Uncertainty in Recurrent Decision Tree Classifiers

Stefan Wezel

Explainable Machine Learning

October 23, 2020

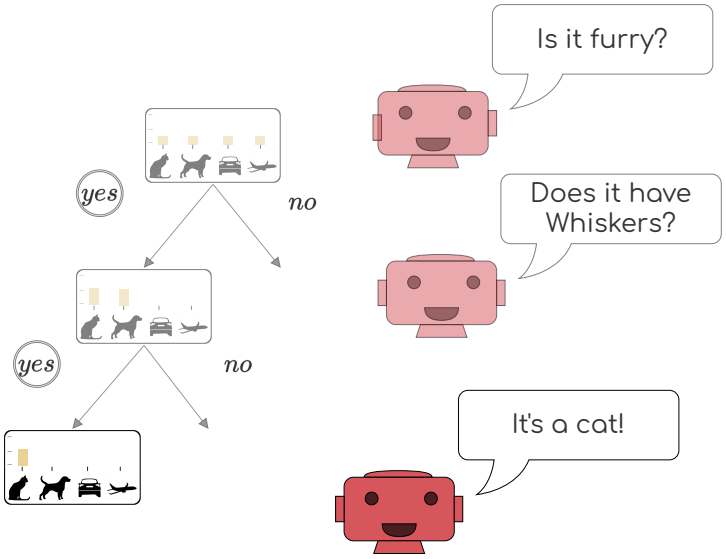
What?

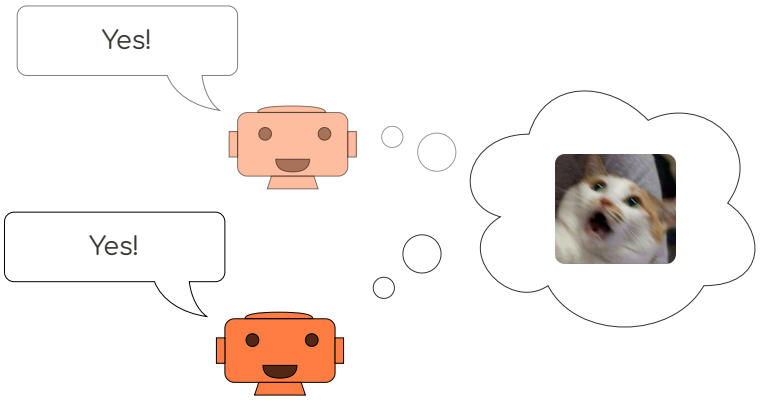
Setting

- There are a lot of architectures that perform great on image classification tasks
- Maybe, most prominently: ResNet
- However, they only yield a classification
- In many settings a classification is not worth much without the reasoning behind it

What?

What is a Recurrent Decision Tree Classifier?

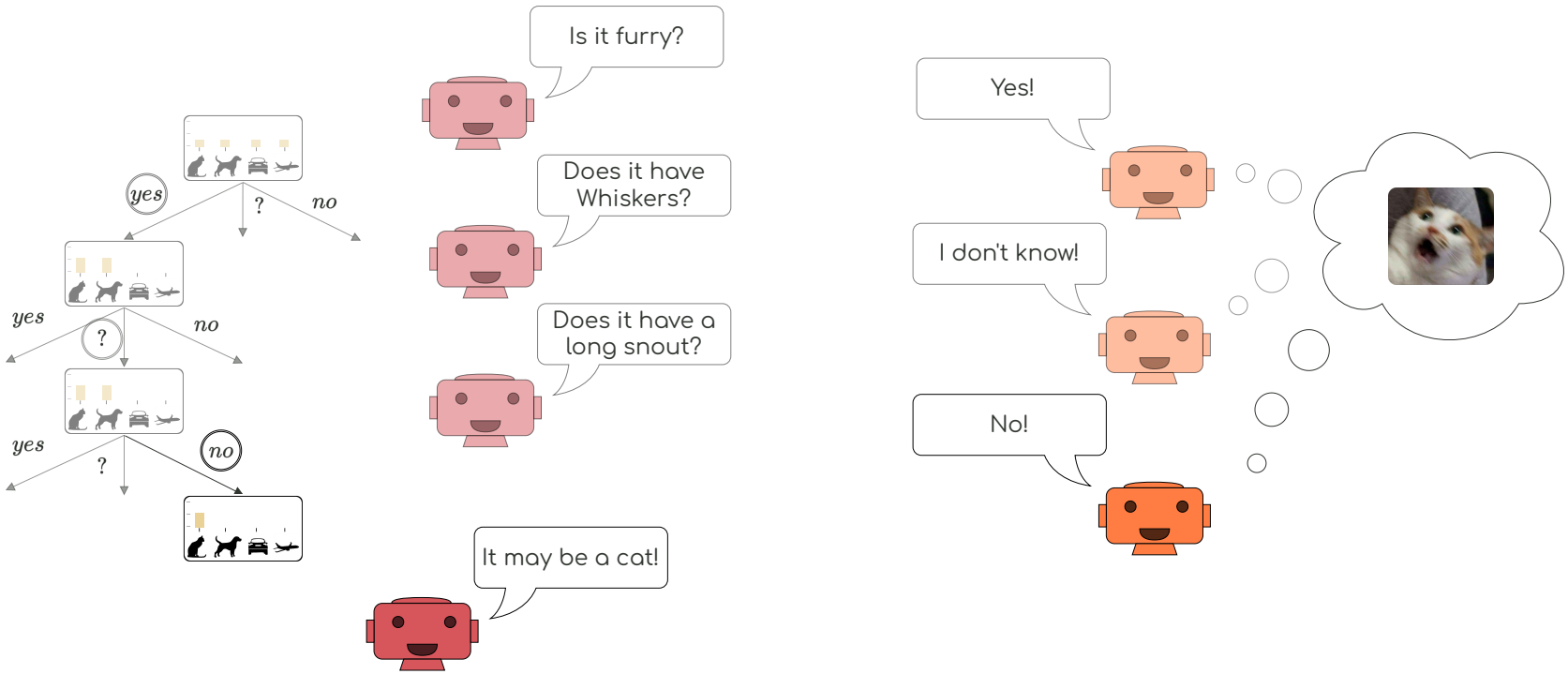




- Two communicating agents
- One (which can not see the data) is asking questions and one (which can see the data) is answering them
- The unfolding decision process is an interpretable tree

What?

