Supplementary Materials
HexaGAN: Generative Adversarial Nets for Real World Classification

Uiwon Hwang [1] **Dahuin Jung** [1] **Sungroh Yoon** [1 2]

1. Proofs

1.1. Global optimality of $p(\mathbf{x}|\mathbf{m}_i = 1) = p(\mathbf{x}|\mathbf{m}_i = 0)$ for HexaGAN

Proof of Theorem 1: Let $D_{MI}(\cdot)$ be $D(\cdot)$, and $G_{MI}(E(\cdot))$ be $G(\cdot)$ for convenience.

The min-max loss of HexaGAN for missing data imputation is given by:

$$V_{MI}(D,G) = \mathbb{E}_{\mathbf{x},\mathbf{z},\mathbf{m}}\left[\mathbf{m}^T D(G(\tilde{\mathbf{x}}|\mathbf{m})) - (\mathbf{1}-\mathbf{m})^T D(G(\tilde{\mathbf{x}}|\mathbf{m}))\right] \tag{1}$$

$$= \mathbb{E}_{\hat{\mathbf{x}},\mathbf{m}}\left[\mathbf{m}^T D(\hat{\mathbf{x}}) - (\mathbf{1}-\mathbf{m})^T D(\hat{\mathbf{x}})\right] \tag{2}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{\mathbf{m}\in\{0,1\}^d} \left(\mathbf{m}^T D(\mathbf{x}) - (1-\mathbf{m})^T D(\mathbf{x})\right) p(\mathbf{x}|\mathbf{m}) d\mathbf{x} \tag{3}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{\mathbf{m}\in\{0,1\}^d} \left(\sum_{i:m_i=1} D(\mathbf{x})_i - \sum_{i:m_i=0} D(\mathbf{x})_i\right) p(\mathbf{x}|\mathbf{m}) d\mathbf{x} \tag{4}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(D(\mathbf{x})_i \sum_{\mathbf{m}:m_i=1} p(\mathbf{x}|\mathbf{m}) - D(\mathbf{x})_i \sum_{\mathbf{m}:m_i=0} p(\mathbf{x}|\mathbf{m})\right) d\mathbf{x} \tag{5}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} D(\mathbf{x})_i p(\mathbf{x}|m_i = 1) - D(\mathbf{x})_i p(\mathbf{x}|m_i = 0) d\mathbf{x} \tag{6}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) D(\mathbf{x})_i d\mathbf{x} \tag{7}$$

For a fixed G, the optimal discriminator $D(\mathbf{x})_i$ which maximizes $V_{MI}(D,G)$ is such that:

$$D^*_G(\mathbf{x})_i = \begin{cases} 1, & \text{if } p(\mathbf{x}|m_i = 1) \geq p(\mathbf{x}|m_i = 0) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Plugging D^*_G back into Equation 7, we get:

$$V_{MI}(D^*_G, G) = \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) D^*_G(\mathbf{x})_i d\mathbf{x} \tag{9}$$

$$= \sum_{i=1}^{d} \int_{\{\mathbf{x}|p(\mathbf{x}|m_i=1)\geq p(\mathbf{x}|m_i=0)\}} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) d\mathbf{x} \tag{10}$$

[1]Electrical and Computer Engineering, Seoul National University, Seoul, Korea [2]ASRI, INMC, Institute of Engineering Research, Seoul National University, Seoul, Korea. Correspondence to: Sungroh Yoon <sryoon@snu.ac.kr>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

Let $\mathcal{X} = \{\mathbf{x}|p(\mathbf{x}|m_i = 1) \geq p(\mathbf{x}|m_i = 0)\}$. To minimize Equation 10, we need to set $p(\mathbf{x}|m_i = 1) = p(\mathbf{x}|m_i = 0)$ for $\mathbf{x} \in \mathcal{X}$.

