CISESS

Roger Yang

August 2023

Overview

01

Introduction

The storm dilemma. Detection vs. forecasting

02

Detection and forecasting

Storm detection and forecasting using ML.

03

Outlook

Analysis of results and outlook

01

Introduction

Introducing some terminology

Detection vs. Forecasting

This presentation will refer to **detection** and **forecasting** models throughout.

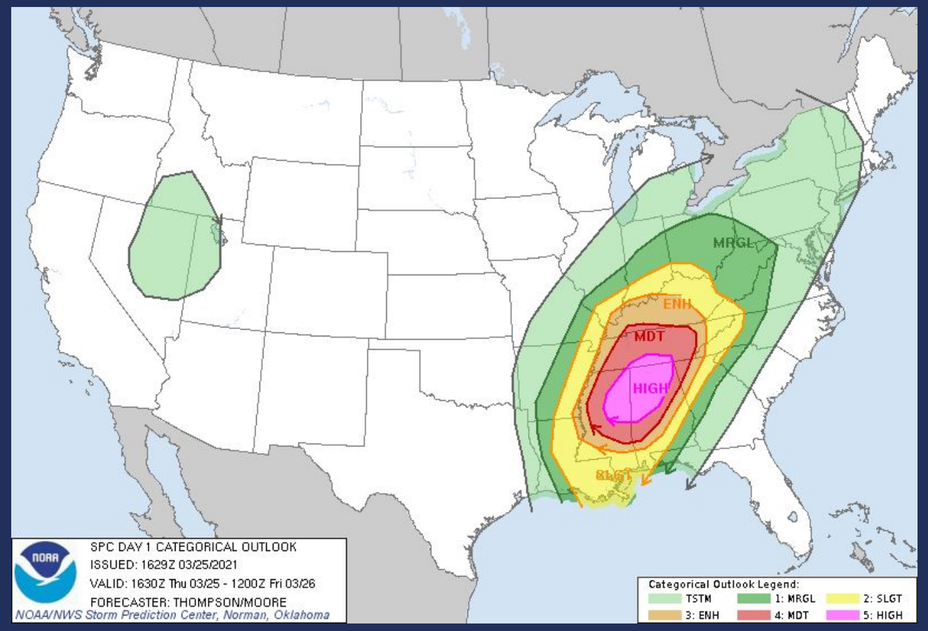
Detection	Identification or classification of historical severe storm events based on non-storm data (like temperature, pressure, humidity, etc.).
Forecasting	Prediction of future severe storm events using historical storm data.

Statistical vs. ML classifiers

Both the detection and forecasting models use some sort of **classifier** to understand complex relationships between input data. There are two main variants of classifiers commonly used for detection or forecasting models:

Statistical classifiers	Use statistical inference techniques to detect correlation between input datasets. A simple example of a statistical classifier is linear regression.
Machine learning classifiers	Use some form of machine learning model to compute weights (for example, using backpropagation) which determine predictive relationships.

Methodology



Below shows the types of classifiers used in the detection and forecasting models.

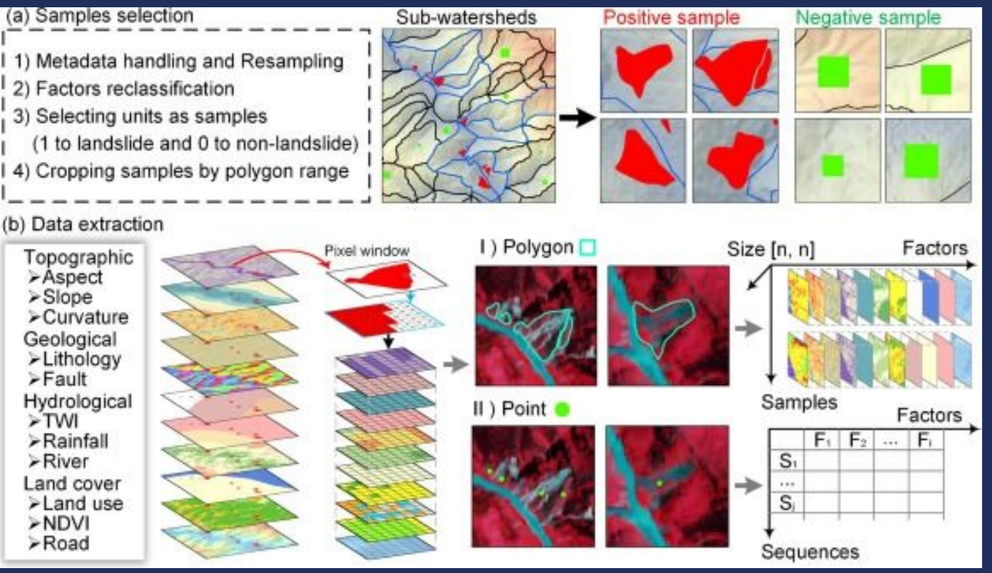
Model	Detection	Forecasting
Inputs	NOAA reports	NOAA reports & NetCDF data
Type of classifier	Statistical	Machine learning
Name of classifier	U-Net	ConvLSTM