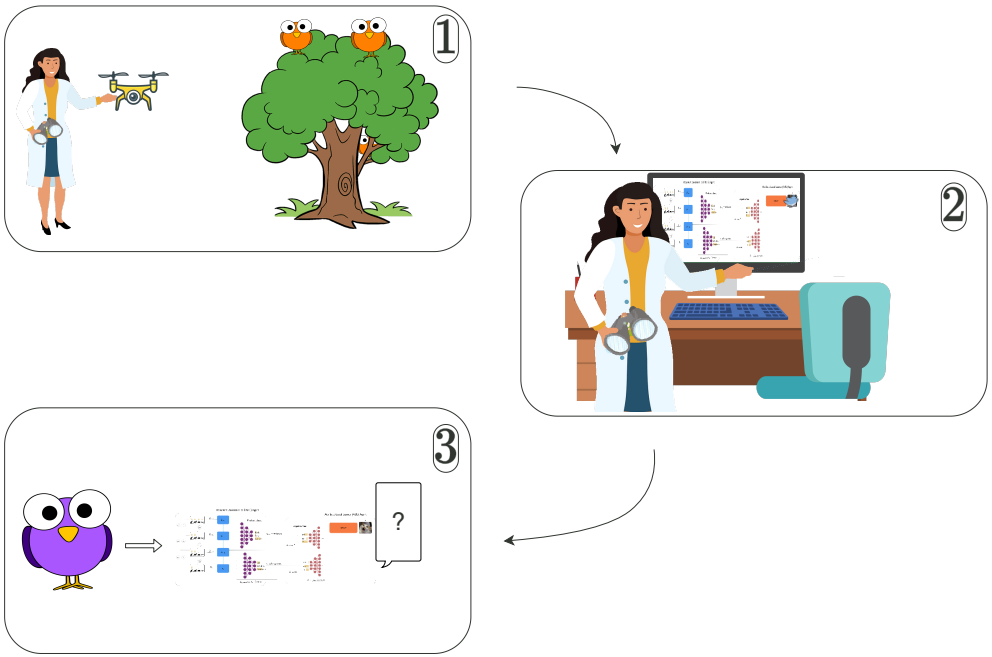
Uncertainty in a Recurrent Decision Tree Classifier



- The agent, seeing the data becomes aware of its uncertainty which is communicated to the other agent

Why do we need uncertainty?

A Practical Example...

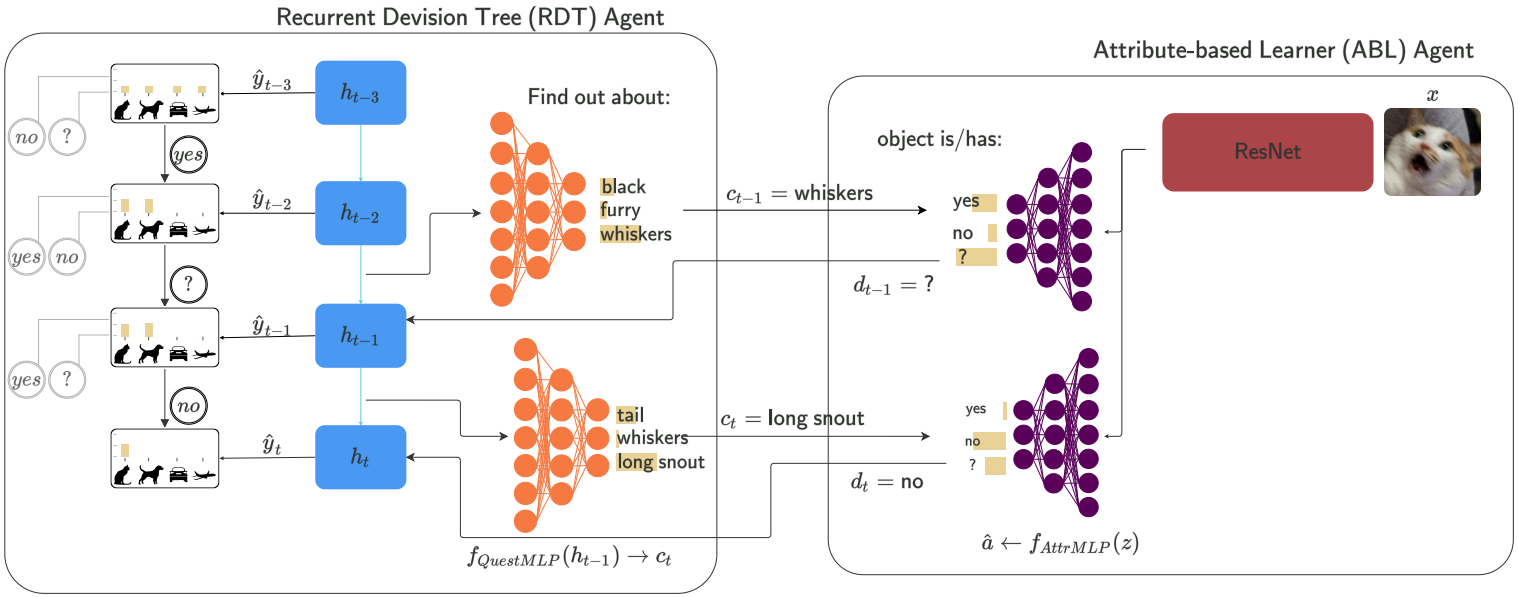


- The ornithologist is tasked to survey bird species, which she automates using a drone and computer vision software
- She uses our model to go through the vast amount of collected data
- Some bird species unknown to the model appear in the data. The model yields high uncertainty and the ornithologist can classify them manually

