Exploring Interpretability of Gender Classification Models

Ashvin Pai, Martin Strauss

27 June 2023

Outline

- Automatic Gender Recognition
- Social Issues of AGR
- Machine Learning Overview
- Basics of Convolutional Neural Networks
- The Grad-CAM Heatmap Algorithm
- Our Work
  - Model Architecture and Training
  - Experiments with Heatmaps
  - Further Work

Automatic Gender Recognition

An automatic gender recognizer is a model which takes data about an individual as an input and then produces a gender classification as an output.

Figure: Spotify Logo (spotify.com)

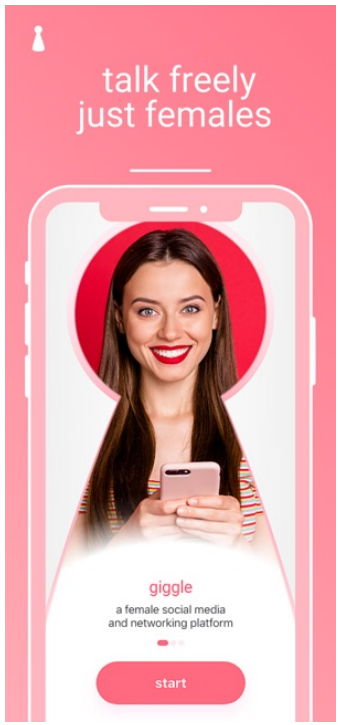


Figure: Giggle App (mzstatic.com)

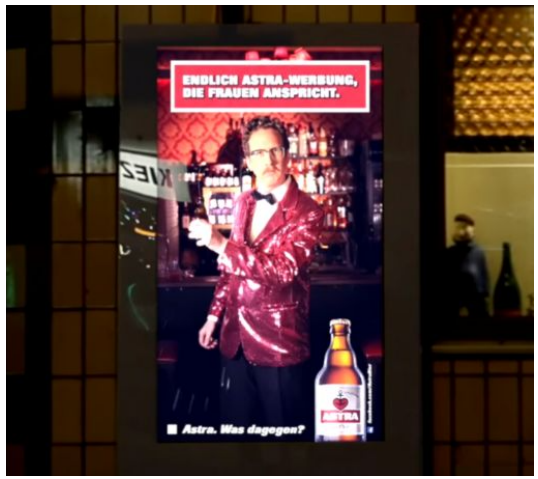


Figure: Digital Billboard (trendhunter.com)

Automatic Gender Recognition

Gender recognition is built into other facial recognition technologies as a feature.

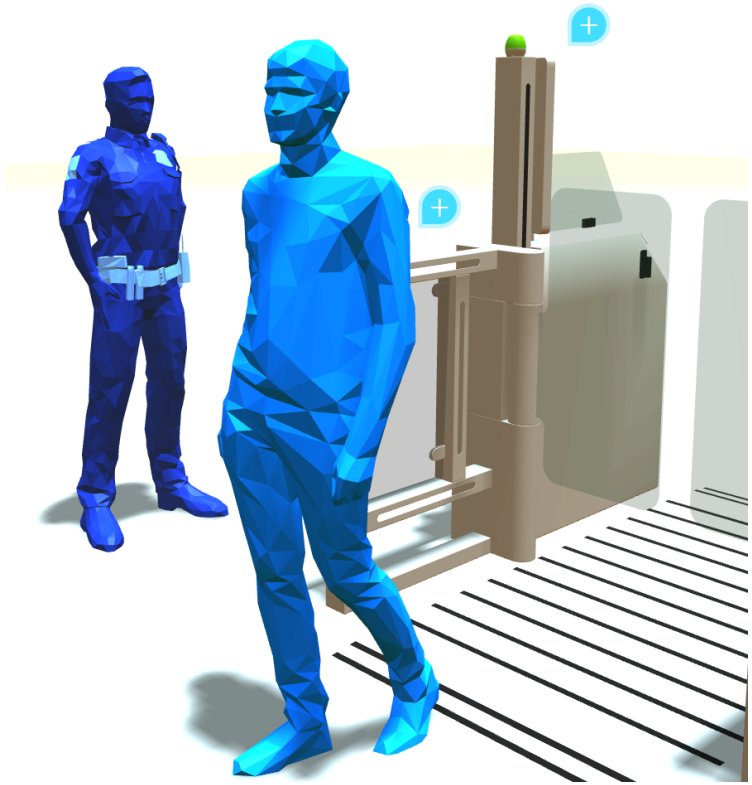


Figure: Biometric Facial Comparison (US Customs and Border Patrol)

Frame Title

However these technologies are very flawed at the moment and numerous people have advocated against their use or development.

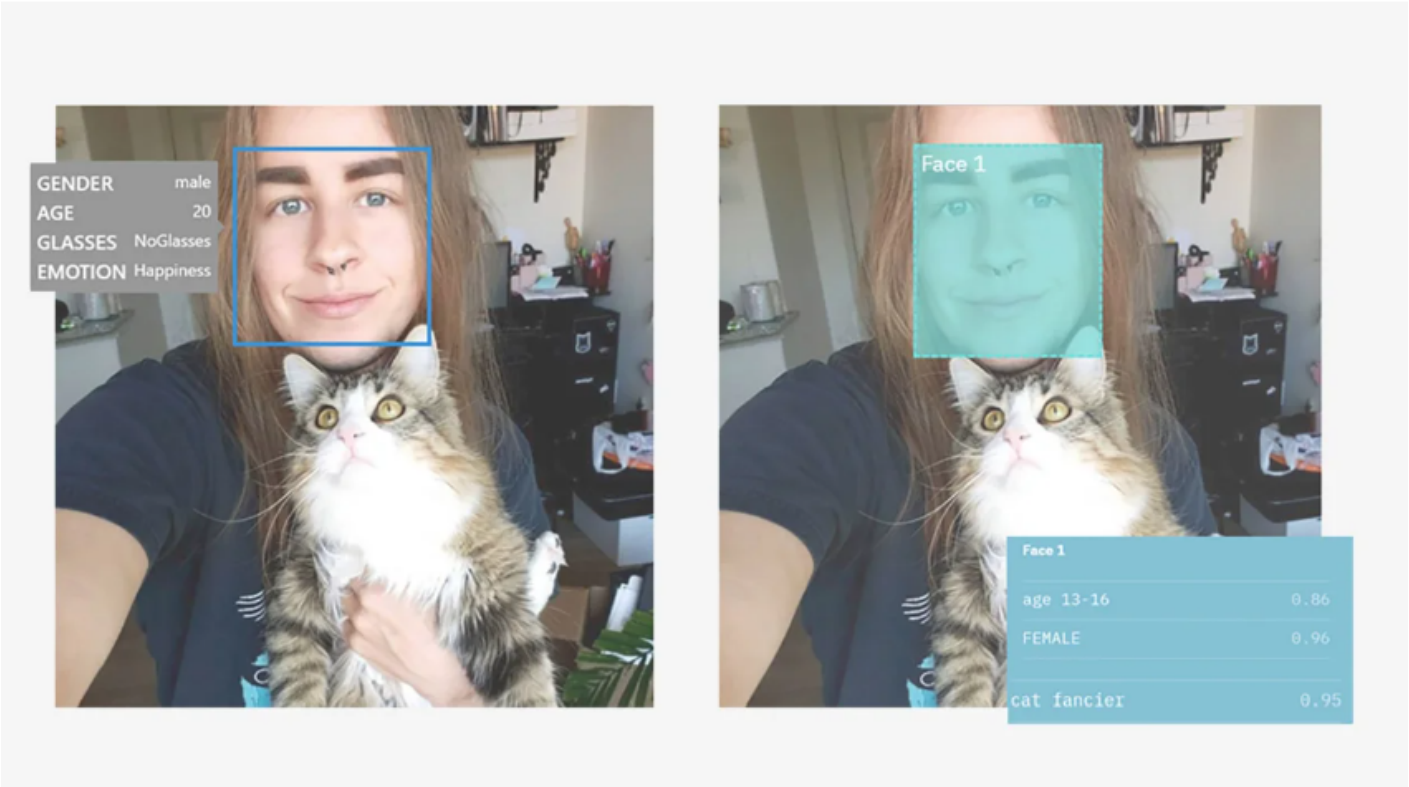


Figure: Classified as different gender by two different softwares (cnn.com)

Social Issues of Gender Classification

Commercial technologies from Microsoft, IBM, and Face++ have a gender and skin color bias with respect to error rate.

