A Proof of Main Theorem

Let $\{f_{(I,J)} : X \to [0, \infty)\}_{(I,J)\in\mathcal{I}}$ be an indexed family of non-negative measurable functions on X, and let $f_k := f_{(\{k\},\emptyset)}$ for $k \in N$. We will consider two following properties:

(a) $\forall (I, J) \in \mathcal{I}, f_{(I,J)} = \sum_{S:I\subseteq S, J\subseteq N\setminus S} f_{(S,N\setminus S)}$

(b) $\forall i, j \in N$ s.t. $i \neq j, \operatorname{supp} f_{(\{i\},\{j\})} \cap \operatorname{supp} f_{(\{j\},\{i\})} = \emptyset$

We first derive some useful lemmas to prove the necessity of Theorem 1.

Lemma 1. *Assume that (a) and (b) hold.* $\forall (I_1, J_1), (I_2, J_2) \in \mathcal{I}$ *s.t.* $I_1 \cap J_2 \neq \emptyset, J_1 \cap I_2 \neq \emptyset$, $\operatorname{supp} f_{(I_1,J_1)} \cap \operatorname{supp} f_{(I_2,J_2)} = \emptyset$.

Proof. Choose any $u \in I_1 \cap J_2, v \in J_1 \cap I_2$. By (a),

$$\begin{aligned} f_{(\{u\},\{v\})} &= \sum_{S:u\in S, v\in N\setminus S} f_{(S,N\setminus S)} \geq \sum_{S:I_1\subseteq S, J_1\subseteq N\setminus S} f_{(S,N\setminus S)} = f_{(I_1,J_1)} \\ f_{(\{v\},\{u\})} &= \sum_{S:v\in S, u\in N\setminus S} f_{(S,N\setminus S)} \geq \sum_{S:I_2\subseteq S, J_2\subseteq N\setminus S} f_{(S,N\setminus S)} = f_{(I_2,J_2)}. \end{aligned} \tag{9}$$

If $u = v$, then $I_1 \cap J_1 \neq \emptyset$ which contradicts the definition of $\mathcal{I}$. Therefore, $u \neq v$. By (b), $\operatorname{supp} f_{(I_1,J_1)} \cap \operatorname{supp} f_{(I_2,J_2)} \subseteq \operatorname{supp} f_{(\{u\},\{v\})} \cap \operatorname{supp} f_{(\{v\},\{u\})} = \emptyset$. ☐

Lemma 2. *Assume that (a) and (b) hold.* $\forall i \in N, \forall I \subseteq N$ *s.t.* $\{i\} \subsetneq I$, $\operatorname{supp} f_{(I\setminus\{i\},\{i\})} \cap \operatorname{supp} \sum_{S:i\in S, I\not\subseteq S\subseteq N} f_{(S,N\setminus S)} = \emptyset$.

Proof. Let $S \subseteq N$ s.t. $i \in S \not\supseteq I$. Then, $\emptyset \neq I \setminus S \subseteq (I \setminus \{i\}) \cap (N \setminus S)$. By Lemma 1, $\operatorname{supp} f_{(I\setminus\{i\},\{i\})} \cap \operatorname{supp} f_{(S,N\setminus S)} = \emptyset$. Therefore, $\operatorname{supp} f_{(I\setminus\{i\},\{i\})} \cap \operatorname{supp} \sum_{S:i\in S, I\not\subseteq S\subseteq N} f_{(S,N\setminus S)} = \emptyset$. ☐

Lemma 3. *Assume that (a) and (b) hold.* $\forall I \subseteq N$ *s.t.* $I \neq \emptyset$, $f_{(I,\emptyset)} = \min_{i\in I} f_i$.

Proof. We will use induction to prove the lemma. Let $P(k)$ be the following statement.

$$P(k) : \forall I \text{ s.t. } 1 \leq |I| = k \leq |N|, \text{then } f_{(I,\emptyset)} = \min_{i\in I} f_i. \tag{10}$$

For the base case $k = 1$, the statement holds by the definition. Assume that the induction hypothesis for $k \leq l < |N|$ holds. Consider $|I| = l + 1$ and choose any $i \in I$. Then,

$$\begin{aligned} \min_{i\in I} f_i &= \min\{f_{(I\setminus\{i\},\emptyset)}, f_{(\{i\},\emptyset)}\} && \text{by the induction hypothesis} \\ &= f_{(I\setminus\{i\},\emptyset)} - \max\{f_{(I\setminus\{i\},\emptyset)} - f_{(\{i\},\emptyset)}, 0\} \\ &= f_{(I\setminus\{i\},\emptyset)} - \max\{f_{(I\setminus\{i\},\{i\})} - \sum_{S:i\in S, I\not\subseteq S\subseteq N} f_{(S,N\setminus S)}, 0\} \\ &= f_{(I\setminus\{i\},\emptyset)} - f_{(I\setminus\{i\},\{i\})} && \text{by Lemma 2} \\ &= f_{(I,\emptyset)} \end{aligned} \tag{11}$$

Therefore, $P(l + 1)$ holds. We conclude that $f_{(I,\emptyset)} = \min_{i\in I} f_i$ for $\emptyset \neq I \subseteq N$. ☐

Theorem 1. *Let* $\{f_{(I,J)} : X \to [0, \infty)\}_{(I,J)\in\mathcal{I}}$ *be an indexed family of non-negative measurable functions on X, and let* $f_k := f_{(\{k\},\emptyset)}$. *Then, the following conditions hold:*

(a) $\forall (I, J) \in \mathcal{I}, f_{(I,J)} = \sum_{S:I\subseteq S, J\subseteq N\setminus S} f_{(S,N\setminus S)}$

(b) $\forall i, j \in N$ *s.t.* $i \neq j, \operatorname{supp} f_{(\{i\},\{j\})} \cap \operatorname{supp} f_{(\{j\},\{i\})} = \emptyset$

if and only if, for every $(I, J) \in \mathcal{I}$,

$$f_{(I,J)} = \begin{cases} (\min_{i\in I} f_i - \max_{j\in J} f_j)^+ & \textit{if } J \neq \emptyset \\ \min_{i\in I} f_i & \textit{otherwise} \end{cases}, \tag{3}$$

where $(\cdot)^+$ *represents the positive part.*

Proof. We first show the necessity of the condition. Assume that (a) and (b) hold. If $J = \emptyset$, then $f_{(I,J)} = \min_{i\in I} f_i$ by Lemma2. Hence, we may assume that $J \neq \emptyset$. Fix $x \in X$, and let $\{a_1, a_2, ..., a_{|J|}\}$ be an arrangement of J so that $f_{a_i}(x) \le f_{a_j}(x)$ for all $i < j$. For every $\emptyset \neq S \subseteq J$, we let $m(S)$ denote the minimum index s such that $a_s \in S$.

Note that

$$\begin{aligned} f_{(I,J)}(x) &= \sum_{S\subseteq J} (-1)^{|S|} f_{(I\cup S,\emptyset)}(x) \quad \text{by Inclusion–exclusion principle} \\ &= \sum_{S\subseteq J} (-1)^{|S|} \min_{i\in I\cup S} f_i(x) \quad \text{by Lemma 3} \end{aligned} \tag{12}$$

We now decompose the last summation into three cases.

