# Coactive Critiquing: Elicitation of Preferences and Features

**Stefano Teso**
stefano.teso@unitn.it
University of Trento
Via Sommarive 9, Povo
Trento, Italy

**Paolo Dragone**
paolo.dragone@unitn.it
University of Trento
TIM-SKIL
Via Sommarive 9, Povo
Trento, Italy

**Andrea Passerini**
andrea.passerini@unitn.it
University of Trento
Via Sommarive 9, Povo
Trento, Italy



**Abstract**

When faced with complex choices, users refine their own preference criteria as they explore the catalogue of options. In this paper we propose an approach to preference elicitation suited for this scenario. We extend Coactive Learning, which iteratively collects manipulative feedback, to optionally query example critiques. User critiques are integrated into the learning model by dynamically extending the feature space. Our formulation natively supports constructive learning tasks, where the option catalogue is generated on-the-fly. We present an upper bound on the average regret suffered by the learner. Our empirical analysis highlights the promise of our approach.

## Introduction

Preference elicitation (Goldsmith and Junker 2009) is the task of interactively inferring preferences of users and it is a key component of personalized recommendation and decision support systems. The typical approach consists of asking the user to rank alternative solutions (Chajewska, Koller, and Parr 2000; Boutilier et al. 2006; Guo and Sanner 2010; Viappiani and Boutilier 2010) and use the resulting feedback to learn a (possibly approximately) consistent user utility function. These algorithms rely on a fixed pool of solutions from which to choose both candidates for feedback and final recommendations. However, when thinking of an interaction between a user and a salesman, one imagines a more active role by the user, who could suggest modifications to candidates. For instance, in a trip planning application, when commenting a candidate trip to New York, the user may reply: "I'd rather visit the MoMA than Central Park". This is especially true when considering fully *constructive* scenarios (Teso, Passerini, and Viappiani 2016), where the task is synthesizing entirely novel objects, like the furniture arrangement of an apartment or a novel recipe for vegan tiramisù. Coactive Learning (Shivaswamy and Joachims 2012) is a recent interactive learning paradigm which allows the user to provide corrected versions of the candidates she is presented with.

While Coactive Learning approaches adapt the preference model based on user-provided option improvements, the set of features that the utility is defined by is assumed given and fixed. This is not always a realistic assumption. When faced with a complex decision, users may not be fully aware of their own quality criteria, especially in large, unfamiliar decision domains (Chen and Pu 2012; Pu and Faltings 2000). Even more so in constructive settings, where the option catalogue is exponentially (possibly infinitely) large and generated on-the-fly. Crucially, the user may become aware of novel preference criteria, in a context-specific fashion, while exploring the decision domain (Payne, Bettman, and Johnson 1993; Slovic 1995).

One way to tackle this problem is to enumerate all potential user criteria in advance, by combining a fixed set of features with one or more operators (e.g. multiplication or logical conjunction). This solution however has drawbacks. First, the number of feature combinations suffers from combinatorial explosion, making learning harder and more computationally demanding. Most importantly, entirely novel and unanticipated user criteria can not be added to the feature space.

Example critiquing (or conversational) recommendation systems (Tou et al. 1982; McGinty and Reilly 2011; Chen and Pu 2012) provide an alternative solution. In this setting, preferences are stated in term of critiques to suggested configurations. Upon receiving one or more proposals, the user is free to reply with statements such as "this trip is too expensive" or "I dislike crowded places". Critiques are integrated into the learner as auxiliary constraints or penalties (Faltings et al. 2004). Options presented at later iterations are chosen based on the collected feedback, focusing the search on more promising items. Example critiquing is explicitly designed to address the above difficulty: by being confronted with a set of concrete items, the user has a chance to realize that she cares about features that she was previously unaware of (Chen and Pu 2012). Unfortunately, typical conversational systems do not support numerical modelling of user preferences (e.g. weights), and often assume noiseless critiquing feedback.

In this paper we present a new algorithm, Coactive Critiquing (CC), that unifies *coactive* learning and example *critiquing*, harnessing the strengths of both strategies. Coactive Critiquing builds on the coactive learning framework by further allowing critique feedback. We view critiques as arbitrarily articulated explanations for the user-provided im-

provements, e.g. the user may explain her reason for suggesting the MoMA over Central Park by stating: "I prefer indoor activities during winter". In this work, we assume that there is an interface between the algorithm and the user which translates the user's critiques into (soft) constraints[1]. Newly acquired constraints are included into the learning problem as additional features. We extend the regret bounds of Shivaswamy and Joachims (2015) to the more general case of growing feature spaces. Our empirical findings highlight the promise of Coactive Critiquing in a synthetic and a realistic preference elicitation problem, highlighting its ability in offering a reasonable trade-off between the quality of the recommendations and the cognitive effort expected from the user.

In the next section we position our work within the related literature. In the Method section we motivate, detail and analyze our proposed method. We describe our empirical findings in the Empirical Evaluation section, and conclude with some final remarks.

## Related Work

There is a large body of work on preference elicitation (Goldsmith and Junker 2009). Due to space restrictions, we focus on the techniques that are most closely related to our approach.

