Training Data Cleaning for Text Classification

Andrea Esuli and Fabrizio Sebastiani
Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
Via Giuseppe Moruzzi, 1 – 56124 Pisa, Italy
andrea.esuli@isti.cnr.it, fabrizio.sebastiani@isti.cnr.it

ABSTRACT

In text classification (TC) and other tasks involving supervised learning, labelled data may be scarce or expensive to obtain. Semi-supervised learning and active learning are two strategies whose aim is maximizing the effectiveness of the resulting classifiers while minimizing the required amount of training effort; both strategies have been actively investigated for TC in recent years. Much less research has been devoted to a third such strategy, *training data cleaning* (TDC), which consists in devising ranking functions that sort the original training examples in terms of how likely it is that the human annotator has misclassified them, thereby providing a convenient means for the human annotator to revise the training set so as to improve its quality. Working in the context of boosting-based learning methods we present three different techniques for performing TDC and, on two widely used TC benchmarks, evaluate them by their capability of spotting misclassified texts purposefully inserted in the training set.

1. INTRODUCTION

In many applicative contexts involving supervised learning, labelled data may be scarce or expensive to obtain. In such situations, once we have trained the classifiers with the available training data (and tested them on the test data, and/or applied them to the unlabelled data that need to be classified), we are often left with the issue of how to improve the effectiveness of the existing classifiers, given that the amount of humanpower needed to perform further labelling is limited. One potential solution is to apply *semi-supervised learning* techniques (see e.g., [15]), which leverage on the availability (as it is often the case) of large quantities of unlabelled data, often sampled from the same distribution as the training and test data. A second solution is to apply *active learning* techniques (see e.g., [19]) which rank a set of unlabelled examples in terms of how much further information they would carry, once manually labelled, for retraining a (hopefully) better classifier. One difference between semi-supervised learning and active learning is that, while the former requires no humanpower to be devoted to further labelling, the latter does require it, but attempts to minimize it, by focusing the human labelling effort on the examples deemed most promising or informative.

Both semi-supervised learning and active learning have been widely studied in the context of text classification (TC) and other IR tasks involving supervised learning. There is instead a third route to solving the above problem that has been studied much less, namely, (computer-assisted) *training data cleaning* (TDC). Similarly to active learning, TDC techniques attempt to minimize the additional effort required from human annotators; however, while in active learning the human annotator is asked to label new *unlabelled* examples, in TDC he/she is required to double-check the manually *labelled* examples, looking for possible misclassified ones. Indeed, training data often contain misclassified items, sometimes as a result of lack of experience on the part of the junior annotators who have performed the labelling, sometimes as a result of tight time constraints under which the labelling activity has been performed. In the same way as a good active learning technique top-ranks the unlabelled examples that, once labelled, would prove the most informative for the training process, a good TDC technique top-ranks the training examples with the highest likelihood of being misclassified. This allows the human annotator to improve the quality of the training set by double-checking the labels attached to the training examples, starting with the ones most likely to be erroneous, and working down the ranked list until s/he sees fit.

We present three different techniques for performing TDC in TC, and test them using a boosting-based supervised learning device that generates confidence-rated predictions. The reason we are using this device is that, as will be apparent in Sections 2 and 3, it has two features that allow us to exemplify our TDC techniques particularly well, i.e., (i) it allows for a notion of confidence in the classifier's classification decisions; and (ii) the classifier it generates is actually a classifier committee.

1.1 Outline of the paper

The rest of the paper is organized as follows. Section 2 gives a brief description of the supervised learning device that we use in our experiments, focusing on the features that are important for understanding the three TDC techniques we study; these latter are presented in Section 3. In Section 4 we describe the results of our tests in which, using two popular TC benchmarks, evaluate these techniques by their capability of spotting misclassified texts that we have purposefully inserted in the training set for experimental purposes. Section 5 describes related research efforts, comparing them with the research described in this paper. Section 6 concludes, pointing at avenues for further research.

2. PRELIMINARIES

This work attempts to identify good TDC techniques for *text classification* (aka *text categorization* – TC), and for *multi-label* text classification (MLTC) in particular. Given a set of textual documents D and a predefined set of *classes* (aka *labels*, or *categories*) $C = \{c_1, \ldots, c_m\}$, MLTC can be defined as the task of estimating an unknown *target function* $\Phi : D \times C \rightarrow \{-1, +1\}$, that describes how documents ought to be classified, by means of a function $\hat{\Phi} : D \times C \rightarrow \{-1, +1\}$ called the *classifier*[1]; here, $+1$ and -1 represent membership and non-membership of the document in the class. Each document may thus belong to zero, one, or several classes at the same time. As usual, we accomplish MLTC by generating m independent binary classifiers $\hat{\Phi}^j : D \rightarrow \{-1, +1\}$, one for each $c_j \in C$, entrusted with the task of deciding whether a document belongs or not to class c_j.

As the learning device for generating our classifiers we use a boosting-based learner, called MP-Boost [6]; boosting is among the classes of supervised learning devices that have obtained the best performances in several learning tasks and, at the same time, have strong justifications from computational learning theory. MP-Boost is a variant of AdaBoost.MH [13] optimized for multi-label settings, which has been shown in [6] to obtain considerable effectiveness improvements with respect to AdaBoost.MH.