Classifier	All	Female	Male	Dark Female	Light Male
MSFT	6.3	10.7	2.6	20.8	0.0
Face++	10.0	21.3	0.7	34.5	0.8
IBM	12.1	20.3	5.6	34.7	0.3

Table: Error Rates of Commercial AGR Systems (Buolamwini and Gebru, 2018)

Social Issues of Gender Classification

Most technologies exclude non-binary people by encoding a man-woman binary in their training data and predictions. Meanwhile in the world...

Folksonomies Turned Hashtag			
	Instagram #		***Instagram #***
AFAB	28,191	***Man***	36,466,751
Agender	1,864,879	*Neutrois*	28,060
AMAB	20,332	***Non-Binary***	2,780,477
Androgyne	223,125	*Pangender*	156,596
Bigender	855,370	*Polygender*	103,620
Cisgender	97,971	*Third Gender*	12,592
Demiboy	597,320	*Trans*	5,933,800
Demigirl	592,703	*Trans Feminine*	26,060
Female	6,379,367	***Trans Man***	843,139
Femme	3,132,240	*Trans Masculine*	132,380
Gender Nonconforming	84,780	***Trans Woman***	452,743
Genderless	236,082	*Transgender*	7,849,435
Genderqueer	1,990,117	*Trigender*	171,539
Male	6,884,437	***Woman***	41,269,789

Figure: Gender Labels of a Dataset Scrapped from Instagram (Scheuerman et. al)

Social Issues of Gender Classification

Exhibited bias against binary transgender men and women with respect to error rate.

TPR Performance Per Gender Hashtag													
Hashtag	***Amazon***			***Clarifai***			***IBM***			***Microsoft***			***All***
	T	*F*	*TPR*	*T*	*F*	*TPR*	*T*	*F*	*TPR*	*T*	*F*	*TPR*	*Avg*
#woman	348	2	99.4%	333	17	95.1%	345	5	98.6%	100	0	100.0%	98.3%
#man	334	16	95.4%	344	6	98.3%	341	9	97.4%	348	2	99.4%	97.6%
#transwoman	317	33	90.6%	271	79	77.4%	330	20	94.3%	305	45	87.1%	87.3%
#transman	216	134	61.7%	266	84	76.0%	250	100	71.4%	255	95	72.8%	70.5%
#agender, #genderqueer, #nonbinary	—	—	—	—	—	—	—	—	—	—	—	—	—

Figure: True Positive Rates (Scheuerman et. al)

Social Issues of Gender Classification

"Put simply, a trans-inclusive system for non-consensually defining someone's gender is a contradiction in terms."
- Keyes, 2018

Social Issues of Gender Classification

But maybe AGR has some benefits for people if it is used consensually and in private?

Figure: Makeup System for "Passing" (Chong et. al, 2021)

Social Issues of Gender Classification

Some reasonable questions you might have now... Why are these model biased? How are they making decisions? Answer: No idea.

Figure: xkcd Comic 1838

Social Issues of Gender Classification

Questions we can raise...

- What insights would a method of interpretability give into how a gender classification model contributes to aformentioned social issues?
- Could interpretability allow us to make recommendations to people, particularly transgender people, on how they might "fool" a gender classification model?

Machine Learning Overview

High-level of model training. This is a supervised learning task.

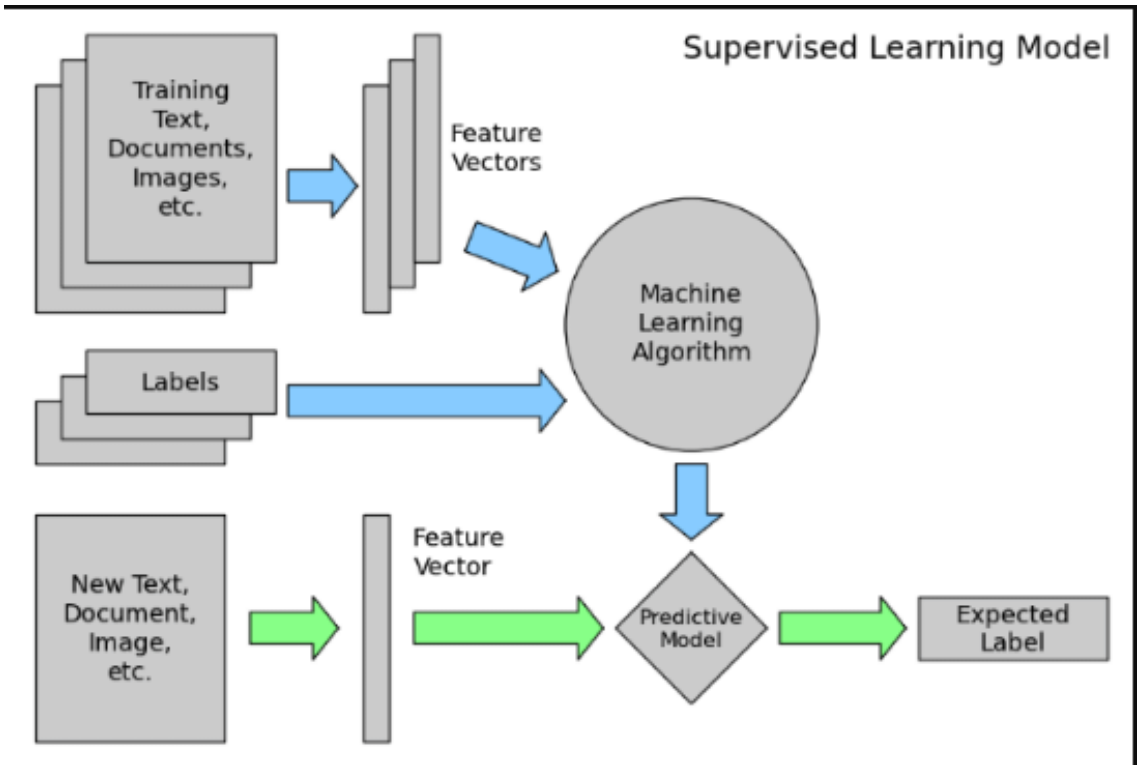
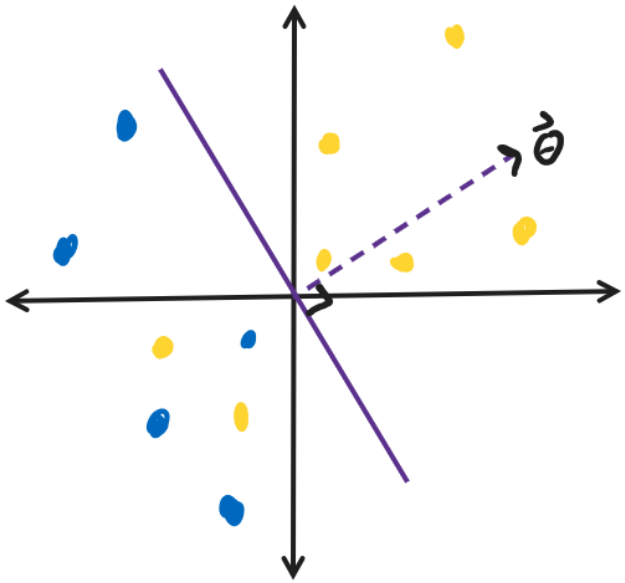


Figure: Supervised Machine Learning, researchgate.net

Machine Learning Overview

What goes on in the training phase?

- Our training data is points in $\mathbb{R}^2$ labeled either maize or blue.
- Goal: learn a separating hyperplane $\bar{\theta} \in \mathbb{R}^2$ that minimizes total error/loss.
- Many different definitions for loss. Simplest is 0-1 loss where we count the number of misclassified points.

