Building neural networks that
know what they don't know

Balaji Lakshminarayanan
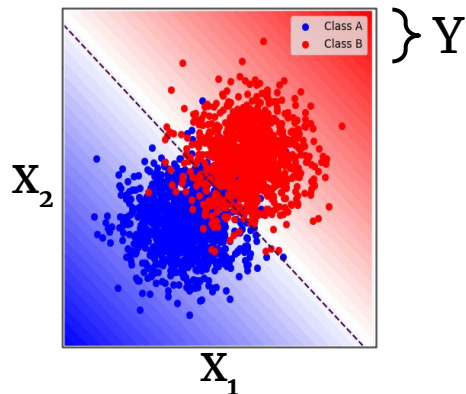balajiln@

Google Research, Brain Team

Motivation

What do we mean by Uncertainty?

Return a distribution over predictions rather than a single prediction.

- **Classification**: Output label along with its confidence.

- **Regression**: Output mean along with its variance.

Good uncertainty estimates quantify **when we can trust the model's predictions**.
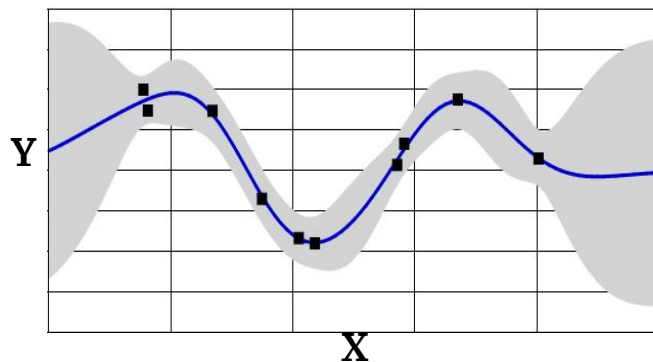

$$p(\mathbf{y}|\mathbf{x})$$

Image credit: Eric Nalisnick

What do we mean by Out-of-Distribution Robustness?

I.I.D. $\qquad p_{\text{TEST}}(y,x) = p_{\text{TRAIN}}(y,x)$
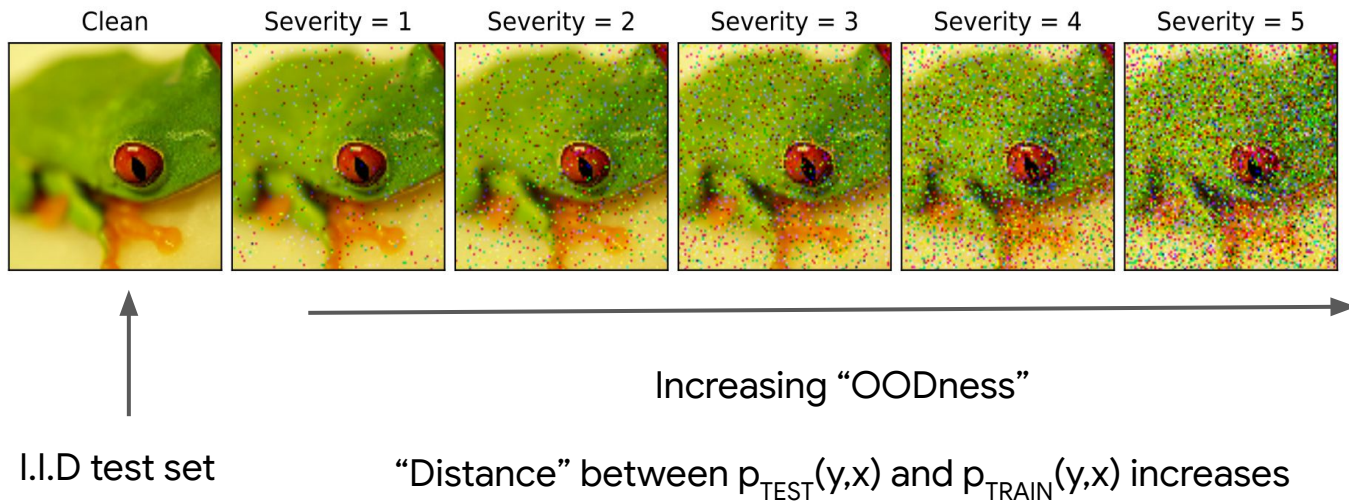
(*Independent and Identically Distributed*)

O.O.D. $\qquad p_{\text{TEST}}(y,x) \neq p_{\text{TRAIN}}(y,x)$
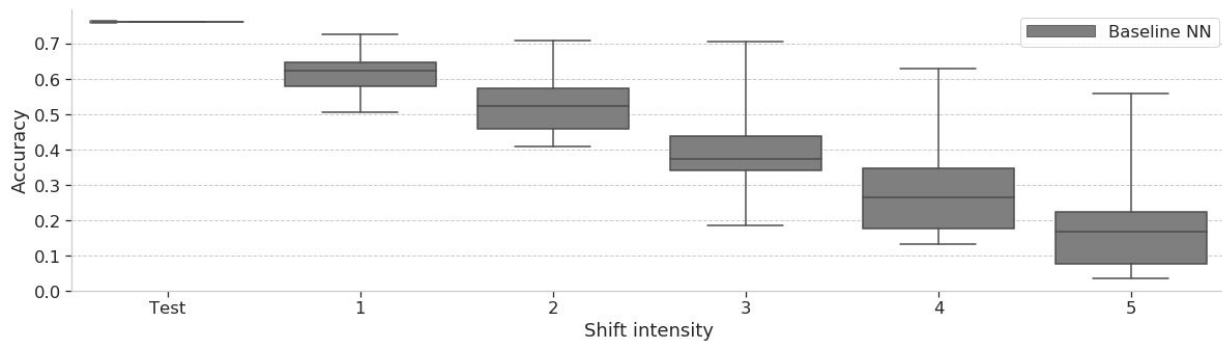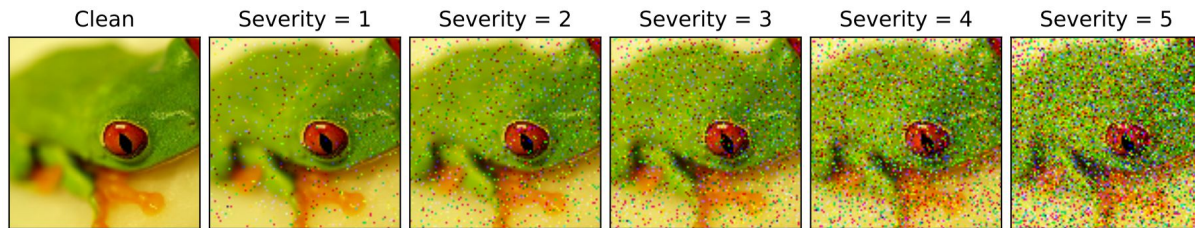
Examples of dataset shift:

- **Covariate shift.** Distribution of features $p(x)$ changes and $p(y|x)$ is fixed.

- **Open-set recognition.** New classes may appear at test time.

- **Subpopulation shift.** Frequencies of data subpopulations changes.

- **Label shift.** Distribution of labels $p(y)$ changes and $p(x|y)$ is fixed.

ImageNet-C: Varying Intensity for Dataset Shift

Clean Severity = 1 Severity = 2 Severity = 3 Severity = 4 Severity = 5

I.I.D test set

Increasing "OODness"

"Distance" between $p_{TEST}(y,x)$ and $p_{TRAIN}(y,x)$ increases

Image source: Benchmarking Neural Network Robustness to Common Corruptions and Perturbations, Hendrycks & Dietterich, 2019.

Neural networks do not generalize under covariate shift

- **Accuracy drops** with increasing shift on Imagenet-C

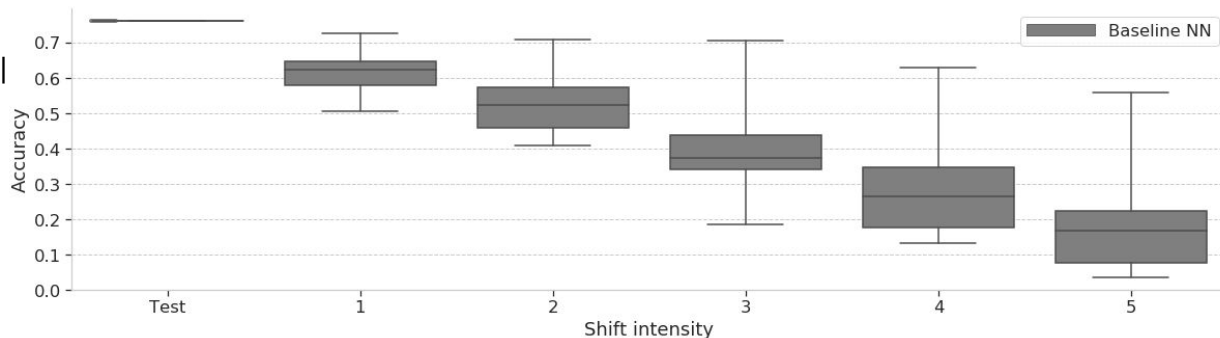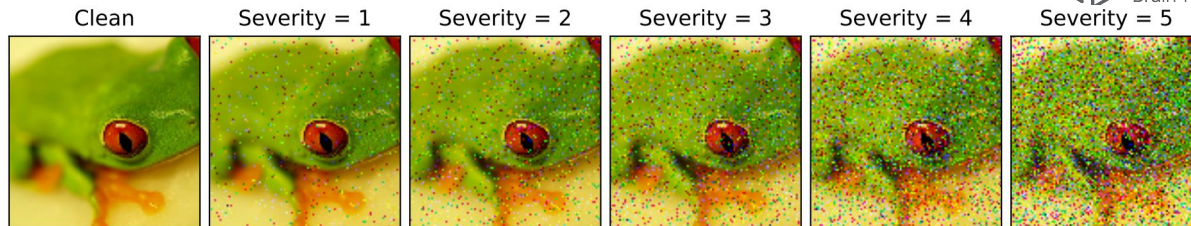- But do the models know that they are less accurate?

Neural networks *do not know when they don't know*
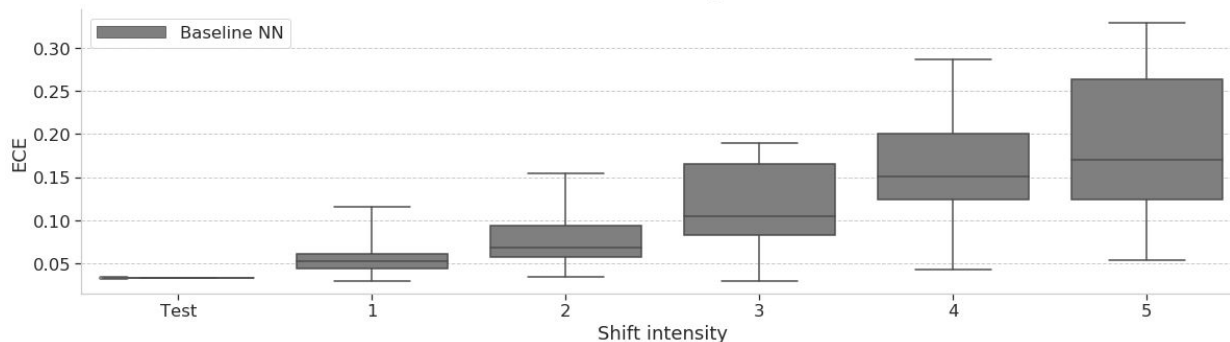

- **Expected Calibration error (↓)**

Calibration Error = |Confidence - Accuracy|

predicted probability of correctness

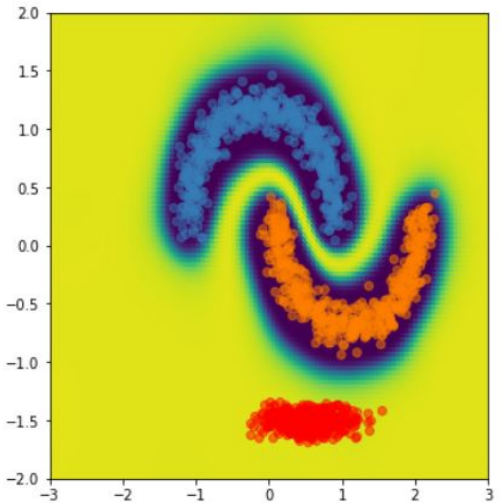observed frequency of correctness

- **Quality of uncertainty degrades with shift -> "overconfident mistakes"**

Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift?, Ovadia et al. 2019

Models assign high confidence predictions far away from training data

Ideal desired behavior

Trust model when x* is close to $p_{TRAIN}(x,y)$

Deep neural networks assign high confidence predictions far away from $p_{TRAIN}(x,y)$

High uncertainty (low confidence)
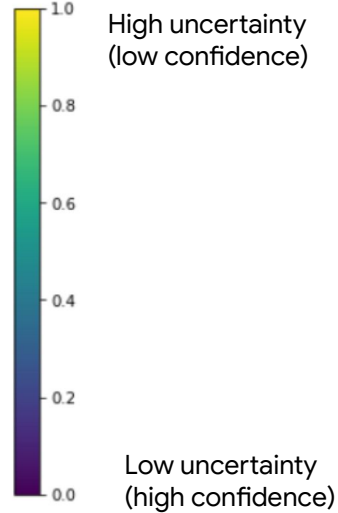
Low uncertainty (high confidence)

Image source: "Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness" Liu et al. 2020

Models assign high confidence predictions far away from training data
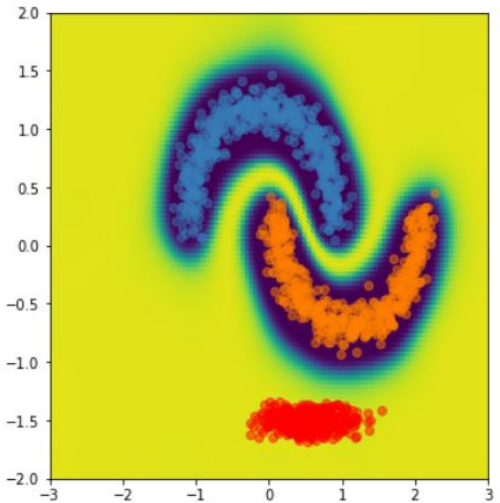
Google AI
Brain Team

Ideal desired behavior

Trust model when x* is close to $p_{TRAIN}(x,y)$

Deep neural networks assign high confidence predictions far away from $p_{TRAIN}(x,y)$

High uncertainty (low confidence)

Low uncertainty (high confidence)

Image source: "Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness" Liu et al. 2020

Applications
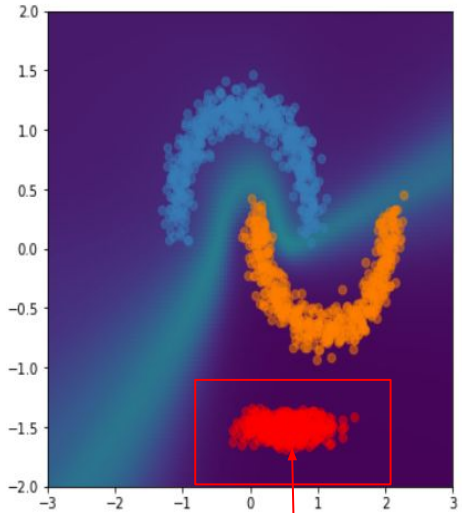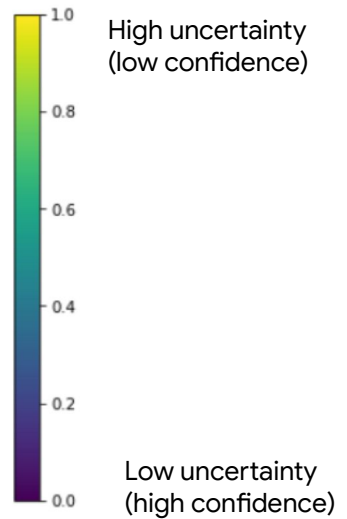
Healthcare

- Use model uncertainty to decide when to trust the model or to defer to a human.