MP-Boost works by iteratively generating, for each class c_j, a sequence $\hat{\Phi}^j_1, \ldots, \hat{\Phi}^j_S$ of classifiers (called *weak hypotheses*). A weak hypothesis is a function $\hat{\Phi}^j_s : D \rightarrow \mathbf{R}$, where D is the set of documents and $\mathbf{R}$ is the set of real numbers. The sign of $\hat{\Phi}^j_s(d_i)$ (denoted by $sgn(\hat{\Phi}^j_s(d_i))$) represents the binary decision of $\hat{\Phi}^j_s$ on whether d_i belongs to c_j, i.e. $sgn(\hat{\Phi}^j_s(d_i)) = +1$ (resp., -1) means that d_i is believed to belong (resp., not to belong) to c_j. The absolute value of $\hat{\Phi}^j_s(d_i)$ (denoted by $|\hat{\Phi}^j_s(d_i)|$) represents instead the confidence that $\hat{\Phi}^j_s$ has in this decision, with higher values indicating higher confidence.

At each iteration s MP-Boost tests the effectiveness of the most recently generated weak hypothesis $\hat{\Phi}^j_s$ on the training set, and uses the results to update a distribution D^j_s of weights on the training examples. The initial distribution D^j_1 is uniform. At each iteration s all the weights $D^j_s(d_i)$ are updated, yielding $D^j_{s+1}(d_i)$, so that the weight assigned to an example correctly (resp., incorrectly) classified by $\hat{\Phi}^j_s$ is decreased (resp., increased). The weight $D^j_{s+1}(d_i)$ is thus meant to capture how ineffective $\hat{\Phi}^j_1, \ldots, \hat{\Phi}^j_s$ have been in guessing the correct c_j*-assignment* of d_i (denoted by $\Phi^j(d_i)$), i.e., in guessing whether training document d_i belongs to class c_j or not. By using this distribution, MP-Boost generates a new weak hypothesis $\hat{\Phi}^j_{s+1}$ that concentrates on the examples with the highest weights, i.e. those that had proven harder to classify for the previous weak hypotheses.

The overall prediction on whether d_i belongs to c_j is obtained as a sum $\hat{\Phi}^j(d_i) = \sum_{s=1}^{S} \hat{\Phi}^j_s(d_i)$ of the predictions of the weak hypotheses. The final classifier $\hat{\Phi}^j$ is thus a *committee* of S classifiers, each classifier casting a weighted vote (the vote being the binary decision $sgn(\hat{\Phi}^j_s(d_i))$, the weight being the confidence $|\hat{\Phi}^j_s(d_i)|$) on whether d_i belongs to c_j. For the final classifier $\hat{\Phi}^j$ too, $sgn(\hat{\Phi}^j(d_i))$ represents the binary decision as to whether d_i belongs to c_j, while $|\hat{\Phi}^j(d_i)|$ represents the confidence in this decision.

See [6] for more details on these and other aspects of MP-Boost.

3. THREE TECHNIQUES FOR TRAINING DATA CLEANING

In the following, by a *TDC technique* we will mean a technique that, given a training set Tr and a class c_j, produces a ranking $r_j(Tr)$ in which the elements of Tr are sorted in decreasing order of their likelihood of being mislabelled for c_j. Different techniques correspond to different ways of estimating this likelihood.

We now present three alternative TDC techniques. For each $c_j \in C$, the first technique (that we dub *the confidence-based technique* – CON, in short) consists in (i) training the classifier $\hat{\Phi}^j$ on Tr; (ii) reclassifying Tr by means of $\hat{\Phi}^j$; and (iii) ranking Tr in increasing order of $\hat{\Phi}^j(d_i) \cdot \Phi^j(d_i)$ value. Note that, while $\Phi^j(d_i)$ is a value in {-1,+1}, $\hat{\Phi}^j(d_i)$ is a value in $(-\infty, +\infty)$, so $\hat{\Phi}^j(d_i) \cdot \Phi^j(d_i)$ is also in $(-\infty, +\infty)$; a positive (resp., negative) value of $\hat{\Phi}^j(d_i) \cdot \Phi^j(d_i)$ indicates correct (resp., incorrect) classification, while a high (resp., low) absolute value of $\hat{\Phi}^j(d_i) \cdot \Phi^j(d_i)$ indicates that this classification decision has been taken with high (resp., low) confidence. The CON technique thus corresponds to (a) top-ranking the examples $d_i \in Tr$ that $\hat{\Phi}^j$ has misclassified, in decreasing order of the confidence $|\hat{\Phi}^j(d_i)|$ with which $\hat{\Phi}^j$ has taken its decision, and (b) appending to this list the examples $d_i \in Tr$ that $\hat{\Phi}^j$ has correctly classified, in increasing order of the confidence $|\hat{\Phi}^j(d_i)|$. The rationale of this technique is that, if $\hat{\Phi}^j$ has misclassified a training example d_i with high confidence, this means that the c_j-assignment made to d_i by the human annotator is highly at odds with the c_j-assignments that the human annotator has made for the other training examples, which indicates that the human annotator may well have misclassified d_i for c_j.

For each $c_j \in C$, the second technique (that we dub *the nearest neighbours technique* – NN) consists in ranking the training examples in terms of how inconsistent their c_j-assignment is with the c_j-assignments of their k nearest neighbours, for a predefined k. More formally, this technique consists in (i) computing, for each $d_i \in Tr$, the value

$$\zeta(d_i, c_j) = \sum_{d_z \in Tr_k(d_i)} sim(d_i, d_z) \cdot \Phi^j(d_z) \qquad (1)$$

where $sim(\cdot, \cdot)$ denotes a measure of similarity between documents and $Tr_k(d_i)$ denotes the k training examples most similar to d_i; and (ii) ranking Tr in increasing order of

[1] Consistently with most mathematical literature we use the caret symbol (ˆ) to indicate estimation.

