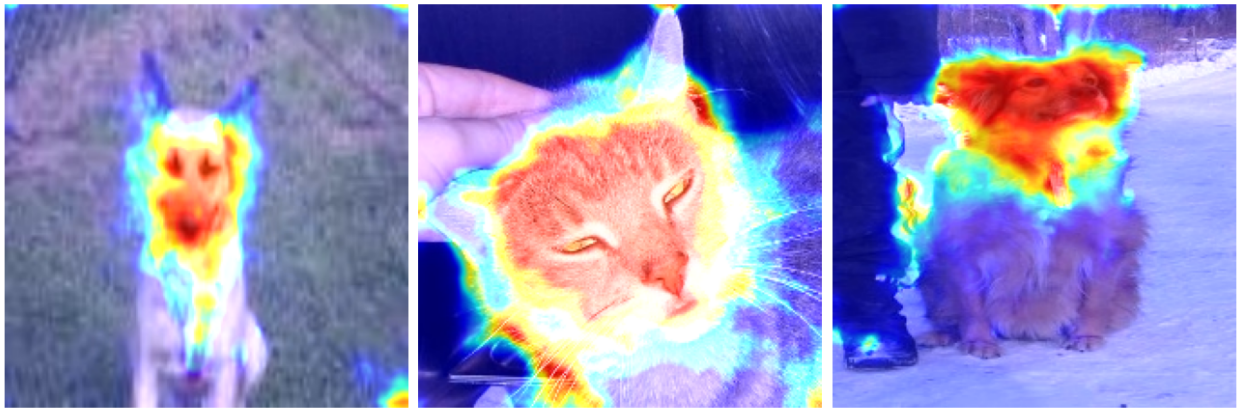
Understand your Algorithm with Grad-CAM

Why should we trust AI enough to drive cars, detect diseases, and identify suspects when it is a black box?

Daniel Reiff



Grad-CAM interpretation of cats & dogs (Image by Author)

Introduction

Is AI just a black box that we started trusting enough to drive cars, detect diseases, identify suspects just because of the hype?

You may have heard of the Netflix documentary, *Coded Bias* (you can watch the film here). The film criticizes deep learning algorithms for their inherent biases; specifically their failure to detect dark-skinned and female faces. The film suggests that the solution to the problem is in government. To “push for the first-ever legislation in the U.S. to govern against bias in the algorithms that impact us all.” Is

there something inherently biased about deep learning algorithms themselves or is it a question about bias in the training data? While a need exists for AI regulation, a large part of the solution sits with machine learning engineers. We need to make our deep learning algorithms more explainable, find areas of weakness and bias, and improve our algorithms.

How can we understand what our convolutional neural network (black box) sees and understands when making a decision?

This blog is all about producing visual explanation heat-maps that will help us understand how deep learning algorithms make decisions. At Forsight, a construction tech startup focused on construction safety and security, our computer vision team uses visual explanation to improve our datasets and models. Read more about our work here. For now, we are going to explore the visual explanation algorithm: **Grad-CAM**.

In this article, we'll show you how to interpret a dataset of cat & dog images using Grad-CAM. You can easily extend this example and use it to debug your own model. We'll show you how to improve accuracy and precision over the existing literature by looking at Grad-CAM heat-maps over earlier layers in the model. Lastly, we'll look into the model's mistakes and see how Grad-CAM can help us improve our model going forward. Let's jump in.

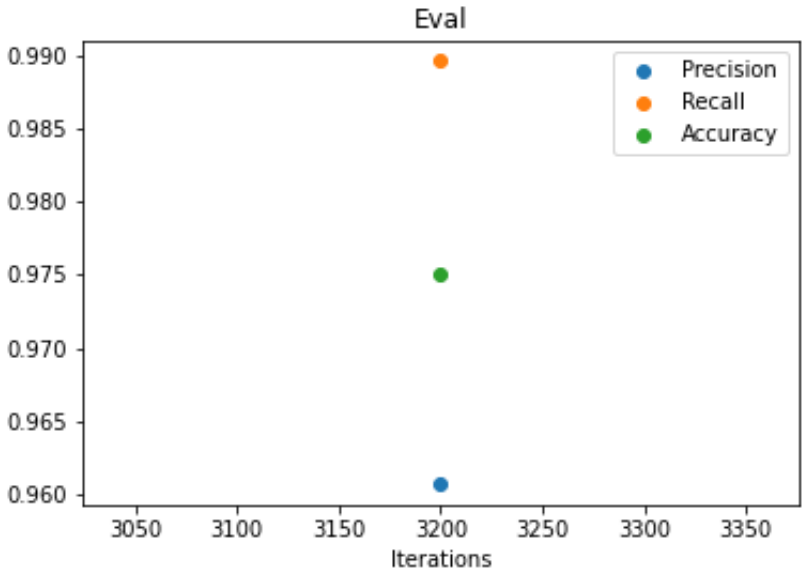
The supplementary Google Colab notebook will help you reproduce the heat-maps produced here. Feel free to copy and use the code for your own projects!

