~~Gradcam~~ Explaining Transformer Models (From a Vision perspective)

Jason Chalom (711985)

Supervisor: Dr Richard Klein

25 October 2023

School of Computer Science and Applied Mathematics

How I got here?

- Im a MSc candidate looking at attribution methods for CNN models
  - Self-driving car problem using end-to-end techniques
- Vision transformers seem to be able to replace CNN models (state of the art)
- Can the same techniques be applied here?

Acknowledgements

- Devon's NLP class had a guest lecture with Aaditya Singh
  - Helped show how much I didn't know
  - Helped give names for techniques
- Im not fully read up on everything happening with ViT explainability, lots of new stuff

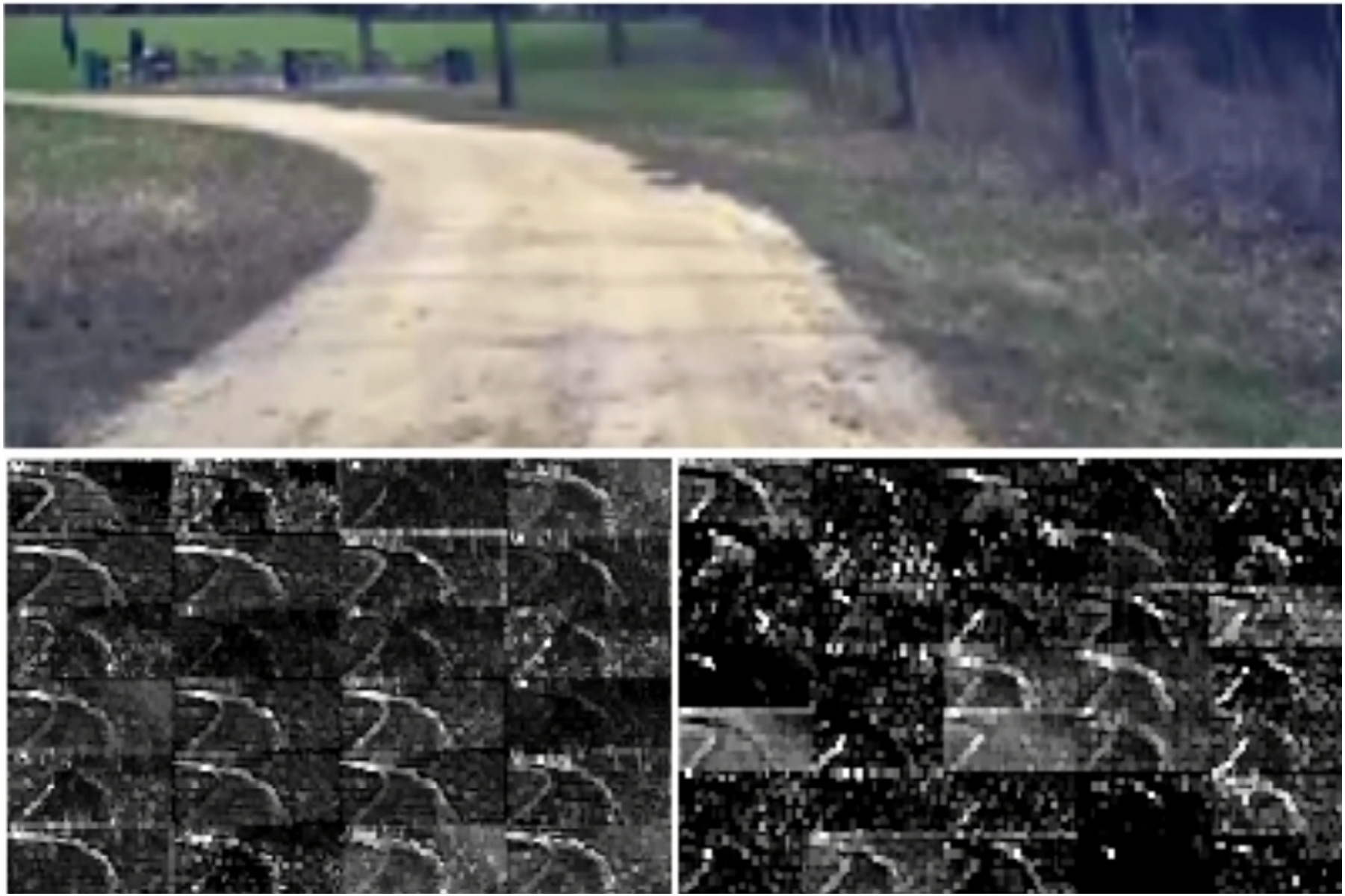
Background

”The CNN is able to learn meaningful road features from a very sparse training signal (steering alone).”

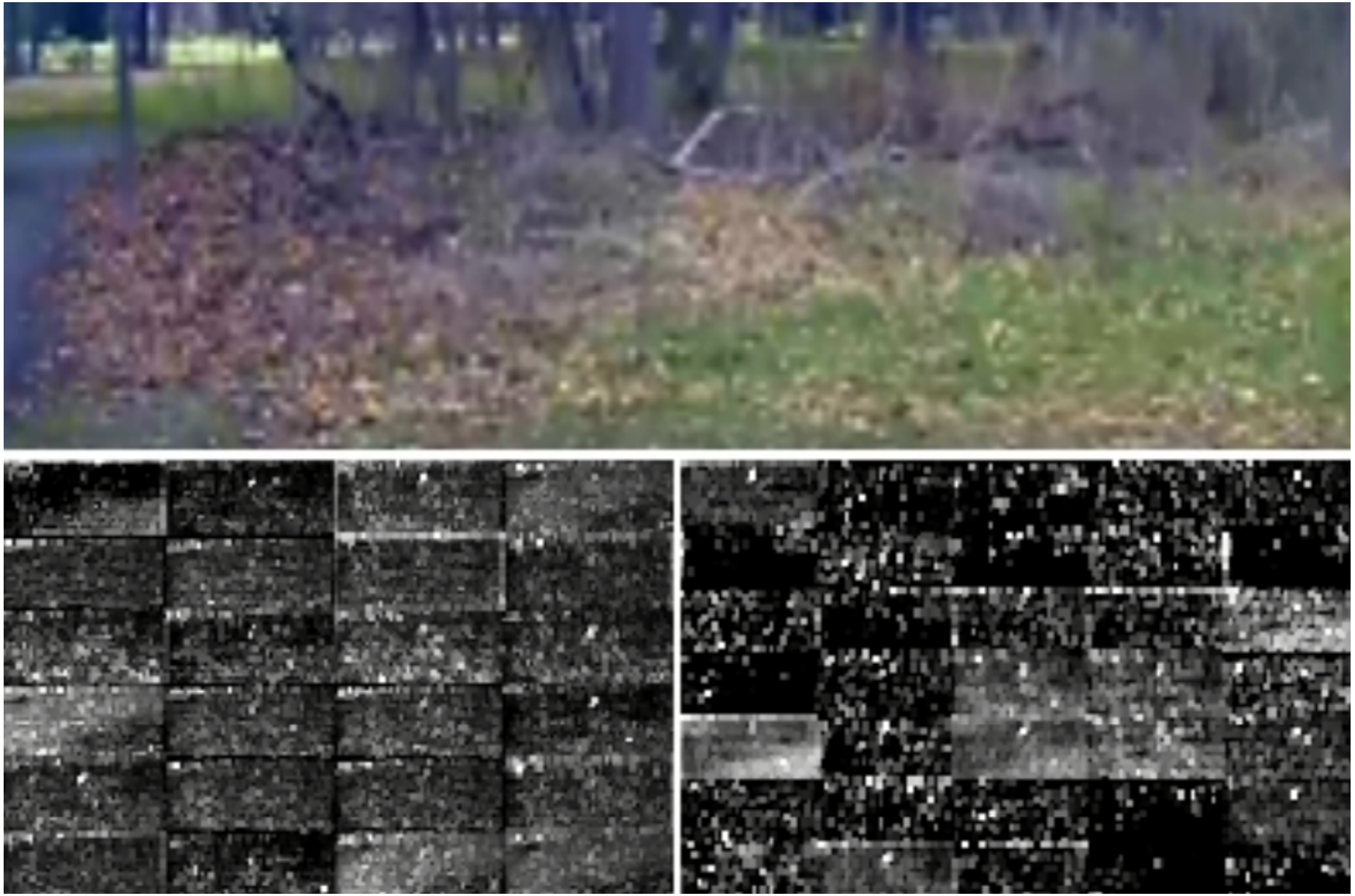
Bojarski et al. (2016)

