Introduction to Uncertainty in Deep Learning

Balaji Lakshminarayanan
balajiln@

Based on NeurIPS tutorial with Dustin Tran & Jasper Snoek

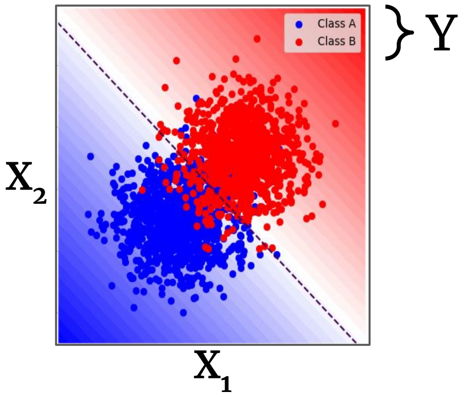
Motivation

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***.



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

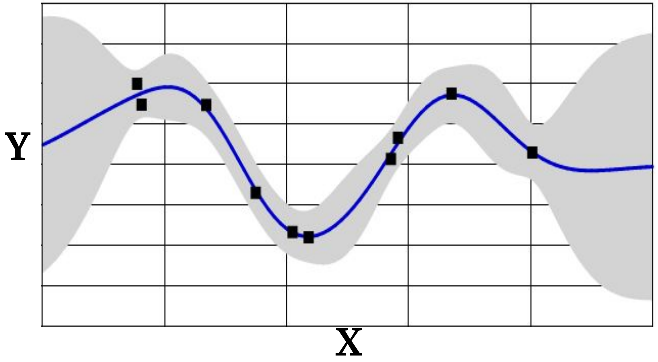


Image credit: Eric Nalisnick

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

Google AI
Brain Team

I.I.D. $p_{TEST}(y,x) = p_{TRAIN}(y,x)$

(Independent and Identically Distributed)

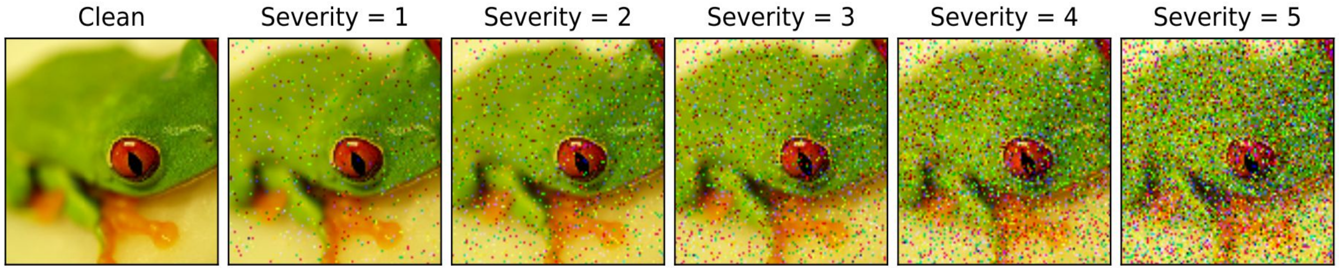
O.O.D. $p_{TEST}(y,x) \neq p_{TRAIN}(y,x)$

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

Google AI
Brain Team



Increasing “OODness”

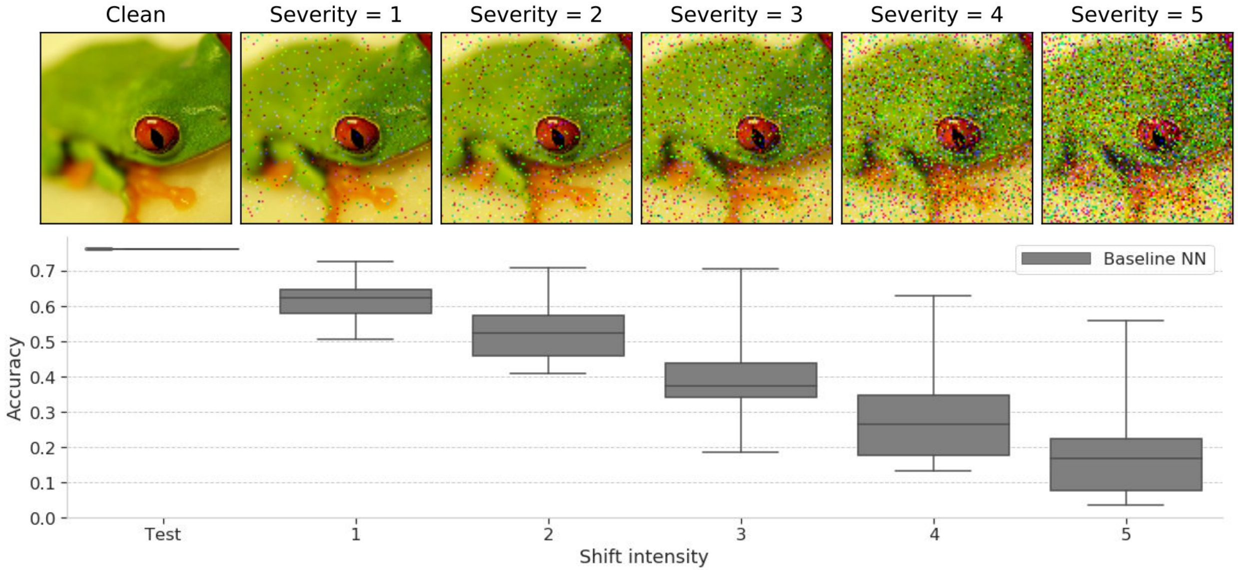
I.I.D test set

“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](https://arxiv.org/abs/1903.12261).

Neural networks do not generalize under covariate shift

Google AI Brain Team



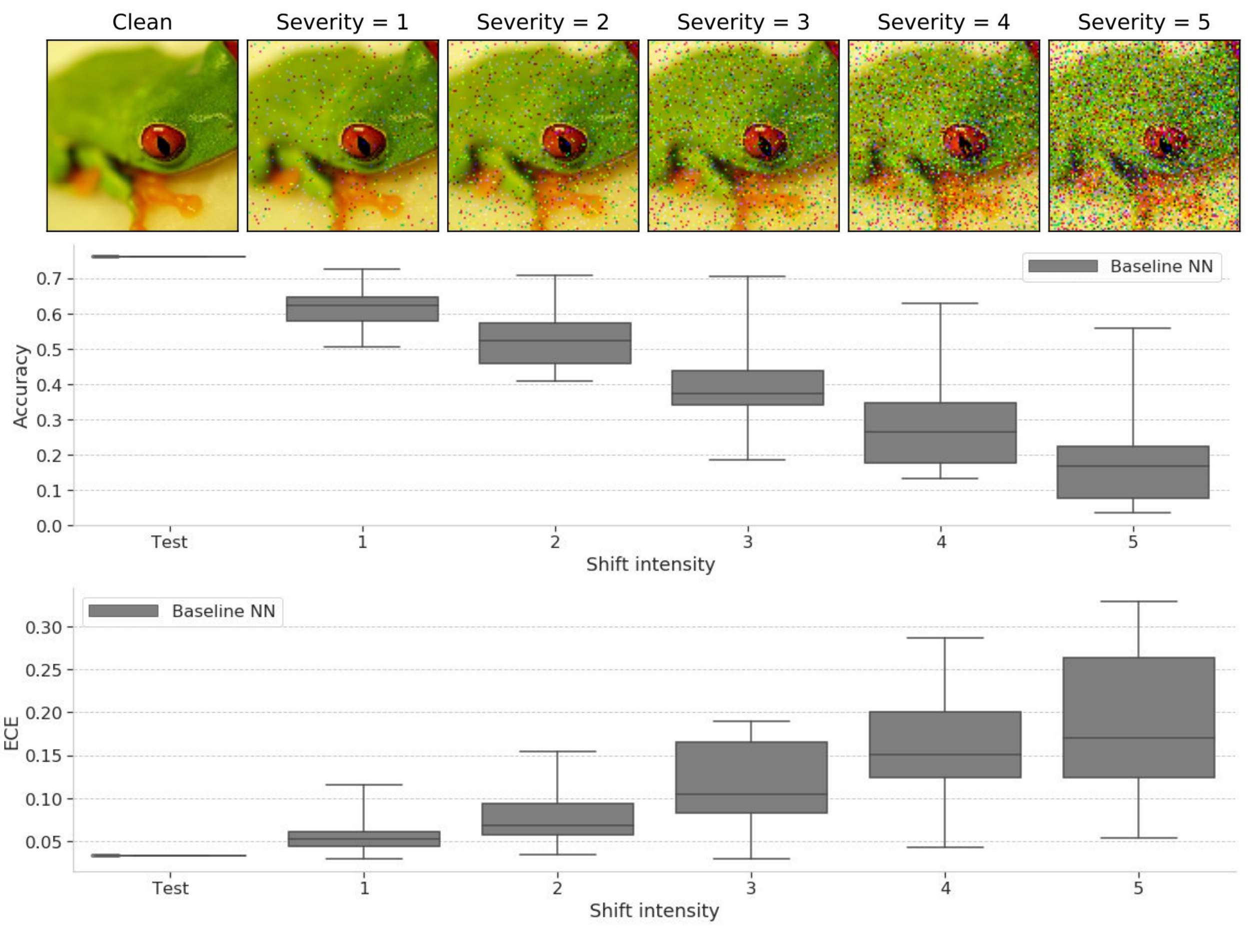
- **Accuracy drops** with increasing shift on Imagenet-C

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

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

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

Google AI
Brain Team

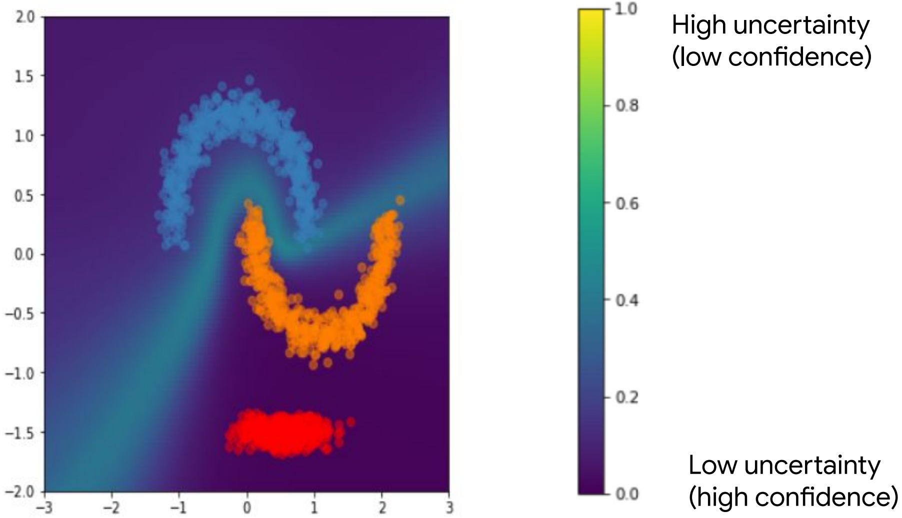


- **Accuracy drops** with increasing shift on Imagenet-C

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

Models assign high confidence predictions to OOD inputs

Google AI
Brain Team

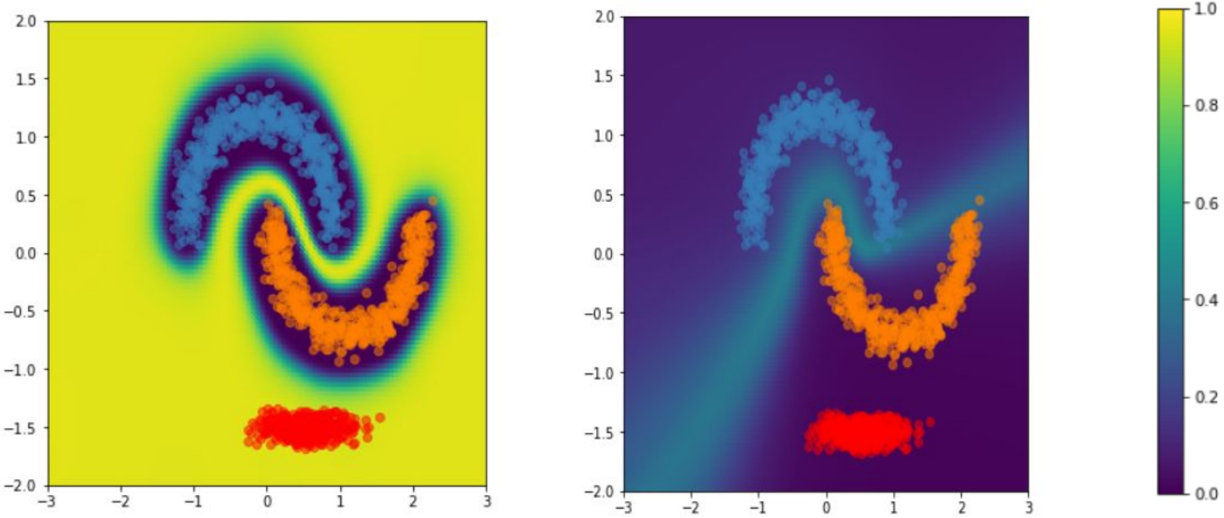


Deep neural networks

Image source: “Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness” [Liu et al. 2020](#)

