Fixing Mislabeling by Human Annotators via Conflict Resolution and the Exploitation of Prior Knowledge

MATTIA ZENI[1]*, WANYI ZHANG[1], ENRICO BIGNOTTI[1], ANDREA PASSERINI[1], and - FAUSTO GIUNCHIGLIA[1,2], (1) Department of Information Engineering and Computer Science, University of Trento, IT (2) College of Computer Science and Technology, Jilin University, Changchun, China, CN

According to the “human in the loop" paradigm, machine learning algorithms can improve when leveraging on human intelligence, usually in the form of labels or annotation from domain experts. However, in the case of research areas such as ubiquitous computing or lifelong learning, where the annotator is not an expert and is continuously asked for feedback, humans can provide significant fractions of incorrect labels. We propose to address this issue in a series of experiments where students are asked to provide information about their behavior via a dedicated mobile application. Their trustworthiness is tested by employing an architecture where the machine uses all its available knowledge to check the correctness of its own and the user labeling to build a uniform confidence measure for both of them to be used when a contradiction arises. The overarching system runs through a series of modes with progressively higher confidence and features a conflict resolution component to settle the inconsistencies. The results are very promising and show the pervasiveness of annotation mistakes, the extreme diversity of the users' behaviors which provides evidence of the impracticality of a uniform fits-it-all solution, and the substantially improved performance of a skeptical supervised learning strategy.

CCS Concepts: • **Human-centered computing** **Ubiquitous computing**; • **Computing methodologies** **Knowledge representation and reasoning**; **Supervised learning**;

ACM Reference Format:
Mattia Zeni[1], Wanyi Zhang[1], Enrico Bignotti[1], Andrea Passerini[1], and Fausto Giunchiglia[1,2]. 2018. Fixing Mislabeling by Human Annotators via Conflict Resolution and the Exploitation of Prior Knowledge. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 0, 0, Article 0 (March 2018), 23 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Nowadays, our smartphones are no longer just miniature computers to send emails or access the Web, but they are becoming more and more like assistants for our everyday life, e.g., reminding us of meetings in our agenda or guiding us in a new city. The critical factor for the change of roles that machines play in our lives is knowledge; in fact, machines need to have access to our knowledge and not just convey it. Knowledge is however diverse and heterogeneous, and what a person knows may differ radically from another, and this requires humans to help machines in understanding the world in the same way that they do. In recent approaches in computer science such as (supervised) machine learning [38] or mobile crowdsensing [20], human knowledge is usually provided

*This is the corresponding author

Authors' address: Mattia Zeni[1], mattia.zeni@disi.unitn.it; Wanyi Zhang[1], wanyi.zhang@unitn.it; Enrico Bignotti[1], enrico.bignotti@unitn.it; Andrea Passerini[1], andrea.passerini@unitn.it; Fausto Giunchiglia[1,2], fausto.giunchiglia@unitn.it, (1) Department of Information Engineering and Computer Science, University of Trento, via Sommarive, 9, Trento, TN, 38123, IT, (2) College of Computer Science and Technology, Jilin University, Changchun, China, No.2699 Qianjin Street, Changchun, JL, 130012, CN.

2474-9567/2018/3-ART0 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

as annotations or labeling of data collected with sensors or other means. This type of human contribution is at the heart of the “human in the loop” paradigm which leverages both human and machine intelligence to create machine learning models by involving humans in training, tuning and testing data for a particular machine learning algorithm. The final goal is to use humans to improve the quality of the results of the machine. However, humans can make mistakes and more often than not unknowingly or unwittingly, due to their biases, which alter their perception and judgment of reality. Research in the Social Sciences provides evidence of the unreliability of people when required to compile self-reports such as time diaries [40] describing their behavior. For instance, respondents have difficulties recalling sequence and relevance of distant activities, i.e., memory bias and cognitive load [42], or may feel discouraged or non-cooperative because they do not understand the instructions or other reasons, i.e., unwillingness to report [4], or change their reported behavior to one that is considered socially desirable, i.e., conditioning [45]. Unfortunately, other areas outside social science such as machine learning are affected by these biases, especially since these approaches assume that the human annotator is an expert, and that her feedback is mostly correct apart from some occasional inaccuracies. Such an assumption is becoming less and less sustainable in a world where non-expert users are involved in annotating a constant stream of data, be it the flow of news in the Web (see, e.g., [10]), or the flow of sensor data in pervasive and ubiquitous computing (see, e.g., [15, 17]). In the case of the latter scenario, the same input can be proposed multiple times, resulting in it being often labeled differently even by the same user [18].

The research presented here is part of, and motivated by, a long-term series of experiments (two experiments per year for two years) aimed at studying the University student life and at correlating the students’ behaviors with their academic performance, measured concerning grades and credits passed. The long-term goal is to support students who should increase their performance, thus also minimizing the possibility of a failure in getting the degree. Nonetheless, the early results from our experiments show that students display very different behaviors, which also include a considerable amount of unreliable information. Thus, motivated by these earlier findings, the goal of the research described here is to deal with the problem of the *untrustfulness of human annotators*, namely with the problem of how to define techniques which allow minimizing the effects that mislabeling has on the machine ability to do proper activity and context recognition. The central intuition behind our solution is that the machine exploits its available knowledge to assess the correctness of both its prediction and of the user label. The machine can obtain a confidence measure not only about itself but also about the user by monitoring the sequence of wrong and correct answers. If a contradiction arises, these measures are employed to make the final decision of what is the case. Notice that this method provides a uniform way to seamlessly integrate the *data-driven* view of the world provided by machine learning algorithms and the *model-driven* view of the world provided by human annotators and by the previously stored knowledge. Thus, for instance, the machine can exploit the information that *car* and *automobile* are synonyms and, in turn, more specific terms of *vehicle*, or that a *Research Department* is a *part-of* a *University*, thus freeing the user from remembering the exact same labels used in the past.

The main contribution of this work is a detailed description of the experiment we have carried out, which highlights the advantages of an interactive, skeptical approach to learning over state-of-the-art but non-skeptical machine learning alternatives. Results show that a *Skeptical Supervised (Machine) Learning (SSML)* architecture improves over an approach relying only on the noise-robustness of the underlying learning algorithm, as is customary in the machine learning community. Questioning user labeling and exploiting semantic knowledge for conflict resolution are crucial aspects underlying the observed increase in performance. The results provided also show the extreme variability of the students’ behaviors and, therefore, the need for a person-centric customized solution to the problem of mislabeling.

The paper is structured as follows. In Section 2 we provide the related work. In Section 3 we describe the SSML system in detail, from the general architecture to the various components. The data collection experiment is discussed in Section 4, and the results of running the SSML system are detailed in Section 5. Finally, conclusions are drawn in Section 6.

2 RELATED WORK

The issue of the reliability of users is a well-known problem in social sciences and psychology where, given the relevance of understanding behavior, self-reports of everyday life are widely used [22]. When reporting using these tools, users may show different response biases causing a lack of congruence between subjects' answers and their adherence to reality [42]. Although there are several types of biases, e.g., memory biases or inadequate recalling (i.e., having issues remembering and reporting correctly) [45], conditioning [4] (i.e., reporting socially desirable behaviors) and unwillingness to respond [24] (i.e., failing to report), there are still no systematic approaches in sociology to address and quantify them [12]. All the work we carry out in our experiments is grounded in this literature which constitutes our starting point for the elaboration of the solutions we propose.

The main population of our experiments is students [19], since they are a relevant sample of population for sociological studies, especially with respect to their time management ability, which includes setting goals and priorities and plays a crucial role in improving students' performance [31]. Another area that focuses on students, due to their wide adoption of smartphones and tech-savviness [3], is computational social sciences, which is based on approaches for extracting and analyzing behaviors using smartphone data such as proximity, location, and call logs [8]. These data are combined with surveys, which may be administered via smartphones, for socio-psychological metrics such as personality traits, daily mood, or sleep quality [3]. Within the work in computational social sciences, the Student Life study [43] is the closest one to ours. For this study, smartphones were used to assess the impact of workload on stress, sleep, activity, mood, sociability, mental well-being and academic performance of a class of 48 students (38 males and 10 females) of a computer science class across a 10-week term at Dartmouth College. Moreover, the SmartGPA study [44] used the data from [43] to show that there is evidence of a link between the students' GPA and their behavioral patterns. In this work, regression analyses were used to develop a behavioral slope and behavioral breakpoints in order to identify changes in a student's behavior on a weekly basis. The temporal granularity and the predictive model does not consider raw data, since the pre-built classifiers feed into a regression model, e.g., the accelerometer. Differently from our work, these studies do not have approaches in place for assessing the quality of the students' annotations. In fact, they rely on heuristics, e.g., spending at least 20 minutes in a library means that the student is studying, and classifiers, which may be inaccurate, e.g., conversation classifiers cannot distinguish a person speaking in a TV from one physically in the room [21].

The problem of noisy supervision is well-known in machine learning. While traditional approaches to concept learning assume perfectly labeled training sets, most recent supervised learning techniques can tolerate a small fraction of mislabelled training instances (see [13] for a comprehensive overview). The most popular solution consists in designing learning models which are robust to (some) label noise [11]. By averaging predictions of multiple learners, ensemble methods typically perform well in terms of noise robustness [6, 34]. The robustness of random forests, the ensemble method we use in this work, was shown theoretically and empirically in a recent work [14]. Nonetheless, label noise badly affects the performance of learning algorithms [30]. Our approach diverges from existing solutions in that it directly involves the user in an interactive process of consistency check and labeling revision. This allows tolerating a much larger amount of noise, achieving substantial improvements over a solution which only relies on the noise-robustness of the learning algorithm. The whole field of statistical relational learning [1] deals with the integration of symbolic and sub-symbolic approaches to learning. Frameworks like Markov Logics [36], Semantic-Based Regularization [7] or Learning Modulo Theories [41] combine logical rules or other types of constraints with learnable weights in order to encourage predictions which are consistent with the available knowledge. The crucial difference here is that knowledge is used as the main means for aligning and enforcing the consistency between the user labels and the output of the machine learning algorithms. In the limit case, where the user and the machine learning algorithms provide mutually consistent labels which are however inconsistent with the prior knowledge, we would end up generating an inconsistency. In this respect, our work and the work cited above are complementary.