- Selective prediction, Cost-sensitive decision making
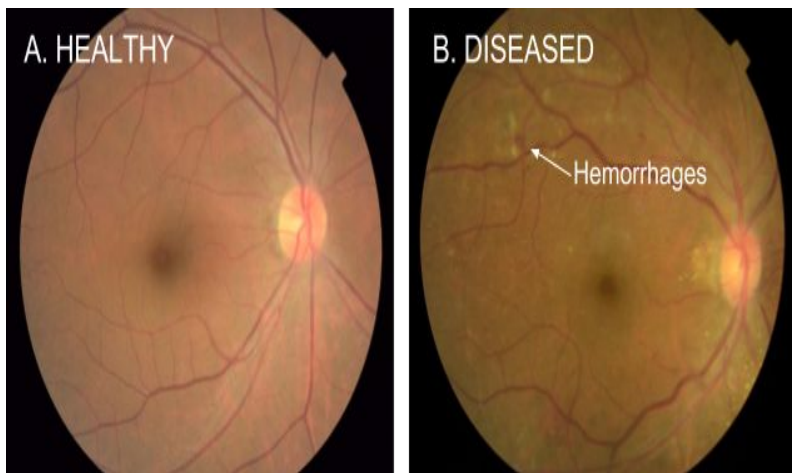
Diabetic retinopathy detection from fundus images
Gulshan et al, 2016

	True label	
	Healthy	Diseased
Predict Healthy	0	10
Predict Diseased	1	0
Abstain/Defer "I don't know"	0.5	0.5

Action

Self-driving cars

Dataset shift:

- Time of day / Lighting
- Geographical location (City vs suburban)
- Changing conditions (Weather / Construction)
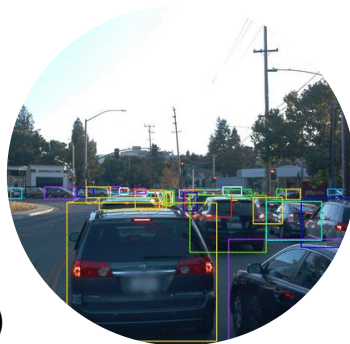
Daylight

Night

Weather

Construction

Image credit: Sun et al, Waymo Open Dataset

Downtown

Suburban

Open Set Recognition
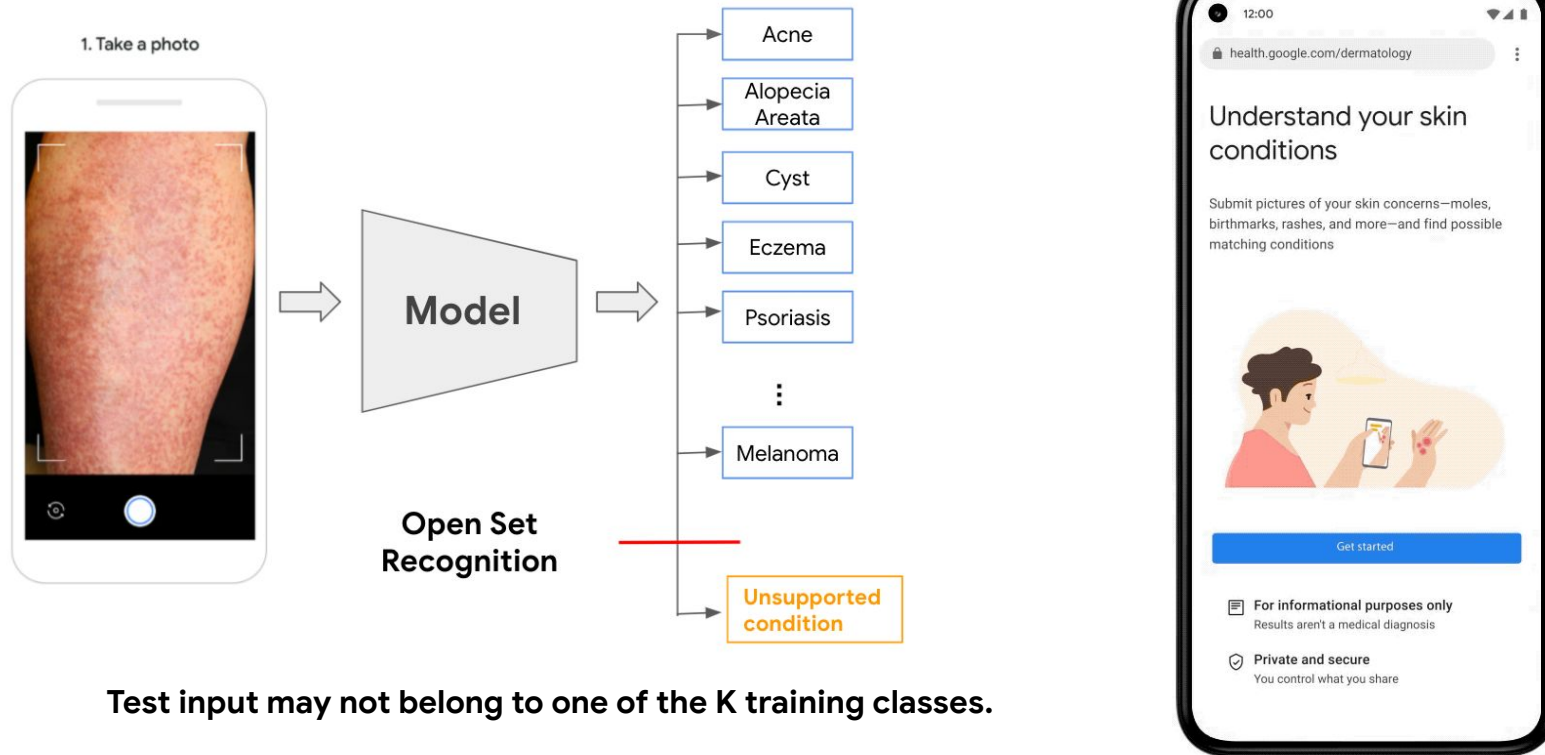
Test input may not belong to one of the K training classes.

High I.I.D. accuracy is not sufficient, need to be able to detect OOD inputs.

Image source: https://blog.google/technology/health/ai-dermatology-preview-io-2021/

Active Learning

- Use model uncertainty to improve data efficiency and model performance in blindspots.

Image source: Active Learning Literature Survey, Settles 2010

Bayesian Optimization and Experimental Design

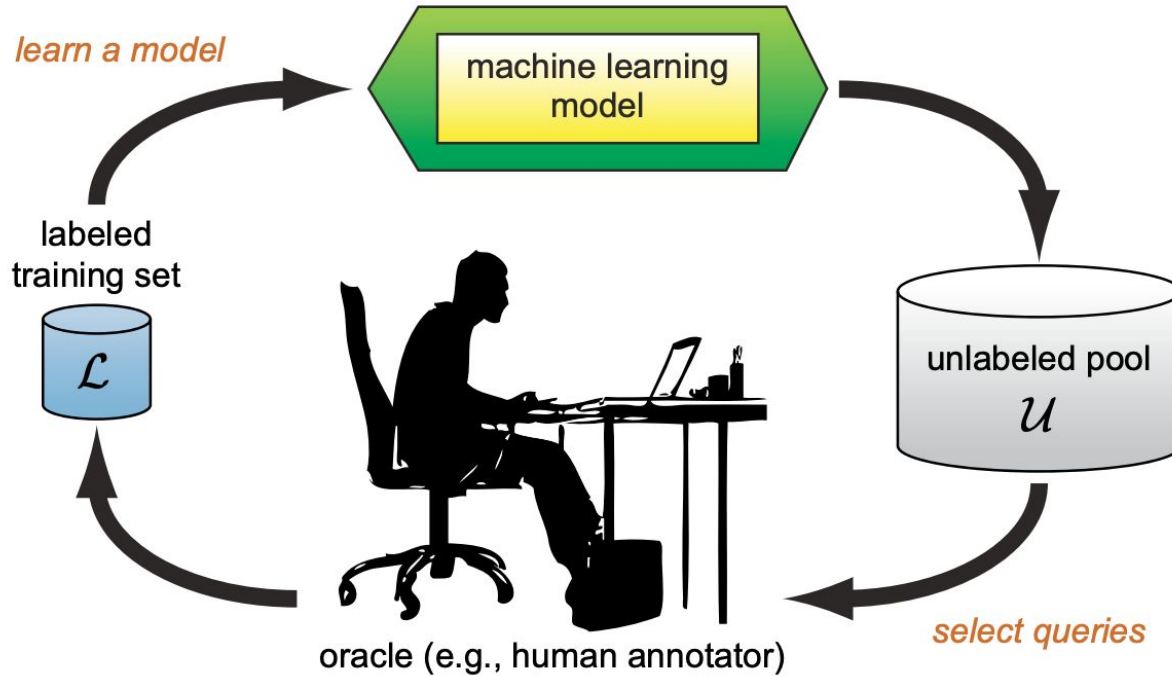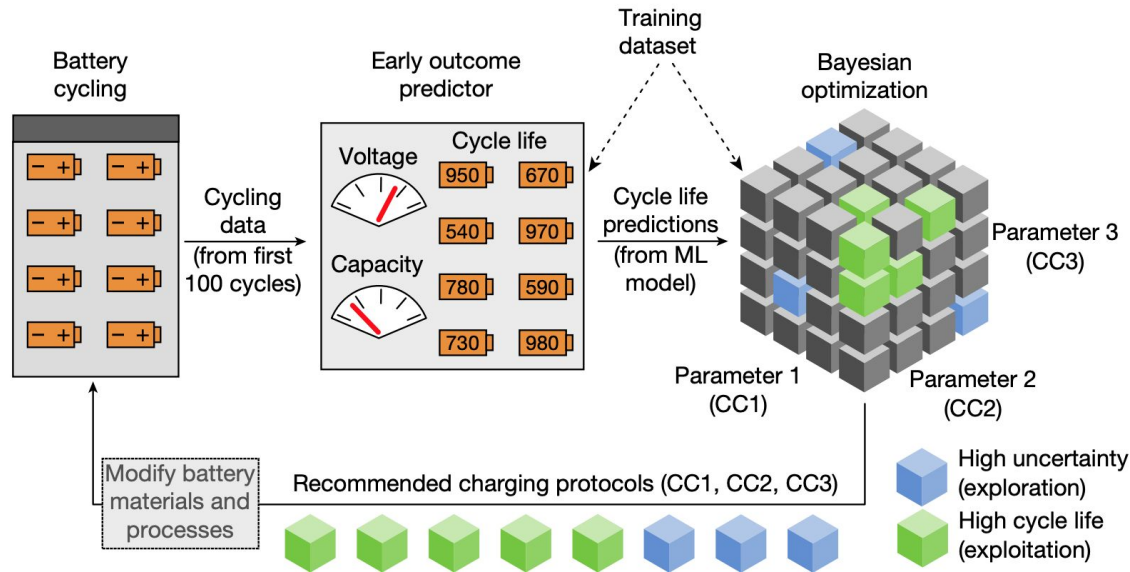- Hyperparameter optimization and experimental design
  - Used across large organizations and the sciences
- Photovoltaics, chemistry experiments, AlphaGo, batteries, materials design

Image source: Attia et al. 2020 Closed-loop optimization of fast-charging protocols for batteries with machine learning

Bandits and Reinforcement Learning

- Decision making with asymmetric losses

$$\ell(\mu) \neq \mathbb{E}_{z \sim N(\mu, \sigma^2)}[\ell(z)]$$

- Modeling uncertainty is crucial for **exploration vs exploitation** trade-off

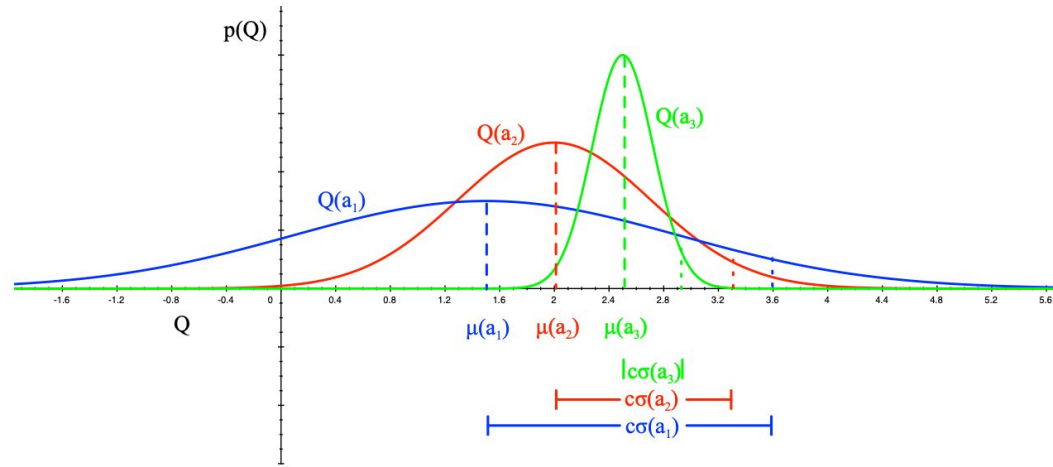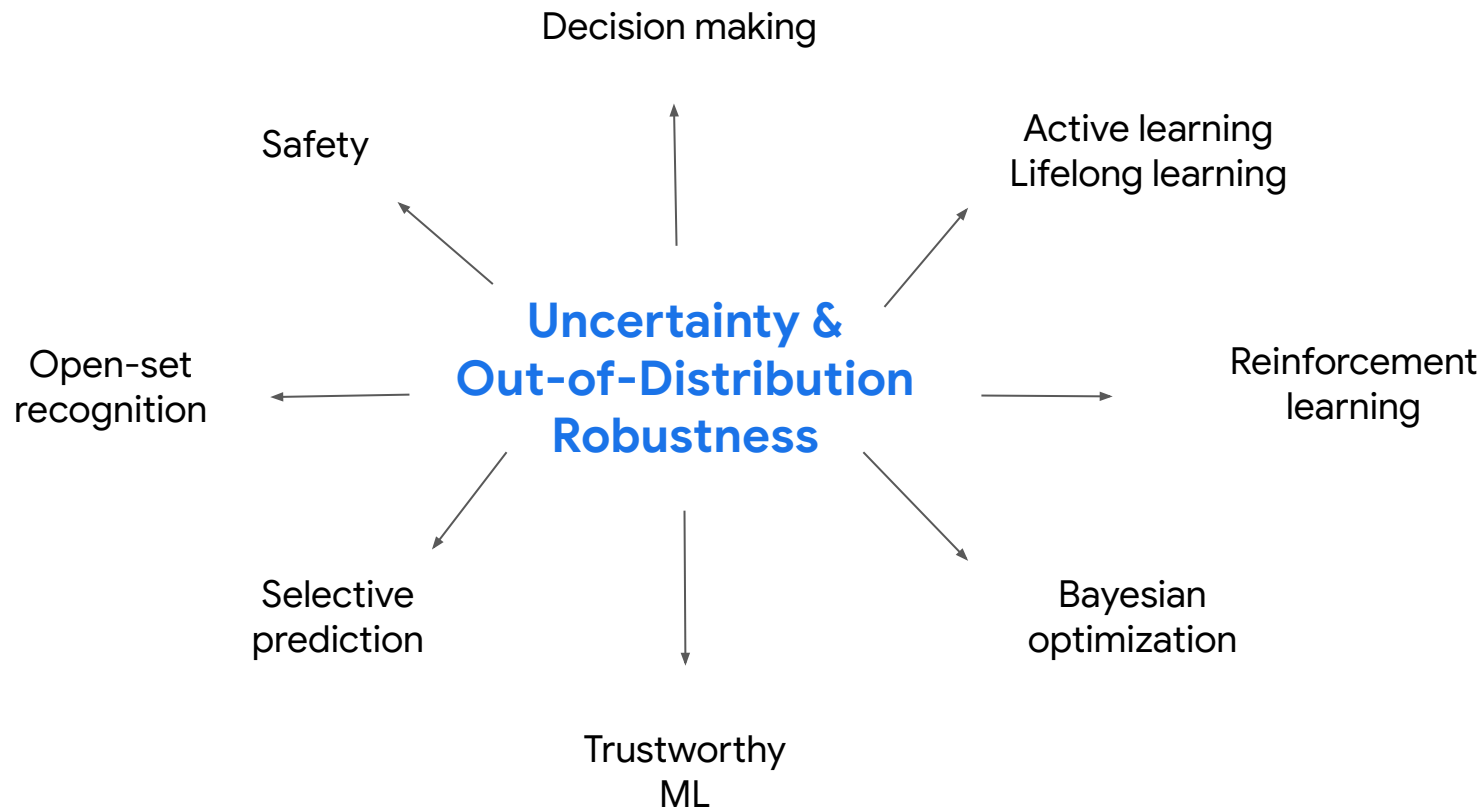*Image source*: David Silver's RL course

- Non-stationarity

All models are wrong, but ~~some~~ *models that know when they are wrong*, are useful.

