Meta Pattern Concern Score: A Novel Metric for Customizable Evaluation of Multi-classification

Yanyun Wang¹,², Dehui Du¹, and Yuanhao Liu¹

¹ Software Engineering Institute, East China Normal University, Shanghai, China
dhdu@sei.ecnu.edu.cn
² Department of Computer Science, The University of Hong Kong, Hong Kong, China
yynwang@connect.hku.hk

Abstract. Classifiers have been widely implemented in practice, while how to evaluate them properly remains a problem. Commonly used two types of metrics respectively based on confusion matrix and loss function have different advantages in flexibility and mathematical completeness, while they struggle in different dilemmas like the insensitivity to slight improvements or the lack of customizability in different tasks. In this paper, we propose a novel metric named Meta Pattern Concern Score based on the abstract representation of the probabilistic prediction, as well as the targeted design for processing negative classes in multi-classification and reducing the discreteness of metric value, to achieve advantages of both the two kinds of metrics and avoid their weaknesses. Our metric provides customizability to pick out the model for specific requirements in different practices, and make sure it is also fine under traditional metrics at the same time. Evaluation in four kinds of models and six datasets demonstrates the effectiveness and efficiency of our metric, and a case study shows it can select a model to reduce 0.53% of dangerous misclassifications by sacrificing only 0.04% of training accuracy.

Keywords: Machine learning · Classification · Evaluation metric.

1 Introduction

A classifier is a computational learning system that can understand, translate and sort a target input into the predicted output, with the characteristics needed to construct the correct output learned automatically through processing labeled samples. While this simplicity lays a solid foundation for its widespread implementation, it also makes it a black-box model with its effectiveness hard to be properly evaluated. A rich body of evaluation metrics have been proposed to pick out the optimal one from the trained models, which can be roughly divided into two types. The metrics based on confusion matrix [1] allow one to define the objective of learning with an intuitive statistics of the output results, while they are insensitive to slight improvements because of their discrete value and fail to make a theoretically complete evaluation of the generalization ability of the model. The metrics come from loss functions [2] are designed with the above
issues considered, but the price is the unreasonable treatment of negative classes in multi-classification and the lack of flexibility in different tasks.

Therefore, an intuitive idea is, if a metric built in a different way from both of them can achieve their advantages and avoid their weaknesses at the same time? In this paper, we propose such an evaluation metric named Meta Pattern Concern Score mainly for the multi-classification (while it also works for binary cases). Through the well-designed abstract representation of the probabilistic prediction result, the metric approaches the loss metrics in the underlying mathematical principle, while it also allows people to determine the specific objective of optimization flexibly. Besides, the process of negative classes is redesigned for multi-classification as a compromise of existing methods, and the insensitivity caused by discrete values is reduced by the finer and adjustable grained intervals.

Evaluation in four different kinds of models and six reality and synthetic datasets demonstrates that our metric is as effective and efficient as the traditional metrics, and a case study in MNIST shows that it can pick out the ideal model different from the one selected by the traditional metrics according to specific requirements without violating them too much. By just sacrificing 0.04% of training accuracy, it can reduce 0.53% of specified dangerous misclassifications, which would be especially useful in the field of safety-critical tasks.

2 Background and Related Work

2.1 Classification Evaluation Basics

A series of evaluation metrics have been proposed to pick out the desired classifier. The metrics based on simply observations and statistics of the output results such as classification accuracy rate are widely used. Most of these metrics are derived from confusion matrix, which is one of the most popular structures to illustrate the results obtained from a classifier [1]. Given a \( c \)-class classification task, let \( m_{ij} \) be the number of samples actually belonging to the \( i \)-th class while that are classified into the \( j \)-th class, the confusion matrix \( \mathcal{M} \) can be defined as

\[
\mathcal{M} = \begin{bmatrix}
m_{11} & m_{12} & \cdots & m_{1c} \\
m_{21} & m_{22} & \cdots & m_{2c} \\
\vdots & \vdots & \ddots & \vdots \\
m_{c1} & m_{c2} & \cdots & m_{cc}
\end{bmatrix}
\]