$\zeta(d_i, c_j) \cdot \Phi^j(d_i)$ value. For class c_j, the examples d_i with c_j-assignments highly consistent with the c_j-assignments of their neighbours will have high $\zeta(d_i, c_j) \cdot \Phi^j(d_i)$ values, which means that the top-ranked examples (which are the ones with the lowest $\zeta(d_i, c_j) \cdot \Phi^j(d_i)$ values) will be the ones with c_j-assignments most dissimilar from those of their closest neighbours. Equation (1), of course, is that of the standard distance-weighted k-NN learning device (see e.g., [16, 17]), the only difference being that, while in the standard case $\Phi^j(d_z)$ ranges on {0,1}, in our case it ranges on {-1,+1}, which means that neighbours with a negative c_j-assignment weigh negatively, instead of having no effect, on $\zeta(d_i, c_j)$. This variant of the k-NN learning device is discussed in [8].

For each $c_j \in C$, the third technique (that we dub *the committee-based technique* – COM) consists in (i) training the classifier $\hat{\Phi}^j$ on Tr; (ii) reclassifying Tr by means of $\hat{\Phi}^j$; and (iii) ranking Tr in increasing order of $\Delta(\hat{\Phi}^j(d_i)) \cdot sgn(\hat{\Phi}^j(d_i)) \cdot \Phi^j(d_i)$ value, where $\Delta(\hat{\Phi}^j(d_i))$ is a measure of the *disagreement* among the S members of $\hat{\Phi}^j$ on whether d_i belongs to c_j or not. This technique is based on the intuition that the examples most in need of double-checking are the ones which $\hat{\Phi}^j$ has misclassified (i.e., are such that $sgn(\hat{\Phi}^j(d_i)) \cdot \Phi^j(d_i) = -1$) *with the most widespread agreement* among its S members; in other words, if the information that a training example provides to the training process is so inconsistent with that provided by the other training data as to have the members of the generated classifier committee misclassify the example, and with widespread agreement, then it is likely that the example might be mislabelled. This technique will thus top-rank the training examples that the committee has misclassified and on which the S members of the committee agree most, mid-rank those on which there is disagreement, and bottom-rank those that the committee has classified correctly and on which the S members of the committee agree most. The key difference between the first technique (CON) and this technique is that here the confidence that a classifier committee has in a certain decision is taken to coincide with the level of (weighted) agreement among its members, and not with the (weighted) sum of the individual opinions. As a measure of disagreement among the S members of the committee we have chosen to use *standard deviation* (denoted σ). This is a natural choice, given that the values $\hat{\Phi}^j_1(d_i), \ldots, \hat{\Phi}^j_S(d_i)$ are real numbers: standard deviation thus allows to measure disagreement by taking into account not only the polarity $sgn(\hat{\Phi}^j_s(d_i))$ of each member's decision, but also its confidence level $|\hat{\Phi}^j_s(d_i)|$, so that two members with views of different polarity are taken to disagree more if they are highly confident in their views, and less if they are not.

Actually, there is a fourth technique (that we dub *the distribution-based technique* – DIS) that might come to mind. For each $c_j \in C$, this technique consists in (i) training the classifiers $\hat{\Phi}^j$ on Tr, and (ii) ranking the examples $d_i \in Tr$ in decreasing order of the $D^j_S(d_i)$ value that MP-Boost has produced as a side effect of the learning process. The rationale of this technique is that, since the training examples that maximize $D^j_S(d_i)$ are the ones that have turned out the most difficult to make sense of during the boosting iterations, they are thus the ones whose c_j-assignment is most highly at odds with the c_j-assignment of the other training examples. The problem with the DIS technique is that it turns out to be equivalent to our first technique (CON), in the sense that CON and DIS always generate identical rankings, a fact that had never been noted in the literature[2]. The only advantage that DIS provides over CON is thus that there is no need to reclassify the training examples by means of $\hat{\Phi}^j$, since the information needed for ranking is already available after training has occurred.

Before discussing the experiments it is worthwhile noting that, although we have described these techniques in the context provided by a boosting-based learner which generates confidence-rated predictions, all of these techniques can be used also in connection with other learning devices. More specifically, CON only needs the classifier to return a score of confidence in its own decision, NN has no specific requirements, and COM requires the classifier to consist of a committee of classifiers. Moreover, the discussed equivalence between CON and DIS has the practical consequence of making available a technique equivalent to DIS to learning devices not based on boosting.

4. EXPERIMENTS

4.1 Experimental protocol

In order to test our TDC techniques we use a standard MLTC dataset $\Omega = \langle Tr, Te \rangle$ split into a training set Tr and a test set Te. We assume that Tr contains no misclassified examples, and we simulate the presence of misclassified training examples by artificially "perturbing" a small number m of training examples; we call the value $p = \frac{m}{|Tr|}$ the *perturbation ratio*. In what follows, "perturbing a training example d_i for class c_j" means changing its c_j-assignment, from positive to negative (in this case we call d_i a *false negative for* c_j) or from negative to positive (a *false positive*); by $\widehat{Tr}$ we denote the training set after perturbation.

