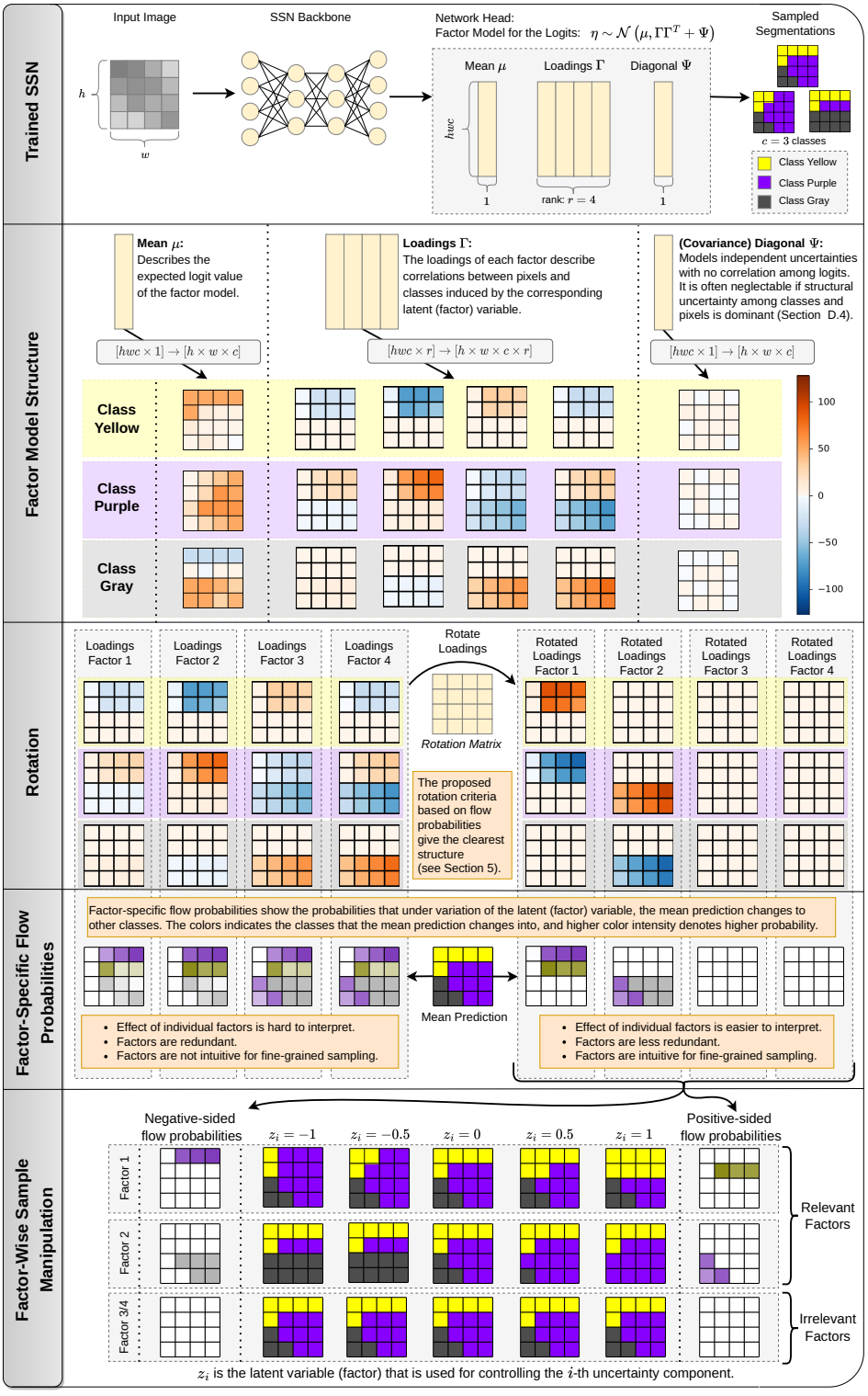
Supplement: Structuring Uncertainty for Fine-Grained Sampling in Stochastic Segmentation Networks (NeurIPS 2022)

A Overview



B Transformation of the latent factor variables in factor models

Here, we show that the marginal distribution of the observed variables in factor models is not affected by certain transformations of the latent variables.

Lemma 1. *Let*

$$(\boldsymbol{\eta}, \boldsymbol{y}) = (\boldsymbol{\eta}, \boldsymbol{A}\boldsymbol{z}) = \begin{pmatrix} \boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} \end{pmatrix} \begin{pmatrix} \boldsymbol{\eta} \\ \boldsymbol{z} \end{pmatrix}$$

be the factor model after a linear transformation of the latent variables $\boldsymbol{z}$ by some matrix $\boldsymbol{A}^{r\times r}$. Then, the joint density of the factor model with transformed latent variables $\boldsymbol{y}$ is given by

$$p(\boldsymbol{\eta}, \boldsymbol{y}) \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi} & \boldsymbol{\Gamma}\boldsymbol{A}^\top \\ \boldsymbol{A}\boldsymbol{\Gamma}^\top & \boldsymbol{A}\boldsymbol{A}^\top \end{pmatrix} \right).$$

The marginal distribution $p(\boldsymbol{\eta}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi})$ is independent from $\boldsymbol{A}$, and for the conditional distribution it holds that

$$p(\boldsymbol{\eta} \,|\, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{A}^{-1}\boldsymbol{y}, \boldsymbol{\Psi}).$$

Proof. Recall that the factor model before the transformation has the joint multivariate Gaussian density

$$(\boldsymbol{\eta}, \boldsymbol{z}) \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi} & \boldsymbol{\Gamma} \\ \boldsymbol{\Gamma}^\top & \boldsymbol{I}_r \end{pmatrix} \right).$$

By standard linear transformation of multivariate Gaussian random vectors it holds that

$$\begin{aligned}(\boldsymbol{\eta}, \boldsymbol{y}) = (\boldsymbol{\eta}, \boldsymbol{A}z) &\sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} \end{pmatrix} \begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} \boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi} & \boldsymbol{\Gamma} \\ \boldsymbol{\Gamma}^\top & \boldsymbol{I}_r \end{pmatrix} \begin{pmatrix} \boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} \end{pmatrix}^\top \right) \\ &= \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi} & \boldsymbol{\Gamma}\boldsymbol{A}^\top \\ \boldsymbol{A}\boldsymbol{\Gamma}^\top & \boldsymbol{A}\boldsymbol{A}^\top \end{pmatrix} \right).\end{aligned}$$

The parameters of a marginal distribution of a multivariate Gaussian distribution are given by the projections onto the dimensions that are kept in the marginal, hence the invariance of $p(\boldsymbol{\eta})$ under the transformation $\boldsymbol{A}$. By general formulas for conditionals of multivariate Gaussian distributions (using the Schur complement) it holds that

$$\begin{aligned}p(\boldsymbol{\eta} \,|\, \boldsymbol{y}) &= \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top)^{-1}\boldsymbol{y}, \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \boldsymbol{\Psi} - \boldsymbol{\Gamma}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{\Gamma}^\top) \\ &= \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{A}^{-1}\boldsymbol{y}, \boldsymbol{\Psi})\end{aligned}$$

For the last line, we used that $(\boldsymbol{A}\boldsymbol{A}^\top)(\boldsymbol{A}\boldsymbol{A}^\top)^{-1} = \boldsymbol{I}_r$, which yields $\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top)^{-1} = \boldsymbol{A}^{-1}$ by multiplying $\boldsymbol{A}^{-1}$ from the left. By afterwards multiplying $\boldsymbol{A}$ from the right we also get $\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top)^{-1}\boldsymbol{A} = \boldsymbol{I}_r$. □

If it holds $\boldsymbol{A} = \boldsymbol{O}^\top$ for an orthogonal matrix $\boldsymbol{O} \in \mathbb{R}^{r\times r}$ in Lemma 1, then the orthogonality of the latent factor variables is preserved as then the covariance matrix of the transformed latent factor variables is $\boldsymbol{A}\boldsymbol{A}^\top = \boldsymbol{I}_r$. Orthogonal rotation criteria are concerned with this type of rotations.

Another class of rotation criteria are obligue rotations. These rotations allow non-orthogonal latent factor variables. For obligue rotations, the only requirement for $\boldsymbol{A}$ is that the diagonal elements of the covariance matrix $\boldsymbol{A}\boldsymbol{A}^\top$ of the latent factor variables are all equal to one, in other words, the rows of $\boldsymbol{A}$ are vectors of length 1. However, we focus on orthogonal rotations because they yield independent uncertainty components since the controlling latent variables are orthogonal.

