Explicit Loss Minimization in Quantification Applications (Preliminary Draft)

Andrea Esuli[†] and Fabrizio Sebastiani[‡]

[†] Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: andrea.esuli@isti.cnr.it

[‡] Qatar Computing Research Institute
Qatar Foundation
PO Box 5825, Doha, Qatar
E-mail: fsebastiani@qf.org.qa

Abstract. In recent years there has been a growing interest in *quantification*, a variant of classification in which the final goal is not accurately classifying each unlabelled document but accurately estimating the prevalence (or "relative frequency") of each class c in the unlabelled set. Quantification has several applications in information retrieval, data mining, machine learning, and natural language processing, and is a dominant concern in fields such as market research, epidemiology, and the social sciences. This paper describes recent research in addressing quantification via explicit loss minimization, discussing works that have adopted this approach and some open questions that they raise.

1 Introduction

In recent years there has been a growing interest in *quantification* (see e.g., [2, 3, 9, 12, 19]), which we may define as the task of estimating the prevalence (or "relative frequency") $p_S(c)$ of a class c in a set S of objects whose membership in c is unknown. Technically, quantification is a *regression* task, since it consists in estimating a function $h : \mathcal{S} \times \mathcal{C} \rightarrow [0, 1]$, where $\mathcal{S} = \{s_1, s_2, ...\}$ is a domain of sets of objects, $\mathcal{C} = \{c_1, ..., c_{|\mathcal{C}|}\}$ is a set of classes, and $h(s, c)$ is the estimated prevalence of class c in set s. However, quantification is more usually seen as a variant of *classification*, a variant in which the final goal is not (as in classification) predicting the class(es) to which an unlabelled object belongs, but accurately

[0] The order in which the authors are listed is purely alphabetical; each author has given an equally important contribution to this work. Fabrizio Sebastiani is on leave from Consiglio Nazionale delle Ricerche.

estimating the percentages of unlabelled objects that belong to each class $c \in \mathcal{C}$. Quantification is usually tackled via supervised learning; it has several applications in information retrieval [8, 9], data mining [11–13], machine learning [1, 20], and natural language processing [4], and is a dominant concern in fields such as market research [7], epidemiology [18], and the social / political sciences [15].

Classification comes in many variants, including *binary* classification (where $|\mathcal{C}| = 2$ and exactly one class per item is assigned), *single-label multi-class* (SLMC) classification (where $|\mathcal{C}| > 2$ and exactly one class per item is assigned), and *multi-label multi-class* (MLMC) classification (where $|\mathcal{C}| \geq 2$ and zero, one or several classes per item may be assigned). To each such classification task there corresponds a quantification task, which is concerned with evaluating at the aggregate level (i.e., in terms of class prevalence) the results of the corresponding classification task. In this paper we will mostly be concerned with binary quantification, although we will also occasionally hint at how and whether the solutions we discuss extend to the SLMC and MLMC cases.

Quantification is not a mere byproduct of classification, and is tackled as a task on its own. The reason is that the naive quantification approach consisting of (a) classifying all the test items and (b) counting the number of items assigned to each class (the "classify and count" method – CC) is suboptimal. In fact, a classifier may have good classification accuracy but bad quantification accuracy; for instance, if the binary classifier generates many more false negatives (FN) than false positives (FP), the prevalence of the positive class will be severely underestimated.

As a result, several quantification methods that deviate from mere "classify and count" have been proposed. Most such methods fall in two classes. In the first approach a generic classifier is trained and applied to the test data, and the computed prevalences are then corrected according to the estimated bias of the classifier, which is estimated via k-fold cross-validation on the training set; "adjusted classify and count" (ACC) [14], "probabilistic classify and count" (PCC) [3], and "adjusted probabilistic classify and count" (PACC) [3] fall in this category. In the second approach, a "classify and count" method is used on a classifier in which the acceptance threshold has been tuned so as to deliver a different proportion of predicted positives and predicted negatives; example methods falling in this category are the "Threshold@50" (T50), "MAX", "X", and "median sweep" (MS) methods proposed in [12]. See also [9] for a more detailed explanation of all these methods.

