Supplementary Material

Random Sum-Product Networks: A Simple but Effective Approach to Probabilistic Deep Learning

Robert Peharz[1], **Antonio Vergari**[2], **Karl Stelzner**[3], **Alejandro Molina**[3], **Xiaoting Shao**[3], **Martin Trapp**[4]
Kristian Kersting[3], **Zoubin Ghahramani**[1,5]
University of Cambridge[1], MPI for Intelligent Systems[2], TU Darmstadt[3], TU Graz[4], Uber AI Labs[5]
{rp587,zoubin}@cam.ac.uk, antonio.vergari@tue.mpg.de,
{stelzner, molina,xiaoting.shao, kersting}@cs.tu-darmstadt.de, martin.trapp@tugraz.at

1 Building RAT-SPNs

Algorithm 1 Construct SPN from Region Graph

procedure CONSTRUCTSPN($\mathcal{R}, C, S, I$)
  Make empty SPN
  for $\mathbf{R} \in \mathcal{R}$ **do**
    if $\mathbf{R}$ is a leaf region **then**
      Equip $\mathbf{R}$ with I distribution nodes
    else if $\mathbf{R}$ is the root region **then**
      Equip $\mathbf{R}$ with C sum nodes
    else
      Equip $\mathbf{R}$ with S sum nodes
  for $\mathcal{P} = \{\mathbf{R}_1, \mathbf{R}_2\} \in \mathcal{R}$ **do**
    Let $\mathbf{N}_{\mathbf{R}}$ be the nodes for region $\mathbf{R}$
    for $\mathsf{N}_1 \in \mathbf{N}_{\mathbf{R}_1}, \mathsf{N}_2 \in \mathbf{N}_{\mathbf{R}_2}$ **do**
      Introduce product $\mathsf{P} = \mathsf{N}_1 \times \mathsf{N}_2$
      Let P be a child for each $\mathsf{N} \in \mathbf{N}_{\mathbf{R}_1 \cup \mathbf{R}_2}$
  return SPN

Recall from the main paper, that given a set of RVs $\mathbf{X}$, a *region* $\mathbf{R}$ is defined as any non-empty subset of $\mathbf{X}$. Given any region $\mathbf{R}$, a *K-partition* $\mathcal{P}$ of $\mathbf{R}$ is a collection of K non-empty, non-overlapping subsets $\mathbf{R}_1, \dots, \mathbf{R}_K$ of $\mathbf{R}$, whose union is again $\mathbf{R}$, i.e. $\mathcal{P} = \{\mathbf{R}_1, \dots, \mathbf{R}_K\}$, $\forall k\colon \mathbf{R}_k \neq \emptyset$, $\forall k \neq l\colon \mathbf{R}_k \cap \mathbf{R}_l = \emptyset$, $\bigcup_k \mathbf{R}_k = \mathbf{R}$. We consider only 2-partitions, which causes all product nodes in our SPNs to have exactly two children. This assumption, frequently made in the SPN literature, simplifies SPN design and seems not to impair performance.

A *region graph* $\mathcal{R}$ over $\mathbf{X}$ is a connected DAG whose nodes are regions and partitions such that i) there is exactly one region $\mathbf{R} = \mathbf{X}$ without parents (i.e. $\mathbf{X}$ is the *root region*), ii) all leaves of $\mathcal{R}$ are regions, iii) all children of regions are partitions and all children of partitions are regions (i.e. $\mathcal{R}$ is bipartite), iv) if $\mathcal{P}$ is a child of $\mathbf{R}$, then $\bigcup_{\mathbf{R}' \in \mathcal{P}} \mathbf{R}' = \mathbf{R}$ and v) if $\mathbf{R}$ is a child of $\mathcal{P}$, then $\mathbf{R} \in \mathcal{P}$. From this definition it follows that a region graph dictates a hierarchical partitioning of the overall scope $\mathbf{X}$.

