DETOXER: A Visual Debugging Tool with Multi-Scope Explanations for Temporal Multi-Label Classification

Mahsan Nourani
University of Florida

Chiradeep Roy
University of Texas at Dallas

Donald R. Honeycutt
University of Florida

Eric D. Ragan
University of Florida

Vibhav Gogate
University of Texas at Dallas

***Abstract*—In many applications, developed deep learning models need to be iteratively debugged and refined to improve the model efficiency over time. Debugging some models, like Temporal Multi-Label Classification (TMLC) where each datapoint can simultaneously belong to multiple classes, can be specially more challenging due to the complexity of the analysis and instances that need to be reviewed. In this paper, focusing on video activity recognition as an application of TMLC, we propose *DETOXER*; an interactive visual debugging system to support finding different error types and scopes through providing multi-scope explanations.**

WITH DEEP LEARNING MODELS used in many applications—often in contexts with high stakes and critical outcomes— iteratively *debugging* the models for errors and further *refining* them becomes an essential step in developing and deploying deep learning models [1]. In this paper, we focus on debugging *Temporal Multi-Label Classification* (TMLC) models, which address the problem of assigning multiple labels (classes) to all data points in an input sequence, assuming all data points in the sequence are related to each other [2]. These models are used in a wide range of applications, including video activity recognition and annotation.

In practice, debugging TMLC models can easily grow exponentially with the number of labels. For example, in each single video frame (used for activity recognition), there are 2^n possible combination of labels to examine for errors (with n as number of labels defined). Therefore,

Published by the IEEE Computer Society

examining all possible combinations of labels within the video becomes exponentially hard. Additionally, with sequential data, an analyst must reason about how errors in one datapoint (e.g., video frame) can propagate and relate to others within a sequence. With deep learning models often trained and verified on hundreds or thousands of instances, analysts need to examine large collections of all possible combinations when debugging TMLC models. Even if exploring all combinations was practical, perceiving model uncertainties and outputs in numerical or tabular formats can be challenging and may obscure error patterns or common problems among instances.

We present research examining video activity recognition, one of the prominent applications of TMLC models, with videos as sequential inputs, frames as data points, and the labels being activity components (i.e., *actions* + *objects* and/or *locations*). Motivated to overcome some of the debugging challenges in this application, we propose *DETOXER*, an explainable, interactive visual analytics tool for debugging video activity recognition models. With the deep learning models being complex and hard to comprehend, we designed our tool to provide multiple scopes of explanations and outputs: 1) frame-level outputs to represent temporal data points and to provide a compact visualization for the combination of possible labels; 2) video-level explanations to provide an overview of the errors and labels for each unique instance; and 3) global-level explanations to reveal error trends and patterns across all the instances.

Our tool is designed to support error detection of variable scopes (errors that occur on instances vs. high-level patterns that can fix problems globally) and types (false positive and false negative errors) through exploration and interactive visualization. This paper contributes our research summarized with the following:

1) We demonstrate the *DETOXER* explainable visualization system for exploratory debugging video activity recognition models.
2) We present the results of a human-subjects evaluation.
3) We provide two case studies on models from two domains to demonstrate how our tool can support model designers in their exploratory model debugging.

1. Design Goals

Explainable AI (XAI) approaches aim to improve a model's decision-making transparency, and our work focuses on explanation for model analysis and debugging. Explanations can help identify *when* the model makes errors and *why* [3], along with model-wide problems. TMLC adds a time dimension to the multi-label classification problem. With these models, the same instance can be assigned to multiple classes (labels), and with the inputs being temporal, what is detected at any given moment relies on what is observed earlier. This paper specifically focuses on the problem of debugging video activity recognition models, a prominent usecase of TMLC. We summarize our main Design Goals (**DG**) in four categories:

DG1) Multi-Scope Explainability. To improve human understanding of how the model works, we aim to provide explanations of various scope. Many applications incorporate instance-level explanations, which are less overwhelming for novices and support debugging edge cases in the model [3]. However, these explanations may limit the ability to find systematic issues with large datasets [4] and require usage over time to achieve that. When working with multi-label classification, instances that are detected as matches for any given class are often a smaller subset than non-matches and easier to examine, while users may be biased towards inspecting more false positives than negatives [4]. We provide additional global explanations to help contextualize instances with model-level information.

