Deep Residual Flow for Out of Distribution Detection

Ev Zisselman and Aviv Tamar

Block-Seminar on Deep Learning for Bio-Medical Data Analysis

Advisor: Özgün Çiçek

What is the problem?

What is the problem?

From: https://www.youtube.com/watch?v=bNW1a1RYGWk

- Unexpected behavior for out-of-distribution images!

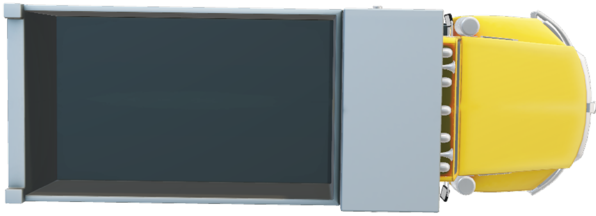
What is the problem?

From: https://www.youtube.com/watch?v=LfmAG4dk-rU&feature=emb_title

- Real world applications are hindered by unexpected behavior!

How do we solve this?

- confidence measure for certainty of the prediction

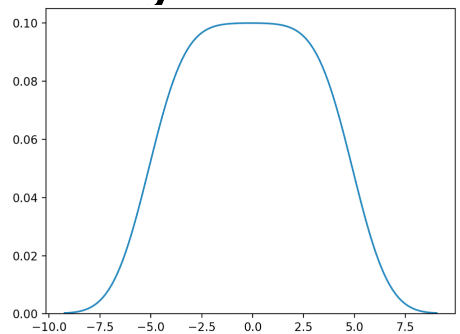


Model

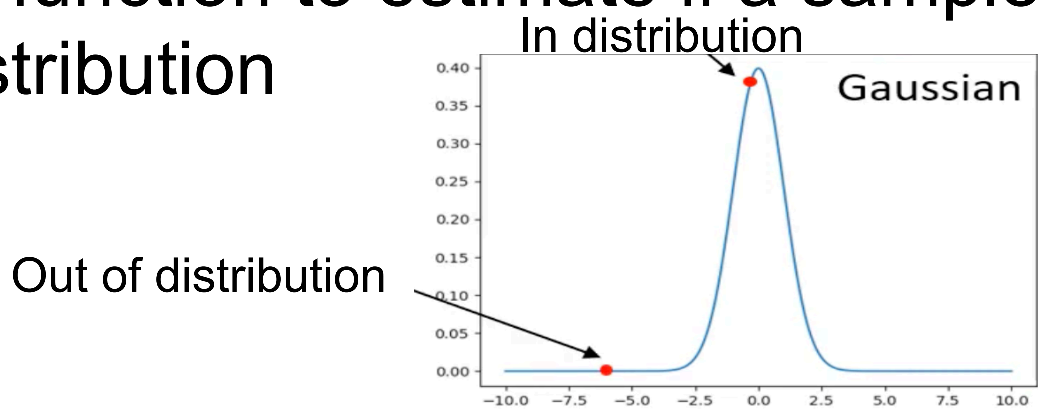
Prediction: free road

Confidence: 2%

- model a probability density function given samples form that distribution



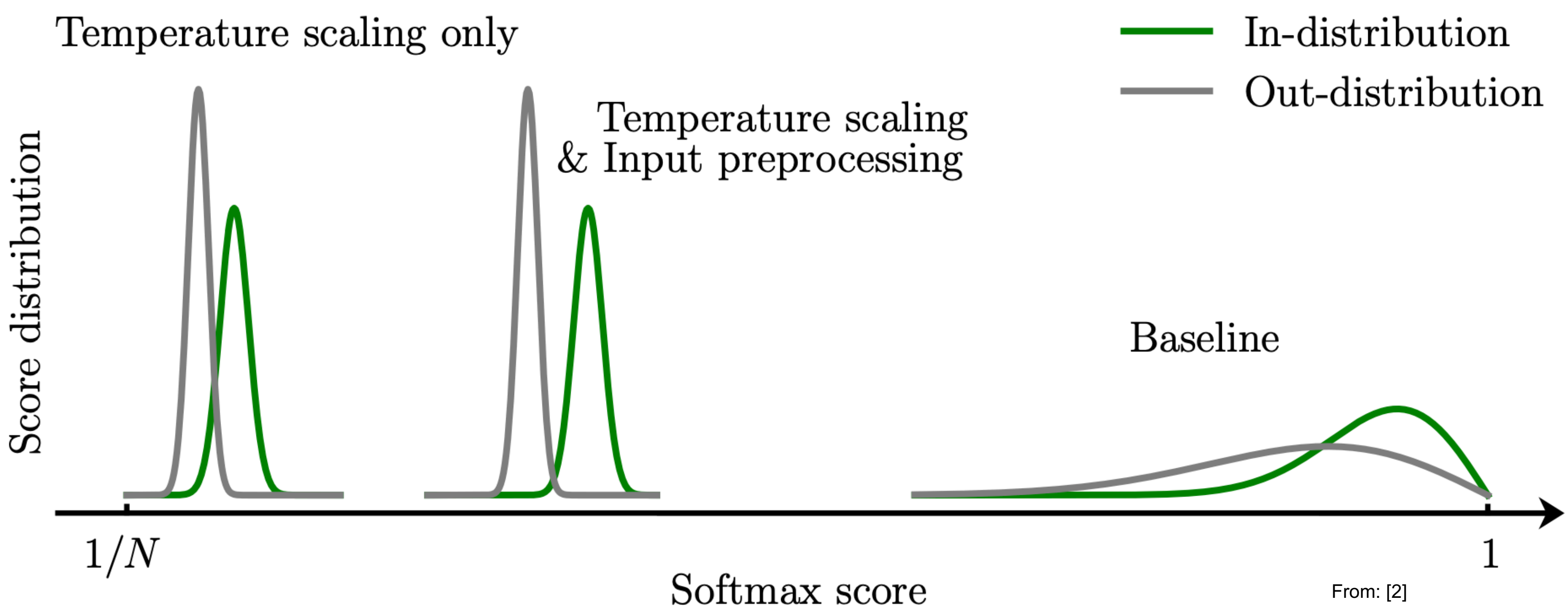
- Use the density function to estimate if a sample is part of the in-distribution



From: https://www.youtube.com/watch?v=bNW1a1RYGWk

Related methods

- Classical methods like: One-Class SVMs
- **Setting:** Trained classification net, labeled data
- Baseline line method by [Hendrycks and Gimpel]
  - Uses soft-max score as the confidence score
- ODIN by [Liang et al.]

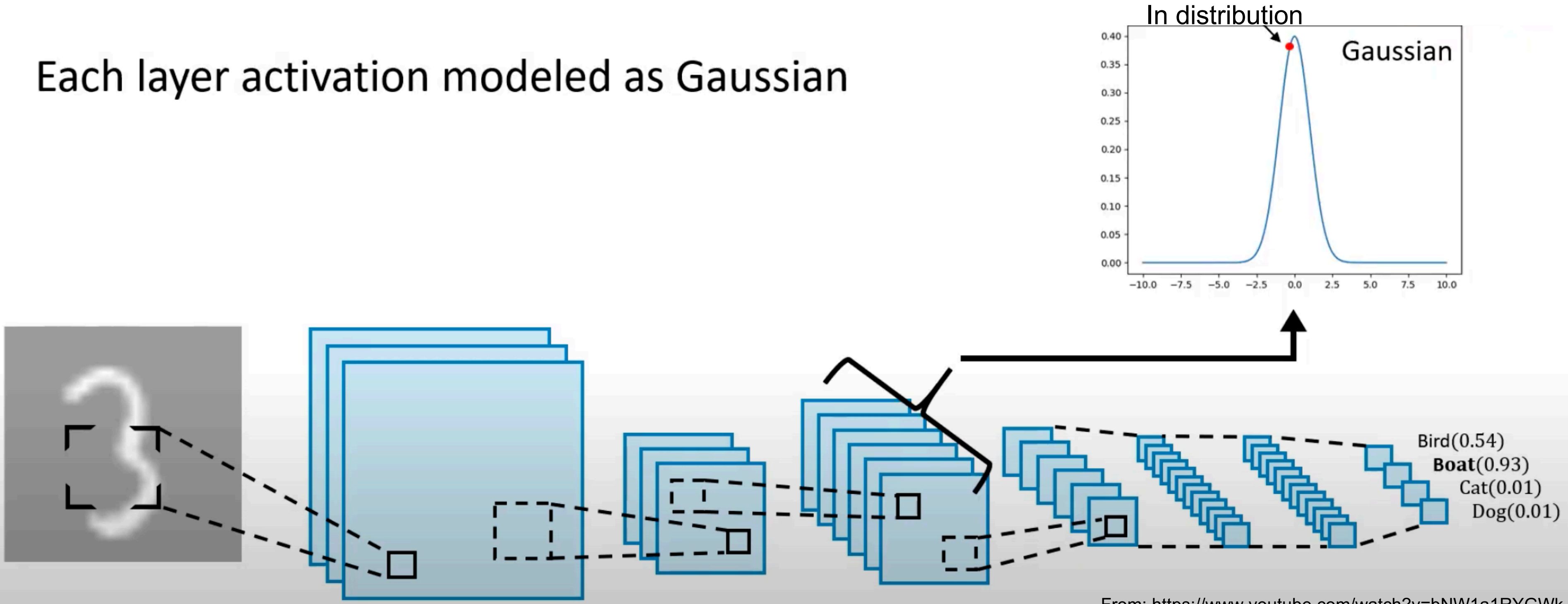


From: [2]

