Predicting Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach

작성일시	@2022년 8월 20일 오후 5:36
최종 편집일시	@2022년 8월 22일 오후 1:25
작성자	
최종 편집자	
원문 링크	https://arxiv.org/abs/1812.11670

1. Introduction (작성자 : 김동연)

몇몇 기관들이 항공기 경로 예측 시스템을 사용 중이다. 하지만 계산 복잡도 때문에 보관된 비행계획이나 과거 경로를 이용하는 결정론적 예측을 사용한다. 수요 예측에서 일정하지 않은, 불안정한 날씨같이 불확실한 정보를 거의 사용하지 않는 결정론적 예측은 불필요한 계획을 만들어낼 수 있다. TBO(Trajectory Based Operations)는 정확한 비행경로 예측, 계획 도구를 요구한다. 비행경로는 4D로 이루어져 있다. 이 논문에서는 4D 경로를 예측하는 방법을 제시할 것이다.

미래 비행경로는 실현된 경로, 보관된 비행계획(2D sequence), 부근의 날씨와 연관된 4D 좌표의 sequence로 구성된다. 따라서 이 과제는 sequence to sequence 학습 문제다. 입력 sequence는 비행계획이고, 출력 sequence는 실제 비행경로다. 이 문제 해결을 위해 encoder-decoder RNN 구조를 사용한다. encoder는 비행계획을 통해 학습하고 decoder는 날씨 정보를 통합한 뒤, 삽입된 비행계획을 4D 경로로 귀납적으로 번역한다. 날씨의 영향을 모델링하기 위해 batch mode와 recursive mode 둘 다 사용 가능한 tree-based 매칭 알고리즘을 적용한다. 그리고 convolutional 층을 decoder 네트워크 파이프라인으로 통합해 날씨의 특징을 추출
한다.

● encoder-decoder RNN을 4D 비행경로 예측에 최초로 사용
● 비행경로에 영향을 주는 많은 요소를 통합하여 생성모델 제시
● 학습, 추측 파이프라인 제공

2. Related work(작성자: 정예준)

Aircraft trajectory prediction approach는 Deterministic, Probabilistic 두개의 카테고리로 나뉨

1. Deterministic : 기체역학적 모델을 이용해 aircraft의 현재 state를 추정하고 Kalman filter
   등을 이용해 미래의 state를 구해냄. 역학 식들을 많이 이용하지만 불확실성을 고려하지 않음(ex. 바람 등 날씨, 파일럿의 control action 등)
2. Probabilistic : 기체역학보다 통계적 모델에 의존함. 과거 경로 데이터셋을 주로 이용
   a. Probabilistic 관련 논문들
      i. A.de Leege "A machine learning approach to trajectory prediction" : Generalized Linear Model(GLM) 을 이용해 aircraft trajectories 를 예측한다. 하지만 논문에서는 4D trajectory가 아닌 도착 시간과 특정 포인트들(ex. 경유지, 목적지)만 예측한다.

ii. Choi and Herbert "Learning and predicting moving object trajectory" : Markov model을 이용해 과거 movement 를 통해 미래 motion 을 예측한다. 하지만 논문에서는 물체의경로를 실제 좌표들로 나타낸 것이 아니라 짧은 segment 의 연속으로 나타낸다.

iii. Ayhan and Sament "Aircraft Trajectory Predictidinates를 예측하도록 적용한다.
Made easy with Predictive Analytics":Choi and Herbert의 논문을 확장시켜 실제 4D coordinates를 예측하도록 적용한다.
Hidden Markov Model 를 이용해 날씨까지 반영한다. 하지만 세가지 문제점이 있는데,

1. 첫째로, 4D coordinate 가 들어갈 hidden state가 정해져 있다. 즉 예측한 track point 는 과
   거의 track points 중 하나로 나올 수 밖에 없다.
2. 둘째로, 모든 track point에서 날씨 조건만 고려되고 있는데, 실제로는 파일럿의 action이
   더 크게 고려되어야 한다.
3. 마지막으로, 날씨 데이터를 clustering 한 result 의 질에 따라 결과가 매우 갈린다.
   Aircraft trajectories는 높은 범위로 명확한 시간적, 공간적 패턴을 가지고 있기 때문에 올
   바른 도구들을 이용한다면 예측 모델에 잘 이용될 수 있음.

