Xplorer: A System for Visual Analysis of Sensor-based Motor Activity Predictions

Marco Cavallo, Çağatay Demiralp

Abstract—Due to the large diffusion of wearable devices, the task of detecting motor activities from sensor data is becoming increasingly common in a wide range of applications. During the development of predictive models for activity recognition, data scientists generally rely on performance metrics (such as accuracy score) for evaluating and comparing the performance of classification algorithms. While these numerical estimates represent a straightforward way to summarize the effectiveness of a model, they convey little insights on the causes of misclassified events, not offering enough clues for data scientists to improve their algorithms.

In this paper we present BlueSky Xplorer, an interactive visualization system to analyze, debug and compare the output of multiple predictive models at different levels of granularity. We combine classification results on multi-sensor data with the context of usage of each sensor and with ground truth information (such as textual labels and videos), representing them as temporally-aligned linear tracks. We then define an algebraic language over these tracks that enables users to quickly identify classification errors and to visually reason on the performance of classifiers.

We demonstrate the usefulness of our tool by applying it to a real-world example, involving the development of models for assessing the symptoms of Parkinson's disease. In particular, we show how Xplorer was used to improve the performance of classification models and to discover problems in data temporal alignment.

Index Terms—Visualization, sequence, sensor data, classification, model validation, track algebra, query, Parkinson.

1 INTRODUCTION

The large diffusion of consumer-level wearable devices has opened many possibilities related to activity monitoring. Smart watches and devices such as Fitbit [1] are increasingly used by people to track their daily motor activity, whereas a wide variety of biosensors is starting to play an important role in patient monitoring. The task of detecting motor activities such as walking from sensor data is thus becoming very popular in the fields of data science and machine learning. The development of these prediction models generally relies on validating their performance on a database of labeled sensor data. Numerical metrics such as accuracy
We build on earlier work on temporal and sequential data visualization. Xplorer proved to be fundamental for visually validating and comparing which of them can be deployed. In less trivial applications, data scientists and between science and business people in general.

The interface of Xplorer (Fig. 1) is composed of a main view, where classification results and labels are represented as linear tracks stacked horizontally bars of variable length. These features are also commonly found in multimedia editors, where tracks typically represent audio or video sources. In the context of data analysis, Time-ART [8] and ChronoViz [4] provide synchronized, interactive visual representations of multiple data sources. In the context of sensor-based predictive models, the computation of metrics also assumes the presence of correct ground-truth labels, assumed to be not always available — especially for manually-generated motion activity labels.

We introduce a novel visualization system, Xplorer, to interactively analyze the classification results of sensor-based predictive models. In particular, Xplorer enables users to debug and compare multiple classifiers up to the level of granularity of a single prediction, providing different ways to validate the performance of each model. Our contribution consists in (1) a new method for visually coordinating classification results with ground-truth information (i.e. labels, video) and in (2) an algebraic language for quickly identifying relevant prediction events. In particular, Xplorer aims at facilitating the interpretation of classification results, enabling data scientists to reason on the causes of misclassifications and to improve their predictive models. We note that our system does not aim at gaining insights on the internal behavior of a predictive model, rather it serves the purpose of analyzing its output.

To illustrate the usefulness of our system, we report a real-life use case, involving the development of predictive models for identifying relevant motor activities in Parkinson’s disease. We study the usage of Xplorer by a mixed group of fourteen people, composed of both data scientists and business people working on the same project. We demonstrate how Xplorer proved to be fundamental for visually validating and comparing predictive models, for reasoning on the causes of mispredictions, and for understanding the trade-offs in the usage of different sensors. We further observe how the system, being independent from the implementation details of each predictive model, facilitated the discussion among data scientists and between science and business people in general.

2 RELATED WORK

We build on earlier work on temporal and sequential data visualization (e.g., [3–5, 8]) along with interactive systems that query and compose visualizations through algebraic operations (e.g., [3, 6, 7]). The visual design of Xplorer takes inspiration from genome browsers. USCS Human Genome Browser [5] visualizes requested portions of the human genome as aligned, linear tracks, which can be added, removed and reordered based on need. Fundamental interactions include zooming and panning to enable fine-grained exploration of the data, often represented as horizontal bars of variable length. These features are also commonly found in multimedia editors, where tracks typically represent audio or video sources. In the context of data analysis, Time-ART [8] and ChronoViz [4] provide synchronized, interactive visual representations of multiple data streams, enabling navigation and annotation of time-coded data.