Previous STOA

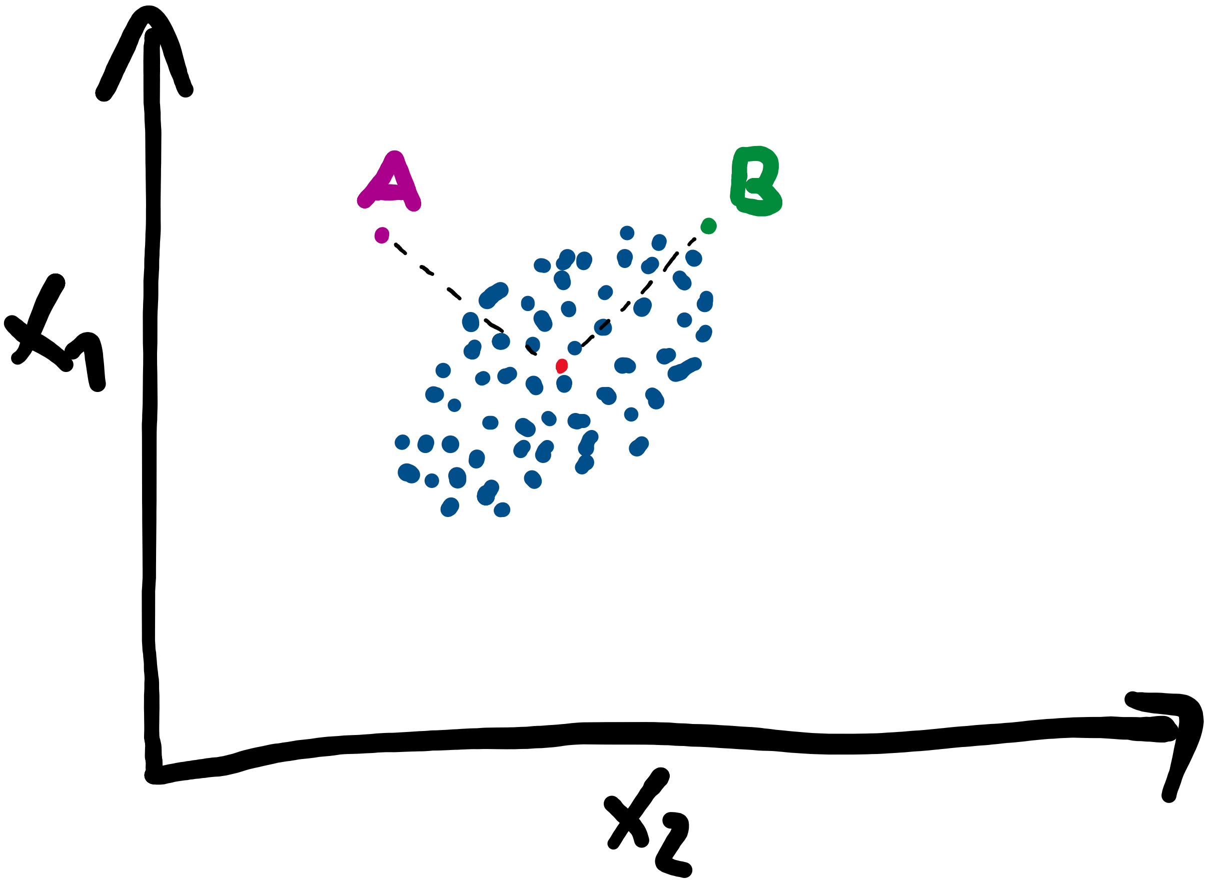
Previous State-of the-Art (Mahalanobis) by: Lee et al.

Main idea:
Model each layer activations as a Gaussian distribution and use the Mahalanobis distance as a confidence score.

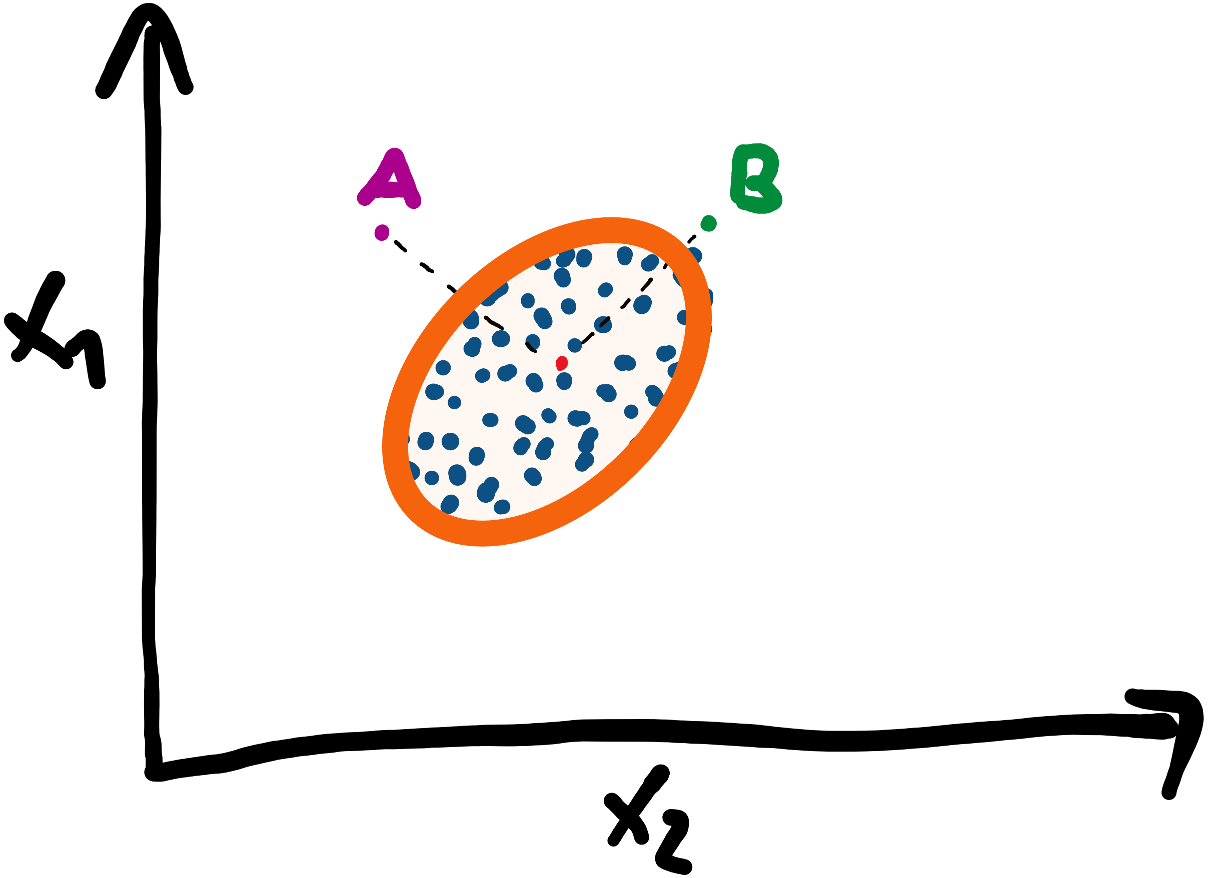


From: https://www.youtube.com/watch?v=bNW1a1RYGWk

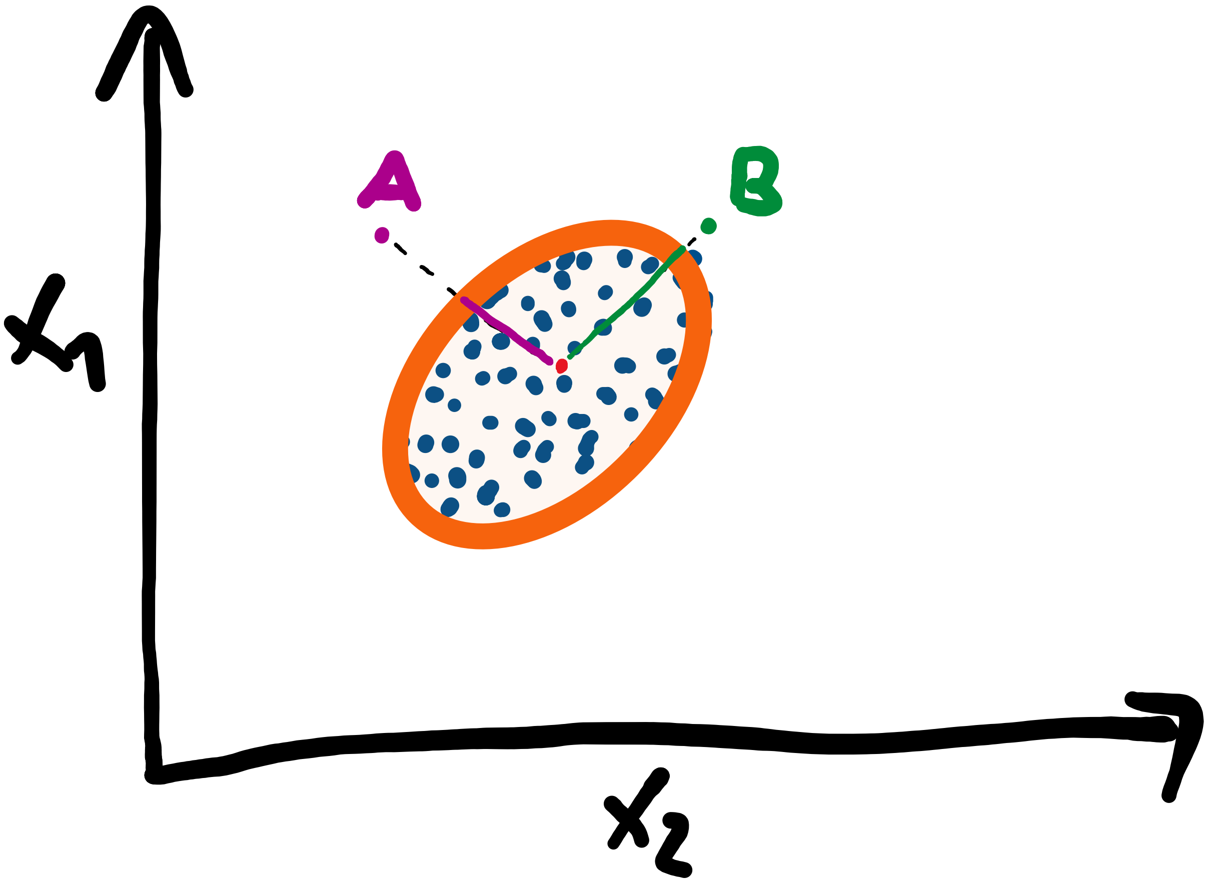
What is the Mahalanobis distance?



