Mining Models of Human Activities from the Web

Mike Perkowitz[1], Matthai Philipose[1], Donald J. Patterson[2], Kenneth Fishkin[1]
[1]Intel Research Seattle
[2]University of Washington
[1]{mike.perkowitz, matthai.philipose, kenneth.p.fishkin}@intel.com [2]djp3@cs.washington.edu

ABSTRACT
The ability to determine what day-to-day activity (such as cooking pasta, taking a pill, or watching a video) a person is performing is of interest in many application domains. A system that can do this requires models of the activities of interest, but model construction does not scale well: humans must specify low-level details, such as segmentation and feature selection of sensor data, and high-level structure, such as spatio-temporal relations between states of the model, for each and every activity. As a result, previous practical activity recognition systems have been content to model a tiny fraction of the thousands of human activities that are potentially useful to detect. In this paper, we present an approach to sensing and modeling activities that provides scalability for a much larger class of activities than before. We show how a new class of sensors, based on Radio Frequency Identification (RFID) tags, can directly yield semantic terms that describe the state of the physical world. These sensors allow us to formulate activity models by translating labeled activities, such as "cooking pasta", into probabilistic collections of object terms, such as "pot". Given this view of activity models as text translations, we show how to mine definitions of activities in an unsupervised manner from the web. We have used our technique to mine definitions for over 20,000 activities. We experimentally validate our approach using data gathered from actual human activity as well as simulated data.

Categories and Subject Descriptors
I.2.6 [**Artificial Intelligence**]: Learning – *knowledge acquisition.*
I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *text analysis.*

General Terms
Algorithms, Performance, Design.

Keywords
Activity inference, activity models, RFID, web mining.

1. INTRODUCTION
Systems that recognize humans performing physical activities have long been recognized as enabling a variety of useful applications. Proposed applications include activity-based actuation (*e.g.* dimming lights when a video is being watched), to prompting (*e.g.* providing directions for someone using unfamiliar facilities and appliances), and notification (*e.g.* informing caregivers when an elderly person fails to perform key activities of daily living). Such applications, and the activity recognition capability that underlie them, exemplify a model of computing, variously called ubiquitous computing [21], proactive computing [18] and disappearing computing [5], that has triggered much recent interest. Unlike the traditional interactive model in which computers respond to explicit human commands, this new model requires computers to implicitly understand people's needs by observing their physical activities, and then to act autonomously to address those needs.

Unfortunately, applications of the kind described above have been slow in materializing, even as research prototypes. A key reason is that the process of developing these applications is not scalable. In particular, developing recognizers for even small classes of activities is a highly specialized activity involving months to years of work by specialists in pattern recognition. The resulting systems are typically only applicable to certain applications and deployment contexts. The cost of developing recognition infrastructure is thus both too high for the ordinary developer and not amortizable across applications, or even over multiple deployments of the same application. Many researchers have therefore recognized the value of developing a general system that recognizes a large and useful class of activities with minimal incremental effort from programmers or end users.

A broadly applicable and easy-to-use system must overcome at least two major challenges. First, the system should be *general-purpose*: it should not, by its very design, be incapable of recognizing new activities of interest. Design decisions that limit a system's generality include the sensors used, the features derived from the sensors, the relationships between these features that comprise activities modeled in the system, and the algorithms used for matching observations to models. Second, the system must *facilitate model construction*: even if the system can in principle recognize complex activities, it should be easy for non-specialists using the system to specify models of these activities. Ideally, the system should be packaged with as many of the models as possible. Allowing non-specialists to create models is challenging because underlying representations of activities are often both specialized (e.g. probabilistic graphical structures such as Dynamic Bayesian Networks and Hidden Markov Models) and complex (even simple activities may have intricate representative structures). On the other hand, packaging models of activities with the system only pushes the modeling problem to designers of the package; at its heart, the model of an activity is a human belief and a scalable activity recognition system must make it very easy to extract and document those beliefs.

Systems that use vision as their primary sensor hold the promise of being very general. They can robustly detectable features and use probabilistic models that can represent elaborate spatio-temporal and causal relations among features. Although such systems can represent a variety of activities in principle, none have reported detecting more than tens of activities in practice. The main problem lies in the fact that, although vision is a very general sensor, the features robustly detectable from it are coarse. For instance, in most cases, current models represent the

WWW 2004, May 17-22, 2004, New York, NY USA.
ACM xxx.

relationships between "blobs" in the image rather than specific objects. Another problem is that many of these systems do not facilitate model extraction, so that, especially given the elaborate models, each activity is expensive to model.

The common approach to facilitating model extraction is to support learning of the models. In some cases [2][13][14] the developers define the structure of the possible models, but the system tunes the parameters of the model based on examples from the user. In others [3][6][15] the system uses pattern recognition techniques to recognize spatio-temporal patterns that may constitute "interesting" activities in an unsupervised manner. However, the user is then expected to label these patterns. Unfortunately the burden of supervising (whether providing examples or perusing patterns to determine appropriate labels) still makes model extraction onerous in both cases. Further, the variety of activities whose models can be extracted is quite restricted: by the scenarios anticipated (and formalized) by the developers in the former case, and by the built-in pattern recognizers in the latter.

At the heart of our technique is a breakthrough in sensing technology. Advances in miniaturization and manufacturing have yielded postage-stamp sized, forty-cent radio transceivers called Radio Frequency Identification (RFID) tags that can be attached unobtrusively to objects as small as a toothbrush. The tags are wireless and battery free. When queried by radio, the tag responds with a globally unique identifier using power scavenged from the querying signal. When combined with special wearable and ambient tag readers and a database mapping identifiers to names, RFID technology can reliably name the objects with which a person is interacting.

