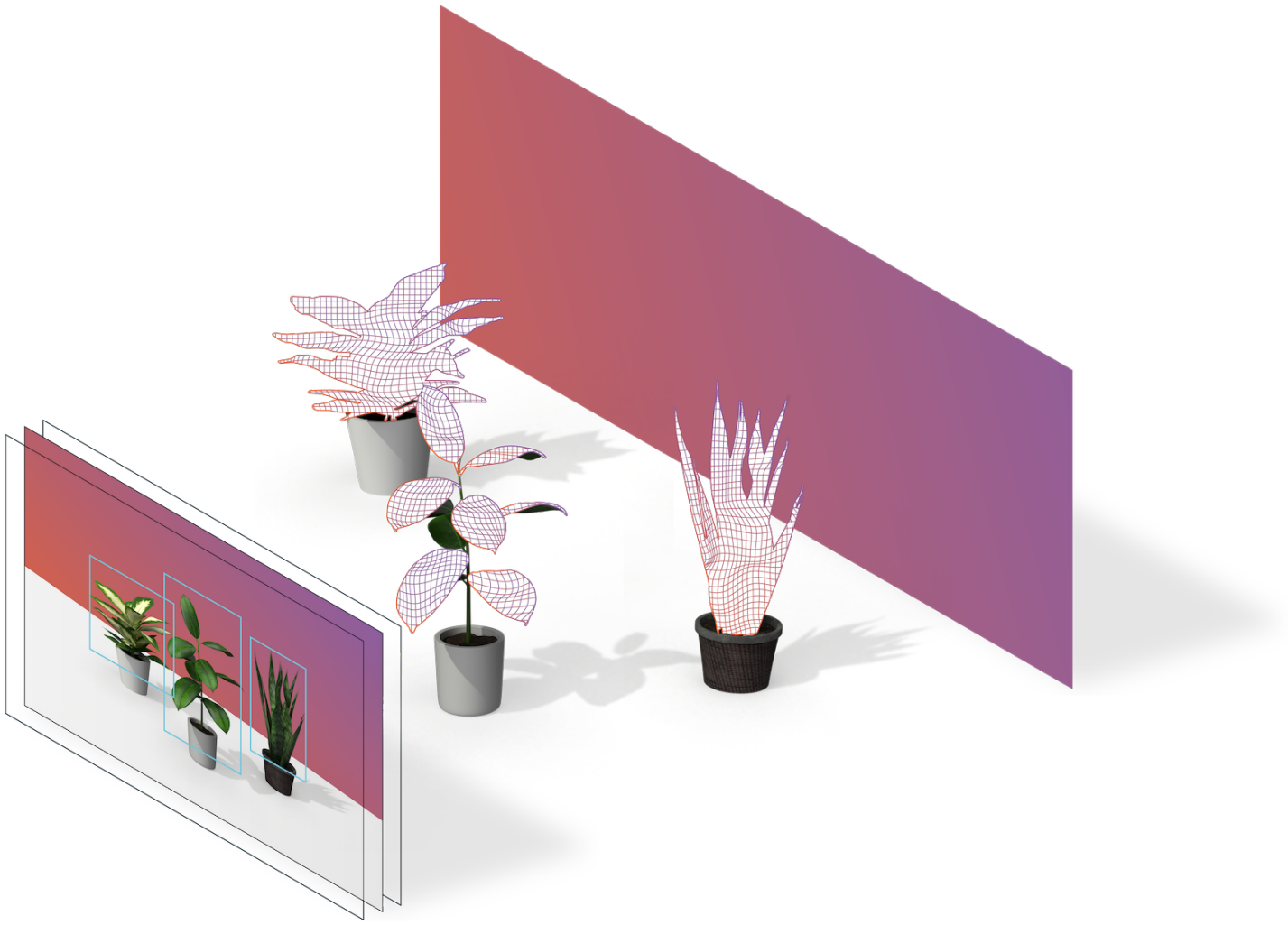
A report from

cdt | Research



APPENDIX

Do You See What I See?

Capabilities and Limits of Automated Multimedia Content Analysis

Carey Shenkman
Dhanaraj Thakur
Emma Llansó

May 2021

The Center for Democracy & Technology (CDT) is a 25-year-old 501(c)3 nonpartisan nonprofit organization working to promote democratic values by shaping technology policy and architecture. The organisation is headquartered in Washington, D.C. and has a Europe Office in Brussels, Belgium.

CAREY SHENKMAN

Carey Shenkman is an independent consultant and human rights attorney.

DHANARAJ THAKUR

Dhanaraj Thakur is the Research Director at CDT, where he leads research that advances human rights and civil liberties online.

EMMA LLANSÓ

Emma Llansó is the Director of CDT's Free Expression Project, where she leads CDT's work to promote laws and policies that support Internet users' free expression rights in the United States, Europe, and around the world.

A report from

cdt | Research

APPENDIX

Do You See What I See?

Capabilities and Limits of Automated Multimedia Content Analysis

Carey Shenkman
Dhanaraj Thakur
Emma Llansó

WITH CONTRIBUTIONS BY

DeVan Hankerson, Hannah Quay-de la Vallee, Samir Jain, and Tim Hoagland.

ACKNOWLEDGEMENTS

We thank Robin Burke for his feedback on sections of this paper. We also thank the various experts from academia, industry, and civil society that we interviewed and who helped inform the analysis in this paper.

This work is made possible through a grant from the John S. and James L. Knight Foundation.

Suggested Citation: Shenkman, C., Thakur, D., Llansó, E. (2021) Do You See What I See? Capabilities and Limits of Automated Multimedia Content Analysis. Center for Democracy & Technology.

IV. Appendix: Automated Multimedia Content Analysis Techniques

Matching Models - Cryptographic Hashing

Cryptographic hashing functions create a string of numbers and letters called a hash, which almost uniquely identifies a file. Similar cryptographic functions are also used to encrypt data in applications like e-mail, Signal or WhatsApp texts, or certain file storage mediums, and are meant to assure recipients of the authenticity of a message or file, down to the last bit. Cryptographic hashing uses a cryptographic function to generate a random hash fingerprint. The cryptographic component makes these functions generally "non-smooth" and extremely sensitive to change. This means even miniscule alterations in the input data will drastically change the resulting hash. For example, changing the shade of one pixel in a high-resolution photo would produce a distinct cryptographic hash. Cryptographic functions are also highly collision-resistant, meaning different pieces of content will produce very different hashes so the likelihood of two different pieces of content producing the same hash (or "colliding") are incredibly low (Engstrom & Feamster, 2017).