DG2) Support for Guided Exploration. Visual analytics tools allow for exploratory approaches to debugging machine learning models, but it can be difficult to find a good starting point for candidate problems. Verifying whether an observed instance-level error is also a systematic problem can also be challenging. To this end, we aim to support guided exploration of the outputs by calling attention to underlying problems with instances and labels in our design.

DG3) Supporting Both Multi-Label and Temporal Comparisons. With thousands of possible combinations of labels to show per video, our goal is to minimize the presented information to improve the exploration while providing enough context for error detection and explorations. Additionally, it is important for the design to integrate the temporal nature of the input, as data points (frames) in a sequential data (video) might depend on the prior data point(s). Our design aims to support comparisons of various labels at any given frame (time).

DG4) Supporting Detection of Type I and II Errors. Debugging machine learning models should go beyond merely finding *any* errors, as some types of errors can be more prominent or even overshadow others. Thus, we aim to encourage attention to different types of errors—specifically, we call attention to *False Positive* (FP or type I) and *False Negative* (FN or type II) errors. Depending on the domain and how critical the task is, detection of either or both these errors becomes vital. For instance, with anomaly detection in surveillance cameras, it is more consequential when malicious activities are not identified (i.e., false negatives). In general, many applications that showcase model-generated outputs fail to highlight both false negatives and false positives. This is because the focus is on highlighting detection hits when displaying outcomes, which comprise true and false positives. Since both false positive and false negatives are important in the debugging process, our goal is to design our tool to intuitively steer user attention to both of these error types.

2. Model

The machine learning model we use in this paper comprises a two-tiered system that uses a CNN based on the BAIR/BLVC GoogleNet architecture [5] and a tractable dynamic probabilistic model to model the temporal dependencies between the labels as a joint probability distribution [6], [7]. With this two-tiered architecture, the first layer serves as a *video classification* black-box model, while the second layer a) generates explanations for probabilistic queries and b) refines the accuracy of noisy labels at each frame to obtain the true ground label. For more information on the model specifics and technical details, we refer the readers to our prior work [8], or the supplemental material of this paper. We emphasize that our application and presented visual designs can work with any TMLC model for video activity recognition as long as individual label probabilities (w.r.t to the model) can be calculated at each time slice.

3. System Design

To address our design challenges, we propose *DETOXER*, an interactive debugging tool that supports detecting errors of various types and scopes in video activity recognition models. Our system supports three levels of explanation/outputs: (1) **Global explanations**, highlighting performance information extracted from the *model*–Figure 1 (D) and (E); (2) **Video-level explanations**, highlighting high-level information about each *video*–Figure 1 (A) and (B); and (3) **Frame-level explanations**, demonstrating information specific to each *frame* (image) of a selected video–Figure 1 (C). Although both (2) and (3) provide explanations on the instance level, (2) can be considered more global on the explanation scope spectrum.

The web-based interface for *DETOXER* is implemented using HTML/CSS, React.js, D3.js, and Material UI (MUI). [1] An overview of our visualization is shown in Figure 1. In this section, we will first describe interface design components in our tool and how the information they present is extracted from the model to support the design. We will then briefly describe how each of these elements can be used concurrently or separately to support our design goals.

3.1. Heatmap View

In our model, an activity consists of up to three components: an action + an object and/or a location. For each of these activity components, we have a set of labels that are defined for the model through data annotations. Our main goal was to allow a high-level view of the activities within a given video where users can get a grasp of activity components detected by the model by simplifying the examination of thousands of

[1]The application is open-sourced and available here: https://github.com/MahsanNourani/DETOXER.