Then, when we consider $\mathcal{X}^c$, the complement of $\mathcal{X}$, $p(\mathbf{x}|m_i = 1) < p(\mathbf{x}|m_i = 0)$ for $\mathbf{x} \in \mathcal{X}^c$. Since both probability density functions should integrate to 1,

$$\int_{\mathcal{X}^c} p(\mathbf{x}|m_i = 1)d\mathbf{x} = \int_{\mathcal{X}^c} p(\mathbf{x}|m_i = 0)d\mathbf{x} \tag{11}$$

However, this is a contradiction, unless $\lambda(X^c) = 0$ where λ is the Lebesgue measure. This finishes the proof. □

1.2. Optimization of components for imputation

From Equation 6,

$$V_{MI}(D, G)_i = \int_{\hat{\mathcal{X}}} p(\mathbf{x}|m_i = 1)D(\mathbf{x})_i - p(\mathbf{x}|m_i = 0)D(\mathbf{x})_i d\mathbf{x} \tag{12}$$

$$= \mathbb{E}_{\tilde{\mathbf{x}},\mathbf{z},\mathbf{m}}\left[m_i \cdot D(G(\tilde{\mathbf{x}}|\mathbf{m}))_i\right] - \mathbb{E}_{\tilde{\mathbf{x}},\mathbf{z},\mathbf{m}}\left[(1 - m_i) \cdot D(G(\tilde{\mathbf{x}}|\mathbf{m}))_i\right] \tag{13}$$

G is then trained according to $\min_G \sum_{i=1}^{d} V_{MI}(D, G)_i$, and D is trained according to $\max_D \sum_{i=1}^{d} V_{MI}(D, G)_i$.

1.3. Relation between pseudo-labeling and the ODM cost

Proof of Theorem 2: Optimizing the adversarial loss functions L_C and $L_{D_{MI}}^{d+1}$ are equivalent to minimizing the Earth Mover distance between $\text{Distr}[C(\hat{\mathbf{x}}_u)]$ and $\text{Distr}[\mathbf{y}]$, where $\text{Distr}[\cdot]$ denotes the distribution of a random variable.

Since converging the Earth Mover distance $W(p, q)$ to zero implies that the two distributions p and q are equal, the following proposition holds

$$W(\text{Distr}[C(\hat{\mathbf{x}}_u)], \text{Distr}[\mathbf{y}]) \to 0 \;\; \Rightarrow \;\; \text{Distr}[C(\hat{\mathbf{x}}_u)] = \text{Distr}[\mathbf{y}] \tag{14}$$

This means that minimizing the Earth Mover distance $W(\text{Distr}[C(\hat{\mathbf{x}}_u)], \text{Distr}[\mathbf{y}])$ matches the distributions of the outputs. Therefore, the adversarial losses of D_{MI} and C satisfy the definition of the output distribution matching (ODM) cost function, concluding the proof. □

2. Training of HexaGAN in details

2.1. Dataset description

Table 1 presents the dataset descriptions used in the experiments. The imbalance ratio of the wine dataset is calculated from the binarized classes by combining classes 2 and 3 into one class, and the numbers of data in the three classes are 59, 71, and 48, respectively.

Table 1. Dataset description. The imbalance ratio indicates the ratio of the number of instances in the majority class to the number of instances in the minority class.

Dataset	# of features	# of instances	Imbalance ratio (1:x)
Breast	30	569	1.68
Credit	23	30,000	3.52
Wine (with binarized class)	13	178	2.02
Madelon	500	4,400	1.00

2.2. Training procedure

Each component of the whole system is updated in order. We should note that the distribution of $\mathbf{h}_l$ is altered by the updating of E; thus, we updated G_{CD} and D_{CG} several times when the other components are updated once, as shown in Algorithm 1. We set the number of iterations for the conditional generation per an iteration for the other components to 10 and the number of iterations for discriminators per an iteration for generators to 5 in our experiments.

Algorithm 1 Training procedure of HexaGAN

Require : n_{CG} - the number of iterations for the conditional generation per an iteration for the other components;
n_{critic} - the number of iterations for discriminators per an iteration for generators

while training loss is not converged **do**

(1) Missing data imputation

for $k = 1, ..., n_{critic}$ **do**

Update D_{MI} using stochastic gradient descent (SGD)