In this paper we review an emerging class of methods, based on *explicit loss minimization* (ELM). Essentially, their underlying idea is to use (unlike the first approach mentioned above) simple "classify and count" without (unlike the second approach mentioned above) any heuristic threshold tuning, but using a classifier trained via a learning method explicitly optimized for quantification accuracy. This idea was first proposed, but not implemented, in a position paper by Esuli and Sebastiani [8], and was taken up by three very recent works [2, 9, 19] that we will discuss here.

The rest of this paper is organized as follows. Section 2 discusses evaluation measures for quantification used in the literature. Section 3 discusses the reason why approaching quantification via ELM is impossible with standard learning algorithms, and discusses three ELM approaches to quantification that have made use of nonstandard such algorithms. Section 4 discusses experimental results, while Section 5 concludes discussing questions that existing research has left open.

2 Loss Measures for Evaluating Quantification Error

ELM requires the loss measure used for evaluating prediction error to be directly minimized within the learning process. Let us thus look at the measures which are currently being used for evaluating SLMC quantification error. Note that a measure for SLMC quantification is also a measure for binary quantification, since the latter task is a special case of the former. Note also that a measure for binary quantification is also a measure for MLMC quantification, since the latter task can be solved by separately solving $|\mathcal{C}|$ instances of the former task, one for each $c \in \mathcal{C}$.

Notation-wise, by $\Lambda(\hat{p}, p, S, \mathcal{C})$ we will indicate a *quantification loss*, i.e., a measure Λ of the error made in estimating a distribution p defined on set S and classes $\mathcal{C}$ by another distribution $\hat{p}$; we will often simply write $\Lambda(\hat{p}, p)$ when S and $\mathcal{C}$ are clear from the context.

The simplest measure for SLMC quantification is *absolute error* (AE), which corresponds to the sum (across the classes in $\mathcal{C}$) of the absolute differences between the predicted class prevalences and the true class prevalences; i.e.,

$$AE(\hat{p}, p) = \sum_{c_j \in \mathcal{C}} |\hat{p}(c_j) - p(c_j)| \tag{1}$$

AE ranges between 0 (best) and $2(1 - \min_{c_j \in \mathcal{C}} p(c_j))$ (worst); a normalized version of AE that always ranges between 0 (best) and 1 (worst) can thus be obtained as

$$NAE(\hat{p}, p) = \frac{\sum_{c_j \in \mathcal{C}} |\hat{p}(c_j) - p(c_j)|}{2(1 - \min\limits_{c_j \in \mathcal{C}} p(c_j))} \tag{2}$$

The main advantage of AE and NAE is that they are intuitive, and easy to understand to non-initiates too.

However, AE and NAE do not address the fact that the same absolute difference between predicted class prevalence and true class prevalence should count as a more serious mistake when the true class prevalence is small. For instance, predicting $\hat{p}(c) = 0.10$ when $p(c) = 0.01$ and predicting $\hat{p}(c) = 0.50$ when $p(c) = 0.41$ are equivalent errors according to AE, but the former is intuitively a more serious error than the latter. *Relative absolute error* (RAE) addresses this problem by relativizing the value $|\hat{p}(c_j) - p(c_j)|$ in Equation 1 to the true class prevalence, i.e.,

$$RAE(\hat{p}, p) = \sum_{c_j \in \mathcal{C}} \frac{|\hat{p}(c_j) - p(c_j)|}{p(c_j)} \tag{3}$$

RAE may be undefined in some cases, due to the presence of zero denominators. To solve this problem, in computing *RAE* we can smooth both $p(c_j)$ and $\hat{p}(c_j)$ via additive smoothing, i.e.,

$$p_s(c_j) = \frac{p(c_j) + \epsilon}{(\sum\limits_{c_j \in \mathcal{C}} p(c_j)) + \epsilon \cdot |\mathcal{C}|} \tag{4}$$

where $p_s(c_j)$ denotes the smoothed version of $p(c_j)$ and the denominator is just a normalizing factor (same for the $\hat{p}_s(c_j)$'s); the quantity $\epsilon = \frac{1}{2 \cdot |s|}$ is often used as a smoothing factor. The smoothed versions of $p(c_j)$ and $\hat{p}(c_j)$ are then used in place of their original versions in Equation 3; as a result, *RAE* is always defined and still returns a value of 0 when p and $\hat{p}$ coincide.