Because of the sensors' accuracy and specificity we model activities in a novel way: we define an activity in terms of the probability and sequence of the objects that are physically involved in the activity. We generate the models by translating textual definitions that are structured like recipes into machine-readable mathematical models. In this paper we show how these models can be generated with no human involvement; in fact, they can be produced completely automatically by mining appropriate web sites. This technique is feasible without extensive natural language tools and enables to scale our system up to tens of thousands of mundane activities, covering many aspects of human life.

To show that the resulting models (and the recognition system as a whole) can be used for detecting actual activities, we present the results of analyzing activities of daily living (ADLs) data from eight subjects in a real home. ADLs are regarded as indicators of patient wellness, and professional caregivers are often required by law to record them. Our users performed subsets of fourteen classes of activities comprising 66 activities in all. To explore the usefulness of the remaining thousands of models, and to evaluate the components of our model extractor more thoroughly, we also present results based on simulated traces and cross-corpus comparisons.

In this paper, we make three main contributions:

1. We formulate the problem of modeling activities as that of generating translations from natural language texts.
2. We describe a simple set of techniques to mine these translations automatically from the web.
3. We validate these techniques via a combination of real-world experiments and simulations.

To the best of our knowledge, this paper is the first to show how to mine useful models of human physical activity from the web.

The system for mining models is part of a larger activity recognition system called the Proactive Activity Toolkit (PROACT)[16]. To place the model extractor in context, section 2 sketches the structure and usage model for PROACT. Section 3 describes in detail how we mine models. Section 4 evaluates the model extractor. Section 5 presents related work. Section 6 summarizes the paper and presents future work.

2. ACTIVITY INFERENCING SYSTEM

Below, we describe how the PROACT activity recognition system, which uses the mined models, is intended to be used. We then describe the high-level structure of PROACT, place the model miner in context.

2.1 Usage model

PROACT assumes that "interesting" objects in the environment contain RFID tags. These can be purchased off the shelf, cost roughly $0.40 each, have the form factor of postage stamps (including adhesive backing), and can withstand day-to-day use for years. PROACT deployment involves tagging tens to hundreds of objects in their environment. This can be done incrementally. As the number of tags increase more accurate and detailed recognition becomes possible. Tagging an object involves sticking an RFID tag on it, and making a database entry mapping the tag ID to a name. Current trends indicate that within a few years, many household objects may be RFID-tagged before purchase, thus eliminating the overhead of tagging [1].

Users employ RFID tag readers to track tag objects they interact with. They may wear tag-detecting bracelets or gloves, place medium-range readers in corners of rooms, or run robots, vacuum cleaners, or janitorial carts, with mounted long-range readers. As users go about their daily activities, the readers detect tags that (a) users touch, (b) are close to them, or (c) are moved by them, and thereby deduce which objects are currently involved in an activity. PROACT uses the sequence and timing of object involvement to deduce what activity is happening.

An application can query PROACT at any time for the likelihood of various activities being tracked or details of those activities (e.g. objects involved or durations), or subscribe for event notification when activities occur with a specified degree of certainty. Programmers name activities using plain English phrases (e.g. "paying bills"). The phrase used can either be chosen from a pre-mined list provided by PROACT, or can be a new one provided by the programmer. For a new phrase, the programmer can define the activity by providing a text document containing an English description of the steps involved in the activity (much like a recipe). The model extractor converts text into activity definitions.

2.2 System Overview

Figure 1 presents the main components of PROACT. It is centered on an inference engine which, given models for activities, and sequences of sensor readings, returns the likelihood of current activities. The models are produced by the model extractor, which extracts them automatically from text documents, including but not limited to websites. The sensor readings are produced while the end-user performs activities. For debugging, PROACT provides an activity viewer, which provides programmers with a real-time view of activities in progress, the sensor data seen, and

an indication of how belief in each activity changes with the data. A more detailed description of the system is in [16]. We elaborate on some of the relevant components below.

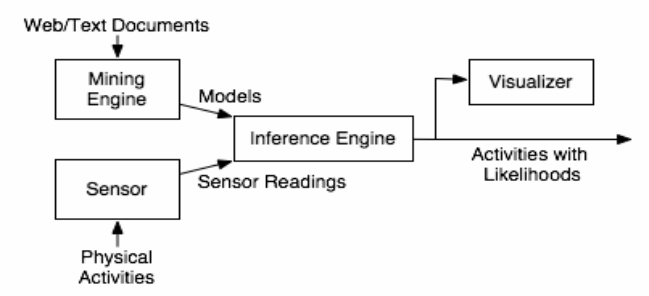

Figure 1: A High-Level View of PROACT

2.2.1 Sensors

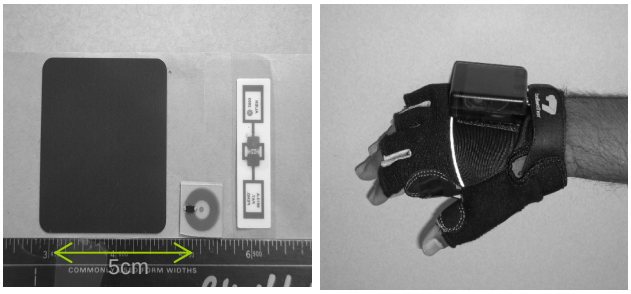

Figure 2: RFID Tags (L) and Glove-Based Reader (R)

PROACT depends on being able to observe objects that are "involved" in activities. As mentioned previously, we detect objects by detecting RFID tags stuck to the objects. The left-hand image in Figure 2 shows three types of off-the-shelf tags that may be used for this purpose. The tags are small and have adhesive backing. The first two have a 3-5 cm range, the third 2-6 meters.

Currently, we use two different kinds of RFID readers to detect two types of involvement. First, we use long-range readers mounted on a mobile robot platform to map the location of objects in the activity space. Coupled with the location of the user, this information gives the set of objects close to a person at a given time. Second, we use a short-range reader built into the palm of a glove that can determine the objects that are touched. The glove-based reader is on the right of Figure 2.

