Deep Learning for Prediction of Meteorlogical Fronts

Keras tutorial (link to Jupyter notebook)

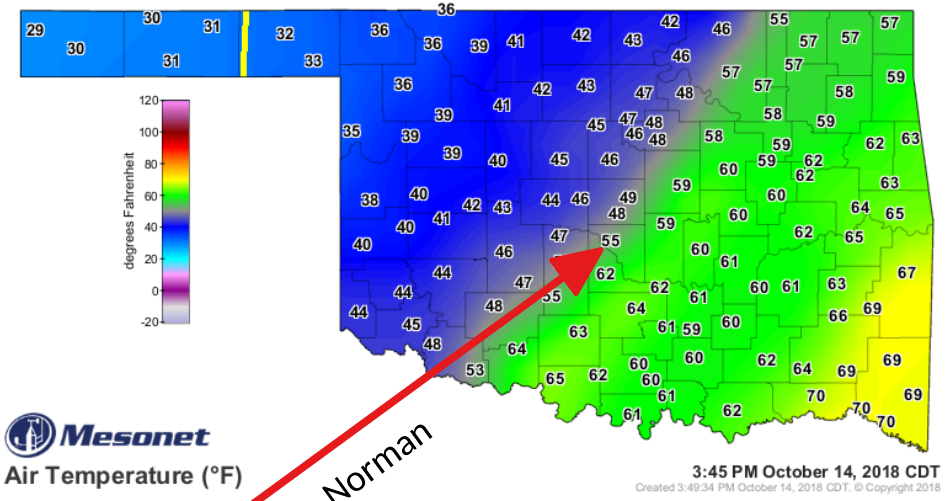
2018 Symposium on Artificial Intelligence and Machine Learning

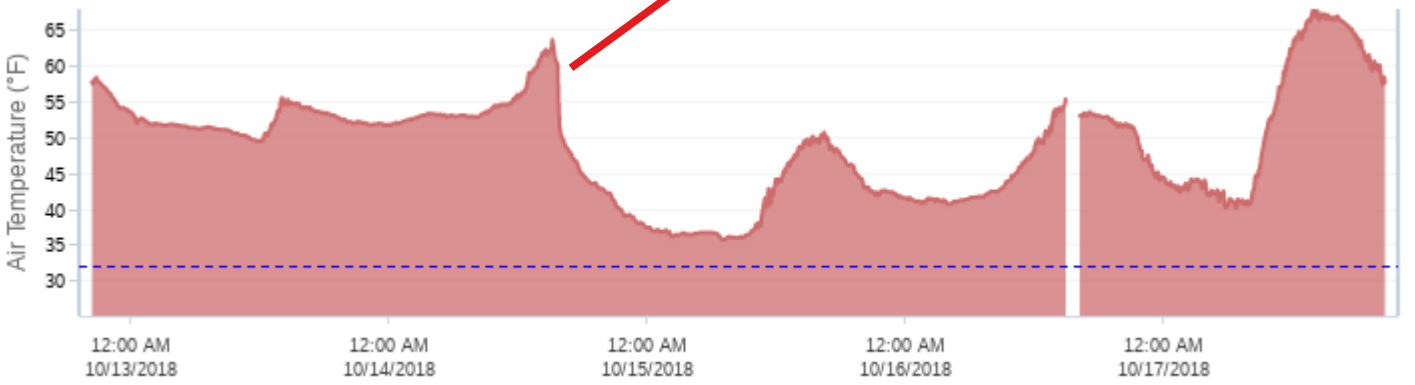
University of Oklahoma

Ryan Lagerquist (ryan.lagerquist@ou.edu)

Section 1: Problem

- Warm and cold fronts are part of global atmospheric circulation.
- They occur mainly in the mid-latitudes (*e.g.*, Oklahoma).
- Often associated with dramatic changes in weather, sometimes with extreme weather such as:
  - Severe thunderstorms
  - Blizzards
  - Heavy rainfall
  - Extreme cold and wind chill

