Classification by Components Including Chow's Reject Option

Mehrdad Mohannazadeh Bakhtiari[1,2(✉)] and Thomas Villmann[1,2(✉)]

[1] Saxon Institute for Computer Intelligence and Machine Learning, Mittweida, Germany
[2] University of Applied Sciences Mittweida, Mittweida, Germany
{mmohanna,thomas.villmann}@hs-mittweida.de.de

Abstract. In this paper, we present an approach how reject options are integrated into Classification-by-Component networks. Classification-by-Component networks are relatively accurate classifiers that offer a fair interpretability. Yet, their performance can be increased by allowing rejection of an uncertain classification. We will modify the original Classification-by-Component model so that the adaptive parameters adapt, taking reject options into account.

Keywords: Classification-by-Components · Reject Option · Neural Networks · Probabilistic Classifier

1 Introduction

Classification by Components (CbC) networks [1] are robust interpretable classifiers with high performance. Nonetheless, reject options [2] can increase the performance of CbC, when a wrong classification costs considerably. Therefore, in this paper, we investigate the details of reject options for CbC networks. There has been much effort gone into modifying reject options since the original work of Chow. However, most of the papers in the field are not directly relatable or applicable to CbC networks. Here, we mention a few of such work. A paper by Musavishavazi et al. [3] has a comparable mathematical formulation to this paper. Nonetheless, it cannot be immediately used for CbC, because of the contrastive CbC objective function. In another paper, Y. Geifman and R. EL-Yaniv [4] designed a selective classifier, that is more suitable for pre-trained deep neural networks. A similar approach is RISAN [5], that has more similar reject-classification formulation to our work in this paper.

Reject options can be introduced for classification systems, in which a wrong decision can lead to catastrophic outcomes [2,6]. In case a classifier is not sure about the class of an input, it can refuse labelling it at a lower cost. Originally, Chow [6] defined reject options and the optimal decision strategy for an optimum system [2] were determined. The labels, for each data, is crisp and the conditional

M.M.B. is Supported by ESP.

M. Tanveer et al. (Eds.): ICONIP 2022, CCIS 1791, pp. 586–596, 2023.
https://doi.org/10.1007/978-981-99-1639-9_49

probabilities are estimated based on a Bayesian estimator. Note that Chow's reject option is originally applied to a previously trained model. Later, variations of reject options emerged [7,8] as the proposed classifiers intended to train the adaptable parameters, while reject options are taken into account. We would like to adopt this approach, in this paper, such that we use Chow's optimal threshold as our threshold. However, the model that estimates conditional probabilities for prediction will be slightly different and, consequently, the prediction probabilities will be different. Also, the Error-Reject curve, as described by Chow [2], will be different for CbC with reject option. Villmann et al. [9] introduced a general loss function that reaches Chow's optimal threshold. The loss function, which is a weighted sum of error and rejection rates, is the inspiration of this paper.

The paper is structured as follows: In Sect. 2, CbC networks are introduced. Section 3 provides a background about reject options. In Sect. 4, reject options are investigated for a classifier, with contrastive loss. Section 5, contains the gradients, with respect to adaptable parameters, for the sake of comparison with the original CbC. The experiments and simulations are in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 The Original CbC

A brief description of CbC model [1] is given in this section. We define components as a set $\mathcal{K} = \{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ existing in data space $X \subset \mathbb{R}^m$. Also, a set $C = \{1, 2, \ldots, c\}$ of data classes is defined. Detection probability function $d_i : X \to [0, 1]$ determines the probability to detect the component $\mathcal{K}_i$ in data point $\mathbb{x}$. Hence, $d_i(\mathbb{x}) = 0$ means the full absence of component $\mathcal{K}_i$ in $\mathbb{x}$ and $d_i(\mathbb{x}) = 1$ means the full presence. For a data point $\mathbb{x}$, we define the detection vector

$$\mathbb{d}(\mathbb{x}) = [d_1(\mathbb{x}), \ldots, d_k(\mathbb{x})]^T \tag{1}$$