Models assign high confidence predictions to OOD inputs

Google AI
Brain Team



High uncertainty
(low confidence)

Low uncertainty
(high confidence)

Ideal behavior

Deep neural networks

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

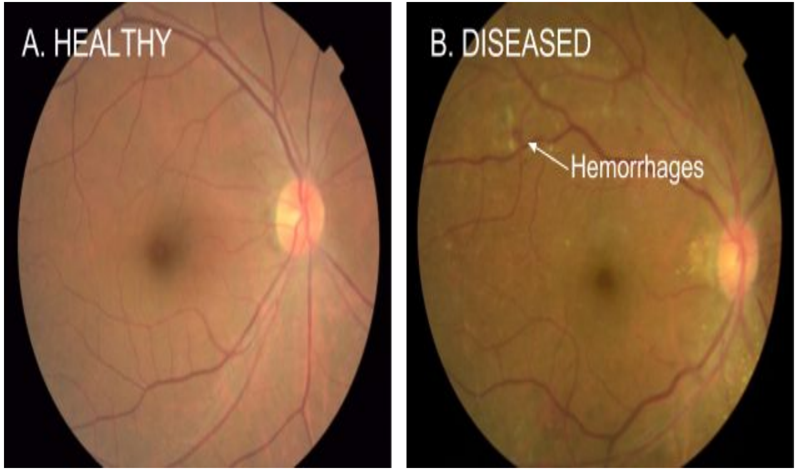
Image source: "Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness" [Liu et al. 2020](#)

Applications

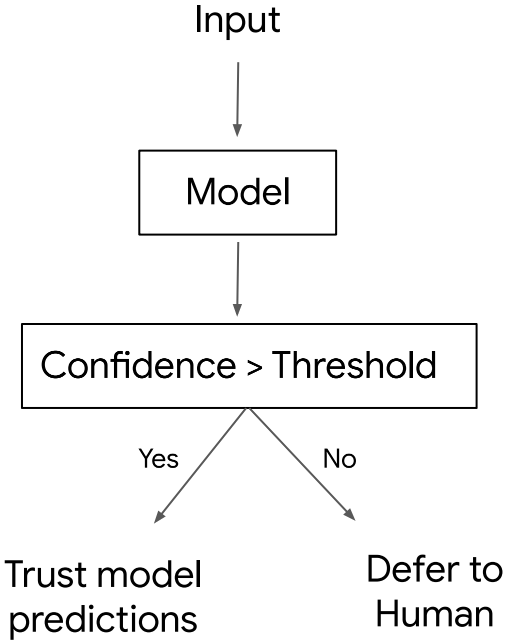
Healthcare

Google AI Brain Team

- Use model uncertainty to decide when to trust the model or to defer to a human.
- Cost-sensitive decision making

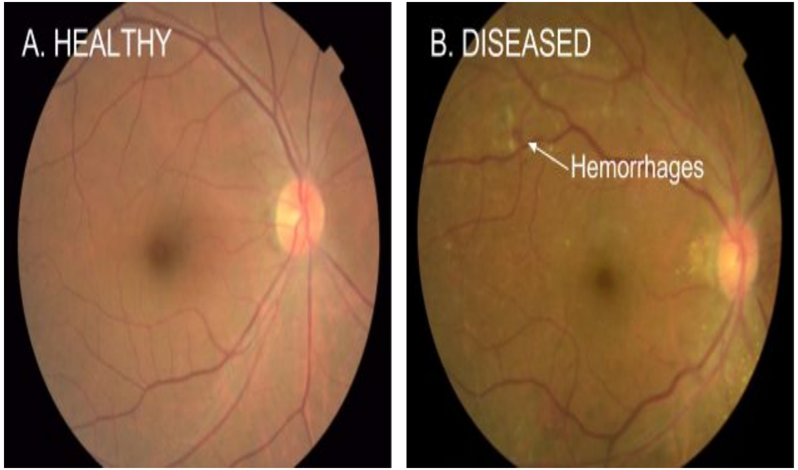


Diabetic retinopathy detection from fundus images
[Gulshan et al, 2016]



Healthcare

- Use model uncertainty to decide when to trust the model or to defer to a human.
- Cost-sensitive decision making



Diabetic retinopathy detection from fundus images
[Gulshan et al, 2016]

True label

Action		Healthy	Diseased
	Predict Healthy	0	10
	Predict Diseased	1	0
	Abstain "I don't know"	0.5	0.5

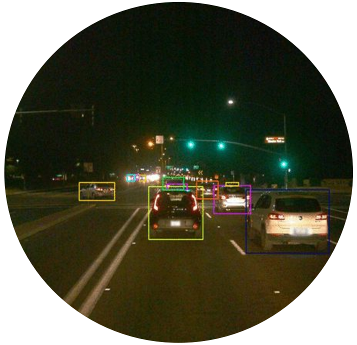
Self-driving cars

Google AI
Brain Team

Dataset shift:

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

Daylight



Night

Weather

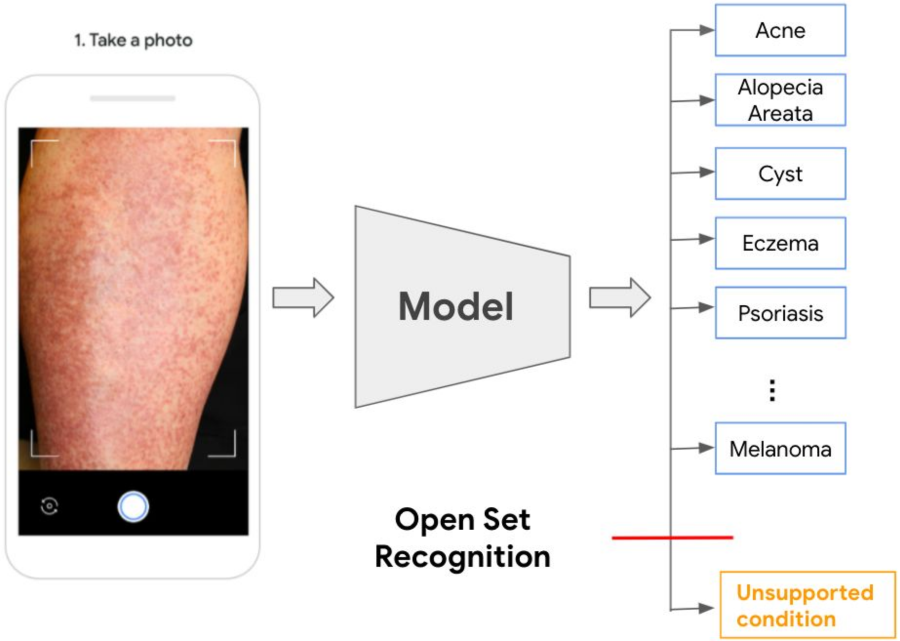
Construction

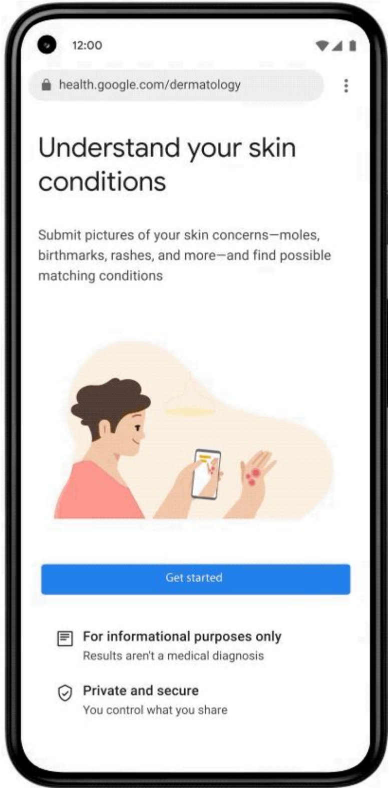
Downtown

Suburban

Image credit: Sun et al, [Waymo Open Dataset]

Open Set Recognition





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

Need to be able to say "none-of-the-above".

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

Open Set Recognition

- Example: Classification of genomic sequences

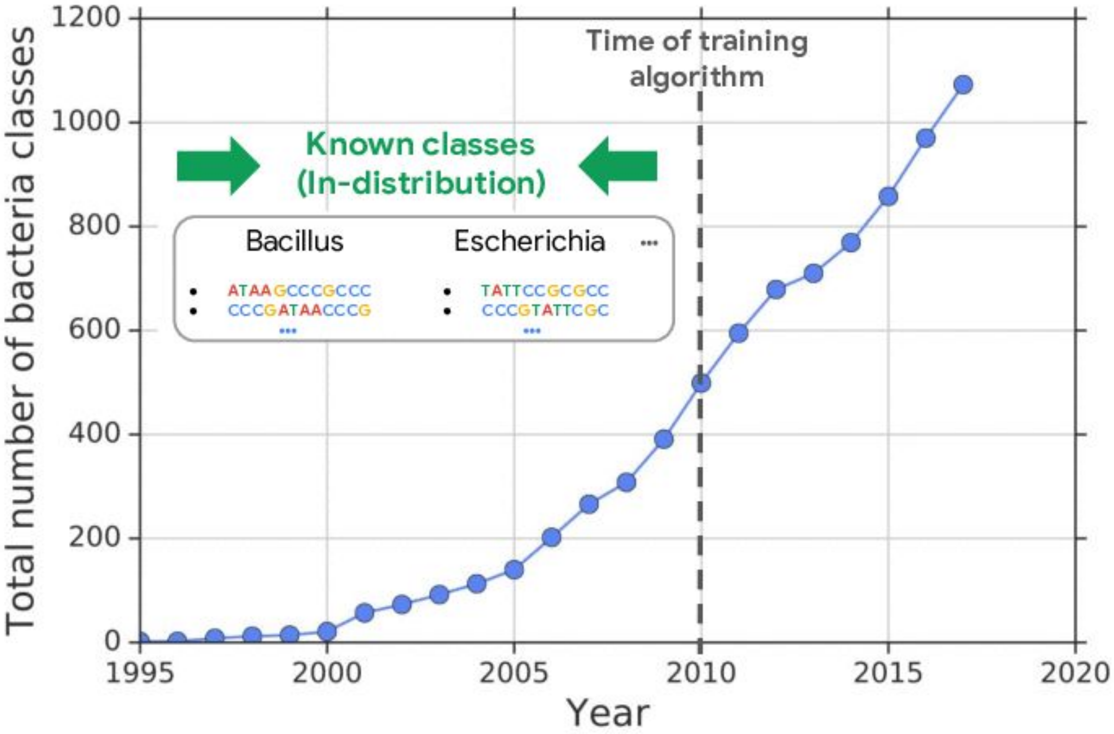


Image source: https://ai.googleblog.com/2019/12/improving-out-of-distribution-detection.html

Open Set Recognition

Google AI
Brain Team

- Example: Classification of genomic sequences
- High i.i.d. accuracy on known classes is not sufficient
- Need to be able to detect inputs that do not belong to one of the known classes