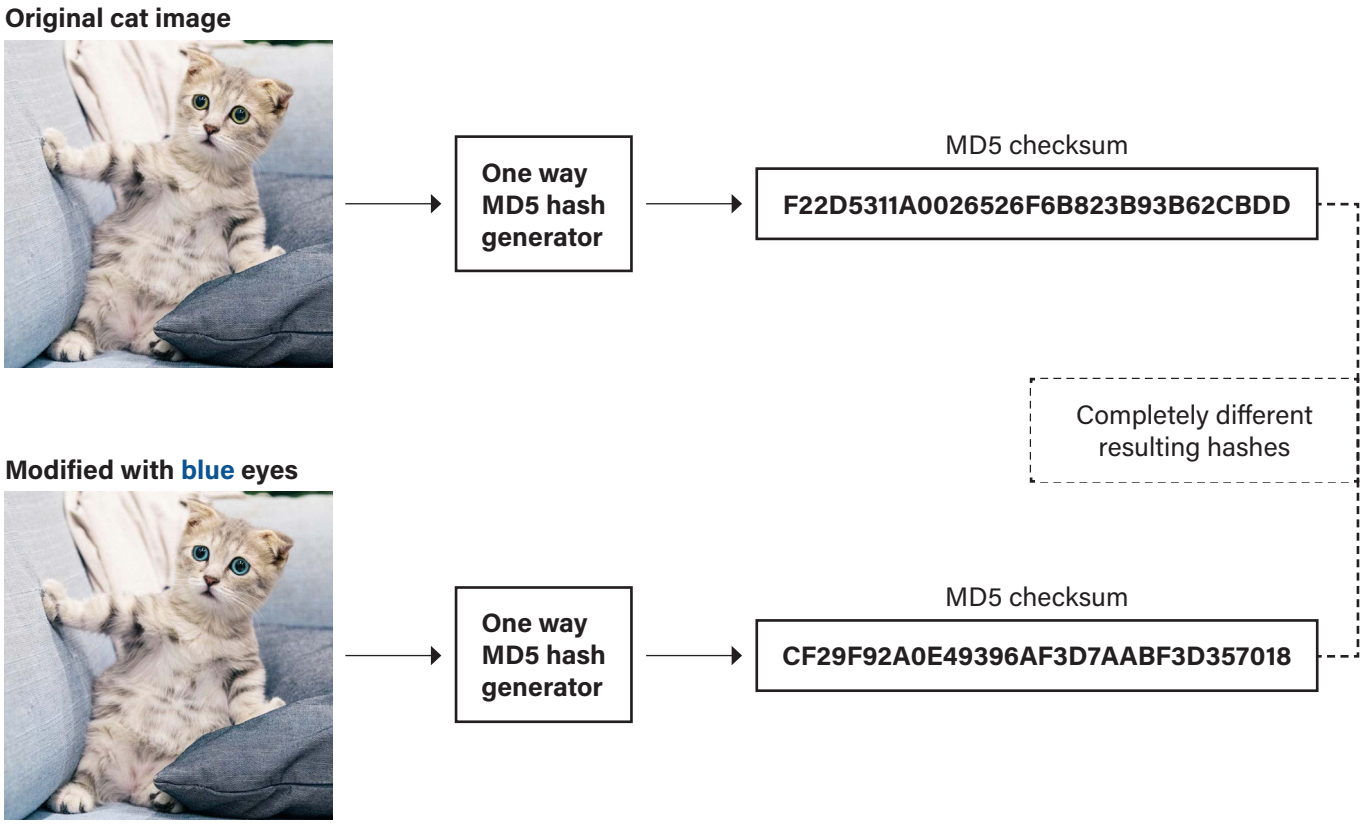


Figure 8. An example of how small changes in input data can lead to very different results in cryptographic hashing. This graphic has been recreated, and based on one by Rosenbaum, K. (2017, June 26). Cryptographic Hashes and Bitcoin, Grokking Bitcoin, Manning Publications. Retrieved December 17, 2020 from https://freecontent.manning.com/cryptographic-hashes-and-bitcoin/.

Cryptographic hashing is highly effective in authenticating known content without alterations. This leads to its primary drawback in its use for automated content analysis, which is its lack of robustness, meaning it is not resistant to minor data distortion. Substantially identical pieces of content may hash very differently. This is particularly problematic in use cases that are adversarial in nature—i.e. an attacker tries to circumvent a hash-based filter and modifies content such that it produces a different hash. Alterations might also occur naturally, simply through the routine transfer of data which may utilize compression to save bandwidth and space. Most modern content sharing systems apply some form of post-processing which would, by nature, change the bits of the file and thus the output of the cryptographic hash. Ideally, a matching system would be *input invariant*, which means that small alterations in input would produce little or no change in the hash.

Matching Models - Perceptual Hashing

Perceptual hashing seeks to determine not whether two pieces of content are identical, but whether they are "alike enough"—i.e. practically identical. For example, if you were shown two photos of the same person, except one single hair in one photo were a slightly different shade, you would likely not notice this miniscule change and consider the photos the same. However, a cryptographic hashing method would consider these completely different. The goal of a perceptual hashing method would be to recognize these as fundamentally the same photo. Perceptual hashing methods aim to better comprehend the nature of a piece of content so that the machine cannot be fooled by imperceptible or non-meaningful changes, such as rotations, resizing, orientation flips, noise, delays in audio or video, or watermarking. Some of these changes might be naturally occurring, or others may be human-designed efforts to circumvent detection.

Some of those methods include ones based on invariant features, local feature points, dimension reduction, and statistics features. (Du et al., 2020).

Perceptual hashing methods involve various methods of pre-processing content, hashing it, and using metrics to compare how alike two pieces of hashed content are. A threshold can be set to determine what degree of difference between hashes is allowed to still consider them matches. Modern perceptual hashing methods apply a range of techniques, including different approaches to create hash fingerprints. For example, by applying a grid and analyzing relationships among pixels in each square, the hash comparison is able to recognize the underlying similarity of the images (see Fig 9).