(1)

When the number of classes \( c \) takes the value 2, we can obtain the typical binary form of the confusion matrix \( \mathcal{M}_b \) as

\[
\mathcal{M}_b = \begin{bmatrix}
m_{11} & m_{12} \\
m_{21} & m_{22}
\end{bmatrix}
\]

(2)

where the two classes are respectively regarded as Positive and Negative class, and the elements \( m_{11}, m_{12}, m_{21}, m_{22} \) represent four different situations named "True Positive" (TP), "False Negative" (FN), "False Positive" (FP) and "True Positive" (TP).
Negative") (TN). In multi-classification scenarios, given the confusion matrix \( \mathcal{M} \) and a target positive label \( r \), these concepts can be further extended as follows

\[
\begin{align*}
TP &= m_{rr} \\
FN &= \sum_{j=1}^{c} m_{rj} - m_{rr} \\
FP &= \sum_{i=1}^{c} m_{ir} - m_{rr} \\
TN &= \sum_{i=1}^{c} \sum_{j=1}^{c} m_{ij} - \sum_{i=1}^{c} m_{ir} - \sum_{j=1}^{c} m_{rj} + m_{rr}
\end{align*}
\]  \( (3) \)

with which the definition of such metrics can be given just as in the binary scenario \([3]\). Specifically, the \textit{Accuracy} given in Equ. \(4\) is the proportion of the number of correct predictions in the whole samples.

\[
\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \quad (4)
\]

The \textit{Precision} represents the number of samples that are actually positive from the total number of samples predicted to be positive as

\[
\text{Precision} = \frac{TP}{TP + FP} \quad (5)
\]

while the \textit{Recall}, also known as \textit{Sensitivity}, is the number of samples actually and predicted as positive from the total number of samples actually positive, which is represented as

\[
\text{Recall} = \frac{TP}{TP + FN} \quad (6)
\]

It is obvious from Equ. \(5\) and Equ. \(6\) that the values of \textit{Precision} and \textit{Recall} are difficult to be optimized simultaneously. As a result, the \textit{F\textsubscript{1}-score} is proposed as a weighted harmonic mean of them, through which we can take both of the metrics into consideration with equal importance.

\[
F\textsubscript{1}-score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)
\]

Matthew’s Correlation Coefficient (\( \text{MCC} \)), coming from the Phi Coefficient in statistics, is also useful in the measure of machine learning based classifications. As a metric generally regarded to be balanced, it takes into account true and false positives and negatives as follow

\[
\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (8)
\]

Another major category of evaluation metrics come from the ML theory. The objective of model training is to find the optimal parameters \( \omega^* \) to minimize the
value of global loss function $\mathcal{L}(\omega)$ over the whole training dataset $\mathcal{X}$, which can be expressed as

$$\omega^* = \arg \min \mathcal{L}(\omega) = \arg \min \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} l(x, \omega)$$

(9)

where $l(x, \omega)$ denotes the specific loss value computed from sample $x$ with parameters $\omega$. Naturally, how well the objective is met can serve as a measure of model quality, so $\mathcal{L}$ is also used to evaluate classifiers in many scenarios. One of the most famous loss function is the mean squared error ($MS$). Given a model $\mathcal{N}$ having fully connected final layer with the Softmax activation, the supervised labels $y$ in the format of one-hot, and the probability outputs $\hat{y}$ predicted by $\mathcal{N}$, the $MS$ can be denoted as

$$\mathcal{L}_{MS} = -\frac{1}{2|\mathcal{X}|} \sum_{i=1}^{c} \sum_{j=1}^{c} (\hat{y}_{i}^{(j)} - y_{i}^{(j)})^2$$

(10)

Currently, the most commonly used loss function for classification problems is, however, the cross entropy error ($CE$) [4]. Under the same condition of Equ. 10, the $CE$ is given as follows

$$\mathcal{L}_{CE} = -\frac{1}{|\mathcal{X}|} \sum_{i=1}^{c} \sum_{j=1}^{c} y_{i}^{(j)} \log(\hat{y}_{i}^{(j)})$$