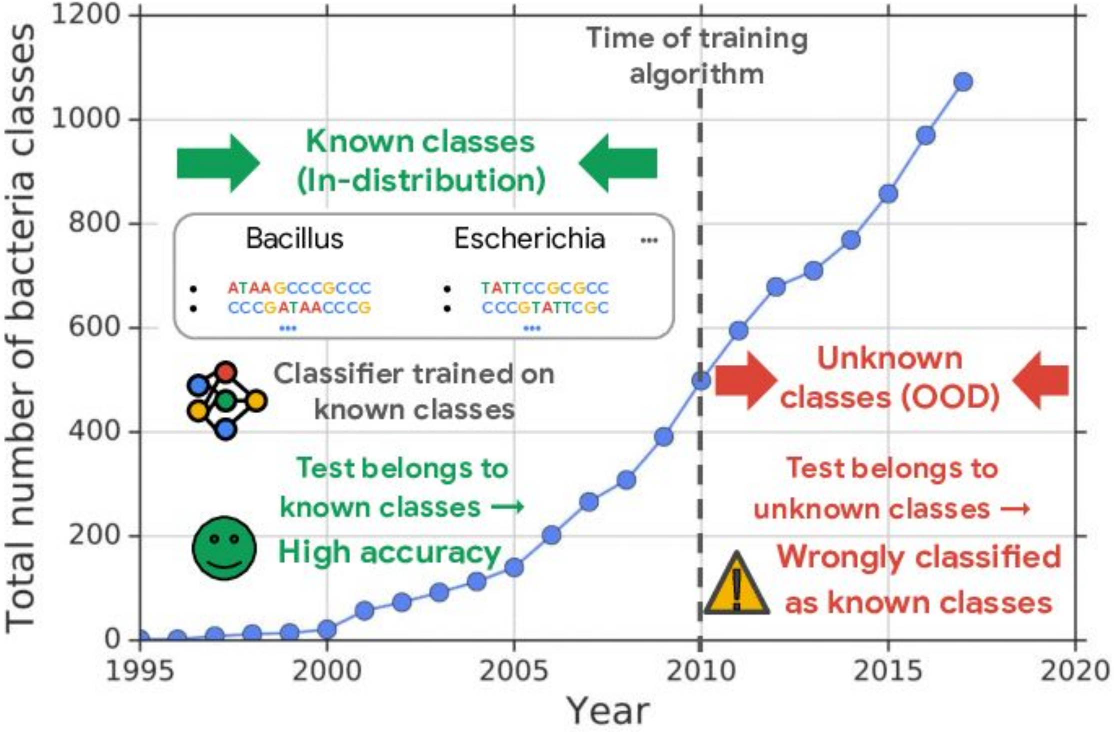


Image source: https://ai.googleblog.com/2019/12/improving-out-of-distribution-detection.html

Active Learning

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

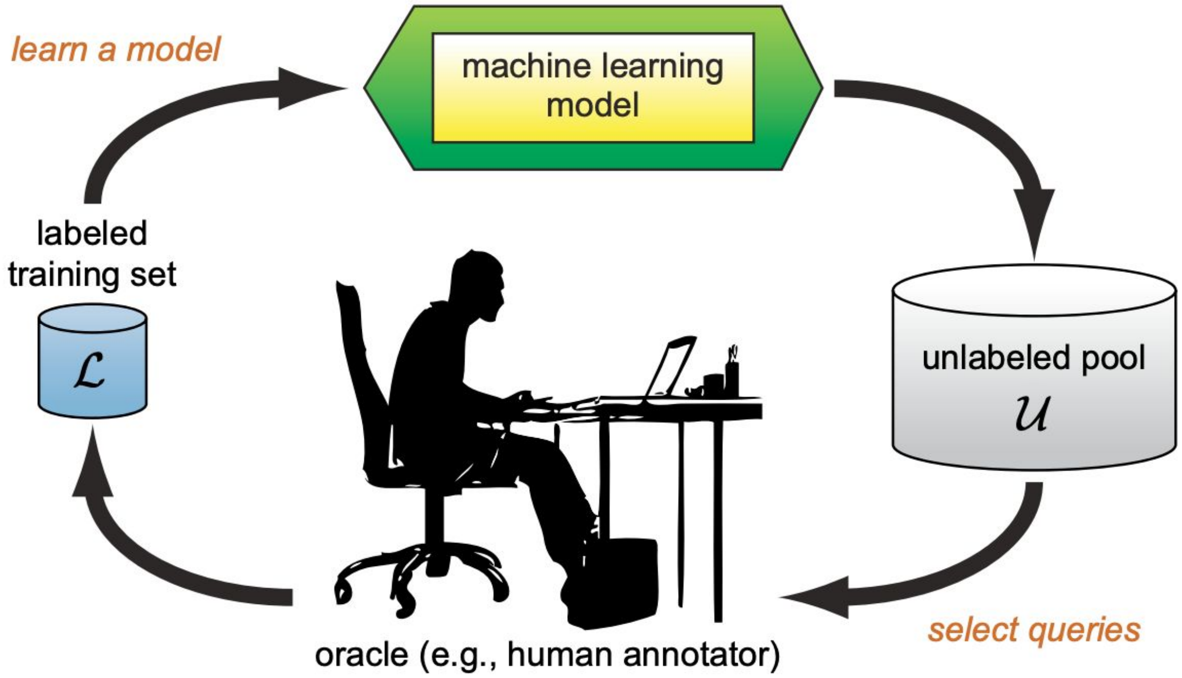


Image source: Active Learning Literature Survey, [Settles 2010]

Bayesian Optimization and Experimental Design

Google AI Brain Team

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

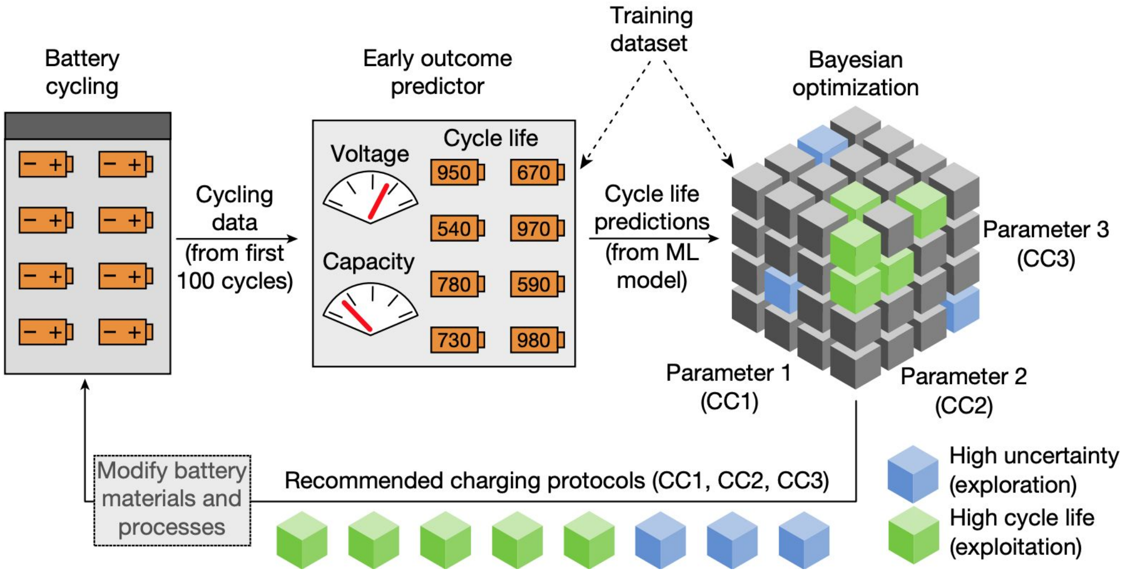


Image source: Attia et al. 2020 Closed-loop optimization of fast-charging protocols for batteries with machine learning

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

- Non-stationarity

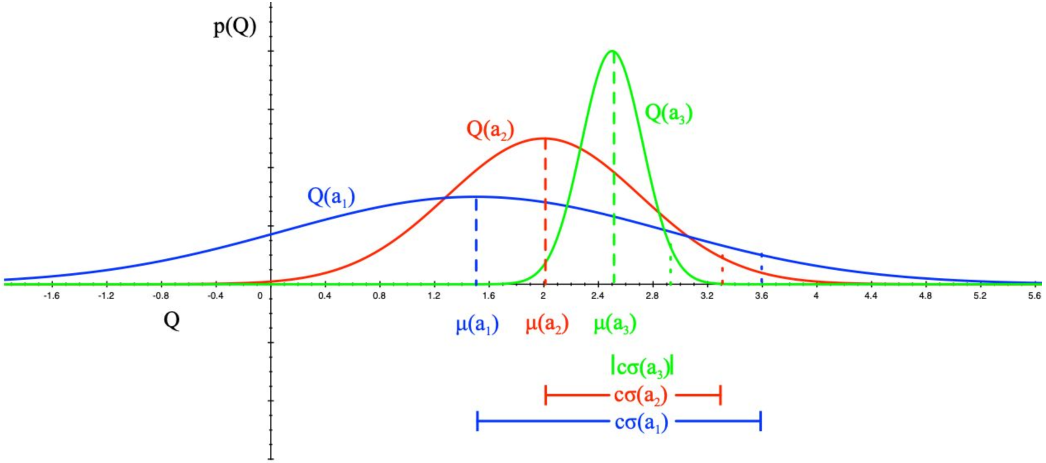
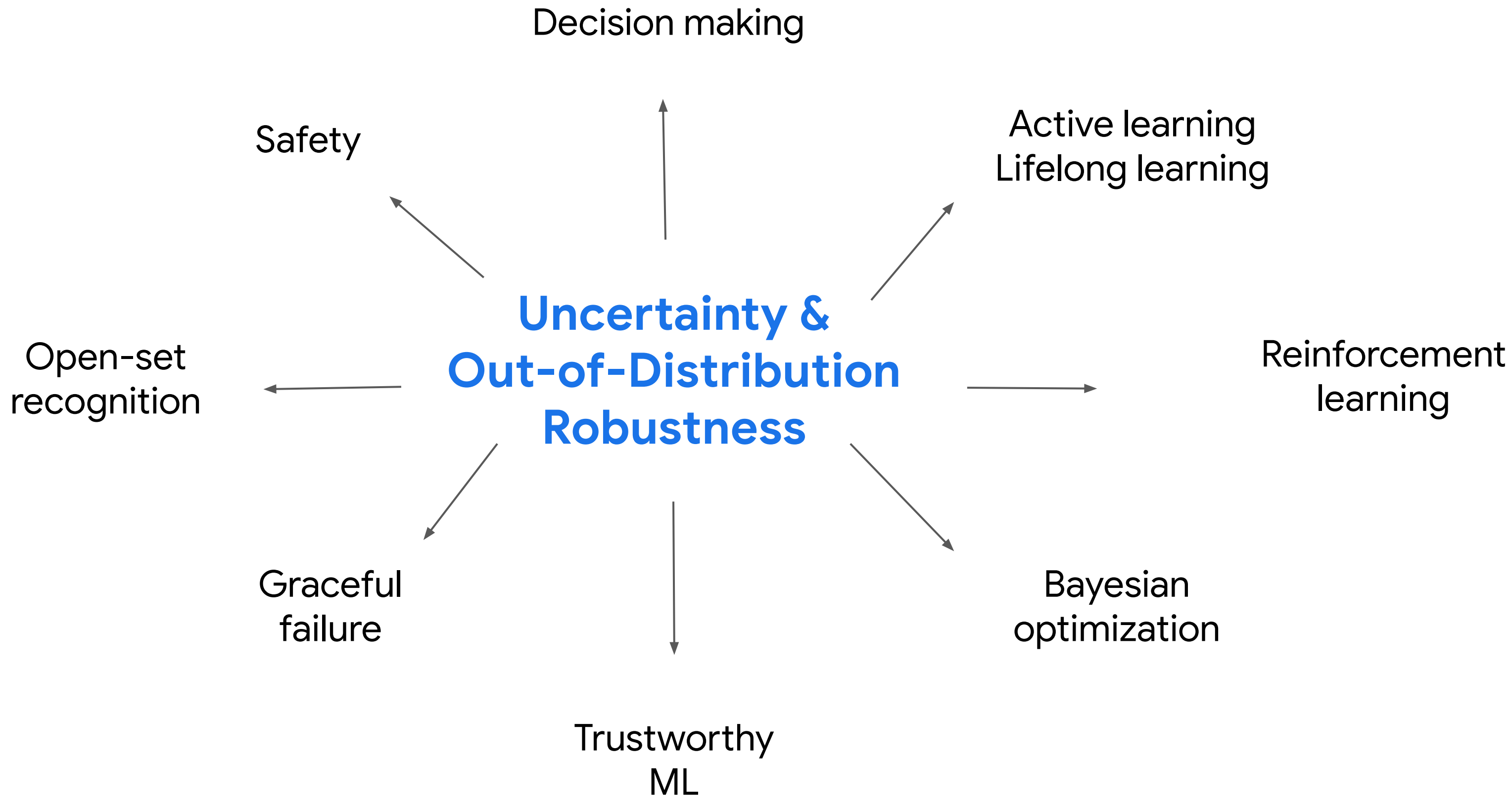


