To provide a more intuitive presentation of the algorithmic principles of the DCBAN model proposed in this thesis, the training and inference processes of DCBAN are presented in pseudocode form, as shown in Algorithm 1. This pseudocode illustrates the complete process, from fMRI data preprocessing and feature extraction to the training of semantic and text prediction models, and finally, to the dynamic confidence adaptive fusion diffusion generation. Through this process, it is clear that the core of DCBAN relies on the organic collaboration of three key modules: Deep Nested Singular Value Decomposition (DeepSVD), Bayesian Adaptive Feature Ridge Regression (BAFRR), and Dynamic Confidence Adaptive Fusion Diffusion (DCAF).

Algorithm 1: DCBAN

Input: Visual image I_{RGB}、raw fMRI data obtained while viewing I_{RGB}、prompt associated with I_{RGB}

Hyperparameters: DeepSVD/BAFRR/DCAF internal hyperparameters (see Table 2. Hyperparameter Settings and instructions)

Output: Reconstructed visual image

Preprocessing of raw fMRI data

1: X ← GLM_denoise_and_beta_average(fMRI raw data)

Construct Training Labels:

2: Semantic features Y_Z are extracted by VAE from I_{RGB}.

3: Text features Y_C are extracted by CLIP from the prompt.

Prediction models for semantic and text features are trained.

for each feature type Y in {Y_Z, Y_C} **do**

Dimension Alignment:

4: Y_{PCA} ← PCA (Y, X.shape[1])

DeepSVD Fine-grained Feature Extraction:

5: λ, V ← DeepSVD(X, Y_{PCA})

Obtain BAFRR Coefficient Solution:

6: δ_{best} ← ARSO(X, Y_{PCA})

7: β ← Train_BAFRR(X, Y, λ, V,)

end for

// After the loop, β_Z and β_C are obtained.

Predict Visual Image Semantic Features:

8: Z← X_{test} β_Z

Predict Visual Image Text Features:

9: C← X_{test} β_C

Dynamic Confidence Adaptive Diffusion to Reconstruct Visual Image:

10: Reconstructed Visual mage← DCAF(Z, C)

Return Reconstructed Visual Image

To provide a more intuitive explanation of the detailed processes involved in the three main modules of DCBAN—DeepSVD, BAFRR, and DCAF—the corresponding pseudocode has been included, as shown in Algorithm 2-4.

Algorithm 2: DeepSVD

Input: fMRI feature X、dimension-reduced visual image features Y_{PCA}。

Hyperparameters: DeepSVD internal hyperparameters (see Table 2. Hyperparameter Settings and instructions)

Output: Highly correlated singular values λ, singular value vectors V

Repeat until early stopping:

    $T_0 = X$

    for i in {1..56} **do**

        $T_i \leftarrow$ 3×3Conv (T_{i-1}; Φ_i)

        f_i, $g_i \leftarrow$ SVD(T_i)

        Compute low-rank constraint loss (see Equation (1))

        $\mathcal{L}_{rank} \leftarrow \mathcal{L}_{rank}$ t(f_i, g_i)

        Compute Mean Squared Error (MSE) (see Equation (3))

        $\mathcal{L}_{MSE} \leftarrow \mathcal{L}_{MSE}$ (T_i, Y_{PCA})

        Global joint loss (see Equation (4))

        $\mathcal{L}_{total} \leftarrow \mathcal{L}_{total}$ ($\mathcal{L}_{rank}$, $\mathcal{L}_{MSE}$)

        Update network parameters (see Equation (5))

        AdamUpdate(Hyperparameters)

    end for

until early_stop

$\mathbf{X}_{\text{lowrank}} \leftarrow$ Fully Connected Layer(T_{56})

Output of Highly Correlated Singular Values (see Equation (6)):

λ ← Calculation_of_strongly_correlated_singular_values($\mathbf{X}_{\text{lowrank}}$)

Output of Highly Correlated Singular Vectors (see Equation (7)):

V ← Calculation_of)singular_vectors ($\mathbf{X}_{\text{lowrank}}$)

Return λ, V

Algorithm 3: BAFRR

Input: fMRI feature X、visual image feature Y、highly correlated singular value λ, singular value vector V