02

Detection

Building the Detection Model

Detection model



The architecture of the detection model is inspired by the model from this landslide detection paper. Specifically, the model first encodes spatial input through a two-step process:

1. The number of spatial channels is reduced to the effective dimension of the channel and the reduced channels are flattened through spatial pyramid pooling (SPP)
2. Further flatten the data using a channel-wise SPP. Feed into a linear layer to eliminate permutation-based hyperparameters

Note that (1) incorporates spatial relationships and (2) incorporates channel-wise relationships. The transformed input data is then fed into a classifier like an NRF or SVM.

The spatial inputs for this version of the model are the precipitation, wind direction, and wind velocity, extracted from Google Earth Engine. These inputs can be extended easily to other datasets.

Encoding

The architecture of the encoding branch is inspired by the model from this landslide detection paper.

Spatial

Step 1

Depthwise Conv.

The first part of a Separable CNN

Iterates over all channels and applies SPP

Channel-wise SPP

Step 2

Pointwise Conv.

The second part of a Separable CNN

Merges all SPP's through concatenation

Concatenation

Step 3

SPP Flattening

Spatial Pyramid Pooling

Interpolates data with linear activation

Linear Layer

Response

Step 4

Concatenation

Combine the two branches to create a unified feature map

Decoding/Classifying

Encoded data is not immediately useful unless it is run through certain processes to extract the desired output. There are two main ways of doing this and they are not mutually exclusive.

Decoding

Taking encoded data which does not contain any immediately discernible data and extracting useful information

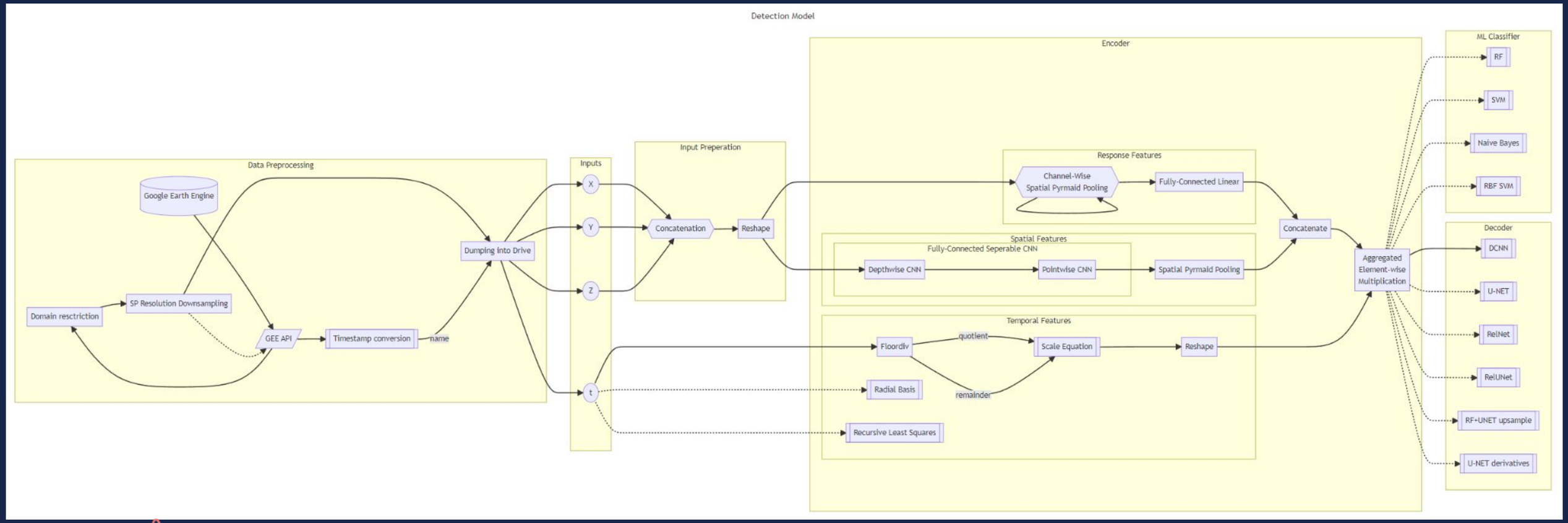
U-Net	Biomedical imaging model used for classification
DCNN	Deep Convolutional Neural Network
ResUNet	Residual U-Net; novel approach to semantic segmentation