Image source: David Silver's [RL course]

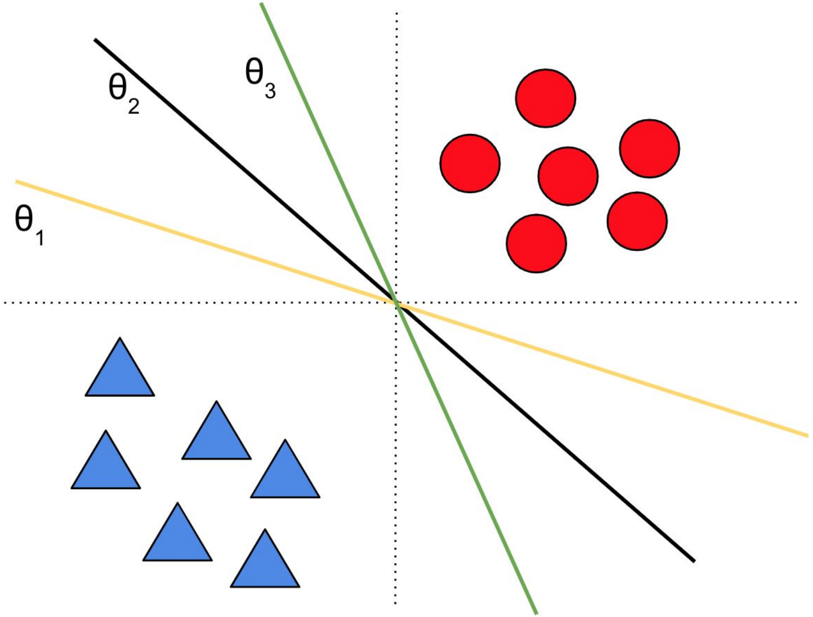
All models are wrong, but ~~some~~ models that know when they are wrong, are useful.



Primer on Uncertainty & Robustness

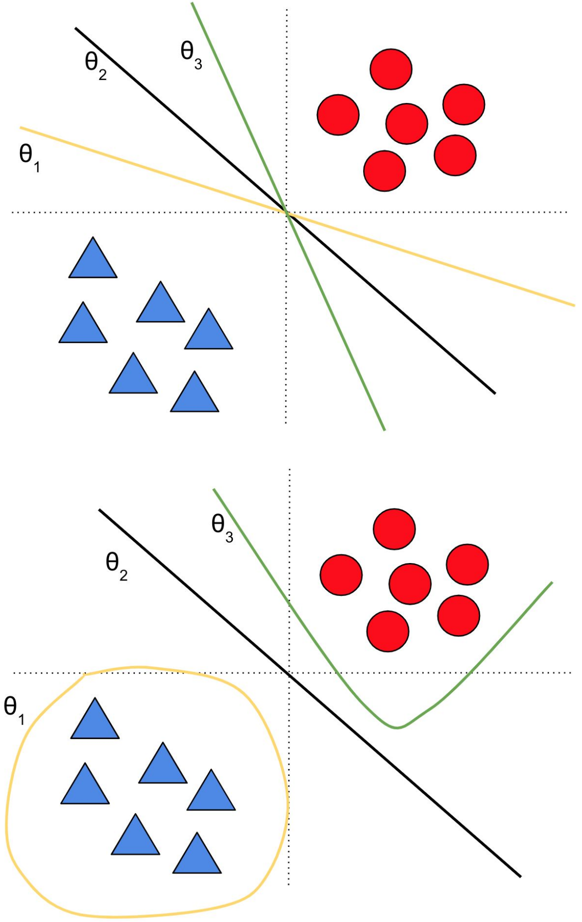
Sources of uncertainty: *Model uncertainty*

- Many models can fit the training data well
- Also known as ***epistemic uncertainty***
- Model uncertainty is "**reducible**"
    - Vanishes in the limit of infinite data (subject to model identifiability)



Sources of uncertainty: *Model uncertainty*

- Many models can fit the training data well
- Also known as ***epistemic uncertainty***
- Model uncertainty is “**reducible**”
  - Vanishes in the limit of infinite data (subject to model identifiability)
- Models can be from same hypotheses class (e.g. linear classifiers in top figure) or belong to different hypotheses classes (bottom figure).



Sources of uncertainty: *Data uncertainty*

- Labeling noise (ex: human disagreement)

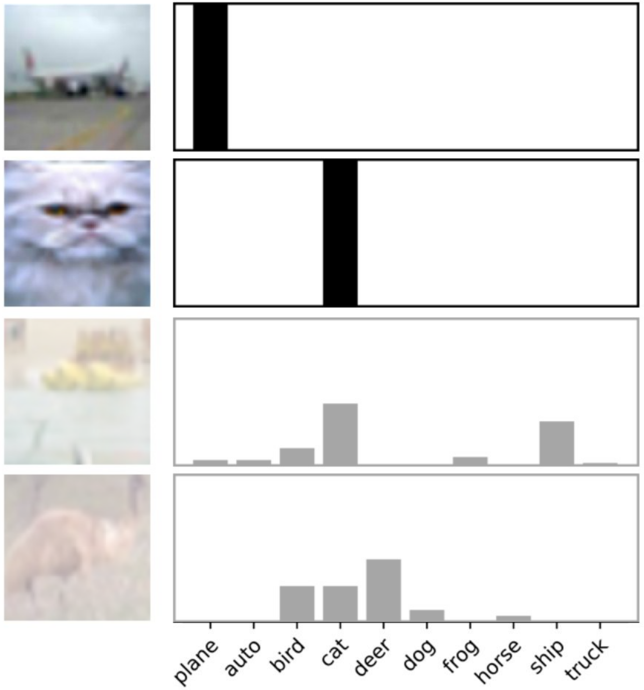


Image source: [Battleday et al. 2019](#) "Improving machine classification using human uncertainty measurements"

Sources of uncertainty: *Data uncertainty*

- Labeling noise (ex: human disagreement)

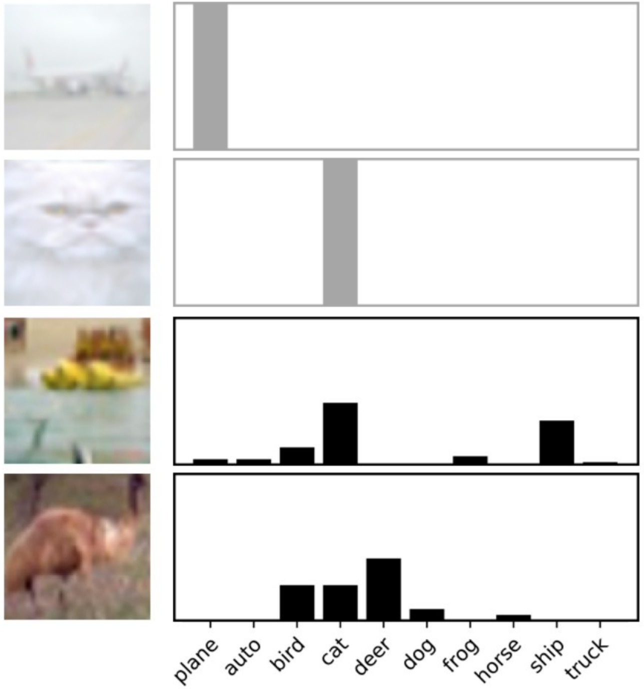


Image source: [Battleday et al. 2019](#) "Improving machine classification using human uncertainty measurements"

Sources of uncertainty: *Data uncertainty*

Google AI Brain Team

- Labeling noise (ex: human disagreement)
- Measurement noise (ex: imprecise tools)
- *Missing* data (ex: partially observed features, unobserved confounders)
- Also known as ***aleatoric uncertainty***
- Data uncertainty is "**irreducible***"
  - Persists even in the limit of infinite data
  - *Could be reduced with additional features/views

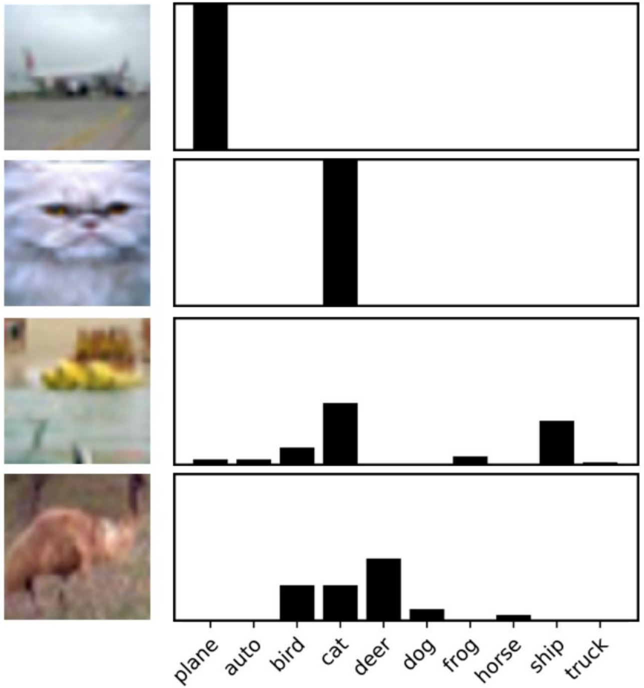


Image source: [Battleday et al. 2019]() "Improving machine classification using human uncertainty measurements"

How do we measure the quality of uncertainty?

Calibration Error = |Confidence - Accuracy|

predicted probability of correctness

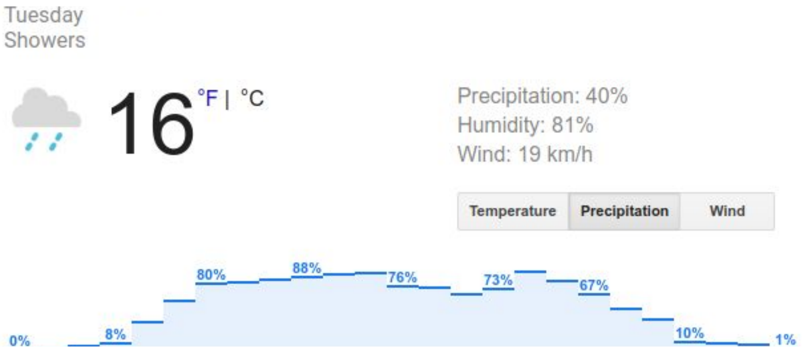
observed frequency of correctness

How do we measure the quality of uncertainty?

Calibration Error = |Confidence - Accuracy|

Of all the days where the model predicted rain with 80% probability, what fraction did we observe rain?

- 80% implies perfect calibration
- Less than 80% implies model is overconfident
- Greater than 80% implies model is under-confident

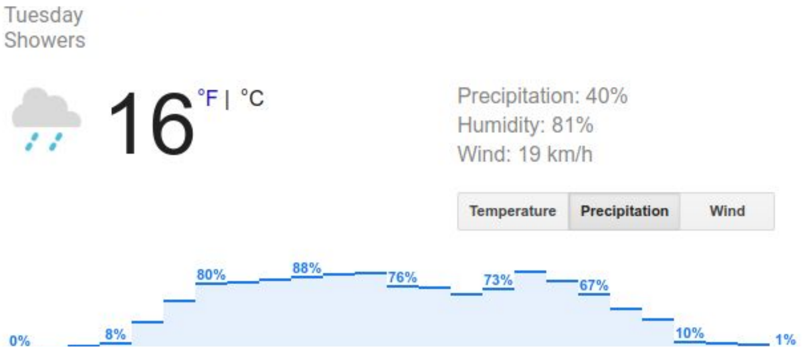


How do we measure the quality of uncertainty?

Calibration Error = |Confidence - Accuracy|

Of all the days where the model predicted rain with 80% probability, what fraction did we observe rain?

- 80% implies perfect calibration
- Less than 80% implies model is overconfident
- Greater than 80% implies model is under-confident



Intuition: For regression, calibration corresponds to coverage in a confidence interval.