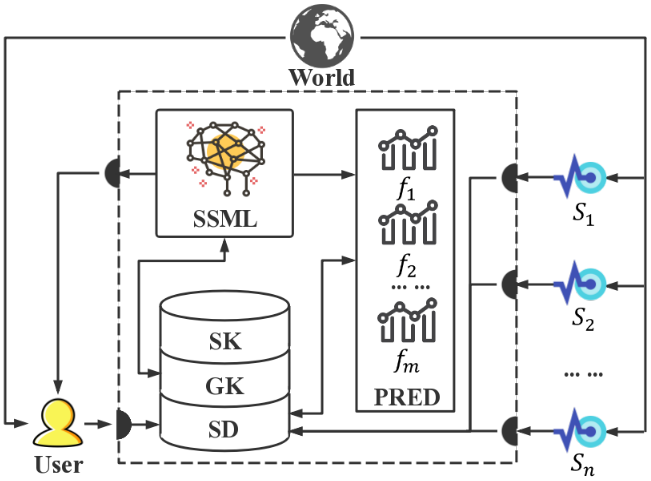


Fig. 1. The SSML Architecture. The disk-like component marked with SK, GK, SD contains the prior knowledge (SK, GK) and the streaming data coming from the sensors and the user labeling activity (SD); PRED is the main machine learning algorithm which in turns is an ensemble of simpler machine learning algorithms; SSML is the main Skeptical Supervised Machine Learning algorithm.

While many machine learning approaches assume or expect the user to be an expert, in other areas of research this condition cannot be met — this is the case of participatory sensing [25] and mobile crowdsensing [20]. The main idea is to have users collect and share sensed data and annotations from their surroundings using their mobile phones. A relevant issue is assessing the quality of users' annotations [35] since it can be negatively affected by several factors, e.g., short duration activities not captured [23] or the interruption needed for annotating or performing them [26]. [28] propose to evaluate how much a user is familiar with certain locations at a given time based on how regular she is in being there. Thus, user's credibility is computed on the basis of his/her past interactions with a geo-referenced crowdsourcing application, with the aim of reflecting the usefulness of his/her contributions as seen by other users. Hence, these two measures are combined together to compute a credibility weight of each user. [33] proposes the mPASS system that combines data obtained by sensing, crowdsourcing and mashing-up with geo-referenced social systems (e.g., Foursquare) in order to map urban and architectural accessibility. This system compares the user-provided data with the information provided by authoritative data sources, i.e., local administrations, which is treated as a gold set, to assess the reliability of users. However, at the time of writing, there has been no direct evaluation of this system. In [47], the authors consider the problems of quality estimation and monetary incentives, which respectively represent a passive and active strategy for obtaining data from users. They propose a quality-based truth estimation and surplus sharing method, which mainly consists of two elements: *i)* quality estimation module and *ii)* surplus sharing module. Focusing on the quality estimation module, it is designed to rely on an unsupervised learning technique to estimate the users' data qualities, characterize their long-term reputations, and generate a reliable estimation of ground truth. To improve the estimation accuracy, anomalous users whose sensory readings are far away from the group consensus can be detected and filtered out. The key difference from this work is that we focus on personal data, e.g., personal context and activities, to be used by the user herself and that, therefore, the quality a user's annotation is not evaluated by comparing it with other annotations from the crowd (which would be very hard if not impossible, since we deal with personal data) but, rather, by comparing it with the machine's knowledge.

3 METHOD

The method we follow is composed of a set of three basic ingredients, namely:

- A reference architecture, described in Section 3.1, which integrates the two main components used in supervised learning, namely the machine learning (ML) algorithms and the user providing feedback with two extra components, namely a knowledge component which stores the prior knowledge and the SSML algorithm. The key idea is that the SSML component, based on its internal strategy, involves the user and the ML algorithms in the most suitable way aimed at solving the mislabeling problem. This strategy may or may not involve using the prior knowledge or asking the user for different tasks, for instance, asking her for a second opinion on a previous labeling, with the goal of solving a contradiction;
- The main SSML algorithm, described in Section 3.2, which exploits the critical idea that this algorithm can be in one of three states, namely: *TrainMode* where the goal is to accumulate enough knowledge about the user dynamics, *RefineMode* where, being confident of the previously acquired knowledge, SSML starts checking the user provided label quality and, finally, *RegimeMode* where SSML deploys an active learning strategy where the user is only occasionally asked for feedback as the way to check that SSML and the user herself are aligned in their ability to interpret the world;
- The conflict management algorithm described in Section 3.3, which solves possible labeling conflict by optionally involving the prior knowledge and an *oracle*, for instance, the user herself on a second opinion, who is called upon making the final decision about what is the case.

We describe these three main components in the remainder of this section.

3.1 The SSML reference architecture

Figure 1 shows the multi-layer architecture of the Skeptical Learning. The fundamental intuition is that the human annotator(s) and the machine learning algorithm(s) are treated as *interpretation channels* which provide their *fallible* perspective on what is the case in the real world. In Figure 1, the *first* layer, inside the dashed box, is the HW/SW *Machine*, e.g., a smartphone with its software, implementing the overall SSML functionality while the *third* and the last layer is the world, which is only indirectly accessible to the Machine. In the *second* layer, Machine and World are connected via, on one side, a set of n sensors $S_1, ..., S_n$ (as they exist, e.g., in a smartphone or a smartwatch) which generate a set of streams $\{x_t\}_{S_i}$, with x_t the value collected at time t by the sensor S_i and $\{x_t\}_{S_i}$ the stream of data generated by S_i and, on the other side, the Machine *reference user*, namely the person who is the direct beneficiary of its services.

The reference user, on request (outgoing arrow on the left of the dashed box) provides a label y_t which is interpreted as the value at time t of a specific property P_j, thus generating, by giving answers to different questions, a set of label streams $\{\{y_t\}_{P_j}\}_{\mathrm{u}}$. For instance, the label "University" which answers the question "Where are you?" (ingoing arrow on the bottom left of the dashed box) is interpreted as $In_t(\mathrm{User}, \mathrm{University})$. It is important to underline how the answer by the user is provided based on her perception of the world and without any clue of the sensor values nor of how the Machine uses her input. The mapping from sensor values to user-provided labels is produced by a set of machine learning algorithms, whose existence may not be even known by the user, as described below. Note how this design decision allows for full freedom in the choice of the machine learning algorithms which can, therefore, be tuned to the specific learning problem.

As from figure 1, the Machine consists of three components:

(1) The SSML main algorithm, hence SSML implementing the skeptical learning strategy as described in the next sections;
(2) A *Predictor* PRED consists of an *ensemble* (see, e.g., [32]) of m learning algorithms $f_1, .., f_m$, each taking in input an array of data $\mathbf{x}_t$ (the concatenation of all sensor readings at a certain time) and producing a score $f_k(\mathbf{x}_t, y)$ for all possible labels $y \in \mathcal{Y}_j$ of a given property P_j. These scores are then aggregated by PRED into

a single label y_t which, similarly to the user, is the value that a certain property P_j takes at time t (from the point of view of PRED). Different properties can be managed using different ensembles (one per property) or having the ensemble score a combination of labels, one for each of the properties. PRED, similarly to the user, provides its internally produced labels on request by SSML, where requests are represented in Figure 1 as right outbound arrows from SSML. As implied in Figure 1, both the user and the predictor do not answer back to SSML but rather store their answers in the knowledge component SD. This process allows for the possibly asynchronous interaction of SSML with the user and PRED, which leaves room for independent loosely connected reasoning strategies;

(3) A *Knowledge component* which stores whatever prior knowledge the machine has accumulated in time. It consists of three sub-components:

(a) The Stream Data storage SD, with SD being

$$\text{SD} \; = < \{x_t\}_{S_i}, \{\{y_t\}_{P_j}\}_{\text{p}}, \{\{y_t\}_{P_j}\}_{\text{u}} >$$

where $\{x_t\}_{S_i}$ is the set of data streams coming from the sensors and $\{\{y_t\}_{P_j}\}_{\text{p}}$ and $\{\{y_t\}_{P_j}\}_{\text{u}}$ are the multiple streams, one per property P_j, provided by the predictor and the user, respectively, as answers to the SSML queries;

(b) the component called GK, for *Ground Knowledge*, contains factual knowledge about the world (in the actual implementation it is stored as a knowledge graph), which can be generated in one of three possible ways: (i) it was provided to the machine as a priori knowledge, (ii) it was independently provided by third parties or (iii) it was previously generated by SSML, as described below. GK is where the Machine cumulates the knowledge learned in time. One example of GK is the knowledge about specific locations, or events, or people, for instance, the fact that Fausto's office is part of the Department building, which in turn is a part of the University premises;

(c) the component called SK, for *Schematic Knowledge* which contains general knowledge, for instance in the form of a hierarchy of concepts stated in a certain language, see, e.g., [16]. One trivial example of SK content is an axiom stating that *resting* is a more general concept than *sleeping*. Another example is general statements about locations and sub-locations, for instance, the fact that Department buildings are always inside the Univesity premises. In this paper, we assume that SK is static and not growing and that it is known by the Machine because it is provided as a priori knowledge.

GK and SK are pivotal for the correct interpretation of the Machine and user's' answers. As described in detail in Section 3.3, they are quite useful in the conflict resolution phase as, whenever this is the case, they allow to infer that the user and the Machine meant the same thing even though they used different labels. Thus, for instance, the SK can be used to infer that the user is *resting* from her assertion that she is *sleeping* while the GK can be used to infer that she is *in the University* from her assertion that she is *in her office*, which is part of the University. The key observation is that the GK and the SK, which provide a model-driven view of the world, are unavoidable components whenever there is an interest in making the machine capable of fully understanding the user input, in its intended semantics and, vice-versa, in enabling the user in providing semantics to (i.e., in fully understanding) the output of the Machine internal data-driven machine learning algorithms.

3.2 The SSML main algorithm

In this section, without loss of generality, we assume that there is a single property of interest P, e.g., the location of the user at a particular time. We represent by $\mathcal{Y}$ the set of possible values for this property.

SSML is an algorithm which takes in input a continuous stream of sensor data as they are stored in SD (see Figure 1). The pseudocode of SSML is reported in Algorithm 1. The algorithm can be in one of three modalities which, for simplicity, we assume are activated sequentially, namely: *Train* mode as in typical supervised learning,

Algorithm 1 Skeptical Supervised Learning (SSML)

```
procedure SSML($\theta$)
    init $\mathbf{c}^{\mathrm{u}} = 1$, $\mathbf{c}^{\mathrm{p}} = 0$
    while TRAINMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
        $\mathbf{x}_t$ = SENSORREADING()
        $y_t$ = ASKUSER()
        $\hat{y}_t$ = PRED($\mathbf{x}_t$)
        TRAIN($\mathbf{x}_t, y_t$)
        UPDATE($\mathbf{c}^{\mathrm{p}}, \hat{y}_t, y_t$)
    while REFINEMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
        $\mathbf{x}_t$ = SENSORREADING()
        $y_t$ = ASKUSER()
        $\hat{y}_t$ = PRED($\mathbf{x}_t$)
        SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t$)
    while True do
        $\mathbf{x}_t$ = SENSORREADING()
        $\hat{y}_t$ = PRED($\mathbf{x}_t$)
        if CONF($\mathbf{x}_t, \hat{y}_t, c^{\mathrm{p}}_{\hat{y}_t}$) $\leq \theta$ then
            $y_t$ = ASKUSER($\hat{y}_t$)
            SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t$)
```

Refine mode which checks the quality of the user answers and, under certain conditions, it challenges them, and the *Regime* mode where it starts being autonomous and only queries the user for particularly ambiguous instances.

