Using Micro-Documents for Feature Selection: The Case of Ordinal Text Classification

Stefano Baccianella, Andrea Esuli and Fabrizio Sebastiani

Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: {firstname.lastname}@isti.cnr.it

Abstract. Most popular feature selection (FS) methods for text classification (TC) such as information gain (a.k.a. mutual information), chi-square, and odds ratio, are based on binary information concerning the presence/absence of the feature in each training document. As such, these methods do not exploit a rich source of information, namely, the information concerning how frequently the feature occurs in each training document (*term frequency*). In order to overcome this drawback we break down each training document of length k into k training "micro-documents", each consisting of a single word occurrence and endowed with the same class information of the original training document. This move has the double effect of (a) allowing all the original FS methods to be still straightforwardly applicable, and (b) making them sensitive to term frequency. We study the impact of this strategy in the case of ordinal TC, using four recently introduced FS functions, two SVM-based learning methods, and two large datasets of product reviews. The experiments show that the use of this strategy substantially improves the accuracy of ordinal TC.

1 Introduction

In IR tasks involving supervised or unsupervised learning, such as text classification or clustering, the high dimensionality of the vector space in which textual documents are represented is problematic. For instance, in text classification (TC) many supervised learners, such as neural networks, do not scale well to large numbers of features, and even those learners that do scale well have a computational cost at least linear in the dimensionality of the vector space. While this negatively impacts on efficiency, effectiveness suffers too, since if the ratio of the number of training examples to the number of features is low, overfitting occurs. For all these reasons, several techniques for reducing the dimensionality of a vector space in text learning tasks have been investigated, the main one being *feature selection* (FS – see e.g., [1, 2]). This latter consists in identifying a subset $S \subset T$ of the original feature set T such that $|S| \ll |T|$ (with $\xi = |S|/|T|$ being called the *reduction level*) and such that S reaches the best compromise

between (a) the efficiency of the learning process and of the classifiers (which is, of course, inversely proportional to $|S|$), and (b) the effectiveness of the resulting classifiers.

In TC, the most popular approach to FS is the *filter* approach. By and large, this consists in a greedy strategy in which a real-valued function f is applied to each feature in T in order to compute its expected contribution to solving the classification task. Only the $|S|$ features with the highest f value are retained.

The most popular instances of function f above, such as information gain (a.k.a. mutual information), chi-square, odds ratio, pointwise mutual information, and the like, are based on *binary* information concerning the *presence/absence* of the feature in each training document. For instance, in pointwise mutual information, defined as $PMI(t_k, c_j) = \log_2 \frac{P(t_k,c_j)}{P(t_k)P(c_j)}$, the value $P(t_k)$ is the probability that feature $P(t_k)$ occurs at all in a random training document. As such, PMI and all the other above-mentioned functions do not exploit a rich source of information, namely, the information concerning *how many times* t_k occurs in a given training document (*term frequency*).

In this paper we propose a filter approach to FS for TC which attempts to overcome this drawback. The approach consists in breaking down each training document d_i into $length(d_i)$ training "micro-documents" (hereafter: μ-documents), each consisting of a single word occurrence and endowed with the same class information of the original training document. In this paper we limit our experiments to the case of "ordinal" text classification (see below); we will test this approach in the context of the better known "single-label" and "multi-label" TC in a future paper.

This paper is organized as follows. In Section 2 we present our μ-document-based approach to FS, and we briefly describe (in Section 2.1) the four FS methods for ordinal TC that we will use as testbeds. Section 3 reports the results of experiments we have conducted using two SVM-based learning methods and two large datasets of product reviews. Section 4 concludes by pointing at avenues for future research.

2 Feature Selection for OC based on Training μ-documents

2.1 Feature Selection Methods for OC

Let us fix some terminology and notation. *Ordinal classification* (OC – also known as *ordinal regression*) consists in estimating (from a training set Tr) a *target function* $\Phi : X \rightarrow R$ which maps each object $x_i \in X$ into exactly one of an ordered sequence (that we here call *rankset*) $R = \langle r_1 \prec \ldots \prec r_n \rangle$ of *ranks* (aka "scores", or "labels", or "classes"). The result of the estimation is a function $\hat{\Phi}$ called the *classifier*[1], which we will evaluate on a test set Te. This problem is

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

somehow intermediate between *single-label classification*, in which R is instead an unordered set, and *metric regression*, in which R is instead a continuous, totally ordered set (typically: the set $\mathbb{R}$ of the reals).