Figure 1: The overview of *DETOXER*. In the center, a video is selected for exploration. Directly under the progress bar, heatmaps demonstrate the model's confidence for any given label per second (C) (*frame-level explanations*). On the left, available videos are shown; for each video, the tool shows top-5 detected labels (A) and the rate of FP and FN errors (B) in the video (*video-level explanations*). The selected video is emphasized with a blue background. On the right, a global information panel displays model performance metrics (D) and *object*-specific FN and FP error rates in two vertically adjacent bar charts (E) (*Global-level explanations*).

possible combinations (**DG3**), while supporting the detection of both error types (**DG4**). Moreover, we sought after a visualization approach that would highlight the temporal aspect of video data (i.e., **DG3**) [9], [10]. To this end, for each activity component defined in our vocabulary (comprised of actions + objects and/or locations), we provide a temporal heatmap under the video progress bar (see Figure 1). Heatmaps represent probabilities extracted from the model, and we simply define them as the model's confidence score for whether a given component is present at the given time. For the currently selected video, the heatmap rows are statically ordered based on relevance of labels.

The component-wise confidence scores in the heatmaps were extracted from the model and show the marginal probabilities conditioned on the evidence from the neural network. If $X_k^{(t)}$ is the random variable associated with the k-th component of the ground label vector at time slice t for video v, then the confidence score is given by $P(X_k^{(t)} = 1|\boldsymbol{y}^{(1:T_v)})$ where T_v denotes the number of frames in video v and $\boldsymbol{y}^{(1:T_v)}$ denotes the noisy labels obtained from the neural network for all the frames in v. We also computed a relevance score for each component label and ordered them in descending order of their scores. The relevance score for each predicted label y in a specific video v was computed as $R(y,v) = F1(y,v) \cdot \#frames(y,v) \cdot \frac{1}{first\_frame(y,v)}$ where $F1(y,v)$ is the F1 score of y in v while $\#frames(y,v)$ and $first\_frame(y,v)$ are the number of frames and the first frame respectively that y is detected in v.

3.2. Global Information Panel

In accordance with **DG1** and **DG2**, we sought to design an information panel where users could review estimations of the model's overall strengths and weaknesses to guide inspection of different errors while debugging. We designed a *global information panel* to show general information—extracted and calculated from the model—to guide user explorations. Note that all the information represented in this panel are

Table 1: The Jaccard Index computation per video frame.

True Positive: $TP(x,y) = x \cdot \mathbb{1}\{x = y\}$	**False Positive:** $FP(x,y) = x \cdot \mathbb{1}\{x \neq y\}$
True Negative: $TN(x,y) = (1-x) \cdot \mathbb{1}\{x = y\}$	**False Negative:** $FN(x,y) = (1-x) \cdot \mathbb{1}\{x \neq y\}$

overall estimations considering the entire data set (i.e., all videos). At the top of the panel, we show three metrics, including model accuracy (higher values better), overall rate of false positives (type I errors), and false negatives (type II errors) in the model (lower values better); see Figure 1 (D). We also visualize the rates of false positive and false negative errors per each object in the vocabulary via two bar charts (Figure 1 (E)). Bar charts are vertically aligned to support easier comparison of type I and II errors for the same object. Higher percentages represent higher number of errors. Prior research demonstrates the effectiveness of the presence of such information on early-on debugging attempts [11], [12].

The overall accuracy of the model was measured using the Jaccard Index which is defined as the ratio between the intersections and the unions of the ground (true) labels and the predicted labels respectively. This metric is commonly used in multi-label classification system since it measures the degree of overlap between the true labels and predicted labels and yields a value of 1 when both sets are exactly the same and 0 when they are disjoint with nothing in common. The Jaccard Index was computed for each frame of each video and then averaged out over all the frames and videos, as seen in Table 1.

The false positive rate (FPR) is defined as the ratio between the number of false positives and the total number of false positives and true negatives. The false negative rate (FNR) is the ratio between the number of false negatives and the total number of false negatives and true positives. These are computed for each video and then averaged over over all videos. FPR and FNR are important metrics because they provide the user with an estimate of how frequently they can expect the model to make FP and FN errors respectively. The usefulness of these metrics for debugging is highlighted in section 5.

3.3. Video Preview

While the global information panel provides high-level context (**DG1**) and supports exploration (**DG2**), it does not incentivize instance-level explorations and might be insufficient as the number of instances to explore grows. To address this problem, we provide summarized, high-level information per video to explain model errors per video and show the most prominent labels detected in the video.