C Flow probabilities

C.1 Visualization of flow probabilities

Let $\boldsymbol{F} \in \mathbb{R}^{wh\times c}$ be a matrix of flow probabilities. Let $\text{color}(k) \in [0,1]^3$ denote a RGB color for the k-th class. We visualize the uncertainty encoded in $\boldsymbol{F}$ by mixing colors as

$$\sum_{k\in[c]} \text{color}(k)^\top \max(\boldsymbol{F}_{:,k}, \boldsymbol{0}) \in [0,1]^{wh\times 3},$$

where the max cuts off negative flow probabilities, that is, for each pixel, the class that is predicted from the mean logits is not considered (remember that flow probabilities highlight the change from the mean prediction).

C.2 Additional material to support theoretical results from the main paper

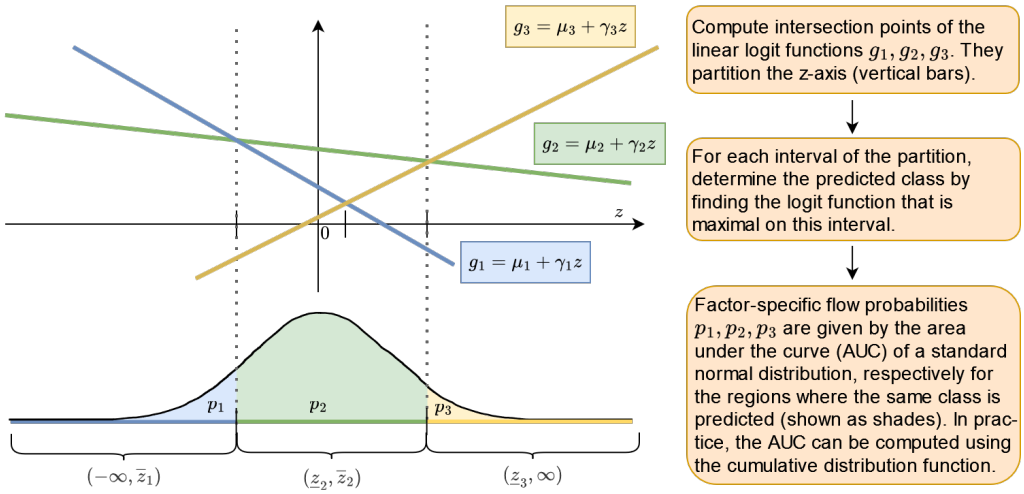


Figure 5: Computation of factor-specific flow probabilities for a single pixel based on its logit functions. For simplicity, here we leave out the pixel index in the notation that we used for Proposition 1.

Proposition 1 in the main paper follows directly from the preceding derivation. In it, the intervals $(\underline{z}_{ik}, \overline{z}_{ik})$ of z-values for which the k-th class is predicted are respectively computed for each pixel $i \in [wh]$ and class $k \in [c]$. Corollary 1 in the main paper is the result of restricting these intervals to positive (negative) values. In the matrix notation of Proposition 1 and Corollary 1, this amounts to computing the element-wise maxima $\max(\mathbf{0}, \overline{\boldsymbol{Z}})$ and $\max(\mathbf{0}, \underline{\boldsymbol{Z}})$ for positive, and correspondingly element-wise minima for negative factor-specific flow probabilites. As in Proposition 1, one-sided flow probabilities are obtained by an element-wise application of the cumulative distribution function ψ of a standard normal random variable.

C.3 Uncertainty overview plots

Recall from Section 3 in the main paper that we compute the probabilities that classes are predicted as an expected value over the distribution of the logits:

$$P^{\text{full}} = \int E(\boldsymbol{\eta}) p(\boldsymbol{\eta}) d\boldsymbol{\eta} = \int E(\boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{z} + \boldsymbol{\Psi}^{1/2}\boldsymbol{\varepsilon}) p(\boldsymbol{z}) p(\boldsymbol{\varepsilon}) d\boldsymbol{z} d\boldsymbol{\varepsilon} \in [0,1]^{(wh)\times c}.$$

Here, we also discuss an alternative and why we do not use it in the end: By replacing the hard argmax function E with a softmax function, we can also compute softmax class probabilities:

$$\begin{aligned} P^{\text{full},s} &= \int \text{softmax}(\boldsymbol{\eta}) p(\boldsymbol{\eta}) d\boldsymbol{\eta} \\ &= \int \text{softmax}(\boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{z} + \boldsymbol{\Psi}^{1/2}\boldsymbol{\varepsilon}) p(\boldsymbol{z}) p(\boldsymbol{\varepsilon}) d\boldsymbol{z} d\boldsymbol{\varepsilon} \in [0,1]^{(wh)\times c}, \end{aligned}$$

where the softmax function is applied pixel-wise (along the class dimension). Analogously to Section 3 from the main paper, we define full *softmax* flow probabilities as

$$F^{\text{full},s}(\boldsymbol{\Gamma}) = P^{\text{full},s} - \text{softmax}(\boldsymbol{\mu}).$$

Softmax flow probabilities are less practical for two reasons: First, by our judgement the color-mixing visualization scheme from Section C.1 looks less good for softmax flow probabilities because more colors contribute to each pixel and it is much harder for the human eye to map the mixed colors

to the original classes (see Figures 9 and 12 below)–generally, uncertainty visualizations based on softmax probabilities are less sharp. The second reason is that *factor-specific* softmax flow probabilities (defined analogously as in Section 3.1 from the main paper) do not have closed-form analytic solutions, which is why they were not suitable for (exact) optimization.

In Figures 6, 9, and 12 below, we show full flow probability (FP) and full softmax flow probability (S-FP) overview plots for the predicted uncertainties. In addition, we show overview plots obtained by computing pixel-wise entropy measures [27] for both hard and softmax class probabilities, that is,

$$E = -(P^{\text{full}} \odot \log(P^{\text{full}}))\mathbb{1}_c \qquad \text{and} \qquad E^s = -(P^{\text{full},s} \odot \log(P^{\text{full},s}))\mathbb{1}_c,$$

where here $\odot$ denotes element-wise multiplication, and the vector $\mathbb{1}_c$ of c ones is used to sum over rows. The visualizations in Figures 6, 9, and 12 below show that entropy and softmax entropy visualizations look much alike, however, softmax entropy yields slightly smoother visualizations. Since entropy measures aggregate over classes, information about class-specific uncertainty is lost. Flow probabilities act complementary by including this information in the visualization.

Finally, confidence scores are another overview plot for pixel-wise uncertainties that is calculated by simply taking the pixel-wise maximum of the expected softmax output $P^{\text{full},s}$ along the class dimension.

D Additional material for experiments

D.1 Data sets and training

The training of all models was conducted on Tesla V100 GPUs with 32GiB RAM.

LIDC [1]. For the data processing we used the provided implementations supplemented to [31] and providing a processed split of $9\,375$ slices for training, $2\,474$ slices for validation, and $3\,247$ slices for testing. Following [31], we ran the training procedure for $500\,000$ iterations with a batch size of 12. For each prediction the loss was evaluated based on 20 samples from the logit distribution and on one ground truth label randomly chosen from the four proposed labels. For the optimization we used the Adam optimizer with default parameters, including a learning rate of 0.001.

SEN12MS [36]. We build upon the implementations provided in [37] and also make use of the proposed data splits into $162\,556$ patches for training and $18\,106$ patches for validation and testing. We follow the training process from [37] and use a UNet, only adapting the network output and the loss to the stochastic segmentation network setup from [31]. Hence, we used the RMSprop optimizer with a learning rate of 0.01 and a momentum of 0.9, weight decay of 0.0005, and a batch size of 32. The training process was run for 100 epochs.

CamVid [5]. The CamVid data set consists of 701 images of 360×480 pixels, split into 367, 101, and 233 images for training, validation, and testing. The scenes show scenes from the perspective of a driver, segmented into 11 different classes. As a backbone, we use the fully convolutional dense net with the same training strategy as the one commonly used in other works: [27] For the optimization, we use RMSprop with a learning rate of 0.001 and early stopping with a patience of 100 as termination criterion.

Quality of trained models. As we noted in the main paper, we do not benchmark SSNs because their capability of yielding state-of-the-art results has already been proven [24, 31]. Nevertheless, for some context, here we report the primary metrics for which reference values exist in the papers from which we took baseline models. LIDC: [31] already used the LIDC data set and extensively compared the SSN approach, see their experiments. However, our reimplemented model has a slight performance degradation compared to the one from [31], for instance, DSC score drops from 43.3 to 39.7. This is because we barely tuned our model since for our purpose (structuring the uncertainty), the quality of the model was already sufficient. SEN12MS: Here, the main metric reported in [37] is average accuracy with a reported value of slightly above 60. Without much tuning, we achieved a value of 59.21 on the test split. CamVid: [27] only report intersection over union with the best score at 0.67. Our model achieves a score that is slightly below at 0.63, where again we omitted extensive tuning because for our purpose (structuring uncertainty predictions) it was not necessary. Even without much tuning, modifying deterministic backbones into SSNs did not hurt segmentation performance much, and we believe that with further tuning results could be brought at least on par.

