

A Extended Related Work

Uncertainty Representation. In machine learning, uncertainty is often represented by Bayesian methods [Mackay, 1992, Blundell et al., 2015]. Frequently, the Bayesian posterior is approximated by ensembles [Lakshminarayanan et al., 2017] or methods such as Dropout [Gal and Ghahramani, 2016] or Laplace approximation [Daxberger et al., 2021]. An important characteristic of such representations, especially for uncertainty tasks, is diversity [D’Angelo and Fortuin, 2021, Wood et al., 2023]. Some works enforce this by means of regularization [de Mathelin et al., 2023], whereas others vary hyperparameters across models within the ensemble [Wenzel et al., 2020] or enforce diversity in the representations [Lopes et al., 2022].

Alternatively, credal sets have been used in the fields of imprecise probability and machine learning to represent model uncertainty [Zaffalon, 2001, Corani and Zaffalon, 2008, Corani and Mignatti, 2015]. Antonucci et al. [2012a] proposed to generate such sets based on relative likelihoods. Relative likelihoods, also referred to as normalized likelihoods, have also been used in machine learning with simple model classes such as logistic regression [Senge et al., 2014, Cella and Martin, 2024]. In this work, we build upon these approaches and address the challenges that emerge when adapting the relative likelihood to a setting with complex predictors.

Recently, credal sets have been applied in the context of machine learning. Wang et al. [2024b] take multiple samples from a Bayesian posterior or ensembles and derive class-wise lower and upper probabilities. Based on these samples, a credal set is constructed by including all probability distributions such that the individual predicted class probabilities are in the respective lower to upper class probability interval. Nguyen et al. [2025] also construct credal sets from ensemble predictions, but with the additional option of discarding potential outliers to prevent the set from becoming too large. This is done by comparing all ensemble predictions to a representative prediction, e.g. the mean prediction, using some distance between distributions, and keeping only $(1 - \alpha) \cdot 100\%$ closest predictions. The credal set is then constructed by taking the convex hull of the remaining probability distributions. Other methods directly train neural networks to predict intervals by explicitly predicting a lower and upper probability for every class [Wang et al., 2024a]. In combination with a custom loss function, consisting of regular cross-entropy for the upper probabilities and cross-entropy computed on the highest loss subset of a batch for the lower probabilities, the claim is that this approach encourages both “optimistic” and “pessimistic” predictions. Credal sets are constructed by taking the same interval-based approach as [Wang et al., 2024b]. Besides this, hybrid methods, combining multiple uncertainty frameworks, have also been proposed. Caprio et al. [2023] combine Bayesian deep learning and credal sets by considering sets of priors over weights of neural networks. Training these neural network by variational inference then results in a set of posteriors. Based on this, a credal set is constructed by sampling from the Bayesian neural networks and taking the convex hull of the sampled probability distributions. Another approach leverages conformal prediction to construct credal sets with validity guarantees [Javanmardi et al., 2024]. However, this work uses distributions over classes to perform the conformal calibration step. Since our method does not require such data, we exclude this method as a baseline. Our work distinguishes itself from aforementioned works by using relative likelihood cuts which allow for an intuitive and adaptable construction of the credal set.

Moreover, our approach is conceptually related to Rashomon sets [Semenova et al., 2022]. Both characterize a collection of models whose performance exceeds a given threshold, thereby facing the same challenge in approximating this set of plausible models [Donnelly et al., 2025]. However, the objectives differ: Rashomon sets are primarily concerned with interpretability and (syntactic) model diversity, whereas our focus is on uncertainty quantification and predictive diversity.

Uncertainty Quantification. Given a credal representation of uncertainty, there are many ways to quantify uncertainty. Some measures consider only the epistemic uncertainty [Abellán and Moral, 2000], whereas others use entropy to reason about the total uncertainty [Abellán and Moral, 2003]. In this line of work, Abellán et al. [2006] also proposed measures based on entropy that decompose the total uncertainty into an aleatoric and epistemic component. Conversely, Antonucci et al. [2012b] quantify uncertainty by measuring the lack of dominance of one class over others in the predictive distributions represented by the credal set. Hüllermeier et al. [2022] offer a critical analysis of these measures and propose an alternative for the dominance-based measure. Recently, Hofman et al. [2024] proposed to quantify credal uncertainty based on a decomposition of scoring rules. In the following, we consider the uncertainty measures by Abellán et al. [2006] in order to ensure a fair comparison with previous works.

B Experimental Details

B.1 Datasets

ChaosNLI The ChaosNLI dataset, introduced by Nie et al. [2020], is a large-scale dataset designed to study human disagreement in natural language inference (NLI) tasks. It comprises 100 human annotations per example for 3,113 examples from the SNLI and MNLI datasets, and 1,532 examples from the α NLI dataset, totaling approximately 464,500 annotations. In line with Javanmardi et al. [2024], we use only the SNLI and MNLI subsets of the ChaosNLI dataset, but for simplicity, we will refer to this as the ChaosNLI dataset. Each example includes metadata such as the unique identifier, counts of each label assigned by annotators, the majority label, label distribution, entropy of the label distribution, the original example text, and the original label from the source dataset. This dataset enables a detailed analysis of the distribution of human opinions in NLI tasks, highlighting instances of high disagreement and questioning the validity of using majority labels as the sole ground truth. ChaosNLI is publicly available under the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) license. We train our models on the 768-dimensional embeddings of the ChaosNLI dataset retrieved from <https://github.com/alireza-javanmardi/conformal-credal-sets>. We refer to [Javanmardi et al., 2024] for more details on the generation of the embeddings.

CIFAR-10 The CIFAR-10 dataset is a widely used benchmark in machine learning and computer vision, introduced by Krizhevsky et al. [2009], and Geoffrey Hinton in 2009. It comprises 60,000 color images at a resolution of 32×32 pixels, evenly distributed across 10 distinct classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is partitioned into 50,000 training images and 10,000 test images, organized into five training batches and one test batch, each containing 10,000 images. The dataset is publicly available and has been utilized extensively for developing and benchmarking machine learning models. While the original dataset does not specify a license, various distributions, such as those provided by TensorFlow Datasets, are released under the Creative Commons Attribution 4.0 License.