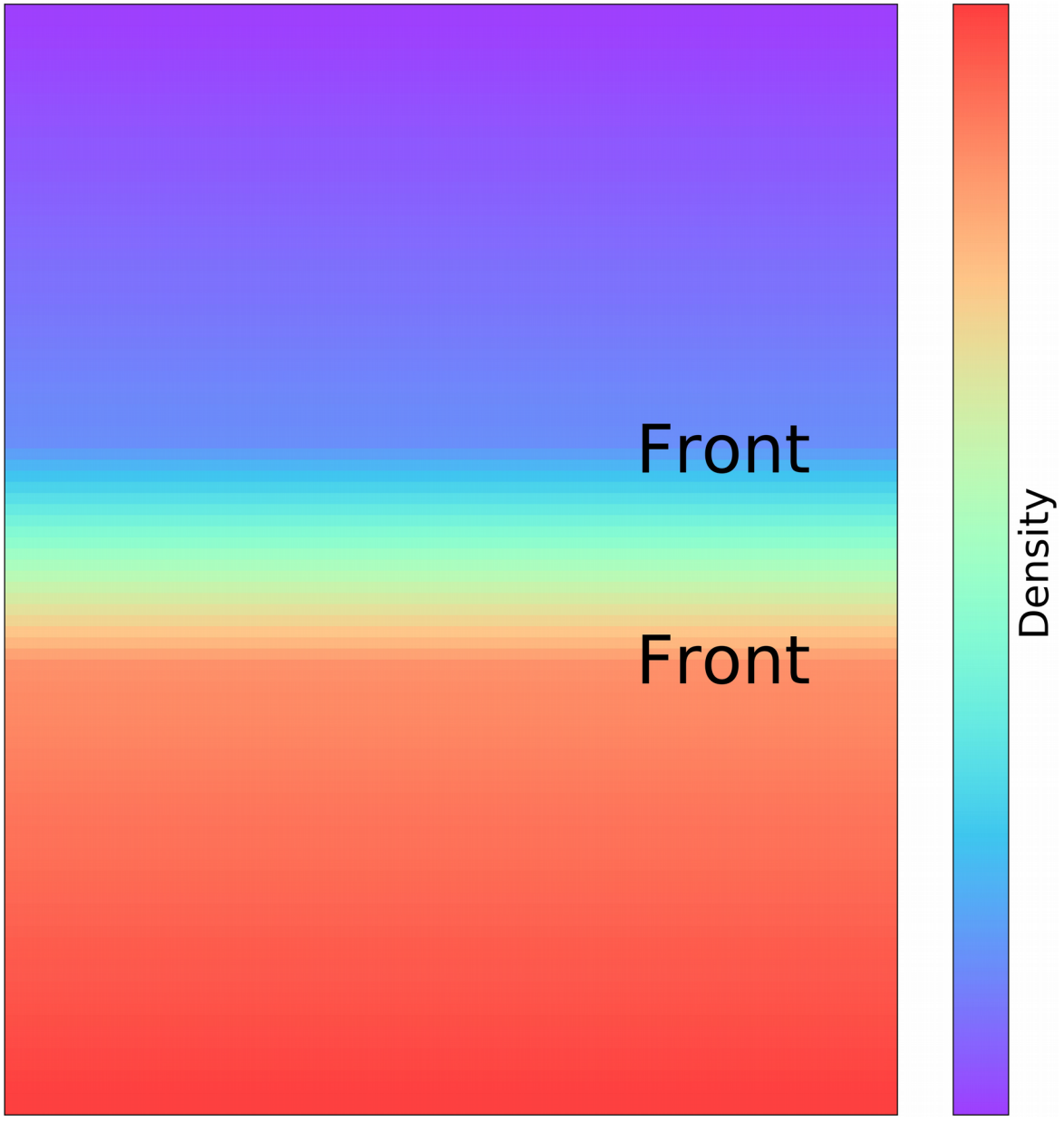




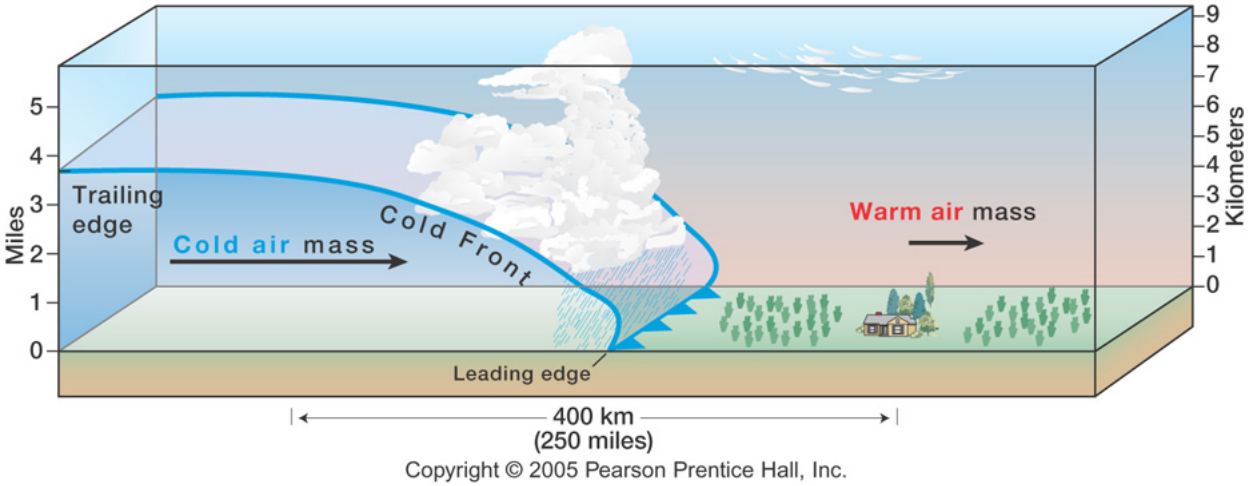
Temperature history in Norman. Both images from http://mesonet.org.