Introduction

- The RDTC model yields a decision tree as explanation, revealing the reasoning behind a classification
- The decision tree is created through communication of two agents
- We introduce uncertainty to this model
- Uncertainty allows the model to i.e. detect OOD examples

How?
Architecture



- The RDT can not see the images but only ask whether an attribute is present in the image
- The AbL can see the image and based on features answer the RDT's questions

Attribute-based Learner

Answering questions

- We extract features from a given image using a ResNet
- A MLP maps extracted features to 'yes-no' answers indicating absence/presence of attributes
- It returns a tensor with the of shape number of attributes $\times$ decision size
- We get a discrete answer from our AbL though applying

$$TempSoftmax(log\ \pi) = \frac{exp((log\ \pi_i)/\tau)}{\sum_{j=1}^{K} exp((log\ \pi_j)/\tau)} = d_t \text{ on } log\ \pi$$

  which are the logit values for either 'Yes', or 'No' per attribute.

- The TempSoftmax serves as differentiable approximation to a one argmax returning a one-hot encoding

Recurrent Decision Tree

Building a decision tree

- LSTM
  - Hidden state based on previous hidden states and new answers
- Explicit Memory
  - Stores all questions and corresponding answers
  - Its content is the decision tree
- $f_{ClassMLP}$
  - Make classification based on LSTM's hidden state
- $f_{QuestMLP}$
  - Find next question to ask based on LSTM's hidden state
  - Next question is the index the $f_{QuestMLP}$ poses
  - To turn logits into a discrete value, Gumbel softmax is used
  - This allows us to sample an index from the logits
  -

$$GumbelSoftmax(log\ \pi) = \frac{exp((log\ \pi_i + g_i)/\tau)}{\sum_{j=1}^{K} exp((log\ \pi_j + g_j)/\tau)}$$

Training

the two agents

- We optimize for class (and attribute accuracy)
-

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}[(1-\lambda)\mathcal{L}_{CE}(y,\hat{y}_t) + \lambda\mathcal{L}_{CE}(\alpha_{y,c_t},\hat{\alpha}_{c_t})]$$

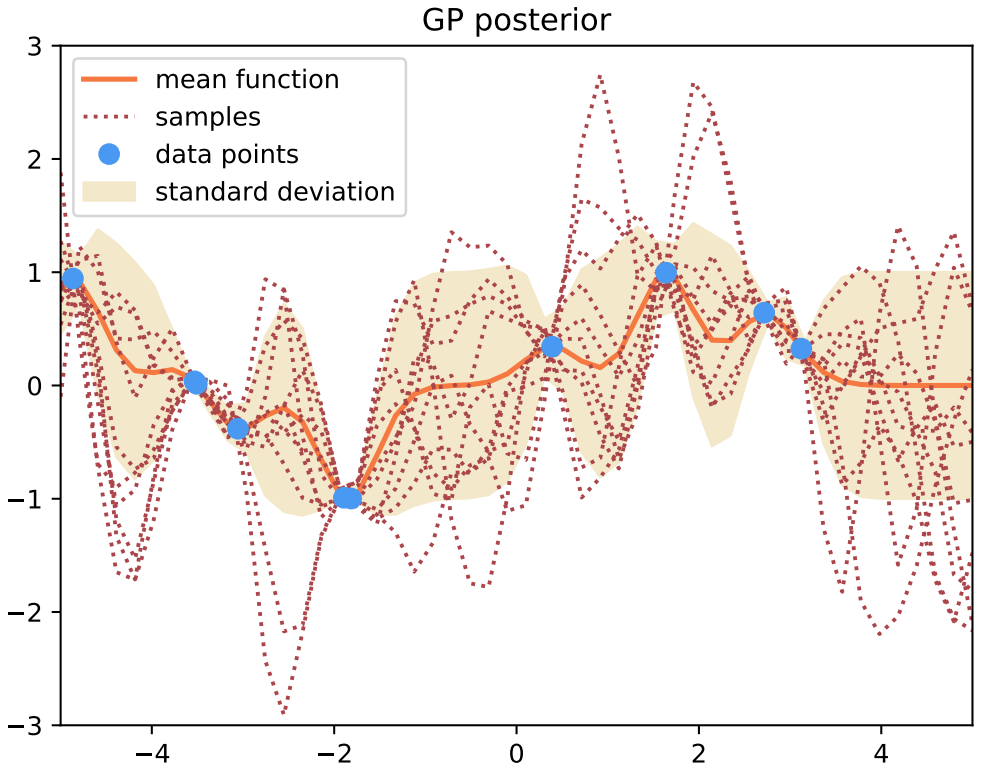
- λ can be used to balance the two loss terms
- For all of our experiments, we use $\lambda = 0.2$

The RDTC Model [1]

- The RDTC is a model consisting of two agents
- The AbL can see the data, and can answer the RDT's questions in advance
- The RDT can then choose attributes it wants to question and get an answer through indexing the AbL's answer
- They have the joint objective of maximizing class accuracy (and if $\lambda > 0$, attribute accuracy)

Background

A small excursion to Gaussian Processes (GP)