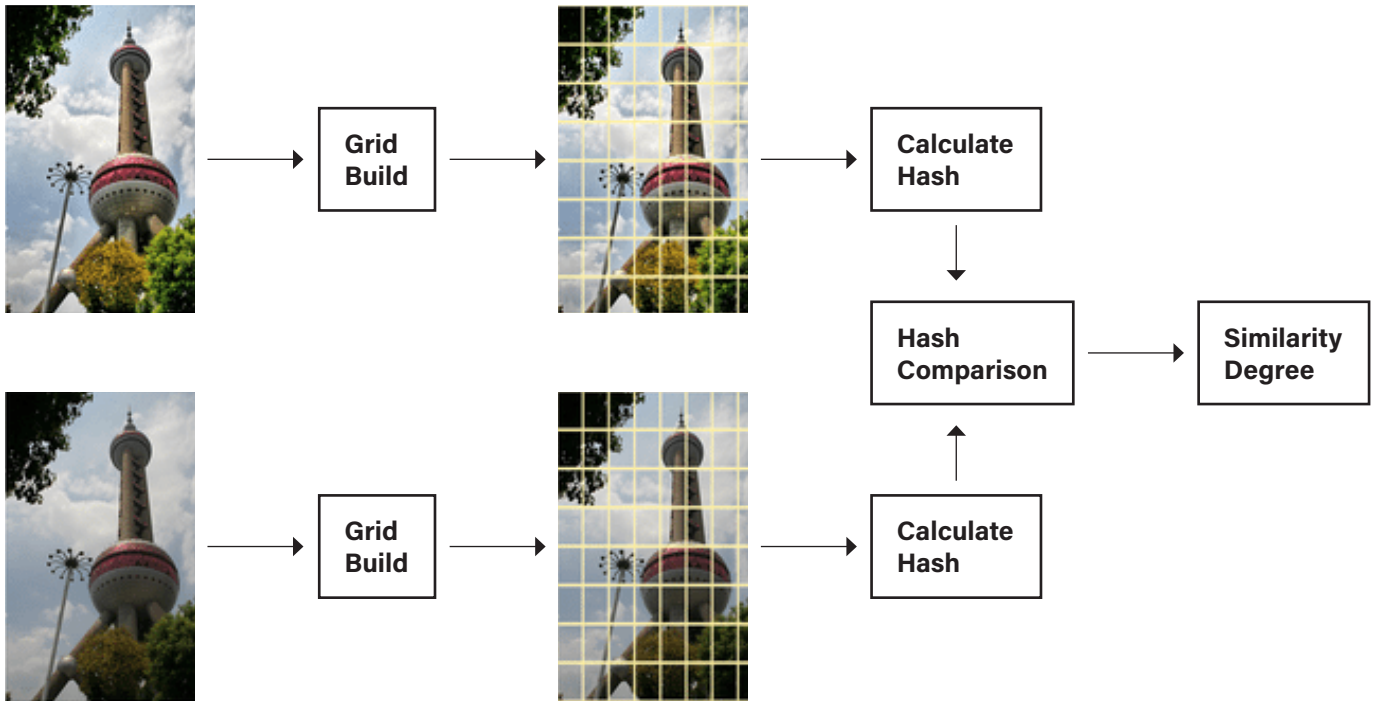


Figure 9. An overview of a hash comparison process of two versions of the same photo but with different levels of color saturation. Source: Souza et al. (2018). https://www.researchgate.net/profile/Veronica-Teichrieb/publication/325521472_Generating_an_Album_with_the_Best_Media_Using_Computer_Vision/links/5b2179a6458515270fc6da3e/Generating-an-Album-with-the-Best-Media-Using-Computer-Vision.pdf.

Perceptual hashing methods offer more flexibility than their cryptographic counterparts. For instance, they can be capable of identifying content that is hidden within other pieces of content, such as a video that is masked within another video (Langston, 2018). In order to evolve as attackers evolve, perceptual hashing functions may utilize techniques like deep learning and convolutional neural networks (discussed in more detail in Box 1) in order to adaptively identify manipulation methods and features. Such methods have shown promise, with the ability to distinguish between substantively distinct images, while also not being fooled by superficial changes (Jiang & Pang, 2018). Some specific implementations of perceptual hash algorithms include systems designed to detect child sexual abuse material (CSAM), terrorist propoganda, and copyrighted content.

CHILD SEXUAL ABUSE MATERIAL (CSAM)

Perceptual hashing has been the primary technology utilized to mitigate the spread of CSAM, since the same materials are often repeatedly shared, and databases of offending content are maintained by institutions like the National Center for Missing and Exploited Children (NCMEC) and its international analogue, the International Centre for Missing & Exploited Children (ICMEC) (Lee et al., 2020). PhotoDNA, developed by Microsoft, is presently the most widespread perceptual matching method for countering CSAM. At a high level, it works by first converting a full-resolution color image to grayscale, then downsizing it to 400 x 400 pixels. A filter is applied, the image is partitioned, and then measurements are extracted onto feature vectors which are compared using a distance metric. PhotoDNA for video applies a similar method to certain video "key frames" (Langston, 2018). More specific information about the PhotoDNA algorithm and the NCMEC database are not publicly available, due to concerns that attackers would use that information to circumvent these protections; however, this lack of transparency also closes off avenues for independent audits and review.

Facebook has open-sourced its PDQ and TMK+PDQF algorithms for image- and video-matching, respectively (Davis & Rosen, 2019). PDQ, based on an algorithm called pHash, stores and compares the outputs of 16 x 16 transformations of images. Other perceptual applications in CSAM include CSAI Match, a proprietary hash-matching technology developed by YouTube, which is utilized by Adobe, Tumblr, and Reddit. Google released an open-source Content Safety API, an AI-powered tool grading the severity of disturbing images, with the Internet Watch Foundation (Todorovic & Chaudhuri, 2018). New methods propose purely metadata-based analysis (meaning they work without examining the actual content of a file) using file paths, which could augment perceptual hashing methods in the fight against CSAM (Pereira et al., 2020). In practice, companies may use a combination of these and other automated tools to detect CSAM on their networks.

See for example the tools used by Pornhub to detect CSAM and non-consensual content. **https://help.pornhub.com/hc/en-us/articles/1260803955549-Transparency-Report/**. Accessed April 2021.

TERRORIST PROPAGANDA

The Global Internet Forum to Counter Terrorism (GIFCT), a consortium founded by Facebook, Microsoft, Twitter, and YouTube and now operating as an independent entity, maintains a shared industry hash database of what they view as terrorist and violent extremist content. Individual companies that are members of the consortium may, depending on the nature of their participation, contribute content they deem to include terrorist propaganda to be catalogued in the shared database. This shared database is not available for independent review or audit.

Joining the consortium involves signing an NDA, MOU, and obtaining licenses to use hashing techniques. As a result, the technical workings of the SIHD are not publicly known. See **https://gifct.org/**. Accessed March 2021.

eGLYPH is another hashing algorithm for terrorist content created by Hany Farid, the same researcher who developed the original PhotoDNA technology. eGLYPH operates very similarly to PhotoDNA, involving the grayscale conversion of images and down-sizing to 400 x 400 fixed resolution. The algorithm can be used to find videos as well, by filtering out redundant frames to reduce length and file size, and then producing arbitrary-length hashes which can be compared using a "longest common substring." Longest common substring is a technique for comparing how similar two alphanumeric strings are by finding the longest contiguous stretch the two strings have in common. For example, consider the random strings "982tiu3hhuiuh" and "293rr928iu3hhu2tiu." These strings have the common substrings "982," "2tiu," and "iu3hhu." Because "iu3hhu" is the longest of these substrings at six characters long, that is the strongest point of similarity and thus the string used to score the strength of similarity between the two longer strings. This approach can also be used to compare audio files (Counter Extremism Project, 2018; Greenemeier, 2017).

COPYRIGHTED CONTENT

Copyright-enforcement tools seek to match user-uploaded content to instances of known, copyrighted content. Perceptual methods are often useful in these efforts, since pirated content might add modifications or watermarks to avoid identification. An example of one tool is the Echoprint API, an open-source fingerprinting library utilized by Echo Nest, a subsidiary of Spotify (Ellis & Whitman, 2013). Echoprint contains three components: 1) a code/fingerprint generator; 2) a query server that stores codes to match against; and 3) codes themselves that are used to match against the fingerprints of any given audio files. Specifically, Echoprint creates time/hash pairs based on relative timing between beat-like onsets, and identifies pieces of audio via these pairs. The fingerprint is based on the relative locations of these onsets (*Welcome to Echoprint*, n.d.).

Another example of a similar fingerprinting technology is YouTube's Content ID, which allows rights holders themselves to create fingerprints of their multimedia (Engstrom & Feamster, 2017). The company Audible Magic produces matching systems utilized by major entertainment studios (Universal Music Group, Warner Bros., Sony, and Disney), as well as platforms such as Facebook, Soundcloud, Twitch, and Tumblr. Audible Magic holds numerous patents in perceptual fingerprinting and automated content recognition, including methods for creating unique audio signatures via segmentation. While its methods are proprietary, those patents indicate that it utilizes principles analogous to segmentation and fingerprinting of spectrograms (visual representations of a spectrum of frequencies in a piece of audio).

Content ID was originally licensed by YouTube in 2006 from Audible Magic; after Google acquired YouTube, it acquired a trademark for "Content ID," after which Audible Magic sued Google over use of the term (Sanchez, 2017).