Section 1: Problem

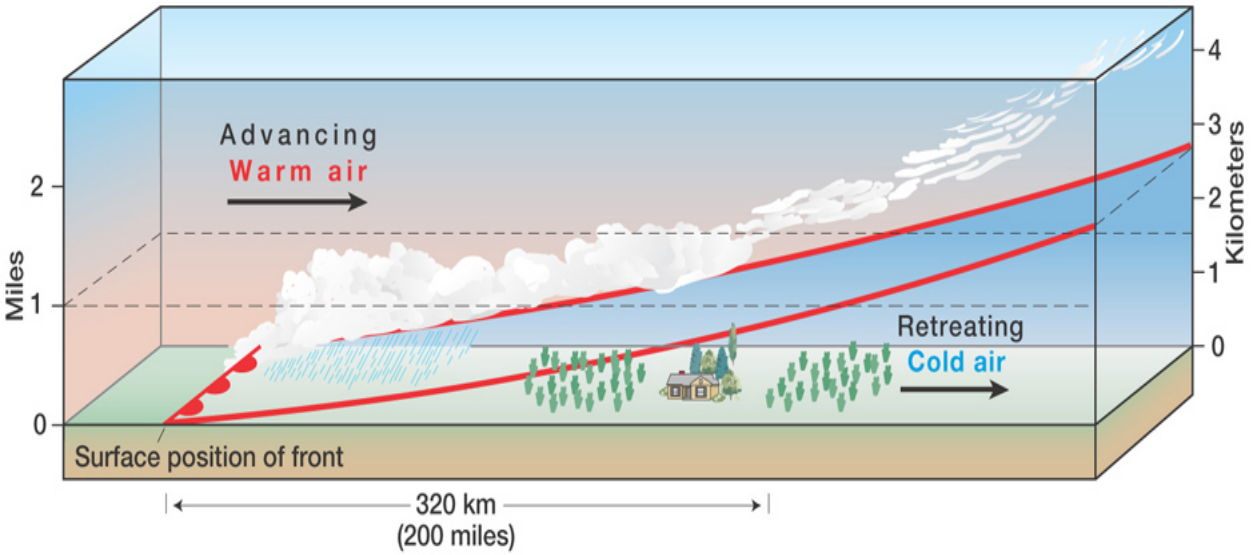
- What is a front?
- Technically: quasi-vertical transition zone between two air masses of different density.
- This assumes homogeneous air masses with discontinuity at front.
- In reality it is more complicated (the atmosphere is a continuous fluid).
- There are small density gradients within air masses, larger gradients at the front.