The algorithm takes as input a confidence threshold θ. It starts by setting the user confidence to one and the predictor confidence to zero for all classes ($|\mathbf{c}^{\mathrm{u}}| = |\mathbf{c}^{\mathrm{p}}| = |\mathcal{Y}|$). Then the training phase begins. The algorithms collect sensor readings ($\mathbf{x}_t$) to be used as input for the predictor. In our current implementation, the prediction procedure PRED simply returns the highest scoring prediction:

$$\textsc{Pred}(\mathbf{x}) := \operatorname*{argmax}_{y \in \mathcal{Y}} \frac{1}{m} \sum_{j=1}^{m} f_j(\mathbf{x}, y) \tag{1}$$

where the score of a prediction y is the average score given to that label by the learners in the ensemble (in the first iteration, when no training has not been performed yet, PRED($\mathbf{x}$) returns a randomly chosen label). The system then asks the user for a label (y_t) to be used as ground truth. The input-output pair is used to train the predictor using the TRAIN procedure. This procedure can either perform batch learning, in which the predictor is retrained from scratch using all input-label pairs stored in memory, or do an online learning step [39], where the novel input-output pair is used to refine the current predictor. The choice between these learning modalities depends on the specific implementation and constraints (e.g., storage capacity), see Section 4 for the details on the actual implementation. After training, the confidence of the predictor is updated using the UPDATE procedure, receiving as input the ground truth label and the predicted one *before* the training step. When the predictor is confident enough to start challenging the user on the correctness of a certain labeling, the training stage is stopped. The confidence in a prediction y for an input $\mathbf{x}$ is obtained from the product of the score of the prediction times the confidence c^{p}_y the predictor has in predicting that label:

$$\textsc{Conf}(\mathbf{x}, y, c^{\mathrm{p}}_y) := c^{\mathrm{p}}_y \cdot \frac{1}{m} \sum_{j=1}^{m} f_j(\mathbf{x}, y) \tag{2}$$

The system remains in training mode as long as the expected probability of contradicting the user does not exceed the threshold:

$$\begin{aligned} &\textsc{trainMode}(\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta) := \\ &E[\mathbb{1}(\textsc{Conf}(\mathbf{x}, \hat{y}, c^{\mathrm{p}}_{\hat{y}}) > c^{\mathrm{u}}_y \cdot \theta)] \leq \theta/2 \end{aligned} \tag{3}$$

where $\hat{y} = \textsc{Pred}(\mathbf{x})$ is the predicted label for input $\mathbf{x}$, y is the label provided by the user for that input, $\mathbb{1}(\varphi)$ evaluates to one if φ is true and zero otherwise, and the expectation is taken over all inputs seen so far. The user is contradicted when the confidence in the predicted label exceeds a factor θ of the confidence of the user in her own label.

Once the system enters the refinement mode, it keeps asking the user for labels, but it starts to compare them with its predictions. The SOLVECONFLICT procedure deals with this comparison, and will be described in detail later on in this section. The refinement stage is stopped when the predictor is confident enough to stop asking for feedback to the user on every input, but selectively query the user on "difficult" cases. In general, it should be the user who decides when to switch mode, trading off system maturity and cognitive load. A simple fully automated option, similar to the one used for the train mode consists of staying in refine mode as long as the expected probability of querying the user exceeding the threshold:

$$\begin{aligned} &\textsc{refineMode}(\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta) := \\ &E[\mathbb{1}(\textsc{Conf}(\mathbf{x}, \hat{y}, c^{\mathrm{p}}_{\hat{y}}) \leq \theta)] \geq \theta/2 \end{aligned} \tag{4}$$

again with expectation taken over all inputs are seen so far.

When leaving the refine mode, the system enters the regime one, where it remains indefinitely. In this mode, the system stops asking feedback for all inputs, and an active learning strategy [37] begins. The system queries the user only if the confidence in a certain prediction is below the "safety" threshold θ. If the system decides to query the user, it includes the tentative label in the query, and then behaves as in refinement mode, calling SOLVECONFLICT to deal with the comparison between the predicted and the user labels.

3.3 The Conflict Management Algorithm

Algorithm 2 shows the SOLVECONFLICT procedure. The procedure takes as input the predictor and user confidence vectors $\mathbf{c}^{\mathrm{p}}$ and $\mathbf{c}^{\mathrm{u}}$, an input $\mathbf{x}$ with its predicted label ($\hat{y}$) and the label given by the user (y). It first compares the two labels according to the ISCOMPATIBLE procedure. In the simplest case, this outputs true if the two labels are identical, and false otherwise. In more complex scenarios, this procedure can use existing knowledge, as stored in the SK or in the GK, to decide whether two distinct labels are compatible, e.g., if the concept denoted by one is a generalization of the concept denoted by the other. In case the labels are compatible, a consensus label is taken as the ground truth, and the predictor and user confidences are updated accordingly. A natural choice for the consensus (and the one we use in our experiments) is being conservative and choosing the least general generalization of the two concepts.

A labeling conflict arises in the case of the two labels are not compatible. In case the confidence of the prediction is not large enough to contradict the user, the user label is taken as ground truth, the predictor is retrained with this additional feedback, and its confidence is updated accordingly. Otherwise, the system queries the user providing the two conflicting labels as input, asking her to solve the conflict. The user is free to stick to her own label, change her mind and opt for the label suggested by the predictor, or provide a third label as a compromise. As we are

Algorithm 2 Procedure for solving labeling conflicts.

```
procedure SOLVECONFLICT($\mathbf{c}^{\text{p}}, \mathbf{c}^{\text{u}}, \mathbf{x}, \hat{y}, y$)
    if ISCOMPATIBLE($\hat{y}, y$) then
        $y^* =$ CONSENSUS($\hat{y}, y$)
        UPDATE($\mathbf{c}^{\text{p}}, \hat{y}, y^*$)
        UPDATE($\mathbf{c}^{\text{u}}, y, y^*$)
    else if CONF($\mathbf{x}, \hat{y}, c^{\text{p}}_{\hat{y}}$) $\leq c^{\text{u}}_{y} \cdot \theta$ then
        TRAIN($f, \mathbf{x}, y$)
        UPDATE($\mathbf{c}^{\text{p}}, \hat{y}, y$)
    else
        $y^* =$ ASKUSER($\hat{y}, y$)
        if not ISCOMPATIBLE($\hat{y}, y^*$) then
            TRAIN($\mathbf{x}_t, y^*$)
        UPDATE($\mathbf{c}^{\text{p}}, \hat{y}, y^*$)
        UPDATE($\mathbf{c}^{\text{u}}, y, y^*$)
```

assuming a non-adversarial the setting, the system eventually trusts the newly provided label (even if unchanged) which becomes the ground truth. At this point, a compatibility check is made in order to verify whether a retrain step is needed, and the predictor and user confidences are updated. Notice how this is only an initial choice which can be extended in many different ways. What is needed at this point in the algorithm is a trusted *oracle* which will give its final word on the dispute. This oracle can be the user herself asked for a second opinion, as discussed above, or some trusted learning algorithm, as in the experiments below, or some kind of social opinion by, e.g., some friends or people nearby, or any combination of these possibilities.

The UPDATE procedure takes as input a confidence vector, a tentative label ($\hat{y}$) and a ground truth label (y^*), and updates the confidence vector according to the relationship between the two labels. The new confidence vector is as a label-wise running average accuracy over the current and past predictions, for a certain window size d.

4 EXPERIMENT

We validate SSML on the data collected in our experiments with students, which aim at understanding the empirical gap concerning students' time allocation and academic performance by providing a detailed description of how their everyday behavior affects their academic performance.

4.1 Data collection

In order to collect data from students, we rely on a mobile application [48] that can simultaneously acquire data from up to thirty sensors, both hardware (e.g., GPS) and software (e.g., running applications), generating streams that are temporarily stored on the device and periodically synchronized with the SD Knowledge Component of SSML (see Fig. 1). The data collection is completely configurable in terms of the number of sensors and sensing parameters, e.g., collection frequency. For our experiment, we configured the application as follows:

- We asked the participants to install the application on their personal smartphone to generate truthful data. In fact, providing them with a second device could have altered the generated data, since they would not have used it in the very same way. Additionally, the fact that we collected data using many different smartphone models from different vendors support the generality of our approach and its usability in the wild;

- We collected data from all the sensors available in the application. A full list of these sensors is provided in Table 1. It is important to mention that not all the smartphones have all the sensors so that each user can have a different set of sensors running;
- Thanks to the configurability of the application, we optimized the sensing process to balance the amount of data generated and battery consumption. Since the participants were using their own devices, we couldn't drain their battery, even if this meant having less frequent sensor readings. The "Sampling Rate" column in table 1 shows the collection frequency for each sensor.

Table 1. Table showing the sensors we selected among the ones available in the application [48] in combination with the selected sampling rate for the experiment.

Sensor	**Sampling Rate**	**Sensor**	**Sampling Rate**
Acceleration	20Hz	Screen Status	On change
Linear Acceleration	20Hz	Flight Mode	On change
Gyroscope	20Hz	Audio Mode	On change
Gravity	20Hz	Battery Charge	On change
Rotation Vector	20Hz	Battery Level	On change
Magnetic Field	20Hz	Doze Modality	On change
Orientation	20Hz	Headset plugged in	On change
Temperature	20Hz	Music Playback	On change
Atmospheric Pressure	20Hz	Location	Once every minute
Humidity	20Hz	WIFI Network Connected to	On change
Proximity	On change	WIFI Networks Available	Once every minute
Detect Incoming Calls	On change	Bluetooth Device Available	Once every minute
Detect Outgoing Calls	On change	Bluetooth Device Available (Low Energy)	Once every minute
Detect Incoming Sms	On change	Running Application	Once every 5 seconds
Detect Outcoing Sms	On change		
Incoming Notifications	On change		

The [48] application also allows administering time diaries to the participants asking about their activities, location and social relations at fixed time intervals. The way these labels are collected accounts only for the user perception of the world while she is not aware of the collected sensor values. The collected answers are then used as user labels inside the machine learning algorithms of the SSML architecture.