What is the Mahalanobis distance?



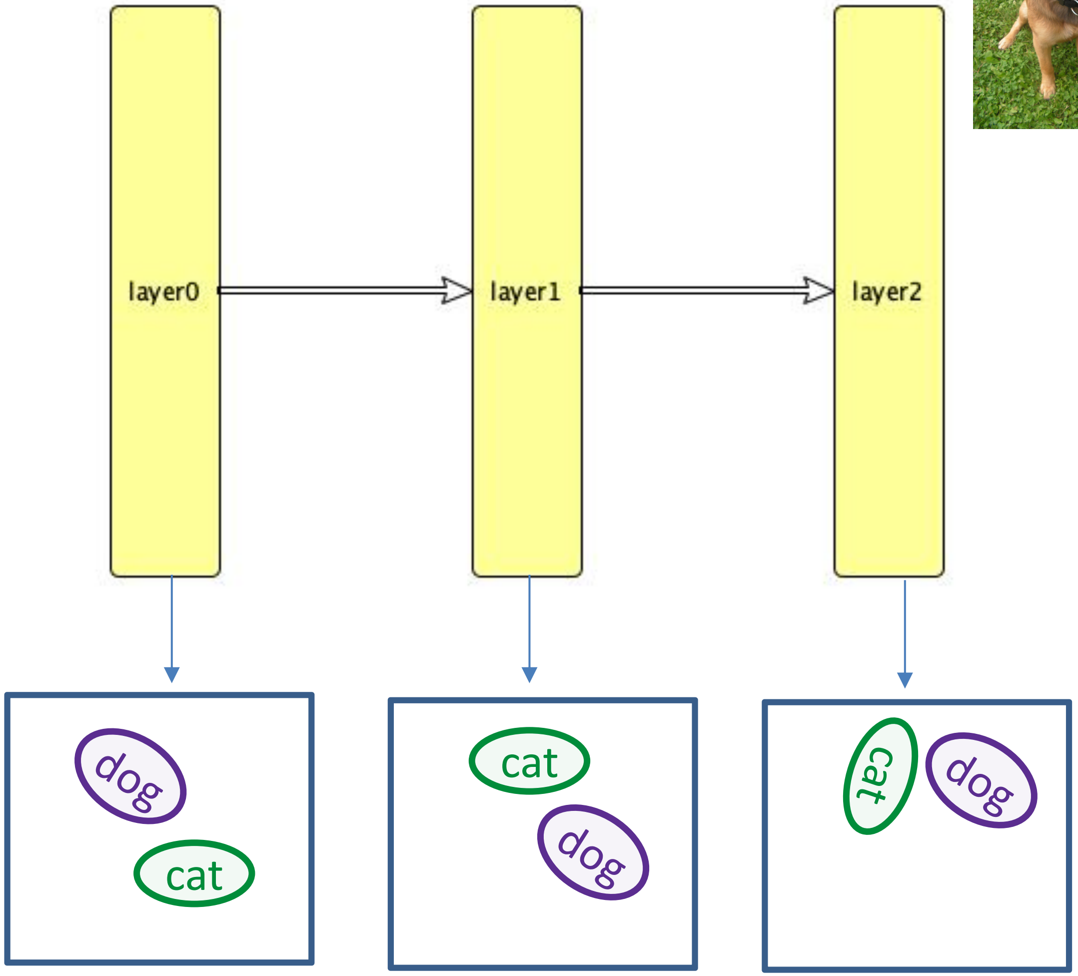
What is the Mahalanobis distance?



Measures of how far away a Point is from a distribution

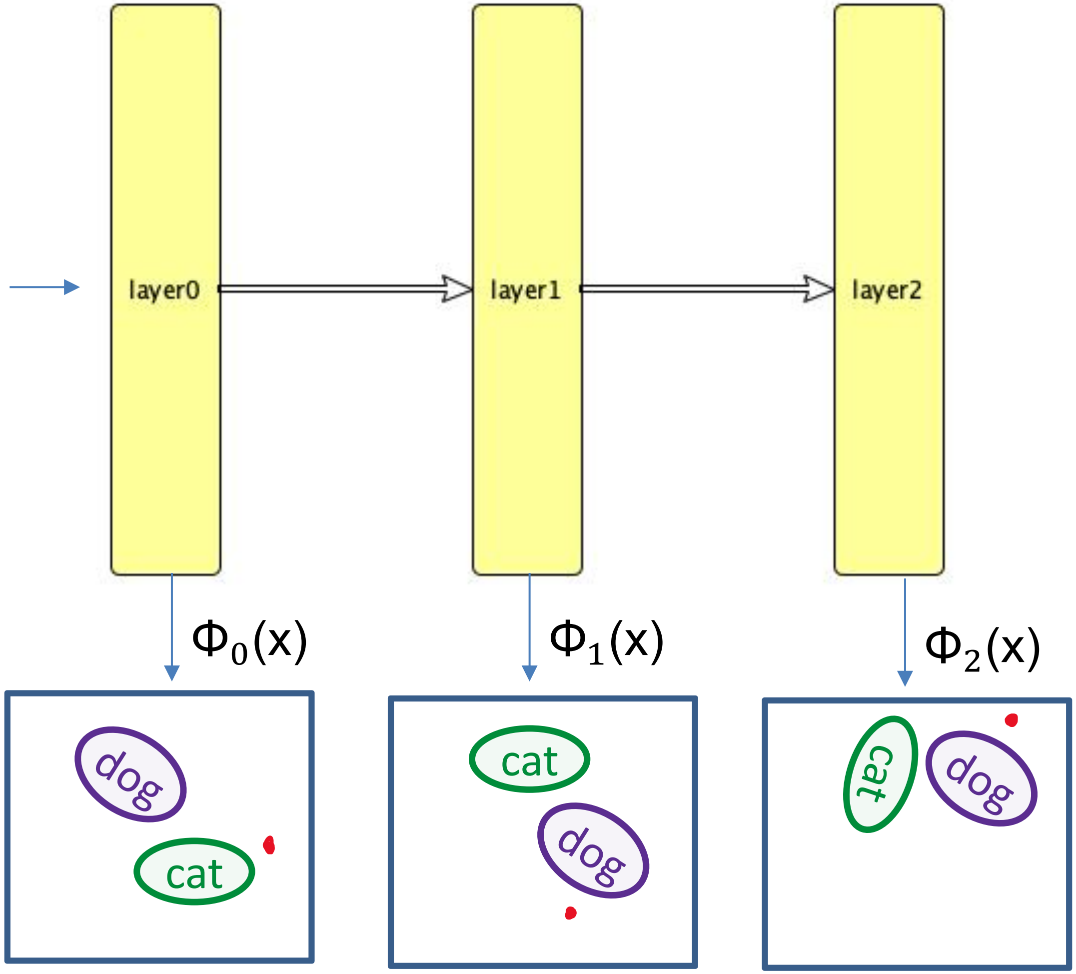
Example: Mahalanobis based confidence score