OTHER APPLICATIONS

Matching algorithms may appear in any case where an organization wants to blocklist content and flag that content when it appears. For instance, online social matchmaking services like OkCupid have utilized perceptual hashing algorithms to scan for re-uploads of banned profiles (Jablons, 2017). Facebook, too, utilizes a large-scale matching infrastructure called SimSearchNet/SimSearchNet++ on "every image uploaded to Instagram and Facebook" to scan against an existing curated database of "misinformation," including COVID-19 misinformation (Facebook, 2020). Amazon utilizes audio fingerprinting to prevent mentions of the word "Alexa" in advertisements from mistakenly triggering Alexa devices and resulting in negative customer experiences (Rodehorst, 2019).

For more background on the audio fingerprinting techniques mentioned here, see Haitsma & Kalker (2003).

Box 1.
Deep Learning as the Foundation for Predictive Models

Deep learning is an attempt to solve complex computational problems by replicating the structure of the human brain. The result are structures called **artificial neural networks (ANNs)** that can "learn" from very large quantities of data. The basic function of ANNs is to ascertain features from inputs. For example, an ANN may learn what features of an image represent a flower, by analyzing millions of images of flowers and non-flowers. ANNs contain layers of functions (called "nodes" or "neurons") which perform various operations on the data that they are fed. The "deep" of deep learning refers to a network having many, many layers, most of which are hidden. ANNs are an *umbrella* and can contain many different types of neural networks.

Think of ANNs (very roughly) like an incredibly large imaginary car factory, larger than any currently on earth, where upwards of millions of workers process smaller components of a very complex car. Assembly of the finished project will typically be broken down into a multitude of sub-tasks. Teams of workers with specialized skills build upon the output of other workers within dedicated teams and may connect with other teams as needed. The outputs of these steps may not, by themselves, look anything like the finished product, much as an ignition coil may not be immediately recognizable as a car part (even to regular users of cars). During the process, tasks and workflow may also be shifted in real-time to make the process more efficient. Thus, someone walking through this factory would likely find it impossible to grasp the immensity of the process or the relationships between various teams and processes.

ANNs can be structured in a variety of ways. One type of ANN, a fully-connected neural network, is good at making classification decisions of simple data. This means each node in a layer is connected to all the nodes in the next layer. However, fully-connected networks suffer from computational inefficiency because they are *dense*. If the first layer contained 1,000 nodes, this would lead to 1 billion parameters after just the first layer, which will increase dramatically with dozens or hundreds of layers, or if color channels are added to the image being evaluated, for example (Elgendy, 2020). This huge number of parameters leads to high computing time, unwieldiness, and overfitting, making ANNs alone ill-suited for computer vision and audition tasks. Another type of neural network, called a convolutional neural network (CNN), seeks to address this issue.

CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks (CNNs) underlie the current most popular method for modern predictive models for content analysis. They utilize *locally connected layers* to attempt to simplify inputs to smaller representations before making classification decisions. Instead of each node being connected to every node in the previous layer and considering the entire input, nodes in a CNN consider smaller windows of the input. CNNs utilize *convolutional layers*, which act like windows (or "kernels") sliding over the input data to extract salient features by applying various *filters*. These filters perform specialized operations such as edge, contour, or corner detection for images (these operations reduce spatial dimension and resolution of the image). Then a *pooling* operation is performed where the results of the high-level features are combined. Like ANNs, CNNs have input layers, output layers, and hidden layers. Predictive models that utilize CNNs often incorporate fully-connected layers or recurrent layers for stages of their analysis.

Here's a walkthrough of a CNN process. Suppose a CNN is used to try to identify that an input image contains a flower. First, CNN layers will apply filters across the image to create a feature map. This means the first layers will extract very rudimentary features like edges and blobs. As these features are combined, an early layer may result in recognizing a rough outline of a flower. Another layer of features may identify a petal, stem, or leaf by their outlines, colors, and textures. Pooling layers simplify the outputs of these various feature maps. They are then "flattened" onto a long vector which expresses the data in a simplified format. These simplified outputs then can be analyzed by fully-connected layers (thus the more computationally expensive part of the calculation is now being done on a much smaller, less expensive, input) which will generate a *prediction* whether the image contains a flower. This prediction is based on the data and the model's training on images containing flowers.

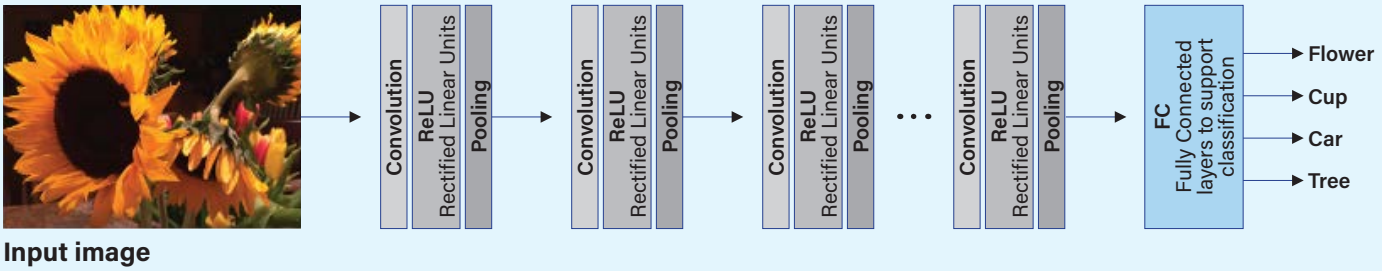
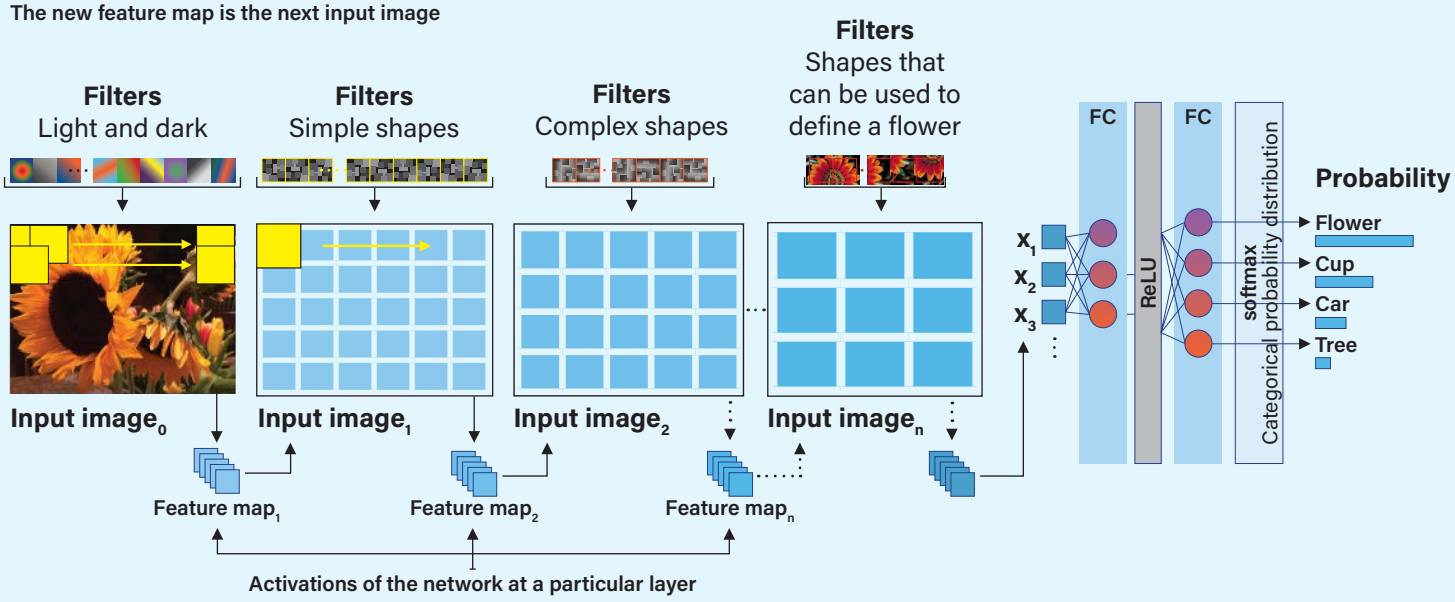


Figure 10. This graphic has been recreated, and based on an illustration of a CNN by MathWorks. Source: Learn About Convolutional Neural Networks. (2020). MathWorks. Retrieved December 17, 2020 from https://www.mathworks.com/help/deeplearning/ug/introduction-to-convolutional-neural-networks.html.



