GHENT UNIVERSITY

FACULTY OF BIOSCIENCE ENGINEERING

Efficient Algorithms for Set-Valued Prediction in Classification

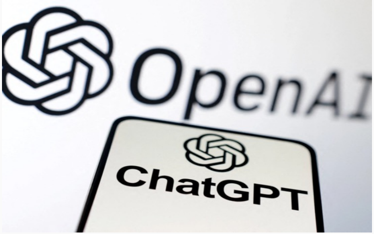
ir. Thomas Mortier

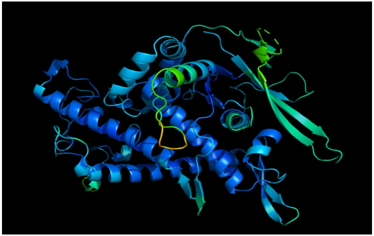
Public PhD defence

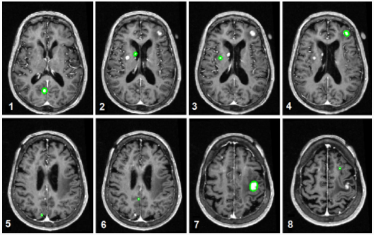
June 23, 2023

Machine learning

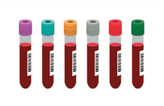
“Machine learning is a subfield of artificial intelligence that gives computers the ability to learn without explicitly being programmed”





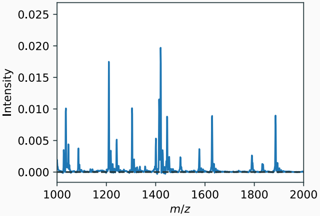


Bacterial species identification using MALDI-TOF MS

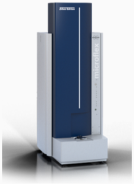


Blood sample

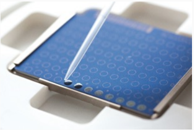
Bacterial culture



Spectrum

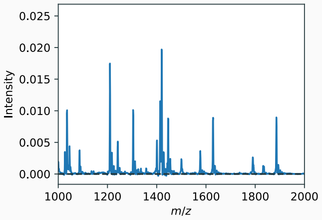


MALDI-TOF MS

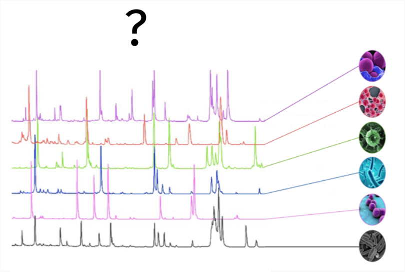


Target plate

Bacterial species identification using MALDI-TOF MS



Spectrum



Reference spectra

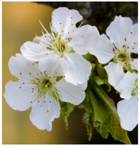
Plant species identification using images


C. roseus


C. trichophyllus


V. major


V. minor


P. avium


P. serrulata


R. canina


R. regosa

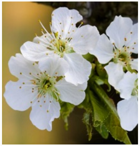
Plant species identification using images

C. roseus

C. trichophyllus

V. major

V. minor



P. avium

P. serrulata

R. canina

R. regosa

Plant species identification using images

V. major

V. minor

?

Overview

1. Introduction to probabilistic classification
2. Set-valued prediction in classification
3. Set-valued prediction in hierarchical classification
4. Conclusion

Introduction to probabilistic classification

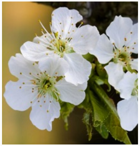
Classification


C. roseus


C. trichophyllus


V. major


V. minor


P. avium


P. serrulata


R. canina


R. regosa

Probabilistic classification

Plug-in classifier

1. Training: learn a probabilistic classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the class with the highest probability

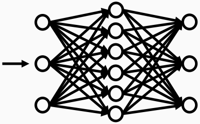
Training problem


V. major

Input x, y

Training problem

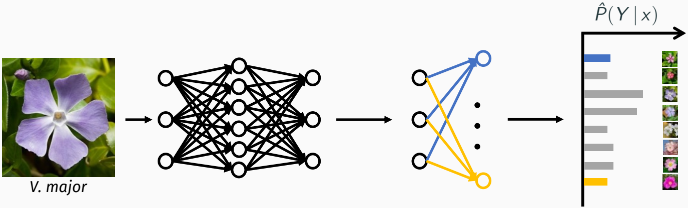
V. major



Input x, y

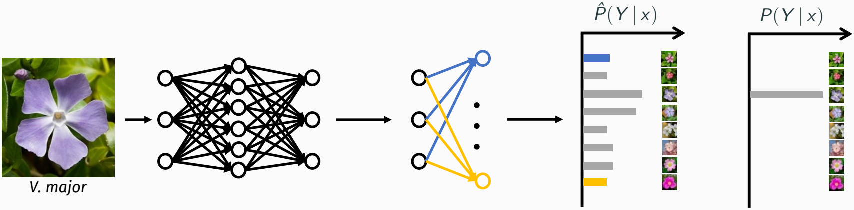
Feature extractor

Training problem



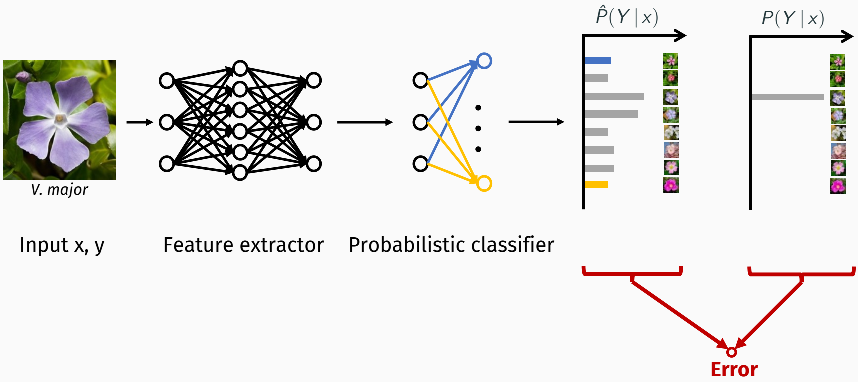
Input x, y Feature extractor Probabilistic classifier

Training problem

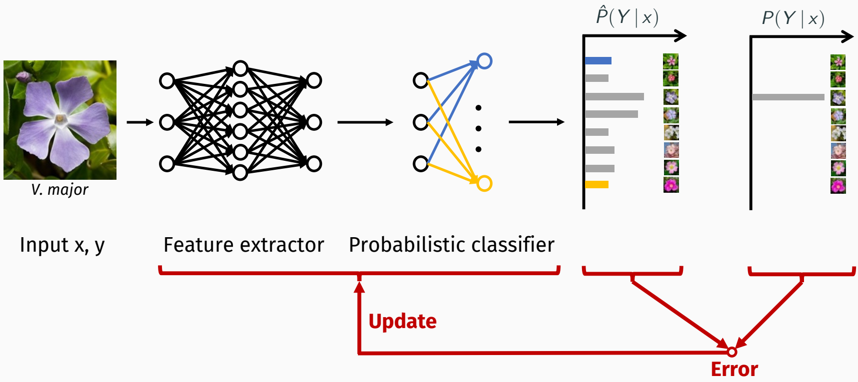


Input x, y   Feature extractor   Probabilistic classifier

Training problem



Training problem



Probabilistic classification

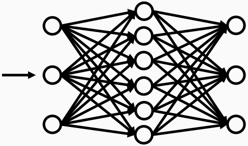
Plug-in classifier

1. Training: learn a probabilistic classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the class with the highest probability

