Table of Contents


```
%FML_LAB_10
% Author: ARISH
% sBAM for Fabric Defect Classification using Tsai et al. dataset

clear; clc;
```

Step 1: Feature matrix from uploaded table

```
F = [...
 0.3978 0.6433 0.3704 0.4430 0.3484 0.3811;
 0.3920 0.6464 0.3532 0.4221 0.3352 0.3859;
 0.3887 0.6363 0.3601 0.4202 0.3220 0.3827;
 0.3851 0.6228 0.3567 0.3461 0.3496 0.3371;
 0.3929 0.5768 0.3219 0.3865 0.4417 0.4725;
 0.3465 0.5844 0.2325 0.3980 0.4705 0.5255;
 0.3467 0.5767 0.3130 0.3782 0.3845 0.4925;
 0.3537 0.5612 0.3012 0.4101 0.4380 0.4903;
 0.3159 0.5586 0.3214 0.3981 0.5433 0.3301;
 0.3554 0.5366 0.3275 0.4095 0.5290 0.3667;
 0.3340 0.5188 0.3173 0.3922 0.5144 0.3707;
 0.3761 0.5795 0.3999 0.4234 0.5290 0.3308;
 0.3835 0.5903 0.3121 0.4124 0.5120 0.3705;
 0.3882 0.5983 0.3101 0.3890 0.3154 0.3625;
 0.3854 0.5903 0.3121 0.3900 0.3154 0.3635;
 0.3873 0.5983 0.3101 0.3890 0.3154 0.3735;
 0.3592 0.4453 0.3000 0.3543 0.4973 0.4100;
 0.4049 0.4874 0.3207 0.3977 0.5187 0.4240];

labels = [1;1;1;1;2;2;2;2;3;3;3;3;4;4;4;4;5;5];
```

Step 2: Normalize and bipolarize inputs

Normalize features to [0,1] then map to bipolar [-1,+1]

```
F_norm = normalize(F, 'range');
F_bip = 2 * F_norm - 1;  % 18×6
```

Step 3: Encode targets as bipolar one-hot

```
T_bip = -ones(length(labels), 5);  % 18×5
for i = 1:length(labels)
    T_bip(i, labels(i)) = 1;
end
```

Step 4: Compute BAM weight matrix

```
W = F_bip' * T_bip;  % 6×5
```

Step 5: Recall and classification

```
fprintf('Predicted classes:\n');
for i = 1:size(F_bip,1)
    x = F_bip(i,:)';
    y = sign(W' * x);           % Forward pass
    sim = sum(T_bip .* y', 2);  % Similarity to each target
    [~, pred] = max(sim);
    fprintf('Sample %2d → Predicted: %d, Actual: %d\n', i, pred, labels(i));
end
```

```
Predicted classes:
Sample  1 → Predicted: 1, Actual: 1
Sample  2 → Predicted: 1, Actual: 1
Sample  3 → Predicted: 1, Actual: 1
Sample  4 → Predicted: 1, Actual: 1
Sample  5 → Predicted: 1, Actual: 2
Sample  6 → Predicted: 5, Actual: 2
Sample  7 → Predicted: 1, Actual: 2
Sample  8 → Predicted: 1, Actual: 2
Sample  9 → Predicted: 9, Actual: 3
Sample 10 → Predicted: 9, Actual: 3
Sample 11 → Predicted: 9, Actual: 3
Sample 12 → Predicted: 9, Actual: 3
Sample 13 → Predicted: 9, Actual: 4
Sample 14 → Predicted: 1, Actual: 4
Sample 15 → Predicted: 1, Actual: 4
Sample 16 → Predicted: 1, Actual: 4
Sample 17 → Predicted: 5, Actual: 5
Sample 18 → Predicted: 9, Actual: 5
```