We test two different perturbation techniques, which we call *random perturbation* (RP) and *targeted perturbation* (TP). As the name implies, in RP the training examples to perturb are picked at random from Tr. The same training examples (x% of the entire lot) are perturbed for all classes $c_j \in C$. TP is instead obtained by (i) training the classifiers $\hat{\Phi}^j$ on Tr, (ii) reclassifying Tr by means of them, (iii) ranking, for each $c_j \in C$, the reclassified examples in increasing order of the confidence $|\hat{\Phi}^j(d_i)|$ that $\hat{\Phi}^j$ had in classifying them, and (iv) perturbing the top-ranked ones, in number equal to x% of the training examples. The rationale of this technique is that the training examples that $\hat{\Phi}^j$ classifies with low confidence are more likely to be "borderline" examples for c_j; as a result, these examples are the ones that, should they be manually labelled, would have the highest probability of being misclassified (either due to lack of experience or to lack of adequate time) by a human annotator. In other words, while RP simulates the perturbation of a training set that might derive from, say, file corruption, TP simulates the

[2] We discovered this fact experimentally in the course of this work. A conversation with Robert Schapire, one of the "fathers" of boosting, later revealed that, while this phenomenon had never been observed before, an *a posteriori* justification can be found for it in the theory that underlies the AdaBoost.MH algorithm, of which MP-Boost is a variant.

Table 1: Percentage of perturbed documents that are false negatives for Reuters-21578 (left) and RCV1-v2 (right).

	Reuters-21578		RCV1-v2	
p	RP	TP	RP	TP
.001	0.77	46.18	2.84	65.43
.010	0.91	19.34	3.18	43.91
.050	0.87	7.32	3.14	22.84
.100	0.86	4.35	3.15	14.96

perturbation that might derive from lack of experience, or lack of care, on the part of the human annotator who has labelled the training set. Unlike in RP, in TP we allow different training examples to be perturbed for different classes $c_j \in C$, since the same document might be controversial, or "borderline", for one class but not for others.

Table 1 illustrates, for each dataset, perturbation technique and perturbation ratio, the percentage of perturbed documents that are false negatives (i.e., positive examples mislabelled as negatives). Of course, for random perturbation this percentage tends to be fairly constant across the different perturbation ratios, since it tends to coincide with the average class frequency of the entire dataset. One insight that we may gain from looking at Table 1 is that in targeted perturbation, while false positives tend to outnumber false negatives, this tendency is increasingly marked as the perturbation rate increases; this is due to the fact that the number of (genuinely) positive examples is limited, so as the number of documents to perturb increases, the number of positive documents that can be perturbed cannot increase proportionally.

4.2 Effectiveness measures

In order to determine which among the three TDC techniques of Section 3 is the best we will measure how good each technique is at ranking $\widehat{Tr}$ in such a way that the perturbed training examples are placed at the top of the ranking. To this end, it seems natural to adopt one of the measures routinely used for evaluating ad-hoc (ranked) retrieval. Of course, ad-hoc retrieval is all about ranking the "good" (i.e., relevant to the information need) examples higher than the bad ones, while TDC aims at ranking the "bad" (i.e., likely misclassified) examples higher than the good ones; but this is of course an inessential difference.

As a measure of ranking quality we will choose *mean average precision* (MAP), which in our context is defined as follows. Let $r_j(\widehat{Tr})$ be the ranking for class c_j, realized according to TDC technique r, of the perturbed training set $\widehat{Tr}$, and let $[r_j(\widehat{Tr})]_k$ be a binary predicate that returns 1 if the example at the k-th position in $r_j(\widehat{Tr})$ is perturbed for class c_j, and 0 otherwise. We define the *precision at n of $r_j(\widehat{Tr})$* as

$$P_n(r_j(\widehat{Tr})) = \frac{1}{n}\sum_{k=1}^{n}[r_j(\widehat{Tr})]_k \qquad (2)$$

We then define the *average precision of $r_j(\widehat{Tr})$* as

$$AP(r_j(\widehat{Tr})) = \frac{\sum_{k=1}^{|\widehat{Tr}|} P_k(r_j(\widehat{Tr}))\cdot[r_j(\widehat{Tr})]_k}{\sum_{k=1}^{|\widehat{Tr}|}[r_j(\widehat{Tr})]_k} \qquad (3)$$

The *mean average precision* (MAP) of TDC technique r on $\widehat{Tr}$ is finally defined as

$$MAP(r(\widehat{Tr})) = \frac{1}{|C|}\sum_{c_j\in C} AP(r_j(\widehat{Tr})) \qquad (4)$$

Aside from a measure of TDC effectiveness we will also need a measure of MLTC effectiveness, so as to determine which effectiveness gains in classification can be obtained if TDC is performed. As a MLTC effectiveness measure that combines the contributions of *precision* (π) and *recall* (ρ) we have used the well-known F_1 function, defined as $F_1 = \frac{2\pi\rho}{\pi+\rho} = \frac{2TP}{2TP+FP+FN}$, where TP, FP, and FN stand for the numbers of true positives, false positives, and false negatives, respectively. Note that F_1 is undefined when $TP = FP = FN = 0$; in this case we take F_1 to equal 1, since the classifier has correctly classified all documents as negative examples. We compute both microaveraged F_1 (denoted by F_1^μ) and macroaveraged F_1 (F_1^M). F_1^μ is obtained by (i) computing the category-specific values TP_i, FP_i and FN_i, (ii) obtaining TP as the sum of the TP_i's (same for FP and FN), and then (iii) applying the $F_1 = \frac{2TP}{2TP+FP+FN}$ formula. F_1^M is obtained by first computing the category-specific F_1 values and then averaging them across the c_j's.

Section 4.4 reports the results of our experiments with the three TDC techniques of Section 3, the two different perturbation techniques, different perturbation ratios, and different datasets Ω.

4.3 The datasets

In our experiments we have used the Reuters-21578 and RCV1-v2 datasets.