Coactive Learning (CL) is an interaction model for learning user preferences from observable behavior (Shivaswamy and Joachims 2012), recently employed in learning to rank and online structured prediction tasks (Shivaswamy and Joachims 2015; Sokolov, Riezler, and Cohen 2015). For an overview of the method, see the next section. The underlying weight learning procedure can range from a simple perceptron (Rosenblatt 1958) to more specialized online learners (Shivaswamy and Joachims 2015). Further extensions include support for approximate inference (Goetschalckx, Fern, and Tadepalli 2014) and multitask learning (Goetschalckx, Fern, and Tadepalli 2015). These extensions are orthogonal to our main contribution, and may prove useful when used in tandem. However, in this paper, we only consider the original formulation, for simplicity. Our approach inherits several perks from CL, including a theoretical characterization of the average regret (Shivaswamy and Joachims 2015) and native support for constructive tasks. The main difference between the two methods, which is also our main contribution, is that in CC the feature space grows dynamically through critiquing interaction. CL instead works with a static feature space, and is therefore incapable of handling users with varying preference criteria.

The concept of critiquing interaction originated in interactive recommender and decision support systems (Chen and Pu 2012; McGinty and Reilly 2011; Tou et al. 1982). Critiquing systems invite the user to critique the suggested configurations, thus supporting the exploration and understanding of the decision domain. Collected critiques play the role of constraints (or penalties) in filtering the available options, allowing the search to focus on the more promising candidates. Our approach is most closely related to user-initiated critiquing protocols, where at each iteration the user articulates one or more critiques (Chen and Pu 2012). In CC critiques are elicited at specific iterations only, selected by a heuristic balancing cognitive cost and expressivity of the acquired feature space (as discussed in the Methods section). Few critiquing recommenders model the user preferences numerically. In contrast, CC associates weights to both basic and acquired features (i.e. critiques). One exception is the method of Zhang and Pu (2006), which employs a learned linear utility model. The user chooses an option from a pool of 5 highest utility options. In this context, critiques are simple textual descriptions of the advantages of each suggestion over the reference option. The estimated utility is updated through a multiplicative update based on the user's pick. CC instead uses the (user-initiated) critiques to improve the expressivity of the feature space. Other critiquing recommenders that include an adaptive component are concerned with developing effective query selection strategies, e.g. (Viappiani, Pu, and Faltings 2007) and (Viappiani and Boutilier 2009).

## Method

We first introduce some notation. We indicate column vectors $\boldsymbol{a}$ in bold and vector concatenation as $\boldsymbol{a} \circ \boldsymbol{b}$. The usual dot product is denoted $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_i a_i b_i$ and the Euclidean norm as $\|\boldsymbol{a}\|$. Later on we will compute dot products between vectors of different lengths. In this case, the shorter vector is implicitly zero padded to the bottom to match the length of the longer one.

**Coactive Learning.** We consider a preference learning setting with coactive feedback; $\mathcal{X}$ is the set of feasible item configurations $x$; these are represented by an $m$-dimensional feature vector $\boldsymbol{\phi}^*(x)$. We assume that the feature vector length is bounded, $\|\boldsymbol{\phi}^*(x)\| \leq R$ for some constant $R$. The attractiveness, or subjective quality, of a configuration is measured by its *utility*, which we assume (Keeney and Raiffa 1976) to be expressible as a linear function of the features $u^*(x) := \langle \boldsymbol{w}^*, \boldsymbol{\phi}^*(x) \rangle = \sum_{i=1}^{m} w_i^* \phi_i^*(x)$. Here $\boldsymbol{w}^* \in \mathbb{R}^m$ encodes the true, unobserved user preferences. We write $x^*$ to indicate a maximal utility configuration. The goal of the system is to suggest high utility configurations without direct access to $\boldsymbol{w}^*$. A common strategy is to iteratively improve an estimate of the true preferences through interaction with the user, while keeping the user's cognitive cost at a minimum.

We follow the Coactive Learning (Shivaswamy and Joachims 2015) paradigm, which we describe briefly[2]. In Coactive Learning, the learner maintains an estimate $\boldsymbol{w}^t$ of the user preferences. At each iteration $t = 1, \ldots, T$, the algorithm computes a most preferable configuration

[1] For instance, it could be a simple form that allows the user to combine attribute values to form critiques. We are currently working on automated approaches based on NLP and rule mining.

[2] We only consider a "context-less" version of Coactive Learning, which is sufficient for our purposes; our method can be trivially extended to support contexts. See Shivaswamy and Joachims (2015) for further details.