Section 1: Problem

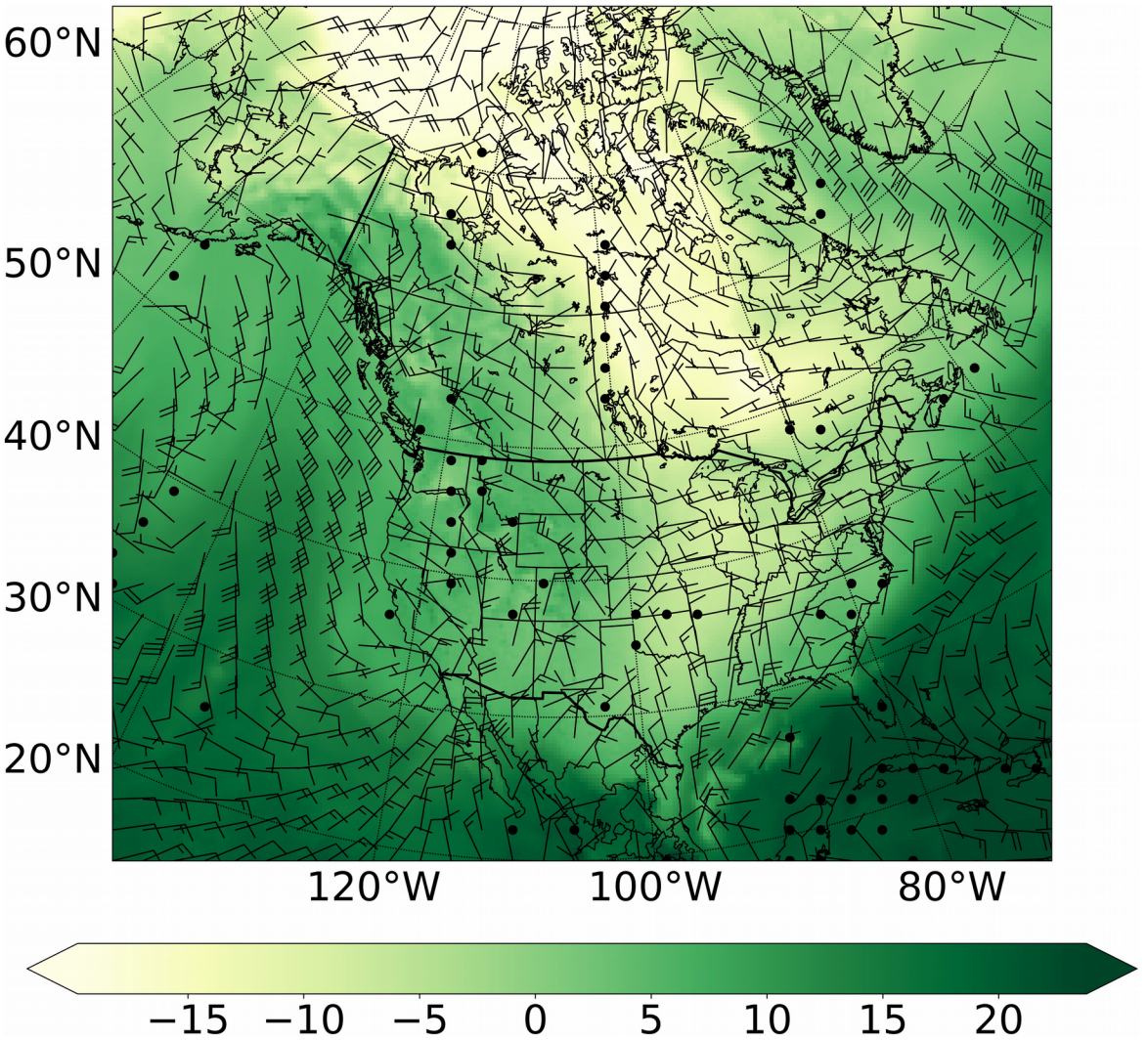


Images downloaded from
http://web.gccaz.edu/~lnewman/gph111/topic_units/airmass_frnt_cyclones/airmass_frnt_cycl/airmass_frnt_cycl2.html.



Section 1: Problem

- Inputs (predictors) come from North American Regional Reanalysis (NARR).
- 32-km grid over North America, every 3 hours, dating back to 1979.
- **Right:** a few NARR variables at one time.
- **Green fill** = wet-bulb potential temperature (°C)
- **Black lines** with weird appendages = wind barbs