Decision making

Safety

Active learning
Lifelong learning

**Uncertainty &
Out-of-Distribution
Robustness**

Open-set
recognition

Reinforcement
learning

Selective
prediction

Bayesian
optimization

Trustworthy
ML

Overview of Methods

Probabilistic Deep Learning

- Parametrize "base model".

$$p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$$

- Specify prior over functions.

$$p(\boldsymbol{\theta})$$

- Capture model uncertainty by approximating the posterior.

$$p(\boldsymbol{\theta} \mid \mathcal{D})$$

- Average predictions over multiple functions (ensemble or Bayesian NN)

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

# Cartoon: Uncertainty/Robustness vs Compute frontier

Ensembles /
Bayesian NNs

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

Quality of Uncertainty / Robustness

Single NN $\quad p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$

Compute (Inference time / Inference memory)
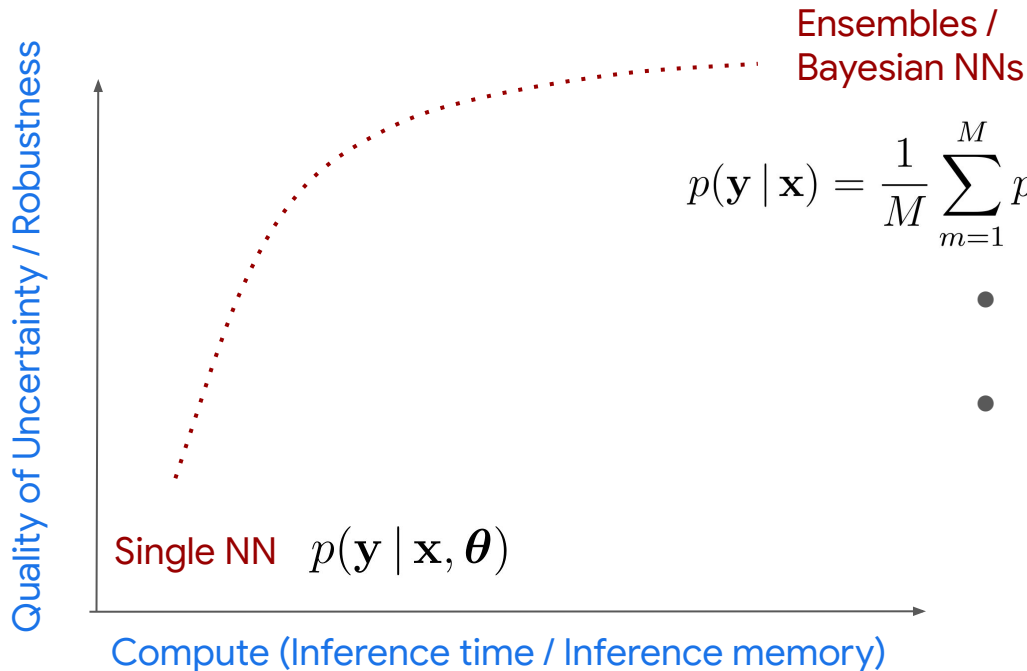
- Practitioners can pick "operating point" depending on constraints of application.
- Probabilistic framework gives an unifying view. Improving p(y|x) improves performance on **all** downstream tasks (accuracy/calibration under shift, selective prediction, open set recognition, etc) as opposed to custom techniques for tasks.

Orthogonal ways of improving performance

Quality of Uncertainty / Robustness (vertical axis)

Ensembles /
Bayesian NNs
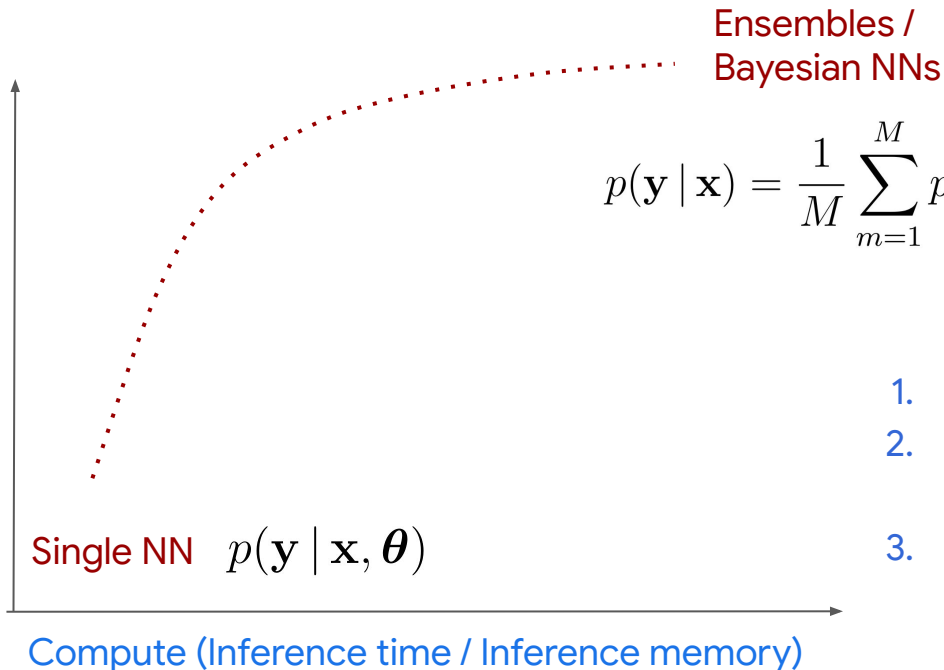
$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

"Composable" toolkit

1. Improve the "base" model $p(y|x,\theta)$
2. Efficiently average predictions over diverse set of functions $\theta_1, \theta_2 \dots \theta_M$
3. Better inductive biases for representations (e.g. pre-training or data augmentation)

Single NN $\quad p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$

Compute (Inference time / Inference memory)
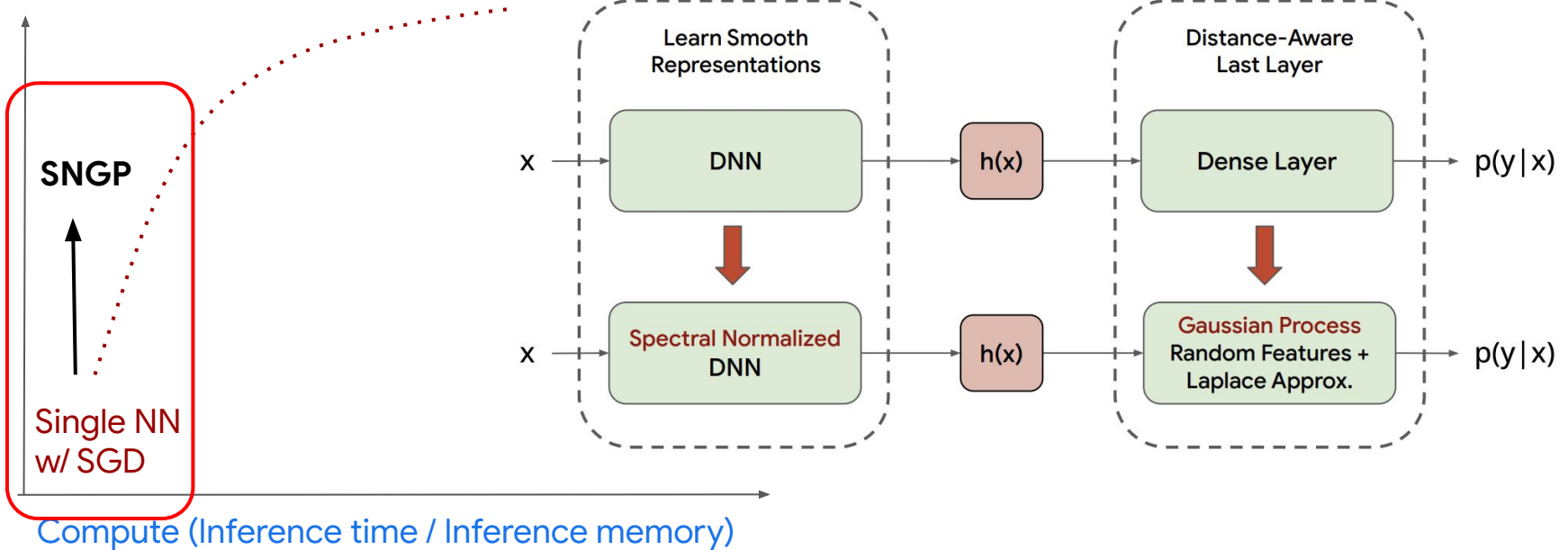
Composing can further improve performance!

Improving Single Model Uncertainty via Distance Awareness
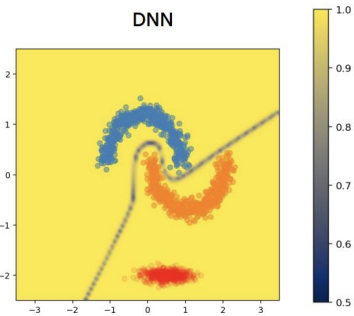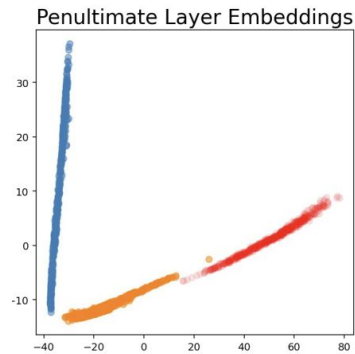
Jeremiah Liu (jereliu@) et al.

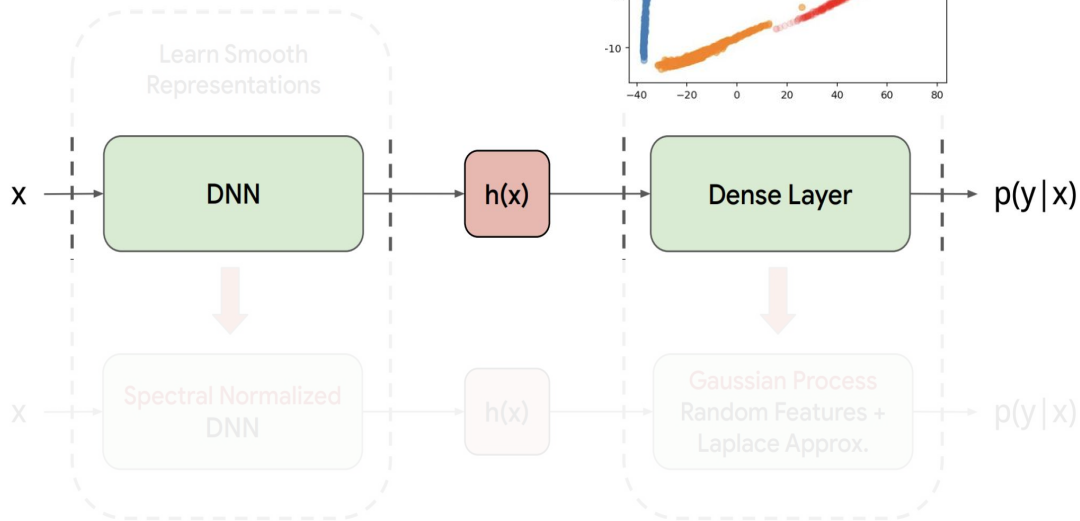# SNGP improves single model uncertainty with two simple changes: Spectral-normalization (SN) + Last-layer Gaussian Process (GP)

Why do vanilla DNNs assign high confidence predictions far away from training data?

Penultimate Layer Embeddings

DNN

x → DNN → h(x) → Dense Layer → p(y|x)

Learn Smooth Representations

Spectral Normalized DNN   h(x)   Gaussian Process Random Features + Laplace Approx.

x → → → → p(y|x)
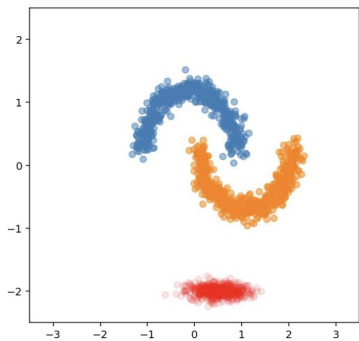
1. Vanilla NNs can map inputs far away in input space to close points in latent space (cf. "shortcut learning").

2. Confidence is a function of distance from boundary (and not the training data).

Google AI Brain Team

Idea 1: Enforce bi-Lipschitz smoothness via spectral normalization
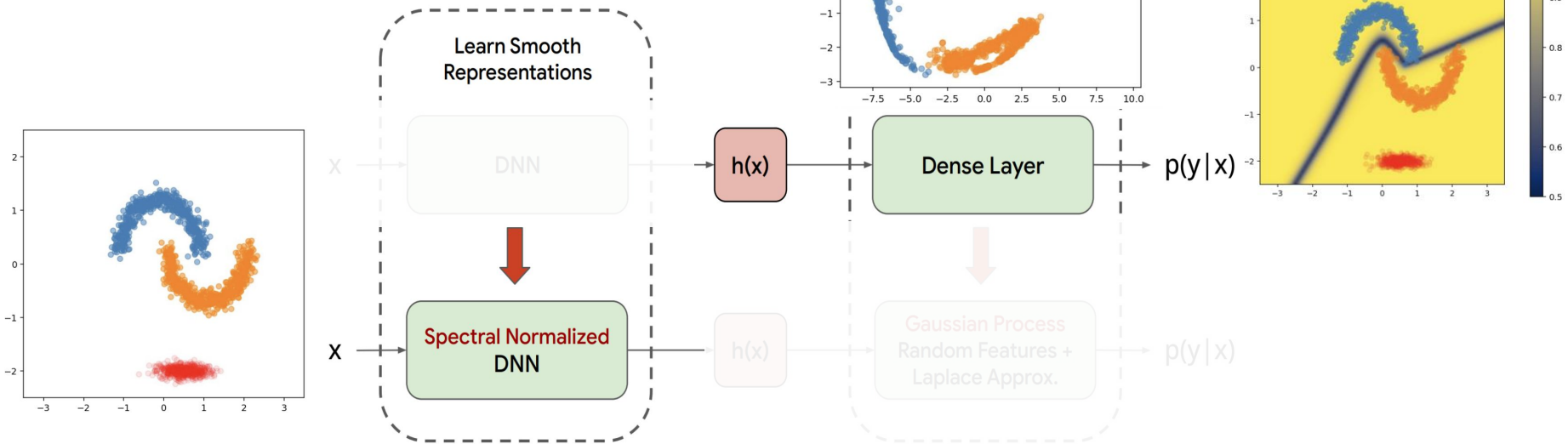
Penultimate Layer Embeddings

DNN-SN

Learn Smooth Representations

DNN

Spectral Normalized DNN

x

h(x)

Dense Layer

p(y|x)

h(x)

Gaussian Process Random Features + Laplace Approx.

p(y|x)

Bi-Lipschitz smoothness discourages inputs far away in input space getting mapped close in latent space.