Reuters-21578 is probably still the most widely used benchmark in MLTC research[3]. It consists of a set of 12,902 news stories, partitioned (according to the "ModApté" split we have adopted) into a training set of 9,603 documents and a test set of 3,299 documents. The documents are labelled by 118 categories; in our experiments we have restricted our attention to the 115 categories with at least one positive training example.

Reuters Corpus Volume 1 version 2 (RCV1-v2)[4] is a more recent MLTC benchmark made available by Reuters and consisting of 804,414 news stories produced by Reuters from 20 Aug 1996 to 19 Aug 1997. In our experiments we have used the "LYRL2004" split, defined in [10], in which the (chronologically) first 23,149 documents are used for training and the other 781,265 are used for testing. Of the 103 "Topic" categories, in our experiments we have restricted our attention to the 101 categories with at least one positive training example. Consistently with the evaluation presented in [10], (i) also categories placed at internal nodes in the hierarchy are considered in the evaluation, and (ii)

[3] http://www.daviddlewis.com/resources/testcollections/~reuters21578/

[4] http://trec.nist.gov/data/reuters/reuters.html

as positive training examples of these categories we use the union of the positive examples of their subordinate nodes, plus their "own" positive examples.

In all the experiments discussed in this paper stop words have been removed, punctuation has been removed, all letters have been converted to lowercase, numbers have been removed, and stemming has been performed by means of Porter's stemmer. Word stems are thus our indexing units; since MP-Boost requires binary input, only their presence/ absence in the document is recorded, and no weighting is performed.

4.4 Results and discussion

Table 2 reports MAP values obtained by ranking the perturbed training sets by means of the three TDC techniques (CON, NN, COM). Results are reported for the full set of classes and for the 30 most infrequent classes of both Reuters-21578 and RCV1-v2. The reason we pay special attention to the most *infrequent* classes is that they are usually the classes for which standard supervised learning techniques produce the lowest classification accuracy, which means that they are the classes which are most in need of effectiveness improvement, by TDC or other technique: a user might typically engage in TDC for these highly problematic classes and forget about the classes for which high enough accuracy has already been achieved.

In all the experiments MP-Boost has been run with a number S of iterations fixed to 1,000. For the NN technique, as the $sim(\cdot,\cdot)$ measure of inter-document similarity we have used the cosine of the angle between the $tfidf$ vectors of the two documents. For the same technique we have used the value $k = 45$, since in using k-NN as a learning device for TC Yang [17] has found this value to yield the best effectiveness and has found negligible differences between values of $k \in [30, 65]$; we defer careful optimization of the k parameter to further work.

A "trivial" baseline to the results of Table 2 is the expected MAP value of the random ranker (RR). Detailed combinatorial analysis shows that this is equal to

$$MAP(RR(\Omega)) = \frac{m-1}{n-1} + \frac{(n-m)H_n}{n(n-1)} \qquad (5)$$

where m is the number of relevant (in our case: misclassified) examples in the document set Ω, n is the total number of examples in Ω, and H_n denotes the n-th harmonic number (i.e., $H_n = \sum_{k=1}^{n} \frac{1}{k}$). Actual computation of this formula shows that $MAP(RR(\Omega))$ is approximated by $\frac{m}{n}$ (and in an especially accurate way for large values of n), which in our case coincides with the perturbation ratio $p = \frac{m}{|Tr|}$. Since for all of our datasets and perturbation ratios approximating Equation (5) to the third decimal digit exactly yields p, the first column of Table 2 also indicates the trivial baseline for the experiments in the corresponding row.

There are several insights that can be gained from observing the results of Table 2. The first observation is that, since picking training examples at random is the only method one can adopt when wanting to perform TDC, unless equipped with a specific TDC technique such as CON, NN or COM, the improvements that the three TDC techniques display in Table 2 over the baseline of Column 1 is noteworthy.

A second observation is that, with few exceptions and all other things being equal, each technique performs better for random perturbation than for targeted perturbation. This is intuitive, since misclassified training examples inserted at random in the training set tend to be easier to spot; conversely, in targeted perturbation we corrupt the label of borderline examples, which are then much more difficult to identify for *any* technique.

The third observation is that, among the three competing TDC techniques, while there is no clear winner, there is certainly one clear loser, namely, the COM technique, which in almost all situations obtains results inferior (and often radically so) to CON and NN. We think that the reason for the bad performance of COM may be found in the fact that MP-Boost generates a committee of classifiers that are not independent of each other. Indeed, each member $\hat{\Phi}^j_s$ of the committee strongly depends on the previously generated member $\hat{\Phi}^j_{s-1}$, since the former is generated according to the distribution resulting from applying $\hat{\Phi}^j_{s-1}$ to Tr. As a consequence, agreement is probably not something one could reasonably expect from the members of *this* kind of committee, since sharp disagreement may derive from reasons different from a bad label, such as the different emphasis that the different members place, by construction, on a given training example.

Leaving COM aside, we may observe that neither CON nor NN systematically outperform the other. CON tends to be the better technique on the RCV1-v2 dataset, while the situation is less clearcut on Reuters-21578; similarly, CON tends to outperform NN on the full set of classes of each dataset, while when we analyse the behaviour of the two techniques on the 30 most infrequent classes of each dataset there is no clear winner. All in all, both techniques turn out to be respectable contenders, often achieving (sometimes surprisingly) high MAP values in absolute terms.

A fourth insight we can gain by looking at Table 2 is that MAP tends to increase with the perturbation ratio p, and may reach extremely high values for high values of p. This is very good news, since this means that if we have reasons to believe that our training set is extremely low-quality, we know that our time in cleaning it will not be wasted, since these techniques will place almost all the bad examples near the top of the ranking.