(11)

2.2 Flaw of the Evaluation Metrics

The two kinds of evaluation metrics mentioned above both have their own advantages in contrast, while they also struggle in different inapplicable scenarios. Compared with loss value, metrics based on confusion matrix have advantages in flexibility and explainability. Through the various combinations of $TP$, $FN$, $FP$ and $TN$, these metrics allow one to define the objective to be optimized in an intuitive and direct manner according to the practical application. For instance, as over-abundance of negative examples is commonly seen in information retrieval, recommender systems and social network analysis [5], one of the sensible metrics in such situations is the $Precision$ who can measure the ability of the classifier not to label as positive a sample that is negative, so that the always-negative classifier will not be over-valued.

Nevertheless, the metrics referring to the classification results directly also struggle in some problems. For one thing, each element in the confusion matrix is essentially a number of classification results having specific actual and predicted labels, which means the values of all the metrics based on the confusion matrix are discrete. As a consequence, these metrics are insensitive to slight improvements in model parameters and only vary discontinuously and abruptly when the prediction results of some samples change [6], which is especially disappointing in the latter stages of model training. For another, these metrics heavily focus
on the existing data, while the ideal model is expected to be capable of reliably classifying previously unseen objects in the real world. So what really matters is, according to the Bayesian classification theory, how well the model approximates the underlying posterior probability distribution [7]. Paper has shown that improving observed accuracy rates and improving posterior probability estimation are not entirely synonymous [8]. In fact, efforts to improve posterior probability estimation may yield lower accuracy rates in the known dataset, but lead to better performance on future reality tasks.

Actually, the problems mentioned above are the important reasons why loss is proposed [6]. Through careful design, loss functions ensure the gradient of most parameters is not \( \vec{0} \) in the training process, so as to guide the continuous optimization of the model. And for the latter concern, the theoretical relationship between Bayesian posterior probability estimation and \( MS \) cost functions had also been explored and widely adopted [9]. However, squared error assumes Gaussian target data, which is violated given the discrete targets used to train classifiers. Besides, it can be found that \( MS \) is likely to be dominated by a few outlier data points which have particularly large errors.

So as a Log-linear error function based on the maximum likelihood estimate approach [4] who are impacted much less by these problems, \( CE \) gradually became more popular in classification tasks until the present. However, entropy based measures are specifically designed for binary targets at the beginning, so when \( CE \) is applied in multi-classification tasks, it has to treat all the classes except the class with the correct label as a collective concept "incorrect class", without considering the distinction between them. What’s worse, \( CE \) merely takes the probability of the correct label in the prediction results into error calculation, which means that of the incorrect labels, who are certainly more in quantity in every single output of multi-classification and may also impact the prediction effectiveness, are totally ignored.

All in all, there is indeed an uncharted territory: is the bias between the two kinds of metrics irreconcilable? To be specific, if there is a metric whose view of the multi-classification is more like the metrics from the confusion matrix, while its underlying principles are closer to the loss metrics, then it may have good customizability and theoretical completeness at the same time. In this paper, we are committed to proposing such a new metric.

3 Motivation

While the ideal aim of the new metric is conceived, its concrete form has not been clarified. In this section, by analyzing the requirements exposed by two reality scenarios, we determine how to design our metric. The scenarios come from a typical safety-critical task, Traffic Light Recognition, which is important regarding the traffic participants' safety in autonomous driving [10].

The first scenario happens when a red light shows, where there are two different kinds of misclassifications according to the recognition result. If it is classified as a green light, the autonomous vehicle will be incorrectly allowed to proceed,
which may endanger both its passengers and others, and is, of course, totally unacceptable. While if the light is predicted to be yellow, which means ”stop, unless it is unsafe to do so” [11], at least in most cases the car will stop and do not break the traffic rules. So it is obviously inappropriate to treat both of the situations in the same way, which is, however, what the traditional metrics do.

