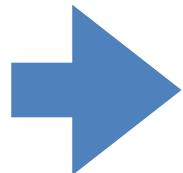
소상공인 창업에 필요한 전국1200개 주요상권현황을 목록화한 자료

상권번호	상권명	시도코드	시도명	시군구코드	시군구명	상권좌표수	좌표(경위도)_wgs84
9145	홍대입구역_2		11 서울특별시	1144	마포구	19	126.922648349739,37
9146	홍대입구역_3		11 서울특별시	1144	마포구	21	126.925311211591,37
9147	홍익대학교		11 서울특별시	1144	마포구	46	126.926047368483,37
9148	서대문역_1		11 서울특별시	1111	종로구	12	126.967096112898,37
9149	서대문역_2		11 서울특별시	1141	서대문구	29	126.965843273243,37
9150	신촌역_2		11 서울특별시	1141	서대문구	38	126.938956730473,37
9151	이대역		11 서울특별시	1141	서대문구	21	126.945669763923,37
9152	숙대입구역		11 서울특별시	1117	용산구	12	126.97354506797,37
9153	신용산역		11 서울특별시	1117	용산구	11	126.971537705886,37
9154	용산,전자상가_1		11 서울특별시	1117	용산구	31	126.960157235116,37
9155	용산,전자상가_2		11 서울특별시	1117	용산구	16	126.967095553325,37

공공데이터포털 www.data.go.kr 상권 데이터 약 7만여개 활용

개발 과정 - 데이터 전처리

13	18	28	16	25
9	21	29	18	19
10	17	16	13	25
7	16		14	17
12	16		14	13
6	14		16	19
9	18		22	19
7	23		11	15
9	12		8	11
12	11	23	5	11
8	13	19	10	11
8	15	11	9	14