The Dataset and Model

Before applying Grad-CAM interpretation to complex datasets and

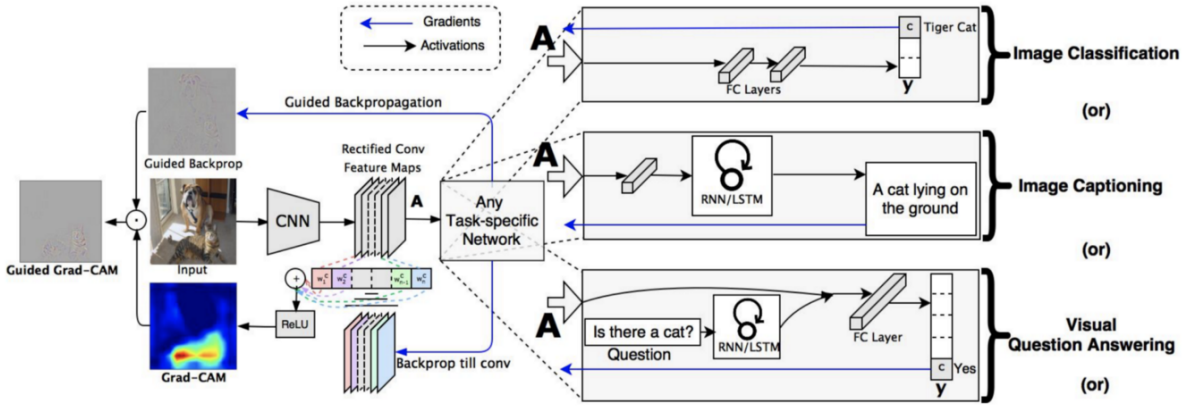
tasks, lets keep it **simple** with a classic image classification problem. We will be classifying cats & dogs with a high quality dataset from kaggle. Here we have a large dataset containing 37,500 images (25,000 train & 12,500 test). The data consists of two classes: cat & dog. The data contains cats & dogs in a variety of backgrounds, differing levels of brightness, and engaging in different activities which will put our visual explanations to the test!

On the Google Colab notebook, you will load in a pre-trained model that can already classify cats & dogs. For those interested in the architecture of the model: it consists of a rescaling layer for standardization followed by **MobileNetV2** layers previously trained on the COCO dataset. After the base model, we included a global average pooling layer, a dropout layer with 10% dropout, and a dense prediction layer. You can see all evaluation set metrics after training in the figure below. Reminder that a false positive indicates a cat falsely classified as a dog. Likewise, a false negative indicates a dog falsely classified as a cat. With a trained model ready to go, we are ready to investigate the question: *how is it making its decisions?*



Model Metrics (Image by Author)

The Grad-CAM



Grad-CAM overview by [Ramprasaath R. Selvaraju et al.] on arxiv.org

Warning, the Grad-CAM can be difficult to wrap your head around.

> *Gradient-weighted Class Activation Mapping (Grad-CAM), uses the gradients of any target concept (say 'dog' in a classification network or a sequence of words in captioning network) flowing into the final convolutional layer to produce a coarse localization map highlighting the important regions in the image for predicting the concept.*

[The technique] is an improvement over previous approaches in versatility and accuracy. It is complex but, luckily, the output is intuitive. From a high-level, we take an image as input and create a model that is cut off at the layer for which we want to create a Grad-CAM heat-map. We attach the fully-connected layers for prediction. We then run the input through the model, grab the layer output, and loss. Next, we find the gradient of the output of our desired model layer w.r.t. the model loss. From there, we take sections of the gradient which contribute to the prediction, reduce, resize, and rescale so that the heat-map can be overlaid with the original image. You can follow the specific steps in the code below and check out

the academic paper which outlines the complete mathematics here.

```
gradModel = Model(inputs=[model.inputs],outputs=[model.ge
```

Equipped with Grad-CAM producing code, we took our pre-trained model and our validation set to see *what our model had learned about cats & dogs*.