Google AI Brain Team

Idea 1: Enforce bi-Lipschitz smoothness

Idea 2: Replace last dense layer with "Gaussian process" layer
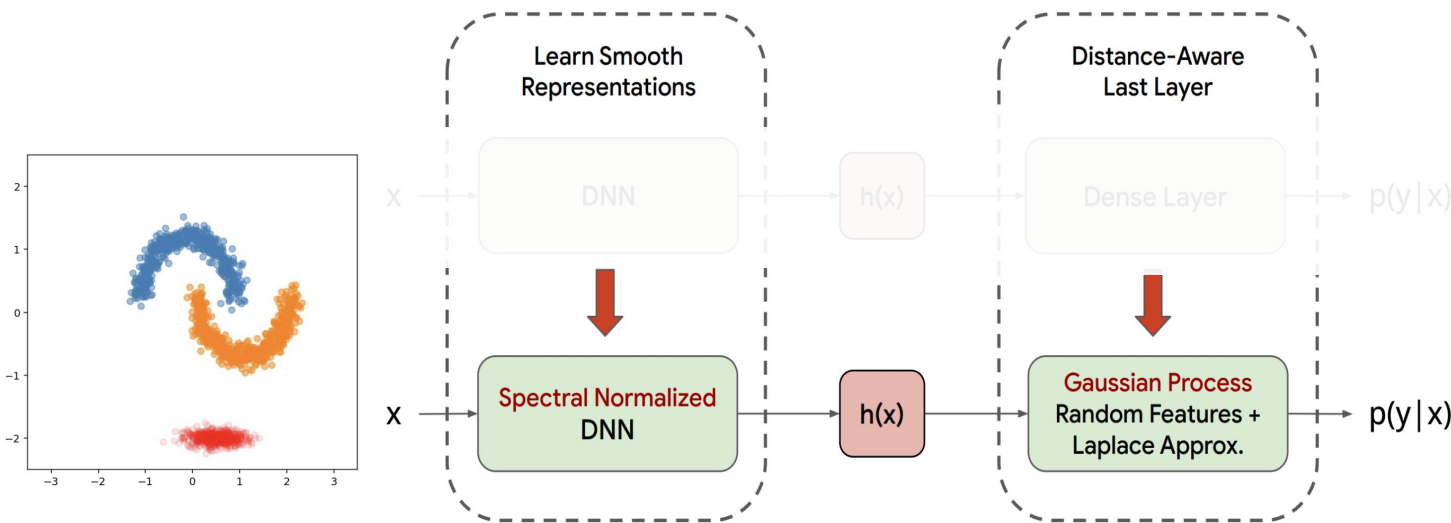
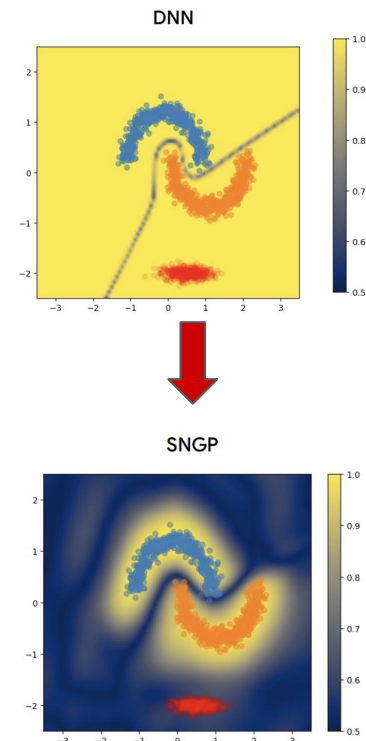Bi-Lipschitz smoothness discourages inputs far away in input space getting mapped close in latent space

For GP layer, confidence is a function of distance from the training data.

Spectral-normalized Neural Gaussian process (SNGP)

Method	Accuracy (↑)	ECE (↓)	OOD AUROC (↑)	AUPR (↑)	Latency (ms / example)
Deterministic	96.5	0.0236	0.8970	0.7573	**10.42**
MCD-GP	95.9	0.0146	0.9055	0.8030	88.38
DUQ	96.0	0.0585	0.9173	0.8058	15.60
MC Dropout	96.5	0.0210	0.9382	0.7997	85.62
Deep Ensemble	**97.5**	0.0128	0.9635	0.8616	84.46
SNGP	96.6	**0.0115**	**0.9688**	**0.8802**	17.36

BERT on an intent detection benchmark

- SNGP improves
  - Accuracy under shift
  - Calibration under shift
  - OOD detection
- Simple to implement
- Low computational/memory overhead
- A building block for better ensembles

Method	Acc (↑)	ECE (↓)	Corrupted Acc/ECE	OOD AUPR (↑) SVHN	CIFAR100	Latency (ms)
Deterministic	96.0	0.023	72.9 / 0.153	0.7810	0.8352	**3.91**
MCD-GP	95.5	0.024	70.0 / 0.100	0.9599	0.8631	29.53
DUQ	94.7	0.034	71.6 / 0.183	0.9733	0.8537	8.68
MC Dropout	96.0	0.024	70.0 / 0.116	0.9714	0.8320	27.10
Deep Ensembles	96.6	**0.010**	**77.9 / 0.087**	0.9640	0.8875	38.10
SNGP (Ours)	95.9	0.018	74.6 / 0.090	**0.9901**	**0.9050**	6.25

Results on CIFAR-10 using Wide ResNet

See also [van Amersfoort+ 2020].
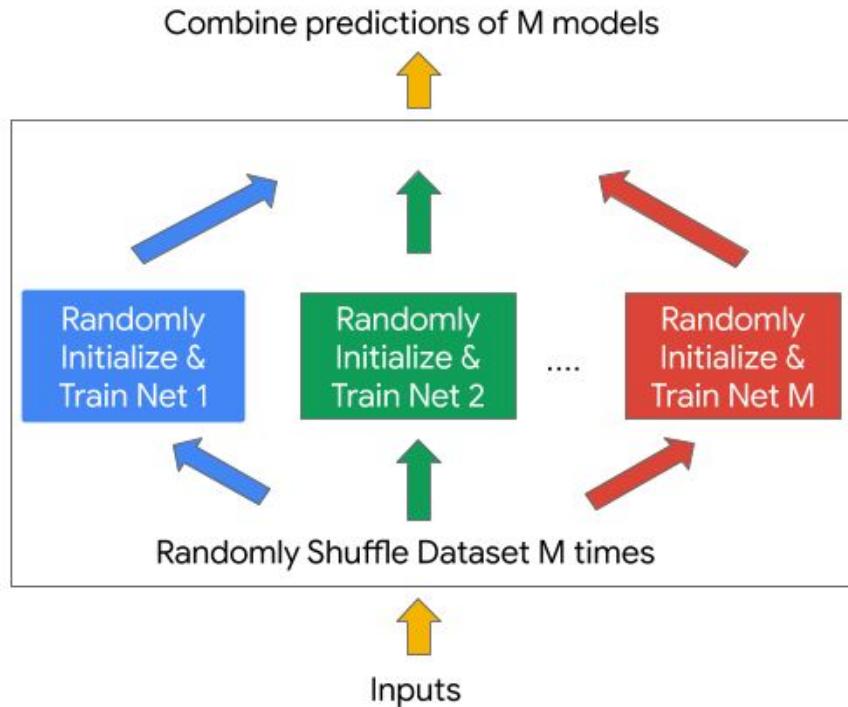
[Liu+ 2020]
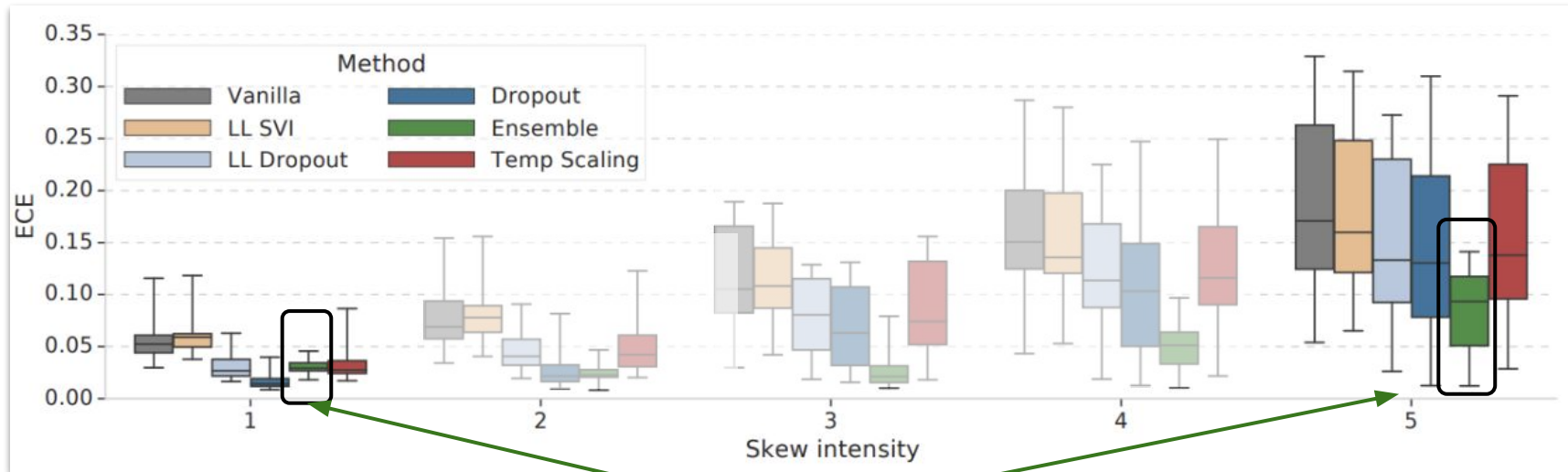
Improving the quality of model uncertainty
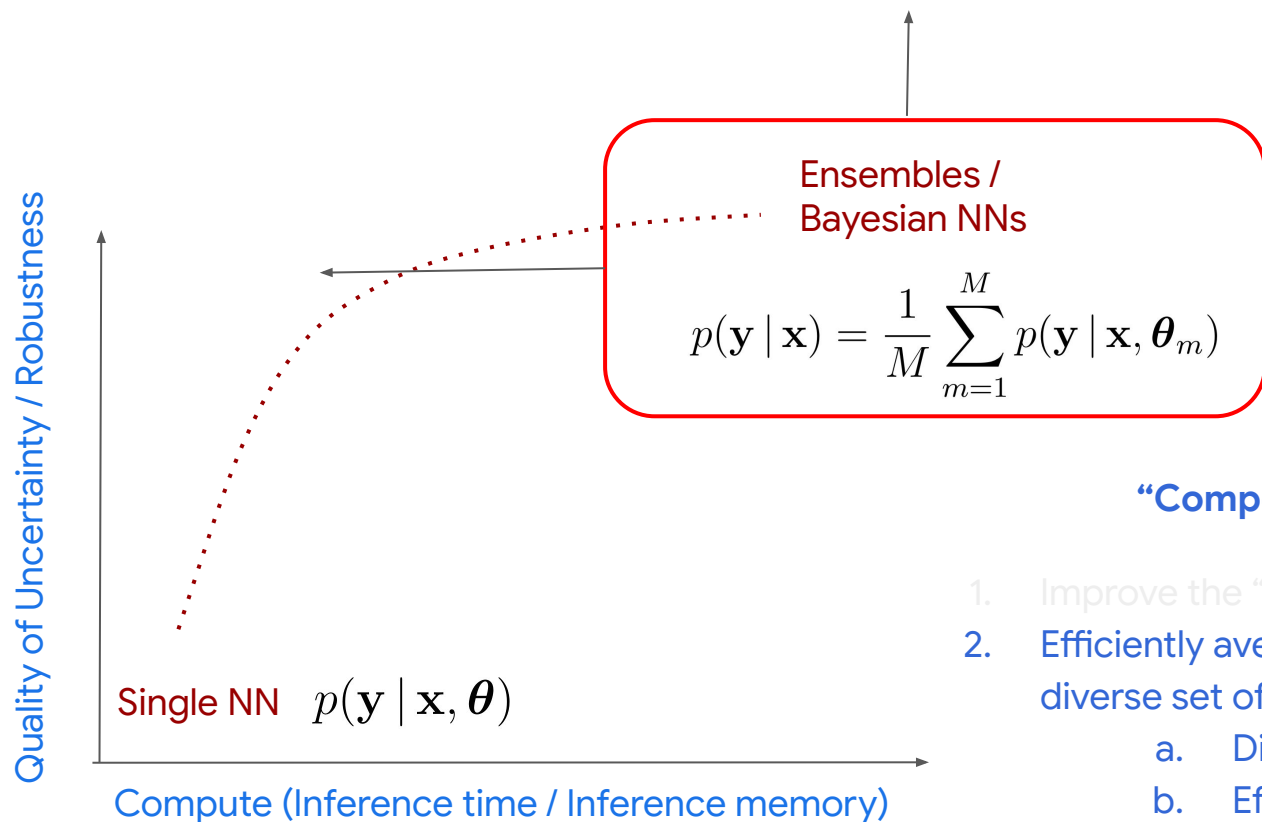
Quality of Uncertainty / Robustness (vertical axis)

Ensembles / Bayesian NNs

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

Single NN $\quad p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$

Compute (Inference time / Inference memory)

"Composable" toolkit

1. Improve the "base" model p(y|x,θ)
2. Efficiently average predictions over diverse set of functions $\theta_1, \theta_2 \dots \theta_M$
   a. Diverse ensembles ↑
   b. Efficient ensembles ←

Surprisingly Simple Baseline: Deep Ensembles

Just re-run standard training but with different random seeds (initializations + SGD shuffling) & combine models.
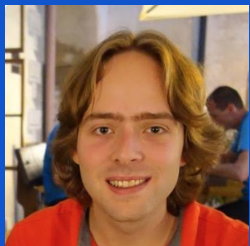

Combine predictions of M models

Randomly Initialize & Train Net 1 .... Randomly Initialize & Train Net 2 .... Randomly Initialize & Train Net M

Randomly Shuffle Dataset M times

Inputs

[Lakshminarayanan+ 2016]

Deep Ensembles improve accuracy and calibration under dataset shift

Deep Ensembles are consistently among the best performing methods, especially under dataset shift

Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift [Ovadia+ 2019]

Deep Ensembles: A Loss Landscape Perspective
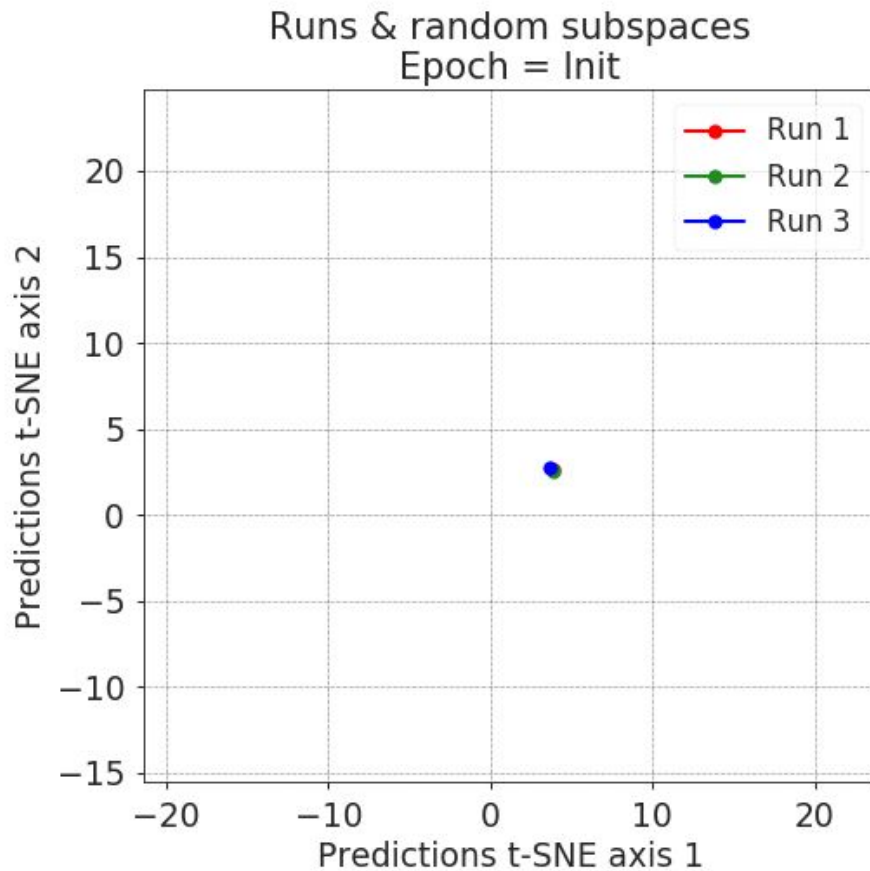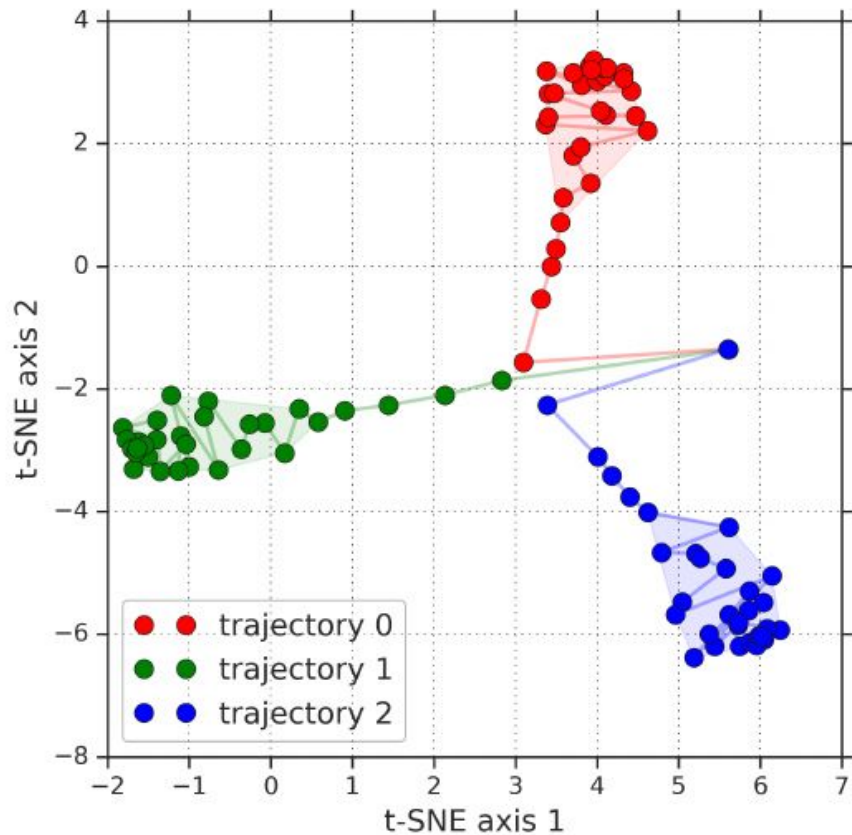
Stanislav Fort*, Clara Huiyi Hu*, Balaji Lakshminarayanan

