July 10, 2019

Using Sensors to Improve User Interaction with Application Notifications

Victor Carbune

Sandro Feuz

Using Sensors to Improve User Interaction with Application Notifications

Abstract:

This publication describes techniques and methods for improving user interaction (*e.g.,* dismissal, expansion) with application notifications on computing devices, such as smartphones, tablets, or smart glasses. The techniques incorporate the utilization of multiple on-device sensors (*e.g.*, a camera sensor, an accelerometer) that can measure various means of user input. The methods as described herein afford users convenient and quick notification interaction options. Machine-learned (ML) models that employ gaze detection, custom gestures (*e.g.*, hand movement, eye clipping), and/or user actions (*e.g.*, shaking or rotating the device) can provide users the ability to expand, individually dismiss, or batch dismiss application notifications exclusive of or in conjunction with application notification priority levels (*e.g.*, the urgency of the notification, preselected user importance levels).

Keywords:

Application notification, computing device, user interaction, machine-learning, gaze detection, eye tracking, eye movement, smartphone, device gestures, lock-screen, notification dismissal, user-interface (UI)

Background:

Computing device (*e.g.*, smartphone, tablet) users often take out their device, glance at the application notifications, and dismiss a majority by swiping their finger(s) left-right across the user-interface (UI) (*e.g.*, screen) for every notification. Depending on the amount of application notifications, this process of dismissing notifications can often be inconvenient and slow. After

dismissing most, users often expand the remaining important notifications by selecting them on the UI with their finger.

It is desirable for users to interact with application notifications by means of multiple on-device sensors (*e.g.*, a camera sensor, an accelerometer) which can measure various forms of user input, to the end that the users can be provided more convenient and quicker methods to dismiss or expand application notifications.

Description:

This publication describes techniques and methods for improving user interaction with application notifications on computing devices. The techniques incorporate the utilization of multiple on-device sensors that can afford computing device users convenient and quick methods to interact with the application notifications. Such methods include the ability to expand, individually dismiss, or batch dismiss application notifications based on user input.

Figure 1 illustrates an example computing device and elements of the computing device that support the methods to dismiss application notifications by user input.

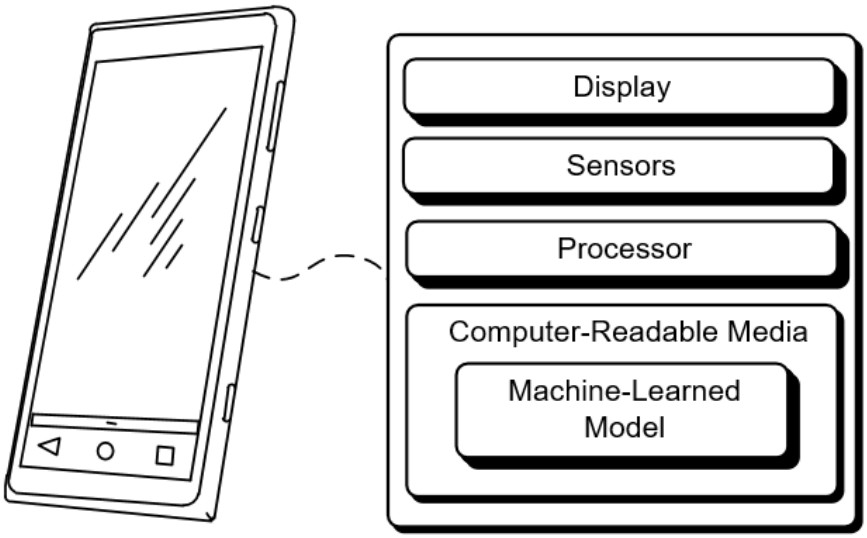

Figure 1

As illustrated in Figure 1, the computing device is a smartphone. However, other computing devices (*e.g.*, tablets, smart glasses, watches) can also support the techniques and methods described in this publication. The computing device includes a display (*e.g.*, a light emitting diode (LED) display, a liquid crystal display (LCD)) and sensors (*e.g.*, an accelerometer, a gyroscope, a camera, a millimeter wave sensor, an infrared position sensor). The computing device also includes a processor and a computer-readable medium (CRM) that may consist of any suitable memory or storage device such as random-access memory (RAM), static RAM (SRAM), dynamic RAM (DRAM), non-volatile RAM (NVRAM), read-only memory (ROM), or flash memory that contains executable instructions for implementing the techniques and methods described in this publication. Further, the CRM may include the operating system (OS) of the wireless-communication device, applications installed on the wireless-communication device, and a machine-learned (ML) model (*e.g.*, neural networks).