Inference problem

Input x

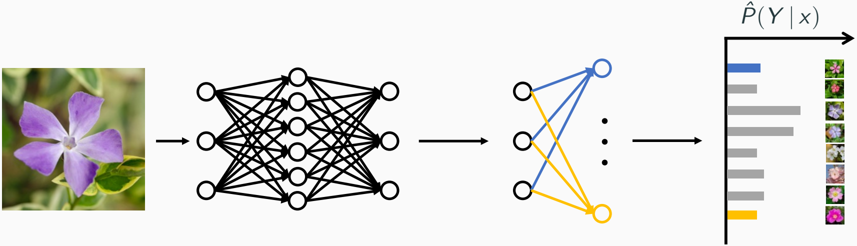
Inference problem



Input x

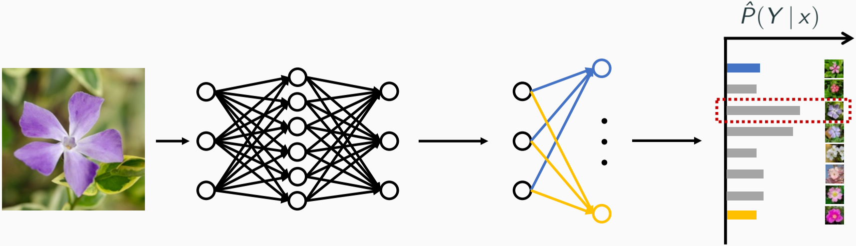
Feature extractor

Inference problem



Input x   Feature extractor   Probabilistic classifier

Inference problem



Input x

Feature extractor

Probabilistic classifier

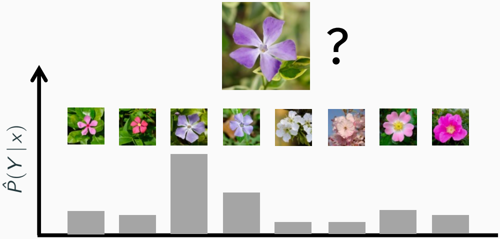
Vinca major

Uncertainty in probabilistic classification

Two different sources of uncertainty:

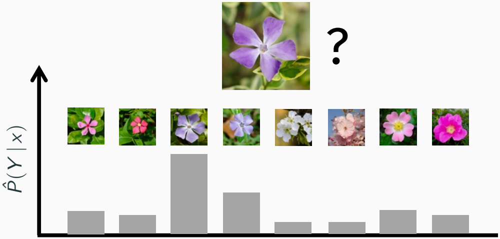
- $\rightarrow$ Aleatoric uncertainty (irreducible), due to an unknown non-deterministic relationship between inputs and labels
- $\rightarrow$ Epistemic uncertainty (reducible), due to a lack of knowledge about the true relationship between inputs and labels

Aleatoric uncertainty

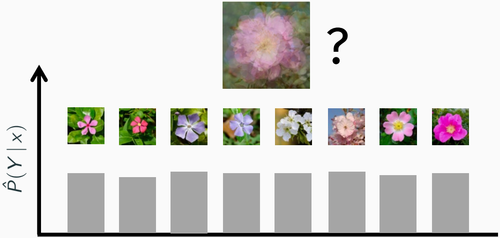


Low

Aleatoric uncertainty



Low



High

Set-valued prediction in classification

Contributions

- Novel decision-theoretic framework for set-valued prediction in classification based on set-based utility maximization
- Efficient inference algorithm for classification problems with a large number of classes

Set-valued prediction

- Find a set-valued classifier that predicts sets of classes in case of high aleatoric uncertainty
  - Plug-in classifier $\rightarrow$ *Vinca minor*
  - Set-valued classifier $\rightarrow$ $\{$*Vinca minor*, *Vinca major*$\}$
- Search is guided by a set-based utility function $u(y, \hat{Y})$ with focus on
  - Recall: the true class y is in the predicted set $\hat{Y}$
  - Precision: the set size $|\hat{Y}|$ is not too large

Vinca major

Set-based utility functions

A general family of set-based utility functions:

$$u(y, \hat{Y}) = \begin{cases} 0\,, & \text{if true class } y \text{ is not in set } \hat{Y} \\ g(|\hat{Y}|)\,, & \text{if true class } y \text{ is in set } \hat{Y} \end{cases}$$

With properties:

1. $g(1) = 1$
2. $g(1), \ldots, g(K)$ is a decreasing sequence

Set-based utility functions

A general family of set-based utility functions:

$$u(y, \hat{Y}) = \begin{cases} 0\,, & \text{if true class } y \text{ is not in set } \hat{Y} \\ g(|\hat{Y}|)\,, & \text{if true class } y \text{ is in set } \hat{Y} \end{cases}$$

With properties:

1. $g(1) = 1$
2. $g(1), \ldots, g(K)$ is a decreasing sequence

Some examples from the literature:

$$g_P(|\hat{Y}|) = 1/|\hat{Y}|\,, \quad g_{F\beta}(|\hat{Y}|) = \frac{1+\beta^2}{|\hat{Y}|+\beta^2}$$

Decision-theoretic approach

Plug-in set-valued classifier for u

1. Training: learn a probabilistic classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the set with the highest expected utility $U(\hat{Y}, \hat{P}, u) = g(|\hat{Y}|)\hat{P}(\hat{Y} \mid \boldsymbol{x})$

Decision-theoretic approach

Plug-in set-valued classifier for u

1. Training: learn a probabilistic classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the set with the highest expected utility $U(\hat{Y}, \hat{P}, u) = g(|\hat{Y}|)\hat{P}(\hat{Y} \mid \boldsymbol{x})$

Inference problem: we need to consider 2^K sets!

Decision-theoretic approach

Plug-in set-valued classifier for u

1. Training: learn a probabilistic classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the set with the highest expected utility $U(\hat{Y}, \hat{P}, u) = g(|\hat{Y}|)\hat{P}(\hat{Y} \mid \boldsymbol{x})$

Inference problem: we need to consider 2^K sets!

$K = 300 \rightarrow$ **more sets than atoms in the universe!**

Example for u_{F1}

$\hat{P}(Y \mid$  $)$

$\hat{P}(Y \mid$  $) \xrightarrow{\text{sort}}$

							
0.50	0.40	0.03	0.02	0.02	0.01	0.01	0.01

Example for u_{F1}

$\hat{P}(Y \mid$  $)$ $\xrightarrow{\text{sort}}$

							
0.50	0.40	0.03	0.02	0.02	0.01	0.01	0.01

$U(\{$  $\}, \hat{P}, u_{F1}) = 0.500$

Example for u_{F1}

$\hat{P}(Y \mid$  $)$ $\xrightarrow{\text{sort}}$

0.50 0.40 0.03 0.02 0.02 0.01 0.01 0.01

$U(\{$  $\}, \hat{P}, u_{F1}) = 0.500$

$U(\{$ ,  $\}, \hat{P}, u_{F1}) = 0.600$

Example for u_{F1}

$\hat{P}(Y \mid$  $)$ $\xrightarrow{\text{sort}}$

							
0.50	0.40	0.03	0.02	0.02	0.01	0.01	0.01

$U(\{$  $\}, \hat{P}, u_{F1}) = 0.500$

$U(\{$  ,  $\}, \hat{P}, u_{F1}) = 0.600$

$U(\{$  ,  ,  $\}, \hat{P}, u_{F1}) = 0.465$

Example for u_{F1}

$\hat{P}(Y \mid$  $)$ $\xrightarrow{\text{sort}}$

							
0.50	0.40	0.03	0.02	0.02	0.01	0.01	0.01

$U(\{$  $\}, \hat{P}, u_{F1}) = 0.500$

$U(\{$  ,  $\}, \hat{P}, u_{F1}) = 0.600$

STOP! $U(\{$  ,  ,  $\}, \hat{P}, u_{F1}) = 0.465$

Example for u_{F1}

$\hat{P}(Y \mid$  $)$ $\xrightarrow{\text{sort}}$        

0.50	0.40	0.03	0.02	0.02	0.01	0.01	0.01

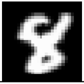
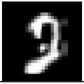
$\hat{Y}_{u_{F1}} = \{$ ,  $\}$