RAE ranges between 0 (best) and $(((1-\min_{c_j \in \mathcal{C}} p(c_j))/\min_{c_j \in \mathcal{C}} p(c_j))+|\mathcal{C}|-1)$ (worst); a normalized version of *RAE* that always ranges between 0 (best) and 1 (worst) can thus be obtained as

$$NRAE(\hat{p}, p) = \frac{\sum_{c_j \in \mathcal{C}} \dfrac{|\hat{p}(c_j) - p(c_j)|}{p(c_j)}}{\dfrac{1 - \min\limits_{c_j \in \mathcal{C}} p(c_j)}{\min\limits_{c_j \in \mathcal{C}} p(c_j)} + |\mathcal{C}| - 1} \tag{5}$$

A third measure, and the one that has become somehow standard in the evaluation of SLMC quantification, is *normalized cross-entropy*, better known as *Kullback-Leibler Divergence* (KLD – see e.g., [5]). KLD was proposed as a SLMC quantification measure in [10], and is defined as

$$KLD(\hat{p}, p) = \sum_{c_j \in \mathcal{C}} p(c_j) \log \frac{p(c_j)}{\hat{p}(c_j)} \tag{6}$$

KLD is a measure of the error made in estimating a true distribution p over a set $\mathcal{C}$ of classes by means of a predicted distribution $\hat{p}$. *KLD* is thus suitable for evaluating quantification, since quantifying exactly means predicting how the items in set s are distributed across the classes in $\mathcal{C}$.

KLD ranges between 0 (best) and $+\infty$ (worst). Note that, unlike *AE* and *RAE*, the upper bound of *KLD* is not finite since Equation 6 has predicted probabilities, and not true probabilities, at the denominator: that is, by making a predicted probability $\hat{p}(c_j)$ infinitely small we can make *KLD* be infinitely large. A normalized version of *KLD* yielding values between 0 (best) and 1 (worst) may be defined by applying a logistic function, e.g.,

$$NKLD(\hat{p}, p) = \frac{e^{KLD(\hat{p},p)} - 1}{e^{KLD(\hat{p},p)}} \tag{7}$$

Also *KLD* may be undefined in some cases. While the case in which $p(c_j) = 0$ is not problematic (since continuity arguments indicate that $0 \log \frac{0}{a}$ should be taken to be 0 for any $a \geq 0$), the case in which $\hat{p}(c_j) = 0$ and $p(c_j) > 0$ is indeed problematic, since $a \log \frac{a}{0}$ is undefined for $a > 0$. To solve this problem, also in computing KLD we use the smoothed probabilities of Equation 4; as a result, KLD is always defined and still returns a value of zero when p and $\hat{p}$ coincide.

The main advantage of KLD is that it is a very well-known measure, having been the subject of intense study within information theory [6] and, although from a more applicative angle, within the language modelling approach to information retrieval [23]. Its main disadvantage is that it is less easy to understand to non-initiates than AE or RAE.

Overall, while no measure is advantageous under all respects, KLD (or $NKLD$) wins over the other measures on several accounts; as a consequence, it has emerged as the *de facto* standard in the SLMC quantification literature. We will hereafter consider it as such.

3 Quantification Methods Based on Explicit Loss Minimization

A problem with the quantification methods hinted at in Section 1 is that most of them are fairly heuristic in nature. A further problem is that some of these methods rest on assumptions that seem problematic. For instance, one problem with the ACC method is that it seems to implicitly rely on the hypothesis that estimating the bias of the classifier via k-fold cross-validation on Tr can be done reliably. However, since the very motivation of doing quantification is that the training set and the test set may have quite different characteristics, this hypothesis seems adventurous. In sum, the very same arguments that are used to deem the CC method unsuitable for quantification seem to undermine the previously mentioned attempts at improving on CC.

Note that all of the methods discussed in Section 1 employ *general-purpose* supervised learning methods, i.e., address quantification by leveraging a classifier trained via a general-purpose learning method. In particular, most of the supervised learning methods adopted in the literature on quantification optimize zero-one loss or variants thereof, and not a quantification-specific evaluation function. When the dataset is imbalanced (typically: when the positives are by far outnumbered by the negatives), as is frequently the case in text classification, this is suboptimal, since a supervised learning method that minimizes zero-one loss will generate classifiers with a tendency to make negative predictions. This means that FN will be much higher than FP, to the detriment of quantification accuracy[1].