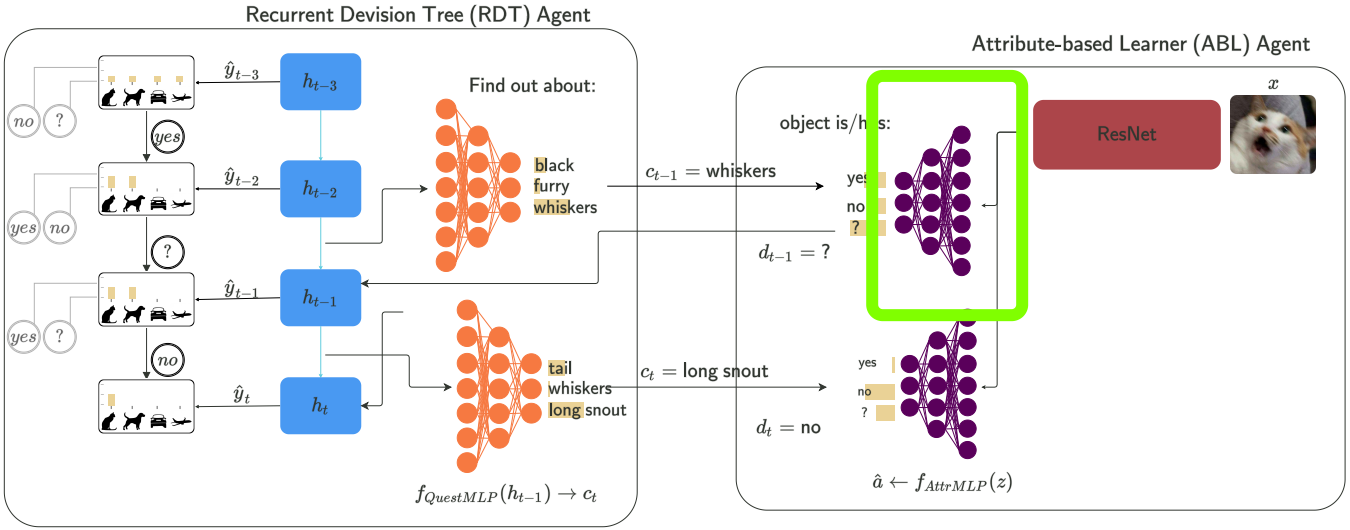


- Build on the notion that data points can be described by (infinitely) many functions
- A GP is a PDF over these functions
- Intuition: → a GP yields a probability for function values at any given index
- Parameterized by mean function and covariance function
- The variance resembles the model uncertainty where no data is given

- Neural network is a set of weighted linear functions, activated by some non-linearity
- If we put a PDF over each weight, we retrieve a finite GP
- So we can view a neural network as an approximation to a GP
- Computing the posterior over functions requires computing integrals
- Often intractable integrals are encountered, and methods of variational inference are used
  - We formulate GP objective as a minimization objective
  - For computing covariance function, we use Monte-Carlo integration
- This allows us to rewrite a GP's objective as to objective of a dropout neural network
- This in turn allows us to interpret variance arising from dropout as model uncertainty

How?

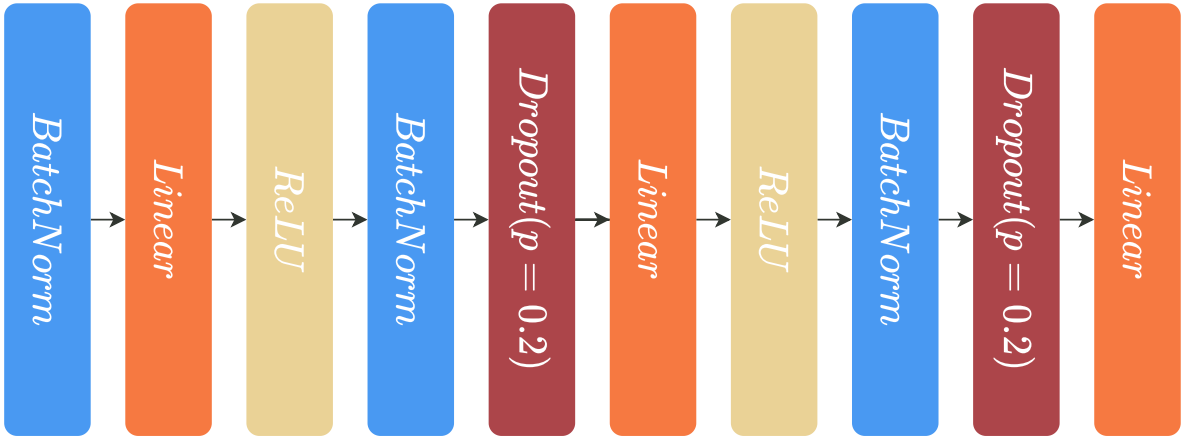
How do we get uncertainty information?



- We use this dropout uncertainty estimation to retrieve uncertainty
- We want the AbL to say 'I don't know in the case of high uncertainty
- However, we want to keep the ResNet as feature extractor
- Thus, make $f_{AttrMLP}$ a dropout MLP
- After extracting features, we do n forward passes and compute variance
- This corresponds to model uncertainty

Getting Uncertainty Information

Estimating Uncertainty in the AbL



- We include dropout layers in our $f_{AttrMLP}$
- We tested different configurations and a combination of batchnorm and dropout worked best