$\nabla_{D_{MI}} \mathcal{L}_{D_{MI}} + \mathcal{L}_{D_{MI}}^{d+1} + \lambda_1 \mathcal{L}_{\mathrm{GP}_{MI}}$

end for

Update E using SGD

$\nabla_E \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$

Update G_{MI} using SGD

$\nabla_{G_{MI}} \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$

(2) Conditional generation

for $i = 1, ..., n_{CG}$ **do**

for $j = 1, ..., n_{critic}$ **do**

Update D_{CG} using SGD

$\nabla_{D_{CG}} \mathcal{L}_{D_{CG}} + \lambda_2 \mathcal{L}_{\mathrm{GP}_{CG}}$

end for

Update G_{CG} using SGD

$\nabla_{G_{CG}} \mathcal{L}_{G_{CG}} + \alpha_2 \mathcal{L}_{G_{MI}} + \alpha_3 \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$

end for

(3) Semi-supervised classification

Update C using SGD

$\nabla_C \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_{l,c}, \mathbf{y}_{l,c}) + \alpha_4 L_C$

end while

2.3. Architecture of HexaGAN

Excluding the experiments in Sections 4.1.2 and 4.2, all six components used an architecture with three fully-connected layers. The number of hidden units in each layer is d, $d/2$, and d. As an activation function, we use the rectified linear unit (ReLU) function for all hidden layers and the output layer of E and G_{CG}, the sigmoid function for the output layer of G_{MI} and D_{CG}, no activation function for the output layer of D_{MI}, and the softmax function for the output layer of C.

Table 2 describes the network architectures used in Sections 4.1.2 and 4.2. In the table, FC(n) denotes a fully-connected layer with n output units. Conv(n, $k \times k$, s) denotes a convolutional network with n feature maps, filter size $k \times k$, and stride s. Deconv(n, $k \times k$, s) denotes a deconvolutional network with n feature maps, filter size $k \times k$, and stride s.

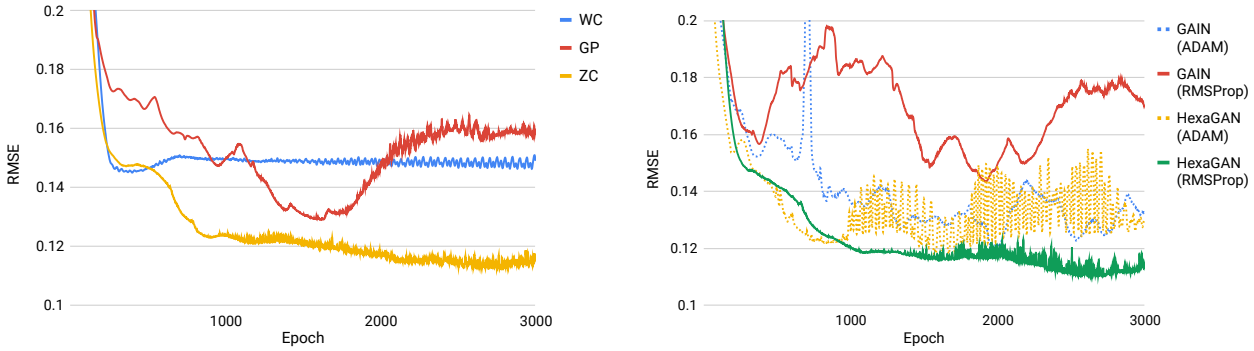
Table 2. Convolutional neural network architectures used for the MNIST dataset

G_{CG}	D_{CG}	E	G_{MI}	D_{MI}	C
FC(512)	FC(1024)	Conv(32, 5×5, 2)	Deconv(64, 5×5, 2)	Conv(32, 5×5, 2)	Conv(32, 5×5, 2)
ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
FC(1024)	FC(512)	Conv(64, 5×5, 2)	Deconv(32, 5×5, 2)	Conv(64, 5×5, 2)	Conv(64, 5×5, 2)
ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
FC(2048)	FC(1)	Conv(128, 5×5, 2)	Deconv(1, 5×5, 2)	Conv(128, 5×5, 2)	Conv(128, 5×5, 2)
ReLU	Sigmoid	ReLU	ReLU	ReLU	ReLU
			FC(784)	FC(785)	FC(10)
			Sigmoid	Sigmoid	Softmax