Our FS methods will typically consist of (a) scoring each feature $t_k \in T$ by means of a function $Score$ that measures the predicted utility of t_k for the classification process (the higher the value of $Score$, the higher the predicted utility), and, (b) given a predetermined reduction level ξ, selecting the $|S| = \xi \cdot |T|$ features based on their $Score$. The $Score$ function will sometimes be rank-specific (i.e., its form will actually be $Score(t_k, r_j)$), while sometimes it will be global to the entire rankset (i.e., its form will actually be $Score(t_k)$); in the former case, a method for selecting a set of features global to the entire rankset from the rank-specific scores will also be needed.

The FS methods that we use in this paper are the $Var * IDF$, $RR(Var * IDF)$, $RR(IGOR)$ and $RR(AC * IDF)$ methods originally defined in [3] (an extended version of [4]). These functions represent the state of the art in FS for ordinal classification, since the experiments reported in [3] have shown that they substantively outperform the only two other such functions (Var and PRP) ever discussed (to the best of our knowledge) in the ordinal regression literature. For reasons of space we only describe $Var * IDF$, $RR(Var * IDF)$, $RR(IGOR)$ and $RR(AC * IDF)$ concisely; for more mathematical detail and for the intuitions that underlie them see [3].

In the first method ($Var * IDF$), $Score(t_k)$ is computed as

$$Score(t_k) = -(Var(t_k) + \epsilon) * (IDF(t_k))^a \tag{1}$$

where

- $Var(t_k)$ is the variance of the distribution of the training documents containing t_k across the ranks in R;
- ϵ is a small positive constant whose purpose is to prevent the first factor in the multiplication from being equal to zero, which would reward those features that occur only once in Tr;
- $IDF(t_k) = \log_e \frac{|Tr|}{\#_{Tr}(t_k)}$ (where $\#_{Tr}(t_k)$ denotes the number of training documents that contain feature t_k) represents inverse document frequency;
- a is a nonnegative real-valued parameter (to be optimized on a validation set) that allows to fine-tune the relative contributions of $(Var(t_k) + \epsilon)$ and $IDF(t_k)$ to the product.

The $\xi \cdot |T|$ features with the highest $Score(t_k)$ value are retained while the others are discarded.

The second method ($RR(Var * IDF)$) also uses Equation 1 for computing $Score(t_k)$, but does not simplistically choose the $\xi \cdot |T|$ top-scoring features. Instead, it provisionally assigns each feature t_k to the rank $r(t_k)$ closest to the mean of the distribution of the training documents containing it. Then, it performs a *round robin* (RR) step, in which (i) for each rank $r_j \in R$ it sorts the features t_k assigned to r_j in descending order of $Score(t_k)$, and then (ii) it allows

the n ranks $r_1, ..., r_n$ to take turns in picking features, one at a time, from the top of their rank-specific orderings, until $\xi \cdot |T|$ features have been picked.

The third method ($RR(IGOR)$) computes scores via a rank-specific function $Score(t_k, r_j)$. For each feature t_k and for each $j = 1, \ldots, (n-1)$ we define $c_j = r_1 \cup \ldots \cup r_j$ and $\overline{c}_j = r_{j+1} \cup \ldots \cup r_n$ and compute

$$Score(t_k, r_j) = IG(t_k, c_j) = \sum_{c \in \{c_j, \overline{c}_j\}} \sum_{t \in \{t_k, \overline{t}_k\}} P(t, c) \log_2 \frac{P(t, c)}{P(t)P(c)} \qquad (2)$$

We then (i) sort, for each of the ranks in $\{r_1, ..., r_{n-1}\}$, the $|T|$ features in decreasing order of their $Score(t_k, r_j)$ value, and (ii) carry out a round robin step as for method $RR(Var * IDF)$, until $\xi \cdot |T|$features have been picked.