(i) $S = \emptyset$

$$(-1)^{|S|} \min_{i\in I\cup S} f_i(x) = \min_{i\in I} f_i(x). \tag{13}$$

(ii) $m(S) < |J|$

$$\begin{aligned} \sum_{S:m(S)<|J|} (-1)^{|S|} \min_{i\in I\cup S} f_i(x) &= \sum_{j<|J|} \sum_{S:m(S)=j} (-1)^{|S|} \min_{i\in I\cup\{a_j\}} f_i(x) \\ &= \sum_{j<|J|} \left(\min_{i\in I\cup\{a_j\}} f_i(x) \right) \left\{ (-1)\cdot 2^{|J|-j-1} + 2^{|J|-j-1} \right\} \\ &= 0. \end{aligned} \tag{14}$$

(iii) $m(S) = |J|$

$$(-1)^{|S|} \min_{i\in I\cup S} f_i(x) = - \min_{i\in I\cup\{a_{|J|}\}} f_i(x). \tag{15}$$

Summing up all of the above terms gives the rest result.

$$\begin{aligned} f_{(I,J)}(x) &= \min_{i\in I} f_i(x) - \min_{i\in I\cup\{a_{|J|}\}} f_i(x) \\ &= \min_{i\in I} f_i(x) - \min\{\min_{i\in I} f_i(x), \max_{j\in J} f_j(x)\} \\ &= \left(\min_{i\in I} f_i(x) - \max_{j\in J} f_j(x) \right)^+ . \end{aligned} \tag{16}$$

To show the sufficiency, assume

$$\forall (I,J) \in \mathcal{I}, f_{(I,J)} = \begin{cases} (\min_{i\in I} f_i - \max_{j\in J} f_j)^+ & \text{if } J \neq \emptyset \\ \min_{i\in I} f_i & \text{otherwise} \end{cases}. \tag{17}$$

Let us assume that $f_{(I,J)} \neq \sum_{S:I\subseteq S, J\subseteq N\setminus S} f_{(S,N\setminus S)}$ for some $(I,J) \in \mathcal{I}$. Choose such I, J so that $|I| + |J|$ is maximum. Note that $I \cup J \subsetneq N$ because $\sum_{S:I\subseteq S,J\subseteq N\setminus S} f_{(S,N\setminus S)}$ is exactly the same expression as $f_{(I,J)}$ for $I \cup J = N$. Hence, we can choose some $k \in N \setminus (I \cup J)$. By the maximality of $|I| + |J|$, the following two equations hold.

$$\begin{aligned} f_{(I,J\cup\{k\})} &= \sum\nolimits_{S:I\subseteq S, J\cup\{k\}\subseteq N\setminus S} f_{(S,N\setminus S)} \\ f_{(I\cup\{k\},J)} &= \sum\nolimits_{S:I\cup\{k\}\subseteq S, J\subseteq N\setminus S} f_{(S,N\setminus S)}. \end{aligned} \tag{18}$$

We use above two equations and consider all possible inequalities among $\min_{i \in I} f_i$, $max_{j \in J} f_j$, and f_k. The following equation always holds regardless of these inequalities.

$$
\begin{aligned}
\sum_{S: I \subseteq S, J \subseteq N \setminus S} f_{(S, N \setminus S)} &= f_{(I, J \cup \{k\})} + f_{(I \cup \{k\}, J)} \\
&= \begin{cases} \left(\min_{i \in I} f_i - \max_{j \in J \cup \{k\}} f_j\right)^+ + \left(\min_{i \in I \cup \{k\}} f_i - \max_{j \in J} f_j\right)^+ & \text{if } J \neq \emptyset \\ (\min_{i \in I} f_i - f_k)^+ + \min_{i \in I \cup \{k\}} f_i & \text{otherwise} \end{cases} \\
&= \begin{cases} \left(\min_{i \in I} f_i - \max_{j \in J} f_j\right)^+ & \text{if } J \neq \emptyset \\ \min_{i \in I} f_i & \text{otherwise} \end{cases} \\
&= f_{(I, J)},
\end{aligned}
\tag{19}
$$

which leads to a contradiction.

Also, for every $i, j \in N$ such that $i \neq j$,

$$
\begin{aligned}
\min(f_{(\{i\},\{j\})}, f_{(\{j\},\{i\})}) &= \min\{(f_i - f_j)^+, (f_j - f_i)^+\} \\
&= (\min\{f_i - f_j, f_j - f_i\})^+ \\
&= 0.
\end{aligned}
\tag{20}
$$

Therefore, (a) and (b) hold. □

B Description for S2M Sampling

B.1 Pseudocode of S2M Sampling

In Section 3.2, we describe how to build S2M sampling upon unconditional GANs and class-conditional GANs. Algorithm 1 illustrates the use of S2M sampling for GANs. This algorithm can be easily modified to the conditional versions by replacing the acceptance probability α (See Equation 7).

Algorithm 1 *S2M Sampling for GANs*

Input: generator G, classifiers D_v^*, D_r^*, intersection index set I, difference index set J, and class prior ratios $\gamma_{1:N}$

Output: filtered sample x

```
Choose any $x \in \operatorname{supp} p_{(I,J)}$.
for $k = 1$ to $K$ do
    Draw $x'$ from $G$.
    Draw $u$ from Uniform(0,1).
    $r_i \leftarrow \gamma_i D_r^*(i|x)$ for every $i \in I \cup J$
    $r'_i \leftarrow \gamma_i D_r^*(i|x')$ for every $i \in I \cup J$
    $\alpha \leftarrow \min\left(1, \frac{\left(\min\{r'_i : i \in I\} - \max\{r'_j : j \in J\} \cup \{0\}\right)^+ (D_v^*(x)^{-1} - 1)}{(\min\{r_i : i \in I\} - \max\{r_j : j \in J\} \cup \{0\})^+ (D_v^*(x')^{-1} - 1)}\right)$
    if $u \leq \alpha$ then
        $x \leftarrow x'$
    end if
end for
```

B.2 Latent Adaptation with Gaussian Mixture Model

In Section 3.2, we describe the latent adaptation technique to improve the sampling efficiency of our proposed S2M sampling method. In this section, we provide an example for the latent adaptation technique using a Gaussian mixture model. The real examples of using latent adaption are shown in Figure 9. After obtaining target latent samples $t_{1:m}$ from S2M sampling, we use those samples to fit a multivariate Gaussian mixture model $\tilde{p}_z(z) = \sum_{i=1}^{M} \phi_i \mathcal{N}(z|\mu_i, \Sigma_i)$. The parameters can be updated using an expectation–maximization algorithm [3] As explained in Section 3.2, we run the MH

---

[3] Christopher M. Bishop. Pattern recognition and machine learning, 5th Edition. 2007.

algorithm where the proposal $x' \sim q(x'|x) = \tilde{p}_G(x')$ is accepted with a probability $\tilde{\alpha}(x', x)$ which is calculated as

$$\tilde{\alpha}(x', x) = \min\left(1, \frac{p_{(I,J)}(x')/\tilde{p}_G(x')}{p_{(I,J)}(x)/\tilde{p}_G(x)}\right) \approx \min\left(1, \frac{r_{(I,J)}(x')(D_v^*(x)^{-1} - 1)p_z(z')/\tilde{p}_z(z')}{r_{(I,J)}(x)(D_v^*(x')^{-1} - 1)p_z(z)/\tilde{p}_z(z)}\right). \tag{21}$$