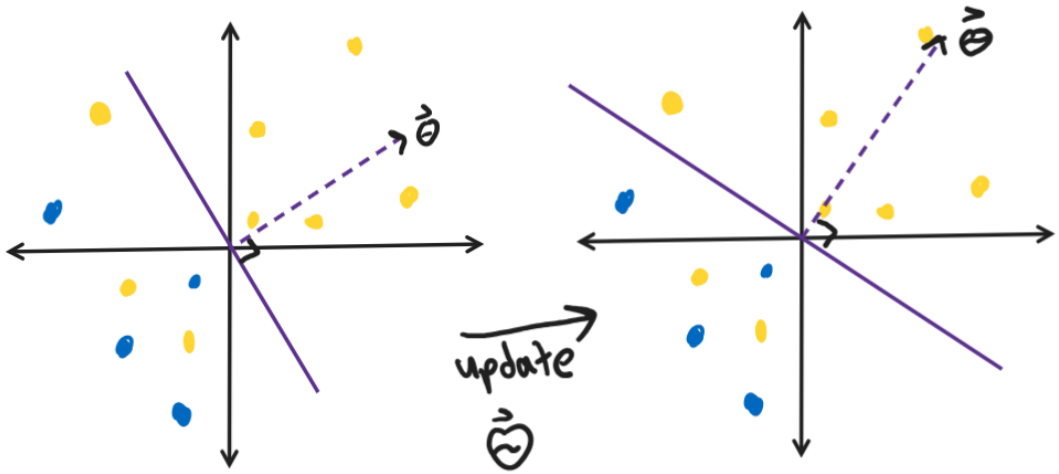


Machine Learning Overview

How to minimize loss?

Ex: Gradient Descent Algorithm

1. Initialize $\bar{\theta}$ randomly.
2. Run through training data and calculate loss.
3. Calculate gradient of loss with respect to each weight.
4. Update weights by taking a small step in direction opposite to gradient.
5. Loop until convergence criteria.



Machine Learning Overview

But doing this from scratch requires lots of data and resources. Instead we can use transfer learning from an existing pre-trained model.

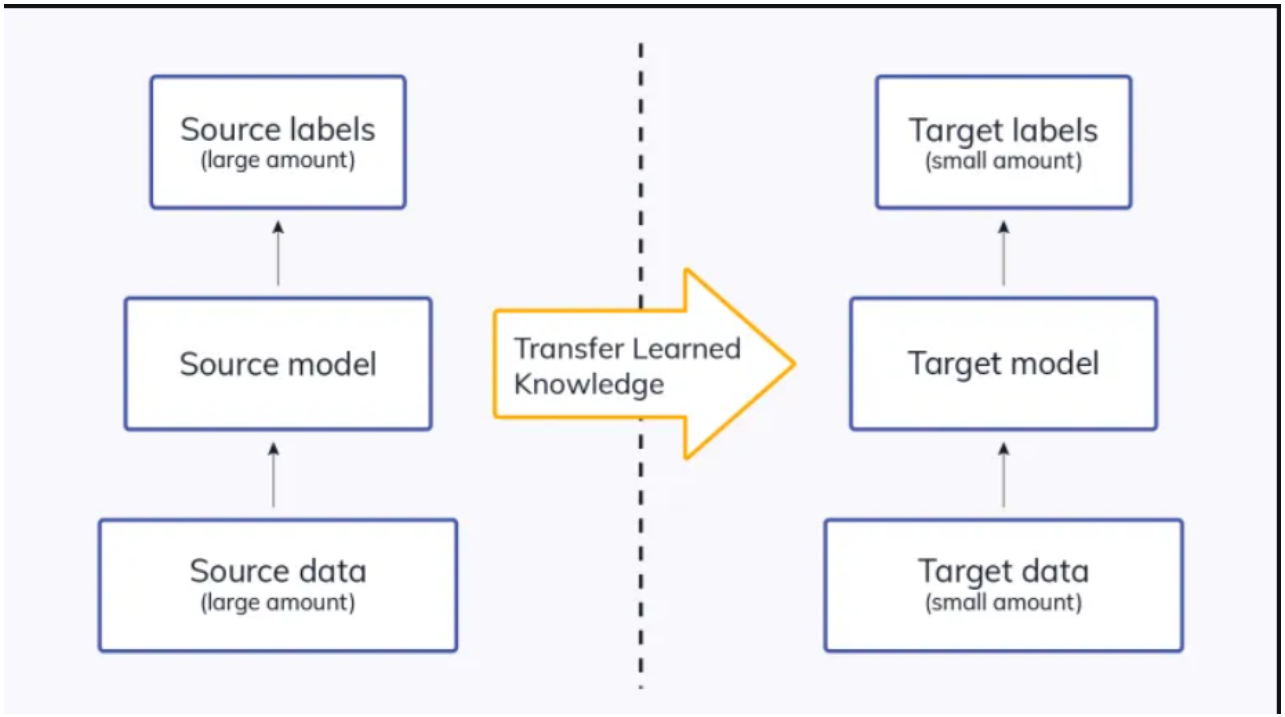


Figure: Transfer Learning, neptune.ai

Machine Learning Overview

For image classification tasks the base model is usually an ImageNet classifier.

Figure: ImageNet, cs.stanford.edu

Basics of Convolutional Neural Networks

In a typical neural network every input neuron is connected to the hidden layer.

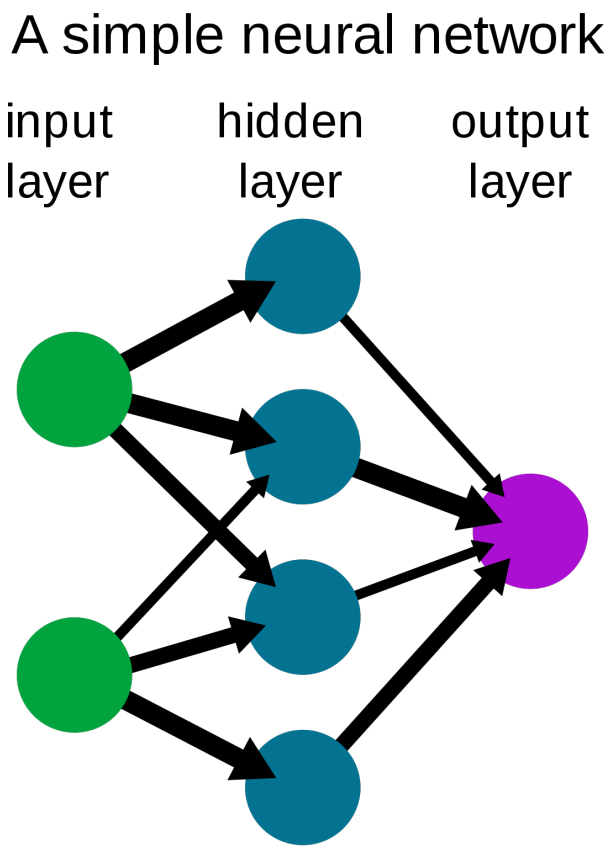


Figure: typical neural network, wikipedia.com

Basics of Convolutional Neural Networks

In a convolutional neural network only a small number of input neurons (i.e. pixel value) is connected to the hidden layer. These are connected by convolution.