$x^t \in \mathcal{X}$, by maximizing the current estimate of the utility $\langle \boldsymbol{w}^t, \boldsymbol{\phi}^*(x^t)\rangle$. The configuration is then presented to the user, who is tasked with providing an improved configuration $\bar{x}^t$, e.g. by direct manipulation of $x^t$. The two options $x^t$ and $\bar{x}^t$ provide an implicit ranking constraint $u^*(\bar{x}^t) > u^*(x^t)$. The latter is employed to update the preference estimate, in the simplest case with a perceptron update[3]:

$$\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t)$$

In the remainder we assume the user to be $\alpha$-*informative* (Shivaswamy and Joachims 2015): if the configuration $x^t$ is not optimal, the user can always produce an improvement $\bar{x}^t$ with higher true utility (modulo mistakes). Formally, $\alpha$-informativity implies that there exists a constant $\alpha \in (0, 1]$ such that, for all $t$, it holds that:

$$u^*(\bar{x}^t) - u^*(x^t) = \alpha \left(u^*(x^*) - u^*(x^t)\right) - \xi^t \qquad (1)$$

Improvement errors are absorbed by the (possibly negative) slack term $\xi^t \in \mathbb{R}$. Under this assumption, the *average regret* incurred by Coactive Learning after $T$ iterations, defined as:

$$REG_T := \frac{1}{T}\sum_{t=1}^{T} \left(u^*(x^*) - u^*(x^t)\right) \qquad (2)$$

is bounded from above as follows.

**Theorem 1** (Shivaswamy and Joachims 2015)**.** *For an $\alpha$-informative user with true preference vector $\boldsymbol{w}^*$ and bounded length feature vectors $\|\boldsymbol{\phi}^*(x)\| \leq R$, the average regret incurred by Coactive Learning after $T$ iterations is upper bounded by*

$$REG_T \leq \frac{2R}{\alpha\sqrt{T}}\|\boldsymbol{w}^*\| + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi^t$$

As a consequence, so long as the user is not too noisy, the slacks will be small enough, and the bound guarantees that the average regret will shrink accordingly. While similar bounds have been proposed for more general users (Shivaswamy and Joachims 2015), here we restrict ourselves to $\alpha$-informative users for simplicity. In our presentation we do not impose any restriction on the type of features used. We note in passing, however, that the choice of feature type can heavily impact the complexity of the inference step. There are however ways to make Coactive Learning work with approximate inference procedures (Goetschalckx, Fern, and Tadepalli 2014).

**Coactive Critiquing.** Coactive Learning presupposes the user and the learner to have unlimited access to the complete feature function $\boldsymbol{\phi}^*(x)$ at all times. This assumption is often unrealistic. It is well known that users may discover their own quality criteria while exploring the option catalogue (Payne, Bettman, and Johnson 1993); further, critique queries can be employed to stimulate the users to discover their own criteria (Faltings et al. 2004). We amend to this deficiency by augmenting Coactive Learning with support for example critiquing interaction.

[3]Other update strategies can be applied, see for instance (Shivaswamy and Joachims 2015); we will stick with the classical perceptron with unit step size for simplicity.

**Algorithm 1** Pseudo-code of the Coactive Critiquing algorithm. Here $\boldsymbol{\phi}^1$ is the initial feature space, and $T$ is the maximum number of iterations. User interaction occurs inside the QUERYIMPROVEMENT and QUERYCRITIQUE procedures.

```
1:  procedure CC(φ^1, T)
2:      w^1 ← 0,  D ← ∅
3:      for t = 1, ..., T do
4:          x^t ← argmax_{x∈X} ⟨w^t, φ^t(x)⟩
5:          x̄^t ← QUERYIMPROVEMENT(x^t)
6:          D ← D ∪ {(x^t, x̄^t)}
7:          if NEEDCRITIQUE(D, φ^t) then
8:              ρ ← QUERYCRITIQUE(x^t, x̄^t)
9:              φ^t ← φ^t ∘ [ρ]
10:             w^t ← w^t ∘ [0]
11:         end if
12:         w^{t+1} ← w^t + φ^t(x̄^t) − φ^t(x^t)
13:         φ^{t+1} ← φ^t
14:     end for
15:     return argmax_{x∈X} ⟨w^T, φ^T(x)⟩
16: end procedure
```

At a high level, Coactive Critiquing works as shown in Algorithm 1. The algorithm maintains estimates of both the user preferences $\boldsymbol{w}^t$ and feature function $\boldsymbol{\phi}^t(x)$. The initial set of features $\boldsymbol{\phi}^1(x)$ is supposedly taken from a reasonable default set, provided by a domain expert, by the user herself (e.g. through a questionnaire), or other sources (Chen and Pu 2012). At each iteration $t$, the algorithm performs an improvement query, as in Coactive Learning (lines 4, 5), but can additionally submit a *critique* query to the user. Critiques are only queried when specific conditions are met (line 7), as described in the next subsection.

Given the proposed and improved configurations, $x^t$ and $\bar{x}^t$ respectively, a query critique (line 8) amounts to asking the user why she thinks the improved configuration is preferable to the suggested one. Ideally, the user would respond with a critique $\rho$ that maximally explains the utility difference between the two configurations. This interaction protocol is based on a modest "local rationality" assumption: when presented with two distinct configurations $x^t, \bar{x}^t \in \mathcal{X}$, the user can state at least *one* critique that contributes a significant utility difference between the configurations. The user is free to reply with suboptimal critiques, according to her current awareness and the required cognitive effort. We will discuss the impact of suboptimal critiques in our theoretical analysis.