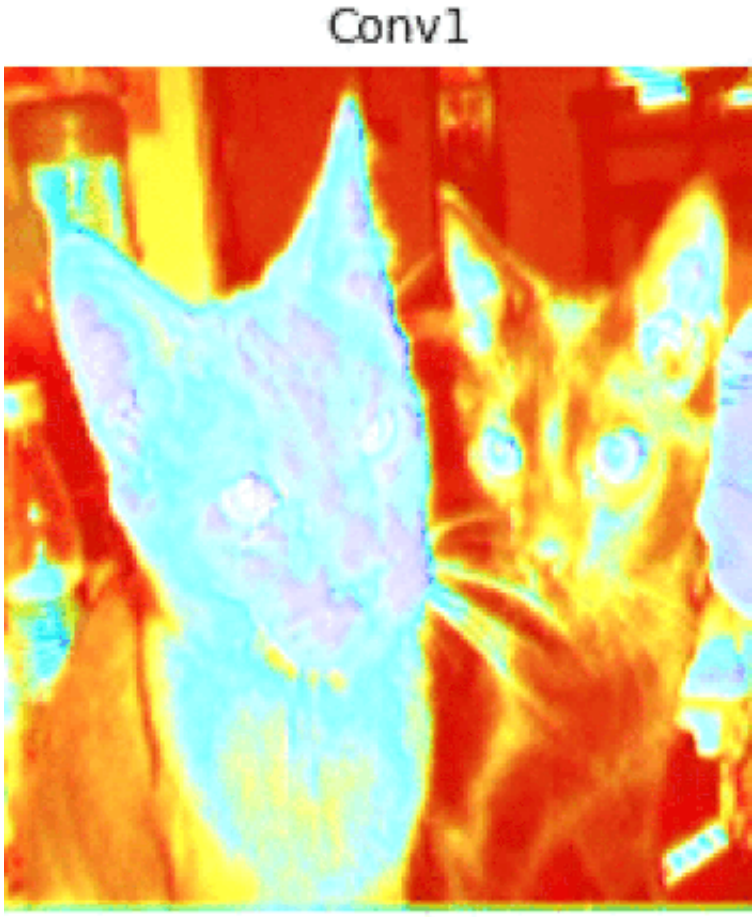
We started out creating Grad-CAM heat-maps for the last convolutional layer in our model, Conv_1. In theory, the heat-map for this layer should display the most accurate visual explanation of the object being classified by the model.

Grad-CAM heat-map of conv_1 layer (Image by Author)

In the gradient math, we are capturing the importances of all successive feature maps leading up to the last convolutional layer. We noticed that while the heat-map does emphasize the classified object, a cat in this case, it isn't very precise. The emphasized region (red) encapsulates regions of the cat but does not fit the cat with much precision. The region includes sections of the human's shirt, the human's hand, and the background. We know the model sees a cat, but we are not quite sure what it is about the cat that convinces the model that this is, in fact, a cat. **How can we make the Grad-**

CAM heat-map more precise?

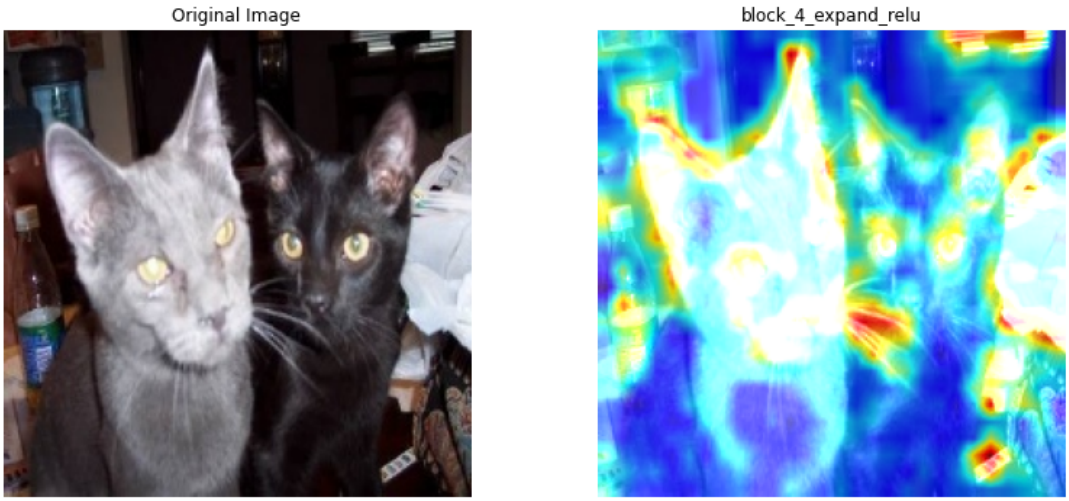
You'll notice that the code above has been wrapped in a function on the Google Colab notebook to allow for the Grad-CAM to be reproduced for different model layers. Lets explore earlier model layers using this function.