CIFAR-10H The CIFAR-10H dataset provides human-derived soft labels for the 10,000 images in the CIFAR-10 test set, capturing the variability in human annotation during image classification tasks. Developed by Peterson et al. [2019], the dataset comprises 511,400 annotations collected from 2,571 Amazon Mechanical Turk workers, with each image receiving approximately 51 labels. Annotators classified images into one of the ten CIFAR-10 categories, enabling the construction of probability distributions over labels for each image. CIFAR-10H is publicly available under the Creative Commons BY-NC-SA 4.0 license.

CIFAR-100 The CIFAR-100 dataset, introduced by Krizhevsky et al. [2009], comprises 60,000 color images at 32×32 resolution, divided into 100 classes with 600 images each. Each image has a “fine” label (specific class) and a “coarse” label (superclass), with the 100 classes grouped into 20 superclasses. The dataset is split into 50,000 training and 10,000 test images. It is a subset of the Tiny Images dataset and is commonly used for evaluating image classification algorithms. While the original dataset does not specify a license, various distributions, such as those provided by TensorFlow Datasets, are released under the Creative Commons Attribution 4.0 License.

QualityMRI The QualityMRI dataset, introduced by Obuchowicz et al. [2020], is part of the Data-Centric Image Classification (DCIC) Benchmark, which aims to evaluate the impact of dataset curation on model performance. The dataset contains 310 magnetic resonance (MRI) images spanning various quality levels and is designed to assess the MRI image quality. The dataset is publicly available under the Creative Commons BY-SA 4.0 license.

SVHN The SVHN dataset, introduced by Netzer et al. [2011], consists of over 600,000 32×32 RGB images of digits (0–9) obtained from real-world house number images in Google Street View. It includes three subsets: 73,257 training images, 26,032 test images, and 531,131 additional images for extra training. The dataset is designed for digit recognition tasks with minimal preprocessing. While the original dataset does not specify a license, various distributions, such as those provided by TensorFlow Datasets, are released under the Creative Commons Attribution 4.0 License.

Places365 Places365, introduced by Zhou et al. [2018], is a large-scale scene recognition dataset containing 1.8 million training images across 365 scene categories. The validation set includes 50 images per category, and the test set has 900 images per category. An extended version, Places365-Challenge-2016, adds 6.2 million images and 69 new scene classes, totaling 8 million images over 434 categories. While the original dataset does not specify a license, various distributions, such as those provided by TensorFlow Datasets, are released under the Creative Commons Attribution 4.0 License.

FMNIST Fashion-MNIST (FMNIST), introduced by Xiao et al. [2017], is a dataset of Zalando’s article images, comprising 70,000 28×28 grayscale images labeled across 10 classes, such as T-shirt/top, Trouser, and Sneaker. It includes 60,000 training and 10,000 test images and serves as a direct replacement for the original MNIST dataset for benchmarking machine learning algorithms. FMNIST is publicly available under the MIT License.

ImageNet ImageNet, introduced by Deng et al. [2009], is a large-scale image database organized according to the WordNet hierarchy, containing over 14 million images across more than 20,000 categories. The ILSVRC subset (ImageNet-1K) includes 1,281,167 training images, 50,000 validation images, and 100,000 test images across 1,000 classes. The dataset is available for free to researchers for non-commercial use.

B.2 Models

Fully-Connected Network We train fully-connected neural networks on the ChaosNLI dataset. The network consists of 4 linear layers with $[768 - 256 - 64 - 16 - 3]$ units with ReLU activations, except for the last layer, which has Softmax transformation to transform the logits into probabilities. We use the hyperparameters (cf. Table 2) similar to the optimal parameters found in Javanmardi et al. [2024].

ResNet18 For experiments on the CIFAR-10 dataset, we use the PyTorch ResNet-18 implementation and hyperparameters provided by <https://github.com/kuangliu/pytorch-cifar>. This model is specifically optimized for CIFAR-10 and is trained from scratch, without any pretraining on ImageNet. For experiments on the QualityMRI dataset, we use the ResNet18 implementation from the PyTorch torchvision package with random initialization, i.e. no pretrained weights.

Hyperparameters Each dataset is trained using a dedicated set of hyperparameters as presented in Table 2. We evaluated multiple configurations and selected the best-performing ones for each dataset. To ensure fair and consistent comparisons, all models trained on a given dataset, both our approach and the baselines, use the same hyperparameter settings. The only exception is the CreBNN, which requires a KL-divergence penalty of $1e - 7$ and zero weight decay when using the Adam optimizer [Kingma and Ba, 2015]. When we apply the SGD optimizer with a learning rate scheduler, namely Cosine Annealing [Loshchilov and Hutter, 2017], CreBNN requires additionally a momentum of 0.9 to enable effective learning.

Table 2: Hyperparameters used for each dataset.

Hyperparameter	ChaosNLI	CIFAR-10	QualityMRI
Model	FCNet	ResNet18	ResNet18
Epochs	300	200	200
Learning rate	0.01	0.1	0.01
Weight decay	0.0	0.0005	0.0005
Optimizer	Adam	SGD	SGD
Ensemble members	20	20	20
LR scheduler	-	CosineAnnealing	CosineAnnealing
Tobias value	100	100	100

B.3 Out-of-Distribution Detection

We use the SVHN [Netzer et al., 2011], Places365 [Zhou et al., 2018], CIFAR-100 [Krizhevsky et al., 2009], Fashion-MNIST [Xiao et al., 2017], and ImageNet [Le and Yang, 2015] datasets for Out-of-Distribution detection.

The Out-of-Distribution task is treated as a binary classification task where the epistemic uncertainty is used as the classification criterion. In order to balance the data, we sample 10000 instances from the test set of the respective datasets. On both the in-Distribution data (CIFAR-10) and the Out-of-Distribution data, the same transforms — normalization and resizing to 32 by 32 pixels — are applied. After computing the epistemic uncertainty the area under the receiver operating characteristics curve (AUROC) is computed and used as the comparison metric.

B.4 Computing Uncertainty

Computing the total and aleatoric uncertainty involves solving a per instance optimization problem, after which the epistemic uncertainty is obtained as their difference (see Equation (5)). This optimization is performed using SciPy’s `minimize` function with the SLSQP solver and the default parameters.

Interval-Based For the interval-based credal sets, the optimization is initialized with the mean of the predicted distributions, bounded between the lower and upper probabilities for each class, and constrained to ensure that the solution forms a valid probability distribution, namely, the class probabilities must sum to 1.

Convex Hull For the credal sets based on the convex hull, the optimization is done on the weights of convex combination, instead of the probability distribution. Uniform weights are used as the initial value, the weights are bounded between 0 and 1, and constrained to sum to 1.

