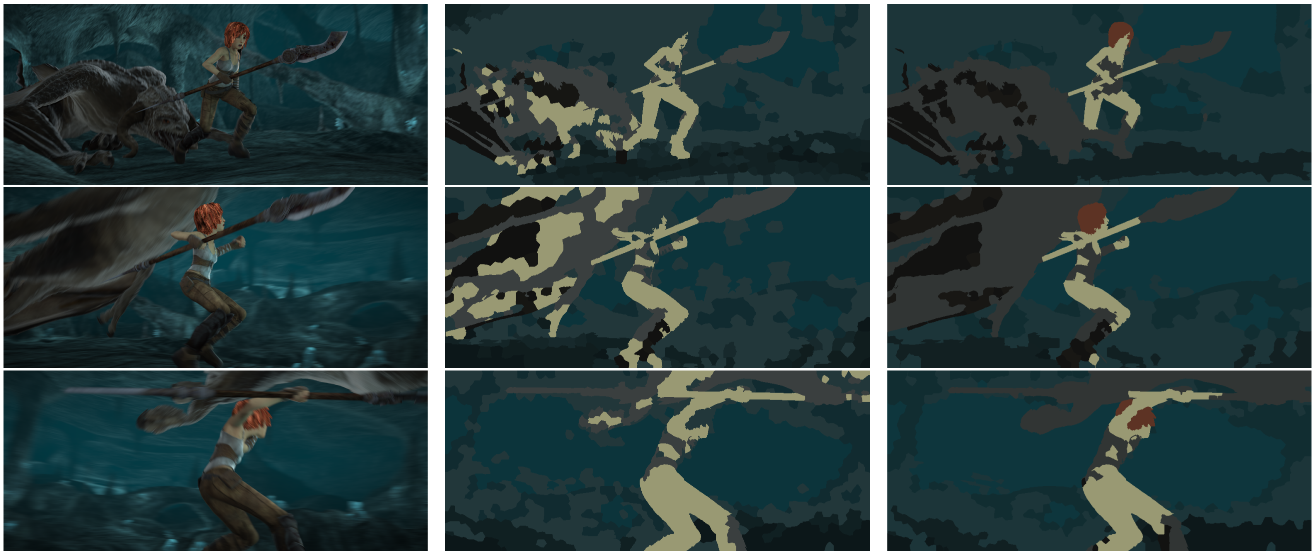
Scalable and Flexible Clustering of Grouped Data via Parallel and Distributed Sampling in Versatile Hierarchical Dirichlet Processes

Supplementary Material

Or Dinari and **Oren Freifeld**
The Department of Computer Science, Ben-Gurion University
dinari@post.bgu.ac.il, orenfr@cs.bgu.ac.il

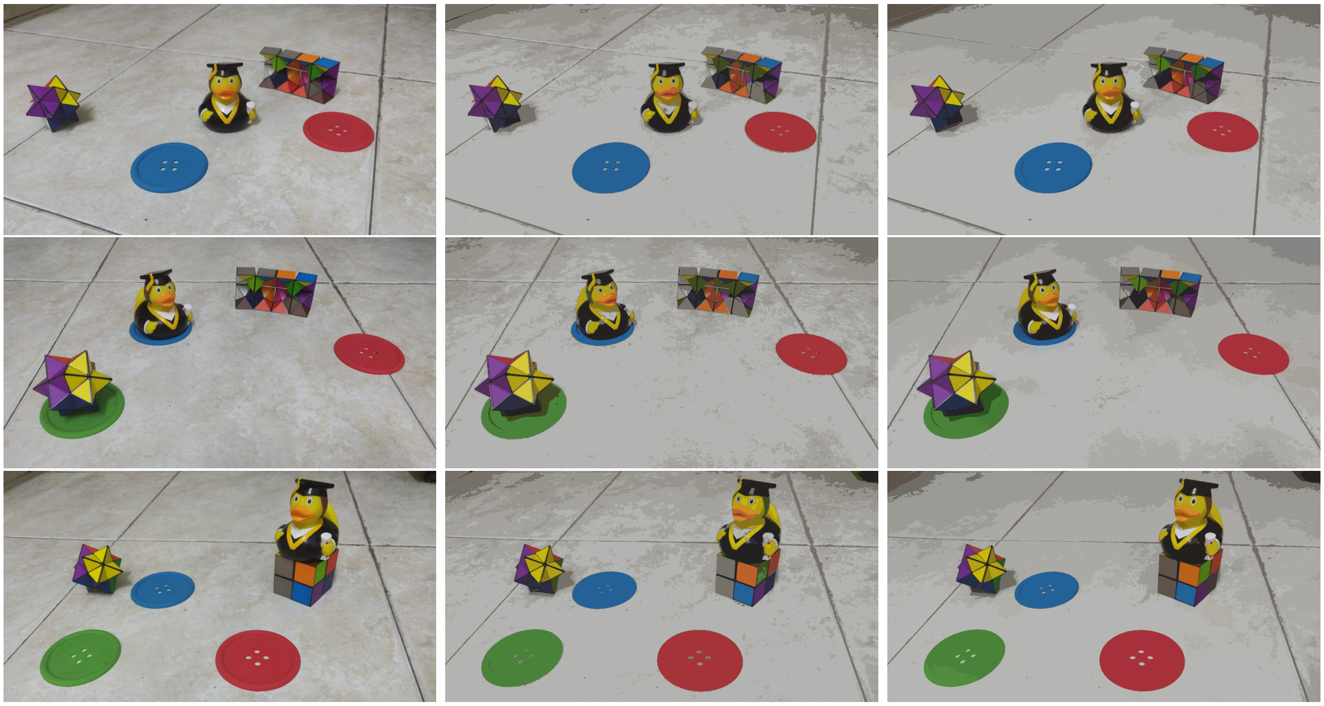
Contents

1 RESULTS FOR THE COSEGEMENTATION EXPERIMENTS



(a) Original (b) HDPMM (c) vHDPMM

Figure 1: Image cosegmentation over superpixels. 3 out of 50 images are shown. (a) Original images [1]. (b) A typical HDPMM result – only global features (*i.e.*, color data from all the images) are used. (c): A typical vHDPMM result – both global (color) and local (location) features are used. In (b) and (c) the colors represent the means of the (global) clusters (we also highlighted, for visualization, the girl's dominant clusters). The local features improve the global clustering, allowing the capturing of small clusters (*i.e.*, her hair) and better color separation.



(a) Original (b) HDPMM (c) vHDPMM

Figure 2: Image cosegmentation over pixels. 3 out of 5 images are shown. Compared with the proposed vHDPMM, the HDPMM misses small details, and has difficulties in color separation, especially for colors that appear infrequently.

2 DETECTING GLOBAL FEATURES: THE THEOREM'S PROOF; INTUITION/VISUALIZATION

We start with a proof of the theorem.