$U(\{$  $\}, \hat{P}, u_{F1}) = 0.500$

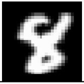
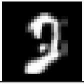
$U(\{$ ,  $\}, \hat{P}, u_{F1}) = 0.600$

STOP! $U(\{$ , ,  $\}, \hat{P}, u_{F1}) = 0.465$

Results on MNIST (K=10) and VOC 2006 (K=10)

		
u_{F5}	$\{\underline{8}\}$	$\{9, 3, \underline{2}, 7, 8, 1\}$
u_{F1}	$\{\underline{8}\}$	$\{9, 3, \underline{2}\}$

Results on MNIST (K=10) and VOC 2006 (K=10)

		
u_{F5}	$\{\underline{8}\}$	$\{9, 3, \underline{2}, 7, 8, 1\}$
u_{F1}	$\{\underline{8}\}$	$\{9, 3, \underline{2}\}$

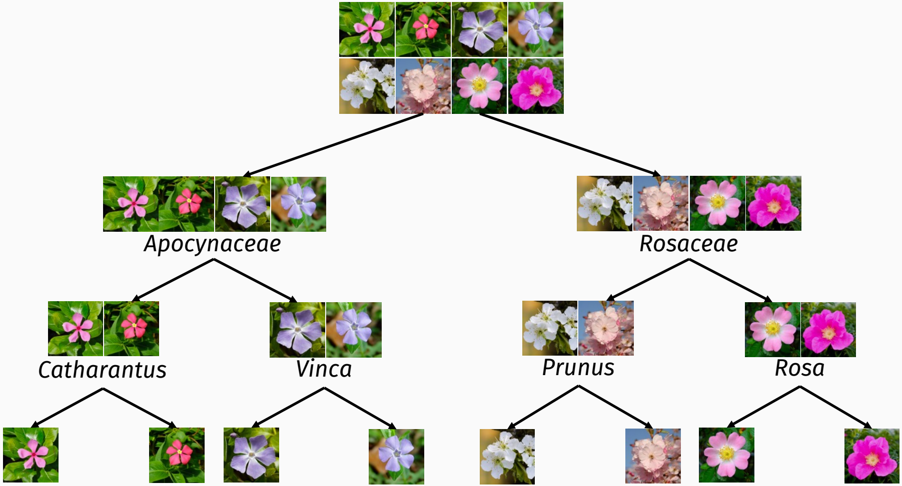
Top-5 = *{sheep, <u>cow</u>, horse, car, motorbike}*
$\hat{Y}_{u_{F1}}$ = *{sheep, <u>cow</u>}*

Set-valued prediction in hierarchical classification

Contributions

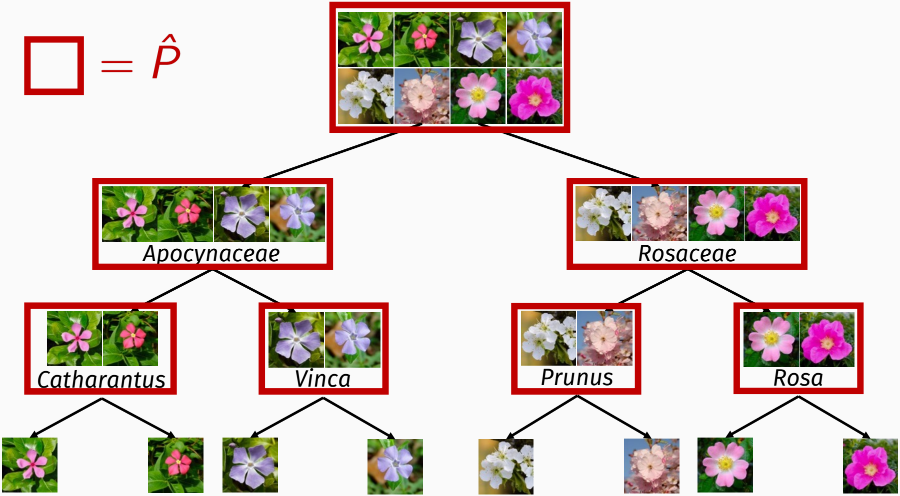
- Novel decision-theoretic framework for set-valued prediction in hierarchical classification
- Restriction on the representation complexity and size of predictions
- Efficient inference algorithm for classification problems with a large number of classes