The fourth and last method ($RR(AC * IDF)$) also computes for each feature t_k, via a rank-specific function $Score(t_k, r_j)$, scores for each of the ranks $r_j \in R$ via the function

$$Score(t_k, r_j) = -\left(\frac{\sum_{\{d_i \in Tr \mid t_k \in d_i\}} E(\tilde{\Phi}_j, d_i)}{|\{d_i \in Tr \mid t_k \in d_i\}|}\right) + \epsilon) * (IDF(t_k))^a \qquad (3)$$

where E is an error measure (here taken to be $|\hat{\Phi}(d_i) - \Phi(d_i)|$), $\tilde{\Phi}_j$ is the "trivial" classifier that assigns all documents to the same rank r_j, and IDF, ϵ and a are as in Equation 1. We then (i) sort, for each of the n ranks $r_j \in R$, the $|T|$ features in decreasing order of their $Score(t_k, c_j)$ value, and (ii) carry out a round robin step as in the two previous methods, until $\xi \cdot |T|$ features have been picked.

2.2 Feature Selection via μ-documents

As noted in the introduction, most popular FS functions from the TC literature only use information concerning the presence/absence of feature t_k in training document d_i, and do not use information on the *number of times* t_k occurs in d_i. This is also true of the four methods we have presented in Section 2.1. In fact:

- In the $Var * IDF$ and $RR(Var * IDF)$ methods, $Var(t_k)$ is the variance of the distribution of the training documents containing t_k; that is, only the fact that a training document contains or does not contain t_k is used.
- Concerning $RR(IGOR)$, in Equation 2 the quantity $P(t)$ is the probability that a random training document contains t at all; the number of times t occurs in the document has no impact.
- In the $RR(AC * IDF)$ method, the first factor of Equation 3 depends, both at the numerator and at the denominator, on the training documents that contain t_k at all; the number of times t_k is contained in them has no impact.

We attempt to overcome this drawback by breaking down each training document d_i into $length(d_i)$ training "μ-documents", each consisting of a single word occurrence and endowed with the same class information of the original training

document. The training set Tr is then replaced, for FS purposes only, by the set of the training "μ-documents" obtained from it. All the original FS methods are obviously still applicable after this move: however, these methods are now *de facto* sensitive to term frequency, since a training document d_j belonging to class c_i and containing r occurrences of feature t_k has generated (among others) r training μ-documents containing (only) t_k and belonging to c_i.

The move from training documents to training μ-documents is, as far as FS is concerned, akin to the move, in naïve Bayesian learners, from a multivariate Bernoulli event model (where documents are events) to a multinomial event model (where word occurrences are events). In the context of TC this move was originally discussed in [5]. However, in that case the authors reported that little difference in performance was found when selecting features via the former model rather than via the latter model (no actual effectiveness figures were given, though). Our work may be seen as exporting that idea outside the realm of naïve Bayesian learners, and outside the realm of single-label TC, neither of which has been done before to the best of our knowledge.

Finally, note that after the reduced set of features S has been identified via our FS mechanisms, classifier training proceeds as usual, i.e., by using "regular" training documents; that is, the training documents are broken up into μ-documents only for the purpose of carrying our FS.

3 Experiments

3.1 Experimental setting

The datasets We have tested the proposed method on two different datasets, whose characteristics are reported in Table 1.

Dataset	$\|Tr\|$	$\|Te\|$	$\|Tr_\mu\|$	1 Star	2 Stars	3 Stars	4 Stars	5 Stars
TripAdvisor-15763	10,508	5,255	2,222,578	3.9%	7.2%	9.4%	34.5%	45.0%
Amazon-83713	20,000	63,713	3,399,721	16.2%	7.9%	9.1%	23.2%	43.6%

Table 1. Main characteristics of the two datasets used in this paper. The first three columns indicate the number of training documents, the number of test documents, and the total number of training μ-documents, respectively. The last five columns indicate the fraction of documents that have a given number of "stars".

The first is the TripAdvisor-15763 dataset first used in [6] and consisting of 15,763 hotel reviews from the TripAdvisor Web site[2]. We use the same split between training and test documents as used in [6], resulting in 10,508 documents used for training and 5,255 for test; the training set contains 36,670 unique words.

[2] The dataset is available for download from `http://patty.isti.cnr.it/~baccianella/reviewdata/`

The second dataset is the Amazon-83713 first used in [3] and consisting of 83,713 home electronics product reviews from the Amazon Web site. We use the same split between training and test documents as in [3], resulting in 20,000 documents used for training and 63,713 for test; the training set contains 138,964 unique words. To the best of our knowledge, Amazon-83713 is still the largest dataset ever used in the literature on ordinal TC.

Both datasets consist of reviews scored on a scale from 1 to 5 "stars"; both datasets are highly imbalanced (see Table 1), with positive and very positive reviews by far outnumbering negative and very negative reviews.

