New Generation Computing, 31(2013)47-70
Ohmsha, Ltd. and Springer

StarTrack: The Next Generation (of Product Review Management Tools)

Stefano BACCIANELLA, Andrea ESULI,
and Fabrizio SEBASTIANI
Istituto di Scienza e Tecnologia dell'Informazione
Consiglio Nazionale delle Ricerche
Via Giuseppe Moruzzi 1 – 56124 Pisa, Italy
stefano.baccianella@gmail.com
andrea.esuli@isti.cnr.it
fabrizio.sebastiani@isti.cnr.it

Received 9 May 2011
Revised manuscript received 26 March 2012

Abstract Online product reviews are increasingly being recognized as a gold mine of information for determining product and brand positioning, and more and more companies look for ways of digging this gold mine for nuggets of knowledge that they can then bring to bear in decision making. We present a software system, called StarTrack, that automatically rates a product review according to a number of "stars," i.e., according to how positive it is. In other words, given a text-only review (i.e., one with no explicit star-rating attached), StarTrack attempts to guess the star-rating that the reviewer would have attached to the review. StarTrack is thus useful for analysing unstructured word-of-mouth on products, such as the comments and reviews about products that are to be found in spontaneous discussion forums, such as newsgroups, blogs, and the like. StarTrack is based on machine learning technology, and as such does not require any re-programming for porting it from one product domain to another. Based on the star-ratings it attributes to reviews, StarTrack can subsequently rank the products in a given set according to how favourably they have been reviewed by consumers. We present controlled experiments in which we evaluate, on two large sets of product reviews crawled from the Web, the accuracy of StarTrack at (i) star-rating reviews, and (ii) ranking the reviewed products based on the automatically attributed star-ratings.

Keywords: Product reviews, Sentiment analysis, Sentiment lexicons, Ordinal regression, Text classification.

§1 Introduction

Online product reviews are becoming increasingly available across a variety of Web sites, and are being used more and more frequently by consumers in order to make purchase decisions from among competing products.[12, 22, 38] For example, according to a study performed by Gretzel and Yoo[22] on TripAdvisor,[*1] one of the most popular online review sites for tourism-related activities with over 10 million travel reviews all posted by travellers, travel review readers perceive reviews posted by other consumers as having several advantages over information obtained from travel service providers. Almost two thirds of the readers think that reviews written by consumers contain up-to-date, enjoyable and reliable information. The same study also highlights the fact that, among the users that use the TripAdvisor online booking system, 97.7% are influenced by other travellers' reviews, and among them 77.9% use the reviews as a help to choose the best place to stay. Almost all respondents taking part in this survey answered that reviews (i) are a good way to learn about travel destinations and products, (ii) help with the evaluation of alternatives, and (iii) help to avoid places they would not enjoy. A clear majority of them also think that reviews increase confidence and help reduce risk by making it easier to imagine how a place will be like.[*2]

The importance of harnessing the information contained in online product reviews, and employing it as a tool for decision making, is now widely recognized.[10, 11, 16, 20] Monitoring online product reviews in order to check how a product is perceived, in order to generate sales forecasts, and in order to steer the design, production, and marketing strategies of the company, is also recognized as essential in order to react dynamically to the evolving needs of the market. As a result, software tools that crawl the Web for product reviews, analyse them automatically, and extract from them indicators useful to analysts and researchers, are going to become increasingly important for applications such as brand positioning, revenue forecasting, and the like.

Among the issues that the designers of these software tools need to address are (a) content aggregation, such as in pulling together product reviews from sources as disparate as e-magazines, newsgroups, blogs, and community Web sites; (b) content validation, as in filtering out fake reviews authored by people with vested interests;[24, 32] and (c) content organization, as in automatically ranking competing products of similar type and price in terms of the satisfaction of consumers who have purchased the product before.

We describe a software tool that we have recently built and that addresses a problem related to issue (c), namely, rating (i.e., attributing a numerical score of satisfaction to) consumer reviews based on a fully automatic analysis of their textual content. This is akin to guessing, based on an analysis of the tex-

[*1] `http://www.tripadvisor.com/`

[*2] See also: Andrew Lipsman, *Online Consumer-Generated Reviews Have Significant Impact on Offline Purchase Behavior*, `http://www.comscore.com/Press_Events/Press_Releases/2007/11/Online_Consumer_Reviews_Impact_Offline_Purchasing_Behavior`, November 29, 2007.

tual content of the review, the score the reviewer herself would attribute to the product. This problem arises from the fact that, while some online product reviews (especially those to be found in specialized product review sites such as Epinions.com,[*3] Amazon,[*4] Ratingz.net,[*5] or Rotten Tomatoes[*6]) consist of a textual evaluation of the product *and* a score expressed on some ordered scale of values, many other reviews (especially those to be found in newsgroups, blogs, and other venues for spontaneous discussion) contain a textual evaluation only, with no score attached. These latter reviews are difficult for an automated system to manage, especially when a comparison among them is needed in order to determine, based on the reviews alone, whether product x is considered by the reviewers to be better than product y, or in order to identify the best perceived product in the lot. The availability of tools capable of interpreting a text-only product review and scoring it according to how positive it is, is thus of the utmost importance.

In particular, our system addresses the problem of rating a review when the value to be attached to it must range on an *ordinal* (i.e., discrete) scale. This scale may be in the form either of an ordered set of *numerical* values (e.g., one to five "stars"), or of an ordered set of *non-numerical* labels (e.g., Disastrous, Poor, Good, VeryGood, Excellent). In mathematics, this rating task is usually called *ordinal regression*, or *ordinal classification*.[*7] The difference between numerical and non-numerical values is inessential to our purposes, since we assume a scale of non-numerical labels to be easily mappable onto one of numerical values.[*8] Since rating a product according to a number of "stars" is commonplace for many types of products (including movies, records, wines, hotels, etc.), hereafter we will assume this to be the rating model. We have accordingly called our software system StarTrack.

As hinted above, the basic operation that StarTrack is capable of performing is "star-rating" (i.e., attributing a certain number of stars, e.g., from one to five, to) a product review based on a fully automatic analysis of its textual content. Based on this capability, StarTrack computes the average rating obtained by a given product (as resulting from star-rating different reviews of the product written by different consumers) and ranks all the products that fulfil a given set of constraints (e.g., all stereo amplifiers in the 500 to 800 Euro range; all horror movies released since 2006 to 2008 and produced in the US; etc.) according to

[*3] `http://www.ratingz.net/`

[*4] `http://www.amazon.com/`

[*5] `http://www.epinions.com/`

[*6] `http://www.rottentomatoes.com/`

[*7] Ordinal regression is intermediate between the task of *single-label classification* (in which there is no order defined on the non-numerical labels) and *metric regression* (in which there is a continuous set of labels (typically: the set of real numbers).

[*8] This assumption entails a further assumption, i.e., that the "distances" between two subsequent non-numerical labels are always the same. This assumption may or may not be satisfied, depending on the context (i.e., is the conceptual distance between Poor and Good equal to the distance between VeryGood and Excellent?). However, for the purpose of this paper we will ignore this issue.

the computed average star-rating. This latter ability thus allows a user to rank a set of comparable products by average reviewer satisfaction; it goes by itself that this allows a researcher to monitor consumer attitudes towards a given product / service / brand easily, so that the company may then respond by revising its production and marketing strategies accordingly. How well does product x fare against competing products? How high is brand y positioned in the reviewers' opinions? Did product z soar in the ranking as a result of last month's massive advertising campaign?

The rest of the paper is organized as follows. Section 2 describes StarTrack, analysing its underlying philosophy and describing its main functionality. Section 3 describes the datasets on which we will perform our laboratory evaluation. Sections 4 and 5 present instead the actual results of this evaluation, in which we measure the ability of StarTrack at guessing the correct star-ratings of a set of manually rated reviews (Section 4), and at guessing the correct ranking of the reviewed products (Section 5). Section 6 discusses related work, while Section 7 concludes by discussing the results obtained and their value for market research.

