MC-SQ: A Highly Accurate Ensemble for Multi-class Quantification (Extended Abstract)

Zahra Donyavi[1], Adriane Serapião[1,2], and Gustavo Batista[1]

[1] School of Computer Science and Engineering, University of New South Wales,
UNSW, Sydney, Australia, 2052
{z.donyavi, g.batista}@unsw.edu.au
[2] São Paulo State University
adriane.serapiao@unesp.br

Abstract. Quantification research proposes methods to estimate the class distribution in an independent sample. Many areas, such as epidemiology, sentiment analysis, political research and ecological surveillance, rely on quantification methods to estimate aggregated quantities. For instance, epidemiologists are often concerned with the dynamics of the number of disease cases across space and time. Thus, while classification predicts individual subjects, quantification is the class of methods that directly estimate the number of cases. Quantification is a thriving research area, and the community has proposed several approaches in the last decade. Nevertheless, most quantification research has focused on binary-class quantifiers, expecting these approaches to extend to multi-class using the one-versus-all (OVA) approach. However, enough empirical evidence indicates that OVA multi-class quantifiers' performance is subpar. This paper has two main contributions. First, we demonstrate why OVA quantifiers are doomed to underperform in multi-class settings due to a distribution shift they cannot handle. Second, we propose a new class of quantifiers based on ensemble learning that boosts the performance of the base quantifiers in the binary and, more importantly, multi-class settings. In one of the most comprehensive experimental setups ever attempted in quantification research, we show that our ensembles are the best-performing quantifiers compared with 33 state-of-the-art (single and ensemble) quantifiers and rank first in a recent quantification competition.

Keywords: Quantification · prevalence estimation · class probability estimation · ensembles · multi-class · machine learning.

1 Introduction

Quantification is the Machine Learning task that proposes methods to estimate the class distribution in an independent sample [8]. It finds applications in areas where we are more interested in understanding the behavior of groups than predicting individual cases. One well-known example is sentiment analysis, in which we often want to understand trends, such as the percentage of users making positive comments about a personality, brand, or product in a given period.

Classify & Count (CC) is the simplest quantifier. It is a direct application of classification to solve quantification problems. However, despite its simplicity, CC is a biased quantifier. Forman [9] reveals that CC contains a systematic bias. For an imperfect classifier, the CC method will underestimate the true proportion of positives $\hat{p}$ in a test set for $\hat{p} > p^*$ and overestimate for $\hat{p} < p^*$, where p^* is the particular proportion at which the CC method estimates correctly. This flaw has motivated a thriving community of researchers to develop novel quantifiers that provide accurate class estimates for the whole spectrum of class distributions.

So far, the quantification community has heavily focused on developing binary quantifiers. The idea is that those binary quantifiers can be extended to multi-class problems using the *one-versus-all* (OVA) approach. An OVA quantifier performs independent binary quantifications for each class versus all others and then normalizes the final estimates to sum to 100%.

However, recent empirical evidence has shown that OVA quantifiers' performance is subpar in multi-class problems [28]. Even more worrisome, multi-class quantifiers perform better than OVA quantifiers but just by a small margin. In this paper, we make two contributions to multi-class quantification. (*i*) For the first time, we explain why OVA quantifiers underperform in multi-class problems. (*ii*) We propose a simple ensemble approach that boosts the performance of existing multi-class quantifiers.

As contributions of this paper, we show that modeling a multi-class quantification problem with a set of OVA datasets induces a distribution shift in $p(\mathbf{x}|y)$. However, existing quantification methods assume that $p(\mathbf{x}|y)$ is constant. Therefore, these methods are doomed to perform poorly in multi-class settings. This finding will sound counter-intuitive to a significant part of the Machine Learning community since OVA is one of the *de-facto* approaches to converting binary classifiers into multi-class.

We show that a simple ensemble can significantly improve the performance of existing quantifiers. In a comprehensive empirical comparison with 33 state-of-the-art quantifiers and 40 datasets, our proposals are the best-performing quantifiers for both binary and multi-class datasets. In addition, our methods rank first in a recent quantification competition.

This paper is organized as follows. Section 2 introduces the basic concepts and the notation used throughout this paper. Section 3 presents the related work, briefly describing the 33 quantifiers used in our experiments. Section 4 discusses the limitations of using the OVA approach for multi-class quantification. Section 5 describes the ensemble approach that constitutes our main technical contribution. Section 6 presents the experimental results in both multi-class and binary quantification settings as well as the LeQua 2022 competition. Finally, Section 7 concludes our work and presents directions for future work.

2 Background

This section introduces the mathematical notation and fundamental concepts employed throughout this work.

A *dataset* is a collection of N examples such that $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$. Each $\mathbf{x}_n \in \mathcal{X}$ is a vector with M attributes, and $y_n \in \mathcal{Y} = \{1, 2, \ldots, C\}$ is the class label associated with $\mathbf{x}_n$.

