Distant Viewing Toolkit

Tutorial Slides 1

Taylor Arnold
Lauren Tilton

https://distantviewing.org

These slides are designed to accompany the Python tutorials that we have built to introduce the theoretical and methodological foundations of *Distant Viewing* alongside an overview of the tools made available through the Distant Viewing Toolkit. The slides contain additional notes, references, and diagrams that we find useful when using the tutorials during in-person workshops. Most of the technical notes and materials are available direction in the notebooks, which can be accessed at the following links:

https://colab.research.google.com/drive/1qQKQw8qHsTG7mK7Rz-z8nBfl98QBMWGf

https://colab.research.google.com/drive/1n7qWm47laCUJwg0-Rdx7pNw3dQWwuyxz

For further information about our work, please visit Distant Viewing Lab website at **https://distantviewing.org** or feel free to email us directly at **tarnold2@richmond.edu** and **ltilton@richmond.edu**.

Tutorial 1

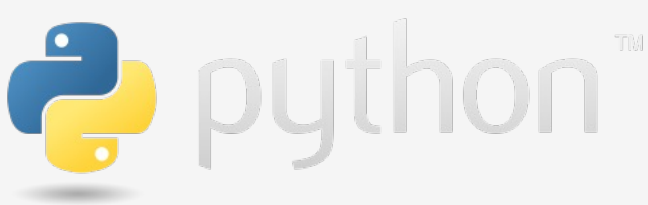
Movie Posters and Color Analysis

1.1 Setup

This tutorial makes use of Google's Colab environment, which allows us to use the Python programming language directly from the browser without needing to install any software locally. Colab can be used for small projects for free, with paid plans available to access more computing resources.

We will make use of three common Python modules for data analysis: **NumPy**, **Pandas**, and **matplotlib**.

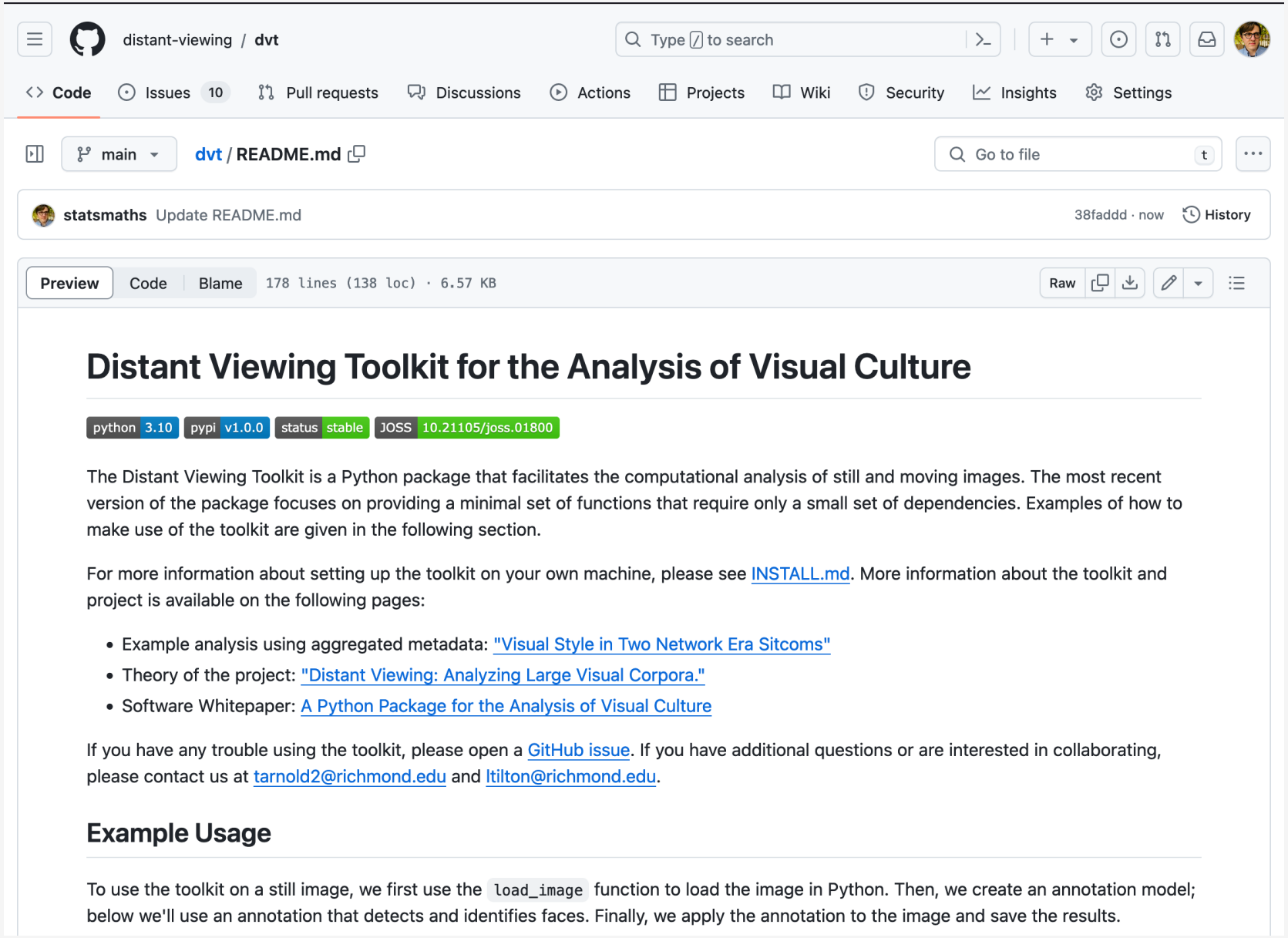
All the software libraries used in the tutorial are available under open-source licenses. The code we work through can all be run on your local machine with just a little bit of extra setup required to get started.



1.1 Setup

We will also be using a Python module that we built specifically for the humanistic analysis of visual materials: the distant viewing toolkit (**dvt**).

We will start by using **dvt** to load images into Python. The full power of the **dvt** module will become clearer as we move towards advanced computer vision algorithms towards the end of the first tutorial.

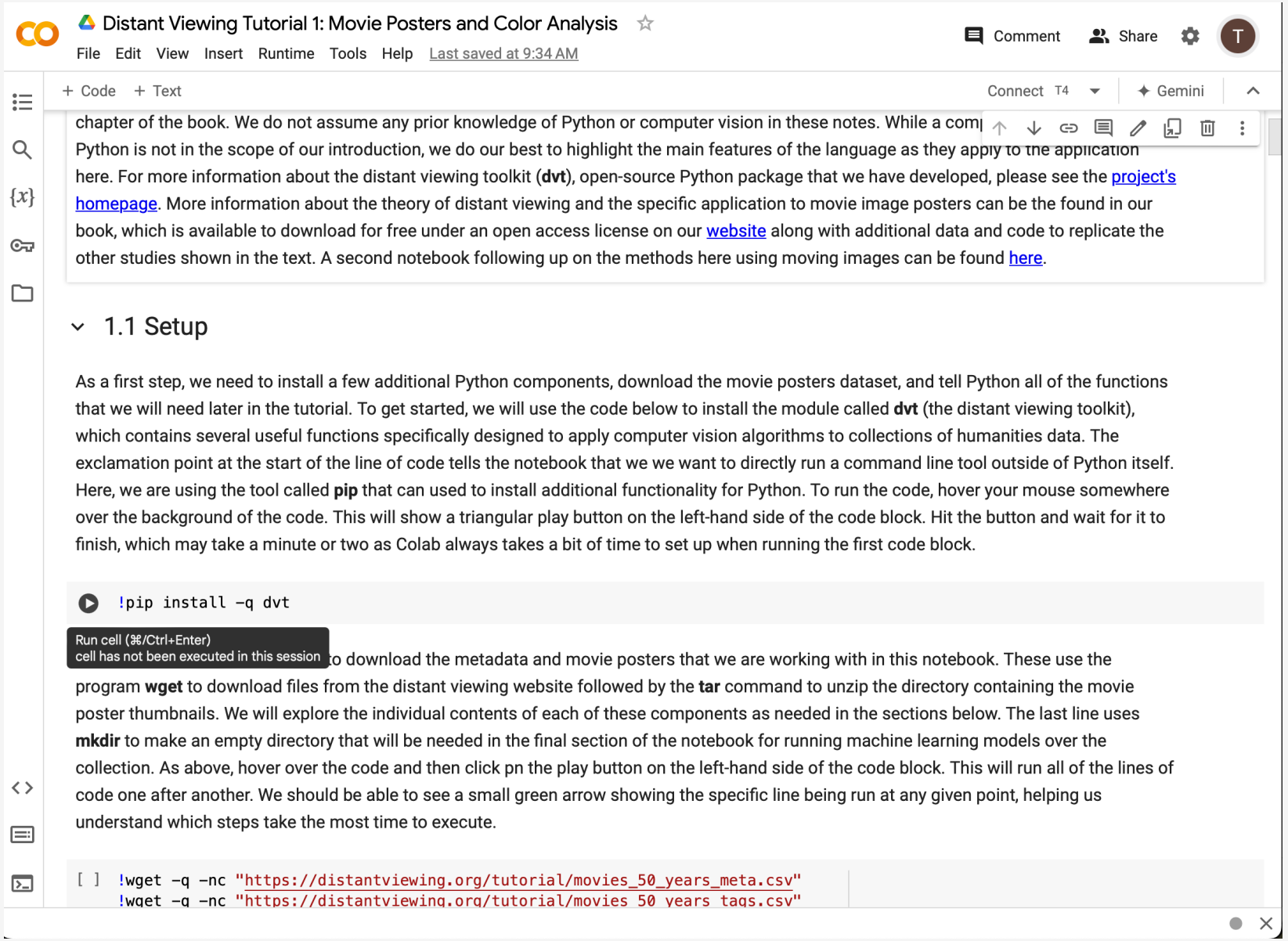


1.1 Setup

When opening the tutorial notebook in Colab, this should open a window in the browser similar to the one on the right-hand side here.

The notebook can be viewed by anyone; in order to run the code itself, we will need to sign into a Google account.