Proof *Let $\boldsymbol{R}'$ be a permutation of $\boldsymbol{R}$ and let $\mathrm{vHDPMM}(H', \gamma, \alpha, L', \eta)$ be the corresponding vHDPMM model, where the first D_g rows of $\boldsymbol{R}'$ are known global features, and H' is of dim D_g. Let s_k denote the data in all the local clusters, across all groups, that are tied to global cluster k;* i.e.,

$$s_k = (s_j^k)_{j=1}^J \tag{1}$$

where s_j^k was defined in Eqn. (4) in the paper. Let C' and $(C_j'^l)_{j=1}^J$ be the optimal global and local clustering, respectively, of $\boldsymbol{R}'$ according to $\mathrm{vHDPMM}(H', \gamma, \alpha, L', \eta)$, where the optimality is defined by

$$p(C', (C_j'^l)_{j=1}^J | \boldsymbol{R}', \mathrm{vHDPMM}(H', \gamma, \alpha, L', \eta)) \geq p(\widetilde{C}, (\widetilde{C}_j^l)_{j=1}^J | \boldsymbol{R}', \mathrm{vHDPMM}(H', \gamma, \alpha, L', \eta)) \tag{2}$$

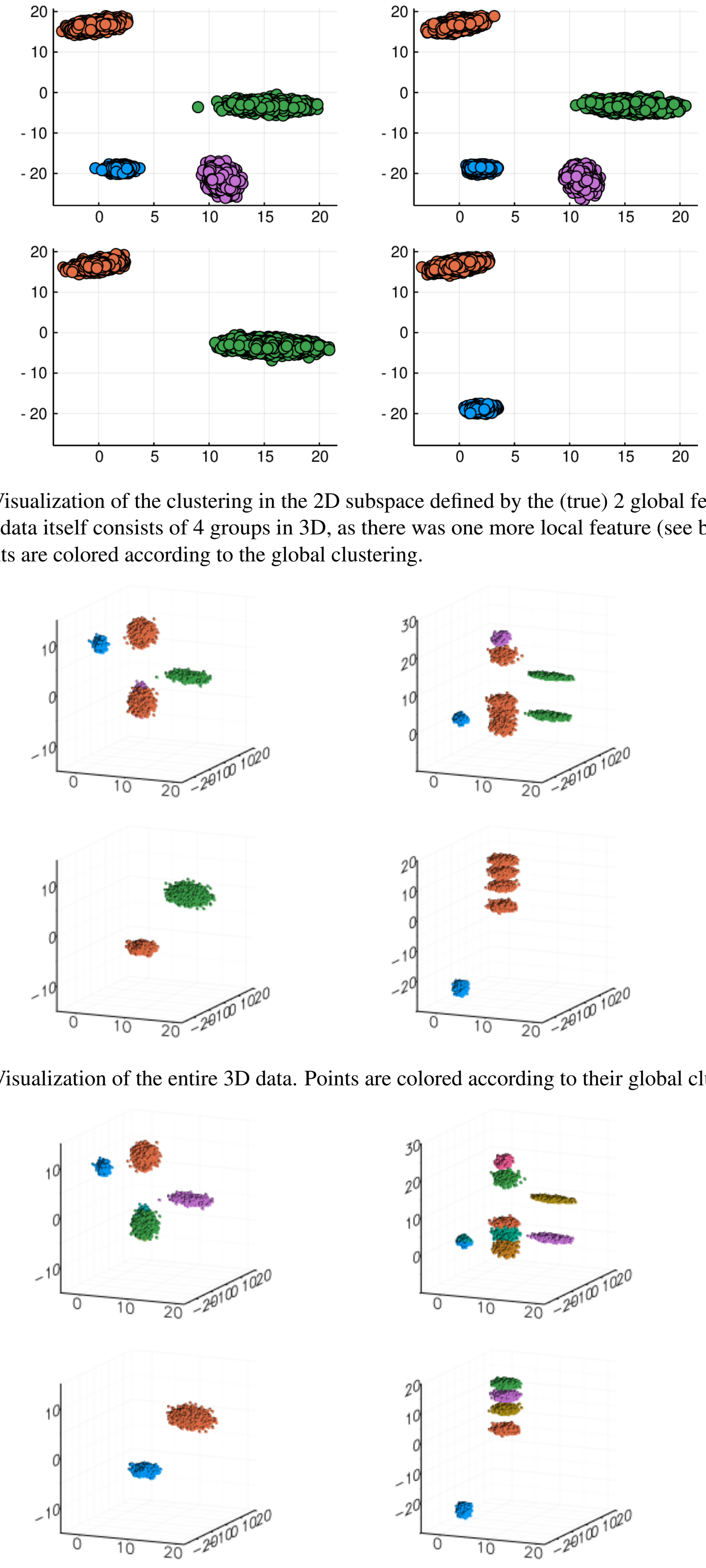
*for every other global clustering, $\widetilde{C}$, and every other local clustering, $(\widetilde{C}_j^l)_{j=1}^J$, of $\boldsymbol{R}'$ according to $\mathrm{vHDPMM}(H', \gamma, \alpha, L', \eta)$. The probability of the global and local clustering given the model takes into account both the likelihood of the data points (how well each point is explained by its associated cluster, at both the local and global parts) and, in addition, the number of different clusters (*e.g.*, while the likelihood is maximized when each point is its own cluster, such clustering is penalized by the too-large number of clusters). In other words, $(C', (C_j'^l)_{j=1}^J)$ is an optimal clustering, taking into account both the likelihood of the points, given the clusters, the likelihood of the cluster parameters given the priors, (H', L'), and likelihood of the number of clusters, given the concentration parameters, (γ, α, η).*
Let $\boldsymbol{R}''$ and $\mathrm{vHDPMM}(H'', \gamma, \alpha, L'', \eta)$ be as defined in the theorem. That is, here the model assumes that there are D_g^{new} global features, where $D_g^{new} = D_g + 1$. The first D_g rows of $\boldsymbol{R}''$ coincide with those of $\boldsymbol{R}'$ and they indeed contain global features. However, the next row is mistakenly viewed as a global feature while, in fact, it contains some local feature. Similarly to before, we denote by C'' and $({C''}^{l}_{j})_{j=1}^J$ the optimal clustering of $\boldsymbol{R}''$ according to $\mathrm{vHDPMM}(H'', \gamma, \alpha, L'', \eta)$. Note the following observations:

- *Moving a feature from local to global (in an optimal setting) does not modify the local clustering of the points. However, when such a move occurs, each global cluster k will no longer well explain the points associated with it, as the local clusters tied to it were distinct from each other. Thus, there exists no single cluster that can well explain the distinct mistakenly-identified-as-global in the different clusters.*

- *In order to accommodate the that move, the global cluster c_k can either 1) split into $(c_g^{k,1}, c_g^{k,2}...c_g^{k,|s_k|})$ different clusters, or 2) drastically increase the variance in f_l in order to fit the distinct points, or 3)combine the previous two solutions.*

Following these observations, note that, in the "best case", where the global part still explains the points as well as before and c_k is split into $|s_k|$ clusters, the data likelihood of $({C''}^{l}_{j})_{j=1}^J$ is similar to $(C_j'^l)_{j=1}^J$. If c_k has been split into fewer clusters than $|s_k|$, there will be at least one local cluster, and its explanation by its associated global cluster will be poorer than before. In addition, the increased amount of global clusters reduces the score, as the previous number of global clusters was, by definition, the best fit (for a given γ). Thus, as the likelihood of C'' and $({C''}^{l}_{j})_{j=1}^J$ is bounded above by that of C' and $(C_j'^l)_{j=1}^J$, it follows that the added local feature reduced the likelihood of the model. While not all local clusters between groups are distinct from each other, in the setting of the proposed model at least some of the local clusters are. The proof holds for any setting where there are at least 2 distinct clusters. As the amount of distinct clusters grows, the differences in the likelihood would only grow larger and larger.

