Decision Making
in Robots and Autonomous Agents

Explainability for Decision Making

Subramanian Ramamoorthy
School of Informatics

22 March 2019

How to build trust in robots' decisions?

http://www.henrypayne.com/wp-content/uploads/2014/05/051314_BackseatSelfDrive_COLOR.jpg

Question of interpretability of a sequence of actions

- Necessity to explain executed path behaviour, not just one-shot decisions
- Explanations might need to be built with query dependent symbols

http://2.bp.blogspot.com/-5QhUSfM8Rnc/VYLF9EpRhgI/AAAAAAAAABU/mI52ESzloV8/s1600/mag_curry01jr_576x324.jpg

Prevalence of AI Systems

- Machine learning based decisions are finally finding their way into deployment
  - From ad placement to autonomous navigation
  - Promise of autonomous systems that will perceive, learn, **decide, and act on their own**.
- Machine learning models can still be opaque, non-intuitive, and difficult for people to understand (especially lay people)
- The effectiveness of these systems will be limited by the machine's inability to explain its thoughts and actions to the human user

What kinds of questions does the user need answers to?

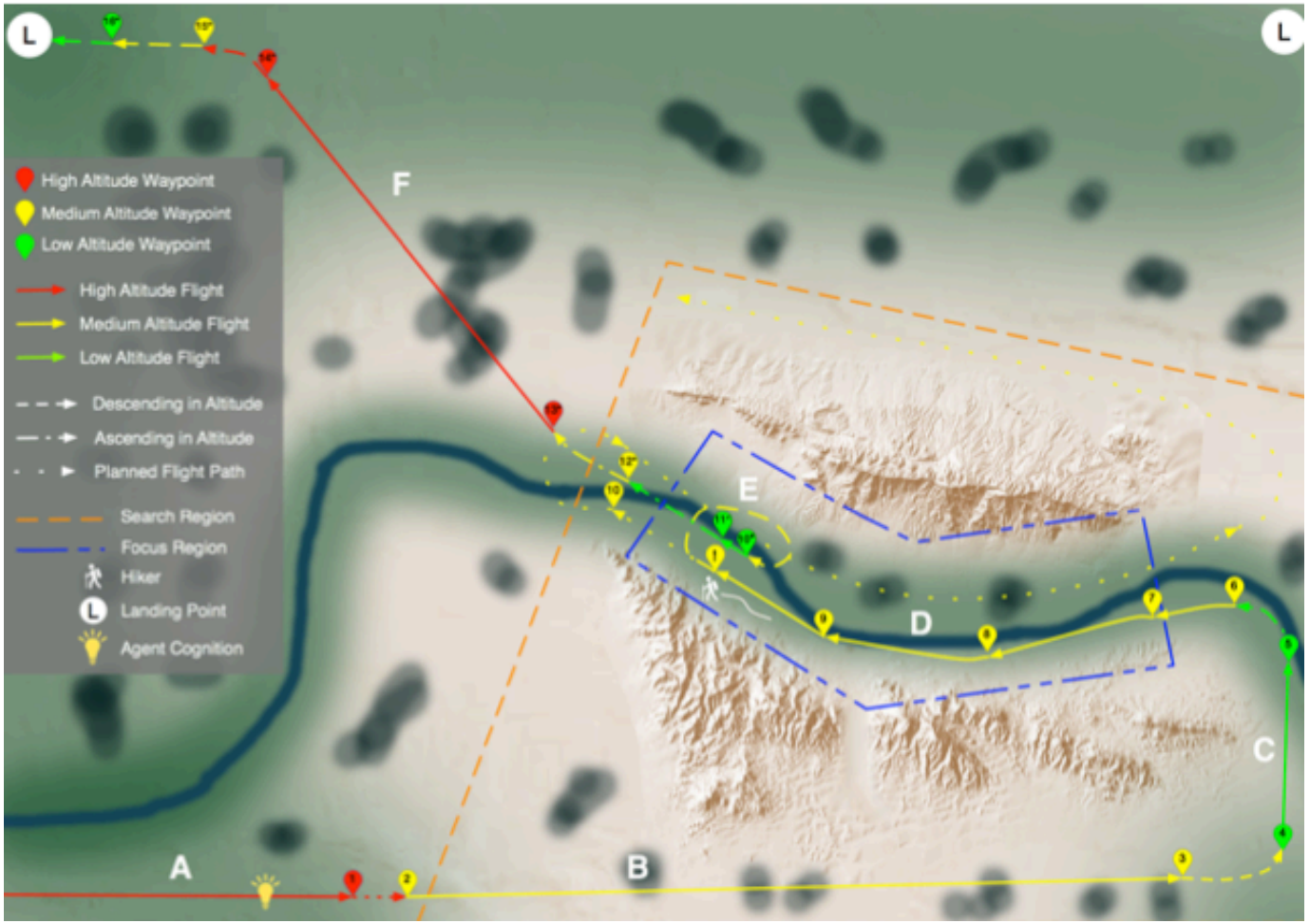
1. Why did you do that?
2. Why not something else?
3. When do you succeed?
4. When do you fail?
5. When can I trust you?
6. How do I correct an error?