On the left-side of our application is a list of all videos available to the model. Each item in the list includes a video preview with a thumbnail (showing the middle frame of the video) and descriptive model information. We show the top-five activity components (labels) the model detected for the video as simple tags (in pink; Figure 1 (A)). We also show the calculated percentage of FPR and FNR for the video (Figure 1 (B)). In the example of Figure 1, the model was more likely to not detect labels/activities that are taking place rather than detecting something by mistake.

The FPR and FNR for each video use the same formula described in subsection 3.2 with the only difference being that we compute these on a per-video basis and therefore average out only over the number of frames present in the current video. To calculate the top-five detected tags, we compute a relevance score (as we did in subsection 3.1) per each activity component (label) defined in the system and pick the components with the top five scores. The relevance score is calculated for a predicted label y and a video v as $T(y,v) = F1(y,v) \cdot \frac{1}{\sum_v present(y,v)}$ where $F1(y,v)$ is the $F1$ score while $present(y,v)$ is a function that evaluates to 1 if label y is on for video v for at least one frame and 0 otherwise. This was motivated by the notion of assigning higher priority to rarer labels that are specific to a select subset of videos.

3.4. System Usage

Each of the designed interface elements can be used separately or together throughout the

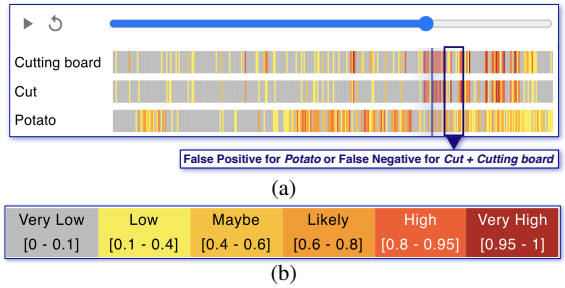

Figure 2: (a) Example heatmaps under the progress bar and comparisons to detect type I and II errors in activities. The vertical blue strip is synced with playback time. (b) Heatmap color map.

debugging process. In this subsection, we describe the intended utility of our visual design via a hypothetical example to showcase a potential debugging workflow that would not require expertise in machine learning.

The global information panel can be beneficial with context-aware error detection as it presents global-level explanations regarding the model (i.e., explanations that are not specific to any instances). As prior work shows, local explanations can limit mental model formations [4]. Our design aims to counteract such problems by including descriptive model information and directing user awareness towards varying levels of system performance.

Furthermore, users might not know where to look at or what to look for when debugging a model. Global explanations in this panel support and contextualize exploratory debugging, especially earlier in the debugging process or when users struggle with error discoveries. In our example approach, users can initially understand the model's performance through inspecting the global information panel (D). For example, they might notice that the model is prone to more FN errors than FP. This can encourage them to explore potential false negatives and decide what components to investigate further. Using the false negative bar charts ((E)), users find components with the highest possible error rate. In case of multiple possibilities, users can compare those based on their false positive errors choose the one that is suspected to maintain the highest error values in both error categories.

Let us assume a user decides to investigate *object X*. The next step is to identify which videos contain the selected object. Extracting and visualizing explanations for the video instances and generating a high-level overview of each video is another aid for users in their debugging process. These features can be used separately or together with the global information. In our example, since there are many videos available, it might be the wiser choice to refer to the top detected components (A) to determine which videos contain *object X*. While this narrows down the search, the decision of which video to explore first remains. Using the detection error information in the video preview (B), the user can decide to explore the video with the highest FNR. Even if the user ignores the global information panel, using these two features is still useful in selecting a video to explore. Additionally, this design supports the discovery of higher-level error patterns that are concurrent across various instances, as users can skim through the available videos for specific components and drill down their search to only those videos. Such technique can generally be used when many videos exist where a video overview can be beneficial. Other techniques, such as filtering, can be utilized to improve interactivity and the debugging process based on the context, domain, and number of data inputs.

From this point forward, exploring videos through using and comparing heatmaps (C) allows for finding FN/FP errors with *object X*. Our heatmap design aligns with our main design goals (i.e., **DG1** and **DG2–3**) and provides support throughout the debugging process. Having one temporal heatmap per activity component (i.e., label) provides an overview of model predictions based on video frames while stacking them allows for comparisons across multiple labels and identifications of various combinations simultaneously. In our example, a user may observe that the model does not detect *object X* when the person is performing *action Y* on it. The heatmaps provide a simple means to visually locate such correlations and to compare the patterns of *action Y* and *object X*. For instance, when exploring the heatmaps from *cut* and *cutting board* in Figure 2a, we can detect a logical correlation between the

Table 2: Summary of *significant* results for user reported helpfulness and usage of different design elements in *DETOXER*. Bold texts represent the more-used elements.