We now provide an illustrative intuition. Figure 3 and Fig. 4 show the inferred model on the same $3D$ data, with $D_g = 2, D_l = 1$. Particularly, Fig. 3 shows the result of choosing the correct features as global, while Fig. 4 shows the result of mistaking one of the local features as global. While in the optimal clustering the 'local' clustering should have remained the same, it is observable that, empirically, converging to the actual optimal local clustering is unlikely when such a mistake has been made.

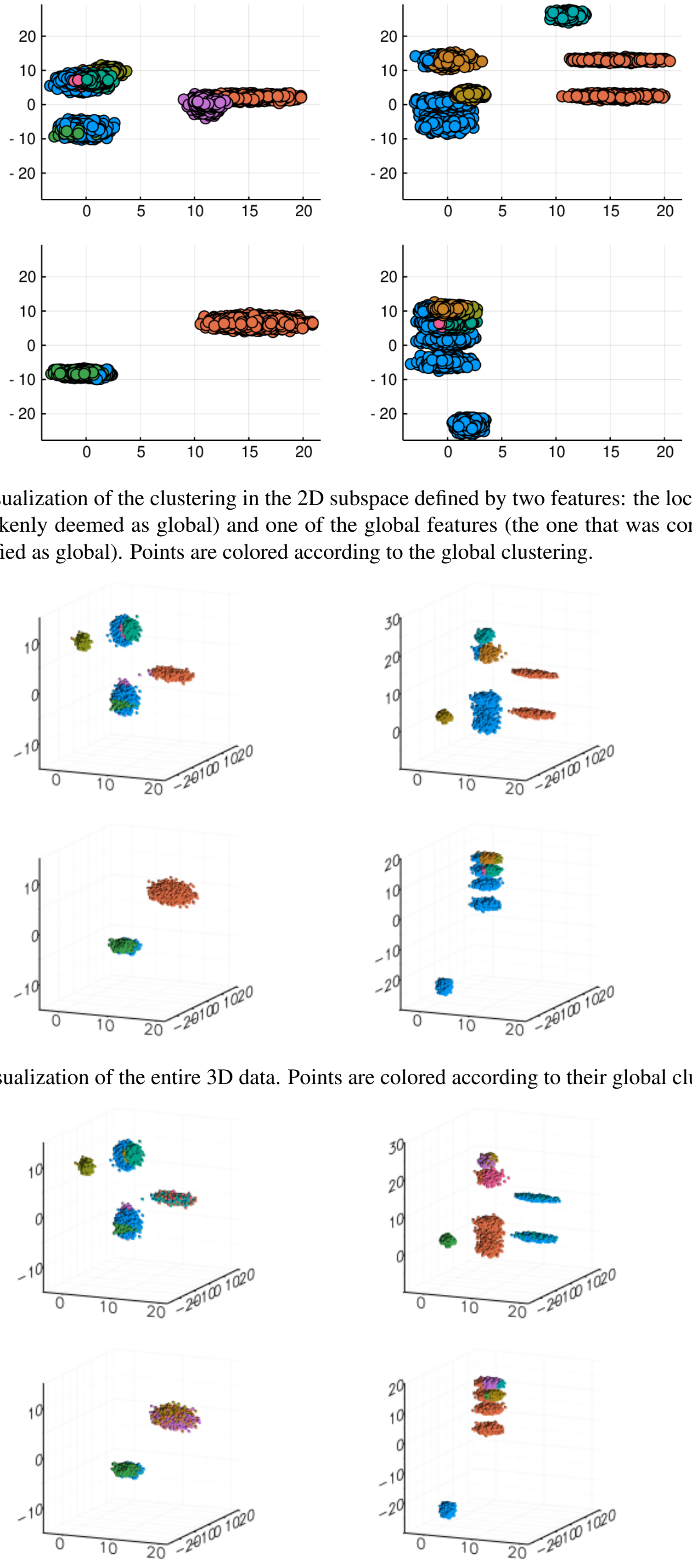


(a) Visualization of the clustering in the 2D subspace defined by the (true) 2 global features. The data itself consists of 4 groups in 3D, as there was one more local feature (see below). Points are colored according to the global clustering.

(b) Visualization of the entire 3D data. Points are colored according to their global clusters.

(c) Visualization of the entire 3D data. Points are colored according to their local clusters.

Figure 3: An example of successful clustering that resulted from correctly identifying (only) the true global features as global.



(a) Visualization of the clustering in the 2D subspace defined by two features: the local one (mistakenly deemed as global) and one of the global features (the one that was correctly identified as global). Points are colored according to the global clustering.

(b) Visualization of the entire 3D data. Points are colored according to their global clusters.

(c) Visualization of the entire 3D data. Points are colored according to their local clusters.

Figure 4: An example of unsuccessful clustering that resulted from mistakenly replacing one of the global features with the local one.

3 VISUALIZATION OF THE AUGMENTED SPACE

The visualization in Fig. 5, for 2D data, provides an intuitive explanation of the augmented space. The figure shows data of a single group (the other groups are not shown). The horizontal axis stands for a global feature while the vertical one stands for a local feature. As can be seen, in that group we see a single global cluster with two local clusters. That is, the two separated 2D point clouds shown in the figure both belong to that global cluster, whose mean is -2.8 (along the horizontal axis). Each of these two clouds, however, is associated with a different local cluster. The means of the two local clusters are at 1.2 and 4.5 (along the vertical axis). Additionally, there are two global sub clusters, with means -2.3 and -3.3 (along the horizontal axis). Likewise, each of the local clusters has two subclusters (along the vertical axis). Thus, there are 8 possible configurations for pairs of a local subcluster label and a global subcluster label. These configurations are represented by the 8 colors of the data in the figure. Among these, 4 are associated with points in the upper local cluster, and the other 4 are associated with the points in the lower subcluster. When a local cluster is split, it is according to the local sub-cluster labels; *e.g.*, in the figure, if the upper local cluster (whose mean is 4.5) is split, two new local clusters will be born. One will consist of the points in orange and purple, and the other will consist of the points in blue and green. When a global split occurs, all the local clusters associated with it will be split according to the global sub-cluster labels; *e.g.*, in the figure, in each local cluster, the points left to -2.8 will form one new local cluster, while the points right to -2.8 will form another new local cluster. Thus, the total number of local clusters will be doubled.