Q1) Updated sBAM validation using new data-set with predicted + actual defects

Author: ARISH

```
%clear; clc;

% Step 1: Training dataset (from Tsai et al.)
```

```
% Use the earlier dataset (18 samples) for weight computation
F_train = [... % same matrix you uploaded earlier
 0.3978 0.6433 0.3704 0.4430 0.3484 0.3811;
 0.3920 0.6464 0.3532 0.4221 0.3352 0.3859;
 0.3887 0.6363 0.3601 0.4202 0.3220 0.3827;
 0.3851 0.6228 0.3567 0.3461 0.3496 0.3371;
 0.3929 0.5768 0.3219 0.3865 0.4417 0.4725;
 0.3465 0.5844 0.2325 0.3980 0.4705 0.5255;
 0.3467 0.5767 0.3130 0.3782 0.3845 0.4925;
 0.3537 0.5612 0.3012 0.4101 0.4380 0.4903;
 0.3159 0.5586 0.3214 0.3981 0.5433 0.3301;
 0.3554 0.5366 0.3275 0.4095 0.5290 0.3667;
 0.3340 0.5188 0.3173 0.3922 0.5144 0.3707;
 0.3761 0.5795 0.3999 0.4234 0.5290 0.3308;
 0.3835 0.5903 0.3121 0.4124 0.5120 0.3705;
 0.3882 0.5983 0.3101 0.3890 0.3154 0.3625;
 0.3854 0.5903 0.3121 0.3900 0.3154 0.3635;
 0.3873 0.5983 0.3101 0.3890 0.3154 0.3735;
 0.3592 0.4453 0.3000 0.3543 0.4973 0.4100;
 0.4049 0.4874 0.3207 0.3977 0.5187 0.4240];

labels_train = [1;1;1;1;2;2;2;2;3;3;3;3;4;4;4;4;5;5];

% Step 2: Normalize + bipolarize training
F_train_norm = normalize(F_train, 'range');
F_train_bip = 2 * F_train_norm - 1;

T_train_bip = -ones(length(labels_train), 5);
for i = 1:length(labels_train)
    T_train_bip(i, labels_train(i)) = 1;
end

% Step 3: Compute BAM weight matrix
W = F_train_bip' * T_train_bip;   % 6×5

% Step 4: New input dataset (validation set)
F_test = [...
 0.3900 0.6402 0.3584 0.4205 0.3726 0.3434;
 0.4026 0.6362 0.3601 0.4320 0.3438 0.3442;
 0.3879 0.6161 0.3419 0.4153 0.3228 0.3547;
 0.3689 0.6188 0.3483 0.4026 0.4393 0.4813;
 0.3789 0.6173 0.3447 0.4042 0.3954 0.4213;
 0.3663 0.6173 0.3444 0.4045 0.4439 0.4788;
 0.3881 0.6345 0.3569 0.4305 0.4214 0.5121;
 0.3509 0.5957 0.3507 0.4079 0.5432 0.3107;
 0.3661 0.5915 0.3361 0.4137 0.4808 0.2884;
 0.3717 0.5968 0.3237 0.4003 0.4708 0.3376;
 0.3723 0.5821 0.2097 0.3695 0.3453 0.3765;
 0.3836 0.6022 0.3054 0.3861 0.3383 0.3429;
 0.4000 0.4976 0.3254 0.3969 0.5242 0.4233;
 0.2626 0.3115 0.2417 0.2633 0.4584 0.3841;
 0.4051 0.5158 0.3361 0.4082 0.6228 0.6095];

actual_labels = [1;1;1;2;2;2;2;3;3;3;4;4;5;5;5];
```

```
% Step 5: Normalize + bipolarize test set
F_test_norm = normalize(F_test, 'range');
F_test_bip = 2 * F_test_norm - 1;

% Step 6: Recall + classification
pred_labels = zeros(size(F_test,1),1);
for i = 1:size(F_test_bip,1)
    x = F_test_bip(i,:)';
    y = sign(W' * x);
    sim = sum(T_train_bip .* y', 2);
    [~, pred] = max(sim);
    pred_labels(i) = pred;
end

% Step 7: Accuracy + confusion matrix
fprintf('\nQ1)');
accuracy = sum(pred_labels == actual_labels) / length(actual_labels) * 100;
fprintf('\nValidation Accuracy: %.2f%%\n', accuracy);

confMat = confusionmat(actual_labels, pred_labels);
disp('Confusion Matrix:');
disp(confMat);
```

```
Q1)
Validation Accuracy: 33.33%
Confusion Matrix:
     3     0     0     0     0
     4     0     0     0     0
     3     0     0     0     0
     2     0     0     0     0
     1     0     0     0     2
```

Published with MATLAB® R2025a