The feedback of the critique query consists of a single, arbitrary critique constraint $\rho$. We interpret the latter as a feature function $\rho(x)$ that captures whether (or how much) the constraint is satisfied. In principle, all kinds of features are acceptable, including indicators and numerical degrees of satisfaction. For instance, the critique "I prefer indoor activities during winter" would equate to a feature that indicates the conjunction of the season being winter and whether the trip includes one or more indoor activities. The feature $\rho(x)$ is appended to the current feature vector $\boldsymbol{\phi}^t(x)$; the weight vector $\boldsymbol{w}^t$ is padded accordingly by appending a zero element (lines 9 and 10). The learner traverses increasingly

more expressive feature spaces $\boldsymbol{\phi}^t, t = 1, \ldots, T$, as critiques are collected. The perceptron update remains the same as in coactive learning (line 12). The algorithm terminates after a fixed number of iterations $T$, or when the user is satisfied (e.g. when the regret of the current suggestion $x^t$ is small enough).

**When to ask for critiques.** Critique queries are key in improving the expressiveness of the feature space. Critiques are only elicited at the iterations selected by the NEEDCRITIQUE procedure (line 7). The design of this procedure is crucial. On one hand, if the procedure is too lazy, not enough critiques are elicited, impairing the representation ability of the traversed feature spaces. This may in turn make it impossible to learn the true utility $u^*(x)$. On the other hand, if the procedure is too eager, the algorithm may end up eliciting more critiques than necessary, thus wasting cognitive effort. We will show in the next section that, unsurprisingly, the design of the procedure affects the regret incurred by the learner.

In order to balance between the two, we design a simple selection criterion, as follows. The idea is to submit a critique query as soon as algorithm realizes the true utility can no longer be represented in the current feature space. Since we do not have access to the true utility, we use the collected ranking feedback (i.e. the dataset, indicated as $\mathcal{D}$ in Algorithm 1) as a proxy. To decide whether to ask for a critique or not, we check for the existence of a weight vector $\boldsymbol{w}$ that correctly ranks the pairwise preference examples in $\mathcal{D}$, i.e. more formally: $\exists \boldsymbol{w} \forall (x, \bar{x}) \in \mathcal{D}\ \langle \boldsymbol{w}, \boldsymbol{\phi}(\bar{x}^t) - \boldsymbol{\phi}(x^t) \rangle > 0$.

This criterion is guaranteed to work in noiseless scenarios. When the user is noisy though, a vector $\boldsymbol{w}$ satisfying the ranking constraints may not exist in any subspace of $\boldsymbol{\phi}^*(x)$. In this case, Coactive Critiquing may end up querying for a critique at every iteration. We did not experience this problem in practice. We also designed a more sophisticated criterion, based on estimating the likelihood of inconsistencies in the dataset being due to noise or lack of features. However, we did not see any improvements using this strategy in our empirical tests.

**Theoretical analysis.** Theorem 1 assumes that the feature space is fixed. In coactive critiquing, however, this is not the case. Our goal is to extend the theorem to this more general case.

In Coactive Learning, at each iteration $t$, the utility gain provided by two configurations $\bar{x}^t$ over $x^t$ is $u^*(\bar{x}^t) - u^*(x^t)$, and is lower bounded by the $\alpha$-informativity assumption (Eq 1). Our algorithm however works in a lower dimensional feature space than the user's one, and has access to the partial utility $u^t(x) = \langle \boldsymbol{w}^*, \boldsymbol{\phi}^t(x) \rangle$ only. In the lower dimensional space, the utility gain amounts to $u^t(\bar{x}^t) - u^t(x^t)$, so it "misses out" on the contribution of the unobserved features. We write $\eta^t$ to denote the missing part, quantified as:

$$\begin{aligned} \eta^t &:= (u^*(\bar{x}^t) - u^*(x^t)) - (u^t(\bar{x}^t) - u^t(x^t)) \\ &= \sum_{i=k^t+1}^{m} w_i^* \left[\phi_i^*(\bar{x}^t) - \phi_i^*(x^t)\right] \end{aligned} \tag{3}$$

where $k^t$ is the number of features acquired up to iteration $t$. Note that $\eta^t$ can be either positive or negative, depending on whether ignoring the missing features worsens or improves the utility gain, respectively. The latter case can occur when the $\boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t)$ update is negatively correlated with $\boldsymbol{w}^*$ with respect to the missing features.

We formalize this intuition in the following proposition, which is an adaptation of Theorem 1.

