# Questionable Image Content Detection

## Goal

I worked as a data scientist for a small digital advertisement company. However, unlike the many publisher startups out there, the focus of this particular startup was to provide key metrics for the beforementioned ad publishers. Websites have many ad slots to fill for the user to consume, and during the loading of the page, the spaces are being auctioned to bidding publishers within milliseconds during loading the page content. The publishers are bidding based on key metrics, mostly their own, bidding more if they feel the user exposed to their particular ad is more likely to convert. Conversion is the internet term for the desired action resulting from seeing an ad. That can be a click on the ad, a visit to the advertisers' website, and of course, the best result: a purchase. Publishers relied on our company for additional metrics. And one key metric we specialized in was how offensive or questionable the website was considered by the brand. Because pretty much all brands that the publishers place ads for have a strong desire to stay away from such websites.

## Data

Web 1.0's text content (articles, product descriptions, etc.) was easily evaluated, as it was exposed through the HTML. Read the HTML, and you had a decent sense of what the page was about. Over the years, with web 2.0 and moving towards web 3.0, this is changed quite a bit. Text content is no longer locally available and fetched on load through JavaScript or other internet protocols. And this fetching isn't necessarily deterministic. Therefore, this updated tech means one must load the website to retrieve the content. This can be done but is time-consuming as the entire page must be loaded. And with billions of websites on the web, ever-changing, it is a complicated problem. However, it does offer a new opportunity. Because with the text, the images are also loaded.

Figure 1 - AlexNet Architecture

## Approach and Model

And this is where I came in. I was tasked with building a model that would score the images in terms of questionability while the NLP team continued analyzing the text. The approach I took was the following.

1. A training data set was created by extracting images from websites where the NLP team had scored either as very questionable or not questionable.

2. The images extracted from the questionable websites were clustered by feeding them to a pretrained image classification algorithm. This model (AlexNet, see figure 1) was trained on 1000 image categories (the ImageNet dataset), so each questionable image thus yielded a 1000-element vector of probabilities. It was in this 1000-dimensional numerical space that the clustering algorithm was applied. Clustering images in the original format (say 256 by 256 pixels) is computationally expensive and makes relatively little sense. Images depicting the same object typically vary significantly in their pixel-by-pixel values (see figure 2).

3. When I felt I had an optimal number of clusters (the first iteration of the model used 21), I manually inspected them and determined that, indeed, most of them were clusters of images that a typical user would deem inappropriate (I will leave it up to the imagination of the reader what kind of categories those were) I retrained the original 1000-category model but adding the 21 clusters as additional categories.

4. After I was satisfied with the newly trained model, I created a few simple metrics that turned the output of the 1021-category model into a score indicating questionability. The metric was a simple weighted average of the 1000 original categories versus the 21 newly questionable categories. The NLP team then integrated this score into their existing scoring metric system.

Figure 2 - Images depicting the same object can vary significantly

## Impact

My approach ultimately was not revolutionary but proved to be useful. First, when there was too little text content, the image model could stand in and take over. Second, combined, the two methods typically provided a more robust metric. Third, when they diverged, both the image and text teams could fine-tune based on the observed discrepancy between models. I can give a handful of amusing examples, but I'd rather not offend the reader. You can think of a few examples of when the text diverges from the images.

## Technologies Used

Python

Torch

Scikit-Learn

AWS EC2

NVidia CUDA