

Binary Classification of Graphene Flakes

Samuel Rager and Alvin
Wang



Main References

- Krizhevsky, Sutskever, & Hinton. "ImageNet classification with deep convolutional neural networks." *Commun. ACM* 60. 2017, accessed 29 October 2024,
[<https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>](https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf).
- Ioffe, Sergey. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).
[<https://arxiv.org/pdf/1502.03167>](https://arxiv.org/pdf/1502.03167).
- Kingma, Diederik P. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014), accessed 29 October 2024,
[<https://arxiv.org/pdf/1412.6980>](https://arxiv.org/pdf/1412.6980).

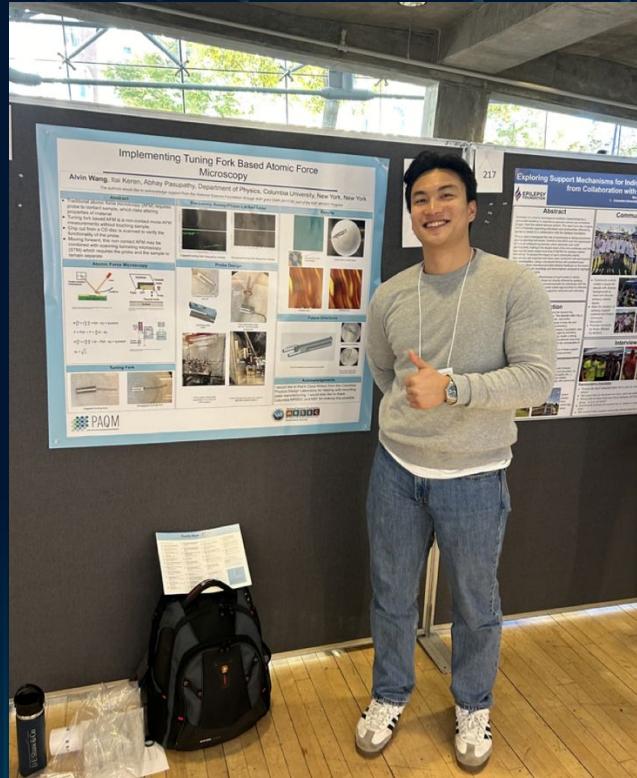
Reproducibility

- GitHub Repository:
<https://github.com/56sarager/Graphene-Classification>
- Google Drive with Data:
https://drive.google.com/drive/folders/1CJks_0wGzySyGWV78xK-VB7FZPO_-qDO?usp=sharing
- Medium Article:
<https://medium.com/@56sarager/alexnet-and-the-binary-classification-of-graphene-ad07dc595867>

Outline

- Speakers' Backgrounds
- The Problem
- State of the Field
- Alternative Solution
- AlexNet Architecture
- Batch Normalization Pseudocode
- Adam Optimizer Pseudocode
- Backpropagation Pseudocode
- Code Demonstration
- Project Results
- Additional Model Architectures and Results
- Future Work

About Us



Why Do We Care?/Why Should You Care?



Condensed Matter Experiment: Studying the electronic properties of 2D materials

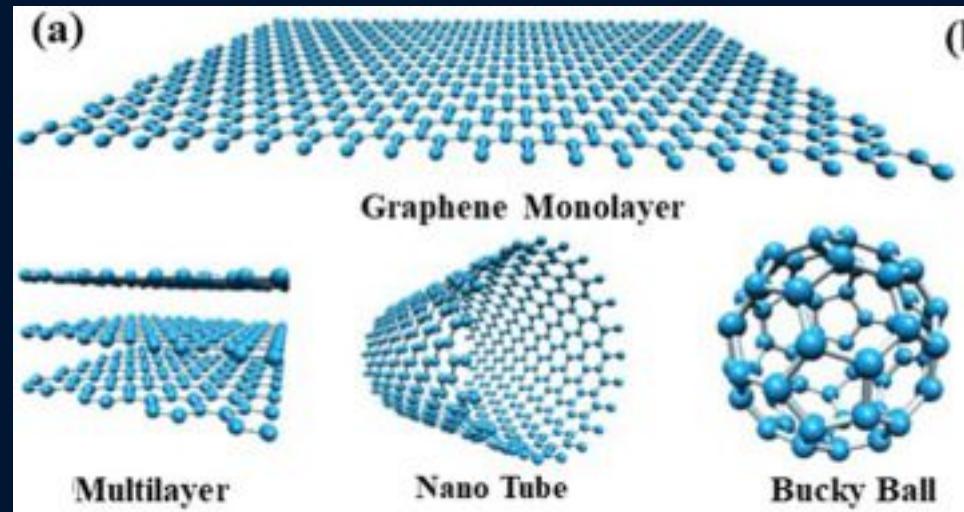
COLUMBIA UNIVERSITY IN THE CITY OF NEW YORK

DEPARTMENT OF PHYSICS

*Condensed Matter Physics / Department of Physics,
www.physics.columbia.edu/content/condensed-matter-physics. Accessed 7 Dec. 2024.*

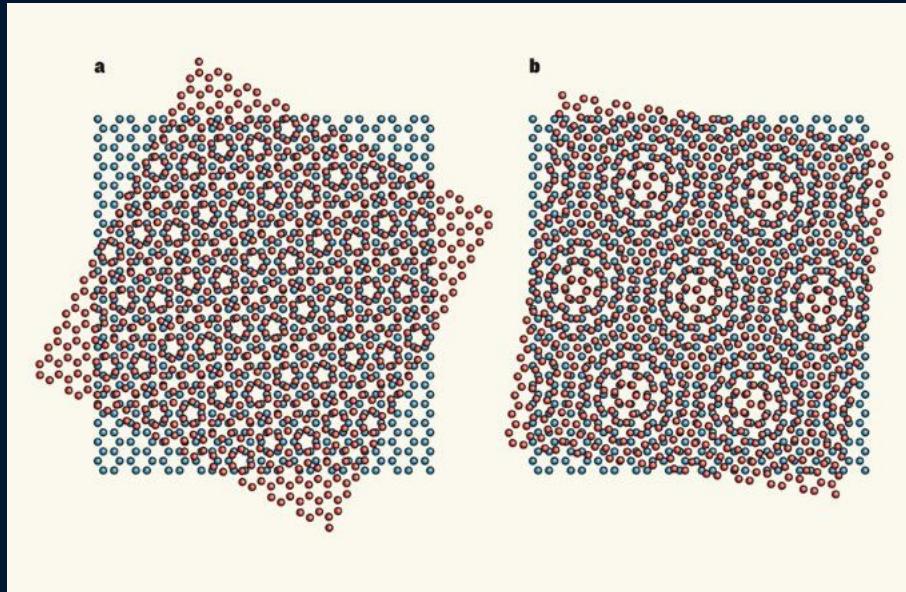


Discovery of Graphene: Professor Andre Geim and Professor Kostya Novoselov



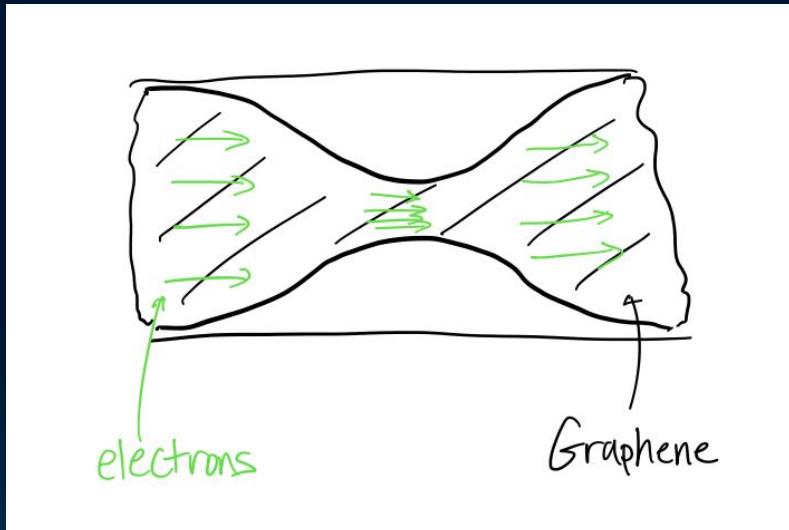
Lloyd-Hughes, James & Jeon, Tae-In. (2012). A Review of the Terahertz Conductivity of Bulk and Nano-Materials. *J Infrared Milli Terahz Waves*. 33. 871. 10.1007/s10762-012-9905-y.

Superconductivity in Moire Superlattices

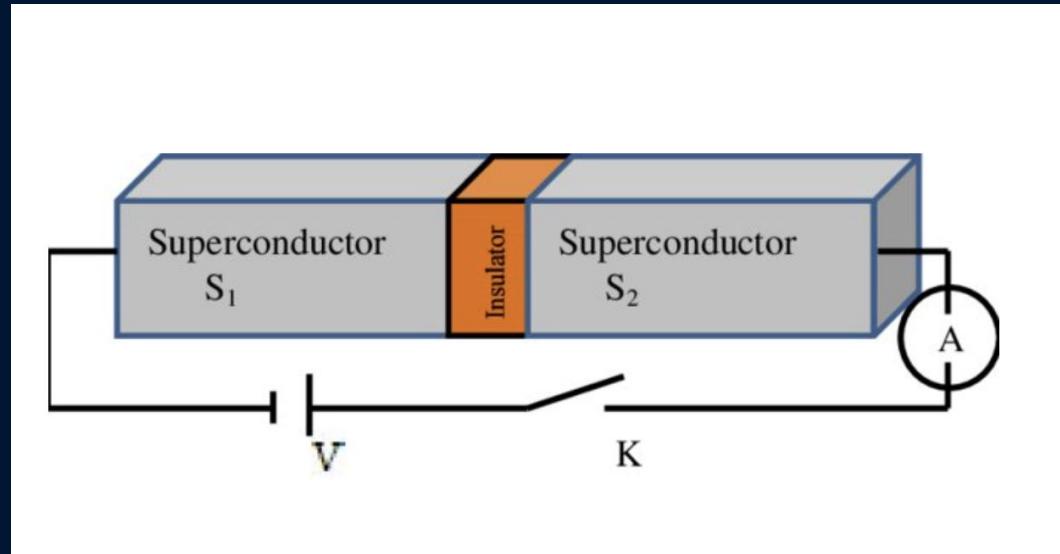


MacDonald, A., Bistritzer, R. Graphene moiré mystery solved?. *Nature* 474, 453–454 (2011).
<https://doi.org/10.1038/474453a>

Work in Pasupathy Lab: Electron Hydrodynamics

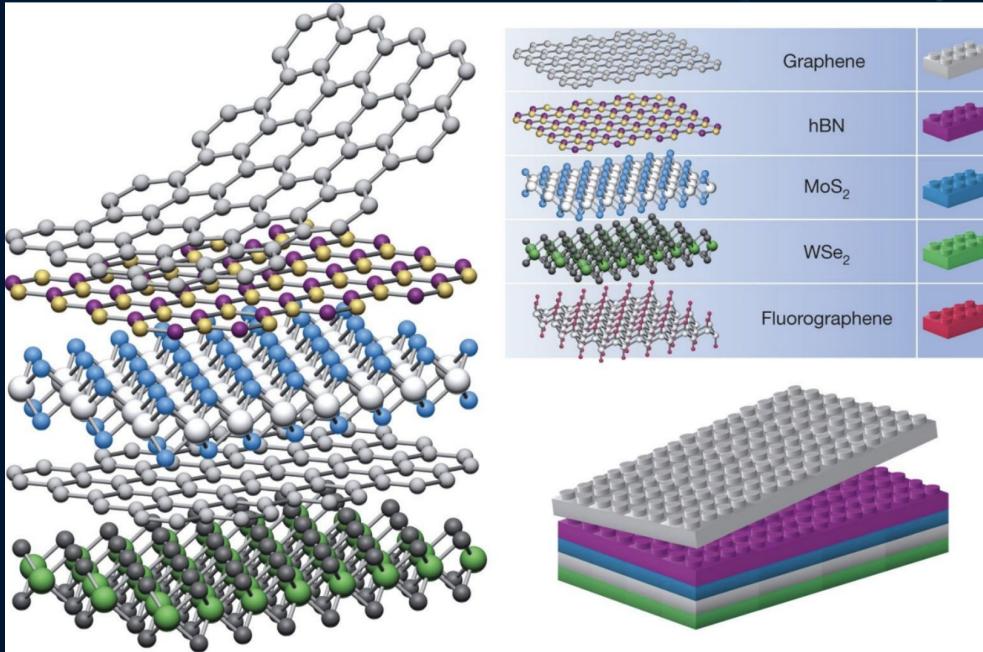


Work in Pasupathy Lab: Josephson Junction



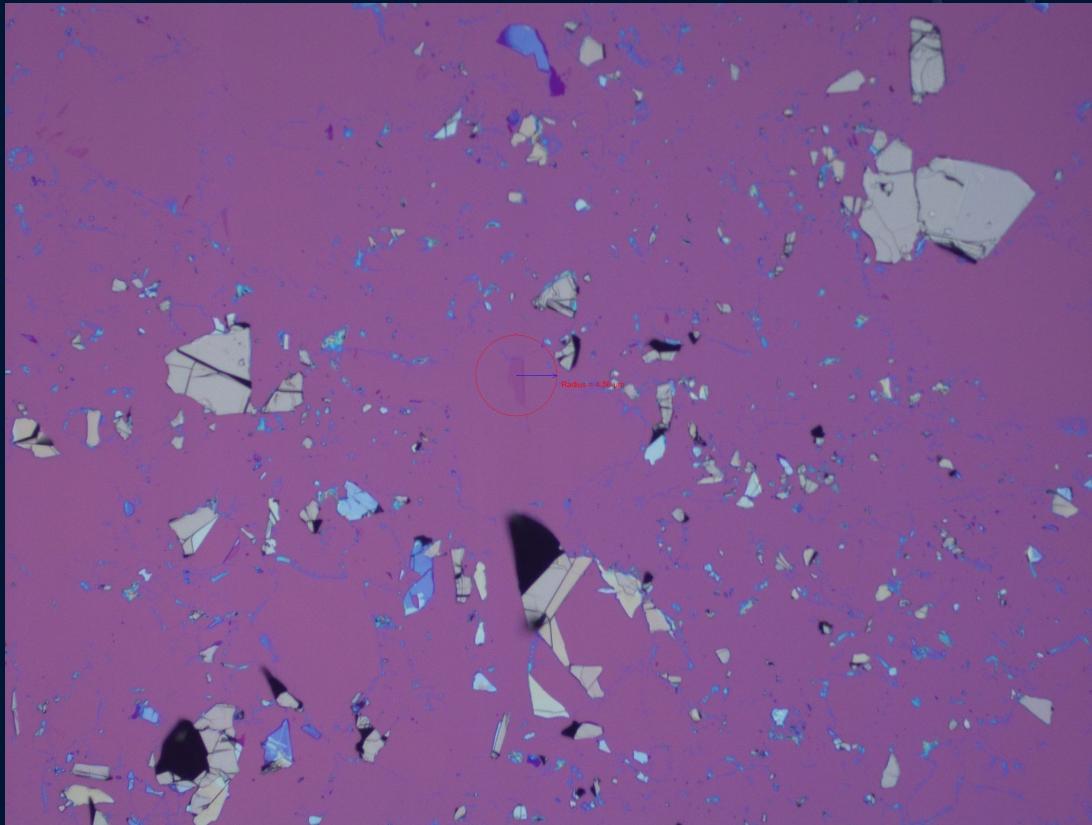
Maruf, H M & Islam, Md. Rafiqul & Chowdhury, F.-U.-Z. (2018). ANALOGY BETWEEN AC JOSEPHSON JUNCTION EFFECTS AND OPTICAL PHENOMENA IN SUPERCONDUCTORS. 105-113.

Everything done in-house

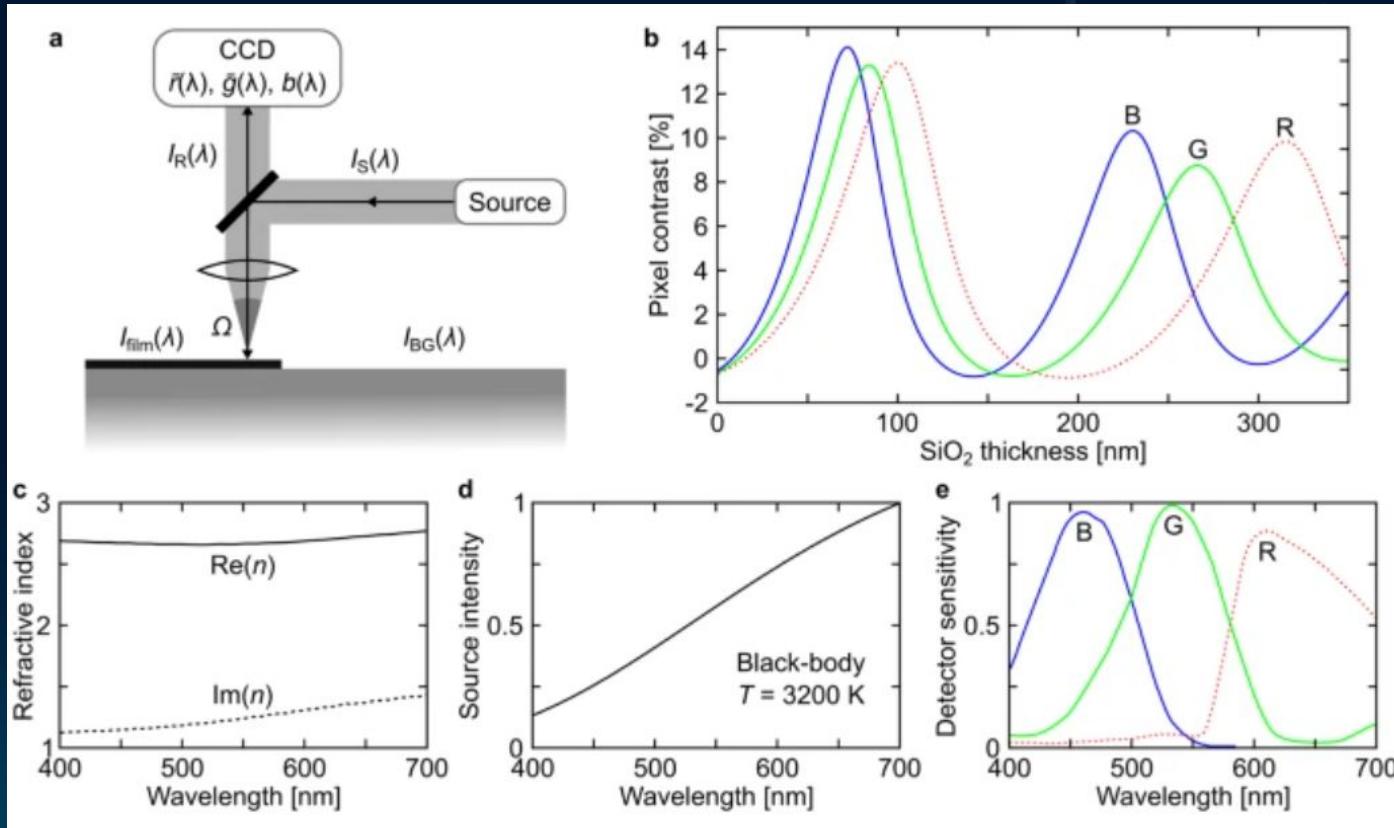


Geim, A. K., and I. V. Grigorieva. "Van Der Waals Heterostructures." *Nature*, vol. 499, no. 7459, 1 July 2013, pp. 419–425, www.nature.com/articles/nature12385, <https://doi.org/10.1038/nature12385>.