We can create a predictive model with the dataset $\mathcal{D}$. The primary goal of *classification* is to predict the class label of each example using the covariates. Hence, the classifier is a predictive model h_c trained from $\mathcal{D}$ such that:

$$h_c : \mathcal{X} \to \{1, 2, \ldots, C\}$$

In this paper, we are interested in quantification problems. We define a *quantifier* as a supervised model learned from a dataset $\mathcal{D}$ to estimate the class prevalence in a test sample. Therefore, the quantifier is a function h_q such that:

$$h_q : \mathcal{S} \to [0, 1]^C$$

where $\mathcal{S}$ represents all possible sets of samples under the representation $\mathcal{X}$. From an unlabeled set $\mathbf{S} \in \mathcal{S}$, h_q outputs a vector $\hat{\mathbf{p}} = \langle \hat{p}_n \rangle_{n=1}^{C}$, where $\hat{p}_n$ is the estimate of the proportion of the class n, such that $\sum_{n=1}^{C} \hat{p}_n = 1$. The aim is to estimate the predicted ratios $\hat{\mathbf{p}}$ as close as possible to the true ratios $\langle p(n) \rangle_{n=1}^{C}$ of the unlabeled set $\mathbf{S}$.

Comparing the functions h_c and h_q, we can notice the similarities and differences in classification and quantification. These two tasks use the same data representation, a labeled tabular dataset $\mathcal{D}$, to induce their models. However, the objectives are distinct. While a classifier outputs a class label for each input instance, a quantifier outputs a class distribution estimate for a given *sample* of examples.

In both classification and quantification, the examples are *independent* of each other. Thus, the occurrence of one instance does not change the probability of the other instances. However, training and test samples are not *identically distributed* in quantification problems, as we expect that the class distribution will change.

Let us introduce one example to make these ideas more concrete. In the case of sentiment analysis, we can create a dataset of, say, tweets and label them as $\{\oplus, \ominus, \odot\}$, representing the positive, negative and neutral classes, respectively. A classifier will take a single tweet as input and output a unique class label. In contrast, a quantifier will take a set of tweets, such as the tweets from the last 24 hours that match the search criteria, and will output a vector $\hat{\mathbf{p}} = \langle \hat{p}_\oplus, \hat{p}_\ominus, \hat{p}_\odot \rangle$. In this example, $\hat{p}_\oplus$ is the estimated percentage of users expressing positive sentiment.

Two final observations about quantifiers. First, we can trivially convert the class probability estimates into counts by multiplying these probabilities with the test sample size. Thus, quantifiers are also known as *counters.* Second, the test sample size can vary according to the application. In the example of tweet sentiment analysis, we can have a test sample with tweets from the last hour, day, week, or month. Thus, it is essential to consider different test sample sizes when assessing quantifiers [19].

We conclude this section by defining a scorer as several quantifiers use them as an intermediate step in their computation. A *scorer* is a model induced from $\mathcal{D}$ such that:

$$h_s : \mathcal{X} \to \mathbb{R}^C$$

A scorer outputs a vector $\mathbf{s} = \langle s_n \rangle_{n=1}^{C}$ of real values called *scores* for a given input example. Each score s_n has a positive correlation with the posterior probability of the class y_n, i.e., $p(y_n|\mathbf{x})$. Accordingly, a higher s_n value means an increased chance for an example belonging to the class y_n.

3 Related Work

This section reviews all existing quantification algorithms in the literature. Due to lack of space, we briefly describe the 29 single quantifiers and one ensemble approach and provide relevant references for readers interested in further details. We organize this section according to the taxonomy proposed by [12], resulting in three groups of methods:

Classify, count & correct: These methods use a classifier to classify each instance and then count them by the class label. They often include an additional step that applies a correction to the counts.

Distribution matching: These methods parametrically model the training distribution and later search for the parameters that provide the best match against the test distribution.

Adaptations of classification algorithms: These methods adapt classification algorithms, transforming them into quantifiers.

We conclude this section by describing the only ensemble quantification approach in the literature.

3.1 Classify, Count & Correct

is a class of methods that count the classes using a classifier and apply a correction factor to obtain the final estimate.

CC (Multi-class): Classify & Count (CC) uses a classifier to count the class predictions for each label. Forman [9] shows that CC is a biased quantifier.

ACC (Binary): Adjusted Classify & Count (ACC) corrects the output of the CC method by employing the following correction factor:

$$p_{ACC}(y = \oplus|\mathbf{S}) = \frac{p_{CC}(y = \oplus|\mathbf{S}) - fpr}{tpr - fpr} \tag{1}$$

where $p_{CC}(y = \oplus|\mathbf{S})$ is the positive class prevalence provided by CC in the test set $\mathbf{S}$, and fpr is the false-positive and tpr is the true-positive rates often estimated in the training set using cross-validation.

PCC and PACC (Binary): Probabilistic Classify & Count (PCC) and Probabilistic Adjusted Classify & Count (PACC) [3] assume that probabilities have richer information than the label predictions of the classifier. PCC is a counterpart of the CC method, averaging the probabilities to estimate the class prevalence. Similar to ACC, PACC corrects the estimate of PCC using Equation 1. Since the class distribution influences the calibration of the classifiers, PCC and PACC approaches suffer from a *chicken-and-egg* problem [9].

GACC and GPACC (Multi-class): The Generalized Adjusted Classify & Count (GACC) and Generalized Probabilistic Adjusted Classify & Count (PACC) are multi-class generalizations of ACC and PACC, respectively [7]. These methods build the following system of equations and solve it via constrained least-squares regression:

$$p(h_c(\mathbf{S}) = n) = \sum_{i=1}^{C} p(h_c(\mathbf{S}) = n|y = i)p(y = i)$$

for $n = 1 : C$. As $P(h_c(\mathbf{S}) = n|y = i)$ is unknown, we estimate it using cross-validation in the training data.