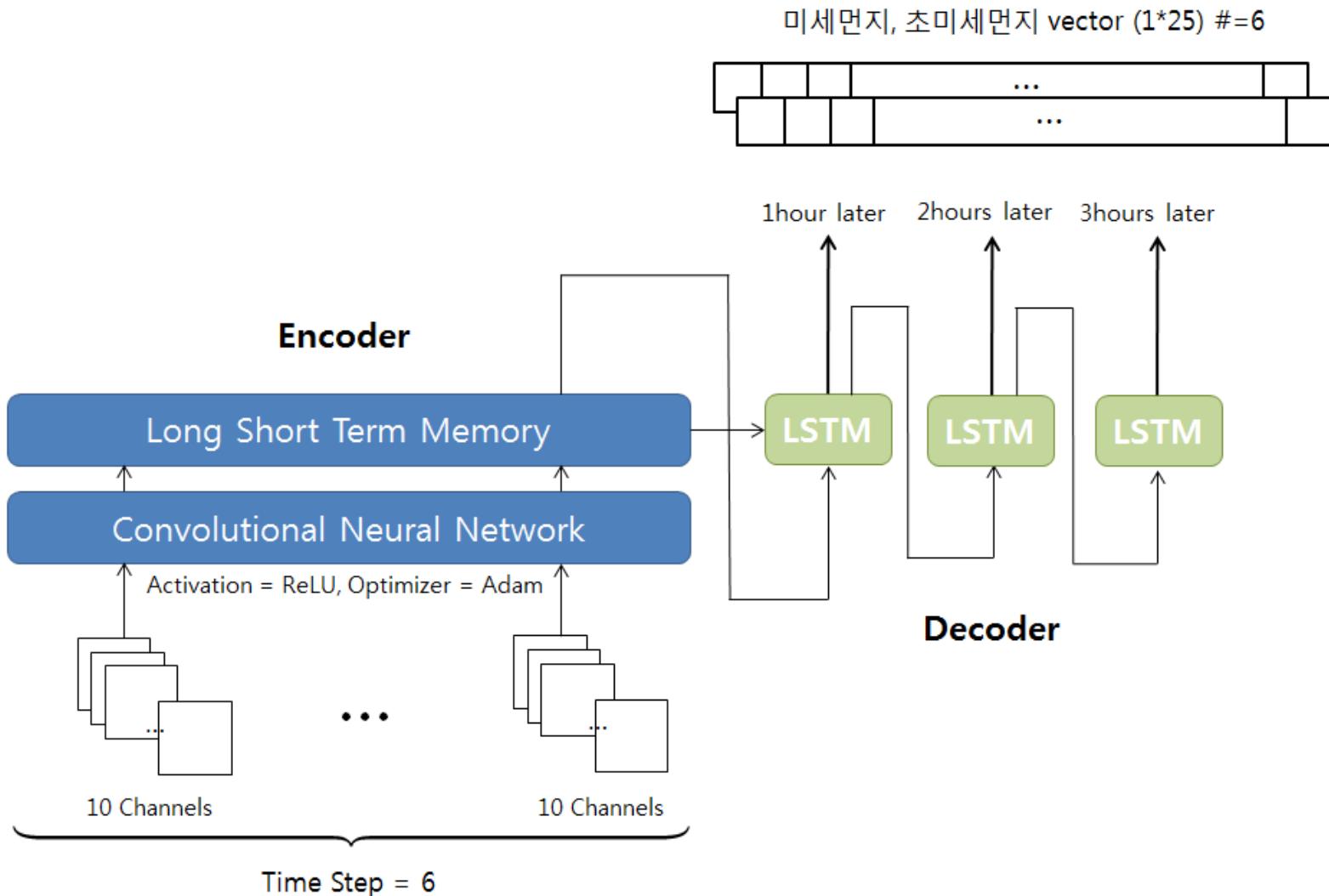


13	18	28	16	25
9	21	29	18	19
10	17	16	13	25
7	16	17	14	17
12	16	18	14	13
6	14	19	16	19
9	18	20	22	19
7	23	21	11	15
9	12	22	8	11
12	11	23	5	11
8	13	19	10	11
8	15	11	9	14

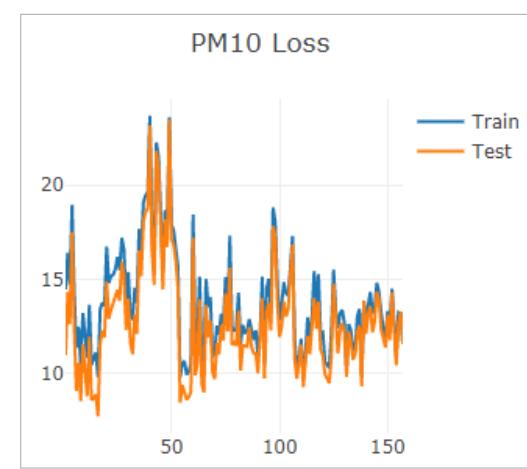
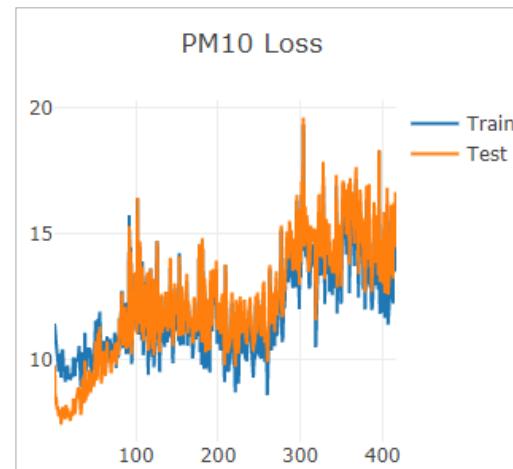
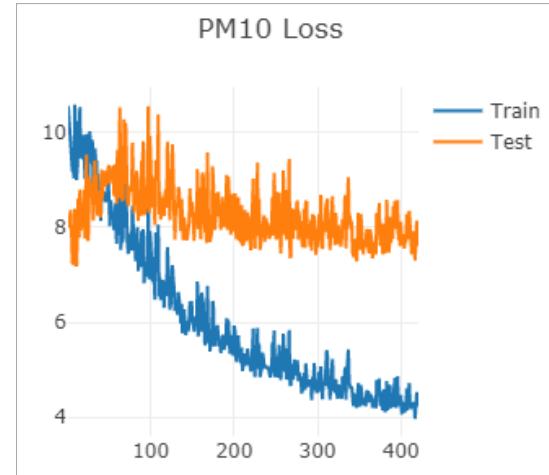
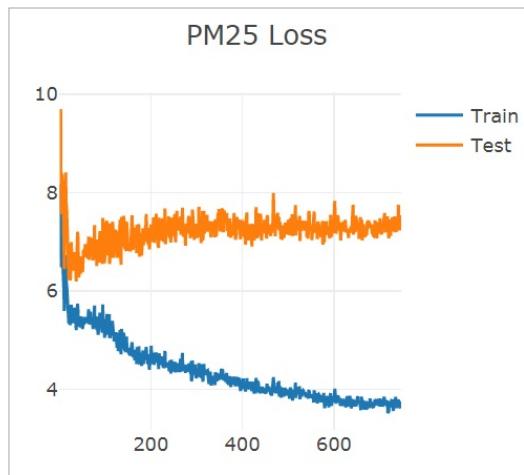
6시간 이내 연속 결측치 : 보간법 활용

6시간 초과 연속 결측치 : 해당 시간대 평균값 대체

개발 과정 - 분석 모델 구축



개발 과정 - 분석 모델 테스트



“모델 학습 (Minimizing Loss)”

개발 과정 - 분석 모델 튜닝



“끝없는 삽질과 기도....”