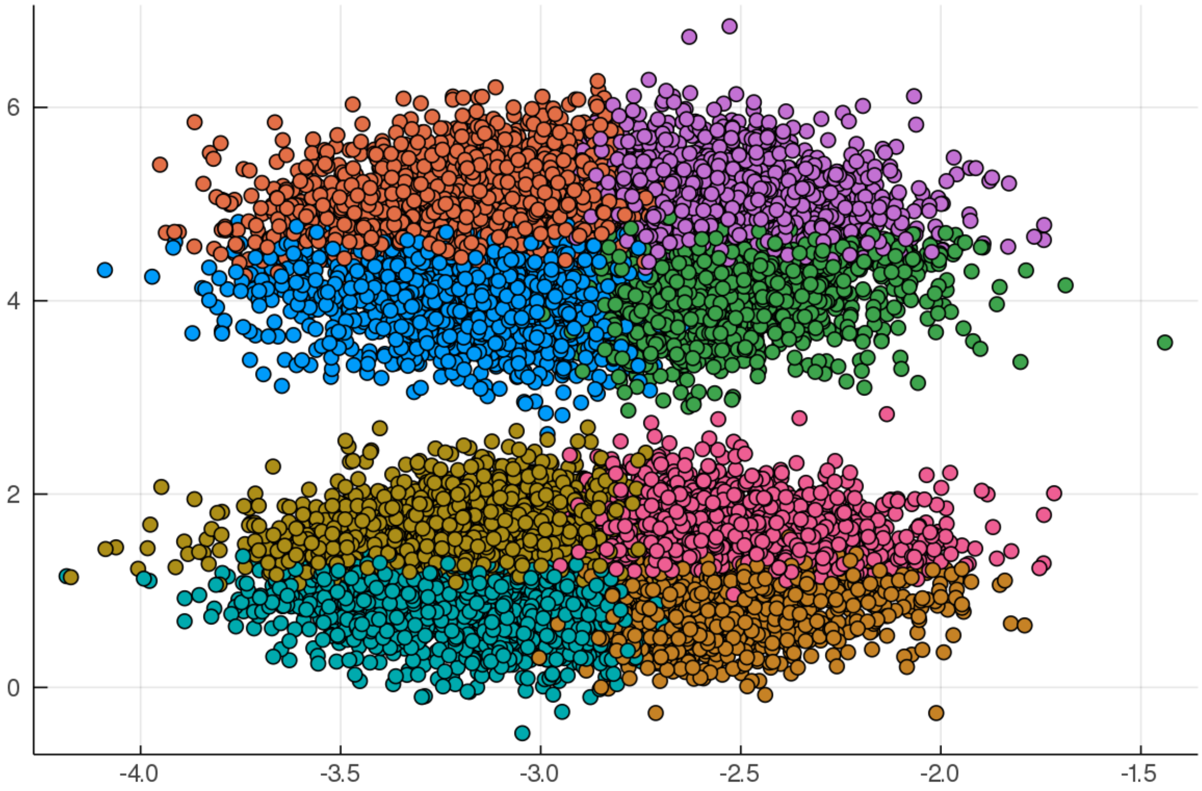


Figure 5: An example, in 2D, of the augmented space. See text for more details.

4 WHERE vHDPMM AND HDPMM DIFFER FROM EACH OTHER EVEN IN THE REDUCED SETTING

In the reduced setting of HDPMM, the two models, vHDPMM and HDPMM, might seem identical at first glance. There are, however, subtle differences. These differences originate from the vHDPMM's full setting, and propagate to the reduced setting as well. For intuition, we will stick with the CRF notation and terminology.

Assume first the full vHDPMM setting. In vHDPMM, two customers at restaurant j cannot sit at two different tables if they have the same pair of dish and side dish. Recall that global dishes are drawn from the discrete distribution, G_j, and that side dishes are drawn from L_j, which can can be either discrete or not. If L_j is non-discrete, then all the drawn pairs will be distinct from each other. If it is discrete, then pairs might not be distinct. In such a case, according to our model we collapse all the tables that share such a pair to a single table. Now, in the reduced setting, because there are no side dishes, a "pair" is just some dish and "no side dish". Because G_j is discrete, repetitions of such "pairs" are likely. Consistently with the full setting, here, in the reduced setting, we still collapse tables sharing a "pair". This implies that, in each of our restaurants in the reduced setting, there is a one-to-one correspondence between tables that are currently occupied and dishes that are currently served. This is different from the HDPMM, where the same dish may appear on several tables at the same restaurant.

In other words, while both vHDPMM and HDPMM use $G_0 \sim DP(\gamma, H)$, the former uses it to model **customers** eating dish k, while the latter uses it to model **tables** serving dish k. Let N_k be the number of customers eating dish k. While both models can be used for inference in the same setting, in the HDPMM the **global** weight of a dish k is proportional to $\log(N_k)$ (that log is an estimate of the number of tables, based on the number of customers), while in vHDPMM the (expected value of the) **global** weight of dish k is proportional to N_k. We note, however, that despite these differences, vHDPMM performs the clustering task well even on data drawn directly from an HDPMM model. See the experiment in the paper where we compared our sampler with an HDPMM CRF-based sampler where the data was drawn from the latter. Despite the success in inferring the labels (*i.e.*, a successful clustering), and because the models are slightly different, we of course do not claim (even in the reduced setting) that the full posterior distributions over the (infinitely-many) parameters of the two models are the same.

5 A CRF-BASED SAMPLER FOR THE vHDPMM

While the vHDPMM CRF-based sampler presented below is of limited practical use (due to shortcomings similar to those of the HDPMM CRF-based sampler), it can help in understanding the more complicated proposed vHDPMM sampler from our paper. To construct a CRF-based sampler for the vHDPMM, we slightly alter the HDPMM CRF-based sampler from [7]. In addition to the notation defined in our paper, we will use additional indicator variables for the CRF model.

Each restaurant is divided to sections, where all the tables at the same section have the same dish (but their side dishes are different from each other). Let $\boldsymbol{s}_j$ denote the collection of sections in restaurant j and let $(s_{j\sigma})_{\sigma=1}^{|\boldsymbol{s}_j|} \in \boldsymbol{s}_j$ denote the sections in that restaurant. Let $\boldsymbol{m}_j^\sigma$ denote the collection of tables in section $s_{j\sigma}$, and let $(m_{jt}^\sigma)_{t=1}^{|\boldsymbol{m}_j^\sigma|} \in \boldsymbol{m}_j^\sigma$ denote the tables in that section. Let $((\theta_{jw}^k)_{k=1}^\infty)_{w=1}^\infty$ denote the side dishes at restaurant j. We now let $t_{ji} \in \{1, \ldots, |\boldsymbol{s}_j|\}$ and $t_{ji}^s \in \{1, \ldots, |\boldsymbol{m}_j^s|\}$ denote the assignments of a customer to a section and a table, respectively. Similarly, $d_{j\sigma}$ and d_{jt}^l are the assignments of a section to a dish and a table to a side dish, respectively. Each table belongs to a single section $s_{j\sigma}$ at any given time, as is implied by the dish served at that table.