The ML model may include a computer vision model for gaze detection and a sequence model for gesture recognition. The gaze detection model can be trained on labeled images of eyes and respective gaze direction (*e.g.*, the point of gaze of the user, the motion of any eye of the user relative to the user's head). The sequence model can be constructed by standard neural network blocks from collected data for gesture types. After sufficient training, the ML model can be deployed to the CRM of the computing device.

Figure 2, below, illustrates an example of how the ML model implemented in the CRM of the computing device may recognize the various forms of user input by which users can interact with the application notifications.

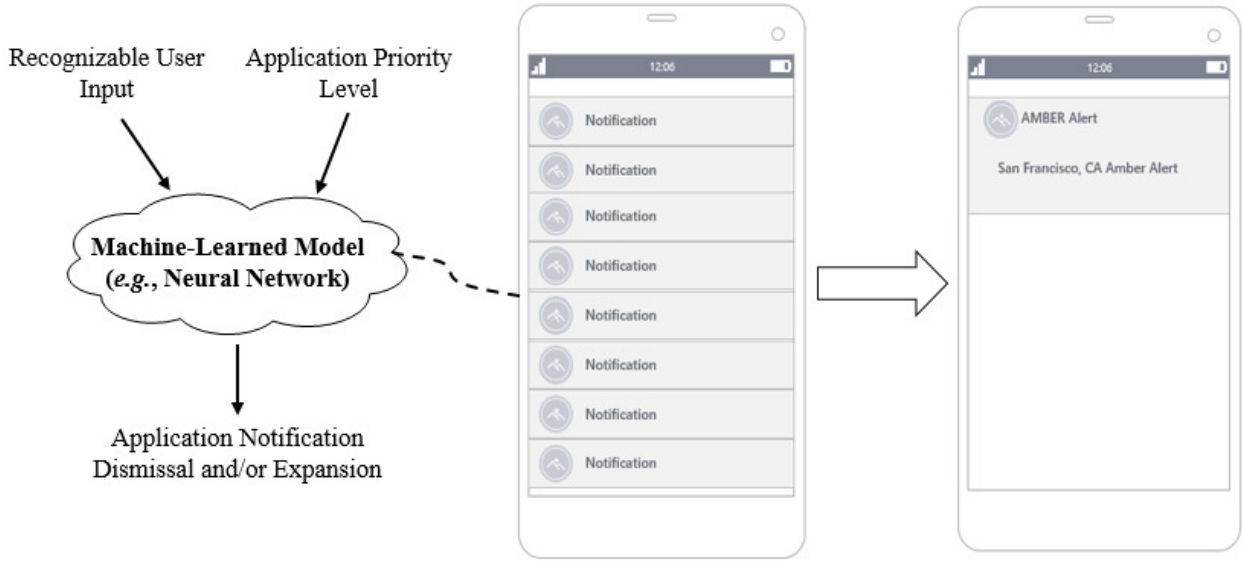

Figure 2

The sensors on the computing device can measure the user input (*e.g.*, gaze detection, custom user gestures, and/or user actions). Furthermore, the ML model can identify a priority level for every application notification based on urgency or preselected settings. The combination of user input and application notification dismissal or expansion can be implemented at the framework level of the computing device by the introduction of a code layer handling the ML models. In other words, the ML model would be inserted into the existing code layer schema, specifically at the computing device's lock-screen layer, so that user input and application notification dismissal or expansion can be achieved. As illustrated in Figure 2, the ML model can measure the user input, dismiss the non-urgent application notifications, and expand the urgent notification.