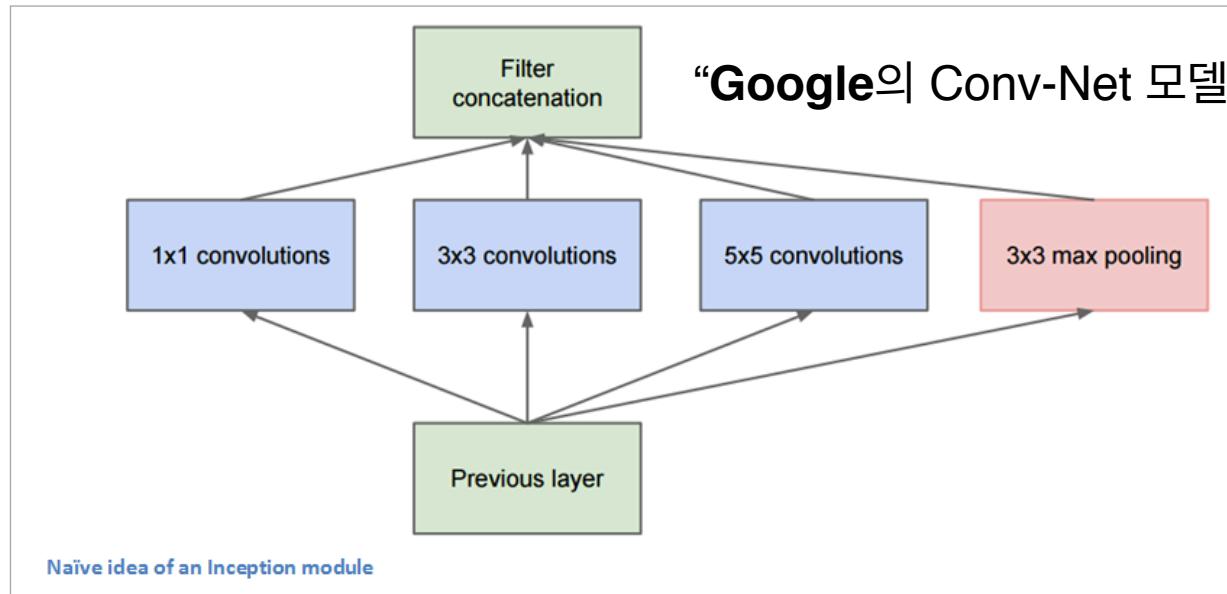
개발 과정 - 분석 모델 완성



```
self.conv1 = nn.Sequential(  
    nn.Conv2d(in_channels=10, out_channels=32, kernel_size=1),  
    nn.AvgPool2d(kernel_size=2, stride=1),  
    nn.BatchNorm2d(32),  
    nn.Dropout2d(dropout_p),  
    nn.ReLU(),  
)
```

```
self.encoder = nn.LSTM(input_size=256, hidden_size=512,  
self.decoder = nn.LSTM(input_size=512, hidden_size=512,  
self.decoder_fc = nn.Sequential(  
    BatchNorm1d(512),  
    Linear(512, 256),  
    BatchNorm1d(256),  
    Dropout(dropout_p),  
    Tanh(),  
    Dropout(dropout_p),  
    Linear(256, 25)
```

“딥러닝 프레임월 Pytorch로 모델 구현”



개발 과정 - 데이터베이스 설계 & 프로세스



“**schtasks** 활용하여 작업 스케줄링
1시간마다 실시간 데이터, 예측 데이터 DB 업데이트”

개발 과정 - DB 연동

“파이썬 cx_Oracle 활용”

```
conn = cx_Oracle.connect('SCOTT/TIGER@localhost:1521/XE')
db = conn.cursor()
for i in range(6):
    rows = [get_air_data()[0]]
    rows.extend(get_air_data()[i+1])
    if i==0:
        db.execute("insert into realtimeCo (집계시각, 강남구, 강동구,
    elif i==1:
        db.execute("insert into realtimeNo2 (집계시각, 강남구, 강동구,
    else:
        db.execute("insert into realtimeO3 (집계시각, 강남구, 강동구,
```