3. Additional experiments

3.1. Learning curve analysis on missing data imputation

Using the breast dataset, we measured the RMSE to evaluate the imputation performance of the proposed adversarial losses ($\mathcal{L}_{D_{MI}}$, $\mathcal{L}_{G_{MI}}$). We excluded $\mathcal{L}_{\text{recon}}$ from the losses of E and G_{MI} and compared the learning curves of weight clipping (WC) proposed by Arjovsky et al. (2017), the modified gradient penalty (GP) of Gulrajani et al. (2017), and the modified zero-centered gradient penalty (ZC, ours) to determine the most appropriate gradient penalty for our framework. As shown in Figure 1(a), ZC shows stable and good performance (small RMSE). In Figure 1(b), we plot learning curves to accurately compare the adversarial losses of GAIN and HexaGAN. We also compare the two optimizers ADAM (Kingma & Ba, 2014) and RMSProp (Tieleman & Hinton, 2012). Our experiment shows that RMSProp is a more stable optimizer than ADAM, and HexaGAN produces a more stable and better imputation performance than GAIN.



(a) Comparison of the gradient penalty

(b) Comparison of the adversarial loss and optimizer

Figure 1. Learning curve comparison for the optimal GAN imputation method

3.2. Imputation performance with respect to the missing rate

We measured the imputation performance of HexaGAN for various missing rates in the credit dataset. To compare the performance with those of competitive benchmarks, we used MICE, which is a state-of-the-art machine learning algorithm, and GAIN, which is a state-of-the-art deep generative model. As seen in Figure 2, HexaGAN shows the best performance for all missing rates except 50%. The comparison of MICE and HexaGAN shows that the gap between the performances of the two methods increases at higher missing rates; therefore, HexaGAN is more robust when there is less information available.

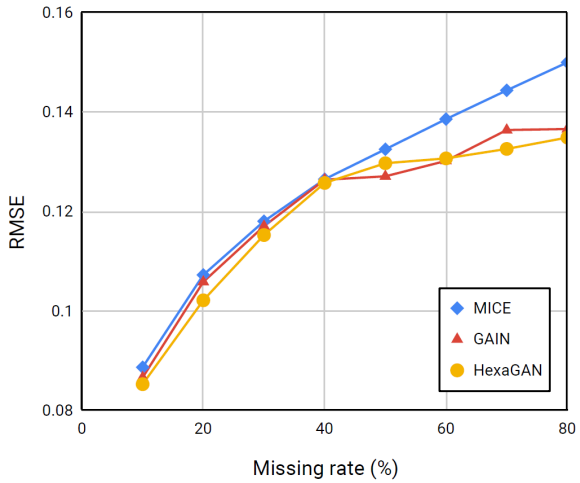


Figure 2. Imputation performance (RMSE) comparison with respect to the missing rate with the credit dataset

3.3. tSNE analysis on conditional generation

Figure 3 is the complete version of the tSNE analysis in Section 4.2.1. The tSNE plot below shows an analysis of the manifold of the hidden space. We confirm that the synthetic data around the original data looks similar to the original data. Therefore, it can be seen that E learns the data manifold well in the hidden space.

