Expert Systems with Applications 40 (2013) 7212–7219

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Customized crowds and active learning to improve classification

Joana Costa [a,b,*], Catarina Silva [a,b], Mário Antunes [c,d], Bernardete Ribeiro [a]

[a] *Department of Informatics Engineering, Center for Informatics and Systems of the University of Coimbra (CISUC), Portugal*
[b] *School of Technology and Management, Computer Science Communication and Research Centre, Polytechnic Institute of Leiria, Portugal*
[c] *Center for Research in Advanced Computing Systems (CRACS), Portugal*
[d] *School of Technology and Management, Polytechnic Institute of Leiria, Portugal*

ARTICLE INFO

Keywords:
Crowdsourcing
Active learning
Classification

ABSTRACT

Traditional classification algorithms can be limited in their performance when a specific user is targeted. User preferences, e.g. in recommendation systems, constitute a challenge for learning algorithms. Additionally, in recent years user's interaction through crowdsourcing has drawn significant interest, although its use in learning settings is still underused.

In this work we focus on an active strategy that uses crowd-based non-expert information to appropriately tackle the problem of capturing the drift between user preferences in a recommendation system. The proposed method combines two main ideas: to apply active strategies for adaptation to each user; to implement crowdsourcing to avoid excessive user feedback. A similitude technique is put forward to optimize the choice of the more appropriate similitude-wise crowd, under the guidance of basic user feedback.

The proposed active learning framework allows non-experts classification performed by crowds to be used to define the user profile, mitigating the labeling effort normally requested to the user.

The framework is designed to be generic and suitable to be applied to different scenarios, whilst customizable for each specific user. A case study on humor classification scenario is used to demonstrate experimentally that the approach can improve baseline active results.

1. Introduction

The World Wide Web created a deluge of data where anyone can publish and search for information. One of the most important drawbacks of this paradigm is the inability to perceive information as relevant, important, accurate or tuned to ones preferences. This problem influences not only the active user that promptly tries to find explicit information, but also the passive user that can be triggered to buy a given product if the correct ad is placed according to its preferences. To tackle this sort of problems, the area of *recommendation systems* has been emerging in the last few years as an active research area in the fields of machine learning and data mining, focusing on customizing models to fulfil user information needs based on personal preferences (Konstan & Riedl, 2012).

Model customization can be specially complex in dynamic environments. In these situations, the learning algorithm needs to cope with the perception of variations and adapt itself accordingly. Although major alterations in classification problems are due to time, other types of drifts may occur, like those related to context variations promoted by different users in recommendation systems. In this case, a model can be fitted to recommend a book to a group of people that share common interests or the same cultural background, but fails to acknowledge the recommendation in a different context. The customization of a learning model is particularly important in recommendation applications where the environment changes, specially when we consider Internet users with so many cultural, educational and geographical differences.

One of the most simple and common techniques used to customize a model is *user profiling* (Webb, Pazzani, & Billsus, 2001). The model needs to previously acquire a certain number of scenarios in which each user can be placed according to its preferences. User profiling can be carried out by defining geographic, age or language boundaries. Although the definition of these boundaries can be suited to some problems, like in news recommendation systems where geographic boundaries are well defined, they fail to deal with more intrinsically subjective problems, like book or movie recommendation systems.

A challenging solution is to use *crowdsourcing* (Howe, 2006; Surowiecki, 2004), an emergent distributed classification method in which a crowdsourcer submits a complex task to groups of people, termed *crowds*, in order to obtain different solutions for further analysis and evaluation. The main idea behind crowdsourcing is to

* Corresponding author at: School of Technology and Management, Polytechnic Institute of Leiria, Portugal. Tel.: +351 244 820 300; fax: +351 244 820 310.

E-mail addresses: joana.costa@ipleiria.pt, joanamc@dei.uc.pt (J. Costa), catarina@ipleiria.pt, catarina@dei.uc.pt (C. Silva), mario.antunes@ipleiria.pt, mantunes@dcc.fc.up.pt (M. Antunes), bribeiro@dei.uc.pt (B. Ribeiro).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.06.072

use the low cost workforce of the users that already introduced their feedback to the recommendation system in order to deal with the newly seen users, learning from previously seen contexts. The crowdsourcing paradigm has been enabled by Web technologies. Its applications use a distributed computer infrastructure and cloud computing facilities provided by the Internet (Brabham, 2008).

In crowdsourcing the distributed problem solving model is based on interoperability between humans (crowd) and computers (evaluation analysis) that work together to solve complex tasks, like those related to annotation, recommendation and classification of contextual examples.

Regarding classification settings, the crowd has to deal with intrinsically subjective tasks, that usually include contextual, semantic and sentiment analysis. However, the classification obtained for a given example differs according to the heterogeneous background (e.g social, cultural, emotional and scientific) expressed by the crowd members. That is, the overall classification provided by the crowd for a contextual example is the result of a different and even opposite classification each member individually contributed.

The *non-expert knowledge* that can be found in such a heterogeneous crowdsourcing scenario, which produces distinct and unrelated examples, may therefore be a source of valuable input for learning systems based on more traditional machine learning methods, e.g. kernel-based methods (Silva & Ribeiro, 2008), such as Support Vector Machines (SVM) (Lebrun, Charrier, Lezoray, & Cardot, 2008; Vapnik, 1999).