The result of an end-to-end AI

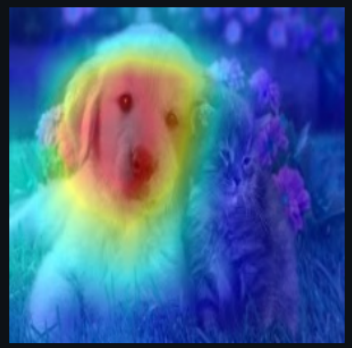
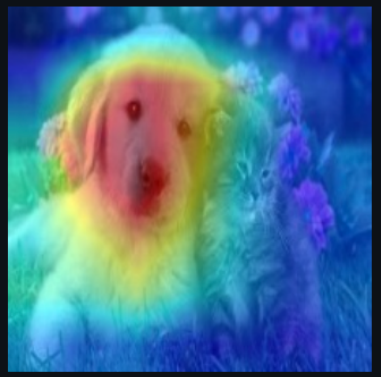
Example output of an end-to-end model (Spryn and Sharma 2018)



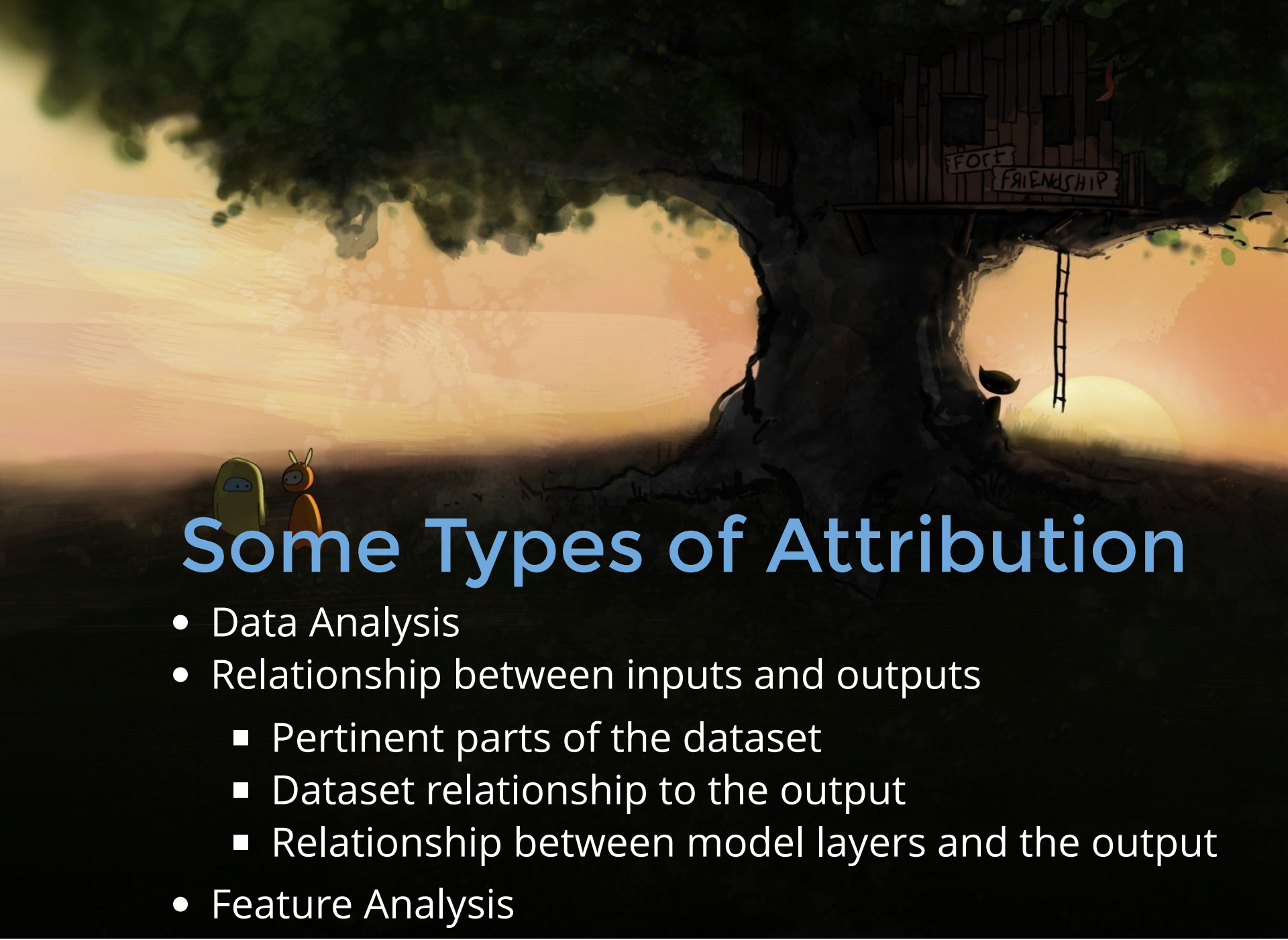
Unpaved Road



Forrest Scene

Category	Image	GradCAM	AblationCAM	ScoreCAM
Dog				
Cat				

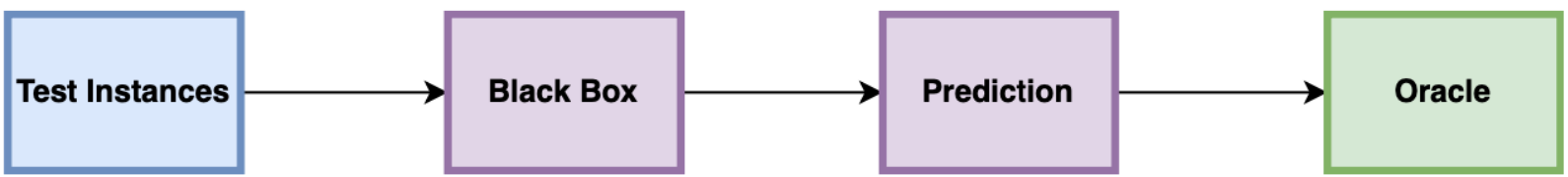
Attribution Techniques



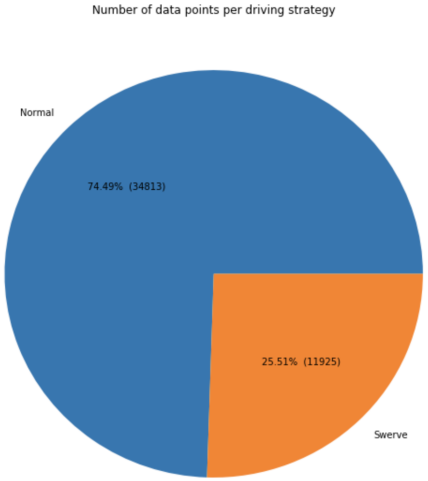
Some Types of Attribution

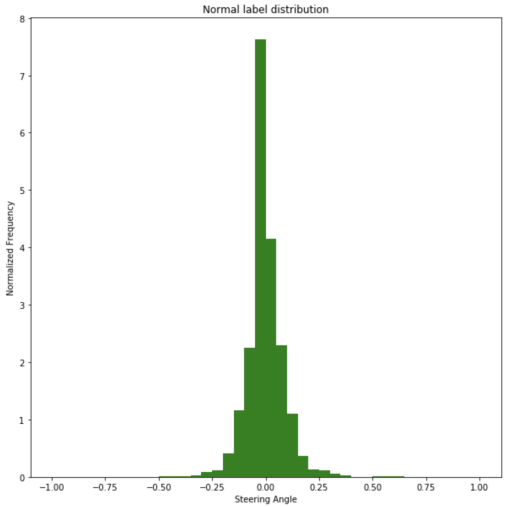
- Data Analysis
- Relationship between inputs and outputs
  - Pertinent parts of the dataset
  - Dataset relationship to the output
  - Relationship between model layers and the output
- Feature Analysis