To initialize the experiment, 312 students enrolled in the first academic year in any of the bachelor courses active in 2016 at our University were contacted through a web survey to ask for their participation. From this initial population, 104 students fulfilled three specific criteria: *i)* to have filled three university surveys in order to obtain their socio-demographics, shown in Table 2, and other characteristics, e.g., psychological and time use related; *ii)* to attend lessons during the period of our project in order to describe their daily behavior during the university experience, and *iii)* to have a smartphone with an Android version 5.0.2 or higher.

Overall, 75 students accepted to participate, but three of them declined during the project due to unexpected technological incompatibility with the application. In the end, the final sample consisted of 72 students to reflect the general population of freshman year students of our University in terms of gender and departments.

The students were asked to attend an introductory presentation where they were presented with the aims of the project and how to use the application. Those who decided to participate after this presentation were presented with

Table 2. Socio-demographics of students from the experiment

Gender		Departments		Scholarship		Age	
Male	**Female**	**Scientific**	**Humanities**	**True**	**False**	**Min**	**Max**
61.1%	39.9%	56.9%	43.1%	37.5%	62.5%	19	22

a consent form to sign, after which they installed the application on their smartphones. Users were informed about all aspects of the management of their personal information concerning privacy, from data collection to storage to processing. Furthermore, before starting the data collection, we obtained approval from the ethical committee of our university.

The experiment lasted two weeks. During the first, students answered to time diaries on the application that collected their sensor data at the same time. During the second, students were only required to have the application running for the sensor data collection.

The resulting dataset contains 20 billion sensor values plus user annotations, resulting in a total size of 110GB. In addition, it is merged with both the pre and post project surveys collecting socio-demographic characteristics of students, their time use habits, some psychological traits measured by validated scales (i.e., pure procrastination scale and goal orientation scale), and academic performance data from the administrative office from our university. All the data were collected in compliance with the latest regulations in terms of privacy and are stored entirely anonymously.

4.2 Data processing and setup

Evaluating the SSML on this dataset required a pre-processing stage that generated a set of 122 feature vectors for each user, using all the available sensors inputs (the number depends on the smartphone, and consequently on the user). The features can be described based on the type of the data they refer to:

- *Periodic data* that include hardware sensors recording data every 1/20 seconds, mostly three-dimensional coordinates. To aggregate this type of data the *norm* of the axes for each row falling in the time slot is computed. This avoids discrepancies deriving from different smartphone positions/orientations at the time of the reading. Furthermore, unreasonable spikes in the reading are discarded. Finally, *mean* and *variance* are calculated from the resulting vector of norms;
- *On change data*, that represent discrete events generated by the device software. Features computed for this category include the *time* in milliseconds a particular state persists within the time slot, e.g., for how long a screen is on and off respectively. Additionally, a value denoting the *presence* of events and one keeping track of the *number of events* seen are recorded;
- Other types of data, including any sensor generating data that does not belong to the above categories. The most important features are the ones about the user location. These consist of a set of features indicating: i) proximity to one of the ten most common locations visited by the students during the experiment (one flag per location). Most common locations were computed using the DBSCAN clustering algorithm [9] using a maximum distance of $12.5m$ for considering two points part of the same neighborhood. To avoid single users with a high number of diverse locations to skew the results in their favor, the clustering was run an additional time on the clustering resulting from the first run, this time with a distance of $10m$. Among the resulting super-clusters, the 10 most common were eventually selected; ii) proximity to the home of the user. The home of the user was identified by running the DBSCAN algorithm with a distance of $108.27m$ (the average accuracy of the considered points) on those location points that were collected by the application in two situations: between 5:30AM and 7:30AM (the best moment according to [27]) and right before the GPS stops collecting data, past midnight, when it is reasonable to assume that the user just reached home to

go to sleep. Among the resulting clusters, we chose the largest cluster's centroid to be the home location. From this point, the feature we considered is the distance calculated in meters from the current position; iii) none of the above. Additionally, the distance between the first and the last detected points and the total distance travelled within the time interval are recorded. For the data relative to proximity networks (WiFi and Bluetooth) information such as *number of unique devices* in range (if at all) is recorded. Finally, for the running applications, for each window the features represent *which categories of applications run for how much time in milliseconds*, (eg. Social : 10000, Tools : 25000) and *the number of events* recorded.

The features were calculated using a window size of 30 minutes, which is the time between two consecutive annotations. The analysis focuses on the locations the participants visited during the two weeks of the experiment. We decided to focus on locations because it is easier to verify the correctness of such data with respect to, for example, activities. To this aim, we created an additional element, *the oracle*, which provides ground truth labels independently of both the predictor and the user annotations. The oracle relies on information regarding the location of the University buildings, and identifies the home of a user by clustering the locations she labels as home via DBSCAN [9] and choosing the cluster where she spends most of the time during the night. Note that SSML has no access to this information. The oracle is used for the evaluation of the performance of the system in predicting actual labels, and to simulate a non-adversarial, collaborative user as detailed in the next section. By comparing the labels provided by the users with the ones generated by the oracle, we discovered that 65 out of 72 participants provided more than 10% wrong labels, while 35 of them provided 30% wrong ones. In order to highlight the potential advantage of the SSML framework in highly noisy scenarios, we focused on the subset of users with more than 20% labeling errors (42 users). Note that this proportion of errors is substantially higher than the one usually expected in machine learning applications. We implemented the predictor as a random forest classifier [2] (with batch training), which is known to be robust to labeling noise [14], in order to evaluate the ability of SSML to improve over an already noise-robust baseline. For simplicity, we used an infinite window ($d = \infty$) for the confidence update, also given the relatively short duration of the experiment. The confidence parameter θ was set to 0.2 in order to achieve a reasonable trade-off between accurate training and reasonable cognitive effort for the user, considering the complexity of the learning task.

5 RESULTS AND EVALUATION

The results are organized into five sections. We first discuss the performance of SSML in the most straightforward setting where no prior knowledge is exploited. We then evaluate the system on a more complex scenario, including schematic knowledge from SK in the form of part-of relationships between locations. The two successive sections deepen the analysis of the results in this latter scenario, focusing respectively on the difference between ground truth and user labels, and on the behavior of the system for different types of users. Finally, we report the results of a third task, namely predicting the means of transportation of users, where we test the ability of SSML to provide improvements in a few-shot learning setting.

5.1 SSML Performance with no prior knowledge

The first experiment we performed is aimed at evaluating the robustness of SSML to labeling noise, without using semantic information on user labels, as obtained from the prior knowledge stored within the system. We collapsed the original labels into three classes, *Home*, *University* and *Others*. In this setting, the oracle can provide accurate ground truth values for all labels, as Others is recognized as not being Home and University. The procedure isCompatible inside Algorithm 2 in this case simply returns true if the two labels are identical and false otherwise. Given that we are working on previously collected data, we cannot query users in real-time in case a conflict arises. To simulate a collaborative, non-adversarial user, we replace the askUser procedure in Algorithm 2 with a call to the oracle, i.e., the user returns the ground truth label when her initial label is contested.

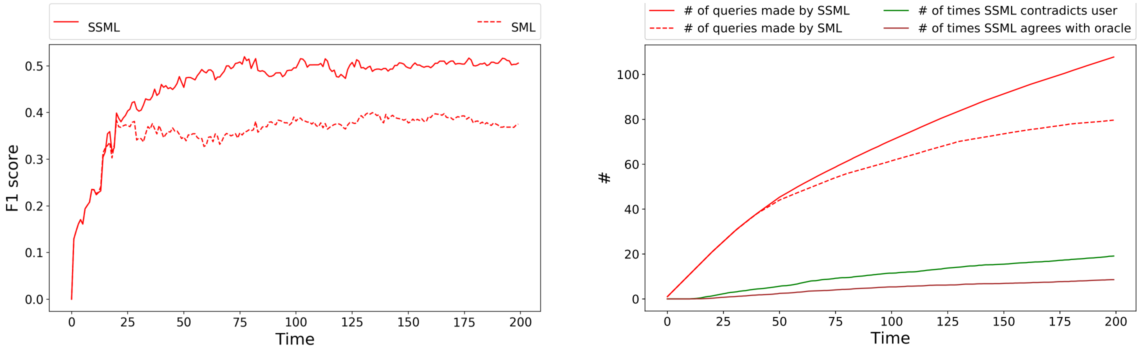


Fig. 2. Results with no prior knowledge: (a) f_1 scores for an increasing number of iterations, for SSML (solid red) and SML (dashed red) respectively. (b) number of queries made by SSML (solid red) and SML (dashed red), number of times the user is contradicted by SSML (green) and number of times SSML ends up being right (brown). The Time axes represent the number of iterations the algorithm is going through.

Figure 2 reports the results of SSML as compared to a solution never contradicting the user (obtained by replacing solveConflict with a train and update step, as happens in the training phase). We refer to this baseline simply as SML, standing for (non-skeptical) Supervised Machine Learning. Results were averaged over all users with a number of training samples greater than 200. Figure 2(a) reports the f_1 score of the SSML and SML predictors for increasing number of iterations. The score was computed on a fixed test set, namely the latest 15% of the all data available for each user, which was not used for training. This provides an estimate of the performance of the algorithms when doing predictions on future data. The results indicate that with a slight increase of queries to the user, the system achieves a 34.2% relative improvement in performance (from $f_1 = 0.38$ to $f_1 = 0.51$). Figure 2(b) reports the number of queries made to the user by SSML and SML respectively (red solid and red dashed). It also reports, for the SSML case, the number of times the user was contradicted (green) and in how many of these cases the user (as simulated by the oracle) ended up agreeing with the predictor (brown). It is interesting to notice that most of the times in which SSML contradicts the user, the oracle agrees with it.

When looking at the performance of individual users, we observe very different behaviors. We range from a very accurately modeled user ($f_1 = 0.96$ at the last iteration) to one where the predictor ends up always predicting the same label ($f_1 = 0.12$). Note that in the former case the conflict management strategy is responsible for the performance, as the non-skeptical alternative learns a degenerate model predicting the same label most of the times. In some cases, SSML too learns a degenerate model. This happens because a long stream of annotations with the same label forces the model to focus on it with very high confidence, move to the regime mode and stop interacting with the user. Smarter initialization strategies could help to avoid this behavior. Our aim here is not to fine-tune the system for this specific problem, but rather highlight the potential of challenging user feedback in general interactive learning scenarios. A more detailed analysis of the behavior of the system with various ”prototypical” users is provided in Section 5.4.

