Aircraft Image Classification Project

Presented by Hao, Steve, and Jeff

Problem

- Inspection takes resources.
- Inspections are tedious and defects can be missed.

Contents

- Proposed solution
- Convolutional Neural Network
- Data Experimentation
- Applications
- Demonstration
- Conclusions

1 Proposed solution

Proposed solution

- Spexi Geospatial provided drone images
- We propose image classification using a convolutional neural network (CNN).
- Images need preprocessing
- Label output for defect vs non-defect?

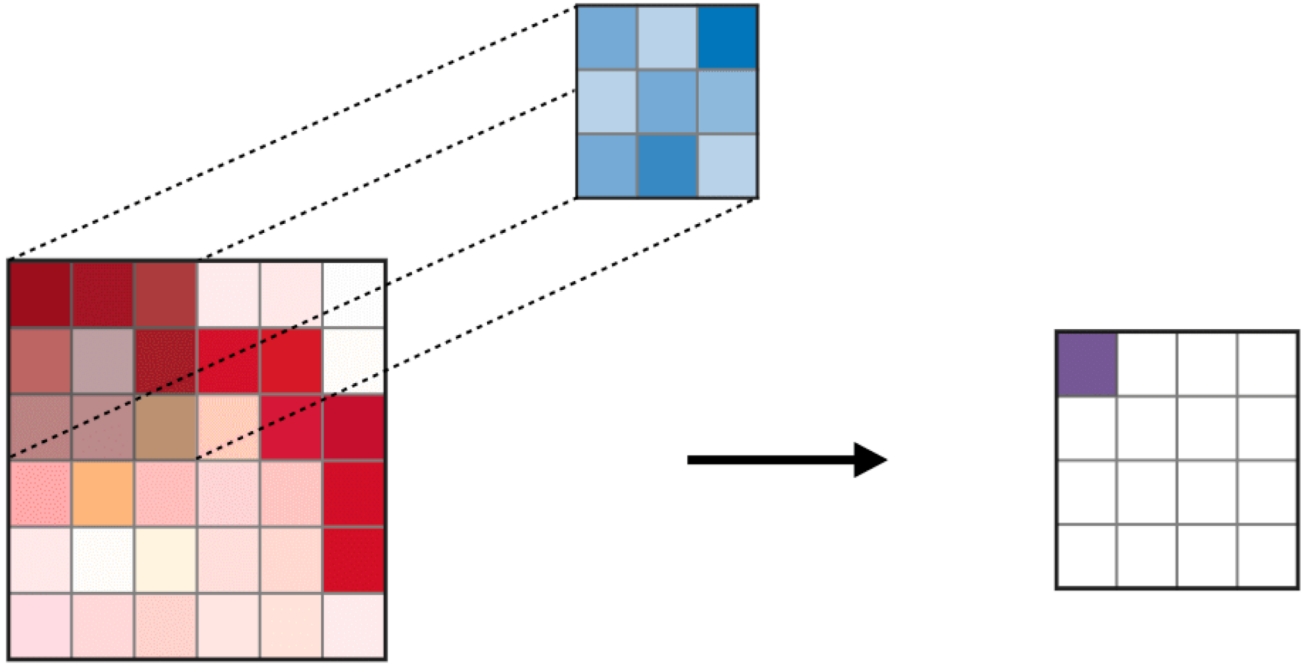
2

Convolutional Neural Network

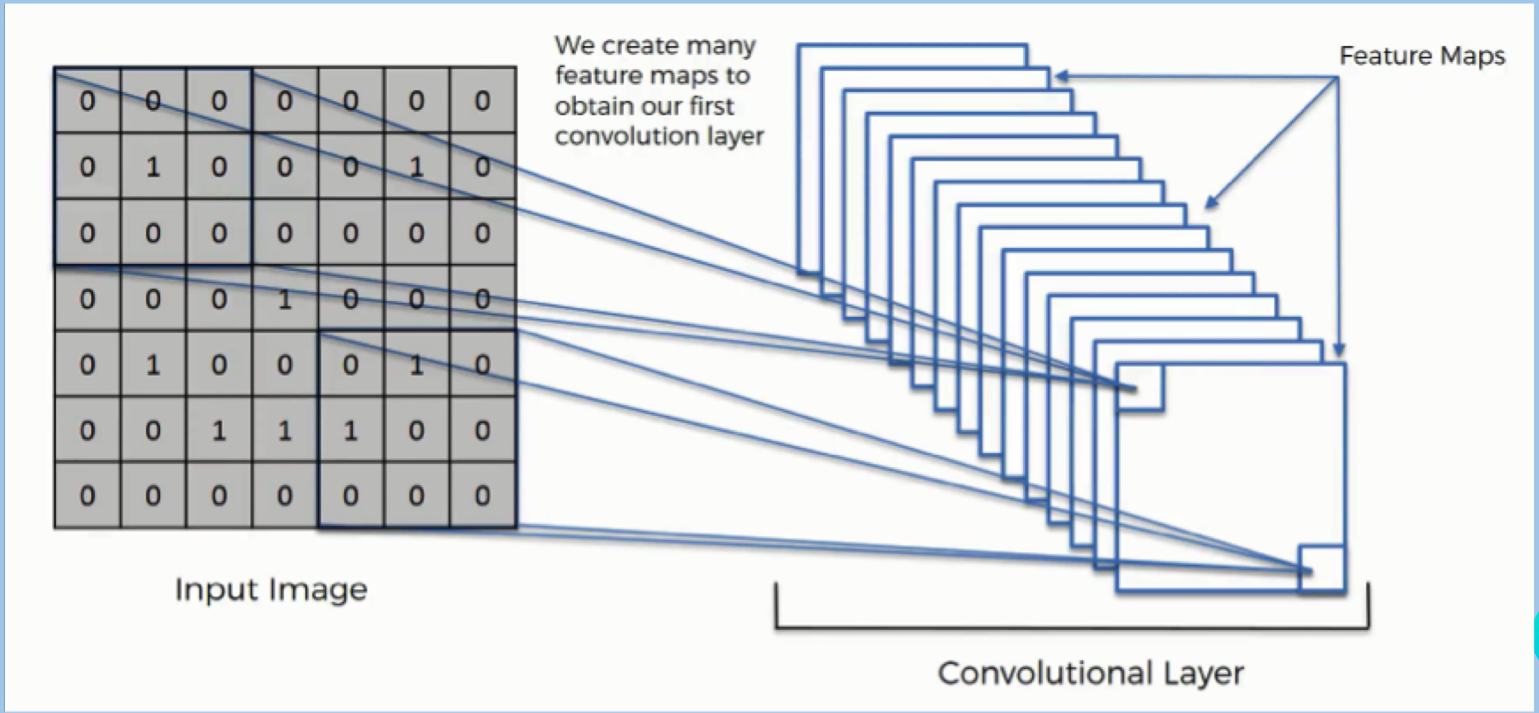
Convolutional Neural Network

- A CNN involves multilayer processing to identify images.
- Input: image, output: prediction
- Key features:
  - Convolutional layer - detection of features.
  - Softmax layer - mapping of previous input to probabilities

Convolutional Layer



Convolutional Layer



Softmax Layer

Input pixels, $\mathbf{x}$

Shape: (3, 32, 32)

Forward propagation

Feedforward output, $\mathbf{y}_i$

cat	dog	horse
5	4	2
4	2	8
4	4	1

Shape: (3,)

Softmax function