If the latent prior of the generator G is the standard multivariate normal distribution $p_z(z) = \mathcal{N}(z|0, I_d)$, we can compute the acceptance probability by explicitly calculating the density ratio $p_z(z)/\tilde{p}_z(z)$ as follows:

$$\frac{p_z(z)}{\tilde{p}_z(z)} = \frac{\mathcal{N}(z|\mu_0, \Sigma_0)}{\sum_{i=1}^M \phi_i \mathcal{N}(z|\mu_i, \Sigma_i)} = \left(\sum_{i=1}^M \phi_i |\Sigma_i|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(z-\mu_i)^T {\Sigma_i}^{-1}(z-\mu_i) + \frac{1}{2}z^T z\right)\right)^{-1}. \tag{22}$$

In our experiments, some determinants of the covariance matrices often approach zero, resulting in numerical errors. One way to readily avoid this problem is to let $\tilde{p}_z$ be a Gaussian mixture with a shared covariance matrix Σ, *i.e.*, $\Sigma_1 = \Sigma_2 = ... = \Sigma_M = \Sigma$. Then, we no longer need to compute the determinants of the covariance matrices to calculate the acceptance probability because $p_z(z')\tilde{p}_z(z)/\tilde{p}_z(z')/p_z(z)$ is simplified as follows:

$$\frac{p_z(z')\tilde{p}_z(z)}{\tilde{p}_z(z')p_z(z)} = \frac{\sum_{i=1}^M \phi_i \exp\left(-\frac{1}{2}(z-\mu_i)^T \Sigma^{-1}(z-\mu_i) + \frac{1}{2}z^T z\right)}{\sum_{i=1}^M \phi_i \exp\left(-\frac{1}{2}(z'-\mu_i)^T \Sigma^{-1}(z'-\mu_i) + \frac{1}{2}z'^T z'\right)}. \tag{23}$$

Algorithm 2 illustrates the sampling process using the Gaussian mixture latent with a shared covariance matrix. After collecting the filtered latent samples $z_{1:s}$ through Algorithm 2, we can get target class samples $x_{1:s}$ by taking $x_i = G(z_i)$ for $i = 1, 2, ..., s$. As explained in Section 3.2, the latent adaptation technique can be applied iteratively by increasing the number of specified attributes one by one. Algorithm 3 illustrates this strategy and Section C provides the computation complexity analysis of the algorithm.

Algorithm 2 *Sampling with Adapted Latent Space*

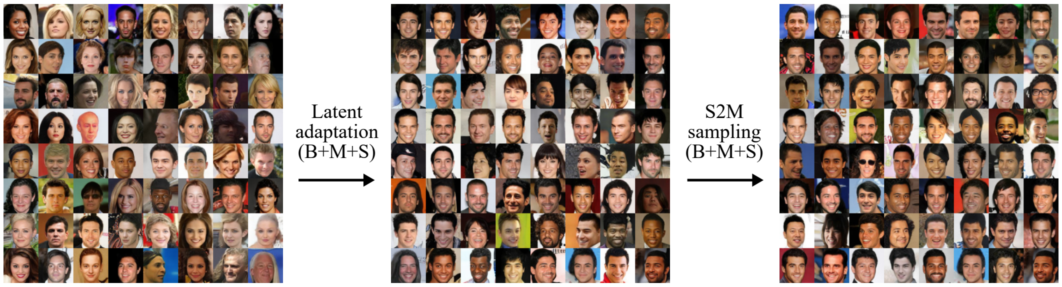
Input: generator G, classifiers D_v^*, D_r^*, intersection index set I, difference index set J, class prior ratios $\gamma_{1:N}$ and target latent distribution parameters $\phi_{1:M}, \mu_{1:M}, \Sigma$

Output: filtered latent sample z

```
Let $\tilde{p}_z(z) \coloneqq \sum_{i=1}^M \phi_i \mathcal{N}(z|\mu_i, \Sigma)$.
Choose any $z$ such that $G(z) \in \operatorname{supp} p_{(I,J)}$.
for $k = 1$ to $K$ do
    Draw $z'$ from $\tilde{p}_z$.
    Draw $u$ from Uniform(0,1).
    $r_i \leftarrow \gamma_i D_r^*(i|G(z))$ for every $i \in I \cup J$
    $r'_i \leftarrow \gamma_i D_r^*(i|G(z'))$ for every $i \in I \cup J$
    $d \leftarrow \frac{\sum_{i=1}^M \phi_i \exp\left(-\frac{1}{2}(z-\mu_i)^T \Sigma^{-1}(z-\mu_i) + \frac{1}{2}z^T z\right)}{\sum_{i=1}^M \phi_i \exp\left(-\frac{1}{2}(z'-\mu_i)^T \Sigma^{-1}(z'-\mu_i) + \frac{1}{2}z'^T z'\right)}$
    $\alpha \leftarrow \min\left(1, \frac{\left(\min\{r'_i : i \in I\} - \max\{r'_j : j \in J\} \cup \{0\}\right)^+ (D_v^*(G(z))^{-1} - 1)}{(\min\{r_i : i \in I\} - \max\{r_j : j \in J\} \cup \{0\})^+ (D_v^*(G(z'))^{-1} - 1)} \cdot d\right)$
    if $u \leq \alpha$ then
        $z \leftarrow z'$
    end if
end for
```

Algorithm 3 *Repeating Latent Adaptation*

```
Input: generator $G$, classifiers $D_v^*, D_r^*$, intersection index set $I$, difference index set $J$, and class prior ratios $\gamma_{1:N}$
Output: filtered latent samples $z_{1:s}$
Let $I_0 \subseteq I$ such that $|I_0| = 1$, and let $J_0 = \emptyset$.
$t_{1:m} \leftarrow$ MHAlgorithm$(G, D_v^*, D_r^*, I_0, J_0, \gamma_{1:N})$    ▷ Obtain latent samples using Algorihtm 1
while $I_0 \cup J_0 \neq I \cup J$ do
    Fit $\tilde{p}_z(z) = \sum_{i=1}^{M} \phi_i \mathcal{N}(z|\mu_i, \Sigma)$ with $t_{1:m}$ using the expectation-maximization algorithm.
    Choose $\alpha \subseteq I \setminus I_0, \beta \subseteq J \setminus J_0$ such that $|\alpha \cup \beta| = 1$.
    $I_0 \leftarrow I_0 \cup \{\alpha\}$
    $J_0 \leftarrow J_0 \cup \{\beta\}$
    $t_{1:m} \leftarrow$ AdaptiveLatentSampling$(G, D_v^*, D_r^*, I_0, J_0, \gamma_{1:N}, \phi_{1:M}, \mu_{1:M}, \Sigma)$    ▷ Obtain latent samples using Algorihtm 2
end while
Fit $\tilde{p}_z(z) = \sum_{i=1}^{M} \phi_i \mathcal{N}(z|\mu_i, \Sigma)$ with $t_{1:m}$ using the expectation-maximization algorithm.
$z_{1:s} \leftarrow$ AdaptiveLatentSampling$(G, D_v^*, D_r^*, I_0, J_0, \gamma_{1:N}, \phi_{1:M}, \mu_{1:M}, \Sigma)$    ▷ Obtain latent samples using Algorihtm 2
```



(1) GAN (Original) (2) GAN w/ **latent adaptation** (3) **S2M sampling** (10 steps)

Figure 9: Results of sequentially applying latent adaptation and S2M sampling for *B+M+S* joint class. (1) Before applying latent adaptation, GANs draw diverse CelebA images. (2) After applying latent adaptation to B+M+S class, the generator produces images of that class with a high probability. (3) By using latent adaptation, S2M sampling can remove samples outside of that class even with a few MCMC iterations.