Copyright and personal information. All data sets that we use for this work (LIDC [1], SEN12MS [36], CamVid [5]) can be used for academic purposes and do not contain any personally identifiable information or offensive content. For our implementations, we build on freely available resources. For the work with the LIDC data, we used the pre-processed data provided in Stefan Knegt's repository, which is under Apache-2.0 license. Further, we made use of a pre-implemented data loader provided in Miguel Monteiro's repository [3], which is also under Apache-2.0 license. For the work with the SEN12MS data set, we used the available data handling from the official SEN12MS repository. For the work with the CamVid data set, we adapted implementations from this repository for the training and evaluation pipeline (MIT license).

D.2 Examples for the data sets

D.2.1 LIDC

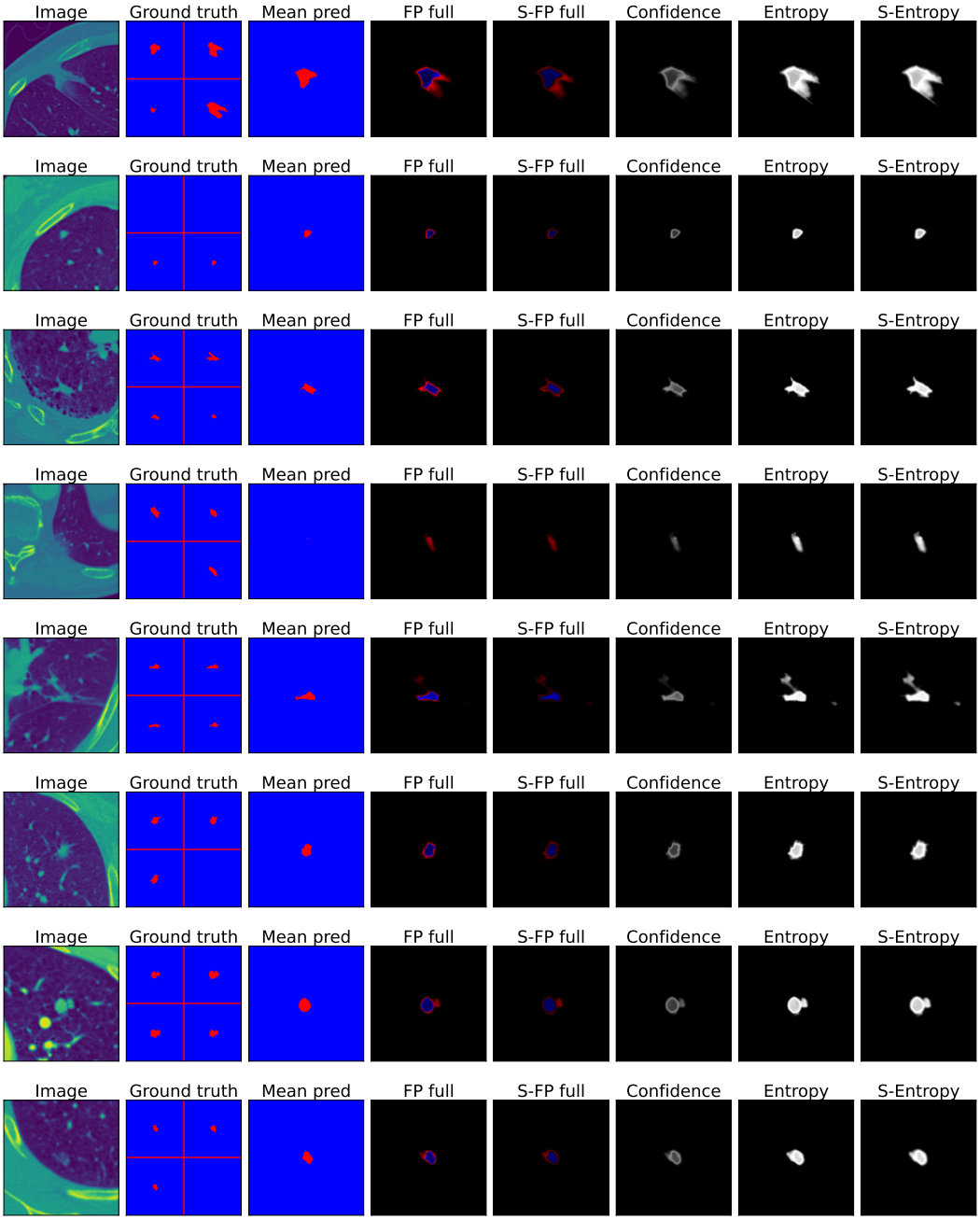


Figure 6: Images from the LIDC test data set with ground truth shown in the second column (split into four patches to show labels from the four experts–lung nodules are red, background is blue) and mean prediction shown in the third column. Overview plots for the predicted uncertainty are shown in the remaining columns: Flow probabilities are denoted by FP, the prefix 'S' is used for softmax versions. Bright colors indicate high uncertainty.

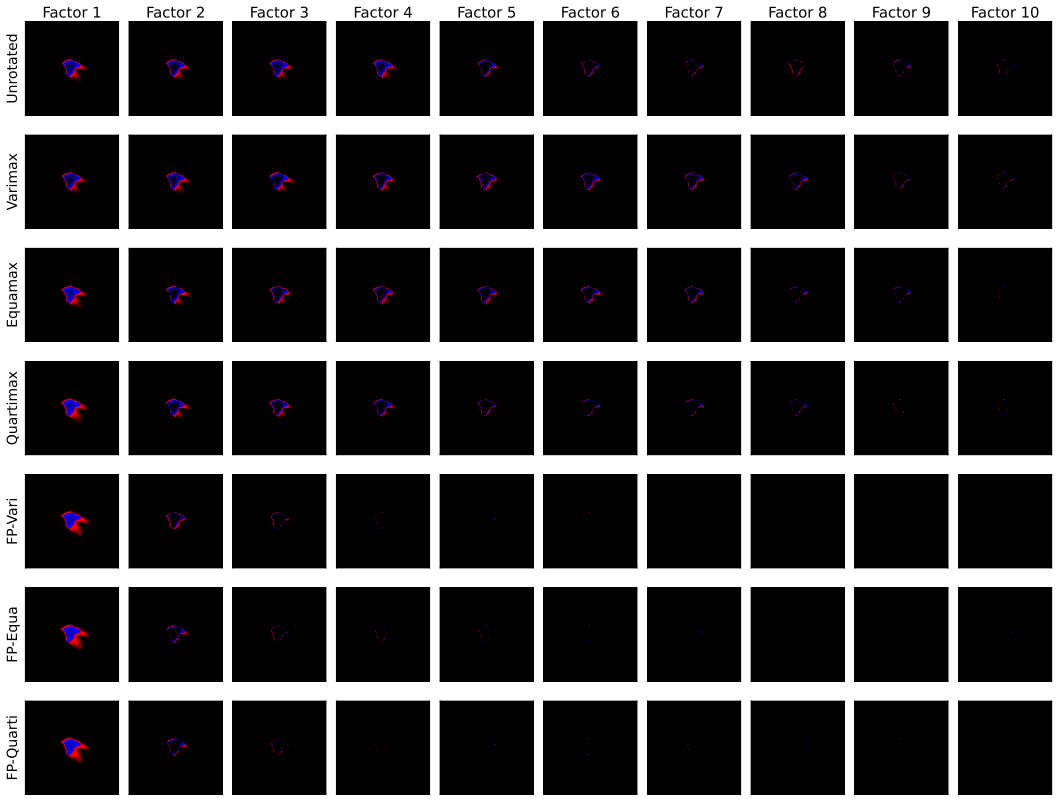


Figure 7: Predicted segmentation uncertainty for an image from the LIDC data set [37]. Factor-specific flow probabilities are shown for different rotations of the latent factor variables. Bright colors mean that the prediction changes with high probability to the class indicated by the color. Flow probability rotations like FP-Quartimax greatly reduce redundancy.