5.2 SSML Performance with prior knowledge

The second experiment is aimed at evaluating the impact of semantics on the overall performance of the SSML, in particular at the Schematic Knowledge level (see Figure 1). To do so, we created a set of labels including all the thirteen classes in the original dataset, plus three novel labels representing superclasses, as shown in Table 3. The latter were the same as the ones used in the first experiment, namely *Home*, *University* and *Others*. Semantics

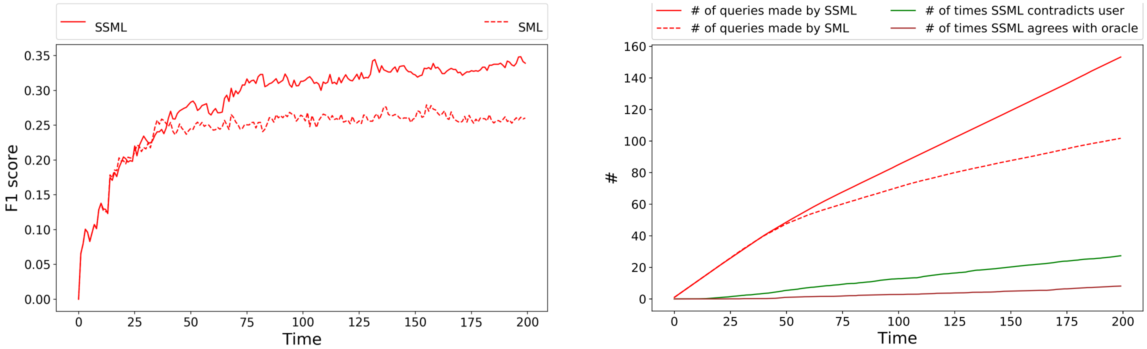


Fig. 3. Results with prior knowledge: (a) f_1 scores for increasing number of iterations, for SSML (solid red) and SML (dashed red) respectively. (b) number of queries made by SSML (solid red) and SML (dashed red), number of times the user is contradicted by SSML (green) and number of times SSML ends up being right (brown). The Time axes represent the number of iterations the algorithm is going through.

Table 3. Table showing the activity labels that the users in the experiment could select and the mapping with the three superclasses we defined.

Bar	Gym	Shop	Outdoors	Workplace	Other Home	Home	Class	Canteen	Study hall	Library
Others						Home	University			

was introduced in terms of part-of relationships, such as *Class* and *Laboratory* being part-of *University*. In this setting, ISCOMPATIBLE returns true if the two labels belong to the same super-class (or if they are the same), false otherwise. The CONSENSUS procedure is implemented as a conservative choice that always returns the super-class, e.g, *University*, *Home* or *Others*. Concerning the oracle (and thus, the replacement of the ASKUSER procedure within Algorithm 2 as well as the one used to generate test labels), the ground truth label is the user provided one if compatible with the one detected by the oracle, otherwise it is the label of the oracle.

Figures 3(a) and 3(b) report the results of this experiment, for f_1 score and number of queries respectively. The overall performance in this setting is lower than in the non-semantic case, despite a higher number of queries to the user, because the larger number of classes substantially increases the complexity of the labeling task. Nonetheless, the trend is very similar, with a relative performance improvement of 30.7% (from $f_1 = 0.26$ to $f_1 = 0.34$).

5.3 Objective vs. Subjective Labelling

Our algorithm aims to learn to predict the ground truth labels, despite being fed with user-provided labels in the first place. We can think of the former, the ones provided by the oracle, as *objective labels* and of the latter, the ones provided by the user when not acting as an oracle, as *subjective labels*. In this section, we investigate the performance of the SSML algorithm with respect to these two types of labels.

Figure 4 shows the average f_1 scores in four different settings, in the scenario including semantic information (the one described in Section 5.2). The solid red curve and the dashed red curve show the same results as in Figure 3(a). They represent f_1 scores of the SSML and SML algorithms, computed on the test set using objective labels as the ground truth. The solid blue curve and the dashed blue curve represent again f_1 scores of the SSML and SML algorithm respectively but computed on the test set using subjective rather than objective labels as the ground truth.

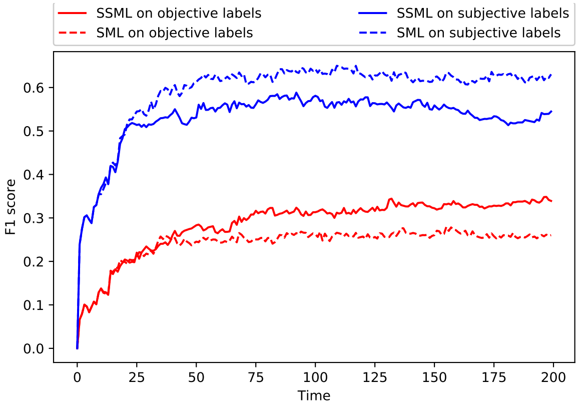
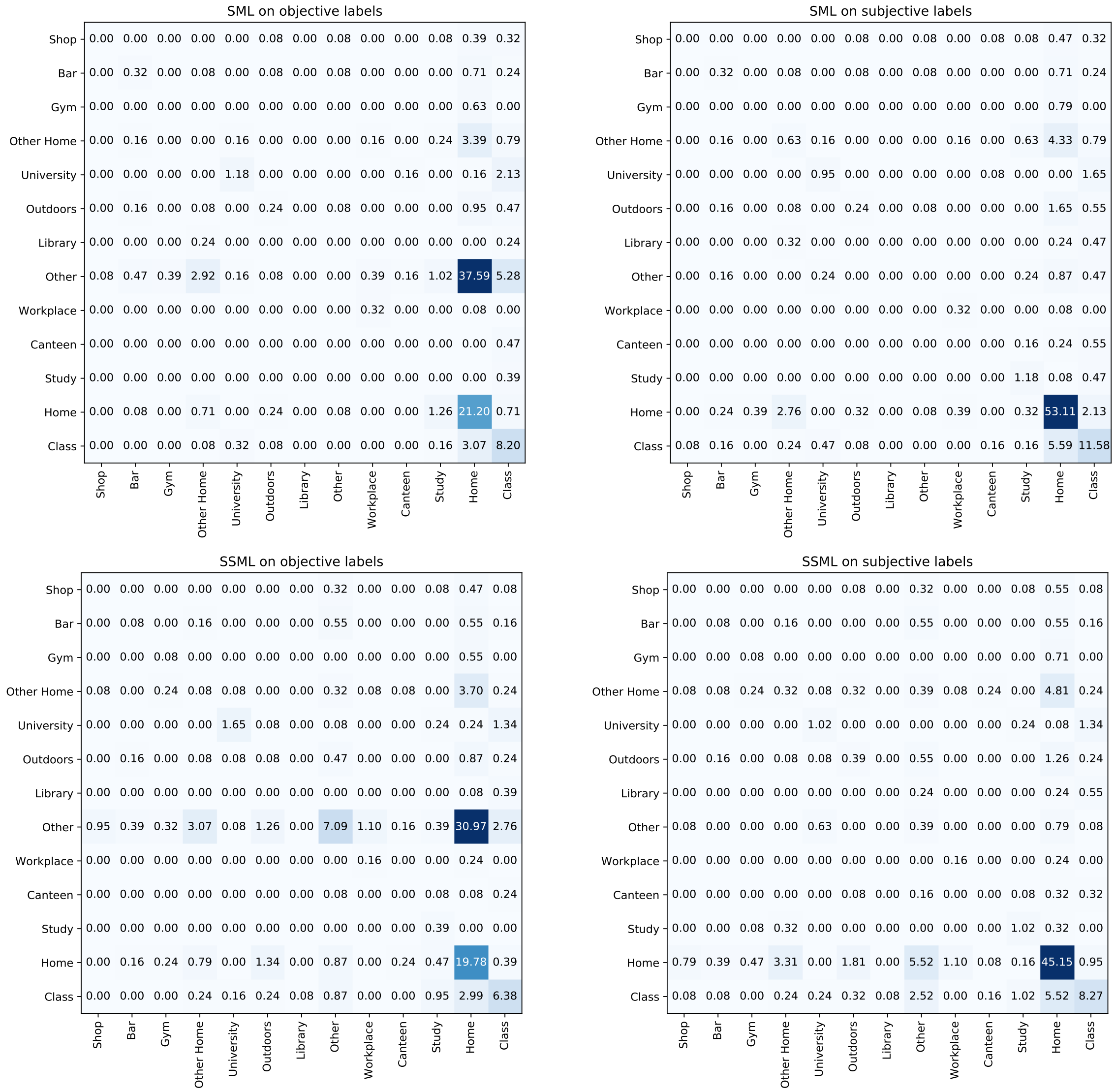


Fig. 4. Average f_1 score of the SSML and SML algorithms on all the users in the experiment. The solid lines represent the performance of the SSML algorithm, whereas the dashes ones represent the performance of the SML baseline. For red lines, the evaluation on the test set was made in terms of objective (i.e., ground truth) labels, while in the case of blue lines the evaluation was made in terms of subjective (i.e., user provided) labels. The Time axes represent the number of iterations the algorithm is going through.

These results provide some interesting insights into the performance of the algorithm. Overall, the performance evaluated on subjective labels (blue curves) is higher than the one evaluated on objective labels (red curves). This means that learning subjective labels is easier than learning objective labels; this is is an expected result, given that the algorithm often receives subjective labels as feedback. When comparing the performance of the SSML algorithm (solid curves) with respect to the SML one (dashed curves), we observe opposite behaviors depending on whether we evaluate them in terms of objective or subjective labels. The SSML algorithm is effective in improving prediction quality on objective labels (solid red curve vs. red dashed curve), as discussed in Sections 5.1, 5.2. Conversely, the conflict management phase of the SSML algorithm is harmful when the evaluation is made in terms of subjective labels (solid blue curve vs. blue dashed curve). This is also an expected result and indicates that challenging the user is pointless if the final goal is adapting to her subjective viewpoint rather than trying to make it match any type of objective knowledge. The decision of which choice to make must be made in a more global setting *i)* as a function of the specific type of knowledge, *ii)* the final goal to be achieved, and *iii)* what the user wants to do with the machine learning results. Thus, for instance, a certain place could objectively be closer to the user than another one, but subjectively farther, given the user's specific knowledge of the paths to both places (including taking some shortcuts not known to the general public). Figure 5 reports the confusion matrices for all settings. The first row refers to SML and the second to SSML, while the first and second columns refer respectively to the objective and subjective labels. Results are normalized over the sum of all the elements in the matrix. The fact that all matrices are quite sparse explains the low f_1 score of the overall results presented in Figure 4. Overall, the *Home* and *Other* labels are the most common. Comparing the matrices on objective and subjective labels, we can see that true positives on *Home* on subjective labels are significantly higher than on objective ones. In the latter case, there are many cases in which the algorithm predicts *Home*, but the actual label was *Other*. This is a clear example of misleading feedback from the user, who always answers *Home* when being in some *Other* location. By looking at the first column, we see that our SSML algorithm is capable of partially fixing this problem, bringing true positives for the *Other* label from 0.0% to 7.09%. This can explain the gap between SSML and SML on objective labels (red curves) in Figure 4.