Another attractive feature of crowdsourcing is its ability to deal with dynamic context drift through time. For example, applications that classify *trends* through time in a particular domain, like musical interests, jokes or news interest. In such dynamic and real world scenarios, the learning systems are able to adjust through time to dynamic variations of concept (*drifts*), according to the classification made by the crowd at each given moment. The challenges are multiple in these scenarios. Firstly, there are different types of drift, namely sudden, gradual, incremental or recurrent drifts (Attar, Chaudhary, Rahagude, Chaudhari, & Sinha, 2012; Minku, White, & Yao, 2010). For instance in the sudden drift, the identification of the change can be easier than in gradual drift, as initially the change can be confused with noise. A learning system can be suited to identify a sudden drift but may fail when gradual drift is present, specially if both occur in the same scenario.

Dynamic contexts, such as the ones described so far, usually lack enough labeled data often hindering classification performance. To tackle this issue active learning methods allow us to design learning algorithms that may effectively filter or choose a subset of examples for being further labeled by a supervisor, termed *oracle* or *teacher*. The reason for using active learning is two-fold: to expedite the learning process and thus to reduce the labeling efforts required by the supervisor (Baram, El-Yaniv, & Luz, 2004); to allow each user to define personal labels and then to build upon a customized learning model that better fits his preferences.

In this work we propose a framework to deal with customization in recommendation systems using crowd-based non-experts. Three different approaches are proposed to improve over the active learning strategy previously proposed and validated by the authors in Costa, Silva, Antunes, and Ribeiro (2011a, 2011b, 2011c). The proposed framework allows non-experts classification performed by crowds to be used to substitute the user profile definition, mitigating the labeling effort normally requested to the user. A case study classification scenario is used to test and validate our efforts, even though the framework is designed to be generic and applicable to different scenarios.

The important analysis to be carried out in this paper is to determine whether specially chosen crowds, that is tuned crowds, are able to retrieve customized user preferences. This can be particularly relevant in real world problems in which it may be unfeasible to have an assertive supervisor.

The rest of the paper is organized as follows. In Section 2 we introduce the necessary background to our work, namely the definition of crowdsourcing, including the discussion of its applicability in learning systems and the discussion of dynamic environments characteristics. Section 3 presents the proposed framework, followed by experimental setup and results in Sections 4 and 5 respectively. Finally, Section 6 presents some conclusions and future work.

2. Background

2.1. Crowdsourcing

In this section we present the background on crowdsourcing, which constitutes the generic knowledge for understanding the approach proposed in this paper. We further discuss the applicability of crowds as a source of non-expert knowledge in learning systems.

2.1.1. Introduction

Since the seminal work of Surowiecki (2004), the concept of *crowdsourcing* has been expanded, mainly through the work of Howe (2006), where the term crowdsourcing was definitely coined. The underpinning idea behind crowdsourcing is that, under the right circumstances, groups can be remarkably intelligent and efficient. Groups do not need to be dominated by exceptionally intelligent people to be smart, and are often smarter than the smartest individual in them, that is, their decisions are usually better than the decisions of the brightest party. As an example, if you ask a large enough group of diverse, independent people, to predict or estimate a probability, and then average those estimates, the errors each one of them makes in coming up with an answer will cancel themselves out, i.e., virtually anyone has the potential to plug in valuable information (Greengard, 2011; Surowiecki, 2004).

Surowiecki (2004) identified four conditions that characterize wise crowds:

1. **Diversity of opinion**, as each person should have some private information, even if it is just an eccentric interpretation of the known facts. For example, if a crowd of individuals think in the same exact way, they are unable to provide the variability that is needed to cancel the errors each one make.
2. **Independence**, related to the fact that people's opinion is not determined by the opinions of those around them, as persuasive individuals can sway others to think in a certain way and null the diversity of opinion.
3. **Decentralization**, in which people are able to specialize and draw on local knowledge. Otherwise, the centralization can narrow and guide the course of information, turning the crowd less wise.
4. **Aggregation**, related to the existing mechanisms for turning private judgments into a collective decision. By using multiple sources to provide a collective decision, mechanisms that combine information are required.

Besides the intelligent use of a group, there is another noteworthy advantage on using crowdsourcing, as there are tasks that are notoriously difficult for an algorithm to perform and quite simple for humans, like speech or image recognition, language understanding, text summarization and labeling (Barr & Cabrera, 2006). Taking advantage of these inherent capabilities, many crowdsourcing platforms emerged, such as the now widely used Amazon Mechanical Turk and Yahoo! Answers. A growing number of real-world problems have also taken advantage of this technique, such as Wikipedia, Firefox or Linux. In the next section, we will address some known applications of crowdsourcing based on these approaches.

2.1.2. Applications

Due to its promising benefits, crowdsourcing has been increasingly studied for the last few years, being the focus of science and research in many fields like biology, social sciences, engineering, and computer science, among others (Leimeister, 2010). In computer science, and particularly in machine learning, crowdsourcing applications are booming. In Tarasov and Delany (2010) crowdsourcing is used for the classification of emotion in speech, by rating contributors and defining associated bias. In Welinder and Perona (2010) people contribute to image classification and are rated to obtain cost-effective labels. Another interesting application is presented in Chen, Hsu, and Liao (2011), where facial recognition is carried out by requesting people to tag specific characteristics in facial images. In Franklin, Kossmann, Kraska, Ramesh, and Xin (2011) crowdsourcing is used to process queries that neither database systems nor search engines can adequately answer, like ranking pictures by subject areas. Another application in data management research field is presented in Marcus, Wu, Karger, and Madden (2011), where Amazon's Mechanical Turk is used to write SQL-like queries to retrieve data that can not be achieved by a relational model. Crowdsourcing is also used in mobile devices like in Burguera, Zurutuza, and Nadjm-Tehrani (2011), where a platform was built to detect and prevent the spread of malware in android-based systems.