Sequence modeling(연속적인 입력으로부터 연속적인 출력을 생성하는 모델) 중 가장 최첨단 모델이 RNN인데, 이를 변형한 LSTM 이 여러 분야(음성 인식, 이미지 캡셔닝, 등)에서 사용되면서 증명되고 있음

- LSTM 관련 논문들
  - Z.Lin "Deep generative models of urban mobility" : IO-HMM을 이용해 도시의 사람들의 활동을 연속된 영역으로 나타내고 LSTM으로 학습시킨다.
  - A.Alahi "Social Istm : Human trajectory prediction in crowded spaces" : 번잡한 곳에서
    사람의 경로를 에측한다. 여러 개개인들의 trajectory를 LSTM network로 모델화한다. Pooling technique를 이용해 이 LSTM network들을 연결한다. 그 결과 단순한 가우시안 process linear 모델보다 좋은 성능을 보였다.

이러한 논문들에 영감을 받아 aircraft 4D trajectories 를 예측하는 encoder-decoder LSTM-based generative model을 개발함.

- 기존 접근과의 차이점 세가지
  1. 과거의 정보로부터 공간, 시간적 패턴을 학습해 완전한 4D trajectory를 생성하는 생성 모델(Generative model)
  2. 날씨 변수들로 생성된 고차원적인 지역적 특성을 이용함
  3. 학습 데이터 셋에서 발견되지 않는 새로운 경로를 예측할 수 있음

3. Preliminaries(작성자: 김석진)

3. 1. Definitions

1. Def 1. 4D trajectory $TP = [p_1, p_2, p_3, ..., p_T]$

   $P_t = [x, y, x, t]$ s.t. $x =$ 경도, $y =$ 위도, $z =$ 고도, $t =$ 시간
2. Def 2. pre-departure last fied flight plan $\tilde{X} = [\tilde{X_1}, \tilde{X_2}, \tilde{X_3}, ..., \tilde{X_{\tilde{T}}}]$

   $\tilde{X_i} = [x, y]$ s.t. $x =$ 경도, $y =$ 위도
3. Def 3. aircraft state $X_t = [x, y, z, \dot{x}, \dot{y}]$

   $X_t = [x, y, z, \dot{x}, \dot{y}]$ s.t. $x =$ 경도, $y =$ 위도, $z =$ 고도, $\dot{x} =$ 경도속도, $\dot{y} =$ 위도속도
4. Def 4. Georeferencing system G

   grid of fixed spatial points in the 3D space
5. Def 5. 4D matching algorithm

   4D trajectory point p_t와 가장 가까운 4D 거리를 가진 interests of dataset과 match하는 알고리즘
6. Def 6. feature cube F_t

   multi-dimentional aray of 4D matched features around a 4D trajectory point p_t

3. 2. Problem Formulation

각 flight마다 last filed flight plan to departure $\tilde{X}$가 있음

시간 t에 대한 aircraft state X_t가 주어질 때, 4D는 대기 온도 및 대류 날씨와 같은 기상 조건과 일치

그리고 다음 시간인 t+1에 대한 상태 예측

정리하면 flight plan, weather datasets, aircraft의 초기 상태가 주어질 때, 다음 상태와 날씨 정보를 일치시키는 4D를 재귀적으로 예측함으로써 마지막 timestamp까지의 전체 비행 궤적을 예측가능

→ translation + sequence generation problem의 결합

(encoder는 last filed flight plan과 일치하는 input sequence를 받고, decoder는 실제 flight trajectory와 일치하는 output sequence를 예측)

3. 3. Data sources and preprocessing

4 datasets from difference source

1. flight tracks dataset (comes from TFMS)

   각 항적에 대한 4D position(경도, 위도, 고도, 시간) 포함, 각 단위는 (1 min, 1 min, 1 feet, 1 min)

   - How to preprocess

     1. 2013년에 george bush 국제 공항에서 Logan 국제 공항으로 가는 flight로 scope 제한
     2. 공간적, 시간적 불연속성이 감지된 flight 제외
     3. 선택된 터미널 영역(0.5도 위도/경도 box) 밖에서 시작. 끝나는 flight 제외
     4. 2 track point 중 하나를 제거하여 flight를 down sizing(연산도 down)
     5. course, latitude and altitude speed 도출

        (각 flight가 두 연손된 지점에서 일정한 지상 속도를 갖는다고 가정)

     → 평균 sequence 길이가 94인 1,679개의 flight 얻음

2. flight plan datasets(comes from FAA TFMS)