Classifying

Taking spatial data and classifying it using certain algorithms

Naive Bayes	Classification technique based on Bayesian probability
NRF	Differentiable form of a Random Forest
RBF-SVM	Radial Basis Function kernel of a Support Vector Machine

Detection model - architecture



03

Forecasting

Conclusion and things to improve

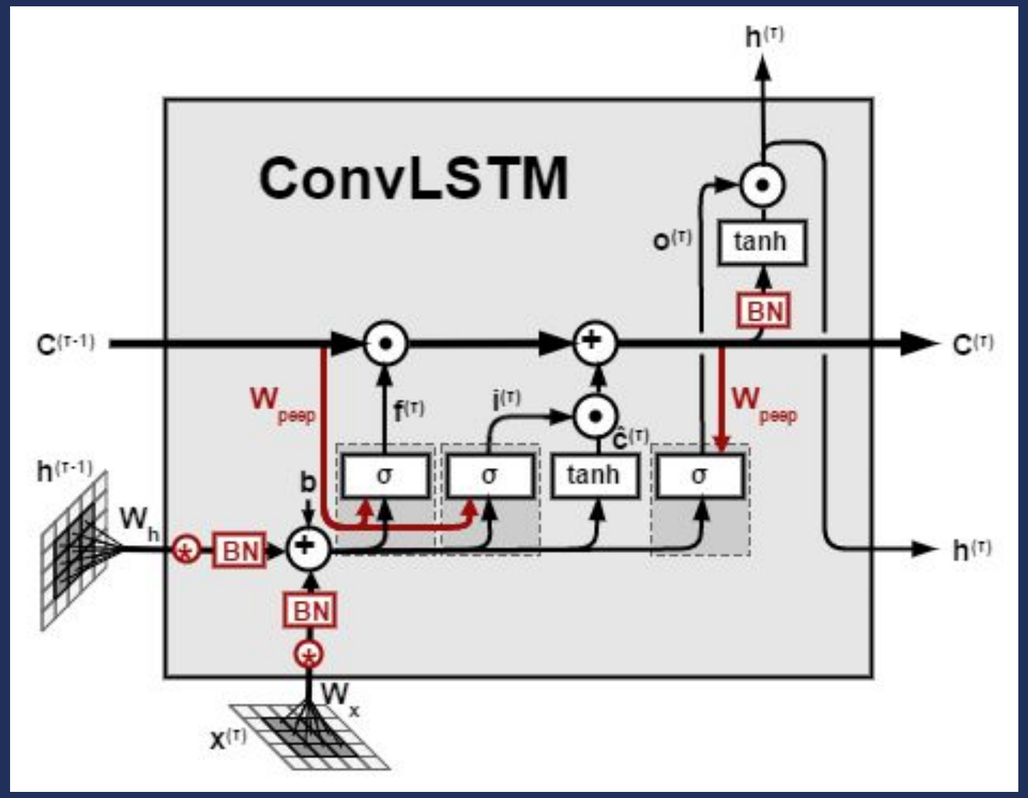
Difficulties forecasting

Anyone who has tried to plan vacations around weather forecasts knows the difficulty of forecasting storms accurately. This difficulty arises from the chaotic, innate dynamics of low-pressure weather systems:

- The general evolution of storm systems is described by fluid dynamics accounting for turbulence and rotation. This is notoriously difficult physics - even the case without rotation or turbulence is an open question.
- But we don't need to predict the evolution of the entire atmosphere, we just need a rough estimate that's approximately right one the order of weeks in the future. Unfortunately, even this is a difficult task due to the massive number of factors that can impact weather.

To deal with these issues, a more exotic solution was required for the forecasting model.

Forecasting model



A novel machine learning model was developed for forecasting - a combination of a traditional convolutional neural network and an LSTM, called a ConvLSTM, modified significantly to accommodate storm input data.