Hierarchical classification

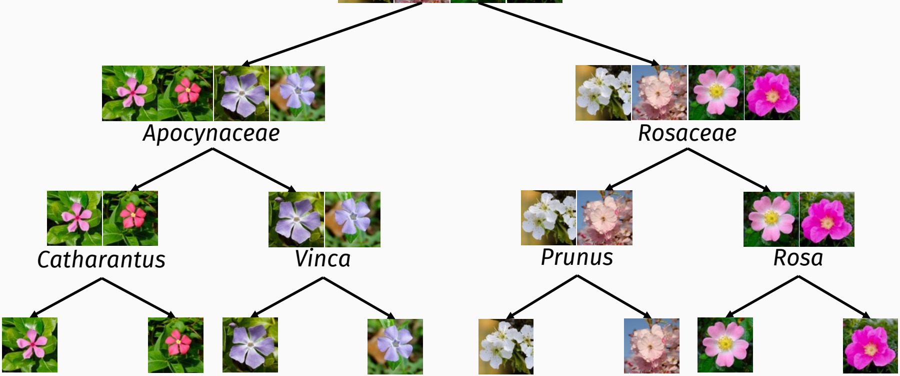


Top-down classifier

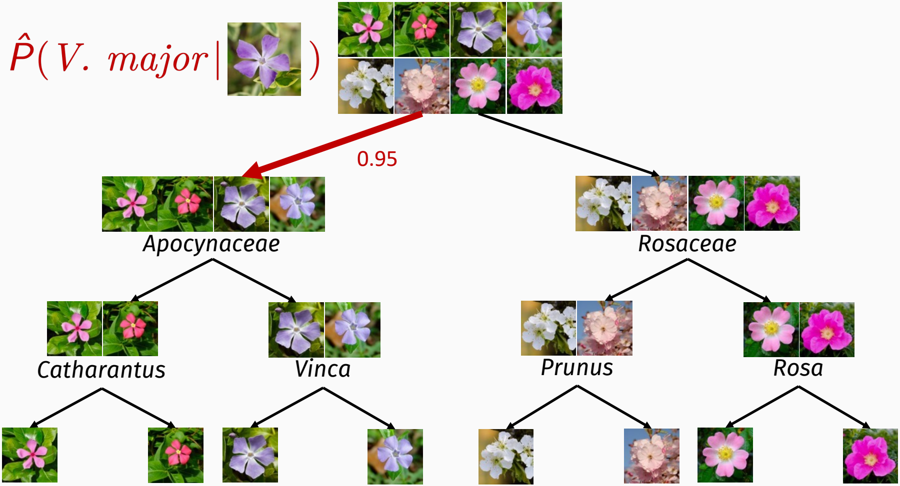
$\square = \hat{P}$



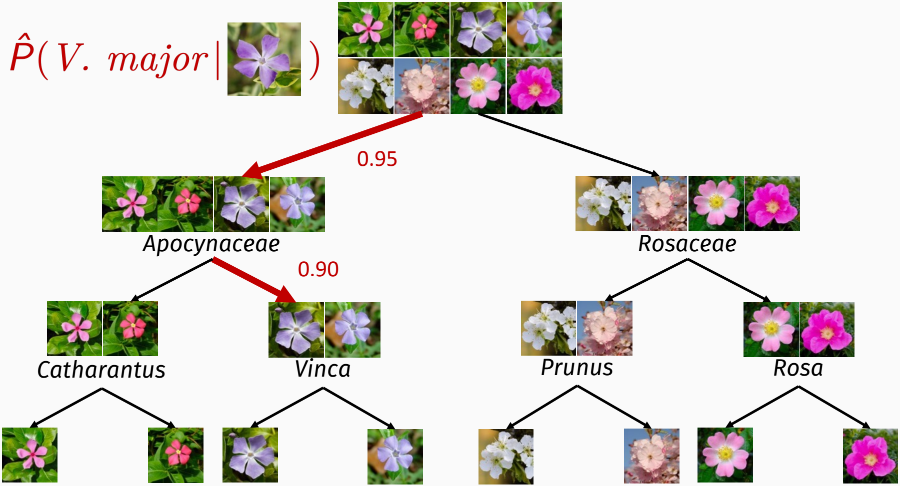
$\hat{P}(V.\ major|$  $)$



Chain rule of probability

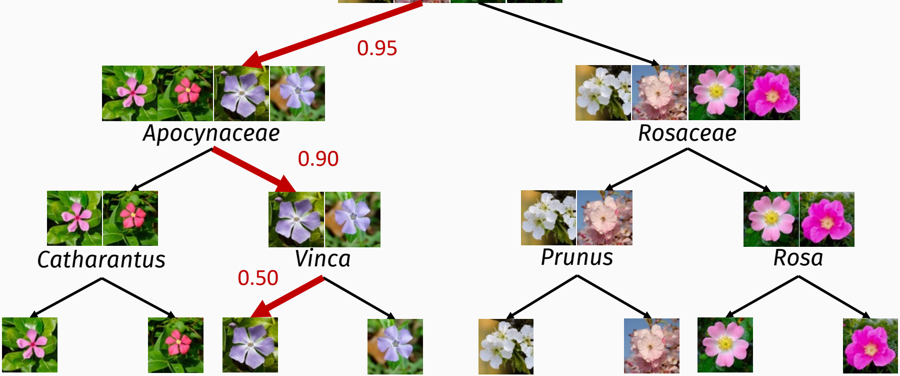


Chain rule of probability



Chain rule of probability

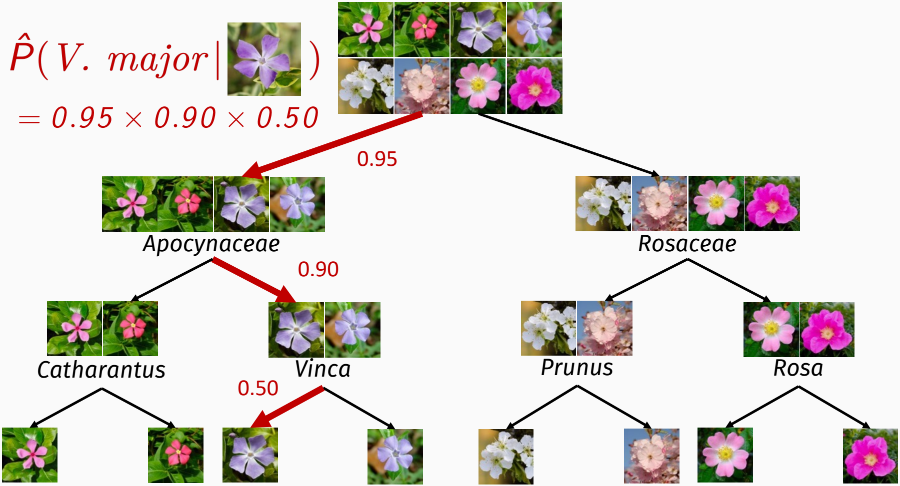
$\hat{P}(V.\ major|$  $)$



Chain rule of probability

$\hat{P}(V.\ major|\ \ )$

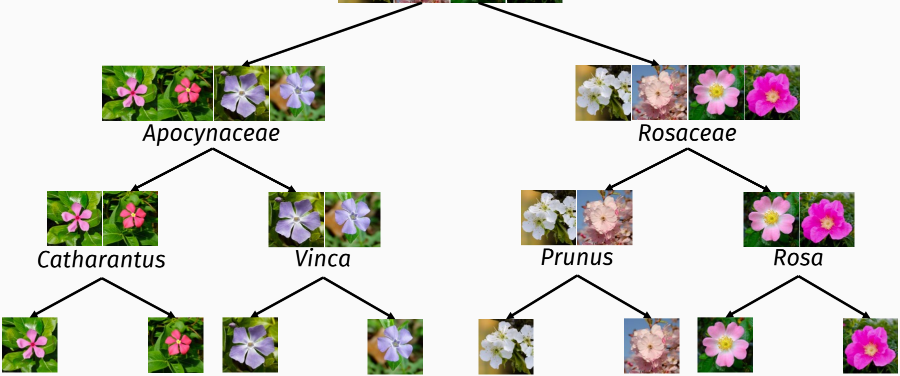
$= 0.95 \times 0.90 \times 0.50$



Restricted set-valued prediction

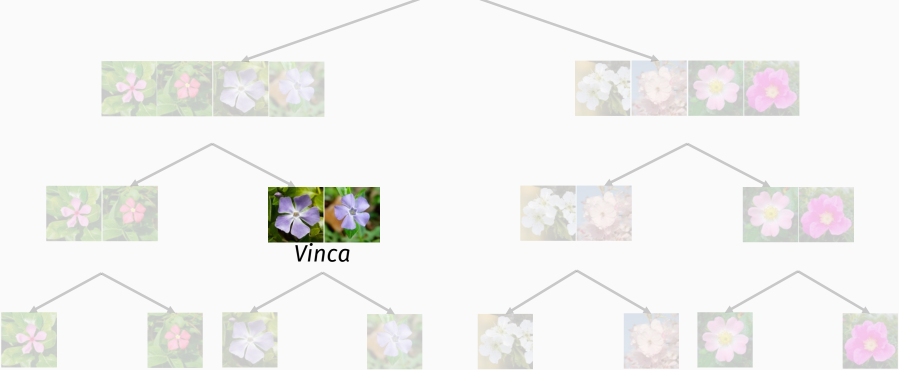
- Find a set-valued classifier that predicts sets of classes in case of high aleatoric uncertainty
- Restriction on the so-called representation complexity of a prediction $R_{\mathcal{T}}(\hat{Y}) \leq r$
    - $r = 1$: traditional set-valued prediction in hierarchical classification (i.e., predictions correspond to nodes in the hierarchy)
    - $r \rightarrow K$: unrestricted set-valued prediction (i.e., as discussed in the previous part)
- Restriction on the size of the prediction $|\hat{Y}| \leq k$