Motivation: Understand why deep ensembles work well

Variational methods capture local uncertainty around a mode

Ensembles identify different modes but ignore local uncertainty and might not pick the best point from each mode
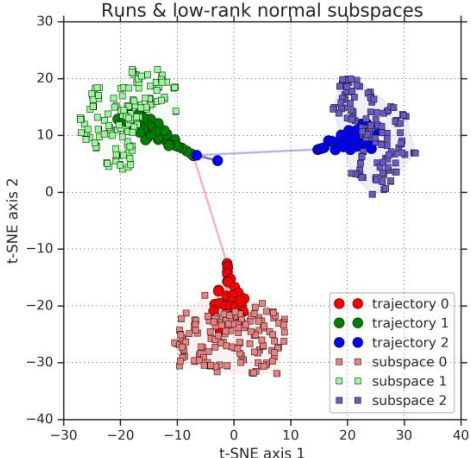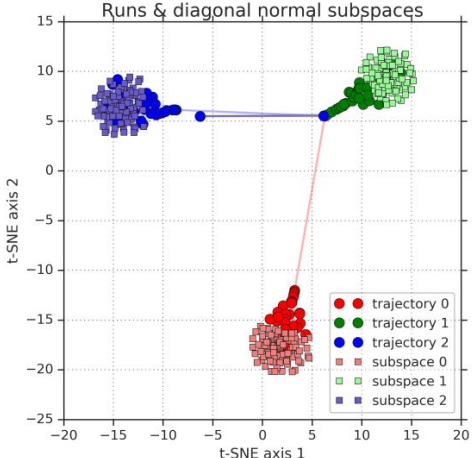
Validation

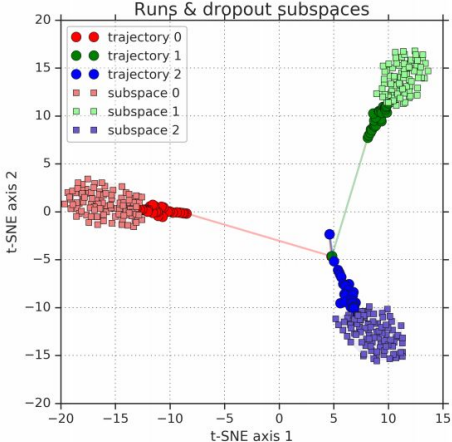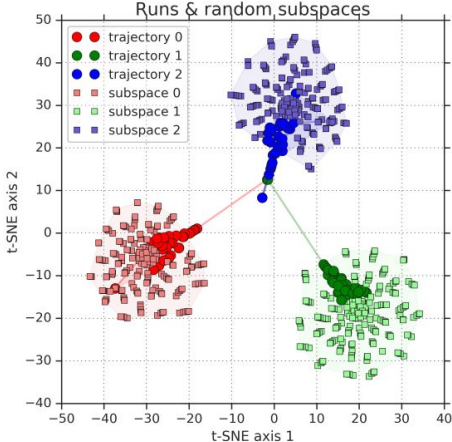Space of solutions

Training
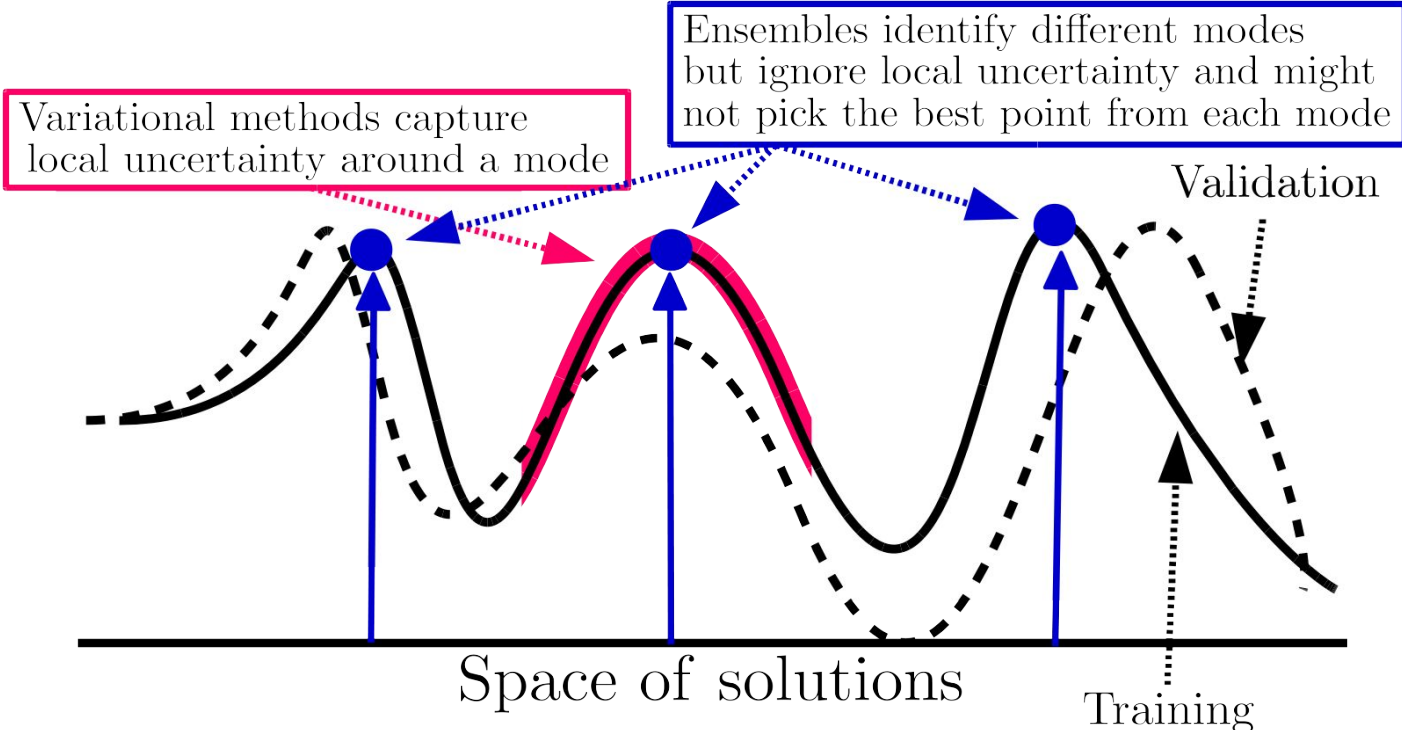
Predictions similarity within and across trajectories

Subspace sampling methods

- Random

- Diagonal Gaussian

- Low-rank Gaussian

- Monte Carlo Dropout

Prediction diversity vs Accuracy

Solution accuracy vs diversity
ResNet20v1 on CIFAR-10
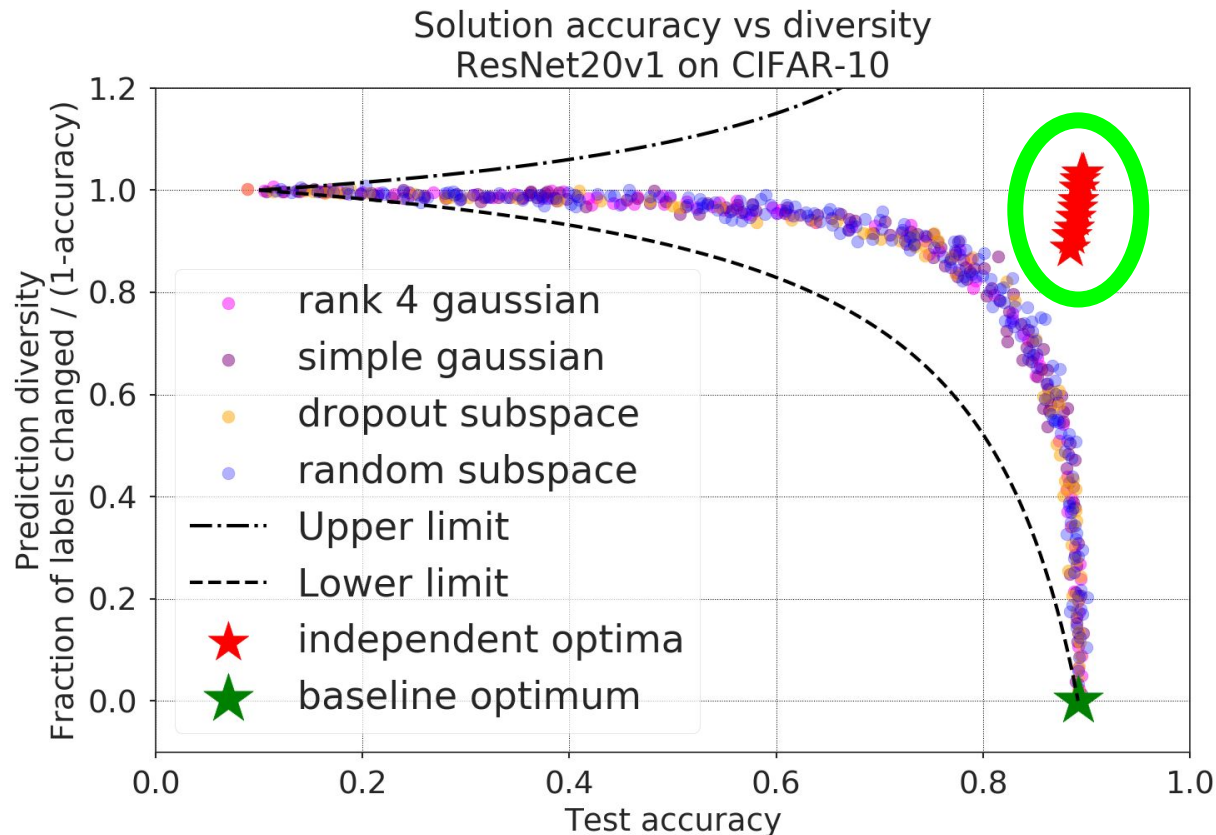
- From a bias-variance tradeoff perspective, we care about low bias as well as diversity.

- We plot fraction of data points where top-1 prediction is different. Note that maximum disagreement depends on the accuracy (completely different mistakes).

- Function diversity is higher across trajectories than within trajectories.

Can we combine uncertainty from subspace (within-mode) and ensembles (across-mode)?
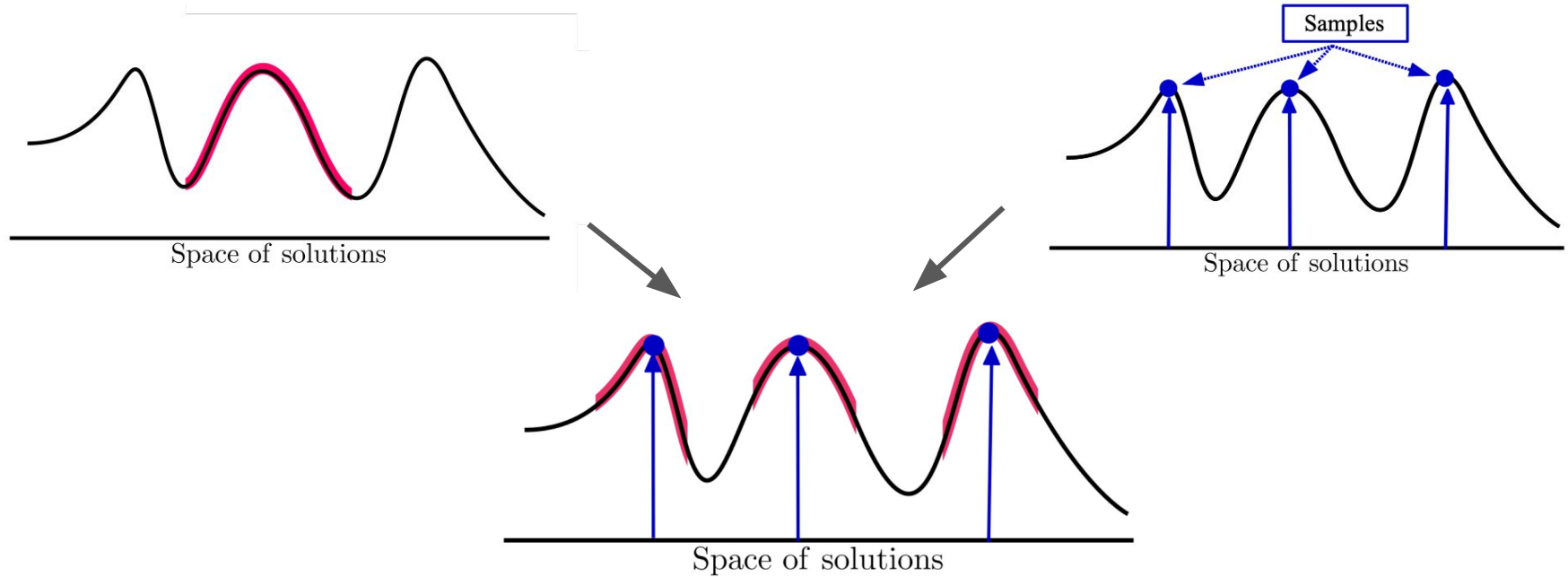
Idea: run multiple trajectories and use subspace sampling within each trajectory
(e.g. Ensemble + Diagonal Gaussian)

Best of both worlds - Results on CIFAR-10 using MediumCNN
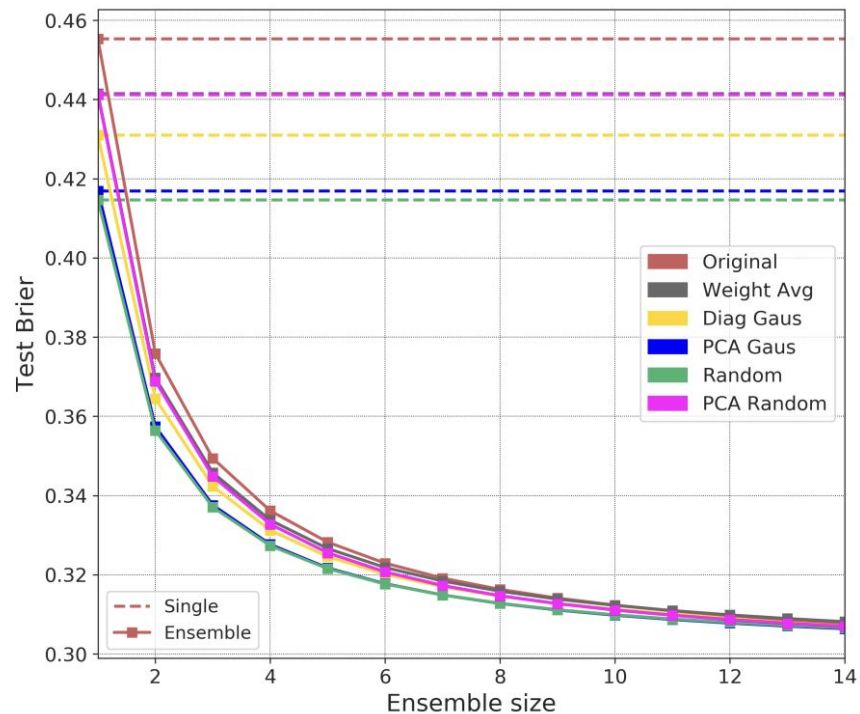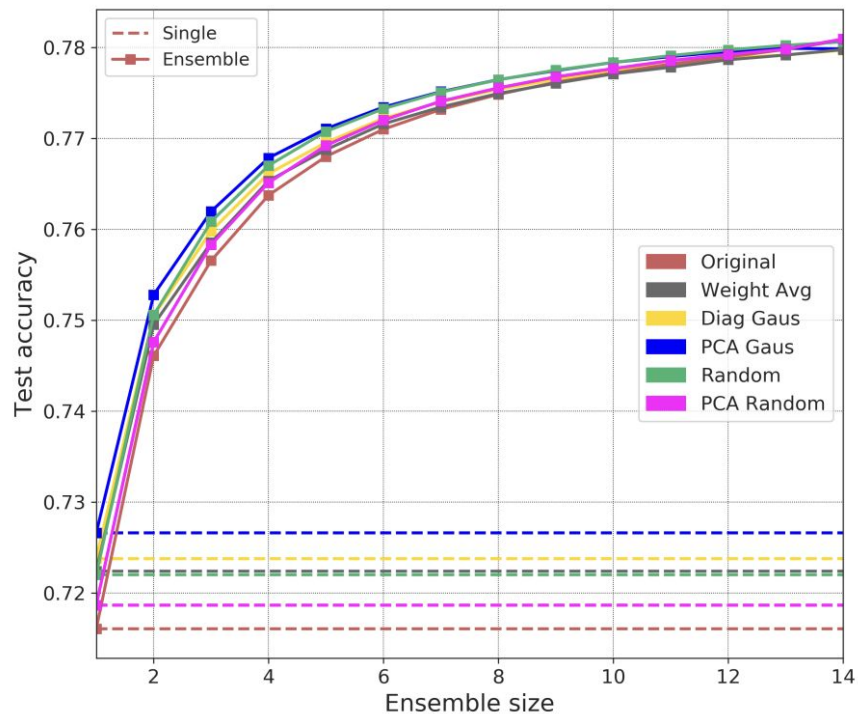
Efficient ensembles lower inference memory and inference time