In order to formulate or validate complex hypotheses about the data, earlier work formulates algebraic operations over visual elements. For example, invis [3] provides a simple algebraic approach to inspecting RNA sequences, where mutations can be visually aggregated using the logical operators AND, OR, and NOT. Polaris [6], introduces a table algebra, extending Wilkinson’s grammar of graphics [7]. Xplorer builds on earlier work and introduces a basic track algebra, facilitating the ability to effectively filter, compose, and compare track representations of classification results.

3 XPLORER

The interface of Xplorer (Fig. 1) is composed of a main view, where classification results and labels are represented as linear tracks stacked vertically. A track visually corresponds to a set of non-overlapping colored blocks, positioned over a common timeline. In the case of classifier tracks (Fig. 1a), each block corresponds to a single prediction or to a set of consecutive identical predictions, which may result in blocks of variable length. If the output of a classifier is binary, the block is made visible only when the activity is detected. If a predictive model outputs a probability score, the block is instead generated by applying a classifier-specific threshold to the continuous prediction (Fig. 2). Here the color opacity of the block is determined by its associated probability score (e.g. the higher the probability of a detected activity, the more intense the color).

In the case of label tracks (Fig. 1b), each block corresponds to a textual label (e.g. “Walking”, “Person is sitting”), characterized by a start and end time which determine its position and length. Labels can be either algorithmically generated or manually defined by a human, and are often used as ground-truth or as a reference for validating classifier tracks. The color of blocks in a label track, different than in classifier tracks, has always the same opacity. A particular type of label track, called protocol track (Fig. 1c), can hold different unique labels on the same timeline, given they do not overlap with each other. The protocol track is generally used as a time reference track, where each block corresponds to a specific task and is identified by a different categorical color.

Whereas all other temporal data is represented as a linear track, video information is shown in a separate container (Fig. 1d), which can be dragged across the interface and freely resized by the user. The interface of Xplorer further includes some auxiliary modal windows and a left sidebar, from which users can decide which tracks to visualize and easily zoom to specific events contained in the protocol track.

3.1 Interactions

Track information can be analyzed at different levels of granularity through zooming and panning, which are performed with the mouse wheel and drag actions. In order to maintain the time alignment among predictions and labels, each movement transform is applied equally to all tracks. By double clicking on a block or by selecting the name of a protocol label from the left sidebar, all tracks are conveniently scaled to show a close-up of the moment of interest. By hovering on a block, information about the correspondent prediction or label is shown (e.g. activity, duration) as a tooltip. In particular, the tooltip includes classifier-specific attribute values (e.g. “tremor frequency”, “angular velocity”) in the case of classifier tracks (Fig. 5).

Each classifier track further includes four buttons, enabling the user to (1) increase its height for better visibility, 2) play consecutively the videos of all detected activities, 3) display information about the underlying predictive model (e.g. sensors, prediction window and threshold used), and 4) switch between two different visualization modes. Fig. 2 explains how a classifier track can be represented also as an area chart, visualizing a continuous probability score over time. This mode is particularly useful for observing how the threshold of a classifier determines which events are positively predicted (and thus generate a block). The threshold can be dynamically changed by the user by moving the red horizontal line shown in Fig. 2, thus avoiding the recomputation of
classification results. All tracks can additionally be sorted by mouse dragging, allowing the user to better compare them visually.

### 3.2 Track Visual Algebra

While analyzing each track separately may be sufficient for some applications, in many cases the possibility to combine different tracks could be essential. For instance, a user may want to analyze the output of a tremor classifier only when a different classifier is predicting no walking movement. Similarly, a user may want to consider all moments in which a subject is stationary, thus needing to unify the labels associated to “Sitting” with the labels associated to “Standing”.

To enable reasoning beyond the scope of single classifiers and labels, we define a visual algebra that allows to generate new tracks as a combination of existing tracks. Operations such as addition, subtraction, logic conjunction and disjunction can be applied to both classifier and label tracks with a different semantic meaning. Fig. 3 illustrates how the most common operators can be used as a form of classifier validation. If we denote A a classifier track and B a label track used as ground-truth, A\(\land\)B corresponds to the intersection of both tracks, that is to the events that were correctly predicted by the model (true positives). Similarly, we can define difference between track A and track B as a new track where all block instances of B are removed from A. This way, the track A\(-\)B will contain all classifier detections that do not match any ground truth label (false positives), while B\(-\)A will conversely represent labeled events that were not identified by the predictive model (false negatives).