Given: Trained Classification network for the classes  and 

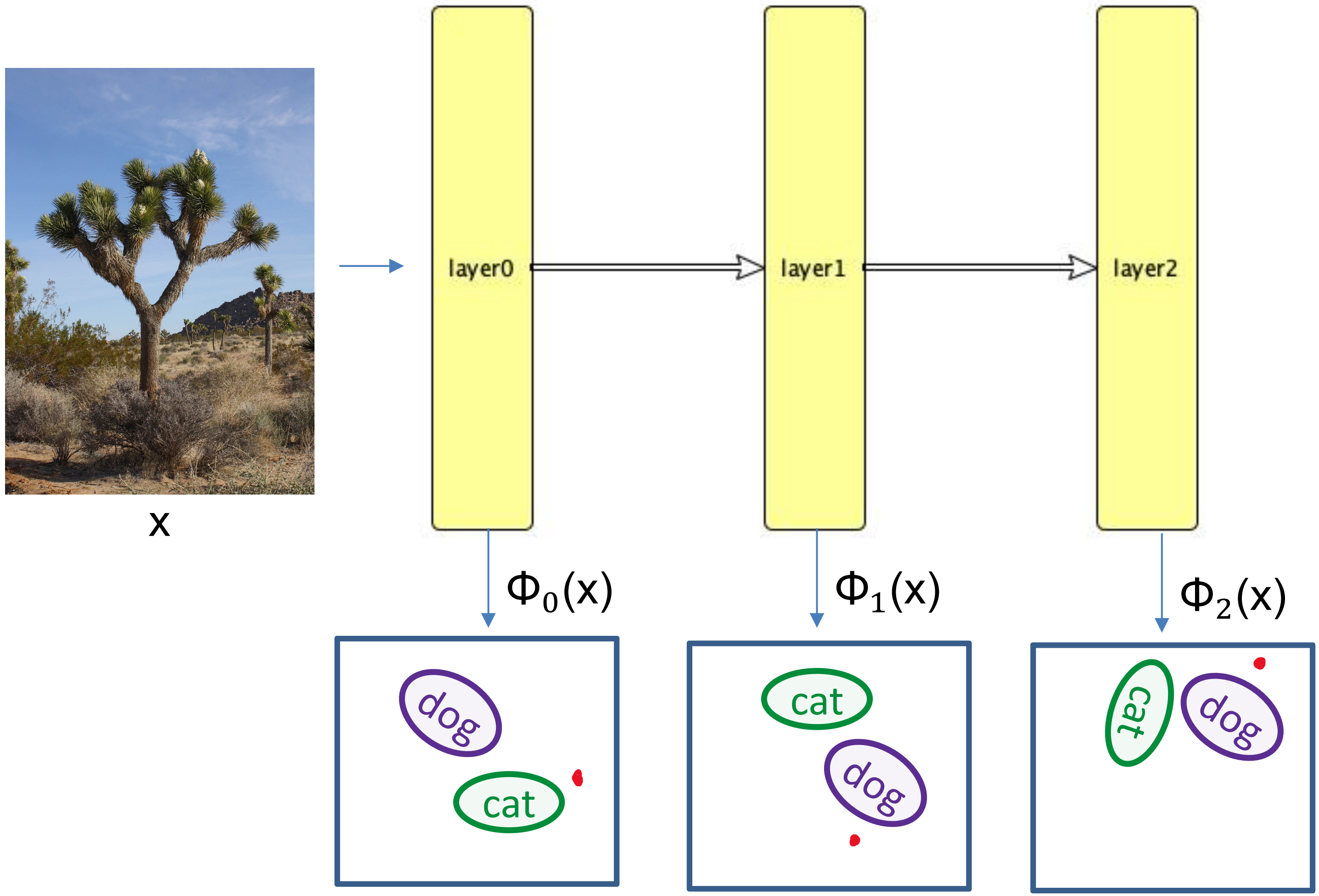


Example: Mahalanobis based confidence score

x



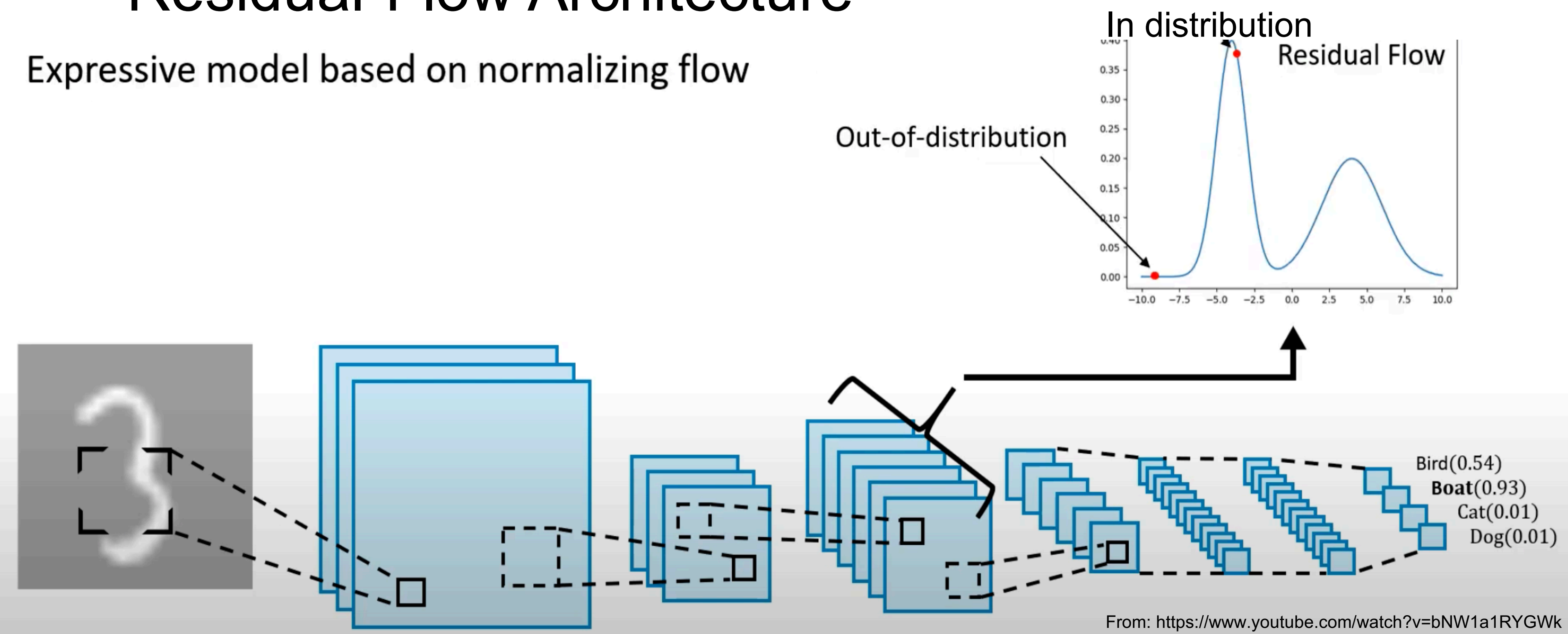
Example: Mahalanobis based confidence score



Confidence score = $\alpha_0\, MD(\bullet, cat) + \alpha_1\, MD(\bullet, dog) + \alpha_2\, MD(\bullet, dog)$

What is the difference to previous work?

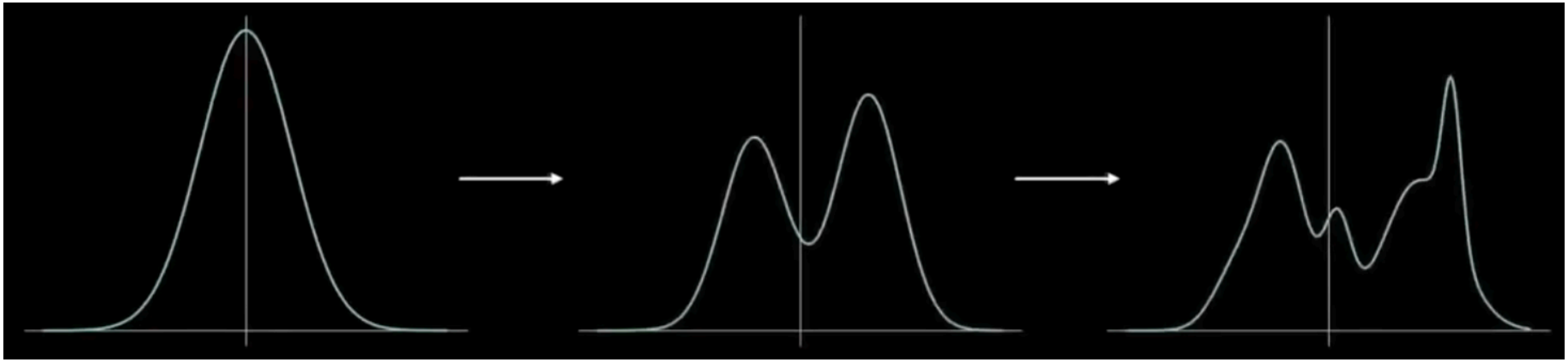
- Uses a more expressive density based on a new Residual-Flow Architecture



From: https://www.youtube.com/watch?v=bNW1a1RYGWk

- Uses the Residual-Flow-Score instead of the Mahalanobis distance as confidence measure

From: https://www.youtube.com/watch?v=i7LjDvsLWCg



$$z \sim p_Z(z) = N(0,1)$$

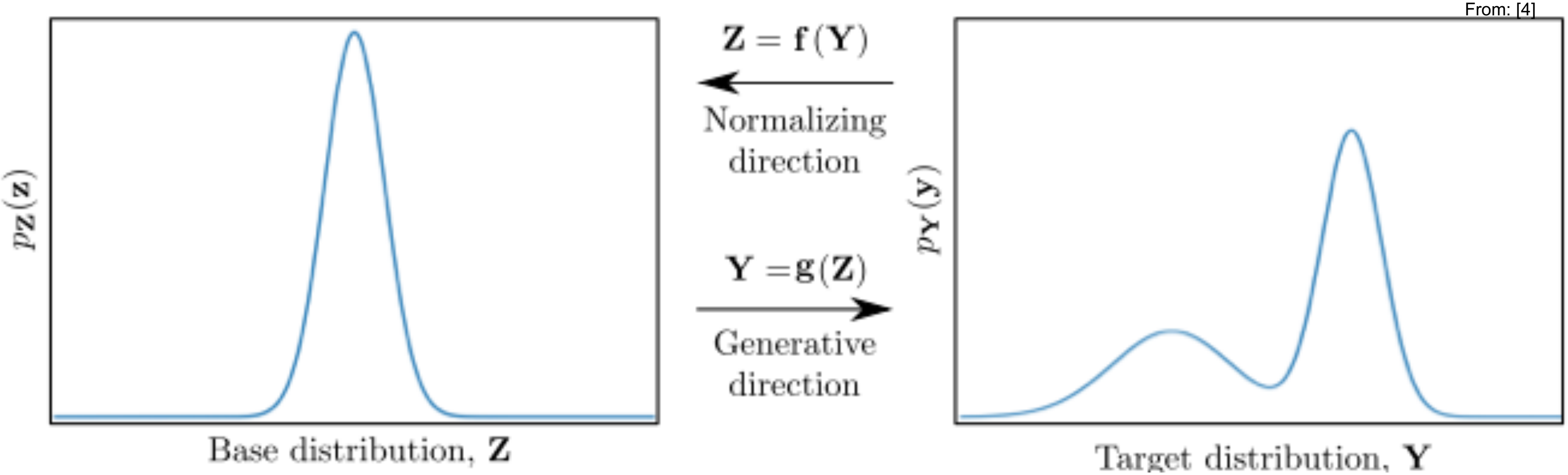
$$x = g(z) = g_k \circ \ldots \circ g_2 \circ g_1(z)$$

each g_i is invertible (bijective)

$$g^{-1} = f$$

Normalizing Flows

From: [4]



What is the density of a given y?