Reverse engineering approach for explaining black-box models

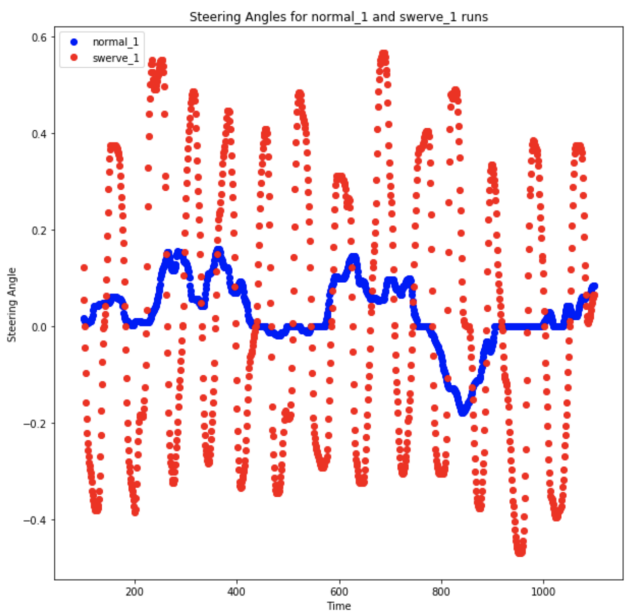


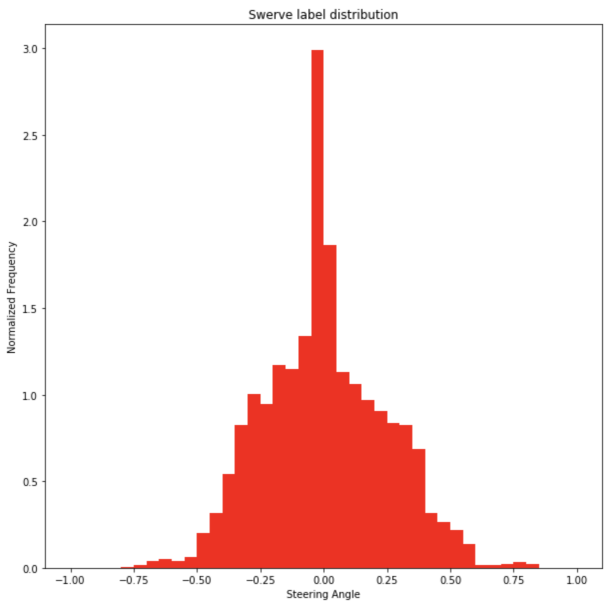
Guidotti et al. 2018





Data Analysis



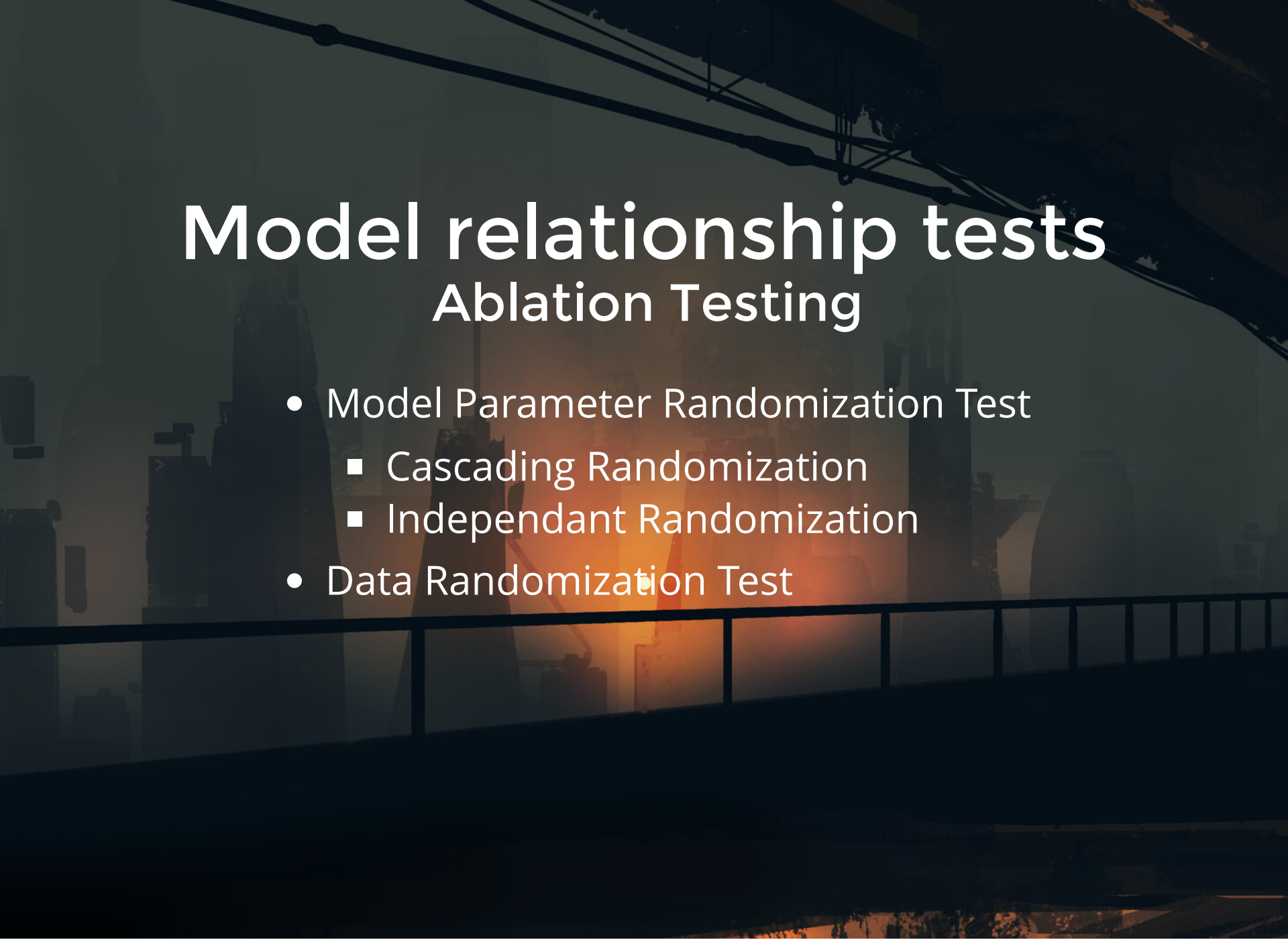


Attribution Methods

- Model relationship tests
- Perturbation Methods
- Gradient methods
- Other methods

Sanity Checks for Saliency Maps (Adebayo et al.)