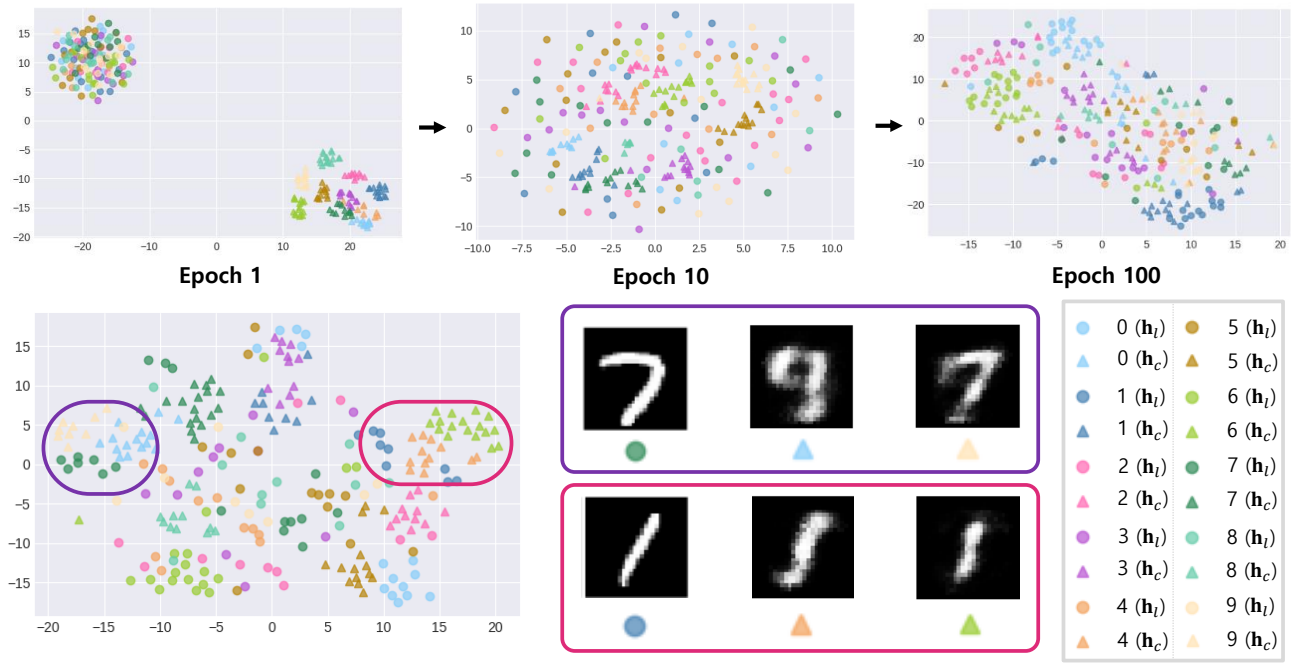


Figure 3. tSNE analysis with the MNIST dataset

3.4. Sensitivity analysis of loss functions

We performed diverse experiments by tuning the hyperparameter of each loss term for the missing data imputation and conditional generation experiments. We utilized the credit dataset and measured the RMSE and F1-score. The first two rows of Table 3 show the imputation performances (RMSE) acheived by tuning hyperparameters α_1 and λ_1, which are multiplied by the auxiliary loss terms for missing data imputation ($\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{GP}_{MI}}$, respectively). The results show that HexaGAN achieves the best missing data imputation performance when both α_1 and λ_1 are set to 10. The last two rows of Table 3 present the classification performances (F-score) acheived by tuning hyperparameters α_2 and α_3, which are multiplied by the auxiliary losses for conditional generation ($\mathcal{L}_{G_{MI}}$ and $\mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$, respectively). As a result, the best classification performance is obtained when α_2 and α_3 are the default values in our paper, at 1 and 0.01, respectively.

Table 3. Sensitivity analysis of the loss functions with the credit dataset

Hyperparameter (Loss)	Setting	1	2	3	4
α_1 ($\mathcal{L}_{\text{recon}}$)	Value	0	1	**10**	100
	RMSE	0.1974	0.1108	**0.1022**	0.1079
λ_1 ($\mathcal{L}_{\text{GP}_{MI}}$)	Value	0	1	**10**	100
	RMSE	0.1110	0.1097	**0.1022**	0.1081
α_2 ($\mathcal{L}_{G_{MI}}$)	Value	0	**1**	10	100
	F1-score	0.4535	**0.4627**	0.4585	0.4523
α_3 ($\mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$)	Value	0	**0.01**	0.1	1
	F1-score	0.4535	**0.4627**	0.4585	0.4523

3.5. Statistical significance

We conducted statistical tests for Tables 1, 2, and 3 in the original paper. Because the results of the experiment could not meet the conditions of normality and homogeneity of variance tests, we used a non-parametric test, the Wilcoxon rank sum test. We additionally measured the effect size using Cohen's d. We validated that all the experiments are statistically significant or showed large or medium effect size, except for GAIN vs. HexaGAN for the wine dataset in Table 1, HexaGAN without D_{MI} vs. HexaGAN for the breast and credit datasets in Table 2, and MICE + SMOTE + TripleGAN vs. HexaGAN for the madelon dataset in Table 3.