Softmax output, $\mathbf{S}(\mathbf{y}_i)$

cat	dog	horse
0.71	0.26	0.04
0.02	0.00	0.98
0.49	0.49	0.02

Shape: (3,)

3 Data Experimentation

Initial Data

- Drone images and spreadsheet labelling defects
- Binary classification for cracks

108	DJI_0107	Wing - Right Tip	cracks
109	DJI_0108	Wing - Right Tip	cracks
110	DJI_0109	Wing - Right Tip	cracks
111	DJI_0110	Wing - Right Tip	cracks
112	DJI_0111	Wing - Right Tip	cracks
113	DJI_0112	Wing - Right Tip	cracks
114	DJI_0113	Wing - Right Tip	cracks
115	DJI_0114	Wing - Right Tip	cracks
116	DJI_0115	Wing - Right Tip	cracks
117	DJI_0116	Wing - Right Tip	cracks
118	DJI_0117	Wing - Right Tip	cracks
119	DJI_0118	Wing - Right Tip	cracks
120	DJI_0119	Wing - Right Tip	cracks
121	DJI_0120	Wing - Right Tip	cracks

Early Identifiable Problem

- Does not show where the crack is
- Too many features in the background that could introduce noise into the model.
- **Solution**: Crop out only the airplane

Experiment 1

- Image with background removed.
- 800 x 800 pixels, downscaled resolution.
- Grayscale to remove color bias.
- Images labeled as containing cracks or no cracks.
- Training to test ratio: 1:1