Table 3 reports instead the micro- and macro-averaged F_1 values obtained before and after perturbation; this is an indication of the improvement in classification effectiveness one obtains by performing TDC if the original training set contains noise at the perturbation ratios indicated. Results are reported for the full set of classes and for the 30 most infrequent classes of our two datasets.

One insight that this table allows to gain is that random perturbation is usually more damaging to effectiveness than targeted perturbation, and this fact tends to become evident as the perturbation rate increases. That targeted perturbation may have less disruptive effects is intuitive, since TP introduces mislabellings on documents that are likely bor-

Table 2: Mean average precision (MAP) of the three TDC techniques (CON, NN, COM) on the full set of classes (top 4 rows) and on the 30 most infrequent classes (bottom 4 rows) of Reuters-21578 (left) and RCV1-v2 (right). Boldface indicates the best performer for a given combination of perturbation ratio (p), perturbation method, and dataset.

		Reuters-21578						RCV1-v2					
		Random perturbation			Targeted perturbation			Random perturbation			Targeted perturbation		
	p	CON	NN	COM	CON	NN	COM	CON	NN	COM	CON	NN	COM
Full Set	.001	**.596**	.458	.305	**.510**	.369	.152	.232	**.238**	.072	**.357**	.082	.125
	.010	.653	**.771**	.517	**.608**	.525	.206	**.752**	.542	.566	**.519**	.376	.194
	.050	**.968**	.907	.808	**.677**	.621	.301	**.927**	.777	.801	**.672**	.512	.417
	.100	**.973**	.961	.874	**.665**	.634	.449	**.945**	.865	.804	**.658**	.593	.520
30 Infr	.001	.748	**.790**	.401	.648	**.681**	.100	.222	**.225**	.099	**.323**	.101	.104
	.010	.674	**.966**	.599	.581	**.670**	.153	**.702**	.476	.533	**.435**	.375	.275
	.050	.982	**.992**	.812	.647	**.701**	.268	**.896**	.716	.747	**.608**	.427	.479
	.100	.981	**.985**	.886	**.673**	.651	.455	**.919**	.845	.760	**.613**	.523	.588

Table 3: Micro- and macro-averaged F_1 values for the full set of classes (top 5 rows) and for the 30 most infrequent classes (bottom 5 rows) of Reuters-21578 (left) and RCV1-v2 (right) after perturbation.

		Reuters-21578				RCV1-v2			
		Random perturbation		Targeted perturbation		Random perturbation		Targeted perturbation	
	p	F_1^μ	F_1^M	F_1^μ	F_1^M	F_1^μ	F_1^M	F_1^μ	F_1^M
Full Set	.000	.852	.606	.852	.606	.572	.423	.572	.423
	.001	.822	.356	.821	.448	.557	.368	.558	.354
	.010	.583	.227	.632	.254	.348	.224	.441	.324
	.050	.138	.074	.209	.094	.105	.096	.211	.160
	.100	.064	.047	.116	.061	.050	.064	.137	.107
30 Infr	.000	.373	.245	.373	.245	.164	.062	.164	.062
	.001	.190	.114	.139	.137	.102	.044	.038	.035
	.010	.038	.036	.056	.052	.025	.024	.063	.039
	.050	.004	.004	.011	.011	.006	.005	.015	.014
	.100	.002	.002	.006	.005	.005	.003	.010	.008

derline examples anyway, i.e., documents that two human annotators might legitimately label in different ways. Mislabelling them may hurt classification accuracy in the thin region of document space close to the surface that separates the positives from the negatives, but does not affect accuracy elsewhere. Conversely, random perturbation may have effects anywhere in document space, and may seriously mislead the classifiers even on cases that would be clearcut otherwise.

A second observation that immediately jumps to the eye is that the decrease in effectiveness deriving from perturbation is noteworthy even for very modest perturbation rates (e.g., .001), and becomes disastrous even for slightly less modest ones (e.g., .010). For instance, for a .001 targeted perturbation rate removing the mislabellings from the Reuters-21578 training set makes F_1^μ jump

- from .821 to .852 for the full set of classes. This is a 3% relative improvement, that in the '90s has taken years of improvement in TC technology to achieve. This shows that one mislabelled document in a thousand can single-handedly defy the efforts of many TC researchers at improving effectiveness;
- from .139 to .373 for the 30 most infrequent classes, a 168% relative improvement. It is not hard to see why the effect of even a few misclassified training examples on the classification accuracy for infrequent classes can be so large. Given a class with very few positive training examples, mislabelling even one or a handful negatives as positives can severely corrupt the set of positive training examples, while mislabelling even one or a handful of positives as negatives has the double effect of depleting the already slim set of positive examples and confusing the learner by presenting it with negative training documents that are very similar to the remaining positive ones.

These two observations hold to an even higher degree for F_1^M; similar observations also hold for random perturbation and RCV1-v2. For reasons of space we do not separately report the results on the $(|C| - 30)$ most frequent classes of our two datasets. In a nutshell, on these classes the decrease in F_1^μ is very similar to the decrease on the full set of classes (since F_1^μ is mostly influenced by the behaviour on the most frequent classes), while the decrease in F_1^M is smaller than the decrease in the full set of classes (since F_1^M is equally influenced by all the classes in C).

Table 4: Micro- and macro-averaged F_1 values for the full set of classes (top 5 rows) and for the 30 most infrequent classes (bottom 5 rows) of Reuters-21578 with classifiers trained after performing TDC by means of the CON technique with $K = 100$.