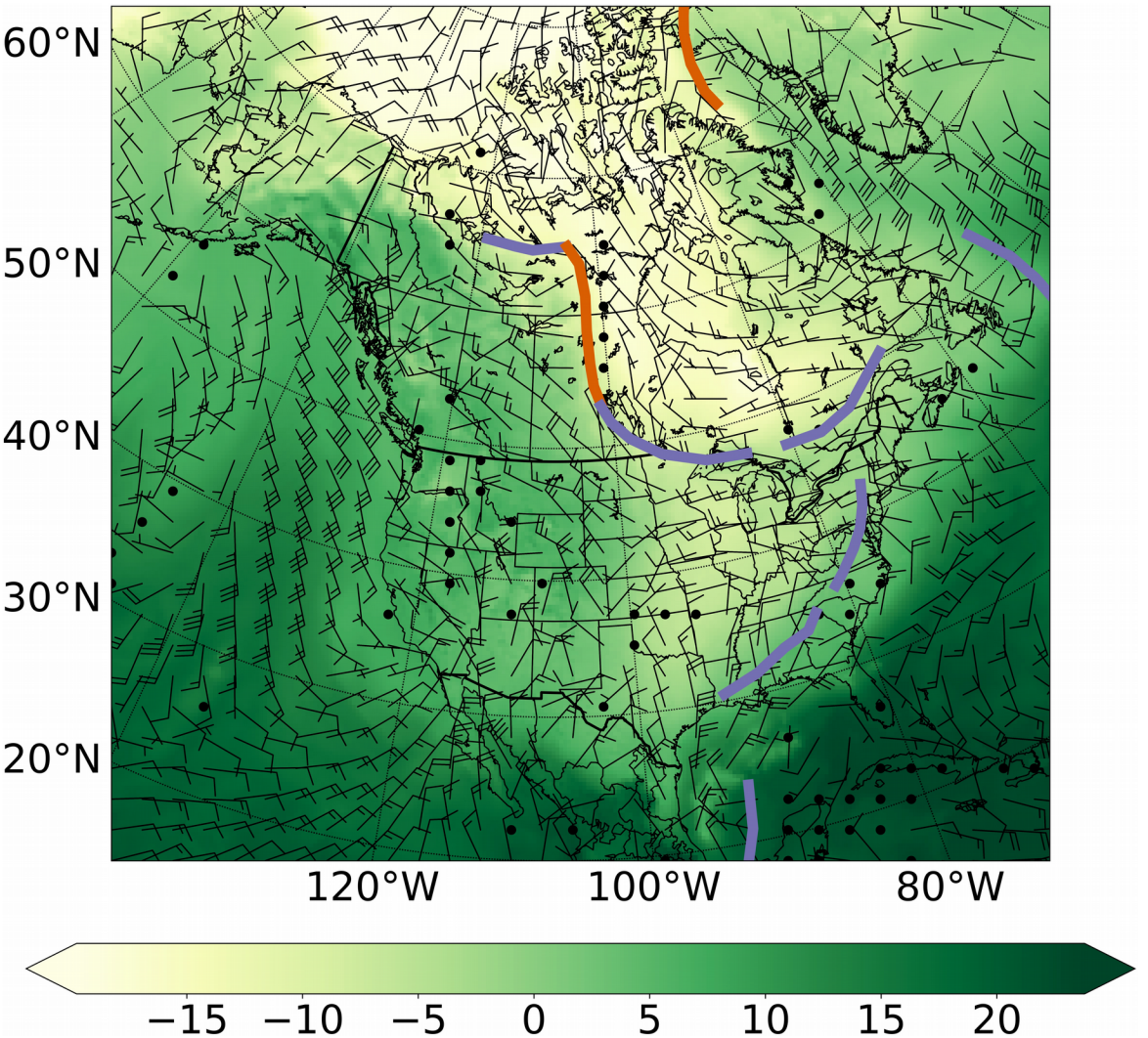


Section 1: Problem

- Labels (verification data) come from Weather Prediction Center (WPC) surface bulletins.
- Surface bulletins include warm and cold fronts drawn by human meteorologists.
- Surface bulletins available every 3 hours back to Nov 5 2008.
- Prediction task is to label each grid cell as:
  - Part of a warm front (WF)
  - Part of a cold front (CF)
  - Non-frontal

Orange lines = warm fronts

Purple lines = cold fronts



Section 2: Convolutional Neural Nets

- Convolutional neural nets (CNN) are a type of deep-learning model.
- CNNs are specially designed to learn from spatial grids (images).
- Traditional machine learning uses images in one of two ways:

    1. Learns from pixel values
        - Pixel values treated as independent scalars, which prevent model from learning spatial structures.

    2. Learns from features
        - “Features” are spatial statistics that describe the image (*e.g.*, mean, median, min, max, standard deviation).
        - Features account for some, but not all, spatial structure.
        - In other words, information is lost.
-

Section 2: Convolutional Neural Nets

- Convolutional neural nets (CNN) are a type of deep-learning model.
- CNNs are specially designed to learn from spatial grids (images).
- Traditional machine learning uses images in one of two ways:

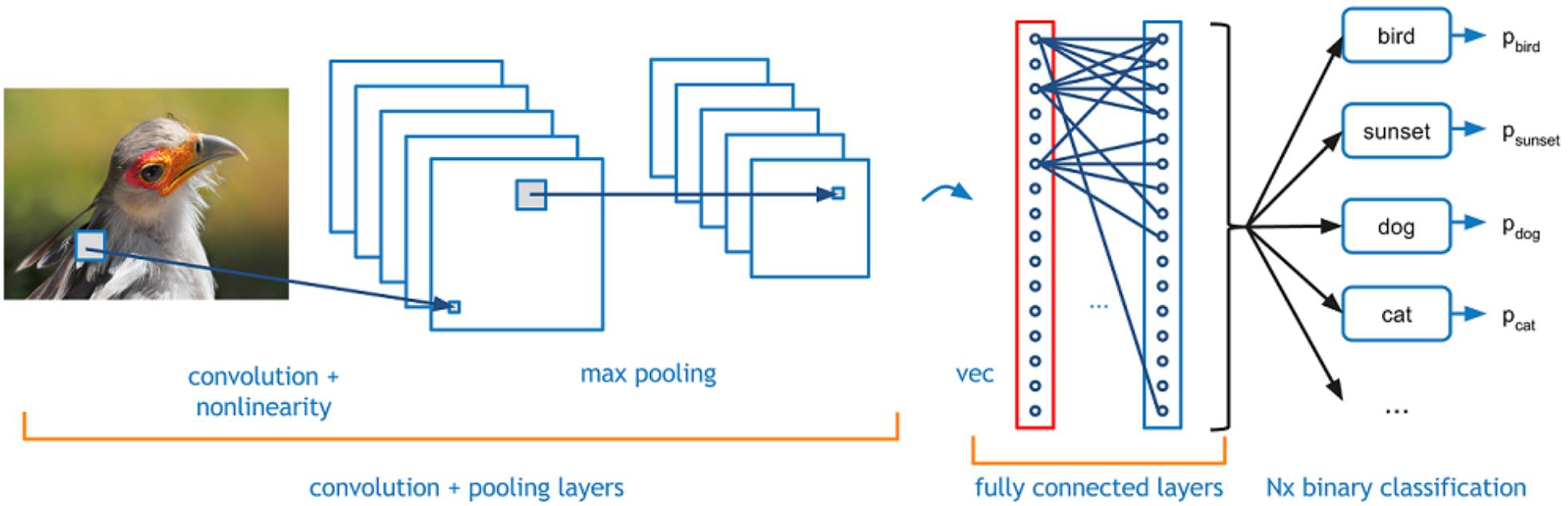
    1. Learns from pixel values
        - Pixel values treated as independent scalars, which prevent model from learning spatial structures.