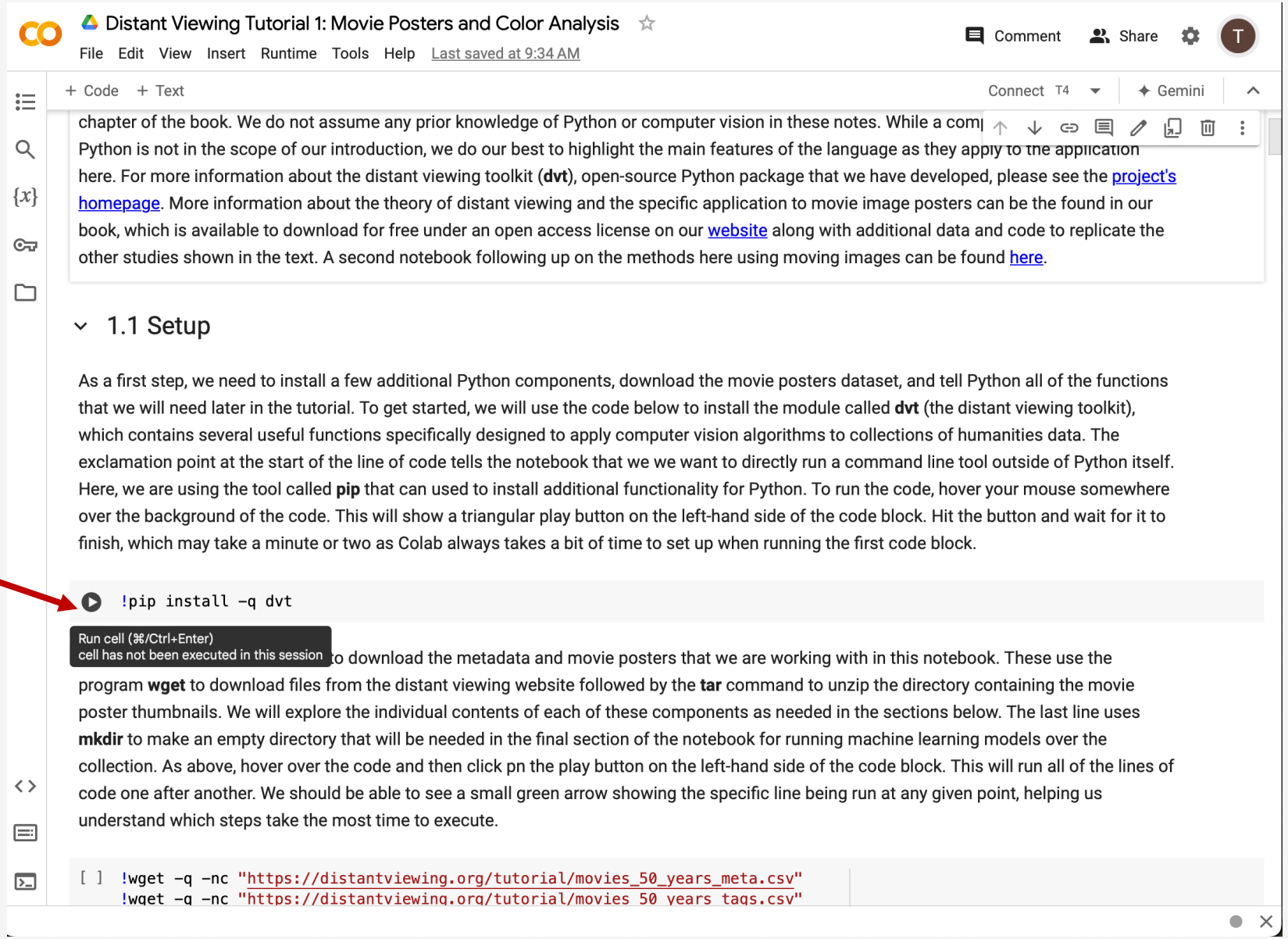
Note: You will be able to edit the code and text locally in your browser. These changes will not be saved for anyone else. So, at any point, feel free to play around with the code!



1.1 Setup

After signing into a Google account, we will be able to run the code block (the parts of the notebook on a grey background). To run the code, hover over the grey background and click on the play button as shown by the red arrow here.

Throughout the tutorial, we need to run each of the code blocks in order. You can go ahead now to read and run all of the setup code shown in the first section of the notebook.

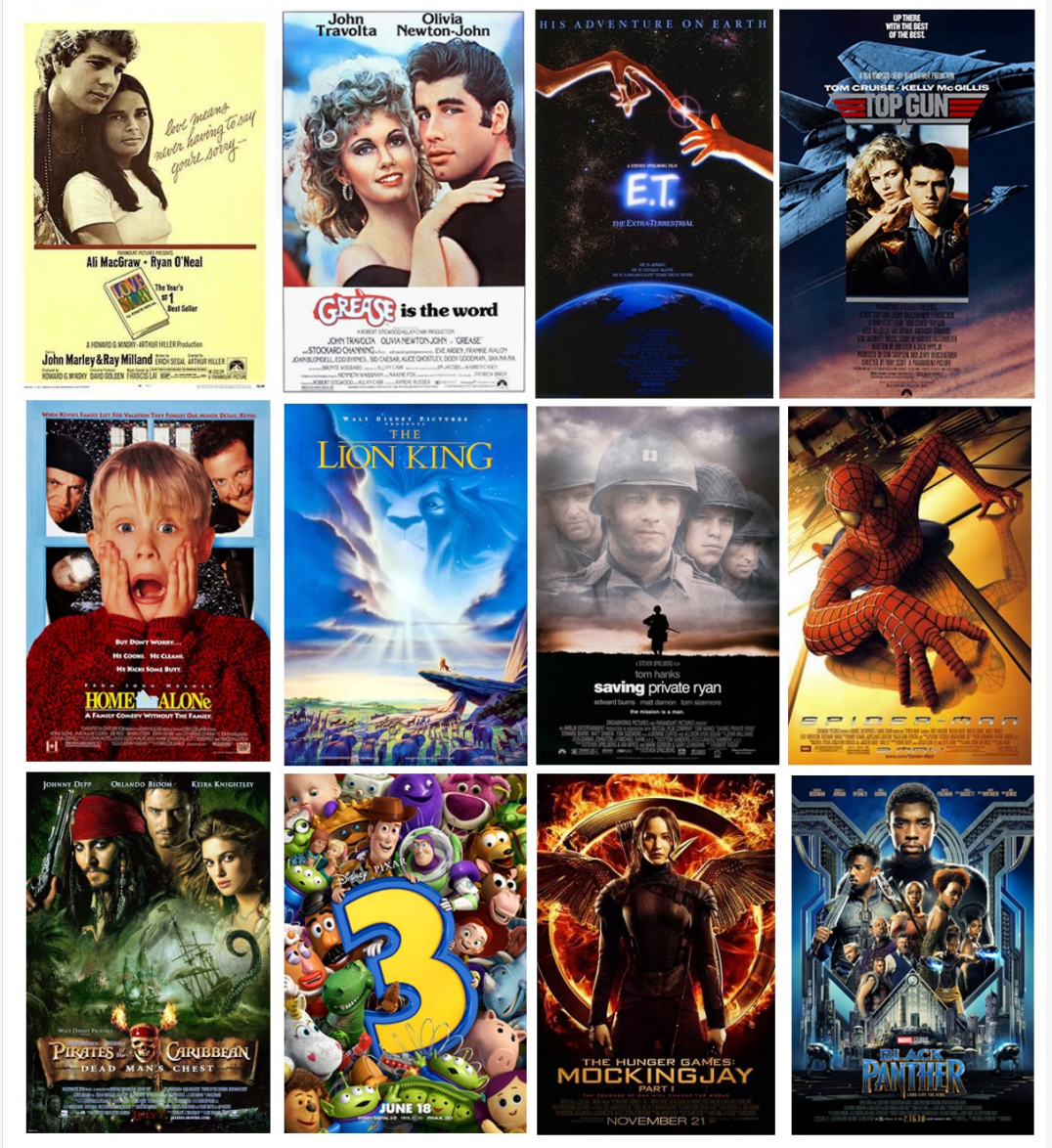


1.2 Movie Poster Dataset

In this tutorial we are going to explore a collection of movie posters from the top-100 grossing films from every year between 1970 and 2019. Our corpus consists of nearly 5000 images (we were unable to locate a small set of older posters). Several examples from across the decades and genres are shown to the right.

Our goal is to understand how the color and composition of the posters has (1) changed over time and (2) is used to convey (or subvert) genre conventions. We want to use computational techniques to be able to identify and understand patterns across all the thousands of images in our collection.

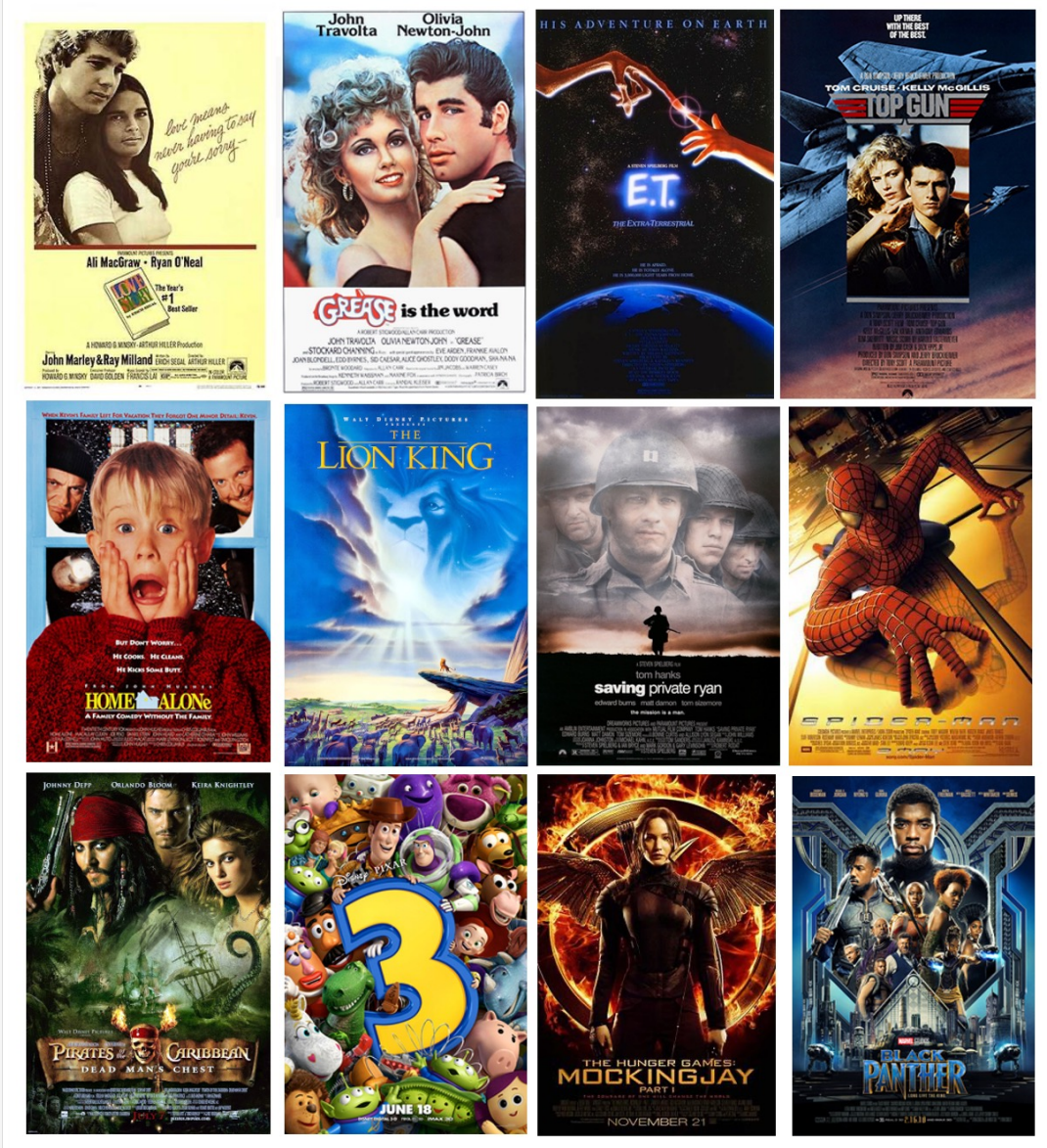
We will start by first understanding the ways that digital images are stored and how we can extract color information from them.