Algorithm 2 Random Region Graph

procedure RANDOMREGIONGRAPH($\mathbf{X}, D, R$)
  Create an empty region graph $\mathcal{R}$
  Insert $\mathbf{X}$ in $\mathcal{R}$
  for $r = 1 \dots R$ **do**
    SPLIT($\mathcal{R}, \mathbf{X}, D$)

procedure SPLIT($\mathcal{R}, \mathbf{R}, D$)
  Draw balanced partition $\mathcal{P} = \{\mathbf{R}_1, \mathbf{R}_2\}$ of $\mathbf{R}$
  Insert $\mathbf{R}_1, \mathbf{R}_2$ in $\mathcal{R}$
  Insert $\mathcal{P}$ in $\mathcal{R}$
  if $D > 1$ **then**
    if $|\mathbf{R}_1| > 1$ **then** SPLIT($\mathcal{R}, \mathbf{R}_1, D-1$)
    if $|\mathbf{R}_2| > 1$ **then** SPLIT($\mathcal{R}, \mathbf{R}_2, D-1$)

Now, given a region graph $\mathcal{R}$, we can easily construct a corresponding SPN as follows: Populate each leaf-region with a collection of I input distributions, and all other regions with a collection of sum nodes. For the root region we spend C sum nodes, and for all internal regions we spend S sum nodes. Finally, take all cross-products of nodes which are co-children of a partition, and connect these product as children of all sums in the parent region of this partition. Pseudo-code for this procedure is given in Algorithm 1. An example for an RAT-SPN by applying Algorithm 2 (Random Region Graph), followed by Algorithm 1, is given in Figure 1, for hyper-parameters $R = 2$, $D = 2$, $I = 2$, $S = 2$, $C = 3$.

2 Classification Datasets

We performed classification experiments on

- 'mnist', a well-known digit classification dataset **?**
- 'fashion-mnist', an in-place substitute for 'mnist', with the goal to classify fashion items rather than digits **?**

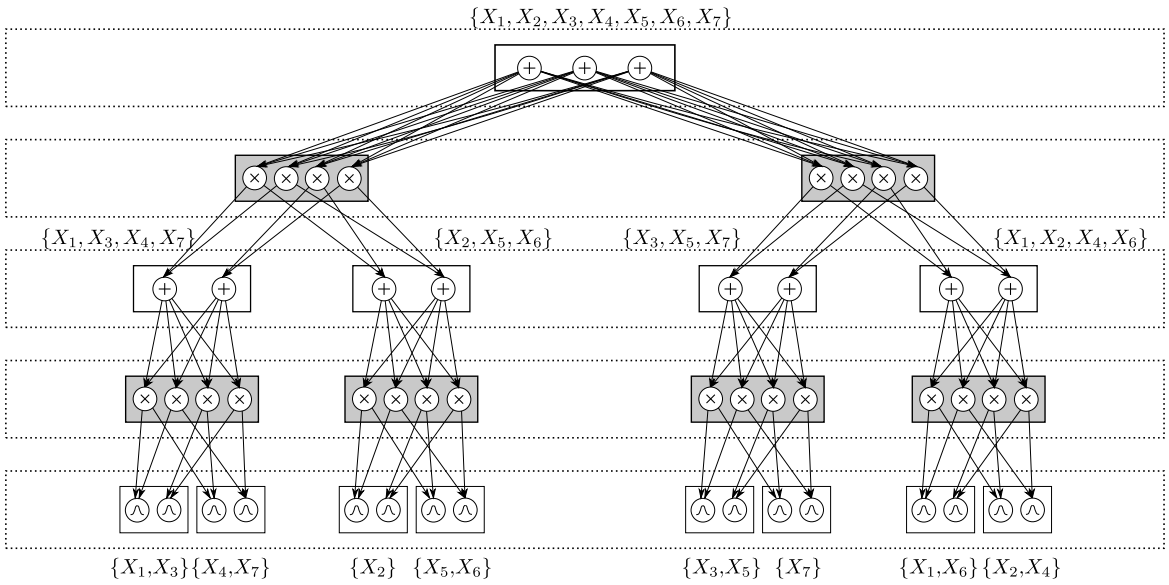

Figure 1: An example RAT-SPN for $R = 2$, $D = 2$, $I = 2$, $S = 2$, $C = 3$.

dataset	domain	#feat.	#train	# val.	# test
mnist	image	784	54k	6k	10k
f-mnist	image	784	54k	6k	10k
imdb	text	200	20k	5k	25k
theorem	logic	51	3670	1224	1224
20ng	text	50	13568	1508	3770
higgs	physics	28	9M	1M	1M
wine	chemistry	11	3899	1299	1299

Table 1: Overview of classification datasets.