To study the effect of components, the reasoning quantity $r_{ij}^{+} \in (0, 1)$ is introduced as the probability that the component $\mathcal{K}_i$ is important and must be detected to support the class hypothesis j according to Biederman's cognitive model [10]. Additionally, negative reasoning is considered in CbC by a reasoning quantity $r_{ij}^{-} \in (0, 1)$, taken as the probability that the component $\mathcal{K}_i$ is important and must not be detected to support the class hypothesis j. Finally, an indefinite reasoning is introduced: If the presence of the component $\mathcal{K}_i$ reveals no information about the occurrence of class j, then component $\mathcal{K}_i$ has a non-vanishing neutral (indefinite) reasoning over class j, i.e. $r_{ij}^{0} \gg 0$ is interpreted as the probability that component $\mathcal{K}_i$ is not important for class hypothesis j. We assume that the three quantities satisfy the restriction $r_{ij}^{+} + r_{ij}^{-} + r_{ij}^{0} = 1$.

Given a classification problem, i.e. a training set $T = \{(\mathbb{x}_b, y_b)\}_{b=1}^{\tau}, \mathbb{x}_b \in X \subset \mathbb{R}^m, y_b \in C$, we consider a set of trainable components $\mathcal{K}_i \in \mathcal{K}$ together with adjustable reasoning parameters $r_{ij}^{+}, r_{ij}^{-}, r_{ij}^{0}$. According to [1], CbC is to minimize the contrastive loss

$$l(\mathbb{x}, y) = \phi\big(max\big\{p_j(\mathbb{x}) | j \neq y, j \in C\big\} - p_y(\mathbb{x})\big) \tag{2}$$

where $(\mathbb{x}, y) \in T$ and $\phi : [-1, 1] \rightarrow \mathbb{R}$ is a monotonically increasing function. The class hypothesis possibilities $\hat{p}_j(\mathbb{x})$ are defined as

$$\hat{p}_j(\mathbb{x}) = \frac{\sum_i r_{ij}^{+} \cdot d_i(\mathbb{x}) + r_{ij}^{-} \cdot (1 - d_i(\mathbb{x}))}{\sum_i (1 - r_{ij}^{0})} \tag{3}$$

The possibilistic vector is normalised to have class probabilities

$$p_j(\mathbb{x}) = \frac{\hat{p}_j(\mathbb{x})}{\sum_i \hat{p}_i(\mathbb{x})} \tag{4}$$

In original CbC, optimal components are found using a Convolutional Neural Network (CNN) feature extractor [11] and the reasoning parameters are found, using Stochastic Gradient Descent (SGD).

3 Reject Options for a Probabilistic Classifier

In this section, we introduce the original reject options [2], that is applied to a probabilistic classifier, after training phase. Then, we introduce an empirical loss function, such that its optimal threshold is the same as the optimal threshold of the original reject options. Optimizing the empirical loss function allows us to train CbC networks, while reject options are taken into account.

The original reject options is as follows. We assume training set $T = \{(\mathbb{x}_b, y_b)\}_{b=1}^{\tau}$, as defined in previous section. A probabilistic classifier $p_i(\mathbb{x})$ is trained to classify a new input $\mathbb{x}$. Note that

$$\sum_i p_i(\mathbb{x}) = 1 \ \forall \mathbb{x} \tag{5}$$

Chow [2] defined the decision-rejection rule, given a threshold t: $\mathbb{x}$ is rejected to be classified if

$$m(\mathbb{x}) < 1 - t \tag{6}$$

where $m(\mathbb{x}) = max_i\{p_i(\mathbb{x})\}$. Otherwise, $\mathbb{x}$ is accepted to be classified. The predicted class of the data $\mathbb{x}$ is