Exercise: Explain this UAV Trace

Why is this Important?

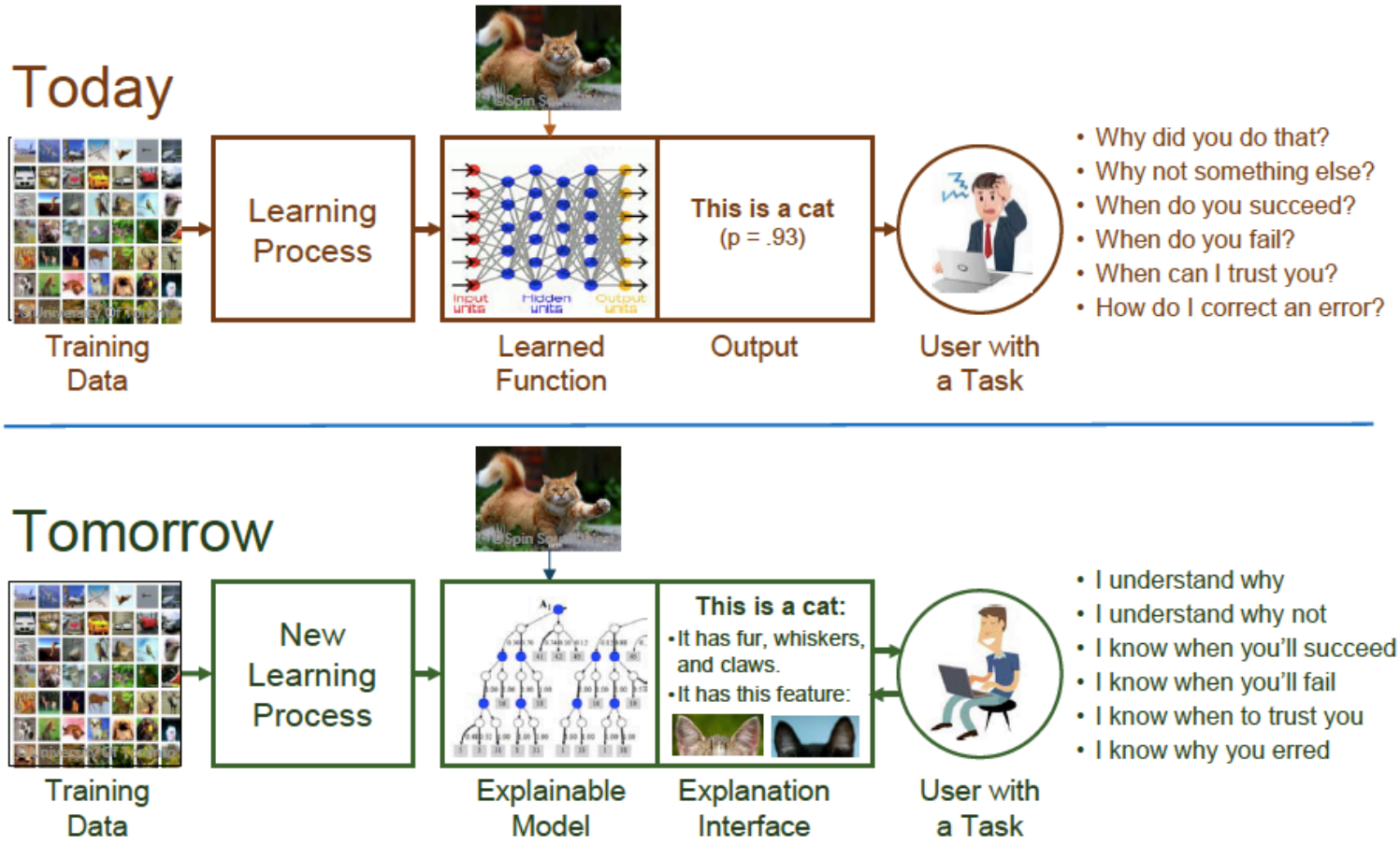
[U.S. Air Force Photo by Tech Sgt Effrain Lopez, 4th Combat Camera; via https://www.airforcetimes.com]