We introduce the notation of $n_{jt}^{s^{-i}}$, which denotes the number of customers at table m_{jt}^s, minus the i^{th} customer if that customer is currently sitting at that table. We use ψ_k to denote the number of customers eating dish k, and $\psi_k^{-j\sigma}$ to denote the number of customers eating dish k, minus the number of customers sitting at section $s_{j\sigma}$ if it serves dish k. The sampling scheme is as follows:

$$p(t_{ji}|(n_{jt}^{\sigma^{-i}}, d_{j\sigma}, (d_{j\tau}^l)_{\tau=1}^{|\boldsymbol{m}_j^\sigma|})_{\sigma=1}^{|\boldsymbol{s}_j|}, (\theta^k)_{k=1}^\infty, x_{ji}) \propto \begin{cases} \alpha f(x_{ji}|\theta^{d_{jt_{ji}}}) & \text{if } t_{ji} = s_{j\sigma^{new}} \\ \sum_{\tau=1}^{|\boldsymbol{m}_j^{t_{ji}}|} n_{j\tau}^{t_{ji}^{-i}} f(x_{ji}|\theta^{d_{jt_{ji}}}) & \text{if } t_{ji} = s_{j\sigma} \in \boldsymbol{s}_j \end{cases} \tag{3}$$

We sample $s_{j\sigma^{new}}$ dish using:

$$d_{j\sigma}|(\psi_k^{-j\sigma})_{k=1}^K, \gamma \sim \frac{\psi_k^{-j\sigma}}{\sum_{k'=1}^K \psi_{k'}^{-j\sigma} + \gamma}\theta^k + \frac{\gamma}{\sum_{k'=1}^K \psi_{k'}^{-j\sigma} + \gamma}\theta_{k_{new}} \tag{4}$$

Where $\theta_{k_{new}}$ is a new sample from H. After sampling $t_{ji} = \sigma$, we sample t_{ji}^s in a similar fashion:

$$p(t_{ji}^s|(n_{j\tau}^{\sigma^{-i}}, d_{j\tau}^l)_{\tau=1}^{|\boldsymbol{m}_j^c|}, (\theta_{jw}^k)_{w=1}^\infty)_{k=1}^\infty, y_{ji}, \eta) \propto \begin{cases} \eta f_j(y_{ji}|\theta_{jd_{jt_{ji}^s}^l}^{d_{j\sigma}}) & \text{if } t_{ji}^s = m_{jt^{new}}^s \\ n_{jt}^{\sigma^{-i}} f_j(y_{ji}|\theta_{jd_{jt_{ji}^s}^l}^{d_{j\sigma}}) & \text{if } t_{ji} = m_{jt}^s \in \boldsymbol{m}_j^s \end{cases} \tag{5}$$

Where for $m_{jt^{new}}^s$ we open a new table and use a new sample from L_j as its d_{jt}^s.
After sampling all the points, we can reassign the section and table labels. This is done first for the sections, and then for all the tables. The sampling for the section is done proportionally to the number of all the customers eating each dish (across all restaurants). The sampling for the table is done proportionally to all the customers siting at a table with a dish and a side dish. Note we allow tables to switch between sections.

As mentioned in the main paper, while the above sampler (which is perhaps more intuitive than the parallel sampler we propose in the paper) demonstrates the inference process in a serial and relatively-simple manner, its scales (very) poorly. Since every sample depends on the rest of entire model, these samples cannot be drawn in parallel, and moreover, such a sampler does not make large moves.

6 MORE DETAILS ABOUT THE RESTRICTED GIBBS SAMPLER

Here we provide the complete details regarding the restricted sampler. An iteration of the restricted sampler starts by updating the global weights and global components via Eq. (6),

$$\begin{aligned} p(\theta_k|c_k; H) &\propto f(c_k; \theta_k)p(\theta_k; H) \\ p(\bar{\theta}_k^a|\bar{c}_k^a; H) &\propto f(\bar{c}_k^a; \bar{\theta}_k^a)p(\bar{\theta}_k^a; H) \quad a \in \{1, 2\} \\ p((\beta_k)_{k=1}^{K+1}|(c_k)_{k=1}^K; \gamma) &= \mathrm{Dir}(|c_1|, |c_2|, \dots, |c_K|, \gamma) \\ p(\bar{\beta}_k|\bar{c}_k^1, \bar{c}_k^2; \gamma) &= \mathrm{Dir}(\tfrac{\gamma}{2} + |\bar{c}_k^1|, \tfrac{\gamma}{2} + |\bar{c}_k^2|)\,, \end{aligned} \tag{6}$$

and updating local weights and local components via Eq. (7),

$$\begin{aligned} p(\boldsymbol{\pi}_j|\boldsymbol{\beta}, (s_j^k)_{k=1}^K; \alpha) &= \mathrm{Dir}(\alpha\beta_1 + |s_j^1|, \dots, \alpha\beta_K + |s_j^K|, \alpha\beta_{K+1}) \\ p(\theta_{jw}^k|c_{jw}^k; L_j) &\propto f_j(c_{jw}^k; \theta_{jw}^k)p(\theta_{jw}^k; L_j) \\ p(\bar{\theta}_{jw}^{k,b}|\bar{c}_{jw}^{k,b}; L_j) &\propto f_j(\bar{c}_{jw}^{k,b}; \bar{\theta}_{jw}^{k,b})p(\bar{\theta}_{jw}^{k,b}; L_j) \quad b \in \{1, 2\} \\ p(\boldsymbol{\pi}_j^k|(c_{jw}^k)_{w=1}^{K_j^k}; \eta) &= \mathrm{Dir}(|c_{j1}^k|, |c_{j2}^k|, \dots, |c_{jK_j^k}^k|, \eta) \\ p(\bar{\pi}_{jw}^k|\bar{c}_{jw}^{k,1}, \bar{c}_{jw}^{k,2}; \eta) &= \mathrm{Dir}(\tfrac{\eta}{2} + |\bar{c}_{jw}^{k,1}|, \tfrac{\eta}{2} + |\bar{c}_{jw}^{k,2}|)\,. \end{aligned} \tag{7}$$

The restricted sampler also uses Eqs. (8)-(9) for drawing labels:

$$p(z_{ji}, z_{ji}^l|x_{ji}, y_{ji}, \boldsymbol{\pi}_j, (\boldsymbol{\pi}_j^k, \theta_k, (\theta_{jw}^k)_{w=1}^{K_j^k})_{k=1}^K) \propto \pi_{jk}\pi_{jw}^k f(x_{ji}; \theta_k)f_j(y_{ji}; \theta_{jw}^k) \tag{8}$$

$$p(\bar{z}_{ji}, \bar{z}_{ji}^l|x_{ji}, z_{ji} = k, z_{ji}^l = w, \bar{\beta}_k, \bar{\pi}_{jw}^k, \bar{\theta}_k, \bar{\theta}_{jw}^k) \propto \bar{\beta}_k^a\bar{\pi}_{jw}^{k,b} f(x_{ji}; \bar{\theta}_k^a)f_j(y_{ji}; \bar{\theta}_{jw}^{k,b}) \quad a, b \in \{1, 2\}\,. \tag{9}$$