**Proposition 2.** *For an $\alpha$-informative user with true preference vector $\boldsymbol{w}^*$ and bounded length feature vectors $\|\boldsymbol{\phi}^*(x)\| \leq R$, the average regret incurred by Coactive Critiquing after $T$ iterations is upper bounded by*

$$REG_T \leq \frac{2R}{\alpha\sqrt{T}}\|\boldsymbol{w}^*\| + \frac{1}{\alpha T}\sum_{t=1}^{T}\left(\xi^t + \eta^t\right)$$

See the Appendix for the proof. The sum $\sum_{t=1}^{T} \eta^t$ on the right hand side depends on the effectiveness of the user's critiques and how often they are asked, as well as the problem structure. The latter factor is beyond our control, but the former can be (partially) controlled by properly designing the interaction with the user. By explicitly asking for the critique $\rho$ contributing the most to the utility gain, we are effectively removing the largest summand from $\eta^t$ (in practice, the user errors may make it decrease by a smaller amount). Furthermore, the amount of critiques may reduce the sum of the $\eta^t$, at the price of additional cognitive effort for the user. In the next section we will show that our proposed NEEDCRITIQUE heuristic offers a good trade-off.

## Empirical Evaluation

We evaluate Coactive Critiquing on two preference elicitation tasks. All experiments were run on a 2.8 GHz Intel Xeon CPU with 8 cores and 32 GiB of RAM. Our implementation makes use of MiniZinc (Nethercote et al. 2007) with the Gecode backend. The CC source code and the full experimental setup are available at: `goo.gl/cTFOFq`.

**User simulation.** We simulated the user feedback as follows. In improvement queries, the user is asked to produce an improvement $\bar{x}^t$ of the suggested configuration $x^t$. A real user would choose the improved configuration by balancing between cognitive effort and perceived quality of the improvement. To account for this fact, our simulated user (line 5 of Algorithm 1) computes the improvement by finding a minimal change to $x^t$ with improved true utility. This is done by solving the combinatorial problem:

$$\begin{aligned} \bar{x}^t := \operatorname*{argmin}_{\bar{x} \neq x^t} \quad & \|\bar{x} - x^t\| \\ \text{s.t.} \quad & \langle \boldsymbol{w}^* + \boldsymbol{\varepsilon}, \boldsymbol{\phi}^*(\bar{x}) \rangle > \langle \boldsymbol{w}^* + \boldsymbol{\varepsilon}, \boldsymbol{\phi}^*(x^t) \rangle \end{aligned}$$

Here $\|\bar{x} - x^t\|$ measures the difference between $\bar{x}$ and $x^t$, and $\boldsymbol{\varepsilon} \in \mathbb{R}^m$ is a normally distributed ($\sigma = 0.1$) perturbation that simulates user noise. The user is $\alpha$-informative as per Eq 1. In order not to artificially advantage our method, our simulated user returns a minimal improvement, consequently providing a minimal utility gain.

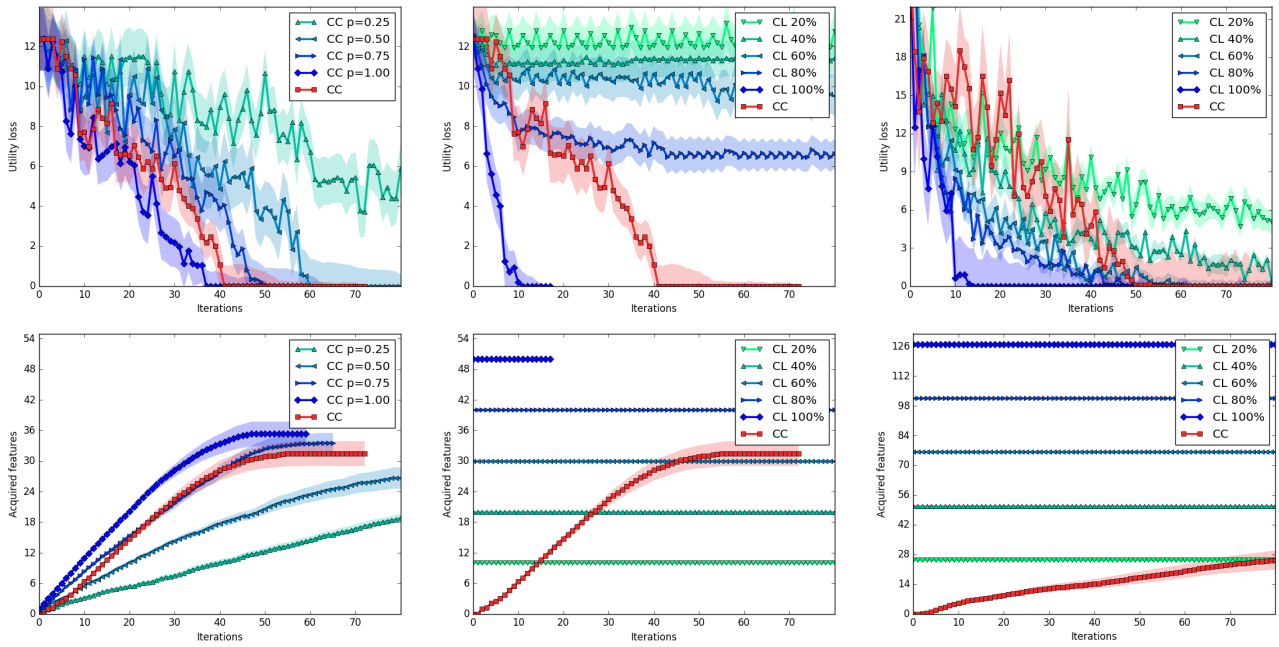


Figure 1: Left: comparison of CC for different choices of NEEDCRITIQUE procedure; median utility loss at the top, average number of acquired features at the bottom. Middle: comparison between CC and CL on the synthetic problem. Right: comparison between CC and CL on the trip planning problem. Best viewed in color.