- 'imdb', a dataset for binary sentiment analysis of popular movie reviews **?**
- '20ng', a dataset of newsgroup posts belonging to 20 different topics **?**
- 'theorem', a task where a suitable heuristic shall be selected for automatic theorem proofing **?**
- 'higgs', a task to detect whether Higgs boson was generated during a high energy particle collision **?**
- 'wine', a dataset where wine quality shall be predicted from various chemical features **?**

An overview of various characteristics of these datasets is shown in Table 1.

For '(fashion-)mnist', we removed pixels with very low variance (in particular, where the pixel's variance is smaller than 0.001 times the average variance). For 'mnist', we were left with 629 pixels and for 'fashion-mnist' with 775 pixels, out of the originally 784 pixels.

For '20ng', the text was pre-processed into a bag-of-words representation by keeping the top 1000 most relevant words according to their Tf-IDF. Then, 50 topics were extracted by LDA **?** and employed as the new feature representation for classification.

For 'imdb', English stopwords have been removed, and only the 1000 most frequent words have been kept to compute the TF-IDF representation of the training document collection. Over this representation, we extracted 200 topics with Non-Negative Matrix Factorization optimizing the KL-divergence **?** and a L2-regularization term with coefficient $\alpha = 0.1$ via multiplicative updates for 1000 iterations after a random initialization.

For 'theorem', 'higgs' and 'wine' we used the provided raw features. For 'wine' we consolidated the two subsets for red and white wine.

Subsequently, we performed for all datasets zero-mean unit-variance normalization. For all classifiers, we used exactly the same pre-processing.

3 Selection of Hyper-Parameters

RAT-SPNs have 5 structural hyper-parameters, i.e.

- *split-depth* D
- *number of repetitions* R

Table 2: Choice of hyper-parameters R, I, and S for generative learning. For $D = 1$, hyper-parameter S has no effect.

D	R	I	S	W_{S}
	10	10	-	1000
1	25	20	-	10000
	50	45	-	101250
	4	5	5	1100
2	10	8	8	10880
	15	15	15	104625
	3	5	3	1089
3	10	6	5	9950
	16	10	10	97600
	3	3	3	1161
4	6	5	5	10650
	10	10	8	95360

- *number of input distributions* I
- *number of root nodes* C
- *number of sum nodes per inner region* S

See the main paper and the previous section (Algorithm 2 and Algorithm 1) for details concerning these hyper-parameters.

In order to adequately set these hyper-parameters, we might target at a particular number of parameters, i.e. sum-weights and parameters of input distributions. Recall from the main paper that the number of sum-weights is given as

$$W_{\mathsf{S}} = \begin{cases} RCI^2 & \text{if } D = 1 \\ R\left(CS^2 + (2^{D-1} - 2)S^3 + 2^{D-1}SI^2\right) & \text{if } D > 1. \end{cases} \tag{1}$$

Similarly, we can count the number of parameters of input distributions, which we assume to factorize into univariate distributions. Since the exponential growth of number of input regions is exactly compensated by the exponential decrease of scope-size per input region, the total number of parameters for the input distributions is given as

$$W_{\mathsf{D}} = RI|\mathbf{X}|P, \tag{2}$$

where P is the number of parameters per univariate distribution.

3.1 Hyper-Parameters for Generative Learning

For generative learning, we cross-validated the split-depth $D \in \{1, 2, 3, 4\}$ and selected, for each D, settings for R, I, S in order to yield RAT-SPNs approximately with 10^3, 10^4 and 10^5 sum-weights. We set $C = 1$, since we are estimating a single density over $\mathbf{X}$. Our particular choice for R, I, S is depicted in Table 2. This choice was found by trying some combinations of R, I, S in (2), until W_{S} was close to the desired value. We kept the values rather balanced, slightly preferring larger values for R and I, since W_{S} grows quickest with S. However, the particular choice for R, I, S was not tuned to any data – only W_{S} was cross-validated.