- Many methods partially reconstruct the input data
- Brittle to noise and interference (misleading results)
- Many of the advanced guided methods dont have an adequate relationship between the input data and output nodes of a network
- Some methods (like some saliency maps) may not work with features that have a negative effect on the output

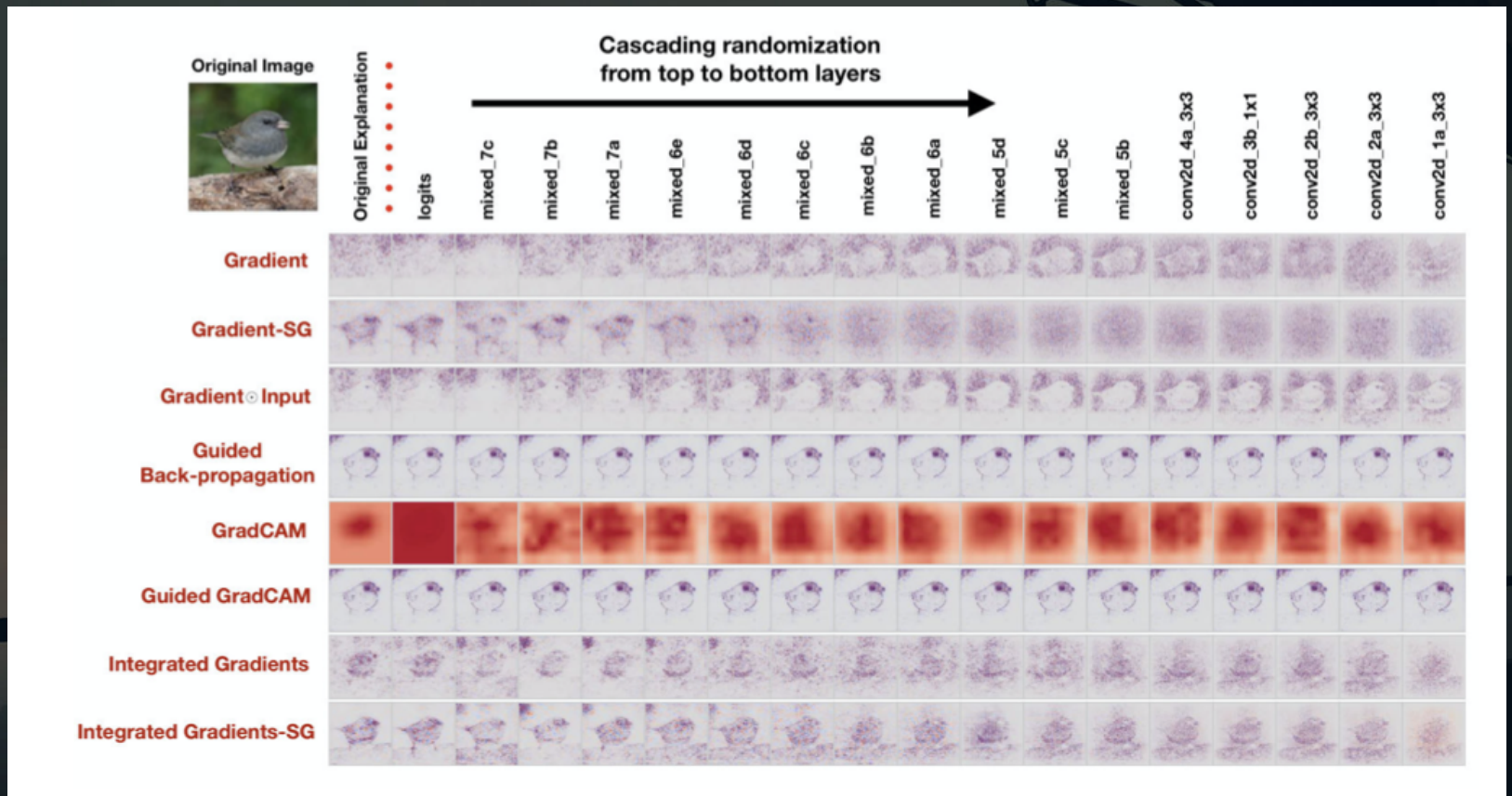


Model relationship tests
Ablation Testing

- Model Parameter Randomization Test
  - Cascading Randomization
  - Independant Randomization
- Data Randomization Test

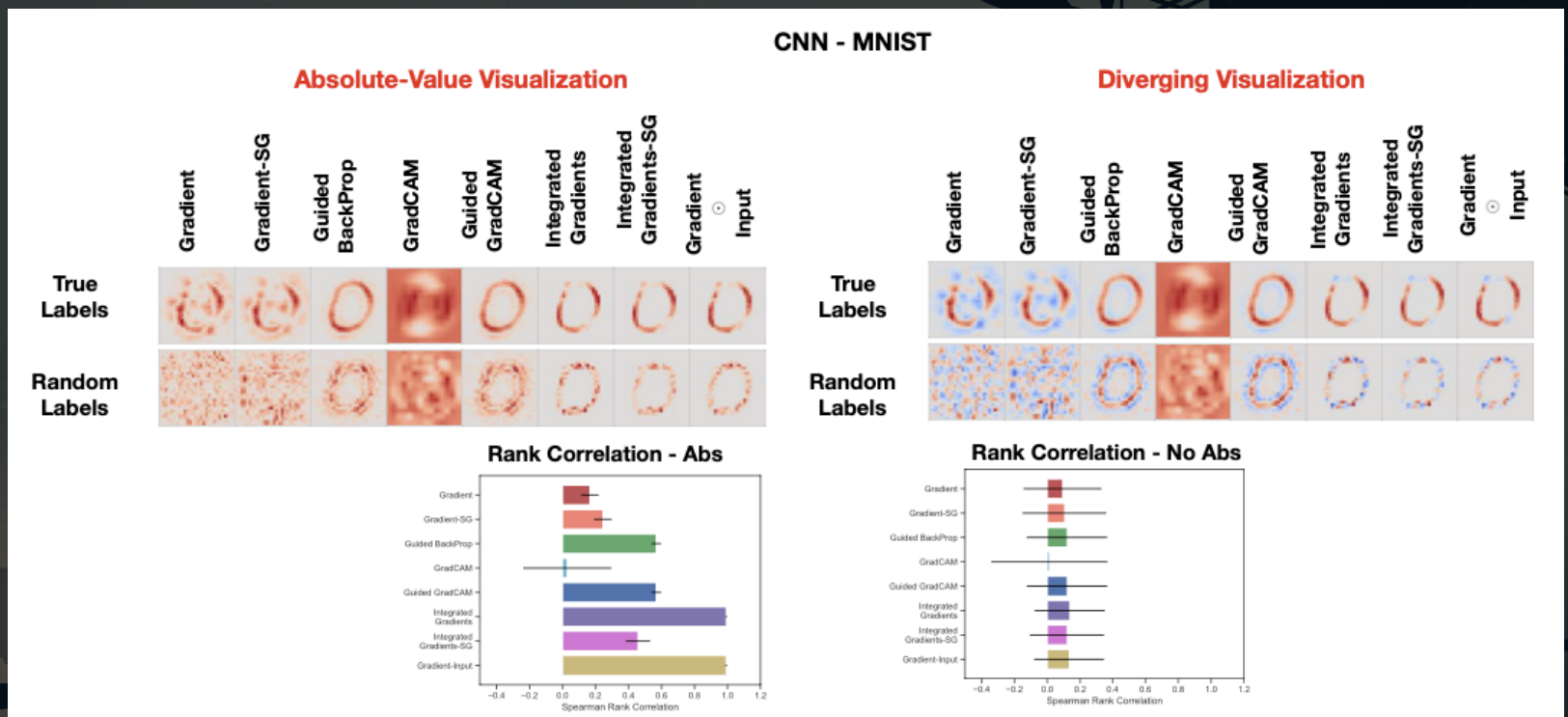
Cascading Randomization

- The weights of a model are randomized over time starting from the top layers and moving down to the bottom ones.
- This test shows the sensitivity of an attribution method to the model's parameters.


Original Image
Cascading randomization
from top to bottom layers
Original Explanation
logits
mixed_7c
mixed_7b
mixed_7a
mixed_6e
mixed_6d
mixed_6c
mixed_6b
mixed_6a
mixed_5d
mixed_5c
mixed_5b
conv2d_4a_3x3
conv2d_3b_1x1
conv2d_2b_3x3
conv2d_2a_3x3
conv2d_1a_3x3
Gradient
Gradient-SG
Gradient ⊙ Input
Guided
Back-propagation
GradCAM
Guided GradCAM
Integrated Gradients
Integrated Gradients-SG

Independant Randomization

- This is done by performing randomization layer-by-layer rather than by weight
- This gives a more granular indication of dependency for an attribution method by the order of the layer.