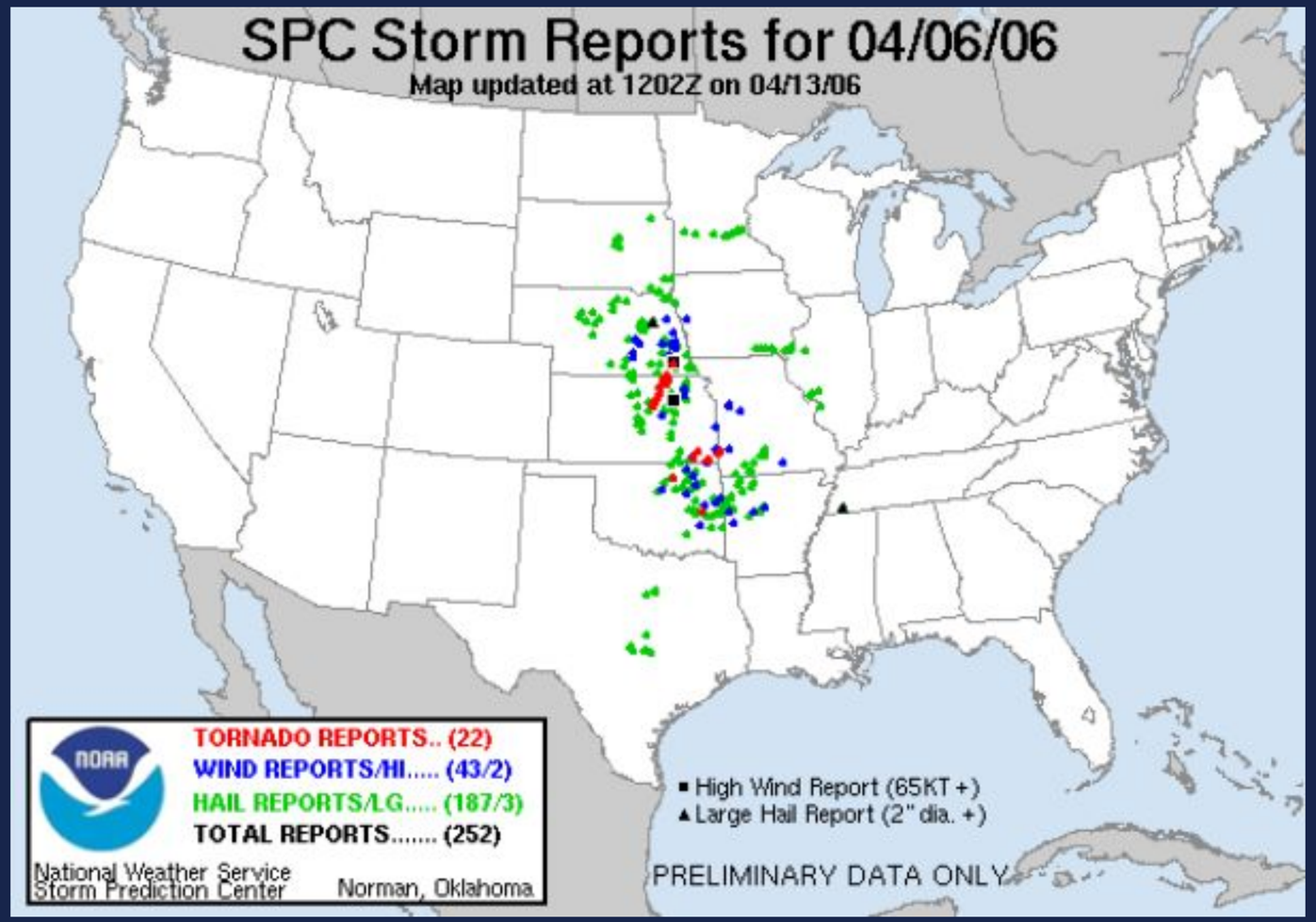
The model is designed in a complex fashion to capture complex spatiotemporaral interactions between the input data.