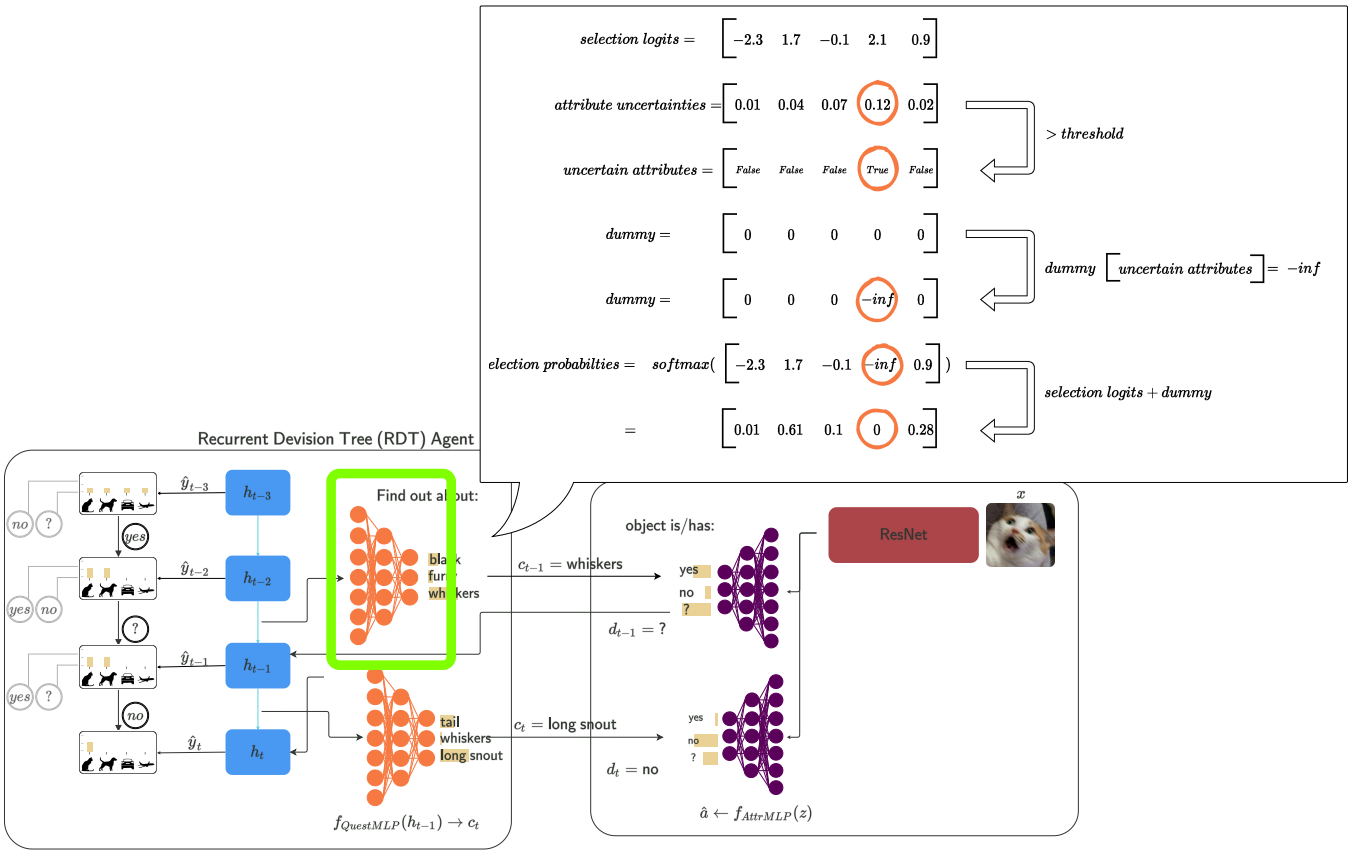
Using Uncertainty Information

as an inductive bias

- Now, we have an uncertainty estimate
- But now, we need a way to use it
- we use two different strategies
  - We prevent the model from asking questions regarding uncertain attributes → remRDTC
  - We give the model the ability to answer with 'I don't know' as extended vocabulary → extRDTC
- For both strategies, we need to make sure the model does not use any gradients coming from uncertain attributes
- This ensures that the uncertainty information only serves as an inductive bias and the model does not misuse it

Using Uncertainty Information

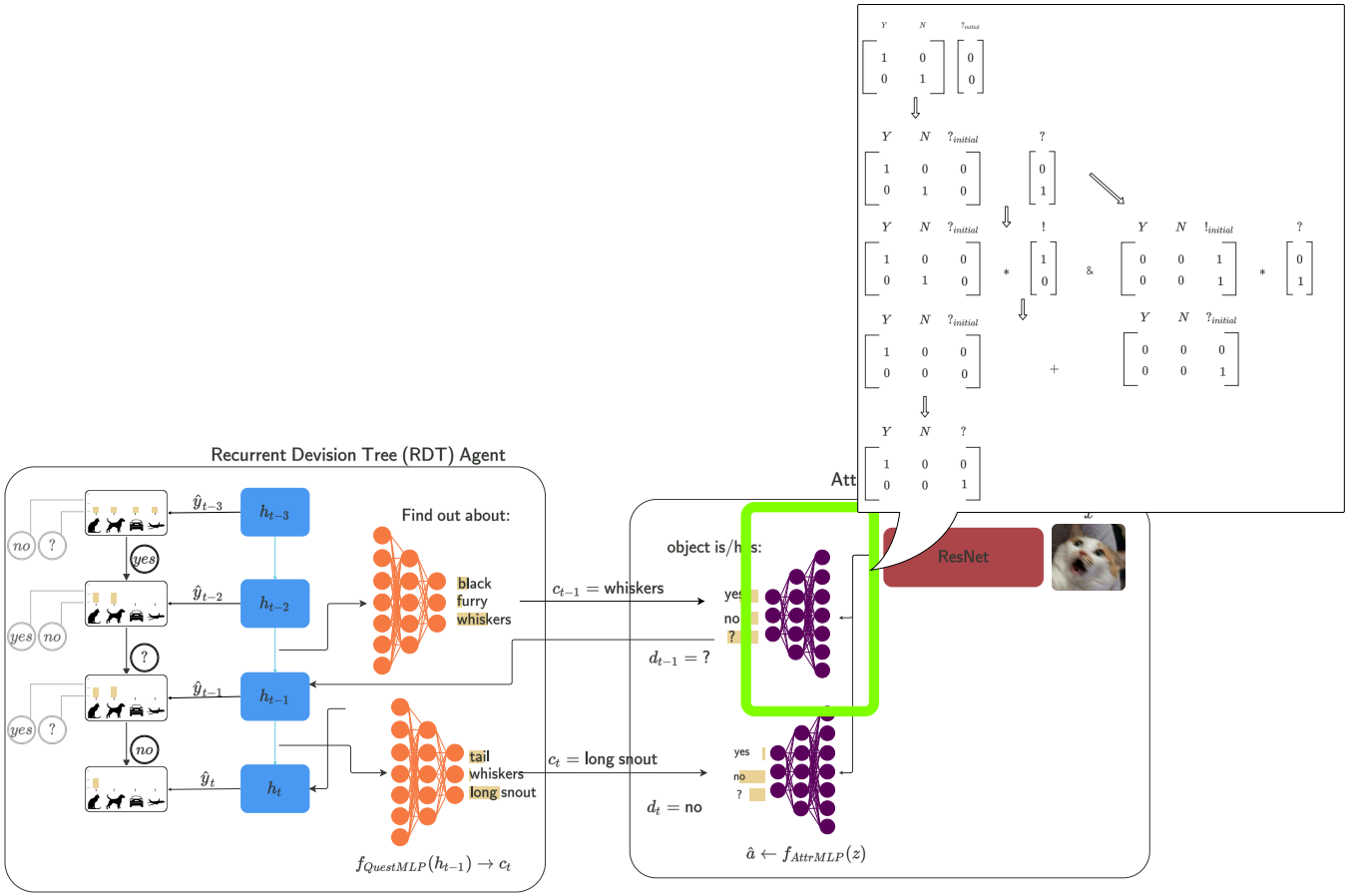
Removing uncertain attributes (remRDTC)



