January 2021

Grip Suppression using a Machine-Learned Technique

Katherine Kuan

Mark Chang

Grip Suppression using a Machine-Learned Technique

Abstract:

This publication describes techniques for grip suppression of a touchscreen display of a computing device using a machine-learned technique. When a user intentionally or unintentionally touches the display (e.g., with a hand or a finger), a Touch Manager of the computing device performs operations to determine a user intent associated with the touch input to prevent false triggering of the display (e.g., by a grip of a hand holding the device). A machine-learned (ML) model calculates the likeliness of an intentional touch input (e.g., a tap, a swipe, or a scroll of a hand or a finger to input or manage information on the device) by identifying and assigning weights to features of the touch input. A total weight is calculated for each touch input and compared to a default threshold (e.g., an accepted threshold associated with an intentional touch input), which may be adjusted to ensure accuracy of user-intent predictions. After the Touch Manager verifies the user-intent predictions using heuristic and/or hysteresis logic, the computing device may perform operations to suppress or enable the touch input.

Keywords:

Machine learning, ML model, deep learning model, convolutional neural network-based learning, grip suppression, phone grip, phone contact, touchscreen display, touch phone, unintentional touch, intended touch, touch input, user interface

Background:

Touchscreens on mobile phones are growing in popularity due to a desire from users to access and manage information on their mobile phone using touch input (e.g., of a finger or a hand). Furthermore, many users trigger their mobile phone to change from an inactive state to an active state with a touch input. For example, a user may tap their touchscreen with their finger to turn on a display of the mobile phone. The user may then access information including messages, search browsers, emails, settings, applications, and so forth by tapping, swiping, or scrolling with their fingers or hand. While touchscreens remain a desirable feature on mobile devices, there are many inherent challenges associated with touchscreens, including identification of unintentional touch inputs (e.g., grip contact of a hand), interference with intended user inputs, degradation of touchscreen performance, or blockage of intended touch inputs.

Description:

This publication describes techniques for grip suppression using a machine-learned technique on a computing device with a touchscreen display. While the example computing device described in this publication is a mobile phone, other types of computing devices can also support the techniques described in this publication.

A computing device may include one or more processors, transceivers for transmitting data to and receiving data from a base station (e.g., wireless access point, another computing device), sensors (e.g., a surface-capacitance sensor, a mutual-capacitance sensor, a self-capacitance sensor, a location sensor, a proximity sensor), a computer-readable medium (CRM), and/or an input/output device (e.g., a touchscreen display, a speaker, a microphone). The CRM may include 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. The CRM includes device data (e.g., user data, multimedia data, applications, and/or an operating system of the device), which are executable by the processor(s) to enable the techniques described herein.

The touchscreen display may include a layer on top of a light-emitting display panel of the computing device that includes a capacitive-touch-sensor array. The user may provide input to or control information on the computing device via the touchscreen display by providing a touch input (e.g., a single touch, a multi-touch, or a motion of a finger, multiple fingers, or hand across a surface of the touchscreen display). While the example computing device described in this publication uses a capacitive-touch-sensor array, other sensors for detecting touch input may support the techniques described herein. The other sensors may include resistive touchscreens, surface acoustic wave (SAW) technology, surface capacitance, projected capacitance, mutual capacitance, self-capacitance, infrared touchscreens, infrared acrylic projection, optical touchscreens, dispersive signal technology, and acoustic pulse recognition. All of these sensors may be used to determine the touch input of the display and produce heatmap images that identify regions touched by the user and corresponding temporal information. However, none of these sensors can determine a user intent associated with the touch input.

It is important to determine the user intent to prevent triggering false touch input. For example, the user may accidentally touch the display of their mobile phone when gripping a housing of the computing device with their hand. In this example, a portion of the palm or an edge of a finger may touch the display unintentionally. If an unintentional touch input triggers the mobile phone to change from the inactive to the active state, the user may then accidentally perform operations on their phone. For example, their palm may accidentally open a browser, or the edge of their finger may accidentally place a call.

To prevent unintentionally triggering the computing device, many devices designate a suppression zone around the touchscreen display. For example, the suppression zone of a mobile phone may include a portion of the touchscreen display (e.g., a border) that is near an edge of the phone where the user typically grips the device. Within the border of the display, the phone may suppress touch input from the user or delay registering the touch input until a motion or lift is detected. However, these devices face many challenges including a lack of flexibility to adjust to gestures and hand/finger sizes of a user. This challenge may degrade a performance of a zoom or rotate operation of the device or even block intentional gestures. Another challenge may include degradation of the latency and sensitivity of the touch input near the border in addition to a long-press gesture degradation near the border. For example, a user may need to click on an object located near the edge of their device to perform desired tasks. If their phone uses a designated suppression zone, their device may not detect intentional clicks from the user along the edge, causing frustration.