There have also been some attempts to use crowdsourcing in complex problems. In Kittur, Smus, Khamkar, and Kraut (2011) a framework is developed to support the coordination dependencies involved in complex and interdependent tasks from many small contributions in crowdsourcing markets. A small set of primitive tasks are identified, namely partition, map and reduce tasks. In Negri, Bentivogli, Mehdad, Giampiccolo, and Marchetti (2011) the main idea is to create a corpora of cross-lingual textual entailment by dividing the associated complex process into small (and simpler) processes. The problem is separated into the creation and annotation of a monolingual textual entailment corpora and then the multilingual dimension.

There are also a few applications of crowdsourcing for text classification. In Brew, Greene, and Cunningham (2010) economic news articles are classified using supervised learning and crowdsourcing. In Costa et al. (2011b) crowdsourcing is used to improve humor classification.

2.1.3. Crowdsourcing as active non-expert knowledge

The key idea behind active learning is that a machine learning algorithm can achieve greater accuracy with fewer training label samples if it is allowed to choose the data from which it learns, which can be helpful in unbalanced settings (Florido, Pomares, & Rojas, 2011). An active learner may pose queries, usually in the form of unlabeled data instances to be labeled by a supervisor (Settles, 2010).

The reason for using active learning is mainly to expedite the learning process and to reduce the labeling efforts required by the supervisor. Active learning is therefore well-motivated in many modern machine learning problems where data may be abundant but labels are scarce or too expensive to obtain (Baram et al., 2004; Settles, 2010).

Active learning methods can be grouped according to the selection strategy, as being *committee-based* and *certainty-based* (Silva & Ribeiro, 2007). In the first group the active examples combine the outputs of a set of committee members, by determining those in which the members disagree the most as the candidates to be labeled (McCallum & Nigam, 1998). The certainty-based methods try to determine the most uncertain examples and point them as active examples to be labeled. The certainty measure depends on the learning method used.

Crowdsourcing and active learning can be successfully combined. Crowdsourcing may enlist a multitude of humans that can label active learning examples. In Zhao, Sukthankar, and Sukthankar (2011) these techniques are used for activity recognition using body-worn inertial sensors by labeling segmented video clips of cooking activities. In Costa et al. (2011c) an empirically evaluation of the performance of a baseline SVM is proposed when active learning examples are chosen and made available for classification to a crowd in a web-based scenario.

2.2. Dynamic environments

In this section we present the background on dynamic environments, that are further detailed in their context and nature. We also discuss the importance of dynamics in recommendation systems.

2.2.1. Introduction

In the context of Internet online users, time can play an important role. A typical example is the prediction of an email importance in our mailbox, as the importance we give to an email changes over time. That is, today we may be involved in a project and emails referring that project are important but, in a couple of months, when we possibly have embraced another project, those emails are no longer relevant. Another example is the pattern of customer's buying preferences that also changes over time. As an example, in winter customers tend to buy warmer clothes while in summer customers prefer fresh ones. The preference pattern can thus change according to the weather, which may depend on complex factors that can be infeasible to predict. This type of problems face additional challenges as they are set in dynamic environments, also called *non-stationary environments*.

Concepts in dynamic environments are dependent on some *hidden context*, not given explicitly in the form of predictive features, with the ability to induce more or less radical changes in the target concept (Widmer & Kubat, 1996). Informally, it refers to a variation of the underlying data distribution that defines the concept to be learned, for which the decision boundary is different from the previously seen examples. This means that a set of examples has legitimate class labels in a given circumstance and has a legitimate label class at another (Kolter & Maloof, 2007; Muhlbaier & Polikar, 2007).

The learning task is particularly challenging in non-stationary environments as the learning algorithm must adapt itself, by distinguishing between an effective change and possible variations that are due to noise in the training data. Another important issue is related to the way arriving instances are treated and how to combine them as important contributors to the final decision (Tsymbal, Pechenizkiy, Cunningham, & Puuronen, 2008). Effective learning in non-stationary environments with hidden context (latent variables) requires a learning algorithm with the ability to detect context changes without being explicitly informed about them (inference), quickly recover from the context change and adjust its hypothesis to the new context. It should also make use of previous experienced situations when old context and corresponding concept reappear (Bishop, 1998; Widmer & Kubat, 1996).

2.2.2. Dynamics in recommendation systems

Traditionally, much of the published research on recommendation systems has focused on the algorithms that power the recommendation process. However, many research challenges remain, especially when it comes to changing environments. In these scenarios, a recommendation system may simply neglect previous information as soon as a drift is detected or, more interestingly from a learning perspective, somehow try to accommodate the novel knowledge.

Research is ongoing to analyse needs and expectations from the users to which recommendations are offered. In particular, when recommenders are applied in domains other than the ones traditionally covered, information regarding the user needs should be retrieved implicitly or explicitly.

In the following section we will present an approach that includes customized user information in the learning process with the goal of fitting the learning model to the user needs.