FM (Multi-class): Friedman's method (FM) [11] also builds a system of equations. However, unlike GPACC, FM only considers a subset of the test instances with probabilities above the training class prevalences.

X, MAX, T50 (Binary): These methods search for different classification thresholds aiming for more reliable estimates for fpr and tpr [10]. X selects the threshold value where the difference between $1 - tpr$ and fpr is minimal. MAX chooses the threshold value that maximizes the denominator in Equation 1. T50 selects the threshold where $tpr \approx 50\%$.

MS (Binary): Median Sweep (MS) [10] returns the median of several applications of the ACC method for a range of classification thresholds. Each threshold estimates the tpr and fpr using cross-validation and then applies ACC correction. We use a variant with a subset of the thresholds that produce a denominator in Equation 1 greater than 0.25.

3.2 Distribution Matching

is a class of methods that parametrically model the training distribution and then search for the parameter that best matches the training and test distributions.

FMM (Binary): Forman's Mixture Method (FMM) [8] models the positive and negative class distributions using cumulative distribution functions (CDFs). As modeling $p(\mathbf{S}|y)$ is often difficult, this method uses the score distribution, i.e., $P(h_s(\mathbf{S})|y)$, which is more amenable since it is a set of unidimensional real values. FMM models the training scores from the positive and negative classes independently, as well as the test scores using CDFs. Then, it compares the test CDF with a mixture of positive and negative class CDFs while varying a mixture parameter. Forman uses the Probability-Probability plot to measure the difference between the training and test CDFs and returns the parameter whose curve produces the minimum difference as the positive class prevalence.

HDx and HDy (Binary): Gonzalez-Castro et al. [13] propose a mixture method similar to FMM that uses histograms to represent data distributions and the Hellinger Distance (HD) to compare those histograms. A weighted sum of the positive and negative class histograms provides a mixture that is compared with the test histogram. HDy uses scores to represent the distributions. Conversely, HDx operates over each feature independently and averages the HD values. The following equation describes the search performed by HDy:

$$p_{\mathrm{HDy}}(y = \oplus | \mathbf{S}^{\oplus}, \mathbf{S}^{\ominus}, \mathbf{S}^{\odot}) = \\ \underset{0 \leq \alpha \leq 1}{\arg\min} \left\{ \mathrm{HD}\left(\alpha H[\mathbf{S}^{\oplus}] + (1-\alpha) H[\mathbf{S}^{\ominus}], H[\mathbf{S}^{\odot}] \right) \right\}$$

where HD is the Hellinger distance and $H[\cdot]$ is a transformation of scores into a histogram representation, and $\mathbf{S}^{\oplus}$, $\mathbf{S}^{\ominus}$, and $\mathbf{S}^{\odot}$ are the positive, negative and test scores, respectively.

DyS (Binary): Distribution y-Similarity (DyS) [18] is a framework of mixture models method for binary quantification, based on HDy, that supports the use of different distance measures besides HD.

ED (Multi-Class): Similar to HDx, Energy Distance Minimization (ED) uses the actual features of the input space to model the distributions. But instead of HD, ED tries to minimize the Energy distance measure as described in [17].

Readme (Multi-class): Readme [15] is similar to HDx, as it also operates directly over features instead of using a classifier. Readme models the feature distribution by counting co-occurrences. Thus, for continuous attributes, this method requires feature discretization. Only a subset is used in an optimization problem solved by general least-squares regression.

EMQ (Multi-class): The Expectation-Maximization Quantifier (EMQ) [27] uses the well-known Expectation-Maximization (EM) algorithm to adjust the output of probabilistic classifiers for changes in the class distribution.

3.3 Adaptations of Classification Algorithms

is a class of methods that adapt an existing classification algorithm to quantification.

QT and QT-ACC (Multi-Class/Binary): Quantification trees (QT) [20] is a quantification method based on a decision tree algorithm. The main difference between QT and classification trees is the node-splitting criterion. QT employs a criterion suitable for the quantification task instead of a measure based on information theory used for classification tasks. QT-ACC is similar to QT with the additional application of the ACC correction (Equation 1).

PWK (Multi-class): The Proportion-Weighted k-Nearest Neighbor algorithm (PWK) [2] is an adaptation of the k-Nearest Neighbor (NN) algorithm to quantification using a weighting scheme which applies less weight on neighbors from the majority class.

CDE (Binary): The Class Distribution Estimation (CDE) [30] applies the cost-sensitive classification principle to update the classifier according to the class distribution change between the training and test sets. CDE is an iterative algorithm that re-trains the classifier according to the cost ratio calculated using the distribution mismatch ratio with the previous iteration's estimate.

SVM-Q and SVM-K (Binary): These methods use the SVM-perf implementation of Support Vector Machines (SVM) optimized for multivariate loss functions [16]. SVM-Q uses the Q-measure [1], and SVM-K uses the Kullback-Leibler Divergence [5].

3.4 Ensembles for quantification.

An ensemble is a set of individually trained models whose predictions are combined to forecast novel instances, often providing more accurate results than their base models [23].