$$C(\mathbb{x}) = argmax_i\{p_i(\mathbb{x})\} \tag{7}$$

Given c_r, c_e, and c_c are costs of rejecting a data, miss-classifying a data, and correctly classifying a data, respectively, Chow [2] derived the optimal threshold to be $t^* = \frac{c_r - c_c}{c_e - c_c}$. Note that the order $c_c < c_r < c_e$ is an important assumption. Here, we assume $c_c = 0$ to have the threshold

$$t^* = \frac{c_r}{c_e} \tag{8}$$

Now, we define the general empirical loss function for a probabilistic classifier with reject option. Inspired by [9], the general loss function, to be optimised, is defined as

$$L(t) = c_e \cdot E(t) + c_r \cdot R(t) \tag{9}$$

where $E(t)$ and $R(t)$ are error rate and reject rate, respectively and they are both functions of the threshold t. It can be proven that, given the decisions are Bayesian optimal, the optimal threshold of (9) is (8). A proof is provided in appendix A.

The empirical version of (9), that is more suitable for learning, is defined as below.

$$L(t) = \frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_e \cdot H\left(m\left(\mathbb{x}\right) + t - 1\right) \cdot \left(1 - \delta\left(C(\mathbb{x}), y\right)\right) + \frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_r \cdot H\left(-m\left(\mathbb{x}\right) - t + 1\right) \tag{10}$$

$\delta(i,j)$, for $i,j \in C$, is the Kronecker delta function and $H(x)$, for $x \in \mathbb{R}$, is the Heaviside function. Note that the argument of Heaviside functions, in (10), are taken from inequality (6). Therefore, $H\left(m\left(\mathbb{x}\right) + t - 1\right)$ is equal to one, when data is accepted to be classified. Also, $1 - \delta\left(C(\mathbb{x}), y\right)$ is a miss-classification indicator. If the prediction and the true label matched for a data point, then $1 - \delta\left(C(\mathbb{x}), y\right)$ would be 0, and 1 otherwise. Similar approach is taken by Shah and Manwani [12], when combining classification and rejection.

Finally, we plug the optimal threshold (8) into (10) to have

$$L = \frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_e \cdot H\left(m\left(\mathbb{x}\right) + t^* - 1\right) \cdot \left(1 - \delta\left(C(\mathbb{x}), y\right)\right) + \frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_r \cdot H\left(-m\left(\mathbb{x}\right) - t^* + 1\right) \tag{11}$$

4 Reject Options for CbC Networks

We still need to modify (11), so the loss function of CbC is involved. CbC uses the contrastive loss function

$$l(\mathbb{x}, y) = \phi\left(\Delta p(\mathbb{x}, y)\right) \tag{12}$$

where

$$\Delta p(\mathbb{x}, y) = p_s(\mathbb{x}, y) - p_y\left(\mathbb{x}\right) \tag{13}$$

and

$$p_s(\mathbb{x}, y) = max\{p_j\left(\mathbb{x}\right) | j \neq y,\ j \in C\} \tag{14}$$

We would like to define function ϕ more accurately, so we can use it as miss-classification indication $1 - \delta\left(C(\mathbb{x}), y\right)$, that appears in (11). We choose the Sigmoid function as below.

$$\phi(x) \equiv \phi_\lambda(x) = \frac{1}{1 + \exp(-\frac{x}{\lambda})} \tag{15}$$

Now, the term $1 - \delta(C(\mathbb{x}, y)$, in (11), can be substituted by $\phi_\lambda(\Delta p(\mathbb{x}, y))$. Note that the function $\phi_\lambda(\Delta p(\mathbb{x}, y))$ goes to zero, given a confident correct classification and it becomes 1, when we have miss-classified an input. In short, the Sigmoid function serves as a smooth indicator function, when λ goes to 0.

Now, we approximate the Heaviside function, appearing in (11), with another Sigmoid function, so we can derive required gradients later.

$$H(x) \equiv H_\gamma(x) = \frac{1}{1 + \exp(-\frac{x}{\gamma})} \tag{16}$$

In the above function, we achieve better approximation of Heaviside function as γ goes to 0.