3.2 Hyper-Parameters for Discriminative Learning

We cross-validated the number of hidden layers L and number of hidden units H in our trained MLPs as depicted in Table 3. We used varying ranges for L and H, since the employed datasets have rather distinct sizes. In order to ensure a fair comparison, we selected hyper-parameters D, R, I and S in order that the number of parameters in RAT-SPNs match the number of parameters in MLPs, see Table 4. Similar as for the generative case, the particular choice of R, I and S (for each D) was made before running any experiments. Thus, only the number of parameters and split-depth D was cross-validated.

4 Detecting Outliers

Besides being robust against under features, an important feature of (hybrid) generative models is that they are naturally able to detect outliers and peculiarities by monitoring the marginal likelihood over inputs $\mathbf{X}$. To this end, we evaluated the likelihoods on the test set for both 'mnist' and 'fashion-mnist', using the respective RAT-SPN post-trained with $\lambda = 0.2$. For illustrative purposes, we divided the test samples into correctly and incorrectly classified ones. From both groups, we selected two examples for each class, namely the one with the lowest input probability (outlier) and the one with the highest input probability (inlier). This yields 4 groups of 10 samples each: outlier/correct, outlier/incorrect, inlier/correct, inlier/incorrect. These samples are shown in Figure 2.

Furthermore, Tables 5 and 6 show detailed class posteriors for the incorrect samples in 'mnist' and 'fashion-mnist', respectively. For both datasets we see that in the inlier/incorrect group, ambiguity seems to be the major cause for miss-classification. For 'mnist', we see that for the inlier/incorrect group, the correct class gets 8 out of 10 times the second highest probability, while for the outlier/incorrect group this happens only 4 out of 10 times. For 'fashion-mnist', we see that for the inlier/incorrect group, the correct class gets 6 out of 10 times the second

Table 3: Choice of hyper-parameters for MLPs, number of hidden layers L and number of hidden units H, for the employed classification datasets.

dataset	L	H	#params
(f-)mnist	1	$\{100, 250, 500, 1000, 2000\}$	64k, 160k, 320k, 640k, 1.28M
	2	$\{100, 250, 500, 1000, 2000\}$	74k, 221k, 574k, 1.60M, 5.25M
	3	$\{100, 250, 500, 1000, 2000\}$	84k, 286k, 821k, 2.64M, 9.28M
	4	$\{100, 250, 500, 1000, 2000\}$	94k, 348k, 1.07M, 3.64M, 13.29M
imdb	1	$\{100, 250, 500, 1000, 2000\}$	20k, 51k, 102k, 203k, 406k
	2	$\{100, 250, 500, 1000, 2000\}$	30k, 114k, 352k, 1.2M, 4.4M
	3	$\{100, 250, 500, 1000, 2000\}$	41k, 176k, 603k, 2.2M, 8.41M
theorem	1	$\{100, 250, 500, 1000\}$	6k, 15k, 29k, 58k
	2	$\{100, 250, 500, 1000\}$	16k, 77k, 280k, 1.05M
	3	$\{100, 250, 500, 1000\}$	26k, 140k, 530k, 2.06M
20ng	1	$\{100, 250, 500, 1000\}$	7k, 18k, 36k, 71k
	2	$\{100, 250, 500, 1000\}$	17k, 81k, 286k, 1.07M
	3	$\{100, 250, 500, 1000\}$	27k, 143k, 536k, 2.07M
higgs	1	$\{100, 250, 500, 1000\}$	3k, 8k, 16k, 31k
	2	$\{100, 250, 500, 1000\}$	13k, 71k, 266k, 1.03M
	3	$\{100, 250, 500, 1000\}$	23k, 133k, 517k, 2.03M
wine	1	$\{100, 250, 500\}$	1k, 4k, 7k
	2	$\{100, 250, 500\}$	12k, 66k, 258k
	3	$\{100, 250, 500\}$	22k, 129k, 508k

Table 4: Choice of hyper-parameters for RAT-SPNs, split-depth D, number of repetitions R, number of input distributions I and number of sum nodes per inner region S, matched to the number of parameters in Table 3. For $D = 1$, hyper-parameter S has no effect.