   각 flight에 대한 last field 2D flight plan 좌표(위도, 경도) 포함

flight plan의 길이는 9 point부터 144 point까지 다양함

→ characteristic points를 identify함으로써 dimension down

> 💡 output sequence의 길이를 control하기 위해 parameter $\alpha \in [1, 2]$를 도입하여 Approximate Trajectory partitioning algorithm의 variant를 구현함

3. atmospheric dataset(comes from NAM)

   풍송, 기온 dataset 포함

   하루 4번(0시, 6시, 12시, 18시 UTC) high-resolution atmostpheric info 생성

   각 예측 주기는 0, 1, 2, …, 6시간 뒤 예측 제공

   - dataset

     Lambert Comfomal projection(지구를 평면에 표현하는 방법 중 하나)

     original horizontal resolution : 614X428

     (위도 $\in [12.19N, 57.33N]$, 경도 $\in [152.88W, 49.42W]$)

     하지만 항적이 미대륙에만 있어서 필요한 부분 제외하고 crop!

     그래서 horizontal resolution : 413 X 336

     dataset은 horizontal resolution외에도 50 ~ 1,000 millibar 에 이르는 39개의 등압 고도층이 포함

> 💡 최종 dataset은 1. westly wind speed, 2. westly wind speed, 3. air temperature로 이루어져 있고 각각은 39X413X336

4. convective weather dataset(comes from NCWF)

   각 record는 아래 내용 포함

   1. convective weather polygon(경계의 좌표 + 태풍의 최고고도)
   2. recoding 시간에 움직임 방향

   dataset은 5분마다 update

아래 과정을 거쳐서 dataset 이산화

1. 고유한 태풍의 고도들(단위 : 1,000 ft)

   [0, 14, 20, 24, 29, 35, 39, 45, 50, 54, 60, 65, 69]

2. 같은 고도 레벨 & 같은 시간(hourly) 그룹에 속하는 storm polygon 결합
3. 한 시간마다 13X413X336 차원의 binary array 생성

   (13 : # of altitude level, 413X336: georefencing system G의 horizontal resolution)

4. 한 시간마다 storm polygon을 G의 같은 고도에 overlay

   G의 grid point가 stom polygon에 cover

   - True : data array에 대응하는 요소가 1
   - False : data array에 대응하는 요소가 0

4. Feature Engineering(작성자: 김동연)

비행기 경로를 3가지 정보를 이용해 예측한다.

1. 제출된 비행계획(filed flight plan)
2. 대류 기상(convective weather)
3. 대기의 상황(atmospheric condition): 특히 공기 온도와 풍속

실제 비행경로 예측을 위해 기초 데이터 세트를 차후 모델의 기저를 제공하는 특성으로 변환해야 한다. 먼저 feature cube를 소개한 뒤, data cube를 비행경로에 효율적으로 매칭시키는 batch mode와 recursive mode 접근을 설명한다.

4.1 Feature Cube Referencing System

비행경로를 적절한 raw 데이터 세트와 매칭하기 위해 주어진 track point와 관련된 4D referencing system을 만든다. 각각의 track point에 대해 그 point를 둘러싸는 사격형 격자를 만든다. 그 격자의 폭과 높이는 각각 위도, 경도의 dx 및 dy이다. 그리고 사각형의 해상도는 nx, ny points다. 비행기 앞의 날씨 상태만 영향을 미치기 때문에 중심 대신 사각형 격자의 한 면을 track point의 중앙에 배치한다. 그 후에, 이전 track point의 경로로 사각형 격자를 회전시킨다.

대기 데이터 세트를 사용하기 위해 track point의 고도와 가장 가까운 압력 고도를 사용한다.

마찬가지로 대류 기상 데이터 세트를 사용하기 위해 L 리스트에서 track point의 고도와 가장 가까운 고도를 사용한다. 대응되는 track point의 timestamp로 사각형 격자의 timestamp를 설정한다.

4.2 Feature Cube Matching

설득력 높은 feature space를 위해 feature cube grid를 날씨 관련 데이터 세트와 매칭한다. 격자와 데이터 세트 둘 다 4D 구조다. 따라서 각각의 격자는 날씨 데이터 세트를 쿼리하고 가장 가까운 4D 기준점의 값을 특성값으로 사용해야 한다. 이 연구에서는 4D 방식으로 비행 경로와 높은 차원의 데이터 세트를 효율적으로 매칭하는 트리 기반 매칭 알고리즘을 제안한다.