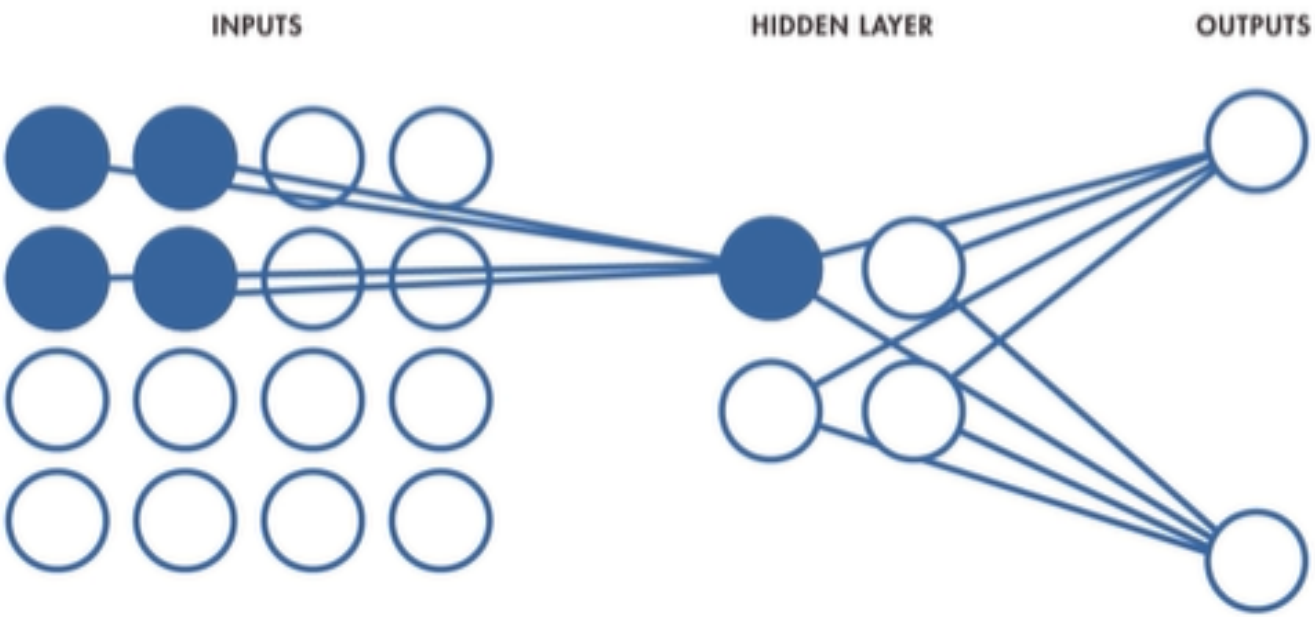


Figure: step 1, mathworks.com

Basics of Convolutional Neural Networks

In a convolutional neural network only a small number of input neurons (i.e. pixel value) is connected to the hidden layer. These are connected by convolution.

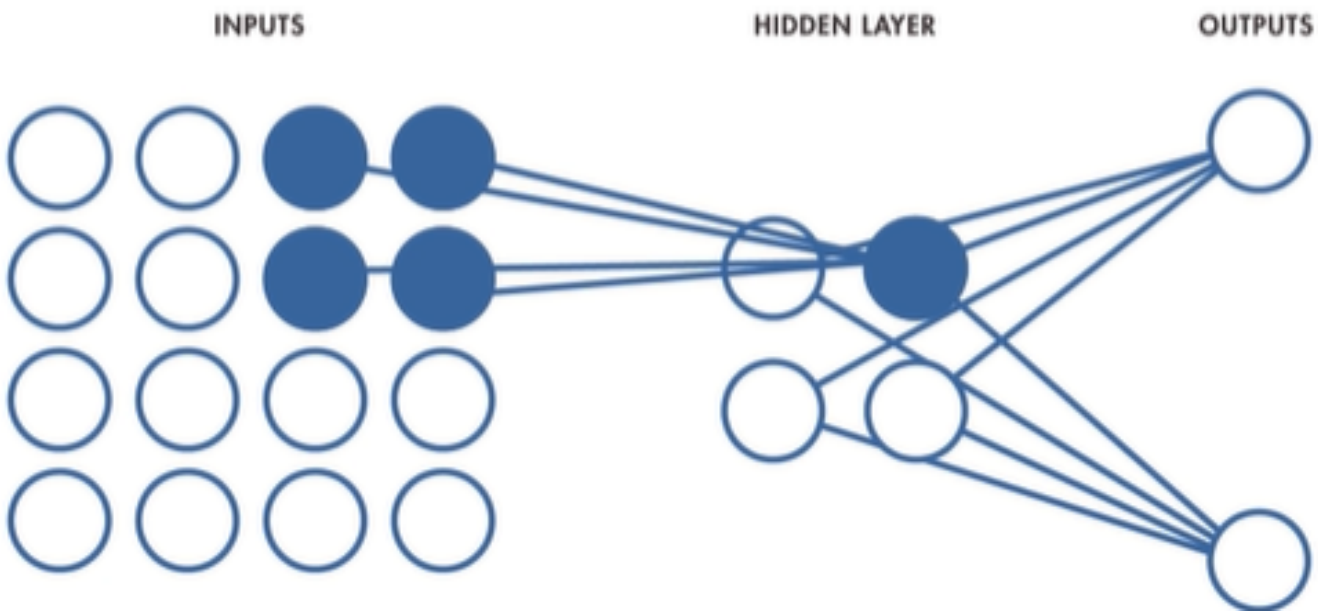


Figure: step 2, mathworks.com

Basics of Convolutional Neural Networks

In a convolutional neural network only a small number of input neurons (i.e. pixel value) is connected to the hidden layer. These are connected by convolution.

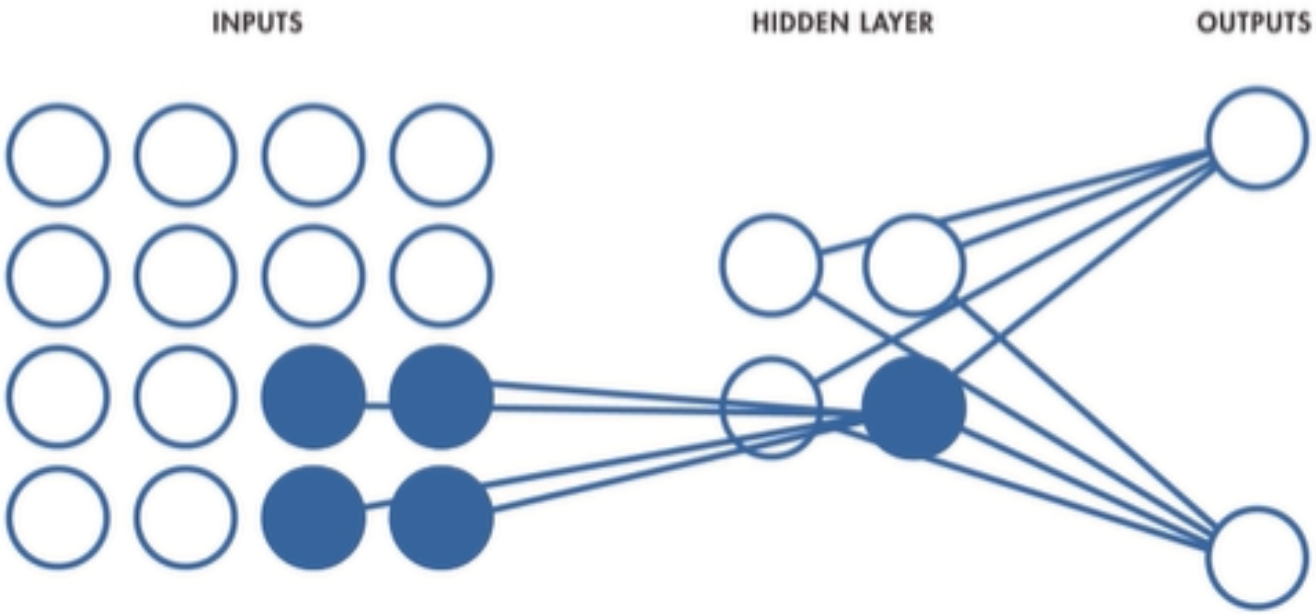


Figure: step 3, mathworks.com

Basics of Convolutional Neural Networks

In a convolutional neural network only a small number of input neurons (i.e. pixel value) is connected to the hidden layer.

