to be a good choice for our experiments using Gaussian likelihoods with dimension $D = 25$ to $D = 50$. For small values like $D = 2$, N' in the low hundreds may be sufficient.

The target dataset $\mathbf{x}'$ contains samples without replacement from the full dataset $\mathbf{x}$ (of size N). For each observed vector $x_n \in \mathbf{x}$, we add it to our subsample $\mathbf{x}'$ if the following test is true:

$$\hat{r}_{nk'} > \tau, \quad \text{with typical value } \tau = 0.1 \tag{23}$$

Here, $\hat{r}_{nk'}$ is interpreted as the posterior responsibility of component k' for data item n. Each observation n has a vector $[\hat{r}_{n1}\ \hat{r}_{n2}\ \cdots\ \hat{r}_{nK}]$ of these responsibilities, where each entry is non-negative and the whole vector sums to one. The value $\hat{r}_{nk'} \in [0, 1]$ will be larger than the threshold τ if the n-th observation is well-explained by component k'.

Intuitively, our simple "threshold" test for adding data to the targeted dataset $\mathbf{x}'$ ensures that the subsample contains data which are significantly explained by component k', while also promoting diversity (since members could also be partially explained by some other component). The threshold of 0.1 strikes a good balance between these competing goals. We did explore a few other values for τ among $\{0.2, 0.5\}$ in preliminary experiments, and found that $\tau = 0.1$ performed slightly better. We stress that this does not need to be fine-tuned for the particular dataset at hand: the same setting was used for all our experiments.

In practice collection is done by visiting each batch in turn, and collecting all relevant data items until the size of $\mathbf{x}'$ exceeds the limit N'. When batch traversal order is randomized at each pass through the data, this has the beneficial effect of randomizing the subsample.

2.2 Creating an expanded model with brand-new components from the targeted dataset

Next, we consider adding new components to our existing model. We first train a fresh DP mixture model with K' brand-new components on $\mathbf{x}'$ via conventional (batch) variational inference, and then later combine these components with the existing K component model.

The process of creating components by a fresh variational analysis is general and elegant. This strategy applies to *any* DP mixture with exponential family likelihoods, re-uses existing code routines needed for the larger learning algorithm, and has a pleasing interpretation as a "divide-and-conquer" strategy. That is, to find the ideal clustering for the large dataset $\mathbf{x}$, we simply need to repeatedly find some broadly related subset $\mathbf{x}'$ and perform a more fine-grained clustering of that subset.

Creation of new components. Given the target dataset $\mathbf{x}'$ as a stand-alone dataset for analysis, we perform one run of standard full-dataset variational inference. We fit a K'-component DP mixture model with exactly the same prior parameters as the original model.

In practice, we initialize by setting fixed-truncation $K' = 10$, which is a reasonable compromise between diversity and speed. To initialize, we select K' observations (uniformly at random) from $\mathbf{x}'$ to seed parameters. We run only for a fixed budget of $I' = 100$ iterations or until convergence of the objective, whichever happens first.

The choices of truncation level K', initialization routine, and number of iterations I' may all impact the performance of the birth move. We found the same settings lead to reasonable performance across all tested datasets. In general, a more intelligent initialization is better. Running for longer will produce more refined components, but at the cost of increased run-time.

After the run, instead of saving estimated parameters we save *summaries* for each new component:

$$\hat{N}' = [\hat{N}_1\ \hat{N}_2\ \cdots\ \hat{N}_{K'}] \tag{24}$$

$$s(\mathbf{x}') = [s_1(\mathbf{x}')\ s_2(\mathbf{x}')\ \cdots\ s_{K'}(\mathbf{x}')] \tag{25}$$

In general, some final components may have very few assignments to data $\mathbf{x}'$. Some may be empty or nearly-empty. We thus post-process results to remove components j which have low expected counts $\hat{N}_j$ for explaining the data $\mathbf{x}'$. Pruning out empty components makes later phases much faster without sacrificing quality.

Specifically, we remove component j if $\hat{N}_j < \epsilon N'$, and we set $\epsilon = \frac{1}{20}$. After this removal, we end up with a set of J' sufficient statistics $\{\hat{N}_j, s_j(\mathbf{x}')\}_{j=1}^{J'}$, where $J' \leq K'$. These sufficient statistics are all we pass along to the next step.