2개의 k-d 트리가 필요하다. (공간 트리, 시간 트리) 공간 트리 TR_S는 데이터 입력을 격자의 위도, 경도의 튜플로 하는 horizontal georeferencing system G에 기반한다. 대기 데이터 세트와 대류 기상 데이터 세트는 같은 G를 공유하기 때문에 TR_S는 일정하다. 날씨 데이터세트에는 고도의 종류가 적어 인덱싱과 그룹을 만들어 빠른 고도 매칭이 가능하기 때문에 TR_S는 고도 레벨을 포함하지 않는다. TR_T는 각각의 대류 기상 데이터 입력에서 미리 특정된 시간에서부터 경과한 시간을 바탕으로 만들어진 일차원 트리다.

- 효율적인 4D 매칭
    1. feature cube grid를 모든 비행 track들에 대해 batch mode로 생성한다. (Algorithm 1)
    2. G에서 가장 가까운 grid point의 인덱스를 반환하는 feature cube 2D grid를 사용하여 TR_S에 쿼리를 일괄처리한다.
    3. feature cube를 고도에 따라 그룹화한다. 하지만 폭풍은 바르게 퍼지기 때문에 현재 고도의 대류뿐 아니라 위아래의 대류도 고려해야 한다. 따라서 고도 범위 안의 데이터를 부분집합으로 모두 사용한다.
    4. 각 그룹 내에서 최대 시간 거리 경계가 있는 feature cube grid timestamp를 사용해 시간 트리에 쿼리하고 날씨 데이터 세트에서 feature cube grid까지 가장 가까운 시간 인덱스를 수집한다.
    5. 행렬의 차원이 (k, nx, ny)인 매칭된 대류 기후를 수집하기 위해 세 개의 인덱스(고도, 2D 공간, 시간)을 사용한다. k는 고도 버퍼의 고도 레벨이다.

6. 일치하는 각각의 feature cube grid에 대해 매칭된 날씨 배열의 모든 k 레이어를 덮어 결과 배열이 (nx, ny) 차원을 가지며 해당 그리드에 대류가 있으면 각 요소가 1이고 그렇지 않으면 0이 되도록 한다.

대기 데이터 세트에는 고도 버퍼를 특정 짓지 않고 유사한 과정을 거친다. 따라서 각 feature cube grid에 대해 최종 매칭된 feature cube는 4개의 레이어가 있다. (각 레이어의 차원은 (nx,ny))

- 1st layer: 대류 날씨를 나타내는 이진 배열
- 2nd layer: 공기 기온
- 3rd layer: 서쪽 풍속
- 4th layer: 남쪽 풍속

5. Module design

항적이 비행 계획에서 이탈하는 경우 많음

그런 '이탈'을 예측하기 위해 문제를 아래와 같이 formulate

problem = translation + sequence generation

(flight plan을 실제 flight 경로로 translate하기 위해 weather infomation을 순회적으로 이용)

→ 우리 모델은 3가지 module 이용 (Figure 4 참고)

1. encoder LSTM (flight plan을 fixed size feature vector로 embed하기 위한)
2. decoder LSTM(fixed-size feature vector를 타겟 flight trajectory sequence로 mapping하기 위한)
3. set of convolutional layers(high-dimension weather-related feature cube를 fixed-size feature representaion로 condense하기 위한)

5. 1. Architecture(작성자: 김석진)

- Encoder-decoder LSTM
  - Encoder LSTM ($\tilde{LSTM}$)
    - input: flight plan sequence($\tilde{X}$)
    - output: sequence of hidden states(Eq 1 참고)

- decoder LSTM

  $\tilde{H}_q$를 initial hidden state로 사용하고 실제 항적 상태의 전체 sequence를 예측

- 가정: 각 timestamp의 state는 decoder network에 의해 학습되는 parameter를 가진 Gaussian mixture를 따름
- 예시) Eq 2 참고

  decoder net은 H_{t-1}($t-1$의 hidden variable), X_t(current aircraft state), convolutional net에서 얻은 weather-related feature representations C_t, 학습할 weight 변수 W를 input으로 하여 Gaussian mixture의 parameter를 구성하는 hidden variable H_t를 output
- Eq 3의 시사점