1.2 Movie Poster Dataset

Before we dive into the computational work, let's take a second to think about the underlying research questions of this work by considering the following questions:

1. What are the primary goals of movie posters? Have these goals changed over time?
2. What kinds of information do the makers of movie posters want to convey to the people who view them?
3. What kinds of information do you infer from the brightness, saturation, and the example movie posters on the right?
4. What hypotheses might we be able to produce about the relationship between the color and composition of a movie poster and genre?



1.3 Digital Images

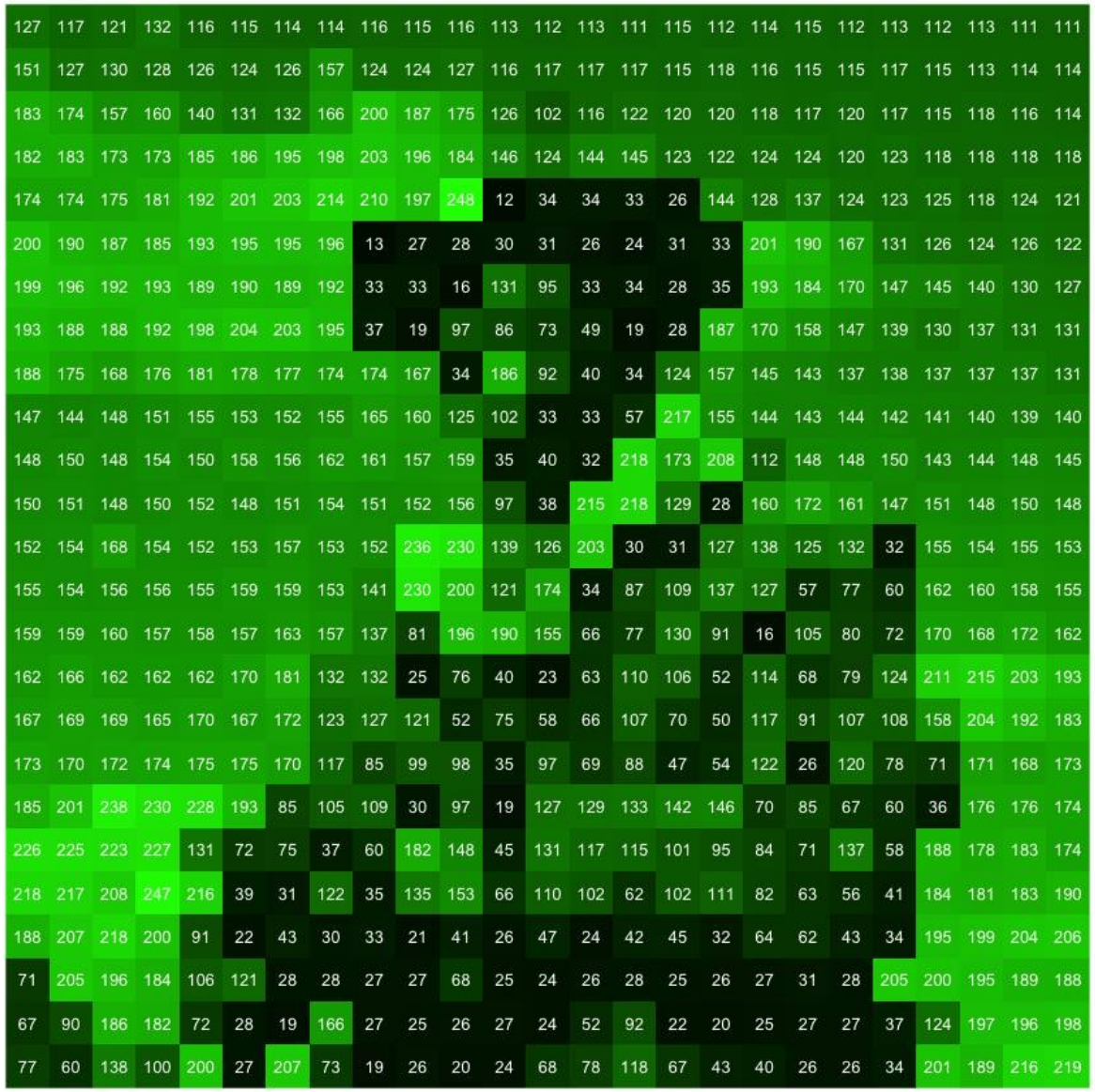
On the right is a cropped image taken from a photograph credited to Russell Lee taken in August 1942 while working for the U.S. Federal government.

While the image was originally taken using physical color film, the version that we have here is a digitized scan stored and displayed on a computer.

When viewing the image with our eyes, we can identify a variety of elements such as the horse, the man, the sky, and the clouds. But how is this information understood by our computer?

1.3 Digital Images

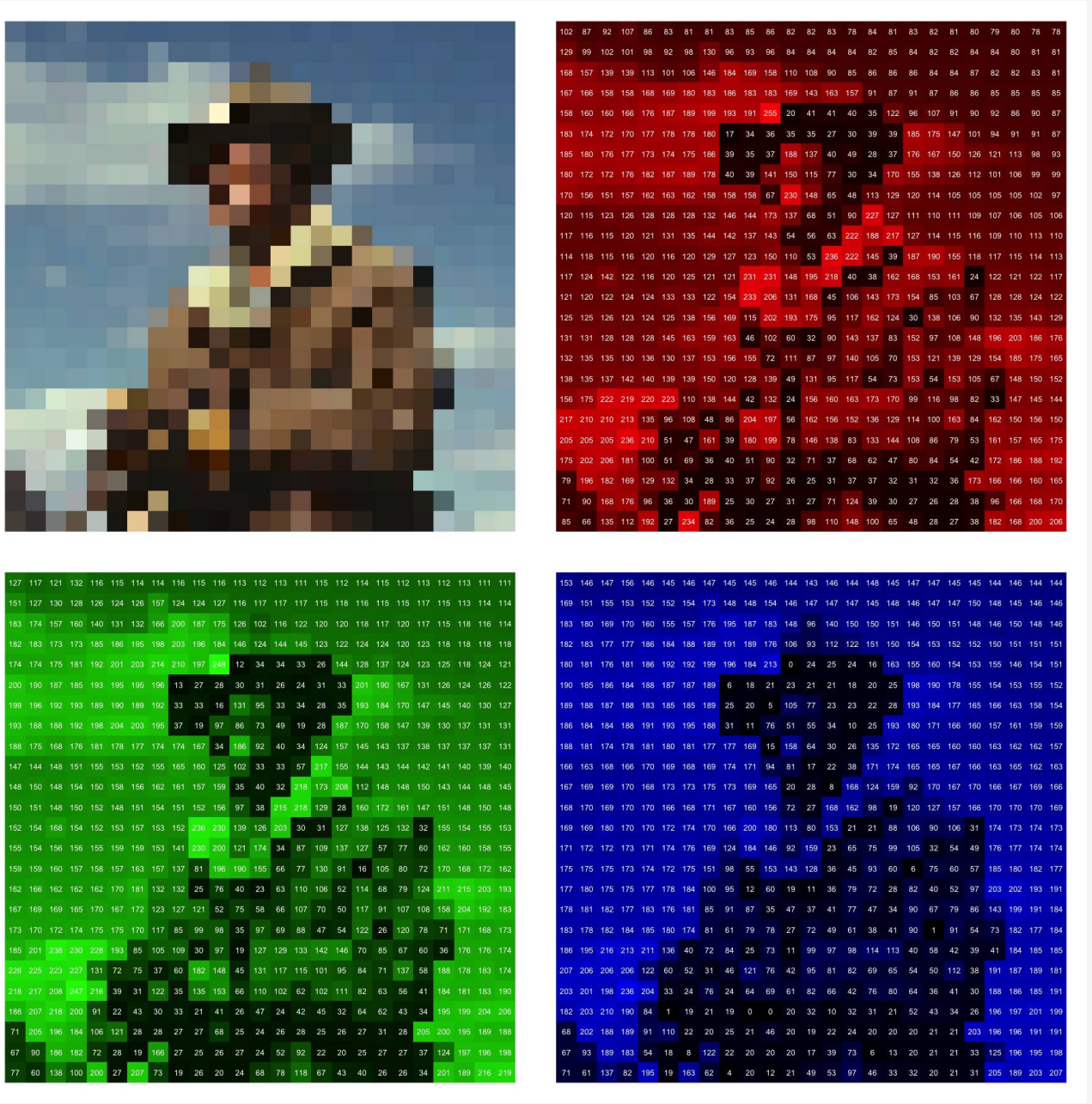
Looking at just the green pixel values, we have sufficient space to see the actual numbers. As with the code in Python, these numbers are on a scale from 0 (no green) to 255 (as much green as possible). Note that there is a lot of green light here even though nothing in the original image is something that we would usually call "green." The green light is blended with red and blue to create other colors such as brown, orange, and white.