(a) Helpfulness	Main Effect
	$\chi^2(5) = 24.9, \eta_p^2 = 0.33, p < 0.001$ *
	Post-Hoc Test
	pairwise comparison failed to detect significant differences.
(b) Usage	Main Effect
	$\chi^2(5) = 32.2, \eta_p^2 = 0.43, p < 0.001$ *
	Post-Hoc Test
	Heatmaps vs. FN Bar charts ($p < 0.001$) *
	Heatmaps vs. FP Bar charts ($p < 0.001$) *
	Heatmaps vs. Overall Performance Metrics ($p < 0.001$) *

two. The video is focused around the activity of cutting a *potato*; however, there are occasions where the *potato* is not detected, although *cut* and *cutting board* are detected. This can either indicate a false negative error for *potato*, or a false positive error for the other two components. We could therefore show that displaying overall label detection coupled with color-coded model confidence for individual activities in the heatmap can reveal both FPs and FNs in our model.

Finally, exploring multiple videos that include *object X* might reveal hidden shared patterns across videos. For instance, assume that the false negative error with *activity Y + X* happens in more than one video; thus, the user can identify this as a higher-level problem and fixing it globally can solve this problem in all or most of the videos.

The debugging process presented in this section provides context for how debugging TMLC models might work using *DETOXER* or similar implementations. Since it supports open-ended exploration, analysts are able to adjust their debugging approach according to their goals and level of machine learning familiarity. Such added flexibility allows analysts to adapt to different error detection techniques over time in the iterative debugging process. In the following sections, we demonstrate how *DETOXER* supports other workflows and usage scenarios for both non-expert users and model experts.

4. User Study

We conducted a user study (approved by our Institutional Review Board) to evaluate the debugging tool and assess the effectiveness of the different interface components in an exploratory debugging scenario. For this study, we used a model (described in Figure 2) that was trained on activity recognition with videos of kitchen activities using the TACoS Corpus dataset [13].

4.1. Study Design and Procedure

We designed a study where participants were asked to a) find false positive and false negative errors with the model and b) identify higher-level problems that are common across multiple inputs. The task was limited to 30 minutes, and participants were free to inspect any number of videos. Additionally, participants were asked to use a think-aloud approach and verbalize errors as they found them. After the main task, participants completed a questionnaire, rating their usage and the helpfulness of each interface element on an ordinal scale, followed by a short interview to better understand users' approach.

We recruited 15 participants (8 females, 7 males) who were undergraduate and graduate students from computing fields with some familiarity with software debugging. The study was conducted online through video conferencing, with the study lasting approximately 60 minutes.

4.2. Study Results

We first present the quantitative analysis for the self-reported usage and helpfulness of the interface elements of *DETOXER* via repeated-measures statistical analysis. With both measures being ordinal, we used a Friedman statistical test for the main effect and a Wilcoxon signed-rank test with Bonferroni correction for pairwise comparisons.

Table 2 shows the summary of the main quantitative findings and the significant comparisons, and Figure 3 shows the distributions of the responses. As demonstrated in Table 2, while the omnibus test for both *usage* and *helpfulness* measures showed evidence of significant differences among interface elements, posthoc testing only found evidence of pairwise significant differences for self-reported *usage*. The posthoc test in usage indicates that participants used the heatmaps (Figure 1 C) significantly more than explanations on the global information panel (Figure 1 D and E). The distribution of the self-reported *helpfulness* of the design elements (as seen in Figure 3)