Estimated Computing Time Here, we provide an estimated upper bound of the computation time for computing the lower and upper entropy of a credal set based on the example of the largest possible credal set, namely the full probability simplex for a 10-class problem, such as CIFAR-10. For this largest set, computing the upper entropy using an interval-based credal set takes on average 0.03 seconds, while computing the lower entropy takes about 0.02 seconds. For the convex hull-based credal set of the same size, the average computation time is 0.07 seconds for upper entropy and 0.03 seconds for lower entropy. To simplify our estimation, we average these times and assume that for a single instance it takes 0.03 seconds to optimize the lower entropy and 0.04 seconds to optimize upper entropy, summing up to 0.07 seconds of computing time per instance.

In the Out-of-Distribution (OoD) detection experiments, we need to compute both lower and upper entropy for each instance in both the in-Distribution (iD) and OoD datasets. Each dataset contains 10,000 instances, resulting in roughly 23 minutes of computation time per model to obtain the epistemic uncertainty across all instances. Since we run three seeds per model and evaluate 12 different models, the total runtime for a single OoD experiment on one dataset is approximately 14 hours. As we evaluate OoD detection across five datasets, the total computing time amounts to around 70 hours excluding any time needed to train models beforehand.

B.5 Computing Coverage

Coverage is evaluated by checking whether the ground-truth distribution lies within the predicted credal set. For the interval-based approach, this involves verifying that each class probability of the ground-truth distribution falls between the corresponding lower and upper bounds of the credal set. For the convex hull approach, we assess whether the ground-truth distribution can be expressed as a convex combination of the extreme points defining the credal set. This optimization is done using the SciPy `linprog` function.

B.6 Baselines

We list all details regarding the implementations of the baselines that were used in the paper. In general, all baselines were implemented in our own code base.

Credal Wrapper (CreWra) The Credal Wrapper was initially implemented in TensorFlow, but we reimplemented it in PyTorch to ensure compatibility with our framework. It follows a standard ensemble learning approach, training multiple models independently. Like our method, the Credal Wrapper constructs credal sets using class-wise upper and lower probability bounds, making it well-aligned with our implementation. Overall, we closely follow both the original paper and their available implementation [Wang et al., 2024b].

Credal Ensembling (CreEns $_{\alpha}$) Since no official code was available for neural network implementations of Credal Ensembling, we reimplemented the method ourselves. Our implementation closely follows all details provided in Nguyen et al. [2025]. The approach builds on standard ensemble training procedures, with inference adapted according to their proposed method of sorting predictions based on a distance measure and selecting only $\alpha\%$ closest predictions to construct credal sets. In our experiments, we use the Euclidean distance measure and evaluate several values of α .

Credal Deep Ensembles (CreNet) As the official implementation of Credal Deep Ensembles is only available in TensorFlow, we reimplemented the method in PyTorch to ensure compatibility with our codebase. Our implementation closely mirrors the original TensorFlow code, particularly in adapting the model architecture and loss function. Specifically, we replace each model’s final linear layer with a head comprising a linear layer outputting $2 \times$ classes (representing upper and lower probability bounds), followed by a batch normalization layer and the custom IntSoftmax layer. We also reimplemented the proposed loss function, which computes a cross-entropy loss for the upper bounds and selectively backpropagates the lower-bound loss only for the $\delta\%$ of samples with the highest loss values, as described in Wang et al. [2024a]. In our experiments, we use $\delta = 0.5$, as suggested in Wang et al. [2024a].

Credal Bayesian Deep Learning (CreBNN) No code or implementation details for Credal Bayesian Deep Learning (CreBNN) were made publicly available, and despite multiple attempts to contact the authors, we received no response or further clarification. As a result, we reimplemented the method ourselves based solely on the high-level description provided in the paper. In our implementation, each ensemble member is a Bayesian neural network (BNN) trained with variational inference using different priors, with prior means μ sampled from $[-1, 1]$ and standard deviations σ from $[0.1, 2]$ to form a diverse prior set. During inference, we draw one sample from each BNN to obtain a finite set of probability distributions, and construct the credal set as the convex hull of these predictions.

B.7 Computing Resources

To run the experiments presented in this work, we utilized the computing resources detailed in Table 3. The total estimated GPU usage amounts to approximately 750 hours.

Table 3: Specifications of Computing Resources.

Component	Specification
CPU	AMD EPYC MILAN 7413 Processor, 24C/48T 2.65GHz 128MB L3 Cache
GPU	$2 \times$ NVIDIA A40 (48 GB GDDR each)
RAM	128 GB (4x 32GB) DDR4-3200MHz ECC DIMM
Storage	$2 \times$ 480GB Samsung Datacenter SSD PM893

C Additional Experiments

This section presents additional results, including ablation studies and alternative hyperparameter configurations, that complement the findings reported in the main paper.

C.1 Out-of-Distribution detection

In addition to the OoD experiments in Section 5, we evaluate a broad range of values α for the CreEns approach.

Table 4: Out-of-Distribution detection based on epistemic uncertainty. CIFAR-10 is used as the in-Distribution dataset. The mean and standard deviation over 3 runs are reported. Best performance is in **bold**.