Data preparation

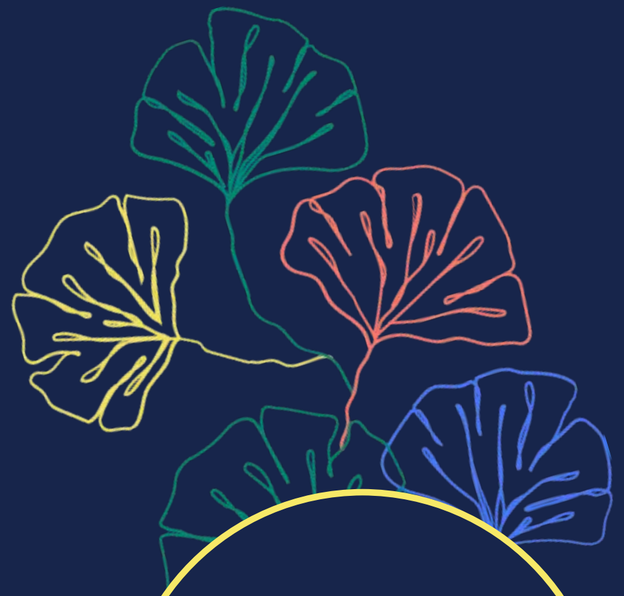
The data for the forecasting model is taken from NOAA historical storm reports, in the form of CSV files of coordinates. The database provides historical reports of severe tornado, wind, and hail events in the continental US, shown on the right.

Regions of input were then generated by constructing and filling alpha-shapes around the storm coordinates in a recursive manner (more on this later).

The full data preparation process is shown in the following slides.



Data Processing (1)



Step 1

CSV Extraction

Through DownThemAll!

Step 2

Reading

In data into a dictionary

Step 3

Alpha-shape Computation

Finds a concave hull to represent the data

Step 4

Uniform Filling

Create a uniform distribution of points in a bounding box

Data Processing (2)

Step 5

Contain filter

Through DownThemAll!

Step 6

Populate BBox

In data into a dictionary

Step 7

Rounding

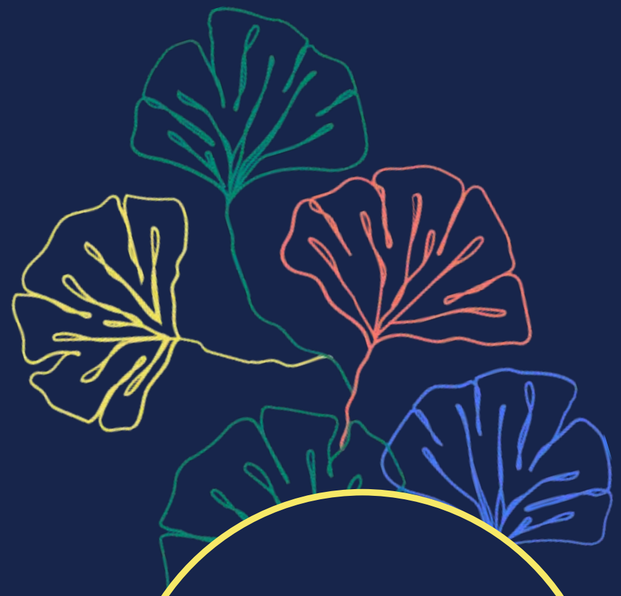
Round all contained with precision sigfig=1

Step 8

Indexing

Convert point coordinates into array indices and populate them with one or true

Data Processing (3)