Ensembles are extremely common in classification, but quantification has not dedicated much attention to this research venue. To the best of our knowledge, the use of ensembles in quantification is restricted to two articles [25,26].

In these papers, the authors explore the drift in $p(y)$ as a factor to generate diversity for the base classifiers. Therefore, they propose to train each base classifier using a different class prevalence. They sample the dataset using random sampling with replacement to vary $p(y)$ while ensuring that $p(\mathbf{x}|y)$ remains constant, a common assumption in quantification learning. The proposal uses the same pair of base classifier and quantifier for all samples and aggregates the individual predictions in a final predicted class prevalence.

We refer to this method as the *class-prevalence ensemble* (CPE) to avoid confusion with the approach proposed in this paper.

4 Multi-class Quantification

Forman [9] was the first to advocate using OVA for multi-class problems. An OVA quantifier performs independent binary quantifications for each class versus all others and then normalizes the final estimates to sum to 100%.

More recently, Schumacher et al. [28] have assessed 29 existing binary and multi-class quantifiers in a comprehensive evaluation involving 40 datasets. They conclude that binary quantifiers allied with OVA "*showed mediocre performance in the multi-class case.*"

What is intriguing here is why this is the case and which factors can make an accurate binary quantifier inaccurate for a multi-class problem transformed into a binary-class dataset with OVA. Schumacher et al. [28] hypothesize that the issue comes with the OVA normalization step. This section demonstrates that the OVA quantifiers perform poorly in multi-class settings due to a change in $p(\mathbf{x}|y)$.

We offer an intuitive explanation for the OVA issues using an example. Suppose we have a multi-class problem with only three classes with red, green, and blue labels. Blue is chosen as the positive ($\oplus$) class during one of the OVA executions, while red and green receive the negative label ($\ominus$) (see Figure 1). As we have a quantification problem, we expect the prevalence of red and green classes to vary independently, i.e., reds' prevalence can increase while greens' decreases and vice-versa. However, as the OVA quantifier sees the instances of these two classes as a single negative class, these prevalence drifts lead to a change in $p(\mathbf{x}|y)$. An intuitive way to realize this is to notice that an increase in reds' prevalence leads to a more complex separation of the positive and negative classes since the red class is closer to the blue class. In contrast, increasing green prevalence leads to an easier separation.

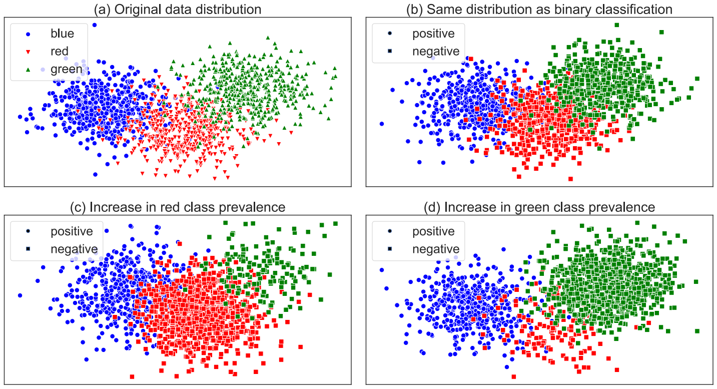

Fig. 1. A hypothetical three-class dataset (a) transformed into a binary-class problem (b) with class blue as positive. The change in the prevalence of the classes red and green causes a concept drift in $p(\mathbf{x}|y = \ominus)$, making the binary classes harder (c) or easier (d) to discriminate.

Suppose we characterize quantification as a $Y \rightarrow X$ problem [6]. The joint probability distribution is factored as $p(\mathbf{x}, y) = p(\mathbf{x}|y)p(y)$. We expect that the class distribution, $p(y)$, will change, as this is the primary motivation of the quantification. However, the quantification literature assumes that the conditional distribution $p(\mathbf{x}|y)$ remains constant. For instance, classify, count and adjust methods estimate the class errors $(p(h_c(\mathbf{S}) = n|y = i))$ on the training set, and the distribution matching methods try to model $p(\mathbf{S}|y)$ or $p(h_s(\mathbf{S})|y)$ using an approximation with training data.

A change in $p(\mathbf{x}|y)$ and $p(y)$ for $Y \rightarrow X$ problems is hardly addressed in the literature, as this problem is so complex it is considered impossible to solve [21]. Therefore, OVA quantification approaches are doomed to underperform, as observed in the literature [28].

5 Proposed Approach

This section presents an ensemble method that is our main technical contribution. We start discussing our main requirements:

Multi-class As we have discussed in Section 4, a binary-class method will not perform well on multi-class problems. Thus, the solution must be intrinsically multi-class since it will naturally apply to any number of labels.

Accurate One of the conclusions of experimental comparisons such as [28] is that multi-class quantification is a difficult problem, as both OVA and multi-class quantifiers perform poorly. Thus, we look for a significantly more accurate solution than current single and ensemble methods.

Simple The method must be simple as our primary motivation is to demonstrate the limitation of the current OVA approach and the directions for future research in multi-class quantification. We hope the community will further develop these ensemble approaches by looking for more complex (and hopefully accurate) variations.

Hyperparameter-free Our approaches must not add new hyperparameters beyond those inherited from the base classifiers and quantifiers. Our performance improvement should not originate from an extensive hyperparameter search.