Everything done in-house...Problem



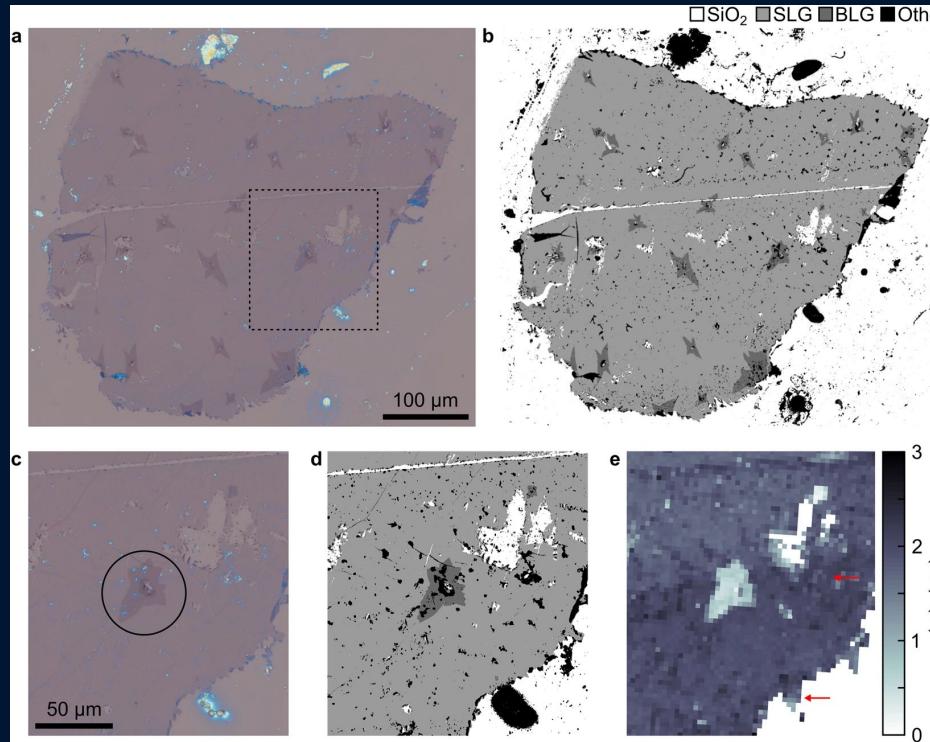
State of the Field- Fresnel Approach



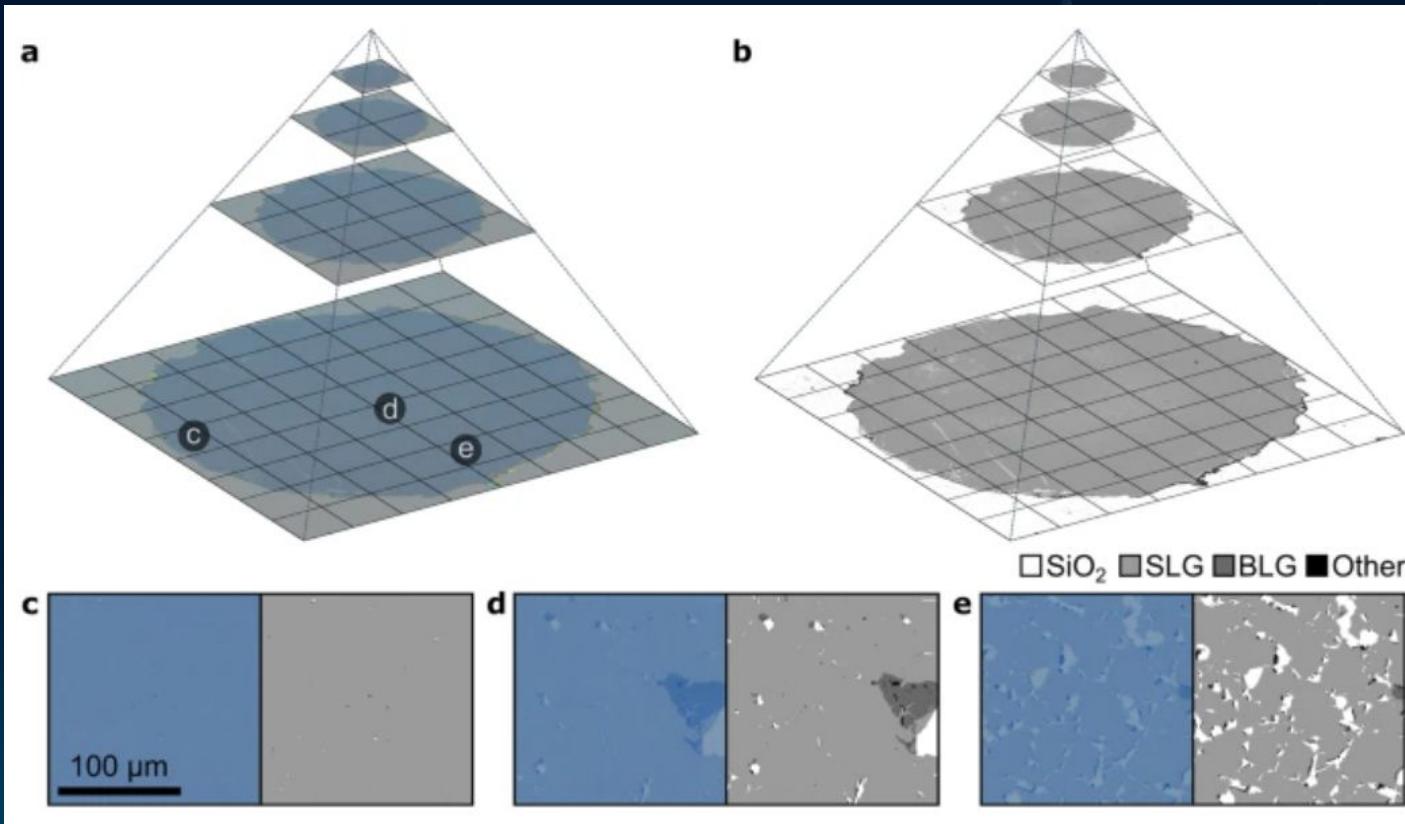
Jessen, B.S., Whelan, P.R., Mackenzie, D.M.A. et al. Quantitative optical mapping of two-dimensional materials. *Sci Rep* 8, 6381 (2018). <https://doi.org/10.1038/s41598-018-23922-1>

State of the Field- Quality

Jessen, B.S., Whelan, P.R., Mackenzie, D.M.A. et al. Quantitative optical mapping of two-dimensional materials. *Sci Rep* 8, 6381 (2018). <https://doi.org/10.1038/s41598-018-23922-1>

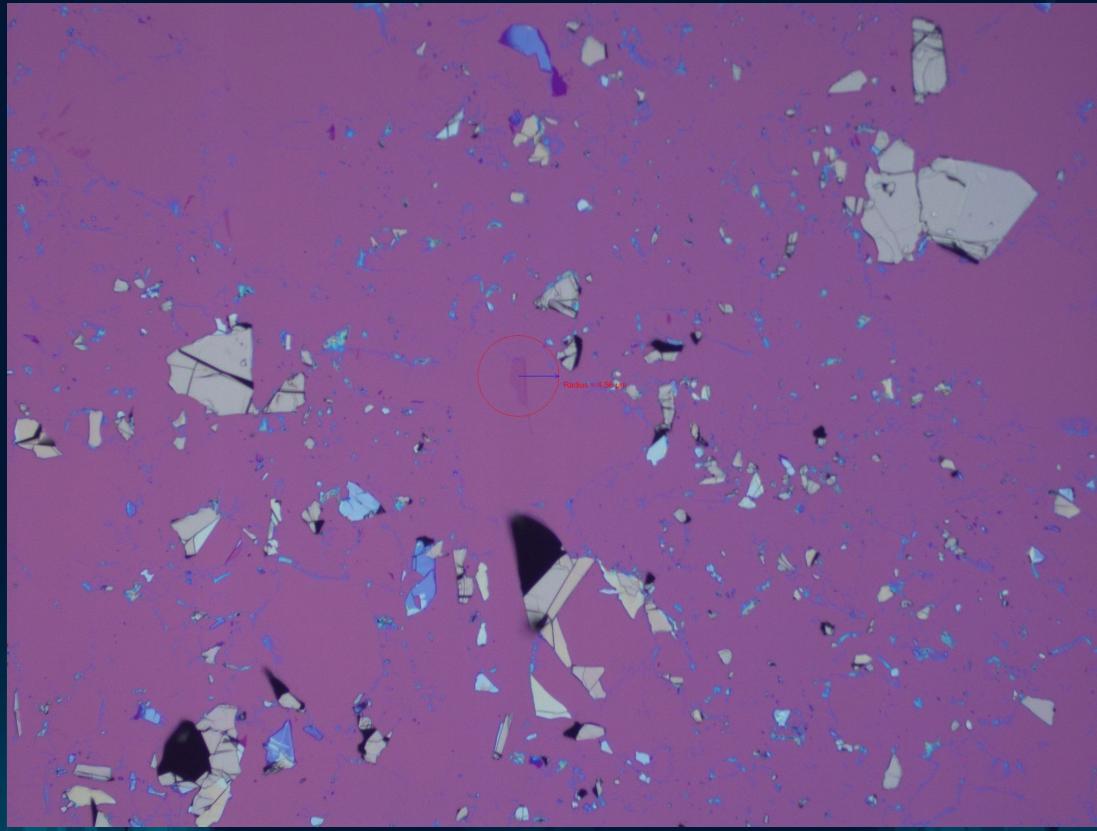


State of the Field- Data Parsing



Jessen, B.S.,
Whelan, P.R.,
Mackenzie,
D.M.A. et al.
Quantitative
optical
mapping of
two-dimensio
nal materials.
Sci Rep 8, 6381
(2018).
[https://doi.org/
10.1038/s41598-
018-23922-1](https://doi.org/10.1038/s41598-018-23922-1)

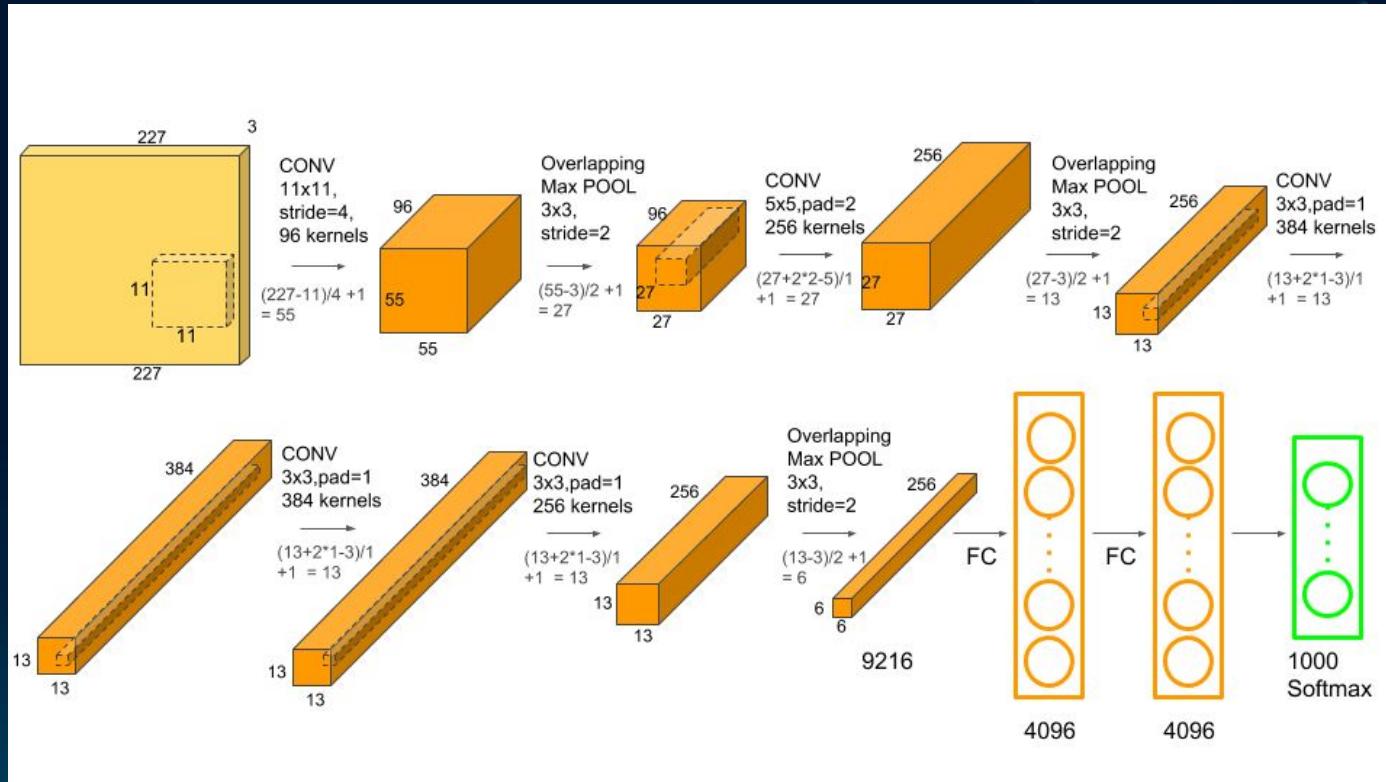
Data Cleaning Methodology



Timeline

2010	2012	2014	2015	2016	2017	2018	2019	2023
•First ImageNet challenge	•AlexNet	•Adam •VGG •GoogleNet	•ResNet •Batch Norm. •GoogleNet renamed Inception •Yolo V1	•Inception v3	•AdamW •Transformer •Last Imagenet Challenge	•Mobile Net v2 (Inverted Residual)	•Efficient Net	•Yolo V8

AlexNet



Askari, Syed Sajjad.
“Alex-Net
Explanation and
Implementation in
Tensorflow and
Keras.” Medium,
Medium, 14 Jan.
2023,
medium.com/@syedsajjad62/alex-net-explanation-and-implementation-in-tensorflow-and-keras-8047efeb7a0f.

Convolution



Sanderson,
Grant. "But
What is a
Convolution?"
Youtube,
3blue1brown,
29 Dec. 2023,
[https://www.youtub...
e.com/watch?v=kpG7I2MOcnI](https://www.youtube.com/watch?v=kpG7I2MOcnI).

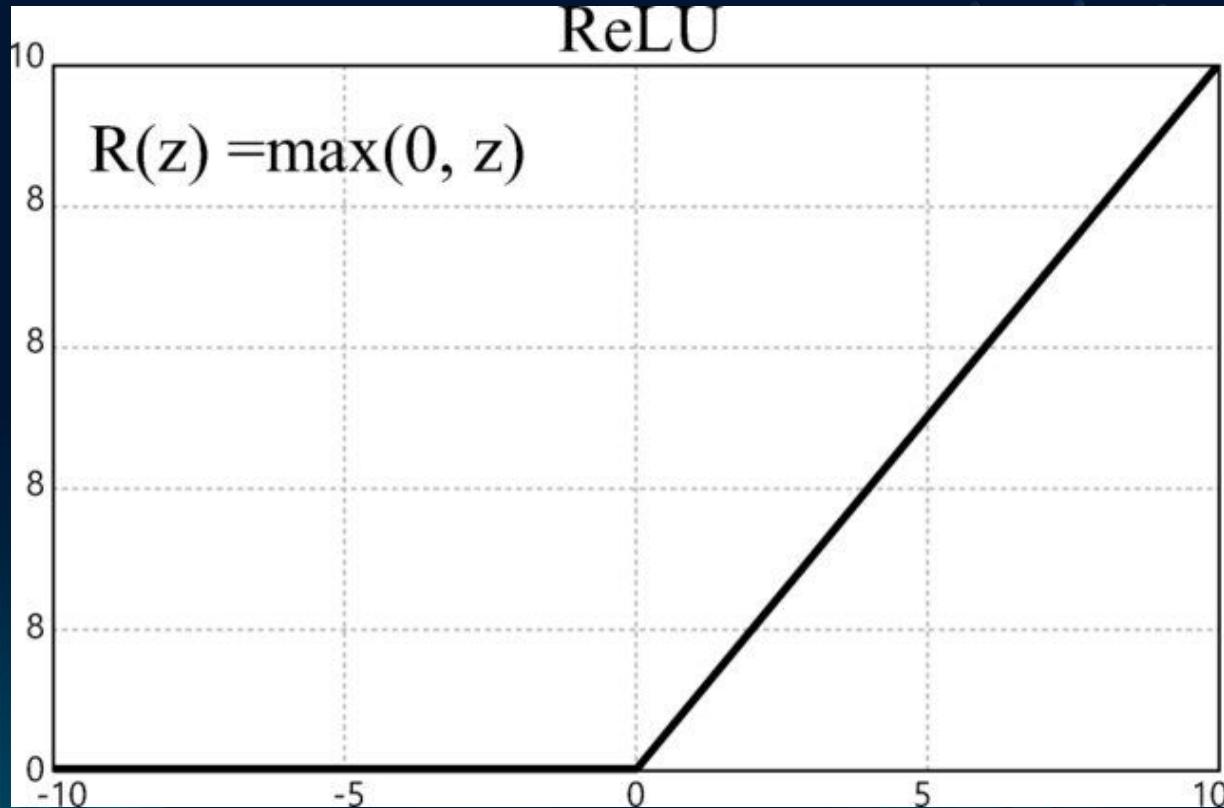
Max Pooling

MAX
POOLING

Ave Coders.
“101 Concepts
of Data
Science and
Machine
Learning”
Youtube,
AveCoders,
28 May. 2024,
[https://www.
youtube.com/
shorts/ykt5PUxs3z8.](https://www.youtube.com/shorts/ykt5PUxs3z8)

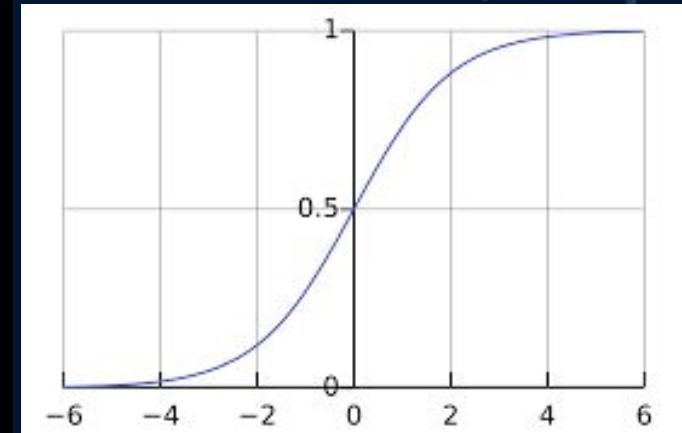
Rectified Linear Function

ReLU



Chima, Precious. "Activation Functions: ReLU & Softmax." Medium, Medium, 5 Apr. 2020, <https://medium.com/@preshchima/activation-functions-relu-softmax-87145bf39288>.