In this paper we look at new, theoretically better-founded quantification methods based upon the use of classifiers explicitly optimized for the evaluation function used for assessing quantification accuracy. The idea of using learning

[1] To witness, in the experiments reported in [9] the 5148 test sets exhibit, when classified by the classifiers generated by the linear SVM used for implementing the CC method, an average FP/FN ratio of 0.109; by contrast, for an optimal quantifier this ratio is always 1.

algorithms capable of directly optimizing the measure (a.k.a. "loss") used for evaluating effectiveness is well-established in supervised learning. However, in our case following this route is non-trivial; let us see why.

As usual, we assume that our training data $Tr = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{|Tr|}, y_{|Tr|})\}$ and our test data $Te = \{(\mathbf{x}'_1, y'_1), ..., (\mathbf{x}'_{|Te|}, y'_{|Te|})\}$ are independently generated from an unknown joint distribution $P(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces, respectively. In this paper we will assume $\mathcal{Y}$ to be $\{-1, +1\}$.

Our task is to learn from Tr a hypothesis $h \in \mathcal{H}$ (where $h : \mathcal{X} \rightarrow \mathcal{Y}$ and $\mathcal{H}$ is the hypothesis space) that minimizes the expected risk $R^{\Lambda}(h)$ on sets Te of previously unseen inputs. Here Λ is our chosen loss measure; note that it is a set-based (rather than an instance-based) measure, i.e., it measures the error incurred by making an entire set of predictions, rather than (as instance-based measures λ do) the error incurred by making a single prediction. Our task consists thus of finding

$$\arg\min_{h\in\mathcal{H}} R^{\Lambda}(h) = \int \Lambda((h(\mathbf{x}'_1), y'_1), ..., (h(\mathbf{x}'_{|Te|}), y'_{|Te|}))dP(Te) \qquad (8)$$

If the loss function Λ over sets Te can be linearly decomposed into the sum of the individual losses λ generated by the members of Te, i.e., if

$$\Lambda((h(\mathbf{x}'_1), y'_1), ..., (h(\mathbf{x}'_{|Te|}), y'_{|Te|})) = \sum_{i=1}^{|Te|} \lambda(h(\mathbf{x}'_i), y'_i) \qquad (9)$$

then Equation 8 comes down to

$$\arg\min_{h\in\mathcal{H}} R^{\Lambda}(h) = \arg\min_{h\in\mathcal{H}} R^{\lambda}(h) = \int \lambda(h(\mathbf{x}', y')dP(\mathbf{x}', y') \qquad (10)$$

Discriminative learning algorithms estimate the expected risk $R^{\Lambda}(h)$ via the empirical risk (or "training error") $R^{\Lambda}_{Tr}(h)$, which by virtue of Equation 9 becomes

$$\hat{R}^{\Lambda}(h) = R^{\Lambda}_{Tr}(h) = R^{\lambda}_{Tr}(h) = \sum_{i=1}^{|Tr|} \lambda(h(\mathbf{x}_i, y_i) \qquad (11)$$

and pick the hypothesis h which minimizes $R^{\lambda}_{Tr}(h)$.

The problem with adopting this approach in learning to quantify is that all quantification loss measures Λ are such that Equation 9 does not hold. In other words, such loss measures are *nonlinear* (since they do not linearly decompose into the individual losses brought about by the members in the set) and *multivariate* (since Λ is a function of all the instances, and does not break down into univariate loss functions). For instance, we cannot evaluate the contribution to KLD of a classification decision for a single item $\mathbf{x}'_i$, since this contribution will depend on how the other items have been classified; if the other items have given rise, say, to more false negatives than false positives, then misclassifying a negative example (thus bringing about an additional false positive) is even beneficial!, since false positives and false negatives cancel each other out when it comes to quantification accuracy. This latter fact shows that *any* quantification loss (and not just the ones discussed in Section 2) is *inherently* nonlinear and multivariate. This means that, since Equations 9–11 do not hold for quantification loss measures Λ, we need to seek learning methods that can explicitly minimize $R^{\Lambda}_{Tr}(h)$ holistically, i.e., without making the "reductionistic" assumption that $R^{\Lambda}(h) = R^{\lambda}(h)$.

As mentioned in the introduction, the idea to use ELM in quantification applications was first proposed, but not implemented, in [8]. In this section we will look at three works [2, 9, 19] that have indeed exploited this idea, although in three different directions.