In critiquing queries (line 8), the user is asked to return the critique $\rho$ contributing the most to the utility gain of $\bar{x}^t$ over $x^t$. Formally, the contribution of feature $\boldsymbol{\phi}_i^*$ is $c_i := w_i^*(\phi_i^*(\bar{x}^t) - \phi_i^*(x^t))$. Ideally, the user would respond with the feature $\rho$ with the highest contribution (with ties broken at random). In practice, she may choose a sub-optimal critique. We simulate user noise by sampling $\rho$ from a multinomial distribution where the probability of choosing $\phi_i^*$ is set to $c_i / \sum_i c_i$. This model favors features with higher contribution, while still leaving room for sub-optimal choices.

**Synthetic Experiment.** First, we evaluate our method on a synthetic task. The configurations are 2D points $x$ with integer coordinates, taking values in a discrete bounding box of size $100 \times 100$, for a total of $10^4$ feasible configurations. There are 50 rectangles $r_1, \ldots, r_{50}$ inside the bounding box. The position and size of the rectangles are sampled uniformly at random once and kept fixed for all runs. Each feature $\phi_i^*(x)$, $i = 1, \ldots, 50$, acts as an indicator for the corresponding rectangle $r_i$: it evaluates to 1 if $x$ is within the rectangle, and to $-1$ otherwise. The true weights $\boldsymbol{w}^* \in \mathbb{R}^{50}$ establish a preference over the rectangles: if $w_i^* > 0$ the user prefers configurations contained in $r_i$, and outside of it otherwise. It can be readily seen that most features are uncorrelated, and thus a sufficiently expressive subset of features is needed to find an optimal solution. The inference and improvement simulators were implemented as mixed integer-linear problems and solved accordingly.

First, we compare our NEEDCRITIQUE heuristic against an uninformed baseline randomly choosing when to ask for critiques. Specifically, we replace our heuristic at line 7 with a binomial distribution, varying the parameter $\theta \in \{0.25, 0.5, 0.75, 1\}$.

We run all methods over 20 users independently sampled from a 50-dimensional standard normal distribution. We compute the median utility loss $u^*(x^*) - u^*(x^t)$ over all users (the lower, the better) as well as the average number of acquired features. Execution times are omitted, as the difference between algorithms is negligible. We report the results in the left column of Figure 1. As shown by the plots, our heuristic strikes a good balance between user satisfaction and cognitive effort. In terms of utility loss, it fares in-between the $\theta = 1$ (most informed baseline) and the $\theta = 0.75$ (second most informed) variants, while eliciting fewer critiques than both. The other baselines are not up to par.

Next, we compare CC with our NEEDCRITIQUE heuristic against CL. CC always starts from 2 features and acquires new ones dynamically through query critiques. In contrast, CL has fixed access to $p\%$ of the features, for $p \in \{20, 40, \ldots, 100\}$. In order not to bias the results, for each $p$ we take the average of five different CL runs, each over a randomly drawn subspace of $\boldsymbol{\phi}^*(x)$ of the appropriate size. We refer to this setting as $\text{CL}^p$. Given that there is no standard, accepted way to estimate the real cognitive cost of replying to improvement or critique queries, we avoid computing a single unified measure of user effort and rather count the number of queries separately. We report the results in the middle column of Figure 1.

In median, $\text{CL}^{100}$ reaches zero loss after 11 iterations,

which is hardly surprising, considering its unrestricted (and unrealistic) access to the full feature space; CC instead takes 41 iterations. All other methods fail to converge. Notably, CC acquires about 30 features to reach zero median loss, and beats $\text{CL}^{80}$ in the same metric after 18 iterations, with 14 acquired features. These results highlight the effectiveness of CC in acquiring relevant features, with consequent savings of cognitive effort.

**Realistic Experiment.** We applied Coactive Critiquing to an interactive touristic trip planning task. We collected a dataset including 10 cities and 15 possible activities from the Trentino Open data website: `http://dati.trentino.it/`. The goal is to suggest a trip route $x$ between (some of) the cities. Each city has a particular offering of activities (e.g. luxury resorts, points of interest, healthcare services) and an overnight cost. Cities may be visited more than once. Traveling between cities takes a time proportional to their distance. In our experiments we set the trip length to 10.

We distinguish between base features $\boldsymbol{\phi}^1(x)$ and full features $\boldsymbol{\phi}^*(x)$. The former include the amount of time spent at each location and the time spent performing each activity, for a total of 25 base features. The latter include the number of distinct visited locations, the total time spent travelling, the total cost, the number of visited geographic regions, among others, for a total of 92 acquirable features. We omit the full list for space restrictions.