Figure 2 illustrates our proposal. It consists of an ensemble of n pairs of classifier and quantifier. We vary the base classifier to provide diversity and fix the base quantifier. Therefore, we name our approach *multiple-classifier, single-quantifier*, or MC-SQ.

We set the number of pairs of classifier-quantifier as seven to eliminate parameters. We employ the following classifiers in our experiments: Random Forest (RF), Nave Bayes (NB), Gradient Boosting (GB), Support Vector Machines (SVM)[3], Linear Discriminant Analysis (LDA), Light Gradient Boosting Machines (LGBM), and Logistic Regression (LR). The motivation for selecting those algorithms is that they represent different learning paradigms and are often shortlisted as the most successful approaches in Machine Learning.

Finally, in our experiments, we employ the quantifiers EMQ, FM, GACC and GPACC as these were shortlisted as the best performing multi-class quantifiers in [28]. We provide further details in the next section.

[3] The SVM implementation in sklearn [24] uses one-versus-one to implement multi-class classifiers. This does not impact our ensembles; they only use the scores these classifiers provide.

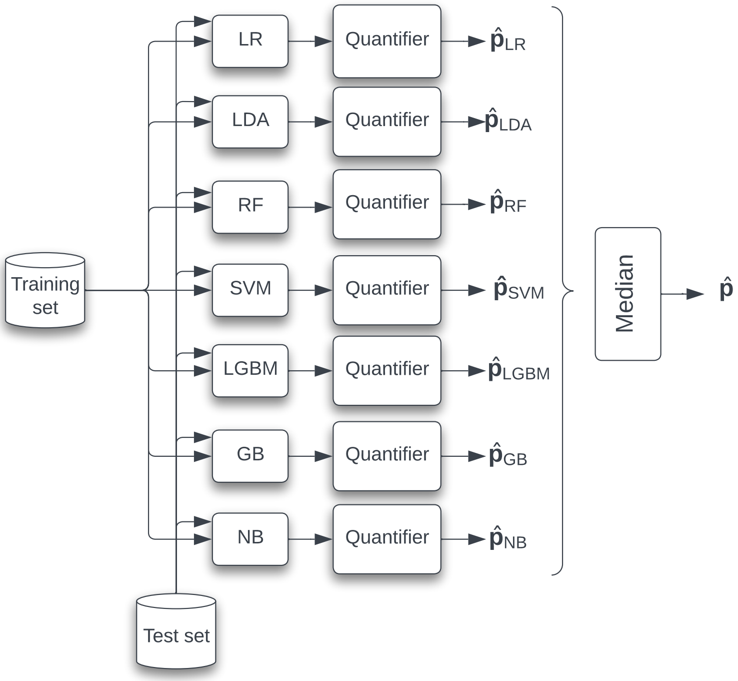

Fig. 2. Schematic of the proposed Multi-Classifier, Single-Quantifier ensemble approach.

6 Experimental Evaluation

This section details the experimental setup and assessment results. We include some ablation studies that provide insights into our method's design decisions.

6.1 Experimental Setup.

We are strongly committed to reproducible research. Therefore, we decided to use the same experimental design as [28], allowing direct comparison with those results. For the base quantifiers, we use the implementation provided in their paper[4]. Also, we created a paper website to store code, figures, tables, and detailed results perpetually[5].

We compare the results obtained by our ensemble methods with the single quantifiers and the class-prevalence ensembles from [26]. We use the ensemble

[4] The only exception is the HDy method which we found to differ significantly from the method described in [13]. In this case, we use our implementation.

[5] https://sites.google.com/view/mc-sq.

implementation provided in QuaPy [22]. As suggested in [22], we produce 50 different training samples with various distributions and apply Linear Regression as the base classifier to get scores for the 50 samples. A base quantifier is applied over the scores, producing 50 quantifiers for each class. The predicted prevalence is the normalized (to sum to 100%) average of the prevalences for each class label. To generate comparable results, we execute our ensembles and the class-prevalence ensembles over the same set of base quantifiers.

The experiments involve 40 benchmark datasets, 23 binary and 17 multi-class. We briefly describe the datasets' main characteristics on the paper's website. We use Absolute Error (AE), Equation 2, as the primary measure to assess our results. AE has several attractive features. For instance, it is easy to interpret and restrained in the interval $[0, 2]$ independently of the number of classes [29].

$$AE(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{C} \sum_{n=1}^{C} |\hat{p}_n - p_n| \tag{2}$$

The experimental setup follows the Artificial-Prevalence Protocol (APP) [14]. It consists of splitting a classification dataset into training and test sets. The test set class prevalence is artificially manipulated through sub-sampling, creating multiple test set samples. The idea is to create test samples with class prevalences that differ significantly from the training class distribution. We train the quantifiers with the training set, assess them in each test sample, and report the average AE across all test sets. We refer to [28] for further details about the experimental setup.

6.2 Experimental Results.

Table 1 shows the numerical results for the multi-class datasets. The proposed MC-SQ methods are the best-performing methods. The last row shows the average performance across all datasets[6]. MC-SQ provides a tremendous improvement over the base quantifiers: 22% for EM, 38% for GACC, 25% for GPACC, and 31% for FM.