- The output from $f_{QuestMLP}$ is an index that indicates the attribute in question
- In case, an attribute is deemed uncertain by the AbL, we replace selection logits at those indices with $-\infty$
- This prevents the Gumbel softmax from picking uncertain attributes as index

Using Uncertainty Information

Extending the vocabulary (extRDTC)



- We create a binary uncertainty vector with 1s where uncertainty is above a given threshold
- This is appended to the initial answer and can be used by the RDT

Introducing Uncertainty to the RDTC Model

- We can interpret the variance arising from multiple forward passes in a dropout neural net as uncertainty [2]
- We do this dropout uncertainty estimation in our $f_{AttrMLP}$
- We use the retrieved uncertainty estimate for either preventing the RDT from asking questions regarding uncertain attributes or extent the AbL's vocabulary

Experiments

- Now, we can make our model aware of, and express its uncertainties
- We use this in our experiments to:
  - Investigate uncertainty and its relationship to other variables
  - Test our model on OOD data
  - Test the model's performance on benchmark datasets

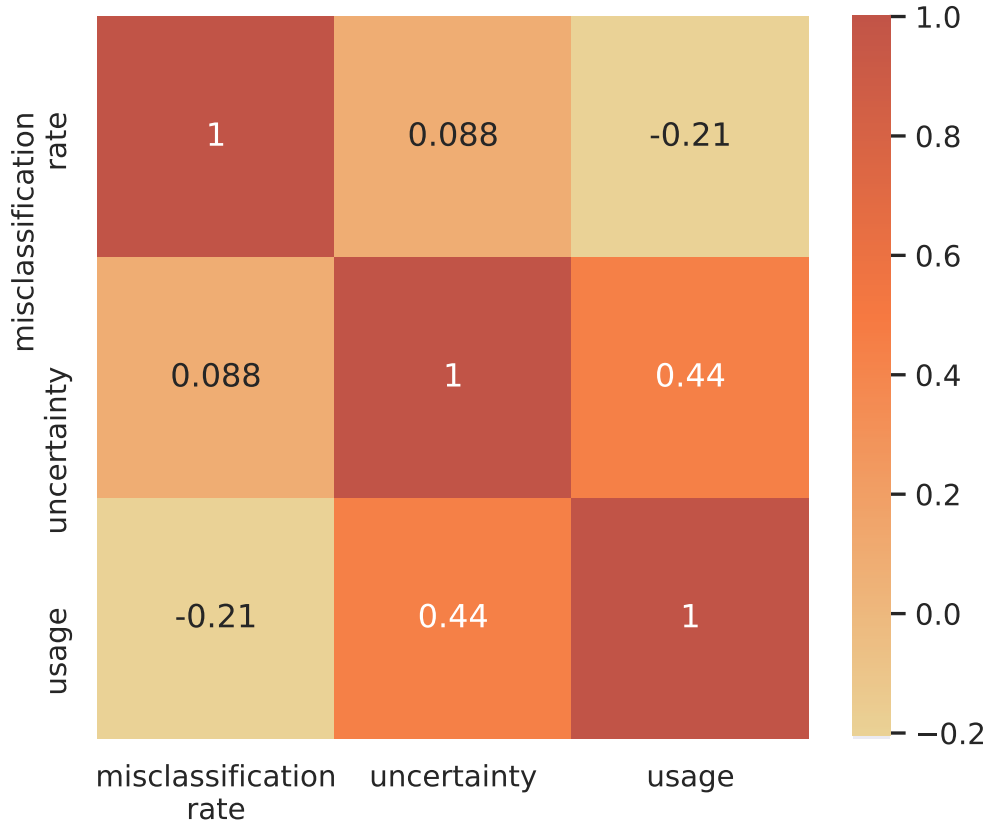
Experiments
Datasets

- Animals with Attributes 2 (AWA2)
  - medium size, coarse grained, image classification
- aPY
  - small size, coarse grained, image classification
- CUB
  - large size, fine grained, image classification

Experiments

Investigating Uncertainties

- We investigate covariance of misclassification rate, uncertainty, and usage of attributes in CUB