Predictive Models - Computer Vision Models for Content Analysis

Computer vision attempts to solve a multitude of sub-problems, using techniques such as deep-learning models and CNNs. Vision problems involve a complex suite of "building block" tasks from analyzing shapes, textures, colors, spatial arrangement, and static and temporal relationships. Computer vision technology is rapidly evolving with the potential to be "the greatest disruptive innovation in a generation" (McBride, 2020). Examples of various computer vision tasks are summarized below, although this list is non-exhaustive:

Computer Vision Task	Function	Sample Output
Classification	Identifies what is in an image, without determining object location in the image.	Image contains at least one person, a hate symbol, and a sign, with a particular degree of confidence.
Object Detection	Identifies the classification and locations of objects in an image via bounding boxes.	A box-shaped region in an image contains a person, another box-region contains another person, another box contains a sign, and another box contains a hate symbol.
Semantic Segmentation	Identifies, at a pixel-level outline, what space in the image belongs to what categories of objects.	The parts of the image that are perceived as people are shaded one color, parts of the image that are signs are another color, and the hate symbol is another.
Instance Segmentation	Identifies objects using a pixel-level outline, differentiating distinct copies of the same object.	The individual people, sign, and symbol are different colors.
Scene Understanding	Identifies what is generally happening in a scene using geometric and content cues.	The scene depicts a person protesting with a sign containing a hate symbol.
Action Recognition	Identifies, using physical cues, what actions are being taken.	The person is holding the sign. The person is yelling.
Object Tracking	Identifies, in a video, where an object moves over time.	The person is swinging the sign back and forth.
3D Pose Estimation	Identifies, using joint positions, what physical action a person is taking.	The person holding the sign is making offensive gestures.

▲ **Table 2.** Examples of various computer vision tasks summarized.

IMAGE CLASSIFICATION (IMAGE LEVEL PREDICTIONS)

A **classifier** is a computer vision algorithm that indicates what an image contains. Image classifiers are one of the simpler computer vision tools, and they are ubiquitous and among the most common in the multimedia content analysis space (Batra, 2019). A current popular classifier is called ResNet-50, which is a CNN that contains fifty layers, is pre-trained on a million images from the ImageNet database, and can classify according to 1000 object categories.

Classification indicates what predefined categories of objects occur in data. A very basic example of a classifier would be one that *predicts* whether or not an image contains a cat or dog. "Prediction" is a term of art used since outputs are typically accompanied by a *confidence score*, which indicates the degree of certainty with which the algorithm has made its prediction (one can also think of it as a "guess").

Classifiers can achieve state-of-the-art performance across many domains. But they are brittle, meaning they are susceptible to external forms of visual interference and distortions called *perturbations* (Stock et al., 2020). Perturbations might include anything from changes to the intensity of single pixels in an image, to image-wide changes such as noise or blur. These perturbations may be environmental, a product of imperfect image capture techniques, or the result of deliberate efforts to fool an image recognition process.

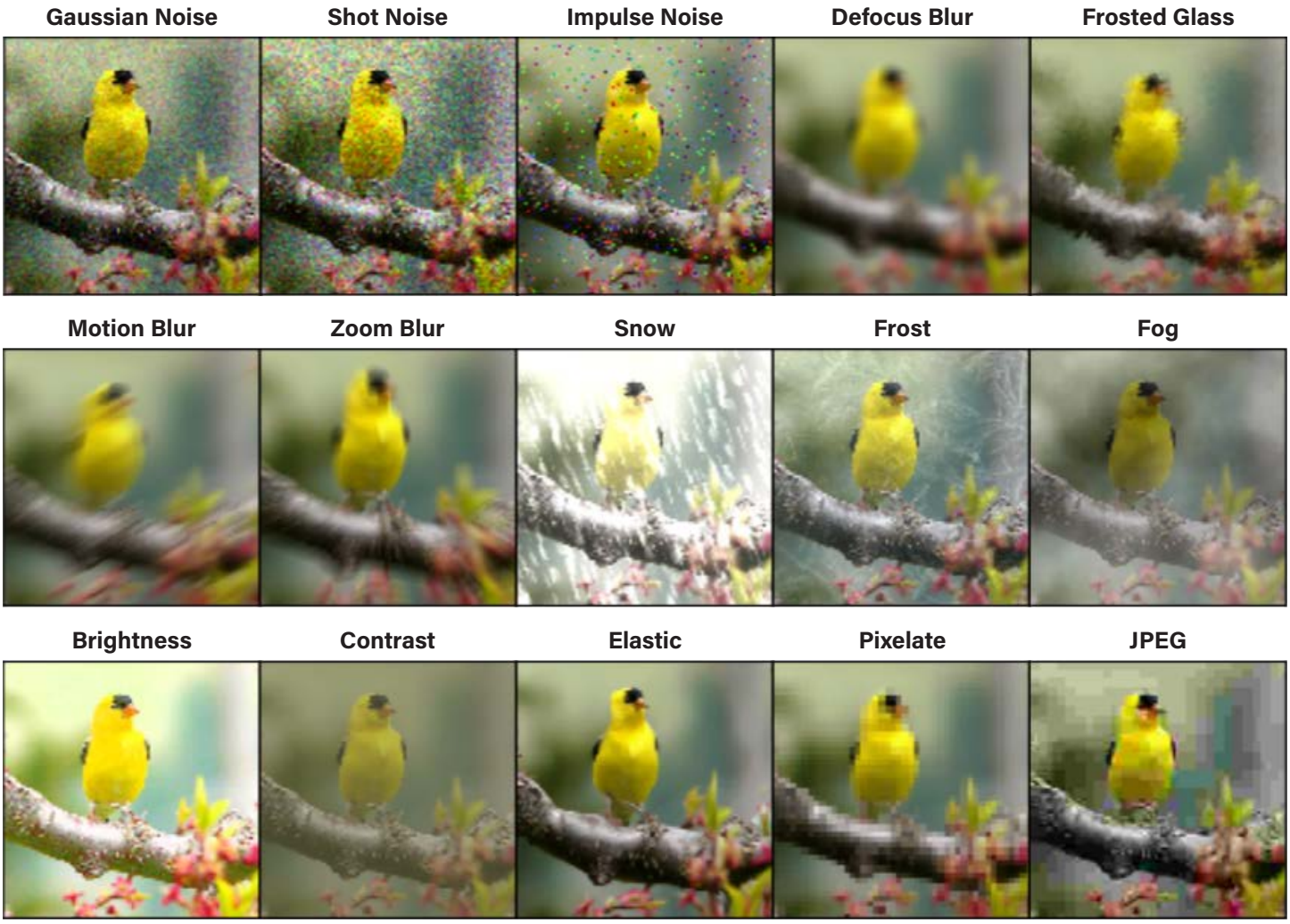


Figure 11. Illustration of image perturbations. Source: (Hendrycks & Dietterich, 2019).