Softmax- Reduces to Sigmoid in Binary Case



$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Alexander, Giffah. "Softmax Activation function explained" Youtube, Giffah_Alexander, 22 Oct. 2024, https://www.youtube.com/shorts/SrJN_hpiuAs.

Sigmoid function. "Sigmoid function." Wikipedia, Wikipedia, 14 Nov. 2024, https://en.wikipedia.org/wiki/Sigmoid_function.

Local Response Normalization

$$b_{x,y}^i = a_{x,y}^i / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a_{x,y}^j)^2 \right)^\beta$$

**For AlexNet, k=2, n=5, alpha=10^-4,
beta=0.75**

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. <https://doi.org/10.1145/3065386>

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_1 \dots m\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Ioffe, Sergey. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).

Input: Network N with trainable parameters Θ ;
subset of activations $\{x^{(k)}\}_{k=1}^K$

Output: Batch-normalized network for inference, $N_{\text{BN}}^{\text{inf}}$

- 1: $N_{\text{BN}}^{\text{tr}} \leftarrow N \quad // \text{Training BN network}$
- 2: **for** $k = 1 \dots K$ **do**
- 3: Add transformation $y^{(k)} = \text{BN}_{\gamma^{(k)}, \beta^{(k)}}(x^{(k)})$ to $N_{\text{BN}}^{\text{tr}}$ (Alg. 1)
- 4: Modify each layer in $N_{\text{BN}}^{\text{tr}}$ with input $x^{(k)}$ to take $y^{(k)}$ instead
- 5: **end for**
- 6: Train $N_{\text{BN}}^{\text{tr}}$ to optimize the parameters $\Theta \cup \{\gamma^{(k)}, \beta^{(k)}\}_{k=1}^K$
- 7: $N_{\text{BN}}^{\text{inf}} \leftarrow N_{\text{BN}}^{\text{tr}} \quad // \text{Inference BN network with frozen parameters}$
- 8: **for** $k = 1 \dots K$ **do**
- 9: // For clarity, $x \equiv x^{(k)}, \gamma \equiv \gamma^{(k)}, \mu_{\mathcal{B}} \equiv \mu_{\mathcal{B}}^{(k)}$, etc.
- 10: Process multiple training mini-batches \mathcal{B} , each of size m , and average over them:

$$\text{E}[x] \leftarrow \text{E}_{\mathcal{B}}[\mu_{\mathcal{B}}]$$

$$\text{Var}[x] \leftarrow \frac{m}{m-1} \text{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$$
- 11: In $N_{\text{BN}}^{\text{inf}}$, replace the transform $y = \text{BN}_{\gamma, \beta}(x)$ with

$$y = \frac{\gamma}{\sqrt{\text{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma \text{E}[x]}{\sqrt{\text{Var}[x]+\epsilon}}\right)$$
- 12: **end for**

Algorithm 2: Training a Batch-Normalized Network

Logits and Sigmoid Function

Logits: Raw outputs of neural networks.

- Numerical values, hard to interpret

Sigmoid function: $\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$

- Turns logits into probabilities, easier to interpret

Binary Cross Entropy Loss

$$L = -\frac{1}{N} \sum_{i=1}^N [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

- N is the number of samples
- y_i is the true label for the i^{th} sample (0 or 1)
- p_i is the predicted probability that the i^{th} sample belongs to Positive class

Sahilcarterr. "Mathematics Behind Simple Linear Regression using Gradient Descent." Medium, Medium, 5 Apr. 2020,
[https://medium.com/@sahilcarterr/mathematics-behind-simple-linear-regression-using-gradient-desc
ent-a09595bf701d](https://medium.com/@sahilcarterr/mathematics-behind-simple-linear-regression-using-gradient-descent-a09595bf701d).

Binary Cross Entropy with Logits Loss

$$\ell_n := \begin{cases} w_n \cdot [(1 - y_n) \cdot x_n + \ln(1 + e^{-x_n})] & \text{if } x_n > 0 \\ w_n \cdot [-x_n \cdot y_n + \ln(e^{x_n} + 1)] & \text{otherwise} \end{cases}$$

jodag. "How Is Pytorch's Class BCEWITHLOGITSLOSS Exactly Implemented?" *Stack Overflow*, 11 July 2022, stackoverflow.com/questions/66906884/how-is-pytorchs-class-bcewithlogitsloss-exactly-implemented.

Backpropagation

Algorithm 1 Backpropagation

```
1: procedure BACKPROPAGATION( $X, \theta$ )
2:    $a^0 \leftarrow x$                                      ▷ Set Input
3:   for  $l \leftarrow [1, L]$  do
4:      $z^l \leftarrow W^l a^{l-1} + b^l$                   ▷ Forward Pass: Compute weighted inputs and activations
5:      $a^l \leftarrow \sigma(z^l)$ 
6:   end for
7:
8:    $\delta_L \leftarrow (a^L - y) \odot \sigma'(z^L)$           ▷ Compute Error in Final Layer
9:   for  $l \leftarrow [L-1, 1]$  do
10:     $\delta^l \leftarrow (W^{l+1})^T \delta^{l+1} \odot \sigma'(z^l)$  ▷ Backpropagate Error
11:   end for
12:
13:   for  $l \leftarrow [1, L]$  do
14:      $\nabla_{b^l} = \delta^l$                             ▷ Compute gradients w.r.t. network parameters
15:      $\nabla_{W^l} = \delta^l (a^{l-1})^T$ 
16:   end for
17:   return  $\nabla_{\theta} = (\nabla_{W^l}, \nabla_{b^l})$ 
18: end procedure
```

Gabbard &
Miller. "Machine
Learning from
Scratch:
Stochastic
Gradient
Descent and
Adam
Optimizer."
18.0851 Project,
accessed 17
November 2024,
https://www.mit.edu/~jgabbard/assets/18085_Project_final.pdf.

ADAM (Adaptive Moment Estimation/ Gradient Descent with Momentum)

Algorithm 1: Adam, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t .

Require: α : Stepsize

Require: $\beta_1, \beta_2 \in [0, 1]$: Exponential decay rates for the moment estimates

Require: $f(\theta)$: Stochastic objective function with parameters θ

Require: θ_0 : Initial parameter vector

$m_0 \leftarrow 0$ (Initialize 1st moment vector)

$v_0 \leftarrow 0$ (Initialize 2nd moment vector)

$t \leftarrow 0$ (Initialize timestep)

while θ_t not converged **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)

$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)

$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)

$\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)

$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)

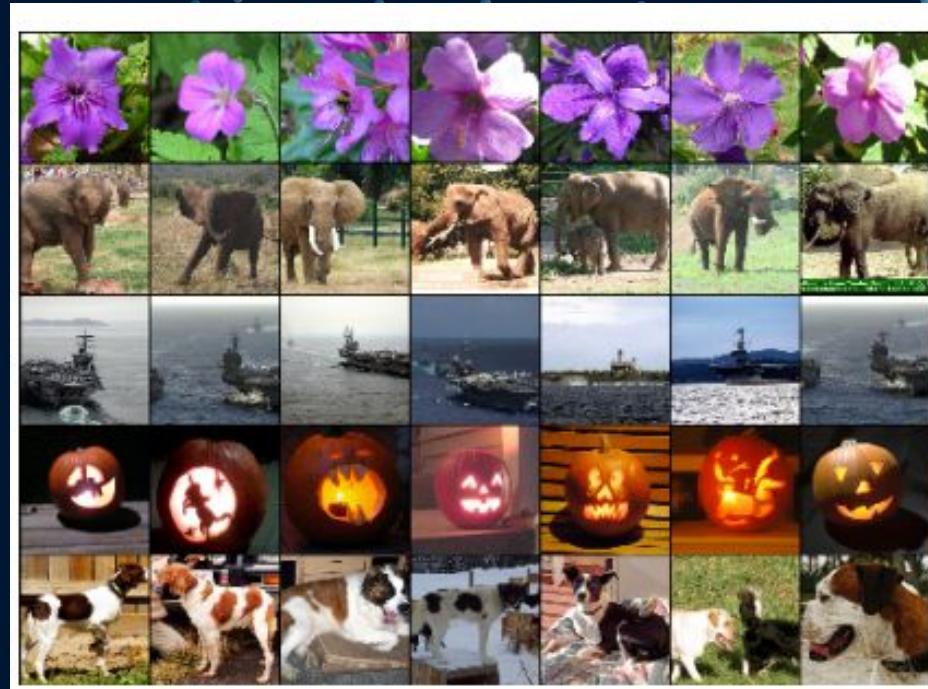
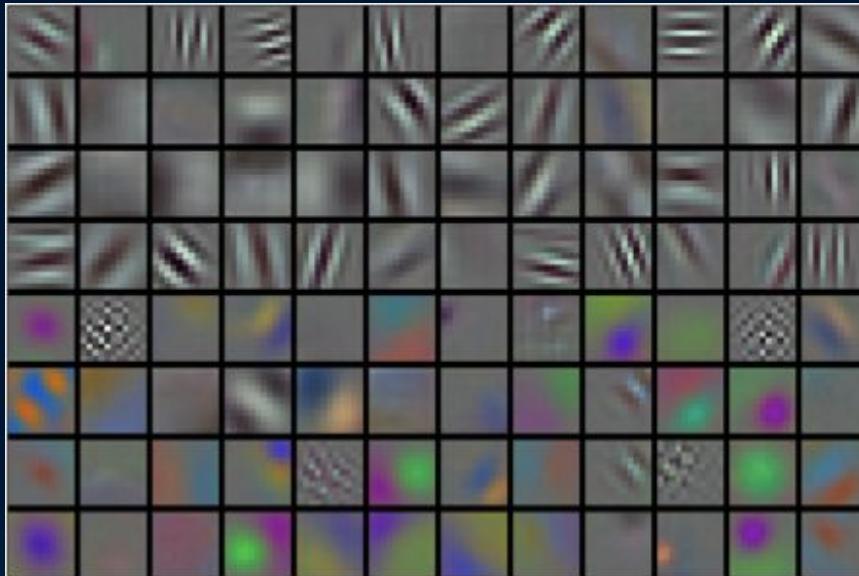
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)

end while

return θ_t (Resulting parameters)

Kingma, Diederik P. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

AlexNet



Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (June 2017), 84–90. <https://doi.org/10.1145/3065386>

Data Structure for Colab Models

Data Structure

Dataset-|

```
| _>train---| _> graphene  
          |_> non_graphene
```

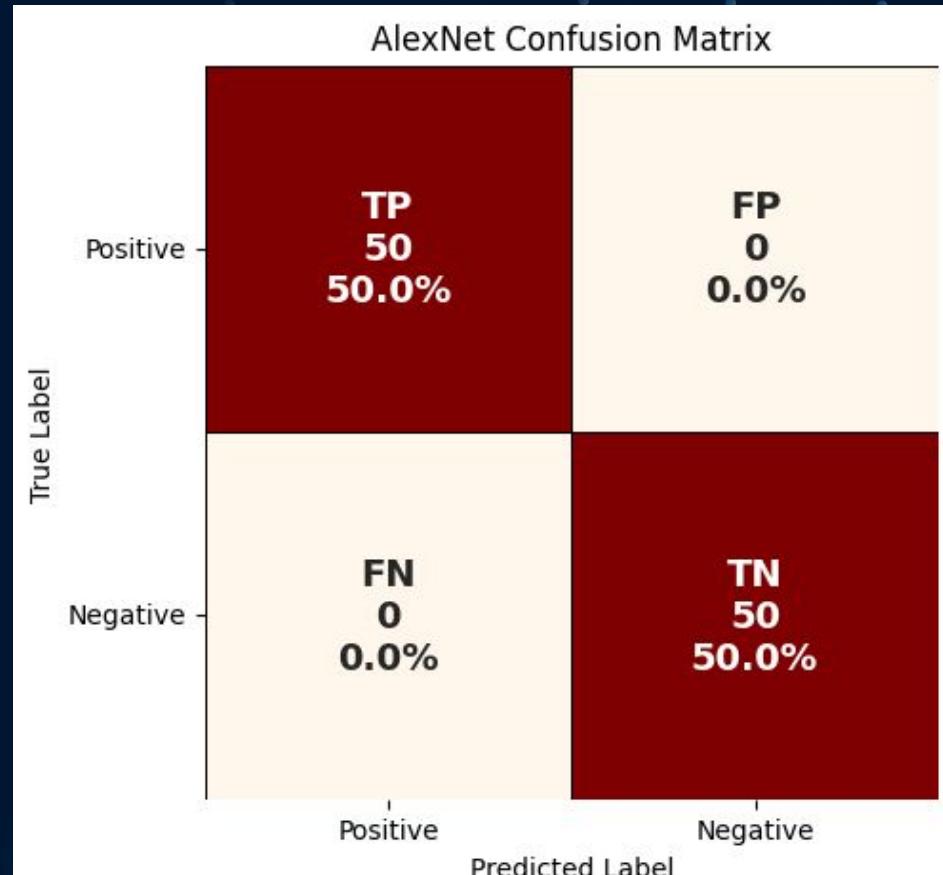
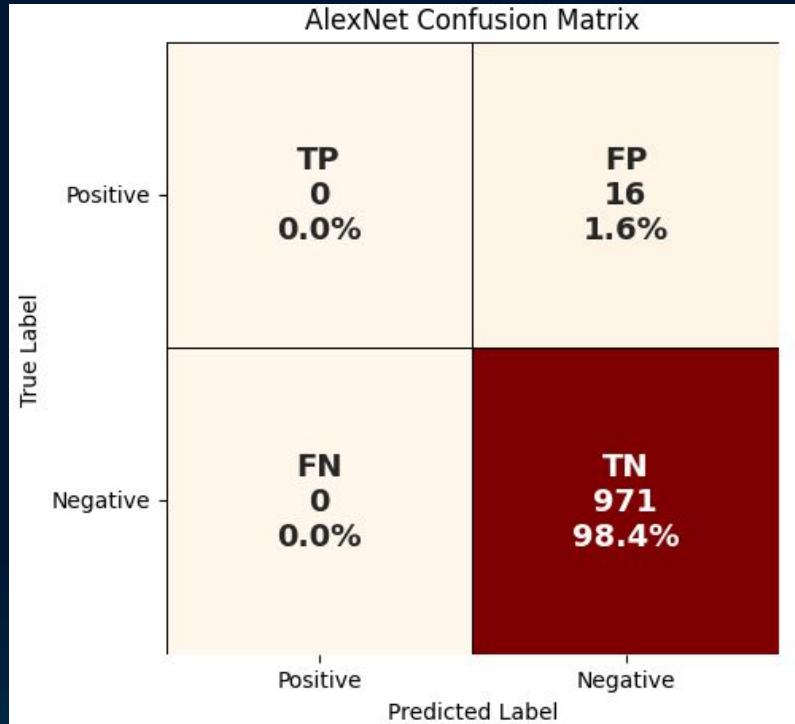
```
| _>test---| _> graphene  
          |_> non_graphene
```

Pre-Trained Models Used & Metrics

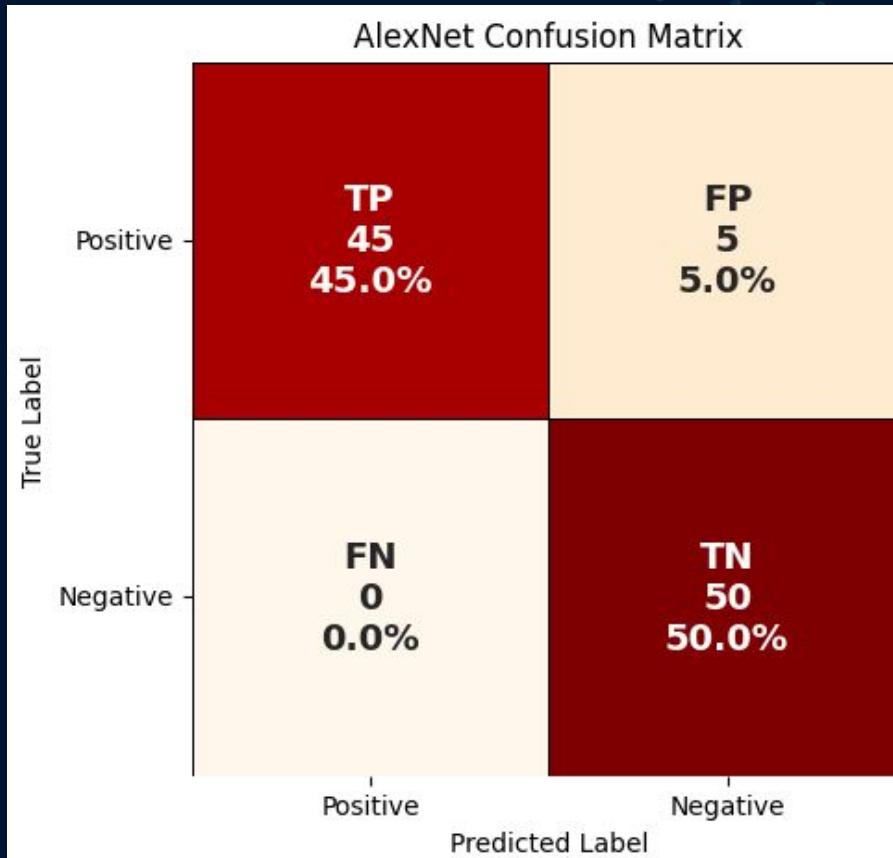
	Metric	AlexNet	VGG16	ResNet50	EfficientNet b0	Inception v3
0	Accuracy	0.98	0.98	0.98	0.99	0.99
1	Precision	0.98	0.98	1.00	0.99	0.99
2	F1	0.99	0.99	0.99	1.00	1.00
3	ROCAUC	0.50	0.40	0.96	1.00	0.97
4	Runtime (sec)	173.46	676.63	484.59	301.84	662.06