GIF of Grad-CAM heat-maps over all model layers (GIF by Author)

There is a lot to digest here. As we studied the heat-maps, a logic in how the model learns began to emerge. The first ~10 layers (blocks 1 through 3) are detecting contours and borders in the image. Depthwise layers de-emphasize objects while project & expand layers de-emphasize contours. The next ~20 layers (blocks 4

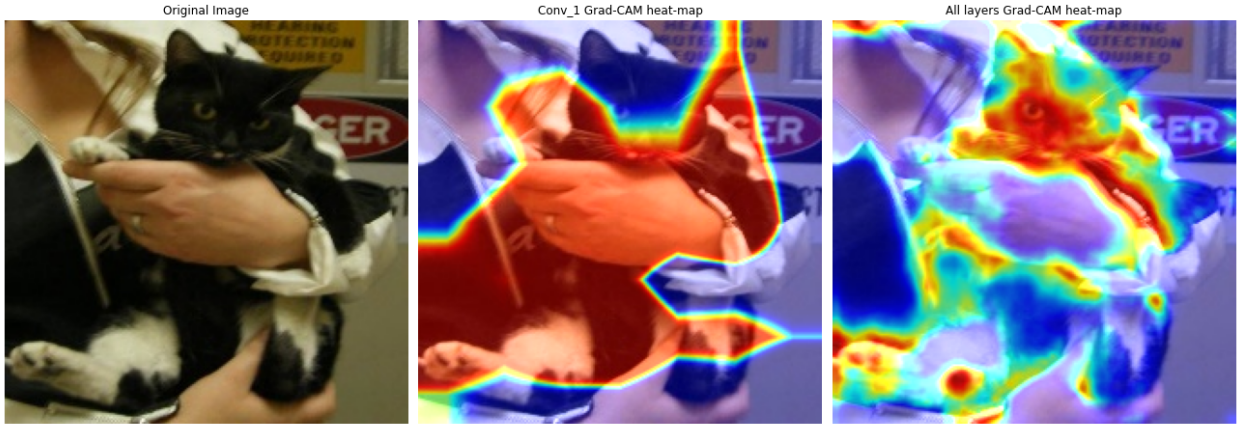
through 11) are detecting concepts in the image. Block 4 expand relu is a great example.



Grad-CAM heat-map of block_4_expand_relu layer (Image by Author)

At this point in the model architecture, shapes that define the characteristics of a cat are starting to stand out from the general contours in the image. You can see how the cat's whiskers, ears, and eyes are emphasized in shades of red and yellow while the rest of the image is colored in blues and greens. You can see how the model is reasoning how a cat's defining characteristics differentiate this image as we get deeper into the model's layers. The final ~16 layers (blocks 12 through 16) try to identify the object in the image using spatial information from the earliest layers and concepts developed more recently. The final layer in the model, Conv_1, does correctly identify the general region of the object, but it does not pick up the nuances of the object that exist in earlier layer Grad-CAM heat-maps. **We used human intuition instead of Grad-CAM mathematics and it improved upon the Grad-CAM results.**

To incorporate earlier layers, we averaged together Grad-CAM heat-maps from **all model layers**. You can see the result below.

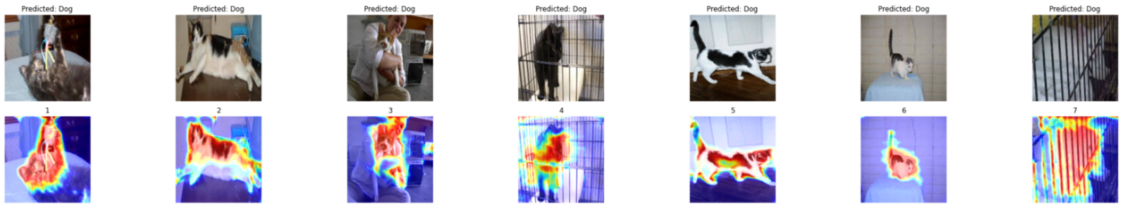


Grad-CAM Heat-map Improvement (Image by Author)

The Grad-CAM heat-map now emphasizes the cat's face, eyes, and paws and de-emphasizes the human's arm. Overall, we have a much more precise region of emphasis that locates the cat. We know that the model classifies this image as a cat due to its intrinsic features, not a general region in the image.