Problems with Complex Specifications e.g., UAV Flight Plans



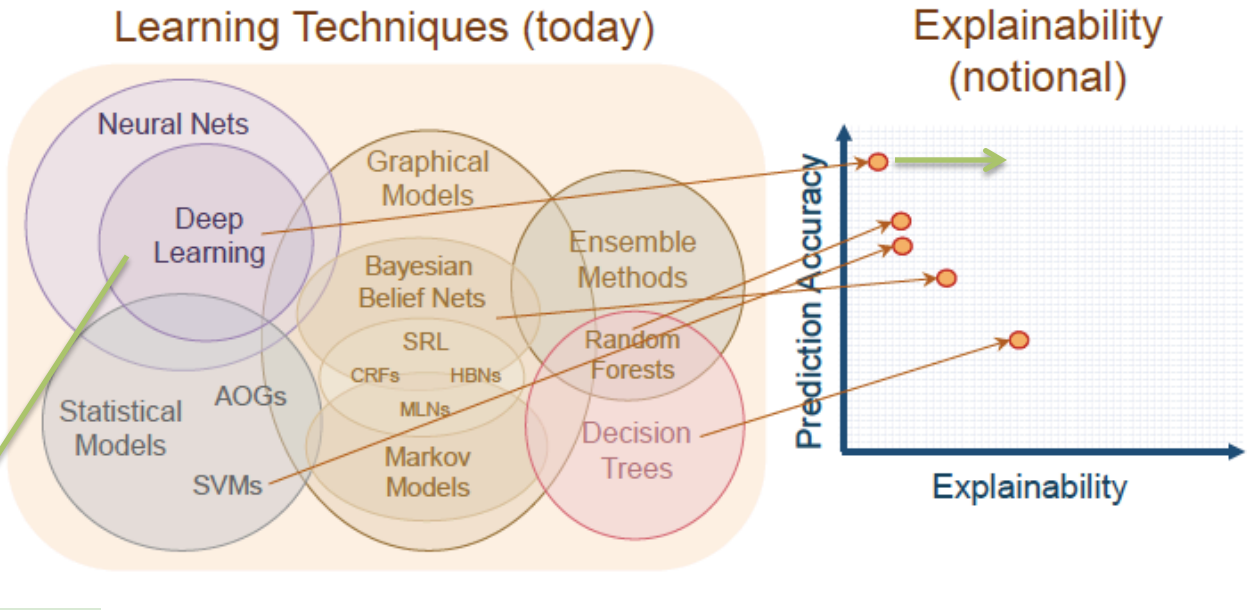
[Src: M. Stefik, PARC]

Two Paradigms

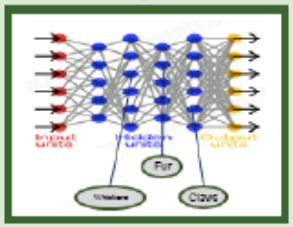


[D. Gunning, DARPA-BAA-16-53]

Levels of Explainability and Aspirations



[D. Gunning, DARPA-BAA-16-53]



Deep Explanation

Modified deep learning techniques to learn explainable features

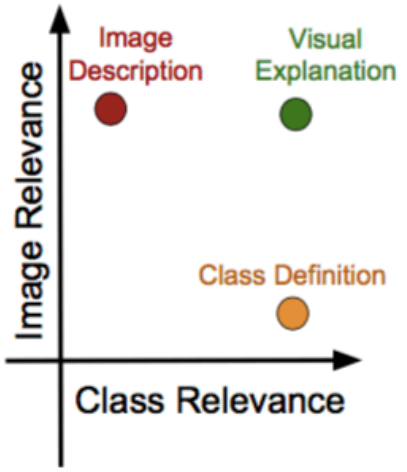
Many Approaches to Explanation

Just as there are many different machine learning models to solve different problems, the desiderata regarding explanations can also diverse, e.g.,

- Explanation through generation of captions
- Programmatically structured representations
- Local approximations with simpler models

  – After looking at an example of each of these, we'll take a closer look at one approach, based on the statistical concept of influence functions

Explanation through Captions



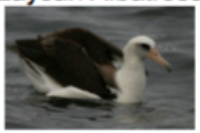
Western Grebe

Description: This is a large bird with a white neck and a black back in the water.
Class Definition: The *Western Grebe* is a waterbird with a yellow pointy beak, white neck and belly, and black back.
Explanation: This is a *Western Grebe* because this bird has a long white neck, pointy yellow beak and red eye.

Laysan Albatross

Description: This is a large flying bird with black wings and a white belly.
Class Definition: The *Laysan Albatross* is a large seabird with a hooked yellow beak, black back and white belly.
Visual Explanation: This is a *Laysan Albatross* because this bird has a large wingspan, hooked yellow beak, and white belly.