Experiment 1 Input

Experiment 1 Results

- Avg. prediction for ‘cracked’ images:

  0.5 (should be 1)
- Avg. prediction for ‘non-cracked’ images:

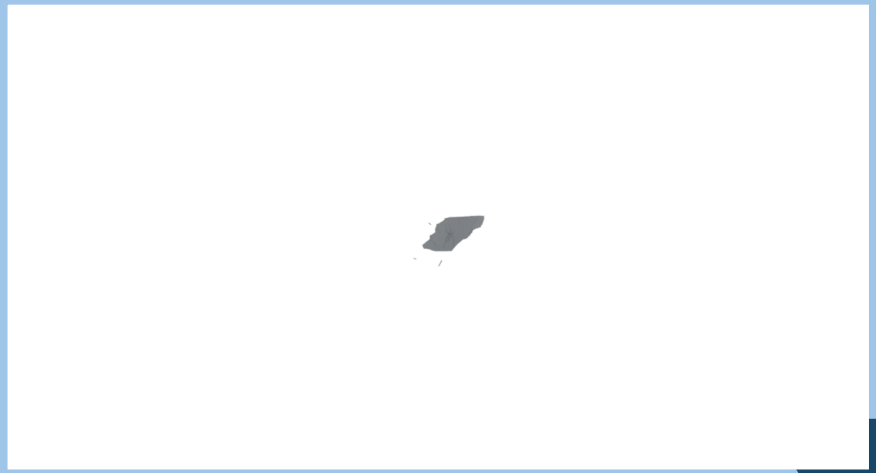
  0.19 (should be 0)

Is it really detecting cracks?

- Attempted a separate trial with the same model using:
  - Wing with crack as training
  - Same wing with crack removed as test
  - Crack from the wing as test

Is it really detecting cracks?

Image without crack

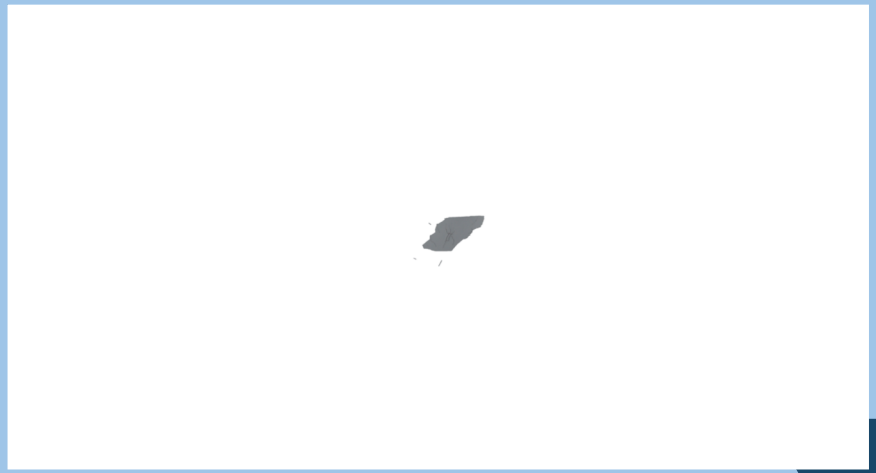


Crack by itself

Is it really detecting cracks?