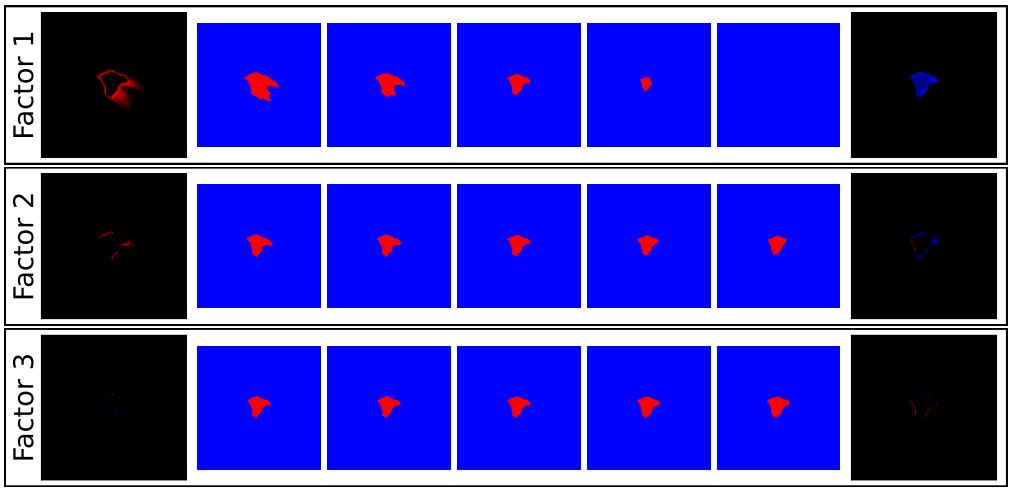


Figure 8: Pseudo-samples obtained by an *individual* manipulation of the three most relevant factors of the FP-Quartimax rotation for the example image from Figure 7. For each factor, five pseudo-samples are respectively obtained by setting the associated factor to $-1, -0.5, 0, 0.5, 1$, while keeping all other factors fixed to zero. Left- and right-sided flow probabilities are shown to the left and right of the boxes. Already the second factor has little impact with nearly zero factor-specific flow probabilities.

D.2.2 SEN12MS

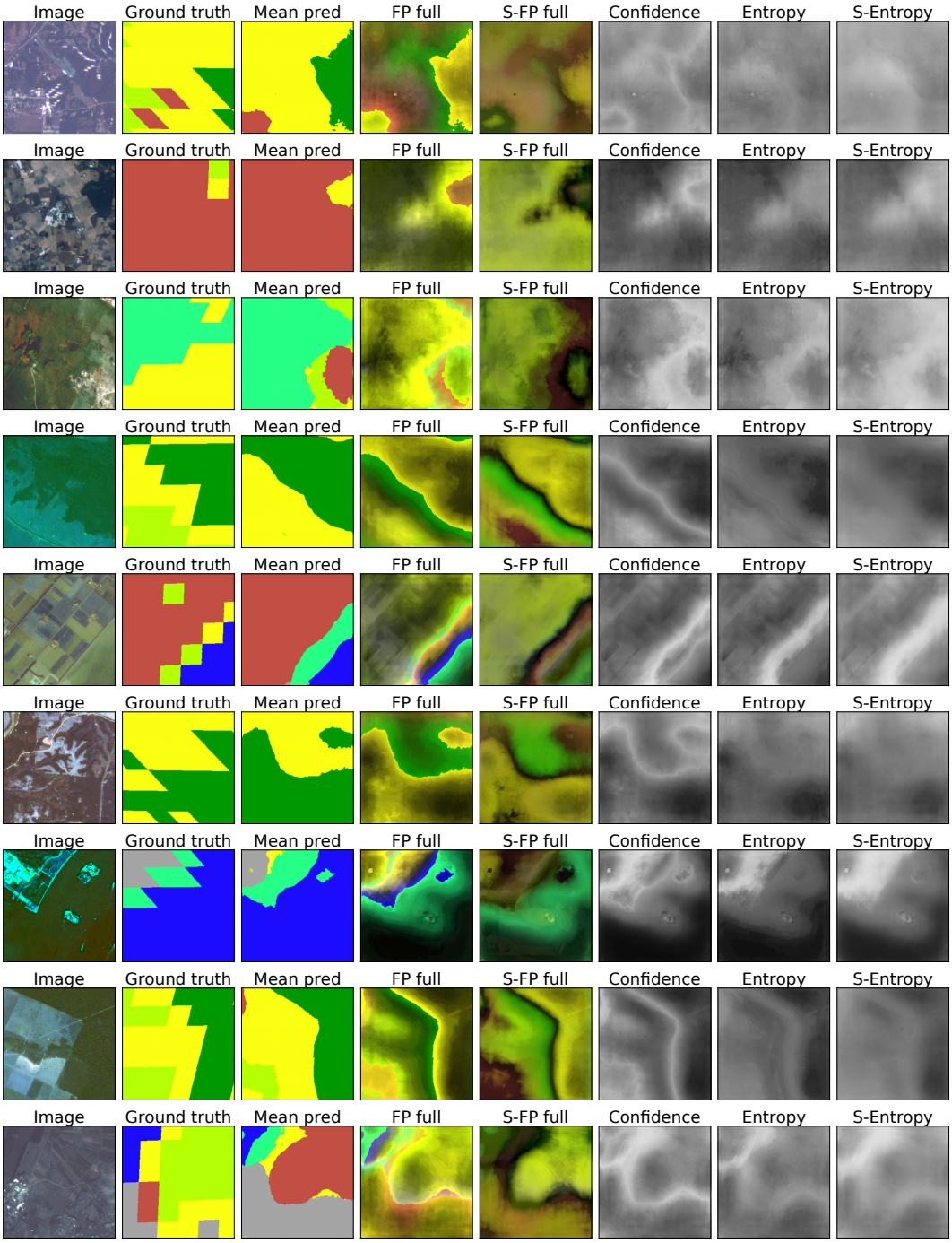


Figure 9: Images from the SEN12MS test data set with ground truth shown in the second column (class colors as in Figure 1 from the main paper) and mean prediction shown in the third column. Overview plots for the predicted uncertainty are shown in the remaining columns: Flow probabilities are denoted by FP, the prefix 'S' is used for softmax versions. Bright colors indicate high uncertainty.

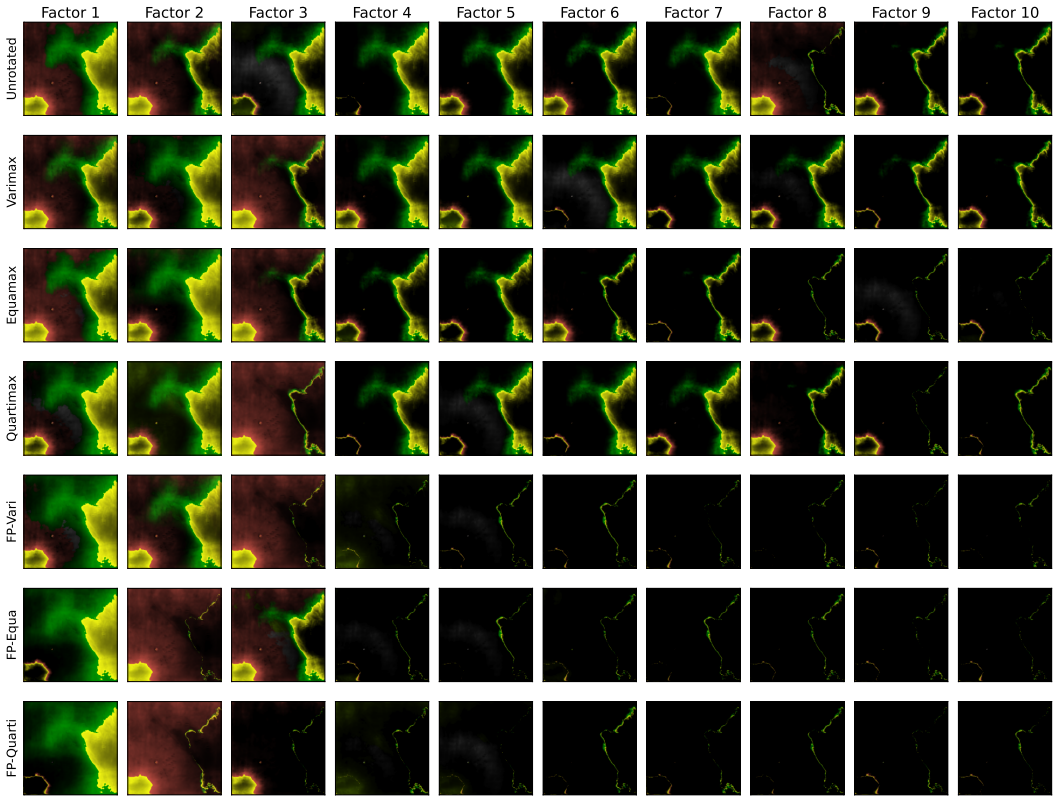


Figure 10: Predicted segmentation uncertainty for an image from the SEN12MS data set [37]. Factor-specific flow probabilities are shown for different rotations of the latent factor variables. Flow-probability rotations like FP-Quartimax greatly reduce redundancy.