2.2.2 Models

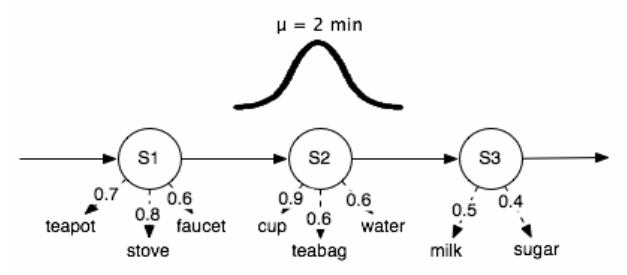

Figure 3: PROACT Model for Making Tea

We now describe our model of activities. As an example, figure 3 shows how the activity of making tea is modeled. Each model is composed of a sequence $s_1,\ldots,s_n$ of steps in the activity. If a step follows another in the sequence, then the latter must temporally follow the former in any valid instance of the activity. The example shows three steps for making tea, each drawn as a circle, corresponding to (A) boiling water, (B) steeping, and (C) flavoring the tea. Each step s_i has:

- An optional duration t_i, modeled as a Gaussian probability distribution (μ_i,σ_i). In this example, steeping the tea is expected to take 2 minutes. The amount of time required to boil the water and flavor the tea is unknown, and does not influence the reasoning.
- The set $\{(o_{i1}, p_{i1}),\ldots,(o_{iNi}, p_{iNi})\}$ of N_i objects o_{ij} involved, along with the probability p_{ij} of involvement of those objects. In Figure 3, for instance, we expect to see a teapot 70% of the time that we are boiling water, whereas sugar is involved in the flavoring phase 40% of the time.

$$P(x_t \mid z_{1:t}) \propto \underbrace{P(z_t \mid x_t)}_{\text{Sensor Model}} \int \underbrace{P(x_t \mid x_{t-1})}_{\text{State Transition}} \underbrace{P(x_{t-1} \mid z_{1:t-1})}_{\text{Prior}} dx_{t-1}$$

Equation 1: The Bayesian update equation

Our model is based on a particle filter implementation of Bayesian reasoning. Equation 1 shows the Bayesian update equation which provides the mathematical interpretation to Figure 3. In this equation the probability of being in a given state, x_t, given the sequence of observations, $z_1 \ldots z_t$, is related to three quantities: a sensor model, a state transition model and a prior distribution. The sensor model accounts for sensor error, and the expectation of seeing a given RFID when engaged in an activity. The state transition model accommodates the graphical structure that links the nodes in the activities, the timing constraints on the nodes and the probabilistic requirement of seeing certain RFID's before an activity can be considered complete. Finally, the prior distribution accounts for the state of the world before any sensors are seen as well as a recursive description of the state of the world at the previous time step.

Our model of activities is quite simple. Possible variations on the basic theme include modeling activities using trees or graphs instead of linear lists; requiring timing information for each step; modeling sources of error, such as sensor error and model error separately; and modeling more complex belief structure at each step. However, we have deliberately chosen to keep our models simple, because they are easier to mine and faster to reason with. A key result of this paper is that even these simple models are quite effective, while supporting automated extraction and real-time tracking.

2.2.3 Inference Engine

The inference engine converts the activity models produced by the mining engine into Dynamic Bayesian Networks. We use a Sequential Monte Carlo (SMC) approximation to probabilistically solve for the most likely activities. The inference engine is adapted from that used in related work on transportation behavior inference [15].

3. THE MODEL EXTRACTOR

The model extractor builds formal models of activities (of the kind specified in section 2.2.2), given definitions for activities

written in natural language by humans. Such definitions (which we will generically call "directions" below) include how-tos (*e.g.* those at ehow.com), recipes (*e.g.* from epicurious.com), training manuals, experimental protocols, and facility/device use manuals. For example, Figure 4 shows directions for making tea.

Making Tea:

1. Fill a **teapot** from the **faucet**. Place kettle on the **stove** and boil.
2. Pour hot **water** into a **cup**, filling ¾ of the cup. Immerse **teabag** in cup for two minutes and dispose of teabag.
3. Add **milk** and **sugar** to taste.

Figure 4: Directions for Making Tea

3.1 Syntactic structure of directions

A key observation in enabling this translation is that, in many cases, the structure of natural language directions closely parallels the structure of our formal model. Typically, each direction consists of:

1. A title t for the activity, and
2. A textual list $r_1,\ldots,r_m$ of steps. Each step r_i has:
   - Possibly a special keyword delimiting duration d_i, and
   - A natural-language description nld_i, typically a paragraph, of what to do during the step, typically mentioning some subset of the objects involved in the step, and often also constraints on the duration of the step.

The example of Figure 4 has title t = "Making Tea", number of steps $m = 3$, no keyword-delimited duration for steps, a two-minute minimum constraint on the duration of step 2. The objects mentioned in each step are highlighted in bold font.

In practice, each corpus of directions we wish to mine has its own concrete syntax (keywords, numbering scheme, indentation, etc), which needs to be parsed into the above abstract syntax. For each corpus, therefore, we require a user to provide a front-end that performs this parsing.

3.2 Converting directions to activity models

The similarity of structure between the formal models and the directions suggests the following scheme for converting the latter into the former. Intuitively, we produce one step in the model for each step in the directions; the objects mentioned in the paragraph for the step are those involved with the step in the model. We name key steps in bold, and provide acronyms for them when we need to use them in the future.

1. **Labeling.** Set label l of the mined model to title t of the directions.
2. **Parsing Steps.** For each step r_i in list $r_1,\ldots,r_m$ generate step s_i as follows:
   a. If r_i has keyword-delimited duration d_i, set the mean duration μ_i for the step to d_i, and standard deviation σ_i to $S(d_i, i, l)$.
   b. Let $O_i=O(nld_i)$ be the set of terms that represents objects in the descriptive paragraph nld_i. For each term o_{ij} in O_i, calculate the probability $P(o_{ij}, i, l)$ that an object named o_{ij} is touched in step i of activity l.
3. **Tagged Object Filtering.** Given the set $O_{deployed}$ of tagged named objects in the space where the activity is happening, we remove from our model all observations related to objects not in $O_{deployed}$. This step exploits the fact that in most actual deployments of the system, the set of objects the system can possibly see is often much smaller than the set of objects named in the models.