7 COMPLETING THE DETAILS FOR SPLITS AND MERGES

Local Splits/Merges: Our local splits/merges are similar to those in Chang and Fisher's (non-hierarchical) DPMM sampler [2]. For the derivation of the Hastings ratio below, see [2]. The Hastings ratio for a local split is

$$H_1^{\mathrm{S}} = \frac{\eta\prod_{a\in\{m,n\}}\Gamma(|\hat{c}_{ja}^k|)f_j(\hat{c}_{ja}^k; L_j)}{\Gamma(|c_{jc}^k|)f_j(c_{jc}^k; L_j)}\,. \tag{10}$$

The Hastings ratio for a local merge is

$$H_{\mathrm{l}}^{\mathrm{M}} = \frac{\Gamma(|\hat{c}_{jc}^{k}|) f_j(\hat{c}_{jc}^{k}; L_j)}{\eta \prod_{a\in\{m,n\}} \Gamma(|c_{ja}^{k}|) f_j(c_{ja}^{k}; L_j)} \,. \tag{11}$$

Global Splits/Merges: The construction of the global splits/merges proposals is similar, but not identical, to the one used in Chang and Fisher HDPMM sampler [3] (also note the different terminologies due to the different contexts: what in [3] is called a 'local split' is what we call a 'global split'). For full derivations of the Hasting ratios and the conditional distributions we refer the reader to [3]. The main difference is that we take into account the number of observations assigned to a component, and not the number of local 'tables' assigned to it. Additionally, in our case each component can be instantiated multiple times in a group, each with its own local part, so we must take into account those instantiations. The following equations describe the creation of the proposals for splits and merges according to the current state of the model:

$$(\hat{c}_m, \hat{c}_n) = \mathrm{split}(c_c, c_{jc}^1, c_c^2) \qquad \hat{c}_c = \mathrm{merge}(c_m, c_n) \tag{12}$$

$$(\widehat{\beta}_m, \widehat{\beta}_n) = \beta_c \cdot \bar{\beta}_c \qquad \widehat{\beta}_c = \beta_m + \beta_n \tag{13}$$

$$(\hat{\theta}_m, \hat{\theta}_n) \sim q(\hat{\theta}_m, \hat{\theta}_n | \hat{c}_m, \hat{c}_n) \qquad \hat{\theta}_c \sim q(\hat{\theta}_c | \hat{c}_c) \tag{14}$$

The split function (Eq. (12), left), splits c_c into clusters $\hat{c}_m, \hat{c}_n$. To that aim it looks at the respective auxiliary labels. For each group $j \in \{1, \ldots, J\}$, and for each cluster $c_{jw}^c \in s_j^c$, we split c_{jw}^c into $\hat{c}_{jw}^m$ and $\hat{c}_{jw}^n$, the process being similar to the local split. For each $x_{ji} \in c_{jw}^c$ we examine the respective auxiliary variables. We assign x_{ji} to $\hat{c}_{jw}^m$ if $x_{ji} \in c_c^1$, and to $\hat{c}_{jw}^n$ otherwise. The procedure above splits all local clusters in all groups assigned to global component c into two clusters. The merge function (Eq. (12), right) merges clusters c_m and c_n into cluster $\hat{c}_c$. Unlike the global split, which modifies the number of clusters in all the local groups, the merge only changes the ties between the local clusters and their respective global component. We modify the global labels of all the data points (across all groups) that belong to each of the merged global clusters (m and n) from their previous values to c. This procedure ties all the local clusters previously tied to c_m and c_n to $\hat{c}_n$. Thus we only take into account the global clusters count. This makes the derivation similar to the local version, leading to the following results:

$$H_{\mathrm{g}}^{\mathrm{S}} = \frac{\gamma \prod_{a\in\{m,n\}} \Gamma(|\hat{c}_a|) f(\hat{c}_a; H)}{\Gamma(|c_c|) f(c_c; H)} \frac{{\pi_c}^{|\hat{c}_m|+|\hat{c}_n|}}{{\hat{\pi}_m}^{|\hat{c}_m|} {\hat{\pi}_n}^{|\hat{c}_n|}} \times \prod_{j=1}^{J} \prod_{w=1}^{K_j^c} \frac{\Gamma(\alpha\pi_c)}{\Gamma(\alpha\pi_c + |c_{jw}^c|)} \prod_{a=m,n} \frac{\Gamma(\alpha\hat{\pi}_a + |\hat{c}_{jw}^a|)}{\Gamma(\alpha\hat{\pi}_a)} \tag{15}$$

(the Hastings ratio for a global split) and

$$H_{\mathrm{g}}^{\mathrm{M}} = \frac{\Gamma(|\hat{c}_c|) f(\hat{c}_c; H)}{\gamma \prod\limits_{a\in\{m,n\}} \Gamma(|c_a|) f(c_a; H)} \frac{{\hat{\pi}_c}^{|c_m|+|c_n|}}{{\pi_m}^{|c_m|} {\pi_n}^{|c_n|}} \tag{16}$$

(the Hastings ratio for a global merge). Note that, in the global case, the ratios for the split and merge are not reciprocals of each other.

8 DISTRIBUTED PARALLEL SAMPLING IN THE vHDPMM AND COMPLEXITY ANALYSIS

Switching the lingo to a technical one, in the context of hyper computing, we note the following definitions (some of which reuse words that previously had different meanings in a different context):

Process A computer process (as opposed to a stochastic process such as the Dirichlet Process, the Hierarchical Dirichlet Process, *etc*.).

Node A machine capable of running computing processes.

Cluster A group of nodes working together (not to be confused with the output of the unsupervised learning task of clustering).

Master Node A node running the main process.

Slave Node A non-master node.

Node Leader A process which distributes the work across other processes in the node.

To use the parallel sampler presented in a distributed fashion, one needs to exploit sufficient statistics. We assume that (in each component of the mixture) the likelihood belongs to an exponential family and that the prior is conjugate to the likelihood. Among other things, this lets us aggregate the sufficient statistics calculated in parts. Particularly, this also allows us to distribute the model. We work in a similar fashion to Dinari *et al*. [4], distributing the data across all the nodes and processes. Sampling and calculating the sufficient statistics in parallel, and deciding on splits/merges in the 'node leaders' and master processes. However, while [4] handled a (non-hierarchical) DPMM, here we face a different challenge, as we would like to minimize the inter-machine communication, and redundant duplications of the data. Hence, we distribute a single group with only one node, not splitting its data between several nodes. This allows the 'Group step' to be completely contained in one node, including the split/merge decisions. The only action that is unique to the master process is deciding on global splits/merges, and only the master node processes sample the global parameters and weights.

For runtime assessment we assume that the model has $J = \#$groups, $D = \#$dim(data), $N = \frac{1}{J}\sum_j n_j$, $M = \#$machines, and $P = \#$processes per machine. We now describe a single iteration of the sampler. Numbers on the left refer to the equivalent line in Algorithm 1 from the paper:

Initialization: Distribute the groups across the different nodes, where each node is assigned J/M groups. Distribute the data of each group across the nodes processes, each process gets $(N/J)/P$ points of data, labels includes.