Figure 3 provides the CD diagram for the results in Table 1. The four proposed ensembles (MC-SQ) occupy the five top-ranking positions. MC-SQ with the base quantifier FM outperforms with statistical significance all existing quantifiers but its sibling MC-SQ ensembles with GPACC and EM as base quantifiers.

Due to a lack of space, we have presented the numerical results for binary datasets on the paper's website. Figure 4 provides the CD diagram for the results in this table. The comparison involves a total of 34 approaches, as we also include DyS as a base quantifier for both ensemble approaches. We decided to include DyS with Tøpsoe distance, which is one of the best-performing binary quantifiers [28].

[6] We understand that computing average AE across datasets can be misleading, but it is often the only way to compare average performance improvement.

Table 1. Experimental results for multi-class datasets. Our proposal, the Multiple-classifier, Single-quantifier (MC-SQ) ensemble, is among the best-performing approaches.

	Single quantifiers									Class-Prevalence Ensembles				**MC-SQ Ensembles**			
Dataset	readme	ED	CC	PWK	QF	EM	GACC	GPACC	FM	EM	GACC	GPACC	FM	EM	GACC	GPACC	FM
bike	0.201	0.176	0.368	0.315	0.638	0.082	0.113	0.073	0.102	0.117	0.101	0.096	0.104	0.096	0.068	**0.059**	0.065
blog	0.180	0.290	0.588	0.422	0.547	0.196	0.360	0.236	0.285	0.256	0.238	0.249	0.264	0.167	0.173	**0.115**	0.122
conc	0.432	0.457	0.915	0.480	0.662	0.498	0.486	0.473	0.510	0.410	0.407	0.381	0.389	0.256	0.275	0.266	**0.245**
cond	0.129	0.093	0.343	0.213	0.431	0.059	0.155	0.066	0.088	0.085	0.078	0.064	0.074	0.054	0.054	**0.045**	0.047
contra	0.424	0.434	0.833	0.572	0.675	0.396	0.600	0.515	0.512	0.409	0.468	0.411	0.402	**0.391**	0.470	0.424	0.419
craft	0.412	0.274	0.752	0.442	0.763	0.191	0.296	0.190	0.190	0.271	0.264	0.206	0.218	0.225	0.186	0.168	**0.156**
diam	0.117	0.209	0.784	0.404	0.501	0.214	0.197	0.098	0.118	0.183	0.196	0.110	0.100	0.042	0.029	**0.027**	0.027
drugs	0.338	0.238	0.465	0.407	0.600	0.218	0.256	0.199	0.181	0.229	0.250	0.252	0.259	0.204	0.206	0.181	**0.163**
ener	0.331	0.169	0.879	0.439	0.925	0.131	0.273	0.115	0.129	0.161	0.225	0.120	0.130	0.158	0.108	**0.084**	**0.084**
fifa	0.221	0.278	0.481	0.384	0.432	0.127	0.313	0.181	0.216	0.198	0.182	0.202	0.211	0.117	0.145	0.111	**0.104**
news	0.446	0.245	0.827	0.471	0.917	0.221	0.498	0.335	0.376	**0.246**	0.288	0.249	0.238	0.260	0.325	0.261	0.268
nurse	0.263	0.049	0.138	0.213	0.399	0.022	0.023	0.019	0.020	0.027	0.016	0.017	0.018	0.015	0.011	0.013	**0.009**
thrm	0.471	0.470	1.042	0.511	0.827	0.494	0.780	0.629	0.663	0.323	0.409	0.337	0.382	0.330	0.344	0.321	**0.302**
turk	0.489	0.356	0.976	0.622	0.834	**0.277**	0.525	0.342	0.392	0.365	0.402	0.338	0.348	0.432	0.408	0.315	0.339
vgame	0.364	0.424	0.590	0.418	0.589	0.322	0.520	0.460	0.474	0.375	0.358	0.364	0.371	**0.315**	0.397	0.391	0.358
wine	0.428	0.440	0.965	0.496	0.613	0.757	0.656	0.575	0.605	0.414	0.416	0.371	0.388	**0.340**	0.440	0.449	0.431
yeast	0.474	**0.289**	0.878	0.295	0.526	0.613	0.567	0.408	0.413	0.546	0.448	0.401	0.411	0.353	0.450	0.476	0.482
Mean	0.336	0.288	0.696	0.418	0.640	0.284	0.389	0.289	0.310	0.272	0.279	0.245	0.253	0.221	0.241	0.218	**0.213**

Similarly to the multi-class case, MC-SQ with FM quantifier is also the best quantifier for binary datasets. The CD diagram shows that MC-SQ with FM outperforms all existing quantifiers but the Median Sweep (MS) with a significant statistical difference. These results are evidence of the performance of the ensemble approaches for quantification, as the MS algorithm can be framed as an ensemble approach.

6.3 Ablation Study: Number of Base Classifiers.

A relevant parameter for our ensembles is the number of base classifier-quantifier pairs. In our experimental results, we fixed this number to seven. However, it is unclear if we could improve performance using a different number of pairs.

We executed experiments with all possible combinations of the number of classifiers and averaged the results, grouping them by the number of base classifiers. Figure 5 shows the CD diagram for this experiment. The ensembles with seven classifiers obtain the best results but with diminishing returns and no statistically significant difference compared to six base pairs.