Method	SVHN	Places	CIFAR-100	FMNIST	ImageNet
CreWra	0.957\pm0.003	0.916 \pm 0.001	0.916\pm0.000	0.952 \pm 0.000	0.890\pm0.001
CreEns _{0.95}	0.500 \pm 0.000	0.500 \pm 0.000	0.500 \pm 0.000	0.500 \pm 0.000	0.500 \pm 0.000
CreEns _{0.9}	0.921 \pm 0.002	0.879 \pm 0.002	0.883 \pm 0.001	0.915 \pm 0.001	0.857 \pm 0.002
CreEns _{0.8}	0.937 \pm 0.001	0.896 \pm 0.001	0.900 \pm 0.001	0.929 \pm 0.001	0.875 \pm 0.001
CreEns _{0.6}	0.944 \pm 0.002	0.902 \pm 0.001	0.906 \pm 0.000	0.935 \pm 0.001	0.881 \pm 0.001
CreEns _{0.4}	0.947 \pm 0.001	0.906 \pm 0.001	0.908 \pm 0.000	0.940 \pm 0.001	0.883 \pm 0.001
CreEns _{0.2}	0.950 \pm 0.001	0.909 \pm 0.001	0.911 \pm 0.000	0.946 \pm 0.001	0.885 \pm 0.001
CreEns _{0.0}	0.955 \pm 0.001	0.913 \pm 0.000	0.914 \pm 0.001	0.949 \pm 0.001	0.888 \pm 0.000
CreNet	0.943 \pm 0.003	0.918 \pm 0.000	0.912 \pm 0.000	0.951 \pm 0.002	0.884 \pm 0.001
CreBNN	0.907 \pm 0.006	0.885 \pm 0.002	0.880 \pm 0.002	0.935 \pm 0.002	0.859 \pm 0.002
CreRL _{1.0}	0.948 \pm 0.003	0.918\pm0.002	0.916\pm0.001	0.957\pm0.002	0.889\pm0.002
CreRL _{0.95}	0.917 \pm 0.013	0.910 \pm 0.001	0.901 \pm 0.000	0.945 \pm 0.004	0.878 \pm 0.002
CreRL _{0.9}	0.918 \pm 0.011	0.907 \pm 0.001	0.896 \pm 0.001	0.944 \pm 0.004	0.874 \pm 0.001
CreRL _{0.8}	0.906 \pm 0.008	0.894 \pm 0.001	0.884 \pm 0.003	0.936 \pm 0.009	0.865 \pm 0.002
CreRL _{0.6}	0.862 \pm 0.035	0.874 \pm 0.003	0.852 \pm 0.002	0.893 \pm 0.005	0.837 \pm 0.003
CreRL _{0.4}	0.739 \pm 0.029	0.821 \pm 0.007	0.796 \pm 0.007	0.815 \pm 0.020	0.788 \pm 0.010
CreRL _{0.2}	0.582 \pm 0.041	0.736 \pm 0.010	0.700 \pm 0.013	0.676 \pm 0.046	0.698 \pm 0.013

C.2 Ablations

In the following ablation study, we evaluate two factors: the impact of varying the initialization constant in our proposed ToBias initialization, and the effect of changing the number of ensemble members. The study is conducted on the ChaosNLI dataset.

C.2.1 ToBias Initialization

We study the impact of the ToBias initialization constant β on the coverage and efficiency of the resulting credal predictor on the ChaosNLI dataset. To do so, we vary the constant, taking values $\beta \in \{5, 10, 20, 30, 50, 80, 100, 200, 500\}$. The ensemble is then trained as proposed in Algorithm 1. The coverage and efficiency Pareto front is shown in Figure 5 with the mean and standard deviation over three runs. We split the results into three Figures for better readability.

For low values of β , the coverage of the ensembles is rather low and as β increases, the coverage also increases. For large values of β , e.g. $\beta = 200$ and $\beta = 500$, the ensembles with large α values are no longer able to reach into the low coverage, high efficiency region. In particular, $\beta = 500$ has a higher coverage and lower efficiency for larger values of α than for smaller values of α . This may be caused by convergence problems due to having a very large bias for a particular class. This causes the individual ensemble members to not always converge to their desired threshold, hence resulting in large credal sets (with high coverage and low efficiency), because the border of the credal set is not reached. In essence, the β value provides the ability to slightly shift the Pareto front to reach the desired coverage, efficiency region (in addition to α).

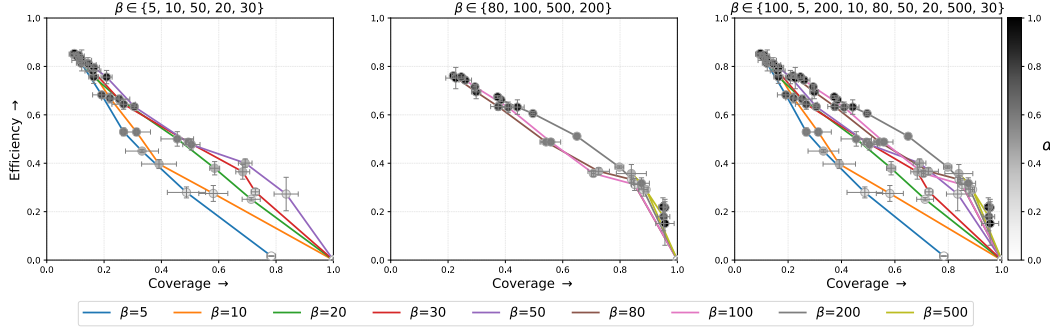


Figure 5: Pareto front between coverage and efficiency for different values of ToBias constant β . **Left:** low values of β , **middle:** high values of β , **right:** all values of β .

C.2.2 Number of Ensemble Members

We study the impact of the numbers of ensemble members M on the coverage and efficiency of the resulting credal predictor on the ChaosNLI dataset. To do so, we vary the constant, taking values $M \in \{1, 2, 3, 4, 5, 10, 20, 30, 50\}$. The ensemble is then trained as proposed in Algorithm 1. The coverage and efficiency Pareto front is shown in Figure 6 with the mean and standard deviation over three runs. We split the results into three Figures for better readability.

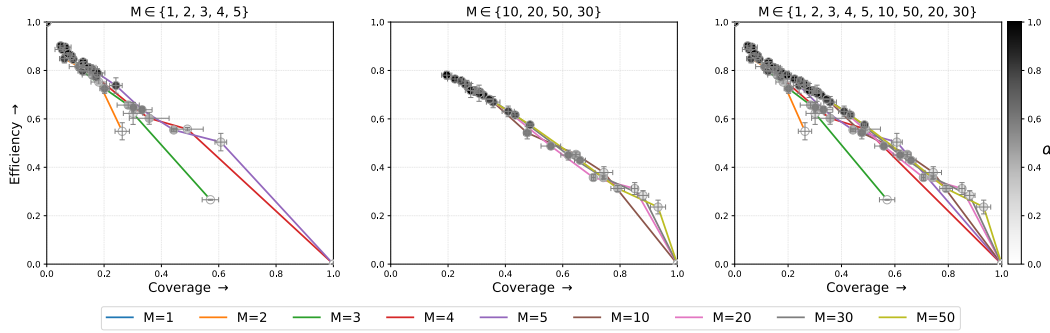


Figure 6: Pareto front between coverage and efficiency for different number of ensemble members M . **Left:** low values of M , **middle:** high values of M , **right:** all values of M .