Evaluation measures As our main evaluation measure we use the *macroaveraged mean absolute error* (MAE^M) measure proposed in [7], and defined as

$$MAE^M(\hat{\Phi}, Te) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{|Te_j|}\sum_{d_i \in Te_j}|\hat{\Phi}(d_i) - \Phi(d_i)| \tag{4}$$

where Te_j denotes the set of test documents whose true rank is r_j and the "M" superscript indicates "macroaveraging". As argued in [7], the advantage of MAE^M over "standard" mean absolute error (defined as

$$MAE^{\mu}(\hat{\Phi}, Te) = \frac{1}{|Te|}\sum_{d_i \in Te}|\hat{\Phi}(d_i) - \Phi(d_i)| \tag{5}$$

where the "μ" superscript stands for "microaveraging") is that it is robust to rank imbalance (which is useful, given the above-mentioned imbalanced nature of our datasets) while coinciding with MAE^{μ} on perfectly balanced datasets (i.e., datasets with exactly the same number of test documents for each rank).

Learning algorithms We have tested our methods with two different SVM-based learning algorithms for ordinal regression: ϵ-SVR [8], originally devised for linear regression and which we have adapted to solve ordinal regression problems, and SVOR [9], which was specifically devised for solving ordinal regression.

ϵ-support vector regression (ϵ-SVR), is the original formulation of support vector (metric) regression as proposed in [8]; we have used the implementation from the freely available LibSvm library[3]. ϵ-SVR can be adapted to the case of *ordinal* regression by (a) mapping the rankset onto a set of consecutive natural numbers (in our case we have simply mapped the sequence [1 Star, ..., 5 Stars] onto the sequence [1, ..., 5]), and (b) rounding the real-valued output of the classifier to the nearest natural number in the sequence.

SVOR [9] consists instead of a newer algorithm that tackles the ordinal regression problem without using any *a priori* information on the ranks, and by finding $n-1$ thresholds that divide the real-valued line into n consecutive intervals corresponding to the n ordered ranks. The authors propose two different

[3] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

variants: the first (nicknamed SVOREX, for "Support Vector Ordinal Regression with EXplicit constraints") takes into account only the training examples of adjacent ranks in order to determine the thresholds, while the second (SVORIM, for "Support Vector Ordinal Regression with IMplicit constraints") determines each threshold by using all the training examples from all of the ranks. Given that the authors have experimentally shown SVORIM to outperform SVOREX, the former (in the implementation[4] available from the authors of [9]) is the variant we have adopted for our experiments.

Both learning algorithms use the sequential minimal optimization algorithm for SVMs [10], and both map the solution onto the real-valued line. The main difference between them is the use of *a priori* information. In fact, when using ϵ-SVR the user needs to explicitly specify a mapping of the rankset onto a sequence of naturals and to set the thresholds in-between these latter, while SVOR automatically derives all the needed information from the training set.

As the baselines against which to test our μ-documents-based approach we have used the results we have obtained in [3] (on the same datasets and with the same learning algorithms) with the versions based on "regular" training documents of the same FS functions.

We have set the γ and C parameters of both learners to the optimal values that we had obtained in the experiments of [3]; this means that the parameters are optimal for the baselines but not necessarily for the methods proposed here, which lends even higher value to the results obtained by these latter.

Experimental protocol The experimental protocol essentially conforms to that of [3]. As a vectorial representation, after stop word removal (and no stemming) we have used standard bag-of words with cosine-normalized $tfidf$ weighting. We have run all our experiments for all the 100 reduction levels $\xi \in \{0.001, 0.01, 0.02, 0.03, \ldots, 0.99\}$. This results in a substantial experimentation effort, consisting of 2 datasets x 2 learners x 100 reduction factors x 4 FS functions = 1600 train-and-test experiments, which add up to the other 1600 whose results we use as baseline and that had been already presented in [3].

For the $Var{*}IDF$, $RR(Var{*}IDF)$ and $RR(AC{*}IDF)$ methods we have set the smoothing parameter ϵ to 0.1. For the same methods we have (individually for each method) used the optimal values of the a parameter that we had obtained in the experiments of [3]; again, this means that the parameters are optimal for the baselines but not necessarily for the methods proposed here. For $RR(AC{*}IDF)$, the E error measure was taken to be $|\hat{\Phi}(d_i) - \Phi(d_i)|$ (i.e., absolute error), given that it is the document-level analogue of MAE^M.