Example of $R_{\mathcal{T}}(\hat{Y}) = 1$ and $|\hat{Y}| = 2$

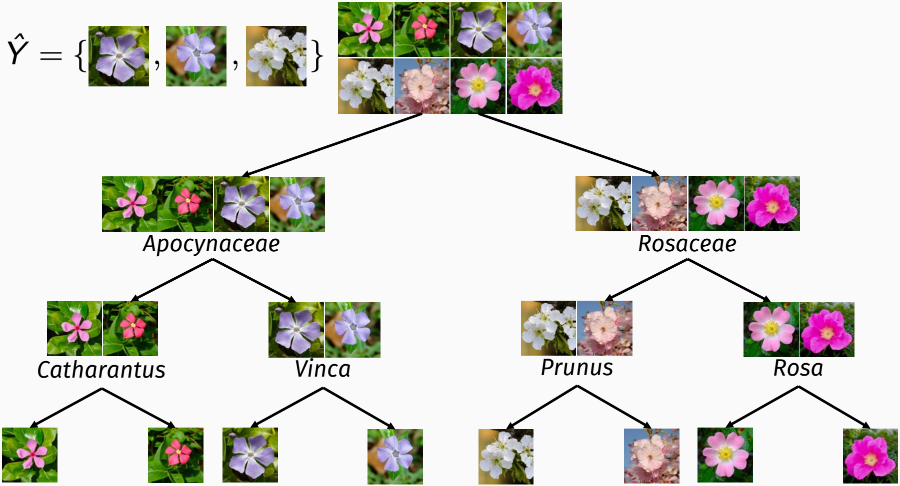


Example of $R_{\mathcal{T}}(\hat{Y}) = 1$ and $|\hat{Y}| = 2$

$\hat{Y} = \{$  ,  $\}$

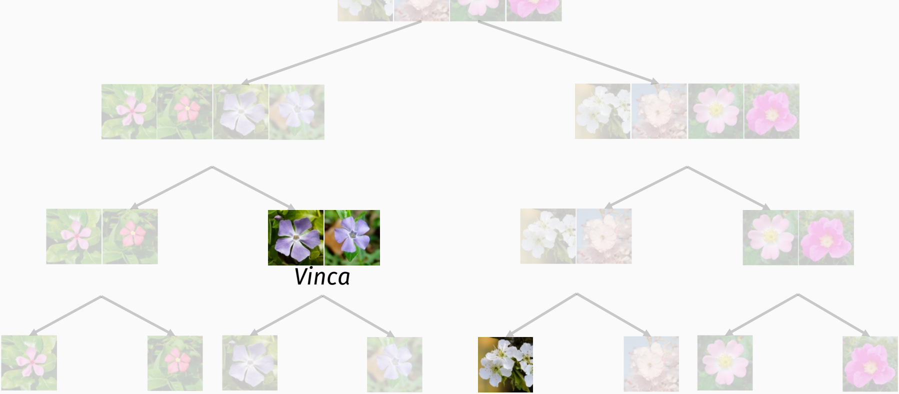


Example of $R_{\mathcal{T}}(\hat{Y}) = 2$ and $|\hat{Y}| = 3$



Example of $R_{\mathcal{T}}(\hat{Y}) = 2$ and $|\hat{Y}| = 3$

$\hat{Y} = \{$  ,  ,  $\}$



Decision-theoretic approach

Plug-in set-valued classifier for r and k

1. Training: learn a top-down classifier $\hat{P}$ on a training set
2. Inference: for any given input $\boldsymbol{x}$, predict the set with the highest probability, with a restriction on
   - the representation complexity: $R_{\mathcal{T}}(\hat{Y}) \leq r$
   - the set size: $|\hat{Y}| \leq k$

Recursive tree search (RTS)

- Exploit hierarchical structure $\rightarrow$ recursive tree search algorithm
- Use a priority queue for storing visited nodes in decreasing order of probability
- Solutions are recursively explored $\rightarrow$ stops when maximum representation complexity r is reached
- Only a limited number of solutions need to be visited in order to find the optimal solution