1.3 Digital Images

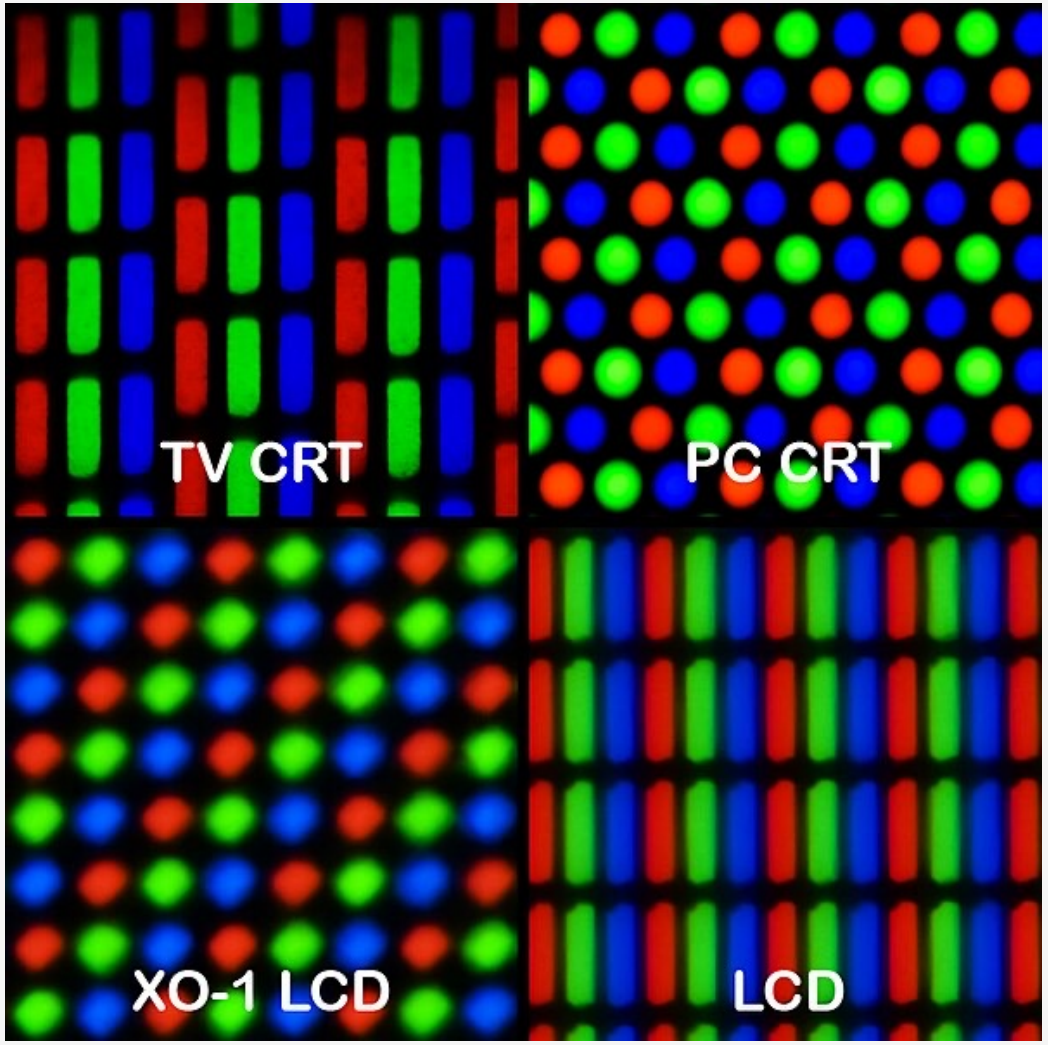
To understand the way that digital images are stored, we will further zoom into our image and consider a lower resolution version.

On the top-left is the modified image centered on the man. The other three parts of the diagram show how the image is represented as *pixel intensities* through the blending of red, green, and blue light. Each point in the combined image is created by defining the amount of red, green, and blue light needed to re-create the given color.



1.3 Digital Images

Usually, particularly on modern screens, the individual pixels are too small to individually see. But know that if we zoomed in far enough, we would see the individual red, green, and blue lights as shown in the image on the right.



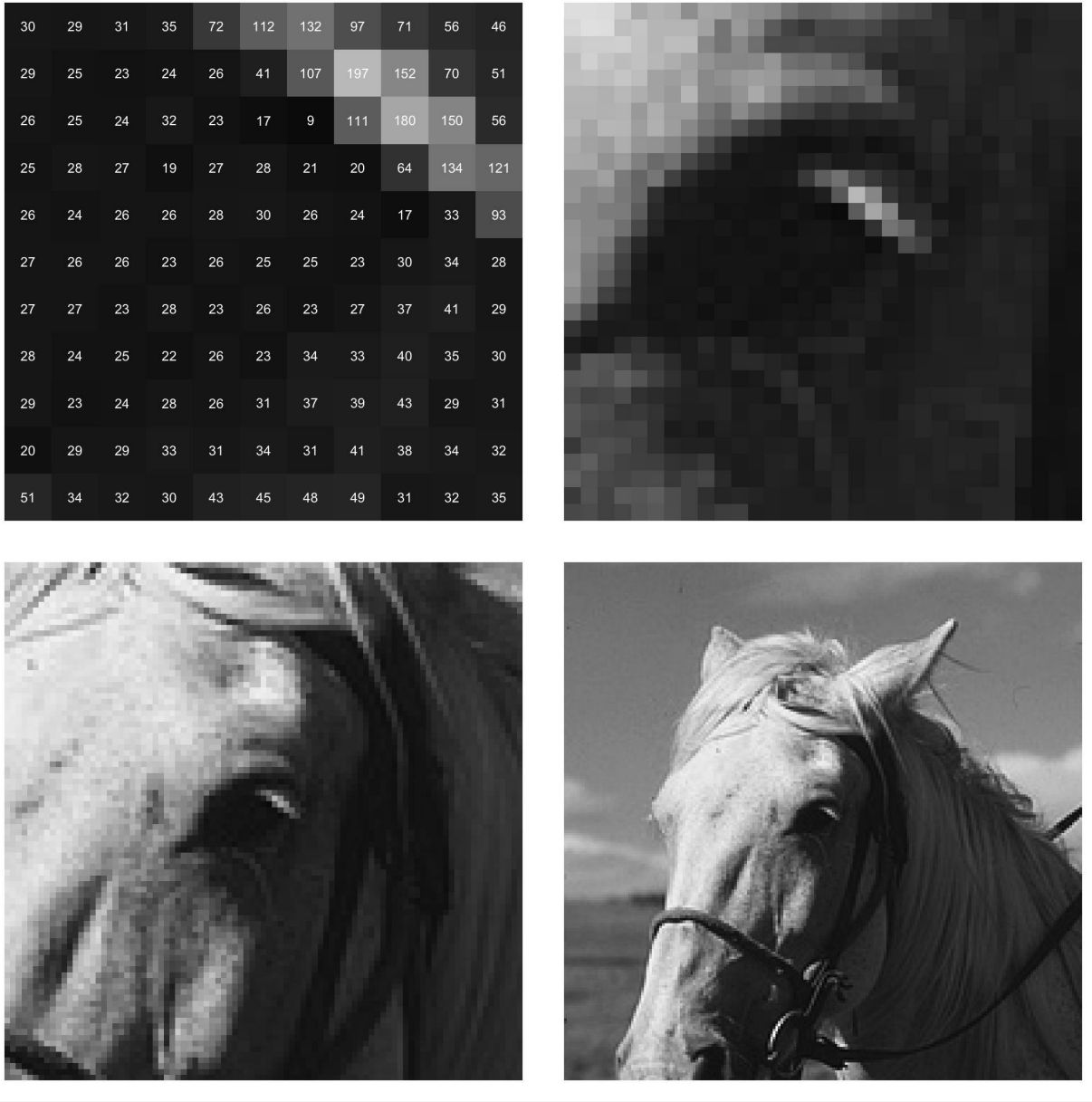
Peter Halasz, CC–SA 3.0, Wikimedia Commons

1.4 Distant Viewing: Theory

On the right is another set of zoomed versions of our example image. Each image is centered on the same point but zoomed out at different levels. For simplicity, we have converted the image into grayscale, which requires only one number for each pixel.

Notice that it is impossible to understand that the numbers in the upper-left hand corner represent the eye of a horse until we see those numbers in the context of a wider version of the entire image.

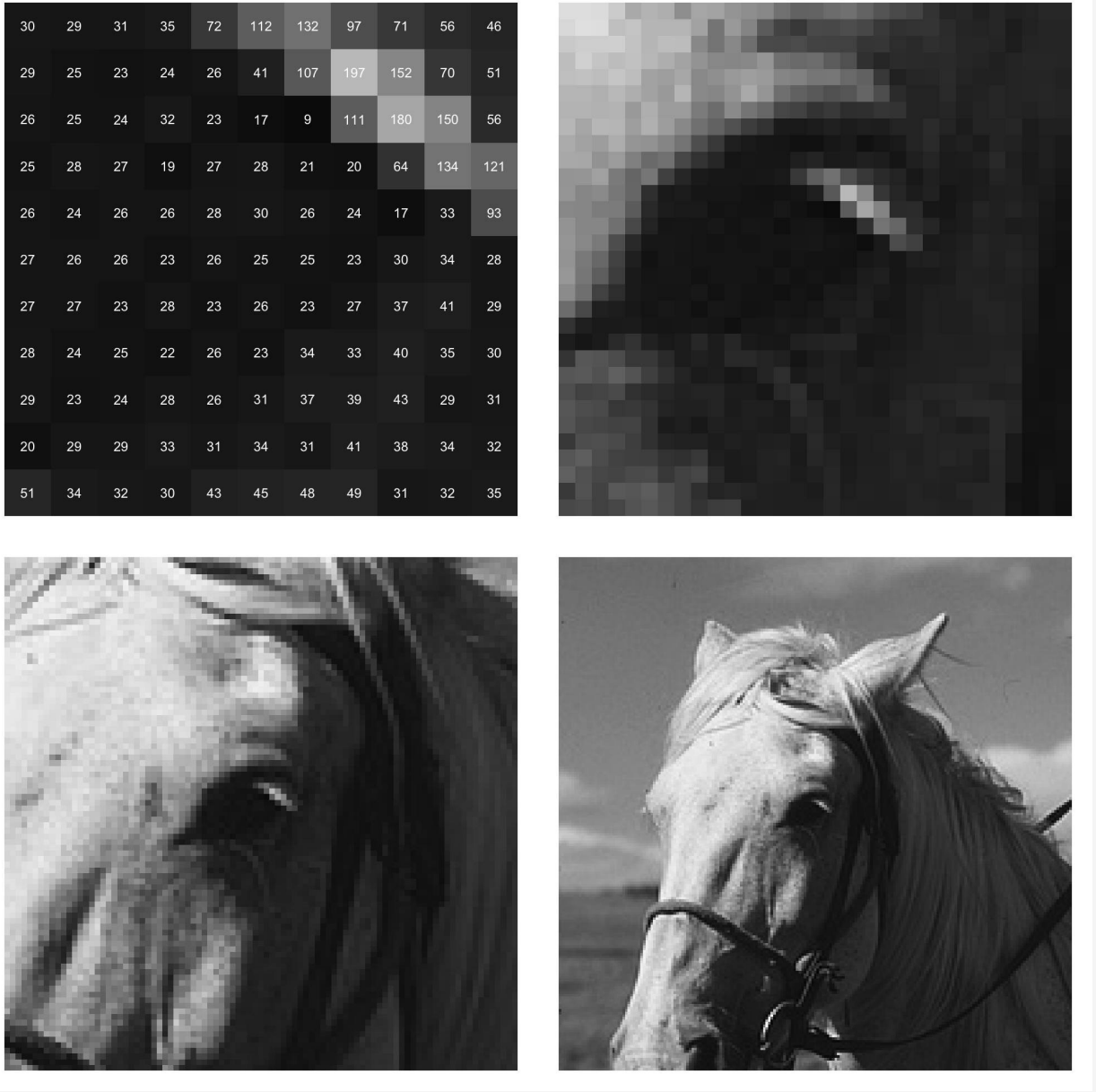
In fact, there is no explicit way that the digital image records that this is a horse. Rather, we recognize this as a horse because it *looks similar to that that a horse looks when we see it with our own eyes*.