Image without crack

Result: Crack



Crack by itself

Crack

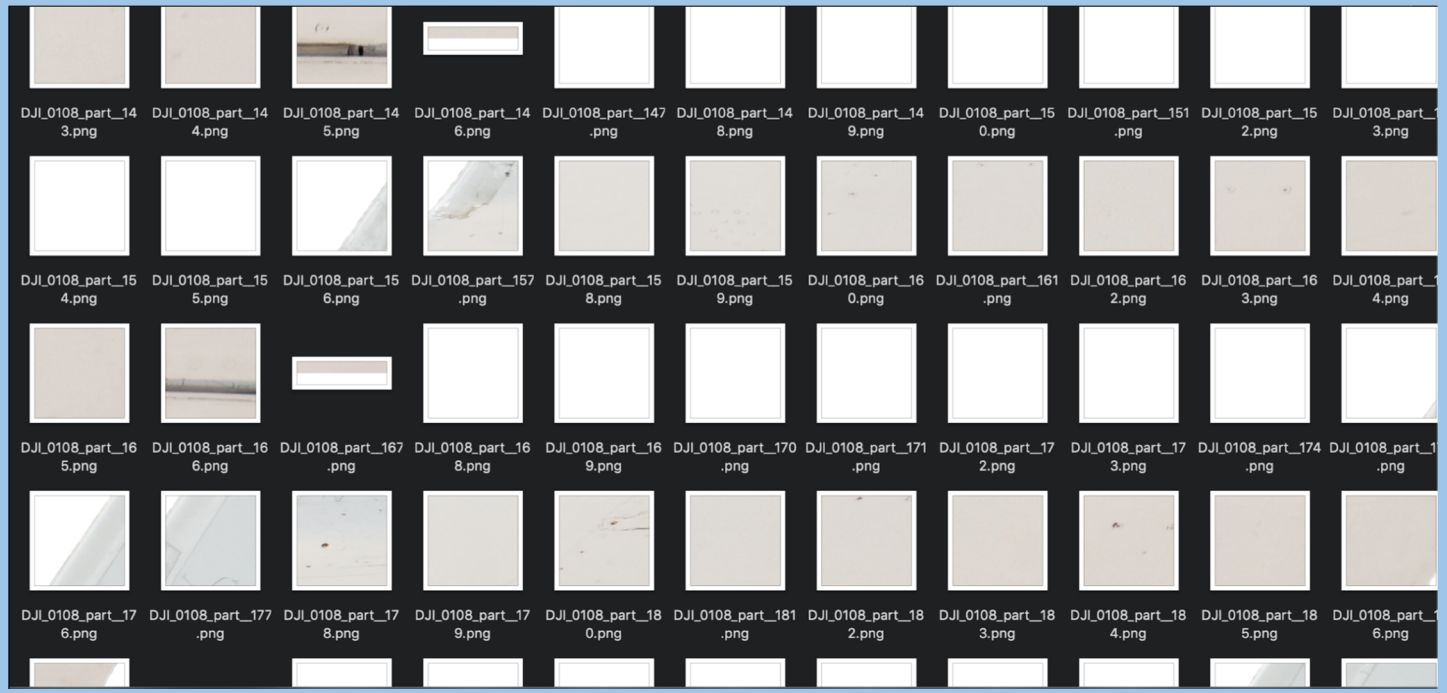
Experiment 1 Conclusion

- We hypothesized compression of image led to loss of detail.
- Model to looks at plane shape as opposed to features of a crack.
- Combat this by taking image subdivisions at 180 x 180 pixel.

Experiment 2

- 120 subimages constructed from original images
- 180 x 180 pixels, no compression
- Gray scale
- Labeled crack, non-crack
- Training to test ratio: 3:1

Experiment 2 Input



Experiment 2 Results

- Avg. prediction for ‘cracked’ images:
  0.68 (1)
- Avg. prediction for ‘non-cracked’ images:
  0.65 (0)
- All images were being identified as cracked.
- Difference in prediction: 0.03

Experiment 2 Conclusion

- Was the number of images a problem?
- Proposed increasing sample size for training, test, measuring new results.

Experiment 3

- **1900** subimages constructed from original images
- 180 x 180 pixels, original resolution
- Gray scale
- Labeled crack, non-crack
- Training to test ratio: 3:1

Experiment 3 Results

- Avg. prediction for ‘crack’ images:
  0.09 (1)
- Avg. prediction for ‘non-crack’ images:
  0.02 (0)
- All images are identified as ‘non-crack’
- Difference in prediction: 0.07

Experiment 3 Conclusion

- Because “non-cracked” was too vague, all images were being identified as non-cracked.
- For binary classification, the two categories should be concrete and distinct.

Experiment 🐱

- 180 x 180 pixels, original resolution
- Gray scale
- Labeled crack, **cat**
- Training to test ratio: 3:1

Experiment 🐱 Results

- Avg. prediction for ‘cracked’ images:
  0.49 (1)
- Avg. prediction for ‘cat’ images:
  0.05 (0)
- All images were being identified as cat.
- Difference in prediction: 0.44