CNN - MNIST
Absolute-Value Visualization
Diverging Visualization
Gradient
Gradient-SG
Guided BackProp
GradCAM
Guided GradCAM
Integrated Gradients
Integrated Gradients-SG
Gradient ⊙ Input
True Labels
Random Labels
Rank Correlation - Abs
Rank Correlation - No Abs
Gradient
Gradient-SG
Guided BackProp
GradCAM
Guided GradCAM
Integrated Gradients
Integrated Gradients-SG
Gradient-Input
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Spearman Rank Correlation

Perturbation

- Type of function which compares two networks
  - The original network
  - A network trained on the dataset where relevant features have been altered
    - Masked
    - Obscured
    - Removed
    - Biased

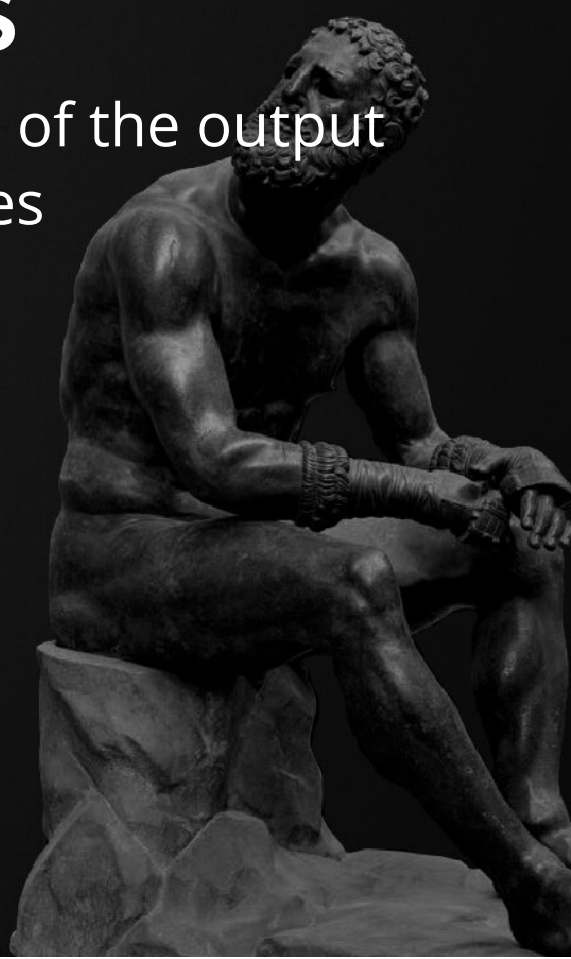
Gradient Methods

- Make use of the gradient based methods in the back-propagation step to attempt to extract spatial information captured in the network

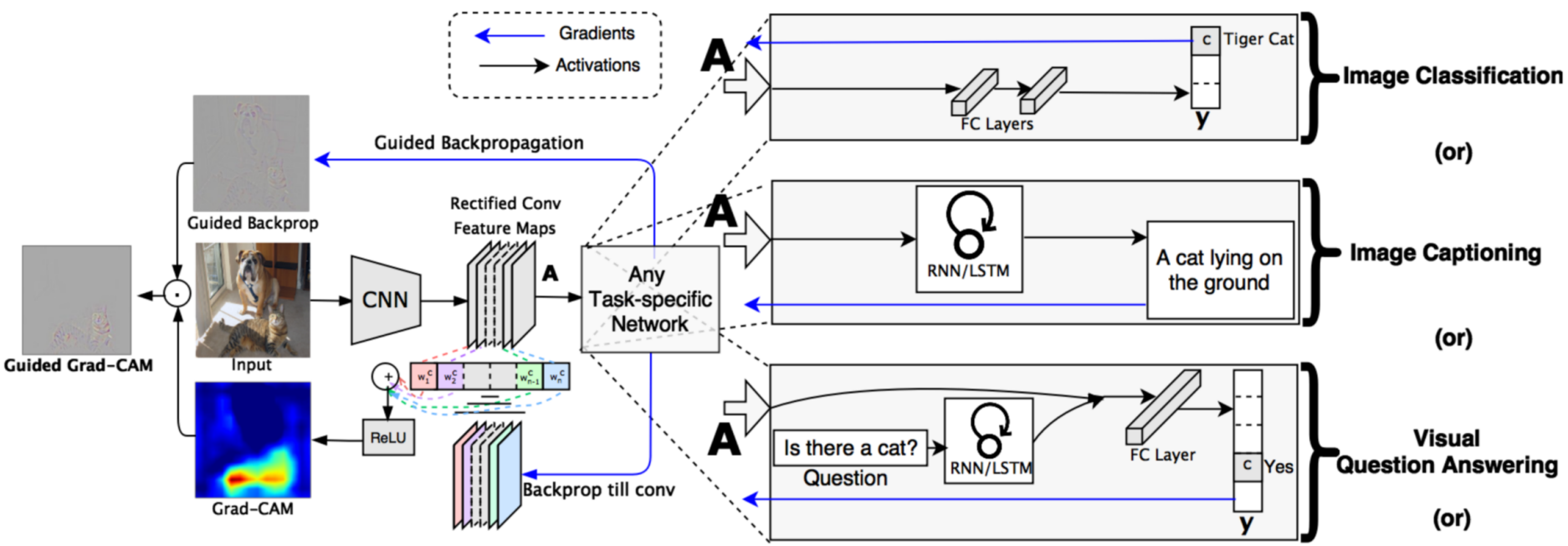
Saliency Maps

- Compute the absolute partial derivative of the output neuron with respect to the input features

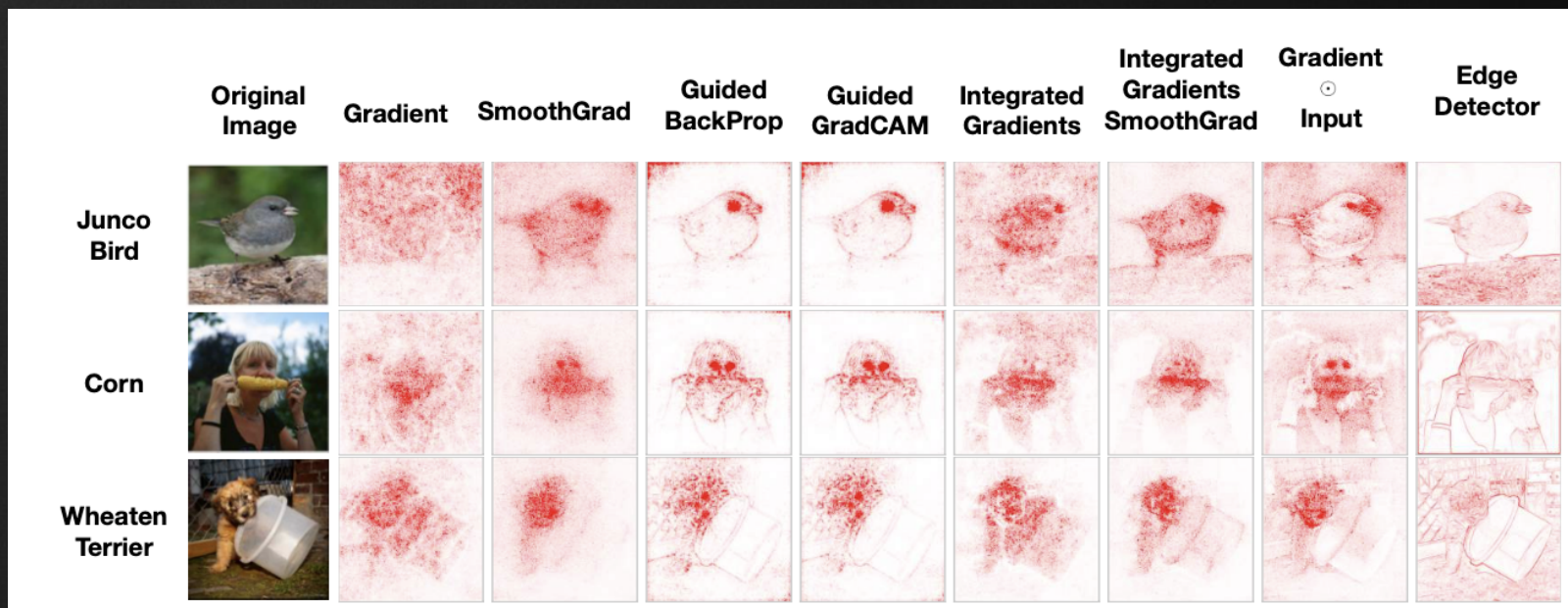
$$\frac{\partial output}{\partial input}$$