1.4 Distant Viewing: Theory

In fact, there is no explicit way that the digital image records that this is a horse. Rather, we recognize this as a horse because it *looks similar to that that a horse looks when we see it with our own eyes*.

To use the language of semiotics, digital images (just as with their analog counterparts) make meaning through their mimetic properties. They convey information by sharing similarities with the things they represent. This is very different from the way that textual data conveys information.



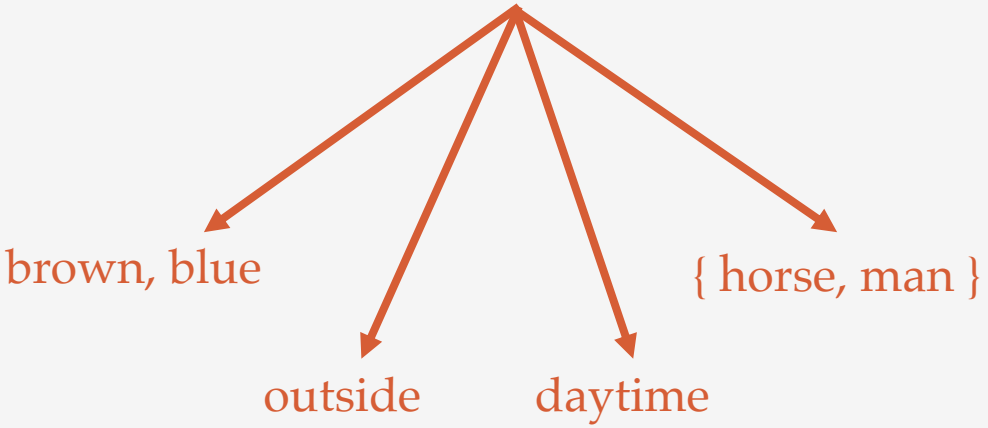
1.4 Distant Viewing: Theory

Because the digital images do not explicitly code semantic elements, we need to construct representations of the information that we want to study as a part of the analytical process.

For example, on the right we see a number of different ways that we can summarize the information about our example image through structured *annotations*. Ideally, we would be able to construct this algorithmically, which we will work towards in the remainder of the notebook.

Can you think of other ways to annotate the image with semantic information?

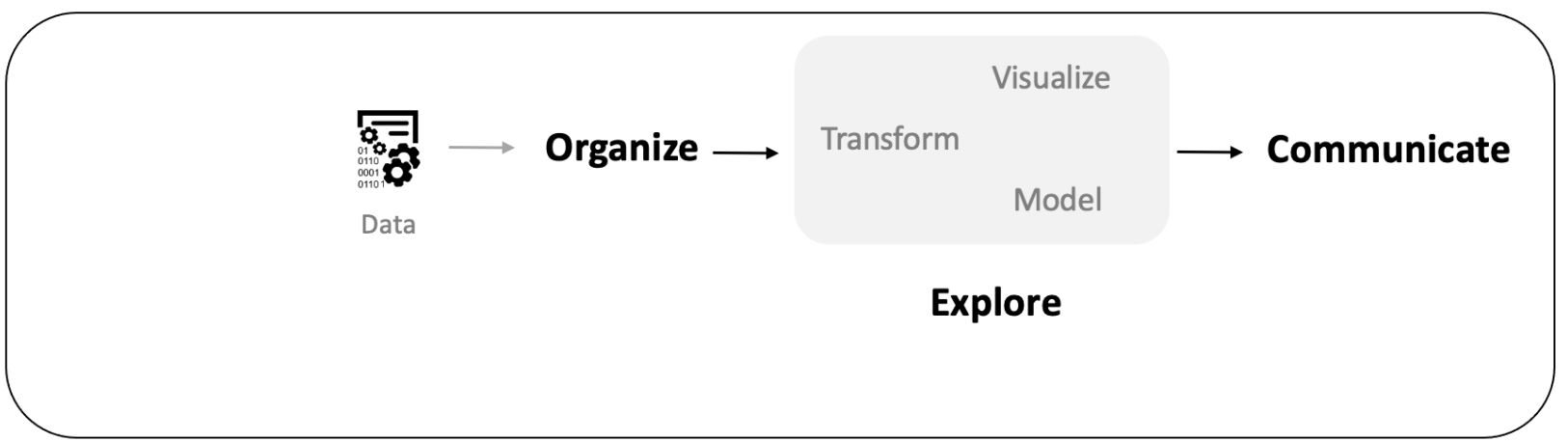
Notice that there is no way to perfectly represent the image through the annotations. We are always both missing information present in the image and making assumptions (and potential errors) in the way that the construct the annotations.



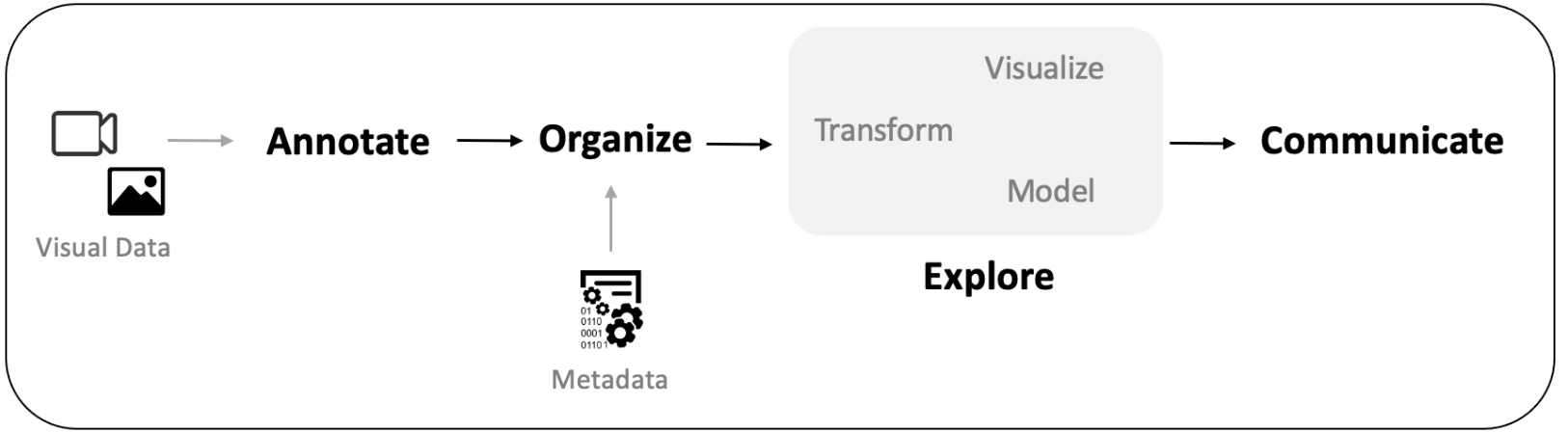
1.4 Distant Viewing: Theory

We can summarize how the process of working with digital images differs from working with structured tabular datasets that are common in the sciences and social sciences.

We need to *annotate* our data before doing any exploratory work. Also, we need to constantly return to the visual data itself to ensure that our annotations capture the information that we are interested in studying.



Data Science Pipeline (Structured Data)



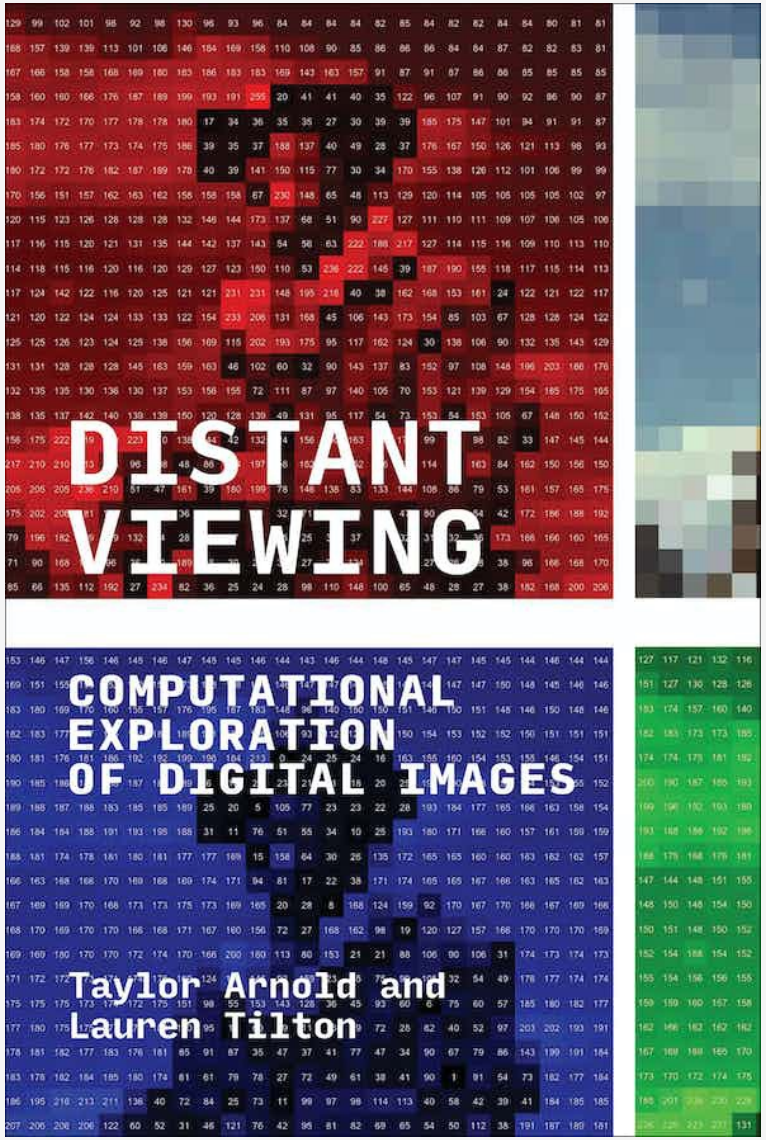
Pipeline for Distant Viewing

1.4 Distant Viewing: Theory

The *theory of distant viewing* combines the observations here with scholarship in semiotics and visual culture to present a theory of how computational exploration of digital images through the application of computer vision works, and why it is needed. We can summarize these results by the following:

1. Mimetic nature of images necessitates that the computational analysis of digital images starts with the extraction of semantic metadata ("annotations") using computer vision.
2. The process of constructing annotations is a computationally mediated form of viewing.
3. The application of computer vision is not a neutral process. It engages in socially mediated ways of seeing and viewing that are encoded into the algorithms and the ways that they are used.