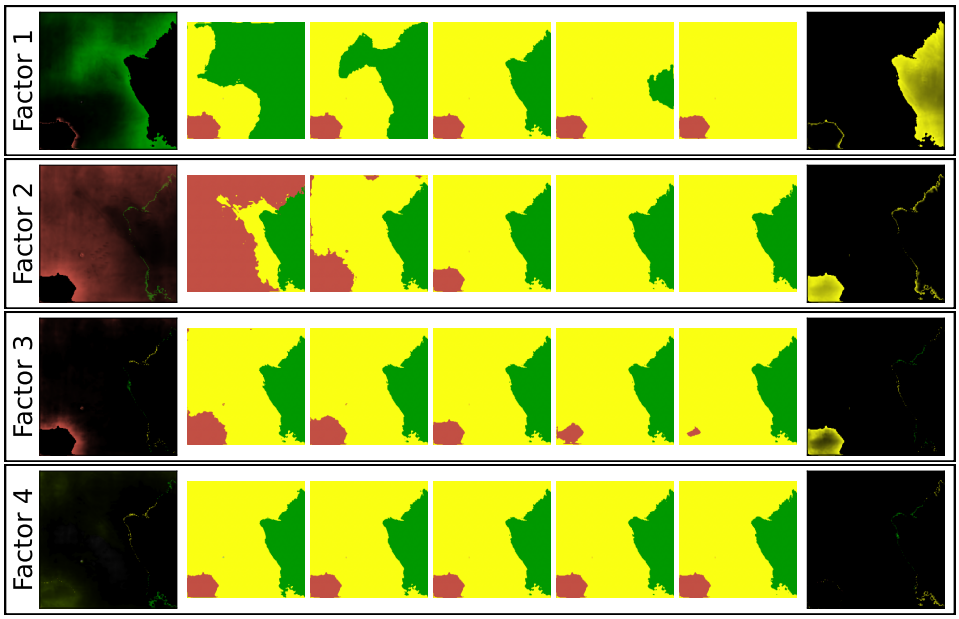


Figure 11: Pseudo-samples obtained by an *individual* manipulation of the four most relevant factors of the FP-Quartimax rotation for the example image from Figure 10 (compare also Figures 1 and 4 from the main paper). For each factor, five pseudo-samples are respectively obtained by setting the associated factor to $-1, -0.5, 0, 0.5, 1$, while keeping all other factors fixed to zero. The fourth factor has little impact, which can also be deduced from its nearly zero flow probabilities.

D.2.3 CamVid

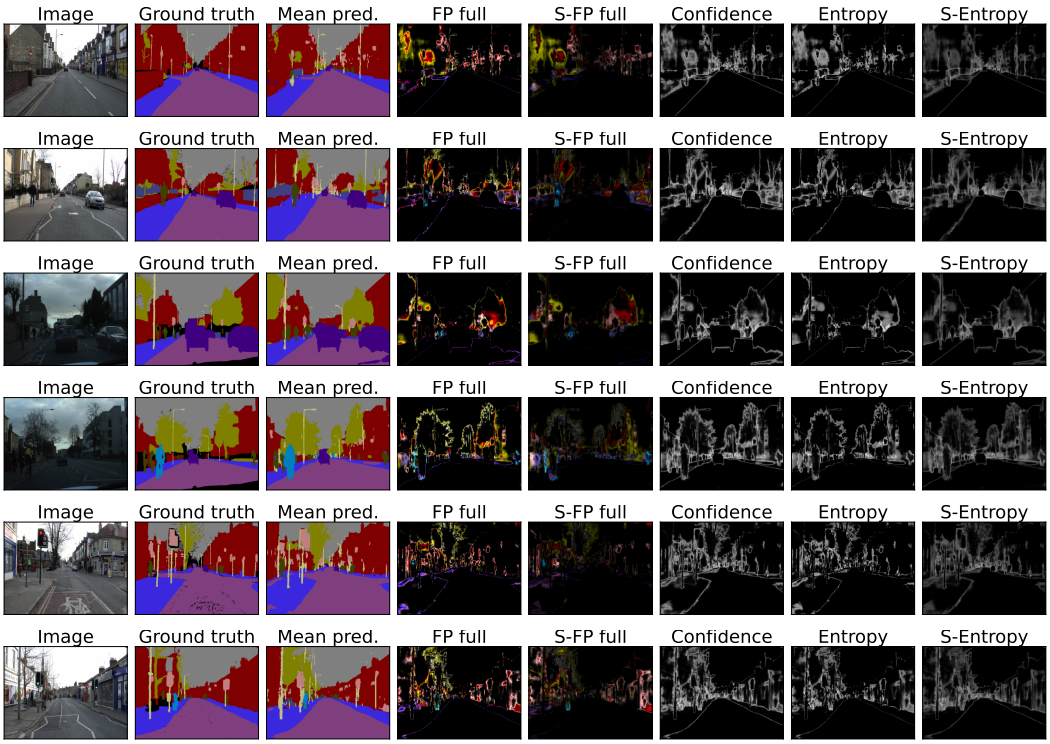


Figure 12: Images from the CamVid test data set with ground truth shown in the second column and mean prediction shown in the third column. Overview plots for the predicted uncertainty are shown in the remaining columns: Flow probabilities are denoted by FP, the prefix 'S' is used for softmax versions. Bright colors indicate high uncertainty.

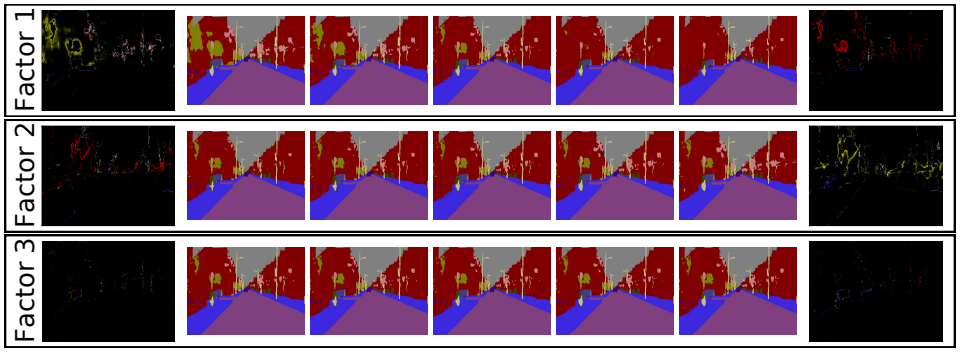


Figure 13: Pseudo-samples obtained by an *individual* manipulation of the three most relevant factors of the FP-Quartimax rotation for the first image from Figure 12. For each factor, five pseudo-samples are respectively obtained by setting the associated factor to $-1.5, -0.75, 0, 0.75, 1.5$, while keeping all other factors fixed to zero. Factor 1 represents the uncertainty associated with the trees on the left side (yellow color code) and Factor 2 represents the uncertainty that is caused by the windows on the right side (pink color code). Factor 3 has little impact: It has nearly zero flow probabilities.

D.3 Additional results on factor relevance, separability, and sparsity

In the following pages, we provide additional material and results about factor relevance, separability, and sparsity.

First, Figures 17, 18, and 19 show more details on the relevance metric n_τ, particularly here we show standard deviations that we did not include in Figure 3 from the main paper for clarity. The standard deviations can be quite large because the amount of predicted uncertainty can differ greatly across test images. This implies that also the number of relevant factors fluctuates across test images. However, the qualitative differences between rotations remains: FP rotations reduce redundancy and tend to reduce the number of relevant factors to a necessary minimum.

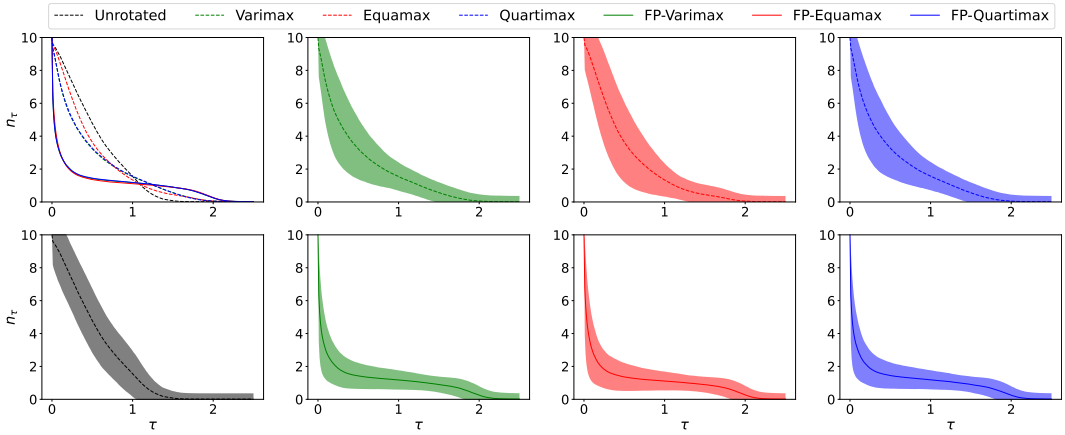


Figure 14: LIDC: Number of relevant factors n_τ including standard deviations.