Hyperparameters: BAFRR internal hyperparameters (see Table 2. Hyperparameter Settings and instructions)

Output: Semantic feature Z、text feature C

Grid Definition (Equation 8):

$\Lambda = \{\delta_i \mid \delta_i = \delta_{\min} + i \cdot \Delta\delta, i = 0,1,2\ldots,N\}$

Use GP surrogate model + Expected Improvement (EI) as the acquisition function for L iterations

Store (α,mse)

Obs = []

for δ_i in Λ **do**:

  Cross-validation evaluation

  for i in {1，…，k} **do**:

    (X_train, Y_train, X_val, Y_val) = SplitKFold(X, Y, i, k)

    β_{δ_i} = BAFRR(Y_train, λ, δ_i)

    Y_pred = X_val @ β_{δ_i}

    mse = KFoldMSE(Y_val, Y_pred)

    Obs.append((δ_i,α, mse))

  end for

end for

Optimal Score Selection:

δ_{best} = min(Obs, key=lambda item: item[2])

def BAFRR (Y, λ, V, δ):

  Preprocess to singular value space

  $\tilde{Y} = V^T Y$

  Unregularized solution (OLS) is solved in singular value space

  $\tilde{\beta}_{OLS} = S^{-2}(\lambda \odot \tilde{Y})$

  Generate search grid A

  $A = (\alpha_k \mid \alpha_k = 10^{\log_{10}(SMALL\_BIAS \cdot \sigma_{\min}^2) + k \cdot \Delta A},\ k = 0,1,\ldots,N)$

  Perform Bayesian optimization for L iterations to generate the adaptive grid A_L

  For i in L **do**:

    Fit Gaussian Process (GP)

    GP.fit(α_i)

    Select the next set of α\alpha$\alpha$ with the highest potential based on EI

    $EI(\alpha_i) = (m(\alpha_{best}) - \mathrm{m}(\alpha_i)) \cdot \Phi(Z) + \sigma(\alpha_i) \cdot \phi(Z)$

    A_L =OptimizeGrid (A)

  end for

  Calculate the A_L **of α values and corresponding γ values in P**

$P = \{(\alpha_1, \gamma_1), (\alpha_2, \gamma_2), \ldots, (\alpha_n, \gamma_n)\}$

Compute Ridge Regression target parameters

α_δ = Interpolate (P, δ)

Solve Ridge Regression

$\tilde{\beta}_{RR} = [\lambda^2 / (\lambda^2 + \alpha_\delta)] \odot \widetilde{\beta}_{OLS}$

$\beta = V \tilde{\beta}_{RR}$

Return β

end

β_Z = BAFRR(Y_Z, λ, V, δ_{best})

β_C = BAFRR(Y_C, λ, V, δ_{best})

$Z = X_{\text{test}} \beta_Z$

$C = X_{\text{test}} \beta_C$

Return Z,C

Algorithm 4: DCAF

Input: Decoded semantic features Z、decoded text features C、time step $niter$

Hyperparameters: Internal hyperparameters of DCAF (see Table 2. Hyperparameter Settings and instructions)

Output: Reconstructed visual image

Feature Alignment

Align(C)

Feature Flattening

$Z_1 = Flatten(C)$

Form Bottleneck Structure

$Z_2 = \mathrm{Re}lu(MLP(Z_1))$

Output Confidence Weights

$conf\_weight = \sigma(Z_2)$

Dynamic Confidence Adaptive Fusion Diffusion for Visual Image Reconstruction

for n in range($niter$) **do**:

  $time\_decay(n) = max(0, 1 - n / niter)$

  $\xi^{(n)} = conf\_weight \times time\_decay(n)$

  Dynamically Couple to Generate Dynamic Text Features

  $C_{dynamic}^{(n)} = \xi^{(n)} \cdot C + (1 - \xi^{(n)}) \cdot u_C$

  Reconstruct Visual Image Using Diffusion Model

  Reconstructed Visual Image← diffusion_model(Z, $C_{dynamic}^{(n)}$)

end

Return: Reconstructed Visual Image