For a more detailed presentation of the theory, please see *Distant Viewing: Computational Exploration of Digital Images* (MIT Press, 2023), which is available under an Open Access license from our website: **https://distantviewing.org/book**

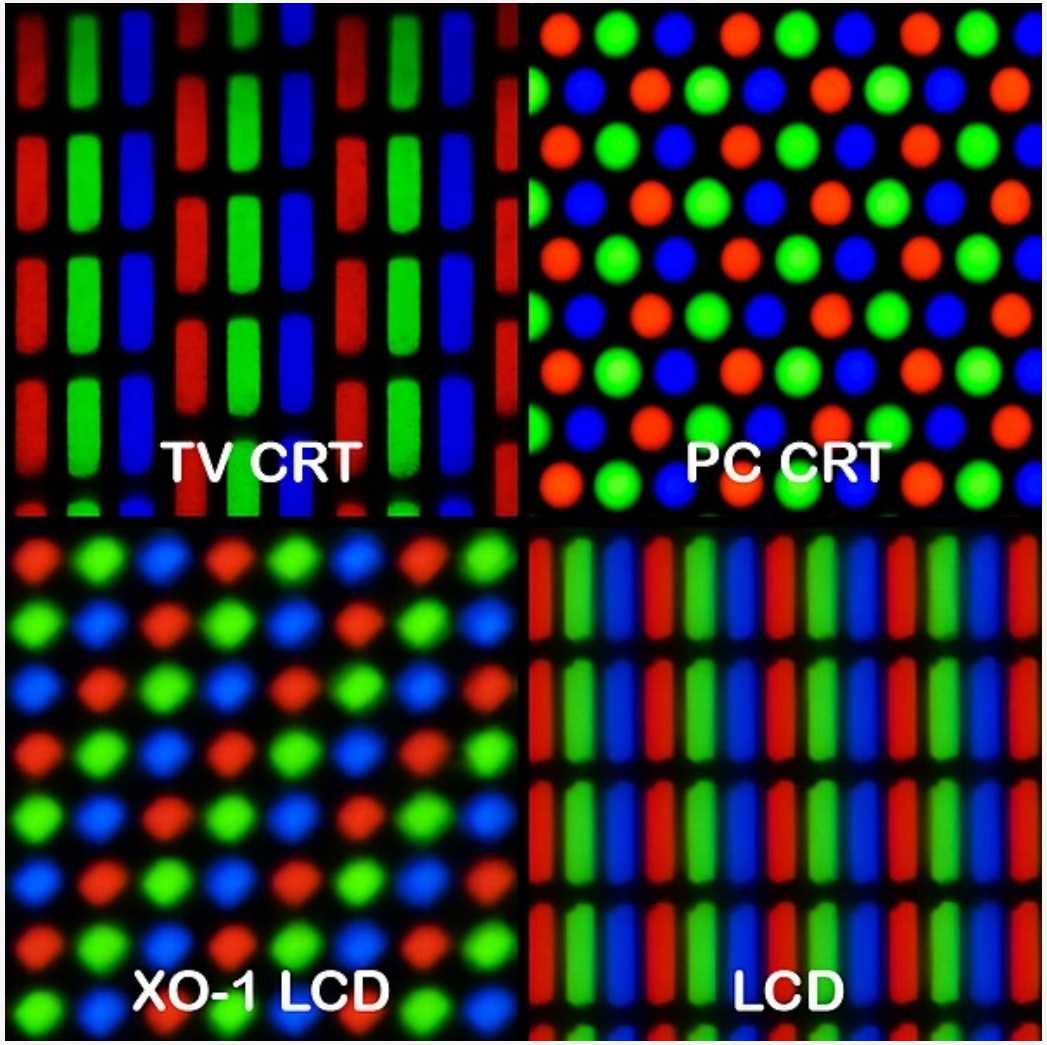


1.5 Annotating Image Brightness

We start with a relatively straightforward annotation that measures the how bright one of our posters is. Before we move to the code consider the following questions:

1. **What changes might there be in the brightness of movie posters over time? What may contribute to these changes?**
2. **How might brightness relate to the genre of a movie poster? Which genres do you expect to be the brightest? Which do you expect to be darkest?**

How should we measure the brightness of an image? Recall that the pixel intensities represent how much red, green, and blue light to use to create a specific color. From this, we see that taking the average pixel values across an image will tell us how much light is used in representing the image, therefore presenting one way of measuring the *brightness* image.

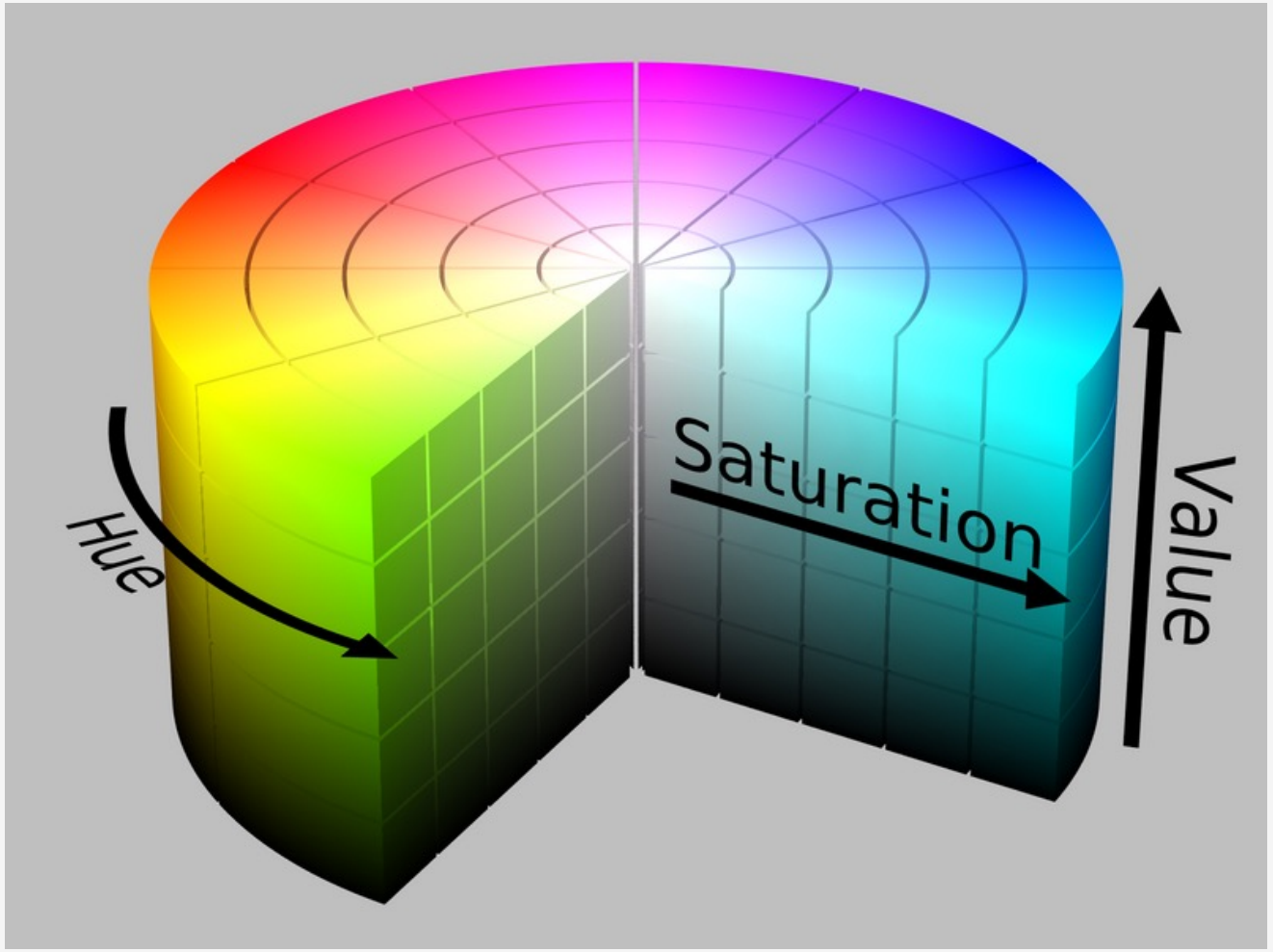


Peter Halasz, CC–SA 3.0, Wikimedia Commons

1.6 Saturation and Chroma

On the right is a visualization of color as hue, saturation, and value (HSV), an alternative way of representing color to the RGB intensities that we have so far been working with.

Saturation is 0 in the middle of the cylinder (shades of grey) and 1 on the exterior of the cylinder. *Value* is an alternative measurement of brightness. We'll return to hue in the next section.

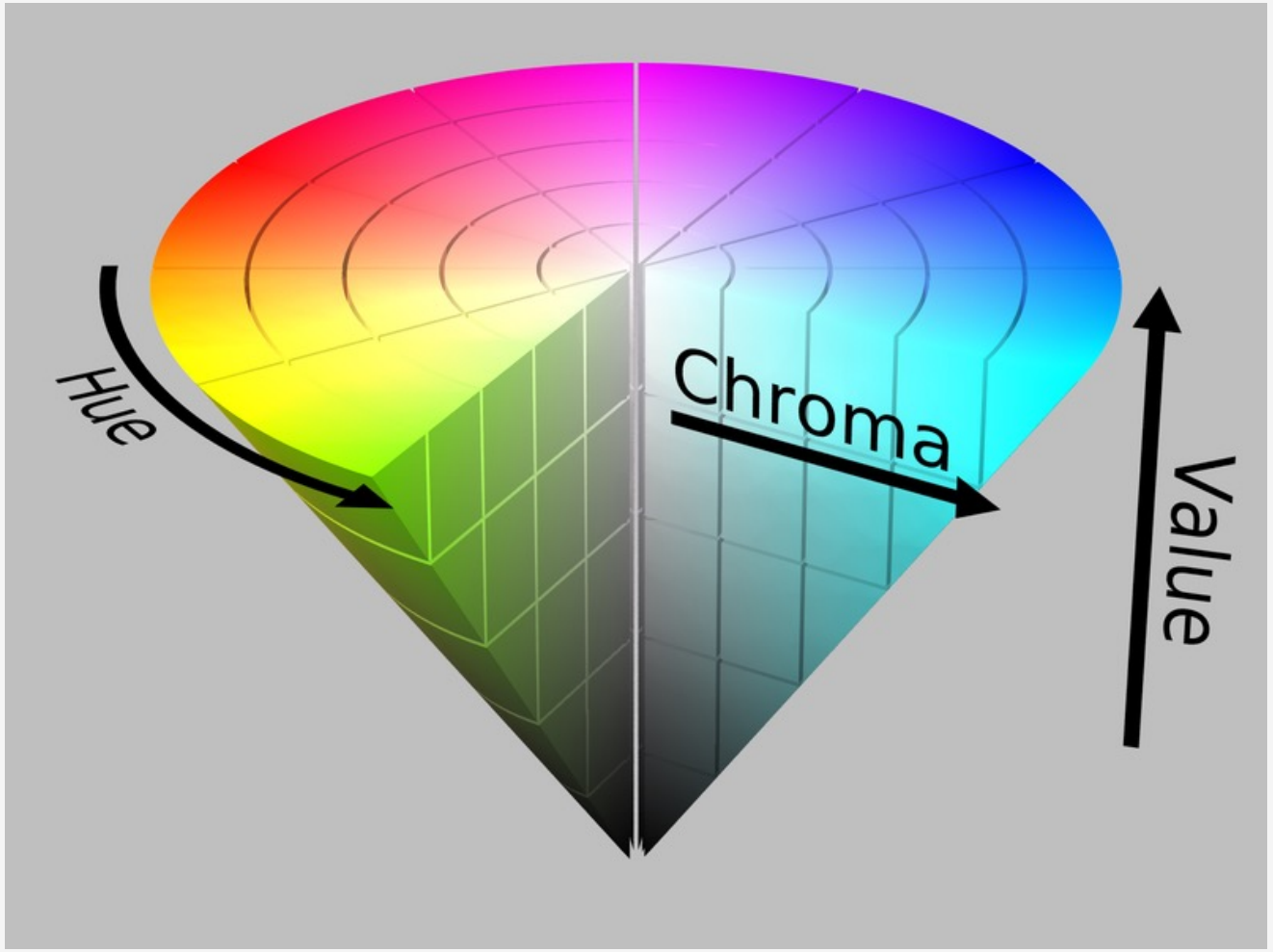


Jacob Rus, CC–SA 3.0, Wikimedia Commons

1.6 Saturation and Chroma

Chroma is an alternative representation of the saturation of a color. It can be geometrically visualized by pinching the bottom of the HSV cylinder to form a cone.