The scheme depends on three helper functions:

- Function $S(d, i, l)$ computes the standard deviation for the duration of step i of activity l (which has mean duration d). A very simple definition is $S(d, i, l) = N$ *seconds*, where is an integer fixed for each corpus being mined.
- Function $O(nld)$ computes the set of terms representing objects in paragraph *nld*. This happens as follows:
  a. **Object Extraction.** Compute the set T of terms in *nld* that represent objects, as per an ontology over the terms. We currently use the WordNet ontology, and include all terms that have either "object" or "substance" as hypernyms.
  b. **Noun-Phrase Extraction.** Compute the subset of terms in T that are used as nouns in the original paragraph, *nld*. We currently use the QTag tagger [12] on the incoming paragraph for part-of-speech tagging. We run the tagged paragraph through a customized regular-expression based noun-phrase extractor, which extracts the set of maximal phrases that contains only nouns.
- Function $P(o, i, l)$ computes the probability that object o_i is involved in step i of the activity labeled l. We consider two alternative approximations for P, neither of which makes an effort to distinguish between two different steps of a model i.e. they ignore parameter i of function P.
  a. **Fixed Probabilities.** $P(o, i, l) = P_{obj}$. where $0<P_{obj}<1$ is fixed for each corpus being mined. In our experiments below we use $P_{obj} = 0.5$.
  b. **Google Conditional Probabilities (GCP).** $P(o, i, l) = GoogleCount("l"+o)/GoogleCount(l)$. $GoogleCount(s)$ is the number of pages on the web matching string s as reported by Google [8], and $s+s'$ is the concatenation of string s and s'. For example, if the phrase *"making tea"* has *24,200* matches, and the phrase *"making tea" cup* has *7,340* matches, we conclude that the conditional probability of a cup being involved in (any step of) making tea is $7340/2400 = 0.3$.

Table 1. GCP for Some Common Activities and Objects

object→ ↓activity	Cup	Diaper	remote control	Stapler	keyboard	wrench
making tea	0.30	0.01	0.01	0.00	0.03	0.01
changing baby	0.09	0.14	0.08	0.00	0.07	0.00
watching television	0.07	0.01	0.06	0.00	0.02	0.00
copying paper	0.03	0.01	0.01	0.01	0.05	0.00
sending e-mail	0.01	0.00	0.01	0.00	0.04	0.00
auto repair	0.09	0.06	0.02	0.00	0.04	0.05

The GCP is predicated on what we call the *mirror assumption*: the probability of an object being involved in an activity in the real world is reflected by the probability of the phrases describing the two co-occurring in human discourse (and therefore on the web). It practice, since we interpret the results of our inference engines as likelihoods rather than probabilities, it is sufficient for our purposes that the probabilities are consistent; if the actual probability of involvement of one object-activity pair is lower than that of the other, the GCP of the one should be lower than that of the other.

Since the GCP may be unfamiliar to readers, we present in Table 1 the GCPs for six activities and six objects. We pick the objects such that the *i*'th object is intuitively one that would be involved in activity *i*, but not in any other. The *i*'th row and *i*'th column of the table contains the GCP of the *i*'th object being involved in the *j*'th activity.

In most cases, as expected, the table has its highest values along the diagonal. Pairings that intuitively seem unlikely (such as "diaper" and "making tea") have substantially lower value than more plausible ones (e.g. "cup" and "making tea"). Although the relative values of the probabilities are on the whole sensible, the absolute values are lower than one might expect in most cases. For instance one expects to use a keyboard more than *4%* of the time when sending e-mail! It is therefore important that any scheme that uses GCP require at most consistency, but not absolute accuracy, of probabilities.

3.3 Example

After **Object Extraction**:

T_1: {kettle, faucet, stove}

T_2: {water, cup, filling, teabag}

T_3: {milk, sugar}

After **Noun Phrase Extraction**:

O_1: {kettle, faucet, stove}

O_2: {water, cup, teabag}

O_3: {milk, sugar}

After **Google Conditional Probabilities**:

s_1: {(kettle, 0.11), (faucet, 0.01), (stove, 0.08)}

s_2: {(water, 0.50), (cup, 0.30), (teabag, 0.01)}

s_3: {(milk, 0.16), (sugar, 0.16)}

After **Tagged Object Filtering**:

$O_{deployed}$ = {kettle, stove, cup, teabag, milk}

t_1: {(kettle, 0.11) , (stove, 0.08)}

t_2: {(cup, 0.30), (teabag, 0.01)}

t_3: {(milk, 0.16)}

Figure 5: Steps in Mining the Directions for Making Tea

To show how the model extractor works, we apply the model-mining steps of the previous subsection to the directions for making tea:

1. **[Labeling]** As per step 1, the label for the new activity is "Making Tea".
2. **[Parsing Steps]** As per step 2, the new activity has *m = 3* steps.
   a. Since none of the steps in the figure have a keyword-delineated duration, we do not ascribe durations to any of the steps.
   b. Figure 5 shows the result of applying the helper functions to each of the paragraphs 1, 2 and 3 of the incoming directions.
      i. Sets T_1 through T_3 show terms describing objects, as extracted by the OT pass for each of the three paragraphs. Note that the terms include "filling", which though a hyponym of "substance" in WordNet, is as a verb here.
      ii. Sets O_1 through O_3 are the subsets of T_1 through T_3, as produced by the POST pass that are used as nouns in the directions. Since "filling" is used as a verb, it is eliminated here.
      iii. Sets s_1 through s_3 are the result of adding to each object the probability that it is involved in the activity as a whole, using GCP to get probabilities.

