Toolkit to Support Intelligibility in Context-Aware Applications

Presented by Mary Salinas
What problem are they trying to solve?

- Context-aware applications are great for helping users to automatically serve users better but the complexity of models can make it difficult for users to understand them.
- Users can become frustrated and lose trust in the applications. Applications must be intelligible and provide explanations of context-awareness.
What did they do?

- Created an architecture for generating a wide range of explanations including Why, Why Not, How To, What, What If, Inputs, Outputs and Certainty.
- Create a library of reference implementations of explanation algorithms for 4 different models.
- Provide automated support for the common explanations to promote good design practices.
Surveyed existing use of model types

![Bar charts comparing model types for all apps and recognition apps.](image)

**Figure 1:** (Left) Counts of model types used in 109 of 114 reviewed context-aware applications. (Right) Counts for 50 recognition applications; classifiers are used most often for applications that do recognition. Key: decision tree (DT), naïve Bayes (NB), hidden Markov models (HMM), support vector machines (SVM), k-Nearest Neighbor (kNN).
Rule-based Decision Model

- Rules are usually if/else logic or simple mapping
- Most popular decision model.
- Used in personalization, activity recognition, monitoring, location guides
Decision Tree Model

- Similar to rules, but decisions are made from top down rather than bottom up.
- Decision trees are built from statistical data so they can model certainty from probability of leaves.

Figure 7: IM Autostatus. Demonstration of various explanations from an IM responsiveness prediction plugin that uses a decision tree to predict when a buddy would respond.
Bayes Decision Model

- Naïve Bayes is the sum of evidence and assumes features are independent of each other.
- Includes prior probabilities of selected class value and from each feature value.
Hidden Markov Model

• Apply weights of evidence as similar to Bayes and include temporal factors as well.
Implementation

- Implemented in Java based on Enactor and Context Toolkit created by Anind.
- Explainer to generate explanations for model-independent types and one Explainer for each of the 4 decision model types.
- Reducer to remove explanations that include too many reasons or each reason is too long.
- Presenter renders the explanation in form suitable for users. Developers can build several Presenters if needed.
Architecture for Rules and Classifiers

Figure 2: Architecture for handling rules and classifiers. The Intelligibility Toolkit adds four components to the Enactor framework of the Context Toolkit. Users ask for explanations with Querier, and invoke Explainer to generate explanations. The explanations may be simplified with a Reducer, and rendered through a Presenter.
Claimed Results

• Allows for fast prototyping of context-aware applications
• Provides lower barrier to providing explanations
• Provides flexibility of using explanations
• Facilitate appropriate explanations automatically.
What has happened since then?

• Active development is continuing by Lim at http://www.contexttoolkit.org/

• Lim’s home page: http://www.brianlim.net/
Any Questions?