▲ **Figure 1.** Visually similar images: chihuahuas and blueberry muffins, or sheepdogs and mops. Source: https://twitter.com/teenybiscuit/status/707670947830968320 (Accessed March 2021).

Classifiers may also be fooled by images that look very similar to one another but represent different objects, such as chihuahuas and blueberry muffins, or sheepdogs and mops (per our previous example in Figure 1).

OBJECT DETECTION (OBJECT/BOUNDING BOX LEVEL PREDICTIONS)

While classifiers merely identify what is in an image, **object detectors** take on a more complex task, which is localizing one or more objects in an image and classifying those objects. Many industry content analysis tools utilize object detectors. For instance, the Amazon Rekognition Content Moderation API, for images and videos, is a deep-learning based detector. It assigns labels to objects in photos including adult content, violence, weapons, visually disturbing content, as well as drugs, alcohol, tobacco, hate symbols, and gestures, all with associated confidence scores (Amazon Web Services, 2020). Google Cloud's Vision API similarly utilizes detectors to identify explicit content and various objects and expressions. Specialized detectors may be used in content analysis, from gunshot detectors, to blood detectors and others.

The output of a detector is typically a location, denoted by a "bounding box," and the class of the object. An object detection algorithm generally begins by proposing regions of interest (ROIs) and then conducting classification tasks, as discussed earlier, on those individual regions. Since several ROIs might initially cover an object, a process called "non-maximum suppression" is utilized to narrow down which ROI most closely frames a given object.

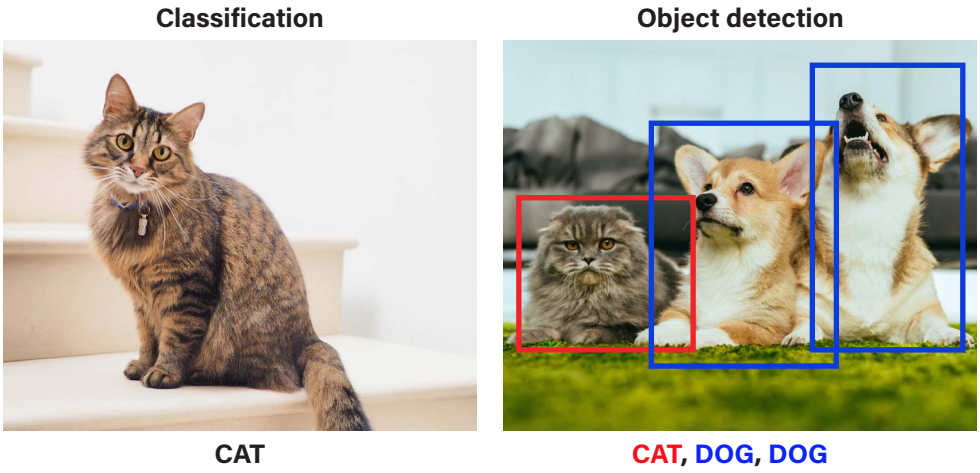


Figure 12. Sample outputs of image classifiers versus detectors. This graphic has been recreated, and based on an illustration by Hulstaert, L. (2018, April 19). A Beginner's Guide to Object Detection, Datacamp. Retrieved December 17, 2020 from https://www.datacamp.com/community/tutorials/object-detection-guide.

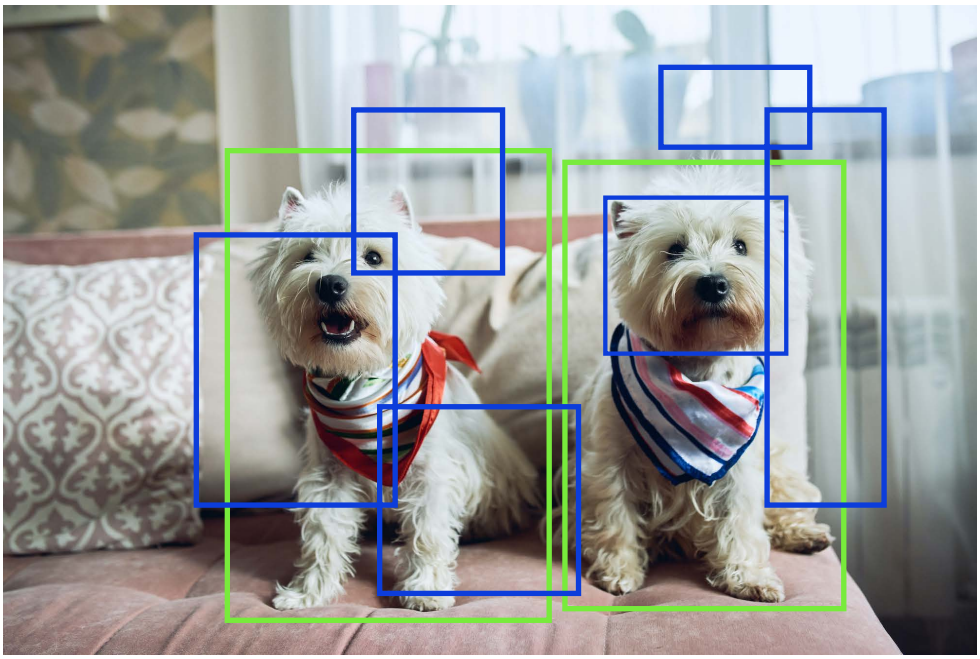


Figure 13. Examples of a proposal process for regions of interest (ROIs). The light green boxes would be the output ROIs because, of all the boxes, they contain the most of a given dog. This graphic has been recreated, and based on an illustration by Chanel, V.S. (2017, September 18). Selective Search for Object Detection (C++/Python), Learn OpenCV. Retrieved from https://www.learnopencv.com/selective-search-for-object-detection-cpp-python/.

For content analysis, detectors may be desirable when the location in an image is relevant for determining its nature. For example, state-of-the-art detectors are being trained to recognize natural disasters such as earthquakes and flash floods, or other emergencies like accidents. These detectors could be used on social media to learn correlations between these events and posting metrics to better respond to emergencies (Weber et al., 2020). Object detection is crucial in analysis of video, which relies on understanding location and movement over time.

Two main evaluation metrics are used to measure the performance of object detectors. Detection speed is evaluated in frames per second (FPS), and network precision is measured via mean average precision (mAP) (Elgendy, 2020). Research shows that detectors generally perform better against efforts to circumvent them than classifiers — "fooling a detector is a very different business from fooling a classifier" (Lu et al., 2017, p. 9). This is because detectors consider a multitude of ROIs around an image, and apply a classification algorithm to each of these. Any circumvention effort must fool all of these boxes, rather than simply one. *Importantly, detectors can come in many forms, and often feature trade-offs depending on the desire for speed or accuracy.*

Improvements to R-CNN include removing the need for analysing separate region proposals (Fast R-CNN) and the use of the selective search algorithm (Faster R-CNN), both of which made computation slower (See Girshick, 2015 and ; S. Ren et al., 2016).