   Note that this resulting model has structure very similar to that of the model in Figure 3, except that the probabilities are different and that the new model has no timing constraints.
3. **[Tagged Object Filtering]** Finally, suppose that we use the model to track activities in a space where the set $O_{deployed}$ of names used for tagged objects is {kettle, stove, cup, teabag, milk}. Sets t_1 through t_3 result from removing from sets s_1 through s_3 objects that are not in this set.

4. EVALUATION

We have used the techniques described in the previous section to mine roughly 21,300 models from two websites that provide directions for activities. We mined ehow.com for roughly 2300 directions on performing domestic tasks (from "boiling water in the microwave" to "change your air filter"), and ffts.com and epicurious.com for a further 400 and 18,600 recipes respectively. In the rest of this section, we evaluate these models. None of the models we mined for this set of experiments have keyword-delimited timing constraints; we have used a default of five seconds per activity step.

We are interested in answering two main questions through our evaluation:

1. **Are the models good enough?** We wish to evaluate whether the models we mined contain sufficient information to enable correct recognition of object traces gathered from actual humans performing activities. The subjects may, of course, perform the activities in a wide variety of ways, most of which will hopefully be captured by the model.
2. **How effective are the various steps that comprise the model extractor?** We wish to evaluate the impact of the various steps (and of alternate design options) of the mining scheme.

We assume for the evaluation that the inference engine is fixed to be the Monte-Carlo based solver sketched in section 2.2.3.

A comprehensive but impractical strategy to determine sufficiency and necessity would be to collect, from real-world activities, traces that exercise all models we have mined. These traces would represent the large variety of ways in which people perform activities. By comparing to ground truth the results reported by

the inference engine on these traces, we could evaluate sufficiency of all the models. By comparing accuracy of results with various (combinations of) mining techniques turned off, we could evaluate the efficacy of those techniques.

Unfortunately, it is impractical to collect actual usage data for thousands of activities. The activities are time consuming, often complex and sometimes unpleasant; getting subjects to perform all of them in various combinations is clearly an impractical task. Instead, we use the following three strategies to approximate comprehensive evaluation. The three options apply to increasingly larger classes of activities, but provide increasingly indirect evidence about the quality of our system:

1. **Human activity-trace recognition (HTR).** We instrumented the home of one of the researchers with 108 RFID tags. Over a period of six weeks, we collected traces (consisting of time-stamped RFID tag numbers) from 14 subjects, while each performed a randomly chosen 12 of 14 Activities of Daily Living (ADLs). Our subjects all wore the glove-based RFID reader (Figure 2) to allow tracking of touched objects. We picked our ADLs from state-mandated ADL lists. Each ADL (*e.g.* housework) corresponds to many ordinary activities (*e.g.* washing dishes, making beds, dusting, vacuuming). In all, our subjects picked from roughly 50 activities of the latter kind. Subjects recorded activities as they performed them, to provide us with ground truth.

   To measure the quality of our models, we select 66 of the models mined from ehow.com. These correspond to the fifty activities being performed by the user (some user activities are described by more than one model). We measure the accuracy with which our inference engine infers activities given these models. We define accuracy below.

2. **Inter-corpus consistency (ICC).** In the absence of actual activity traces, we cannot measure the ability of our models to represent real activities directly. However, we can still indirectly measure the ability of our models to represent the diverse ways in which activities may be performed. Specifically, if an activity model mined from a particular corpus is to represent most instances of the corresponding activity, it must, in particular be compatible with another representation of the same activity (since the latter description presumably represents some person's way of performing the activity).

   Based on the above insight, we use two different corpora (epicurious.com and ffts.com) containing the same activities to test the quality of our models. Each corpus contains roughly 120 recipes for making cookies. We first generate models for all activities in the two corpora. Next, for each model derived from one corpus, we generate traces compatible with model: for each state in the model, in order, we include in the trace a subset of the objects corresponding to that state. Each object is picked with probability equal to its conditional probability p_{ij}. We measure the accuracy with which we infer activities on these traces given the models generated from the second corpus.

3. **Intra-corpus distinguishability (ICD).** Unfortunately, even inter-corpus comparisons are difficult to perform extensively due to lack of data. Most activities in ehow.com, for instance, are not easily available from other sources. We therefore adopt a third strategy. Once again, we resort to an indirect technique for measuring if the mined models are sufficient for recognizing traces from real activities. We observe that if the models are good enough to recognize their corresponding activities, then in particular each model should have a stronger match than any of the other models on activities that conform perfectly to it. In other words, the models must contain enough information to distinguish themselves from each other.

   To keep the number of activities inferred simultaneously manageable, we restrict ourselves to testing distinguishability between models from the same activity domain. To this end, we select seven domains (ADLs, automobiles, food, grooming, parties and the two cookie datasets described above). As a measure of distinguishability, we measure the accuracy with which traces generated from models in a given domain are detected, given the set of all models in the domain. Since the traces generated are "perfect" with respect to the model being matched against, the distinguishability metric is in a sense a "limit" on how well the models can do given our inference engine.

The three strategies above rely on measuring the accuracy with which activity traces can be detected. To determine accuracy over a trace, we split the trace into five-second windows. Accuracy then refers to the fraction of windows where our inference engine gives the same result as ground truth.

In the following subsections, we present the limit measurements from our ICD study, followed by the more direct quality measures from the HTR and ICC measurements. Next, we analyze the contribution of the component techniques of the model extractor towards accuracy. Finally, we measure the effect of some of these techniques in producing more compact models.