Example for r=2 and k=3

$$\hat{Y}_{r,k} = \{\}$$

Q = {}

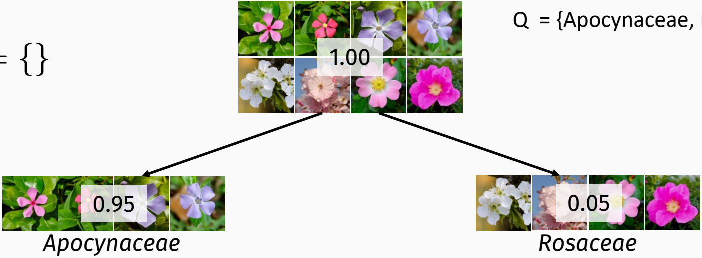
Example for r=2 and k=3

$$\hat{Y}_{r,k} = \{\}$$

Q = {}

Example for r=2 and k=3

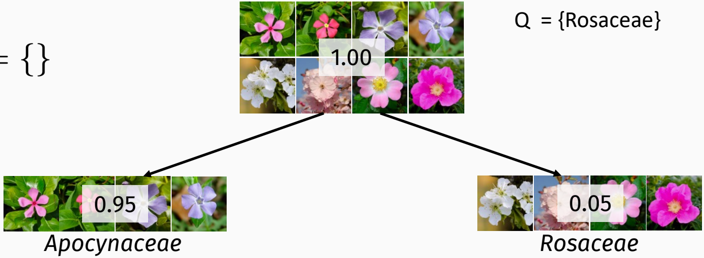
$$\hat{Y}_{r,k} = \{\}$$



Q = {Apocynaceae, Rosaceae}

Example for r=2 and k=3

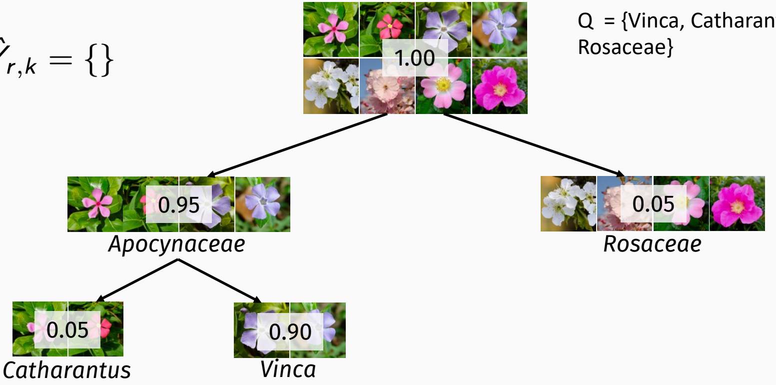
$$\hat{Y}_{r,k} = \{\}$$



Q = {Rosaceae}

Example for r=2 and k=3

$\hat{Y}_{r,k} = \{\}$



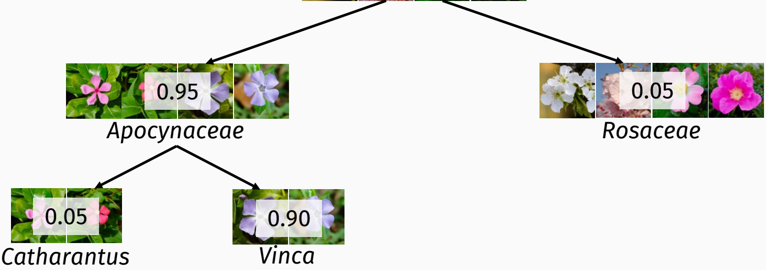
Q = {Vinca, Catharantus, Rosaceae}

Example for r=2 and k=3

$\hat{Y}_{r,k} = \{$  ,  $\}$

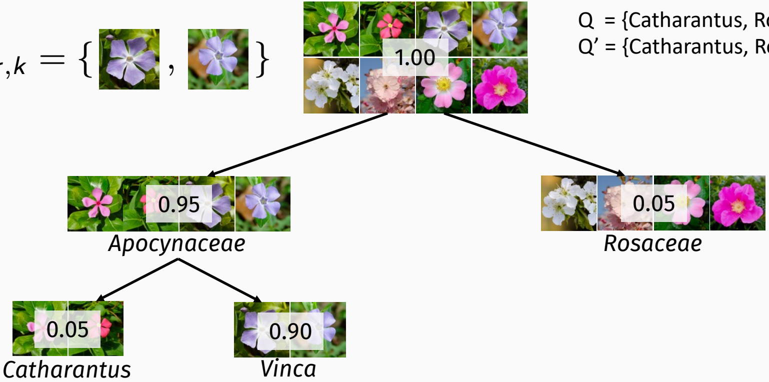
Q = {Catharantus, Rosaceae}

Copy Q + recursive call



Example for r=2 and k=3

$\hat{Y}_{r,k} = \{ \ , \ \}$

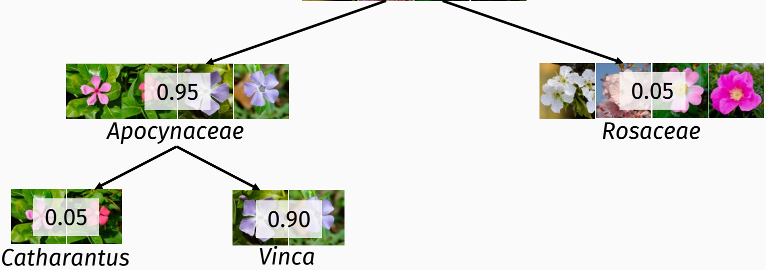


Q = {Catharantus, Rosaceae}
Q’ = {Catharantus, Rosaceae}

Example for r=2 and k=3

$\hat{Y}_{r,k} = \{$  ,  $\}$

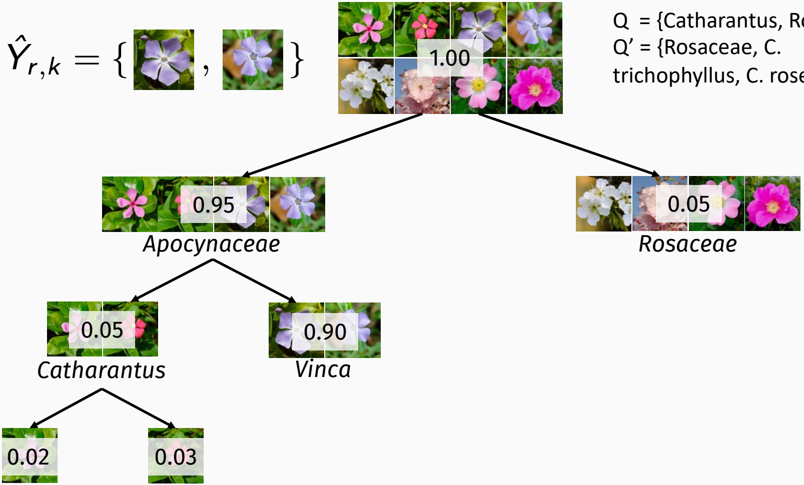
Q = {Catharantus, Rosaceae}
Q’ = {Rosaceae}

Example for r=2 and k=3

$\hat{Y}_{r,k} = \{ \; , \; \}$

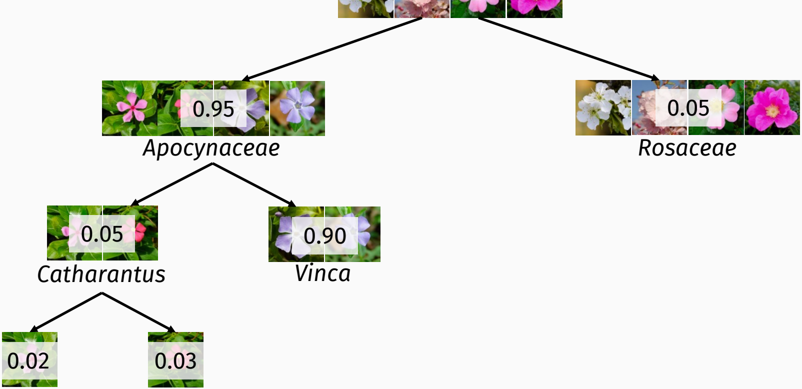
Q = {Catharantus, Rosaceae}
Q’ = {Rosaceae, C. trichophyllus, C. roseus}



Example for r=2 and k=3

$\hat{Y}_{r,k} = \{$  ,  $\}$

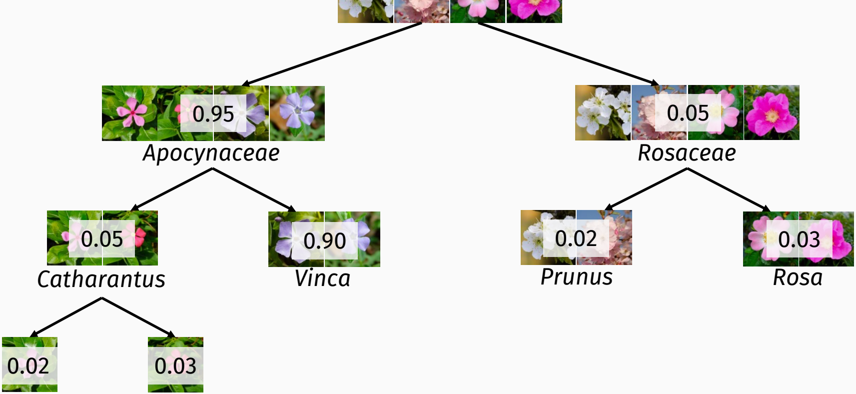
Q = {Catharantus, Rosaceae}
Q’ = {C. trichophyllus, C. roseus}



Example for r=2 and k=3

$\hat{Y}_{r,k} = \{$  ,  $\}$

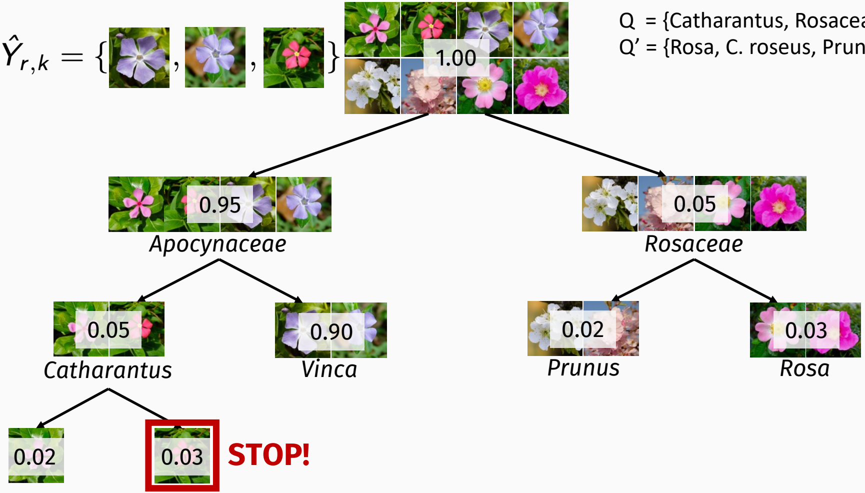
Q = {Catharantus, Rosaceae}
Q' = {C. trichophyllus, Rosa, C. roseus, Prunus}