Change of variables formula:

$$p_Y(y) = p_Z(f(y)) * \left|\det \frac{\delta f(y)}{\delta y}\right|$$

Linear Flow

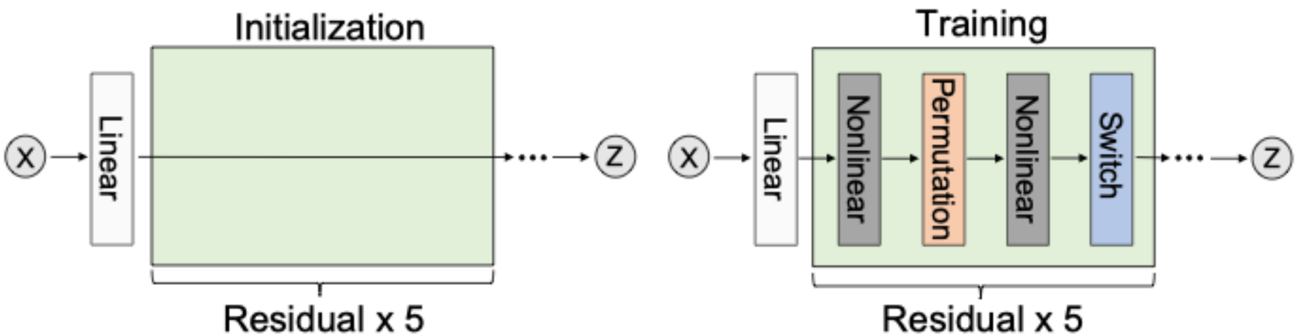
- $g(x) = Ax + b$, where A is invertable

- Training a linear flow model on the feature space of a neural network is equal to fitting a Gaussian distribution via LDA

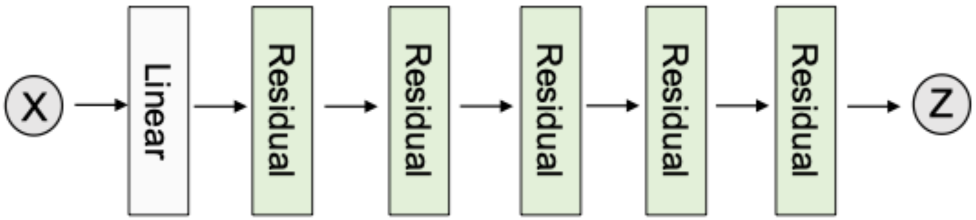
- Linear flow transfomation can be obtained analytically

Residual Flow

- Residual Flow = linear flow + non linear residual component
- Non-linear block component: is a DNN
- Permutations are used to diversifying the inputs of the non-linear components



(a) Residual Flow blocks during initialization and training.



(b) The complete Residual Flow architecture $Z = f(X)$.

From [1]

How does it work?

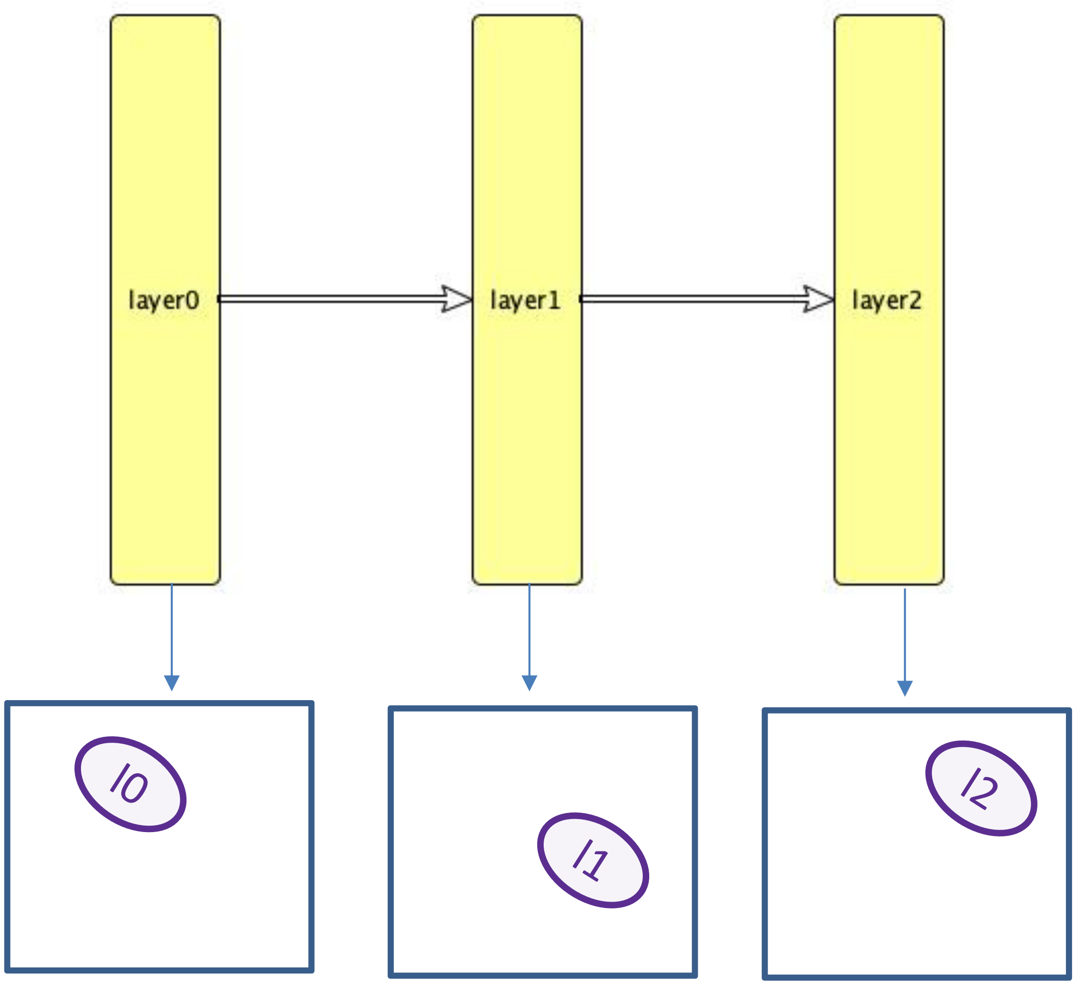
- Given: trained image classifier and labelled pictures of dogs and cats:

1. for each sample x in our training data, we extract the network activation in layer l: $\Phi_l(x)$
2. for each network layer l, we extract the mean activation in the training data for each class label: $\mu_{l,c}$
3. calculate centred feature training set:

$$\hat{\Phi}_l(x) = \Phi_l(x) - \mu_{l,c}$$

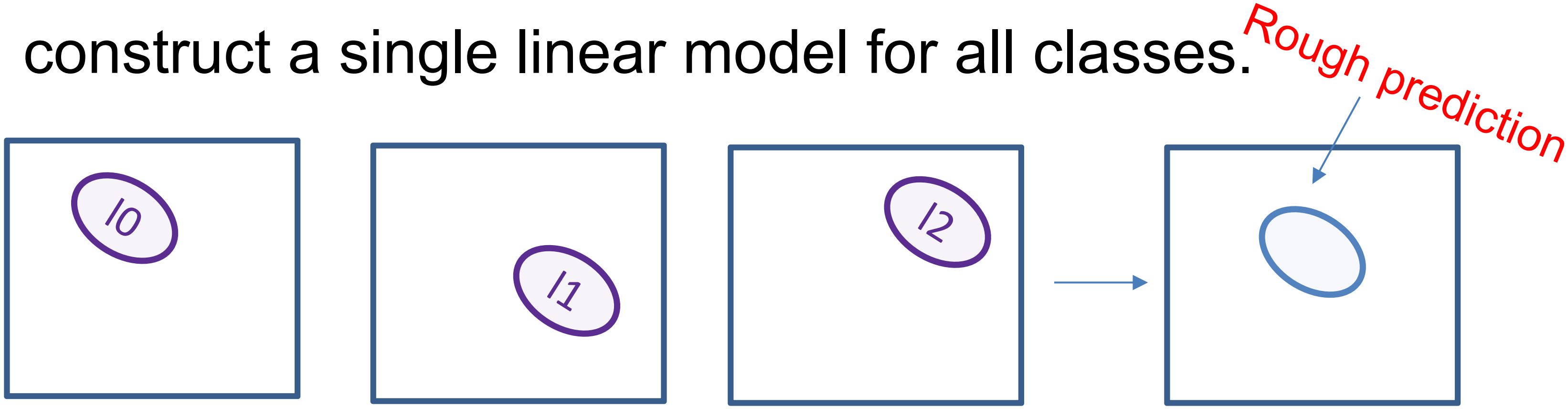
How does it work?