The new loss function for CbC, with reject options, is achieved when we integrate the modifications (15) and (16) into (11) as follows.

$$\begin{aligned} L = &\frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_e \cdot H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right) \cdot \phi_\lambda(\Delta p(\mathbb{x}, y)) \\ &+ \frac{1}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} c_r \cdot H_\gamma \left(-m\left(\mathbb{x}\right) - t^* + 1\right) \end{aligned} \tag{17}$$

It can be shown that

$$H_\gamma\left(-x\right) = 1 - H_\gamma\left(x\right) \tag{18}$$

Therefore, we can further manipulate the loss function to have

$$L = \frac{c_e}{|T|} \cdot \sum_{(\mathbb{x},y)\in T} H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right) \cdot \left(\phi_\lambda(\Delta p(\mathbb{x}, y)) - \frac{c_r}{c_e}\right) \tag{19}$$

Recalling (8), we substitute $\frac{c_r}{c_e}$ with t^* in the above equation. Also, the multiplier $\frac{c_e}{|T|}$, in the above equation, does not change the optimum of the function and, hence, it is dropped as well. So, we get the final loss for CbC, when reject options are taken into account.

$$L = \sum_{(\mathbb{x},y)\in T} H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right) \cdot \left(\phi_\lambda(\Delta p(\mathbb{x}, y)) - t^*\right) \tag{20}$$

The loss function (20) has the term $\phi_\lambda(\Delta p(\mathbb{x}, y)) - t^*$, that is a shifted version (by a constant t^*) of CbC loss (12). However, reject option plays role as a multiplier this time, with the term $H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right)$. The loss function (20) is used to train the CbC network, using SGD. Since the optimization method of interest is stochastic gradient descent, we further define the local loss $\bar{L} = \bar{L}(\mathbb{x}, y)$, associated with training pair $(\mathbb{x}, y)$, to be a single term of the summation (20). In other words $\bar{L}(\mathbb{x}, y) = H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right) \cdot \left(\phi_\lambda(\Delta p(\mathbb{x}, y)) - t^*\right)$. Assuming that the arguments of the local loss $\bar{L}(\mathbb{x}, y)$ are known, we may drop the arguments and simply denote the local loss as $\bar{L}$.

5 Derivation and Comparison of Gradients

In this section, we first mention the gradients of original CbC, with respect to the reasoning parameters. Then, the gradients of CbC, with reject option, with respect to the same parameters are included. We would like to mark the differences between the adaptation rules of the methods.

For CbC, we take ϕ to be (15). Then, the gradients of CbC loss function (12) , with respect to general adaptive parameter r, is as below.

$$\frac{\partial l}{\partial r} = \Psi(\mathbb{x}, y, \lambda) \cdot \frac{\partial \Delta p(\mathbb{x}, y)}{\partial r} \tag{21}$$

with

$$\Psi(\mathbb{x}, y, \lambda) = \frac{\partial l}{\partial \Delta p(\mathbb{x}, y)} = \frac{1}{\lambda} \cdot \phi_\lambda \left(\Delta p \left(\mathbb{x}, y\right)\right) \cdot \left(1 - \phi_\lambda \left(\Delta p \left(\mathbb{x}, y\right)\right)\right) \tag{22}$$

The partial derivative in (21) is

$$\frac{\partial \Delta p(\mathbb{x}, y)}{\partial r} = \frac{\partial p_s(\mathbb{x}, y)}{\partial r} - \frac{\partial \Delta p_y(\mathbb{x})}{\partial r} \tag{23}$$

Based on the above gradients, in each learning step with training pair $(\mathbb{x}, y)$, a parameter is updated to raise the probability $p_y(\mathbb{x})$ and lower the probability $p_s(\mathbb{x}, y)$.