We work with chroma in our analysis because we think it better represents the richness of a color. The choice between these two measurements is a good example of the kinds of subtle subjective decisions that influence applications of distant viewing and need to constantly be accounted for in our analysis.

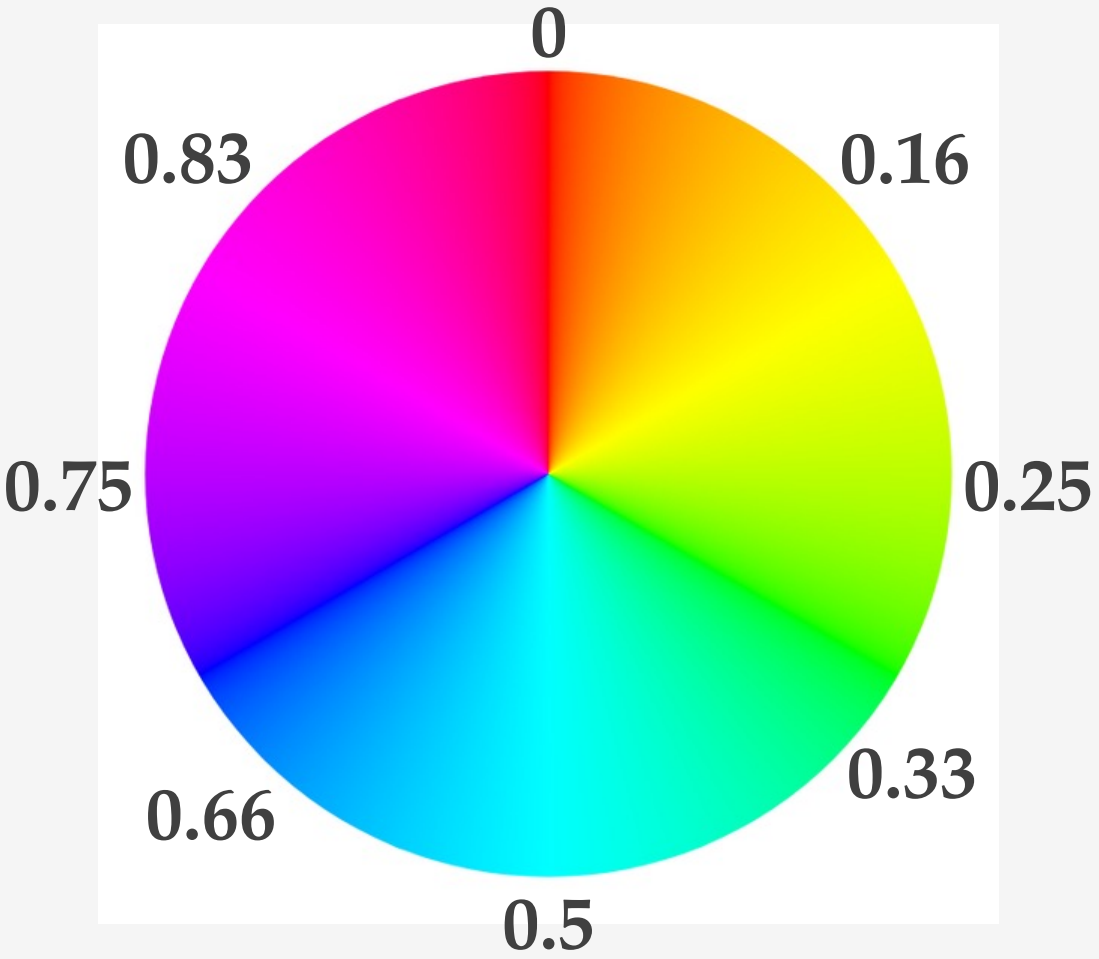


Jacob Rus, CC–SA 3.0, Wikimedia Commons

1.7 Dominant Color

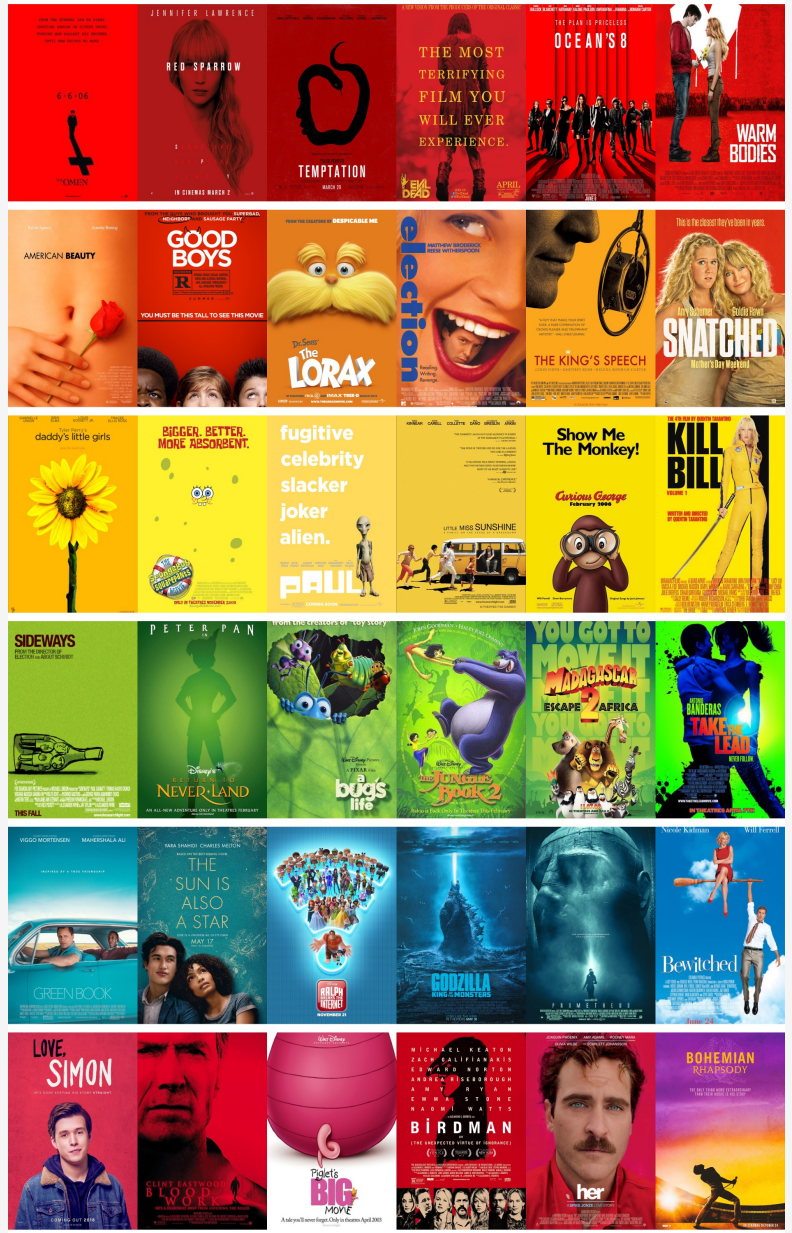
The *hue* of a color represents the common name that we would typically give to a color, such as red, green, or purple. We will use a number between 0 and 1 to represent hue, corresponding to the numbers placed on the color wheel on the right.

Notice that hue is a circular measurement, with values close to 0 being a similar shade of red to values near 1. This means that it rarely makes sense to take average values when working with hue.



1.7 Dominant Color

The Python notebook contains code to find the posters that have the most amount of any given dominant color name according to the buckets that we defined. For reference, on the right are the top six posters most associated with each hue.



As we look at the posters and think about cultural significant of color, consider the following questions before looking at the aggregative results:

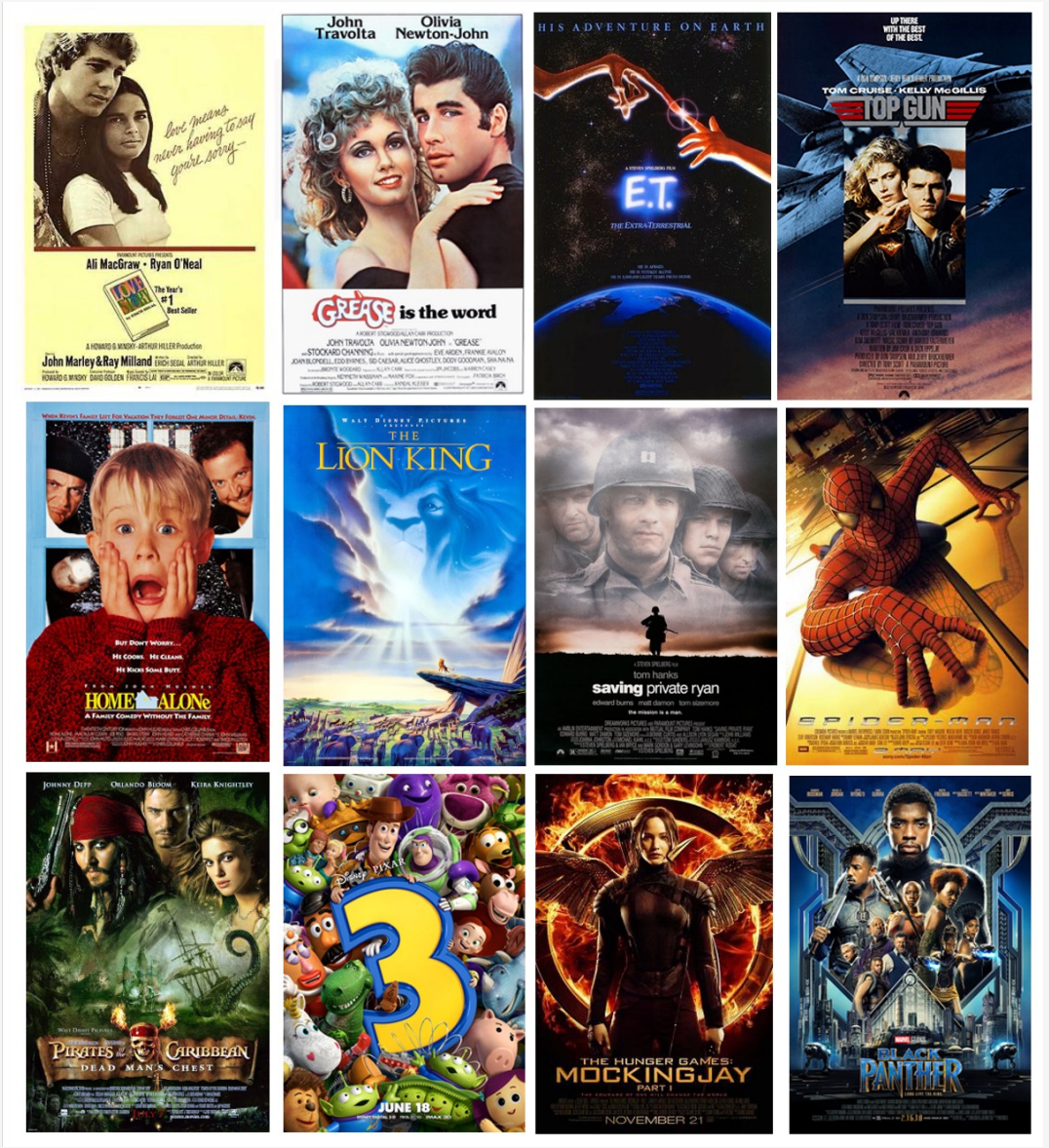
1. **What connections to you expect exist between individual colors and genre? Do you expect specific colors to be frequently indicative of a specific genre?**
2. **What colors do you think are most commonly used in movie posters? Which colors do you think are the least used?**