    2. Learns from features
        - "Features" are spatial statistics that describe the image (*e.g.*, mean, median, min, max, standard deviation).
        - Features account for some, but not all, spatial structure.
        - In other words, information is lost.

- Benefit of deep learning: it creates features **and** converts them to predictions.
- Feature creation and prediction are integrated, so deep learning can discover optimal features for prediction task.
- Usually results in better prediction than choosing features *a priori*.

Section 2: Convolutional Neural Nets

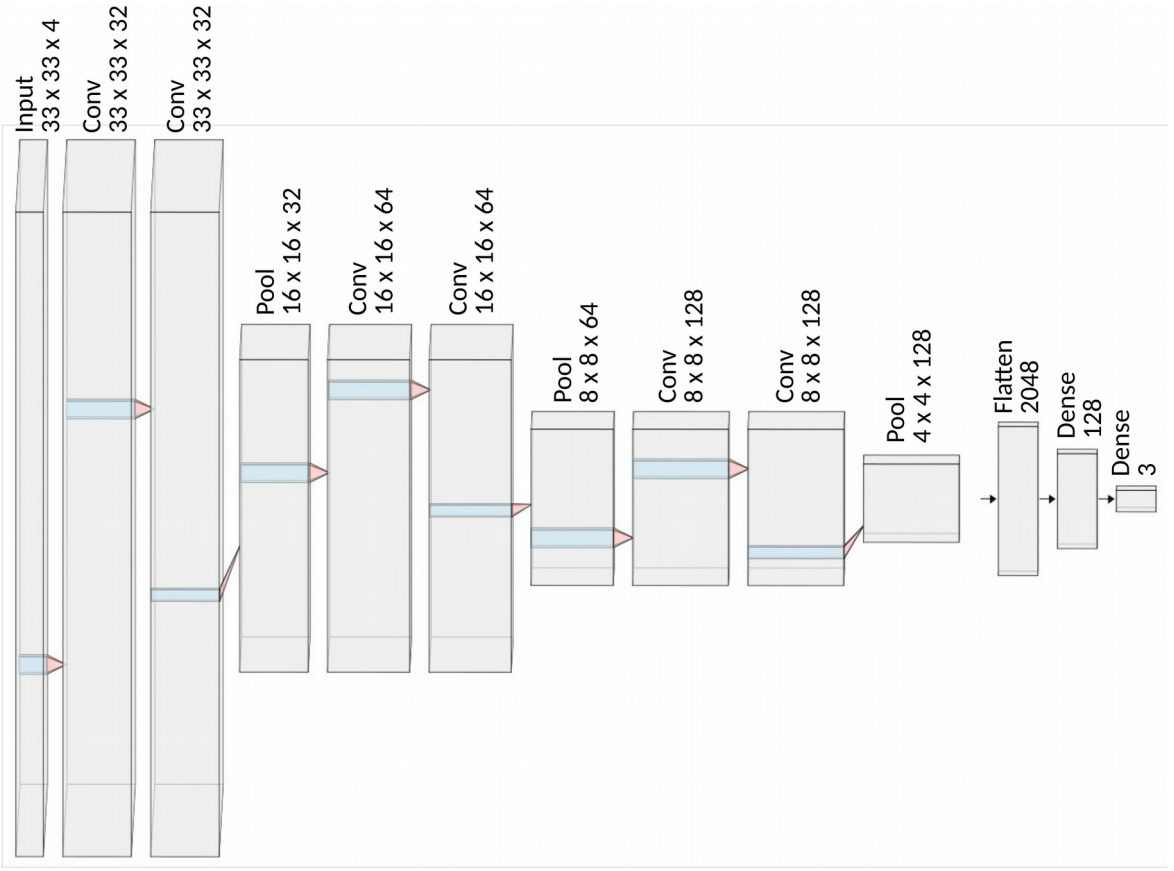
- Main components of CNN are convolutional, pooling, and dense layers.
- Convolutional layers use convolutional filters with trainable weights to create features.
- Pooling layers decrease image resolution, which allows following convolutional layers to learn larger-scale features.



Section 2: Convolutional Neural Nets

- Dense (“fully connected”) layers, from traditional neural nets, convert learned features to prediction.

- Thus, CNN solves two tasks simultaneously:
  - Learning the best features
  - Learning the mapping from features to output (prediction)



Section 3: Pre-processing

- This section describes data-processing done before CNN-training.