1) 공공 API 실시간 데이터 조회 및 DB insert

```
d = realtime_data()
pm10 = CRNN(1.0)
pm10.load_state_dict(torch.load('w/PM10.pth'))

pm25 = CRNN(1.0)
pm25.load_state_dict(torch.load('w/PM25.pth'))
a = getData(d)

m_array = pm10(Variable(a)).data.numpy()[0]
cm_array = pm25(Variable(a)).data.numpy()[0]
```

```
query = 'insert into predictedData values(:1, :2, :3, :4, :5, :6, :7, :8)'
db.execute(query, [oneHourAgoFromNow(), loc_list[i], int(mhour1[i]), int(cmhour1[i]),
db.execute('commit')
```

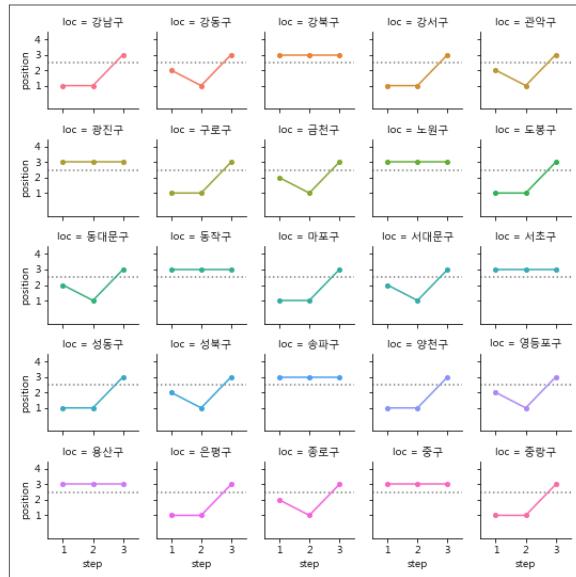
3) 예측 결과 데이터 DB insert

```
cmhour1 = cm_array[0]
cmhour2 = cm_array[1]
cmhour3 = cm_array[2]
```

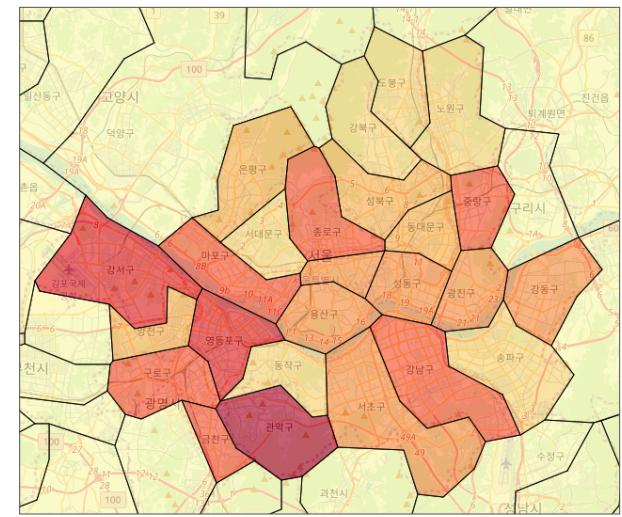
개발 과정 - 데이터 시각화



“Matplotlib”



“Seaborn”



“Folium”

```
schtasks /create /tn plot /tr C:\Users\user\Desktop\makePlots.py /sc hourly /mo 1
```

1시간마다 img파일 생성 및 local 경로에 저장

개발 과정 - WEB (시연)

10.1.43.90/Soup/index.jsp

한계점 및 보완점

한계점

- 국외 요인, 교통, 공장 관련 데이터 등 주요 **데이터** 부재
- 시간, 하드웨어 자원 부족
- “*Does the flap of a butterfly's wings in Brazil set off a tornado in Texas?*”

보완점

- 서울 > **전국**으로 예측 확대
- 랜덤 추천 > 사용자 성향에 맞춘 **개인화** 추천
- 모델 **최적화** 및 예측 시간 **장기화**

기대 효과 및 활용 방안

기대 효과

- 수치 예보 모델과 병행하여 사용 가능한 **저비용** 예측 모델 개발
- 예측 성능 개선을 통한 황사, 고농도 초미세먼지 등의 상황에 **신속한 대응** 가능
- 공기 상태에 알맞은 활동(실내, 실외)로 기관지 질환 **예방**

활용 방안

- 나이, 주거 지역, 활동 형태에 따라 **보험** 상품 가격 **차별화**
- 축제, 관광지 등과 연계하여 타겟 **마케팅**

Thank you!

Q&A