4.1 Distinguishability

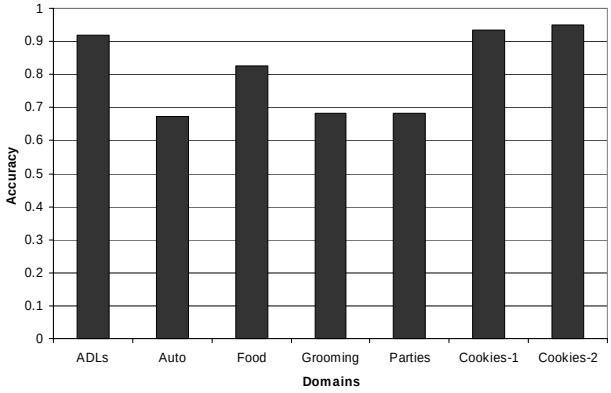

Figure 6: Distinguishability Within Activity Domains

Figure 6 shows the distinguishability for each of the seven domains mentioned above. The distinguishability of a domain is the accuracy across the result of concatenating all traces for that domain i.e. the fraction of all 5-second windows across all activities in the domain that we labeled correctly. The numbers range from 67% for autos to 95% for the second cookie domain. The implication is that if the sensor trace from an activity conforms closely to that expected by the model for that activity, the models we mined for the domains are distinct enough, and our activity inference engine sensitive enough, that we can correctly identify the activity much of the time.

4.2 Human and inter-corpus trace recognition

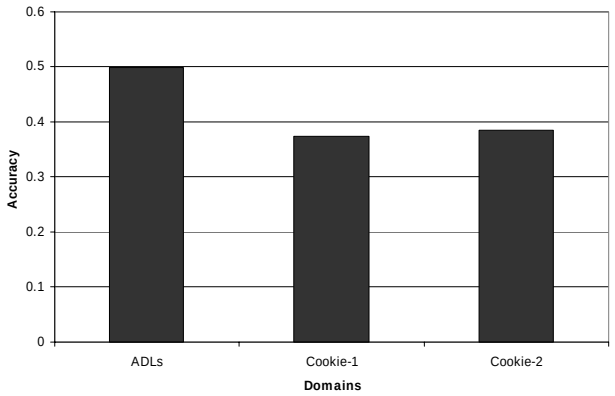

Figure 7: Accuracy of Recognition for Three Datasets

Figure 7 presents the accuracy achieved on the HTR measurements (testing against measured human traces, labeled ADLs), and the ICC measurements (testing inter-corpus consistency, labeled Cookie-1 and Cookie-2).

The bar labeled "ADLs" represents the accuracy across the unified trace that results from concatenating the traces from all 14 subjects across all 12 activities that they performed. It therefore says that for 50% of all 5-second timeslots, we are able to correctly infer the ongoing activity. Although the number may seem modest, we point out for comparison that a random assignment of activities to slots, given that there are 66 possible activities being modeled, will yield only 1.5% accuracy. We also note that we are unaware of any previous work that recognizes such a large variety of activities being performed by such a large number of subjects in an unmodified home (even with handcrafted activity models) with at least this level of accuracy.

The fact that recognition rates for ADLs on real traces is significantly lower than the distinguishability implies that the traces seen in practice deviate from models. There are two main reasons for this deviation. First, the traces violate some of the assumptions underlying our models. For instance, many objects that were touched and in the models were not tagged, tagged objects were sometimes missed, and objects from different activities were interleaved. Second, the models were not fully general representations of the corresponding activities. For instance, they imposed extraneous ordering constraints, omitted possible objects or included optional objects.

The bar labeled Cookie-1 represents the accuracy of Cookie-1 models across the unified trace that results from concatenating the individual traces generated from the models of Cookie-2, and vice versa for the bar labeled Cookie-2. As explained previously, these bars address the question of how generally valid the models mined from each of these corpora are. In comparison with the 95% distinguishability rates, the accuracy is roughly 40% in both cases. There are two reasons for this discrepancy. One, of course, is that the identical recipe can have quite different structure in the two corpora. The other is that for some of the recipes, there is no (unique) counterpart in the other corpus. Our results may have improved had we removed these.

4.3 Impact of techniques on accuracy

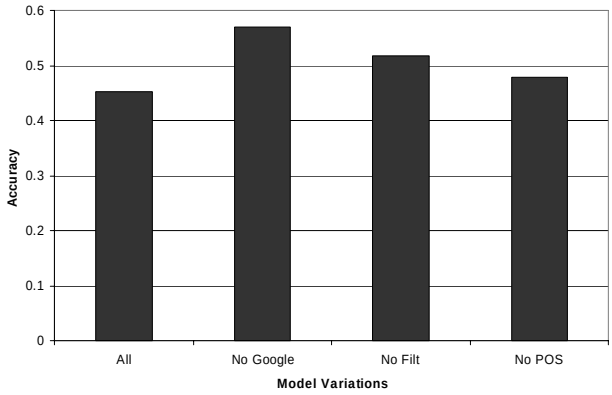

Figure 8(a): Impact of Selectively Disabling Mining Techniques on ADL Dataset

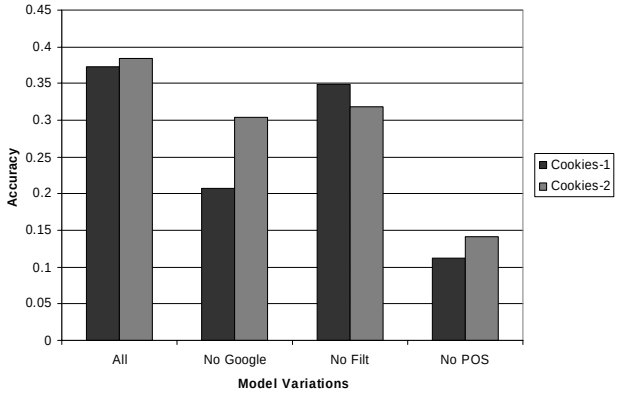

Figure 8(b): Impact of Selectively Disabling Mining Techniques on Cookies Dataset