6.4 Case Study: The LeQua2022 Competition.

Recently, Esuli, Moreo and Sebastiani [4] organized the LeQua 2022 competition for quantification learning. The competition released a large dataset of product reviews from Amazon.

The competition was organized into four streams: T1-A and B released tabular datasets consisting of binary and multi-class problems. Similarly, T2-A and B released textual binary and multi-class datasets. In this section, we focus on the T1-B task as we do not want the feature extraction methods to influence the methods' performance. We are primarily interested in multi-class quantification.

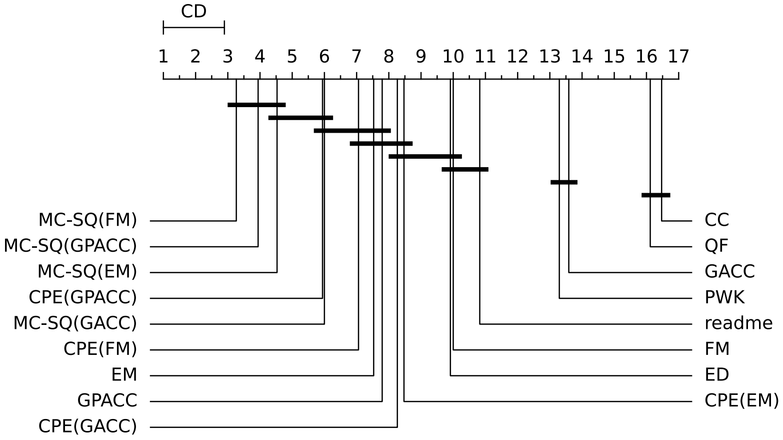

Fig. 3. CD diagram for multi-class datasets.

The released dataset has 28 classes with 20,000 training instances, and the competitors also had access to 1,000 development samples of 1,000 examples each. Finally, all methods were assessed in a hidden test set consisting of 5,000 test samples of 1,000 examples each.

Our methods use the default parameters. We assessed our ensembles with four base quantifiers: EM, FM, GACC and GPACC, using the development set and chose the best-performing one, GPACC as our representative. Finally, we assessed MC-SQ GPACC in the test set. Table 2 summarizes the results, with our proposal ranked first.

The competition uses Relative Absolute Error (RAE) as the main assessment criterion. RAE is defined as:

$$RAE(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{C} \sum_{n=1}^{C} \frac{|\hat{p}_n - p_n|}{p_n}$$

7 Conclusion and Future Work

In this paper, for the first time, we clarified the shortcomings of OVA quantification approaches in a multi-class context. We concluded that using OVA causes a distribution shift in $p(\mathbf{x}|y)$, which contradicts a common assumption of quantification methods.

We proposed an accurate multi-class ensemble method for quantification that naturally works for binary and multi-class problems. MC-SQ is a simple and parameter-free ensemble method that uses seven classifiers and the same base quantifier. We investigated its performance through extensive experiments

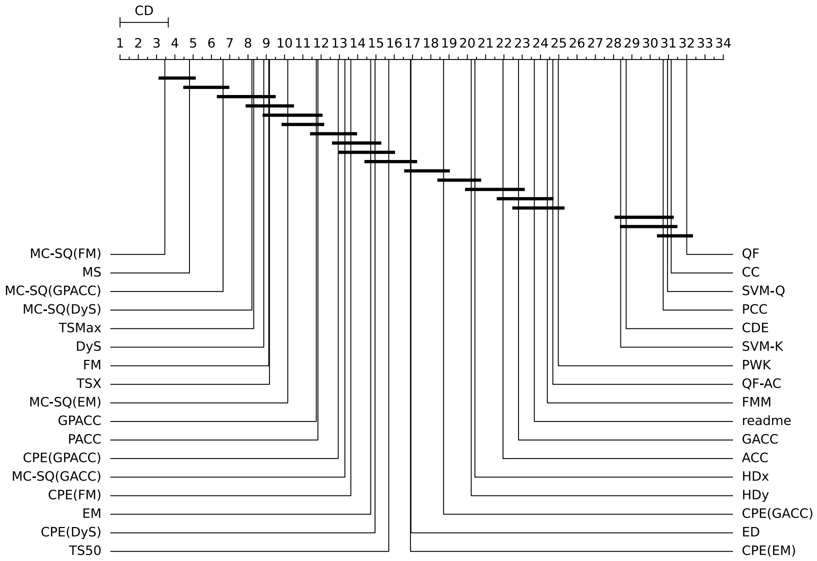

Fig. 4. CD diagram for binary datasets.

showing that MC-SQ is the best-performing quantifier for binary and multi-class problems.

In future work, we plan to investigate other ensemble variations, such as methods that use more than one quantification approach.

Acknowledgments

The authors would like to thank FAPESP (Process 2021/12278-0) for the financial support.

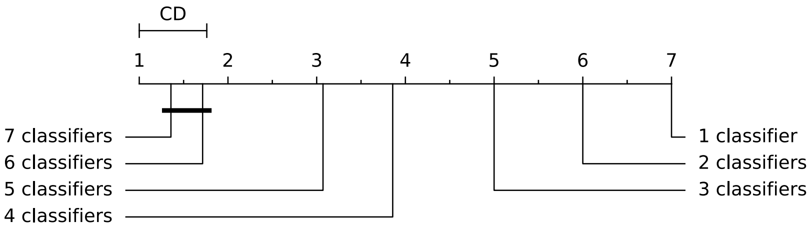

Fig. 5. CD diagram for the number of classifier-quantifier pairs.