§2 StarTrack: An Automatic Tool for Making Sense of Product Reviews

StarTrack has its roots in disciplines such as information retrieval, machine learning, and computational linguistics; it is outside the scope of this paper to describe its underlying model in detail, and we leave it to the mathematically conscious reader to check[4, 5] for details. In this paper, only a high-level description will be given that mostly tries to appeal to intuition.

StarTrack is based on a supervised machine learning approach, according to which StarTrack learns, from a set of manually star-rated reviews, the characteristics a given review should have in order to be attributed a given number of stars. Therefore, StarTrack does not need to be programmed with explicit rating rules: it only needs to be trained to star-rate reviews through exposure to a representative set of correctly star-rated reviews (therefore called *training reviews*). Figure 1 illustrates the basic process according to which StarTrack works.

StarTrack can thus potentially work as a building block for other larger systems that implement more complex functionality. For instance, given a community Web site containing product reviews whose users only seldom rate their own reviews, StarTrack can be used in order to learn, from the few rated reviews, to rate the others. Given another community Web site containing only unrated product reviews, StarTrack can be used to learn, from rated reviews of different provenance, to rate the Web site's own reviews. And given a Web site that acts as a "meta" review site, i.e., as an aggregator of the reviews contained in other Web sites (prominent examples of such meta-sites are Metacritic[*9] or the

[*9] http://www.metacritic.com/

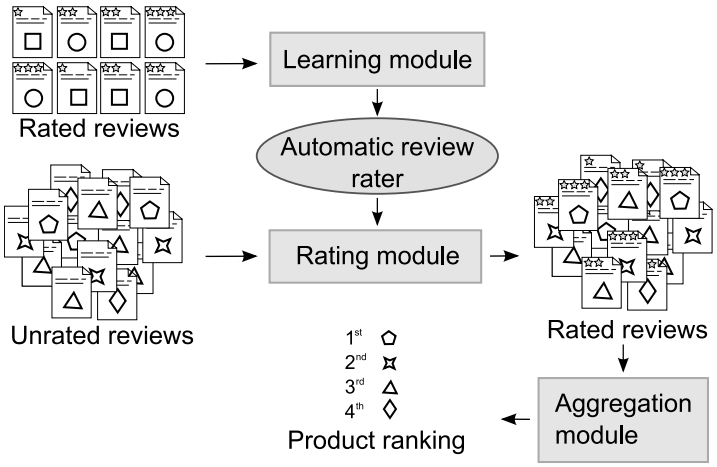

Fig. 1 The Basic Process According to which StarTrack Works

Movie Review Query Engine[*10]), StarTrack can rate all its reviews according to the meta-site's own ordered scale, irrespectively of the possibly different ordered scales used by the contributing sites.

The section that follows takes a slightly more detailed look at the internal workings of StarTrack.

2.1 The Internals of StarTrack: Learning and Feature Selection

The supervised machine learning module upon which StarTrack relies is an implementation (available from `http://www.gatsby.ucl.ac.uk/~chuwei/svor.htm`) of SVORIM,[13] a learning method for "Support Vector Ordinal Regression with IMplicit constraints." Essentially, SVORIM builds a set of binary classifiers via a binary SVM-based learning method, where each such classifier separates the objects in one class from the ones in the class that immediately follows it in the ordering (e.g., separating 3 Stars from 4 Stars objects).

The input to SVORIM consists of representations of the training and test objects as vectors in a high-dimensional space. When classifying product reviews, these representations cannot simply consist of the usual bag-of-words representations commonly used in classifying texts *by topic* (i.e., where a document is represented by the set of words appearing in it, weighted according to frequency considerations). Classifying texts *by opinion* (which is the key contents of reviews) requires much subtler means:[34] two expressions such as "A great hotel in a horrible town!" and "A horrible hotel in a great town!" would receive identical bag-of-words representations, while expressing opposite evaluations of the hotel. As a result, StarTrack is endowed with a linguistic analysis module that, given a review, extracts from it several types of linguistically motivated, higher-order features (hereafter: *LM-features*). As a result, the vector space that SVORIM uses is defined at training time by

[*10] `http://www.mrqe.com/`

1. taking the set consisting of all words and all LM-features (collectively: *features*) occurring in the training set, and
2. performing a phase of *feature selection*, in which only the most promising such features are retained.

Phase 2 is necessary since the set of all words and (especially) all LM-features extracted from a training set of even moderate size could be in the tens, or even in the hundreds of thousands. For instance, in the training set of TripAdvisor-15763, the smaller of the two datasets discussed in Section 4, there are 38,447 unique words and 171,894 unique LM-features; using them all would degrade accuracy (due to overfitting) and efficiency (at both training time and classification time).

For Phase 2 StarTrack relies on $RR(NC * IDF)$, a feature selection technique for ordinal regression that we have proposed in [5], and that in previous experimentation has given consistently good results.[*11] $RR(NC * IDF)$ attributes a score to each feature, after which only the highest-scoring features are retained. This score is a measure of how much the feature is correlated to a given rating. For instance, if a given feature mostly occurs in training reviews rated 4 Stars, with the possible exception of a few training reviews rated 3 Stars or 5 Stars, then it is deemed to be highly correlated with the 4 Stars rating. Such a feature is useful, since when it is discovered in a yet unrated review, it will bring evidence that the rating of this review might be close to or exactly 4 Stars. In other words, $RR(NC * IDF)$ works well on the ordinal regression task because it has been designed with ordinal regression in mind. In particular, the NC component measures how bad an indicator of rating r feature f is, and "bad" is defined in terms of an error measure appropriate for ordinal regression and not for other tasks.

Upon encountering a yet unrated review, StarTrack extracts all the features contained in it, but retains only those which have not been discarded in Phase 2 at training time. These features, weighted according to the usual $tf * idf$ model, constitute then the vectorial representation of the review, that will be fed to the classifiers previously trained by SVORIM.

2.2 The Internals of StarTrack: Sentiment-based Feature Extraction

Our first move away from the simplistic bag-of-words representation has consisted in spotting units of text larger than words that have the potential to be useful additional features. For this purpose, we have defined a module (based on part-of-speech tagging and a simple grammar of phrases – see [4] for details) that (a) extracts complex phrases, such as `hotel(NN) was(Be) very(RB) nice(JJ)`

[*11] The name "$RR(NC * IDF)$" stands for *round robin on negative correlation times inverse document frequency*, and refers to the fact that the technique consists in computing, for each feature, a score resulting from its inverse document frequency and its negative correlation with a given rating, and then choosing the features according to a policy that "round-robins" across the ratings. The interested reader can check [5] for details.

"**Great location**"! We loved the location of this hotel **the area was great** for **affordable restaurants**, bakeries, **small grocers** and near **several good restaurants**. Do not overlook the **lovely church** next door quite a treat! **The rooms were servicable** and some seemed to have been more recently refurbished. Just stay away from room 54 for the money it was a suite **the comfort was not worth** the price, **poor heater** and **horrible shower**, not a single shelf in the bathroom to hold a bar of soap. But 38 also a suite was much nicer. **The basic twin rooms were fine and small** as to be expected. I recommend this hotel overall but do not expect much help from the front desk as all but one of the staff bordered on surly. That was the most disappointing aspect of this **otherwise nice hotel**, **the breakfast was fine** and the breakfast **room was lovely**.

Fig. 2 An example hotel review from the TripAdvisor-15763 dataset discussed in Section 3. The expressions identified by our phrase extraction module are shown in **boldface**.

`and(CC) good(JJ) located(V)`,[*12] and (b) converts them into a canonical form, so as to achieve higher statistical robustness (e.g., the example above is converted into the two canonical forms `very(RB) nice(JJ) hotel(NN)` and `good(JJ) located(V) hotel(NN)`). An example product review, together with the complex phrases that we have extracted from it, is displayed in Fig. 2.