Figure 8(a) and (b) present the efficacy of the various techniques that comprise our model extractor on the ADLs and the two cookie datasets respectively. In each chart, the bar labeled "All" uses all available techniques. The other bars represent our models generated without Google probabilities (using a fixed probability of 0.5 for all objects), without performing tagged object filtering, and without performing part-of-speech tagging and noun phrase extraction, respectively. The markedly different shapes of the two charts make clear that the domains of ADLs and Cookies are quite different. The ADL domain is fairly sparse, with many activities involving only a few objects. In most cases, the presence of a particular object serves to distinguish one activity from another, as in the "vacuuming" activity, which is inextricably linked to the vacuum cleaner. In this domain, the nuances of Google probabilities and complex activity structure only get in the way. In the Cookie domain, on the other hand, each activity model involves many more objects. Furthermore, many activities use the same objects; sugar, flour, butter, and eggs, for example, appear in most recipes. Here, the nuances of how likely each object is and at which step it is introduced become vital for distinguishing the activities. To look at the bigger picture, choosing the best technique for a particular problem domain may depend highly on the structure of that domain.

4.4 Impact of techniques on compactness

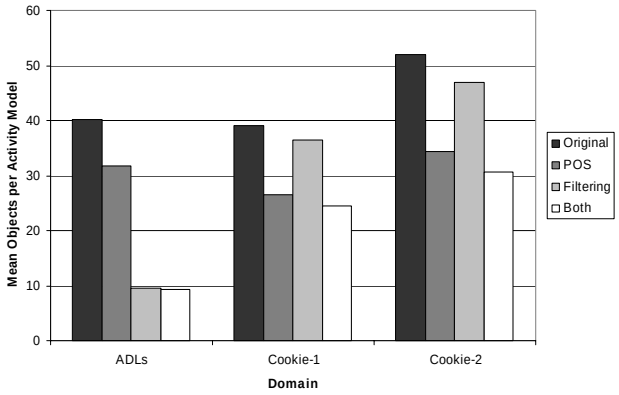

Figure 9: Impact of Mining Techniques on Model Size

The various techniques we apply in the extractor have significant impact on the resulting models. Figure 9 shows how the average number of objects per activity varies as we apply part-of-speech tagging and tagged object filtering to models in the ADL and Cookie domains. As discussed in Section 4.3, these two domains differ significantly in their structure. ADL activity models are only somewhat reduced when we filter for noun phrases, but drastically reduced by tagged object filtering. The resulting models are significantly smaller. In fact, approximately 30% of activity steps become empty; when we remove these empty nodes, we change the structure of the resulting activities. On the other hand, the Cookie activities are barely affected by tagged object filtering, but are significantly reduced by part-of-speech tagging. These models contain many more objects, and these techniques do not create any significant change in structure.

5. RELATED WORK

To the best of our knowledge our work is the first to mine models of human physical activities directly from textual or web-based descriptions. Related work falls into two main categories, and we view our work as being complementary to both.

In the past few years, much work has gone into extracting models of human beliefs (other than activities) such as shared interests [18], preferences [4], reputations [7], concept definitions [11] and spam [17] by mining virtual communities such as the web, chat rooms and newsgroups. In many cases, these efforts have revolved around a mixture of techniques similar to those we use: lightweight natural language processing (NLP), document co-occurrence analysis for detecting semantic proximity, mining topic and community-specific corpora. Even the notion of Google Conditional Probability (GCP) has in essence been used for extracting conditional probability of beliefs [9], although our application of Google to real-world phenomena (via the mirror assumption) is novel. These projects also employ a variety of techniques (such as link-structure analysis and more advanced NLP techniques) that could profitably be used in our system. In some cases, the mined models are even used by virtual sensors to recognize new data; for instance, spam classification software uses mined models to classify incoming data. None of this work, however, makes the connection between mining models from data and recognizing activities in the physical world.

An extensive literature exists on activity recognition, most of it using computer vision. In most cases [2][13][14], the focus is on identifying features, model structure and mining algorithms that enable robust detection of small subsets of activities. The models used in these systems are constructed and labeled by hand. In some of these cases, the parameters of the model are learned on the fly, whereas the structure and labels are provided by clients of the system.

A smaller set [3][10] of systems learn quantitative models of classes of activities; they observe video footage depicting the movement of pre-defined robust features and extract probability density functions over possible sequences of feature configurations as "interesting" models. The models so extracted are sufficient for their intended purpose (*e.g.* enabling computers to mimic facial movements and gestures, and detecting periodic phenomena in footage of people's lives). However, in order to use them as part of a library of activities, the extracted models still have to be labeled. Viewing the quantitative models and labeling them is a challenging task for humans.

A final small class of systems [6] extracts models automatically from video, but provides qualitative descriptions of the footage. For instance, Fernyhough et al are able to detect one car overtaking another in video footage and report the phenomena as a sequence of three operators: traveling *behind-right* in the *same* direction, traveling *right* in the *same* direction, and traveling *infront-right* in the *same* direction, rather than as a probabilistic graphical model over low-level features. The resulting description models are, of course, much easier for humans to parse and then label. However, it is conceivable that more models with more steps will get more difficult to comprehend or specify even in this qualitative fashion. Further, the system designer has to build in the primitives (such as *infront-right*) and the grammar to combine them.

We view our techniques as being complementary to the ones discussed above. In particular, our models are suited to modeling activities that are characterized by the identity of objects involved. In many cases, however, activities are better characterized by geometry, color and texture information; the above techniques are more suited for such activities.

6. SUMMARY AND FUTURE WORK

We show how the use of novel sensing technology (RFID tags) allows us to view human activity models as probabilistic translations from natural language phrases to words describing objects. Given this view of models, we provide a suite of techniques that allow conventional natural language descriptions of activities (such as how-tos and recipes) can be completely automatically converted into formal models of activities. We show, through a combination of measured data of human activities and simulations, that the models we generate are useful. We also provide an analysis of the contribution of our mining techniques to improving the accuracy and compactness of our models. Our techniques have allowed us to construct many orders of magnitude more models for human activities than described previously.