datset	D	(R, I, S)	#params
(f-)mnist	1	$\{(9, 10, -), (14, 15, -), (19, 20, -), (29, 25, -), (40, 33, -)\}$	66k, 164k, 315k, 637k, 1.27M
	2	$\{(8, 10, 10), (12, 15, 15), (19, 20, 18), (30, 25, 25), (40, 37, 35)\}$	74k, 221k, 574k, 1.6M, 5.28M
	3	$\{(10, 8, 8), (12, 14, 12), (15, 20, 18), (30, 25, 20), (40, 35, 30)\}$	87k, 277k, 844k, 2.57M, 9.28M
	4	$\{(5, 10, 9), (10, 15, 10), (14, 20, 14), (28, 20, 20), (40, 30, 26)\}$	93k, 344k, 1.06M, 3.6M, 12.73M
imdb	1	$\{(9, 10, -), (15, 15, -), (21, 20, -), (30, 26, -), (40, 36, -)\}$	20k, 52k, 101k, 197k, 392k
	2	$\{(10, 8, 8), (14, 14, 12), (20, 20, 16), (30, 26, 25), (40, 38, 35)\}$	28k, 109k, 346k, 1.21M, 4.45M
	3	$\{(10, 8, 7), (15, 14, 9), (20, 18, 15), (30, 23, 22), (40, 35, 30)\}$	42k, 172k, 605k, 2.2M, 8.39M
theorem	1	$\{(10, 5, -), (13, 7, -), (17, 9, -), (20, 12, -)\}$	4k, 8k, 16k, 30k
	2	$\{(15, 6, 5), (18, 10, 10), (24, 15, 15), (40, 20, 20)\}$	12k, 56k, 213k, 777k
	3	$\{(13, 7, 5), (17, 10, 10), (21, 15, 15), (40, 20, 19)\}$	23k, 121k, 470k, 1.89M
20ng	1	$\{(10, 5, -), (13, 7, -), (17, 9, -), (20, 12, -)\}$	8k, 17k, 35k, 70k
	2	$\{(15, 6, 5), (18, 10, 10), (24, 15, 15), (40, 20, 20)\}$	17k, 81k, 288k, 1M
	3	$\{(13, 7, 5), (17, 10, 10), (21, 15, 15), (40, 20, 19)\}$	27k, 145k, 536k, 2.09M
higgs	1	$\{(10, 7, -), (16, 10, -), (20, 14, -), (23, 20, -)\}$	3k, 8k, 16k, 31k
	2	$\{(10, 10, 5), (15, 14, 10), (20, 20, 15), (40, 25, 20)\}$	13k, 68k, 260k, 1.06M
	3	$\{(9, 10, 5), (16, 12, 10), (24, 15, 15), (40, 20, 20)\}$	23k, 133k, 507k, 1.97M
wine	1	$\{(5, 10, -), (8, 12, -), (13, 14, -)\}$	2k, 3k, 7k
	2	$\{(5, 10, 10), (10, 15, 14), (20, 20, 15)\}$	12k, 69k, 253k
	3	$\{(9, 10, 5), (11, 15, 10), (20, 20, 13)\}$	22k, 125k, 515k

Figure 2: Examples of outliers (lowest input probability in test set) and inliers (highest input probability) for 'mnist' and 'fashion-mnist', for each class. The respective top row shows the outliers, the bottom row the inliers. All samples on the left hand side were classified correctly, all samples on the right hand side were classified incorrectly.

highest probability, while for the outlier/incorrect group this happens only 3 out of 10 times. This is also reflected in the predictive uncertainties, measured as cross-entropy of the class-posterior. In particular, we have for 'mnist':

- outlier/correct: CE = 0.031
- inlier/correct: CE = 0.031
- outlier/incorrect: CE = 0.125
- inlier/incorrect: CE = 0.301

For 'fashion-mnist', we have:

- outlier/correct: CE = 0.023
- inlier/correct: CE = 0.019
- outlier/incorrect: CE = 0.173
- inlier/incorrect: CE = 0.432

Table 5: Class posteriors for incorrect outliers and incorrect inliers on 'mnist'.