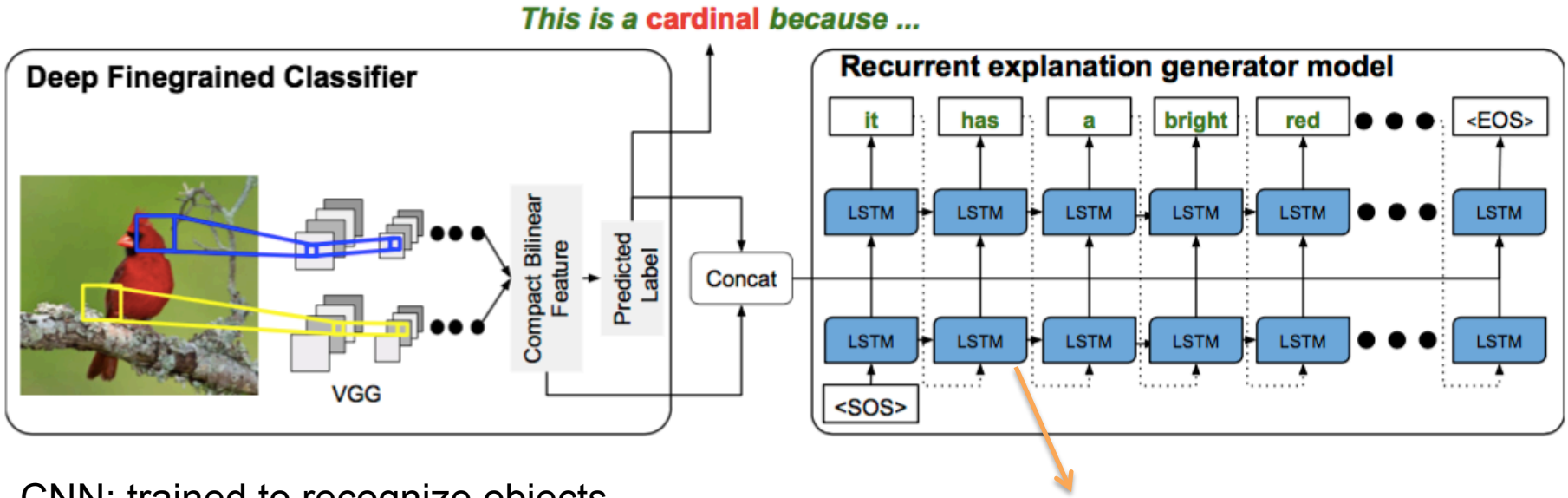
Laysan Albatross



Description: This is a large bird with a white neck and a black back in the water.
Class Definition: The *Laysan Albatross* is a large seabird with a hooked yellow beak, black back and white belly.
Visual Explanation: This is a *Laysan Albatross* because this bird has a hooked yellow beak white neck and black back.

[Hendricks, L.A, et al. (2016). Generating Visual Explanations, arXiv:1603.08507v1]

How to generate captions?
Joint use of multiple models

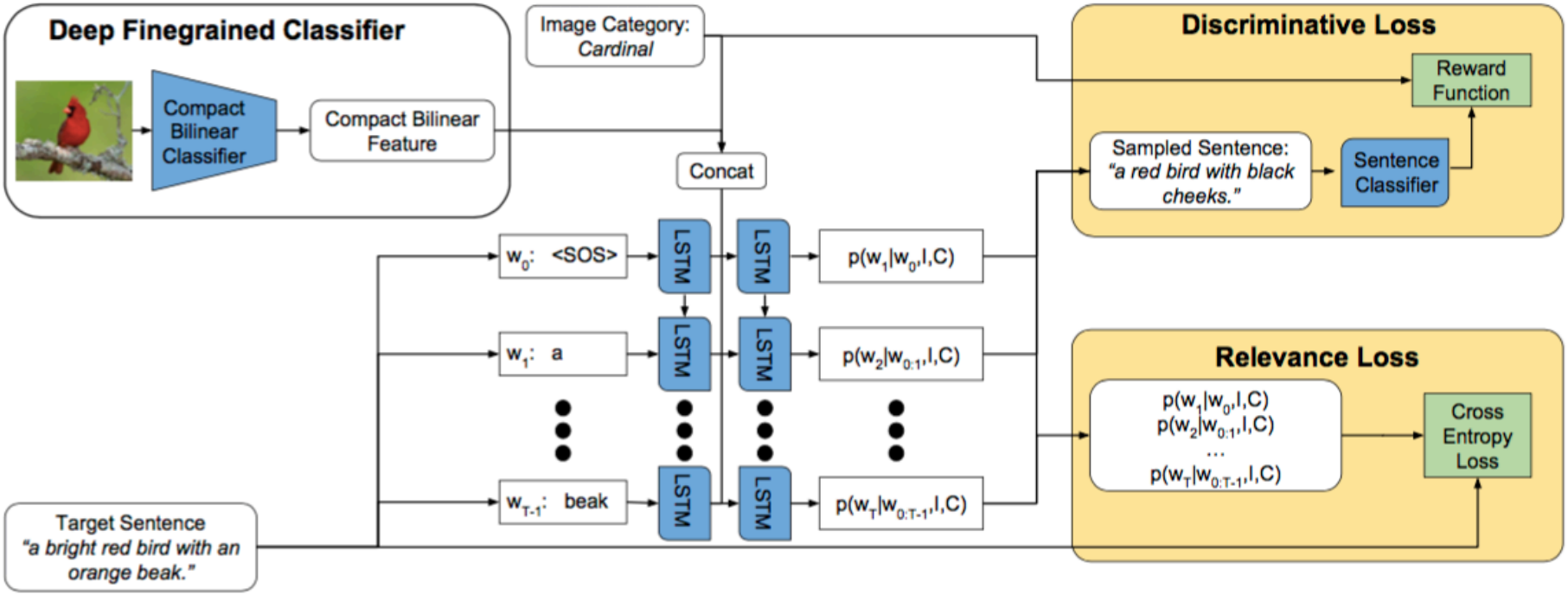


CNN: trained to recognize objects
Language generating RNN: trained to translate features of the CNN into words and captions.