	Metric	AlexNet	VGG16	ResNet50	EfficeintNet b0	Inception v3
Accuracy	0.95		0.91	0.84	0.89	0.96
Precision	0.91		0.85	0.77	0.82	0.93
F1	0.95		0.92	0.86	0.90	0.96
AUC of ROC	0.97		0.91	0.99	0.97	1
Runtime(sec)	189.64		418.92	583.52	355.70	816.22
Parameter #	61,100,840		138,357,544	25,557,032	5.3M	27,161,264

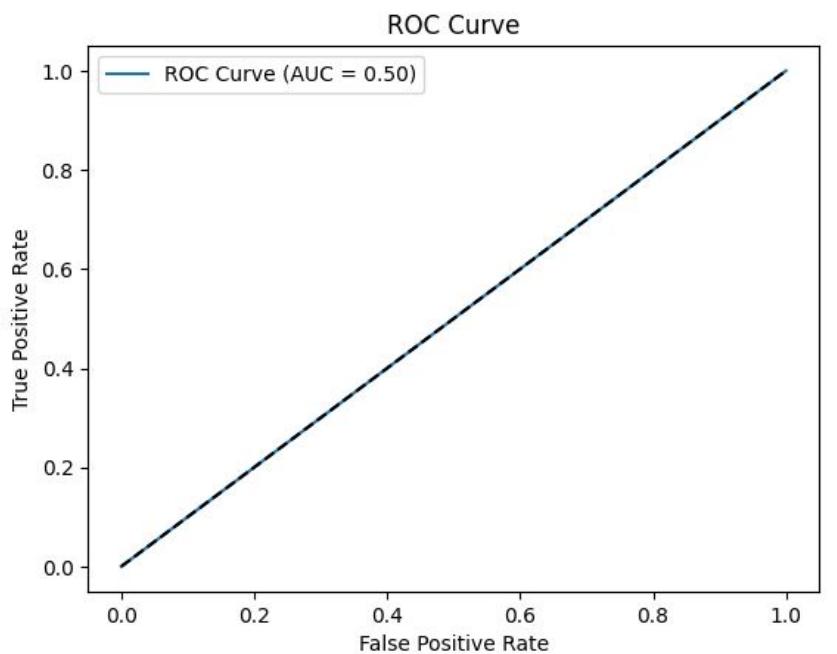
AlexNet Confusion Matrix



AlexNet Confusion Matrix

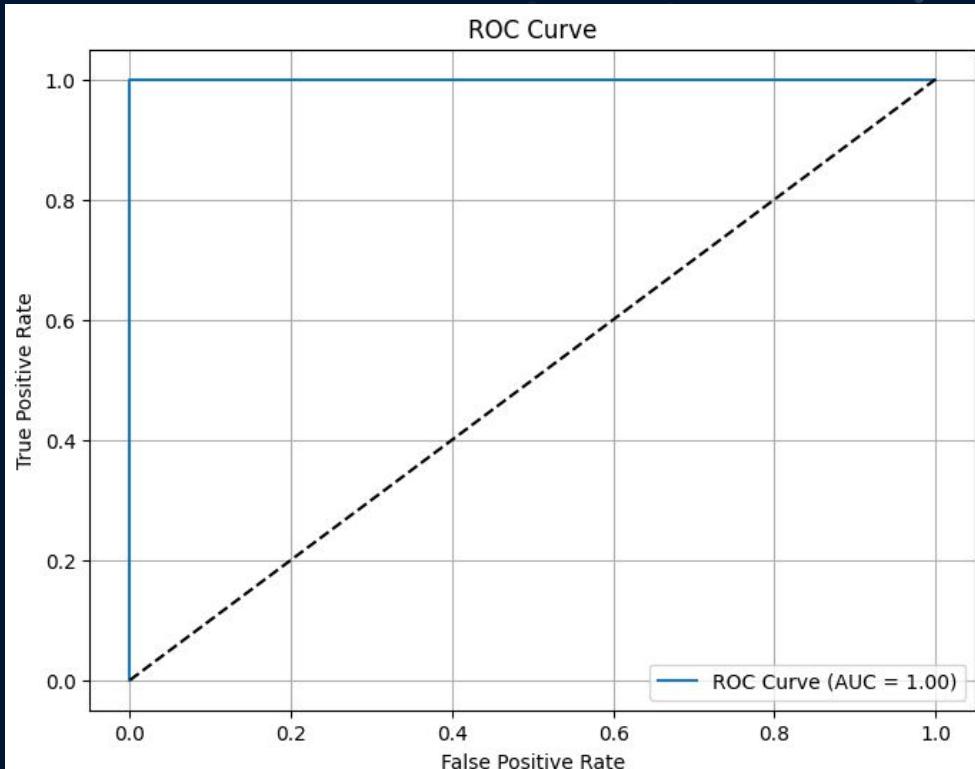


AlexNet ROC/ AUC Plot

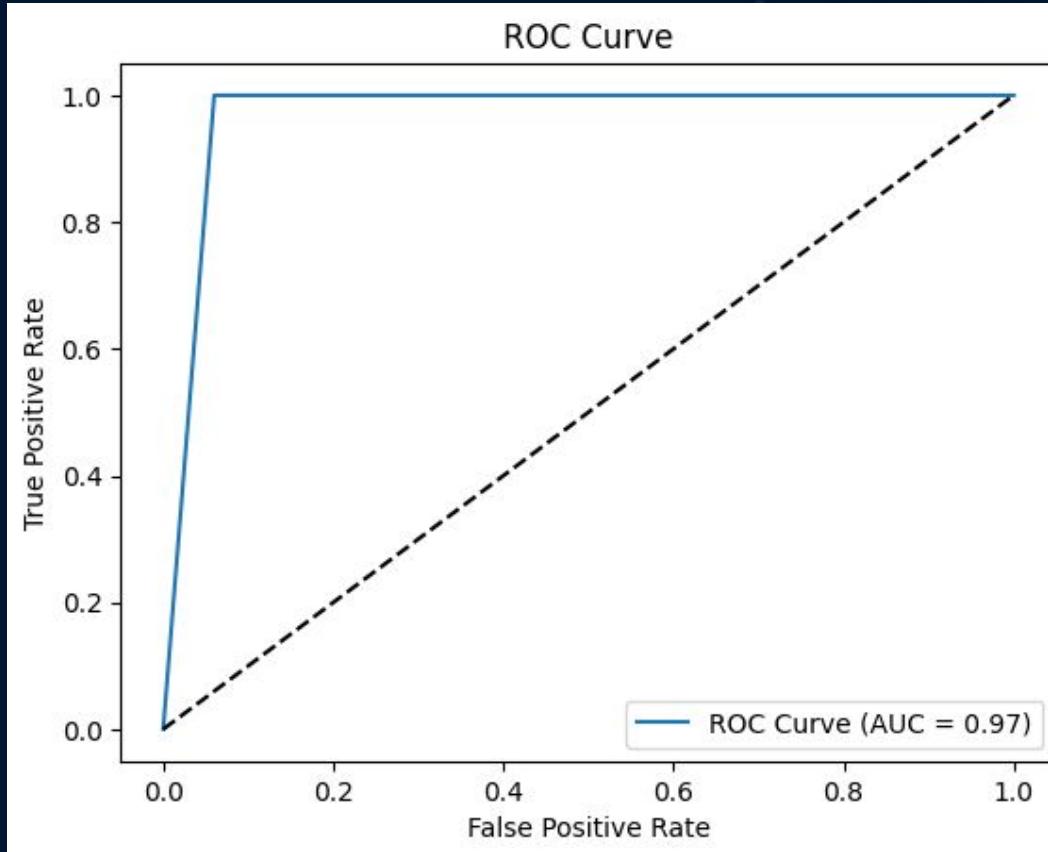


$$TPR = \frac{TP}{TP + FN}$$

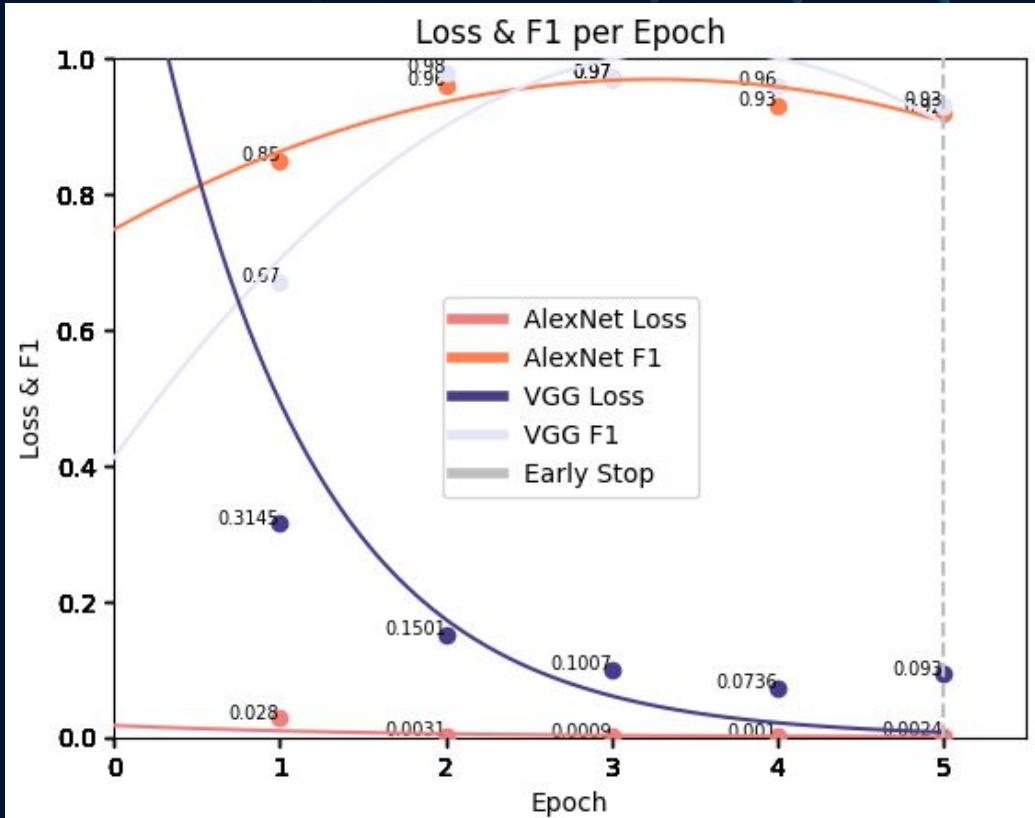
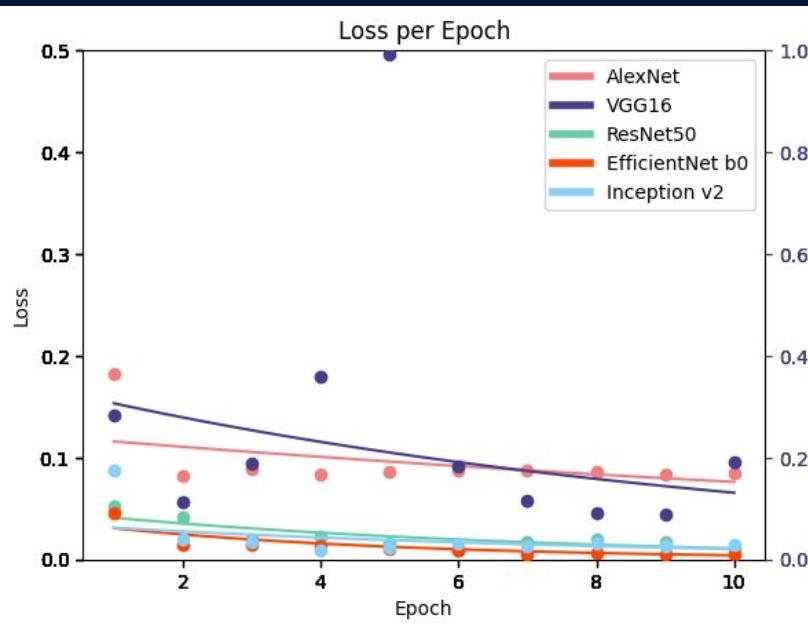
$$FPR = \frac{FP}{FP + TN}$$



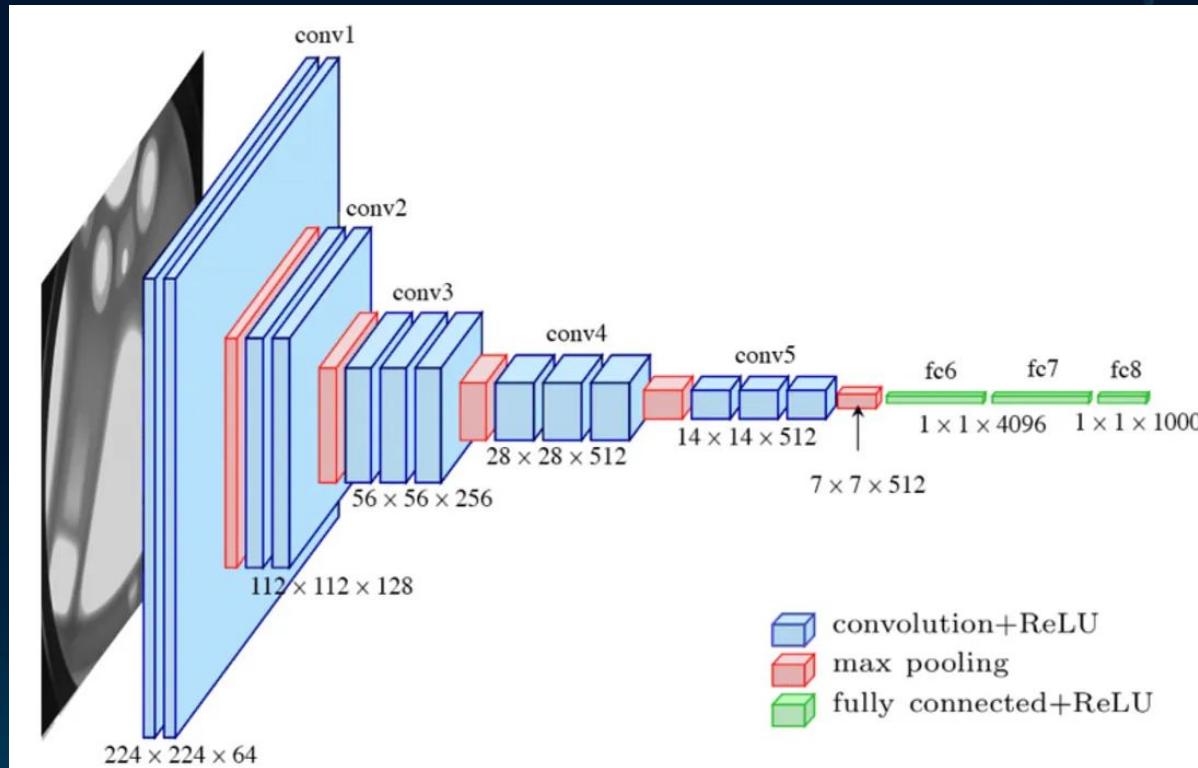
AlexNet ROC/ AUC Plot



Loss Per Epoch w/ Curve Fit

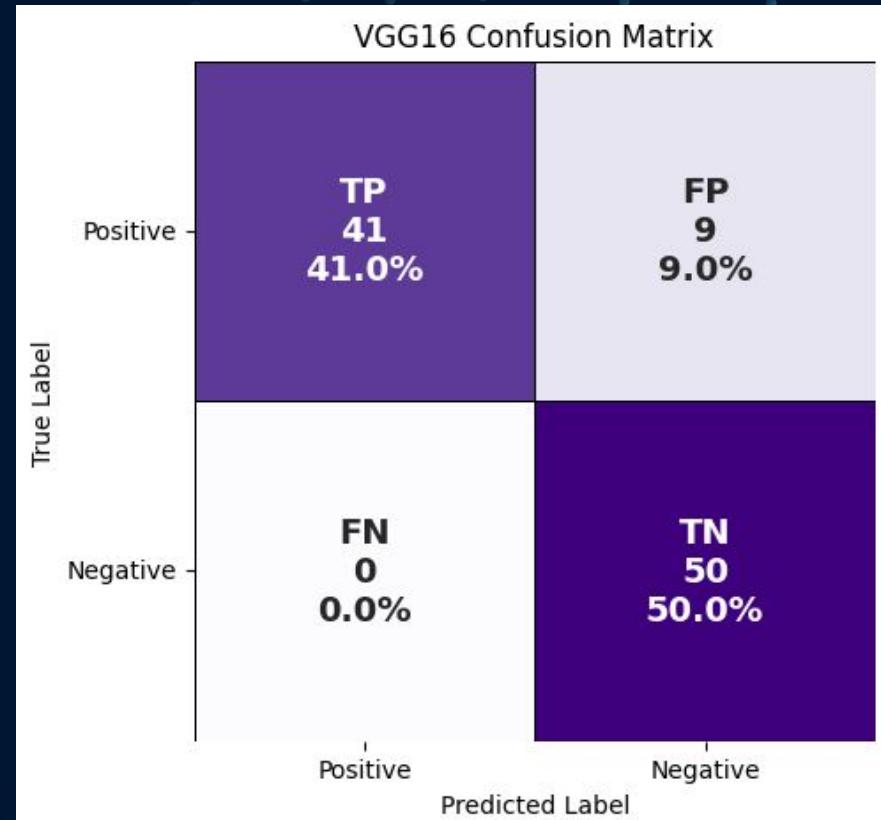
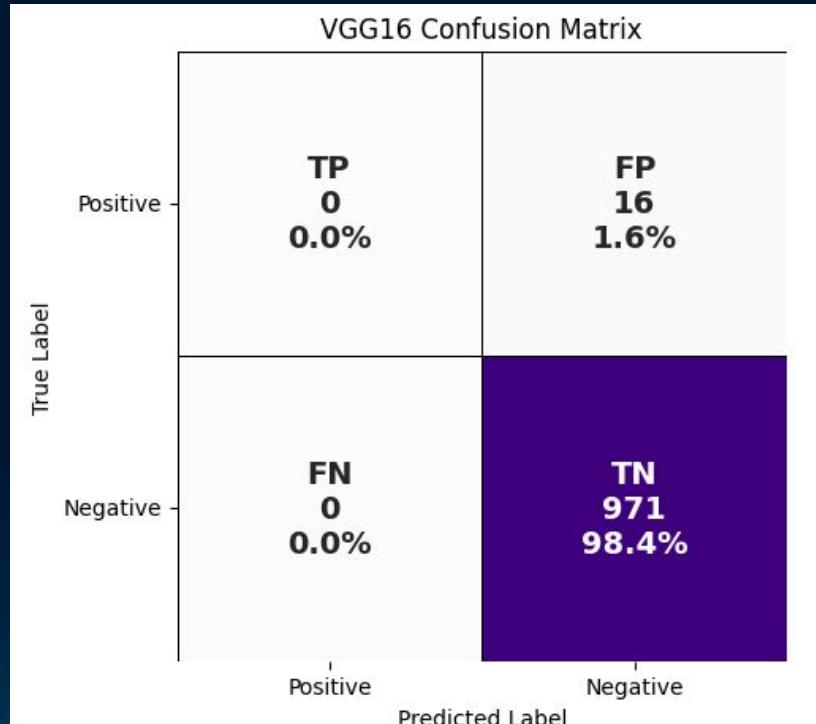