Then, run the code in the Python notebook to see how your hypothesis match the results.

1.8 Face Detection

Looking again some of the example movie posters, we see that one common feature of movie posters are images of one or more main characters from the movie. As we did with color, let's start by thinking about some questions about how the faces convey meaning through the movie poster:

1. **What kinds of information do you think is being conveyed through use of character images on posters?**
2. **What kinds of genre-specific patterns do you expect there to be between the number and placement of the faces on the poster?**
3. **Do you think there are changes over time between the number and size of the faces on the poster?**

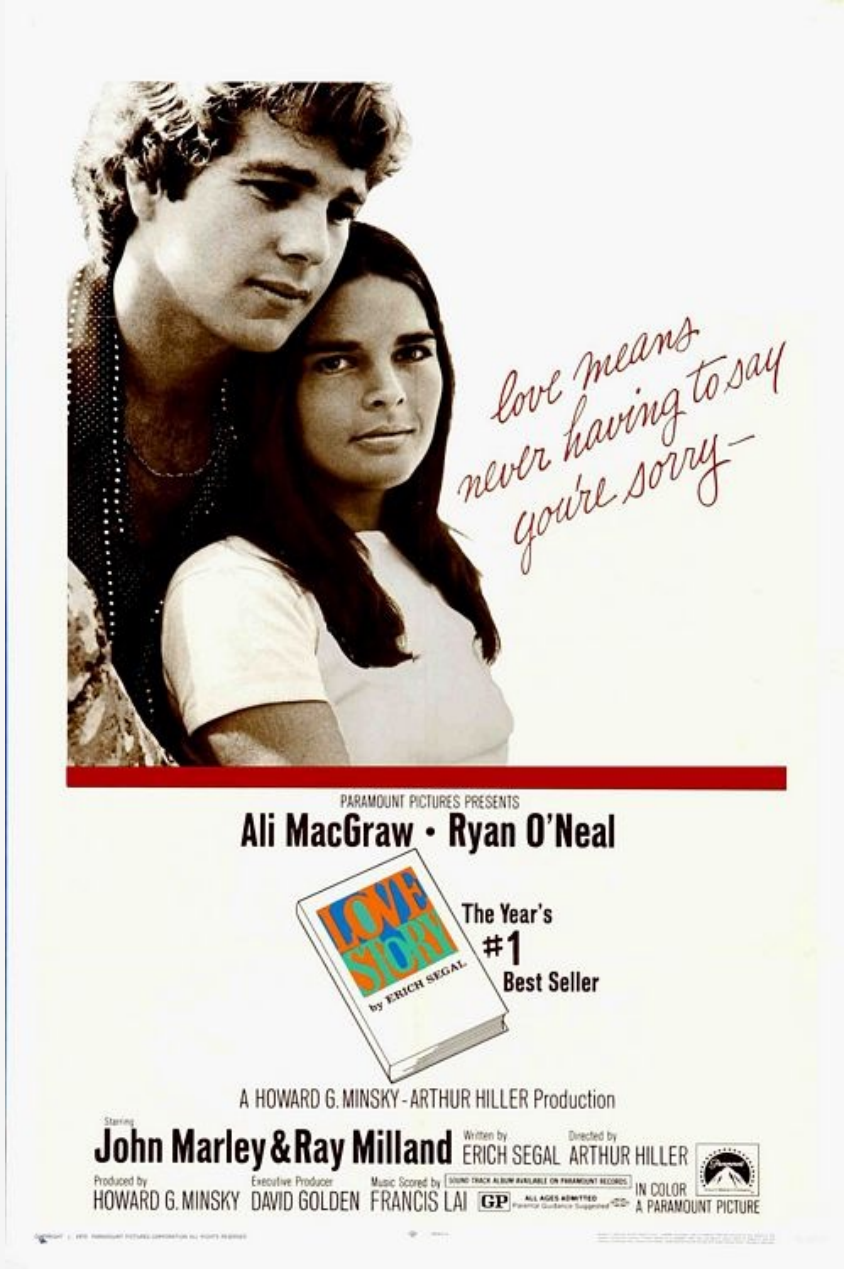


1.8 Face Detection

In order to study these questions, we need to create new annotations that show the location of the faces in each poster. Unlike the color analyses that we have worked on so far, the detection of faces cannot be done by simple mathematical summaries of the pixel intensities. A more complex computer vision algorithm is needed.

This is where the distant viewing toolkit (**dvt**) becomes particularly powerful by allowing us to quickly download and apply a modern computer vision algorithm to detect faces in an image.

As shown in the notebook, for example, we are able to find the two faces of the characters shown in the movie poster for the movie *Love Story* (1970).



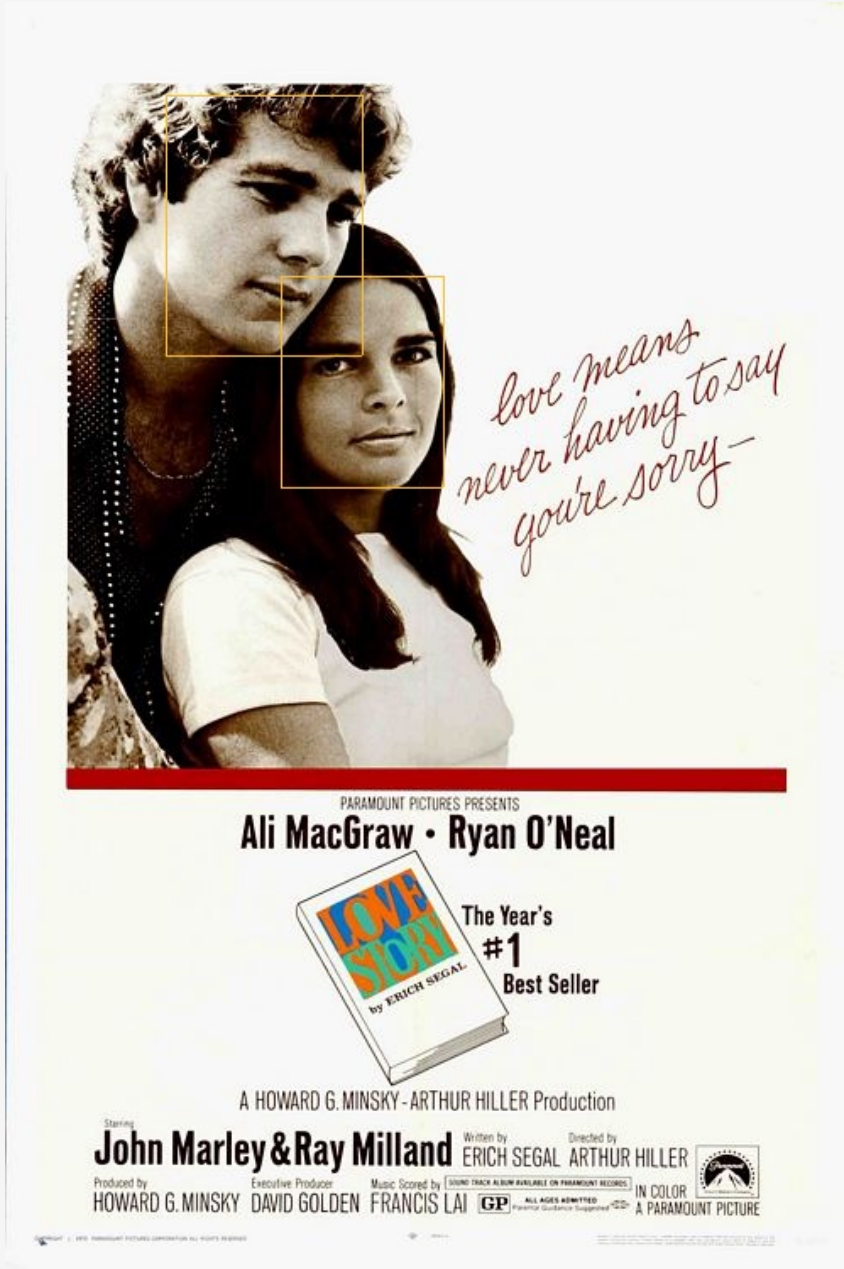
1.8 Face Detection

The output of the annotation can be seen visually in the image on the right, which can also be created within the Python notebook.

We also can represent the detected faces as a structured dataset, as shown here:

	face_id	x	xend	y	yend	prob
0	0	95	211	55	209	0.999969
1	1	163	259	162	287	0.999953

These show the locations of the faces in pixels from the top-left corner of the image along with a probability showing how confident the algorithm is in its determination that it has found a face.



1.9 Conclusions and Next Steps

There are many directions to explore after following these notes:

1. To see how to complete the analysis presented here, check out Chapter 3 of the *Distant Viewing*.
2. If you are new to programming, see the links in the notebook to learn a bit more about working with Python.
3. To learn more about the use of computer vision algorithms and how to work with moving images, see the second tutorial, which works through an example using a set of U.S. television shows from the 1960s and 1970s.

Thank you for taking time to engage with us. Please reach out with any questions or feedback!

https://distantviewing.org

tarnold2@richmond.edu
ltilton@richmond.edu