Gradient-weight Class Activation Mapping

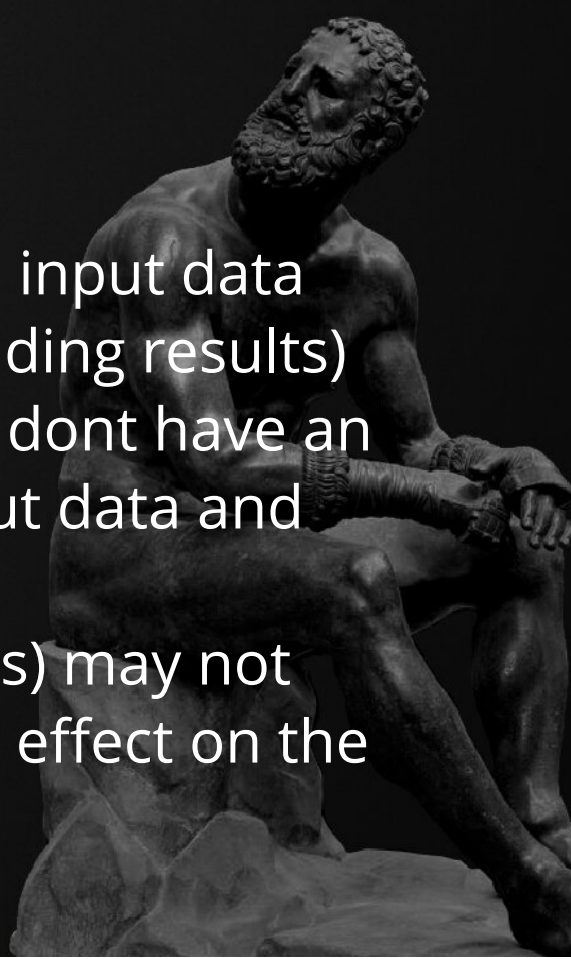


Selvaraju et al. (2016)


Original Image
Gradient
SmoothGrad
Guided BackProp
Guided GradCAM
Integrated Gradients
Integrated Gradients SmoothGrad
Gradient ⊙ Input
Edge Detector
Junco Bird
Corn
Wheaten Terrier

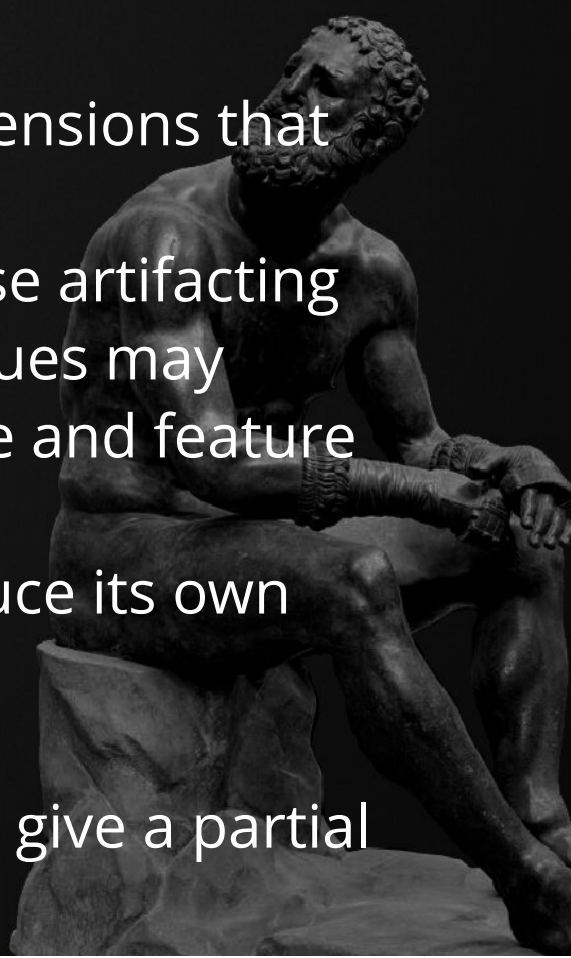
Issues with Gradient Based Methods

- Many methods partially reconstruct the input data
- Brittle to noise and interference (misleading results)
- Many of the advanced guided methods dont have an adequate relationship between the input data and output nodes of a network
- Some methods (like some saliency maps) may not work with features that have a negative effect on the output



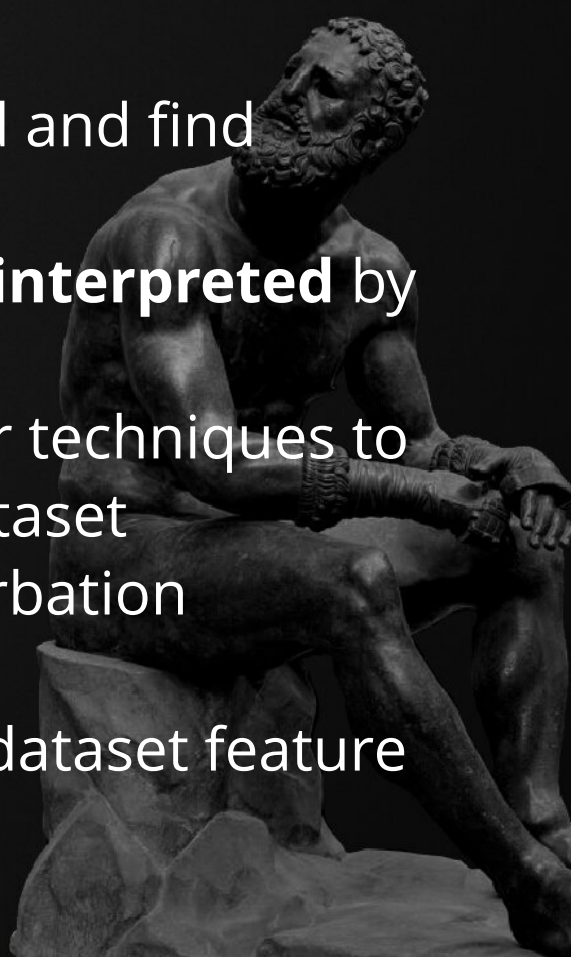
Issues with Attribution

- The outputs of networks may have dimensions that are much smaller than the inputs
- Noise becomes a factor which may cause artifacting
- The boundaries of visualisation techniques may become distorted due to network shape and feature extractor size
- The final step of attribution may introduce its own bias i.e. upscaling, resizing etc
- Computational cost
- Selection of samples analysed may only give a partial interpretation of the model



Evaluating Attribution

- Take the result of an attribution method and find **reasoning** from it
- Generate visualisations which are then **interpreted** by a human
- Perturbation techniques on top of other techniques to find **localised importance** from the dataset
- Model performance **metrics** and perturbation **analysis**
- Calculating **similarity** between a truth dataset feature map and an **extracted** feature map

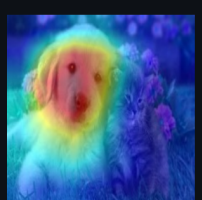
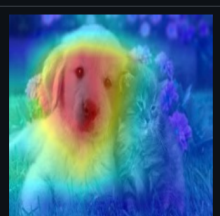
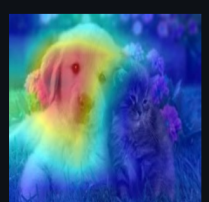
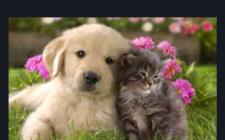
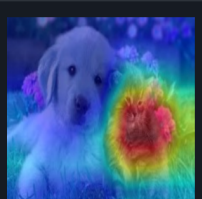
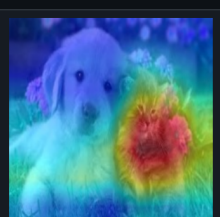
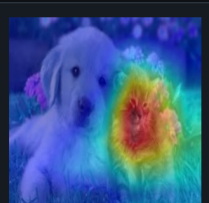