The power of the track algebra consists in enabling users to quickly combine tracks to validate complex hypotheses about the classification process. In particular, in presence of ground-truth labels, it makes the identification of misclassified events visually straightforward. In combination with the video functionality, it also enables to play consecutively all false positive and all false negative predictions for a particular classifier. This way, the user can visually validate the performance of his predictive model and reason on the causes of each single misprediction.

### 3.3 Command Line

Xplorer interface features a command line interface for enabling users to quickly perform complex interactions, such as track manipulation through visual algebra. Fig. 3.4 shows a list of the most common commands that can be executed from the command line. Each command is composed of one operator and one or two operands, which can be track identifiers or numerical values. A track identifier is automatically generated as a combination of the track name, author and version (e.g. the first version of the “Sleeping” classifier created by author “John” will generate the id “SleepingJohn1.0”) and is made available through auto-completion. For instance, while typing “threshold sle” the command line will automatically infer which available track is best suited for the operator “threshold”, highlight it in the main view and offer the suggested completion “threshold SleepingJohn1.0”. When a command generates a new track, this one is added to the main view and its name and identifier are automatically defined based on the operation performed.

### 3.4 Classifiers Validation

While observing a classifier track A and a ground-truth label track B next to each other, it is intuitive to understand that the performance of the predictive model depends on how much the blocks of each track are aligned with each other. Optimally, for each block in A there should exist a block in B of equal length, whose start end and points match the ones of A. Misclassifications and other prediction-related errors may however make one of this two blocks absent or misaligned. A straightforward, numerical way to quantify the visual overlap of two tracks is the Jaccard index, computed as their intersection over union. By sampling each track into a sequence of prediction values or binary labels, it is possible to compute different performance metrics commonly used in data science, such as accuracy score, precision, recall and F1 score.

Note that the value of all these metrics often depends on the threshold applied to the continuous prediction of a classifier. The choice of the threshold is often critical since it allows to balance the amount of true positives and false negatives allowed for a predictive model. For this reason, we include in the Xplorer interface a modal window displaying also a Receiver Operating Characteristic (ROC) plot (Fig. 4) with its related Area Under the Curve (AUC) score, a threshold-independent performance metric. In particular, the user can observe the percentage of true positives and false negatives associated to the current threshold and see how this percentage changes by modifying it.

### 4 USE CASE: BLUE SKY PROJECT

BlueSky project [2] aims at deploying predictive models to automatically assess the symptoms of Parkinson’s disease using wearable sensors. Xplorer was used by a team of fourteen data scientists and business people as a companion tool over most of the project.

A total of six wearable IMU sensors were used, worn by Parkinson’s disease subjects over sessions (visits) of about one hour. The sensors measured accelerometer, gyroscope and magnetometer information at 128Hz and were placed on the wrists, feet, chest and back of the patients. During each visit, all subjects performed the same set of predefined tasks, according to a single clinical protocol. A group of external technicians took care of recording sessions, labeling specific activities and time-stamping the execution of tasks. Sensor data, ground-truth labels and video files were all stored in a single database, that all data scientists could access during the development of their predictive models.

Each time a data scientist produced a new version of a model or algorithm, its classification results were loaded into our visualization...
with their legs crossed. Based on these insights, data scientists decided walking (false positives). By observing the video and the task labels, data scientists observed an incoherent labeling and an unclear definition of movements such as walking and turning. Should few short steps considered a walk? Should a larger rotation of the chest be considered a turn? If so, would they be useful to consider for the purpose of the project? Similarly, false positives showed the absence of a large amount of labels, which were not annotated by human operators because the subject was out of camera. Data scientists further noticed correct classification results were often misaligned with ground-truth labels: with the help of the video feature, it was demonstrated that human-generated labels generally occurred before a subject started a movement. For instance, a technician would annotate a walk whenever a subject showed the intention to move, without waiting for them to make a first step.

Finally, our visualization system played a relevant role in the phase of model validation. Balancing the amount of true positives and false negatives was of fundamental importance for the project, which required being certain of positive predictions more than correctly predicting all events. The possibility to dynamically change the threshold applied to classifier tracks, combined with the ROC curve visualization, helped data scientists visually and quantitatively observe prediction changes.

### 5 Conclusion

In this paper we present a new visualization tool, Xplorer, to query and visually analyze classification results originated from sensor temporal data. In particular, Xplorer offers contextual information such as ground truth labels and video that data scientists can use to better interpret the performance of predictive models. Through a real-life use case, we demonstrate how our system is suitable for understanding the causes of misclassifications, improving classifiers performance and identifying early on possible systemic errors in the data.

### References