3.2 Results

The results of our experiments are displayed in Figures 1 to 4. In each such figure the effectiveness (measured via MAE^M) of one of our FS functions is

[4] `http://www.gatsby.ucl.ac.uk/~chuwei/svor.htm`

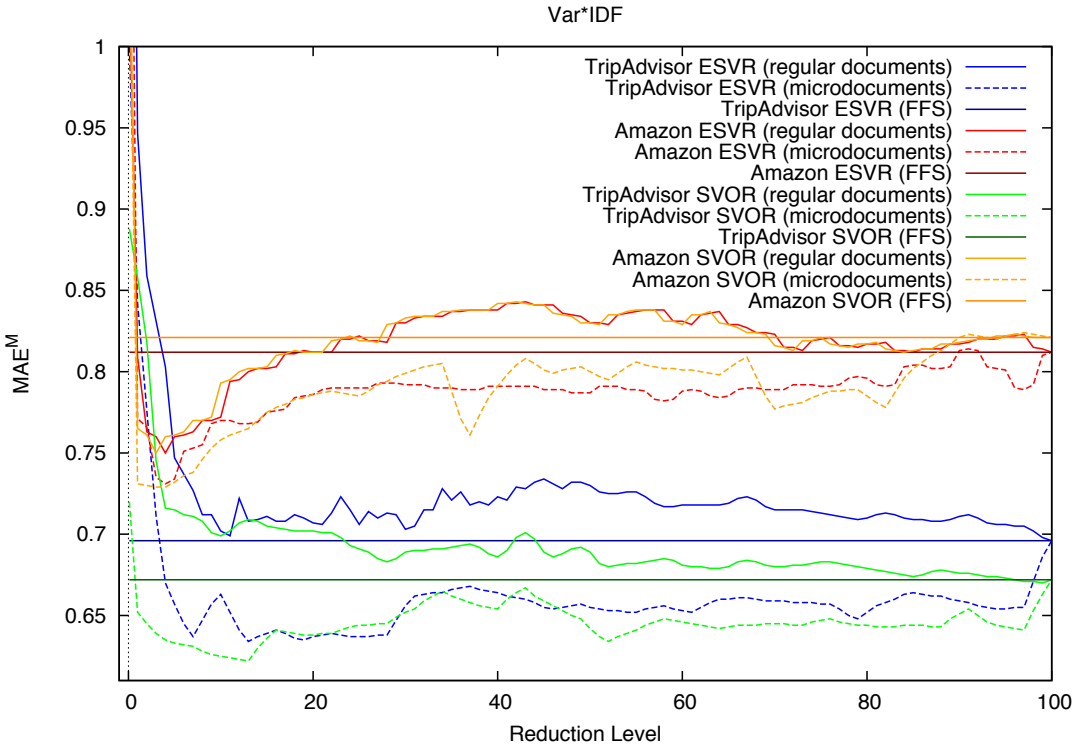

Fig. 1. Results obtained with the two variants (based on "regular" documents and on μ-documents, respectively) of the $Var*IDF$ FS function on the TripAdvisor-15763 and Amazon-83713 datasets with the ϵ-SVR and SVORIM learners. Results are evaluated with MAE^M; lower values are better. "FFS" refers to the use of the full feature set (i.e., $\xi = 1$).

plotted as a function of the tested reduction level; in each figure eight curves are reported, deriving from the choice of two learners, two different datasets and two different interpretations (based on "regular" documents – indicated by solid lines – or on μ-documents – indicated by dotted lines) of the same function. The horizontal line indicates the effectiveness obtained by using the full feature set ($\xi = 1$). Overall, these plots are thus the result of a massive experimental work, consisting of (100 reduction levels $\times$ 2 datasets $\times$ 2 learners $\times$ 4 FS functions =) 1600 new train-and-text experiments (which are additional to the other 1600 that produced out baselines and that were already presented in [3]).