How do we measure the quality of uncertainty?

Expected Calibration Error [Naeini+ 2015]:

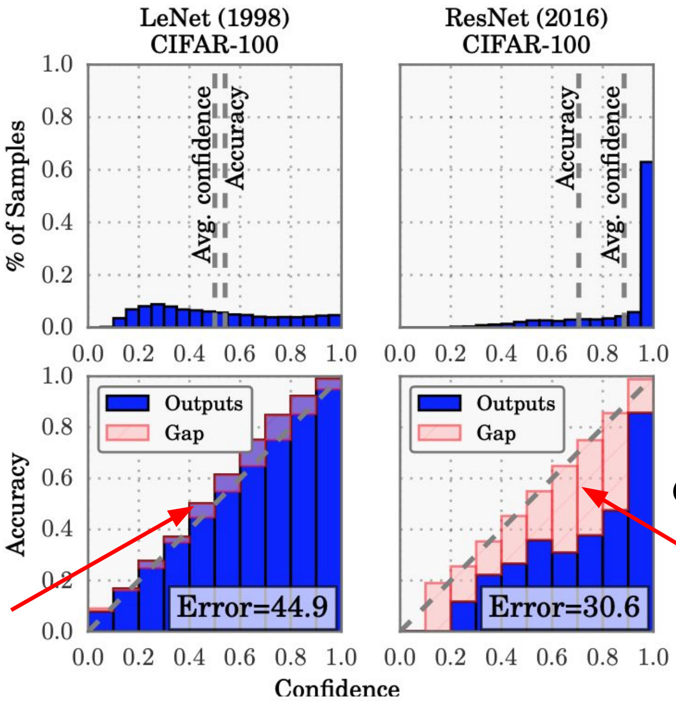
$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} \left|\mathrm{acc}(b) - \mathrm{conf}(b)\right|$$

- Bin the probabilities into B bins.
- Compute the within-bin accuracy and within-bin predicted confidence.
- Average the calibration error across bins (weighted by number of points in each bin).

How do we measure the quality of uncertainty?

Expected Calibration Error [Naeini+ 2015]:

$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} \left|\mathrm{acc}(b) - \mathrm{conf}(b)\right|$$



Confidence < Accuracy

=> Underconfident

Confidence > Accuracy

=> Overconfident

Image source: Guo+ 2017 “On calibration of modern neural networks”

How do we measure the quality of uncertainty?

Expected Calibration Error [Naeini+ 2015]:

$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} \left|\mathrm{acc}(b) - \mathrm{conf}(b)\right|$$

Note: Does **not** reflect **accuracy**.

Predicting class frequency p(y=1) = 0.3 for all the inputs achieves perfect calibration.

True label	0	0	0	0	0	0	0	1	1	1	Accurate?	Calibrated?
Model prediction	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	❌	✅

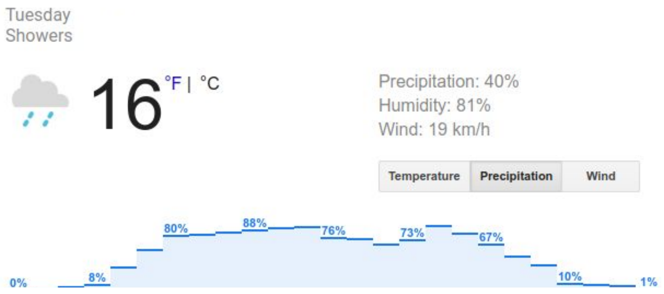
How do we measure the quality of uncertainty?

Proper scoring rules [Gneiting & Raftery 2007]

- Negative Log-Likelihood (NLL)
  - Also known as *cross-entropy*
  - Can overemphasize tail probabilities
- Brier Score
  - Quadratic penalty (bounded range [0,1] unlike log).

$$\mathrm{BS} = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \left[p(y|\mathbf{x}_n, \theta) - \delta(y - y_n) \right]^2$$

  - Can be numerically unstable to optimize.

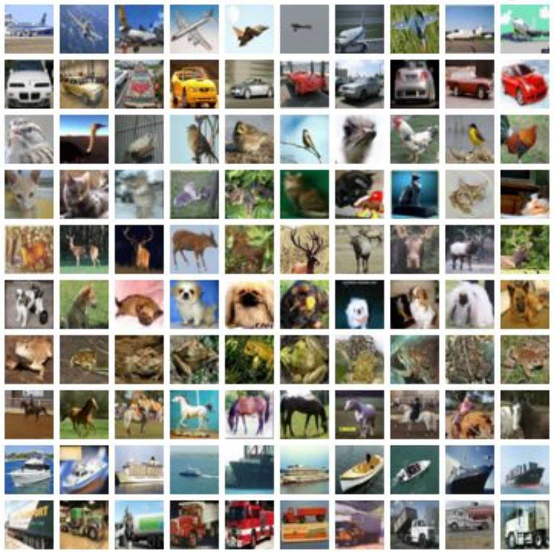


How do we measure the quality of uncertainty?

Google AI
Brain Team

Evaluate model on **out-of-distribution (OOD) inputs** which do not belong to any of the existing classes

- Max confidence
- Entropy of p(y|x)



CIFAR-10 (IID test inputs)

SVHN (OOD test inputs)

CIFAR-10 classifier

Confidence on IID inputs > Confidence on OOD inputs ?

Downstream cost (unifying accuracy & OOD detection)

Cost		Action of the Model	
		Prediction as Inlier	Abstain
Ground Truth	Inlier	0.0 (Correct) 1.0 (Incorrect, mistake that erodes trust)	0.5 (Incorrect, abstaining inliers)
	Outlier	1.0 (Incorrect, mistake that erodes trust)	0.0 (Correct)

Incorrect inlier predictions

Abstaining is better than incorrect prediction, but it's still worse than correctly predicting

mistake that erodes trust

Image source: Roy*, Ren* et al. 2021 "Does Your Dermatology Classifier Know What It Doesn't Know? Detecting the Long-Tail of Unseen Conditions"

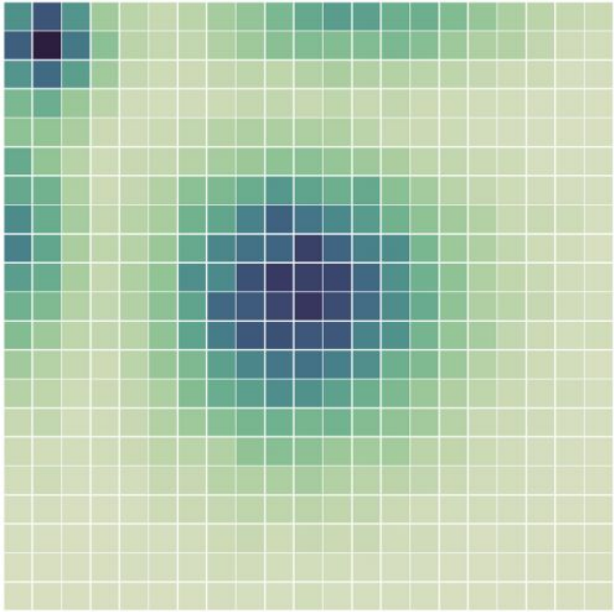
Fundamentals to Uncertainty & Robustness Methods

Neural Networks with SGD

Google AI Brain Team

Nearly all models find a single setting of parameters to maximize the probability conditioned on data.

$$
\begin{aligned}
\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y}) \\
&= \arg\min_{\boldsymbol{\theta}} -\log p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \\
&=^* \arg\min_{\boldsymbol{\theta}} \sum_k \mathbf{y}_k \log \mathbf{p}_k + \boldsymbol{\lambda} \|\boldsymbol{\theta}\|^2
\end{aligned}
$$



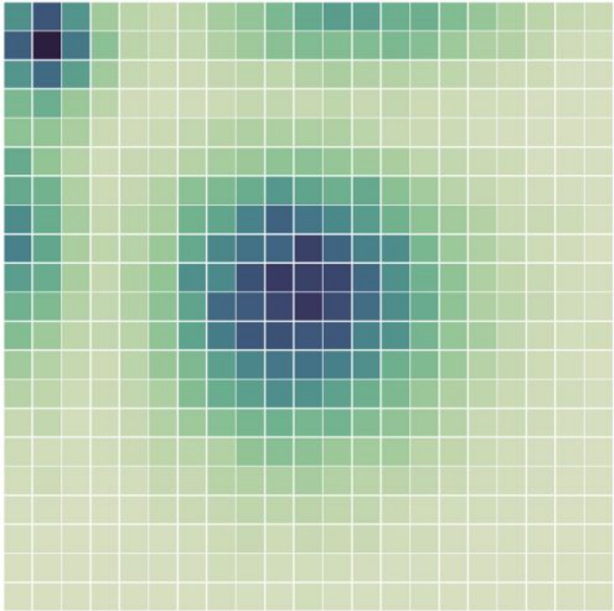
Special case: softmax cross entropy with L2 regularization. Optimize with SGD!

Image source: [Ranganath+ 2016]

Neural Networks with SGD

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y})$$

Problem: results in just one prediction per example
No model uncertainty



How do we get uncertainty?

- Probabilistic approach
  - Estimate a full distribution for $p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y})$
- Intuitive approach: Ensembling
  - Obtain multiple good settings for $\boldsymbol{\theta}^*$

Image source: Ranganath+ 2016

Probabilistic Machine Learning

Model: A probabilistic model is a joint distribution of outputs **y** and parameters $\boldsymbol{\theta}$ given inputs **x**.

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

Training time: Calculate the Bayesian **posterior**, the conditional distribution of parameters given observations.

$$p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{y}, \boldsymbol{\theta} \mid \mathbf{x})}{p(\mathbf{y} \mid \mathbf{x})} = \frac{p(\mathbf{y} \mid \mathbf{x}) p(\boldsymbol{\theta})}{\int p(\mathbf{y}, \boldsymbol{\theta} \mid \mathbf{x}) \, \mathrm{d}\boldsymbol{\theta}}$$

Prediction time: Compute the likelihood given parameters, each parameter configuration of which is weighted by the posterior.

$$p(\mathbf{y} \mid \mathbf{x}, \mathcal{D}) = \int p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathcal{D}) \, \mathrm{d}\boldsymbol{\theta} \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}^{(s)})$$

[Murphy 2012]

General Recipe

Parametrize "base model" with desired inductive biases.

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

Specify prior over functions.

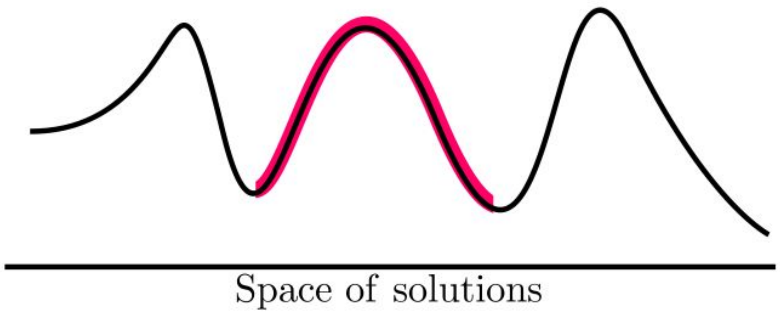
$$p(\boldsymbol{\theta})$$

Capture model uncertainty by approximating the posterior.

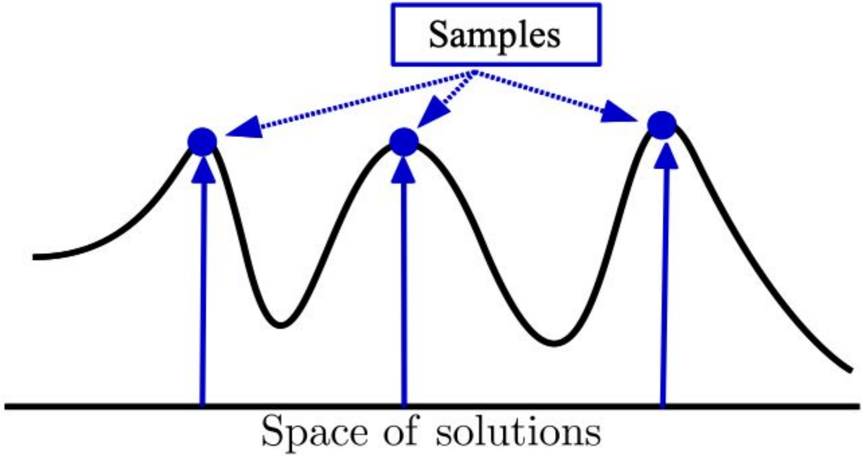
$$p(\boldsymbol{\theta} \mid \mathcal{D})$$

Approximating the posterior

$p(\boldsymbol{\theta} \mid \mathcal{D})$ is multimodal and complex, so how do we estimate and represent it?