Example for r=2 and k=3

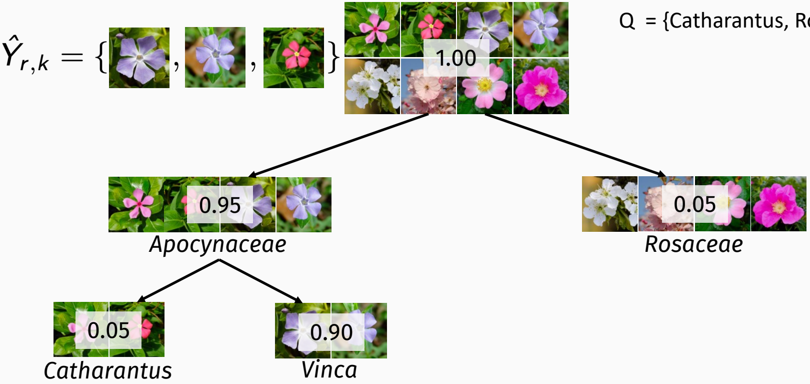
$\hat{Y}_{r,k} = \{ \; , \; , \; \}$

Q = {Catharantus, Rosaceae}
Q' = {Rosa, C. roseus, Prunus}



Example for r=2 and k=3

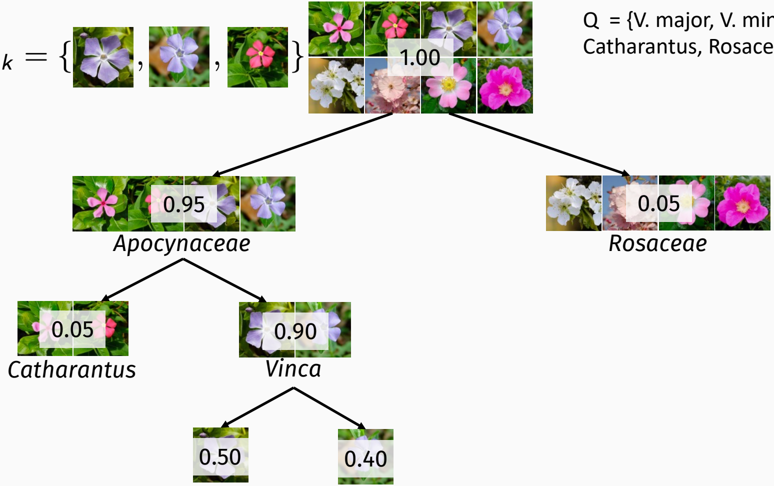
Q = {Catharantus, Rosaceae}



Example for r=2 and k=3

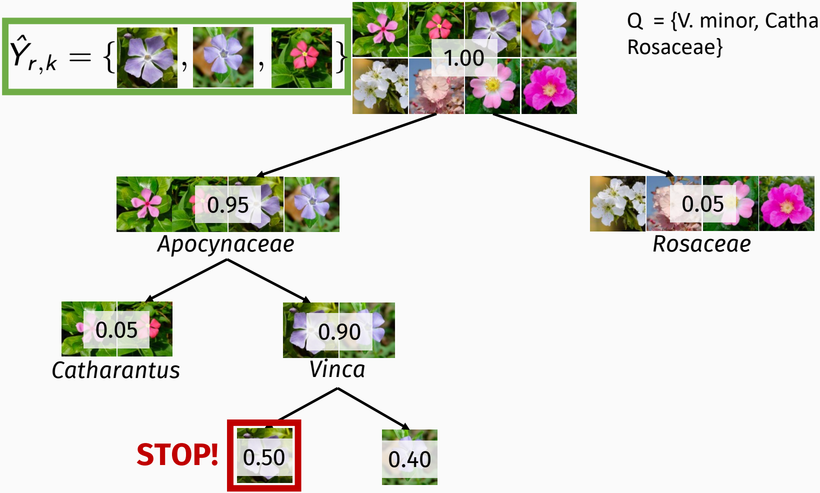
$\hat{Y}_{r,k} = \{$ $\}$

Q = {V. major, V. minor, Catharantus, Rosaceae}



Example for r=2 and k=3

Q = {V. minor, Catharantus, Rosaceae}



Results on PlantCLEF 2015 ($K = 1000$)

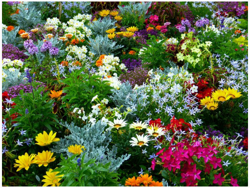
- $\hat{Y}_{r=1,k=5}$ = {*Carduus defloratus*}
- $\hat{Y}_{r=2,k=5}$ = {*Carduus defloratus, Carduus negrescens*}
- $\hat{Y}_{r=3,k=5}$ = {*Carduus defloratus, Carduus negrescens, Leontodon hispidus*}

Leontodon hispidus

Conclusion

Other contributions

- A statistical test that evaluates the validity of probabilistic set-valued predictions for the representation of epistemic uncertainty
- Thresholding methods vs. decision-theoretic approach for optimizing the F_β-measure in multi-label classification
- Large-scale benchmarking study of bacterial species identification using Matrix Assisted Laser Desorption/Ionisation Time-of-Flight Mass Spectrometry (MALDI-TOF MS) data

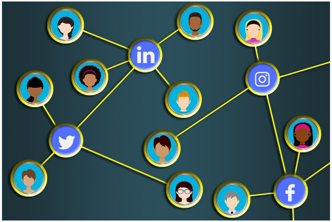


Conclusions

- Probabilistic classification + (aleatoric) uncertainty $\rightarrow$ set-valued prediction
- A novel decision-theoretic framework for unrestricted and restricted set-valued prediction
- Efficient inference algorithms that can calculate the optimal solution for a large number of classes K

Future perspectives

- Further improve efficiency of the inference algorithms (in particular for restricted set-valued prediction)
- Generalize to other hierarchical structures such as graphs
- Extend frameworks to the case of probabilistic set-valued prediction (i.e., second-level set-valued prediction)



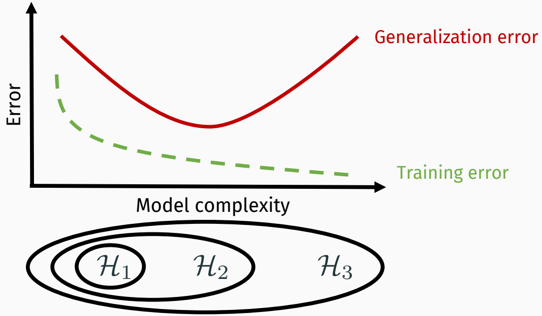
Thank you!

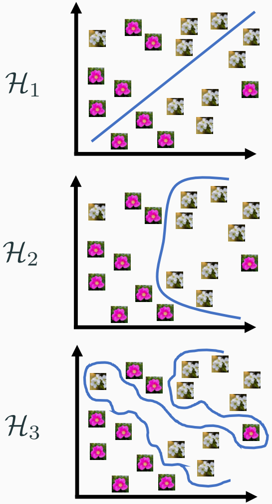
tfmortie/setvaluedprediction

Appendix

Introduction to probabilistic classification

Overfitting





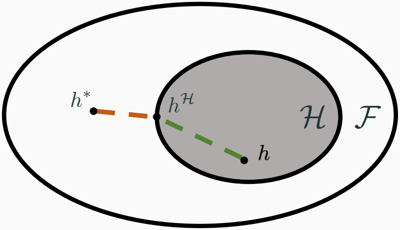
Generalisation (ii)

- The *regret* is given by:

$$R(h) - R(h^*) =$$
$$\underbrace{\Big(R(h^{\mathcal{H}}) - R(h^*)\Big)}_{\text{approximation error}} + \underbrace{\Big(R(h) - R(h^{\mathcal{H}})\Big)}_{\text{estimation error}}$$

- With the *Bayes classifier* and *best-in-class classifier*:

$$h^* = \operatorname*{arg\,inf}_{h \in \mathcal{F}} R(h)\,, \quad h^{\mathcal{H}} = \operatorname*{arg\,inf}_{h \in \mathcal{H}} R(h)$$