Since this kind of scenario with negative classes having different damages is commonly seen in practice, it would make sense to design for it. But before providing formal definitions and descriptions in the Sec. 4, showing our idea in an intuitive way firstly might be found useful. As mentioned previously, $CE$ is only affected by the predicted value of the correct label. Just as shown in Fig. 1 this is marked as: the significance of the correct (Red) class is $+1$, while that of others is assigned to be $0$ to represent no influence. On the other hand, both the metrics from confusion matrix and $MS$ treat the prediction as True or False, respectively corresponding to $+1$ and $−1$ in the ”Others” line.

In our idea, we give $−0.5$ instead of $−1$ to the incorrect prediction ”Yellow” to represent it is relatively less destructive. What’s more, we assign that of the ”Red” to be $−((−0.5) + (−1)) = +1.5$, which is actually a compromise between $CE$ and other metrics. Specifically, we neither ignore negative classes completely, nor make each of them as important as the positive class, while we set that the positive class and the whole of negative classes are of equal significance. These figures correspond to a concept to be discussed later called concern degree.

The second scenario is presented aiming at illustrating the flaw of loss. As shown in Fig. 2 there are two classifiers trained for a task with a training size 100 and the loss $CE$. The classifier A makes a mistake in one of the input samples, predicting the probability of the correct label to be only $0.49$, while the classifier B makes correct predictions in the whole dataset. However, the confidence of all the correct predictions of classifier A is $0.99$, while that of B is $0.98$. In this case, it is indeed that the $CE$ value of classifier A is less than that of B, but does that really mean A is a better choice? At least in Traffic Light Recognition,
the answer is no because it is the failure of prediction that is more intolerable compared with the minor weakness of confidence in such a safety-critical task. Note that although here we take $CE$ as an instance, $MS$ and other loss functions also have the same problem to some extent.

![Table 1]

**Fig. 2.** The figure shows a practical situation that a classifier with lower loss value is not the better choice for Traffic Light Recognition.

To cope with this demand, we can set a threshold value for the confidence and keep the model from being punished in specific samples once the confidence value of the correct label exceeds it. In our idea shown in Fig. 3, we further divide the entire confidence field into several intervals according to the granularity determined by the threshold, and calculate the punishment of a specific confidence value based on the distance between the interval it falls into and the interval representing the highest confidence.

![Diagram 2]

**Fig. 3.** The figure illustrates our idea to divide the confidence field and calculate its punishment value to deal with the dilemma in the second scenario.
4 Approach

According to the above analysis and idea, we establish an abstract representation of the multi-classifier’s output named meta pattern, based on which we propose our novel metric, Meta Pattern Concern Score (MPCS). The roadmap of our approach can be divided into three main parts as shown in Fig. 4. The first part is about running the original classifier and normalizing its result to a probability distribution using Softmax. The second part is where the two generators capture and build meta patterns from the processed results. Finally, in the third part, concern degrees and interval punishments are calculated and the MPCS on the entire training set is acquired. We will introduce the details in this section.

To begin with, obviously, it is not sufficient enough for our idea to merely take the label earning the highest confidence into consideration, while considering all the labels might lead to a lot of unnecessary calculation costs sometimes, so we introduce a hyper-parameter $k$ to indicate the first $k$ classes in the order of confidence decreasing to be considered according to the practical task.

**Definition 1.** With $l_i$ denoting a specific label in a classification task, the prediction pattern is denoted as a vector $P_{\text{pred}}$ with length $k$:

$$P_{\text{pred}} = [l_1, l_2, \ldots, l_k]$$

where the value of $k$ is between 1 and the number of label categories of the task.

To realize the idea described in Fig. 3, we divide the confidence field into $t$ intervals and determine the confidence level of a specific label under an input sample by the interval in which its confidence value falls and whether the label is the correct one or not.

