A Framework for Multimodal Data Collection, Visualization, Annotation and Learning

Anne Loomis Thompson
Microsoft Research
One Microsoft Way
Redmond, WA, US
annelo@microsoft.com

Dan Bohus
Microsoft Research
One Microsoft Way
Redmond, WA, US
dbohus@microsoft.com

ABSTRACT
The development and iterative refinement of inference models for multimodal systems can be challenging and time intensive. We present a framework for multimodal data collection, visualization, annotation, and learning that enables system developers to build models using various machine learning techniques, and quickly iterate through cycles of development, deployment and refinement.

Categories and Subject Descriptors
D.2.m **[Software Engineering]:** Miscellaneous – *Rapid Prototyping* I.2.6 **[Artificial Intelligence]:** Learning – *Parameter Learning*

General Terms
Algorithms, Design, Measurement.

Keywords
Multimodal systems, tools, visualization, annotation, learning.

1. INTRODUCTION
Systems that engage in open-world interaction, such as embodied conversational agents or robots, typically rely on models that fuse multiple streams of evidence to enable them to "understand" the surrounding environment, and the activities, beliefs, and intentions of people around them. These models are often constructed in a data-driven manner, by using machine learning techniques over corpora of collected and annotated data. In general, this is a time consuming process.

A number of tools for multimodal data visualization, annotation, and analysis have been developed and are used in the research community [1, 2, 3, 4, 5]. We present a framework that provides these functions, and additionally supports model building using a variety of machine learning techniques. By bringing these functions together in a single platform, the framework enables its users to explore data in place, gain insights for model development, and quickly iterate through cycles of deployment and refinement.

2. FRAMEWORK
At the center of the data analysis and learning framework lies an infrastructure for capturing the values of floating point temporal variables into *feature streams*. For example, the x-location of a face tracked in a video can be captured into a feature stream, *XCenter*, visualized in Figure 1; a binary feature stream can capture whether speech is in progress or not; and so forth. In general, feature streams can store information gathered from any sensory modality, or produced by the system's components at runtime.

System developers work with feature streams as strongly typed variables in code. Like standard variables, feature streams can be created, destroyed, assigned to and read from; in addition, the infrastructure manages the feature stream timeline, and values are automatically logged to disk in a compact format at every time point. Feature streams can be organized in hierarchical collections, and derived feature streams can be computed by applying operators to existing streams. For instance, the horizontal speed of a face *XCenter.Slope* can be computed by applying a *Slope* operator to the *XCenter* feature stream.

A log exploration tool, shown in Figure 1, displays the data logged by a system along a timeline. It visualizes feature streams and events logged by system components, such as detected utterances and corresponding recognition results. The tool supports variable speed audio and video replay, and overlays the results of scene analysis computations performed by the system. It supports the construction of manual annotations, and allows for user-defined tagging schemes and segmentation.

To enable rapid development of machine learned models, the framework provides a graphical user interface that allows system engineers to manage the multiple stages of the model building process. We outline them briefly below.

Define a learning problem. Model construction begins by defining the structure of the learning problem, *i.e.* how training instances and corresponding labels are generated from logged data. The framework supports binary, regression, and multinomial classification problems. Training instances can be generated for every time point within custom-defined intervals, *e.g.* where a feature stream exists, or where a condition over a feature stream holds. This mechanism enables the specification of a diverse set of multimodal inference problems, such as: predict at every point a face is visible whether the face is tracked correctly; or predict at every point between utterances the next time someone will speak.

Features used to train a model can be selected from the set of feature streams logged by the system. Novel features can also be constructed by applying operators to existing streams. For example, derived features such as "the average value of the *FaceConfidence* feature stream in the last 2 seconds" can be added easily. A number of basic statistical and signal processing operators over time intervals are supported, and developers can

ICMI'13, December 9–13, 2013, Sydney, Australia.
ACM 978-1-4503-2129-7/13/12.
http://dx.doi.org/10.1145/2522848.2531751