SML on objective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.08	0.00	0.00	0.08	0.39	0.32
Bar	0.00	0.32	0.00	0.08	0.00	0.08	0.00	0.08	0.00	0.00	0.00	0.71	0.24
Gym	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.63	0.00
Other Home	0.00	0.16	0.00	0.00	0.16	0.00	0.00	0.00	0.16	0.00	0.24	3.39	0.79
University	0.00	0.00	0.00	0.00	1.18	0.00	0.00	0.00	0.00	0.16	0.00	0.16	2.13
Outdoors	0.00	0.16	0.00	0.08	0.00	0.24	0.00	0.08	0.00	0.00	0.00	0.95	0.47
Library	0.00	0.00	0.00	0.24	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24
Other	0.08	0.47	0.39	2.92	0.16	0.08	0.00	0.00	0.39	0.16	1.02	37.59	5.28
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.47
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.39
Home	0.00	0.08	0.00	0.71	0.00	0.24	0.00	0.08	0.00	0.00	1.26	21.20	0.71
Class	0.00	0.00	0.00	0.08	0.32	0.08	0.00	0.00	0.00	0.00	0.16	3.07	8.20

SML on subjective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.08	0.00	0.08	0.08	0.47	0.32
Bar	0.00	0.32	0.00	0.08	0.00	0.08	0.00	0.08	0.00	0.00	0.00	0.71	0.24
Gym	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.79	0.00
Other Home	0.00	0.16	0.00	0.63	0.16	0.00	0.00	0.00	0.16	0.00	0.63	4.33	0.79
University	0.00	0.00	0.00	0.00	0.95	0.00	0.00	0.00	0.00	0.08	0.00	0.00	1.65
Outdoors	0.00	0.16	0.00	0.08	0.00	0.24	0.00	0.08	0.00	0.00	0.00	1.65	0.55
Library	0.00	0.00	0.00	0.32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.47
Other	0.00	0.16	0.00	0.00	0.24	0.00	0.00	0.00	0.00	0.00	0.24	0.87	0.47
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.24	0.55
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.18	0.08	0.47
Home	0.00	0.24	0.39	2.76	0.00	0.32	0.00	0.08	0.39	0.00	0.32	53.11	2.13
Class	0.08	0.16	0.00	0.24	0.47	0.08	0.00	0.00	0.00	0.16	0.16	5.59	11.58

SSML on objective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.47	0.08
Bar	0.00	0.08	0.00	0.16	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.55	0.16
Gym	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.55	0.00
Other Home	0.08	0.00	0.24	0.08	0.08	0.00	0.00	0.32	0.08	0.08	0.00	3.70	0.24
University	0.00	0.00	0.00	0.00	1.65	0.08	0.00	0.08	0.00	0.00	0.24	0.24	1.34
Outdoors	0.00	0.16	0.00	0.08	0.08	0.08	0.00	0.47	0.00	0.00	0.00	0.87	0.24
Library	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.39
Other	0.95	0.39	0.32	3.07	0.08	1.26	0.00	7.09	1.10	0.16	0.39	30.97	2.76
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.00	0.00	0.24	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.08	0.08	0.24
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.39	0.00	0.00
Home	0.00	0.16	0.24	0.79	0.00	1.34	0.00	0.87	0.00	0.24	0.47	19.78	0.39
Class	0.00	0.00	0.00	0.24	0.16	0.24	0.08	0.87	0.00	0.00	0.95	2.99	6.38

SSML on subjective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.32	0.00	0.00	0.08	0.55	0.08
Bar	0.00	0.08	0.00	0.16	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.55	0.16
Gym	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.71	0.00
Other Home	0.08	0.08	0.24	0.32	0.08	0.32	0.00	0.39	0.08	0.24	0.00	4.81	0.24
University	0.00	0.00	0.00	0.00	1.02	0.00	0.00	0.00	0.00	0.00	0.24	0.08	1.34
Outdoors	0.00	0.16	0.00	0.08	0.08	0.39	0.00	0.55	0.00	0.00	0.00	1.26	0.24
Library	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.00	0.00	0.00	0.24	0.55
Other	0.08	0.00	0.00	0.00	0.63	0.00	0.00	0.39	0.00	0.00	0.00	0.79	0.08
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.00	0.00	0.24	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.16	0.00	0.00	0.08	0.32	0.32
Study	0.00	0.00	0.08	0.32	0.00	0.00	0.00	0.00	0.00	0.00	1.02	0.32	0.00
Home	0.79	0.39	0.47	3.31	0.00	1.81	0.00	5.52	1.10	0.08	0.16	45.15	0.95
Class	0.08	0.08	0.00	0.24	0.24	0.32	0.08	2.52	0.00	0.16	1.02	5.52	8.27

Fig. 5. Confusions matrices of the SSML and SML algorithms computed at the last iteration reported in Figure 4. The first row refers to SML (dashed lines in Figure 4) while the second one to SSML (continuous lines in Figure 4). The first column refers to objective labels (red lines in Figure 4) and the second to subjective ones (blue lines in Figure 4). The scale is normalized, and the numbers represent the percentage of labels among the total. In each matrix, rows represent actual labels (objective labels on the left, subjective labels on the right) and columns represent predicted ones.

5.4 Variability of users

In this section, we present a breakdown of the performance of the SSML algorithm with respect to a recognizable pattern of user behaviour. We analysed the performance graphs of the different users and identified four patterns

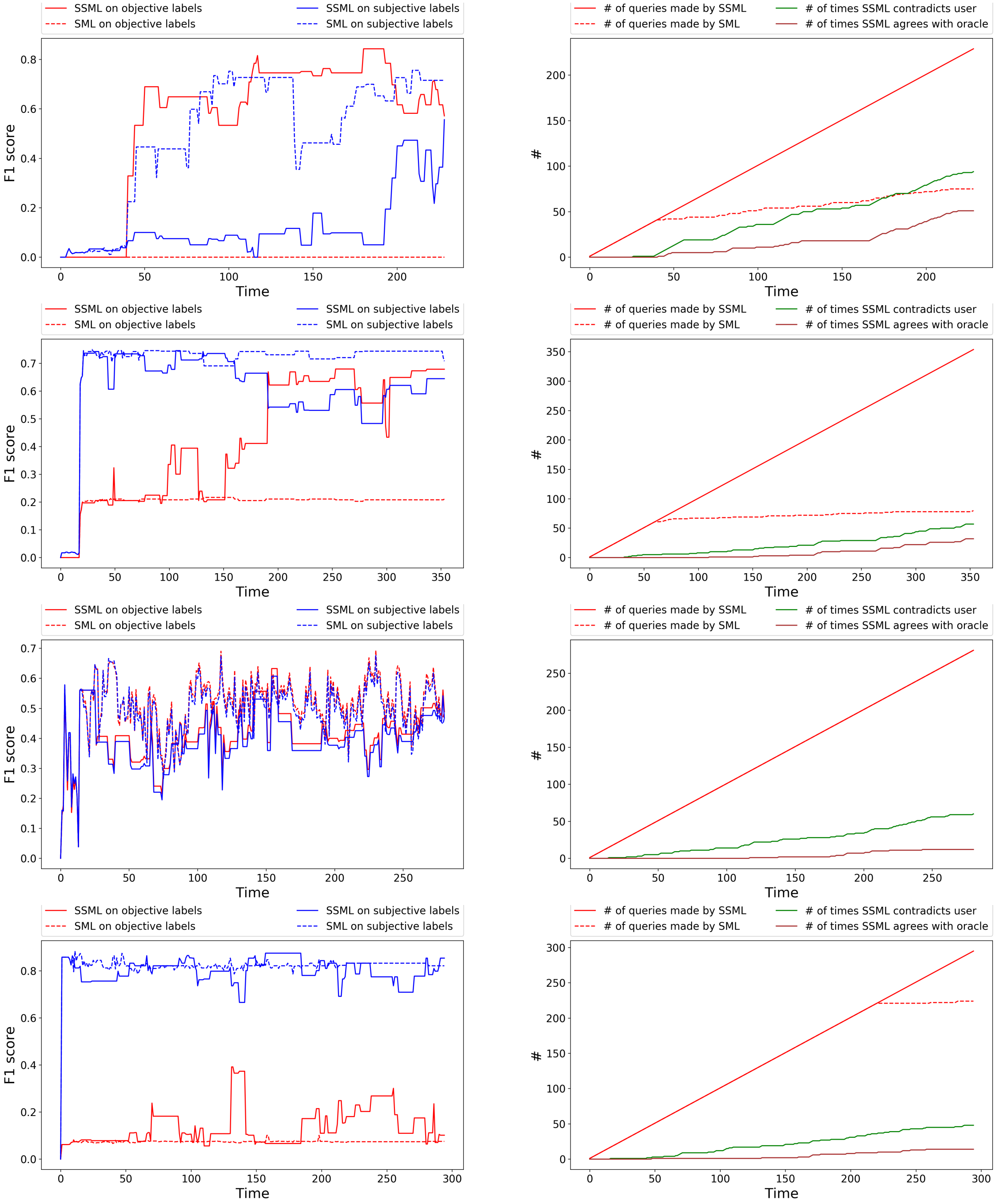


Fig. 6. Results for four different prototypical users: the first row shows an *inattentive user*; the second one a *predictable user*; the third one a *reliable user* and the last one a *tricksy user*. The images on the left report the f_1 scores in different settings while the ones on the right report information about the number of queries and agreement with the user and the oracle. The Time axes represent the number of iterations the algorithm is going through.

as highly common. These performance patterns can be related to some distinct behavioural patterns, which we discuss in the following. Figure 6 reports the results for these prototypical users, where each row refers to a specific user. Left figures report f_1 scores in different settings, which are the very same as in Figure 4 (i.e., objective vs subjective labels, SSML vs SML). Right figures report information on the number of queries and agreement with the user and with the oracle, same as in Figures 2(b) and 3(b).