Visual Geometry Group (VGG) 16

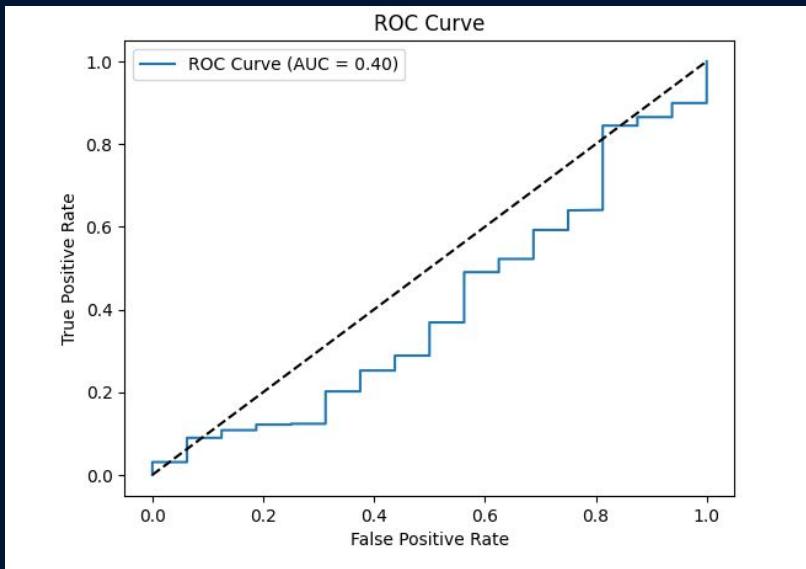


Bangar, Siddhesh. "VGG-Net Architecture Explained." Medium, Medium, 28 Jun. 2022, <https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f>.

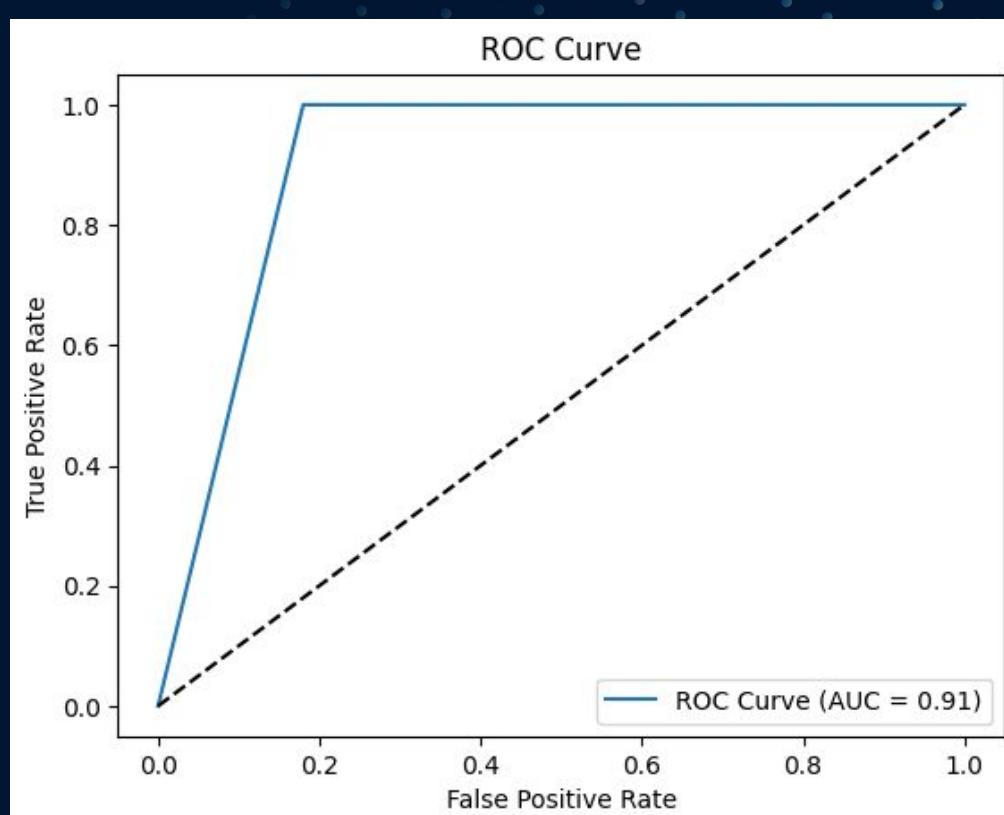
VGG16 Confusion Matrix



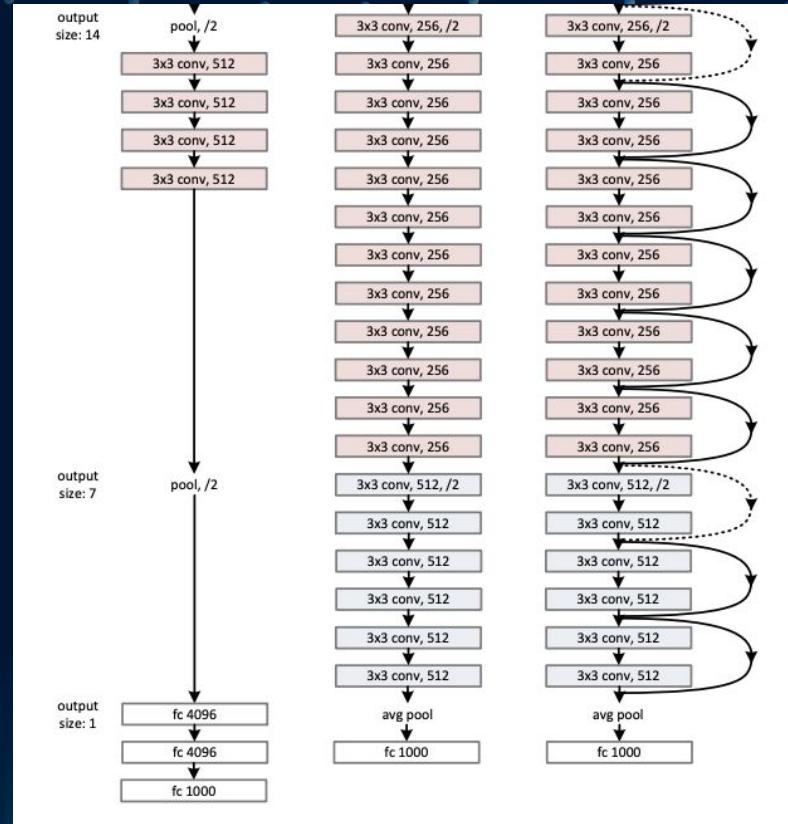
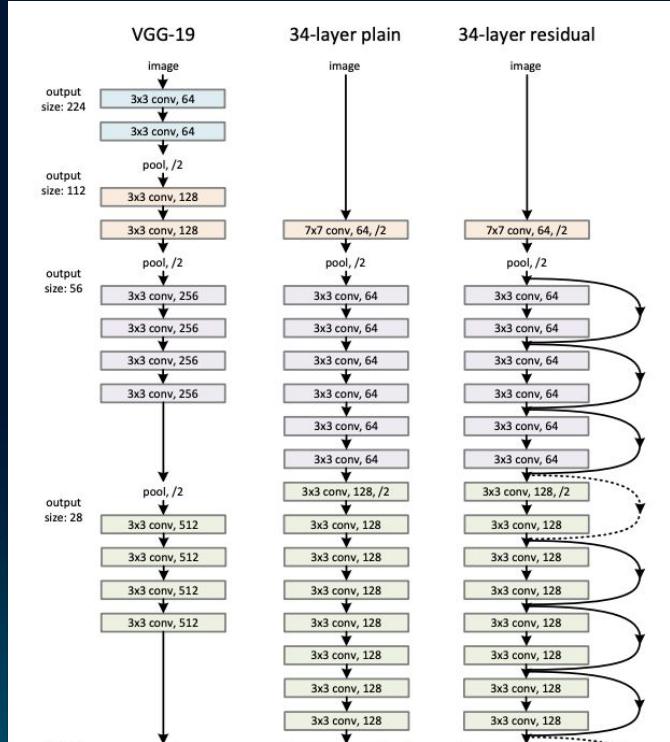
VGG16 ROC/ AUC Plot



Note sign flip^



ResNet Architecture



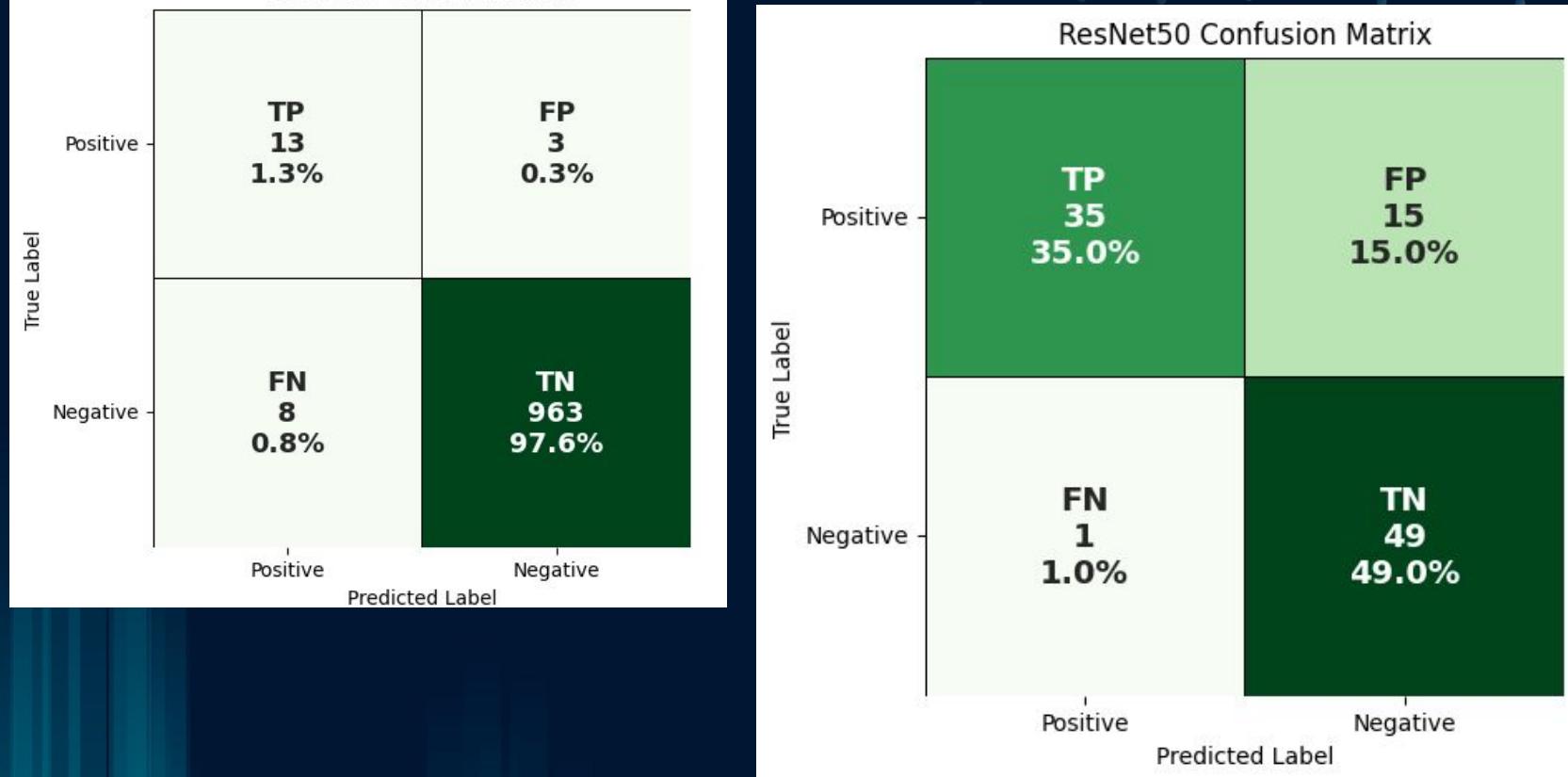
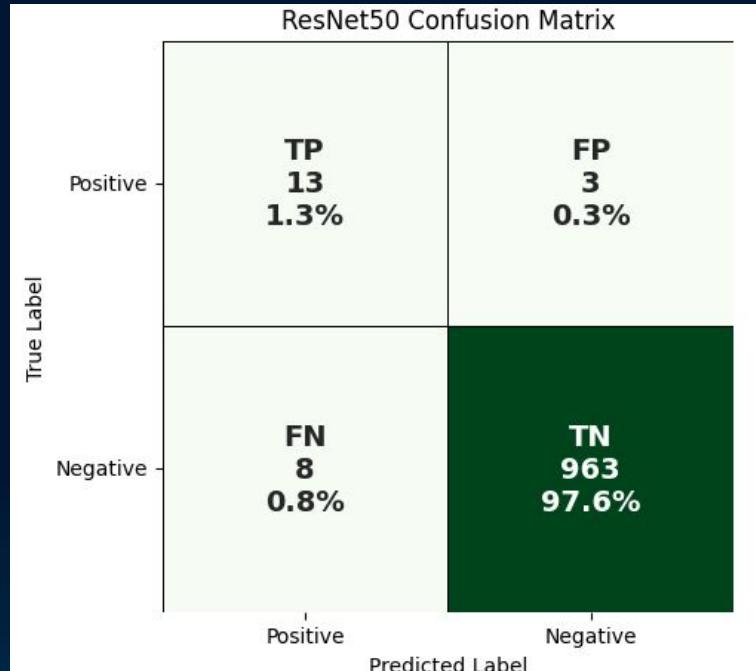
He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

ResNet 50 Architecture

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112			7×7, 64, stride 2		
				3×3 max pool, stride 2		
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1			average pool, 1000-d fc, softmax		
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

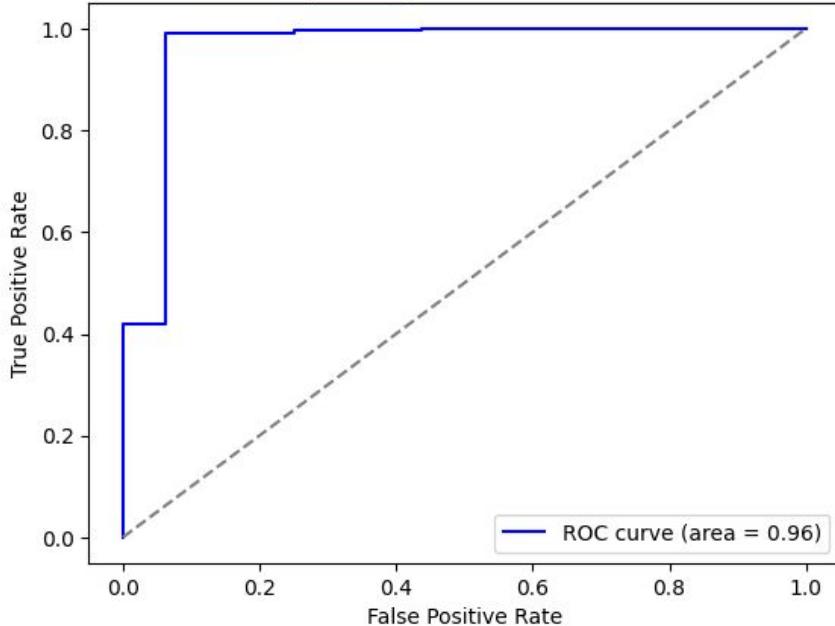
He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