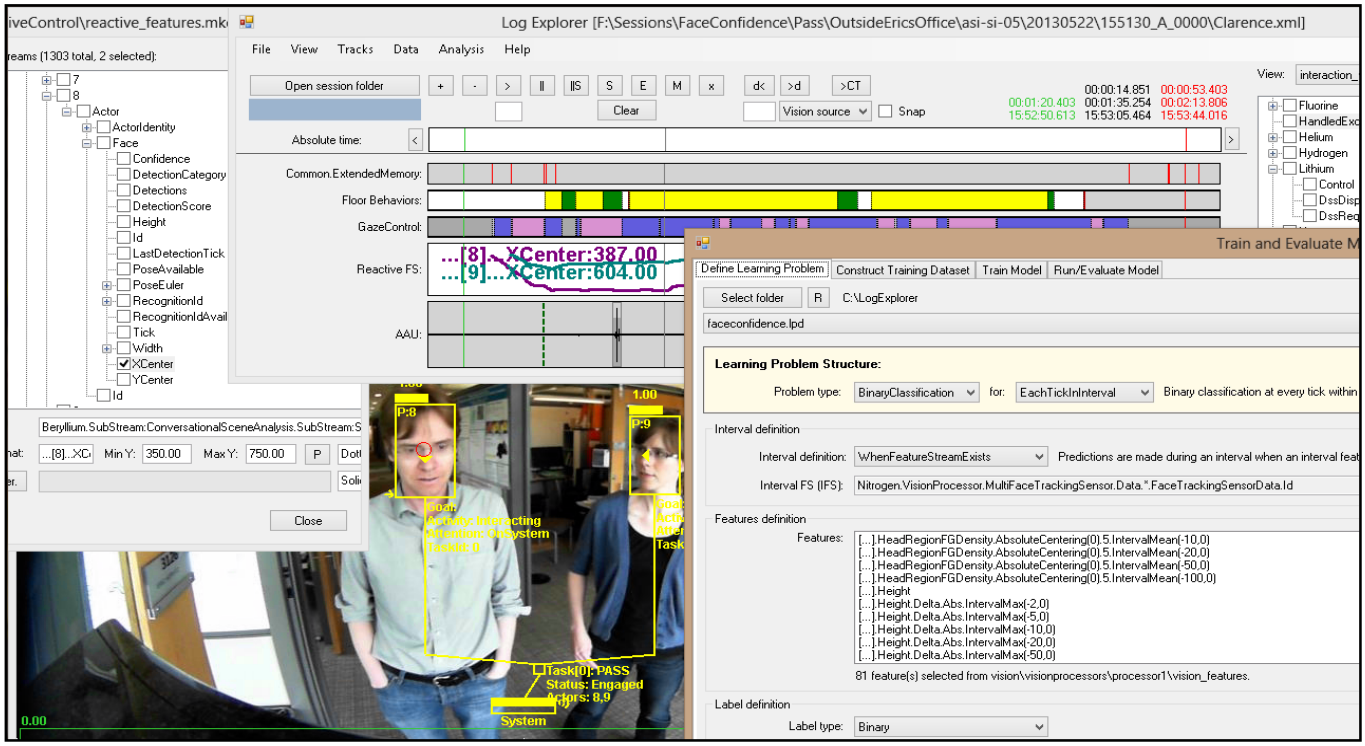

Figure 1. The log explorer tool enables visualization, annotation and learning.

extend this set. Derived feature streams can be visualized in-place, supporting insight and allowing for quick iteration over their design.

Labels can be generated based on manual tags, or automatically, by various parametric mechanisms. For instance, to predict whether an actor will start speaking within the next second, a label can be generated automatically based on a condition on an existing feature stream. Multiple label generation mechanisms are available and we continue to extend the set.

Construct a training dataset. Given a learning problem definition and control file specifying input sessions, a training data set is constructed from recorded data and output in a simple text format that can easily be imported to other tools such as Matlab or Excel. Any derived feature streams or automatically-generated labels specified in the problem definition are computed. In addition to the training data files, this process produces a report containing basic global and per-feature statistics.

Train a model. The framework supports a variety of machine learning approaches, such as boosted decision trees, linear SVMs, and logistic regression. The training process produces a model file and a report containing performance metrics such as accuracy, log-likelihood, and mean squared error.

Run/evaluate a model. The resulting models are easily integrated into our systems for use at runtime; derived feature streams are computed automatically by the framework and no additional code needs to be written. A model may also be run and evaluated offline on any collection of sessions or extracted datasets. The model predictions can be output to simple text files, or to feature streams, which can be immediately visualized in the log explorer.

Together, the feature streams infrastructure, log exploration and learning tools enable quick iterations through model building and support a variety of problem types and machine learning approaches. We continue to work on extending this repertoire and the overall capabilities of the framework.

3. DEMONSTRATION PLAN

Our demonstration will showcase this multimodal analysis and learning framework to conference participants – see accompanying video at [6]. We will illustrate the different steps of the model-building process with several learning problems we have explored using data collected from deployed systems. We will discuss and highlight the important aspects of the overall framework, including the feature streams infrastructure, the construction and in-place visualization of derived feature streams, automatic label generation, different learning problem structures, and quick iteration over model building with different machine learning techniques. We will seek feedback from demonstration participants to shape the future development of this platform.

4. REFERENCES

[1] Kipp, M. 2001. Anvil – A Generic Annotation Tool for Multimodal Dialogue. In *Proc. of 7th European Conference on Speech Communication and Technology*, pp. 1367-1370.

[2] Clow, J., and Oviatt, S. 1998. STAMP: A Suite of Tools for Analyzing Multimodal System Processing. In *Proc. of ICSLP'1998*, pp. 277-280.

[3] Wittenburg, P., Brugman, G., Russel, A., Klassman, A., and Sloetjes, H. 2006. ELAN: a Professional Framework for Multimodality Research. In *Proc. of LREC'200*6, Paris.

[4] Schmidt, T. 2004. Transcribing and annotating spoken language with Exmaralda. In *Proc. of LREC Workshop on XML based richly annotated corpora*, Paris.

[5] Rose, T., Quek, F., and Shi. Y. 2004. MacVissta: A system for multimodal analysis. In *Proc. of ICMI'2004*, New York.

[6] Thompson, A. L., and Bohus, D., 2013. Demonstration Video. http://research.microsoft.com/~dbohus/videos/icmi13.wmv