Now, the gradients of a local loss of the main loss function (20) for CbC, with reject option, are found.

$$\frac{\partial \bar{L}}{\partial r} = H_\gamma(\mathbb{x}, t^*) \cdot \Psi \tag{24}$$

with

$$\Psi = \frac{1}{\gamma} \cdot \left(1 - H_\gamma(\mathbb{x}, t^*)\right) \cdot \left(\phi_\lambda(\mathbb{x}, y) - t^*\right) \cdot \frac{\partial m(\mathbb{x})}{\partial r} + \frac{\partial \phi_\lambda(\mathbb{x}, y)}{\partial r} \tag{25}$$

Also, we have made the following shorthand notations in (24) and (25) .

$$H_\gamma(\mathbb{x}, t^*) = H_\gamma \left(m\left(\mathbb{x}\right) + t^* - 1\right) \tag{26}$$

and

$$\phi_\lambda(\mathbb{x}, y) = \phi_\lambda(\Delta p(\mathbb{x}, y)) \tag{27}$$

Note that the term $\frac{\partial \phi_\lambda(\mathbb{x}, y)}{\partial r} = \frac{\partial l}{\partial r}$, appearing in (25), is basically the original CbC gradient (21).

In (24), if $H_\gamma(\mathbb{x}, t^*) \approx 1$, meaning data $\mathbb{x}$ is accepted to be classified, then the gradient (24) becomes

$$\frac{\partial \bar{L}}{\partial r} \approx \frac{\partial l}{\partial r} \approx \frac{\partial \phi_\lambda(\mathbb{x}, y)}{\partial r} \tag{28}$$

which is simply the gradient of original CbC that is mentioned in (21).

If $H_\gamma(\mathbb{x}, t^*) \approx 0$, then the gradient (24) obviously vanishes. This means that we prefer to keep uncertain data points in rejection area to keep the overall risk low. However, if $0 \ll H_\gamma(\mathbb{x}, t^*) \ll 1$, that happens when $\mathbb{x}$ is close to rejection border (as defined by Chow [2]), then the term $\phi_\lambda(\mathbb{x}, y) - t^*$, in (25), plays an important role. The term $\phi_\lambda(\mathbb{x}, y) - t^*$ basically makes a certain balance between error rate and rejection rate, since the sign of the coefficient of $\frac{\partial m(\mathbb{x})}{\partial r}$ in (25) depends on the term. We suppose the case that $\mathbb{x}$ is correctly classified, which means $m(\mathbb{x}) = p_y(\mathbb{x})$. Then, $\frac{\partial m(\mathbb{x})}{\partial r} = \frac{\partial p_y(\mathbb{x})}{\partial r}$ and the gradient (24) becomes

$$\frac{\partial \bar{L}}{\partial r} = H_\gamma(\mathbb{x}, t^*) \cdot \left[\beta \cdot \frac{\partial p_y(\mathbb{x})}{\partial r} + \frac{\partial \phi_\lambda(\mathbb{x}, y)}{\partial r} \right] \tag{29}$$

where $\beta = \frac{1}{\gamma} \cdot (1 - H_\gamma(\mathbb{x}, t^*)) \cdot (\phi_\lambda(\mathbb{x}, y) - t^*)$.

Since the term $\frac{\partial p_y(\mathbb{x})}{\partial r}$ exists in the derivative $\frac{\partial \phi_\lambda(\mathbb{x}, y)}{\partial r}$ (23), with a negative coefficient, the magnitude of coefficient of $\frac{\partial p_y(\mathbb{x})}{\partial r}$ becomes even larger in (29), if $\phi_\lambda(\mathbb{x}, y) - t^* < 0$. This happens when t^* is relatively large, meaning that cost of misclassification is low or comparable to cost of rejection. In this case the algorithm is less cautious and alter the parameters with larger rates. On the other hand, if t^* is relatively small (cost of misclassification is higher than cost of rejection), then we potentially have $\phi_\lambda(\mathbb{x}, y) - t^* > 0$ and, consequently, the coefficient of $\frac{\partial p_y(\mathbb{x})}{\partial r}$ in (29) becomes small, since we fear misclassifying a correctly classified input.