Local approximations

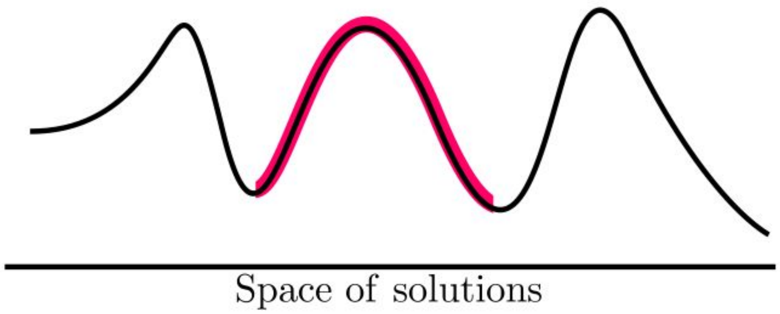


Sampling

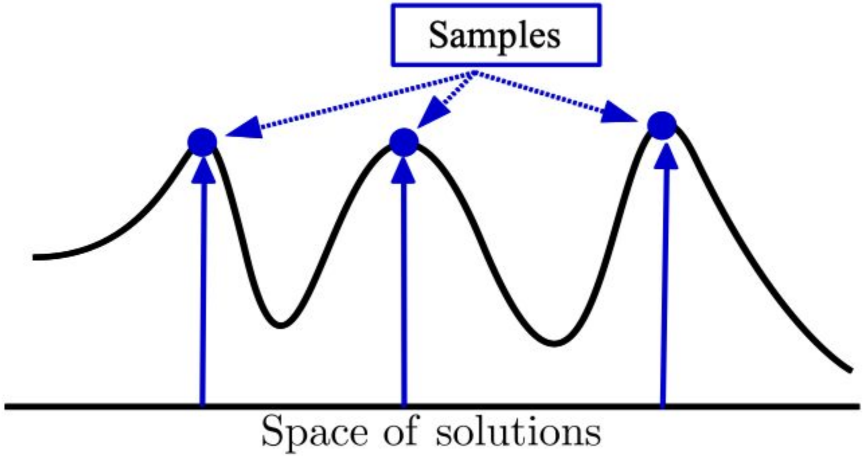
- Locally, covering one mode well
  e.g. with a simpler distribution $q(\boldsymbol{\theta}; \boldsymbol{\lambda})$
    - Variational inference
    - Laplace approximation

Approximating the posterior

$p(\boldsymbol{\theta} \mid \mathcal{D})$ is multimodal and complex, so how do we estimate and represent it?



Local approximations



Sampling

- Summarize using samples
  - MCMC
  - Hamiltonian Monte Carlo
  - Stochastic Gradient Langevin Dynamics

Ensemble Learning

Google AI
Brain Team

- A prior distribution often involves the complication of approximate inference.
- *Ensemble learning* offers an alternative strategy to aggregate the predictions over a collection of models.
- Often winner of competitions!
- There are two considerations: the collection of models to ensemble; and the aggregation strategy.

Popular approach is to average predictions of independently trained models, forming a mixture distribution.

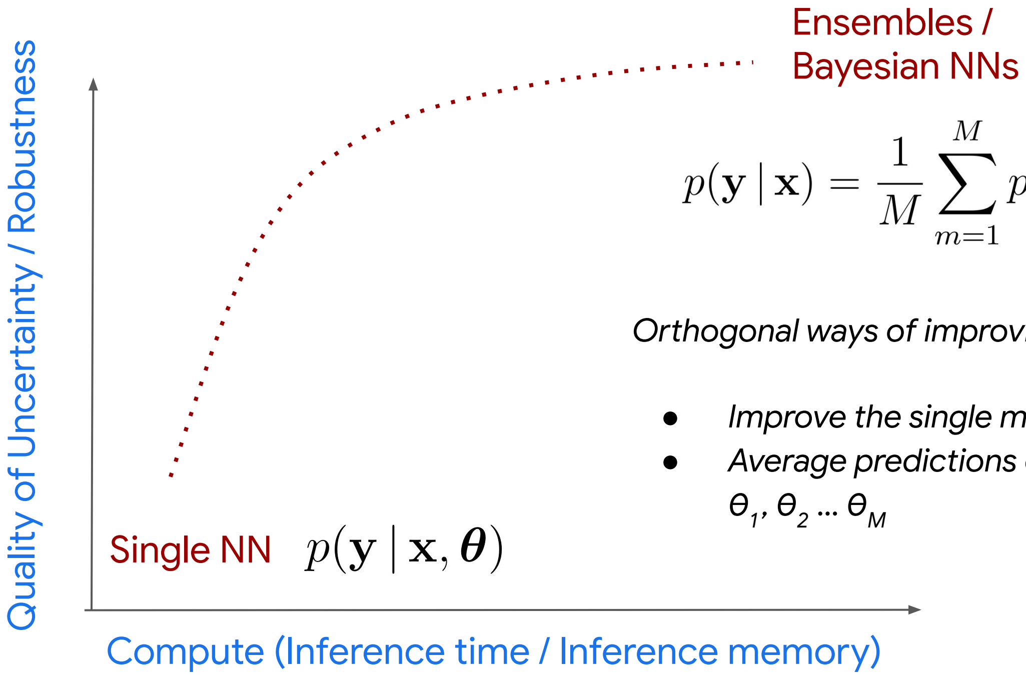
$$p(\mathbf{y} \,|\, \mathbf{x}) = \frac{1}{K} \sum_{k=1}^{K} p(\mathbf{y} \,|\, \mathbf{x}, \boldsymbol{\theta}_k)$$

Many approaches exist: bagging, boosting, decision trees, stacking.

[Dietterich 2000]

Overview of Methods

Cartoon: Uncertainty/Robustness vs Compute frontier

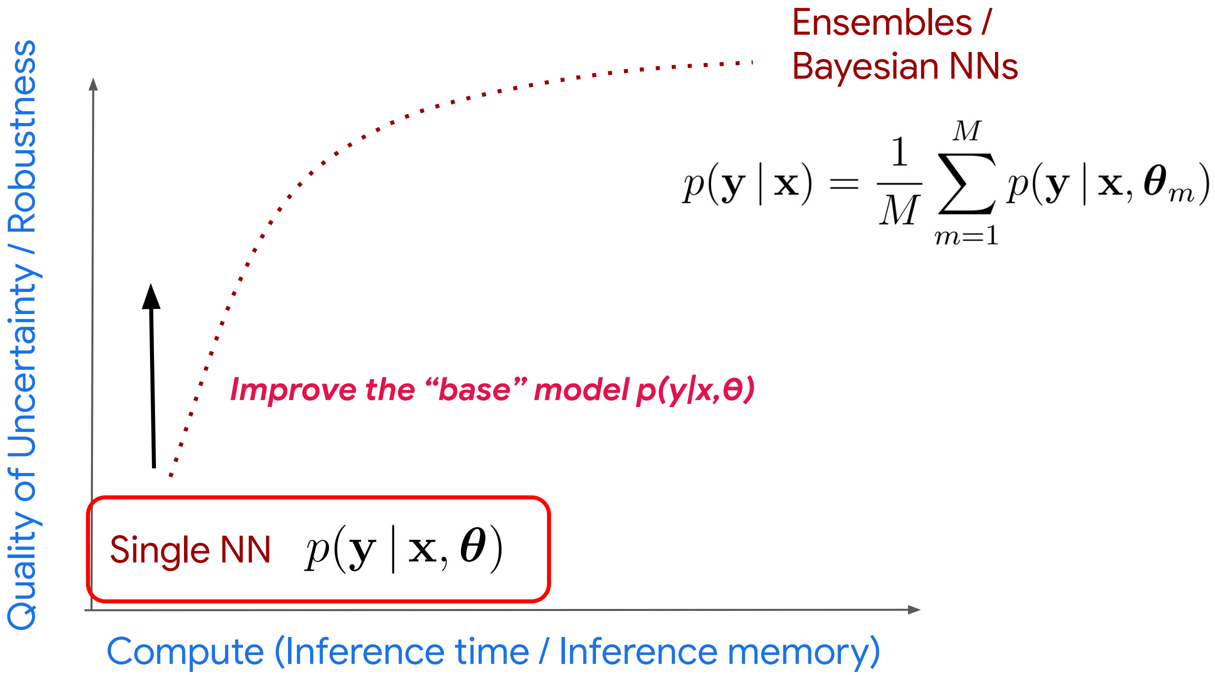


$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_m)$$

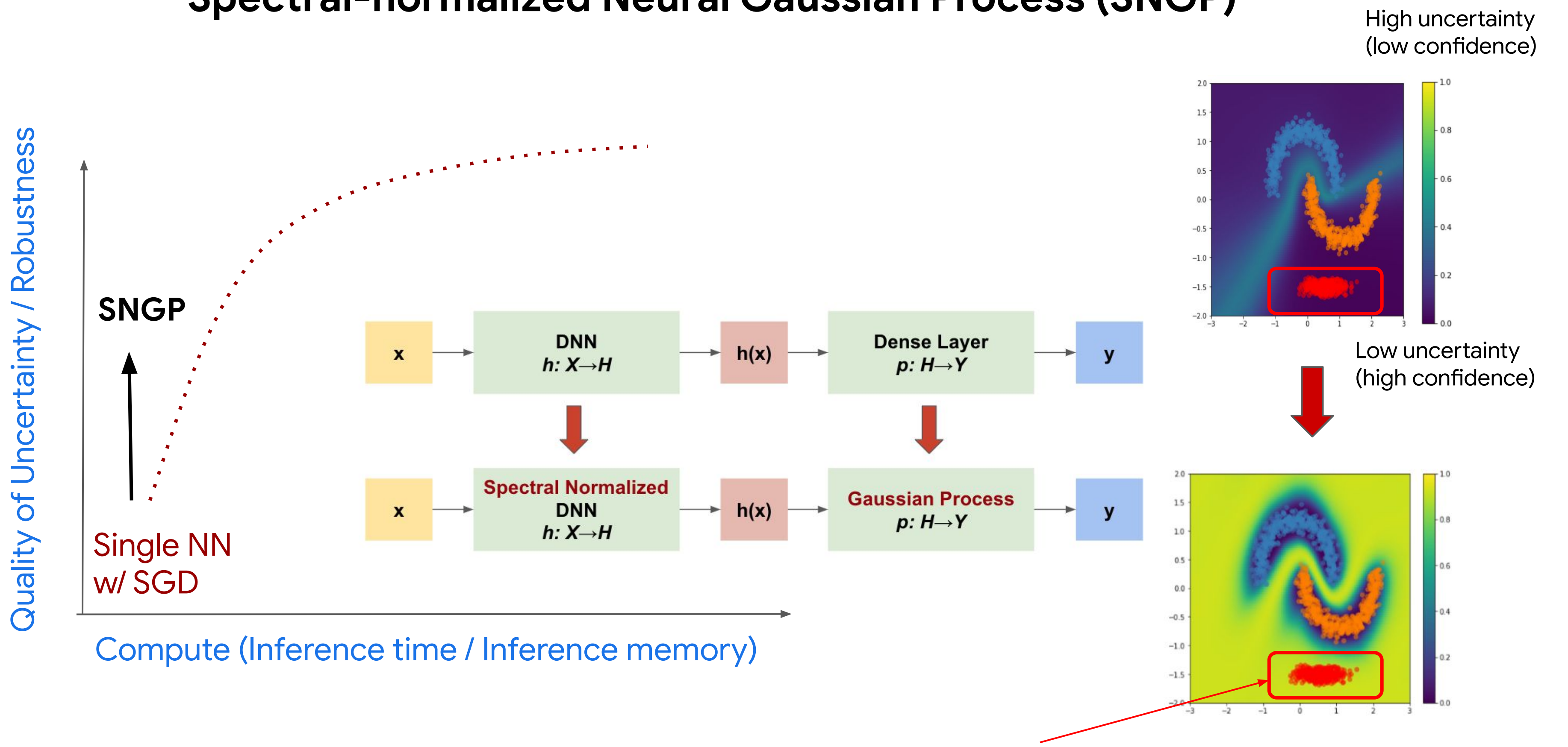
Orthogonal ways of improving performance

- *Improve the single model p(y|x,θ)*
- *Average predictions over diverse set of functions* $\theta_1, \theta_2 \dots \theta_M$

Improving single model performance



Adding *distance-awareness* using **Spectral-normalized Neural Gaussian Process (SNGP)**



SNGP assigns lower confidence predictions to inputs far away from the training data

Imposing distance awareness

Google AI
Brain Team

"Models should be distance aware:
uncertainty increases farther from training data."