Sentiment analysis[34] plays a major role in the LM-feature extraction phase. Indeed, a product review management tool could hardly do without a sentiment analysis phase, since product reviews are mostly about reviewers *opinions*, which are heavily sentiment-laden. In StarTrack, sentiment analysis is performed by mapping (via the use of sentiment-specific lexical resources) the extracted phrases into ones in which the sentiment conveyed, if any, is made explicit. For example, `very(RB) nice(JJ) hotel(NN)` might be turned into `[Increase] [Positive] hotel(NN)`. This has the advantage that different expressions conveying similar sentiment (e.g., `very(RB) nice(JJ) hotel(NN)` and `very(RB) good(JJ) hotel(NN)`) are mapped into the same LM-feature, whose occurrence statistics thus become more robust than those of the original expressions.

In order to do this, we map each expression in canonical form into an LM-feature by using a sentiment lexicon. We use three different sentiment lexicons (to be discussed below) in parallel; each expression in canonical form may thus give rise to several LM-features, since different lexicons may give rise to different LM-features for the same canonical form expression.

[1] The General Inquirer

The first lexicon we use is the [Positive]/[Negative] subset of the General Inquirer (GI),[40] a set of 1,915 (resp., 2,291) English words marked as having a

[*12] Letters in parentheses denote the part-of-speech tags attributed by our POS tagger, with `NN Be RB JJ CC V` standing for noun, verb "To be", adverb, adjective, conjunction, and generic verb, respectively. Any ill-formed or clumsy English expression in the examples displayed in this paper is genuine, i.e., it appears somewhere in the review datasets we have used for testing StarTrack (see Section 3 for details).

positive (resp., negative) polarity. Examples of positive terms in GI are "advantage," "fidelity" and "worthy," while examples of negative terms are "badly," "cancer," and "stagnant." In GI, words are also marked according to an additional, finer-grained set of sentiment-related tags; some of them denote the magnitude of the sentiment associated with the word, while others denote specific emotions and feelings evoked by the word. For instance, `friendly(JJ)` is not simply described as `[Positive]`, but is described as `[Emot][Virtue][Positive]`, to denote the *emotional* character of this positivity. This allows us to cover the sentiment-carrying expressions that occur in our reviews in a finer-grained way. For each POS-tagged phrase (e.g., `friendly(JJ) staff(NN)`), we generate both a *simple* GI-based LM-feature, which takes only positivity and negativity into account (for the example above, `[Positive] staff(NN)`), and a *complex* one, which takes into account further qualifications of positivity and negativity (for the example above, `[Emot][Virtue][Positive] staff(NN)`). Both types of features have advantages and disadvantages: the former are statistically more robust (since they occur more often) but semantically less informative, while the opposite is true for the latter. We generate both types and let them all compete for a spot high up in the feature ranking generated by the feature selection phase.

[2] The Appraisal Lexicon

The second lexicon we use is what we here call the Appraisal Lexicon, a sentiment lexicon based on "Appraisal Theory"[29] and described in [2]. The Appraisal Lexicon contains 1,939 lexical entries (words qualified by their part of speech); all of them are either sentiment-laden terms or modifiers (i.e., words indicating negation, intensification, etc.). Each sentiment-laden word is described along three dimensions:

- Orientation: determines whether the appraisal is [`Positive`] or [`Negative`];
- Force: describes the intensity of the appraisal being expressed, as one of [`Low`], [`Median`], [`High`], or [`Max`];
- Attitude: specifies the type of appraisal being expressed; [`Appreciation`], [`Affect`], and [`Judgment`] are the main types, which are further specialized into subtypes and sub-subtypes (see Fig. 3).

Words indicating negation are simply mapped into the `[Flip]` category, indicating that the polarity of the expression that follows them must be inverted, while modifiers that act as intensifiers (e.g., `very`) or downtoners (e.g., `little`) are mapped into the `[Increase]` or the `[Decrease]` categories, respectively.

In order to generate the appraisal-enriched LM-features we map each POS-tagged, sentiment-laden word into the sequence of categories that specify the three appraisal-related dimensions of the lexical entry (ignoring only the too common tag [`Median`] of type Force); for instance, `beautiful(JJ) room(NN)` is mapped into `[Median] [Positive] [Quality] room(NN)`, and from this into `[Positive] [Quality] room(NN)`. For any term marked in the appraisal lexicon as a negation (indicated in the lexicon with the label `[Flip]`) we re-

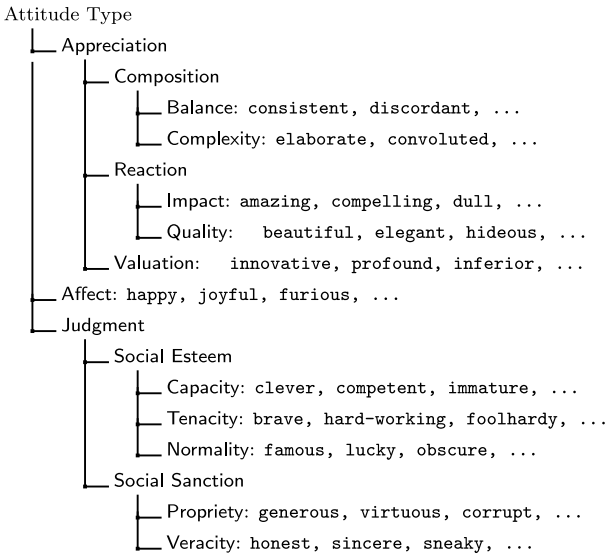

Fig. 3 The Attitude taxonomy, with examples of adjectives having sentimental valence (from [2]).

place the category following it with its inverse (e.g., `[Flip][Positive]` → `[Negative]`, `[Flip][Decrease]` → `[Increase]`). For example, the POS-tagged phrase `not(CC) very(RB) beautiful(JJ) room(NN)` is mapped into `[Flip] [Increase] [Positive] [Quality] room(NN)` and then into the final LM-feature `[Decrease] [Positive] [Quality] room(NN)`.

[3] SentiWordNet

SentiWordNet 3.0 [6] is an automatically generated annotation by sentiment of WordNet 3.0.[19] To each WordNet "synset" s (i.e., set of synonymous word senses), SentiWordNet associates three nonnegative scores $Pos(s)$, $Neg(s)$ and $Obj(s)$ of positivity, negativity and neutrality, respectively, with $Pos(s) + Neg(s) + Obj(s) = 1$.

In WordNet, each of the 117,659 synsets is labelled with a part of speech, and contains one or more words. The same word may appear in more than one synset, reflecting the fact that the same word may have more than one sense. Each sense of a given word is identified by a sense number, in decreasing order of estimated frequency of use in the English language (so that, e.g., the most frequent sense of the noun `bank` is identified as `bank(NN,1)`).

From SentiWordNet we have created a word-level dictionary (that we call SentiWordNet$_w$) in which each POS-tagged word w (rather than each word sense s, as in full-fledged SentiWordNet) is associated to a score $\sigma(w)$ that is meant to indicate its sentimental valence, averaged across its word senses. We have empirically obtained this score by computing a weighted average $\sigma(w) = \sum_i \frac{1}{i}(Pos(s_i(w)) - Neg(s_i(w)))$ of the differences between the positivity and

negativity scores assigned to the various senses $s_1(w), s_2(w), ...$ of w. In this weighted average, the weight is the inverse of the sense number, thus lending more prominence to the most frequent senses.

In order to generate the SentiWordNet-enriched patterns, we have mapped the scores in SentiWordNet$_w$ to a seven-point ordinal scale: [`Strongly Negative`] for words whose SentiWordNet$_w$ scores are in [-1, -.75], [`Negative`] for (-.75, .5], [`Weakly Negative`] (-.5, .25], [`Neutral`] for (-.25, .25), [`Weakly Positive`] for [.25, .5), [`Positive`] for [.5, .75), and [`Strongly Positive`] for [.75, 1].