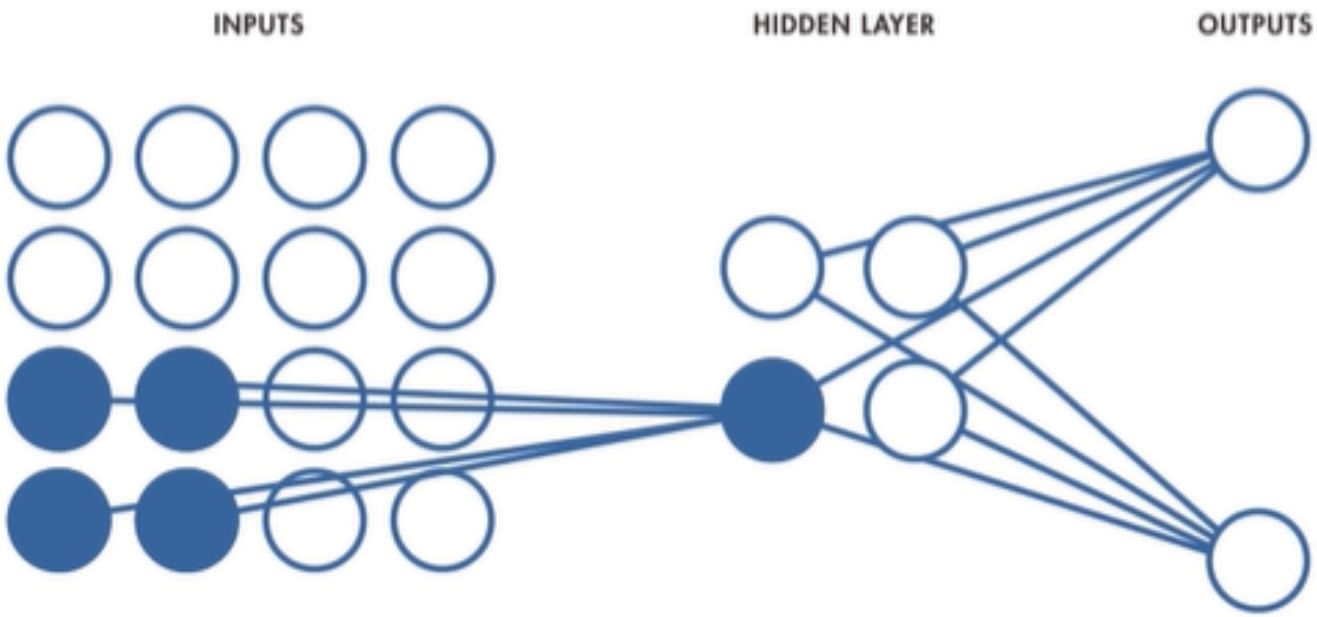


Figure: step 4, mathworks.com

Basics of Convolutional Neural Networks

Weights and biases for neurons in a given hidden layer are all the same. All hidden neurons in a layer are detecting the same feature $\Rightarrow$ The network is stable w.r.t. translation.

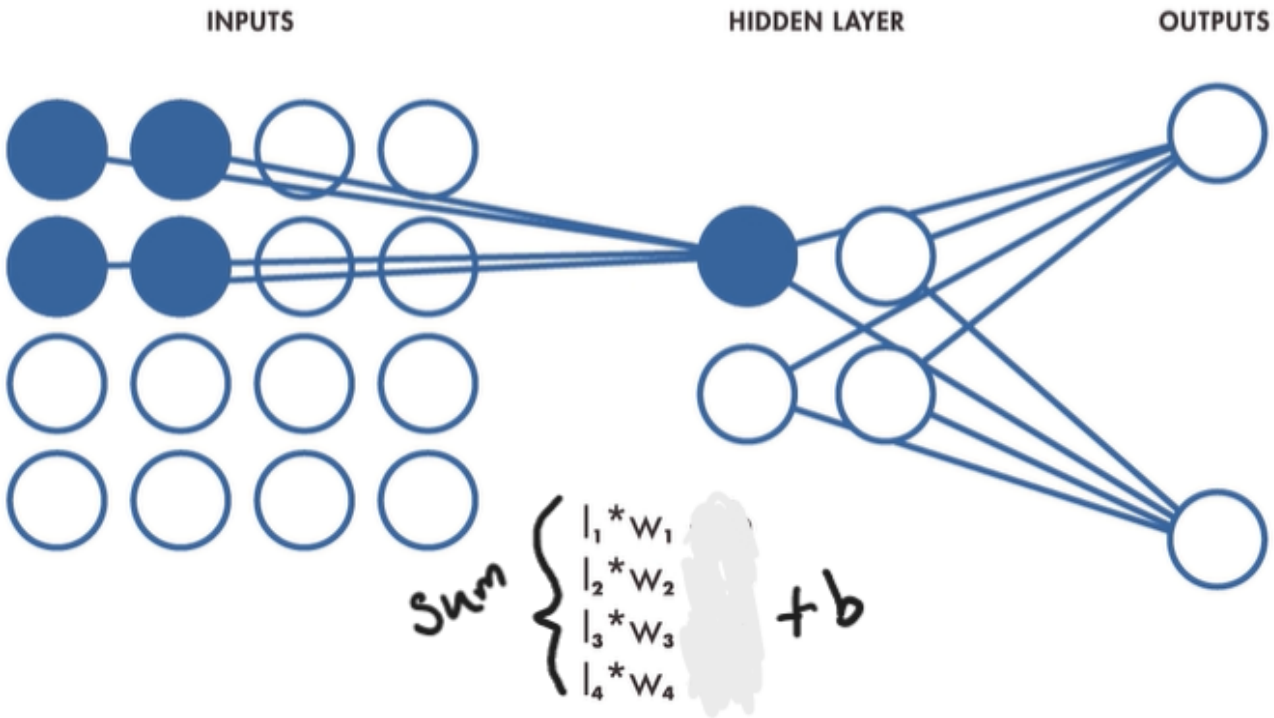


Figure: Weights and Bias, mathworks.com

Basics of Convolutional Neural Networks

Example of a vertical edge filter for a black-white image:

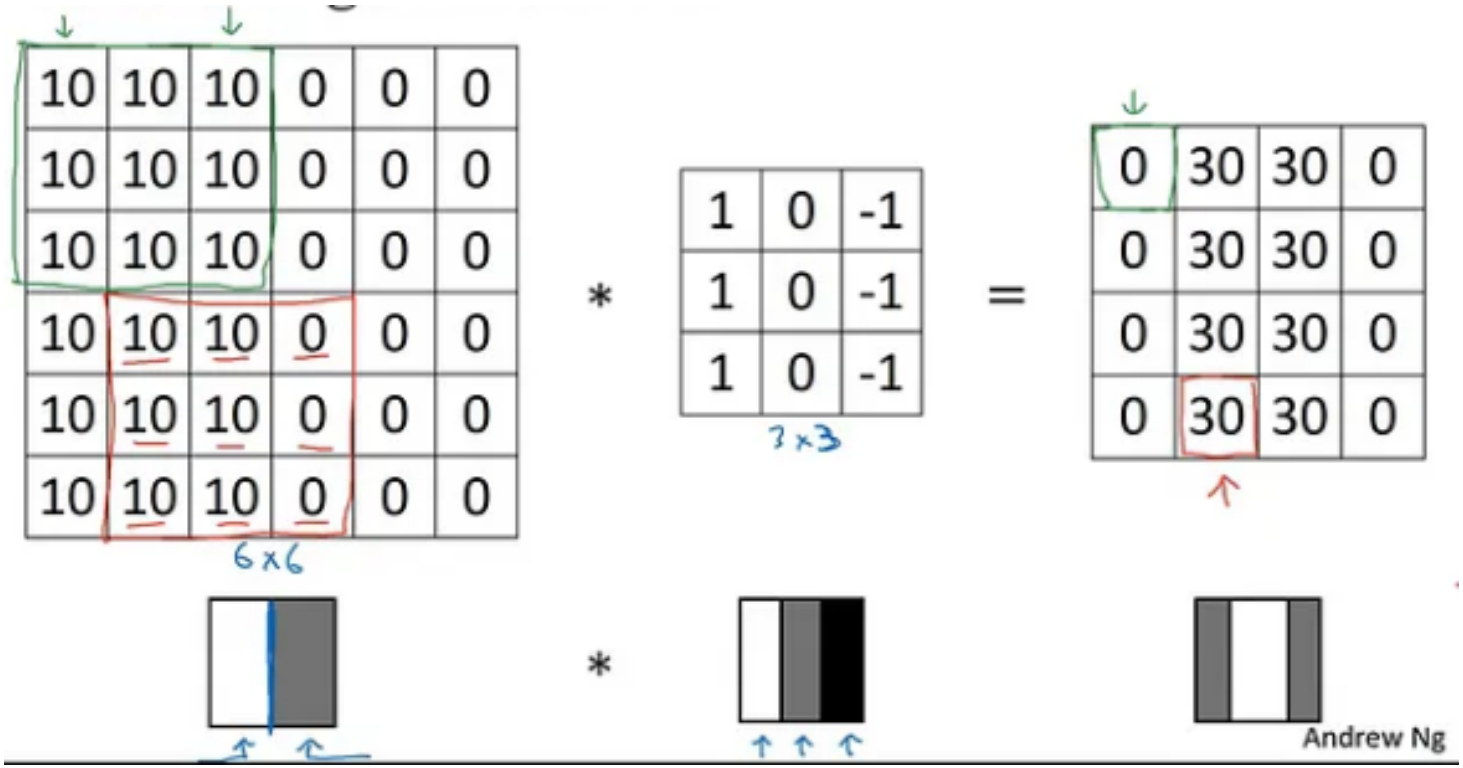


Figure: Vertical Edge Detection, Andrew Ng on Coursera

Basics of Convolutional Neural Networks

To introduce non-linearity and reduce parameters CNNs use activation and pooling. This allows us to learn much more complicated features.

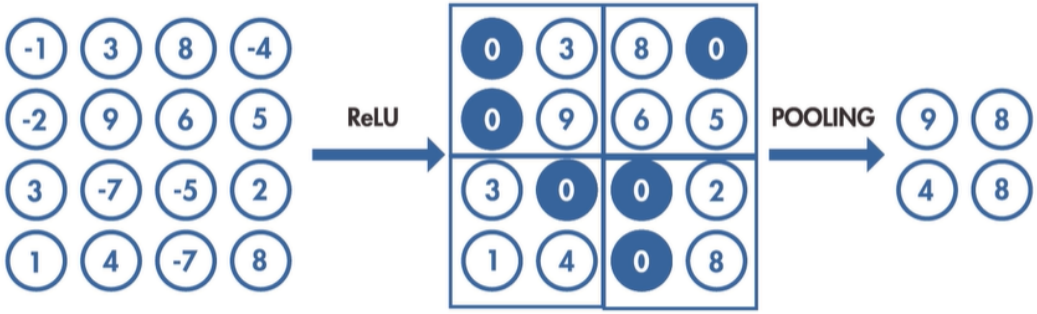


Figure: ReLU Activation + Max Pooling, mathworks.com

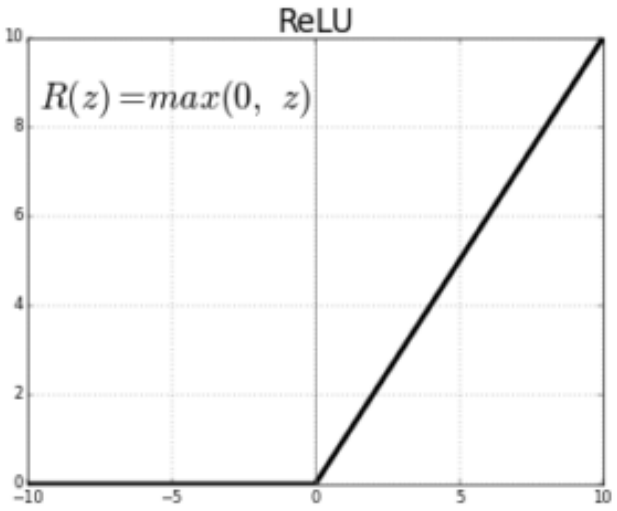


Figure: ReLU Function

Basics of Convolutional Neural Networks

Stack everything up and tack on a regular neural network before the output layer. During training we learn the weights of our kernels. What's going on inside? No idea...

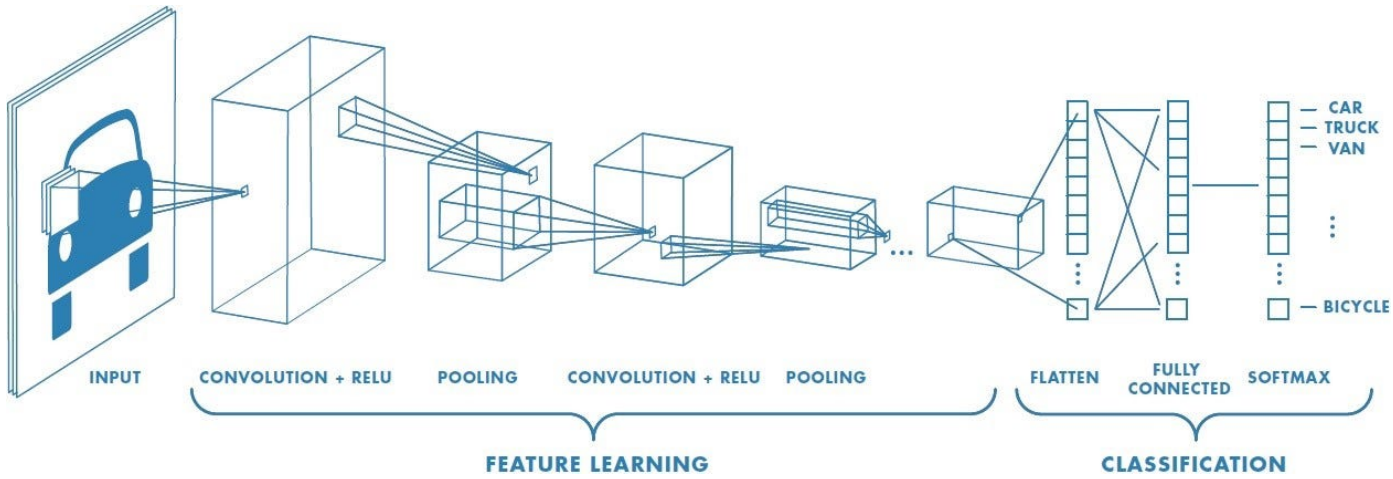


Figure: Typical CNN Architecture, saturncloud.io

The Grad-CAM Algorithm

Gradient-Weighted Class Activation Mapping uses the gradients of a target class with respect to the last convolutional layer to produce a heatmap highlighting the important regions in the image for predicting the concept.

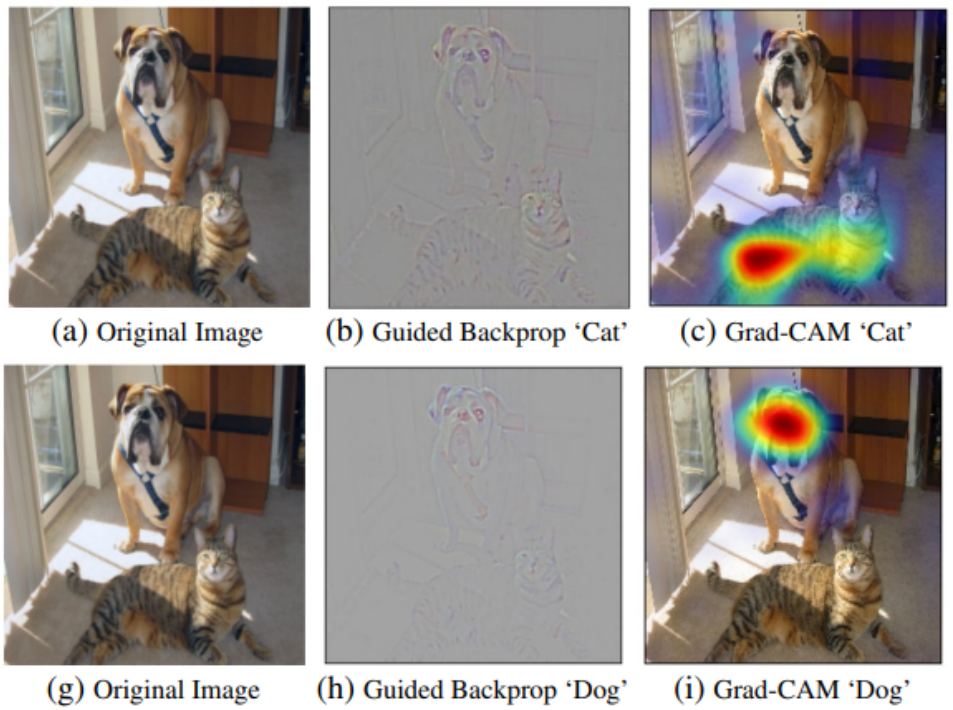


Figure: Selvaraju et. al, 2019

The Grad-CAM Algorithm

Given an input image of dimension $u \times v$ let us have:

- c: target class
- y^c: probability score given by model for target class c
- A^k: k-th feature map of the last convolutional layer
- $L^c_{\text{Grad}-\text{CAM}} \in \mathbb{R}^{u \times v}$: heatmap

Then our weight for the k-th feature map is

$$a_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{1}$$

And our heatmap is produced by

$$L^c_{\text{Grad}-\text{CAM}} = \text{ReLU}(\sum_k a_k A^k) \tag{2}$$

Model Training

- For our Base Model we used the EfficientNetB0 network pretrained on ImageNet dataset.
- We then performed transfer learning and fine-tuning on the GENDER-COLOR-FERET dataset.
- This is a "nice" dataset as it doesn't require much pre-processing.

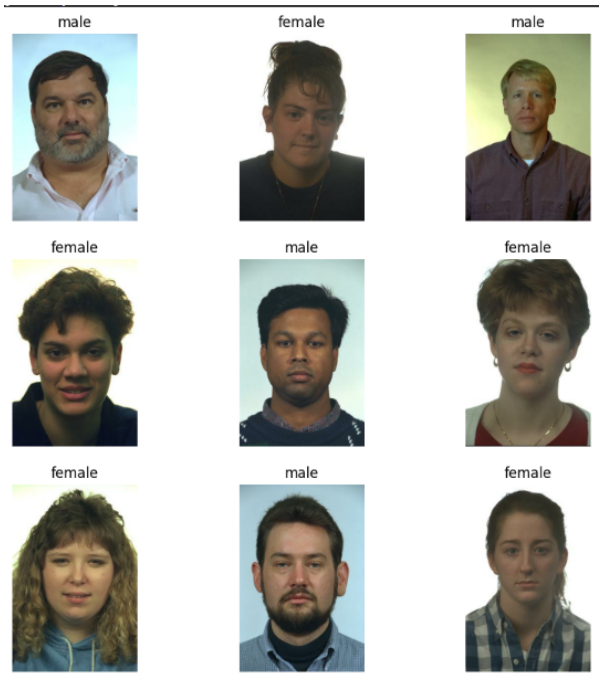


Figure: Example images from the GENDER-COLOR-FERET dataset

Model Training

Training Phase	Binary Accuracy	Loss
Initial	81.82%	0.4347
Fine-Tuning	88.04%	0.3856

Table: Training Results on Heldout Test set

Could have gotten higher accuracy but for our purposes this is pretty good.

Experiments with Grad-CAM

First bias of note was that the model misclassified women at a higher rate than men.

Gender	Number Misclassified	Error Rate
Women	33	15.79%
Men	10	4.78%

Table: Misclassification by Gender

Experiments with Grad-CAM

Can look at heatmaps for an explanation. This reveals the model looks at less features of women than men when making decisions.

Figure: Examples of correctly classified men and women.

Experiments with Grad-CAM

Further, it looks like women are being classified for attributes around the shirt/collar area, though its not apparent why.

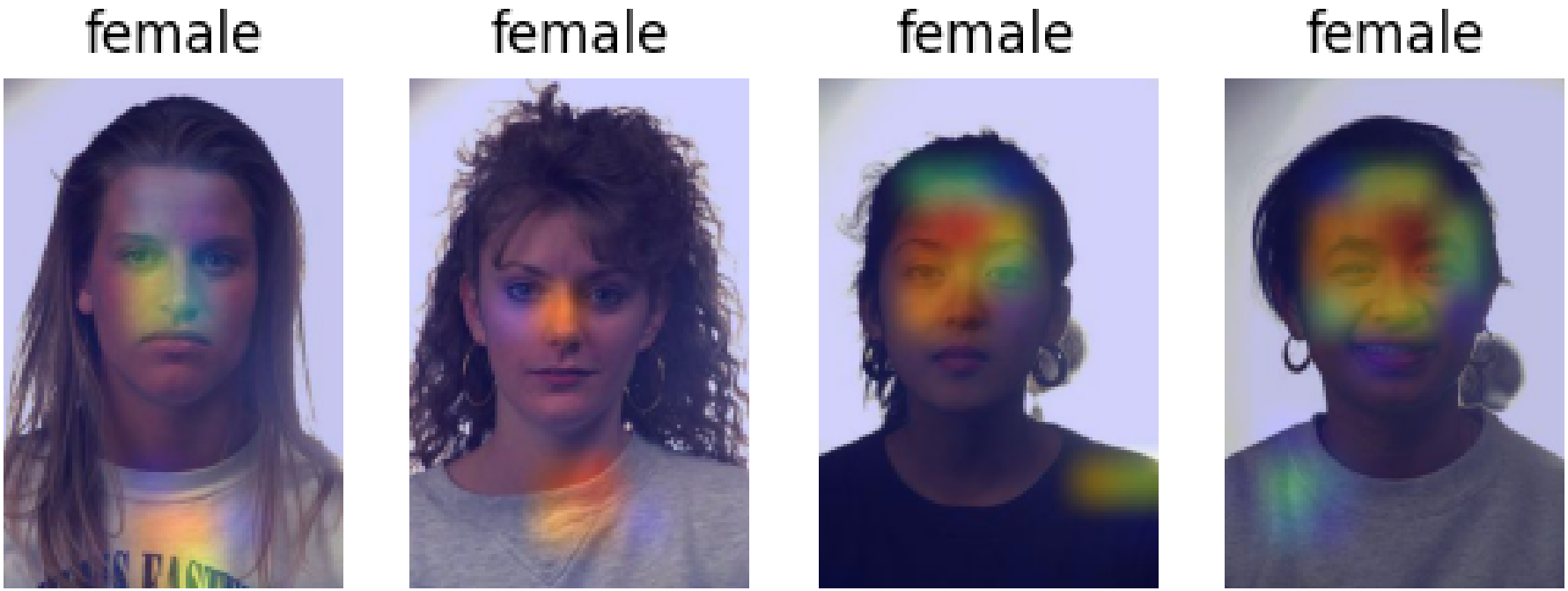


Figure: Examples of women correctly classified with strong presence in shirt/collar area

Experiments with Grad-CAM

This helped us reveal a bias within the dataset: most men are wearing collared shirts in the picture. Maybe this is because...

- When data was collected could have given off impression that it was a formal or professional event.
- Men have less options when it comes to formal clothing.
- Women's formal clothing is more diverse in terms of appearance.

Experiments with Grad-CAM

Exploring further we found several women misclassified as men based on wearing men's shirts (i.e. buttons are on the right side).