This paper is essentially an introduction to the idea of mining activity detection from the web. We have identified an extensive research agenda that follows up on these ideas.

Our first order of business for the future is to perform a more comprehensive evaluation of our models. We intend to collect human traces on many more activities, with many more tagged objects in the activity space (a few thousand as opposed to the

current hundred or so), and on more challenging, but common, patterns of activity such as multi-person and interleaved activities. We would like to find corpora that provide alternate descriptions for many of our activities, both for performing inter-corpus consistency comparisons, and more importantly to examine techniques for combining multiple descriptions for a given activity. Finally, we have not yet attempted to mine timing constraints on activity steps on any large scale; we hope to locate corpora where timing constraints can be extracted easily and to test the usefulness of the timed models that result.

We also plan to explore techniques for improving the effectiveness of our mined models. We are examining how to mine observables relevant to sensors other than RFID tags. In particular, we intend to include location in our models, extracting location information from textual descriptions. For example, the activity representing "baking cookies" might be annotated with the location "kitchen". The location observables in the model are then triggered by location sensors as opposed to RFID sensors.

Close examination of how we fail in recognizing certain activities has led us to believe that it will be profitable to model activities as partial orders, as opposed to lists. We are considering simple natural-language processing techniques to determine dependencies between two steps of a list of directions. In addition, human activities are often described in terms of each other, as when the recipe "basic pie crust" is given as a component of the recipe "apple pie". We intend to augment our model of activities to allow references to other activities, and to then mine these references based on syntactic cues. We expect that, in many cases, clients of our system may be interested in activities for which no step-by-step description is known (e.g. "paying bills"). In these cases, we suspect that we can still create simple naïve-Bayesian models by simply identifying object names that are "semantically close" to the phrase describing activities. Our early forays in this direction have been quite promising.

A difficulty in connecting mined activities with tagged objects is that the activity models may refer to objects synonymously. For example, we might have an object in our kitchen tagged "skillet" while our pancake recipe calls for a "frying pan". One possible solution to the problem is to use synsets to represent objects. Synsets – collections of synonymous words – can be extracted from WordNet. Another solution would be to classify objects with an object ontology – again, an ontology can be constructed from WordNet – and recognize objects that are close in the ontology can sometimes be used interchangeably in activities.

In this paper, we assumed that the spaces in which activities are performed have been extensively tagged with RFID tags. In practice, however, the person tagging the space may have only a limited number of tags available, and may value feedback on which objects are most profitable to tag, given the set of activities to be detected. We call this the "eigenobject" problem, and have preliminary ideas on solving it.

7. ACKNOWLEDGMENTS

We thank Suzi Soroczak and Meliha Yetisgen-Yildiz for performing the noun phrase extraction. We thank Henry Kautz and Tanzeem Choudhury for feedback and discussions.

8. REFERENCES

[1] Auto-ID center. www.autoidcenter.org/ abouttheteсh_idiotsguide6.asp

[2] Aaron F. Bobick, Yuri A. Ivanov: "Action Recognition Using Probabilistic Parsing. CVPR 1998: 196-202 .

[3] Brian Clarkson, "Life Patterns: Structure from Wearable Sensors". PhD Thesis, MIT, 2002.

[4] Kushal Dave, Steve Lawrence, David M. Pennock: Mining the peanut gallery: opinion extraction and semantic classification of product reviews. WWW 2003: 519-528

[5] The Disappearing Computer Initiative. www. disappearing-computer.net/

[6] J. Fernyhough, A. G. Cohn, and D. Hogg: Constructing qualitative event models automatically from video input", Image and Vision Computing, 18, 2000, pages 81-103.

[7] D. Gibson. "Communities and Reputation on the Web". PhD Thesis, UC Berkeley, 2002.

[8] Google. http://www.google.com

[9] F. Heylighen. "Mining associative meanings from the web: from word disambiguation to the global brain". Proceedings of Trends in Special Language and Language Technology, R. Temmerman (ed.), Standaard Publishers, Brussels, 2001.

[10] Tony Jebara, Alex Pentland: Action Reaction Learning: Automatic Visual Analysis and Synthesis of Interactive Behaviour. ICVS 1999: 273-292

[11] Bing Liu, Chee Wee Chin, Hwee Tou Ng: Mining topic-specific concepts and definitions on the web. WWW 2003: 251-260.

[12] Oliver Mason. QTag home page. http://web.bham.ac.uk/ o.mason/software/tagger/index.html

[13] Darnell J. Moore, Irfan A. Essa, and Monson H. Hayes III. "Exploiting human actions and object context for recognition tasks". In IEEE International Conference on Computer Vision, volume 1, pages 80-86, Corfu, Greece, September 1999.

[14] N. Oliver, E. Horvitz, and A. Garg. "Layered representations for human activity recognition". In Fourth IEEE Int. Conf. on Multimodal Interfaces, pages 3--8, 2002.

[15] D. Patterson, L. Liao, D. Fox, H. Kautz. "Inferring High-Level Behavior from Low-Level Sensors". Ubicomp 2003.

[16] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Haehnel, "The Probabilistic Activity Toolkit: Towards Enabling Activity Aware Computer Interfaces", submitted to CHI'03.

[17] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In AAAI-98 Workshop on Learning for Text Categorization, 1998.

[18] M. F. Schwartz and D. C. M. Wood. "Discovering shared interests using graph analysis". Communications of the ACM, 36(8):78--89, 1993.

[19] David Tennenhouse, "Embedding the Internet: proactive computing," Communications of the ACM, vol. 43, no. 5, pp. 43-50, May 2000.

[20] WordNet. www.cogsci.princeton.edu/~wn

[21] Mark Weiser. "Ubiquitous computing". IEEE Computer, 26(10):71-72, 1993.