As a result, an expression such as `good(JJ) room(NN)` is mapped to `[Positive] room(NN)`, since SentiWordNet$_w$ associates a score of 0.514 to `good(JJ)`.

[4] Putting all together

The reason why, in the generation of LM-features, we do not confine ourselves to a single lexicon is that, as apparent from the sections above, each of the three lexicons we use has strengths and limitations, and as such they have the potential to complement each other well. For instance, SentiWordNet has the drawback that it is the result of an automatic annotation, hence it may be of lower quality than the other two lexicons. On the other hand, it has the advantage that its positivity and negativity labels are graded (while they are binary for GI and only coarsely graded for the Appraisal Lexicon), thus providing finer grain, and has the additional advantage of a much larger coverage of the English language than the other two lexicons (a difference of almost two orders of magnitude).

In sum, all the LM-features thus generated (from any of the three lexicons), together with all single words extracted from the training set, are pooled together and are subjected to the feature selection pass described in Section 2.1. This pass is such that only the best features are retained, be they simple words or complex LM-features.

§3 The Datasets

In the next two sections, we present the results of a laboratory evaluation of StarTrack that we have conducted on datasets of real review data. Specifically, we have conducted this evaluation according to a *train-and-test* evaluation methodology, according to which a set of manually rated reviews is split into two non-overlapping sets:

1. A *training set* of reviews, from which StarTrack learns to rate reviews. It is by analysing the reviews in the training set that StarTrack learns the characteristics that a yet unrated review should have in order to have a certain rating. This is called the *training phase* of StarTrack.
2. A *test set* of reviews, on which the ability of StarTrack at correctly guessing the star-ratings of textual reviews, and at correctly ranking products based on the guessed star-ratings, is tested. Specifically, the star-rating manually attached to the test reviews are assumed correct (as is the product ranking deriving from them), and hidden from StarTrack. This

latter tries to guess both the correct star-ratings and the correct product ranking, based on the training received in the training phase. Evaluation consists in checking how closely the true and the predicted star-ratings / product rankings match. Here, the exact meaning of "close match" is mathematically specified by an *evaluation function*; two different such functions need to be employed, one for evaluating the correctness of star-ratings, and one for evaluating the correctness of product ranking.

As the 1st dataset for conducting our experiments, we use the TripAdvisor-15763 dataset that we have assembled ourselves in a previous work [4], and consisting of 15,763 hotel reviews from the TripAdvisor Web site (see Table 1, 1st row). See [4] for a more detailed description of this dataset. We use the same split between training and test reviews of [4], resulting in 10,508 reviews used for training and 5,255 used for test. Altogether, the reviewed hotels are 323 (on average, this means 48.80 reviews per hotel); 6 hotels have a single review while 1 hotel has 368 reviews.

As the 2nd dataset we have chosen a set (that we here call Amazon(MP3)-15071) consisting of 15,071 reviews of MP3 players from the Amazon Web site (see Table 1, 3rd row). Amazon(MP3)-15071 is actually a small subset of the dataset (consisting of more than 5 million reviews from the Amazon site) originally crawled by Jindal and Liu for spam review detection purposes,[24] and consists of *all* the reviews of MP3 players contained in it at the time of crawling. We have randomly picked 9,998 reviews to be used for training, and we use the remaining 5,073 reviews for test. Altogether, the reviewed products are about 1,102 (on average, this means 13.68 reviews per product); 295 products have a single review while one product has 298 reviews.

Both datasets consist of reviews scored on a scale from 1 Star to 5 Stars (scores with "half stars" are not allowed). As clear from Table 1, both datasets are highly imbalanced, with positive and very positive reviews by far outnumbering mild and negative reviews (this is especially true for TripAdvisor-15763); the fact that the ratings of online consumers tend to be positive was noted e.g., in [12], and studied in depth in [23].[*13]

Table 1 Main characteristics of the datasets used in this paper. The 2nd and 3rd columns indicate the number of training and test reviews in the dataset, respectively. Columns from 4th to 8th indicate the fraction of training reviews that have a given number of "stars."

	$\|Tr\|$	$\|Te\|$	1 Star	2 Stars	3 Stars	4 Stars	5 Stars
TripAdvisor-15763	10,508	5,255	4.10%	7.16%	10.01%	34.69%	44.04%
TripAdvisor-15763(s)	5,255	10,508	4.55%	7.19%	9.97%	34.84%	43.40%
Amazon(MP3)-15071	9,998	5,073	16.76%	8.37%	9.33%	26.30%	39.24%
Amazon(MP3)-15071(s)	5,073	9,998	22.14%	8.69%	8.85%	22.85%	37.47%

[*13] Both datasets are available for download from `http://patty.isti.cnr.it\~baccianella/reviewdata/`

§4 Evaluating the Ability of StarTrack at Rating Product Reviews

4.1 The Evaluation Measure

We evaluate the ability of StarTrack at correctly predicting the star-rating of a product review by a mathematical measure called *macroaveraged mean absolute error* (noted MAE^M); this measure, which is more fully discussed in [3], is presented here only briefly.

Essentially, a "good" software system for star-rating reviews is a system that, as often as possible, guesses the correct rating of a review either exactly *or approximately*. This means that, if a given review's true rating is 5 Stars, predicting that its rating is 4 Stars is a better guess than predicting 2 Stars. In other words, evaluating such a system should take into consideration the numerical distance between the true and the predicted rating. This distance is called *absolute error*; for instance, predicting 2 Stars when the true rating is 5 Stars incurs into an absolute error of 3, and predicting 5 Stars when the true rating is 2 Stars also incurs into an absolute error of 3 (that is, overrating and underrating are equally penalized). *Mean* absolute error (MAE) refers to the fact that absolute error is computed for all reviews in the test set that have a certain true rating (say, 2 Stars) and the mean of the absolute errors is computed. *Macroaveraged* MAE refers to the fact that the mean is separately computed as above for each possible rating (e.g., 1 Star, 2 Stars, etc.), and the average of these means is then computed to yield the final value.

MAE^M presents a global view of how accurate a system for guessing star-ratings is. Lower values are better, and the best possible accuracy corresponds to $MAE^M = 0$. However, such a result is not attainable in practice, since the star-rating a hypothetical human annotator would attribute to a given textual review is highly subjective; this is just another facet of the well-known phenomenon of *inter-rater (dis)agreement* (see e.g., [27, pp. 219–250]).[*14]

4.2 Results and Discussion

The results of our experiments are displayed in the 2nd column of Table 2, where we can see that on TripAdvisor-15763, StarTrack obtained a MAE^M result of 0.663. In other words this means that, for an average review, StarTrack's predicted rating is little more than half a star away from the true rating of the review. On the analogous experiment on Amazon(MP3)-15071, StarTrack obtained a MAE^M result of 0.757. While there are certainly margins of improvement, these results are noteworthy, and for several reasons:

[*14] The *worst* possible accuracy corresponds to a MAE^M value equal to the distance between the first and the last class in the set, e.g., $MAE^M = 4$ for the set {1 Star, ..., 5 Stars}. However, only on the datasets in which all of the reviews have a true rating of either 1 Star or 5 Stars a value $MAE^M = 4$ is possible: in a dataset in which at least one review has a true rating of, say, either 2 Stars, 3 Stars or 4 Stars, a value $MAE^M = 4$ is not possible, since the maximum error that can be made on this review is ≤ 3. In general, characterizing the worst MAE^M value possible for a given dataset would require taking, for each review, the maximum possible error a classifier might make on it, and macroaveraging these values.

Table 2 Results obtained on the four datasets of Table 1 as measured by MAE^M. The 2nd column indicates the results obtained by StarTrack, while the 3rd and 4th column indicate the results obtained by using StarTrack with ϵ-SVR and MSVMs, respectively, in place of SVORIM. Lower values are better, **boldface** indicates the best performer.