ResNet50 Confusion Matrix

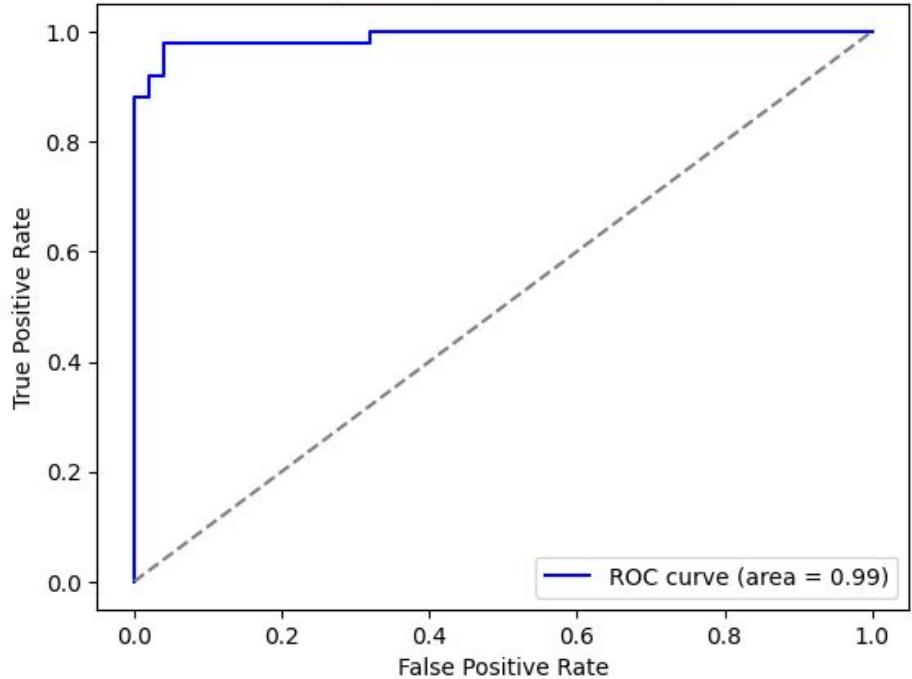


ResNet50 ROC/ AUC Plot

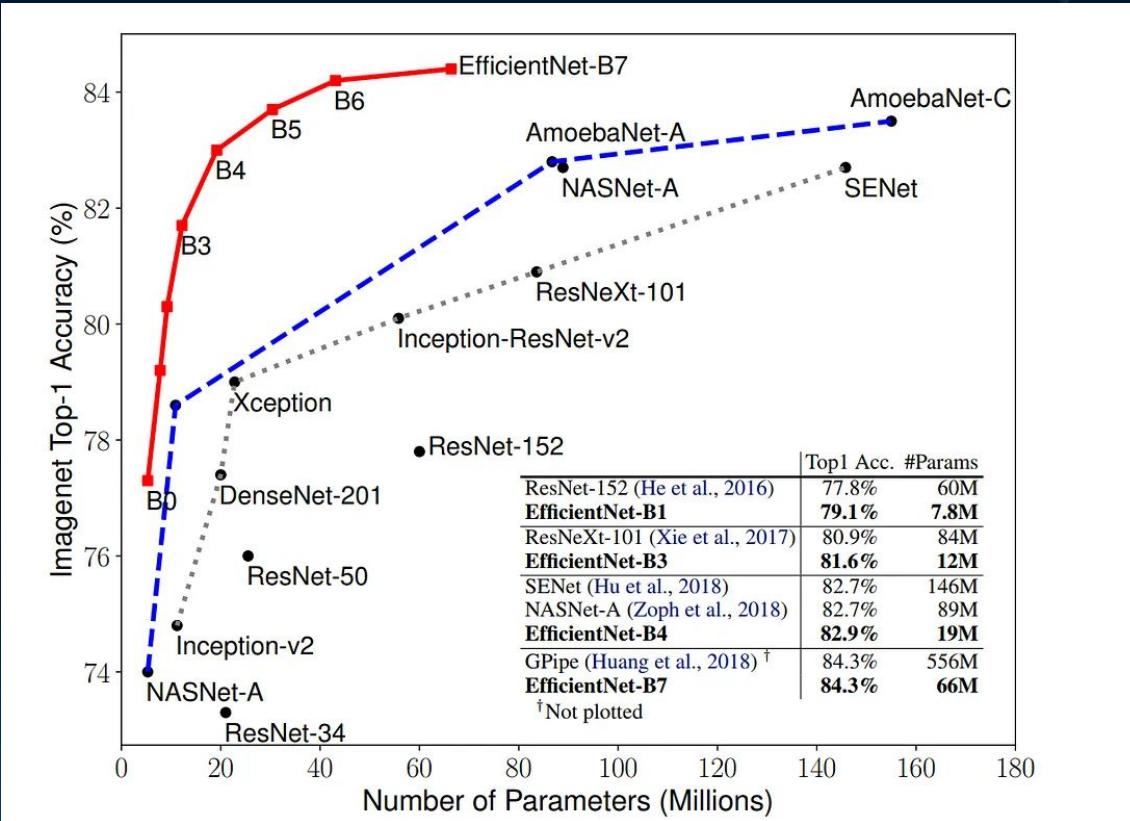
Receiver Operating Characteristic (ROC) Curve



Receiver Operating Characteristic (ROC) Curve



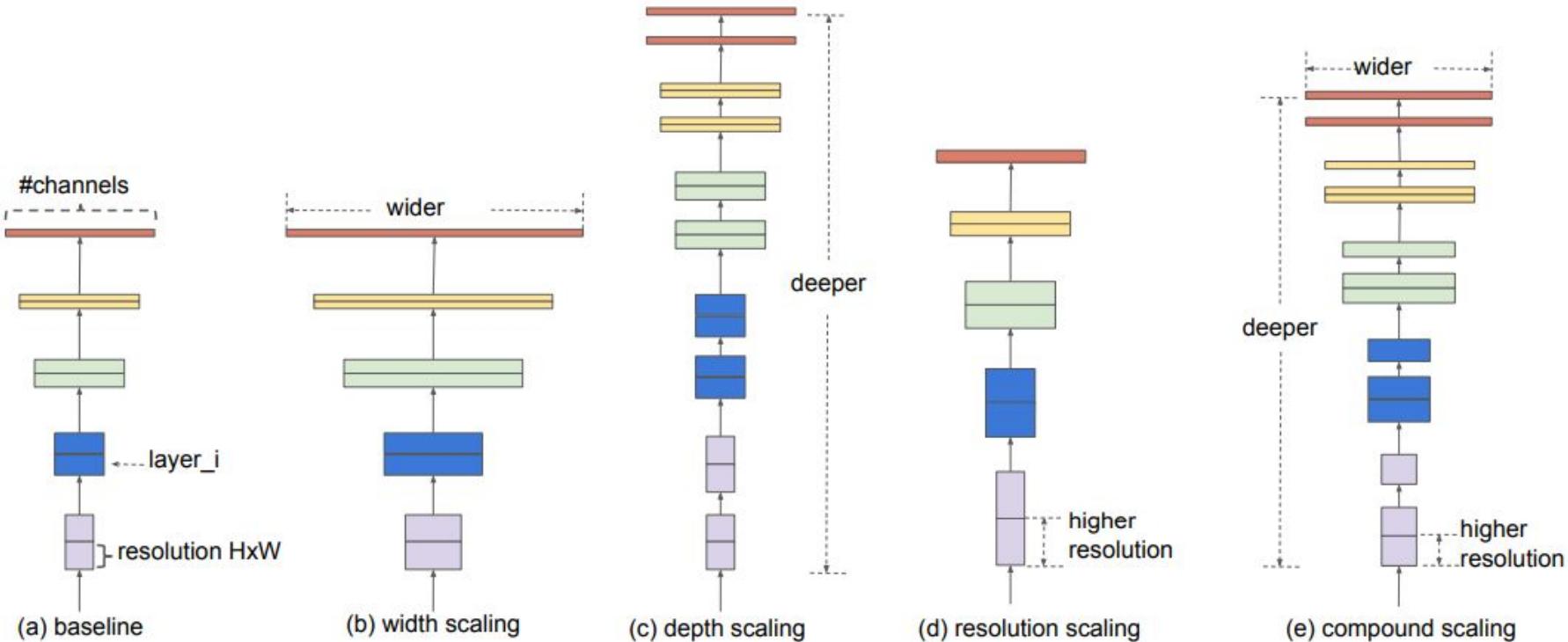
EfficientNet B0



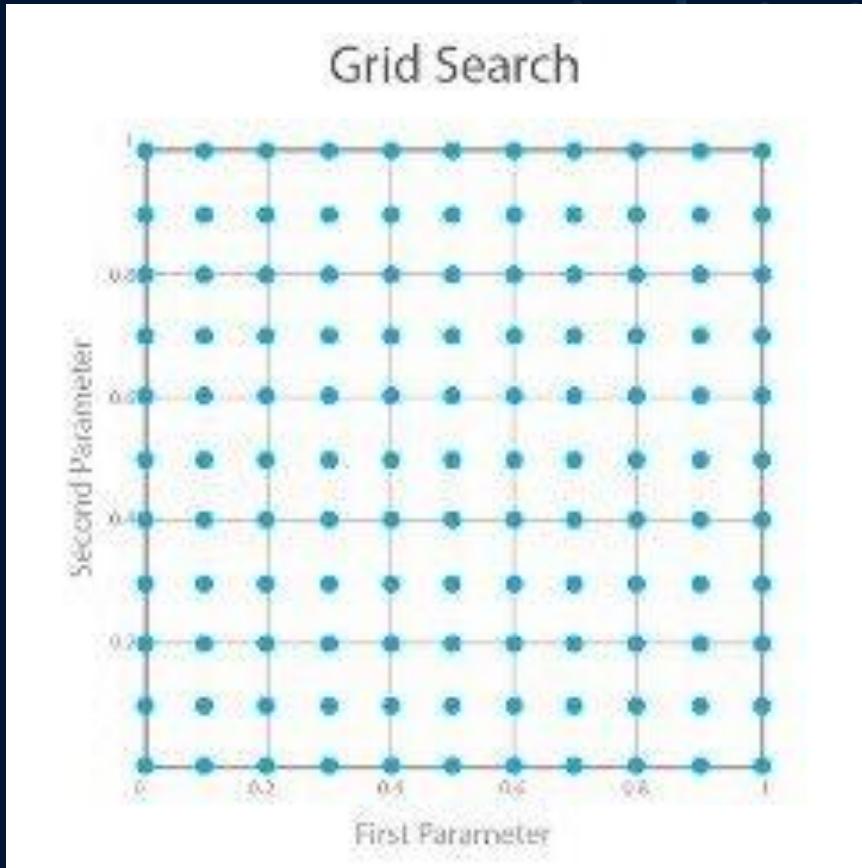
Tan, Mingxing, and Quoc Le.
"Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.

Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.

EfficientNet B0



Grid Search



Hien, Ngo Le Huy. "Grid Search space Illustration" ResearchGate, ResearchGate, https://www.researchgate.net/figure/Grid-Search-space-illustration_fig9_346571789.

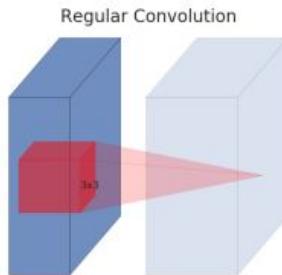
Tan, Mingxing, and Quoc Le.
"Efficientnet: Rethinking model
scaling for convolutional neural
networks." *International conference
on machine learning*. PMLR, 2019.

EfficientNet B0 Architecture

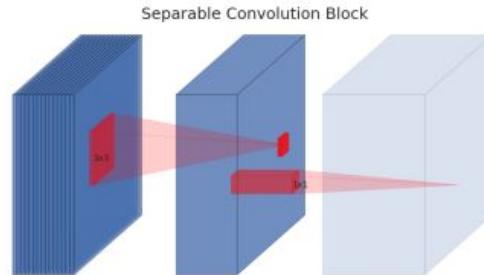
Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels \hat{C}_i	#Layers \hat{L}_i
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Inverted Residual Block

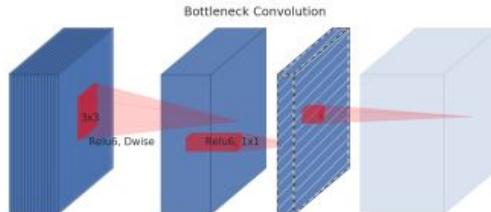
(a) Regular



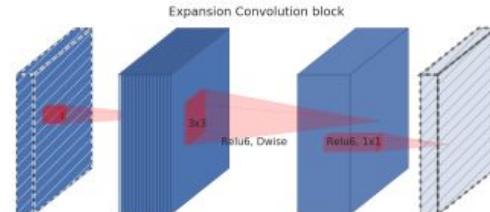
(b) Separable



(c) Separable with linear bottleneck



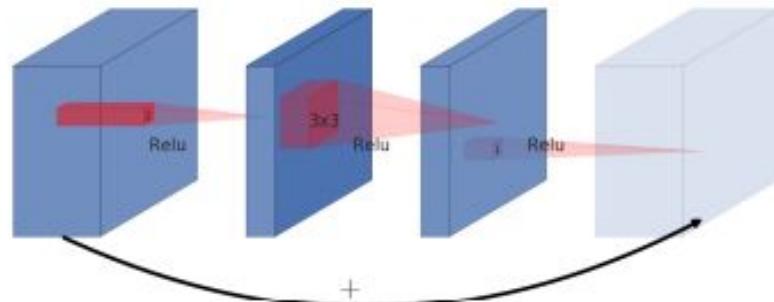
(d) Bottleneck with expansion layer



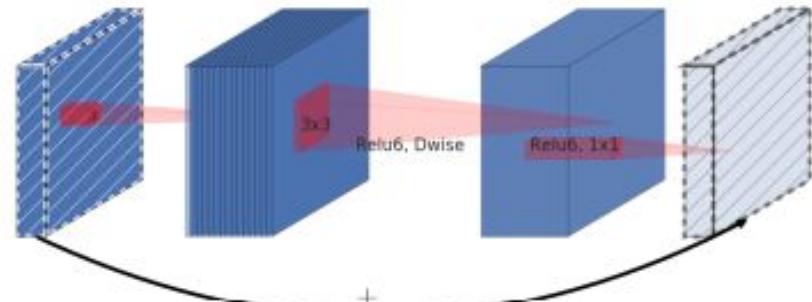
Sandler, Mark, et al.
"Mobilenetv2: Inverted residuals and linear bottlenecks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Inverted Residual Block

(a) Residual block

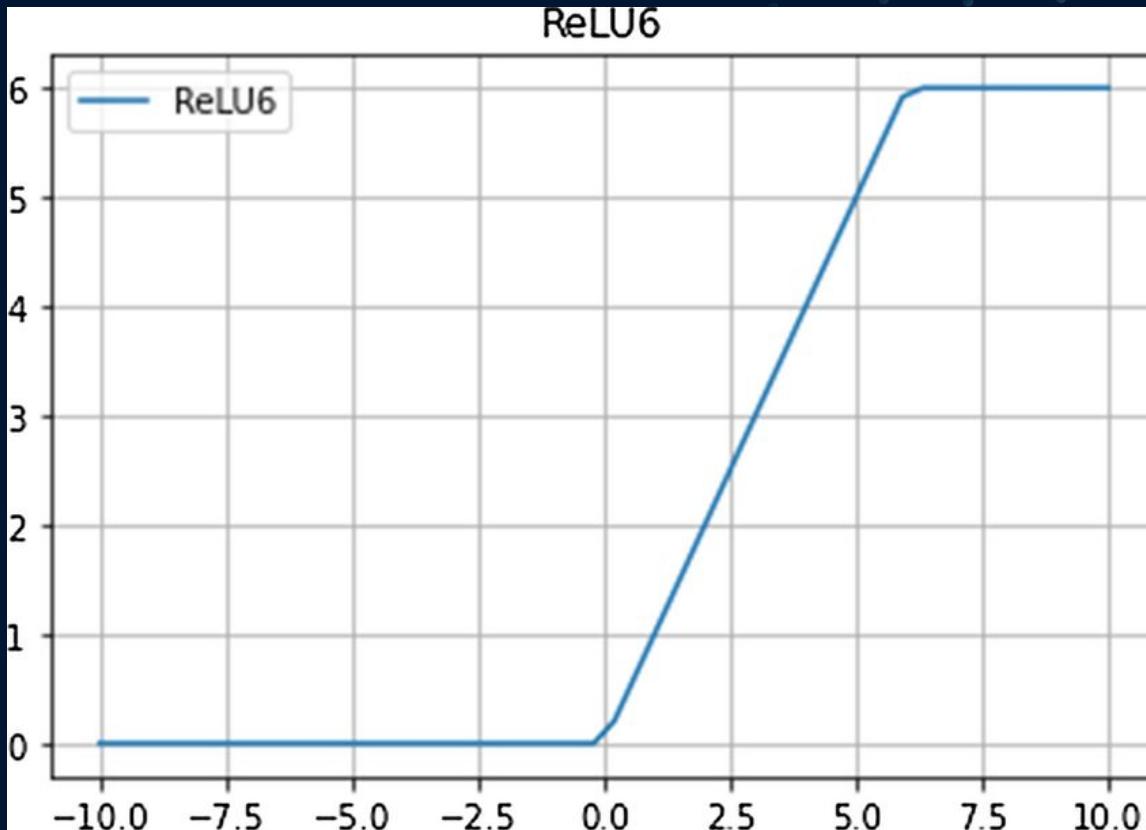


(b) Inverted residual block



Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

ReLU6



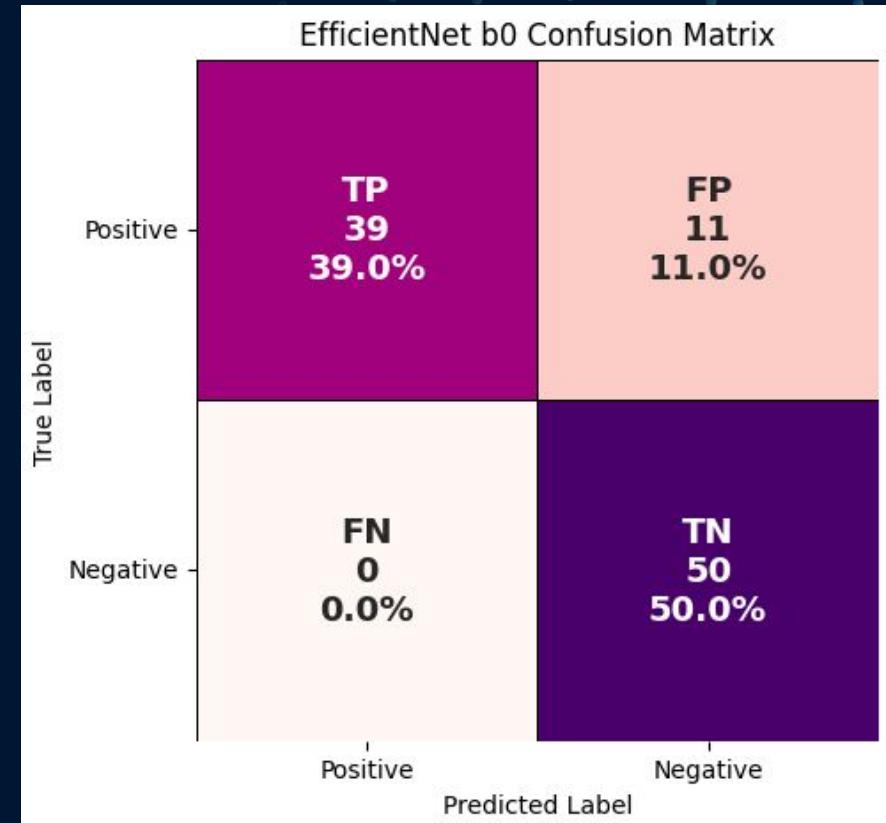
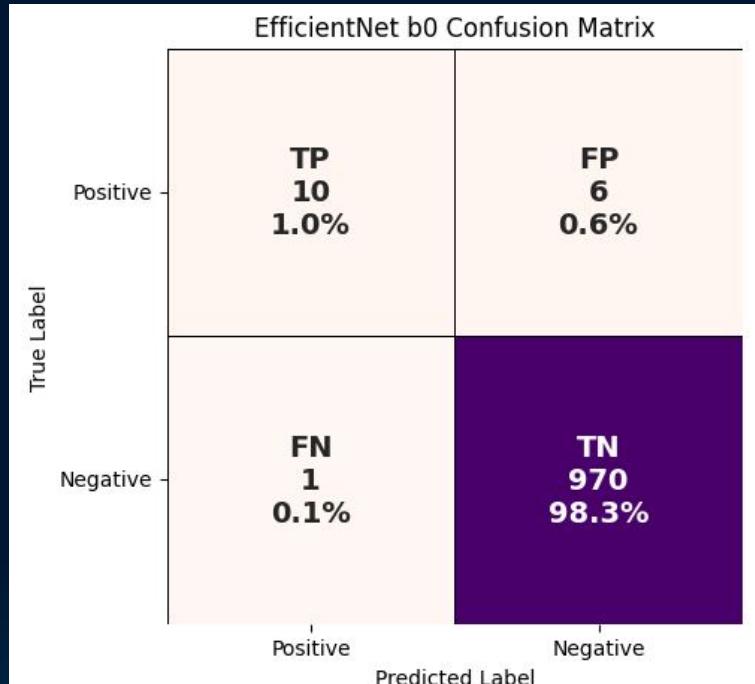
Neural
Computing and
Applications.
“Graph for ReLU6
activation
function”
ResearchGate,
ResearchGate,
https://www.researchgate.net/figure/Graph-for-ReLU6-activation-function_fig5_371374851.