Step 9

Parsing dates

Converting datetimes into a deltatime into a numeric value of days

Step 10

Regressing data

Precomputing previous data for initial conditions

Step 11

Splitting data

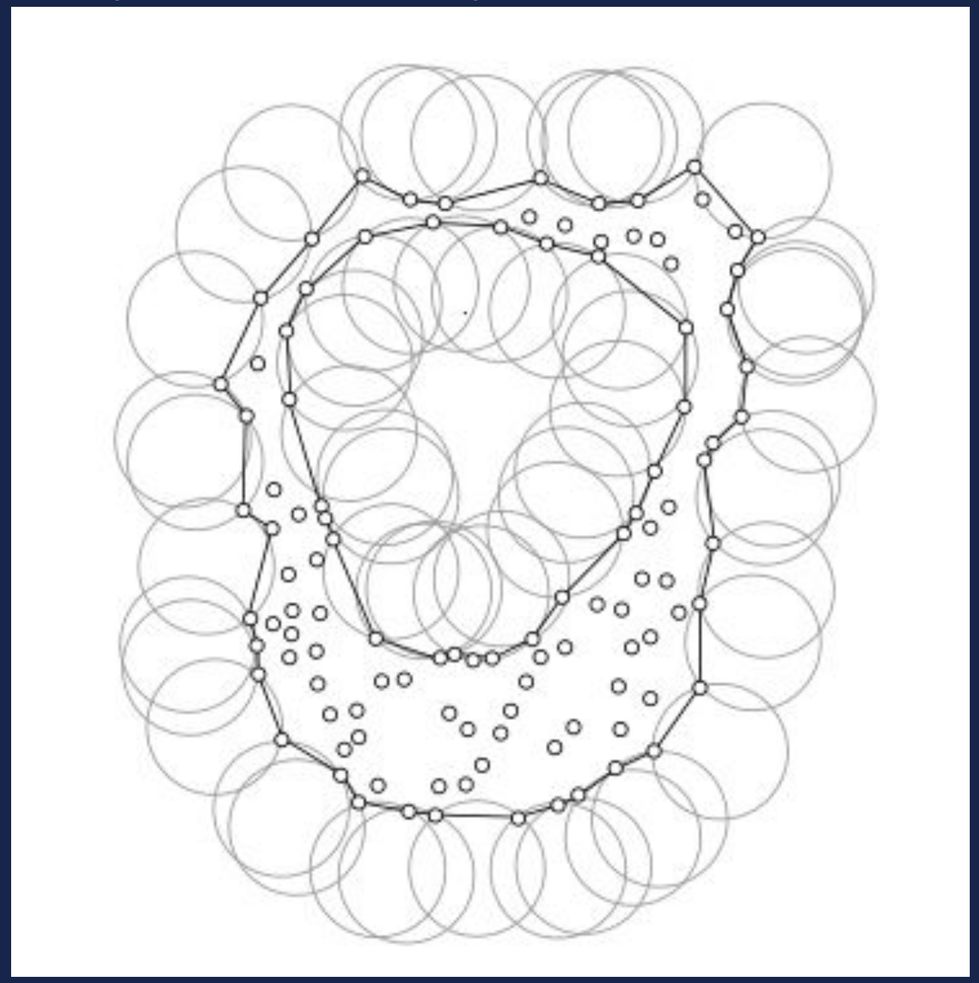
Into training and testing via sklearn's train_test_split