Dataset	StarTrack	ϵ-SVR	MSVMs
TripAdvisor-15763	**0.663**	0.719	0.844
TripAdvisor-15763(s)	**0.670**	0.721	0.865
Amazon(MP3)-15071	**0.757**	0.802	0.881
Amazon(MP3)-15071(s)	**0.762**	0.809	0.897

1. Because analysing online product reviews is a hard task, since the language used by their authors is often ungrammatical, colloquial, and ridden with typos and abbreviations.
2. Because *rating* product reviews is a hard task. Guessing the number of stars that the reviewer has attributed (or would attribute) to it would be difficult also for a human annotator. It is not clear at all that, given a set of test reviews, the human annotator would obtain a much better level of MAE^M when guessing the true star-ratings of these reviews. One reason for this difficulty is that different reviewers may use very similar language to express different ratings, depending on how prudent or radical they are in their ratings; two reviewers writing approximately the same rebuttal of a given product might rate it 1 Star and 2 Stars, respectively.
3. Because the average accuracy of the system is heavily influenced by ratings (such as 2 Stars and 3 Stars) that are infrequent, and as such have few training reviews. In fact, each of the five different ratings counts the same (by design – see [3)]) when computing MAE^M.

Table 3 provides another look at the same results, in the form of *contingency tables* which display, for each pair of ratings r_1 and r_2, how many reviews whose true rating is r_1 have been rated erroneously as r_2. Table 3 shows that StarTrack performs quite well, as witnessed by the fact that high numbers of reviews tend to be concentrated on the diagonal (which represents perfectly correct decisions - more than half of the total number of test reviews in each dataset are in this category) or in the vicinity of the diagonal (which represents venial errors which any human annotator trying to manually rate the review might potentially make), and by the fact that the cells far away from the diagonal (which represent blatant mistakes) are very scarcely populated.

Finally, one may wonder how much computer time it takes to run StarTrack on these problem sizes. For example, for the experiment on the TripAdvisor-15763 dataset StarTrack required 4 hours 10 mins for the training phase (an average of about 1.5 sec per training review) and 1 hour 24 mins for the rating phase (an average of about 1 sec per test review). This latter figure means that, once trained, StarTrack is capable of rating reviews at a rate of approximately 24,000 reviews per hour, which allows it to easily tackle large or

Table 3 Contingency tables for the experiments on the TripAdvisor-15763 dataset (top table) and on the Amazon(MP3)-15071 dataset (bottom table). Each cell contains the number and percentage of reviews with the given true rating which obtained the given predicted rating. Cells on the diagonal (white) represent perfectly correct decisions; cells near the diagonal (light grey) represent less serious mistakes while cells faraway from the diagonal (dark grey) represent more blatant mistakes.

		Predicted ratings											
		1 Star		2 Stars		3 Stars		4 Stars		5 Stars		**Totals**	
True ratings	1 Star	75	(1.43%)	35	(0.67%)	8	(0.15%)	1	(0.02%)	0	(0.00%)	119	(2.26%)
	2 Stars	103	(1.96%)	150	(2.85%)	82	(1.56%)	24	(0.46%)	0	(0.00%)	359	(6.83%)
	3 Stars	46	(0.88%)	114	(2.17%)	147	(2.80%)	141	(2.68%)	21	(0.40%)	469	(8.92%)
	4 Stars	14	(0.27%)	76	(1.45%)	263	(5.00%)	1080	(20.55%)	600	(11.42%)	2033	(38.69%)
	5 Stars	1	(0.02%)	3	(0.06%)	24	(0.46%)	585	(11.13%)	1662	(31.63%)	2275	(43.29%)
	Totals	239	(4.55%)	378	(7.19%)	524	(9.97%)	1831	(34.84%)	2283	(43.44%)	5255	(100.00%)

		Predicted ratings											
		1 Star		2 Stars		3 Stars		4 Stars		5 Stars		**Totals**	
True ratings	1 Star	724	(14.27%)	173	(3.41%)	54	(1.06%)	24	(0.47%)	10	(0.20%)	985	(19.42%)
	2 Stars	207	(4.08%)	95	(1.87%)	86	(1.70%)	60	(1.18%)	27	(0.53%)	475	(9.36%)
	3 Stars	82	(1.62%)	77	(1.52%)	81	(1.60%)	80	(1.58%)	49	(0.97%)	369	(7.27%)
	4 Stars	99	(1.95%)	88	(1.73%)	187	(3.69%)	456	(8.99%)	460	(9.07%)	1290	(25.43%)
	5 Stars	13	(0.26%)	10	(0.20%)	43	(0.85%)	541	(10.66%)	1347	(26.55%)	1954	(38.52%)
	Totals	1125	(22.18%)	443	(8.73%)	451	(8.89%)	1161	(22.89%)	1893	(37.32%)	5073	(100.00%)

very large rating jobs. All the experiments described in this paper were run on a standard consumer PC, equipped with an Intel Centrino Duo 2×2Ghz processor and 2GB RAM.

[1] The features

It is important to note that all these experiments (and all those described in the next section) were run with a 0.02 "reduction level", i.e., by discarding all but the 2% best features in the feature selection phase. This decision resulted from the fact that different experiments performed on TripAdvisor-15763 with different reduction levels (we tested 0.01, 0.02, 0.03, 0.04, 0.05, 0.10, 0.20, 0.30) revealed that 0.02 was the one yielding the best performance; as a result, we chose 0.02 as the reduction level for all our experiments, including those on Amazon(MP3)-15071.[*15]

Table 4 shows sample LM-features selected by our feature selection algorithm for two different star ratings and for two different reduction levels (0.02 – which is the one we have adopted for StarTrack– and 0.10). For the TripAdvisor-15763 dataset we can note that, while the features selected at reduction level 0.02 all seem strongly correlated to the chosen rating, the correlation at level 0.10 seems intuitively a bit weaker; for example, feature `[Weak positive] room`

[*15] Strictly speaking, the TripAdvisor-15763 results may thus be considered a bit overoptimistic, since the reduction level for the TripAdvisor-15763 experiments was optimized on the TripAdvisor-15763's own test set. In practice these results are instead realistic, since on TripAdvisor-15763 StarTrack delivered a pretty stable performance also for the other reduction levels tested.

Table 4 Sample LM-features selected by our feature selection algorithm for two different star ratings and at two different reduction levels (0.02, 0.10).

TripAdvisor-15763			
0.02		0.10	
1 Star	5 Star	1 Star	5 Star
[Negative] room	[Positive] room	[Negative] room	[Positive] room
not [Positive] hotel	[Positive] hotel	[Negative] hotel	[Positive] hotel
[Negative] hotel	[Increase] [Positive] hotel	[Weak positive] room	[Weak positive] hotel
[Weak negative] staff	very [Positive] room	[Weak positive] hotel	not [Negative] room
[Weak negative] room	[Increase] [Positive] room	not [Positive] room	[Weak positive] staff

Amazon(MP3)-15071			
0.02		0.10	
1 Star	5 Star	1 Star	5 Star
[Negative] thing	[Positive] thing	[Strong] [Negative] colors	great [Weak] [Negative] price
[Negative] price	[Positive] [Virtue] thing	[Weak positive] equalizer	have [Weak negative] options
[Weak Negative] use	[Positive] price	[Decrease] [Positive] MP3	[Weak negative] equalizer
[Negative] player	[Positive] use	not very [Positive] MP3	[Positive] thing
[Weak Negative] player	[Positive] sound	[Weak positive] thing	[Positive] price

is retained for 1 Star, which seems intuitively a dubious choice. In this sample, we can note the effect of the features generated from the different lexicons. For instance, `[Increase] [Positive] room` is the version generated via the Appraisal Lexicon of the feature `very [Positive] room` generated by the two other lexicons.

[2] The sentiment lexicons

In Table 5, we report the "penetration levels" of the lexicons we have used, i.e., the percentages of the set of retained features that come from a given lexicon. Note that the same feature can be generated by more than one lexicon (e.g. `[Positive] room`), so the sum of the percentages for a given experiment may be higher than 100%.