1 Global parameters and weight sampling, parallelized across the master node's processes. Runtime completely is $O(1)$ per cluster, thus $O(D\log N/P)$ in total.

3 Sample the local parameters and weights. This is done independently for each group, while also using within-node parallelism. Runtime of $O(DJ\log N/(MP))$.

5 Sample the point labels. The points are distributed across all nodes and processes. Runtime complexity is $O(DN/(MP))$.

6 Calculate the sufficient statistics. Each process calculates for its own chunk of the data. Local component data is aggregated in the node leader process, while global component data is transmitted to the master node and aggregated there for all groups. Runtime complexity is $O(DN/(MP))$.

8-11 Propose local splits/merges and accept/reject. Runtime complexity is $O((N + J\log N/J)/(MP))$ + $O((N + J\log^2 N/J)/(MP))$ in the 'worst' case where all the splits/merges have been proposed and accepted.

12 Propose global splits/merges and accept/reject. Runtime of $O(N/(MP) + \log N)$+$O(N/(MP) + \log^2 N)$ in the 'worse' case where all the splits/merges have been proposed and accepted.

As often $N \gg M, P, J$, the overall final runtime complexity is $O((J \cdot D \cdot N \cdot \log N)/(MP))$ for a single iteration. We refer the reader to Fig. 6 and Fig. 7 for an overview of the model.

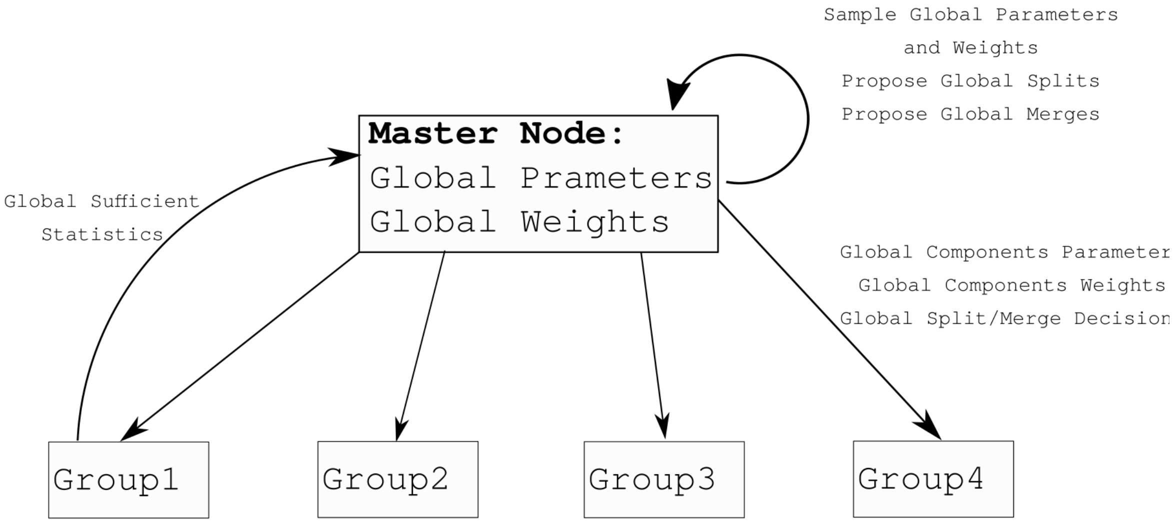


Figure 6: Cluster architecture of a cluster with 4 groups, 4 nodes with 4 processes each, and a master node with a master process.

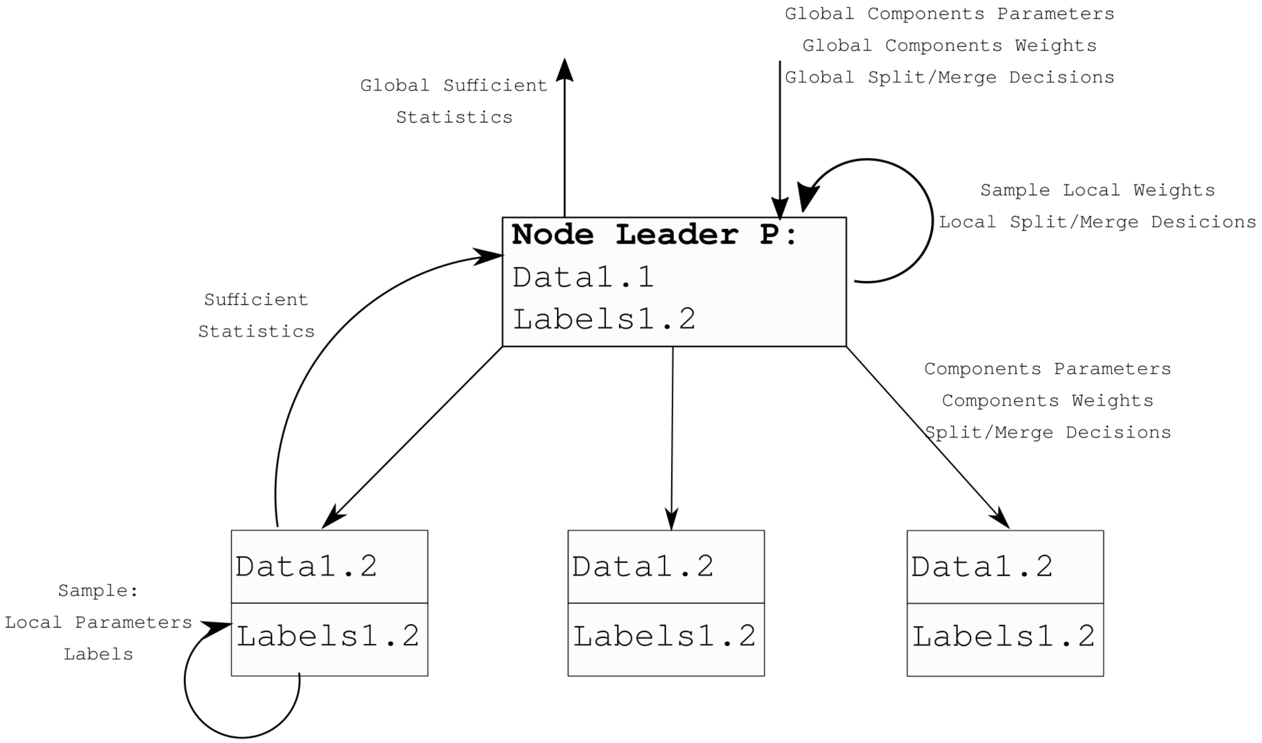


Figure 7: Node architecture of a slave node with 4 processes.