  X_t는 각각 weight ϕ_t^i를 가지는 K개의 독립된 Gaussian 분포로부터 도출됨
- loss function = negative log likelihood(Eq 7 참고)

- Convolutional layers

  deep CNN은 분류뿐만 아니라 high dimensional input signal로부터 feature representation 학습가능

  각 track point p_t는 feature engineering을 통해 high dimension cube인 F_t와 match

  → feature cube는 width, height가 항공기 위치 주변 relevant region의 크기이고 channel의 개수가 weather-related feature의 개수인 multi-channel 'image'로 간주됨

  > 💡 feature cube는 4 channel
  > (westly, southly wind speed, air temperature, convective weahter)

  feature representation 추출을 위해 multiple convolutional layer 도입

  (각 timestamp마다 feature cube를 fixed-size feature vector로 abstract하기 위한 small filter 사용하여 decoder LSTM net에 직접 feed)

  → train하는 동안 loss는 모든 convolutional layer의 weight로 back propagate

5. 2. Inference Process(작성자: 정예준)

inference를 위해 필요한 정보 : last filed flight plan, first T' states(1<= T' <=T),

corresponding T' weather-related feature cubes

trained network 에 위 데이터들을 넣고 순전파를 돌린 후 Gaussian mixture parameter set을 얻는다. 그 다음 Gaussian mixture를 이용해 $X_{t'+1}$ state를 sample 한 뒤 algorithm 1,2를 이용해 $F_{t'+1}$을 얻는다. 이것을 계속 반복하다보면 T' 이후의 전체 tragectory를 얻을 수 있다. 간단한 방법이지만, 크게 불일치가 생기다보니 sampling을 여러 point에서 하자는 의견이 있지만, Gaussian distribution 의 대칭성 때문에 "zigzagged" prediction이 생길수 있다.
그래서 sampling technique를 이용하지 않고 각각의 Gaussian component의 평균 벡터를 이용한다. 결과 값인 log likehood를 내림차순으로 rank를 매겨 가장 높은 값을 얻어낼 것이다. 먼저 같은 방법으로 Gaussian mixture parameter들을 얻은 뒤 Eq.8을 이용해 log likelihood를 계산한다. mean vector가 Adaptive Kalman filter 에 의해 outlier(다른 데이터에 비해 동떨어져있는, 불필요한 데이터)로 판별된 경우 log 함수의 최종값이 음수로 나오게 된다. Eq.8에서 보면 각각의 log 에 가중치를 매기는데, 올바른 Gaussian component를 선택하기 위해서이다. 또한 식에서 L_{t-1}을 이용하는데, 따라서 우리가 순전파로 매 t마다 K개의 Gaussian mixture 값을 얻는데 각각의 mixture parameter를 얻을 때마다 그 전 부모 state로 돌아가 log likelihood를 계산해서 값을 가져와야 한다. 그리고 Adaptive Kalman filter를 통해 기존의 t-1 때의 평균과 지고 때의 평균과 분산을 더욱 정확하게 얻어낼 수 있다. Eq.9에서 볼 수 있듯 결과로 나오는 M_t의 값은 input으로 넣어준 mean vector 값이 outlier인지 아닌지를 알려주는 값이다. L_t를 내림차순으로 rank 한 뒤에 가장 큰 Nbs sequcences를 가져오고 Algorithm 1,2,를 이용해 aircraft state와 weather feature를 매칭한 후 순전파를 통해 Gaussian mixture parameter를 얻는다. 이걸 마지막 T까지 계속 반복해서 가장 뛰어난 Lt의 sequence를 가져온 뒤 RTS(Rauch-Tung-Striebel)를 이용해 예측을 향상시킨다.

5. 3.Implementation Details(작성자: 김동연)

- Feature Engineering
  feature cube grid의 수평 사이즈:2x2 (해상도: 20×20)
  약 120mile×120mile 정도의 면적이다. 각 feature cube의 차원은 20×20×4이다.

  4개의 레이어 중 대류 날씨를 제외한 특성들을 정규화한다.

  잠재적으로 다양한 공항에 이 모델을 적용하기 위해서 출발 공항의 좌표를 제거해 학습과 추측에 사용한다. 이 경로 또한 정규화한다.