The main observation to be made from these plots is that the use of training μ-documents substantially enhances the accuracy of ordinal TC, since it is practically always the case that the MAE^M values of the μ-document-based versions are better than the corresponding values of the "regular document" -based ver-

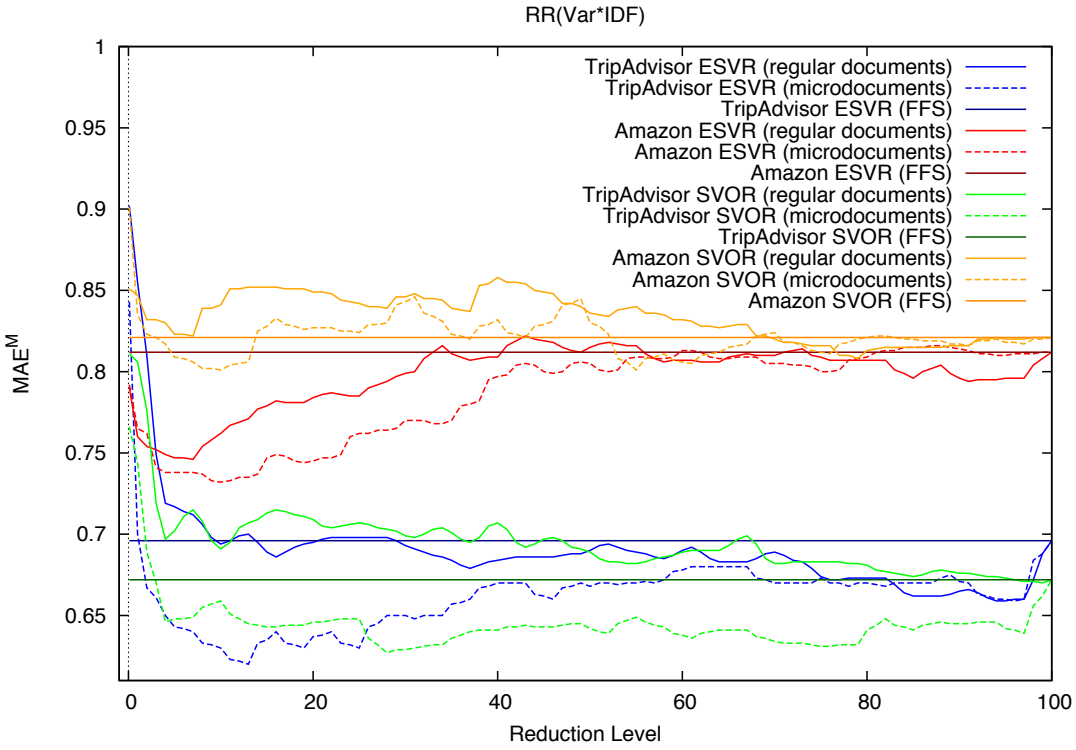

Fig. 2. Same as Figure 1 but with $RR(Var * IDF)$ in place of $Var * IDF$.

sions, irrespective of FS function, dataset, and learner. Table 2 reports MAE^M values as averaged, for a given combination of dataset and learner, across the 100 values of ξ. A further interesting observation that this table allows to draw is that the improvements brought about by the μ-documents technique are much higher for ϵ-SVR than for SVOR; the fact that ϵ-SVR is practically always a better performer than SVOR lends thus to the μ-documents technique even higher value.

4 Further work

One of the problems with the μ-documents-based method we have proposed is that, by transforming each word occurrence into a μ-document and then considering it as a training example, it *de facto* enforces a notion of term frequency in which this latter grows *linearly* with the number $\#(t_k, d_i)$ of occurrences of feature t_k in document d_i, i.e., $tf(t_k, d_i) = \#(t_k, d_i)$. It is instead well-known that the best-performing variants of the tf function are the ones in which $tf(t_k, d_i)$