Spectral-normalized Neural Gaussian process

1. Replace output layer with "GP layer".
2. Apply spectral normalization to preserve input distances within internal layers.

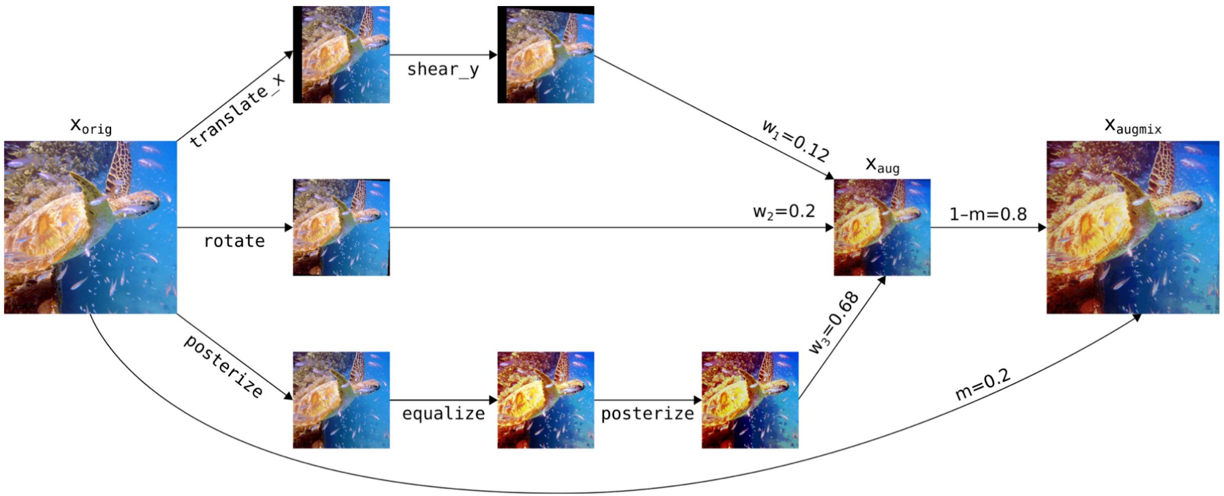
See also [van Amersfoort+ 2020].

BERT on an intent detection benchmark

Method	Accuracy (↑)	ECE (↓)	OOD AUROC (↑)	OOD 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

[Liu+ 2020]

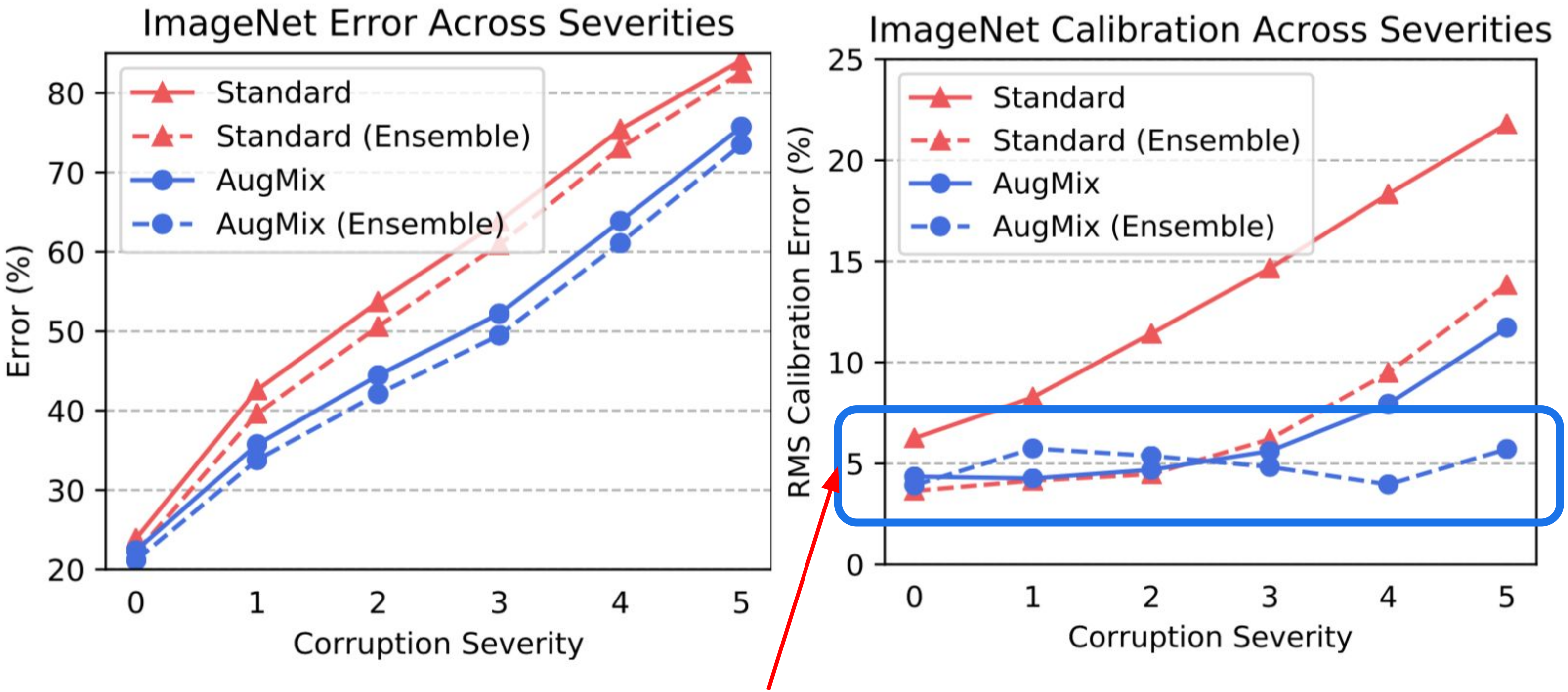
Better representations via data augmentation, e.g. AugMix



Composing base operations and 'mixing' them can improve accuracy and calibration under shift.

[Hendrycks+ 2020]

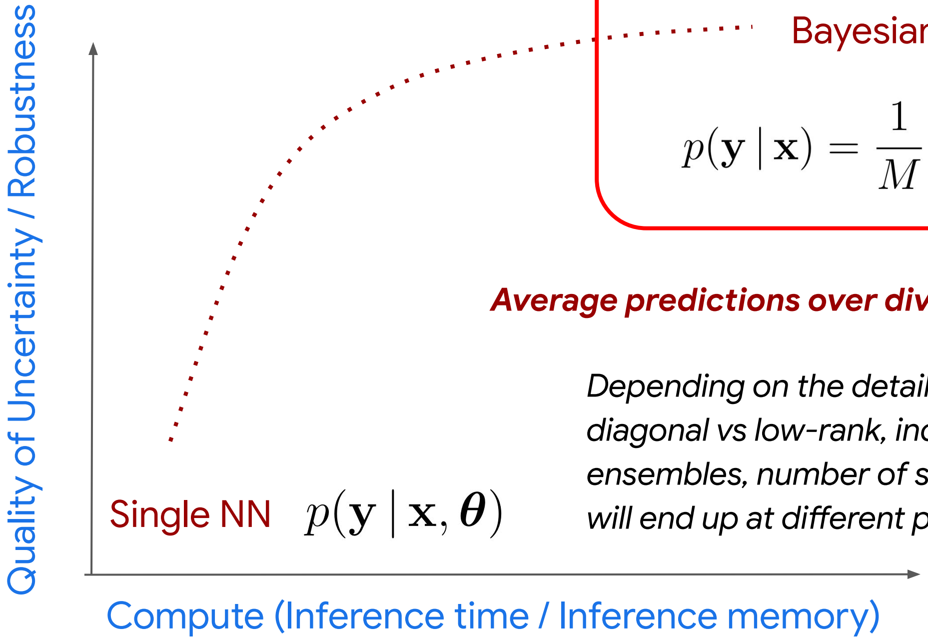
AugMix improves accuracy & calibration under shift



Data augmentation can provide complementary benefits to ensembling.

[Hendrycks+ 2020]

Improving the quality of model uncertainty



Ensembles /
Bayesian NNs

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

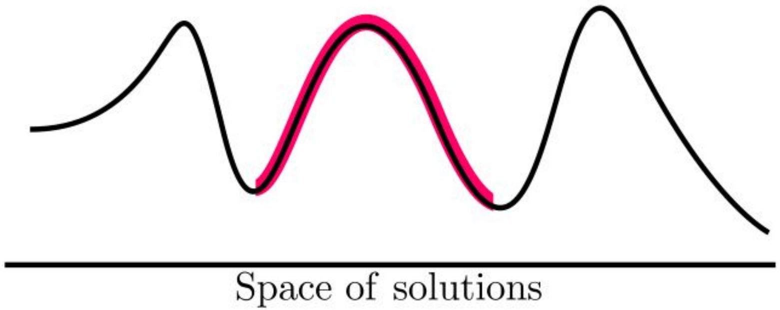
Average predictions over diverse set of functions $\theta_1, \theta_2 \ldots \theta_M$

Depending on the details of the approximation (e.g. diagonal vs low-rank, independent ensembles vs efficient ensembles, number of samples/ensemble members), we will end up at different points along this curve.

Simple Baseline: SWAG + Laplace

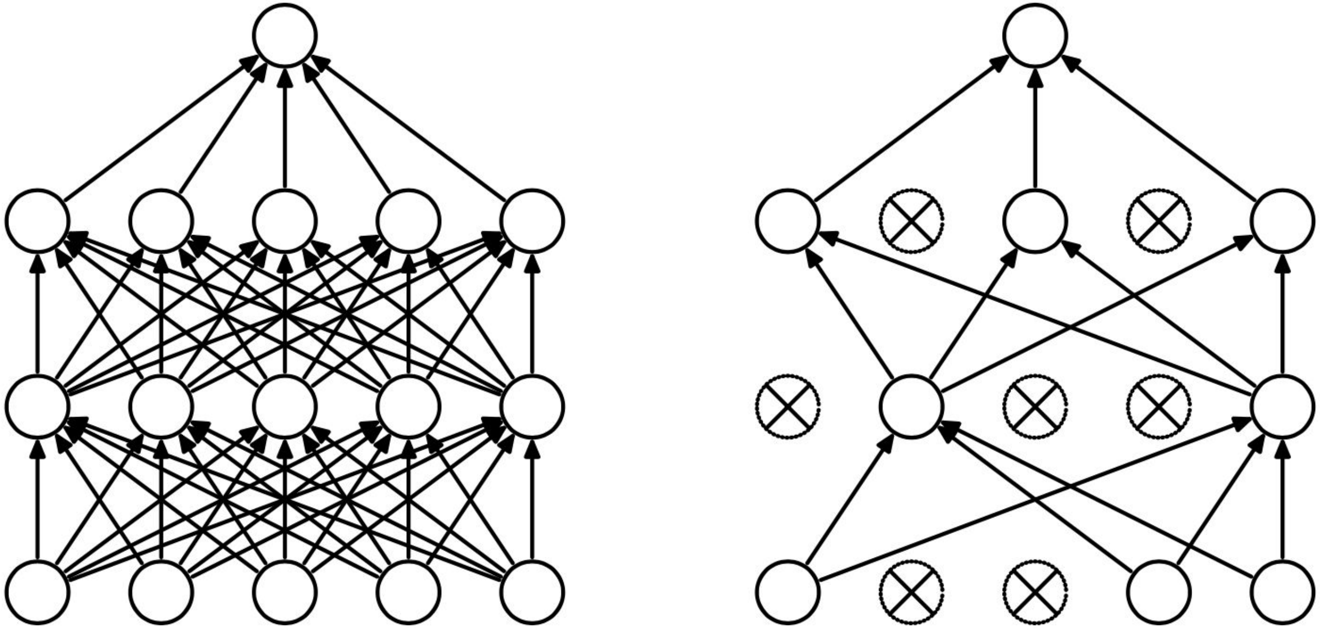
Fit a simple distribution to the mode centered around the SGD solution

- SWAG: Fit a Gaussian around averaged weight iterates near the mode
- Laplace: Fit a quadratic at the mode, using the Hessian or Fisher information



[Maddox+ 2019]

Simple Baseline: Monte Carlo Dropout



(a) Standard Neural Net (b) After applying dropout.