Idea: Create multiple input heads + multiple output heads (MIMO). Train each head on different SGD batch.
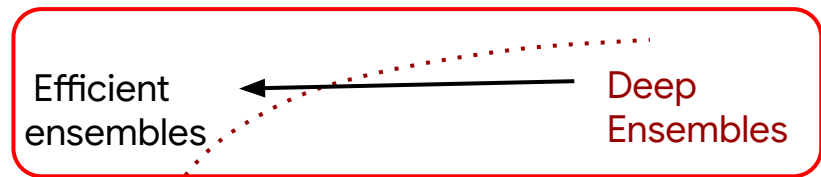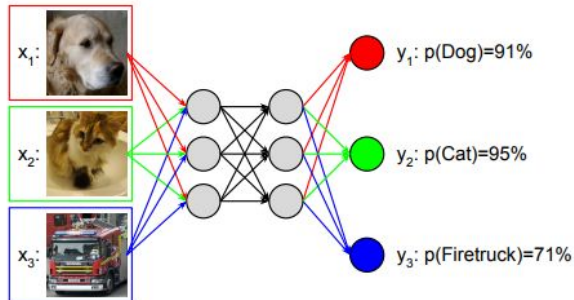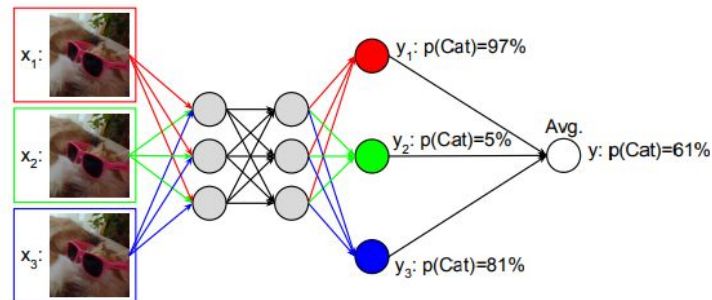
[Training Independent Subnetworks for Robust Prediction. Havasi+ 2020]

Better inductive biases for representations

Quality of Uncertainty / Robustness
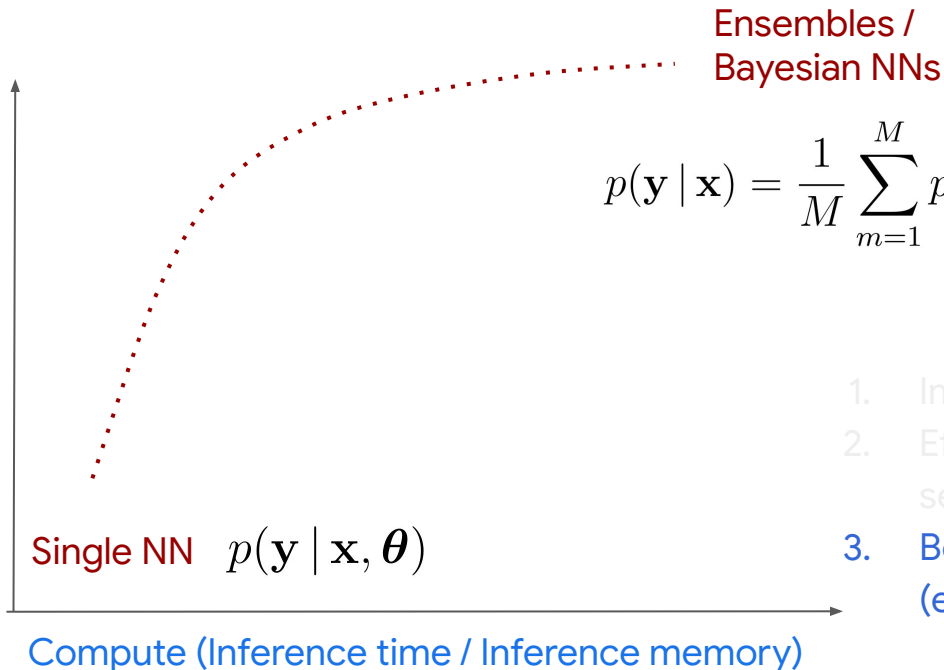
Compute (Inference time / Inference memory)

Ensembles /
Bayesian NNs

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

Single NN $\quad p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$

"Composable" toolkit

1. Improve the "base" model p(y|x,θ)
2. Efficiently average predictions over diverse set of functions $\theta_1, \theta_2 \dots \theta_M$
3. Better inductive biases for representations (e.g. pre-training or data augmentation)

Exploring the limits of OOD detection

Stanislav Fort*, Jie Ren*, Balaji Lakshminarayanan
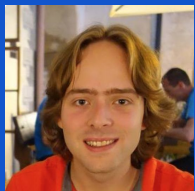
Goal: Improve SOTA on hard OOD detection tasks

Far OOD, AUROC = 99%
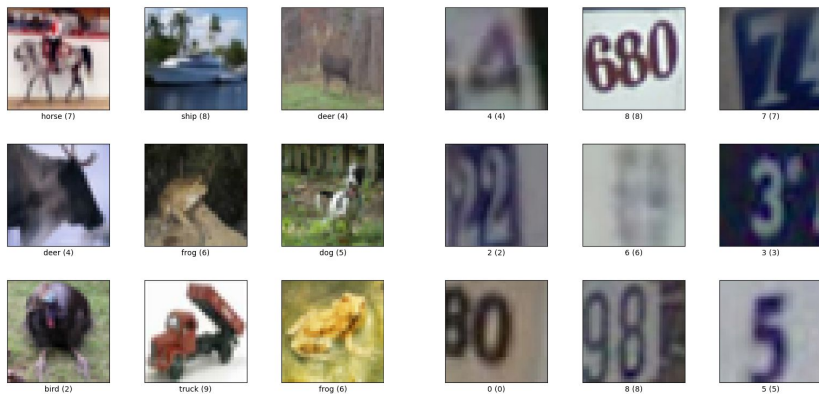
Near OOD, AUROC = 85%
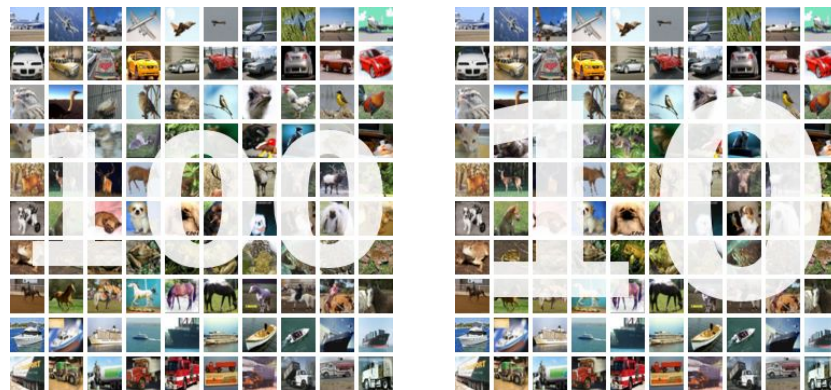
CIFAR-10 (ID)   **SVHN (OOD)**   **CIFAR-100 (ID)**   **CIFAR-10 (OOD)**

Improving near-OOD detection

CIFAR-100 → CIFAR-10

AUROC (%)

85.6%

Hybrid models
for open sets
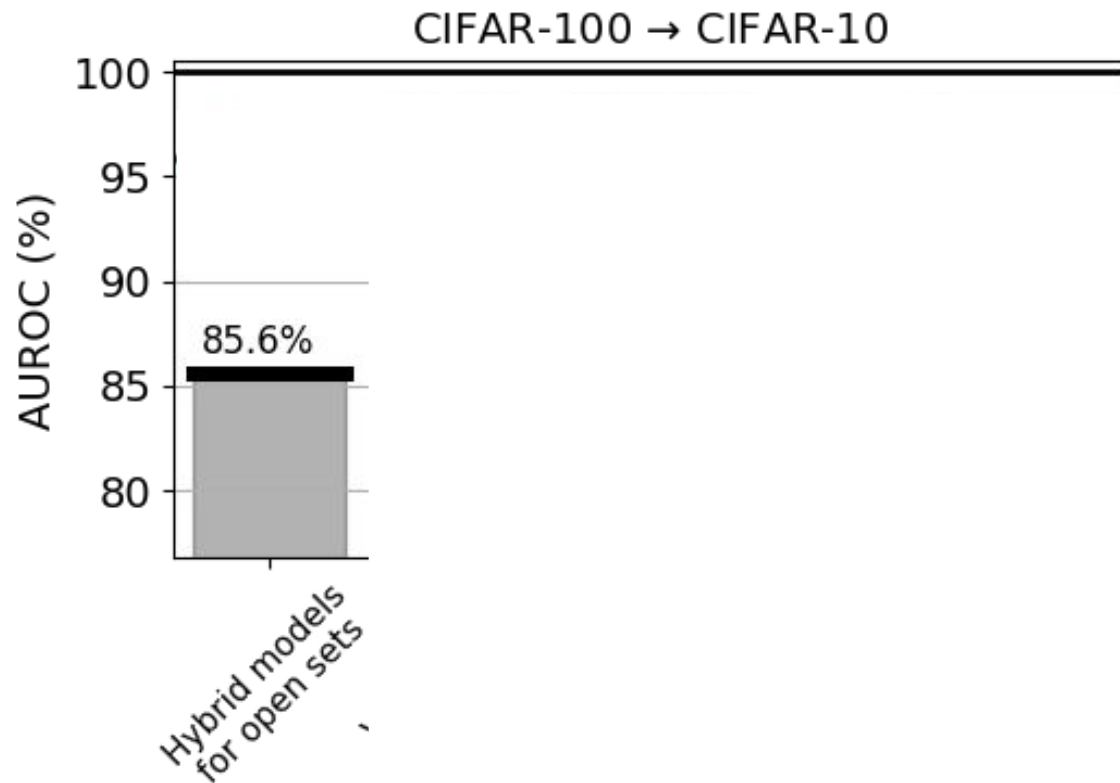
Benchmarking Human Performance (S.F. is the human)

Near-OOD human benchmarking on CIFAR-10 and CIFAR-100.