		Random perturbation		Targeted perturbation	
	p	F_1^μ	F_1^M	F_1^μ	F_1^M
Full Set	.000	.852	.606	.852	.606
	.001	.846	.466	.850	.498
	.010	.749	.399	.780	.412
	.050	.607	.252	.632	.312
	.100	.173	.090	.213	.208
30 Infr	.000	.373	.245	.373	.245
	.001	.260	.187	.202	.197
	.010	.219	.174	.201	.183
	.050	.077	.064	.080	.072
	.100	.013	.013	.020	.019

Note that Table 3 only gives us a picture of the improvement that might be obtained by cleaning the *entire* training set. Aside from probably being infeasible in many real-world situations, this is something that would defy the purpose of the TDC techniques we have presented. A study should thus be performed that, for any combination of TDC technique, perturbation method, perturbation ratio, and dataset, plots the effectiveness of the classifiers generated after TDC has been performed, as a function of K, the number of top-ranked training examples that the human annotator has double-checked for misclassifications. This is obviously a daunting experimentation, since for each such combination and each value of K the classifiers should be retrained from scratch and the test examples should be reclassified anew. In Table 4 we provide a sample such experiment, in which for different perturbation methods and ratios we test the effectiveness values resulting, on Reuters-21578, from performing TDC by the CON technique and "un-perturbing" the perturbed documents found at the top $K = 100$ positions in the ranking. For instance, with targeted perturbation and $p = .001$, the MAP value of .510 that CON obtains guarantees (see Table 2) that F_1^μ, that perturbation had brought down from .852 to .821 (see Table 3), jumps back to .850, and that F_1^M, that perturbation had brought down from .606 to .448, jumps back to .498.

All these results are indicative of the fact that TDC is an important and cost-effective way of improving accuracy for all the datasets of less-than-perfect quality of annotation.

5. RELATED WORK

Several works have used TDC in learning tasks other than TC, especially within the realm of computational linguistics. For instance, TDC has been applied to POS tagging [1, 4, 5, 12, 18], verb modality identification [11], PP-attachment [1], and word segmentation for East Asian languages [14]. Some of these works use task-independent TDC techniques while others do not. Among the former, [1, 14] use the DIS technique discussed at the end of Section 3, while [12] uses a technique analogous to DIS that exploits the characteristics of SVMs; [5] uses instead a generative probabilistic model based on a mixture of a majority distribution and an anomalous distribution, and for each training example computes the probabilities that the example has been generated by either of the two distributions, deeming the example a misclassified one if the ratio between the two falls below a certain threshold. Other works use instead task-specific techniques; for instance, in a POS-tagging application [4] top-ranks multiple occurrences of the same word that have been classified with different parts of speech in similar linguistic contexts, a technique that is obviously applicable to POS-tagging only and not to tasks such as TC.

To the best of our knowledge the only work that deals with TDC in the context of TC is [7]. The proposed method consists in training an SVM, removing from the training set the support vectors that the SVM has identified, training a naive Bayesian classifier on the modified training set, and reclassifying the removed support vectors with this classifier, declaring mislabelled the support vectors whose original label does not match the newly assigned label. The intuition behind this technique is that if a training example has a wrong c_j-assignment, then it likely ends up being a support vector for the generated classifier. Unlike our techniques, this technique is strictly learner-dependent, since it only works with SVMs as learners. Additionally, the method is only limited to cleaning the support vectors; our method examines (and ranks) instead the entire training set; as a result, experimentally comparing the technique of [7] with ours would be problematic.

All of the works above adopt an *a posteriori* evaluation methodology, i.e., they perform no training set perturbation, and evaluate their techniques by ranking the original training sets and then asking human annotators to look for misclassified examples throughout the first k ranks, thus reporting precision-at-k results. We prefer the *a priori* evaluation methodology, since (i) it allows us to work with different perturbation ratios, thus addressing the fact that different real-world applications may be characterized by different levels of quality in their data; (ii) it is exempt from evaluator bias, which the *a posteriori* methodology especially suffers from when (as is frequently the case) it is the authors themselves that engage in post-checking the results; (iii) it allows to compute MAP, while the *a posteriori* methodology only allows to compute precision for a specific, usually low value of k (i.e., the misclassified items from the $(k + 1)$-st position onwards have no impact on the evaluation); and (iv) it allows one researcher to replicate the results of the other, while the *a posteriori* methodology does not.

Finally, let us note that the COM technique is somehow reminiscent of the *query-by-committee* active-learning method (see e.g., [2]), in which *unlabelled* examples (and not labelled ones, as in our case) are ranked for human annotation in decreasing order of the disagreement among a committee of classifiers that try to classify them. As a measure of disagreement, [2] uses entropy. We have instead proposed using standard deviation, since entropy can only take into account the binary decisions of the various classifiers, and not the real-valued confidence in their decision; conversely, standard deviation can naturally account for predictions expressed as real numbers, and is thus a better fit in our case.

6. CONCLUSIONS AND FUTURE WORK

We have tested three techniques for training data cleaning on two popular MLTC benchmarks, checking their ability at spotting and top-ranking a set of training examples whose class assignment we have purposefully corrupted for experimental reasons. This experimental protocol allows to conveniently study *in vitro* the behaviour of these TDC techniques, and to precisely measure the relative merits of the various techniques by means of evaluation measures, such as MAP, standard in the field of ranked retrieval. Studying three TDC techniques with two different perturbation models, at five different perturbation levels, across two datasets (one of which consisting of more than 800,000 documents), and studying both the quality of the resulting rankings *and* the increase in effectiveness that carrying out TDC may bring about, our work probably qualifies as the first truly-large scale experimentation of TDC in either computational linguistics or IR.