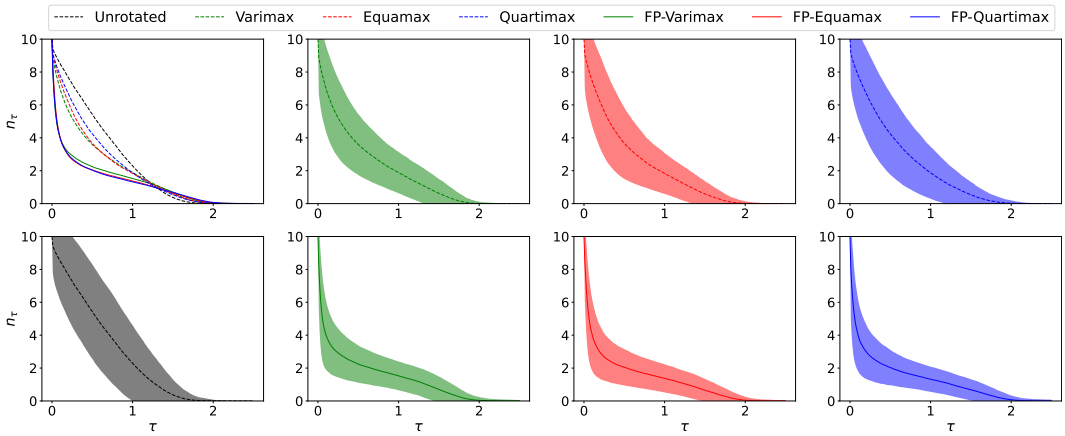


Figure 15: SEN12MS: Number of relevant factors n_τ including standard deviations.

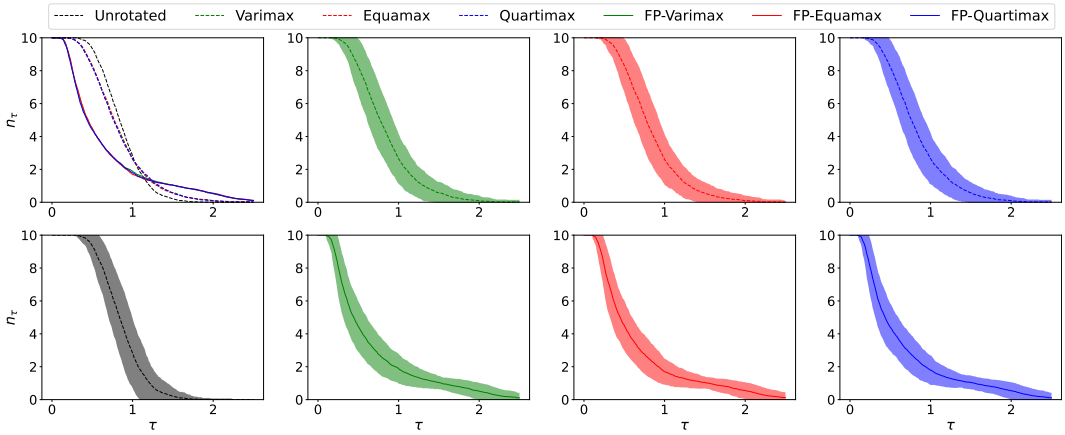


Figure 16: CamVid: Number of relevant factors n_τ including standard deviations.

Next, Figures 17, 18, and 19 give more details on the separability metric (bottom of Figure 3 from the main paper), including standard deviations.

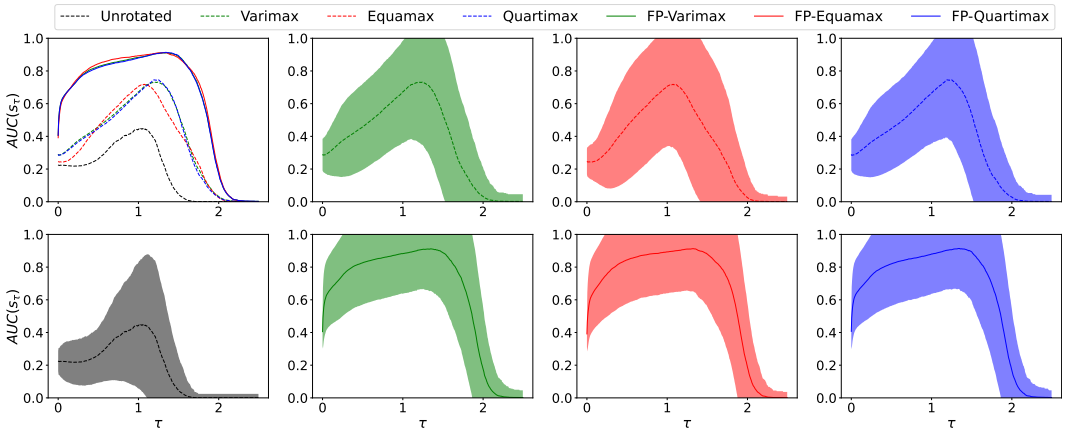


Figure 17: LIDC: Degree of separation among the relevant factors, based on cosine similarities of factor-specific flow probabilities.

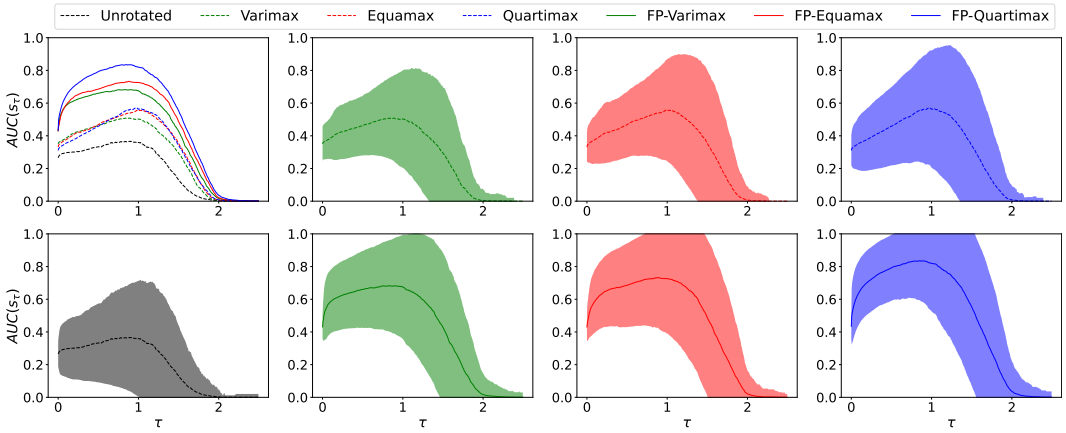


Figure 18: SEN12MS: Degree of separation among the relevant factors, based on cosine similarities of factor-specific flow probabilities.

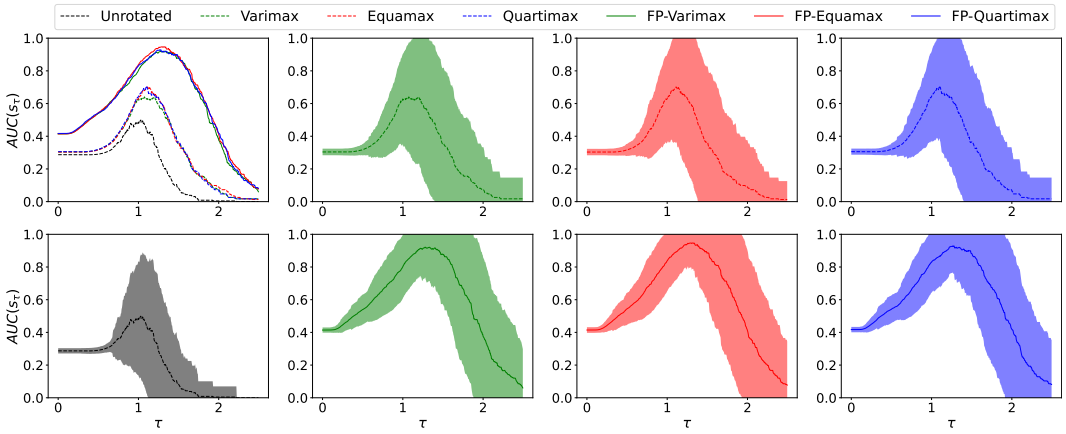


Figure 19: CamVid: Degree of separation among the relevant factors, based on cosine similarities of factor-specific flow probabilities.