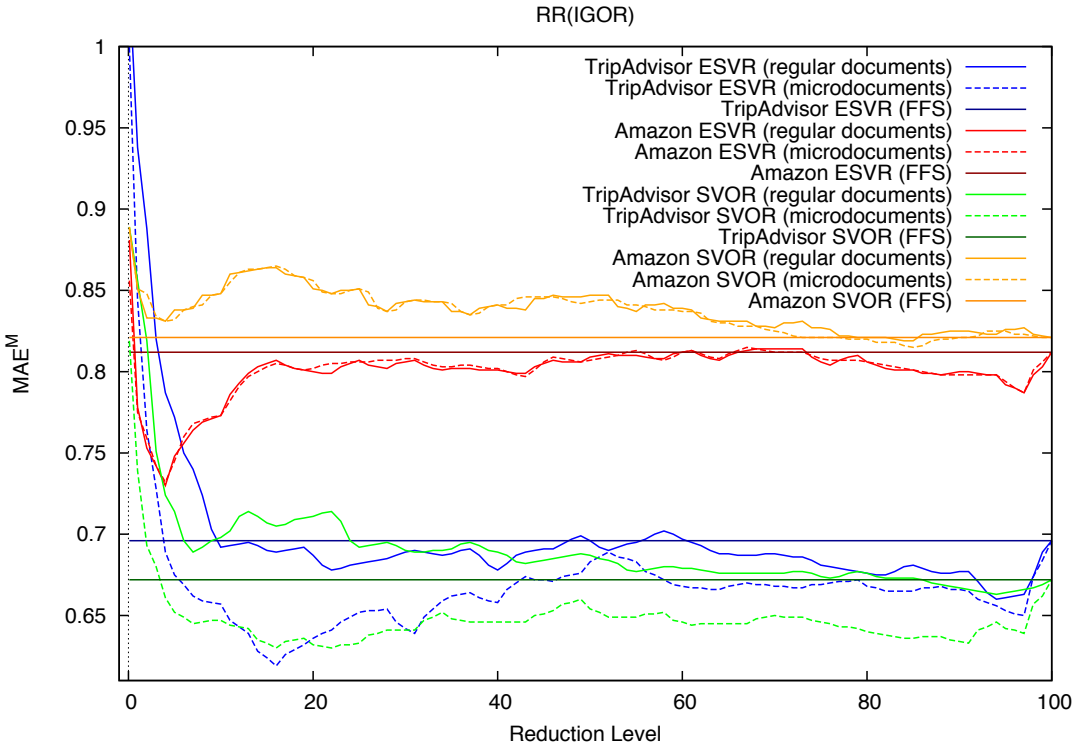

Fig. 3. Same as Figure 1 but with $RR(IGOR)$ in place of $Var * IDF$.

grows *sublinearly* with $\#(t_k, d_i)$ [11, 12]; a simple example is the well-known form

$$tf(t_k, d_i) = \begin{cases} 1 + \log \#(t_k, d_i) & \text{if } \#(t_k, d_i) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

In order to address this shortcoming we are currently implementing a variant of this approach in which, after breaking a training document down into μ-documents, only a part of these μ-documents are retained for FS. Given a nonzero number $\#(t_k, d_i)$ of occurrences of feature t_k in training document d_i, this would allow retaining, e.g., only $1+\lfloor \log \#(t_k, d_i) \rfloor$ (or $1+\lceil \log \#(t_k, d_i) \rceil$) of the $\#(t_k, d_i)$ μ-documents consisting of feature t_k and derived from breaking up d_i, which would *de facto* enforce the notion of term frequency formalized by Equation 6.