Compound Scaling

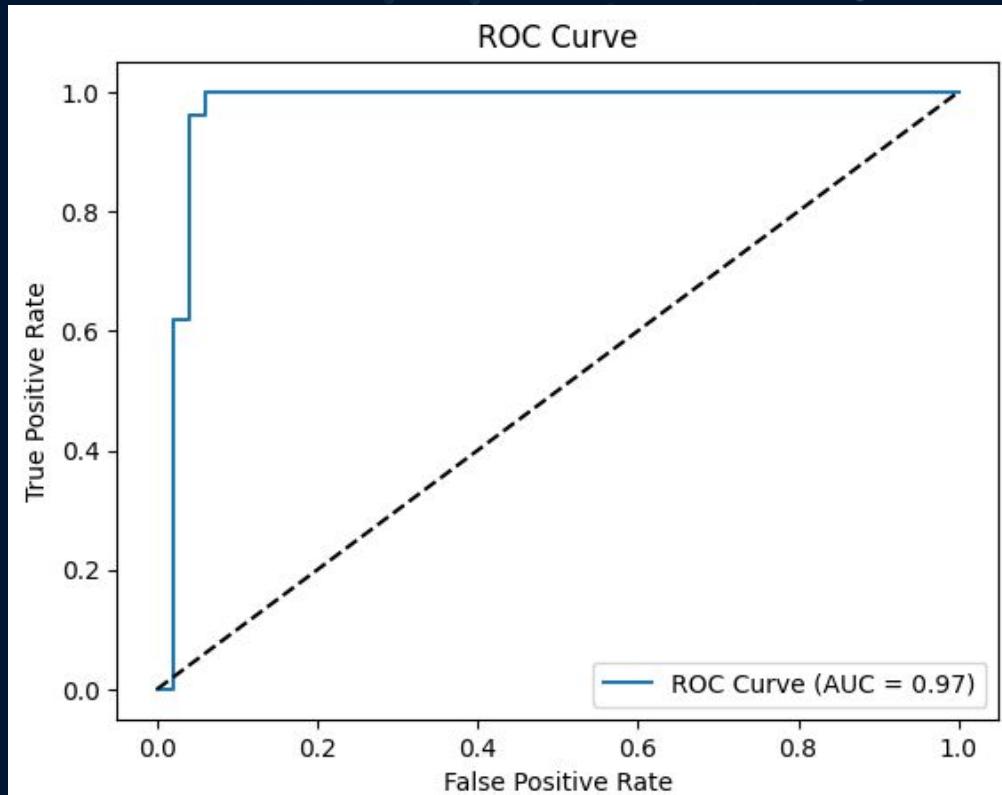
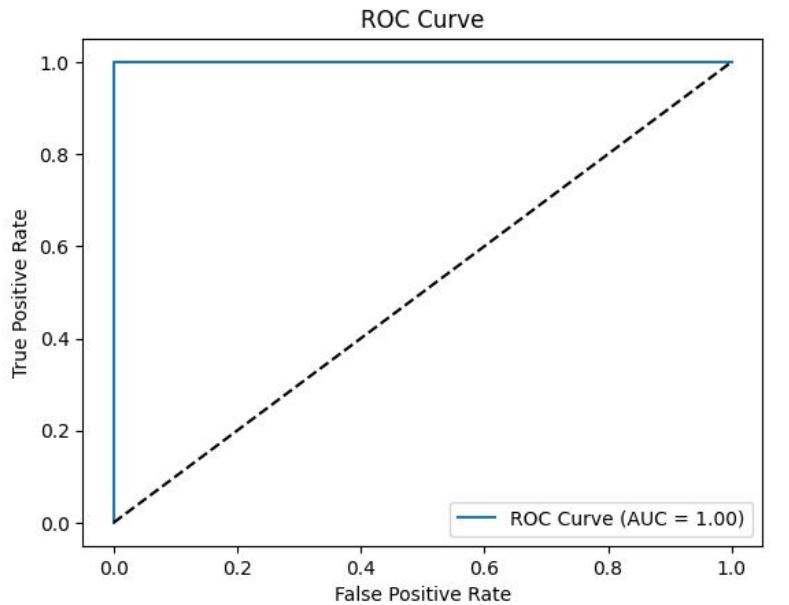
Baseline ResNet-50 (He et al., 2016)	4.1B	76.0%
Scale ResNet-50 by depth ($d=4$)	16.2B	78.1%
Scale ResNet-50 by width ($w=2$)	14.7B	77.7%
Scale ResNet-50 by resolution ($r=2$)	16.4B	77.5%
ResNet-50 compound scale	16.7B	78.8%

Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.

EfficientNet b0 Confusion Matrix



EfficientNet b0 ROC/ AUC Plot



Inception (GoogleNet)

Derevianko, Ivan. "BLAZOR - IT'S TIME TO FORGET JAVASCRIPT" *Ivan Derevianko*, Ivan Derevianko, 16 July 2019,
<https://ivanderevianko.com/2019/07/blazor-its-time-to-forget-javascript>.

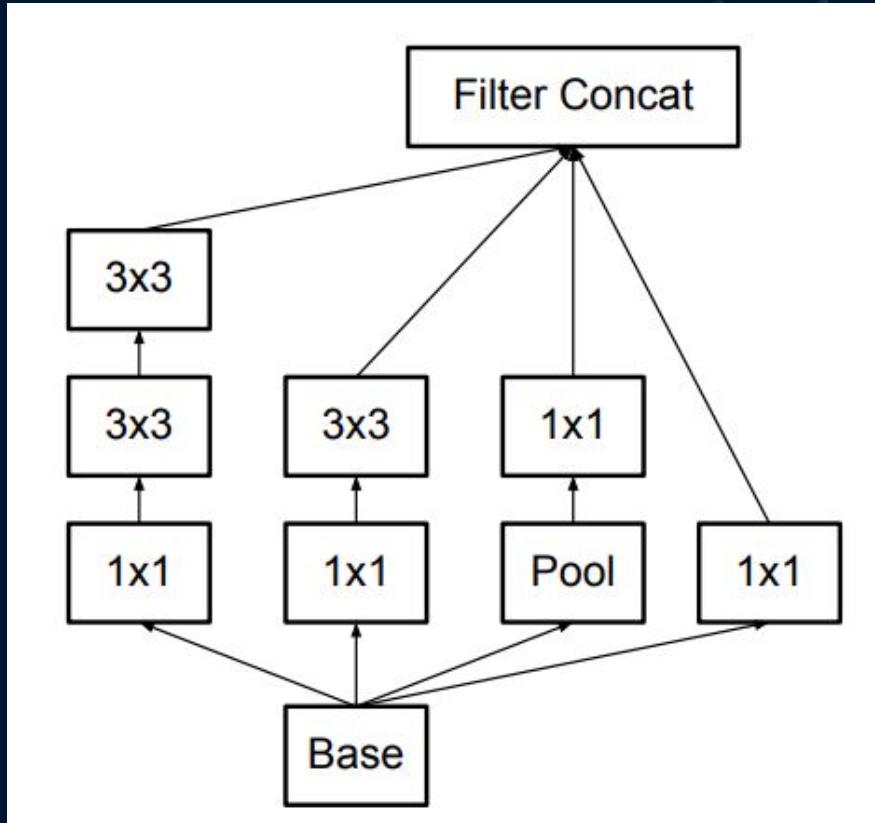


Inception v3 Architecture

type	patch size/stride or remarks	input size
conv	$3 \times 3 / 2$	$299 \times 299 \times 3$
conv	$3 \times 3 / 1$	$149 \times 149 \times 32$
conv padded	$3 \times 3 / 1$	$147 \times 147 \times 32$
pool	$3 \times 3 / 2$	$147 \times 147 \times 64$
conv	$3 \times 3 / 1$	$73 \times 73 \times 64$
conv	$3 \times 3 / 2$	$71 \times 71 \times 80$
conv	$3 \times 3 / 1$	$35 \times 35 \times 192$
$3 \times$ Inception	As in figure 5	$35 \times 35 \times 288$
$5 \times$ Inception	As in figure 6	$17 \times 17 \times 768$
$2 \times$ Inception	As in figure 7	$8 \times 8 \times 1280$
pool	8×8	$8 \times 8 \times 2048$
linear	logits	$1 \times 1 \times 2048$
softmax	classifier	$1 \times 1 \times 1000$

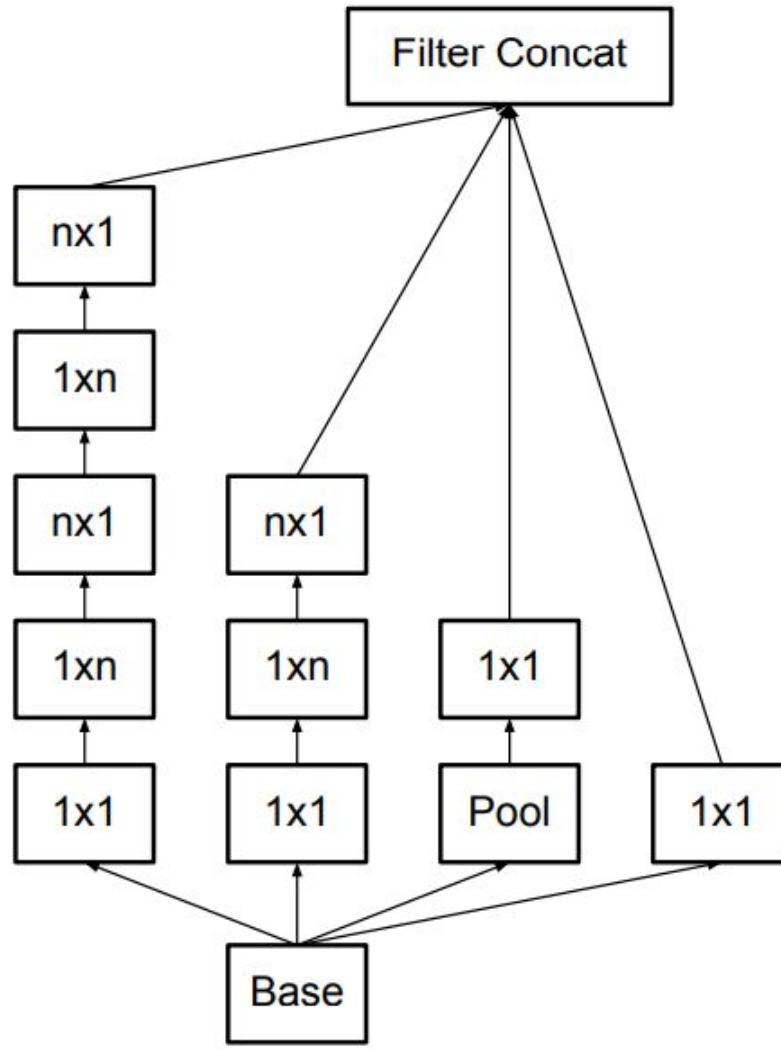
Szegedy, Christian, et al.
"Rethinking the
inception architecture
for computer vision."
*Proceedings of the IEEE
conference on
computer vision and
pattern recognition.*
2016.

Figure 5



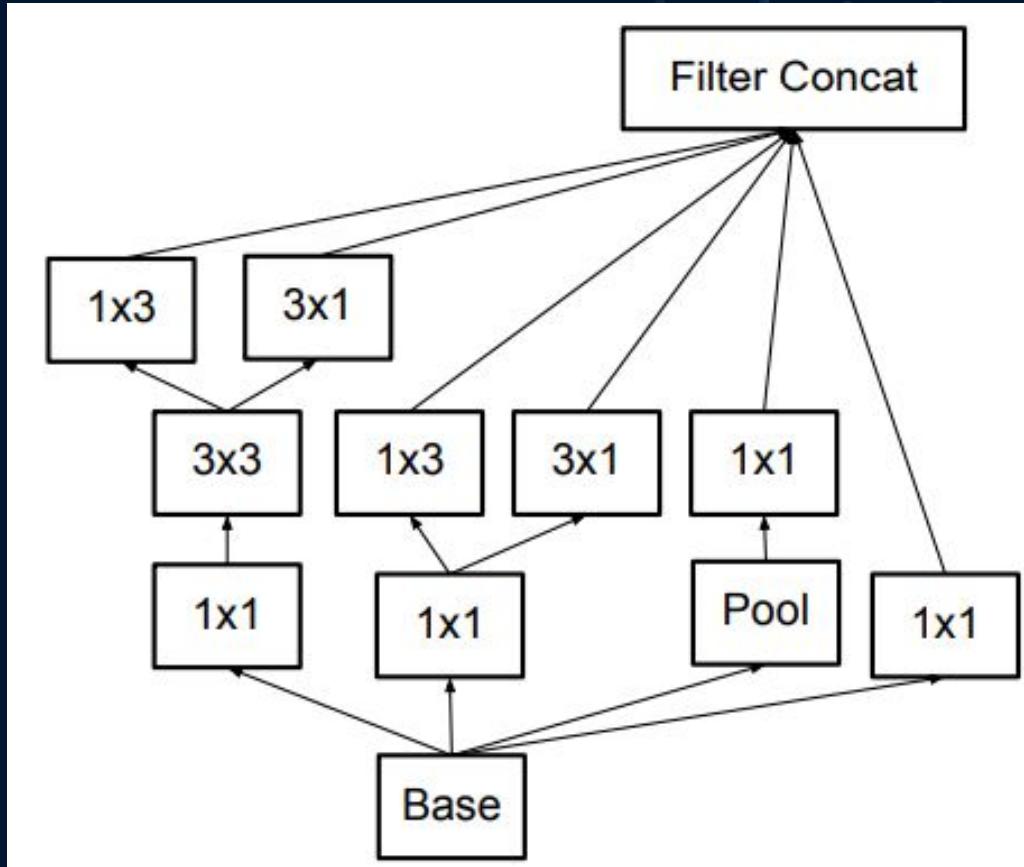
Szegedy, Christian, et al.
"Rethinking the inception
architecture for computer vision."
*Proceedings of the IEEE
conference on computer vision
and pattern recognition*. 2016.

Figure 6



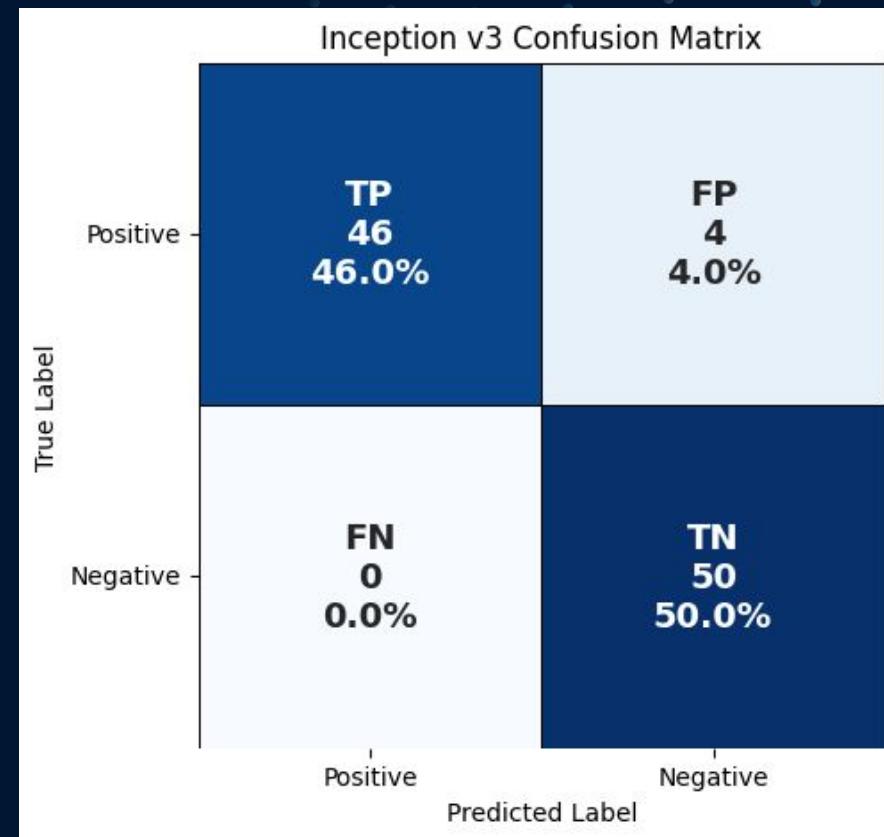
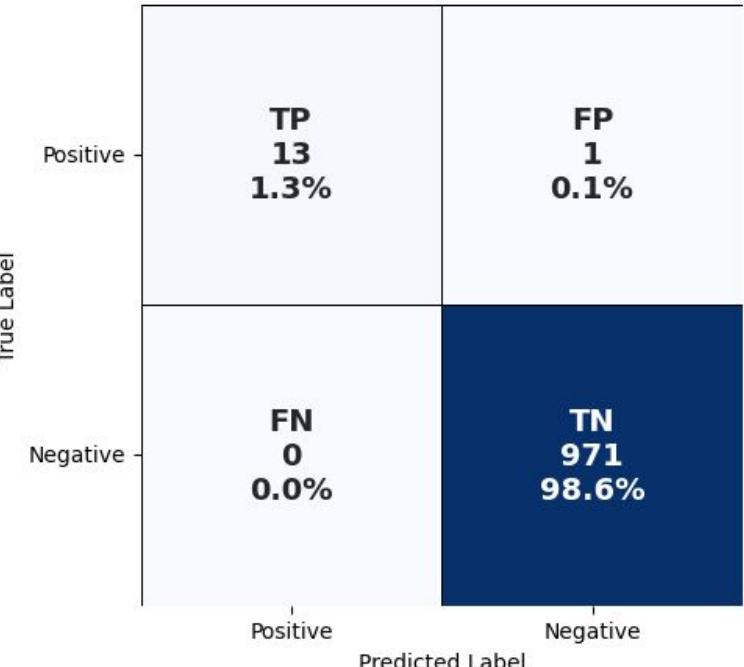
Szegedy, Christian, et al.
"Rethinking the inception
architecture for computer
vision." *Proceedings of the
IEEE conference on
computer vision and
pattern recognition*. 2016.