This section is summarized as follows for gradients of CbC, with reject options. In the region that data is accepted to be classified, the method roughly applies the original CbC rules to data points. Also, if a data is firmly rejected, it stays in rejection region with a high probability. Close to the border of rejection and accepting, the method carefully looks for the best performance, based on the term $\phi_\lambda(\mathbb{x}, y) - t^*$, that shows the compromise between error and rejection.

6 Experiments and Simulations

For experiments, the "Two Moons" data set (TMDS) serves to visualize the difference between components of the original CbC and CbC with reject options. Further, the MNIST data set is considered that has been used in the original CbC [1] for comparison.

Particularly, a sample of TMDS with added Gaussian noise of standard deviation 0.1, is considered. We suppose 4 components $\mathbb{w}_t$, $t \in \{1, \ldots, 4\}$ in the 2d-data space initialized as random data points from the training set. The detection functions, for a component t, is defined as the Gaussian $d_t(\mathbb{x}) = \exp(-2 \cdot ||\mathbb{x} - \mathbb{w}_t||^2)$. The parameter λ, in (15), was set to 0.01. In all experiments of this paper, the training data is split into training and test data and a 6-fold cross validation, with 3×10^3 learning steps, is used. After applying original CbC, we report the accuracy $(95 \pm 0.05)\%$ for the test data. The learnt components, as well as the corresponding reasoning parameters, are depicted in the first column of Fig. 1.

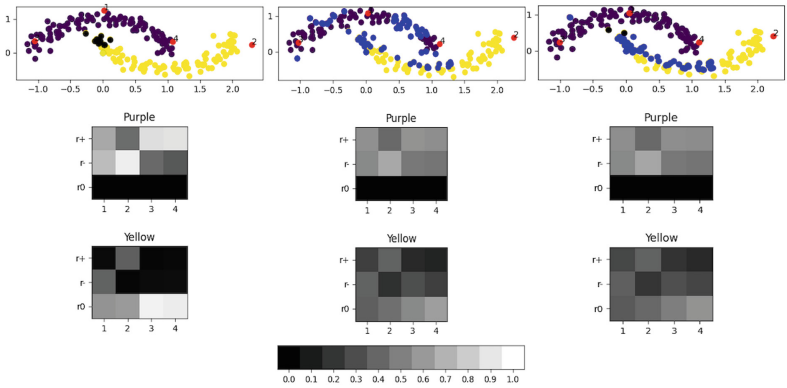

Fig. 1. Visualization of Two Moons. First, second, and third columns, are for normal CbC, CbC with reject option ($t^* = 0.49$), and CbC with reject option ($t^* = 0.495$), respectively. In each row, the top image shows the location of learnt components in red. The black points represent the misclassified data points. Blue points represent rejected samples. The second and third images, from top, depicts reasoning parameters of Purple and Yellow classes, respectively. The last row shows a gray-scale map for reading values. (Color figure online)

Then, we applied CbC, with reject options, to the same problem. The initialization was same as the original CbC. Theoretically, γ is required to be close to 0. In practice, however, a small γ makes a large fraction $\frac{1}{\gamma}$, that appears in the coefficient of the gradient (24). This reduces the effect of the gradient $\frac{\partial \phi_\lambda(\varkappa,y)}{\partial r}$, in (24), that is responsible for making the predictions accurate. Therefore, we need to slowly reduce γ during learning. The simulations showed that, for both TMDS and MNIST problems, $\gamma = \frac{1}{l}$ (l is the learning step) makes a suitable starting value and the final value will be desirably small. We report a higher accuracy of $(99.9 \pm 0.01)\%$ and the rejection rate $(23 \pm 0.02)\%$, when $t^* = \frac{c_r}{c_e} = 0.49$. The learnt components, as well as the corresponding reasoning parameters, are depicted in the second column of Fig. 1. A careful balancing between reject and misclassification costs is needed. Therefore, we increased reject cost to have $t^* = \frac{\hat{c_r}}{\hat{c_e}} = 0.495$. The accuracy is $(97 \pm 0.02)\%$, while the rejection rate is $(21 \pm 0.04)\%$. See the third column of Fig. 1.