Finally, Figure 20 visualizes the results from Table 1 from the main paper about row sparsity. Again, it can be seen that FP rotation yield the most row sparse representations, that is, uncertainty components are disentangled in the sense that they tend to affect distinguished regions of the input image.

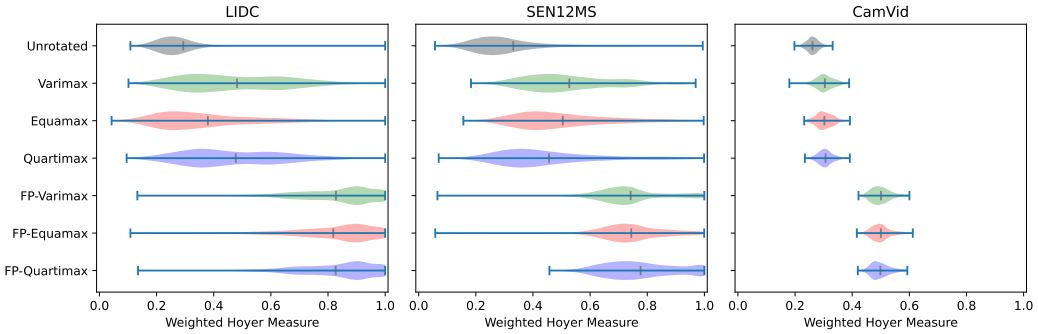


Figure 20: Weighted Hoyer measure for different rotations evaluated on the LIDC, SEN12MS, and CamVid test data sets. The vertical bars respectively show minimum, mean, and maximum values.

D.4 Importance of the diagonal for factor models in SSNs

For representing the uncertainty from the factor model, we largely focus on the structural component $\mathbf{\Gamma} \boldsymbol{z}$ that accounts for the indirect correlations among the logits induced by the latent factor variables $\boldsymbol{z}$. We thereby neglect the diagonal $\mathbf{\Psi}$, which contains scaling parameters for independent noise contributions to the logits. Here, we show empirically that leaving out the diagonal often does not lead to significant differences in sampled segmentations. For that, we compute flow probabilities for the reduced factor model *without* the diagonal noise terms, that is, for Monte Carlo samples $z^{(1)}, \dots, z^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_r)$ for the latent variables we calculate

$$F^{\text{full}-\mathbf{\Psi}} \approx \frac{1}{m} \sum_{j=1}^{m} E(\boldsymbol{\mu} + \mathbf{\Gamma} \boldsymbol{z}^{(j)}) - E(\boldsymbol{\mu}) \in [-1, 1]^{(wh) \times c}.$$

The results in Table 1 show the sample diversities [31] and the scaled ℓ_1-norms of $\|F^{\text{full}-\Psi}\|_1$ and $\|F^{\text{full}}\|_1$ of the flow probabilities for the factor model, with and without considering the diagonal $\mathbf{\Psi}$. As a scaling factor, we use $100/(wh)$ to normalize w.r.t. the image size for easing comparison. Further, to show that the diagonal does also affect the location of the uncertainty only to a minor extent, we compute the scaled differences $100 \cdot \|F^{\text{full}-\Psi} - F^{\text{full}}\|_1/(wh)$. For computing the sample diversity and the full flow probabilities, we respectively use 100 Monte Carlo samples per image.

The results clearly show that in expectation, the diagonal has only a minor effect on sampled segmentations. The high standard deviations for sample diversities and full flow-probabilities are due to variations in the overall uncertainty across different images from the test data sets. The scaled FP differences show that the expected uncertainty per pixel is almost not affected. As a conclusion, it seems sufficient to focus on the structured part for the representation of the uncertainty.

Table 1: Evaluation of the effect of the diagonal $\mathbf{\Psi}$ of the factor model on the resulting predictions. The effect is evaluated based on the changes in the sample diversity and the resulting flow probabilities. The results are stated as the mean and the standard deviation over the test data sets of the LIDC, SEN12MS, and CamVid.

	Sample Diversity		Scaled L1-FP		Scaled FP-Diff
	w/ diag	w/o diag	w/ diag	w/o diag	
LIDC	0.296±0.051	0.296±0.051	0.358± 0.478	0.358± 0.478	$9.2{\cdot}10^{-5}{\pm}9.8{\cdot}10^{-5}$
SEN12MS	0.261±0.075	0.260±0.075	35.018±11.392	35.017±11.392	0.0011±0.0009
CamVid	0.256±0.034	0.256±0.034	2.349± 0.745	2.349± 0.745	0.0001±0.0001

D.5 Computational performance

All computations in this section were performed on an Intel Xeon Platinum 8260 using Python 3.7.

Flow probabilities. For a given factor model, the computation times of the factor-specific flow probabilities depends linearly on the number of pixels and the rank, and it depends quadratically on the number of classes since our implementation relies on the calculation of intersection points of the c straight lines g_{ik}, $k \in [c]$ (see the main paper). Table 2 shows the computation times on a single core for the data sets that we used. Factor-specific flow probabilities for the unrotated representation compute slightly faster because no new factor loadings need to be calculated (currently, we store only rotation matrices for memory efficiency). Next, the computation time of full flow probabilities scales linearly in the number of used Monte Carlo samples. For 100 Monte Carlo samples, a single-core computation of the full flow probabilities took 0.083 ± 0.001 seconds for LIDC, 4.268 ± 0.261 seconds for SEN12MS, and 15.58 ± 0.48 seconds for CamVid (standard deviations over 500 test images, selected uniformly at random). There is a huge potential for speed-ups using parallelization.

Table 2: Mean and standard deviation for the computation times (in seconds) of the factor-specific flow probabilities on the LIDC, SEN12MS, and CamVid data sets. The values are computed based on respectively 500 randomly selected test images.

	LIDC	SEN12MS	CamVid
Unrotated	0.032±0.007	3.807±0.105	18.87±0.63
Varimax	0.042±0.005	3.788±0.103	17.87±0.39
Equamax	0.044±0.006	3.772±0.107	17.86±0.40
Quartimax	0.041±0.005	3.778±0.108	17.86±0.39
FP-Varimax	0.042±0.004	3.782±0.113	17.89±0.41
FP-Equamax	0.044±0.005	3.781±0.111	17.88±0.39
FP-Quartimax	0.042±0.007	3.781±0.112	17.89±0.40

Optimization. For optimizing w.r.t. the different rotation criteria, we adapted projected gradient descent algorithms from [4]. Table 3 shows the computation times, where we report average cost per iteration and the total cost for the optimization, respectively. For the interpretation of these running times, we note that at this point our implementation of the optimization procedure is prototypical.

Generally, optimization is faster for smaller problem dimensions (size of input image, number of classes, and rank). Therefore, computation times for LIDC are significantly faster than those for SEN12MS and CamVid. Because of the more complex objective function, optimizing flow-probability based rotation criteria takes roughly ten times more time than optimizing classic rotation criteria. Given the non-convex nature of the optimization problems, especially when using flow-probability based rotation criteria, it may be useful to perform multiple restarts. Moreover, an initialization with solutions from classic rotation criteria as a warm start may be helpful. For the results presented in the paper, we respectively did five restarts with random initializations (Table 3 reports average cost per run). With our implementation, rotations should be pre-computed whenever possible. However, if desired, we also see great potential for increasing performance, for instance, by improving the optimization scheme, introducing more parallelization, and engineering Python-specific bottlenecks.

Table 3: Mean computation time and standard error in seconds for the computation of the different rotations: average cost per single iteration and total time for the optimization, respectively for the LIDC, SEN12MS, and CamVid data sets. The computation times are based on five repetitions per image, averaged over all images from the respective test data sets.