3.6. Classification performance with the CelebA dataset

We used a more challenging dataset, CelebA. It is a high-resolution face dataset for which it is more difficult to impute missing data. CelebA consists of 40 binary attributes with various imbalance ratios (1:1 1:43). We used 50,000 and 10,000 labeled and unlabeled training images, respectively, and 10,000 test images. The size of each image is 218x178x3, which means that the data dimension is 116,412. Therefore, we could evaluate our method on the setting where the data dimension is less than the sample size. Then, half of the elements were removed from each image under the 50% missingness (MCAR) assumption.

For comparison, we utilized a class rectification loss (CRL) (Dong et al., 2018) which is the most recent method developed for the class imbalance problem. Since an image has 40 labels simultaneously, we simply balanced the class of data entered into C by setting the class condition to $\mathbf{1} - \mathbf{y}$. Additionally, the data dimension was too large to calculate $\mathcal{L}_{\mathrm{GP}_{MI}}$, therefore we replaced the regularization for discriminator learning with weight clipping. We measured the F1-scores for 40 attributes for three cases: GAIN + TripleGAN, GAIN + CRL + TripleGAN, and HexaGAN. The same structure and hyperparameters were used for the classifier for a fair comparison. Table 4 shows the imbalance ratio of each attribute and the classification performance (F1-score) of each combination. Comparing the average F1-score of 40 attributes, GAIN + TripleGAN shows a performance of 0.5152, GAIN + CRL + TripleGAN has a performance of 0.5519, and HexaGAN has a performance of 0.5826. HexaGAN outperforms all the compared methods.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pp. 214–223, 2017.

Dong, Q., Gong, S., and Zhu, X. Imbalanced deep learning by minority class incremental rectification. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pp. 5767–5777, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Table 4. Classification performance comparison with the CelebA dataset (F1-score)

Attribute	Imb. ratio (1:x)	GAIN + TripleGAN	GAIN + CRL + TripleGAN	HexaGAN
Arched eyebrows	3	0.53	0.50	**0.55**
Attractive	1	**0.78**	0.74	0.74
Bags under eyes	4	0.30	0.44	**0.49**
Bald	43	0.37	**0.42**	0.35
Bangs	6	0.70	**0.77**	0.71
Big lips	3	0.17	0.20	**0.39**
Big nose	3	0.41	0.47	**0.49**
Black hair	3	0.67	**0.72**	0.69
Blond hair	6	**0.77**	0.74	0.71
Blurry	18	0.02	**0.16**	0.15
Brown hair	4	0.49	0.49	**0.57**
Bushy eyebrows	6	0.48	**0.55**	0.49
Chubby	16	**0.49**	0.33	0.45
Double chin	20	0.34	0.36	**0.46**
Eyeglasses	14	0.64	**0.81**	0.79
Goatee	15	0.41	0.48	**0.50**
Gray hair	23	0.46	0.55	**0.59**
Heavy makeup	2	0.80	**0.84**	**0.84**
High cheekbones	1	0.78	0.79	**0.80**
Male	1	0.91	**0.93**	**0.93**
Mouth slightly open	1	0.81	**0.83**	0.82
Mustache	24	0.36	**0.58**	0.49
Narrow eyes	8	0.17	0.25	**0.28**
No beard	5	0.95	**0.95**	0.92
Oval face	3	0.16	0.24	**0.47**
Pale skin	22	0.34	**0.45**	0.39
Pointy nose	3	0.49	0.31	**0.52**
Receding hairline	11	0.22	**0.46**	0.44
Rosy cheeks	14	0.45	0.53	**0.55**
Shadow	8	0.45	**0.49**	0.46
Sideburns	17	0.50	0.58	**0.60**
Smiling	1	0.85	**0.87**	**0.87**
Straight hair	4	0.30	0.07	**0.38**
Wavy hair	2	0.52	0.50	**0.57**
Wearing earrings	4	0.44	0.48	**0.53**
Wearing hat	19	0.65	0.67	**0.70**
Wearing lipstick	1	**0.88**	**0.88**	**0.88**
Wearing necklace	7	0.04	0.11	**0.35**
Wearing necktie	13	0.62	**0.65**	0.63
Young	4	**0.89**	**0.89**	0.76
Mean	-	0.5152	0.5519	**0.5826**