Naturally, when $M = 1$, the coverage is 0 and efficiency 1, because the credal set reduces to a point prediction. With $M = 2$ and $M = 3$, the $\alpha = 0$ ensemble does not have coverage 1 and efficiency 0, because there are not enough ensemble members, and hence probability distribution, to span the whole probability simplex for this 3 class problem. As M increases, the Pareto front starts to span more of the Pareto figure, until stabilizing around $M = 20$. After this, the number of ensemble members does not have a significant impact on the resulting Pareto front anymore.

C.3 Data Uncertainty

The datasets used in our experiments, namely ChaosNLI, CIFAR-10, and QualityMRI, differ in their inherent levels of (aleatoric) uncertainty. This is reflected, for example, in the degree of disagreement among the collected annotations, which serve as the ground-truth distributions in our evaluations. To provide further insight, we report the average entropy of these ground-truth distributions in Table 5.

Table 5: Average entropy of ground-truth distributions in the test sets of ChaosNLI, CIFAR-10 and QualityMRI.

Dataset	Avg. Entropy
ChaosNLI	0.932
CIFAR-10	0.223
QualityMRI	0.782

C.4 Performance of Ensemble Members

We visualized the trade-off between coverage and efficiency in Figure 4, and provide the corresponding numerical values in Table 6. We also report the accuracies of the individual predictors within each ensemble for both the baselines and our proposed method to provide insight into their standalone performance (cf. Tables 7 and 8). For CreNet, whose ensemble members directly predict probability intervals, we use their intersection probability technique to derive pointwise predictions [Wang et al., 2024a].

Table 6: Coverage and efficiency for different methods across datasets.

Approach	ChaosNLI		CIFAR-10		QualityMRI	
	Coverage	Efficiency	Coverage	Efficiency	Coverage	Efficiency
CreRL _{0.0}	1.000±0.000	0.000±0.000	1.000±0.000	0.000±0.000	1.000±0.000	0.000±0.000
CreRL _{0.2}	0.851±0.011	0.312±0.025	0.754±0.005	0.766±0.003	0.796±0.068	0.261±0.053
CreRL _{0.4}	0.707±0.007	0.358±0.013	0.680±0.004	0.824±0.004	0.747±0.033	0.293±0.010
CreRL _{0.6}	0.559±0.034	0.488±0.012	0.634±0.004	0.862±0.004	0.677±0.023	0.360±0.016
CreRL _{0.8}	0.410±0.021	0.631±0.022	0.581±0.005	0.894±0.003	0.613±0.035	0.417±0.025
CreRL _{0.9}	0.294±0.007	0.716±0.010	0.556±0.001	0.911±0.001	0.548±0.046	0.475±0.038
CreRL _{0.95}	0.260±0.022	0.743±0.012	0.543±0.002	0.920±0.001	0.511±0.046	0.508±0.037
CreRL _{1.0}	0.246±0.012	0.757±0.008	0.498±0.001	0.950±0.000	0.500±0.086	0.526±0.036
CreWra	0.453±0.050	0.607±0.037	0.450±0.001	0.963±0.000	0.355±0.057	0.608±0.021
CreNet	0.001±0.002	0.978±0.008	0.094±0.010	0.999±0.000	0.188±0.020	0.792±0.003
CreBNN	0.195±0.018	0.649±0.017	0.331±0.005	0.871±0.003	0.898±0.133	0.090±0.124
CreEns _{0.0}	0.298±0.041	0.607±0.037	0.000±0.000	0.963±0.000	0.329±0.040	0.611±0.023
CreEns _{0.2}	0.165±0.007	0.769±0.010	0.000±0.000	0.983±0.000	0.177±0.060	0.787±0.020
CreEns _{0.4}	0.096±0.008	0.834±0.006	0.000±0.000	0.986±0.000	0.118±0.059	0.844±0.021
CreEns _{0.6}	0.043±0.003	0.886±0.005	0.000±0.000	0.988±0.000	0.081±0.057	0.899±0.012
CreEns _{0.8}	0.012±0.002	0.941±0.002	0.000±0.000	0.993±0.000	0.054±0.033	0.950±0.008
CreEns _{0.9}	0.000±0.000	0.976±0.001	0.000±0.000	0.997±0.000	0.016±0.023	0.983±0.002
CreEns _{0.95}	0.000±0.000	1.000±0.000	0.000±0.000	1.000±0.000	0.000±0.000	1.000±0.000

Table 7: Accuracy per ensemble member of CreRL $_{\alpha}$ for varying α values on ChaosNLI, CIFAR-10, and QualityMRI.