To resolve these challenges, the device data may include a Touch Manager. The computing device performs operations under direction of the Touch Manager to determine if input (e.g., a touch of a finger or hand) from a user on the touchscreen display of the computing device is intentional touch input or unintentional touch input. For example, an intentional touch input may include a tap, scroll, or swipe of the display with a finger or hand to input or manage information on the computing device, whereas an unintentional touch input may include a hand gripping the phone and accidentally making contact with the display, especially near edges or corners.

The Touch Manager utilizes a two-dimensional capacitive-touch-sensor array to generate a heatmap image and determine features of the touch input, as illustrated in Figure 1. The Touch Manager may not use a fixed suppression zone or delay reporting the touch input until the motion or lift is detected. Furthermore, the techniques described herein are applicable to a computing device in both a portrait (e.g., vertical orientation) and a landscape (e.g., horizontal orientation) mode. The Touch Manager first collects two-dimensional capacitive-touch-sensor array signals (e.g., the heatmap image) and pre-processes the heatmap image into a format that enables the techniques described herein. For example, the Touch Manager may select data from the heatmap image that is within a region of interest (ROI). Then, the data is analyzed, for example, using a Spatial Classifier and a Temporal Classifier, to determine the most likely intent of the user.

The techniques described herein utilize a machine-learned (ML) model that can run either on touch-controlled firmware (e.g., as part of a firmware running on a touch controller) on the CRM, as an extension of a touch driver on an application processor (e.g., a host or driver), or on a power-efficient system (e.g., using flash memory). The ML model may be a standard neural-network-based model with corresponding layers required for processing input features like fixed-side vectors, text embeddings, or variable-length sequences. The ML model may be implemented as one or more of a support-vector machine (SVM), a recurrent neural network (RNN), a convolutional neural network (CNN), a dense neural network (DNN), one or more heuristics, other machine-learning techniques, a combination thereof, and so forth. The ML model is trained to classify features of the touch input and best predict an intent of the user using the Spatial Classifier.

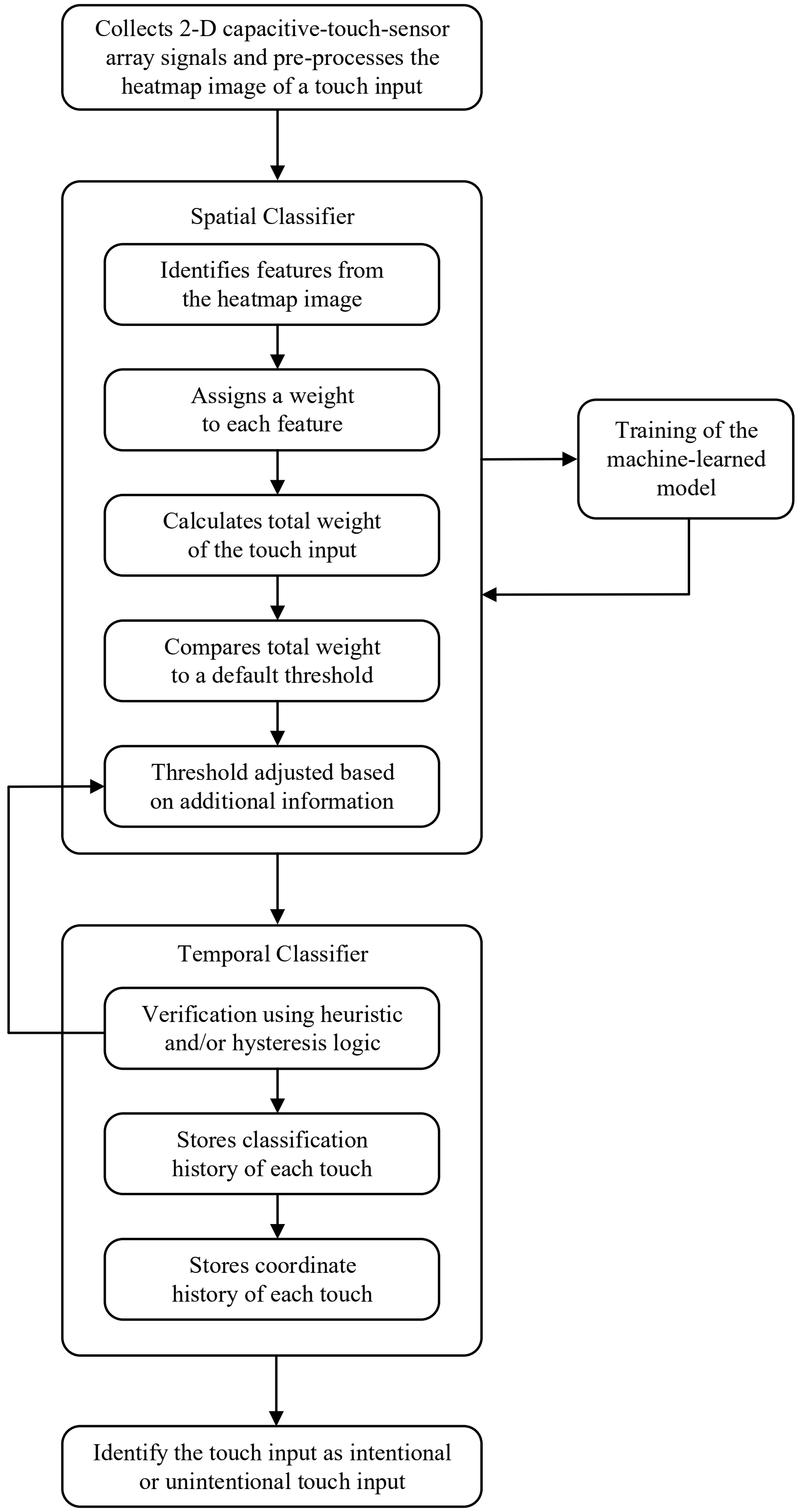
Collects 2-D capacitive-touch-sensor array signals and pre-processes the heatmap image of a touch input
Spatial Classifier
Identifies features from the heatmap image
Assigns a weight to each feature
Calculates total weight of the touch input
Compares total weight to a default threshold
Threshold adjusted based on additional information
Training of the machine-learned model
Temporal Classifier
Verification using heuristic and/or hysteresis logic
Stores classification history of each touch
Stores coordinate history of each touch
Identify the touch input as intentional or unintentional touch input