	outlier/incorrect											
label	$p(C \mid \mathbf{X})$										CE	LL
0	0.288	0.000	0.000	0.000	0.712	0.000	0.000	0.000	0.000	0.000	0.601885	-1252.031372
1	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000230	-912.313721
2	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000177	-9215.482422
3	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000606	-1609.093994
4	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000182	-12721.799805
5	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000158	-10736.949219
6	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000229	-3916.864502
7	0.000	0.000	0.000	0.971	0.000	0.000	0.000	0.029	0.000	0.000	0.132454	-2431.549316
8	0.000	0.000	0.000	0.000	0.000	0.990	0.000	0.000	0.010	0.000	0.054752	-2403.748535
9	0.000	0.000	0.825	0.000	0.000	0.000	0.000	0.000	0.000	0.175	0.463652	-2462.062256
	inlier/incorrect											
label	$p(C \mid \mathbf{X})$										CE	LL
0	0.252	0.000	0.000	0.000	0.000	0.000	0.748	0.000	0.000	0.000	0.564740	-682.124390
1	0.000	0.132	0.000	0.000	0.013	0.000	0.000	0.852	0.000	0.002	0.473643	-670.625916
2	0.000	0.266	0.000	0.000	0.000	0.000	0.000	0.000	0.734	0.000	0.582508	-675.940247
3	0.000	0.000	0.000	0.486	0.000	0.000	0.000	0.000	0.514	0.000	0.693245	-677.803772
4	0.000	0.000	0.000	0.000	0.006	0.000	0.000	0.000	0.000	0.994	0.035805	-663.004822
5	0.000	0.001	0.002	0.001	0.019	0.034	0.000	0.004	0.000	0.937	0.307368	-715.435852
6	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000124	-669.673096
7	0.000	0.970	0.000	0.000	0.000	0.000	0.000	0.030	0.000	0.000	0.135932	-660.813416
8	0.000	0.000	0.998	0.000	0.000	0.000	0.000	0.000	0.002	0.000	0.014296	-685.410889
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.950	0.000	0.050	0.198429	-658.176208

Table 6: Class posteriors for incorrect outliers and incorrect inliers on 'fashion-mnist'.

	outlier/incorrect											
label	$p(C \mid \mathbf{X})$										CE	LL
0	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.003652	-4592.819336
1	0.001	0.000	0.003	0.000	0.000	0.000	0.996	0.000	0.000	0.000	0.030717	-1112.185669
2	0.001	0.000	0.000	0.000	0.007	0.000	0.992	0.000	0.000	0.000	0.049527	-4694.665527
3	0.014	0.000	0.733	0.241	0.012	0.000	0.001	0.000	0.000	0.000	0.686189	-1166.876465
4	0.001	0.000	0.076	0.000	0.089	0.000	0.834	0.000	0.000	0.000	0.568838	-3858.755859
5	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.001925	-4278.995117
6	0.000	0.000	0.000	0.000	0.000	0.000	0.116	0.000	0.883	0.000	0.366590	-6546.127930
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.002169	-1089.765991
8	0.001	0.000	0.000	0.000	0.000	0.000	0.999	0.000	0.000	0.000	0.009618	-5635.616699
9	0.001	0.000	0.000	0.000	0.000	0.000	0.999	0.000	0.000	0.000	0.009860	-3401.843994
	inlier/incorrect											
label	$p(C \mid \mathbf{X})$										CE	LL
0	0.267	0.000	0.000	0.000	0.000	0.000	0.732	0.000	0.000	0.000	0.585017	-825.450012
1	0.001	0.000	0.000	0.996	0.000	0.000	0.002	0.000	0.000	0.000	0.027871	-850.614075
2	0.000	0.000	0.005	0.000	0.722	0.000	0.273	0.000	0.000	0.000	0.616855	-834.262268
3	0.000	0.000	0.001	0.489	0.496	0.000	0.013	0.000	0.000	0.000	0.763959	-831.182373
4	0.000	0.000	0.921	0.000	0.077	0.000	0.002	0.000	0.000	0.000	0.284489	-818.538147
5	0.000	0.000	0.000	0.000	0.000	0.492	0.000	0.508	0.000	0.000	0.693101	-855.898193
6	0.854	0.000	0.000	0.000	0.000	0.000	0.145	0.000	0.000	0.000	0.420682	-821.650879
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000	0.998	0.015364	-853.953491
8	0.003	0.000	0.001	0.201	0.010	0.000	0.670	0.000	0.115	0.000	0.911972	-872.540466
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000525	-847.545105