**Definition 2.** The confidence pattern is denoted as a vector $P_{\text{conf}}$ with length $k$ that must be the same as $P_{\text{pred}}$ in one task:

$$P_{\text{conf}} = [C(l_1), C(l_2), \ldots, C(l_k)]$$
in which \(C(l_i)\) denotes the confidence level of the label \(l_i\). Given a hyper-parameter \(t \geq 1\) and the correct label \(l\), and let \(c_i\) be the confidence of label \(l_i\), \(C(l_i)\) can be calculated as follows:

\[
C(l_i) = \begin{cases} 
  \lceil t \cdot c_i \rceil, & l_i = l \\
  t - \lfloor t \cdot c_i \rfloor - 1, & l_i \neq l
\end{cases}
\]

Notice that there is a correspondence between each pair of elements in the same position of \(P_{\text{pred}}\) and \(P_{\text{conf}}\) in any specific sample. As the two patterns record the key information to uniquely represent any specific output of the classifier together, we refer to them collectively as the meta pattern for convenience.

**Definition 3.** The **meta pattern** of a classification result is the conceptual collective name for its prediction pattern \(P_{\text{pred}}\) and confidence pattern \(P_{\text{conf}}\).

Then it is the time to recall the idea shown in Fig. 1. We introduce a release list as a preset with every single element indicating a practical less destructive situation, as well as a factor to indicate that in what extent the concern of them could be released. In a specific sample, for every label selected into its \(P_{\text{pred}}\), there are three kinds of values of the corresponding element in its concern degree depending on whether the label is the correct one and whether it exists in any element of release list together with the correct label if not.

**Definition 4.** Given a prediction pattern \(P_{\text{pred}}\), its **concern degree** can be denoted as a vector \(I\) with the same length \(k\):

\[
I = [D(l_1), D(l_2), \ldots, D(l_k)]
\]

Given the correct label \(l\), a release list \(R\) with \(r_\alpha\) representing any element of it, and a release factor \(f_R\), \(D(l_i)\) can be calculated as:

\[
D(l_i) = \begin{cases} 
  \sum_{j \in P_{\text{pred}}, j \neq i} D(l_j), & l_i = l \\
  f_R, & l_i \neq l, [l, l_i] \in r_\alpha \\
  1, & l_i \neq l, [l, l_i] \notin r_\alpha
\end{cases}
\]

Now we can propose our metric \(\text{MPCS}\) based on the above definitions. As for how to convert the interval distance into a concrete punishment value, we select the \(\log\) function like CE.

**Definition 5.** Given the confidence pattern \(P_{\text{conf}}\) and the concern degree \(I\) of a specific sample’s classification result, its **meta pattern concern score** \(S\) can be calculated as:

\[
S = \sum_{i=1}^{k} \left( -\log \left( \frac{P_i}{t - 1} \right) \times \frac{I_i}{\sum_{j=1}^{k} I_j} \right)
\]

where \(P_i\) and \(I_i\) respectively represents the \(i\)-th element in the \(P_{\text{conf}}\) and \(I\), and \(k, t\) are the hyper-parameters defined previously.
Algorithm 1 Calculation of Meta Pattern Concern Score

\textbf{Input:} Classifier $N$, Dataset $(X, Y)$, Release List $R$, Release Factor $f_R$, Hyper-parameters $k$ and $t$

\textbf{Output:} Meta Pattern Concern Score $S$

1: $S' \leftarrow 0$
2: while $x, y \in (X, Y)$ do
3: \hspace{1em} $P_{\text{pred}} \leftarrow \text{argsort}(-N(x))[1: k]$
4: \hspace{1em} $P_{\text{conf}} \leftarrow t - \text{floor}((-\text{sort}(-N(x))[1: k]) \times t) - 1$
5: \hspace{1em} $P_{\text{conf}}[P_{\text{conf}} = -1] \leftarrow 0$
6: \hspace{1em} $I \leftarrow \text{ones}(k)$
7: \hspace{1em} if exist $i \leftarrow P_{\text{pred}}.\text{index}(y)$ then
8: \hspace{2em} $P_{\text{conf}}[i] \leftarrow t - P_{\text{conf}}[i] - 1$
9: \hspace{1em} while $r \in R$ and $P_{\text{pred}}[i] = r[0]$ do
10: \hspace{2em} while $j$ from 0 to $k - 1$ do
11: \hspace{3em} if $j \neq i$ and $P_{\text{pred}}[j] \in r[1:]$ then
12: \hspace{4em} $I[j] \leftarrow f_R$
13: \hspace{2em} end if
14: \hspace{2em} end while
15: \hspace{1em} end while
16: \hspace{1em} $I[i] \leftarrow \text{sum}(I) - 1$
17: \hspace{1em} end if
18: \hspace{1em} $P_{\text{conf}}[P_{\text{conf}} = 0] \leftarrow 1e-7$
19: \hspace{1em} $P' \leftarrow -\text{log}(P_{\text{conf}}/(t - 1))$
20: \hspace{1em} $I' \leftarrow I/\text{sum}(I)$
21: \hspace{1em} $S' \leftarrow S' + \text{sum}(\text{multiply}(P', I'))$
22: end while
23: $S \leftarrow S' / |X|$
24: \textbf{return} $S$