Figure 1

The Spatial Classifier first identifies features of the touch input from the heatmap image. These features may include normalized-edge slopes, percentage of edge amplitudes, geometry ratio of a length and width, an area, a normalized-peak amplitude, a distance to a closest corner, and temporal information of the touch input. Each feature is then assigned a weight based on a likeliness of an intentional touch input. The Spatial Classifier totals the weight of all features and compares that weight to a default threshold for an intentional touch input. For example, if a total weight is calculated at six (on a scale of 0–10, where 0 indicates the unintentional touch input and 10 indicates the intentional touch input), and a default threshold is set at five, then the touch input is likely intentional. However, the touch input is not yet classified as intentional. The Spatial Classifier may adjust the default threshold based on additional information to produce a best prediction of the intent of the user. The additional information may include a total number of touch inputs on the display, a number of touch inputs near an edge of the device, a number of touch inputs at a center of the display, results of a same touch input from previous frames, and a duration of the touch input.

The ML model uses an output of the Spatial Classifier as an input to the Temporal Classifier. The Temporal Classifier verifies the output (e.g., predicted result of user intent) using heuristic and/or hysteresis logic. For example, if a specific touch input is classified as intentional, the Temporal Classifier will ensure that every instance of that specific touch input is classified as intentional. If, at a later time, that specific touch input is classified as unintentional by the Spatial Classifier, the Temporal Classifier will identify and change a result to be consistent with precedent. The Temporal Classifier can also store a classification history of each touch input in addition to a coordinate history (e.g., history of pixel locations).

After the Temporal Classifier has verified the output of the Spatial Classifier, the ML model determines the intent of the user, and the computing device may perform operations in accordance. For example, if the ML model determines a touch input (e.g., in a region near the border) is intentional, the computing device may allow the user to provide information using that touch input within that region. If, instead, the touch input near the border is determined to be unintentional via the ML model, the computing device may suppress touch input to prevent false triggers and enable a positive user experience. In another example, if the user is gripping the device and their palm accidentally touches the display, the ML model may determine that the user does not intend to access the display of the device. Therefore, the device may suppress a grip of the user.

References:

[1] Patent Publication: US20160291731A1. Adaptive enclosure for a mobile computing device. Priority Date: December 24, 2013.

[2] Patent Publication: US20180121052A1. Classification of touch input as being unintended or intended. Priority Date: September 12, 2014.

[3] Patent Publication: US20170116545A1. Touch Classification. Priority Date: July 11, 2014.