Improving near-OOD detection

CIFAR-100 → CIFAR-10

human
95.9%

85.6%

Hybrid models for open sets
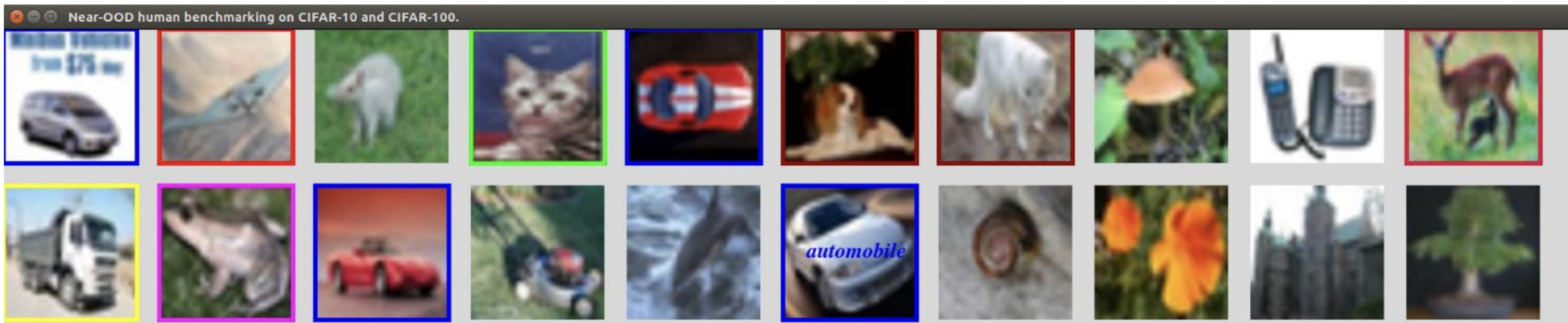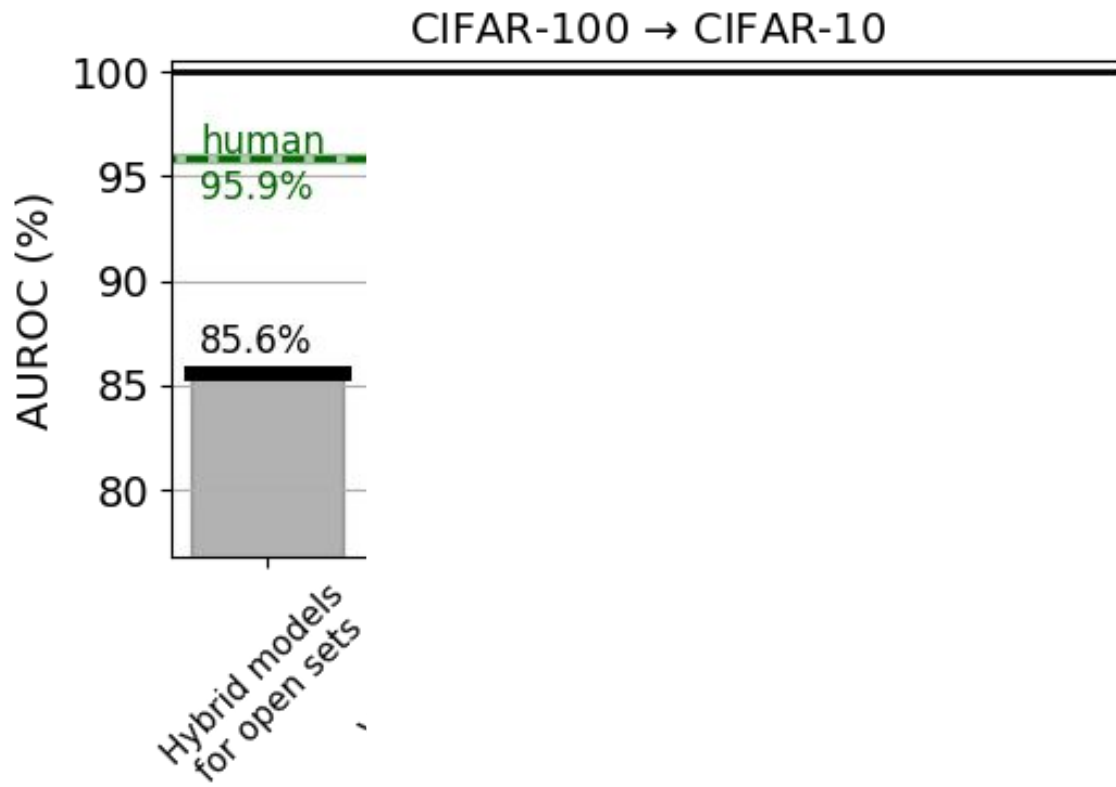
Pre-trained Vision Transformers improve near-OOD detection
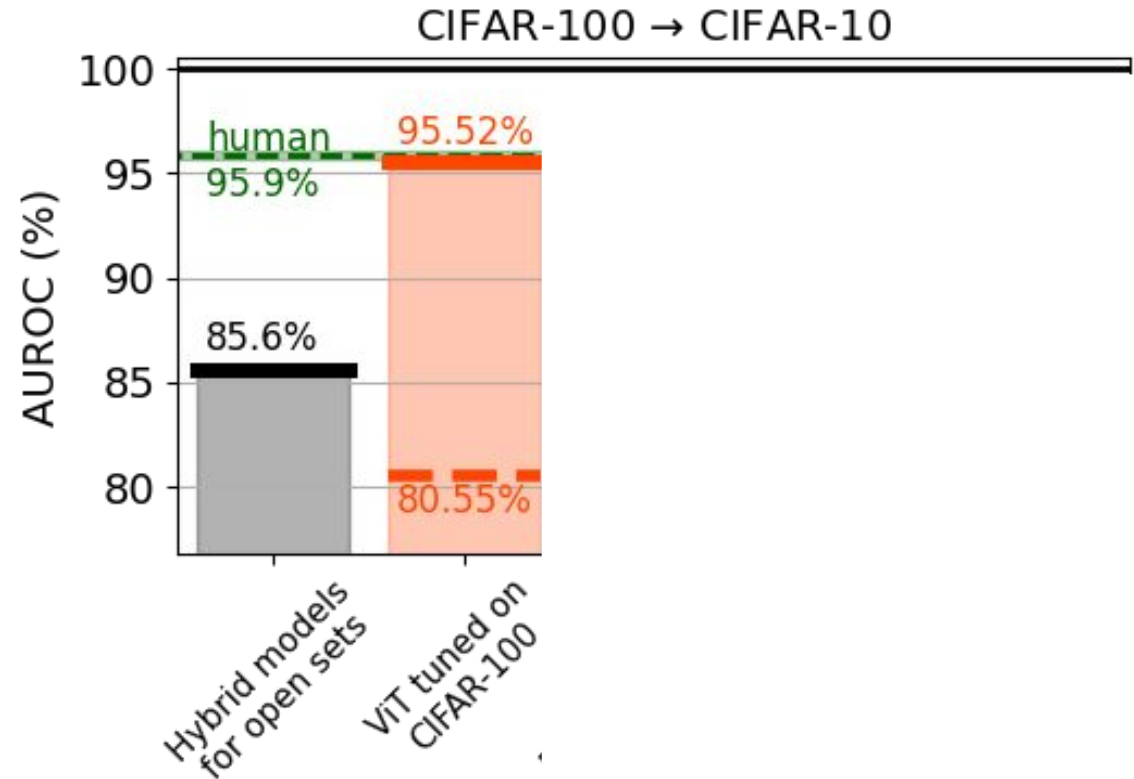
Pre-trained ViT improves near-OOD detection
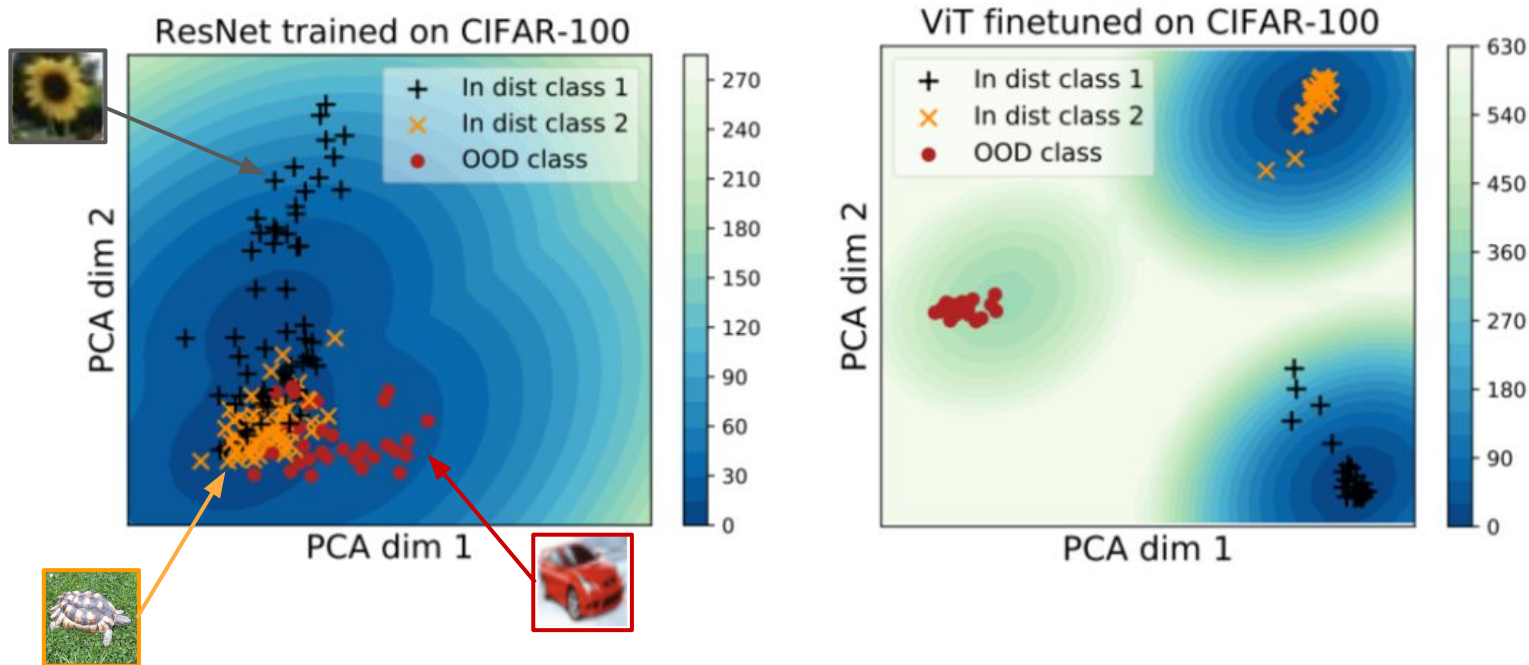
Figure: 2D PCA project of the space of embedding.
Color coding shows Mahalanobis outlier score.

Pre-trained Vision Transformers improve near-OOD detection
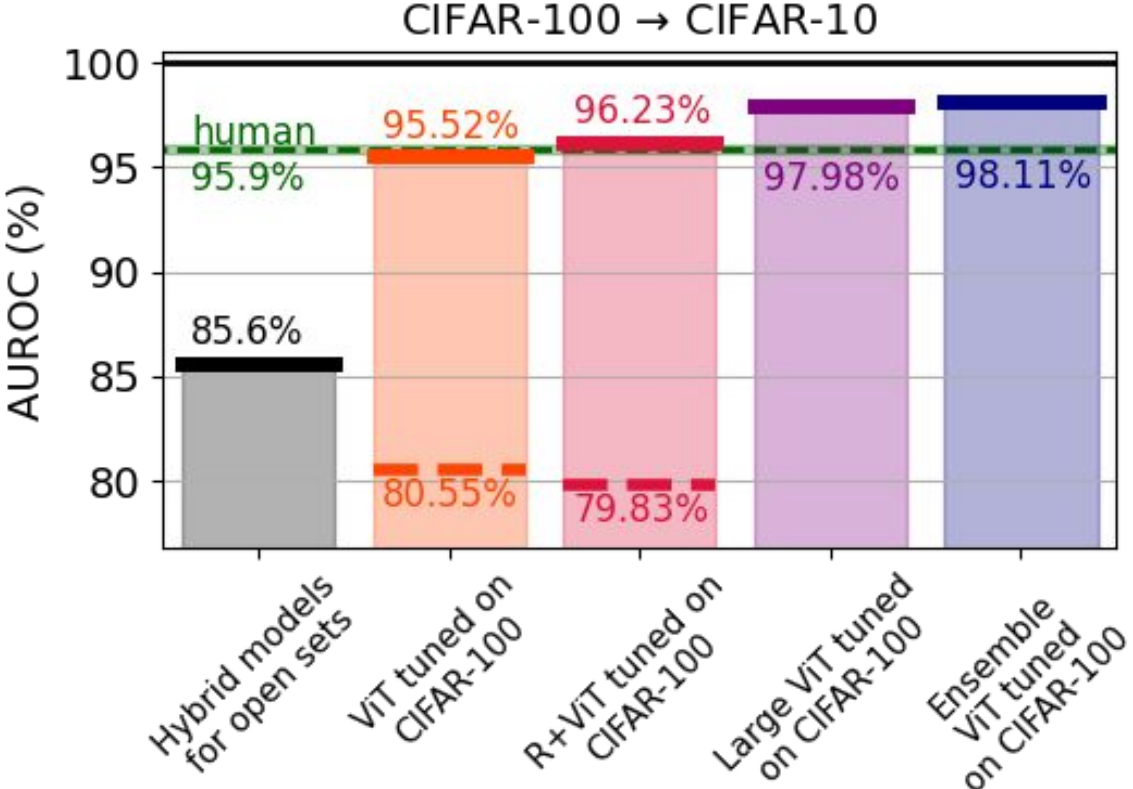
CIFAR-100 → CIFAR-10

Qualitative failure cases of ViT OOD detection

- Most false positives are due to mislabeling or ambiguity
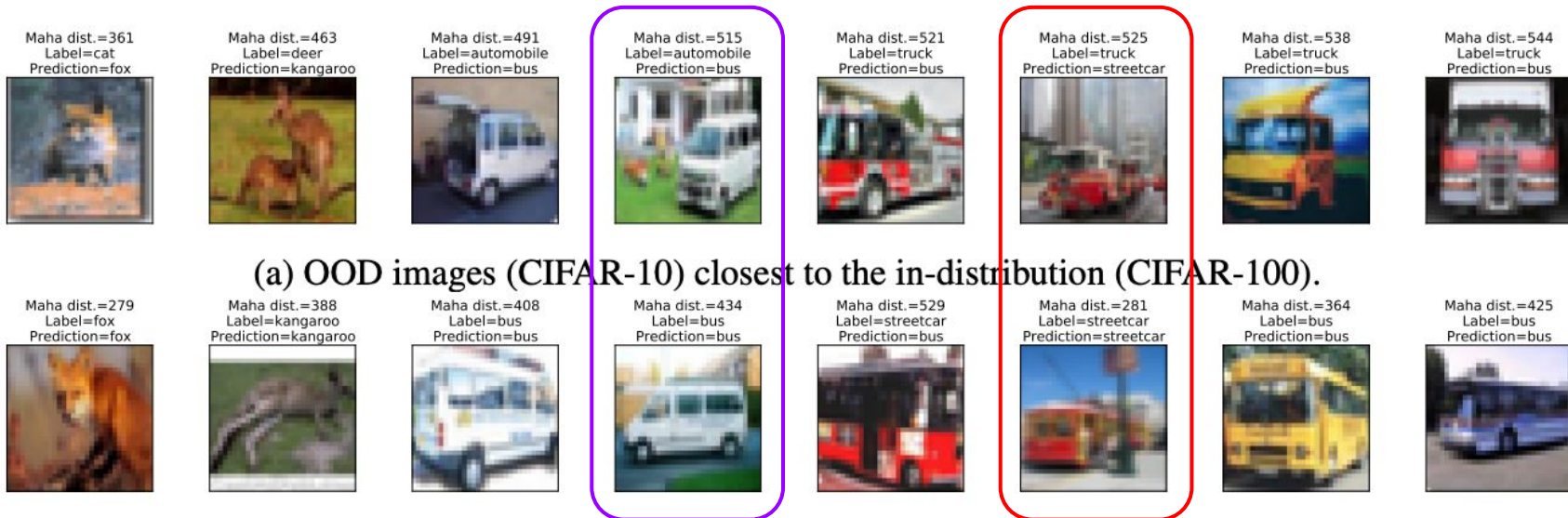
(a) OOD images (CIFAR-10) closest to the in-distribution (CIFAR-100).

(b) The in-distribution (CIFAR-100) images with the closest embedding vector to images in Figure 11a.

Qualitative failure cases of ViT OOD detection

- SVHN digits are classified as CIFAR-100 "worms"

Maha dist.=426
Label=2
Prediction=worm

Maha dist.=531
Label=2
Prediction=worm

Maha dist.=533
Label=5
Prediction=worm

Maha dist.=546
Label=5
Prediction=worm

Maha dist.=555
Label=6
Prediction=worm

Maha dist.=556
Label=5
Prediction=worm

Maha dist.=576
Label=5
Prediction=worm

Maha dist.=576
Label=3
Prediction=cloud
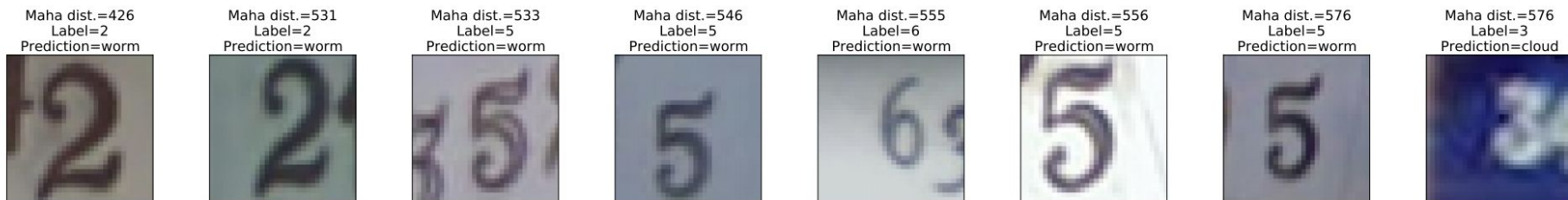
(a) OOD images (SVHN) closest to the in-distribution (CIFAR-100).
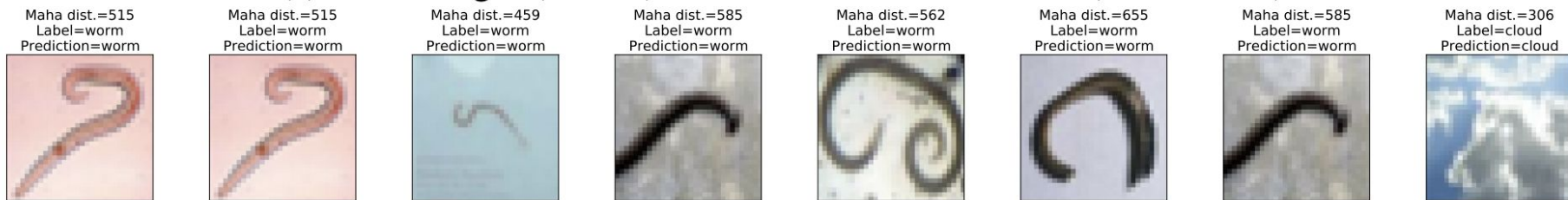
Maha dist.=515
Label=worm
Prediction=worm

Maha dist.=515
Label=worm
Prediction=worm

Maha dist.=459
Label=worm
Prediction=worm

Maha dist.=585
Label=worm
Prediction=worm

Maha dist.=562
Label=worm
Prediction=worm

Maha dist.=655
Label=worm
Prediction=worm

Maha dist.=585
Label=worm
Prediction=worm

Maha dist.=306
Label=cloud
Prediction=cloud

(b) The in-distribution (CIFAR-100) images with the closest embedding vector to images in Figure 11a.

The same story for genomics
(here pre-trained in a self-supervised way)
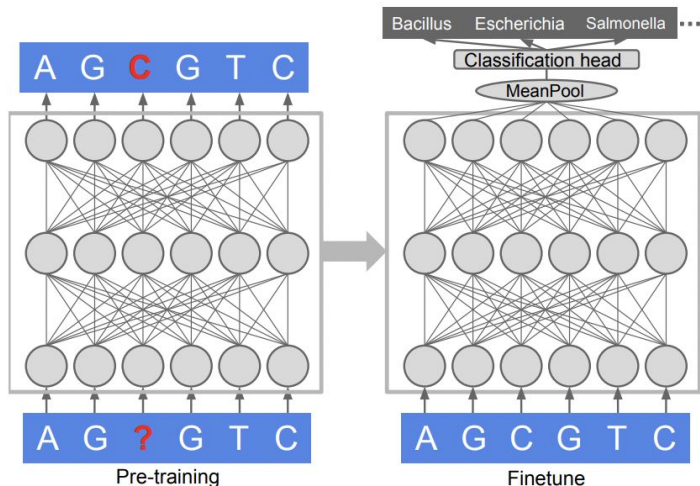
(a) BERT pre-training and fine-tuning for genomics.