It can be noticed that \textit{MPCS} approaches \textit{CE} in the limit of $k = 1$ and $t \to \infty$. In other words, \textit{CE} can be viewed as a special case of \textit{MPCS} to some extent. We elaborate the whole process of \textit{MPCS} calculation as an algorithm in Alg. 1 including the establishment of meta pattern (line 3-5, 8), the calculation of concern degree $I$ (line 6-7, 9-17), and the computation of the metric \textit{MPCS} (line 18-21).

5 Evaluation

In this section, we will evaluate the effectiveness of \textit{MPCS} in two aspects. For one thing, through the comparison of value and trend between \textit{MPCS} and some other metrics, we verify its ability of classifier evaluation in the training process. For another, by illustrating the time cost of their calculation, we clarify that \textit{MPCS} has considerable efficiency as others.

There are six datasets from four sources used in the experiments, namely the IRIS and DIGITS from scikit-learn, the MNIST, the pedestrian counting dataset CT and the smooth subspace clustering dataset SS from UCR time-series
Table 1. The Spearman similarity between MPCS and the five selected metrics Accuracy (ACC), $F_1$-score (F1), MCC, MS and CE in the ten experiments under different combinations of the four models and six datasets.

| No. | Model | Dataset | Similarity | Similarity | Similarity | Similarity | Similarity |
|-----|-------|---------|------------|------------|------------|------------|------------|
|     |       |         | ACC        | F1         | MCC        | MS         | CE         |
| Exp 1 | MLP  | IRIS    | -0.9091    | -0.9092    | -0.9011    | 0.9873     | 0.9827     |
| Exp 2 | MLP  | DIGITS  | -0.9886    | -0.9936    | -0.9935    | 0.9989     | 0.9989     |
| Exp 3 | MLP  | MNIST   | -0.9881    | -0.9886    | -0.9880    | 0.9880     | 0.9841     |
| Exp 4 | CNN  | DIGITS  | -0.9106    | -0.9106    | -0.9107    | 0.9547     | 0.9547     |
| Exp 5 | CNN  | MNIST   | -0.9231    | -0.9229    | -0.9220    | 0.9299     | 0.9168     |
| Exp 6 | RNN  | UCR-CT  | -0.9576    | -0.9549    | -0.9587    | 0.9618     | 0.9619     |
| Exp 7 | RNN  | UCR-SS  | -0.9846    | -0.9844    | -0.9827    | 0.9929     | 0.9928     |
| Exp 8 | LSTM | UCR-SS  | -0.9382    | -0.9352    | -0.9373    | 0.9700     | 0.9718     |
| Exp 9 | LSTM | ADD     | -0.9721    | -0.9735    | -0.9706    | 0.9815     | 0.9862     |
| Exp 10| LSTM | MNIST   | -0.9757    | -0.9754    | -0.9760    | 0.9774     | 0.9763     |

Inspired by the evaluation method adopted in [14], we choose the Spearman correlation coefficient to calculate the similarity value between MPCS and the selected metrics, and regard this kind of relation as verification of its effectiveness. We illustrate the results in Table 1 with corresponding figures as a supplement in Fig. 5. From the results, we can find that all the similarity values exceed ±0.9, and more than half of them are even above ±0.97, based on which we can give an empirical inference that MPCS is highly correlated with the selected metrics, so as to indeed serve as a metric having practical significance.