Our experimental results show that two techniques, the confidence-based technique and the nearest neighbours technique, achieve good MAP values across different settings deriving from the choice of different datasets, different class frequency, different perturbation ratios, and different types of perturbation, but also show that neither one clearly outperforms the other. A further result of this paper is that a fourth technique, which had been proposed before and which was specific to boosting-based learners, is equivalent to the confidence-based technique proposed here, which is instead applicable to all learners equipped with a notion of confidence in the classification decision.

Our results also show that TDC is important, since they show that even a single misclassified example in a thousand training examples can bring about deteriorations in effectiveness that are simply noteworthy in general, and are no less than dramatic for the most infrequent classes and for macroaveraged F_1 in general.

Note also that TDC techniques are important not only for *training data* cleaning, but also for cleaning generic sets of labelled text: the very same techniques discussed here might be applied by a human annotator in order to clean a manually annotated text corpus (e.g., the entire RCV1-v2), regardless of the fact that the entire corpus is then going to be used for training a text classifier. For instance, this is useful for cleaning *test* sets, since incorrectly labelled test examples prevent the accurate measurement of effectiveness, but it is also useful for cleaning labelled datasets produced within organizations that entirely rely on manual classification.

This work still leaves some questions unanswered, that might thus be the subject of future research. One interesting question is whether the committee-based technique, that has proven to wildly underperform the other two, might instead perform radically better with classifier committees whose members, unlike those generated by boosting methods, are independent of each other, as is the case for the classifier committees generated by "bagging" methods [3]. As mentioned in Section 3, member independence seems to better conform to the intuition that underlies the COM method.

A second question is whether spotting and correcting a training example misclassified as positive has the same value as spotting and correcting a training example misclassified as negative. While in this paper we have made the simplifying assumption that the two are equally important, future research could address the issue of attributing different importance values to the two cases, thus bringing about the need of evaluating TDC techniques in terms of cost-sensitive evaluation functions such as normalized discounted cumulative gain [9], in place of the cost-insensitive MAP.

Yet a third question arises is we want to compare TDC and active learning as effectiveness-enhancing techniques that attempt to minimize the additional effort requested from a human annotator. Assuming that the annotation of a new unlabelled document requires an effort x times as large as double-checking an existing labelled document (for some $x \in [0, \infty)$), is it more cost-effective to annotate the n unlabelled documents top-ranked by an active learning technique, or to double-check the $x \cdot n$ documents top-ranked by a TDC technique? Presumably, the answer is a function of the perturbation ratio of the training set, with high (resp., low) perturbation ratios making TDC (resp., active learning) more cost-effective. Individuation of the perturbation ratio that acts as a threshold between the two cases would be extremely interesting.

7. REFERENCES

[1] S. Abney, R. E. Schapire, and Y. Singer. Boosting applied to tagging and PP attachment. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC'99)*, pages 38–45, College Park, US, 1999.

[2] S. Argamon-Engelson and I. Dagan. Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360, 1999.

[3] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[4] M. Dickinson and W. D. Meurers. Detecting errors in part-of-speech annotation. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, pages 107–114, Budapest, HU, 2003.

[5] E. Eskin. Detecting errors within a corpus using anomaly detection. In *Proceedings of the 1st Conference of the North American chapter of the Association for Computational Linguistics (NAACL'00)*, pages 148–153, Seattle, US, 2000.

[6] A. Esuli, T. Fagni, and F. Sebastiani. MP-Boost: A multiple-pivot boosting algorithm and its application to text categorization. In *Proceedings of the 13th International Symposium on String Processing and Information Retrieval (SPIRE'06)*, pages 1–12, Glasgow, UK, 2006.

[7] F. Fukumoto and Y. Suzuki. Correcting category errors in text classification. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04)*, pages 868–874, Geneva, CH, 2004.

[8] L. Galavotti, F. Sebastiani, and M. Simi. Experiments on the use of feature selection and negative evidence in automated text categorization. In *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'00)*, pages 59–68, Lisbon, PT, 2000.

[9] K. Järvelin and J. Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval (SIGIR'00)*, pages 41–48, Athens, GR, 2000.

[10] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[11] M. Murata, M. Utiyama, K. Uchimoto, H. Isahara, and Q. Ma. Correction of errors in a verb modality corpus for machine translation with a machine-learning method. *ACM Transactions on Asian Language Information Processing*, 4(1):18–37, 2005.

[12] T. Nakagawa and Y. Matsumoto. Detecting errors in corpora using support vector machines. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 1–7, Taipei, TW, 2002.

[13] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[14] H. Shinnou. Detection of errors in training data by using a decision list and Adaboost. In *Proceedings of the IJCAI'01 Workshop on Text Learning Beyond Supervision*, Seattle, US, 2001.

[15] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear SVMs. In *Proceedings of the 29th ACM International Conference on Research and Development in Information Retrieval (SIGIR'06)*, pages 477–484, Seattle, US, 2006.

[16] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorisation and retrieval. In *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 13–22, Dublin, IE, 1994.

[17] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):69–90, 1999.

[18] M. Yokoyama, T. Matsui, and H. Ohwada. Detecting errors in POS-tagging using ILP. In *Proceedings of the 8th International Conference on Discovery Science (DS'05)*, pages 75–80, Singapore, SN, 2005.

[19] K. Yu, S. Zhu, W. Xu, and Y. Gong. Non-greedy active learning for text categorization using convex transductive experimental design. In *Proceedings of the 31st ACM International Conference on Research and Development in Information Retrieval (SIGIR'08)*, pages 635–642, Singapore, SN, 2008.