ChaosNLI								
α	0.0	0.2	0.4	0.6	0.8	0.9	0.95	1.0
Member m	1	0.678 \pm 0.017	0.678 \pm 0.017	0.678 \pm 0.017	0.678 \pm 0.017	0.678 \pm 0.017	0.678 \pm 0.017	0.678 \pm 0.017
	2	0.370 \pm 0.012	0.526 \pm 0.075	0.512 \pm 0.063	0.577 \pm 0.011	0.608 \pm 0.055	0.664 \pm 0.028	0.682 \pm 0.016
	3	0.453 \pm 0.010	0.549 \pm 0.008	0.407 \pm 0.020	0.604 \pm 0.045	0.634 \pm 0.016	0.668 \pm 0.003	0.663 \pm 0.027
	4	0.178 \pm 0.014	0.468 \pm 0.022	0.483 \pm 0.045	0.483 \pm 0.046	0.651 \pm 0.016	0.660 \pm 0.012	0.661 \pm 0.010
	5	0.370 \pm 0.012	0.568 \pm 0.018	0.425 \pm 0.046	0.560 \pm 0.042	0.624 \pm 0.018	0.656 \pm 0.016	0.677 \pm 0.017
	6	0.453 \pm 0.010	0.535 \pm 0.015	0.530 \pm 0.023	0.585 \pm 0.025	0.647 \pm 0.017	0.663 \pm 0.012	0.674 \pm 0.007
	7	0.178 \pm 0.014	0.453 \pm 0.015	0.506 \pm 0.041	0.594 \pm 0.044	0.637 \pm 0.020	0.647 \pm 0.003	0.664 \pm 0.005
	8	0.370 \pm 0.012	0.585 \pm 0.027	0.494 \pm 0.119	0.585 \pm 0.039	0.646 \pm 0.038	0.673 \pm 0.022	0.677 \pm 0.024
	9	0.453 \pm 0.010	0.493 \pm 0.074	0.595 \pm 0.042	0.604 \pm 0.029	0.663 \pm 0.007	0.668 \pm 0.002	0.661 \pm 0.023
	10	0.178 \pm 0.014	0.476 \pm 0.022	0.514 \pm 0.050	0.552 \pm 0.054	0.650 \pm 0.004	0.648 \pm 0.008	0.664 \pm 0.013
	11	0.370 \pm 0.012	0.533 \pm 0.041	0.584 \pm 0.046	0.587 \pm 0.035	0.676 \pm 0.029	0.684 \pm 0.018	0.684 \pm 0.012
	12	0.453 \pm 0.010	0.644 \pm 0.011	0.638 \pm 0.015	0.650 \pm 0.011	0.659 \pm 0.011	0.670 \pm 0.013	0.672 \pm 0.005
	13	0.178 \pm 0.014	0.510 \pm 0.065	0.584 \pm 0.079	0.633 \pm 0.021	0.652 \pm 0.009	0.664 \pm 0.006	0.668 \pm 0.010
	14	0.370 \pm 0.012	0.560 \pm 0.040	0.603 \pm 0.015	0.632 \pm 0.031	0.668 \pm 0.020	0.675 \pm 0.014	0.684 \pm 0.014
	15	0.453 \pm 0.010	0.644 \pm 0.019	0.643 \pm 0.013	0.662 \pm 0.017	0.679 \pm 0.014	0.685 \pm 0.013	0.677 \pm 0.007
	16	0.178 \pm 0.014	0.630 \pm 0.021	0.637 \pm 0.001	0.647 \pm 0.013	0.651 \pm 0.005	0.668 \pm 0.017	0.658 \pm 0.007
	17	0.370 \pm 0.012	0.623 \pm 0.045	0.647 \pm 0.017	0.655 \pm 0.010	0.684 \pm 0.014	0.688 \pm 0.024	0.685 \pm 0.020
	18	0.453 \pm 0.010	0.652 \pm 0.005	0.655 \pm 0.009	0.664 \pm 0.014	0.684 \pm 0.008	0.669 \pm 0.011	0.663 \pm 0.009
	19	0.178 \pm 0.014	0.650 \pm 0.008	0.659 \pm 0.003	0.663 \pm 0.015	0.673 \pm 0.006	0.676 \pm 0.005	0.667 \pm 0.001
	20	0.370 \pm 0.012	0.689 \pm 0.017	0.677 \pm 0.014	0.679 \pm 0.024	0.684 \pm 0.029	0.688 \pm 0.024	0.671 \pm 0.026
CIFAR-10								
α	0.0	0.2	0.4	0.6	0.8	0.9	0.95	1.0
Member m	1	0.935 \pm 0.003	0.935 \pm 0.003	0.935 \pm 0.003	0.935 \pm 0.003	0.935 \pm 0.003	0.935 \pm 0.003	0.943 \pm 0.001
	2	0.100 \pm 0.000	0.490 \pm 0.027	0.711 \pm 0.030	0.792 \pm 0.014	0.860 \pm 0.003	0.889 \pm 0.003	0.898 \pm 0.005
	3	0.100 \pm 0.000	0.582 \pm 0.043	0.725 \pm 0.027	0.809 \pm 0.004	0.873 \pm 0.008	0.891 \pm 0.002	0.898 \pm 0.004
	4	0.100 \pm 0.000	0.591 \pm 0.009	0.722 \pm 0.008	0.816 \pm 0.009	0.871 \pm 0.008	0.891 \pm 0.000	0.902 \pm 0.001
	5	0.099 \pm 0.000	0.645 \pm 0.036	0.754 \pm 0.012	0.837 \pm 0.011	0.874 \pm 0.002	0.886 \pm 0.004	0.906 \pm 0.003
	6	0.098 \pm 0.000	0.704 \pm 0.010	0.775 \pm 0.002	0.843 \pm 0.012	0.879 \pm 0.006	0.891 \pm 0.003	0.903 \pm 0.003
	7	0.100 \pm 0.000	0.697 \pm 0.012	0.782 \pm 0.004	0.851 \pm 0.008	0.881 \pm 0.003	0.893 \pm 0.006	0.902 \pm 0.002
	8	0.100 \pm 0.000	0.748 \pm 0.015	0.801 \pm 0.011	0.850 \pm 0.006	0.880 \pm 0.006	0.897 \pm 0.003	0.906 \pm 0.004
	9	0.101 \pm 0.000	0.765 \pm 0.025	0.813 \pm 0.020	0.848 \pm 0.004	0.890 \pm 0.004	0.896 \pm 0.005	0.908 \pm 0.001
	10	0.100 \pm 0.000	0.779 \pm 0.007	0.830 \pm 0.007	0.862 \pm 0.011	0.886 \pm 0.012	0.895 \pm 0.004	0.904 \pm 0.003