The first observation we can derive from Table 5 is that, while the number of features that do not involve any of the three lexicons is high, as expected, the highest number of sentiment-laden features is obtained via SentiWordNet, which can be explained by its wide coverage. Another observation is that the percentage of features coming from any of the three lexicons is higher for the 0.02 reduction level than for the 0.10 reduction level, which means that the lexicons tend to generate few highly relevant features.

Table 5 Where do the selected features come from? For each dataset and for each reduction level, the table indicates the fraction of the retained features that originate from each lexicon after feature selection has been performed.

	TripAdvisor-15763		Amazon(MP3)-15071	
	0.02	0.10	0.02	0.10
SentiWordNet 3.0	10.1%	8.8%	9.3%	7.8%
General Inquirer	5.6%	4.2%	4.9%	3.9%
Appraisal Lexicon	1.9%	1.1%	1.7%	0.8%
Other	85.4%	89.9%	87.4%	92.0%

4.3 Rating Reviews When Training Data are Scarce

In order to assess how sensitive StarTrack is to the number of training examples, for both datasets we performed another experiment by switching the roles of training and test set: the reviews that belonged to the training set were put into the test set, and vice versa. The resulting datasets (here called Amazon(MP3)-15071(s) and TripAdvisor-15763(s), where "(s)" stands for "switched" – see Table 1, 2nd and 4th rows) have a much higher number of test reviews than the original datasets (which lends higher statistical value to the results), and a much lower number of training reviews (which gives indications as to the ability of StarTrack to perform well even when training reviews are scarce).

In these experiments (see Table 2) we obtained MAE^M results of 0.670 for TripAdvisor-15763(s) and 0.762 for Amazon(MP3)-15071(s), only marginally worse than the 0.663 and 0.757 results obtained on the original datasets. This shows that StarTrack may perform well even when training reviews do not abound.

4.4 Comparing SVORIM with Other State-of-the-art Learning Algorithms

In order to demonstrate that the use of SVORIM is an appropriate learning algorithm to use within StarTrack, we have compared the results obtained with SVORIM to the results obtained with two other state-of-the-art learning algorithms:

- *ϵ-support vector regression* (ϵ-SVR), the original formulation of support vector regression as proposed in [17]. ϵ-SVR can be adapted to the case of ordinal regression by (a) mapping the rating scale 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;
- *Multi-class SVMs* (MSVMs),[37] a learning algorithm originally devised for single-label multi-class classification, which can be adapted to the case of ordinal regression by mapping the rating scale onto an unordered set of classes.

For both algorithms, the implementations from the freely available LibSvm library[8] were used.

The results, reported in Table 2, show that in all of the four datasets used in this paper (a) SVORIM clearly outperforms ϵ-SVR and (b) ϵ-SVR clearly outperforms MSVMs. Fact (a) can be explained by the fact that ϵ-SVR was not originally designed for ordinal regression, while SVORIM was, and confirms earlier findings reported in [13]. Fact (b) can be explained by the fact that MSVMs were not originally devised for ordinal regression *and* by the fact that they are not able to exploit the ordered nature of the rating scale.

§5 Evaluating the Ability of StarTrack at Ranking Products based on Their Reviews

We now move to evaluating the ability of StarTrack at correctly predicting the ranking of a set of products, based on the star-ratings it has attributed itself to their reviews. Generating such a ranking is accomplished by (a) taking all the reviews pertaining to a given product, (ii) averaging their (StarTrack-generated) ratings, and (iii) ranking the products in descending order of their average ratings.

It could be objected that this evaluation exercise simply duplicates that of Section 4, and adds nothing to our understanding of StarTrack. Actually, this is not true, since a software tool x might be better than another software tool y at guessing star-ratings but the opposite might be true for product ranking. To see this, assume there are three reviews A, B and C, with the (true) ratings $((A,3),(B,2),(C,1))$; assume also that software tool x has returned the prediction $((A,3),(B,1),(C,2))$ while software tool y has returned prediction $((A,5),(B,4),(C,3))$. Tool x is obviously better than y in terms of the predicted star-ratings since, assuming that possible ratings range between 1 and 5, x has MAE^M equal to 0.4 while y has MAE^M equal to 1.2. However, y is better than x in terms of the product rankings they have predicted, since y has perfectly guessed the true ranking while x has not. So, evaluating the ability at star-rating and evaluating the ability at ranking are two independently interesting exercises, although related.

We evaluate the ability of StarTrack to correctly rank a set of products via *(normalized) Kendall distance with penalization* 0.5 (noted $K^{0.5}$). This measure, a variant of the well known "Kendall tau rank correlation coefficient,"[25] is presented here only briefly; see e.g., [1,18] for a detailed discussion. $K^{0.5}$ measures the degree to which a predicted ranking of n objects coincides with the true ranking of these objects. Smaller values of $K^{0.5}$ are better, since $K^{0.5}$ returns 0 when the two rankings coincide and 1 when they are the reverse of each other. Essentially, $K^{0.5}$ returns a value proportional to the number of swaps of two objects that are needed to convert the predicted ranking into the true ranking. $K^{0.5}$ also caters for "ties" (i.e., objects that are tied either in the predicted or in the true ranking), discarding from consideration ties in the true ranking and penalizing the predicted ranking for tying two objects that are not tied in the true ranking.

It should be observed, however, that in our case ties are inevitably going to be in high numbers in both the true and the predicted ranking, given that in our datasets, there are only five possible ratings and many, many more products. However, should we restrict our analysis to the products that have at least two test reviews, ties would be more infrequent, since there would now be nine values (including also values such as 1.5, 2.5, 3.5, 4.5) across which the average ratings of the various products would be distributed. And should we restrict our analysis to the products that have at least three, or four, or more, test reviews, ties would be even more infrequent. As a result, we have computed $K^{0.5}$ by restricting our analysis to the products that have at least x test reviews, for all values of x

Table 6 Values of K^p when computed by restricting the ranking to the products that have at least x reviews, for all values of x between 1 and 6 (1 means that all products are considered). Each cell also indicates in parentheses how many products indeed satisfy the corresponding constraint.

Dataset	1		2		3		4		5		6	
TripAdvisor-15763	0.219	(315)	0.179	(294)	0.146	(271)	0.123	(251)	0.102	(233)	0.086	(215)
TripAdvisor-15763(s)	0.156	(321)	0.150	(314)	0.144	(304)	0.128	(289)	0.116	(274)	0.108	(268)
Amazon(MP3)-15071	0.141	(785)	0.054	(495)	0.031	(380)	0.019	(304)	0.013	(261)	0.009	(220)
Amazon(MP3)-15071(s)	0.178	(964)	0.089	(693)	0.055	(546)	0.036	(452)	0.028	(401)	0.019	(344)

between 1 and 6 . The results of this computation are displayed in Table 6.

For instance, for $x = 1$ (i.e., the full set of 785 products is considered) on Amazon(MP3)-15071 StarTrack obtained a $K^{0.5}$ result of 0.141. Given that $K^{0.5}$ ranges between 0 (best) and 1 (worst), this indicates that StarTrack does a very good job at correctly ranking products based on how favourably they have been reviewed. Switching to Amazon(MP3)-15071(s) instead produced a notable deterioration, since $K^{0.5}$ rose to 0.178; this means that the ability of StarTrack at ranking products is more sensitive to the number of available training examples than StarTrack's ability at rating reviews is.