LSTM: Long short term memory, a model that generates sequences

[Hendricks, L.A, et al. (2016). Generating Visual Explanations, arXiv:1603.08507v1]

Approach to Training Joint Model



Programmatically Structured Models

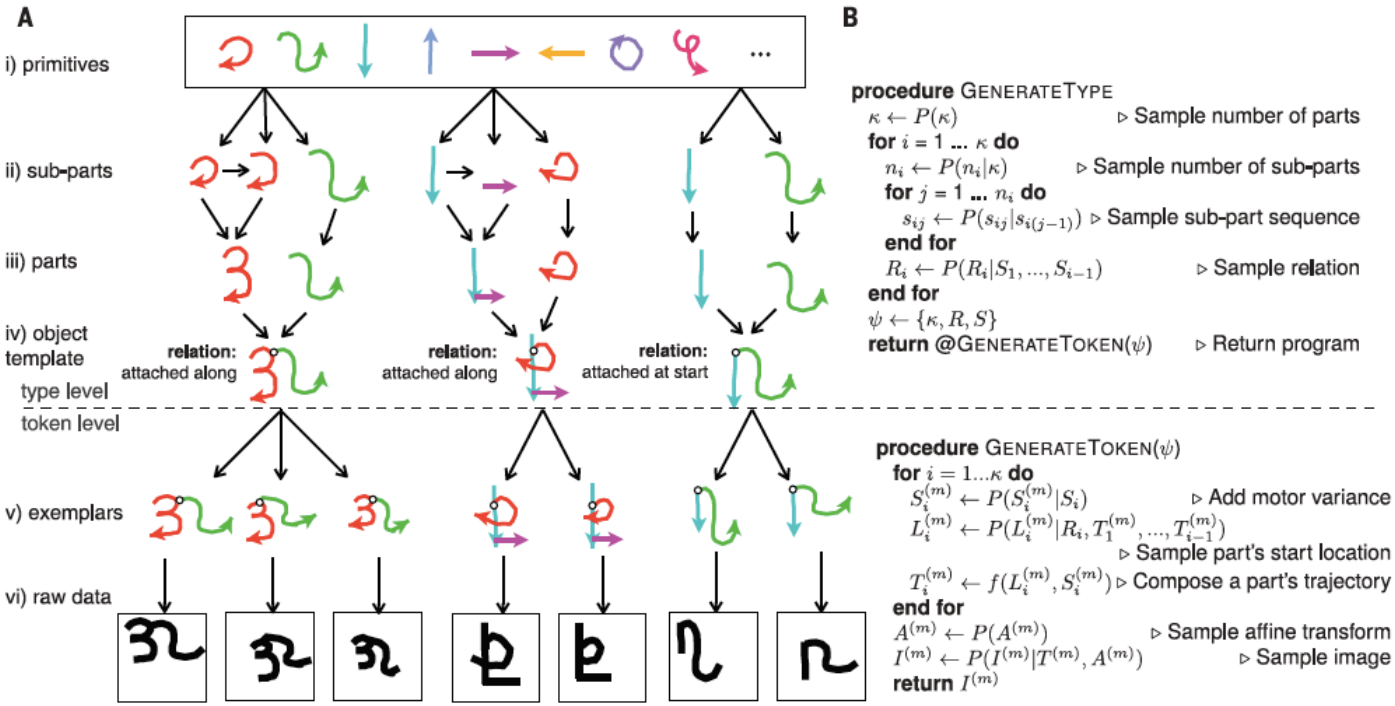


Fig. 3. A generative model of handwritten characters. (**A**) New types are generated by choosing primitive actions (color coded) from a library (i), combining these subparts (ii) to make parts (iii), and combining parts with relations to define simple programs (iv). New tokens are generated by running these programs (v), which are then rendered as raw data (vi). (**B**) Pseudocode for generating new types ψ and new token images $I^{(m)}$ for $m = 1, \ldots, M$. The function $f(\cdot, \cdot)$ transforms a subpart sequence and start location into a trajectory.

Lake, B.H., Salakhutdinow, R., & Tennenbaum, J.B. (2015). Human-level concept learning through probabilistic program induction. *Science* 350: 1332-1338.

Are there statistical models that are "inherently" explainable? Decision Lists

A decision list for the "Titanic dataset",
in parantheses is the 95% credible interval for survival probability:

if male **and** adult **then** *survival probability* 21% (19%–23%)
else if 3rd class **then** *survival probability* 44% (38%–51%)
else if 1st class **then** *survival probability* 96% (92%–99%)
else *survival probability* 88% (82%–94%)

Bayesian Rule Lists (BRL), produce a posterior distribution over permutations of if. . . then. . . rules, starting from a large, predetermined set of possible rules.

[B. Letham et al. (2015). Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. Annals of Applied Statistics Vol. 9, No. 3, 1350-137]