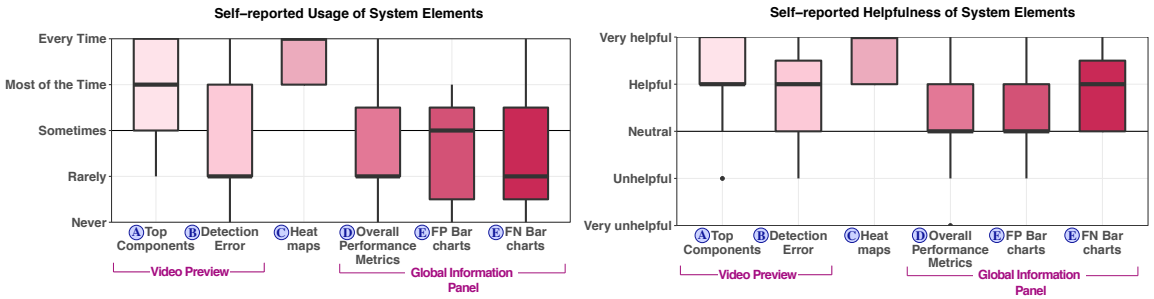

Figure 3: Participants' self-reported usage and helpfulness for the main elements of *DETOXER*.

shows that, overall, most of the participants found them helpful, despite variations in usage. For instance, despite using them less than heatmaps, participants still found the global information panel helpful in their task.

To better understand these results, we interviewed participants about their debugging approach and usage. Here, we summarize the findings of qualitative analysis focusing on error types and usages of interface elements.

The heatmaps and video preview panel were the most used design components (Figure 3); 86% and 60% of the participants mentioned they used them. Additionally, around 40% of participants explicitly mentioned heatmaps supported the identification of and brought their focus to both FP and FN errors at the same time. Following our design motivations (**DG4**), this observation provides further evidence that *DETOXER* promotes attention to both error types.

We further found three debugging trends based on the affinity diagram:

1) **Instance-level debugging**: Participants relied heavily on video-specific explanations and designs; i.e., video preview explanations (Figure 1 Ⓐ and Ⓑ), heatmaps (Figure 1 Ⓒ), and the video). The total of 67% participants incorporated this approach, who stated that they either explored videos (1) in no specific order, (2) in the order they appeared, or (3) based on the top components or the error detection per video to decide what to explore.
2) **Global-level debugging**: 13% of participants relied on the global information panel to first identify objects with the highest known errors (Figure 1 Ⓔ), and then looked for videos containing those items to investigate why the system is making these mistakes.
3) **Multi-level debugging**: 20% of participants changed their approach at different times (i.e., a hybrid approach). These people reported relying more on the global panel later in their debugging process or verified their exploration directions by referring to the provided global explanations.

These trends confirm that *DETOXER* provides support for various debugging approaches, as users were able to focus on both global and local inspections depending on their goals (**DG1**). For example, one participant explained their **global-level debugging** approach:

"*...I looked at the false positive and false negative rates per object [bar charts] ... [and] picked objects from either of these categories with higher [error] values and associated them with the videos in the top components. So when I looked at a video and the heatmap, it provided a better insight into [the activity].*"

Another participant described their **instance-level debugging** as:

"*...I was watching the video and seeing the heatmaps. That was really my process, to see if [the heatmaps] were accurate. I didn't use the [global information] panel as much, I looked at them but I wasn't really thinking about the percentages; more if they were accurate or not*".

The findings from this study demonstrate *DETOXER*'s flexibility for enabling different approaches to model debugging, and the results provide evidence that general users are able to understand and effectively use such design, which aligns with the intended *system usage* discussed in subsection 3.4.

5. Case Studies in Model Debugging

In this section, we describe *DETOXER*'s case studies through two different activity recognition models trained with two video data sets: the TACoS dataset of kitchen activities [13] and video dataset of wet laboratory procedure [14]. As opposed to the user study, the goal in the case study is to demonstrate the utility of *DETOXER* for experienced model architects in diagnosing both system-wide and instance-level model errors. For both models, the activities included *action* and *object*, with the cooking dataset including *location* as well. In this section, we briefly describe four examples of error types identified through debugging with *DETOXER*. *More details of the case study can be found in the supplemental material.*

5.1. Confusion Pairs

A confusion pair is when a single ground label x is mistaken for a predicted label y such that $x \neq y$. *DETOXER* makes identifying these errors much more efficient by providing a visualization for the distribution of confidence scores for each label through heatmaps (Figure 1 C). For instance, with the cooking dataset, we examined videos including *pineapple* as a top-detected component (Figure 1 B) to investigate why they have the highest FPR. After thorough examination, we found that the model mistakenly detected pineapples when the edible object was cucumber.