Table 6 also shows that results improve dramatically when x increases; for instance, if we restrict our attention to the products that have at least six test reviews, $K^{0.5}$ is equal to 0.009 for the 220 products of the original Amazon(MP3)-15071 dataset and 0.019 for the 344 products of the switched dataset. Given that perfect ranking performance is denoted by $K^{0.5} = 0$, these values indicate *almost* perfect ranking performance. The reason why StarTrack excels at ranking products that have many test reviews is that the star-ratings that reviewers assign do not always faithfully mirror what the reviewers have written in their reviews. For example, one reviewer may write a review that reads 2 Stars all the way, and instead rate it 1 Star. When there is one single test review for a given product, StarTrack may fall victim of these "mismatches" between the text of reviews and their star-ratings: in fact $K^{0.5}$ takes the ratings attributed by reviewers at face value, and penalizes StarTrack for not complying with them perfectly. When there are two or more reviews for the same product, instead, there is a smaller probability that *all* these reviews suffer from this problem: while the rating attached to the occasional review may be excessively high, the ratings of other reviews of the same product may be reasonable, and the rating of yet another review of the same product may instead be excessively low, thus compensating for the first. In other words, the high number of reviews that a given product has, tends to reduce the influence on StarTrack of occasional "outlier" reviews, and allows StarTrack to perform more correctly.

The results from TripAdvisor-15763 essentially confirm the observations above. However, note that in this case the improvements obtained when x increases are less dramatic, for the simple reason that in TripAdvisor-15763 most hotels have already many reviews anyway; see Table 6, rows 3 and 4 for details.

§6 Related Work

In recent years, the textual analysis of online product reviews has attracted a lot of interest from the scientific community, due to the evident commercial interest that underlies them. Several types of textual analysis have been carried out on product reviews, ranging from summarization,[42, 43] quality assessment,[9] spam detection,[24, 28] and the prediction of review utility for recommendation purposes.[31, 44, 45] In this section, however, we mostly concentrate on reviewing related work on the *rating* of product reviews, focusing on the differences between previous approaches and ours.

Concerning how to represent product reviews vectorially, several works have shown that, when sentiment classification is at stake, sophisticated linguistic analysis (employing e.g., valence shifters,[26, 41] phrases and other types of multiwords,[4, 30] and specialized lexicons[4, 14]) brings about substantial improvements in accuracy with respect to the pure bag-of-words representation; this shows that sentiment classification is radically different from topical classification, in which such improvements have not been observed in the past.

Concerning how to learn to rate product reviews, Blitzer et al.,[7] Dave et al.,[15] and Popescu et al.[36] address rating inference in a simplified way: while the reviews in the training set are labelled according to a five-point scale, the systems these authors describe are only capable of assigning labels in the sets {Positive, Negative}[7, 15] and {Positive, Neutral, Negative},[36] thus "compressing" the original rating scale to a coarser one. This is very different from what we do, since our system is capable of predicting labels on ordinal scales containing an arbitrary number of labels, and is thus capable of adhering to the original rating scale adopted by the data providers.

Unlike the works discussed in the previous paragraph, Pang and Lee[33] address product review scoring with respect to an uncompressed ordinal scale. Unlike ours, their work is exclusively focused on the learning approach to be used, rather than also on the approach for generating vectorial representations for the reviews. They propose and compare experimentally a multi-class SVM classifier, the ϵ-SVR approach of [37], and a meta-algorithm based on a metric labelling formulation of the problem. These authors experiment on a single dataset (not publicly available, which prevented us from doing explicit comparisons with their work), and much smaller than ours. With respect to this work, our approach has the advantage that the learning algorithm that we use (SVORIM) was explicitly devised for performing ordinal regression, which is not true of multi-class SVMs (which are meant to solve single-label multi-class problems) and ϵ-SVR (which was devised to solve metric regression problems).

A related work is that of Goldberg and Zhu,[21] where a semisupervised algorithm is applied that learns to rate product reviews from both rated and unrated training reviews; because of its semisupervised nature their system is not directly comparable to ours, which is instead purely supervised. Also devoted to testing learning algorithms for rating product reviews is the work of Shimada and Endo,[39] which addresses multi-facet review rating on a corpus of Japanese reviews; again, a direct experimental comparison between our work and their

work is impossible since the sentiment lexicons we have used in our work are for the English language only.

Finally, Pekar and Ou[35] rank online hotel reviews in a way similar to ours. The authors manually build a lexicon of expressions conveying either positive or negative sentiment with respect to the domain of hotel reviews. However, their experimental evaluation is weak, since a very small test set of reviews (about 250) is used, and the evaluation simply consists in ranking pairs of reviews according to which one is more positive than the other.

§7 Conclusions

The controlled experiments we have presented show that StarTrack delivers consistently good accuracy in automatically rating product reviews across two very different domains (hotel rooms and MP3 players), and with no reprogramming needed for moving from one to the other. This shows that StarTrack can be ported across a variety of domains and situations, and with no additional costs involved apart from those of obtaining the training reviews. We think this is a noteworthy accomplishment because rating product reviews is a hard task, due to the inherent subjectivity of the rating task and to the often ungrammatical and colloquial nature of the language used in these reviews.

The fact that hundreds of thousands of reviews can be rated in a few hours' computing time shows also that massive data sets can be analysed, with all the ensuing benefits in terms of reliability of the conclusions obtained.

While the accuracy of StarTrack at star-rating product reviews (as witnessed by the MAE^M results) can be considered pretty good, its accuracy at ranking the products (as witnessed by the $K^{0.5}$ results) based on the average of the star-ratings that StarTrack has attributed to them, is no less than excellent. This is especially evident when products that have two or more reviews are ranked. This shows that StarTrack can reliably be used to detect where a given product or brand stands relative to a competitor, or relative to the rest of the pack.

Acknowledgements

This work was carried out in the context of the "Advanced Search Services and Enhanced Technological Solutions for the European Digital Library" (ASSETS) project, funded by the Commission of the European Communities under the ICT Policy Support Programme (ICT PSP).

References

1) Aiolli, F., Cardin, R., Sebastiani, F. and Sperduti, A., "Preferential text classification: Learning algorithms and evaluation measures," *Information Retrieval, 12, 5*, pp. 559–580, 2009.

2) Argamon, S., Bloom, K., Esuli, A. and Sebastiani, F., "Automatically determining attitude type and force for sentiment analysis," in *Proc. of the 3rd Language*

Technology Conference (LTC 2007), pp. 369–373, Poznań, PL, 2007.

3) Baccianella, S., Esuli, A. and Sebastiani, F., "Evaluation measures for ordinal text classification," in *Proc. of the 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA 2009)*, pp. 283–287, Pisa, IT, 2009.
4) Baccianella, S., Esuli, A. and Sebastiani, F., "Multi-facet rating of product reviews," in *Proc. of the 31st European Conference on Information Retrieval (ECIR 2009)*, pp. 461–472, Toulouse, FR, 2009.
5) Baccianella, S., Esuli, A. and Sebastiani, F., "Feature selection for ordinal regression," in *Proc. of the 25th ACM Symposium on Applied Computing (SAC 2010)*, pp. 1748–1754, Sierre, CH, 2010.
6) Baccianella, S., Esuli, A. and Sebastiani, F., "SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in *Proc. of the 7th Conference on Language Resources and Evaluation (LREC 2010)*, Valletta, MT, 2010.
7) Blitzer, J., Dredze, M. and Pereira, F., "Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proc. of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pp. 440–447, Prague, CZ, 2007.
8) Chang, C.-C. and Lin, C.-J., *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/\~{}cjlin/libsvm.
9) Chen, C. C. and Tseng, Y.-D., "Quality evaluation of product reviews using an information quality framework," *Decision Support Systems, 50, 4*, pp. 755–768, 2011.
10) Chen, Y. and Xie, J., "Third-party product review and firm marketing strategy," *Marketing Science, 23, 2*, pp. 218–240, 2005.
11) Chen, Y. and Xie, J., "Online consumer review: Word-of-mouth as a new element of marketing communication mix," *Management Science, 54, 3*, pp. 477–491, 2008.
12) Chevalier, J. A. and Mayzlin, D., "The effect of word of mouth on sales: Online book reviews," *Journal of Marketing Research, 43, 3*, pp. 345–354, 2006.
13) Chu, W. and Keerthi, S. S., "Support vector ordinal regression," *Neural Computation, 19, 3*, pp. 145–152, 2007.
14) Dang, Y., Zhang, Y. and Chen, H., "A lexicon-enhanced method for sentiment classification: An experiment on online product reviews," *IEEE Intelligent Systems, 25, 4*, pp. 46–53, 2010.
15) Dave, K., Lawrence, S. and Pennock, D. M., "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proc. of the 12th International Conference on the World Wide Web (WWW 2003)*, pp. 519–528, Budapest, HU, 2003.
16) Dellarocas, C., Zhang, X. and Awad, N. F., "Exploring the value of online product reviews in forecasting sales: The case of motion pictures," *Journal of Interactive Marketing, 21, 4*, pp. 23–45, 2007.
17) Drucker, H., Burges, C. J., Kaufman, L., Smola, A. and Vapnik, V., "Support vector regression machines," in *Proc. of the 9th Conference on Neural Information Processing Systems (NIPS 1996)*, pp. 155–161, Denver, US, 1997.