As in the synthetic experiment, we compare CC against variants of CL obtained by varying the percentage $p$ of available features over 20 users sampled from a 127-dimensional standard normal distribution. We report the results in the right column of Figure 1.

The problem is significantly more difficult than the synthetic one, due to the combinatorial size of the space of configurations. The plots show that CC is very critique-effective: by the last iteration it acquires about as many features as $\text{CL}^{20}$ (approx. 27), which is the least informed method, but it performs considerably better. The baselines $\text{CL}^{60} - \text{CL}^{100}$ converge faster, having access to most of the features from the beginning. Our approach performs comparably to $\text{CL}^{60}$ from iteration 50 onwards, notwithstanding the much fewer acquired features ($\sim$20 versus $\sim$75, respectively). Although $\text{CL}^{40}$ uses (from iteration 1) about twice the number of features eventually acquired by CC, it is surpassed by the latter roughly at the $40^{\text{th}}$ iteration.

## Conclusion

In this paper we described an approach to preference elicitation that combines Coactive Learning with example critiquing interaction. Contrary to coactive learning, the feature space is acquired dynamically through interaction with the user. We discussed the theoretical guarantees of the method, and a heuristic query selection strategy that balances between user effort and expressivity of the acquired feature space. We presented experimental evidence in support of our findings. Coactive Critiquing is competitive with more informed baselines, often requiring many less features to obtain comparable (or better) recommendations. Like conversational recommenders, Coactive Critiquing could in principle handle free-form textual or speech critiques, see for instance Grasch, Felfernig, and Reinfrank (2013).

Coactive Critiquing is especially suited for constructive preference elicitation tasks (Teso, Passerini, and Viappiani 2016). Given that the computational cost of inference can become prohibitive in these settings, it may be fruitful to integrate support for approximate inference, as discussed by Goetschalckx, Fern, and Tadepalli (2014). Another promising research direction involves allowing the user to reply with non-feasible improved configurations. In this case, the projection of the improvement on the feasible space may break $\alpha$-informativity. We are currently investigating how to tackle this issue.

**Acknowledgments** ST is supported by the CARITRO Foundation through grant 2014.0372. PD is a fellow of TIM-SKIL Trento and is supported by a TIM scholarship.

## Appendix A: Proof of Proposition 2

We split the proof in three steps.

(i) The update equation of Algorithm 1 (line 12) is

$$\boldsymbol{w}^{T+1} := \boldsymbol{w}^T + \boldsymbol{\phi}^T(\bar{x}^T) - \boldsymbol{\phi}^T(x^T)$$

We expand the dot product $\langle \boldsymbol{w}^{T+1}, \boldsymbol{w}^{T+1} \rangle$ using the above, obtaining

$$\langle \boldsymbol{w}^T, \boldsymbol{w}^T \rangle + 2\langle \boldsymbol{w}^T, \boldsymbol{\phi}^T(\bar{x}^T) - \boldsymbol{\phi}^T(x^T) \rangle + \langle \boldsymbol{\phi}^T(\bar{x}^T) - \boldsymbol{\phi}^T(x^T),\ \boldsymbol{\phi}^T(\bar{x}^T) - \boldsymbol{\phi}^T(x^T) \rangle$$

The optimality of $x^T$ in the current feature space $\boldsymbol{\phi}^T(x)$ (line 4) implies that the second term is no greater than zero. Given that $\|\boldsymbol{\phi}^T(x)\| \leq \|\boldsymbol{\phi}^*(x)\| \leq R$ by assumption, it follows that

$$\langle \boldsymbol{w}^{T+1}, \boldsymbol{w}^{T+1} \rangle \leq \langle \boldsymbol{w}^T, \boldsymbol{w}^T \rangle + 4R^2 \leq 4R^2 T \qquad (4)$$

(ii) Let $\boldsymbol{z}^T$ be a 0-1 vector, of the same shape as $\boldsymbol{w}^*$ such that the only non-zero elements of $\boldsymbol{z}^T$ are those corresponding to the features elicited up to iteration $T$. We expand the dot product $\langle \boldsymbol{w}^*, \boldsymbol{w}^{T+1} \rangle$ using the above update rule to obtain

$$\langle \boldsymbol{w}^*, \boldsymbol{w}^T \rangle + \langle \boldsymbol{w}^*, \boldsymbol{\phi}^T(\bar{x}^T) - \boldsymbol{\phi}^T(x^T) \rangle = \langle \boldsymbol{w}^*, \boldsymbol{w}^T \rangle + \langle \boldsymbol{w}^*, \boldsymbol{z}^T \odot \left[ \boldsymbol{\phi}^*(\bar{x}^T) - \boldsymbol{\phi}^*(x^T) \right] \rangle$$

where $\odot$ is the element-wise product. We unroll the recursion to get

$$\begin{aligned} \langle \boldsymbol{w}^*, \boldsymbol{w}^{T+1} \rangle &= \textstyle\sum_{t=1}^T \ \langle \boldsymbol{w}^*, \boldsymbol{z}^t \odot [\boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t)] \rangle \\ &= \textstyle\sum_{t=1}^T \ \langle \boldsymbol{w}^*, \boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t) \rangle - \\ &\qquad \langle \boldsymbol{w}^*, (\mathbf{1} - \boldsymbol{z}^t) \odot [\boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t)] \rangle \end{aligned}$$