5.2. Error Rules

Almost all of the errors made by the system can be succinctly summarized as probabilistic rules of the form $p^{(i)} : \boldsymbol{x^{(i)}} \rightarrow \boldsymbol{y^{(i)}}$ where $p^{(i)}$ is the probability of the ith rule, $\boldsymbol{x^{(i)}}$ is a set of ground labels and $\boldsymbol{y^{(i)}}$ is a set of predicted labels. The system supported such error debugging for both models. For example, in the wet lab model, we observed the FNR was relatively high for certain objects (over 30%) despite the model's high accuracy (around 84%). The label *salt bottle* had a FNR of around 70%. We examined the videos and the heatmaps to find that *salt bottle* was almost always missing whenever *get* and *pipette* were present. Querying the model returned a probability of 0.98 giving us the error rule: $0.98 : get, pipette, salt\ bottle \rightarrow get, pipette$.

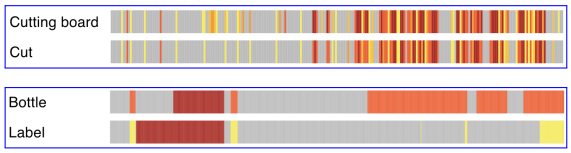

Figure 4: Heatmap persistence comparison between cooking (top two: lower persistence) and wet lab (bottom two: higher persistence) datasets.

The heatmap helped us quickly isolate videos to confirm these error patterns—something that would have been difficult to detect otherwise.

5.3. Low Confidence Negations

These are false negatives with mid to low confidence scores (i.e. between 0.2 to 0.6). For instance, for the debugged model for cooking videos, *bowl* had a relatively high FNR (Figure 1 E). By checking and isolating heatmap areas with low to mid confidence scores in videos with *bowl*, we identify that while *bowl* was *detected* by the model, the model had lower confidence in its prediction; thus, increasing FNR. *DETOXER* helps highlight such low confidence negations that could be fixed by improving train data to improve model's confidence.

5.4. Persistence-based Errors

In the context of TMLC problems, we define "*persistence*" of a system to be its tendency to keep labels on after they are first detected. A highly persistent model would tend to keep a label on for a longer period of time, once activated. In our case studies, the cooking model had low persistence and the wet lab model had high persistence (see Figure 4). This was clearly visible upon examining the heatmaps for each video. For example, action labels like *cut* and *peel* have heatmaps that are highly fragmented into multiple tiny sections (around a second or two long). Due to *DETOXER* showcasing this visually, analysts can become aware of the problem and rectify by introducing solutions like adding regularization terms.

6. Discussion and Conclusion

DETOXER provides multiple explanation scopes to support systematic understanding of TMLC model errors both globally and locally. The interactive visualization tool provides: *frame-level*

model predictions through temporal heatmaps to support comparing combinations of labels over time; *video-level* information specific to video via showing the top-five detected labels and the overall FPR and FNR; and *global-level* explanations including model performance measures and bar charts showing FPR/FNR for *all* the activity components of objects to support systematic exploration and analysis when detecting errors. The presented usability study demonstrates the effectiveness of *DETOXER* for various exploratory debugging workflows, and the two case studies showcase how it can be utilized by machine learning specialists for elaborate, technical error detection.

The system is a novel, open-sourced debugging application in the context of video activity recognition. Our visual analytics approach is not domain-dependent and may be used for any temporal model where individual label probabilities can be calculated at every time frame. Our approach/system can extend to other video activity recognition or TMLC applications, regardless of the number or length of the videos. For instance, in the anomaly detection domain, each anomaly can be represented by one heatmap, and the colors can vary based on the importance of the anomaly or model's confidence in detecting them. Furthermore, combinations of anomalies might be explored by comparing various heatmaps at the same time when a higher-level anomaly consists of smaller irregularities or when new anomaly behaviour patterns need to be explored.