Three of the most popular algorithms for object detection are called R-CNN, SSD (Single Shot Detector), and YOLO (You Only Look Once). R-CNN is the least sophisticated of the three. It first uses a selective search algorithm to identify the most likely regions where the object exists, runs each proposed region separately through the CNN to compute its features, and then uses a classifier to determine what the object is. These steps partly explain why the use of R-CNN architectures is slow and computationally expensive. For this reason they are called *multi-stage* detectors. SSD and YOLO attempt to address the multi-stage issue by being "one shot"—in other words, convolutional layers simultaneously predict whether ROIs contain an object while also conducting the classification step. These detectors are considerably faster, and thus are often used in real-time video or camera applications (Redmon & Farhadi, 2018). However, they tend to be more prone to mistakes than multi-stage detectors.

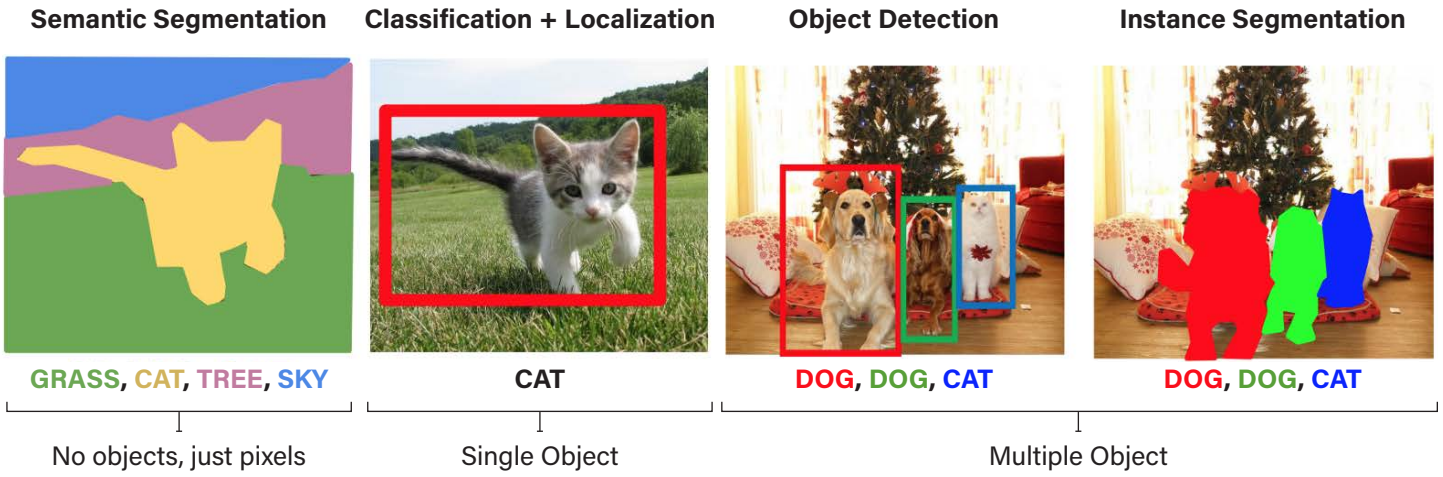


Figure 2. Differences between computer vision tasks. Note that for instance segmentation, the two adjacent dogs are differentiated. In semantic segmentation, these would be the same color and not differentiated. Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf#page=53 (Accessed May 2021).

SEMANTIC SEGMENTATION AND INSTANCE SEGMENTATION

Segmentation tasks are important for content analysis because they are the building blocks for parsing *relationships* between objects in images or video. **Semantic segmentation** seeks to be more granular than detection, by assigning a class label to each individual pixel in an image. **Instance segmentation** seeks to be even more precise and identify individual object boundaries. A popular technique for this is called Mask R-CNN, which is an extension of Faster R-CNN for object detection. It works by generating bounding boxes and then adding a step to produce "masks" or object outlines (Mittal, 2019). **Video instance segmentation** takes this further, where individually segmented instances are then linked and tracked over an entire sequence. For instance, researchers at Facebook developed an approach to instance segmentation to track objects in video sequences using a method called MaskProp. Other state-of-the-art methods in *panoptic segmentation* seek to merge both semantic and instance segmentation into one task (Kirillov et al., 2019).

The simplest architecture for semantic segmentation is the Fully-Convolutional Net (FCN), an encoder-decoder process. In FCN, an input image is down-sampled to a smaller size through a series of convolutions (the encoder), and then that encoded output is up-sampled. Up-sampling can occur via processes such as bilinear interpolation or transpose-convolutions (Long et al., 2015). The encoding process may, however, lead to artifacts and poor boundary resolution. More modern architectures include multi-scale models like the Pyramid Scene Parsing Network (PSPNet), which performs multiple convolution operations of varying dimensions (hence the "pyramid" title) (Zhao et al., 2017).

The MaskProp technique predicts clip-level instances in order to simultaneously classify, segment, and track object instances in video sequences. It is billed as more robust against motion blur and object occlusions in videos (Bertasius & Torresani, 2020).

SCENE UNDERSTANDING

Scene understanding seeks to comprehend a scene by considering the geometric and semantic relationships of its contents (Naseer et al., 2019). Scene understanding algorithms have important applications in content analysis, as they piece together the larger correlations between individual objects. For example, an image containing "fire" might be a campfire or it could be a natural disaster or violent scene. An image containing "blood" might be a gruesome image, or it may be an educational photo of a surgery. Researchers from UCLA utilized scene understanding and visual sentiment analysis to develop a visual model to recognize protesters, describe their activities, and estimate the level of perceived violence in the image (Won et al., 2017). They identified that emotions such as anger and fear were often correlated with perceived violence, and implemented *object detection* of labels such as signs, photos, fire, law enforcement, children, and flags.

Scene understanding is a compound task that involves a number of the aforementioned "building block" tasks. Hence a scene understanding algorithm is not simply one algorithm but involves the application of a number of CNNs: classification; object detection; segmentation; monocular depth estimation; pose estimation; and / or sentiment analysis, among others.

OBJECT TRACKING

The task of **object tracking** in either pre-recorded video or a live stream means following the location of a given object over time. To imagine the difficulty of this, picture being with a friend in a busy crowd. Consider the steps the brain must take to watch a friend moving through the crowd and not lose sight of them. This involves identifying individual humans in the crowd, recognizing the friend among the other humans, and differentiating the friend (or perhaps only one or more features or perspectives of the friend due to obscuration). At some moments the friend may be close or far (Asad et al., 2020). Multiple objects, lighting discrepancies, or temporary disappearances from view are just some of the problems tracking algorithms may face (Nixon & Aguado, 2019).

Video understanding is a significantly more difficult task than identification of objects in static images because it involves a temporal dimension. This dimension creates *dependencies* between various points in time (i.e., the order matters). An example of this is the act of climbing up a ladder, which can appear to be climbing down if an algorithm gets the frame-order wrong. Examples of tasks that may need to occur in video are object tracking, video object segmentation, video prediction, and pose estimation. Many current video analysis tools will approximate videos using specific frames. The Microsoft Azure content moderation system, for instance, divides content into differing "shots" and identifies specific key frames on which to run a static image analysis on whether that image is inappropriate or prohibited content.

Object tracking is utilized for a variety of use cases, such as following the motion of humans or vehicles. One key representation benefitting tracking and motion estimation is *optical flow*, or the pixel-level correspondence between images. These can help ascertain and differentiate forms of movement.

Traditionally, classical methods infer optical flow by minimizing what is called a "loss function." Modern methods utilize unsupervised learning to circumvent the need for labels. These approaches are advantageous because they yield faster results and improved performance. Examples of these approaches include OAFlow and DDFlow (Jonschkowski et al., 2020).