More Complex Decision Lists

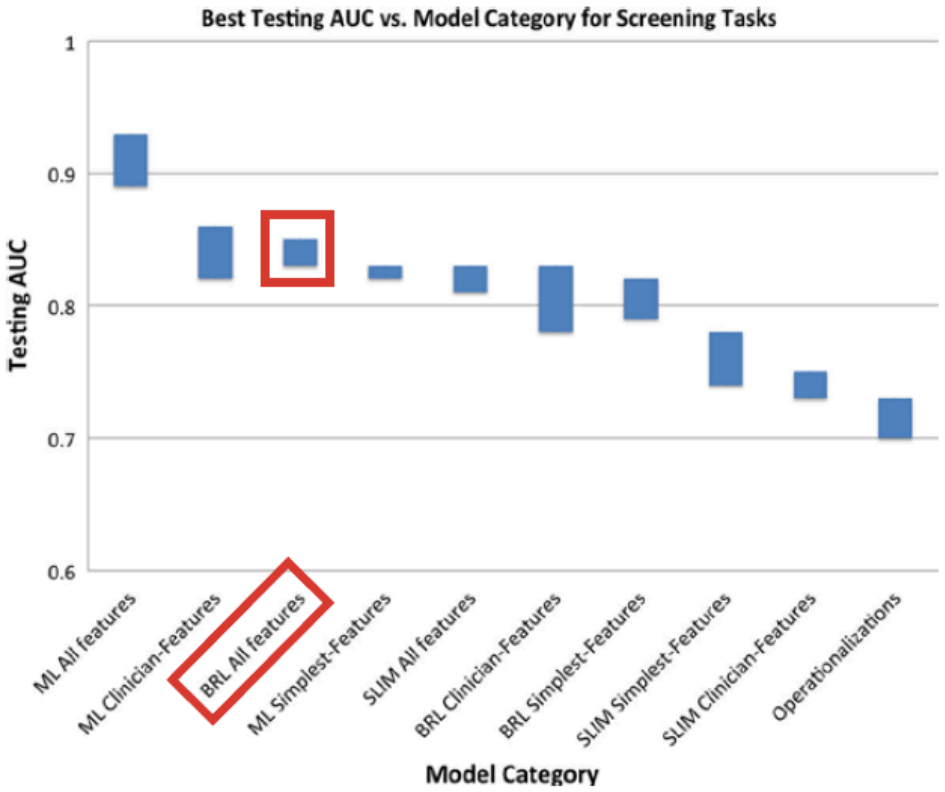
if hemiplegia **and** age >60 **then** *stroke risk* 58.9% (53.8%–63.8%)
else if cerebrovascular disorder **then** *stroke risk* 47.8% (44.8%–50.7%)
else if transient ischaemic attack **then** *stroke risk* 23.8% (19.5%–28.4%)
else if occlusion and stenosis of carotid artery without infarction **then** *stroke risk* 15.8% (12.2%–19.6%)
else if altered state of consciousness **and** age >60 **then** *stroke risk* 16.0% (12.2%–20.2%)
else if age ≤ 70 **then** *stroke risk* 4.6% (3.9%–5.4%)
else *stroke risk* 8.7% (7.9%–9.6%)

Decision list for determining 1-year stroke risk following diagnosis of atrial fibrillation from patient medical history. The risk given is the mean of the posterior consequent distribution, and in parentheses is the 95% credible interval.

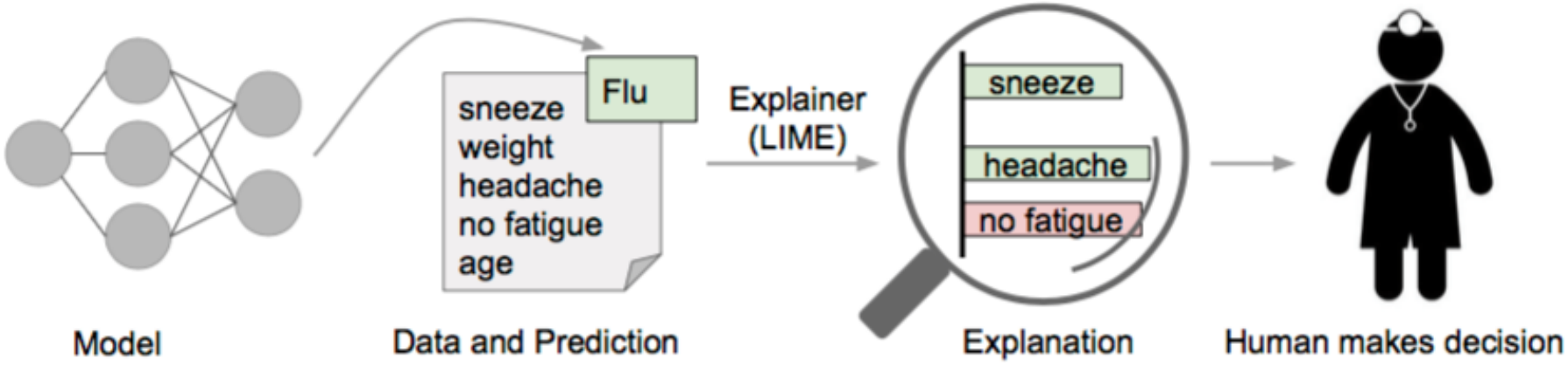
Power of Bayesian Rule Lists

- BRLs discretize a high dimensional, multivariate feature space into a series of simple, readily interpretable decision statements.