4. fit a Gaussian distribution to the centered dataset by constructing a linear flow model for each layer

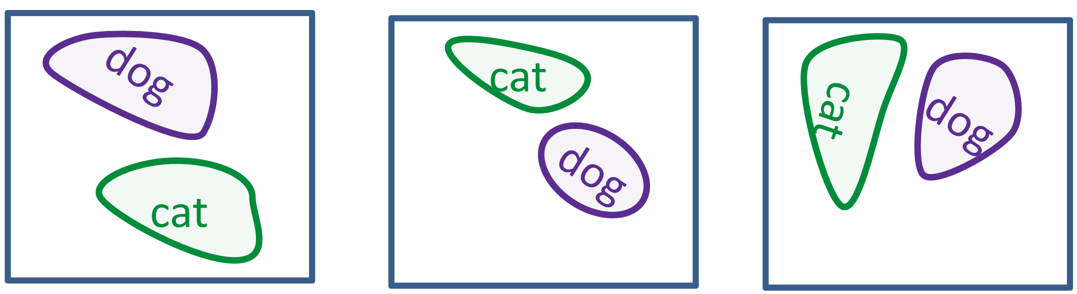


How does it work?

5. construct a single linear model for all classes.



6. for each layer l, and for each class c, we train a residual flow model by training the non-linear flow blocks.

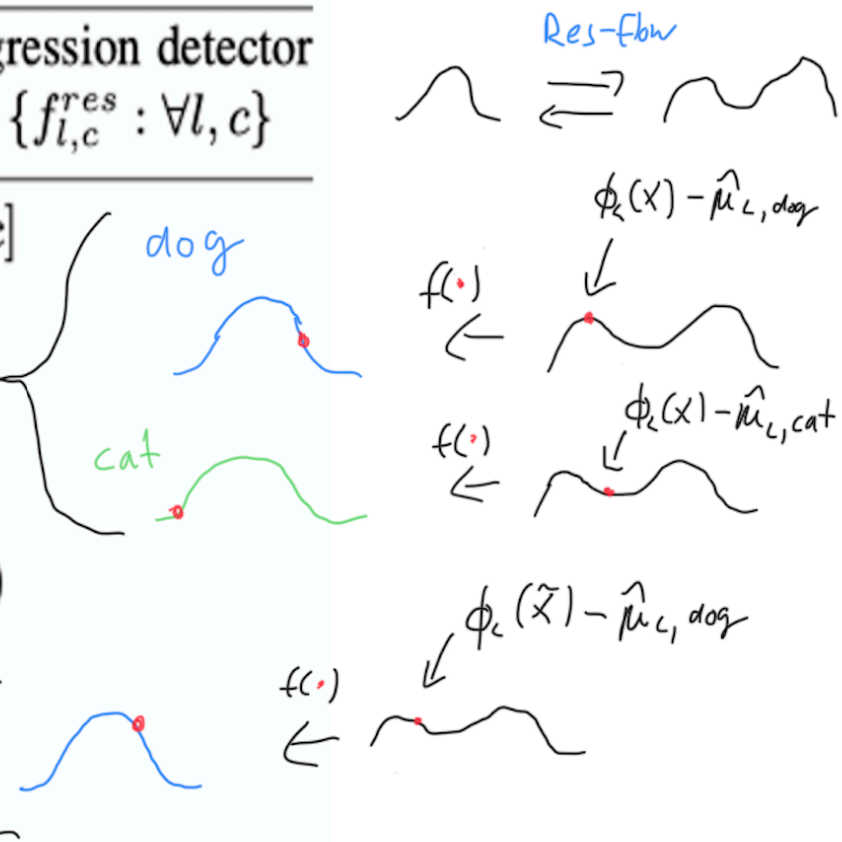


layer0 layer1 layer2

Algorithm 1 Computing the Residual-Flow score S_l.

Input: Test sample x, weights of logistic regression detector α_l, noise ε and C residual-flow for each layer: $\{f^{res}_{l,c} : \forall l, c\}$

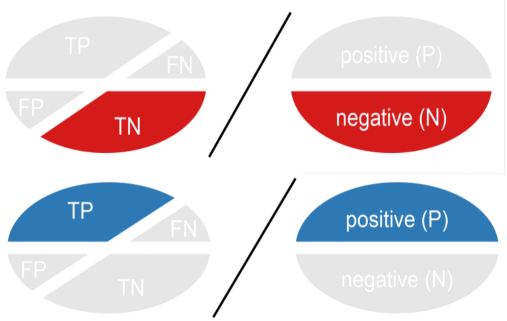
Initialize score vectors: $\mathbf{S}_{RF}(x) = [S_{l,c} : \forall l, c]$
for each layer $l \in 1, \dots, L$ **do**
  Find the most probable class:
    $\widehat{c} = \arg\max_c \; p_c(\phi_l(x) - \widehat{\mu}_{l,c})$
  Add small noise to test sample:
    $\tilde{x} = x + \varepsilon \text{sign} \, \nabla_x \, p_{\widehat{c}} \, (\phi_l(x) - \widehat{\mu}_{l,\widehat{c}})$
  Computing confidence score:
    $S_l = \max_c p_c \, (\phi_l(\tilde{x}) - \widehat{\mu}_{l,c})$
end for
return Confidence score for test sample $\sum_l \alpha_l S_l$



From [1]

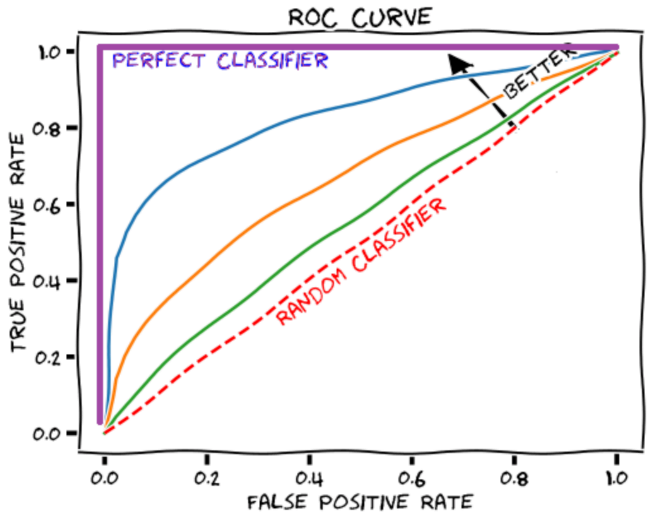
Evaluation

– Same evaluation technique as Lee et al.
  - Same datasets and architectures, …
– Most important performance measures:
  - true negative rate at 95% true positive rate



From: https://moredvikas.wordpress.com/2017/09/12/what-is-true-positive-and-true-negative-confusion-matrix/

  - area under the receiver operating characteristic curve



From: https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/

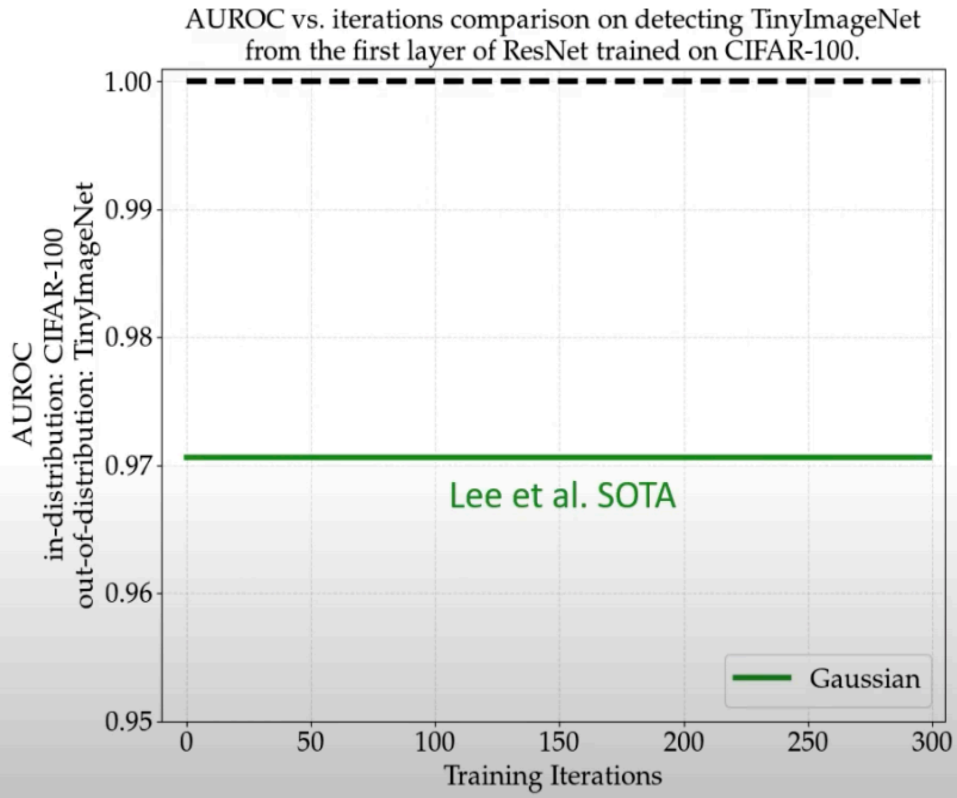
Results

How does the OOD detection method compare with state-of-the-art?
They report an principled improvement on the current STOA