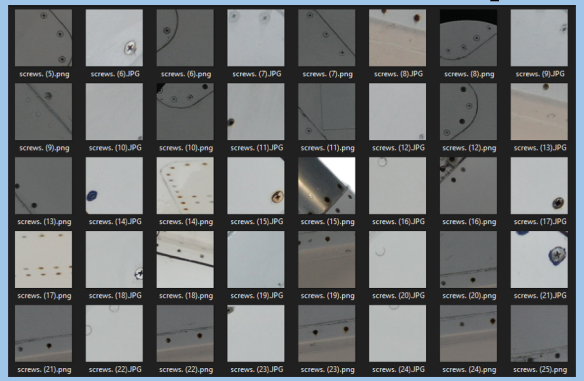
Experiment 🐱 Conclusion

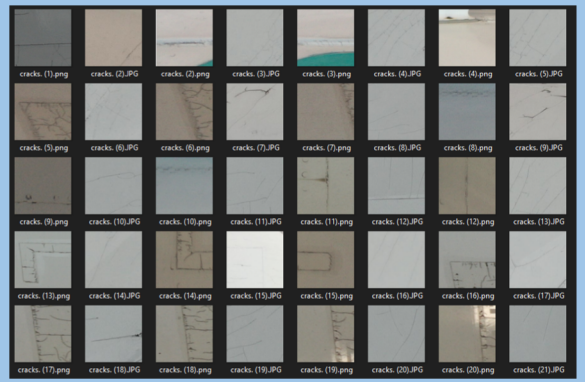
- The difference in complexity between crack and cat is too dramatic.

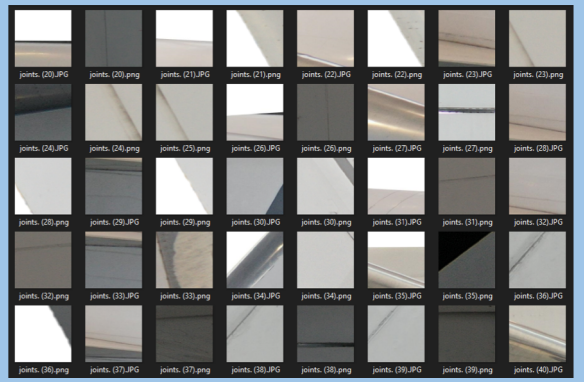
Experiment 5

- **900** subimages cropped from original images (300 per feature).
- 180 x 180 pixels, original resolution, gray scale
- Labeled as:
  - Crack - jagged line.
  - Joint - straight line.
  - Screw - round object.
- Training to test ratio: 9:1

Experiment 5 Input







Experiment 5

- Categorical instead of binary
  - Example output: [0.12, 0.80, 0.08]

Experiment 5 Results

- Manual testing with 19 (180x180) images
- Accuracy 42.1%
- Most likely to be mistaken as ‘joint’

Experiment 5 Conclusion

- The feature ‘joint’ (Straight lines) is too simple
- It’s time to end this
- Model refinement

Model Adjustment

- To improve model performance, we also increased number of epochs from 10 to 100.
- This gives the model more ‘generations’ to learn over, producing a more accurate model.
- Tradeoff is computation time.
- Used different prediction thresholds for different features.

4 Applications

Applications

- The completed model allows us to identify if a subimage contains a crack, joint, or screw.
- Simple:
  - Feed in subimage, get prediction
- More complex:
  - Feed in full image, get image “heatmap”
  - Each 180x180 subregion has been identified as crack, joint, screw.

5 Demonstration

6 Conclusion

Results

- Model was influenced by background, level of detail, quality of features.
- Our best model - Experiment 5 gave results:
  - Able to extract accuracy = 42.1%

Insights Gained

- Data is very important!
  - Need enough data
  - Data should be correctly labeled
- Labels should be intelligently selected
  - Labels should be distinct
  - Labels should be well defined
  - There should be enough images for each label

Is there a solution?

- Based on the data presented, and the nature of defects on aircraft, we do NOT believe there is a CNN solution to detecting aircraft defects.

Thanks!

Any questions?

References

- Convolutional Neural Networks [Digital Image]. Retrieved from https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks
- Convolutional Neural Networks [Digital Image]. Retrieved from https://www.freecodecamp.org/news/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050/
- Convolutional Neural Networks [Digital Image]. Retrieved from https://www.superdatascience.com/blogs/the-ultimate-guide-to-convolutional-neural-networks-cnn
- Softmax [Digital Image]. Retrieved from https://ljvmiranda921.github.io/notebook/2017/08/13/softmax-and-the-negative-log-likelihood/
- Rajat Garg. Kaggle “Dogs vs. Cats” Challenge — Complete Step by Step Guide — Part 2. Retrieved from https://medium.com/@mrgarg.rajat/kaggle-dogs-vs-cats-challenge-complete-step-by-step-guide-part-2-e9ee4967b9
- Jason Brownlee. Multi-Class Classification Tutorial with the Keras Deep Learning Library. Retrieved from https://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/