Table 1: NMI Scores on synthetic data (10 groups, 20K points per group, 10 runs per model)

	$\mathcal{G}3/2/5/1$		$\mathcal{G}5/2/10/1$		$\mathcal{G}3/5/5/5$		$\mathcal{G}5/5/10/5$		$\mathcal{M}10/5/20/5$		$\mathcal{M}10/200/20/200$	
Method	*G*	*L*	*G*	*L*	*G*	*L*	*G*	*L*	*G*	*L*	*G*	*L*
HDPMM-all	0.705	-	0.788	-	0.537	-	0.632	-	0.720	-	0.853	-
HDPMM-global-only	0.769	-	0.867	-	0.922	-	0.888	-	0.925	-	0.972	-
DPMM-merged	-	0.076	-	0.105	-	0.261	-	0.193	-	0.255	-	0.232
DPMM-separated	-	0.667	-	0.705	-	0.927	-	0.903	-	0.917	-	0.897
vHDPMM (ours)	**0.862**	0.746	**0.928**	0.756	**0.989**	0.898	**0.959**	0.872	**0.982**	0.935	**0.985**	0.912
vHDPMM-seperated (ours)	-	**0.751**	-	**0.757**	-	**0.997**	-	**0.972**	-	**0.976**	-	**0.974**

9 EXPERIMENT SETTINGS AND MACHINE SPECS

Our implementation of the vHDPMM parallel sampler was done in `Julia`, for the reasons stated in the text. As the model proposed is novel, no direct comparison of our sampler with other inference methods from the literature could be done. As for the vHDPMM CRF-based sampler (described earlier in this document), we found it to be $100\times$ slower in a simple setting, and much slower in more complex ones.

Computer Infrastructure. All the experiments, except the one described in Table 5 in the main paper (*i.e.*, running time versus the number of machines), were done on a single `Ubuntu 16.04` machine, with `Intel(R) i5-6600 CPU @ 3.30GHz` processor (4 cores) with `16 GB RAM`, using `Julia 1.3.0`. As for the experiment from Table 5 (in the main paper), it was performed on a cluster of 4 `Intel(R) i7-6800K CPU @ 3.40GHz, 6 cores (2 hyperthreads each), 32GB RAM` machines.
Both the CRF experiments, in either the vHDPMM or HDPMM settings were done using software we implemented for this purpose in python. The DPMM experiments were run using the state-of-the-art sampler from [4]. The GMM experiments were run using [6].

Synthetic Data. Whenever we use Gaussian components, we accompanied these with a weak NIW prior: $\text{NIW}(\kappa = 1, \mu = [0] \times D, \nu = D + 3, \mathbf{\Psi} = \boldsymbol{I}_{D\times D})$. Similarly, for multinomial components we use a weak Dirichlet-distribution prior: $\text{Dir}([1] \times D)$. As for the concentration parameters, we used the same $\eta = 100, \alpha = 100, \gamma = 1000$ for all the HDPMM and vHDPMM experiments, $\alpha = 1000$ for the DPMM-merged, and $\alpha = 100$ for the DPMM-separated. These values were found to give the best results. Additional results are in Table 1.

Image cosegmentation. For the cosegmentation experiments with superpixels, in both the HDPMM setting and the vHDPMM setting, the sampler was run with the same $\eta = 1000, \alpha = 1000$ (in the vHDPMM setting $\gamma = 10000$ was used). NIW. The NIW base measure for the global features (colors) was

$$\begin{aligned}\text{NIW}(1.0, [47.56, 45.40, 27.82]]/25, 5.0,\\ [[0.86628170.783232820.41225376];\\ [0.783232820.741703840.50340258];\\ [0.412253760.503402580.79185577]] \cdot 0.2)\end{aligned} \tag{17}$$

(these values stand for the sample mean and sample covariance of the colors in the entire dataset) while the NIW base measure for the local features (spatial locations) was

$$\text{NIW}(1.0, [217.857, 511.084]./250, 1000.0, \boldsymbol{I}_{2\times 2} \cdot 0.4) \tag{18}$$

respectively. Superpixels were created using [8]. From each superpixel we took the the color and spatial means of its pixels as our data point, after dividing the color mean by 25 and the spatial mean by 250.
For the experiments with pixels (as opposed to superpixels) we used

$$\text{NIW}(1.0, [64.0, 65.00, 64.82]./25, 50000.0, \boldsymbol{I}_{3\times 3} \cdot 0.2) \tag{19}$$

for the colors, and

$$\text{NIW}(1.0, [370.7, 374.084]./100, 400.0, \boldsymbol{I}_{2\times 2} \cdot 0.30) \tag{20}$$

for the spatial information. **Clustering with Missing Data.** We used $\eta = 100, \alpha = 100, \gamma = 100$ in the vHDPMM experiments, and $\alpha = 100$ in the DPMM experiments. As priors in the DPMM and vHDPMM experiments we used $\mathrm{NIW}(\kappa = 1, \mu = [0] \times D, \nu = D + 3, \boldsymbol{\Psi} = \boldsymbol{I}_{D\times D} \cdot 0.1)$ for both global and local priors.

vHDPMM in the reduced setting. For the synthetic data experiment, we used $\alpha = 10, \gamma = 1$ for the CRF inference, vHDPMM inference. The data was sampled from a CRF prior with $\alpha = 10, \gamma = 1$ for the first 2 experiments, and $\alpha = 10, \gamma = 0.5$ for the third. The base measure for the Gaussian components (for both the vHDPMM and CRF inference) was $\mathrm{NIW}(\kappa = 1, \mu = [0] \times D, \nu = D + 3, \boldsymbol{\Psi} = \boldsymbol{I}_{D\times D})$. For the visual-topic modeling (where the components were multinomials) we used $\alpha = 0.1, \gamma = 0.1$, and a Dirichlet-distribution base measure of $\mathrm{Dir}(ones(D) \cdot 600)$ for both the global and local parts.

Running time versus the number of machines. As data, we used the first 200 frames from the 'Giraffe01' video in [5] test set. The components were Gaussians and the base measures were

$$\mathrm{NIW}(1.0, ones(3) * 0.5, 75.0, \boldsymbol{I} \cdot 0.1) \tag{21}$$

for the RGB color values, and

$$\mathrm{NIW}(1.0, ones(2), 1000.0, \boldsymbol{I} \cdot 30) \tag{22}$$

as for the XY spatial-location values. The concentration parameters were $\alpha = 10000, \gamma = 100$ and $\eta = 10000$.

References

[1] Butler et al. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 2

[2] Chang and F. III. Parallel sampling of DP mixture models using sub-cluster splits. In *NIPS*, 2013. 8

[3] Chang and F. III. Parallel sampling of HDPs using sub-cluster splits. In *NIPS*, 2014. 9

[4] Dinari et al. Distributed MCMC inference in Dirichlet process mixture models using Julia. In *CCGRID HPML Workshop*, 2019. 10, 12

[5] Ochs et al. Segmentation of moving objects by long term video analysis. *IEEE TPAMI*, 2013. 13

[6] Pedregosa et al. Scikit-learn: Machine learning in Python. *JMLR*, 2011. 12

[7] Teh et al. Sharing clusters among related groups: Hierarchical Dirichlet processes. In *NIPS*, 2005. 7

[8] Uziel et al. Bayesian adaptive superpixel segmentation. In *ICCV*, 2019. 12