3. Proposed approach

In this section we describe the proposed active learning strategy that boosts classification using dynamic customized crowds. The focus on dynamic environments is particularly challenging, since classifiers must adapt to deal with changes usually dependent on hidden drifting contexts. These contexts may arise from different sources, making harder the problem of predicting the right class. Time is usually the most obvious and pervasive cause of drifting contexts. Models should adapt over time to changes in the concept to be learned. For instance, an email message considered legitimate at some point in time, may later drift to spam (undesired email) for any number of reasons (e.g. repetition of emails, specific keywords).

In this work, we focus on a specific type of context drift that depends on user preferences. Using again the spam classifier as an example, current generic classifiers are often found insufficient to fulfill user's expectations, since they can vary tremendously among users. An email can be perfectly legitimate for a given user, but undoubtedly spam for another user. To tackle the problem of user dynamics, we propose an active learning strategy that uses crowds as source of annotated information to train the model to each user. Such a system, able to adapt to user drifts, can be specially relevant in the customization process of a recommendation system, where usually highly subjective issues arise, like joke, book or movie recommendations.

The rationale behind our approach is the possibility of models customization based on user preferences casted as a classification problem in dynamic environments, since the classification model must adapt to the preferences of each new user. Those preferences can be dependent on age, cultural context, geographic location and others, which turn out to be the hidden context previously referred. Thus, model selection can become a problem, since ground-truth is often difficult to determine, as it is quite improbable that all users have the same opinion about a given item, drastically reducing the possibility of having labels that are applicable in all possible scenarios and/or users. Hence, two questions arise: (i) the subjectivity and (ii) the reliability of crowd users (annotators).

To adapt (tune) the model for a given user it is necessary to have complementary information that allows the learning machine to implicitly identify the user profile. This information can be supplied explicitly by user profiles, or implicitly by using user feedback.

In this approach we deal with the associated dynamics of customization in recommendation systems using user feedback to characterize each user, and choosing the appropriate crowd to minimize the personal feedback required. We will firstly present the baseline approach (see Fig. 1), and then follow to the approaches that permit the tuning of the initial model, namely (i) active approach, (ii) personal active approach and (iii) custom active approach (see Fig. 2).

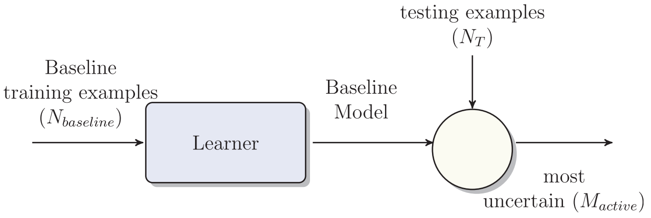

Fig. 1. Baseline approach.

3.1. Baseline approach

The baseline approach is the starting point of our methodology, and is depicted in Fig. 1. The confidence in the classification of the testing set is used to identify which are the most informative examples of the testing set, i.e. those classified with less confidence, in order to use them as active examples in the subsequent approaches.

The examples are equally splitted into training ($N_{baseline}$) and testing examples (N_T). Then, the baseline model is constructed using the training examples, referred from now on as baseline training examples, and tested using the testing examples. The results on the testing examples are used to report individual performances and to choose the active examples to be used further on. In this baseline approach the training examples are generic in the sense that they include the contribution of all the users of the recommendation system, regardless of any personal feature or profile.

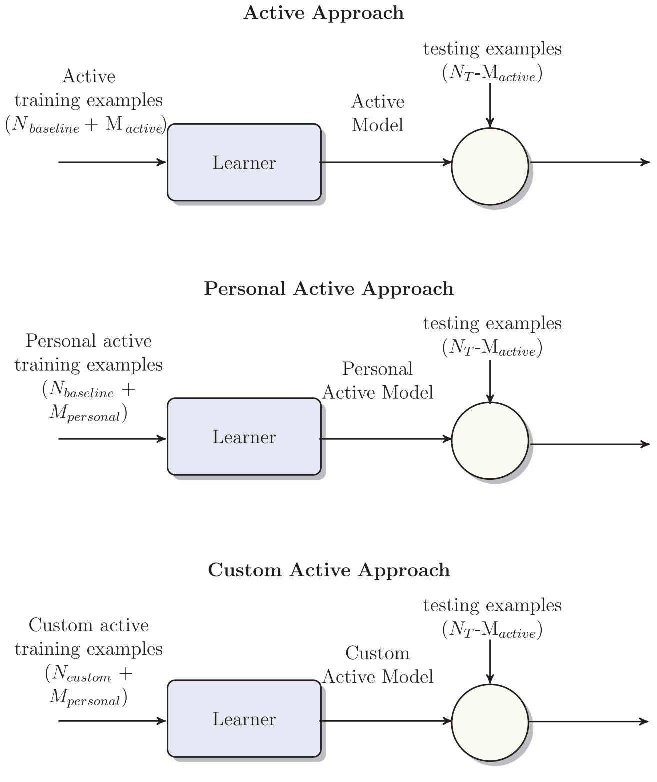

Fig. 2. Active approach, personal active approach and custom active approach.

A certainty-based strategy is then put forward, by using this model to determine the most uncertain examples (M_{active}), also referred from now on as active learning examples, and pointing them to be used in an active learning strategy. The certainty is dependent on the learning method. Using margin-based algorithms, e.g. SVM, we can use the classification margin provided by the baseline model.

The underpinning idea of selecting a subset of examples is to define a smaller number of examples based on their informativeness, in order to manually classify them in an active user feedback process. The relevance of using active learning in documents classification, along with a profound explanation of this initially strategy and its results, has already been introduced in previous work by the authors (Costa et al., 2011a).