3.1 Quantification via Structured Prediction I: SVM(KLD) [9]

In [8] Esuli and Sebastiani also suggested using, as a "holistic" algorithm of the type discussed in the previous paragraph, the *SVM for Multivariate Performance Measures* (SVM$_{perf}$) learning algorithm proposed by Joachims [16][2].

SVM$_{perf}$ is a learner of the Support Vector Machine family that can generate classifiers optimized for any non-linear, multivariate loss function that can be computed from a contingency table (as all the measures presented in Section 2 are). SVM$_{perf}$ is an algorithm for *multivariate prediction*: instead of handling hypotheses $h : \mathcal{X} \rightarrow \mathcal{Y}$ mapping an individual item $\mathbf{x}_i$ into an individual label y_i, it considers hypotheses $\bar{h} : \bar{\mathcal{X}} \rightarrow \bar{\mathcal{Y}}$ which map entire tuples of items $\bar{\mathbf{x}} = (\mathbf{x}_1, ..., \mathbf{x}_n)$ into tuples of labels $\bar{\mathbf{y}} = (y_1, ..., y_n)$, and instead of learning hypotheses of type

$$h(\mathbf{x}) : sign(\mathbf{w} \cdot \mathbf{x} + b) \tag{12}$$

it learns hypotheses of type

$$\bar{h}(\bar{\mathbf{x}}) : \arg\max_{y' \in \mathcal{Y}} (\mathbf{w} \cdot \Psi(\bar{\mathbf{x}}, \bar{y}')) \tag{13}$$

where $\mathbf{w}$ is the vector of parameters to be learnt during training and

$$\Psi(\bar{\mathbf{x}}, \bar{y}') = \sum_{i=1}^{n} \mathbf{x}_i y'_i \tag{14}$$

(the *joint feature map*) is a function that scores the pair of tuples $(\bar{\mathbf{x}}, \bar{y}')$ according to how "compatible" the tuple of labels $\bar{y}'$ is with the tuple of inputs $\bar{\mathbf{x}}$.

While the optimization problem of classic soft-margin SVMs consists in finding

$$\begin{aligned} &\arg\min_{\mathbf{w}, \xi_i \geq 0} \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{|Tr|} \xi_i \\ &\text{such that} \quad y_i[\mathbf{w} \cdot \mathbf{x}_i + b] \geq (1 - \xi_i) \text{ for all } i \in \{1, ..., |Tr|\} \end{aligned} \tag{15}$$

[2] In [16] SVM$_{perf}$ is actually called $SVM\Delta_{multi}$, but the author has released its implementation under the name SVM$_{perf}$; we will indeed use this latter name.

the corresponding problem of SVM$_{perf}$ consists instead of finding

$$\begin{aligned} &\arg\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}\cdot\mathbf{w}+C\xi \\ &\text{such that} \;\; \mathbf{w}\cdot[\Psi(\bar{\mathbf{x}},\bar{y})-\Psi(\bar{\mathbf{x}},\bar{y}')+b]\geq\Lambda(\bar{y},\bar{y}')-\xi \;\; \text{for all } \bar{y}'\in\bar{\mathcal{Y}}/\bar{y} \end{aligned} \tag{16}$$

Here, the relevant thing to observe is that the sample-based loss Λ explicitly appears in the optimization problem.

We refer the interested reader to [16, 17, 21] for more details on SVM$_{perf}$ and on SVMs for structured output prediction in general. From the point of view of the user interested in applying it to a certain task, the implementation of SVM$_{perf}$ made available by its author is essentially an off-the-shelf package, since for customizing it to a specific loss Λ one only needs to write a small module that describes how to compute Λ from a contingency table.

While [8] only went as far as *suggesting* the use of SVM$_{perf}$ to optimize a quantification loss, its authors later went on to actually implement the idea, using KLD as the quantification loss and naming the resulting system SVM(KLD) [9]. In Section 4 we will describe some of the insights that they obtained from experimenting it.