Original CbC was applied to MNIST [1], without feature extraction to achieve 89.5% accuracy. We chose $\lambda = 0.01$ (15) and $\gamma = \frac{1}{l}$, for ϕ_γ. 10 prototypes are initialized, at the center of each class, in the feature space. The data points, as well as the prototypes, are kept normalized at all time. The detection function is $d_t(\varkappa) = max(\langle \varkappa, \wp_t \rangle + b, 0)$, where $\langle ., . \rangle$ indicates the inner product and b is a margin parameter. We have taken $b = 0.3$ by trial and error. The error and rejection rate for $t^* = \frac{c_r}{c_e} = 0.9$ are $(23 \pm 0.1)\%$ and $(0.1 \pm 0.11)\%$, respectively. For $t^* = \frac{\hat{c_r}}{\hat{c_e}} = 0.89$, we have a careful balance between the error and rejection costs. The error and rejection rates are $(19 \pm 0.09)\%$ and $(36 \pm 0.1)\%$ respectively. For case $t^* = 0.89$, we have depicted the learnt components and the reasoning

parameters in Fig. 2. Then, we reduced the number of prototypes to 9 and initiated the prototypes at the center of the first 9 classes (class 0 to 8). The rest of the setting is the same as the case with 10 prototypes. The error and rejection rate for $t^* = \frac{c_r}{c_e} = 0.9$ are $(31 \pm 0.1)\%$ and $(0.1 \pm 0.1)\%$, respectively. For $t^* = \frac{\hat{c_r}}{\hat{c_e}} = 0.88$, the error and rejection rates considerably changed to $(0.2 \pm 0.1)\%$ and $(69 \pm 0.09)\%$ respectively. This relatively high rejection rate suggests that using feature extractors [1] for components might be necessary, when there are 9 or less components.

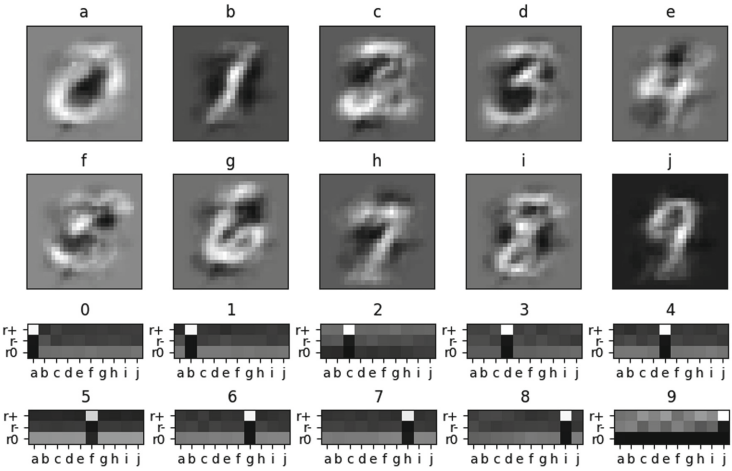

Fig. 2. Visualization of CbC, with reject options ($t^* = \frac{\hat{c_r}}{\hat{c_e}} = 0.89$) and 10 components, applied to MNIST. The top two rows show the components, denoted by letters $\{a, b, c, d, e, f, g, h, i, j\}$. The bottom two rows show the reasoning parameters for each class $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Each class is denoted by its digit.

7 Conclusion

In this paper, we integrated reject options in CbC networks. The new method keeps Chow's optimal threshold, but let the system know about reject options for better adaptation. We have improved the performance of CbC in a scenario that false classification is costlier than rejection. The method was examined using a toy and MNIST data sets.