C Computational Complexity Analysis

Inference time of different architectures. In this section, we compare the inference time of several architectures as shown in Table 3. The classifiers and Gaussian mixture sampler add a small amount of time to the inference process for S2M sampling. To account for these components, we measure the inference time per iteration during the sampling procedure. We use a single RTX 3090 GPU for this experiment.

Table 3: Inference time (ms) of several architectures on CelebA-BMS. PD denote the projection discriminator inference, CLS denote the classifier inference, and GMM denote the gaussian mixture sampler inference. We use the batch size of 64.

Method	CP-GAN	BigGAN	BigGAN + PD	BigGAN + CLS	BigGAN + CLS + GMM
Time (ms)	31.61	29.56	31.66	35.86	36.18

Time complexity of sampling algorithms. To simplify the analysis, we assume that a generator distribution is close to a real distribution, and let α be the probability of a generated sample belonging to a target joint class. Then, we have $\frac{p_{(I,J)}(x)}{p_G(x)} \leq \frac{1}{\alpha}$ for all x. By Theorem 2.1 of Mengersen *et*

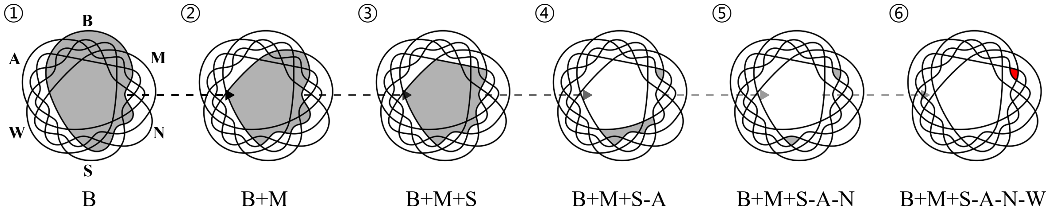


Figure 10: Our latent adaptation can be repeatedly applied for the efficiency of S2M sampling. For instance, to generate images of *B+M+S-A-N-W* joint class (6) with S2M sampling, we can first search the space of *Black-hair* attribute (1) and then gradually increase the number of attributes.

al. [28] the convergence speed of the independent MH algorithm in this condition is given by

$$||P^t(x,\cdot) - p_{(I,J)}||_{TV} \leq (1-\alpha)^t, \tag{24}$$

where $||\cdot||_{TV}$ is the total variation distance and $P^t(x,\cdot)$ is the t-step transition probability density for an initial state x. To analyze the inference time complexity, we now compute the number of MCMC iterations required until the distance is small enough. For a given fixed small ϵ, the number of steps required for the distance to fall within ϵ is $t = \frac{\ln \epsilon}{\ln(1-\alpha)} = O(\frac{1}{\alpha})$. Intuitively, the sampling algorithm without latent adaptation (Algorithm 1) takes time inversely proportional to the relative ratio of the target class samples out of the entire dataset used to train the generator.

Given a fixed number of data, as the number of attributes increases, α decreases, and the algorithm converges slowly. To alleviate such inefficiency, we additionally proposed latent adaptation which collects a certain amount of target class samples using Algorithm 1 and then uses it fit to a new generator distribution close the target distribution. If the newly obtained generator distribution is sufficiently closed to the target distribution, the sampling algorithm using the new generator will take a constant time to produce the target class samples. The remaining question is how long it take to apply latent adaptation.

To discuss about the time complexity of latent adaptation scale with the number of attributes, we let q_i be the probability of a generated sample belonging to the i-th class and assume conditional independence among the classes, *i.e.*, $p(y_i|x, y_1, ..., y_{i-1}, y_{i+1}, ..., y_n) = p(y_i|x)$. Let m be the number of latent samples used to fit the new proposal distribution and c be the overhead introduced by fitting the latent distribution. We consider two scenarios of applying latent adaptation and analyze the time complexity taken to completely fit the latent space in each scenario: (i) Applying latent adaptation by searching the latent space of target joint class samples at once. (ii) Applying latent adaptation repeatedly by increasing the number of specified attributes one by one as shown in Figure 10 (See Algorithm 3 for details). For the case (i), the algorithm runs the MH algorithm once to draw the target class samples and the probability of each generated sample belonging to that class is $\prod_{i\in I} q_i \prod_{j\in J}(1-q_j)$. Hence, it takes $O(m(\prod_{i\in I} q_i \prod_{j\in J}(1-q_j))^{-1} + c)$. For the case (ii), the algorithm runs the MH algorithm once for each class of I and each class of J, so it takes $O(m(\sum_{i\in I} q_i^{-1} + \sum_{j\in J}(1-q_j)^{-1}) + c(|I|+|J|))$. That is, Algorithm 3 can run efficiently as it takes time linearly proportional to the number of attributes. When the number of attributes is small, (i) is also a good way to use latent adaptation, since it has a small overhead of fitting the latent distribution.

To validate the empirical effectiveness of applications of latent adaptation, we select three joint classes of CelebA-ABMNSW, whose ratio of training samples is lower than 3%. For the method (i), sampling for obtaining latent samples is performed by 90 MCMC iterations. For the method (ii), the latent sampling is performed by 15 MCMC iterations for each attribute. For both cases, we collect $10K$ latent samples to perform latent adaptation. After fitting a new latent distribution suit for a target joint class, we again perform a sampling algorithm to draw target joint class samples. We measure the number of MCMC iterations required until accuracy converges. As shown in Table 4, latent adaptation drastically shortens the MCMC chain for all cases, and the method (ii) requires fewer MCMC iterations to draw the target joint class samples than the method (i).

Table 4: Comparison of applications of latent adaptation on CelebA-ABMNSW. For each joint class, we denote the ratio of training samples of the class in the parentheses. "-" in LA type indicates that latent adaptation is not applied. # of searching iterations refers to the total number of MCMC steps performed to fit the latent distribution. # of sampling iterations refers to the number of MCMC steps required until accuracy converges. Accuracy and FID are measured at this step.

Class (Ratio)	LA type	# of searching iterations	# of sampling iterations	Accuracy	FID
B+M+S-A-N-W (1.95%)	-	0	320	53.51%	44.72
	(i)	90	20	57.87%	45.37
	(ii)	90	15	59.08%	46.41
B+M+N+S-A-W (2.68%)	-	0	165	43.78%	38.64
	(i)	90	35	42.70%	38.34
	(ii)	90	5	44.14%	37.15
A+N+W-B-M-S (2.89%)	-	0	335	84.72%	32.03
	(i)	90	40	86.20%	32.22
	(ii)	90	5	86.94%	33.89

D Experiments Details

D.1 MNIST and FMNIST

Each of MNIST and FMNIST dataset consists of a training set of $60k$ images and a test set of $10k$ images. We use 10% of the training set as the validation set. To make the training dataset annotated by single positive labels, we distribute the images belong to the joint classes equally to each corresponding classes of the single positive labels. We evaluate accuracy on MNIST and FMNIST dataset using LeNet5 [36] trained with fully annotated dataset.