References

1. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182

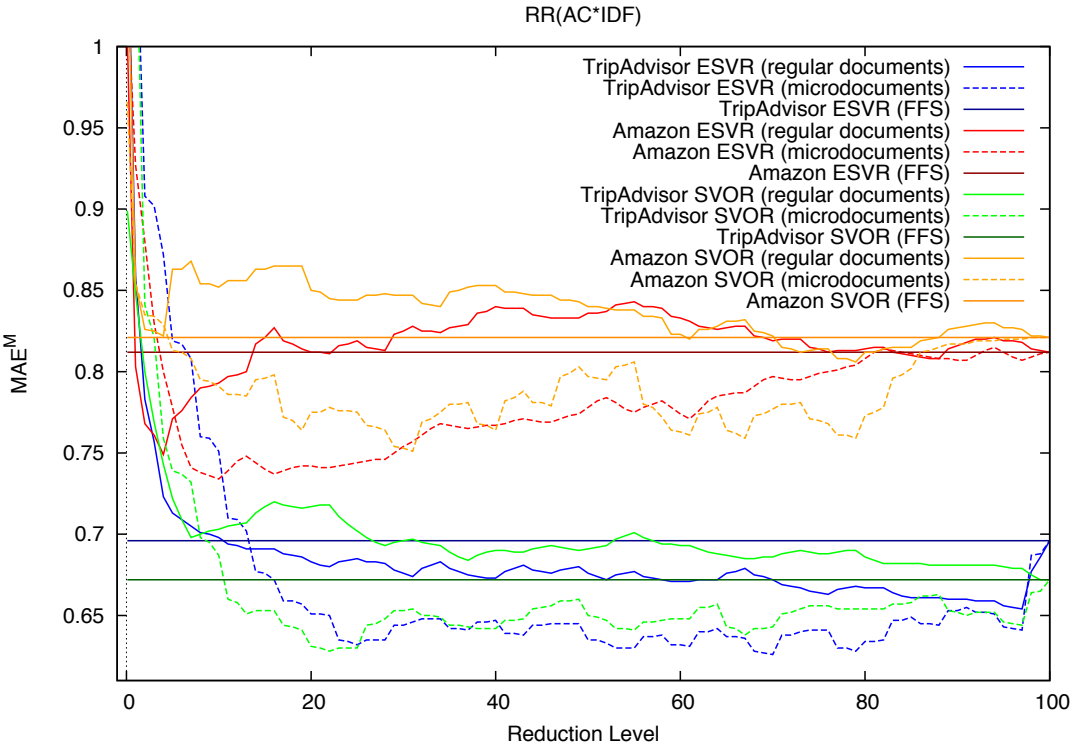

Fig. 4. Same as Figure 1 but with $RR(AC * IDF)$ in place of $Var * IDF$.

2. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proceedings of the 14th International Conference on Machine Learning (ICML'97), Nashville, US (1997) 412–420
3. Baccianella, S., Esuli, A., Sebastiani, F.: Feature selection for ordinal text classification. Technical Report 2010-TR-014, Istituto di Scienza e Tecnologie dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, IT (2010)
4. Baccianella, S., Esuli, A., Sebastiani, F.: Feature selection for ordinal regression. In: Proceedings of the 25th ACM Symposium on Applied Computing (SAC'10), Sierre, CH (2010) 1748–1754
5. McCallum, A.K., Nigam, K.: A comparison of event models for naive Bayes text classification. In: Proceedings of the AAAI Workshop on Learning for Text Categorization, Madison, US (1998) 41–48
6. Baccianella, S., Esuli, A., Sebastiani, F.: Multi-facet rating of product reviews. In: Proceedings of the 31st European Conference on Information Retrieval (ECIR'09), Toulouse, FR (2009) 461–472
7. Baccianella, S., Esuli, A., Sebastiani, F.: Evaluation measures for ordinal text classification. In: Proceedings of the 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA'09), Pisa, IT (2009) 283–287

	TripAdvisor-15763						Amazon-83713					
	ϵ-SVR			SVOR			ϵ-SVR			SVOR		
	RDs	μDs	Δ	RDs	μDs	Δ	RDs	μDs	Δ	RDs	μDs	Δ
$Var * IDF$	0.722	0.658	(-8.84%)	0.818	0.787	(-3.82%)	0.691	0.645	(-6.69%)	0.818	0.790	(-3.51%)
$RR(Var * IDF)$	0.688	0.660	(-4.10%)	0.797	0.787	(-1.36%)	0.694	0.644	(-7.26%)	0.833	0.821	(-1.52%)
$RR(IGOR)$	0.695	0.665	(-4.25%)	0.800	0.800	(-0.00%)	0.689	0.646	(-6.23%)	0.838	0.837	(-0.12%)
$RR(AC * IDF)$	0.680	0.666	(-2.09%)	0.818	0.780	(-4.71%)	0.697	0.662	(-4.99%)	0.837	0.787	(-5.92%)
Average	0.696	0.662	(-4.87%)	0.808	0.788	(-2.48%)	0.693	0.649	(-6.29%)	0.831	0.808	(-2.77%)

Table 2. Average values of MAE^M computed across the 100 different values for ξ; Δ indicates the relative reduction in average MAE^M obtained by replacing regular training documents (RDs) with μ-documents (μDs).

8. Drucker, H., Burges, C.J., Kaufman, L., Smola, A., Vapnik, V.: Support vector regression machines. In: Proceedings of the 9th Conference on Neural Information Processing Systems (NIPS'96), Denver, US (1997) 155–161
9. Chu, W., Keerthi, S.S.: Support vector ordinal regression. Neural Computation **19**(3) (2007) 145–152
10. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C.J.C., Smola, A.J., eds.: Advances in kernel methods: Support vector learning. MIT Press, Cambridge, US (1999) 185–208
11. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24**(5) (1988) 513–523
12. Zobel, J., Moffat, A.: Exploring the similarity space. SIGIR Forum **32**(1) (1998) 18–34