18) Fagin, R., Kumar, R., Mahdiany, M., Sivakumar, D. and Veez, E., "Comparing and aggregating rankings with ties," in *Proc. of ACM International Conference on Principles of Database Systems (PODS 2004)*, pp. 47–58, Paris, FR, 2004.

19) Fellbaum, C. ed., *WordNet: An Electronic Lexical Database*, MIT Press, Cambridge, US, 1998.

20) Gao, G., Gu, B. and Lin, M., "The dynamics of online consumer reviews," in *Proc. of the Workshop on Information Systems and Economics (WISE 2006)*, Evanston, US, 2006.

21) Goldberg, A. B. and Zhu, X., "Seeing stars when there aren't many stars: Graphbased semi-supervised learning for sentiment categorization," in *Proc. of the HLT/NAACL Workshop on Graph-based Algorithms for Natural Language Processing*, pp. 45–52, New York, US, 2006.

22) Gretzel, U. and Yoo, K. Y., "Use and impact of online travel review," in *Proc. of the 2008 International Conference on Information and Communication Technologies in Tourism*, pp. 35–46, Innsbruck, AT, 2008.

23) Hu, N., Zhang, J. and Pavlou, P. A., "Overcoming the J-shaped distribution of product reviews," *Communications of the ACM, 52, 10*, pp. 144–147, 2009.

24) Jindal, N. and Liu, B., "Review spam detection," in *Proc. of the 16th International Conference on the World Wide Web (WWW 2007)*, pp. 1189–1190, Banff, CA, 2007.

25) Kendall, M., "A new measure of rank correlation," *Biometrika, 30*, pp. 81–89, 1938.

26) Kennedy, A. and Inkpen, D., "Sentiment classification of movie reviews using contextual valence shifters," *Computational Intelligence, 22, 2*, pp. 110–125, 2006.

27) Krippendorff, K., *Content analysis: An introduction to its methodology*, Sage, Thousand Oaks, US, 2004.

28) Lim, E.-P., Nguyen, V.-A., Jindal, N., Liu, B. and Lauw, H. W., "Detecting product review spammers using rating behaviors," in *Proc. of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, pp. 939–948, Toronto, CA, 2010.

29) Martin, J. R. and White, P. R., *The Language of Evaluation: Appraisal in English*, Palgrave, London, UK, 2005.

30) Min, H.-J. and Park, J. C., "Toward finer-grained sentiment identification in product reviews through linguistic and ontological analyses," in *Proc. of the 47th Annual Meeting of the Association for Computational Linguistics (ACL 2009)*, pp. 169–172, Singapore, SN, 2009.

31) O'Mahony, M. P. and Smyth, B., "Using readability tests to predict helpful product reviews," in *Proc. of the 9th International Conference on "Recherche d' Information Assistée par Ordinateur" (RIAO 2010)*, pp. 164–167, Paris, FR, 2010.

32) Ott, M., Choi, Y., Cardie, C. and Hancock, J. T., "Finding deceptive opinion spam by any stretch of the imagination," in *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pp. 309–319, Portland, US, 2011.

33) Pang, B. and Lee, L., "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proc. of the 43rd Meeting of*

the Association for Computational Linguistics (ACL 2005), pp. 115–124, Ann Arbor, US, 2005.

34) Pang, B. and Lee, L., "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval, 2, 1/2*, pp. 1–135, 2008.

35) Pekar, V. and Ou, S., "Discovery of subjective evaluations of product features in hotel reviews," *Journal of Vacation Marketing, 14, 2*, pp. 145–156, 2008.

36) Popescu, A.-M. and Etzioni, O., "Extracting product features and opinions from reviews," in *Proc. of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, pp. 339–346, Vancouver, CA, 2005.

37) Schölkopf, B., Smola, A. J., Williamson, R. C. and Bartlett, P. L., "New support vector algorithms," *Neural Computation, 12, 5*, pp. 1207–1245, 2000.

38) Sénécal, S. and Nantel, J., "The influence of online product recommendations on consumers' online choices," *Journal of Retailing, 80*, pp. 159–169, 2004.

39) Shimada, K. and Endo, T., "Seeing several stars: A rating inference task for a document containing several evaluation criteria," in *Proc. of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2008)*, pp. 1006–1014, Osaka, JP, 2008.

40) Stone, P. J., Dunphy, D. C., Smith, M. S. and Ogilvie, D. M., *The General Inquirer: A Computer Approach to Content Analysis*, The MIT Press, Cambridge, US, 1966.

41) Wei, W., Gulla, J. A. and Fu, Z., "Enhancing negation-aware sentiment classification on product reviews via multi-unigram feature generation," in *Proc. of the 6th International Conference on Intelligent Computing (ICIC 2010)*, pp. 380–391, Changsha, CN, 2010.

42) Yang, J.-Y., Kim, H.-j. and Lee, S.-g., "Feature-based product review summarization utilizing user score," *Journal of Information Science and Engineering, 26, 6*, pp. 1973–1990, 2010.

43) Zhan, J., Loh, H. T. and Liu, Y., "Gather customer concerns from online product reviews - a text summarization approach," *Expert Systems with Applications, 36, 2*, pp. 2107–2115, 2009.

44) Zhang, R. and Tran, T. T., "An information gain-based approach for recommending useful product reviews," *Knowledge and Information Systems, 26, 3*, pp. 419–434, 2011.

45) Zhang, Z. and Varadarajan, B., "Utility scoring of product reviews," in *Proc. of the 15th ACM International Conference on Information and Knowledge Management (CIKM 2006)*, pp. 51–57, Arlington, US, 2006.

Stefano Baccianella: He has been with the Italian National Research Council as a research associate since 2009 to 2011, and is now an entrepreneur. He was the recipient of the 2009 Franco Denoth Award for his thesis work on mining product reviews. He has published several research articles at international conferences on themes related to product review mining, ordinal regression, and text classification.

Andrea Esuli, Ph.D.: He has been with the Italian National Research Council as a research associate (since 2005) and then as a researcher (since 2010). He holds a Ph.D. in Information Engineering from the University of Pisa. He is the recipient of the 2010 Cor Baayen Award, from the European Research Council for Informatics and Mathematics. He has authored several scientific articles in international journals and conferences in the fields of information retrieval, computational linguistics, text classification, opinion mining and content-based image search.

Fabrizio Sebastiani: He has been with the Italian National Research Council as a researcher (since 1988) and then as a senior researcher (since 2002). He is the co-editor-in-chief of Foundations and Trends in Information Retrieval (Now Publishers), an associate editor for ACM Transactions on Information Systems (ACM Press) and AI Communications (IOS Press), and a member of the editorial boards of Information Retrieval (Kluwer) and Information Processing and Management (Elsevier). He has authored several scientific articles in international journals and conferences in the fields of information retrieval, machine learning, computational linguistics, text classification and opinion mining.