Table 2. Results of LeQua2022 Task T1B, including our proposal MC-SQ GPACC ranked first.

Methods	RAE
MC-SQ GPACC	**0.861**
UniDortmund	0.880
UniOviedo(Team1)	0.884
UniOviedo(Team2)	1.114
KULeuven	1.178
SLD	1.182
PACC	1.305
ACC	1.421
CC	1.894
PCC	2.265
MLPE	4.577

References

1. Barranquero, J., Díez, J., del Coz, J.J.: Quantification-oriented learning based on reliable classifiers. Pattern Recognit **48**(2), 591–604 (2015)
2. Barranquero, J., González, P., Díez, J., del Coz, J.J.: On the study of nearest neighbor algorithms for prevalence estimation in binary problems. Pattern Recognit **46**(2), 472–482 (Feb 2013)
3. Bella, A., Ferri, C., Hernández-Orallo, J., Ramirez-Quintana, M.J.: Quantification via probability estimators. In: ICDM. pp. 737–742. IEEE (2010)
4. Esuli, A., Moreo, A., Sebastiani, F.: LeQua@CLEF2022: Learning to quantify. In: ECIR. pp. 374–381. Springer (2022)
5. Esuli, A., Moreo Fernández, A., Sebastiani, F.: A recurrent neural network for sentiment quantification. In: CIKM. pp. 1775–1778. ACM (2018)
6. Fawcett, T., Flach, P.A.: A response to webb and ting's on the application of roc analysis to predict classification performance under varying class distributions. Mach Learn **58**(1), 33–38 (2005)
7. Firat, A.: Unified framework for quantification. arXiv preprint arXiv:1606.00868 (2016)
8. Forman, G.: Counting positives accurately despite inaccurate classification. In: ECML. pp. 564–575. Springer (2005)
9. Forman, G.: Quantifying counts and costs via classification. Data Min Knowl Discov **17**(2), 164–206 (2008)
10. Forman, G., Kirshenbaum, E., Suermondt, J.: Pragmatic text mining: minimizing human effort to quantify many issues in call logs. In: SIGKDD. pp. 852–861. ACM (2006)
11. Friedman, J.H.: Class counts in future unlabeled samples (2014), https://jerryfriedman.su.domains/talks/HK.pdf
12. González, P., Castaño, A., Chawla, N.V., Coz, J.J.D.: A review on quantification learning. CSUR **50**(5), 1–40 (2017)
13. González-Castro, V., Alaiz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the hellinger distance. Information Sciences **218**, 146–164 (2013)
14. Hassan, W., Maletzke, A.G., Batista, G.: Pitfalls in quantification assessment. In: CIKM Workshops. ACM (2021)

15. Hopkins, D.J., King, G.: A method of automated nonparametric content analysis for social science. Am J Pol Sci **54**(1), 229–247 (2010)
16. Joachims, T.: A support vector method for multivariate performance measures. In: ICML. pp. 377–384 (2005)
17. Kawakubo, H., Du Plessis, M.C., Sugiyama, M.: Computationally efficient class-prior estimation under class balance change using energy distance. IEICE Trans Inf Syst **99**(1), 176–186 (2016)
18. Maletzke, A., dos Reis, D., Cherman, E., Batista, G.: Dys: a framework for mixture models in quantification. In: AAAI Conference. vol. 33, pp. 4552–4560 (2019)
19. Maletzke, A.G., Hassan, W., dos Reis, D.M., Batista, G.E.: The importance of the test set size in quantification assessment. In: IJCAI. pp. 2640–2646 (2020)
20. Milli, L., Monreale, A., Rossetti, G., Giannotti, F., Pedreschi, D., Sebastiani, F.: Quantification trees. In: ICDM. pp. 528–536. IEEE (2013)
21. Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognit **45**(1), 521–530 (2012)
22. Moreo, A., Esuli, A., Sebastiani, F.: Quapy: a python-based framework for quantification. In: CIKM. pp. 4534–4543. ACM (2021)
23. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. J Mach Learn Res **11**, 169–198 (1999)
24. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J Mach Learn Res **12**, 2825–2830 (2011)
25. Pérez-Gállego, P., Quevedo, J.R., del Coz, J.J.: Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. Inf Fusion **34**, 87–100 (2017)
26. Préz-Gállego, P., Castano, A., Ramón Quevedo, J., José del Coz, J.: Dynamic ensemble selection for quantification tasks. Inf Fusion **45**, 1–15 (2019)
27. Saerens, M., Latinne, P., Decaestecker, C.: Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. Neural Comput **14**(1), 21–41 (2002)
28. Schumacher, T., Strohmaier, M., Lemmerich, F.: A comparative evaluation of quantification methods. arXiv preprint arXiv:2103.03223 (2021)
29. Sebastiani, F.: Evaluation measures for quantification: An axiomatic approach. Inf Retr J **23**(3), 255–288 (2020)
30. Xue, J.C., Weiss, G.M.: Quantification and semi-supervised classification methods for handling changes in class distribution. In: KDD. pp. 897–906. ACM (2009)