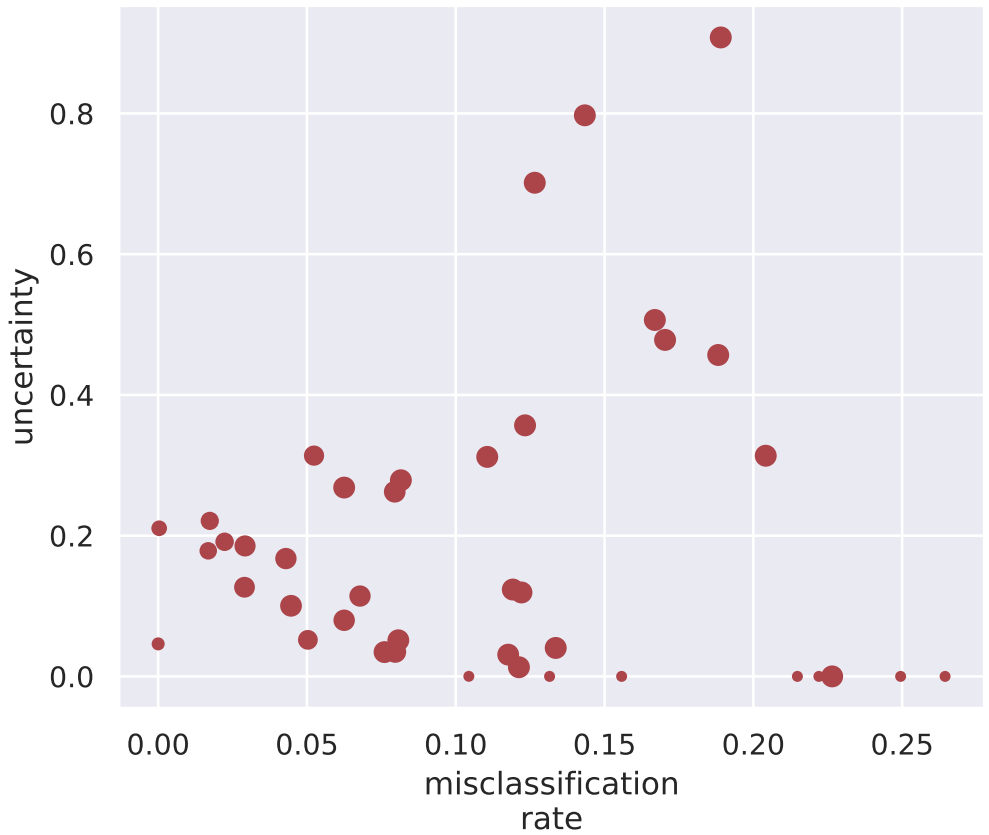


- Negative correlation between usage and misclassification rate
- Almost no correlation between uncertainty and misclassification rate

Experiments

Investigating Uncertainties

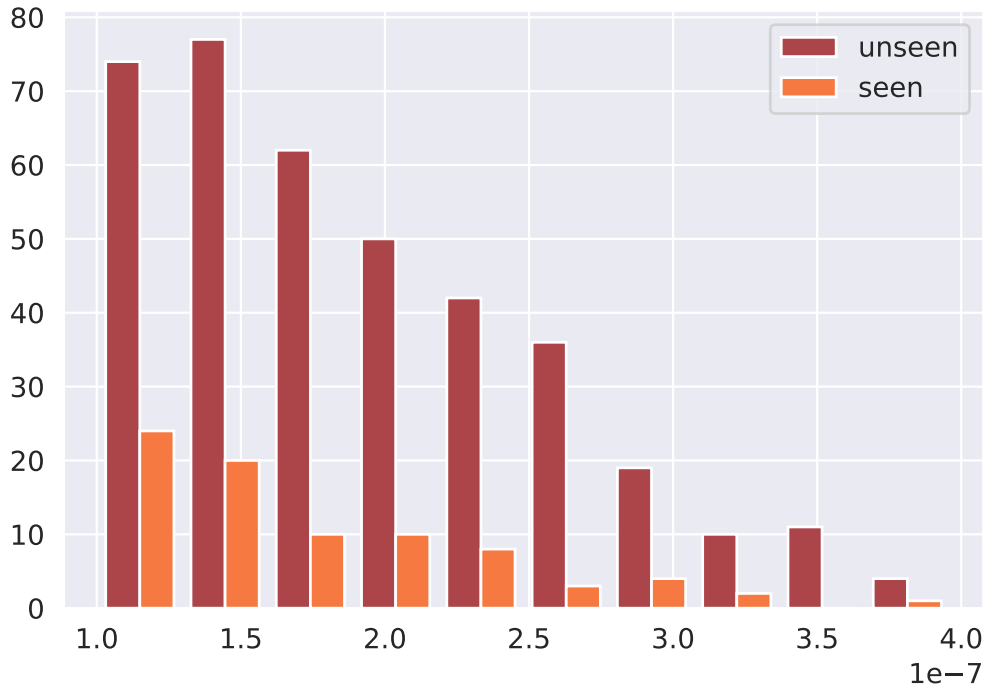
- Let's have a look at the actual values
- Uncertainty and misclassification rate (usage is represented through size)



- Uncertain attribute seem to be misclassified more often

Experiments
OOD Detection

- We test extRDTC in a zero shot setting (using CUB)
- A quarter of all classes were never seen during training
- We compare uncertainty values of seen and unseen classes



- Unseen classes have higher uncertainty values

Experiments

Comparison to other models

- Decision Tree (DT)
  - Use extracted ResNet features
  - Try to split until every leaf node corresponds to one class
- Explainable Decision Tree (XDT)
  - Same as DT, but using learned attribute representations instead of features
  - Attribute representations are created using $f_{AttrMLP}$ as head for ResNet
- dNDF [4]
  - Every node in the tree is a parametric differentiable function
  - Every route through the tree leads to a leaf node representing a class distribution
  - Objective is to learn the optimal route through the tree for each example
- aRDTC
  - RDTC with $\lambda > 0$
- ResNet
  - Not explainable
  - Trained on ImageNet and then fine-tuned for specific datasets

Experiments

Results on Benchmark Datasets

	AWA2	aPY	CUB
ResNet [3]	98.2± 0.0	85.1± 0.6	79.0± 0.2
DT	78.0± 0.4	64.3± 0.6	19.3± 0.3
dNDF[4]	97.6± 0.2	85.0± 0.6	73.8± 0.3
RDTC[1]	98.0± 0.1	85.7± 0.7	78.1± 0.2
XDT	73.9± 0.9	59.9± 1.5	4.9± 1.3
aRDTC[1]	98.6	86.1	**77.9**± 0.6
remRDTC(ours)	**98.7**	**86.4**	77.7
extRDTC(ours)	**98.7**	85.4	77.8