- Experiments show that BRLs can have predictive accuracy on par with the current top ML algorithms

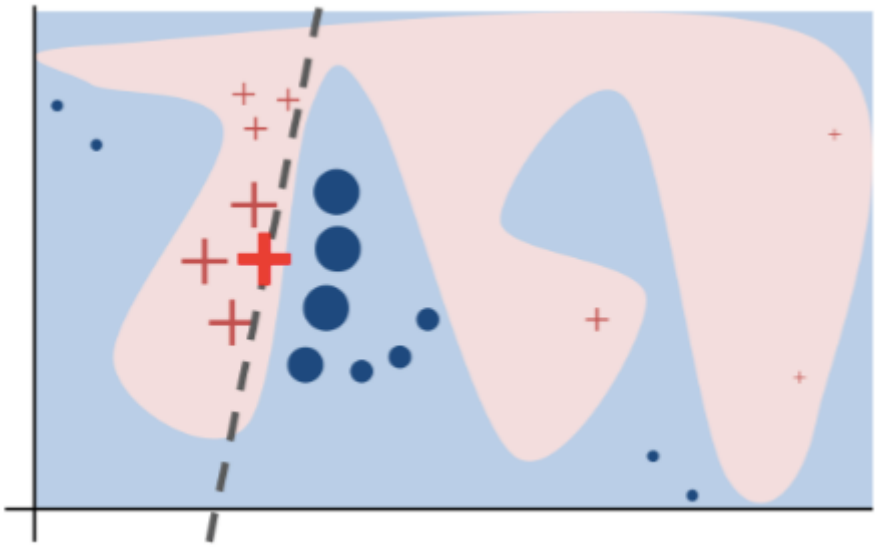


Analyzing Local Model Properties to Explain Individual Predictions



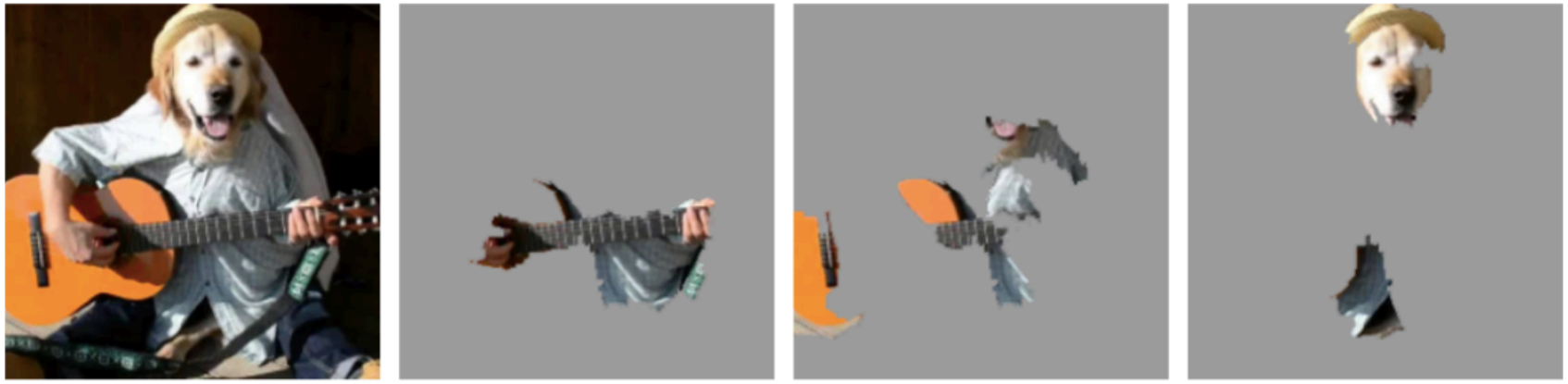
[M.T. Ribeiro et al., Why should I trust you?: Explaining the predictions of any classifier. In Proc. ACM SIGKDD 2016.]

LIME: An approach based on approximating locally with an interpretable model



The black-box model's complex decision function f (unknown to LIME) is represented by the blue/pink background, which cannot be approximated well by a linear model. The bold red cross is the instance being explained. LIME samples instances, gets predictions using f, and weighs them by the proximity to the instance being explained (represented here by size).
The dashed line is the learned explanation that is locally (but not globally) faithful.

Explaining an Image Classification



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

p = 0.32 p = 0.24 p = 0.21

[M.T. Ribeiro et al., Why should I trust you?: Explaining the predictions of any classifier. In Proc. ACM SIGKDD 2016.]

Influence Functions in Statistics

- The empirical influence function is a measure of the dependence of the estimator on the value of one of the points in the sample
- Model-free measure: relies on calculating the estimator again with a different sample
- Consider a set of random variables and an iid sample:

$$(x_1, x_2, \cdots, x_n) \text{ drawn from variables } X_1 X_2 \cdots X_n$$

- If T_n is an estimator based on this, then the empirical influence function is,

$$EIF_i : x \to n.(T_n(x_1, ..., x_{i-1}, x, x_{i+1}, ..., x_n) - T_n(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n))$$

What is an Influence Function Capturing?

- We are replacing the i^{th} value in the sample by an arbitrary value and looking at the output of the estimator
- If this data point is not 'important' to the output of the estimator, then the influence function should output a low value. Why?

$$EIF_i : x \to n.(T_n(x_1, ..., x_{i-1}, x, x_{i+1}, ..., x_n) - T_n(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n))$$

- How is this useful for explainability?

Using Influence Functions for ML Models

- Address the counterfactual question: what would happen if we did not have this training point, or if the values of this training point were changed slightly?