Image source: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

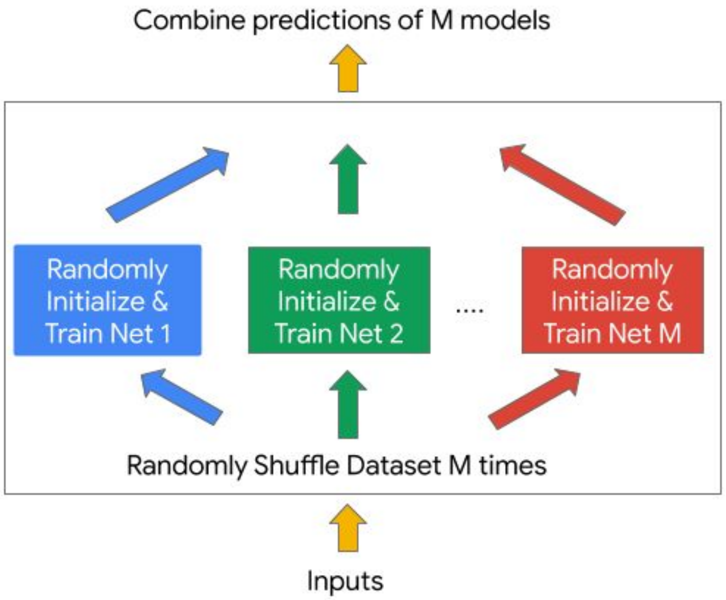
[Gal+ 2015]

Simple Baseline: Deep Ensembles

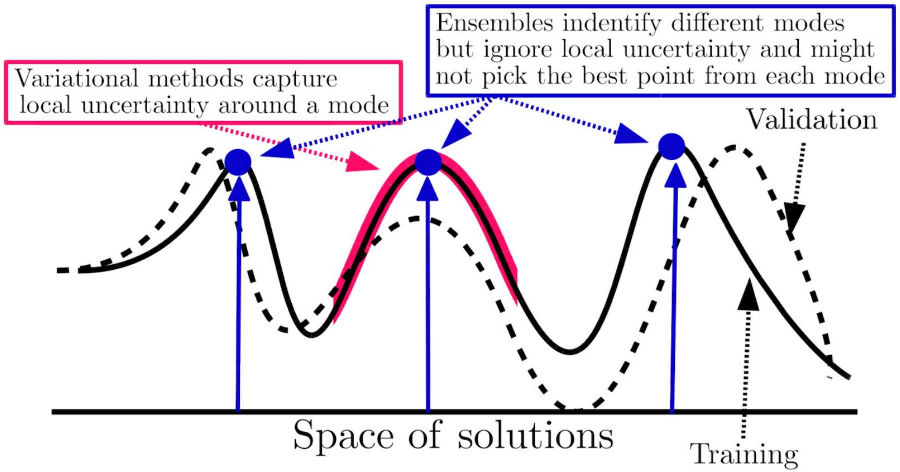
Google AI
Brain Team

Idea: Just re-run standard SGD training but with different random seeds and average the predictions.

Deep ensembles can capture different modes in function space.



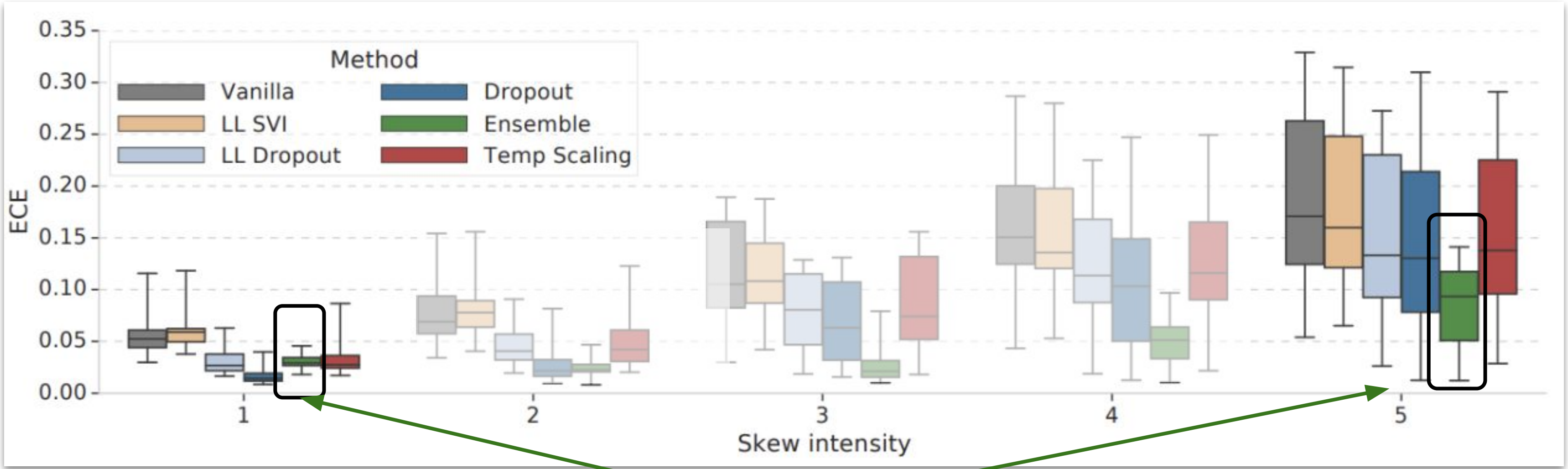
[Lakshminarayanan+ 2016]



[Fort+ 2019]

Google AI
Brain Team

Deep Ensembles work surprisingly well in practice

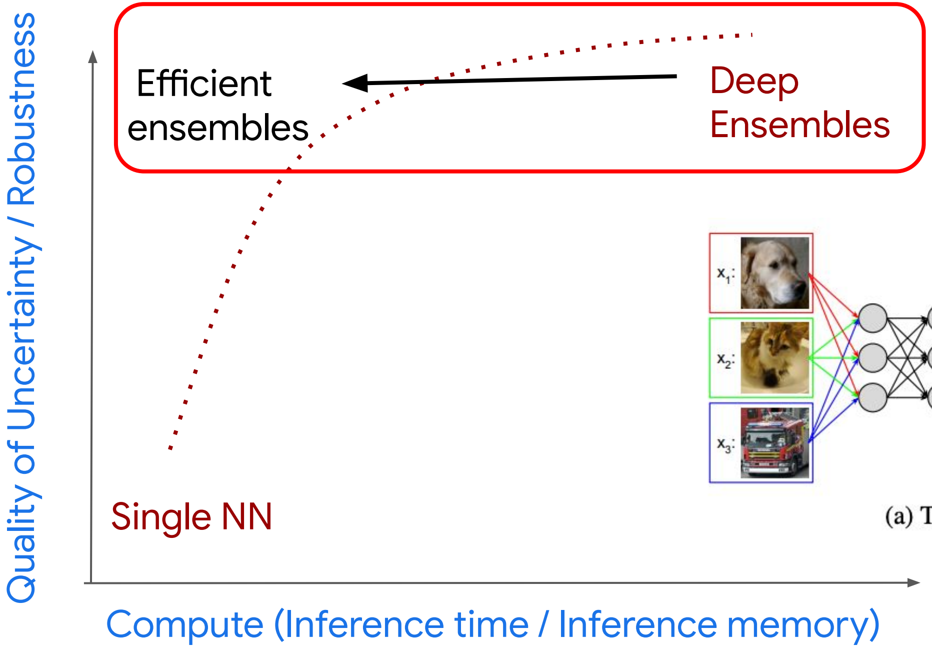


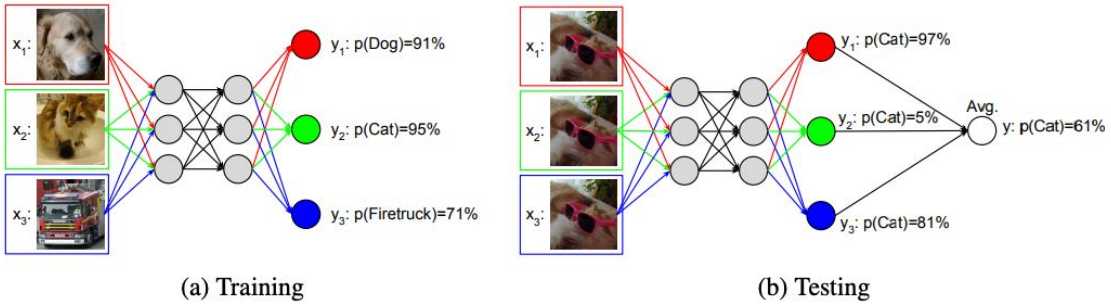
Deep Ensembles are consistently among the best performing methods, especially under dataset shift

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

Google AI Brain Team

Efficient ensembles lower inference memory and/or inference time





[Training Independent Subnetworks for Robust Prediction. Havasi+ 2020]

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

Google AI
Brain Team

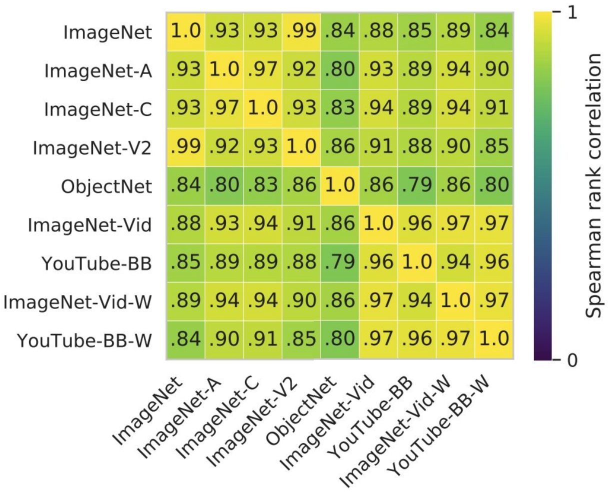
Robustness Metrics

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.
- Benchmark models with calibration error and a large collection of OOD shifts.
- Probabilistic ML, ensemble learning, and optimization provide a foundation.
- The best methods advance two dimensions: combining multiple neural network predictions; and imposing priors and inductive biases.

Links to papers: http://www.gatsby.ucl.ac.uk/~balaji/

Check out recent ICML workshop on Uncertainty and Robustness in Deep Learning

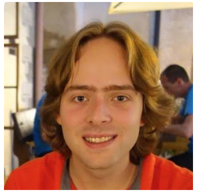
Google AI
Brain Team

Thank you!


Ben Adlam


Josip Djolonga


Mike Dusenberry

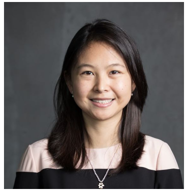

Stanislav Fort


Justin Gilmer


Marton Havasi

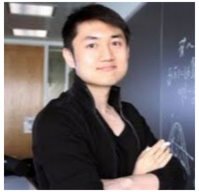

Danijar Hafner


Dan Hendrycks


Clara Hu


Rodolphe Jenatton


Ghassen Jerfel


Jeremiah Liu


Mario Lucic


Zelda Mariet


Rafael Müller


Kevin Murphy

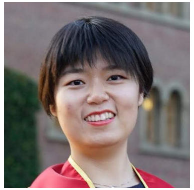

Zack Nado


Eric Nalisnick


Kathleen Nix


Jeremy Nixon


Shreyas Padhy


Jie Ren


D. Sculley


Yeming Wen


Florian Wenzel

& 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.

Google AI Brain Team

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.
- **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*
- **Improving Calibration of BatchEnsemble with Data Augmentation**. *Y. Wen, G. Jerfel, R. Muller, M. Dusenberry, J. Snoek, B. Lakshminarayanan and D. Tran. link*

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 arXiv 2006.09273

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 *arXiv 2106.03004*.