	LIDC		SEN12MS		CamVid	
	Iteration	Total	Iteration	Total	Iteration	Total
Varimax	0.071±0.006	5.43±1.18	2.289±0.093	100.05± 25.23	6.19±0.11	625.65± 11.52
Equamax	0.067±0.005	5.11±0.81	2.544±0.776	217.48± 21.68	6.17±0.12	624.00± 11.96
Quartimax	0.038±0.002	3.30±0.51	1.536±0.051	144.41± 12.00	3.84±0.09	388.18± 9.55
FP-Vari	0.767±0.033	65.34±4.84	15.878±1.001	1016.12±259.62	51.17±3.17	5161.89±318.28
FP-Equa	0.774±0.034	65.57±5.14	15.878±0.992	1055.14±279.66	51.60±3.86	5176.84±383.96
FP-Quarti	0.727±0.031	62.42±4.12	15.971±0.909	1048.48±267.65	48.45±2.81	4858.54±279.20

E Interface for fine-grained sample control

Alongside the anonymous code repository accompanying this submission, we provide a video for a quick impression how the interface (demo) works. Below in Figures 21, 22, and 23 we show sample views of the interface for the demo. The interface is comprised of the following elements:

- top: drop-down menu to select image file and load pre-computed rotations
- top row: display of image and available ground truth labels (can be more, see the LIDC example in Figure 23)
- control row that consists of:
  - drop-down menu to select the rotation,
  - a *Reset* button to set all control variables to zero,
  - a *Resample* button to sample a random segmentation from the factor model for the predicted uncertainty,
  - a field *pseudo density*, which we included with the aim of giving an impression of how likely a sample is. We compute it as the ratio of (a) the density value of the factor model with latent variables set according to values of corresponding sliders and noise variables set to zero and (b) the value of the factor-model density at the mean (normalization constant).
- bottom left: a box that contains sliders to control the contributions of individual factors/uncertainty components: the effect of each uncertainty component is visualized by one-sided flow probabilities
- bottom right: the fixed mean prediction, and below the prediction when all factors/uncertainty components contribute as currently indicated by their corresponding control variables.

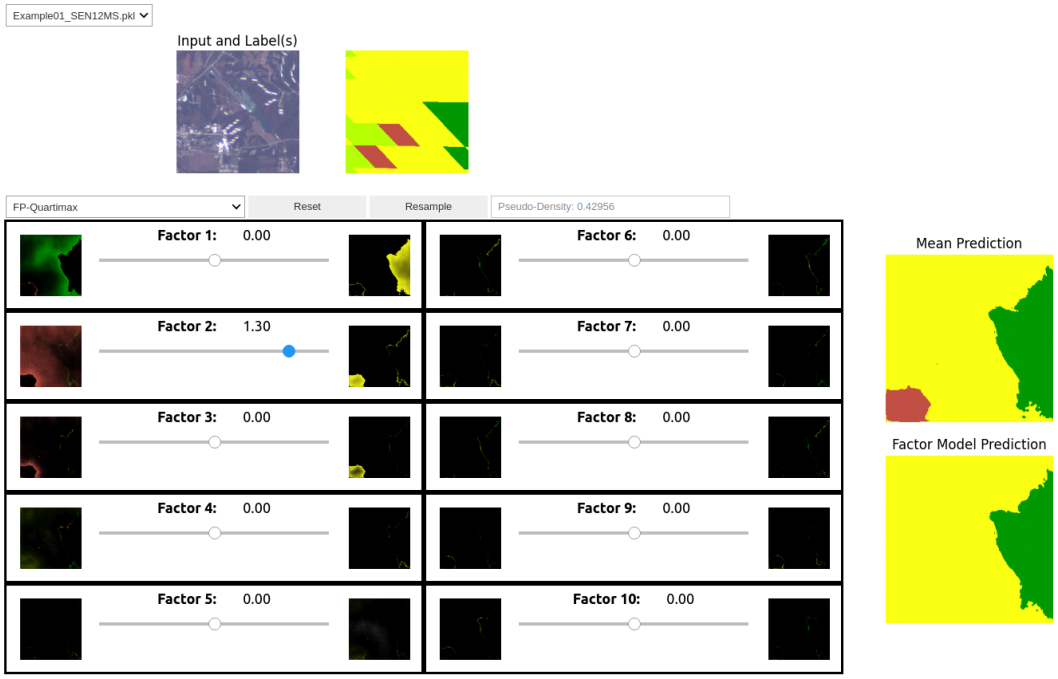


Figure 21: View of the interface for fine-grained sampling for a SEN12MS example.

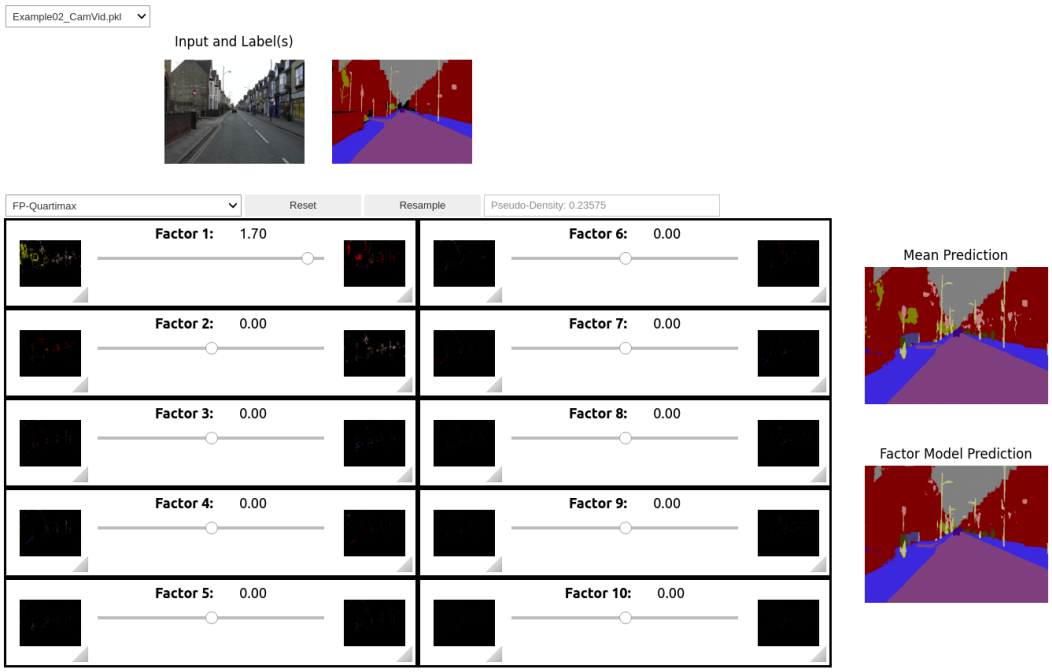


Figure 22: View of the interface for fine-grained sampling for a CamVid example.

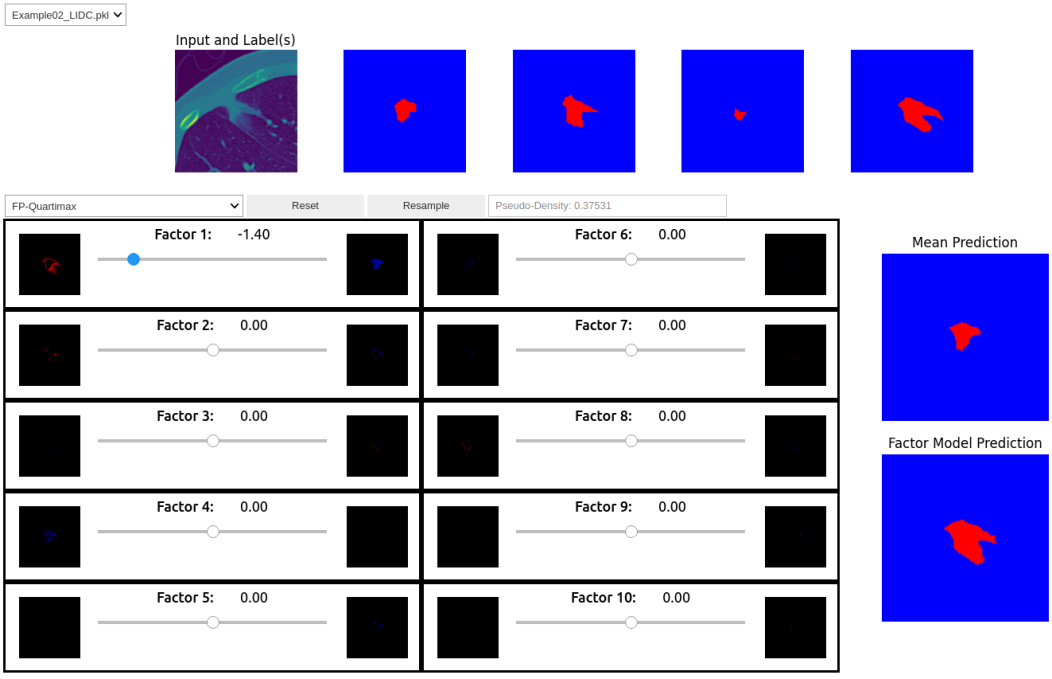


Figure 23: View of the interface for fine-grained sampling for an LIDC example.