The model is especially impressive because this is a difficult image! The human is wearing a white and black shirt which is the same color as the cat. Yet, the model can differentiate the black cat face from the black human shirt. Through the all-layer Grad-CAM we've learned about the models strengths. **How can we use Grad-CAM to learn about our model's weaknesses?**

Lets take a closer look at 7 images of cats falsely classified as dogs as you can see below.



Model False Positives (Image by Author)

We can use the Grad-CAM heat-maps to give us clues into why the

model had trouble making the correct classification. It appears the model is having some trouble differentiating a cat when it sits behind a cage (4, 7). We can see from the Grad-CAM that the model is emphasizing bars of the cage and having trouble finding cat features. In 5 and 6, cat tails are distinctive features in the image. But from the Grad-CAM, we can see that the model is having trouble recognizing this feature as its colored in shades of green and blue. The clearest example of what Grad-CAM heat-maps can provide is in # 1. The model is confused by the toy the cat is playing with. From the heat-map we can see that the model believes that the toy is part of the cat. The confusion has caused a false classification by the model. Using this knowledge, we can find more examples of cats playing with toys and include it in our dataset which will help our model learn and improve.

Improving the Model

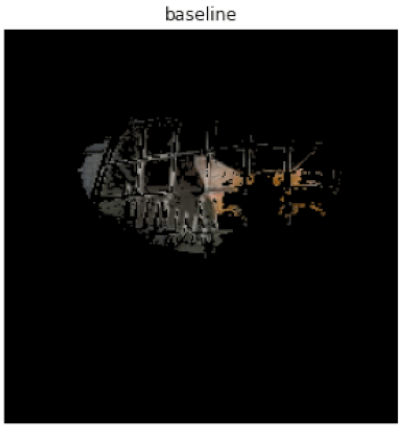
Noticing that our model was having trouble with cages, we presented it with some more specific examples from google images; specifically, transporter crates.

The model has trouble with transporters (Image by Author)

As you can see from the red areas on the grad-CAM heat-map, the model is focusing on the transporter crate surrounding the cat rather than the cat itself and it is leading to a false classification. *How can we help the model differentiate cats when they are in transporter crates?*

We created a supplemental dataset of cats & dogs in transporter crates from google images. Using albumentations, we beefed up the dataset with additional image augmentations. Armed with the data we needed, we began experimenting. We created a series of models with our original dataset and added 25%, 50%, 75% and 100% of the supplemental transporter crate dataset. At each iteration of our experiment, we went back to the Grad-CAM to see if the model adjusted its region of emphasis on the original image.

To see where the model is really keying in on, we created a mask that uses thresholding to capture the most intense (red) areas of the Grad-CAM. We used the mask to then segment out the most important pixels of the original image.



The model improves as we add data from the augmented set (GIF by Author)

As you can see from the filtered image above, the model's focus shifts from the crate to the cat's face and features as we add more

images from the supplemental dataset to our training dataset. And it doesn't take long before the model starts getting the classification right! When the model is trained on the original dataset plus just 25% of the supplemental transporter crate dataset, the model assigns a 90% probability of the above image being a cat.

Using the same experiment, we looked at other images of cats in crates to see if we could find the same progression in model focus. Below are a couple that stand out.

Baseline Model

GIF by Author

Baseline Model

GIF by Author

Conclusion

In this blog, we've hopefully provided some useful insights into and tools for convolutional neural network visual explanation using the Grad-CAM. With the help of cats & dogs, we've explored how a model differentiates classes. We've improved the Grad-CAM results from the final convolutional layer by considering heat-maps from all layers of the model. We've used Grad-CAM to zero in on areas of bias and weakness in the model. **Most importantly, we've used the Grad-CAM to improve our algorithms.** Machine learning is an iterative process where our models are never good enough. We hope that the techniques presented here will help provide more transparency and create more trust in models throughout the improvement process. At Forsight, we use Grad-CAM heat-maps to identify erroneous areas of emphasis by our models.

In order to help ML/AI enthusiasts, and anyone interested in helping to solve this problem in general, we've created and shared a Google Colab notebook which gives you the ability to play around with and produce the Grad-CAM heat-maps we've presented in this blog.

If you're interested in this topic and you would love to work on deep learning algorithm explanation, please reach out to us.

References

1. Ramprasaath R. Selvaraju et al., 2019, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization", https://arxiv.org/pdf/1610.02391.pdf
2. Dogs vs. Cats, https://www.kaggle.com/c/dogs-vs-cats/data
3. Mark Sandler et al., 2019, "MobileNetV2: Inverted Residuals and

Linear Bottlenecks", https://arxiv.org/abs/1801.04381
4. *Coded Bias*, 2020, https://www.codedbias.com/