Examples of user input combined with priority levels to dismiss or expand application notifications include: 1) using the ML model to batch dismiss all application notifications when it has been determined that the user has gazed at all the notifications; 2) using the ML model to dismiss application notifications one-by-one by repeated precise gaze identification and implicit

visual confirmation; 3) utilizing the ML model to measure action-based input (*e.g.*, shake, rotation) to batch or individually dismiss application notifications predicated on priority levels; or 4) utilizing the ML model to measure action-based input to expand the highest priority level application notification or dismiss the lowest level application notification(s). The ML model, in the preceding examples, received various forms of user input by means of the computing device's sensors (*e.g.*, an accelerometer, a gyroscope, a camera, a millimeter wave sensor, an infrared position sensor). In these examples, the user input acted in conjunction with the determined application notifications' priority levels, to the end that the ML model could determine whether to expand or dismiss the application notifications and whether to batch or individually dismiss the application notifications.

Alternatively, the ML model can dismiss or expand application notifications exclusively based on user input. Examples of the dismissal or expansion of application notifications without associated priority levels include: 1) using the ML model to identify which application notifications to expand and 2) utilizing the ML model to measure the user shaking the computing device to dismiss the application notifications. In these examples, the user input operated exclusive of priority levels in dismissing the application notifications.

Figure 3A and 3B illustrate two user input example forms utilized to interact with application notifications at the lock-screen.

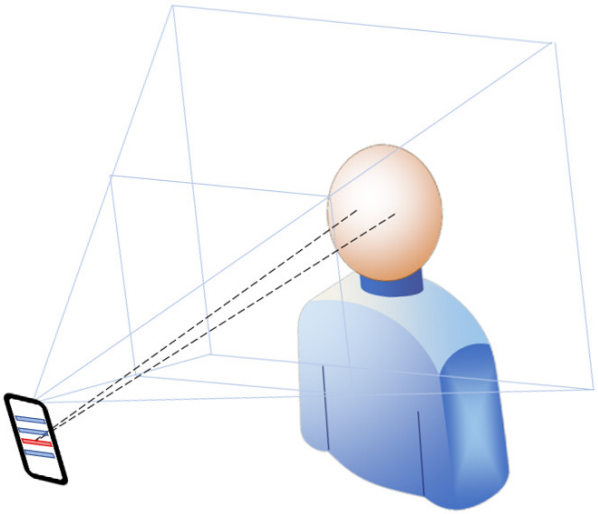

Figure 3A

Figure 3B

As illustrated in Figures 3A and 3B, the computing device is a smartphone. Figure 3A illustrates the smartphone's sensors, specifically a front-facing camera, measuring visual input relating to the user. The ML model identified which application notification the user was looking at, specifically the third notification, and depending on the user's level of gaze focus (*e.g.*, the duration of the gaze, whether the user's eyes are squinting), the ML model can direct the operating system to expand the notification. Alternatively, if the ML model identifies the activity of the user's eyes as quickly glancing over the different application notifications, then the device can batch dismiss the notifications once the user is done.

Figure 3B illustrates a user's hand shaking the computing device. A motion sensor (*e.g.*, an accelerometer) on the device can measure the user shaking the phone. The ML model can then dismiss all application notifications on the lock-screen of the computing device.

The techniques and methods described in this publication can be used to improve user interaction with application notifications on computing devices. The utilization of multiple sensors to measure user input on the lock-screen of computing devices can afford users convenient and quick methods to interact with application notifications. Machine-learned methods, such as gaze

detection, custom gestures, and/or user actions, permit users the ability to expand, individually dismiss, or batch dismiss application notifications exclusive of or in conjunction with application notification priority levels.

References:

[1] Suzanne Marion Beaumont. Method and Device for Notification Preview Dismissal. US Pub. 2016/0227107, filed February 2, 2015, and published August 4, 2016.

[2] Dzenan Dzemidzic, Erland George-Svahn, Rebecka Lannsjo, Ida Nilsson, Anders Olsson, Marten Skogo, and Anders Vennstrom. System for Gaze Interaction. US Pub. 2017/0090566, filed December 14, 2016, and published March 20, 2017.

[3] Jason Wu. Synchronous Interfaces for Wearable Computers. Master's thesis, Georgia Institute of Technology, 2018. Atlanta: Georgia Institute of Technology, 2018. 1-25.