In-dist (model)	Out-of-dist	TNR at TPR 95%	AUROC	Detection accuracy	AUPR in	AUPR out
		Mahalanobis [27]/ Res-Flow without pre-processing / Res-Flow with pre-processing				
CIFAR-10 (DenseNet)	SVHN	85.8 / **94.9 / 94.9**	96.6 / **98.9 / 98.9**	91.9 / **95.3 / 95.3**	98.7 / **99.5 / 99.5**	88.8 / **97.5 / 97.5**
	ImageNet	95.3 / **96.4 / 96.4**	98.9 / **99.2 / 99.2**	95.2 / **96.0 / 96.0**	98.9 / **99.2 / 99.2**	98.7 / **99.2 / 99.2**
	LSUN	97.9 / **98.2 / 98.2**	99.3 / **99.5 / 99.5**	96.8 / **97.1 / 97.1**	99.3 / **99.6 / 99.6**	98.2 / **99.5 / 99.5**
CIFAR-100 (DenseNet)	SVHN	82.9 / 73.0 / **84.9**	96.1 / 95.2 / **97.5**	90.9 / 88.7 / **91.9**	98.5 / 97.5 / **99.0**	89.0 / 91.1 / **95.1**
	TinyImageNet	85.8 / **93.0/ 93.0**	96.6 / **98.5 / 98.5**	91.2 / **94.1 / 94.1**	96.9 / **98.5 / 98.5**	95.5 / **98.5 / 98.5**
	LSUN	83.6 / **96.3 / 96.3**	94.9 / **98.9 / 98.9**	89.9 / **95.7 / 95.7**	95.7 / **99.0 / 99.0**	93.0/ **98.8 / 98.8**
SVHN (DenseNet)	CIFAR-10	96.5 / **99.0 / 99.0**	98.9 / **99.5 / 99.5**	95.9 / **97.4 / 97.4**	95.6 / **97.8 / 97.8**	99.6 / **99.8 / 99.8**
	TinyImageNet	99.8 / **100.0 / 100.0**	99.9 / **100.0 / 100.0**	98.8 / **99.4 / 99.4**	99.6 / **99.8 / 99.8**	**100.0 / 100.0 / 100.0**
	LSUN	**100.0/ 100.00 / 100.00**	99.9 / **100.0 / 100.0**	99.3 / **99.7 / 99.7**	99.7 / **99.9 / 99.9**	**100.0 / 100.0 / 100.0**
CIFAR-10 (ResNet)	SVHN	96.4 / 94.5 / **96.5**	**99.1** / 98.9 / **99.1**	**95.8** / 94.9 / **95.8**	**99.6 / 99.6 / 99.6**	**98.3** / 97.6 / **98.3**
	TinyImageNet	97.1 / **97.8 / 97.8**	99.5 / **99.6 / 99.6**	96.3 / **96.9 / 96.9**	99.5 / **99.6 / 99.6**	99.5 / **99.6 / 99.6**
	LSUN	98.9 / **99.0 / 99.0**	99.7 / **99.8 / 99.8**	97.7 / **97.8 / 97.8**	99.7 / **99.8 / 99.8**	99.7 / **99.8 / 99.8**
CIFAR-100 (ResNet)	SVHN	92.0 / 88.8 / **93.0**	98.4 / 97.8 / **98.5**	93.7 / 92.6 / **94.5**	99.3 / 99.1 / **99.3**	96.4 / 95.3 / **97.1**
	TinyImageNet	90.8 / **95.0** / 94.6	98.2 / **98.9 / 98.9**	93.3 / **95.0 / 95.0**	98.1 / **98.9 / 98.9**	98.2 / **98.9** / 98.8
	LSUN	90.9 /**96.7** / 96.2	98.2 / **99.1** / 99.0	93.5 / 96.0 / **95.7**	97.8 / **99.0** / 98.9	98.4 / **98.8** / 98.6
SVHN (ResNet)	CIFAR-10	98.5 / 99.3 / **99.4**	99.3 / **99.6 /99.6**	96.9 / **97.7 / 97.7**	97.0 / **98.3 / 98.3**	99.7 / **99.9 / 99.9**
	TinyImageNet	99.9 / **100.0 / 100.0**	99.9 / **100.0** / 99.9	99.1 / **99.5 / 99.3**	99.1 / **99.8** / 99.7	99.9 / **100.0 / 100.0**
	LSUN	99.9 / **100.0 / 100.0**	99.9 / **100.0 / 100.0**	99.5 / **99.7 / 99.7**	99.2 / **99.8 / 99.8**	99.9 / **100.0 / 100.0**

From [1]