By applying the definition of utility $u^*(x)$ to the first term and the definition of $\eta^T$ to the second one, we get

$$\langle \boldsymbol{w}^*, \boldsymbol{w}^{T+1} \rangle = \textstyle\sum_{t=1}^T [u^*(\bar{x}^t) - u^*(x^t)] - \sum_{t=1}^T \eta^t \qquad (5)$$

(iii) The Cauchy-Schwarz inequality states that $\langle \boldsymbol{w}^*, \boldsymbol{w}^{T+1} \rangle \leq \|\boldsymbol{w}^*\| \|\boldsymbol{w}^{T+1}\|$. We plug Equations 4 and 5 to obtain:

$$\textstyle\sum_{t=1}^T \langle \boldsymbol{w}^*, \boldsymbol{\phi}^*(\bar{x}^t) - \boldsymbol{\phi}^*(x^t) \rangle \leq 2R\sqrt{T}\|\boldsymbol{w}^*\| + \sum_{t=1}^T \eta^t$$

Now we use the $\alpha$-informativity assumption (Eq 1) and the definition of average regret (Eq 2) to obtain the claim.

## References


[2006] Boutilier, C.; Patrascu, R.; Poupart, P.; and Schuurmans, D. 2006. Constraint-based Optimization and Utility Elicitation using the Minimax Decision Criterion. *Artifical Intelligence* 170:686–713.

[2000] Chajewska, U.; Koller, D.; and Parr, R. 2000. Making rational decisions using adaptive utility elicitation. In *Proceedings of AAAI'00*, 363–369.

[2012] Chen, L., and Pu, P. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction* 22(1-2).

[2004] Faltings, B.; Pu, P.; Torrens, M.; and Viappiani, P. 2004. Designing example-critiquing interaction. In *ACM IUI'04*, 22–29.

[2014] Goetschalckx, R.; Fern, A.; and Tadepalli, P. 2014. Coactive learning for locally optimal problem solving. In *AAAI'14*.

[2015] Goetschalckx, R.; Fern, A.; and Tadepalli, P. 2015. Multitask coactive learning. In *IJCAI'15*, 3518–3524.

[2009] Goldsmith, J., and Junker, U. 2009. Preference handling for artificial intelligence. *AI Magazine* 29(4).

[2013] Grasch, P.; Felfernig, A.; and Reinfrank, F. 2013. Recomment: Towards critiquing-based recommendation with speech interaction. In *RecSys'13*, 157–164.

[2010] Guo, S., and Sanner, S. 2010. Real-time multiattribute bayesian preference elicitation with pairwise comparison queries. In *Proceedings of AISTAT'10*, 289–296.

[1976] Keeney, R. L., and Raiffa, H. 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*.

[2011] McGinty, L., and Reilly, J. 2011. On the evolution of critiquing recommenders. In *Recommender Systems Handbook*. 419–453.

[2007] Nethercote, N.; Stuckey, P. J.; Becket, R.; Brand, S.; Duck, G. J.; and Tack, G. 2007. Minizinc: Towards a standard cp modelling language. In *CP'07*.

[1993] Payne, J. W.; Bettman, J. R.; and Johnson, E. J. 1993. *The adaptive decision maker*.

[2000] Pu, P., and Faltings, B. 2000. Enriching buyers' experiences: the smartclient approach. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 289–296.

[1958] Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65(6):386.

[2012] Shivaswamy, P., and Joachims, T. 2012. Online structured prediction via coactive learning. In *ICML'12*.

[2015] Shivaswamy, P., and Joachims, T. 2015. Coactive learning. *JAIR* 53(1).

[1995] Slovic, P. 1995. The construction of preference. *American psychologist* 50(5):364.

[2015] Sokolov, A.; Riezler, S.; and Cohen, S. B. 2015. A coactive learning view of online structured prediction in statistical machine translation. *CoNLL'15* 1.

[2016] Teso, S.; Passerini, A.; and Viappiani, P. 2016. Constructive preference elicitation by setwise max-margin learning. In *IJCAI'16*.

[1982] Tou, F. N.; Williams, M. D.; Fikes, R.; Henderson, A.; and Malone, T. 1982. Rabbit: an intelligent database assistant. In *AAAI'82*.

[2009] Viappiani, P., and Boutilier, C. 2009. Regret-based optimal recommendation sets in conversational recommender systems. In *RecSys'09*, 101–108.

[2010] Viappiani, P., and Boutilier, C. 2010. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Proceedings of NIPS'10*, 2352–2360.

[2007] Viappiani, P.; Pu, P.; and Faltings, B. 2007. Conversational recommenders with adaptive suggestions. In *RecSys'07*, 89–96.

[2006] Zhang, J., and Pu, P. 2006. A comparative study of compound critique generation in conversational recommender systems. In *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, 234–243.