3.2. Active learning approaches

An active learning approach is based on the idea that the learning algorithm has the ability to choose the learning examples more adequate to the learning process (Costa et al., 2011c; Dan, 2004; McCallum & Nigam, 1998; Tong & Koller, 2002). Hence, the definition of a subset of active examples is crucial to the forthcoming approaches, as an active learning strategy will be put forward based on these examples.

Considering a recommendation system, the main idea of using an active subset of examples is to integrate user feedback into the learning process, by either asking each new user, or a *customized* crowd, to classify those examples to construct the appropriate user profile, just before building the customized model. In Fig. 2 we illustrate the three learning approaches proposed, namely active approach, personal active approach and custom active approach.

3.3. Active approach

The key factor in any active approach is the determination of the active examples. As explained in Section 3.1, we use a certainty-based strategy to determine the most uncertain examples (M_{active}), named active learning examples. These active examples are added to the baseline training examples ($N_{baseline}$) and removed from the testing training examples. The newly defined testing set is defined as N_T - M_{active}. The classification of these new training examples is generic, in the sense that it is not customized for any given user and is obtained with the contribution of all the users of the recommendation system regardless any personal feature or profile.

Using a recommendation system as example, where a multitude of users (crowd) rates a given item, the active approach enriches the training set with the most uncertain examples, a common active approach.

3.4. Personal active approach

In the SVM personal active approach, the active learning examples are not generically classified, but the user is requested to classify them, in order to promote the customization. These examples are then named personal active training examples ($M_{personal}$). Differently from the previous approach, the new user presented to the recommendation system must classify the active learning examples according to his preferences, in order to define his profile. This information is then used along with the generically training examples, i.e. the $N_{baseline}$ examples, to train a more customized model.

In this approach the number of personal active training examples has to be necessarily small, since the user is required to provide a classification. Nonetheless, the adequacy of this number is user and task dependent.

Using a recommendation system as example, the active examples are directly classified by the user instead of being classified by the crowd, providing an obvious advantage in terms of representativeness of the training dataset.

3.5. Custom active approach

Finally, in the SVM custom active approach we take the strategy one step ahead. On one hand we use the active examples personally classified by the user ($M_{personal}$) in the previous approach, but we also customize the baseline examples that were used so far. To achieve this customization, instead of using the crowd contribution to determine the classification of the baseline examples ($N_{baseline}$), we choose a customized crowd, i.e. a crowd with closer preferences to our target user, resulting in a customized set of examples (N_{custom}).

As already stated, the new user profile is defined by his classification in the active learning examples, therefore the closeness between individuals can just take into account the classification of this subset. The baseline training examples classified by this customized group from the crowd is referred as custom examples (N_{custom}).

The main idea behind this approach is to use not only the classification of the user, but adjusting the baseline training examples by restricting the contribution of the previously seen individuals to those that are closely related to the new user. The underpinning idea is that the information provided by them can be more valuable, as it avoids the bias provided by using remarkably different users when compared to the one we intent to customize our model for. It is also important to refer that this approach also avoids asking the new user to manually classify the whole training examples, specially when it is sometimes unfeasible to ask for such contribution, and thus decreasing variance. In this regard, this approach reaches the best trade-off for the bias-variance dilemma.

In the next section we will present the appropriate closeness metrics that can be applied to determine the customized crowd.

3.6. Closeness metrics

Considering a generic recommendation system, we usually have a large set of ratings for every item. When comparing two users, a straightforward technique is to use the sum of the absolute differences between items being classified.

Taking I as the collection of items, a and b as two different users, one can estimate the closeness using:

$$\sum_{i \in I} |C_i^a - C_i^b|, \tag{1}$$

where C_i^a is the classification of item i given by user a.

Using such a similitude measure to determine the closeness between two users, we can then choose a subset of the users that compose a crowd, using only the k users closer (more similar) to a given user. Such a customized crowd is then suited to provide information for customizing a model to classify documents matching the user preferences.

4. Experimental setup

In this section we start by describing the case study used to evaluate the proposed approach, where users have to rate a set of jokes from 1 to 10. The humor classification example is next described, followed by the introduction to the learning mechanism. We finish by detailing pre-processing methods and presenting evaluation assessment metrics for the proposed framework.

4.1. Case study: humor classification

Humor research in computer science has two main research areas: humor generation (Binsted & Ritchie, 1994; Stock & Strapparava, 2003) and humor recognition (Mihalcea & Strapparava, 2005, 2006; Reyes, Potthast, Rosso, & Stein, 2010). With respect to the latter, research carried out so far considers mostly humor in short sentences, like *one-liners*, that is jokes with only one line sentence, and the improvement of interaction between applications and users.

Humor classification is intrinsically subjective. Each one of us has its own perception of fun, hence automatic humor recognition is a difficult learning task that is gaining interest among the scientific community. Classification methods used thus far are mainly text-based and include diverse classifiers, like SVM classifiers, *naïve Bayes* and decision trees.

In Mihalcea and Strapparava (2006) a humor recognition approach based on *one-liners* is presented. A data set was built by grabbing *one-liners* from the web, using web search engines. This humorous data set was then compared with non-humorous data sets like headlines from news articles published in the Reuters newswire and a collection of proverbs.