- Easy way is to directly perform sensitivity analysis: perturb the data point, retrain model, evaluate change.
  - Prohibitively expensive for complex models!

- This is where influence functions come in – help evaluate this sensitivity

Basic Setup of Influence Function for Black-box Predictions

Input Space: $\mathcal{X}$, (e.g., images)

Output Space: $\mathcal{Y}$, (e.g., labels)

Training points: $(z_1, ..., z_n)$, where $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$

For a point z and model parameters θ, let the loss be $L(z, \theta)$ and the empirical risk, $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

The minimizer of this empirical risk is,
$\hat{\theta} = \arg\min_\theta \frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

Set up empirical risk to be twice differentiable and convex

Effect of a training point on prediction

What if we did not have the training point at all?

Consider training set with the point z removed.

The change in model parameters due to removal of the point is $\hat{\theta}_{-z} - \hat{\theta}$ where,

$\hat{\theta}_{-z} = \arg\min_\theta \frac{1}{n} \sum_{z_i \neq z} L(z_i, \theta)$

Retraining the model to calculate this can be slow!

Effect of training point on prediction

Consider the situation where z is upweighted by a small amount ϵ,

$$\hat{\theta}_{\epsilon,z} = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z, \theta)$$

The influence of upweighting z on the parameters $\hat{\theta}$ is,

$$\mathcal{I}_{up.param}(z) = \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}|_{\epsilon=0} = -\mathrm{H}_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$

Effect of training point on prediction

Removing a point z is the same as upweighting by $\epsilon = -\frac{1}{n}$

So we can linearly approximate parameter change due to removing z by computing,

$\hat{\theta}_{-z} - \hat{\theta} \sim -\frac{1}{n}\mathcal{I}_{up.param}(z)$

This does not need the model to be retrained.

How does upweighting z change functions of parameters?

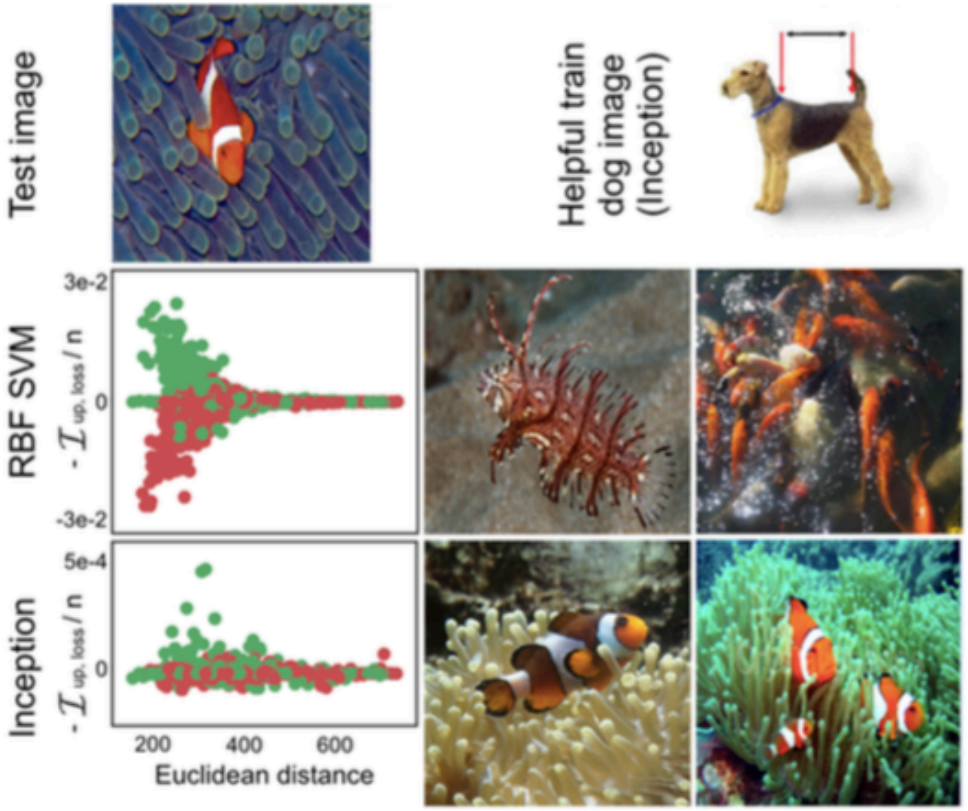
Influence of upweighting z on the loss at test point z_{test}:

$$\mathcal{I}_{up.loss}(z, z_{test}) = \frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}|_{\epsilon=0}$$

$$\mathcal{I}_{up.loss}(z, z_{test}) = \nabla_\theta L(z_{test}, \hat{\theta})^T \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}|_{\epsilon=0}$$

$$\mathcal{I}_{up.loss}(z, z_{test}) = \nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$

Example Use of Influence Functions



[Koh et al., ICML 2017]

Figure 4. **Inception vs. RBF SVM. Bottom left:** $-\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}})$ vs. $\|z - z_{\text{test}}\|_2^2$. Green dots are fish and red dots are dogs. **Bottom right:** The two most helpful training images, for each model, on the test. **Top right:** An image of a dog in the training set that helped the Inception model correctly classify the test image as a fish.