Step 12

Dumping

Pickling data into a tuple of information for later use

Alpha shape construction

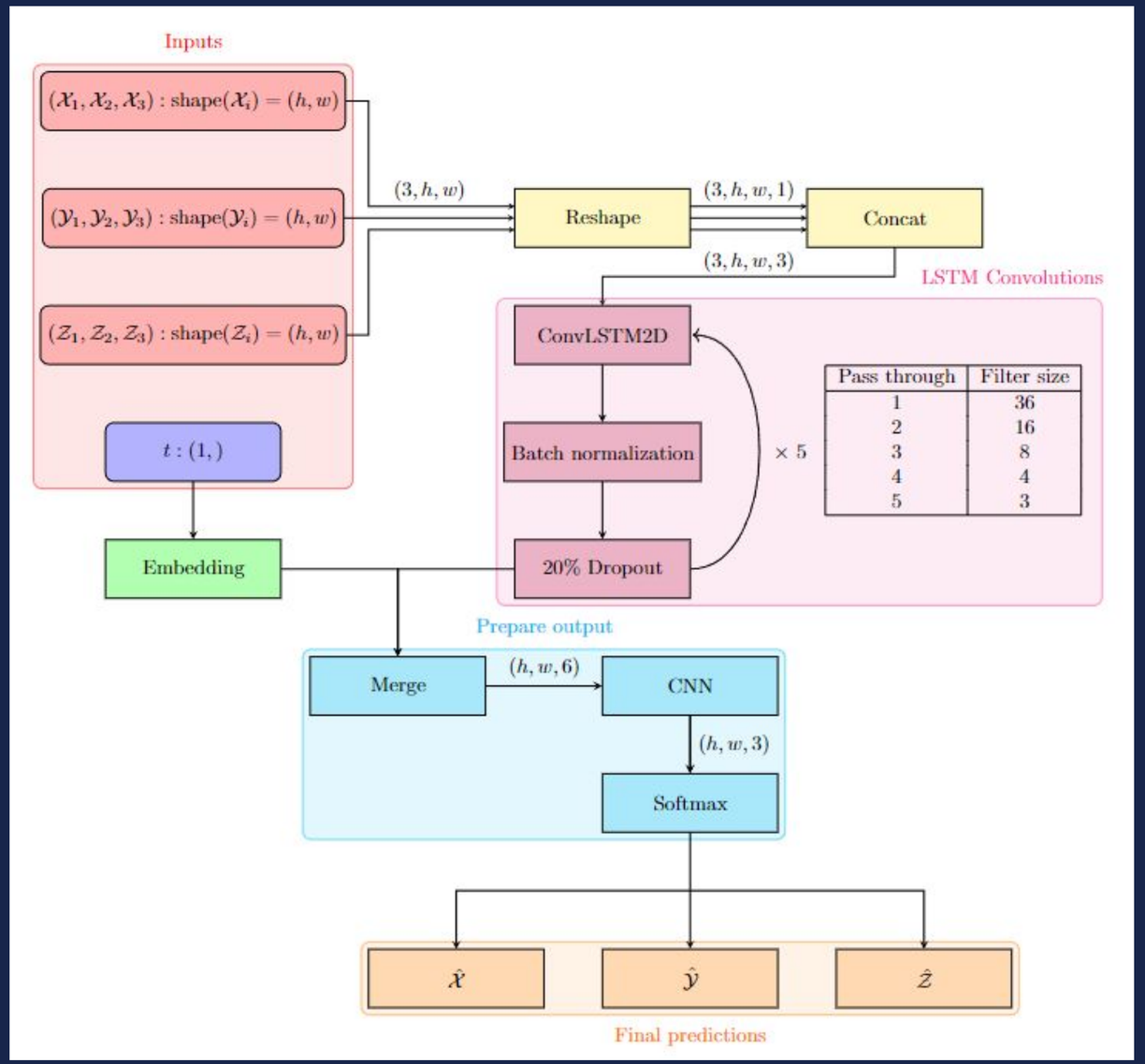


- Alpha-shapes are bounding polygons that enclose coordinates
- An alpha-shape is constructed by Delaunay triangulating the data, finding the local circumcircle of each vertex, and filtering the circumradii such that they are no more than 1/α
- Alpha-shapes were chosen for their customizability and as a middle-ground between convex hulls and MSTs
- To accommodate for degenerate data cases (some days have many storms, some have none) a novel, recursive method was written that recursively decrements the parameter α to produce a descriptive alpha-shape regardless of data density

Forecasting model - architecture

After all is said and done, we can now feed the processed and filled alpha-shape data into a ConvLSTM model as input. The full model architecture is shown on the right.

- Note that the spatial data is fed through several convolutional LSTM layers and then merged with the original datetime data. This allows for the preservation of spatiotemporal relationships
- The model takes the last three days of tornado, wind, and hail data to produce predicted regions of tornado, wind, and hail



Forecasting model - results (1)

The forecasting model has been successfully run and early metrics indicate it is quite accurate.

To the right are the loss/accuracy tables for the training and validation data over five epochs. The model stabilizes at around 70% validation accuracy for tornado and wind, but only manages 0% for hail.

Epoch #	Torn loss	Hail loss	Wind loss	Torn acc	Hail acc	Wind acc
1	5.8032	0.7184	9.6077	0.0145	0.0023	0.2569
2	5.1505	0.7325	8.9438	0.0172	$2.4528 \cdot 10^{-5}$	0.3877
3	5.0135	0.7307	8.8987	0.0232	$2.3785 \cdot 10^{-5}$	0.5717
4	4.9628	0.7298	8.8573	0.0837	$2.5271 \cdot 10^{-5}$	0.5815
5	4.9237	0.7292	8.8176	0.3672	$1.7095 \cdot 10^{-5}$	0.6742