On the other hand, we verify the efficiency of MPCS by the comparison of the calculation time cost between it and the selected metrics. Specifically, we calculate them in the whole training dataset for every epoch of the training process with a total of 150 epochs and take the average time cost as the final results. As illustrated in Table 2, it can be found that the time cost of all the metrics are about the same order of magnitude in general, which is more clear in relatively complex models.
6 Discussion

Now we have confirmed that $MPCS$ can serve as a "normal" training metric, but someone may still wonder why we need it and what benefits can it bring. So in this section, we show how $MPCS$ is "special" compared with other metrics. We present a case study with MLP and MNIST like Exp 3 as a foundation, and determine the less destructive misclassifications according to the t-SNE projection \cite{15}, a technique that can plot 2D embeddings for high dimensional datasets, while keeping the distance between the samples the same. To be specific, if the incorrect prediction result of a sample is adjacent to its correct label in the projected clusters, this mistake is less destructive compared to confusing non-adjacent labels. So we pick all that kinds of mistakes, which are framed in red in Fig. 6 into the release list $\mathcal{R}$. Notice that here we take this approach just as it has a real physical meaning, while actually the release condition can be set depending on any specific requirements in practice.

We train MLP for 150 epochs and record model parameters every epoch as a candidate, and then use different metrics respectively to pick out the optimal models among them, with the $f_\mathcal{R}$, $k$ and $t$ of $MPCS$ set to be 0.5, 5 and 200. In this case, all the metrics but $MPCS$ pick out the same model, and the prediction results of the two models are shown in Fig. 6 as the form of the confusion matrix. As can be calculated from the matrices, although the total number of misclassification of the model picked by the $MPCS$ 5128 is greater than that of the model picked by other metrics 5108, the number of destructive cases of the former model 2605 is less than that of the latter model 2621. In other words, we
Table 2. The calculation time cost of MPCS and the five selected metrics in the experiments shown in Table 1.

| No. | Time Cost ($\times 10^{-3}$s) |
|-----|------------------------------|
|     | ACC  | F1   | MCC  | MS   | CE   | MPCS |
| Exp 1 | 0.28 | 1.76 | 2.01 | 0.49 | 0.49 | 4.27 |
| Exp 2 | 0.33 | 1.62 | 2.79 | 0.59 | 0.49 | 13.66 |
| Exp 3 | 26.15 | 50.21 | 88.63 | 33.88 | 33.74 | 497.49 |
| Exp 4 | 31.56 | 32.98 | 33.90 | 31.80 | 35.55 | 43.98 |
| Exp 5 | 692.03 | 696.62 | 702.07 | 693.26 | 788.27 | 710.92 |
| Exp 6 | 7.25 | 8.34 | 8.17 | 7.26 | 7.59 | 7.08 |
| Exp 7 | 21.30 | 22.27 | 22.34 | 21.39 | 22.43 | 22.40 |
| Exp 8 | 38.92 | 40.05 | 39.99 | 39.04 | 40.07 | 38.89 |
| Exp 9 | 228.87 | 230.86 | 232.93 | 229.05 | 245.07 | 230.59 |
| Exp 10 | 2672.93 | 2703.11 | 2746.43 | 2685.93 | 2927.46 | 2802.63 |

pick out a model that trades off training accuracy from 91.49% to 91.45% for a reduced destructive rate from 51.31% to 50.78% by applying MPCS. This result makes sense in some real-world applications, especially in the field of safety-critical tasks.