The generative models for MNIST and FMNIST are discussed in Section 4.1. As similar to the original setting of GenPU, the generator consists of ReLU activations and fully connected layers of input size: 100-256-256-784. The discriminator consists of ReLU activations and fully connected layers of input size: 784-256-256. As for the GAN objective, we follow the settings introduced by the authors for baselines, and use WGAN-GP [39] for our model. We train all generative models using Adam optimizer [45] with a learning rate of 0.0001, $\beta_1 = 0.5, \beta_2 = 0.999$, and a batch size of 64. The generator is trained for $200k$ iterations, and two updates of the discriminator are performed for every update of the generator.

Classification networks used for S2M sampling are obtained from multiple branches of LeNet5 architecture. We train the classifier using Adam optimizer. For MNIST 3/5 dataset, the classifier is trained for 10 epochs with a learning rate of 0.001, and the temperature of D_r is set to 2. For MNIST and FMNIST Even dataset, the classifier is trained for 50 epochs with a learning rate of of 0.0001, and the temperature of D_v is set to 4. Each γ_k corresponding to the intersection set is set to 0.1 for both MNIST and FMNIST.

D.2 CIFAR-10 and CelebA

For CIFAR-10 dataset, we use 10% of the training set as the validation set. We follow the original partition description and resize images to 64×64 for training efficiency on CelebA dataset. To make the training datasets annotated by single positive labels, we distribute the images belong to the joint classes equally to each corresponding classes of the single positive labels. We evaluate accuracy on CIFAR-10 and CelebA dataset using MobileNet V2 [46] trained with fully annotated dataset.

The generative models for CIFAR-7to3 and CelebA datasets are discussed in Section 4.2. We use BigGAN [2] architecture for all generative models and follow the PyTorch implementation[4]. We use hinge loss [47] as the GAN objective and apply spectral normalization [48]. We train all models using Adam optimizer [45] with a learning rate of 0.0002, $\beta_1 = 0.5, \beta_2 = 0.999$, and a batch size of of 64.

---

[4] `https://github.com/POSTECH-CVLab/PyTorch-StudioGAN`

The generator is trained for $100k$ iterations, and five updates of the discriminator are performed for every update of the generator. We select the model achieving best FID on the validation dataset.

Classification networks used for S2M sampling are obtained from multiple branches of MobileNet V2 architecture. We first train the classifier with only $\mathcal{L}_r$ during 200 epochs for CIFAR-7to3 and 30 epochs for CelebA-BMS, CelebA-ABMNSW dataset. We use SGD optimizer with a learning rate of 0.1 and cosine annealing for this training. Then, the classifier is trained with the sum of all classification losses for $30k$ iterations. For CIFAR-7to3 and CelebA-BMS dataset, we set the temperature of D_r and D_f as 0.2, 1.0, and 1.2 when the size of difference index set is 0, 1, and 2, respectively. For CelebA-ABMNSW dataset, we set the temperature of D_r and D_f as 0.1, 0.1, 1.0, 1.0, 1.6 and 1.6 when the size of difference index set is 0, 1, 2, 3, 4, 5 and 6, respectively. Each γ_k corresponding to the intersection set is set to 0.1 for CelebA-BMS and 0.5 for CelebA-ABMNSW. On CIFAR-7to3 dataset, we set 0.5 and 0.8 to this parameter for unconditional GAN and cGAN-PD, respectively.

D.3 CelebA-HQ

For CelebA-HQ dataset, we follow the original partition description and use the images with the resolution of 256×256. To make the training dataset annotated by single positive labels (*i.e.*, Black hair, Man and Smiling), we distribute the images belong to the joint classes equally to each corresponding classes of the single positive labels.

We use the pretrained StyleGAN V2 [49]. Classification networks used for S2M sampling are obtained from multiple branches of MobileNet V2 [46] architecture. We first train the classifier with only $\mathcal{L}_r$ during 30 epochs. We use SGD optimizer with a learning rate of 0.1 and cosine annealing for this training. Then, the classifier is trained with the sum of all classification losses for $1k$ iterations. We set the temperature of D_r as 0.2, 1.0, and 1.2 when the size of difference index set is 0, 1, and 2, respectively. Each γ_k corresponding to the intersection set is set to 0.5.

E Additional Experimental Results

E.1 2×16 Gaussians Example

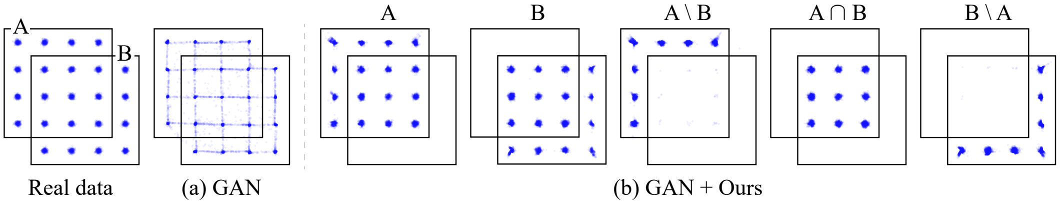


Figure 11: Example of 2×16 Gaussians. Using the outputs of original GAN, S2M Sampling samples high-quality points within various conditions $(\mathrm{A}, \mathrm{B}, \mathrm{A} \setminus \mathrm{B}, \mathrm{B} \setminus \mathrm{A}, \mathrm{A} \cap \mathrm{B})$.

To provide an illustrative example, we modify *25 Gaussians* [20, 21, 25] to have two 4×4 grids of two-dimensional Gaussians of two classes A and B (See Figure 11). The 23 modes in 2×16 Gaussians are horizontally and vertically spaced by 1.0 and have a standard deviation of 0.05. We first train a GAN to randomly draw points within two grids using WGAN-GP [39] as the GAN objective. To apply S2M sampling, we train the classifier to classify each point into two classes A and B. The generator, discriminator, and classification networks consist of ReLU activations and fully connected layers of input size: 2-512-512-512.

In this setting, S2M sampling attempts to draw points of given classes (A, B), the overlapping class $(A \cap B)$, and the non-overlapping classes $(A \setminus B, B \setminus A)$. We obtain samples at 400 MCMC iterations. As shown in Figure 11, S2M sampling correctly draw points of the target classes. On the other hand, GAN tends to generate spurious lines between the points. For the quantitative analysis, we report the accuracy, high-quality ratio, and mode standard deviation. We generate $10k$ samples and assign each point to the mode with the closest L_2 distance for measuring the accuracy. Following Turner *et al.* [21], samples whose L_2 distances are within four standard deviations are considered as

Table 5: Accuracy (%), high-quality ratio (%), and mode standard deviation on 2×16 Gaussians.