- Training

- 인코더
  처음으로 비행 계획 좌표를 32차원의 embedding layer에 입력한다. 그 결과값을 2개의 layer를 가진 LSTM에 투입한다. K=3인 Gaussian mixture components가 실제 비행 경로의 확률 분포를 모델링하도록 선정된다. 각 timestamp에서 디코더 LSTM의 파라미터 수는 45개다.
- convolutional layer
  날씨 feature cube로부터 representation을 학습한다. 3개의 convolutional layer와 1개의 fully-connected layer로 구성된다. 첫 번째 layer는 feature cube를 받고 6X6X64 크기의 필터 16개와 stride 2를 사용한다. 두 번째 layer는 3×3×16 크기의 필터 16개와 stride 1을 사용한다. 세 번째 layer는 3×3×16 크기의 필터 32개와 stride 1을 사용한다. fully-connected layer는 32개의 뉴런을 가진다. 날씨의 위치 정보를 중요하게 다루기 때문에 pooling과 padding은 사용하지 않는다.
- 디코더
  디코더 네트워크도 2개의 layer를 가진 LSTM 구조이고 인코더 LSTM의 마지막 hidden state를 initial state로 받는다. convolutional layer의 feature representation, 항공기 상태 변수들은 디코더 LSTM에 들어가기 전에 embedding layer를 거친다. LSTM의 결과는 Gaussian mixture의 파라미터를 판단하는 데 사용된다. 모든 활성함수에 exponenetial linear unit(ELU)를 사용한다.
- optimizer: Nesterov Momentum optimizer
- learning rate: 0.001 -> 매 1000번 반복마다 감소

- Inference
  누적 log likelihood를 계산할 때 Gaussian 요소에 더 가중치를 둔다. $\pi_1 = 0.8, \pi_2 = 0.2$
  adaptive Kalman filter에서 각각의 timestamp에 대해 항공기는 수평으로 일정한 속도로 이동하고 수직으로 속도가 0이라는 간단한 선형 역학을 가정한다. 수평 속도와 고도는 측정(학습된 neural network의 예측)에 의해 확인되고 선형 역학 시스템의 에러로 다뤄진다. 과정 에러 Q는 대각행렬로 간주한다. 그 측정값이 이상점인지 방향 조절인지 또는 둘 다 아닌지 구분하기 위해 두 개의 임계값 $e_1 = 0.8, e_2 = 0.3$을 선택한다.즉, 위도 경도의 절대 에러가 e_1보다 큰 경우 그 측정은 이상값이다. 이때 음의 값인 $l(M = 1) = -9$가 궤도의 누적 log likelihood에 더해질 것이다. 만약 에러가 e_2 보다 크다면 그 측정을 방향 조절로 취급하고 과
  정 에러 Q를 $Q_S = 10$의 인자로 증가시키고 Kalman filter 과정을 반복한다.

6. Experimental Results(작성자: 정예준)

- Dataset: 1679 flights, 80% training, 20% evaluation set으로 이용됨

- Inference process에서 evaluation set 안에 20개의 track points와 feature cube를 observed sequence로 이용하고 나머지 flight track을 algorithm 4를 통해 예측함

Predicted trajectories가 actual flight track deviation을 보였다. 이 deviation 도 신뢰 구간 안에서 커버가 가능했다.

이 모델의 성능을 평가하기 위해 4가지 종류의 에러를 명시한다.

1. Point-wise horizontal error : 예측 포인트와 실제 포인트의 2d latitude와 longitude 좌표 거리 차이
2. Point-wise vertical error(유일하게 부호가 존재) : 예측 포인트와 실제 포인트의 altitude 거리 차이
3. Trajectory-wise horizontal error: point-wise horizontal error
4. Trajectory-wise vertical error: point-wise vertical error

Point-wise horizontal error는 그래프에서 왼쪽으로 분포가 많이 치우쳤고 평균은 49.60 nautical miles이다.

Point-wise vertical error는 주로 -5000ft에서 5000ft에 분포가 몰려있다.
Trajectory-wise horizontal, vertical error는 둘 다 왼쪽으로 분포가 치우쳤고 평균값은 point-wise error와 비슷했다.
마지막으로 outlier flight 같은 특이한 departure procedure의 경우 큰 에러를 관찰할 수 있었는데, 추후research를 통해 해결 방법을 모색할 필요성이 있다.

최종적으로, neural network의 convolutional layer들을 시각화하면서 이 section을 마침.
Feature cube를 시각화함
Figure 9: input feature cubes

First row: convective weather layer
Second row: air temperature laayer
last two rows: Westly and southerly wind speed layer

Figure 10, 11: output feature cubes