Now we are ready to answer the question of what can MPCS contribute to the community. Theoretically, MPCS is designed regarding the advantages of the existing two main categories of training metrics. It provides an abstract view of model output to retain the customizability as the confusion matrix based metrics, and calculates its value in a way that approaches CE in the limit case to make the selected model approximate the posterior probability distribution as much as possible like Loss value. For the negative labels, MPCS neither ignores them completely like CE, nor gives them the same importance as MS, while it adopts a compromise to avoid the shortcomings of them and makes the evaluation more reasonable. Although MPCS can also not avoid discrete values in non-limit cases, not only the granularity is adjustable, but also the discrete degree is much lower than the confusion matrix based metrics in normal conditions. On the other hand, practically, different from other metrics having a fixed form all the time, the MPCS allows people to declare what they care more about of the model flexibly according to different practices, and try to cater for their specific will to pick out the optimal model, under the premise of not violating the traditional metrics too much and having a similar time cost as them.

7 Conclusion

In this paper, to make the evaluation more customizable and reasonable (mainly) for multi-classification, we proposed a novel metric called Meta Pattern Concern Score based on the abstract representation of the probabilistic prediction and
targeted design for negative classes processing and value discreteness reducing that can achieve advantages and avoid weaknesses of the two types of commonly used metrics. The experiments have confirmed the considerable effectiveness and the efficiency of MPCS and the case study empirically declares its effect in practice conforms to our design expectation. The work to implement the calculation of MPCS as a Python module and to explore the possibility that further using it to guide the learning process is in progress at present.

Acknowledgments

This work was mainly done when Yanyun Wang was an undergraduate student at the Software Engineering Institute, East China Normal University under the supervision of Prof. Dehui Du and in cooperation with a master’s student Yuanhao Liu. Yanyun Wang is currently a master’s student in the Department of Computer Science, The University of Hong Kong.

References

1. A. Luque, A. Carrasco, A. Martín, and A. de Las Heras, “The impact of class imbalance in classification performance metrics based on the binary confusion matrix,” *Pattern Recognition*, vol. 91, pp. 216–231, 2019.
2. C. Chen, H. Xu, W. Wang, B. Li, B. Li, L. Chen, and G. Zhang, “Communication-efficient federated learning with adaptive parameter freezing,” in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2021, pp. 1–11.
3. D. M. Ibrahim, N. M. Elshennawy, and A. M. Sarhan, “Deep-chest: Multi-classification deep learning model for diagnosing covid-19, pneumonia, and lung cancer chest diseases,” *Computers in biology and medicine*, vol. 132, p. 104348, 2021.
4. R. Das and S. Chaudhuri, “On the separability of classes with the cross-entropy loss function,” *arXiv preprint arXiv:1909.06930*, 2019.
5. P. Flach and M. Kull, “Precision-recall-gain curves: Pr analysis done right,” *Advances in neural information processing systems*, vol. 28, 2015.
6. S. Weidman, *Deep Learning from Scratch: Building with Python from First Principles*. O’Reilly Media, 2019.
7. R. O. Duda, P. E. Hart et al., *Pattern classification and scene analysis*. Wiley New York, 1973, vol. 3.
8. D. M. Kline and V. L. Berardi, “Revisiting squared-error and cross-entropy functions for training neural network classifiers,” *Neural Computing & Applications*, vol. 14, no. 4, pp. 310–318, 2005.
9. M. Hung, M. Hu, M. Shanker, and B. Patuwo, “Estimating posterior probabilities in classification problems with neural networks,” *International Journal of Computational Intelligence and Organizations*, vol. 1, no. 1, pp. 49–60, 1996.
10. C. Fernández, C. Guindel, N.-O. Salscheider, and C. Stiller, “A deep analysis of the existing datasets for traffic light state recognition,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 248–254.
11. HMG, “The traffic signs regulations and general directions 2016,” 2016.
12. H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, “The ucr time series archive,” *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1293–1305, 2019.
13. M. Zhang, D. Du, M. Zhang, L. Zhang, Y. Wang, and W. Zhou, “A meta-modeling approach for autonomous driving scenario based on std.” *International Journal of Software & Informatics*, vol. 11, no. 3, 2021.
14. T. Baluta, Z. L. Chua, K. S. Meel, and P. Saxena, “Scalable quantitative verification for deep neural networks,” in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 2021, pp. 312–323.
15. L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research*, vol. 9, no. 11, 2008.