While the usage of our heatmap representation is more specific to temporal and sequential data, the usage of the model-agnostic design features can be extended to other models and tools. For instance, the model performance metrics and the FPR and FNR (both local and global variations) can be incorporated by other models to improve explainability for debugging purposes or to improve user mental models.

We hope our visual analytics tool motivates future debugging applications and systems for TMLC models, and our model-agnostic design features can inspire future visualization of transparent machine learning models.

7. ACKNOWLEDGMENT

The authors thank *Jeremy Block* and *Shivraat Arya* for their contributions. This work was supported by the DARPA XAI Program under award number N66001-17-2-4032 and by NSF award 1900767.

REFERENCES

1. S. Liu, X. Wang, M. Liu, and J. Zhu, "Towards better analysis of machine learning models: A visual analytics perspective," *Visual Informatics*, vol. 1, no. 1, pp. 48–56, 2017.
2. E.-S. Kim, K.-W. On, J. Kim, Y.-J. Heo, S.-H. Choi, H.-D. Lee, and B.-T. Zhang, "Temporal attention mechanism with conditional inference for large-scale multi-label video classification," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.
3. S. Mohseni, N. Zarei, and E. D. Ragan, "A survey of evaluation methods and measures for interpretable machine learning," *arXiv preprint arXiv:1811.11839*, vol. 1, 2018.
4. M. Nourani, C. Roy, J. E. Block, D. R. Honeycutt, T. Rahman, E. Ragan, and V. Gogate, "Anchoring bias affects mental model formation and user reliance in explainable ai systems," in *26th International Conference on Intelligent User Interfaces*, 2021, pp. 340–350.
5. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
6. C. Roy, T. Rahman, H. Dong, N. Ruozzi, and V. Gogate, "Dynamic cutset networks," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 3106–3114.
7. T. Rahman, S. Jin, and V. Gogate, "Cutset bayesian networks: A new representation for learning rao-blackwellised graphical models," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019, pp. 5751–5757.
8. C. Roy, M. Nourani, D. R. Honeycutt, J. E. Block, T. Rahman, E. D. Ragan, N. Ruozzi, and V. Gogate, "Explainable activity recognition in videos: Lessons learned," *Applied AI Letters*, vol. 2, no. 4, p. e59, 2021.
9. Y. Goyal, A. Mohapatra, D. Parikh, and D. Batra, "Towards transparent ai systems: Interpreting visual question answering models," *arXiv preprint arXiv:1608.08974*, 2016.

10. H. Arnout, M. El-Assady, D. Oelke, and D. A. Keim, "Towards a rigorous evaluation of xai methods on time series," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2019, pp. 4197–4201.
11. B. Alsallakh, A. Hanbury, H. Hauser, S. Miksch, and A. Rauber, "Visual methods for analyzing probabilistic classification data," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 1703–1712, 2014.
12. J. Krause, A. Dasgupta, J. Swartz, Y. Aphinyanaphongs, and E. Bertini, "A workflow for visual diagnostics of binary classifiers using instance-level explanations," in *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 2017, pp. 162–172.
13. M. Regneri, M. Rohrbach, D. Wetzel, S. Thater, B. Schiele, and M. Pinkal, "Grounding action descriptions in videos," *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 25–36, 2013.
14. I. Naim, Y. Song, Q. Liu, H. Kautz, J. Luo, and D. Gildea, "Unsupervised alignment of natural language instructions with video segments," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, Jun. 2014.

Mahsan Nourani is a Ph.D. candidate at the University of Florida and the primary contact of this paper (mahsannourani@ufl.edu). Her research interests include human-computer interaction, human-centered AI/XAI, and decision-making.

Chiradeep Roy is a Ph.D. candidate at the University of Texas at Dallas. His research interests include probabilistic graphical and tractable models and XAI.

Donald R. Honeycutt is a Ph.D. student at the University of Florida. His current research is centered around human-in-the-loop machine learning, mixed initiative systems, XAI.

Eric D. Ragan is an Assistant Professor at University of Florida. His research includes of human-computer interaction, visual analytics, virtual reality, and XAI systems.

Vibhav Gogate is an Associate Professor at the University of Texas at Dallas. His focus is on probabilistic graphical models, their first-order logic based extensions, and probabilistic programming.