Algorithm 1 Iterative Meta-Rule Set Refinement

input Initial Meta-Rule Set R
output Finalized Meta-Rule Set R'

```
1:  R' ← R
2:  while R' not converge do
3:    for each r ∈ R ∪ R' do
4:      if r is well-balanced then
5:        if ∄r* ∈ R' similar to r then
6:          /* Add Rules */
7:          R' ← R' ∪ { r }
8:        else
9:          /* Merge Rules */
10:         R' ← R' ∪ {r ⊕ r*} \ {r, r*}
11:       end if
12:     else
13:       /* Prune Rules */
14:       R' ← R' \ { r }
15:     end if
16:   end for
17: end while
18: return R'
```

Iterative meta-rule set refinement To systematically perform the refinement, we define three operations that G can use to update the meta-rule set R. (1) ADD: $R \cup \{r\} \rightarrow R$. This operation instructs G to directly add the rule $r \in R_i$ derived from the training sample (x_i, y_i) to the meta-rule set R. This step is still necessary as MERGE and PRUNE (described below) can be very aggressive and some useful rules have been inadvertently removed. (2) MERGE: $R \cup \{r_i \oplus r_j\} \setminus \{r_i, r_j\} \rightarrow R$. This operation instructs G to merge two rules $r_i, r_j \in R$ by replacing them with an updated rule $r_i \oplus r_j$. (3) PRUNE: $R \setminus \{r\} \rightarrow R$. This operation instructs G to remove the rule r from the meta-rule set R.

Algorithm 1 shows the procedures of updating the editing rule set R. Before adding each r (ADD and MERGE) to R, we prompt G to assess whether it is a balanced rule, i.e., neither too generic nor too specific (line 4). If not, we apply PRUNE to discard it (line 14). Otherwise, we prompt G again to decide whether we should ADD or MERGE r (line 5). Specifically, G will be prompted to decide whether there exists a rule in R similar or identical to r. We apply MERGE (line 10) if yes and ADD (line 7) if not.