My work

GradCAM

+

Segmentation Maps (Truth DS)

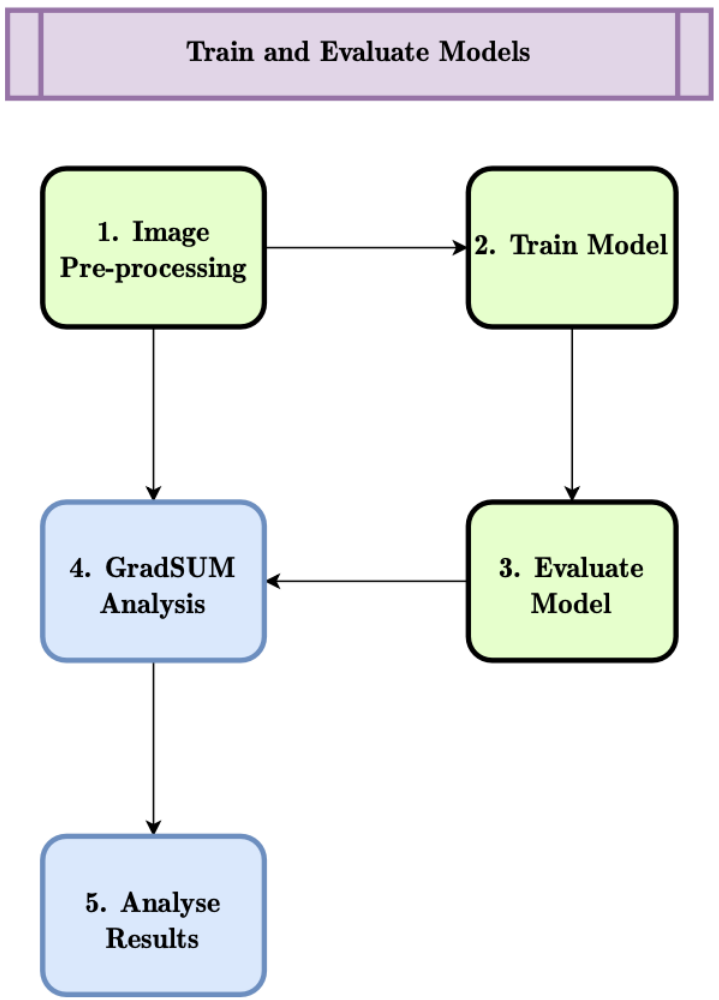
Category	Image	GradCAM	AblationCAM	ScoreCAM
Dog				
Cat				

https://github.com/jacobgil/pytorch-grad-cam

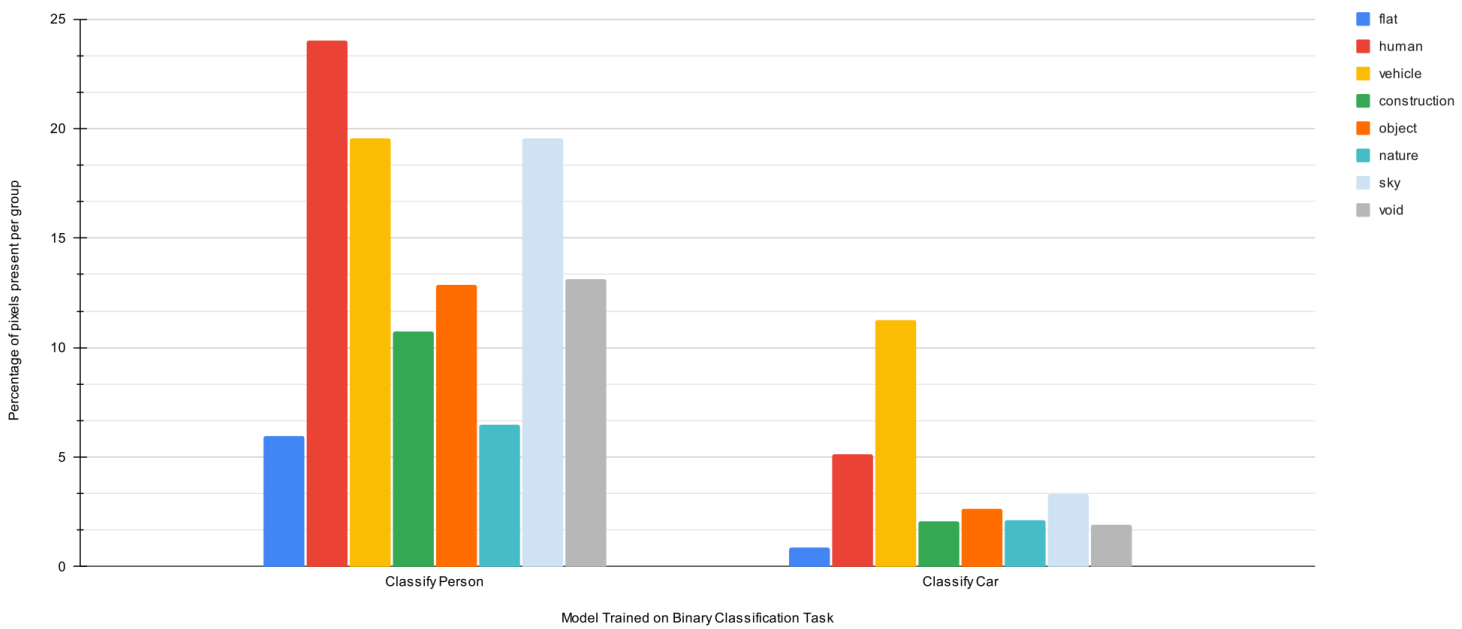
GradSUM

Using a ground truth dataset (Cityscapes), generate a sum (per segmentation class), of each pixel of that class weighted by the activation map and divided by the total number of pixels of that class

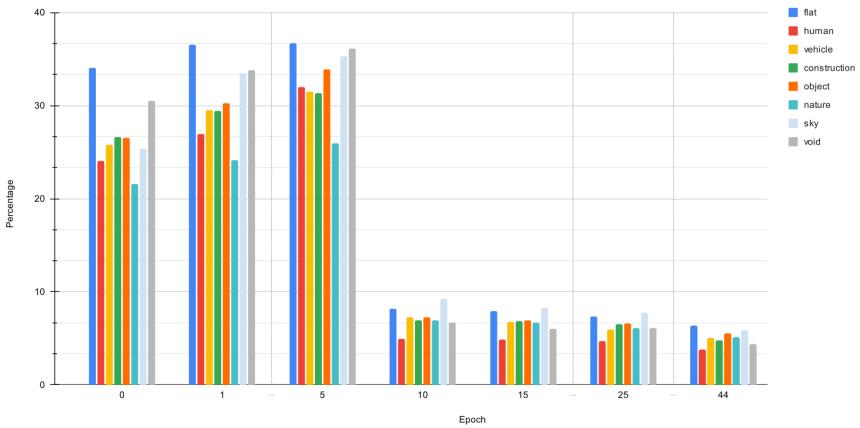
GradSUM


Train and Evaluate Models
1. Image Pre-processing
2. Train Model
3. Evaluate Model
4. GradSUM Analysis
5. Analyse Results

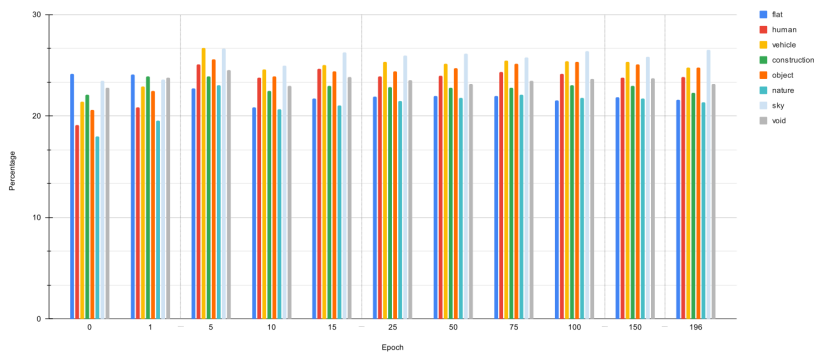
GradSUM


25
20
15
10
5
0
Percentage of pixels present per group
Classify Person
Classify Car
Model Trained on Binary Classification Task
flat
human
vehicle
construction
object
nature
sky
void