Table 1: **Training**

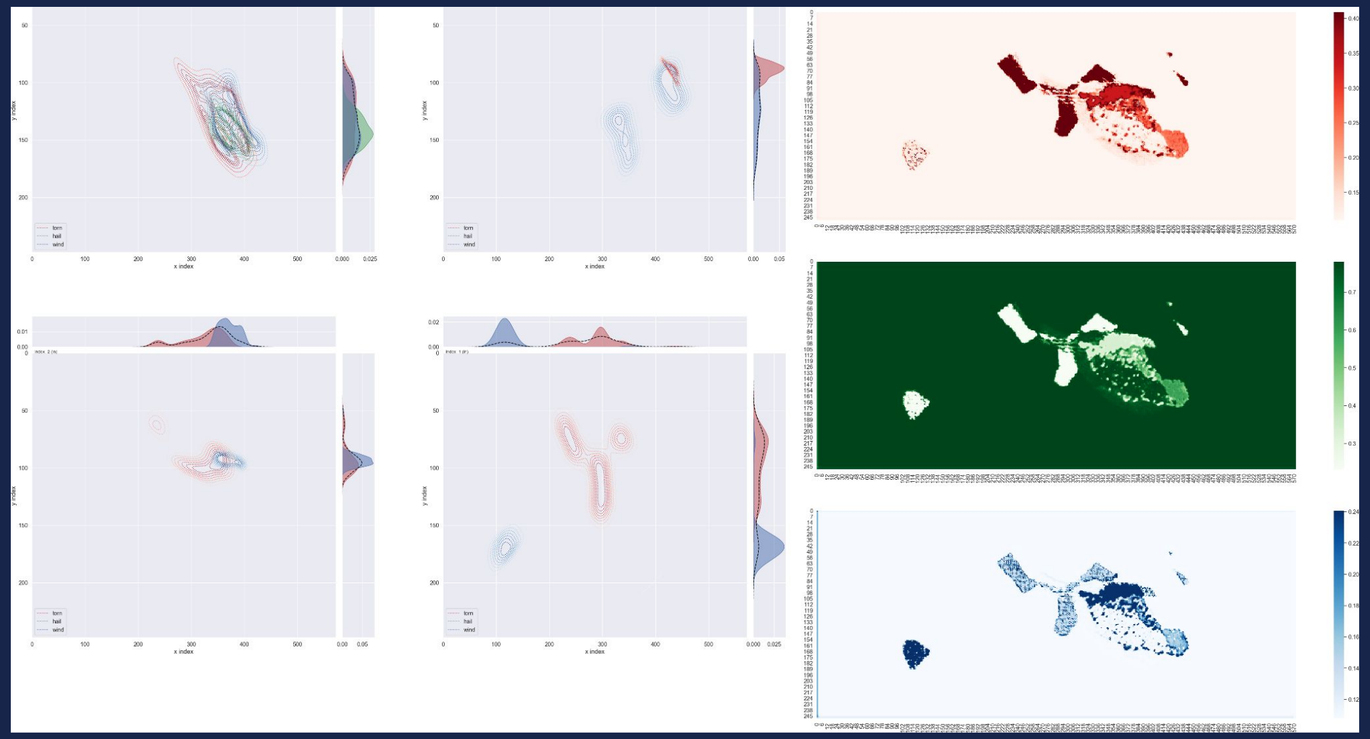
Epoch #	Torn loss	Hail loss	Wind loss	Torn acc	Hail acc	Wind acc
1	6.5738	0.6747	11.6576	$2.2999 \cdot 10^{-4}$	$4.1816 \cdot 10^{-5}$	0.7960
2	6.2969	0.6785	10.9007	0.0015	$3.2855 \cdot 10^{-5}$	0.7217
3	5.2992	0.6948	9.4190	0.0098	$1.4934 \cdot 10^{-5}$	0.6887
4	5.2027	0.7359	9.3074	0.6740	$1.7095 \cdot 10^{-5}$	0.7142
5	5.1819	0.7505	9.2296	0.6950	0	0.7257

Table 2: **Validation**

Forecasting model - results (2)

Here is a sample of predicted regions.

- On the far right are three predicted regions for tornado, hail, and wind. To the left are four pictures - the top left, top right, and bottom left are the previous three days of data, and the bottom right is the actual data.
- All three predicted regions strangely have the same shape, with cycling confidence levels. This may be because of the merging that was done in the model.
- Despite this spatial redundancy, the tornado and wind predictions line up quite well with the actual data. We can now see that the hail suffers because the entire background region gets cycled to high confidence.



Summary of achievements

1. Models were constructed to detect and forecast severe storm events. Both used elements of machine learning and/or statistical inference to establish spatiotemporal patterns.
2. Novel techniques to process weather data (separable CNNs and alpha shape processing) were developed.
3. Novel model architectures for storm detection/forecasting (modified ConvLSTM) were developed.
4. The forecasting model was successfully run and despite spatial redundancy was shown to achieve surprisingly high accuracy.

To do:

1. Eliminate spatial redundancy in forecasting model.
   a. Could be done by switching to a tripartite model that does not merge the input data
2. Successfully run detection model.
3. Adjust hyperparameters (batch size, filter number, etc.) to maximize forecasting and detection accuracy.
4. Improve existing documentation to facilitate usability and adaptability for other future users.
5. Incorporate other datasets like precipitation and wind velocity to forecasting model to further bolster accuracy.

Conclusion

Detecting and forecasting severe storm events is a difficult but highly important task. Hundreds of Americans are killed by severe storms every year and thousands more are displaced or injured by them. With continued research into the field of storm prediction, we hope to get closer to solving this task.

Thank you!