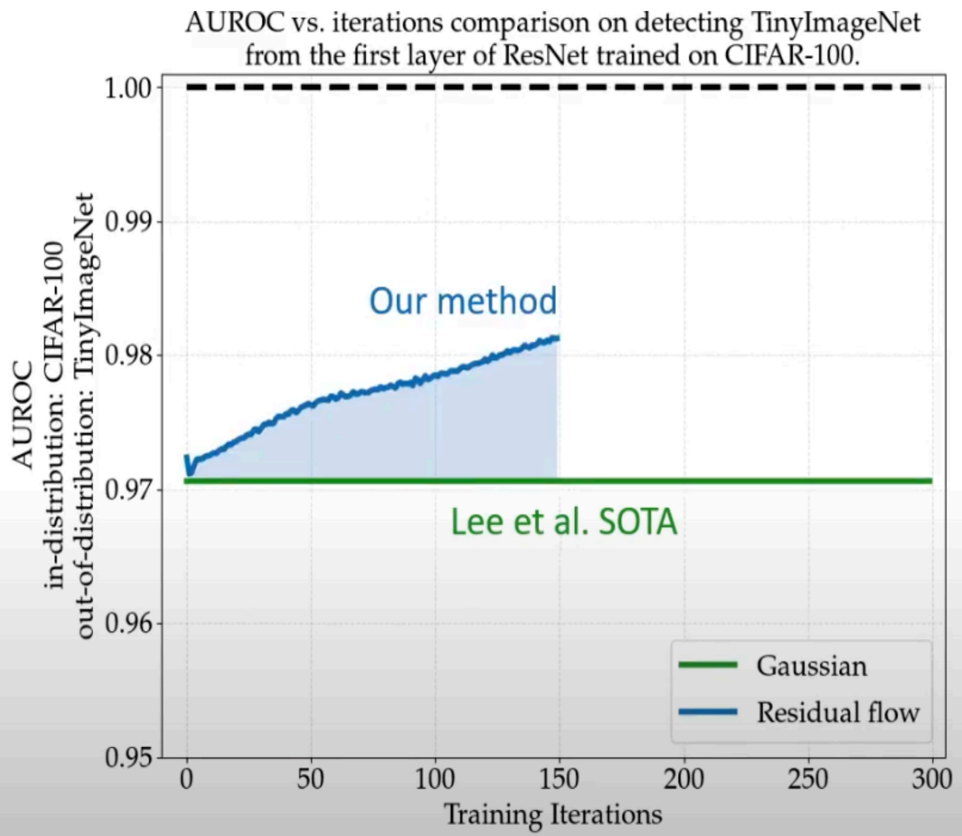
Results

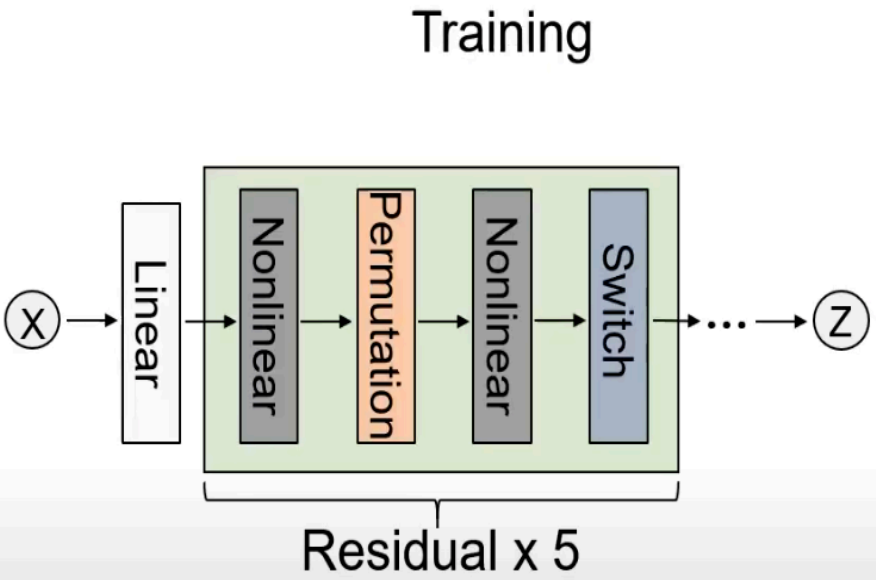


Initialization

From: https://www.youtube.com/watch?v=bNW1a1RYGWk

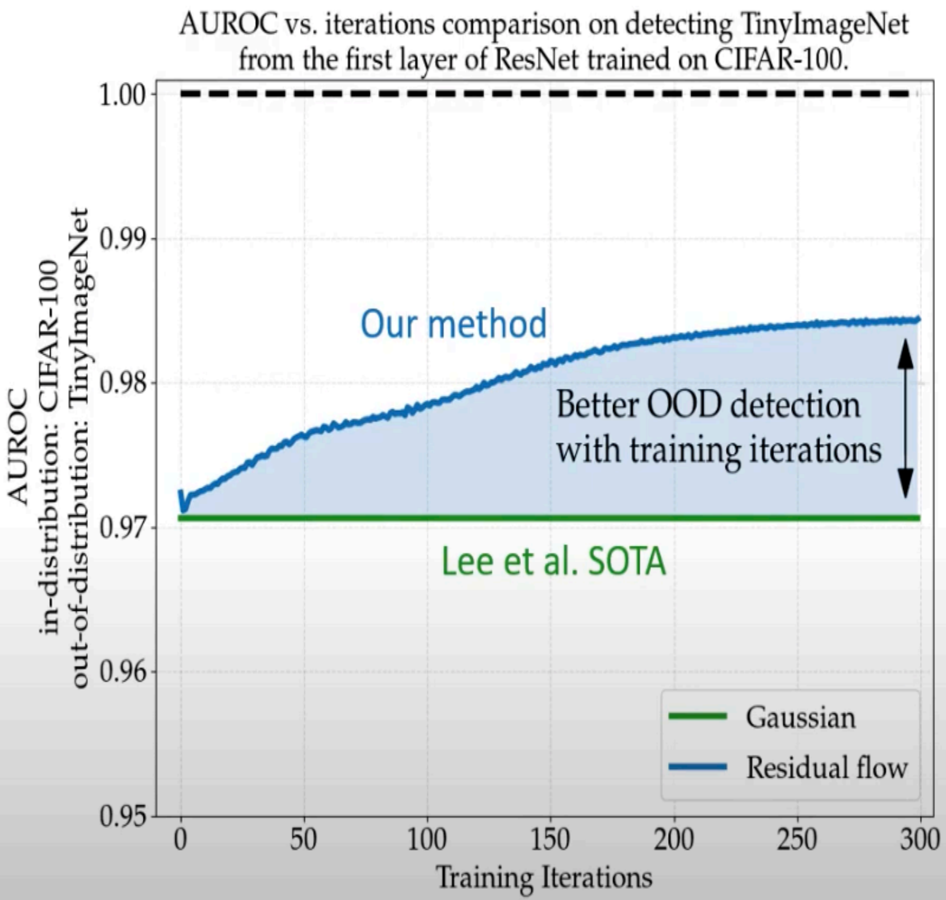
Results





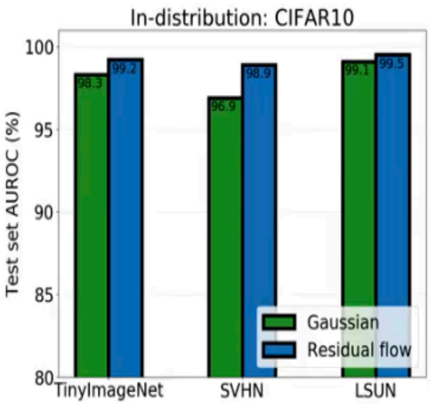
From: https://www.youtube.com/watch?v=bNW1a1RYGWk

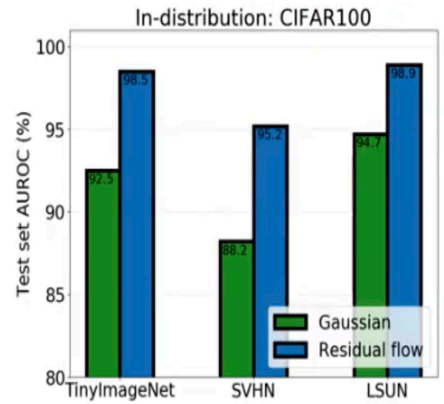
Results



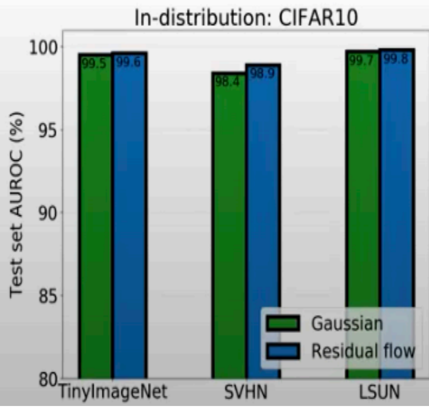
From: https://www.youtube.com/watch?v=bNW1a1RYGWk

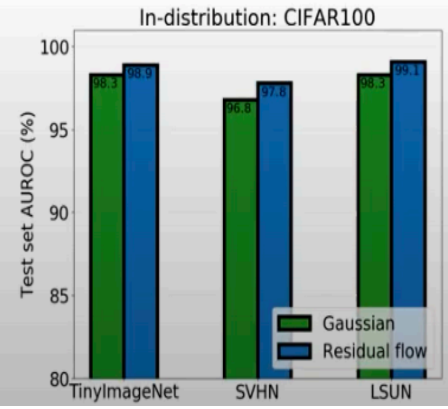
DenseNet





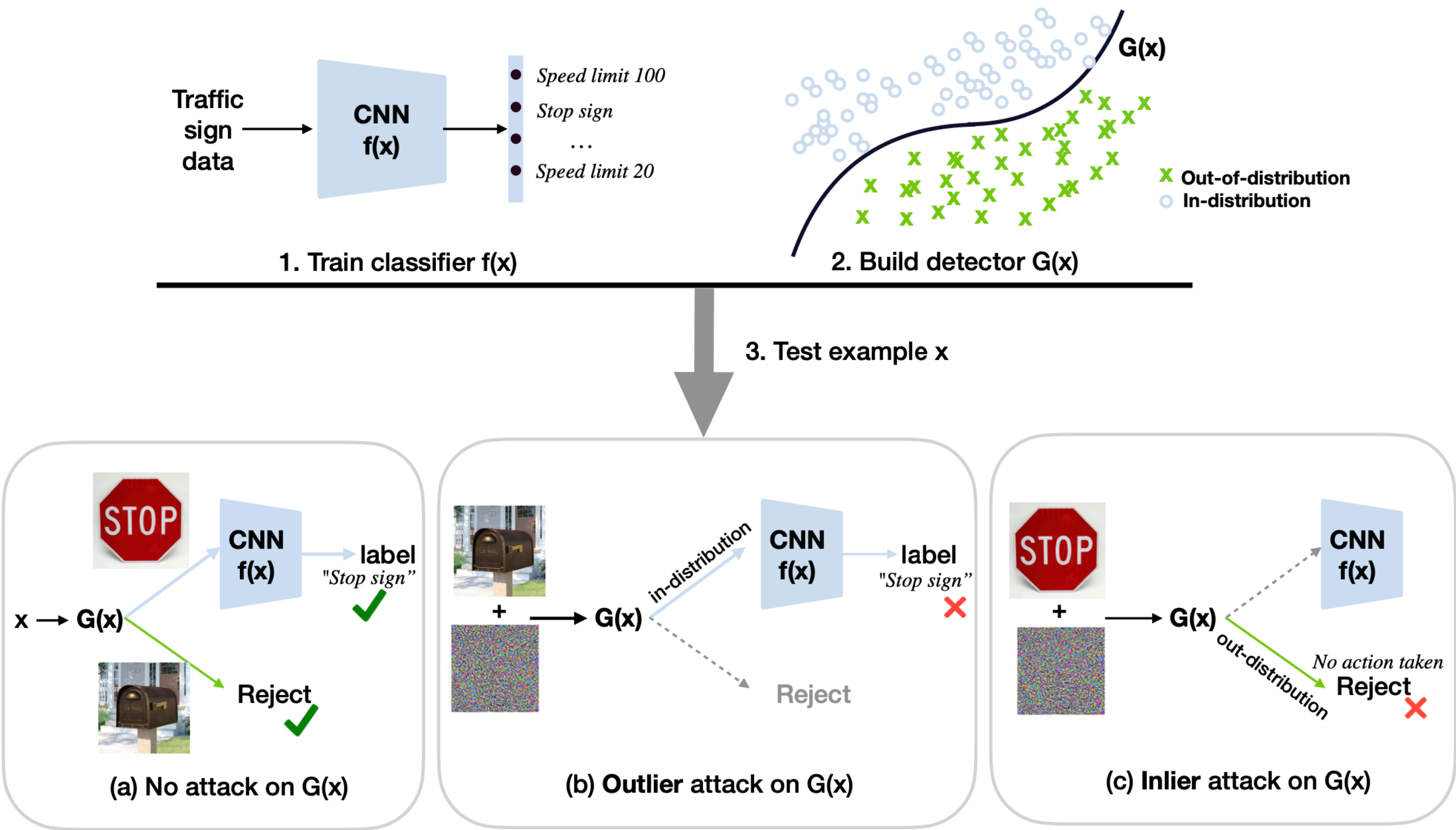
ResNet





Future work

- Applying this general model to different data: speech recognition and natural language processing
- Fixing adversarial attack problems [Chen et al.]



From: [3]

Conclusion

1. Can be applied to a trained network without retraining or changing the underlying architecture
2. No compromise on classification accuracy
3. Performance improvement
4. Approach can be applied to different data e.g. NLP
5. Method has problems with In- and Outlier attacks

Thank you!

References

- [1] Deep Residual Flow for Out of Distribution Detection, Ev Zisselman and Aviv Tamar, https://arxiv.org/pdf/2001.05419.pdf
- [2] Principled Detection of Out-of-Distribution Examples in Neural Networks , Liang et al., https://arxiv.org/pdf/1706.02690v1.pdf
- [3] Robust Out-of-distribution Detection for Neural Network, Chen et al., https://arxiv.org/pdf/2003.09711.pdf
- [4] Normalizing Flows: An Introduction and Review of Current Methods, Kobyzev et al, https://arxiv.org/pdf/1908.09257.pdf
- Tree picture: slide 12, 13: https://upload.wikimedia.org/wikipedia/commons/thumb/4/49/Joshua_Tree_01.jpg/1200px-Joshua_Tree_01.jpg
- Cat picture: slide 11, 19: https://upload.wikimedia.org/wikipedia/commons/d/dc/Grumpy_Cat_%2814556024763%29_%28cropped%29.jpg
- Dog picture: slide 11, 19: https://c1.peakpx.com/wallpaper/463/347/648/dog-portrait-a-hybrid-brown-wallpaper.jpg