3.2 Quantification Trees and Quantification Forests [19]

Rather than working in the framework of SVMs, the work of Milli and colleagues [19] perform explicit loss minimization in the context of a decision tree framework. Essentially, their idea is to use a quantification loss as the splitting criterion in generating a decision tree, thereby generating a *quantification tree* (i.e., a decision tree specialized for quantification). The authors experiment with three different quantification loss measures: (a) (a proxy of) absolute error, i.e., $D(\hat{p},p)=\sum_{c_j\in\mathcal{C}}|FP-FN|$; (b) KLD; (c) $MOM(\hat{p},p)=\sum_{c_j\in\mathcal{C}}|FP_j^2-FN_j^2|$. Measure (c) is of particular significance since it is not a "pure" quantification loss. In fact, notice that $MOM(\hat{p},p)$ is equivalent to $\sum_{c_j\in\mathcal{C}}(FN_j+FP_j)\cdot|FN_j-FP_j|$, and that while the second factor ($|FN_j-FP_j|$) may indeed be seen as representing quantification error, the first factor (FN_j+FP_j) is a measure of *classification* error. The motivation behind the authors' choice is to minimize at the same time classification and quantification error, based on the notion that a quantifier that has good quantification accuracy but low classification accuracy is somehow unreliable, and should be avoided.

The authors go on to propose the use of *quantification forests*, i.e., random forests of quantification trees, where these latter as defined as above. For more details we refer the interested reader to [19].

It should be remarked that [19] is the only one, among the three works we review in this section, that directly addresses SLMC quantification. The other two works that we have discussed [2, 9] instead address binary quantification only; in order to extend their approach to SLMC quantification, binary quantification has to be performed independently for each class and the resulting class prevalences have to be rescaled so that they sum up to 1. This is certainly suboptimal, but better solutions are not known since a SLMC equivalent of SVM$_{perf}$, which is binary in nature, is not known.

3.3 Quantification via Structured Prediction II: SVM(Q) [2]

Barranquero et al.'s forthcoming work [2] proposes an approach to binary quantification that combines elements of the works carried out in [9] and [19]. As suggested in [8], and as later implemented in [9], Barranquero et al. also use SVM$_{perf}$ to directly optimize quantification accuracy. However, similarly to [19], they optimize a measure (which they call *Q-measure*) that combines classification accuracy and quantification accuracy. Their Q-measure is shaped upon the famous F_β measure [22, Chapter 7], leading to a loss defined as

$$\Lambda = (1 - Q_\beta(\hat{p}, p)) = (1 - \frac{(\beta^2+1)\Gamma_c(\hat{p},p)\cdot\Gamma_q(\hat{p},p)}{\beta^2\Gamma_c(\hat{p},p)+\Gamma_q(\hat{p},p)})$$

where Γc and Γ_q are a measure of classification "gain" (the opposite of loss) and a measure of quantification gain, respectively, and $0 \leq \beta \leq +\infty$ is a parameter that controls the relative importance of the two; for $\beta = 0$ the Q_β measure coincides with Γ_c, while when β tends to infinity Q_β asymptotically tends to Γ_q.

As a measure of classification gain Barranquero et al. use recall, while as a measure of quantification gain they use $(1 - NAE)$, where NAE is as defined in Equation 2. The authors motivate the (apparently strange) decision to use recall as a measure of classification gain with the fact that, while recall by itself is not a suitable measure of classification gain (since it is always possible to arbitrarily increase recall at the expense of precision or specificity), to include precision or specificity in Q_β is unnecessary, since the presence of Γ_q in Q_β has the effect of ruling out anyway those hypotheses characterized by high recall and low precision / specificity (since these hypotheses are indeed penalized by Γ_q). The experiments presented in the paper test values for β in {0.5,1,2}.

4 Experiments

The approaches that the three papers mentioned in this section have proposed have never been compared experimentally, since the experimentation they report use different datasets. The only paper among the three where the experimentation is carried out on high-dimensional datasets is [9], where tests are conducted on text classification datasets, while [19] and [2] only report test on low-dimensional ones; in the case of [19], this is due to the fact that the underlying technology (decision trees) does not scale well to high dimensionalities.

We are currently carrying out experiments aimed to compare the approaches of [2] and [9] on the above high-dimensional datasets, testing a range of different experimental conditions (different class prevalence, different distribution drift, etc.) similarly to what done in [9]. We hope to have the results ready in time for them to be presented at the workshop.