Experiments

Results on Benchmark Datasets

	aRDTC [1]	Random Baseline	remRDTC	extRDTC
AWA2				
Class	**98.6**	98.5	98.7	98.7
Attribute	80.4	84.6	**87.5**	82.31
aPY				
Class	86.1	**86.5**	86.4	85.4
Attribute	86.4	86.2	**87.6**	87.1
CUB				
Class	**77.9**	76.8	77.7	77.8
Attribute	68.6	70.0	77.4	**82.6**

- We have seen that the RDTC uses attributes that it rarely misclassifies
- Unseen classes yield higher uncertainty
- Both our proposed strategies perform well on benchmark datasets

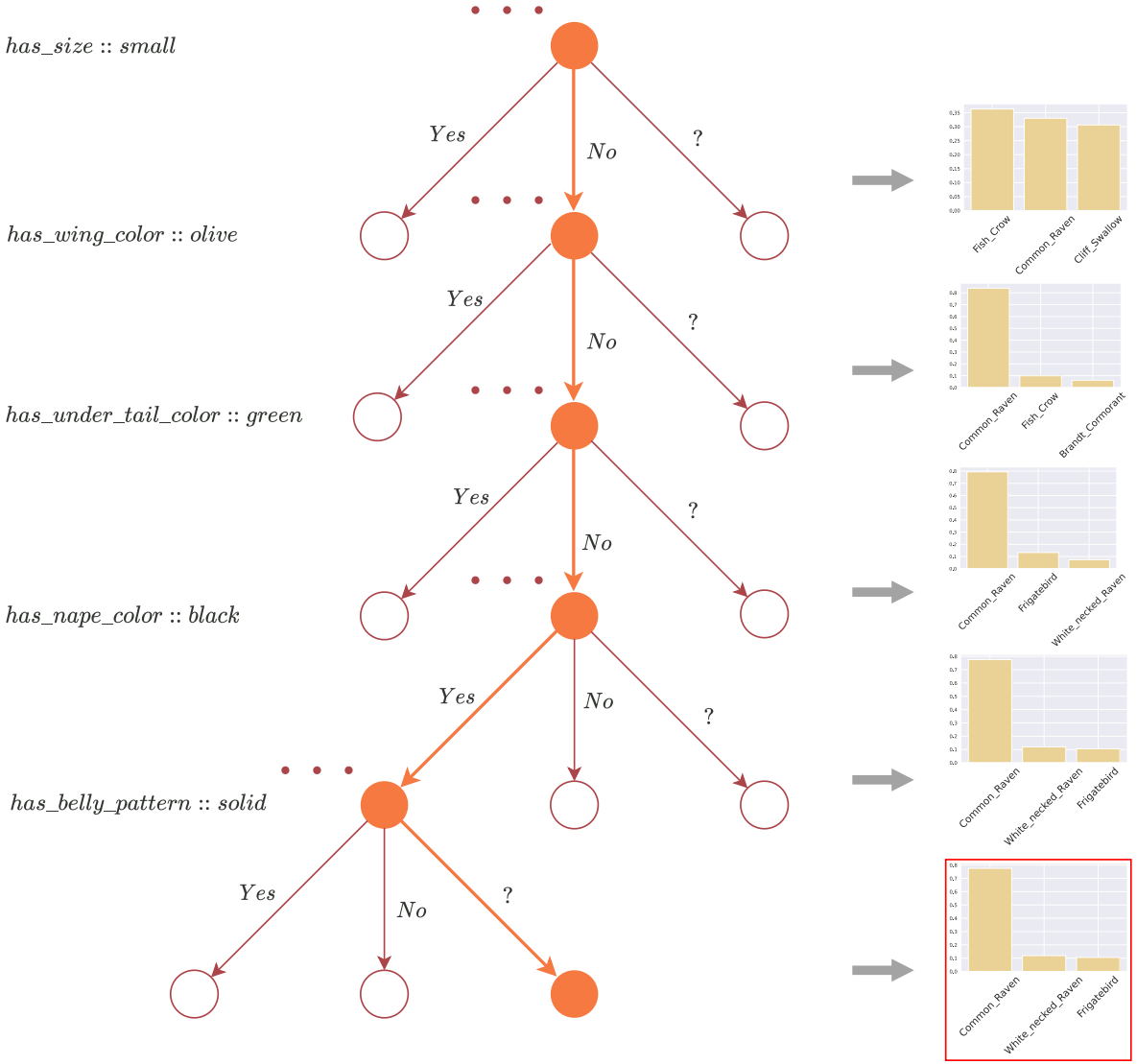
Discussion

Looking back...

- The right kind of uncertainty?
  - We only consider model uncertainty (epistemic uncertainty)
  - Uncertainty arising from noise or occlusions in the data is not considered
- Other methods besides dropout uncertainty estimation
  - Unfortunately, they are often computationally expensive
  - Uncertainty estimation in neural networks is challenging in itself
- Beyond attribute uncertainty
  - Estimate uncertainty in other parts of the model (i.e. $f_{ClassMLP}$)

Conclusions

A qualitative Example



Conclusions

- We estimate uncertainty in the RDTC model
- This uncertainty is used in the remRDTC and extRDTC strategies
- We investigate uncertainty and its relationship to other variables
- We show that OOD examples yield high uncertainty
- The RDTC models using uncertainty information achieve state of the art accuracy on benchmark image classification tasks

[1] S. Alaniz and Z. Akata. Explainable observer-classifier for explainable binary decisions. arXiv preprint arXiv:1902.01780, 2019.

[2] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059, 2016.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[4] P. Kontschieder, M. Fiterau, A. Criminisi, and S. Rota Bulo. Deep neural decision forests. In Proceedings of the IEEE international conference on computer vision, pages 1467–1475, 2015.