Condition	GAN			**GAN + Ours**		
	Accuracy	High Quality	Std. Dev.	Accuracy	High Quality	Std. Dev.
A, B	69.83$\pm$0.35	84.39$\pm$0.60	0.106$\pm$0.002	100.00$\pm$0.00	98.94$\pm$0.40	0.052$\pm$0.002
A \ B, B \ A	30.17$\pm$0.35	88.87$\pm$0.50	0.090$\pm$0.002	99.52$\pm$0.36	98.67$\pm$0.51	0.051$\pm$0.002
A ∩ B	39.66$\pm$0.46	80.98$\pm$0.82	0.118$\pm$0.003	100.00$\pm$0.00	99.73$\pm$0.14	0.050$\pm$0.001

high-quality samples. As shown in Table 5, S2M sampling accurately draw samples for all conditions, and the ratio of high-quality samples is improved by 14.36% on average.

E.2 CelebA-HQ 256×256

To validate whether S2M sampling can be adopted to state-of-the-art architecture on high resolution (256 × 256) image dataset, we evaluate the performance of S2M sampling with the pretrained StyleGANv2 [49] on CelebA-HQ dataset. The quantitative results for StyleGANv2 with and without S2M sampling are shown in Table 6, and the qualitative results are depicted in Section E.5.

Table 6: Quantitative results of StyleGANv2 with S2M sampling.

Method	Accuracy (↑)	FID (↓)
StyleGANv2	15.44%	17.08
StyleGANv2 + **Ours**	77.18%	14.64

E.3 Data Embedding Visualization.

To visually probe the effect of S2M sampling, we perform t-SNE [50, 51] on generated samples. We embed Inception-V3 activations of samples generated by various models on CIFAR-7to3. As shown in Figure 12, S2M sampling accurately draw samples for all classes compared to other generative models.

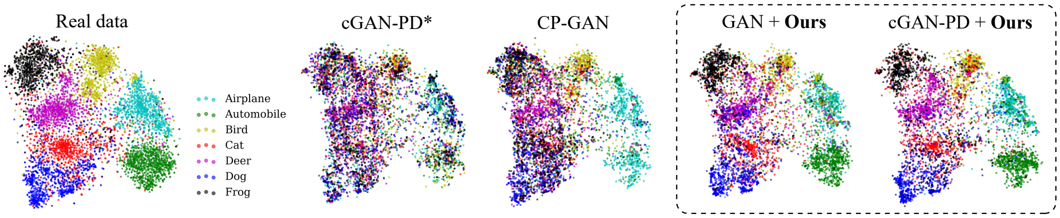


Figure 12: t-SNE visualization results for generated samples on CIFAR-7to3 dataset. Samples drawn from S2M sampling are embedded similarly to the real data.

E.4 Ablation Study

Since classifiers used in S2M sampling are not optimal in practice, we correct the sampling algorithm by scaling the temperatures of the classifiers and adjusting γ_k. Temperature scaling [30] is useful technique to control the confidence of classifier. If the temperature of D_f and D_r is small, samples that are likely to belong to overlapping classes tend to be drawn mostly in S2M sampling. Another approach is to adjust γ_k. We can obtain samples clearly distinct from classes of a difference set by decreasing γ_i for all $i \in I$.

To validate the effects of the temperature scaling and γ adjustment in S2M sampling, we perform the ablation studies on CIFAR-7to3 and CelebA-BMS dataset. The hyperparameter values used are explained in Appendix D. In Table 7, we report the average accuracy and FID when S2M sampling is applied to unconditional GAN and cGAN-PD. With the proper adjustment of the hyperparameters, accuracy can be highly improved while FID is maintained or slightly degraded.

Table 7: Ablation study for the hyperparameters of S2M sampling.

Method	Metric	CIFAR-7to3		CelebA-BMS	
		GAN	cGAN-PD	GAN	cGAN-PD
Sampling w/ actual logits	Acc. (↑) FID (↓)	62.73±1.63 14.99±0.17	72.33±0.52 14.11±0.23	71.14±1.46 9.86±1.41	73.32±4.62 10.29±0.83
+ temperature scaling	Acc. (↑) FID (↓)	68.40±0.82 14.81±0.23	76.32±1.44 14.12±0.31	74.99±1.95 9.81±1.21	78.50±2.63 10.21±0.41
+ γ adjustment	Acc. (↑) FID (↓)	77.65±1.22 14.42±0.55	80.62±2.08 14.14±0.34	85.22±4.27 10.50±0.97	90.44±1.05 10.63±0.29

E.5 Qualitative Results

In this section, we provide the qualitative results for the experiments on CIFAR-7to3, CelebA-BMS, and CelebA-HQ. The qualitative results for CIFAR-7to3 and CelebA-BMS are shown in Figure 13 and Figure 14, respectively. One of advantages of S2M sampling is that it can be readily built upon existing state-of-the-art GANs. Figure 15 represents the samples with the resolution of 256 × 256 when we adopted S2M sampling to pretrained StyleGAN V2 [49] on CelebA-HQ dataset.

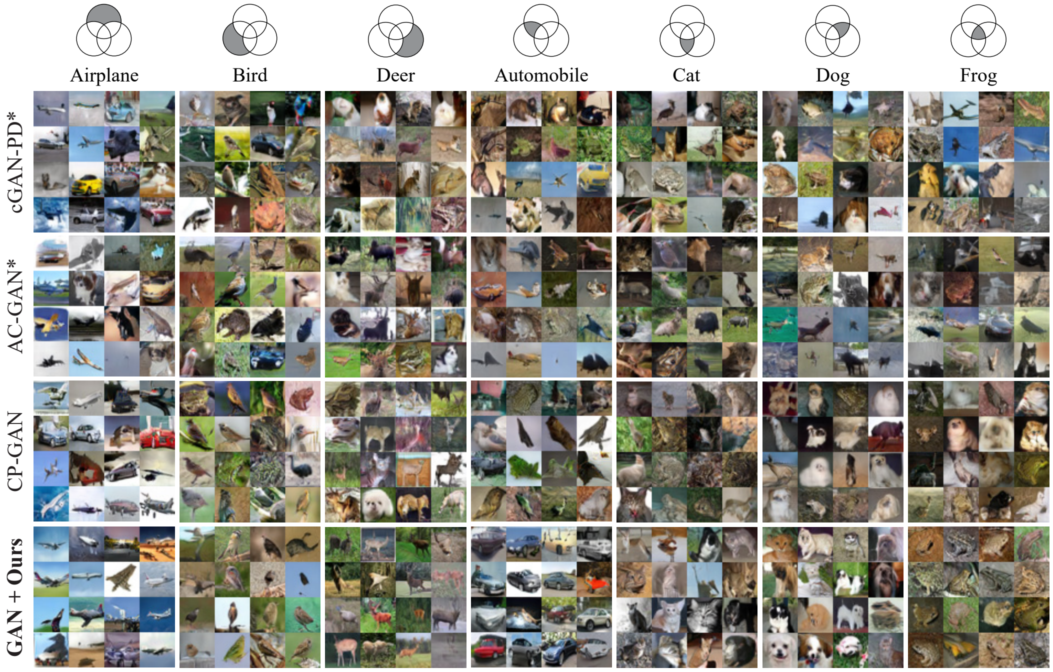


Figure 13: Qualitative results for cGAN-PD*, AC-GAN*, CP-GAN, and S2M sampling with GAN on CIFAR-7to3. For cGAN-PD*, AC-GAN* and CP-GAN, we provide $1/n$ as the labels for each class to generate images belonging to n classes.