	11	0.100 \pm 0.000	0.790 \pm 0.003	0.831 \pm 0.003	0.865 \pm 0.005	0.889 \pm 0.007	0.904 \pm 0.001	0.908 \pm 0.004
	12	0.100 \pm 0.000	0.821 \pm 0.009	0.854 \pm 0.013	0.863 \pm 0.008	0.890 \pm 0.002	0.902 \pm 0.004	0.907 \pm 0.003
	13	0.100 \pm 0.000	0.824 \pm 0.001	0.850 \pm 0.002	0.875 \pm 0.003	0.884 \pm 0.009	0.904 \pm 0.003	0.907 \pm 0.002
	14	0.100 \pm 0.000	0.840 \pm 0.002	0.860 \pm 0.008	0.874 \pm 0.009	0.891 \pm 0.002	0.906 \pm 0.006	0.912 \pm 0.004
	15	0.099 \pm 0.000	0.849 \pm 0.007	0.868 \pm 0.006	0.879 \pm 0.004	0.898 \pm 0.003	0.905 \pm 0.001	0.911 \pm 0.000
	16	0.098 \pm 0.000	0.868 \pm 0.009	0.873 \pm 0.006	0.887 \pm 0.004	0.904 \pm 0.004	0.910 \pm 0.003	0.911 \pm 0.002
	17	0.100 \pm 0.000	0.863 \pm 0.003	0.879 \pm 0.003	0.888 \pm 0.004	0.901 \pm 0.005	0.908 \pm 0.002	0.909 \pm 0.002
	18	0.100 \pm 0.000	0.879 \pm 0.004	0.892 \pm 0.005	0.898 \pm 0.002	0.902 \pm 0.005	0.913 \pm 0.005	0.916 \pm 0.001
	19	0.101 \pm 0.000	0.887 \pm 0.007	0.897 \pm 0.004	0.902 \pm 0.001	0.909 \pm 0.002	0.915 \pm 0.002	0.915 \pm 0.001
	20	0.100 \pm 0.000	0.903 \pm 0.005	0.904 \pm 0.003	0.908 \pm 0.002	0.910 \pm 0.004	0.920 \pm 0.002	0.921 \pm 0.001
QualityMRI								
α	0.0	0.2	0.4	0.6	0.8	0.9	0.95	1.0
Member m	1	0.624 \pm 0.033	0.602 \pm 0.020	0.602 \pm 0.020	0.602 \pm 0.020	0.602 \pm 0.020	0.602 \pm 0.020	0.624 \pm 0.033
	2	0.376 \pm 0.008	0.511 \pm 0.050	0.532 \pm 0.023	0.570 \pm 0.046	0.575 \pm 0.020	0.597 \pm 0.057	0.586 \pm 0.055
	3	0.624 \pm 0.008	0.608 \pm 0.015	0.672 \pm 0.027	0.629 \pm 0.026	0.618 \pm 0.015	0.624 \pm 0.055	0.656 \pm 0.042
	4	0.376 \pm 0.008	0.586 \pm 0.015	0.548 \pm 0.103	0.543 \pm 0.020	0.581 \pm 0.013	0.618 \pm 0.027	0.618 \pm 0.053
	5	0.624 \pm 0.008	0.608 \pm 0.027	0.645 \pm 0.035	0.602 \pm 0.042	0.597 \pm 0.070	0.629 \pm 0.013	0.651 \pm 0.038
	6	0.376 \pm 0.008	0.543 \pm 0.020	0.586 \pm 0.040	0.581 \pm 0.023	0.591 \pm 0.020	0.570 \pm 0.008	0.597 \pm 0.035
	7	0.624 \pm 0.008	0.618 \pm 0.042	0.624 \pm 0.020	0.608 \pm 0.020	0.618 \pm 0.030	0.640 \pm 0.042	0.629 \pm 0.013
	8	0.376 \pm 0.008	0.597 \pm 0.099	0.586 \pm 0.027	0.575 \pm 0.046	0.597 \pm 0.060	0.591 \pm 0.027	0.602 \pm 0.020
	9	0.624 \pm 0.008	0.640 \pm 0.020	0.651 \pm 0.055	0.683 \pm 0.046	0.613 \pm 0.023	0.645 \pm 0.013	0.629 \pm 0.013
	10	0.376 \pm 0.008	0.565 \pm 0.035	0.602 \pm 0.008	0.548 \pm 0.035	0.602 \pm 0.059	0.591 \pm 0.038	0.570 \pm 0.008
	11	0.624 \pm 0.008	0.629 \pm 0.026	0.624 \pm 0.020	0.608 \pm 0.027	0.591 \pm 0.033	0.613 \pm 0.066	0.581 \pm 0.013
	12	0.376 \pm 0.008	0.538 \pm 0.015	0.570 \pm 0.055	0.640 \pm 0.008	0.591 \pm 0.008	0.602 \pm 0.038	0.570 \pm 0.030
	13	0.624 \pm 0.008	0.688 \pm 0.053	0.613 \pm 0.013	0.677 \pm 0.023	0.656 \pm 0.008	0.602 \pm 0.042	0.618 \pm 0.042
	14	0.376 \pm 0.008	0.532 \pm 0.035	0.581 \pm 0.035	0.570 \pm 0.046	0.586 \pm 0.020	0.591 \pm 0.020	0.554 \pm 0.033
	15	0.624 \pm 0.008	0.634 \pm 0.040	0.651 \pm 0.008	0.586 \pm 0.050	0.651 \pm 0.059	0.629 \pm 0.023	0.618 \pm 0.050
	16	0.376 \pm 0.008	0.565 \pm 0.023	0.618 \pm 0.040	0.586 \pm 0.062	0.570 \pm 0.030	0.597 \pm 0.026	0.554 \pm 0.008
	17	0.624 \pm 0.008	0.624 \pm 0.040	0.677 \pm 0.035	0.629 \pm 0.047	0.618 \pm 0.027	0.591 \pm 0.053	0.624 \pm 0.040
	18	0.376 \pm 0.008	0.575 \pm 0.055	0.586 \pm 0.008	0.597 \pm 0.013	0.591 \pm 0.020	0.570 \pm 0.008	0.602 \pm 0.027
	19	0.624 \pm 0.008	0.608 \pm 0.050	0.656 \pm 0.046	0.634 \pm 0.055	0.629 \pm 0.040	0.624 \pm 0.038	0.618 \pm 0.040
	20	0.376 \pm 0.008	0.608 \pm 0.027	0.570 \pm 0.065	0.608 \pm 0.020	0.597 \pm 0.026	0.608 \pm 0.020	0.554 \pm 0.020