A Proof of Optimality of Chow's Threshold

Here , we prove $t = \frac{c_r}{c_e}$ is the optimal solution for the function

$$L(\varkappa, y, t) = c_e \cdot E(t) + c_r \cdot R(t)$$

The derivative of the above function, with respect to t, is

$$\frac{\partial}{\partial t}L(\mathbb{x}, y, t) = c_e \cdot \frac{\partial E(t)}{\partial t} + c_r \cdot \frac{\partial R(t)}{\partial t}$$

or

$$\frac{\partial}{\partial t}L(\mathbb{x}, y, t) = c_e \cdot \frac{\partial E(t)}{\partial R(t)} \cdot \frac{\partial R(t)}{\partial t} + c_r \cdot \frac{\partial R(t)}{\partial t} \tag{30}$$

Section *IV* of [2] contains a relation between $E(t)$ and $R(T)$, given a Bayes optimal decision system, in form of Riemann-Stieltjes integral. The relation is

$$E(t) = -\int_{t=0}^{t} \tau \cdot dR(\tau)$$

and the differential form is

$$\frac{\partial E(t)}{\partial R(t)} = -t$$

We use the above result in (30) to have the following.

$$\frac{\partial}{\partial t}L(\mathbb{x}, y, t) = -c_e \cdot t \cdot \frac{\partial R(t)}{\partial t} + c_r \cdot \frac{\partial R(t)}{\partial t}$$

The above derivative it set to 0.

$$\frac{\partial R(t)}{\partial t} \cdot (-c_e \cdot t + c_r) = 0$$

Then, we solve for t and the result is $t^* = \frac{c_r}{c_e}$.

References

1. Saralajew, S., Holdijk, L., Rees, M., Asan, E., Villmann, T.: Classification-by-components: probabilistic modeling of reasoning over a set of components. Adv. Neural Inf. Process. Syst. **32** (2019)
2. Chow, C.K.: On optimum recognition error and reject tradeoff. IEEE Trans. Inf. Theor. **16**(1), 41–46 (1970)
3. Musavishavazi, S., Bakhtiari, M., Villmann, T.: A mathematical model for optimum error-reject trade-off for learning of secure classification models in the presence of label noise during training. In: International Conference on Artificial Intelligence and Soft Computing, pp. 547–554 (2020)
4. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
5. Kalra, B., Shah, K., Manwani, N.: RISAN: robust instance specific deep abstention network. In: Uncertainty in Artificial Intelligence, pp. 1525–1534 (2021)
6. Chow, C.K.: An optimum character recognition system using decision functions. IRE Trans. Electron. Comput. **4**, 247–254 (1957)
7. Villmann, T., et al.: Self-adjusting reject options in prototype based classification. In: Advances in Self-Organizing Maps and Learning Vector Quantization, pp. 269–279 (2016)

8. Fischer, L., Nebel, D., Villmann, T., Hammer, B., Wersing, H.: Rejection strategies for learning vector quantization-a comparison of probabilistic and deterministic approaches pp. 109–118 (2014)
9. Villmann, T., Kaden, M., Nebel, D., Biehl, M.,: Learning vector quantization with adaptive cost-based outlier-rejection. In: International Conference on Computer Analysis of Images and Patterns pp. 772–782 (2015)
10. Biederman, I.: Recognition-by-components: a theory of human image understanding. Psychol. Rev. **94**(2), 115 (1987)
11. Jogin, M., Madhulika, M., Divya, G., Meghana, R., Apoorva, S., et al.: Feature extraction using convolution neural networks (CNN) and deep learning. In: 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 2319–2323 (2018)
12. Shah, K., Manwani, N.: Sparse reject option classifier using successive linear programming. Proc. AAAI Conf. Artif. Intell. **33**(1), 4870–4877 (2019)