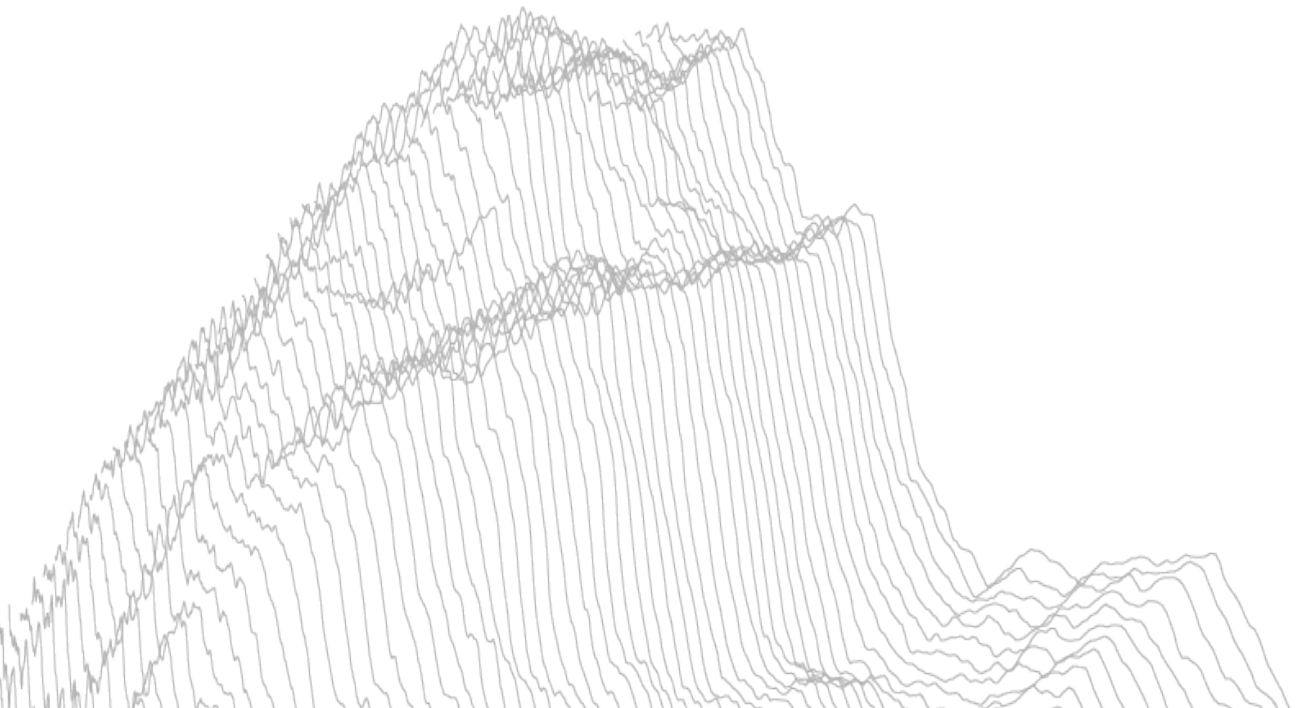
ACTION RECOGNITION AND 3D POSE ESTIMATION

Advances in **action recognition** are a current priority in computer vision, given the volume of video content being produced on devices and platforms. Many action recognition algorithms are highly specialized. Tools may only consider specific subjects at a time. For example, state-of-the-art models in **violence recognition** propose to break down violence into categories such as blood, explosions, fights, fire, and firearms (Peixoto et al., 2019). **3D pose estimation** involves predicting the 3D position of

human joints in images. Most reliable data is obtained using elaborate sensors and bodysuits which is impractical for collecting volumes of data and, importantly, does not exist for data obtained "in the wild" (Pavllo et al., 2019). In light of that, current research focuses on *estimation* of 3D keypoints from 2D images, historically using estimations to reference the pelvis joint. Pose estimation allows for better action recognition, as well as enabling research into human gestures. Audio cues can be combined with gestures to analyze and predict gestures from speech (Ginosar et al., 2019).

ISSUES WITH LIVE VIDEO

Live video presents some of the most challenging problems to content analysis. It requires the application of all of the aforementioned prediction tasks. Not only must the outputs of those tasks be synthesized, but the live component requires them to be done quickly. This is enormously computationally expensive, because videos (especially high resolution ones) are large data files, and hence generally impractical to monitor for social media platforms. Use cases of screening live video for violence, for example, may thus still be far off. Facebook executives, for example, reportedly said that AI may still be years away from being able to moderate live video at scale (Kahn, 2019).
However, current technologies do apply forms of live object detection. Self-driving cars must understand objects in real time (Chin et al., 2019). Even so, these technologies are typically applying detection of objects, which is a much simpler task than parsing context about whether a scene contains violence.



Predictive Models - Computer Audition Models for Content Analysis

Computer audition seeks to understand audio content. Where audio involves humans speaking, speech will often first be *transcribed* to a text form and analyzed with natural language processing (NLP) methods. This may compound errors that are misheard (such as if "porn" is misheard as "born," potentially changing an analyzed context). Google's "AI Autobahn" combines its Natural Language API and Jigsaw's Perspective APIs to first do speech-to-text analysis, then apply textual sentiment and toxicity analysis. NLP methods and their strengths and limitations are covered in detail in CDT's previous *Mixed Messages* report (Duarte et al., 2017).

Deep learning applications for computer audition mirror many of the use cases in computer vision. However, they are typically conducted on spectrograms (graphic frequency depictions) of audio, rather than on images. State-of-the-art image classification techniques are also capable of achieving positive results on audio classification tasks (Hershey et al., 2017). Tasks of audio classification are often analogous to their image counterparts. Scene recognition, for example, has an audio counterpart (computational auditory scene recognition, or CASR) (Petetin et al., 2015). The foundational "cats and dogs" image classification task even has an audio counterpart for barks and meows (Takahashi et al., 2016).

Some unique challenges presented in computer audition include **mitigating noise**, **data variations**, and **language biases**. Isolating salient audio from noise is the subject of current research, which is attempting to isolate *sources* of audio in mixed-audio recordings (Gfeller et al., 2020). Sound samples themselves may be inconsistent, with varied loudness, sample quality, and time durations (Saska et al., 2019). Some algorithms exist for noise reduction, including spectral noise gating, which aims to eliminate consistent background noise by "gating" out any noise that falls in a certain frequency range. This can help eliminate certain types of consistent background noise, like eliminating the frequencies of a coffee grinder from a recording of ambient sounds in a coffee shop. This could be useful, for example, in a tool that is trying to identify the song playing over the coffee shop's loudspeakers. But gating out the coffee-grinder frequencies could also affect, for example, the ability of a matching algorithm to identify a song that uses those same frequencies.

Finally, **automatic speech recognition (ASR)** is challenged by the fact speech can occur in many different languages, accents, or dialects. Different recognition models may be trained on "high resource languages" (languages for which many data resources exist) versus "low resource languages" (for which there are few data resources available). Many near-extinct languages, dialects, or primarily oral languages have not generated electronic data (C. Wang et al., 2020). These considerations present challenges for the widespread application of computer audition tools for predictive applications.

cdt.org

cdt.org/contact

Center for Democracy & Technology
1401 K Street NW, Suite 200
Washington, D.C. 20005

202-637-9800

@CenDemTech

CENTER FOR
DEMOCRACY
& TECHNOLOGY