Table 8: Accuracy per ensemble member of baselines on ChaosNLI, CIFAR-10, and QualityMRI.

ChaosNLI					
		CreWra	CreEns _{α}	CreBNN	CreNet
Member m	1	0.658 \pm 0.013	0.671 \pm 0.011	0.680 \pm 0.013	0.525 \pm 0.051
	2	0.648 \pm 0.012	0.676 \pm 0.013	0.674 \pm 0.031	0.488 \pm 0.098
	3	0.641 \pm 0.020	0.676 \pm 0.013	0.517 \pm 0.079	0.542 \pm 0.030
	4	0.636 \pm 0.012	0.666 \pm 0.008	0.683 \pm 0.030	0.485 \pm 0.037
	5	0.651 \pm 0.032	0.667 \pm 0.011	0.533 \pm 0.112	0.554 \pm 0.051
	6	0.660 \pm 0.016	0.674 \pm 0.021	0.667 \pm 0.008	0.470 \pm 0.115
	7	0.637 \pm 0.014	0.677 \pm 0.009	0.675 \pm 0.012	0.518 \pm 0.036
	8	0.667 \pm 0.016	0.670 \pm 0.010	0.675 \pm 0.029	0.446 \pm 0.030
	9	0.584 \pm 0.086	0.663 \pm 0.005	0.610 \pm 0.104	0.462 \pm 0.113
	10	0.663 \pm 0.035	0.674 \pm 0.029	0.668 \pm 0.017	0.531 \pm 0.064
	11	0.648 \pm 0.017	0.653 \pm 0.015	0.529 \pm 0.096	0.484 \pm 0.047
	12	0.660 \pm 0.016	0.662 \pm 0.016	0.684 \pm 0.019	0.549 \pm 0.071
	13	0.664 \pm 0.014	0.655 \pm 0.017	0.526 \pm 0.105	0.487 \pm 0.040
	14	0.650 \pm 0.011	0.654 \pm 0.006	0.668 \pm 0.030	0.505 \pm 0.039
	15	0.635 \pm 0.038	0.651 \pm 0.018	0.673 \pm 0.009	0.458 \pm 0.143
	16	0.652 \pm 0.026	0.654 \pm 0.007	0.616 \pm 0.108	0.486 \pm 0.012
	17	0.628 \pm 0.019	0.633 \pm 0.015	0.675 \pm 0.030	0.518 \pm 0.057
	18	0.652 \pm 0.009	0.605 \pm 0.021	0.598 \pm 0.109	0.519 \pm 0.031
	19	0.660 \pm 0.012	0.581 \pm 0.037	0.628 \pm 0.116	0.487 \pm 0.050
	20	0.651 \pm 0.013	0.480 \pm 0.074	0.536 \pm 0.116	0.560 \pm 0.043
CIFAR-10					
		CreWra	CreEns _{α}	CreBNN	CreNet
Member m	1	0.942 \pm 0.001	0.955 \pm 0.000	0.875 \pm 0.004	0.941 \pm 0.000
	2	0.944 \pm 0.001	0.954 \pm 0.000	0.877 \pm 0.005	0.943 \pm 0.000
	3	0.944 \pm 0.001	0.952 \pm 0.001	0.867 \pm 0.015	0.944 \pm 0.001
	4	0.943 \pm 0.002	0.952 \pm 0.000	0.881 \pm 0.003	0.942 \pm 0.001
	5	0.943 \pm 0.001	0.952 \pm 0.001	0.872 \pm 0.004	0.942 \pm 0.001
	6	0.943 \pm 0.000	0.952 \pm 0.000	0.869 \pm 0.006	0.942 \pm 0.002
	7	0.941 \pm 0.001	0.952 \pm 0.001	0.877 \pm 0.001	0.943 \pm 0.001
	8	0.942 \pm 0.001	0.951 \pm 0.001	0.880 \pm 0.006	0.943 \pm 0.002
	9	0.943 \pm 0.001	0.951 \pm 0.000	0.879 \pm 0.005	0.944 \pm 0.001
	10	0.943 \pm 0.000	0.953 \pm 0.000	0.873 \pm 0.007	0.943 \pm 0.003
	11	0.942 \pm 0.001	0.953 \pm 0.001	0.882 \pm 0.004	0.943 \pm 0.000
	12	0.943 \pm 0.000	0.953 \pm 0.000	0.879 \pm 0.003	0.942 \pm 0.000
	13	0.943 \pm 0.000	0.953 \pm 0.001	0.874 \pm 0.000	0.943 \pm 0.002
	14	0.945 \pm 0.000	0.952 \pm 0.001	0.872 \pm 0.006	0.941 \pm 0.001
	15	0.942 \pm 0.002	0.948 \pm 0.000	0.856 \pm 0.019	0.942 \pm 0.001
	16	0.942 \pm 0.000	0.941 \pm 0.000	0.870 \pm 0.002	0.942 \pm 0.001
	17	0.942 \pm 0.001	0.930 \pm 0.001	0.854 \pm 0.035	0.942 \pm 0.002
	18	0.944 \pm 0.001	0.921 \pm 0.001	0.872 \pm 0.007	0.943 \pm 0.002
	19	0.943 \pm 0.001	0.906 \pm 0.002	0.873 \pm 0.001	0.942 \pm 0.001
	20	0.942 \pm 0.001	0.872 \pm 0.002	0.860 \pm 0.012	0.942 \pm 0.001
QualityMRI					
		CreWra	CreEns _{α}	CreBNN	CreNet
Member m	1	0.457 \pm 0.099	0.430 \pm 0.170	0.548 \pm 0.115	0.581 \pm 0.137
	2	0.452 \pm 0.139	0.409 \pm 0.163	0.484 \pm 0.115	0.559 \pm 0.112
	3	0.425 \pm 0.124	0.419 \pm 0.160	0.468 \pm 0.126	0.570 \pm 0.110
	4	0.419 \pm 0.126	0.446 \pm 0.187	0.548 \pm 0.115	0.575 \pm 0.102
	5	0.398 \pm 0.135	0.387 \pm 0.139	0.468 \pm 0.126	0.548 \pm 0.126
	6	0.403 \pm 0.117	0.398 \pm 0.158	0.554 \pm 0.118	0.554 \pm 0.084
	7	0.409 \pm 0.175	0.414 \pm 0.161	0.446 \pm 0.107	0.554 \pm 0.095
	8	0.430 \pm 0.119	0.441 \pm 0.182	0.554 \pm 0.118	0.559 \pm 0.133
	9	0.392 \pm 0.137	0.419 \pm 0.181	0.473 \pm 0.110	0.586 \pm 0.141
	10	0.446 \pm 0.177	0.430 \pm 0.140	0.462 \pm 0.130	0.570 \pm 0.118
	11	0.425 \pm 0.137	0.414 \pm 0.153	0.554 \pm 0.118	0.602 \pm 0.107
	12	0.398 \pm 0.122	0.430 \pm 0.153	0.554 \pm 0.118	0.565 \pm 0.103
	13	0.398 \pm 0.130	0.430 \pm 0.140	0.446 \pm 0.107	0.575 \pm 0.090
	14	0.409 \pm 0.066	0.430 \pm 0.142	0.457 \pm 0.122	0.581 \pm 0.070
	15	0.409 \pm 0.146	0.414 \pm 0.132	0.468 \pm 0.126	0.624 \pm 0.133
	16	0.414 \pm 0.107	0.425 \pm 0.167	0.554 \pm 0.118	0.608 \pm 0.100
	17	0.419 \pm 0.139	0.414 \pm 0.119	0.554 \pm 0.118	0.581 \pm 0.091
	18	0.403 \pm 0.168	0.414 \pm 0.129	0.468 \pm 0.126	0.581 \pm 0.117
	19	0.425 \pm 0.151	0.414 \pm 0.073	0.554 \pm 0.118	0.565 \pm 0.142
	20	0.430 \pm 0.154	0.446 \pm 0.068	0.548 \pm 0.117	0.581 \pm 0.103