1. Split time period into training, validation, and testing.

   (a) Training data are used to fit the CNN (adjust weights).

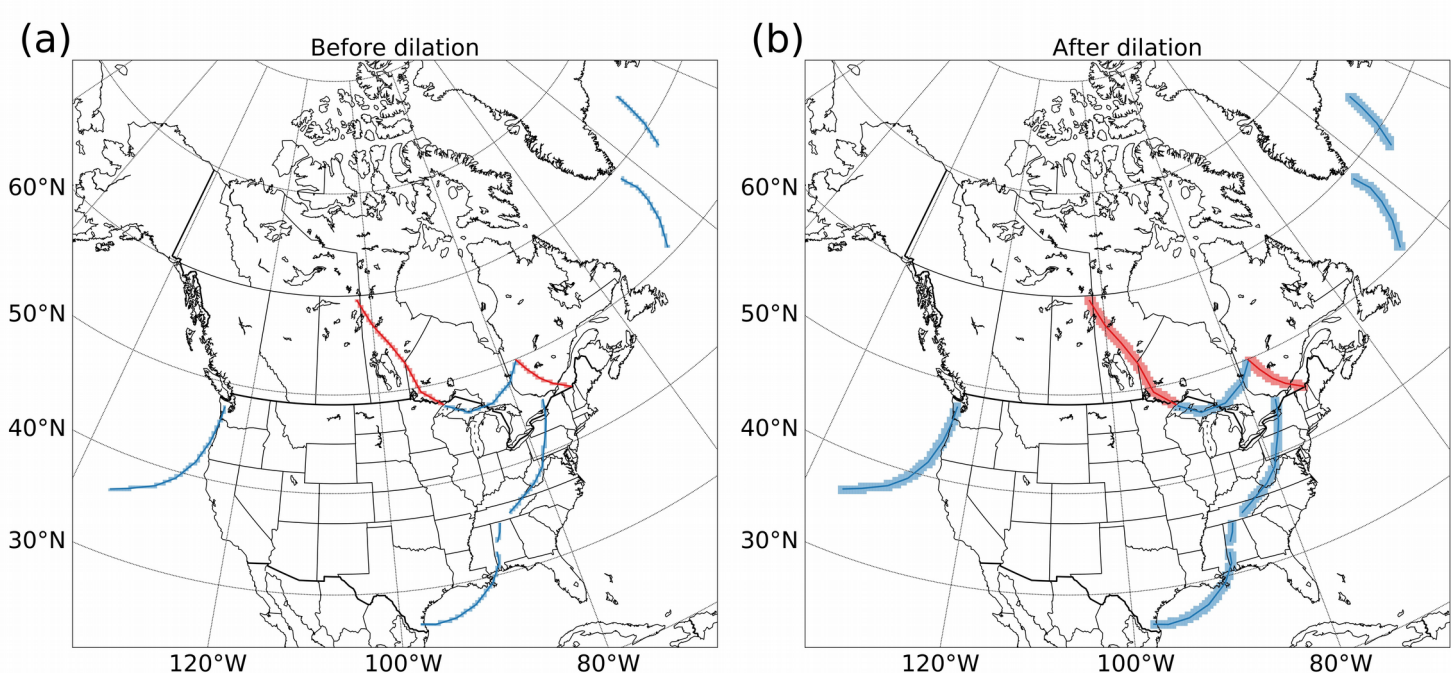
   (b) Validation data are used to:
   - Monitor CNN during training
   - Choose the best CNN (best architecture or settings)

   (c) Testing data are used to evaluate performance (predictive skill) on previously unseen data.

Dataset	Time Period
NARR	0000 UTC 1 Jan 1979 – 2100 UTC 31 Dec 2017
WPC fronts	1500 UTC 5 Nov 2008 – 1200 UTC 18 Jan 2018
Training period	1500 UTC 5 Nov 2008 – 2100 UTC 24 Dec 2014
Validation period	0000 UTC 1 Jan 2015 – 2100 UTC 24 Dec 2016
Testing period	0000 UTC 1 Jan 2017 – 2100 UTC 31 Dec 2017