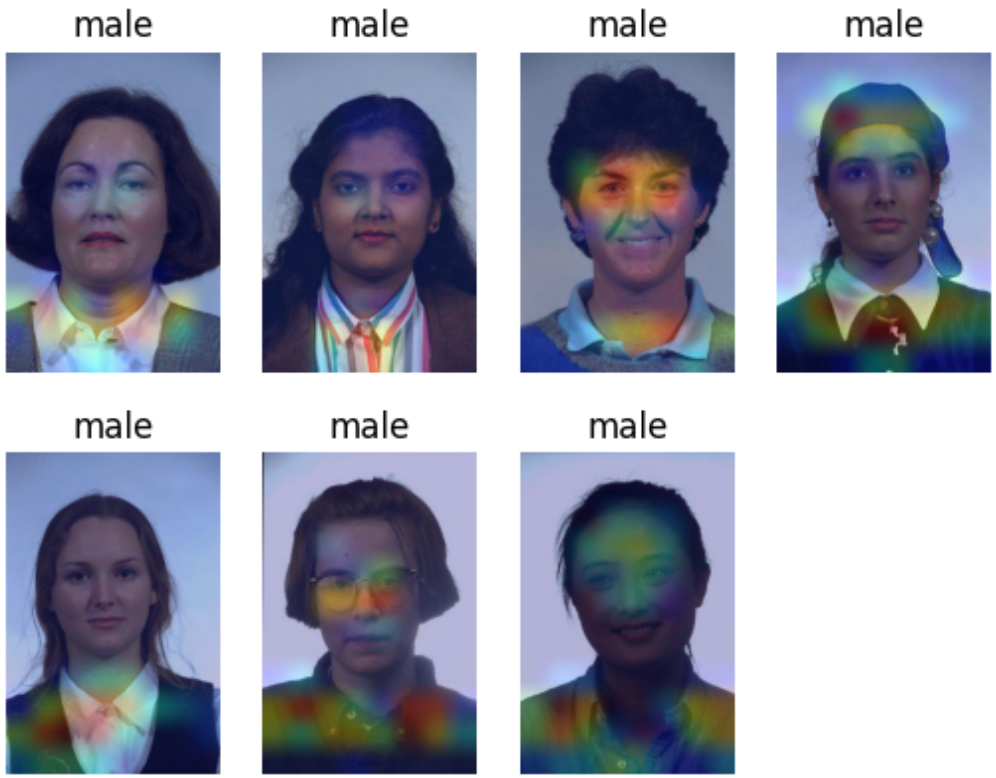


Figure: Misclassified women based on wearing buttoned shirt

Experiments with Grad-CAM

Hypothesized that model might be picking up difference between men's and women's shirt. Flipped images so that buttons would appear on opposite side, all still ended up misclassified.

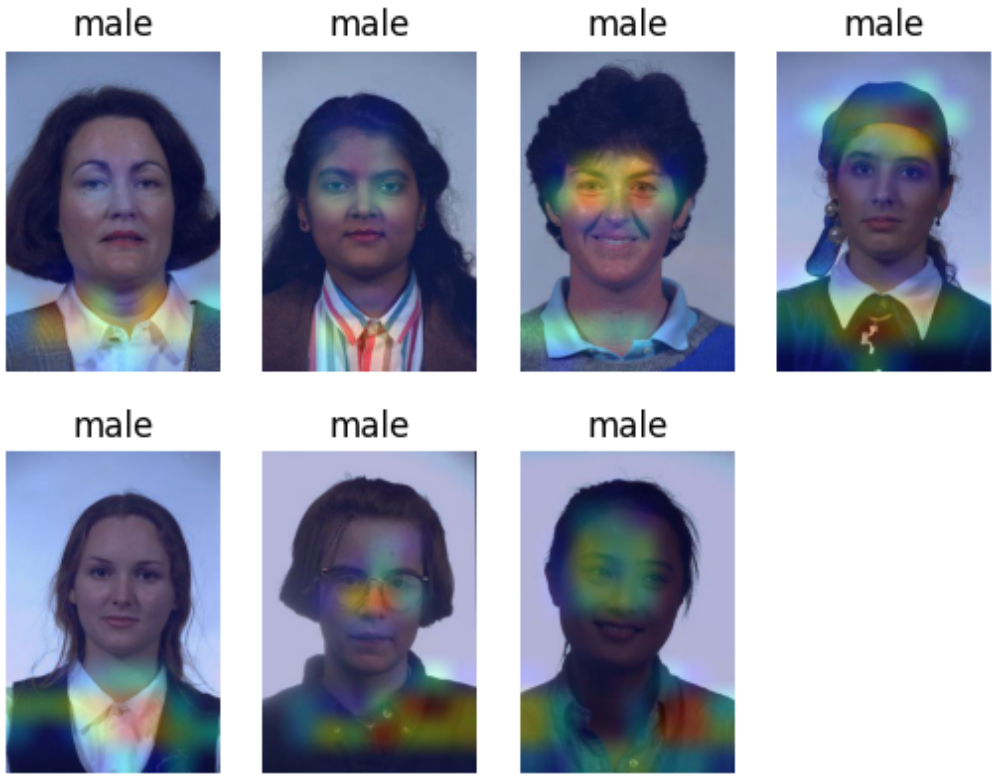


Figure: When flipped they are still misclassified

Experiments with Grad-CAM

Finally tested if appearance of button itself was causing misclassification. Blacked out button from images and five out of seven were classified correctly.

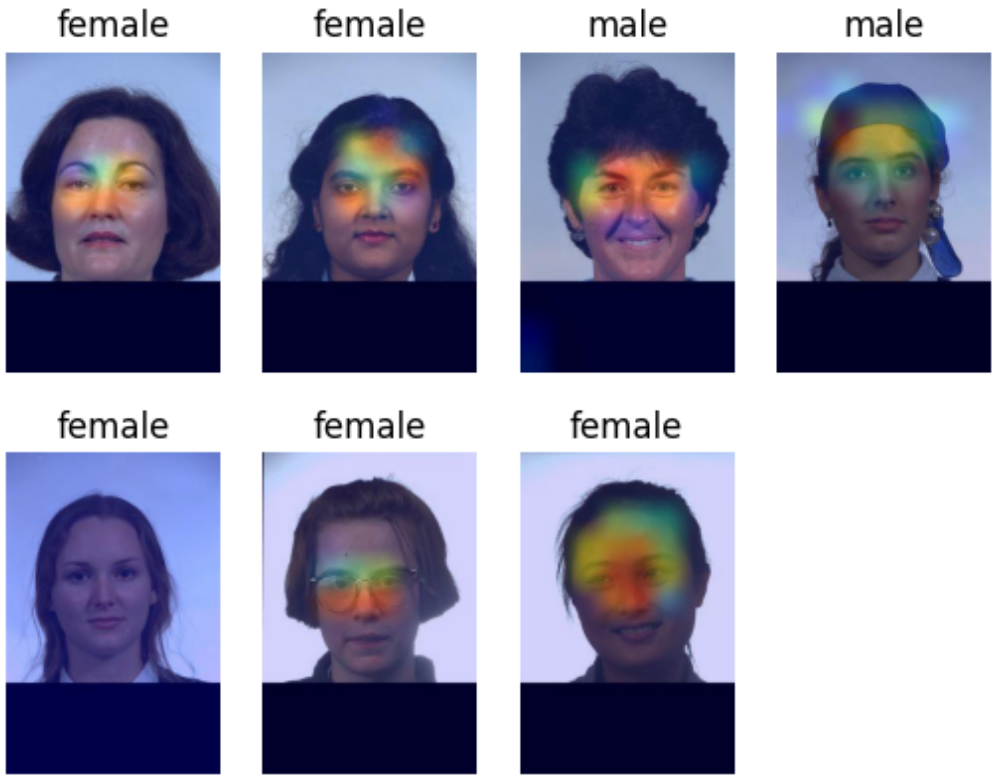


Figure: When buttons and collars are removed accuracy increases

Further Work

So we could potentially advise transgender people that for models trained on the GENDER-COLOR-FERET dataset if they wish to be classified as male they can wear a collared shirt buttoned all the way up. However...

- would need to do more work on a larger class of models to generalize such a recommendation
- would also need to do human subject studies to understand whether interpretability methods affect how transgender people interact with or view automatic gender recognition systems
- building on the work of Chong et. al, we can attempt to incorporate our heatmap system into a make-up recommendation system and study whether that is an effective tool to help transgender people pass in public