Figure 7

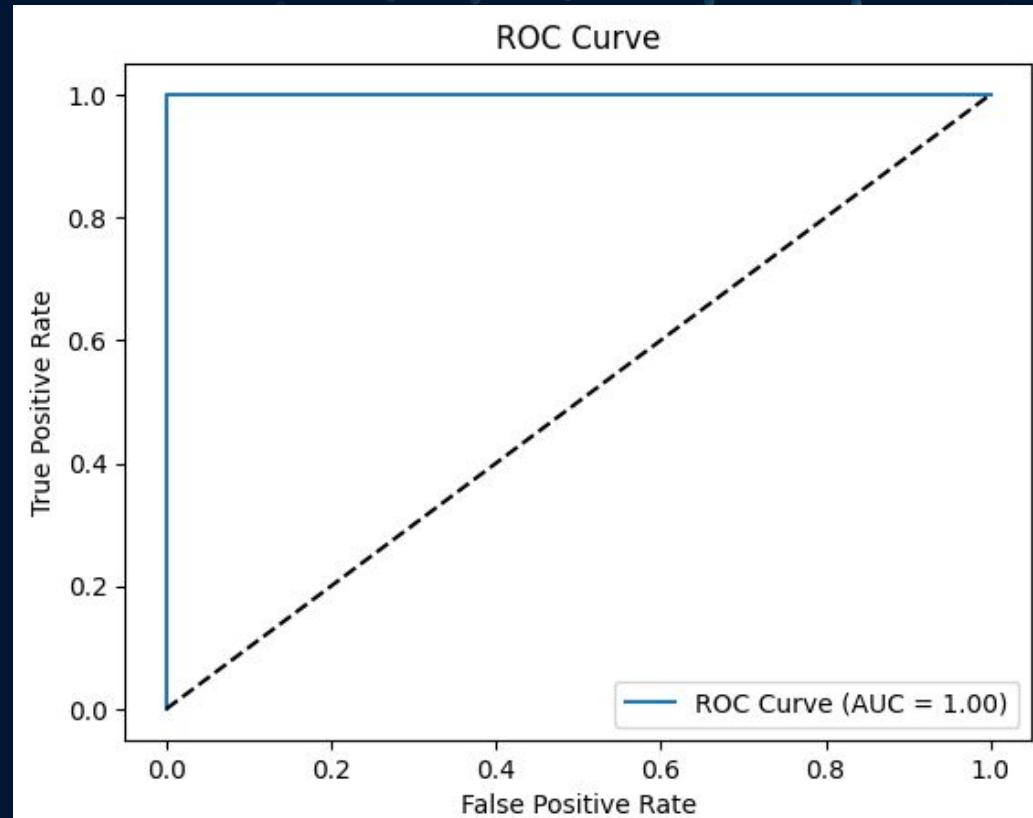
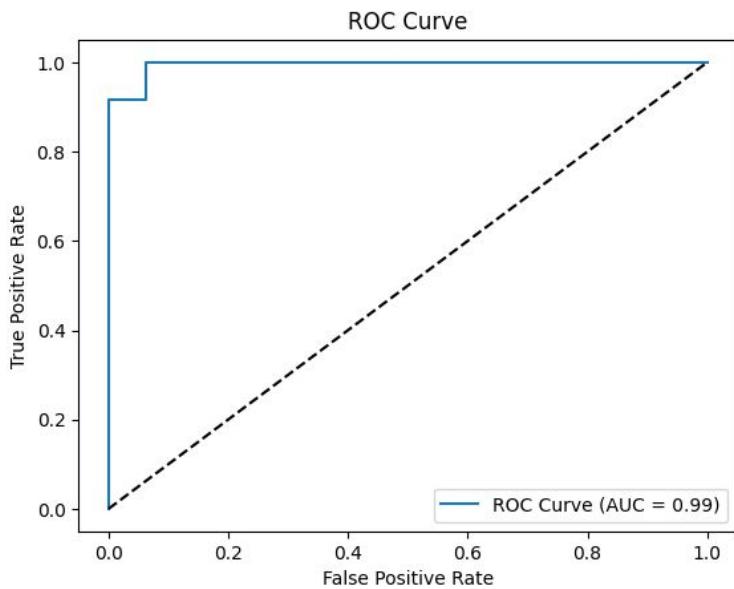


Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

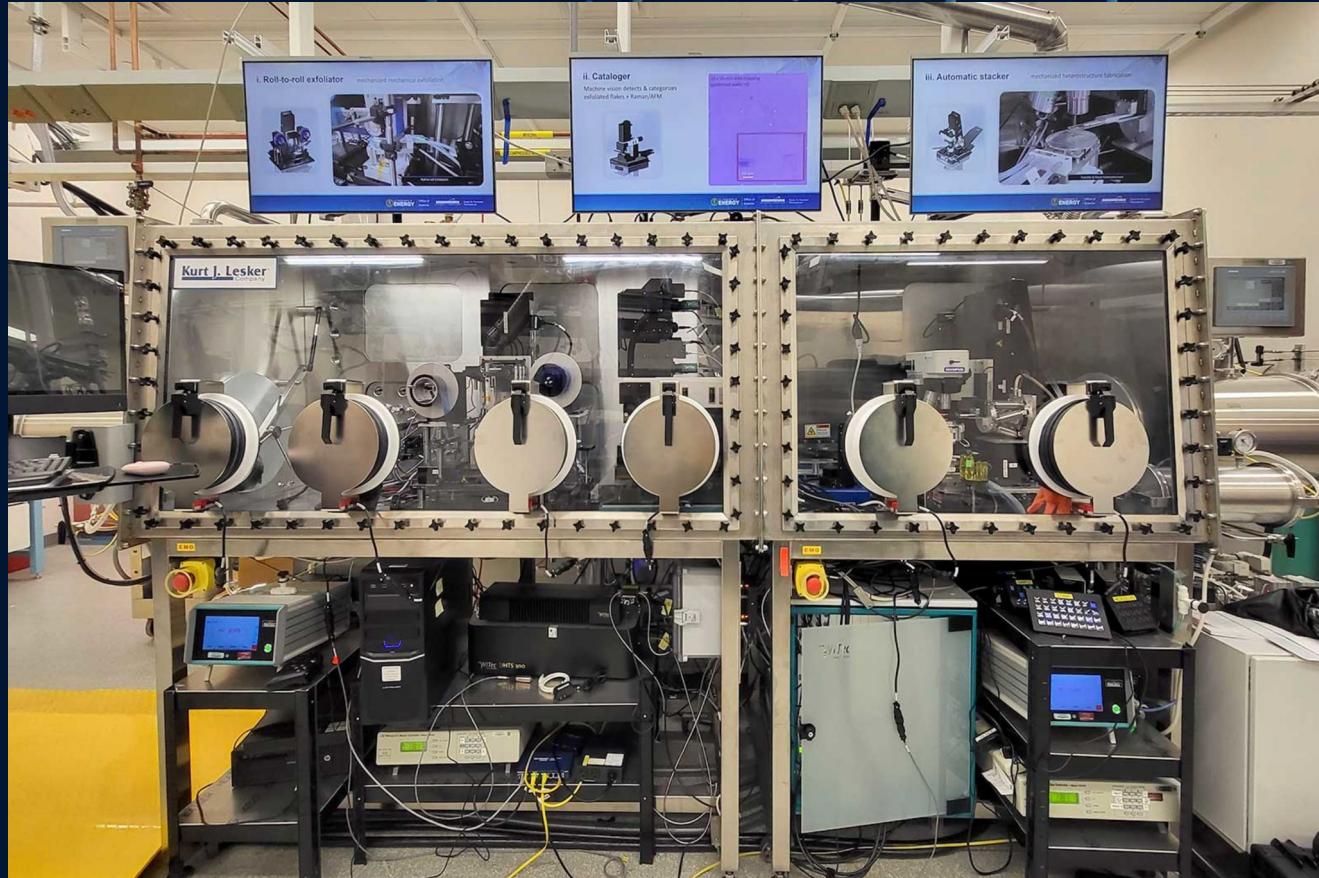
Inception v3 Confusion Matrix



Inception v3 ROC/ AUC Plot



Future Work



“BNL | QPress Cluster Tool.” Bnl.gov, 2024, www.bnl.gov/qpress/modules/cluster-tool.php. Accessed 8 Dec. 2024.

Our Futures

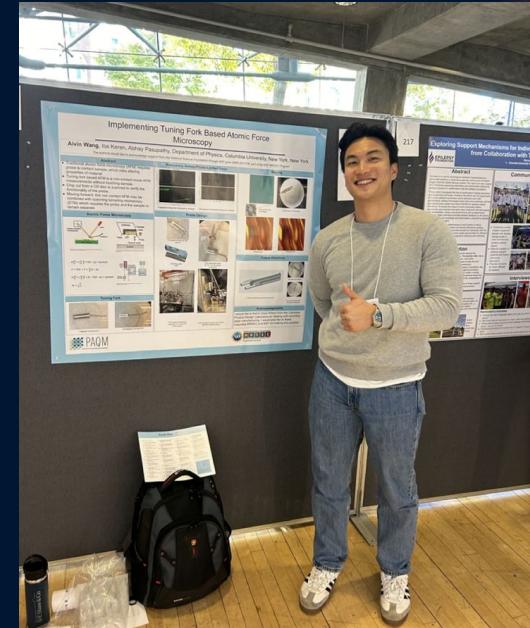
R&D

Boster, Seth. "Smokejumping, Other Outdoor Extremes Shape Life of Colorado Woman." *Colorado Springs Gazette*, 25 Apr. 2022, gazette.com/premium/smokejumping-other-outdoor-extremes-shape-life-of-colorado-woman/article_55b68c42-bc07-11ec-8b01-c3ddee3115f4.html.



COLUMBIA UNIVERSITY IN THE CITY OF NEW YORK

DEPARTMENT OF PHYSICS



References

1. *Condensed Matter Physics / Department of Physics*, www.physics.columbia.edu/content/condensed-matter-physics. Accessed 7 Dec. 2024.
2. Lloyd-Hughes, James & Jeon, Tae-In. (2012). A Review of the Terahertz Conductivity of Bulk and Nano-Materials. *J Infrared Milli Terahz Waves*. 33. 871. 10.1007/s10762-012-9905-y.
3. MacDonald, A., Bistritzer, R. Graphene moiré mystery solved?. *Nature* 474, 453–454 (2011). <https://doi.org/10.1038/474453a>
4. Maruf, H M & Islam, Md. Rafiqul & Chowdhury, F.-U.-Z. (2018). ANALOGY BETWEEN AC JOSEPHSON JUNCTION EFFECTS AND OPTICAL PHENOMENA IN SUPERCONDUCTORS. 105-113.
5. BEIJING IWHR CORPORATION. “Spillway Crest Gates: Controlling Water Flow in Dam Structures.” BEIJING IWHR CORPORATION, 2024, www.bic-iwhr.com/news/spillway-crest-gates-controlling-water-flow-in-dam-structures.html. Accessed 8 Dec. 2024.
6. Nanoscience Instruments. “Scanning Tunneling Microscopy - Nanoscience Instruments.” Nanoscience Instruments, 2018, www.nanoscience.com/techniques/scanning-tunneling-microscopy/.
7. Yamada, Yuya, et al. “Fundamental and Higher Eigenmodes of QPlus Sensors with a Long Probe for Vertical-Lateral Bimodal Atomic Force Microscopy.” *Nanoscale Advances*, vol. 5, no. 3, 26 Dec. 2022, pp. 840–850, pubs.rsc.org/en/content/articlehtml/2023/na/d2na00686c, <https://doi.org/10.1039/d2na00686c>. Accessed 8 Dec. 2024.
8. Geim, A. K., and I. V. Grigorieva. “Van Der Waals Heterostructures.” *Nature*, vol. 499, no. 7459, 1 July 2013, pp. 419–425, www.nature.com/articles/nature12385, <https://doi.org/10.1038/nature12385>.

References

9. Askari, Syed Sajjad. "Alex-Net Explanation and Implementation in Tensorflow and Keras." *Medium*, Medium, 14 Jan. 2023, medium.com/@syedsajjad62/alex-net-explanation-and-implementation-in-tensorflow-and-keras-8047efeb7a0f.
10. Sanderson, Grant. "But What is a Convolution?" *Youtube*, 3blue1brown, 29 Dec. 2023, <https://www.youtube.com/shorts/kpG7I2MOcnI>.
11. Ave Coders. "101 Concepts of Data Science and Machine Learning" *Youtube*, AveCoders, 28 May. 2024, <https://www.youtube.com/shorts/ykt5PUxs3z8>.
12. Chima, Precious. "Activation Functions: ReLU & Softmax." *Medium*, Medium, 5 Apr. 2020, <https://medium.com/@preshchima/activation-functions-relu-softmax-87145bf39288>.
13. Alexander, Giffah. "Softmax Activation function explained" *Youtube*, Giffah_Alexander, 22 Oct. 2024, https://www.youtube.com/shorts/SrJN_hpiuAs.
14. Sigmoid function. "Sigmoid function." *Wikipedia*, Wikipedia, 14 Nov. 2024, https://en.wikipedia.org/wiki/Sigmoid_function.
15. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (June 2017), 84–90. <https://doi.org/10.1145/3065386>
16. Ioffe, Sergey. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).

References

17. Kingma, Diederik P. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).
18. Gabbard & Miller. "Machine Learning from Scratch: Stochastic Gradient Descent and Adam Optimizer." *18.0851 Project*, accessed 17 November 2024, https://www.mit.edu/~jgabbard/assets/18085_Project_final.pdf.
19. Sahilcarterr. "Mathematics Behind Simple Linear Regression using Gradient Descent." *Medium*, Medium, 5 Apr. 2020, <https://medium.com/@sahilcarterr/mathematics-behind-simple-linear-regression-using-gradient-descent-a09595bf701d>.
20. Bangar, Siddhesh. "VGG-Net Architecture Explained." *Medium*, Medium, 28 Jun. 2022, <https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f>.
21. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).
22. He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
23. Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.
24. Hien, Ngo Le Huy. "Grid Search space Illustration" *ResearchGate*, ResearchGate, https://www.researchgate.net/figure/Grid-Search-space-illustration_fiq9_346571789.
25. Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

References

26. Neural Computing and Applications. "Graph for ReLU6 activation function" *ResearchGate*, ResearchGate, https://www.researchgate.net/figure/Grid-Search-space-illustration_fig9_346571789.
27. Derevianko, Ivan. "BLAZOR - IT'S TIME TO FORGET JAVASCRIPT" *Ivan Derevianko*, Ivan Derevianko, 16 July 2019, <https://ivanderevianko.com/2019/07/blazor-its-time-to-forget-javascript>.
28. Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
29. Pytorch. "Vision/Torchvision/Models/Alexnet.Py at Main · Pytorch/Vision." *GitHub*, github.com/pytorch/vision/blob/main/torchvision/models/alexnet.py. Accessed 20 Nov. 2024.
30. Pytorch. "Vision/Torchvision/Models/ResNet.Py at Main · Pytorch/Vision." *GitHub*, <https://github.com/pytorch/vision/blob/main/torchvision/models/resnet.py>. Accessed 20 Nov. 2024.
31. Pytorch. "Vision/Torchvision/Models/VGG.Py at Main · Pytorch/Vision." *GitHub*, <https://github.com/pytorch/vision/blob/main/torchvision/models/vgg.py>. Accessed 20 Nov. 2024.
32. Nvidia. "DeepLearningExamples/Pytorch/Classification/Convnets/Efficientnet at Master · Nvidia/Deeplearningexamples." *GitHub*, github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/Classification/ConvNets/efficientnet. Accessed 20 Nov. 2024.
33. Pytorch. "Vision/Torchvision/Models/Inception.Py at Main · Pytorch/Vision." *GitHub*, <https://github.com/pytorch/vision/blob/main/torchvision/models/inception.py>. Accessed 20 Nov. 2024.
34. "BNL | QPress Cluster Tool." Bnl.gov, 2024, www.bnl.gov/qpress/modules/cluster-tool.php. Accessed 8 Dec. 2024.

References

35. Boster, Seth. "Smokejumping, Other Outdoor Extremes Shape Life of Colorado Woman." *Colorado Springs Gazette*, 25 Apr. 2022, gazette.com/premium/smokejumping-other-outdoor-extremes-shape-life-of-colorado-woman/article_55b68c42-bc07-11ec-8b01-c3ddee3115f4.html.
36. jodag. "How Is Pytorch's Class BCEWITHLOGITSLOSS Exactly Implemented?" *Stack Overflow*, 11 July 2022, stackoverflow.com/questions/66906884/how-is-pytorchs-class-bcewithlogitsloss-exactly-implemented.