Inattentive user The results of the first row in Figure 6 show that the highest score is achieved by the SSML algorithm evaluated on objective labels. All other settings achieve substantially lower performance. This behavior can be explained in terms of an *inattentive* user, who often provides subjective labels which are different from the objective ones (difference between red and blue curves), and is largely inconsistent in the initial feedback, which makes predicting subjective labels harder than objective ones (solid red vs. dashed blue), even if most of the feedback is subjective anyhow. The inconsistency of the user is also reflected in the right graph of the first row of Figure 6, showing a rather large fraction of times in which the user is contradicted (because there is no agreement) and the predictor agrees with the oracle. This is the type of user for which the SSML algorithm is most beneficial. Note that the fact that SSML manages to correct user inconsistencies indicates that the system reaches a confidence which is sufficient to start challenging the user, i.e., the user is a "detectably" inconsistent one.

Predictable user The second case is a particularly interesting one. For the first 20 iterations, both SSML and SML are completely incapable of predicting the user labels. After this initial step, the algorithm learns to predict subjective labels with a high accuracy (solid red against solid blue). This happens because the user is consistent in providing feedback, but her subjective labels are largely different from the objective ones. At a certain point, the system starts challenging the user and soon afterward (around iteration 190), the system learns to predict objective labels with an higher accuracy with respect to subjective ones. We refer to this user as "predictable". Albeit subjective labels are mostly different from objective ones for this user, both of them can be predicted with high accuracy, once the system receives the appropriate feedback. This is confirmed by the high number of times in which predictor and oracle agree, as shown in the right figure. A predictable user is thus another case in which the benefits of SSML are substantial, even if it takes some time for the system to figure out the discrepancy between subjective and objective labels.

Reliable user The third row of Figure 6 shows the results of a user for which the performance of the SSML and SML algorithms are basically the same. This is because the user is already reliable in providing initial feedback, as can be seen by the substantial overlap between the red and blue curves. Indeed, the user is contradicted only occasionally (green curve in right figure), and even rarer are the cases in which the oracle agrees with the predictor against the subjective label of the user (brown curve). This is a user for whom SSML is not helpful, but also not harmful, as it ends up asking about the same number of queries as SML (solid and dashed red curves in the right figure). Both systems never exit the refinement phase and keep asking for feedback, because the location of this specific user keeps changing and, most importantly, almost all the labels in the label space are selected.

Tricksy user The last one is a case in which the SSML algorithm completely fails to predict user behavior. By looking at the difference between red and blue curves, it is apparent that the problem is in the difference between subjective and objective labeling. The algorithm keeps learning the former, even when given the chance to question user labeling. The right figure shows that this chance is rarely taken by the algorithm, and almost never leads to discovering the cases in which subjective and objective labels disagree. The user here succeeds in fooling the system by convincing it of the correctness of her own feedback, namely the fact of being at home way more often than what was actually the case. Note that additional prior knowledge would substantially help the system in figuring out that something strange is going on. Indeed, GPS information clearly shows that "home" is in too many places for this user. We kept this information hidden from the

Table 4. Table showing the means of transportation considered in the experiment and the mapping with the three superclasses used.

Label	On foot	Bus	Train	Car	Motorbike	Bike
Superclass	On foot	In vehicle				On bicycle

system in this study because we used it to implement the oracle. However, in a real setting with actual users as oracles, this information would be part of the prior knowledge of the system and contribute to its capacity to identify inconsistencies in user feedback.

Additional types of users could be identified, e.g., an "average" user behaving similarly to Figure 4, or an "unpredictable" one for whom the system was incapable of learning neither subjective nor objective labels. Overall, these results highlight how the performance of SSML is strongly affected by the behavior of the user, and call for additional work to deal with the difficult cases. For instance, finding a way to make the system aware of cases when the user is not being helpful would allow it to start searching for some additional source of information, e.g., nearby users, in order to compensate for the lack of reliable feedback.

5.5 SSML Performance in Few-Shot Learning

In order to evaluate the generality of the skeptical learning algorithm, we ran a second test aimed at recognizing the means of transport of users during their movements. We decided to focus specifically on movement activities since as for the location experiment, we could simulate a collaborative user with an oracle replacing the ASKUSER procedure in Algorithm 2, as will be shown in the following. During the data collection process, the user could indicate one of six possible means of transportation if in movement, as shown in Table 4. We created a hierarchy of labels by introducing three novel labels representing superclasses, namely *On foot*, *In vehicle* and *On bicycle*. In this setting, ISCOMPATIBLE works as for the experiment presented in Section 5.3, it returns true if the two labels belong to the same super-class or if they are the same, false otherwise. The CONSENSUS procedure is again implemented as a conservative choice that always returns the super-class, e.g., *On foot*, *In vehicle*, and *On bicycle*. We created an oracle for the prediction of means of transportation leveraging on the Google data that users were asked to provide at the end of the data collection experiment. By default, Google keeps track of the users' movements and detects how the user was moving at fixed time intervals. At every detection, a list of possible labels (namely *On foot*, *In vehicle*, *On bicycle*, *Still*, *Tilting*, *Exiting vehicle* and *Unknown*) is provided with an attached confidence value, from 0 to 100. In processing Google data, we ignored the labels *Unknown*, *Still*, *Tilting* and *Exiting vehicle*, which have no correspondence in the set of candidate user answers, and focused on *On foot*, *In vehicle* and *On bicycle*. Google data, when available, have a frequency of a label per minute on average. On the other hand, users provide labels once every 30 minutes. We combined Google provided labels into oracle labels by taking the most common label over the 30-minute interval. As for the prediction of location with prior knowledge, the oracle only provides labels for the superclasses of the hierarchy (see Table 4). As for that setting, the ground truth label is the user provided one if compatible with the one detected by the oracle; otherwise, it is the label of the oracle. Out of 72 users participating in the experiment, only 31 of them provided Google data. Furthermore, movement data are sparser than location ones, with just 15 iterations labelled as moving activities on average for each user over the two weeks of the experiment. This is in line with the expected behavior of an average student that passes most of her time either at home or at the University, commuting only few times a day for short trips. This challenging setting is known as few-shot learning in the machine learning literature and allows us to test the ability of SSML to achieve improvements even with just a handful of labeled examples.

Results are shown in Figure 7. As for the previous settings, the left graph reports the f1 score of the SSML and SML predictors for increasing number of iterations, while the right graph reports the number of queries made to the

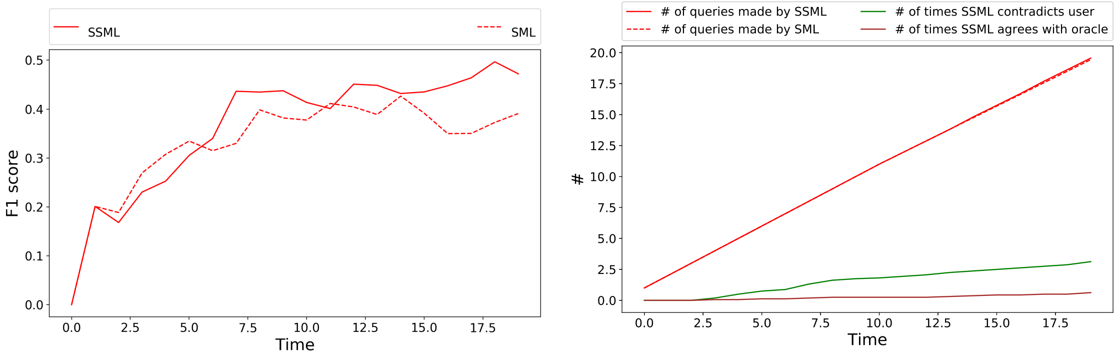


Fig. 7. Results of the task of predicting the means of transportation: (a) f1 scores for increasing number of iterations, for SSML (solid red) and SML (dashed red) respectively. (b) the number of queries made by SSML (solid red) and SML (dashed red), number of times the user is contradicted by SSML (green) and the number of times SSML ends up being right (brown). The Time axes represent the number of iterations the algorithm is going through.

user by SSML and SML, the number of times SSML contradicts the user and in how many of these cases, the user (as simulated by the oracle) ends up agreeing with the predictor. While having larger oscillations and closer curves because of the minimal number of examples, the f1 prediction results show a clear trend towards an advantage of SSML over SML. The difference in the number of iterations (i.e., training instances) between these graphs and those in Figures 2 and 3 can give an idea of the amount of supervision available in this setting and confirm the capability of SSML to exploit even very limited feedback to improve prediction performance. Figure 8 reports confusion matrices of the SML (left) and SSML (right) algorithms at the last iteration. The advantage of SSML is due to its ability to better predict the *On foot* and *Bus* labels, at the cost of a moderate decrease in the accuracy of the *Train* and *Car* ones.

6 DISCUSSION AND FUTURE WORK

In this paper, we have addressed the issue of the *untrustfulness of human annotators* within ubiquitous experiments by integrating the model-driven input from the user with the data-driven input provided by machine learning. The primary goal was to exploit this integration to minimize the impact of labeling mistakes both on the machine and on the user side. The experimental results are promising, and the general idea seems generalizable in many dimensions. Some such examples are: allowing for the possibility to ask third parties (e.g., a friend or the crowd) whenever a conflict arises, allowing for multiple machine learning algorithms and for various inputs from the crowd while providing a uniform way to measure their truthfulness, dealing with adversarial learning [5, 29] and adversarial label contamination [46]. These dimensions will enable a much more powerful role of semantics in the future work, leading to building *evolvable* knowledge, contrary to its static nature in this work, that *adapts* and *evolves* in order to accommodate for the ever coming new knowledge, as it is the case in our everyday lives.

ACKNOWLEDGMENTS

This work was supported by the European Commission Horizon 2020 Research and Innovation Project QROWD, (http://qrowd-project.eu/) under grant agreement No 732194.

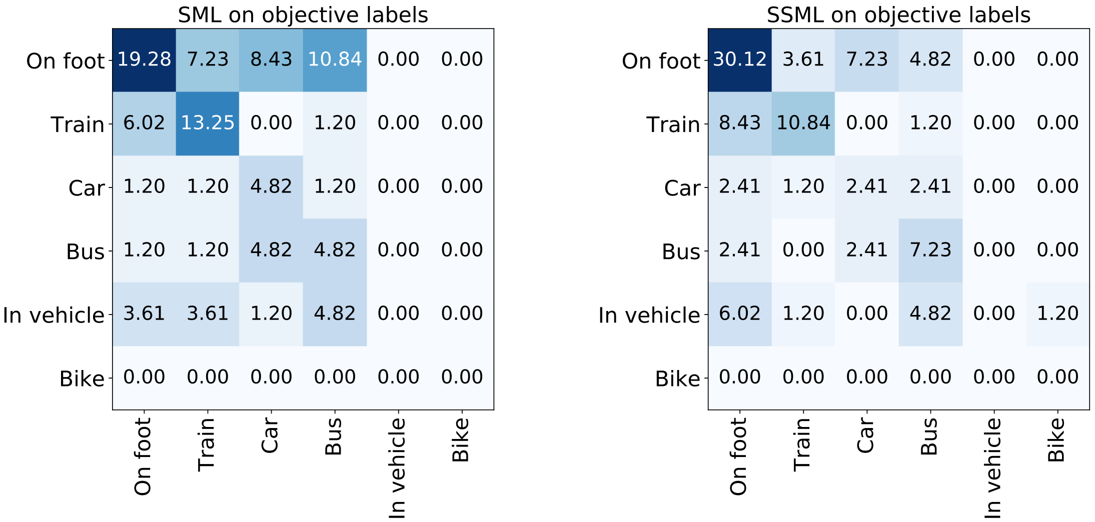


Fig. 8. Confusions matrices of the SML (left) and SSML (right) algorithms computed at the last iteration reported in Figure 7. The scale is normalized, and the numbers represent the percentage of labels among the total. In each matrix, rows represent actual objective labels and columns represent predicted ones.