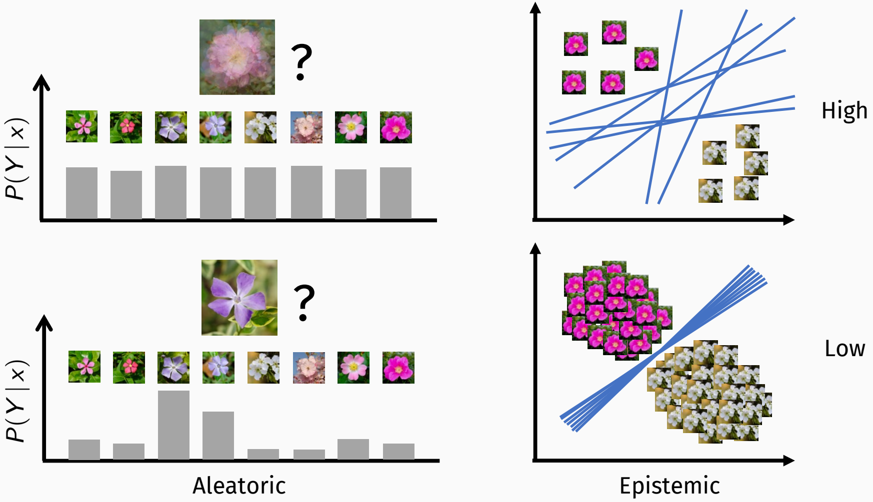
Regret bound plug-in classifier for ℓ_{01}

Theorem

Assume a multi-class classification multi-class classification problem, i.e., $\mathcal{Y} = \{1, \ldots, K\}$, with the zero-one loss ℓ_{01}. For any P, given the Bayes classifier h^*_{01} and the plug-in classifier $\hat{h}_{01}$, an upper bound for the regret is given by:

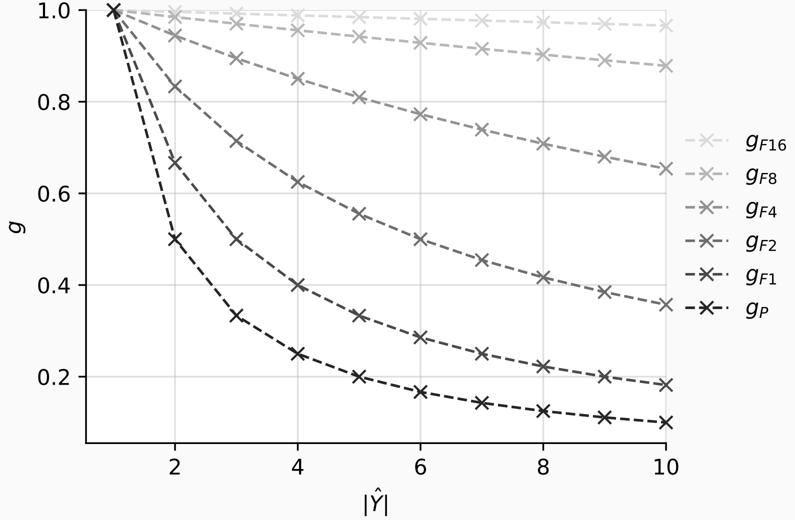
$$R_{01}(\hat{h}_{01}) - R^*_{01} \leq \sqrt{2\,\mathbb{E}_{\boldsymbol{x}\sim P(\boldsymbol{x})}\left[D_{KL}\Big(P(Y\,|\,\boldsymbol{x}) \| \hat{P}(Y\,|\,\boldsymbol{x})\Big)\right]}\,.$$

Aleatoric vs. epistemic uncertainty



Set-valued prediction in classification

Examples of set-based utility functions



Set-Valued Prediction (SVP-Full)

One can show that $U(\hat{Y}, \hat{P}, u) = g(|\hat{Y}|)\hat{P}(\hat{Y} \mid \boldsymbol{x})$.

(2.a) Inner maximization:

$$\begin{aligned}\hat{Y}_u^s &= \operatorname*{arg\,max}_{|\hat{Y}|=s} g(s)\hat{P}(\hat{Y} \mid \boldsymbol{x}) \\ &= \operatorname*{arg\,max}_{|\hat{Y}|=s} \hat{P}(\hat{Y} \mid \boldsymbol{x}), \quad \forall s \in \{1, \ldots, K\}\end{aligned}$$

(2.b) Outer maximization:

$$\hat{\boldsymbol{h}}_u(\boldsymbol{x}) = \operatorname*{arg\,max}_{\hat{Y} \in \{\hat{Y}_u^1, \ldots, \hat{Y}_u^K\}} g(|\hat{Y}|)\hat{P}(\hat{Y} \mid \boldsymbol{x})$$

Regret bound plug-in classifier for u

Theorem

Assume a multi-class classification problem, i.e., $\mathcal{Y} = \{1, \ldots, K\}$*, with the family of utility functions* u*. For any* P*, given the Bayes classifier* $\boldsymbol{h}_u^*$ *and the plug-in classifier* $\hat{\boldsymbol{h}}_u$*, an upper bound for the regret is given by:*

$$R_u(\hat{\boldsymbol{h}}_u) - R_u(\boldsymbol{h}_u^*) \leq \sqrt{8\,\mathbb{E}_{\boldsymbol{x}\sim P(\boldsymbol{x})}\left[D_{KL}\Big(P(Y\,|\,\boldsymbol{x})\|\hat{P}(Y\,|\,\boldsymbol{x})\Big)\right]}\,.$$

Results (2)

Table 1: Performance versus runtime for the SVP-Full, SVP-ANNS and SVP-HF inference algorithms, tested on LSHTC1 ($K = 12166$) for the u_{F1} utility. Notation: $|\hat{Y}|$ – avg. set size, t_{train} – CPU train time in seconds, t_{test} – CPU test time in milliseconds / number of test samples

Inference	t_{train}	Top-1 accuracy	Recall	$\lvert\hat{Y}\rvert$	t_{test}
SVP-Full	71509	0.4200	0.4538	1.29	46.13
SVP-ANNS	72361	0.4152	0.4486	1.30	8.28
SVP-HF	557	0.3982	0.4479	1.42	0.52

Set-valued prediction in hierarchical classification

Regret bound top-down classifier for ℓ_{ce}

Theorem

Assume a hierarchical multi-class classification problem, i.e., $\mathcal{Y} = \{1, \dots, K\}$ with a hierarchical tree structure $\mathcal{T}$, and with the cross-entropy loss ℓ_{ce}. For any P, given the Bayes error R^*_{ce}, the following cross-entropy loss regret for the top-down classifier is obtained:

$$R_{ce}(\hat{P}) - R^*_{ce} = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\sum_{j=1}^{d} reg\big(\hat{P}(V \,|\, Path(y)_{j+1}, \boldsymbol{x})\big)\right],$$

with d the maximum depth of the tree structure $\mathcal{T}$ and

$$reg\big(\hat{P}(V \,|\, Path(y)_{j+1}, \boldsymbol{x})\big)$$

the regret of the local classifier $\hat{P}(V \,|\, Path(y)_{j+1}, \boldsymbol{x})$.

Results (2)

Table 2: Performance versus runtime for the MVM, KCG and RTS inference algorithms, tested on Proteins ($K = 3485$). Notation: $|\hat{Y}|$ – avg. set size, t_{test} – CPU test time in milliseconds / number of test samples

Inference	Top-1 accuracy	Recall ($k=5$)	$\lvert\hat{Y}\rvert$ ($k=5$)	t_{test} ($k=5$)	Recall ($k=10$)	$\lvert\hat{Y}\rvert$ ($k=10$)	t_{test} ($k=10$)
MVM-1	0.7699	0.7766	1.3152	0.0489	0.7829	2.2505	0.0500
KCG-1	0.7667	0.7728	1.3245	0.4748	0.7802	2.3300	0.4739
KCG-2		0.8439	2.3042	0.4758	0.8494	4.2730	0.4751
KCG-3		0.8734	3.2057	0.4837	0.8765	5.8075	0.4861
KCG		0.9003	4.9320	0.4888	0.9219	9.8309	0.4906
RTS-1	0.7806	0.7936	1.3045	0.0004	0.8012	2.2052	0.0003
RTS-2		0.8610	2.3161	0.0004	0.8664	3.6366	0.0005
RTS-3		0.8842	3.2457	0.0005	0.8885	4.7484	0.0006