In Reyes et al. (2010) another interesting approach is proposed to distinguish between an implicit funny comment and a not funny one. The authors used a 600,000 web comments data set, retrieved from the Slashdot news Web site. These web comments were tagged by users into four categories: funny, informative, insightful, and negative. Data set was then split in humorous and non-humorous comments.

4.2. Dataset

In this paper we used the Jester data set as a benchmark. It contains 4.1 million continuous ratings (-10.00 to +10.00) of 100 jokes from 73,421 users and is available at: http://eigentaste.berkeley.edu. It was generated from Ken Goldberg's joke recommendation website, where users rate a core set of 10 jokes and receive recommendations from other jokes they could also like. As users can continue reading and rating and most of them end up rating all the 100 jokes, the data set is quite dense.

The data set is provided in three parts: the first one contains data from 24,983 users, the second one from 23,500 users and the third one contains data from 24,938 users. The users from part one and two have rated 36 or more jokes, while the users from the third part have only rated between 15 and 35 jokes. The experiments were carried out using the first and the second part as they contain a significant number of users that rated all jokes.

For classification purposes, a joke classified on average above 0.00 is a recommendable joke, being non recommendable a joke below that value. Jokes were split into two equal disjoint sets: training and test. The data from the training set is used to select learning models, while data from the testing set is used to evaluate performance.

4.3. Learning

Jokes classification is considered a binary task that can be formalized as approximating the unknown target function $f : \mathcal{J} \times \mathcal{C} \longrightarrow \{-1, 1\}$ that corresponds to how jokes would be classified by a human. The function f is the jokes classifier, $\mathcal{C} = \{c_1, c_2, \cdots, c_{|\mathcal{C}|}\}$ and $\mathcal{J}$ is a set of jokes. Each joke $\mathbf{d}$ has a simple document representation, which is the vector space model also known as Bag of Words. The joke is represented as a set of features, usually words, $\mathcal{W} = \{w_1, w_2, \cdots, w_{|\mathcal{W}|}\}$ with each one as a vector $\mathbf{d}_i = (w_{i1}, w_{i2}, \cdots, w_{i|\mathcal{W}|})$ where w_{ik} describes each feature representation for the specified joke. In this representation each joke is indexed with the bag of the terms occurring on it, i.e., is a vector with one component for each term occurring in the whole collection. When $f(\mathbf{d}_i, c_j) = 1$ $\mathbf{d}_i$ is a positive example or member of classe c_j otherwise is a negative example of c_j. Since we have a binary classification problem the cardinality of classes is two ($|C| = 2$).

We will now detail the setup for each approach. Regarding the active approach, every user, despite its profile, contributes equally to the classification of the generically classified examples. Although there are different ways to define the resultant classification based on multiple contributions, like majority voting or weighted voting, we propose an equally weighted voting system based on numeric values. The difference is that a user contributes not only with a binary decision, like liking a book, or not, but also contributes with the corresponding numeric value that scales the likeness ratio.

The learning method used in this experimental setup is the well known Support Vector Machine (SVM) (Vapnik, 1999). Given our case study, a joke text classification setup, the SVM is an obvious choice (Joachims, 2002). Nevertheless, the proposed strategies can be applied to any learning algorithm that provides some confidence level in the classification.

New unlabeled examples are classified by the SVM according to which side of the Optimal Separating Hyperplane (OSH) they fall into, although not all of them are classified with the same margin to the OSH, as depicted in Fig. 3. Examples close to the margin are those where the SVM puts less confidence, as slight deviations of the OSH would change their given class.

4.4. Pre-processing

A joke is represented as the most common, simple and successful document representation, which is the vector space model, also known as *bag of words*. Each joke is indexed with the *bag* of the terms occurring on it, having a value that takes into account the number of times the term appear in the joke. It was also considered the simplest approach in the definition of term, as it was defined as any space-separated word.

Considering the proposed approach and the use of text-classification methods, pre-processing methods were applied in order to reduce feature space. These techniques, as the name reveals, reduce the size of the joke representation and prevent the mislead classification as some words, such as articles, prepositions and conjunctions, called *stopwords*, are non-informative words, and occur more frequently than informative ones. These words could also mislead correlations between jokes, so *stopword* removal technique was applied. *Stemming* method was also applied. This method consists in removing case and inflection information of a word, reducing it to the word stem. Stemming does not alter significantly the information included, but it does avoid feature expansion.

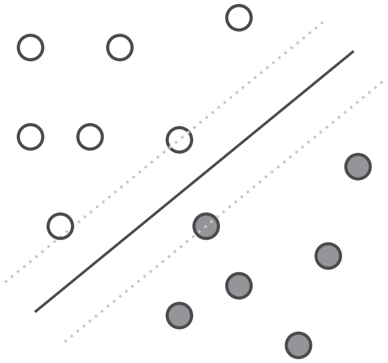

Fig. 3. 2-D dimensional example of the SVM margin and unlabelled examples.

Table 1
Contingency table for binary classification.

	Class positive	Class negative
Assigned positive	*a* (True positives)	*b* (False positives)
Assigned negative	*c* (False negatives)	*d* (True negatives)

4.5. Performance metrics

In order to evaluate a binary decision task we first define a contingency matrix representing the possible outcomes of the classification, as shown in Table 1.