REFERENCES

[1] Gükhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. 2007. *Predicting Structured Data (Neural Information Processing)*. The MIT Press.

[2] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (01 Oct 2001), 5–32. https://doi.org/10.1023/A:1010933404324

[3] Simone Centellegher, Marco De Nadai, Michele Caraviello, Chiara Leonardi, Michele Vescovi, Yusi Ramadian, Nuria Oliver, Fabio Pianesi, Alex Pentland, Fabrizio Antonelli, et al. 2016. The Mobile Territorial Lab: a multilayered and dynamic view on parents' daily lives. *EPJ Data Science* 5, 1 (2016), 3.

[4] Louise Corti. 1993. Using diaries in social research. *Social research update* 2, 2 (1993).

[5] N Dalvi, P Domingos, Mausam, S Sanghai, and D Verma. 2004. Adversarial Classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*. ACM, New York, NY, USA, 99–108. https://doi.org/10.1145/1014052.1014066

[6] Thomas G. Dietterich. 2000. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning* 40, 2 (01 Aug 2000), 139–157. https://doi.org/10.1023/A:1007607513941

[7] Michelangelo Diligenti, Marco Gori, and Claudio Saccà. 2017. Semantic-based regularization for learning and inference. *Artificial Intelligence* 244 (2017), 143 – 165. https://doi.org/10.1016/j.artint.2015.08.011 Combining Constraint Solving with Mining and Learning.

[8] Nathan Eagle and Alex Sandy Pentland. 2006. Reality mining: sensing complex social systems. *Personal and ubiquitous computing* 10, 4 (2006), 255–268.

[9] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *Kdd*, Vol. 96. 226–231.

[10] Geli Fei, Shuai Wang, and Bing Liu. 2016. Learning Cumulatively to Become More Knowledgeable.. In *KDD*. 1565–1574.

[11] A. Folleco, T. M. Khoshgoftaar, J. Van Hulse, and L. Bullard. 2008. Identifying learners robust to low quality data. In *2008 IEEE International Conference on Information Reuse and Integration*. 190–195. https://doi.org/10.1109/IRI.2008.4583028

[12] Vicki A Freedman, Jessica Broome, Frederick Conrad, and Jennifer C Cornman. 2013. Interviewer and respondent interactions and quality assessments in a time diary study. *Electronic international journal of time use research* 10, 1 (2013), 55.

[13] Benoît Frénay, Ata Kabán, et al. 2014. A comprehensive introduction to label noise.. In *ESANN*.

[14] Aritra Ghosh, Naresh Manwani, and P. S. Sastry. 2017. On the Robustness of Decision Tree Learning Under Label Noise. In *Advances in Knowledge Discovery and Data Mining*, Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon (Eds.). Springer International Publishing, Cham, 685–697.

[15] Fausto Giunchiglia, Enrico Bignotti, and Mattia Zeni. 2017. Personal context modelling and annotation. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on*. IEEE, 117–122.

[16] Fausto Giunchiglia, B Khuyagbaatar, and B Gabor. 2017. Understanding and exploiting language diversity. IJCAI.

[17] Fausto Giunchiglia, Mattia Zeni, and Enrico Bignotti. 2018. Personal Context Recognition via Reliable Human-Machine Collaboration. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2018 IEEE International Conference on*. IEEE, in print.

[18] Fausto Giunchiglia, Mattia Zeni, Enrico Bignotti, and Wanyi Zhang. 2018. Assessing Annotation Consistency in the Wild. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2018 IEEE International Conference on*. IEEE, in print.

[19] Fausto Giunchiglia, Mattia Zeni, Elisa Gobbi, Enrico Bignotti, and Ivano Bison. 2018. Mobile social media usage and academic performance. *Computers in Human Behavior* 82 (2018), 177–185.

[20] Bin Guo, Zhiwen Yu, Xingshe Zhou, and Daqing Zhang. 2014. From participatory sensing to mobile crowd sensing. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*. IEEE, 593–598.

[21] Gabriella M Harari, Sandrine R Müller, Min SH Aung, and Peter J Rentfrow. 2017. Smartphone sensing methods for studying behavior in everyday life. *Current Opinion in Behavioral Sciences* 18 (2017), 83–90.

[22] Mattias Hellgren. 2014. Extracting More Knowledge from Time Diaries? *Social Indicators Research* 119, 3 (2014), 1517–1534.

[23] Stephen S Intille, Ling Bao, Emmanuel Munguia Tapia, and John Rondoni. 2004. Acquiring in situ training data for context-aware ubiquitous computing applications. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1–8.

[24] F Thomas Juster and Frank P Stafford. 1991. The allocation of time: Empirical findings, behavioral models, and problems of measurement. *Journal of Economic literature* 29, 2 (1991), 471–522.

[25] Salil S Kanhere. 2011. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*, Vol. 2. IEEE, 3–6.

[26] Nicky Kern, Bernt Schiele, and Albrecht Schmidt. 2007. Recognizing context for annotating a live life recording. *Personal and Ubiquitous Computing* 11, 4 (2007), 251–263.

[27] John Krumm. 2007. Inference attacks on location tracks. In *International Conference on Pervasive Computing*. Springer, 127–143.

[28] Afra J Mashhadi and Licia Capra. 2011. Quality control for real-time ubiquitous crowdsourcing. In *Proceedings of the 2nd international workshop on Ubiquitous crowdsouring*. ACM, 5–8.

[29] DJ Miller, X Hu, Z Qiu, and G Kesidis. 2017. Adversarial Learning: A Critical Review and Active Learning Study. *CoRR* abs/1705.09823 (2017). arXiv:1705.09823 http://arxiv.org/abs/1705.09823

[30] David F. Nettleton, Albert Orriols-Puig, and Albert Fornells. 2010. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review* 33, 4 (01 Apr 2010), 275–306. https://doi.org/10.1007/s10462-010-9156-z

[31] Sarath A Nonis, Melodie J Philhours, and Gail I Hudson. 2006. Where does the time go? A diary approach to business and marketing students' time use. *Journal of Marketing Education* 28, 2 (2006), 121–134.

[32] Robi Polikar. 2006. Ensemble based systems in decision making. *IEEE Circuits and systems magazine* 6, 3 (2006), 21–45.

[33] Catia Prandi, Stefano Ferretti, Silvia Mirri, and Paola Salomoni. 2015. Trustworthiness in crowd-sensed and sourced georeferenced data. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*. IEEE, 402–407.

[34] Gunnar Rätsch, Bernhard Schölkopf, Alexander Johannes Smola, Sebastian Mika, Takashi Onoda, and Klaus-Robert Müller. 2000. Robust Ensemble Learning for Data Mining. In *Knowledge Discovery and Data Mining. Current Issues and New Applications*, Takao Terano, Huan Liu, and Arbee L. P. Chen (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 341–344.

[35] Francesco Restuccia, Nirnay Ghosh, Shameek Bhattacharjee, Sajal K Das, and Tommaso Melodia. 2017. Quality of Information in Mobile Crowdsensing: Survey and Research Challenges. *ACM Transactions on Sensor Networks (TOSN)* 13, 4 (2017), 34.

[36] Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning* 62, 1 (01 Feb 2006), 107–136. https://doi.org/10.1007/s10994-006-5833-1

[37] Burr Settles. 2009. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison. http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf

[38] B Settles. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 1 (2012), 1–114.

[39] Shai Shalev-Shwartz. 2012. Online Learning and Online Convex Optimization. *Found. Trends Mach. Learn.* 4, 2 (Feb. 2012), 107–194. https://doi.org/10.1561/2200000018

[40] Pitirim Aleksandrovich Sorokin and Clarence Quinn Berger. 1939. *Time-budgets of human behavior*. Vol. 2. Harvard University.

[41] Stefano Teso, Roberto Sebastiani, and Andrea Passerini. 2017. Structured learning modulo theories. *Artificial Intelligence* 244 (2017), 166 – 187. https://doi.org/10.1016/j.artint.2015.04.002 Combining Constraint Solving with Mining and Learning.

[42] Roger Tourangeau, Lance J Rips, and Kenneth Rasinski. 2000. *The psychology of survey response*. Cambridge University Press.

[43] Rui Wang, Fanglin Chen, Zhenyu Chen, Tianxing Li, Gabriella Harari, Stefanie Tignor, Xia Zhou, Dror Ben-Zeev, and Andrew T Campbell. 2014. StudentLife: assessing mental health, academic performance and behavioral trends of college students using smartphones. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 3–14.

[44] Rui Wang, Gabriella Harari, Peilin Hao, Xia Zhou, and Andrew T Campbell. 2015. SmartGPA: how smartphones can assess and predict academic performance of college students. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous*

Computing. ACM, 295–306.

[45] Brady T West and Jennifer Sinibaldi. 2013. The quality of paradata: A literature review. *Improving Surveys with Paradata* (2013), 339–359.

[46] Huang Xiao, Battista Biggio, Blaine Nelson, Han Xiao, Claudia Eckert, and Fabio Roli. 2015. Support vector machines under adversarial label contamination. *Neurocomputing* 160 (2015), 53 – 62. https://doi.org/10.1016/j.neucom.2014.08.081

[47] Shuo Yang, Fan Wu, Shaojie Tang, Xiaofeng Gao, Bo Yang, and Guihai Chen. 2017. On designing data quality-aware truth estimation and surplus sharing method for mobile crowdsensing. *IEEE Journal on Selected Areas in Communications* 35, 4 (2017), 832–847.

[48] Mattia Zeni, Ilya Zaihrayeu, and Fausto Giunchiglia. 2014. Multi-device activity logging. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 299–302.