F S2M Sampling with Different Attributes

In this section, we examine S2M sampling with various attributes appearing in CelebA dataset. Concretely, we conduct two additional datasets: (i) CelebA-BBM consisting of classes of *Brown-hair*, *Bushy-eyebrows*, and *Mouth-slightly-opens* attributes, (ii) CelebA-HBW consisting of classes of *High-cheekbones*, *Bags-under-eye*, and *Wavy-hair* attributes. Those attributes are chosen due to their strong visual impact. For both datasets, we adopt S2M sampling to pretrained StyleGANv2.

Quantitative Results. Table 8 shows the quantitative results of S2M sampling. As shown in Table 8, we can observe that S2M sampling correctly draws samples corresponding to each joint class for

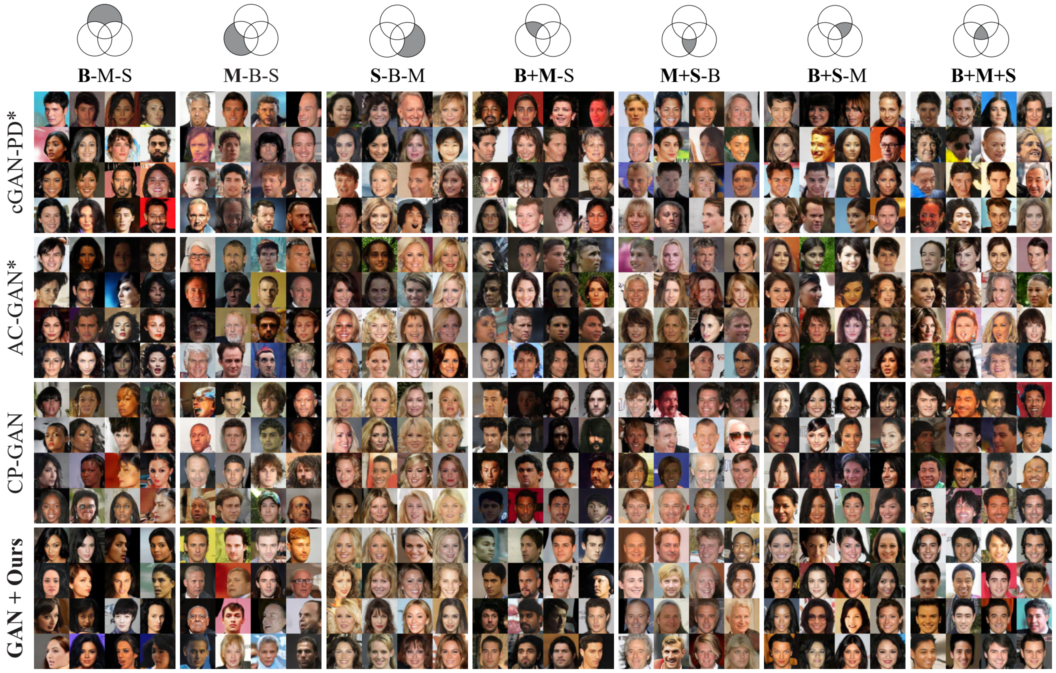


Figure 14: Qualitative results for cGAN-PD*, AC-GAN*, CP-GAN, and S2M sampling with GAN on CelebA-BMS. For cGAN-PD*, AC-GAN* and CP-GAN, we provide $1/n$ as the labels for each class to generate images belonging to n classes. Intersections and differences are denoted by plus signs and minus signs, respectively.

Table 8: Quantitative results of StyleGANv2 with S2M sampling on two variants of CelebA-BMS: CelebA-BBM and CelebA-HBW.

Dataset	Method	Accuracy (↑)	FID (↓)
CelebA-BBM	StyleGANv2	18.41%	15.57
	StyleGANv2 + **Ours**	**74.97%**	**14.41**
CelebA-HBW	StyleGANv2	12.68%	15.80
	StyleGANv2 + **Ours**	**70.94%**	**14.54**

both datasets, which indicates that S2M sampling can be efficiently performed with various types of attributes.

Qualitative Results. For the qualitative results, we depict samples drawn by S2M sampling for three joint classes of CelebA-BBM and CelebA-HBW in Figure 16. For all joint classes, the samples in Figure 16 are randomly selected from the outputs of S2M sampling.

G Consideration of other sampling approaches for applying S2M Sampling

In this section, we provide insight to apply S2M sampling with other sampling methods. We mainly considered applying three sampling algorithms which can be found in previous GAN sampling studies [20, 21, 25, 29]; Rejection sampling, Independent Metropolis-Hastings algorithm, and Langevin dynamics. Here, we briefly discuss the pros and cons of each sampling method we faced in our problem settings.

Rejection Sampling As discussed in DRS [20], the rejection sampling can be applied to sample from the target distribution $p_t(x)$ if we can compute the ratio between the target density $p_t(x)$ and the generator density $p_G(x)$, and the upper bound of the density ratio $p_t(x)/p_G(x)$. However, it is in general difficult and expensive to compute this upper bound in the high dimensional data space.