5 Discussion

The ELM approach to quantification combines solid theoretical foundations with state-of-the-art performance, and promises to provide a superior alternative to the mostly empirical approaches that have been standard in the quantification literature. The key question that the (few) past works along this line leave open is: should one (a) optimize a combination of a quantification and a classification measure, or rather (b) optimize a pure quantification measure? In other words: how fundamental is classification accuracy to a quantifier? Approach (a) has indeed intuitive appeal, since we intuitively tend to trust a quantifier if it is also a good classifier; a quantifier that derives its good quantification accuracy from a high, albeit balanced, number of false positives and false negatives makes us a bit uneasy. On the other hand, approach (b) seems more in line with accepted machine learning wisdom ("optimize the measure you are going to evaluate the results with"), and one might argue that being serious about the fact that classification and quantification are fundamentally different means that, if a quantifier delivers consistently good quantification accuracy at the expense of classification accuracy, this latter fact should not be our concern. Further research is needed to answer these questions and to determine which among these contrasting intuitions is more correct.

Acknowledgements

We thank Thorsten Joachims for making SVM$_{perf}$ available, and Jose Barranquero for providing us with the module that implements the Q-measure within SVM$_{perf}$.

References

1. Rocío Alaíz-Rodríguez, Alicia Guerrero-Curieses, and Jesús Cid-Sueiro. Class and subclass probability re-estimation to adapt a classifier in the presence of concept drift. *Neurocomputing*, 74(16):2614–2623, 2011.
2. Jose Barranquero, Jorge Díez, and Juan José del Coz. Quantification-oriented learning based on reliable classifiers. *Pattern Recognition*, 48(2):591–604, 2015.
3. Antonio Bella, Cèsar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana. Quantification via probability estimators. In *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2010)*, pages 737–742, Sydney, AU, 2010.
4. Yee Seng Chan and Hwee Tou Ng. Estimating class priors in domain adaptation for word sense disambiguation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL 2006)*, pages 89–96, Sydney, AU, 2006.
5. Thomas M. Cover and Joy A. Thomas. *Elements of information theory.* John Wiley & Sons, New York, US, 1991.
6. Imre Csiszár and Paul C. Shields. Information theory and statistics: A tutorial. *Foundations and Trends in Communications and Information Theory*, 1(4):417–528, 2004.

7. Andrea Esuli and Fabrizio Sebastiani. Machines that learn how to code open-ended survey data. *International Journal of Market Research*, 52(6):775–800, 2010.
8. Andrea Esuli and Fabrizio Sebastiani. Sentiment quantification. *IEEE Intelligent Systems*, 25(4):72–75, 2010.
9. Andrea Esuli and Fabrizio Sebastiani. Optimizing text quantifiers for multivariate loss functions. *ACM Transactions on Knowledge Discovery and Data*, 2014. Forthcoming.
10. George Forman. Counting positives accurately despite inaccurate classification. In *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, pages 564–575, Porto, PT, 2005.
11. George Forman. Quantifying trends accurately despite classifier error and class imbalance. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 157–166, Philadelphia, US, 2006.
12. George Forman. Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206, 2008.
13. George Forman, Evan Kirshenbaum, and Jaap Suermondt. Pragmatic text mining: Minimizing human effort to quantify many issues in call logs. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 852–861, Philadelphia, US, 2006.
14. John J. Gart and Alfred A. Buck. Comparison of a screening test and a reference test in epidemiologic studies: II. A probabilistic model for the comparison of diagnostic tests. *American Journal of Epidemiology*, 83(3):593–602, 1966.
15. Daniel J. Hopkins and Gary King. A method of automated nonparametric content analysis for social science. *American Journal of Political Science*, 54(1):229–247, 2010.
16. Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 377–384, Bonn, DE, 2005.
17. Thorsten Joachims, Thomas Hofmann, Yisong Yue, and Chun-Nam Yu. Predicting structured objects with support vector machines. *Communications of the ACM*, 52(11):97–104, 2009.
18. Gary King and Ying Lu. Verbal autopsy methods with multiple causes of death. *Statistical Science*, 23(1):78–91, 2008.
19. Letizia Milli, Anna Monreale, Giulio Rossetti, Fosca Giannotti, Dino Pedreschi, and Fabrizio Sebastiani. Quantification trees. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM 2013)*, pages 528–536, Dallas, US, 2013.
20. Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure. *Neural Computation*, 14(1):21–41, 2002.
21. Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.
22. Cornelis J. van Rijsbergen. *Information Retrieval*. Butterworths, London, UK, second edition, 1979.
23. ChengXiang Zhai. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–213, 2008.