Several measures have been defined based on this contingency table, such as, error rate ($\frac{b+c}{a+b+c+d}$), recall ($R = \frac{a}{a+c}$), and precision ($P = \frac{a}{a+b}$), as well as combined measures, such as, the van Rijsbergen F_β measure (van Rijsbergen, 1979), which combines recall and precision in a single score.

$$F_\beta = \frac{(\beta^2+1)P \times R}{\beta^2 P + R}. \tag{2}$$

F_β is one of the best suited measures for text classification used with $\beta = 1$, i.e. F_1, an harmonic average between precision and recall (3), since it evaluates well unbalanced scenarios that usually occur in text classification settings.

$$F_1 = \frac{2 \times P \times R}{P + R}. \tag{3}$$

5. Experimental results and analysis

In Table 2 we present the results for the baseline approach. These results are just informative, and not comparable with the following approaches, since they constitute macro-averaging over all users.

Although the value of 86.42% for F1 is rather acceptable, one should keep in mind that it would only be valid for a user with preferences rather similar to the average preferences.

Table 3 shows the recall and precision results for both levels of crowd customization. We considered that 10 jokes were deemed sufficiently non-intrusive for a user to classify, and the closer crowd in the custom active approach was heuristically defined to have the $k = 1000$ users closest preferences to the user (see Section 3.6).

We may also observe that whilst precision values are rather similar, there is a relevant difference of circa 5% in recall values. This difference results in more relevant items being discovered and made available to a user in the recommendation system. One may argue that this relevant items can make the difference in the user's evaluation of the service provided.

Table 2
Baseline approach performance on Jester dataset.

	Precision	Recall	F1
Baseline approach	81.40%	92.11%	86.42%

Table 3
Precision and recall performances for active approaches.

	Precision	Recall
Personal active approach	72.78± 0.17%	82.69± 0.13%
Custom active approach	72.35± 0.18%	87.32± 0.20%

Regarding F1 we have macro-averaged values of 77.42% for the personal active approach and 79.13% for the custom active approach. As expected by the difference in recall values, the custom approach presents a better overall performance. While the personal active approach only uses the 10 active examples, it is outperformed by the personal approach that takes customization one step further by using the similarity measure defined in Section 3.6 (1), to choose the crowd that is closer to user preferences.

Finally, notice that the values of Tables 2 and 3 are not directly comparable. While the baseline results are generic, the active results are personalized. The goal in the baseline approach was set as the average of the classification of each joke, while the goal in the active approaches is distinct for each user, thus much harder to learn.

6. Conclusions and future work

In this paper we proposed a framework for active learning that uses crowd-based non-expert information to capture the drift between user preferences in a recommendation system. The proposed strategy incorporates a baseline approach and a suite of learning approaches, namely active, personal active and custom active approaches. The methodology starts by using the baseline approach to identify the *active examples*, i.e. those that were classified with less confidence. Then, an active learning approach is applied to the active examples.

The presented framework allows non-experts classification performed by crowds to be used to substitute the user profile definition, mitigating the labeling effort normally requested to the user. We evaluated the overall approach with the Jester dataset, based on Ken Goldberg's joke recommendations website.

The results obtained revealed the usefulness of using crowds in the adjustment of user preferences. More precisely, we determined that specially chosen crowds are able to retrieve customized user preferences. Moreover, the results have also shown that active learning plays a crucial role in the overall classification process, as the training set becomes supplemented with the most uncertain examples obtained by the baseline model.

Our research will expand the framework to deal with temporal drifts in user preferences. We also aim to evaluate the appropriateness of using this framework and the research strategy defined in other distinct contextual environments suitable to recommendation systems.

Acknowledgements

This work is partly funded by the ERDF, through the COMPETE Programme, by FCT within project PEst-C/EEI/LA0014/2011 and by iCIS (CENTRO-07-ST24-FEDER-002003).

References

Attar, V., Chaudhary, P., Rahagude, S., Chaudhari, G., & Sinha, P. (2012). An instance-window based classification algorithm for handling gradual concept drifts. *Agents and Data Mining Interaction*, 156–172.

Baram, Y., El-Yaniv, R., & Luz, K. (2004). Online choice of active learning algorithms. *The Journal of Machine Learning Research, 5*, 255–291.

Barr, J., & Cabrera, L. F. (2006). AI gets a brain. *Queue, 4*, 24–29.

Binsted, K., & Ritchie, G. (1994). An implemented model of punning riddles. In *Proceedings of the national conference on artificial intelligence* (pp. 633). John Wiley & Sons, LTD.

Bishop, C. (1998). Latent variable models. *Learning in Graphical Models*, 371.

Brabham, D. C. (2008). Crowdsourcing as a model for problem solving an introduction and cases. *Convergence: The International Journal of Research into New Media Technologies, 14*, 75–90.

Brew, A., Greene, D., & Cunningham, P. (2010). The interaction between supervised learning and crowdsourcing. In *24th Annual conference on neural information processing systems (NIPS'10)*.

Burguera, I., Zurutuza, U., & Nadjm-Tehrani, S. (2011). Crowdroid: Behavior-based malware detection system for android. In *Proceedings of the 1st ACM workshop on security and privacy in smartphones and mobile devices* (pp. 15–26). ACM.

Chen, Y.-Y., Hsu, W. H., & Liao, H.-Y. M (2011). Learning facial attributes by crowdsourcing in social media. In *Proceedings of the 20th international conference companion on World Wide Web* (pp. 25–26). ACM.

Costa, J., Silva, C., Antunes, M., & Ribeiro, B. (2011a). The importance of precision in humour classification. In *Proceedings of intelligent data engineering and automated learning – IDEAL 2011* (pp. 271–278). Springer.