Section 3: Pre-processing

2. Dilate fronts.
   - Accounts for representativity error in NARR grid and human analysis.

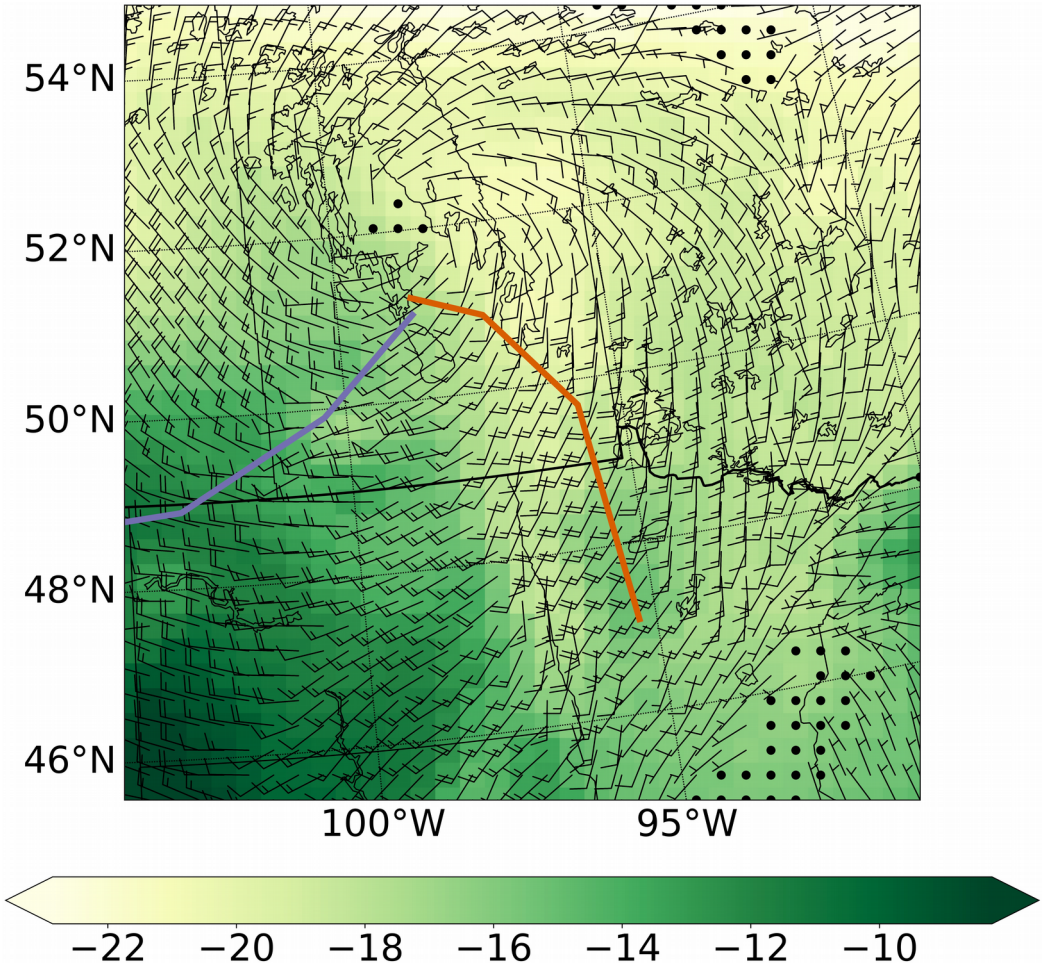


3. Choose predictor variables:
   - Temperature
   - Specific humidity (kg of water vapour per kg of air)
   - Wind velocity
   - Wet-bulb potential temperature
   - Geopotential height (similar to pressure)

4. Normalize all predictors to the same scale, ~[0, 1].

Section 3: Pre-processing

5. Crop 65 x 65 subsets of grid.
   - Target value (NF for no front, WF for warm front, CF for cold front) based on center grid cell.
   - Target value = WF in example shown below.



Section 4: Model Evaluation

- CNN outputs 3 values (with respect to center grid cell):
  - Probability of no front (NF)
  - Probability of warm front (WF)
  - Probability of cold front (CF)

- To make things easier (avoid issues beyond scope of tutorial), we transform CNN predictions in two ways:
  1. **Determinize:** convert probabilities to deterministic labels (NF, WF, or CF).
  2. **Binarize:** turn problem into "front or no front".

- Then we create contingency table.
  - *a* = number of true positives (prediction = observation = yes)
  - *b* = number of false positives (prediction = yes, ob = no)
  - *c* = number of false negatives (prediction = no, ob = yes)
  - *d* = number of true negatives (prediction = ob = no)

Observation	Yes	No
Prediction		
Yes	*a*	*b*
No	*c*	*d*

Section 4: Model Evaluation

- Contingency table used to compute the following metrics.
- Don’t worry about memorizing equations (repeated in notebook).
- This slide is here as a quick reference.

Observation	Yes	No
Prediction		
Yes	*a*	*b*
No	*c*	*d*

$$\text{Probability of detection} = \text{POD} = \frac{a}{a+c}$$

$$\text{Probability of false detection} = \text{POFD} = \frac{b}{b+d}$$

$$\text{Success ratio} = \text{SR} = \frac{a}{a+b}$$

$$\text{False-alarm rate} = \text{FAR} = \frac{b}{a+b} = 1 - \text{SR}$$

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{a+d}{N}$$

$$\text{Critical success index} = \text{CSI} = \frac{a}{a+b+c} = \frac{a}{N-d}$$

$$\text{Frequency bias} = \text{FB} = \frac{a+b}{a+c}$$