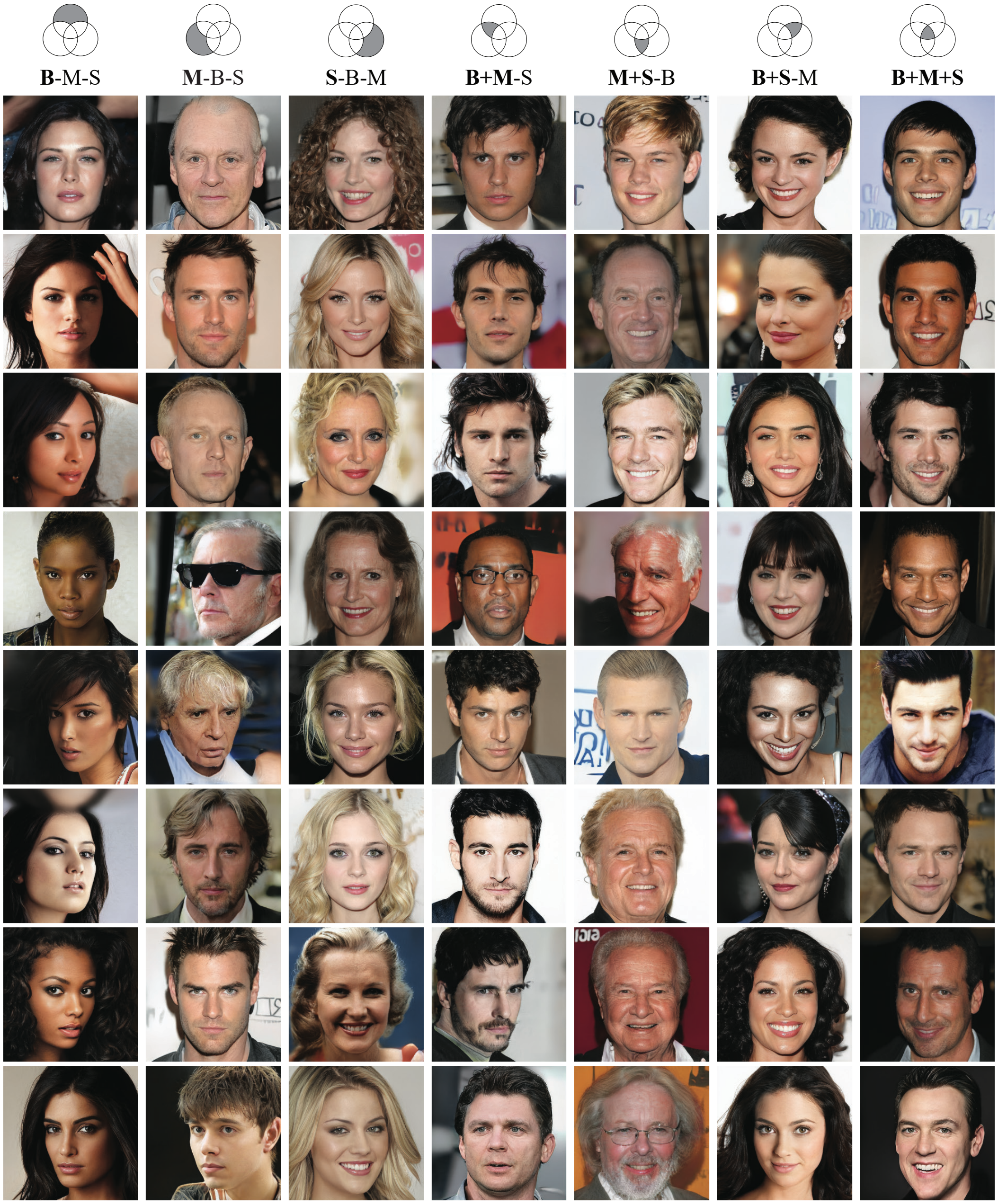


Figure 15: Qualitative results of applying S2M sampling to StyleGANv2 on CelebA-HQ. Intersections and differences are denoted by plus signs and minus signs, respectively.

We need several heuristics to mitigate the issues, e.g., shifting the logit score of the classifier used to compute the density ratio as introduced by DRS [20], which may introduce additional non-trivial efforts for hyperparameter searching.

Independent Metropolis-Hastings algorithm This algorithm can also be used to draw samples from the target distribution if we can compute the density ratio $p_t(x)/p_G(x)$. Unlike Rejection sampling, we do not need to compute the upper bound of this density ratio, but a sequence of samples forming a Markov chain is required. Our study mainly used this algorithm because it empirically performed well without complex heuristics and the sampling accuracy can be readily controlled at the cost of MCMC steps. To further mitigate the sample efficiency in our problem, we suggest the latent adaptation technique.

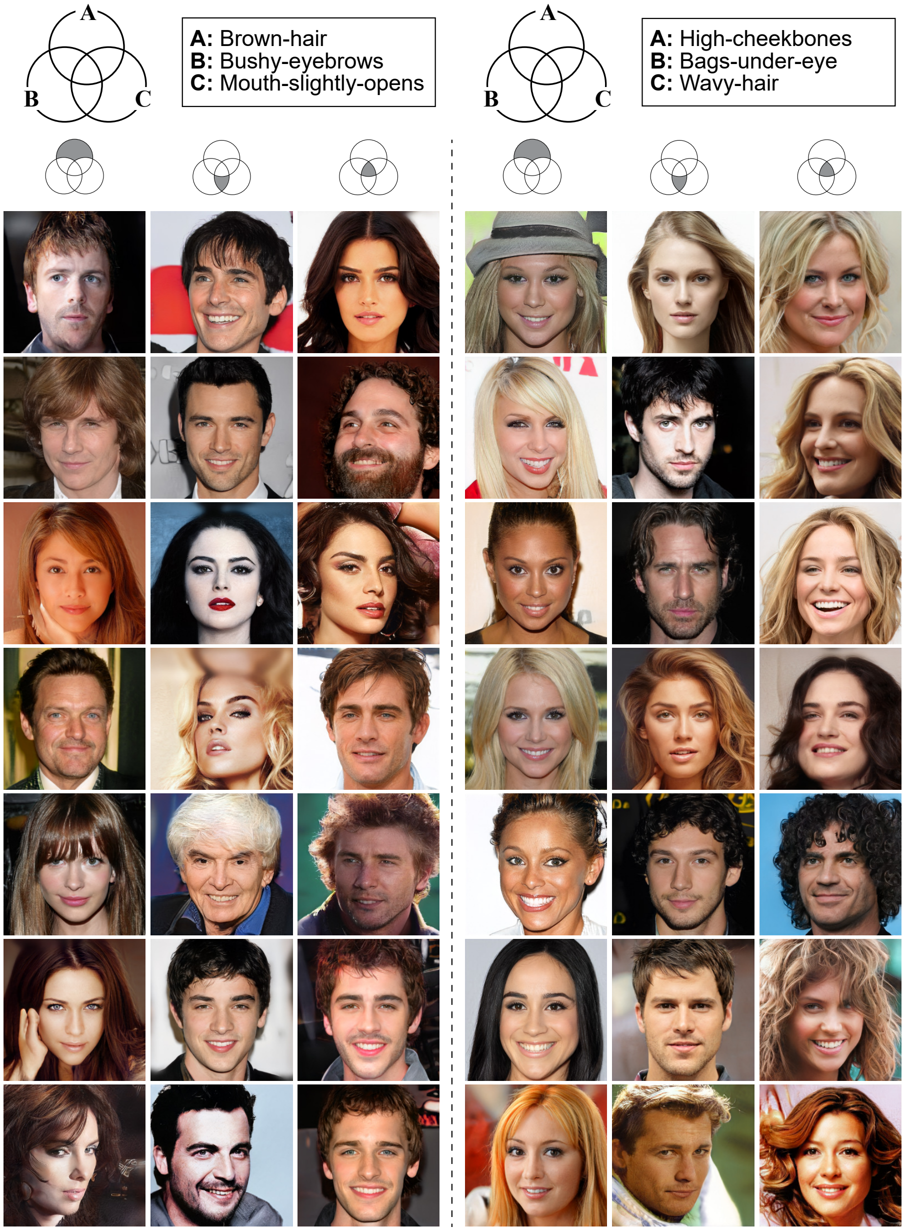


Figure 16: Qualitative results of applying S2M sampling to StyleGANv2 on two varaints of CelebA-BMS: CelebA-BBM (left) and CelebA-HBW (right).

Langevin dynamics Langevin dynamics is a gradient-based MCMC approach which can also be used when we can compute the density ratio $p_t(x)/p_G(x)$. Several studies [25, 29] employ its Euler-Maruyama discretization to improve the quality of GAN samples. While this algorithm can efficiently push a chain of samples towards the target distribution, the step size of the algorithm is very sensitive to the sampling cost and quality. Especially, in our problem, we need to deal with the case that the sample falls within the space where the gradient is not well-defined, i.e. $\operatorname{supp} p_G \setminus \operatorname{supp} p_t$. We did not use the algorithm as we could not find an effective way to address these issues.

H Potential Societal Impacts

This work demonstrates that it is possible to generate multi-label data from limited labels. In addition, this work can be freely adopted to unconditional GANs trained with a large amount of unlabeled data. Hence, our work can reduce the high annotation cost that research groups face in common. Despite the fact that deep learning models tend to struggle from learning underrepresented data [52], properly calibrated sampling algorithm does not readily ignore rarely appearing data, meaning that it is unlikely to introduce bias into generative models.