Model	Test Accuracy	Mahalanobis AUROC	MSP AUROC
1D CNN [Ren et al., 2019]	85.93±0.11%	64.75±0.73%	65.84±0.46%
BERT pre-train and fine-tune	89.84±0.00%	**77.49±0.04%**	73.53±0.03%

Wrapping up

Uncertainty Baselines

github.com/google/uncertainty-baselines

High-quality implementations of baselines on a variety of tasks.

Ready for use: **7** settings, including:

- Wide ResNet 28-10 on CIFAR
- ResNet-50 and EfficientNet on ImageNet
- BERT on Clinc Intent Detection

14 different baseline methods.

Used across **10** projects at Google.

Collaboration with OATML @ Oxford, unifying

github.com/oatml/bdl-benchmarks.

		dustinvtran and edward-bot Retune VI baseline for CIFAR. ...		Latest commit 9379550 3 hours ago

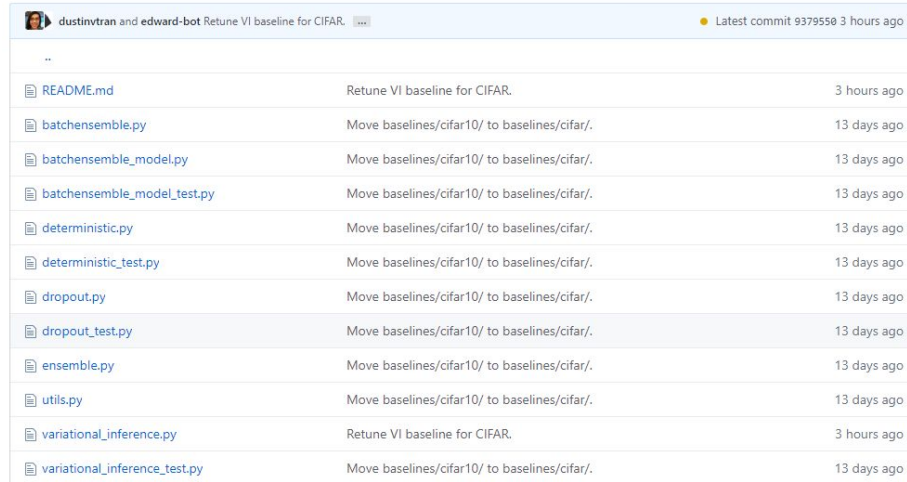|	README.md	Retune VI baseline for CIFAR.	3 hours ago
	batchensemble.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	batchensemble_model.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	batchensemble_model_test.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	deterministic.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	deterministic_test.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	dropout.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	dropout_test.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	ensemble.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	utils.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago
	variational_inference.py	Retune VI baseline for CIFAR.	3 hours ago
	variational_inference_test.py	Move baselines/cifar10/ to baselines/cifar/.	13 days ago

README.md

Wide ResNet 28-10 on CIFAR

CIFAR-10

Method	Train/Test NLL	Train/Test Accuracy	Train/Test Cal. Error	cNLL/cA/cCE	Train Runtime (hours)	# Parameters
Deterministic	1e-3 / 0.159	99.9% / 96.0%	1e-3 / 0.0231	1.29 / 69.8% / 0.173	1.2 (8 TPUv2 cores)	36.5M
BatchEnsemble (size=4)	0.08 / 0.143	99.9% / 96.2%	5e-5 / 0.0206	1.24 / 69.4% / 0.143	5.4 (8 TPUv2 cores)	36.6M
Dropout	2e-3 / 0.160	99.9% / 95.9%	2e-3 / 0.0241	1.35 / 67.8% / 0.178	1.2 (8 TPUv2 cores)	36.5M
Ensemble (size=4)	2e-3 / 0.114	99.9% / 96.6%	-	-	1.2 (32 TPUv2 cores)	146M
Variational inference	1e-3 / 0.211	99.9% / 94.7%	1e-3 / 0.029	1.46 / 71.3% / 0.181	5.5 (8 TPUv2 cores)	73M

Robustness Metrics

github.com/google-research/robustness_metrics

Lightweight modules to evaluate a model's
robustness and uncertainty predictions.

Ready for use:

- 10 OOD datasets
- Accuracy, uncertainty, and stability metrics
- Many SOTA models (TFHub support!)
- Multiple frameworks (JAX support!)

Enables large-scale studies of robustness
[Djolonga+ 2020].

Collaboration lead by Google Research, Brain Team @ Zurich.
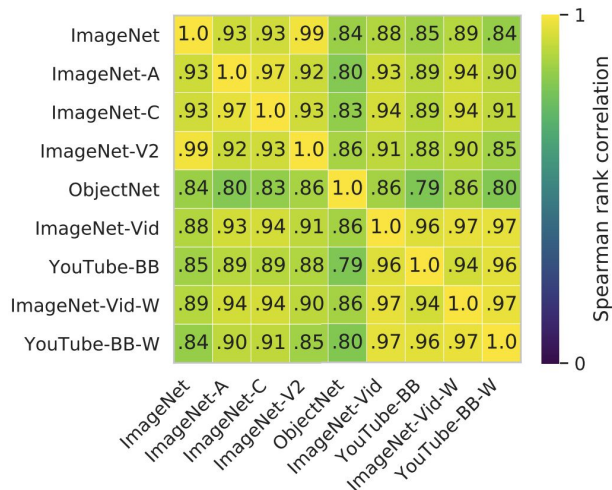
Takeaways

- Uncertainty & robustness are critical problems in AI and machine learning.

- Best performance achieved by composing orthogonal techniques
  - Single model uncertainty
  - Ensembling multiple neural networks
  - Imposing inductive biases on representations

- Open questions
  - Understand relationship between different types of OOD shifts
  - What other tools do we need in the "composable" toolkit?

- Links to papers available in my webpage: http://www.gatsby.ucl.ac.uk/~balaji/

Not covered in this talk: Unsupervised anomaly detection

- Caveat: Using density p(x) from deep generative model trained on $p_{TRAIN}(x)$ is not always reliable
- [Anomaly Detection using Deep Generative Models: Pitfalls and Promises](#) (overview talk that covers the following papers)

- **Do deep generative models know what they don't know?** E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, *B. Lakshminarayanan*. *[ICLR 2019](#)*.
- **Likelihood ratios for out-of-distribution detection**. J. Ren, P. Liu, E. Fertig, J. Snoek, R. Poplin, M. DePristo, J. Dillon, *B. Lakshminarayanan*. *[NeurIPS 2019](#)*.
    - *See also [A Simple Fix to Mahalanobis Distance for Improving Near-OOD Detection](#) where we extended this to density models of representations.*
- **Detecting out-of-distribution inputs to deep generative models using a test for typicality.** E. Nalisnick, A. Matsukawa, Y. W. Teh, *B. Lakshminarayanan*. *[arXiv 2019](#)*.
- **Density of States Estimation for Out-of-Distribution Detection** W. R. Morningstar, C. Ham, A. G. Gallagher, B. Lakshminarayanan, A. A. Alemi, J. V. Dillon [AISTATS 2021](#)
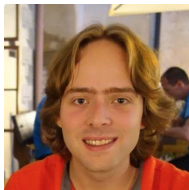
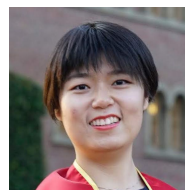# Thank you!

Dustin Tran

Jasper Snoek

Jeremiah Liu

Stanislav Fort
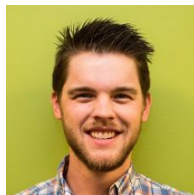
Clara Huiyi Hu

Jie Ren

Dan Hendrycks

Norman Mu

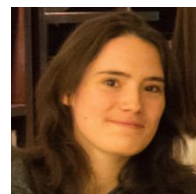Kevin Murphy

Mike Dusenberry

Rodolphe Jenatton

Shreyas Padhy
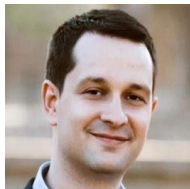
Mario Lucic

Zelda Mariet

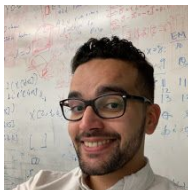Ben Adlam

Zack Nado

Eric Nalisnick
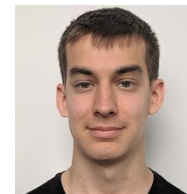
Josip Djolonga

Jeremy Nixon

Ghassen Jerfel

Justin Gilmer

D. Sculley

Yeming Wen

Marton Havasi

& others!

Appendix

References

*This list is intended just as a starting point for exploring other related work using Google Scholar or Connected papers.
Feel free to email me if you think there's a reference that should be included here.*

Survey papers

- **A Survey of Uncertainty in Deep Neural Networks**. J. Gawlikowski et al., *arXiv 2107.03342*.
- **A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges**. M. Abdar et al. *arXiv 2011.06225*

Bayesian neural networks

- **A practical Bayesian framework for backpropagation networks** D. MacKay Neural Computation 1992
- **Keeping Neural Networks Simple by Minimizing the Description Length of the Weights**. G. Hinton, D. Van Camp. COLT 1993.
- **An Introduction to Variational Methods for Graphical Models**. M. Jordan+. Machine Learning 1999.
- **Bayesian Learning for Neural Networks.** R. Neal. Technical Report 1994.
- **Bayesian Learning via Stochastic Gradient Langevin Dynamics**. M. Welling, Y. Teh. ICML 2011.
- **Weight Uncertainty in Neural Networks**. C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra. ICML 2015.
- **Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning** Y. Gal, Z. Ghahramani ICML 2016
- **Automatic Differentiation Variational Inference**. A. Kucukelbir, *D. Tran*, R. Ranganath, A. Gelman, D. M. Blei. JMLR 2017.
- **A Scalable Laplace Approximation for Neural Networks** H. Ritter, A. Botev, D. Barber ICLR 2018
- **Noise Contrastive Priors for Functional Uncertainty**. D. Hafner, *D. Tran*, T. Lillicrap, A. Irpan, J. Davidson. UAI 2019.
- **A Simple Baseline for Bayesian Uncertainty in Deep Learning** W. Maddox, T. Garipov, P. Izmailov, D. Vetrov, A. G. Wilson. NeurIPS 2019.
- **Practical Deep Learning with Bayesian Principles** K. Osawa, S. Swaroop, A.Jain, R. Eschenhagen, R. E. Turner, R. Yokota, M. E. Khan. NeurIPS 2019.
- **Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors**. M. W. Dusenberry, G. Jerfel, Y. Wen, Y. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, *D. Tran*. ICML 2020.

References

Ensembles

- **Simple and scalable predictive uncertainty estimation using deep ensembles** *B. Lakshminarayanan*, A. Pritzel, C. Blundell. NeurIPS 2017.
- **BatchEnsemble: An Alternative Approach to Efficient Ensemble and Lifelong Learning**. Y. Wen, *D. Tran*, J. Ba. ICLR 2020.
- **Why Aren't Bootstrapped Neural Networks Better?** J. Nixon, D. Tran and B. Lakshminarayanan ICBINB workshop @ NeurIPS 2020.
- **Training independent subnetworks for robust prediction**. M. Havasi, R. Jenatton, S. Fort, J. Z. Liu, J. Snoek, B. Lakshminarayanan, A. M. Dai, D. Tran ICLR 2021.

Understanding marginalization

- **Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data**. G. K. Dziugaite, D. M. Roy. UAI 2017.
- **Deep ensembles: A loss landscape perspective**. S. Fort, H. Hu, *B. Lakshminarayanan*. *arXiv 1912.02757*.
- **Bayesian Deep Learning and a Probabilistic Perspective of Generalization** A. G. Wilson and P. Izmailov arXiv 2002.08791

Gaussian processes and Neural Tangent Kernel

- **Deep Neural Networks as Gaussian Processes.** J. Lee, Y. Bahri, R. Novak, S. Schoenholz, J. Pennington, J. Sohl-Dickstein, ICLR 2018.
- **Neural Tangent Kernel: Convergence and Generalization in Neural Networks.** A. Jacot, F. Gabriel, C. Hongler. NeurIPS 2018.
- **Approximate Inference Turns Deep Networks into Gaussian Processes** M. Emtiyaz Khan, Alexander Immer, Ehsan Abedi, M. Korzepa NeurIPS 2019
- **Bayesian Deep Ensembles via the Neural Tangent Kernel** B. He, *B. Lakshminarayanan* and Y.W. Teh *NeurIPS 2020*
- **Exploring the Uncertainty Properties of Neural Networks' Implicit Priors in the Infinite-Width Limit**. B. Adlam, J. Lee, L. Xiao, J. Pennington and J. Snoek link

References

Practical guidance

- See the codebases! E.g. https://github.com/google/uncertainty-baselines
- **Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift**. Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D Sculley, S. Nowozin, J. Dillon, *B. Lakshminarayanan, J. Snoek*. NeurIPS 2019.
- **Simple, Distributed, & Accelerated Probabilistic Programming**. *D. Tran*, M. Hoffman, D. Moore, C. Suter, S. Vasudevan, A. Radul, M. Johnson, R. A. Saurous. NeurIPS 2018.
- **Bayesian Layers: A Module for Neural Network Uncertainty**. *D. Tran*, M. W. Dusenberry, M. van der Wilk, D. Hafner. NeurIPS 2019.

Data Augmentation and Invariances

- **Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty** *D. Hendrycks, M. Mazeika, S. Kadavath, D. Song NeurIPS 2019*
- **AugMix: A simple data processing method to improve robustness and uncertainty**. D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, *B. Lakshminarayanan. ICLR 2020*
- **Combining Ensembles and Data Augmentation can Harm your Calibration**. *Y. Wen, G. Jerfel, R. Muller, M. Dusenberry, J. Snoek, B. Lakshminarayanan and D. Tran. ICLR 2021*

Calibration

- **On calibration of modern neural networks** C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger *ICML 2017*
- **Revisiting the Calibration of Modern Neural Networks**. M. Minderer, J. Djolonga, R. Romijnders, F. Hubis, X. Zhai, N. Houlsby, D. Tran, M. Lucic *arXiv 2106.07998*.
- **Measuring Calibration in Deep Learning**. J. Nixon, M. Dusenberry, L. Zhang, G. Jerfel, *D. Tran*. arXiv 1904.01685

References

Proper Scoring Rules, Types of Uncertainty

- **Strictly Proper Scoring Rules, Prediction and Estimation**, *Gneiting & Raftery,* [JASA 2007](#)
- **What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?** *A. Kendall, Y. Gal* [NeurIPS 2017](#)
- **Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods** *E. Hüllermeier, W. Waegeman* [arXiv 1910.09457](#)

Deep Generative Models and Hybrid models

- **Hybrid models with deep and invertible features** E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, *B. Lakshminarayanan*. [ICML](#) [2019](#).
- **Do deep generative models know what they don't know?** E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, *B. Lakshminarayanan*. [ICLR 2019](#).
- **Likelihood ratios for out-of-distribution detection**. J. Ren, P. Liu, E. Fertig, J. Snoek, R. Poplin, M. DePristo, J. Dillon, *B. Lakshminarayanan*. [NeurIPS 2019](#).
- **Detecting out-of-distribution inputs to deep generative models using a test for typicality.** E. Nalisnick, A. Matsukawa, Y. W. Teh, *B. Lakshminarayanan*. [arXiv 1906.02994](#).
- **Density of States Estimation for Out-of-Distribution Detection** W. R. Morningstar, C. Ham, A. G. Gallagher, B. Lakshminarayanan, A. A. Alemi, J. V. Dillon [AISTATS 2021](#)

Detecting Out-of-Distribution Inputs

- **A Baseline for Detecting Misclassified & Out-of-Distribution Examples in Neural Networks** D. Hendrycks, K. Gimpel [ICLR 2017](#)
- **A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks** K. Lee, K. Lee, H. Lee, J. Shin [NeurIPS 2018](#)
- **Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks** S. Liang, Y. Li, R. Srikant [ICLR 2018](#)
- **Deep Anomaly Detection with Outlier Exposure** D. Hendrycks, M. Mazeika, T. Dietterich [ICLR 2019](#)
- **Exploring the Limits of Out-of-Distribution Detection**. S. Fort, J. Ren, B. Lakshminarayanan [NeurIPS 2021](#).