Costa, J., Silva, C., Antunes, M., & Ribeiro, B. (2011b). Get your jokes right: Ask the crowd. In *Proceedings of 1st international conference on model and data engineering – MEDI 2011* (pp. 469–474). Obidos, Portugal: Springer.

Costa, J., Silva, C., Antunes, M., & Ribeiro, B. (2011c). On using crowdsourcing and active learning to improve classification performance. In *Proceedings of 11th international conference on intelligent systems design and applications – ISDA* (pp. 469–474). IEEE.

Dan, S. (2004). Multi-criteria-based active learning for named entity recognition. Master's thesis, National University of Singapore.

Florido, J., Pomares, H., & Rojas, I. (2011). Generating balanced learning and test sets for function approximation problems. *International Journal of Neural Systems, 21*, 247–263.

Franklin, M. J., Kossmann, D., Kraska, T., Ramesh, S., & Xin, R. (2011). Crowddb: Answering queries with crowdsourcing. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 61–72).

Greengard, S. (2011). Following the crowd. *Communications of the ACM, 54*, 20–22.

Howe, J. (2006). The rise of crowdsourcing. *Wired Magazine, 14*, 1–4.

Joachims, T. (2002). *Learning text classifiers with support vector machines*. Dordrecht, NL: Kluwer Academic Publishers.

Kittur, A., Smus, B., Khamkar, S., & Kraut, R. E. (2011). Crowdforge: Crowdsourcing complex work. In *Proceedings of 24th annual ACM Symposium on User interface software and technology* (pp. 43–52).

Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research, 8*, 2755–2790.

Konstan, O., & Riedl, J. (2012). Deconstructing recommender systems. *IEEE Spectrum, 10*, 1–7.

Lebrun, G., Charrier, C., Lezoray, O., & Cardot, H. (2008). Tabu search model selection for svm. *International Journal of Neural Systems, 18*, 19–31.

Leimeister, J. M. (2010). Collective intelligence. *Business & Information Systems Engineering, 2*, 245–248.

Marcus, A., Wu, E., Karger, D., & Madden, S. (2011). Crowdsourced databases: Query processing with people. In *5th Biennial conference on innovative data systems research (CIDR)* (pp. 211–214).

McCallum, A. K., & Nigam, K. (1998). Employing EM and pool-based active learning for text classification. In *Proceedings of ICML-98, 15th international conference on, machine learning* (pp. 350–358).

Mihalcea, R., & Strapparava, C. (2005). Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the conference on human language technology and empirical methods in natural language processing* (pp. 531–538). Association for Computational Linguistics.

Mihalcea, R., & Strapparava, C. (2006). Technologies that make you smile: Adding humor to text-based applications. *Intelligent Systems, 21*, 33–39.

Minku, L., White, A., & Yao, X. (2010). The impact of diversity on on-line ensemble learning in the presence of concept drift. *IEEE Transactions on Knowledge and Data Engineering, 2*, 730–742.

Muhlbaier, M. D., & Polikar, R. (2007). An ensemble approach for incremental learning in nonstationary environments. In *Multiple classifier systems* (pp. 490–500). Springer.

Negri, M., Bentivogli, L., Mehdad, Y., Giampiccolo, D., & Marchetti, A. (2011). Divide and conquer: Crowdsourcing the creation of cross-lingual textual entailment corpora. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 670–679). Association for Computational Linguistics.

Reyes, A., Potthast, M., Rosso, P., & Stein, B. (2010). Evaluating humor features on web comments. In *Proceedings of the 7th International Conference on, Language Resources and Evaluation* (pp. 1138–1141).

Settles, B. (2010). Active learning literature survey, CS Technical Report 1648, University of Wisconsin–Madison.

Silva, C., & Ribeiro, B. (2007). On text-based mining with active learning and background knowledge using svm. *Soft Computing, 11*, 519–530.

Silva, C., & Ribeiro, B. (2008). Towards expanding relevance vector machines to large scale datasets. *International Journal of Neural Systems, 18*, 45–58.

Stock, O., & Strapparava, C. (2003). Getting serious about the development of computational humor. *International joint conference on artificial intelligence* (Vol. 18, pp. 59–64). Lawrence Erlbaum Associates, Ltd.

Surowiecki, J. (2004). *The wisdom of crowds*. Doubleday.

Tarasov, A. & Delany, S. (2010). Using crowdsourcing for labelling emotional speech assets. In *ECAI - prestigious applications of intelligent systems* (pp. 1–11).

Tong, S., & Koller, D. (2002). Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research, 2*, 45–66.

Tsymbal, A., Pechenizkiy, M., Cunningham, P., & Puuronen, S. (2008). Dynamic integration of classifiers for handling concept drift. *Information Fusion, 9*, 56–68.

van Rijsbergen, C. (1979). *Information retrieval*. Butterworths Ed..

Vapnik, V. (1999). *The nature of statistical learning theory*. Springer.

Webb, G., Pazzani, M., & Billsus, D. (2001). Machine learning for user modeling. *User Modeling and User-Adapted Interaction, 11*, 19–29.

Welinder, P., & Perona, P. (2010). Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *IEEE computer society conference on computer vision and pattern recognition workshops* (pp. 25–32). IEEE.

Widmer, G., & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine learning, 23*, 69–101.

Zhao, L., Sukthankar, G., & Sukthankar, R. (2011). Robust active learning using crowdsourced annotations for activity recognition. In *Proceedings of AAAI 2011 workshop on human computation* (pp. 74–79).