GradSUM



(a) NetSVF Profile



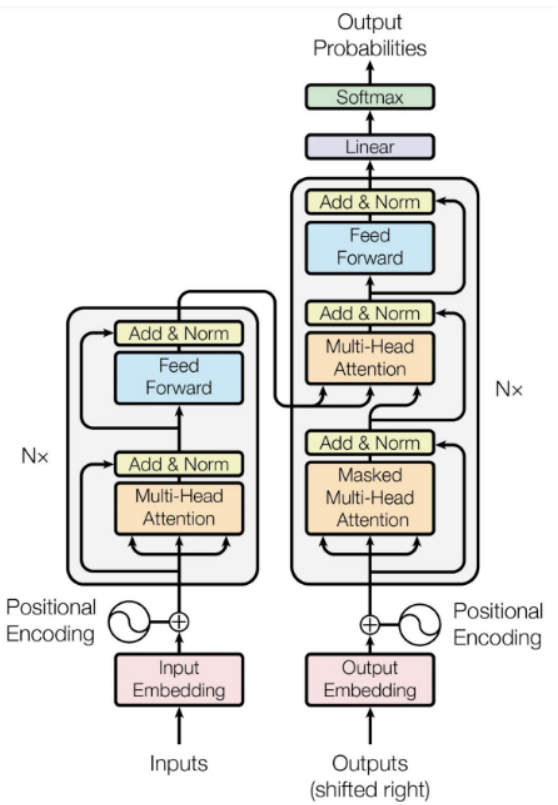
(b) NetHVF Profile

Possible Issues

- Performance
- Needing semantic data
- Accuracy and granularity of the semantic data
- Transferability of model behaviour between datasets


ANS
FORMERS

What is a Transformer?



https://jinglescode.github.io/2020/05/27/illustrated-guide-transformer/

What is a Transformer?

- Comes from NLP research
- Has an encoder stage and a decoder stage
- Vectorises the input (Tokenizer)
- Embedding to represent meaning
- Has a positional vector for each component
- Uses attention maps to determine what parts of the input should be focused on

What is a Transformer?

- Fed into an Encoder attention block
- Generates attention vectors for every token

What is a Transformer?

- The decoder block is fed the input of data you want to transform the working set into i.e. The output language in a translation problem
- The attention vectors are also fed in
- The meaning of each word is encoded at the embedding layer
- Maps attention vectors between Encoder and Decoder
- Predicts next output (classification etc) using a feed-forward network
- Repeats until the end of the sentence or input is reached

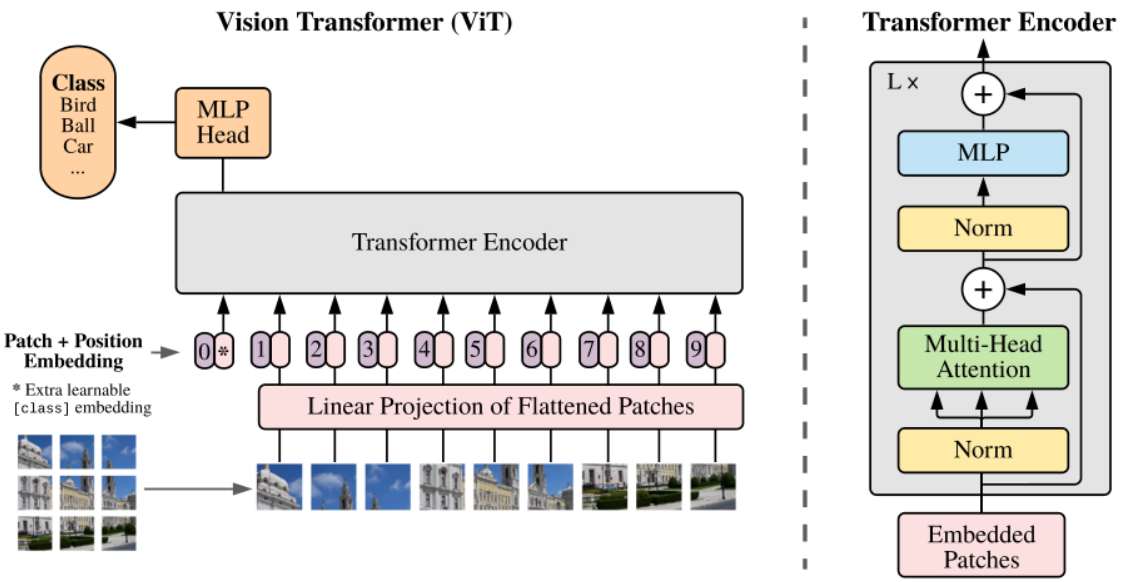
AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE an overview

Dosovitskiy Et. Al.

What is a vision transformer (ViT)?

- In this paper they propose an architecture to replace a conventional CNN architecture
- Can be combined with a CNN -> They argue causes complex engineering and performance challenges

What is a vision transformer (ViT)?



What is a vision transformer (ViT)?

- Many other steps added to training a Large Language Model.
- Addition of reinforcement learning and adversarial training
- Possibly multiple stages of fine-tuning
- Pre and Post-processing of data and outputs
- Biasing Datasets to maximise results
- Input data needs to be able to be decomposed into square patches

Mechanistic Interprability

Why?

- Comparing models
- Model compression / Reduction
- Production continuous integration and testing
- Biasing the models and confirming the bias
- Identifying issues within the input data
- Possibly identifying "expert" parts of a ViT model to transfer it (Open question)
- Compression of Models

Explainability Techniques

- Linear Probing
- Attention Maps (Qualitative Visulisation)
- Semantic Correlations (GradSUM, concepts)
- Perturbation analysis (Causual Ablation)
- Information flow modelling inside the model (Induction Heads) - Still somewhat open for research
- And others
- Mostly by hand analysis of qualitative results

Explainability Techniques

- Tehniques all limited by scale of the problem
  - Size of transformer models
  - Multiplicative size of the input data
- Implementation specific details and "magic" sauce
  - May make the models not differentiable
  - Throw away important information such as the attention maps
  - Aggressive approximations
  - Reproducibility of input data
  - Genetic algorithms used to generate model architecture - dropout

GradCAM for Transformers (Vision)

- Its theoretically possible
- Computationally expensive
- Attempted with 16x16 words ViT models
- Encountered memory leaks
- *Have new ideas here

https://github.com/jacobgil/pytorch-grad-cam/blob/master/tutorials/vision_transformers.md

GradCAM for Transformers (Vision)

- Alternative approach. Apply GradCAM to the tokenizer CNN only
- Use the attention maps in a GradSUM approach
- Combine results in a general anlysis

GradCAM for Transformers (Vision)

- Ive managed to get some partial preliminary results (For regression problem - steering angle)
- Appears the tokenizer CNN (which is of a similar structure to my simplest CNN model for self-driving cars) is doing the heavy lifting for these specific models
- Attention maps appear close to uniform and have small probabilities.
- Still encountering implementation problems

Next Steps

- Attempt to use the example simple model from the GradCAM technical reference (deit_tiny model from Facebook Research)
  - Apply GradCAM over the entire model
  - Apply split technique
- Attempt to train, evaluate, and analyse on the binary classification problems to identify the bias introduced to the model
- Attempt to apply to pre-trained weights and the corresponding dataset to confirm results
- Do more reading
- Get a paper written out of it

Implementation Pains

- Throw away data
- Dropout in unusual places preventing backward passes
- Memory leaks in implementation when storing attention weights
- Missing layers
- Half precision not behaving as expected
- Scaling of ViT models leads to huge memory and computation requirements
- Getting a Truth dataset can be difficult
- Automating the analysis of model behaviour is still tricky