

Emoji Tree: The Fastest and Most Personalized Emoji Input Software

<https://emoji.saranshgrover.com>

Saransh Grover

Stony Brook University, CSE 323

Abstract

Emojis are the most modern way of communication online. Using Emojis in text messages has become commonplace for a large population on the internet and emoji input programs are used as often as text input devices. All modern operating systems now come with an emoji input tool, and many social media and communication applications have their own ways of inputting emojis. This widespread adaption has only increased the popularity of emojis to being one of the most essential methods of communication on the internet. In this paper, I analyze the most popular and frequently used methods of emoji input with an HCI lens. I focus on three aspects of these programs: ease of use, accuracy and speed, and prediction. These 3 aspects are often the most important for text input, and thereby I believe these measures translate well the Emoji Inputs. After analyzing these aspects, I propose a new method of inputting emojis, as well as display a proof of concept for the method. We perform an informal evaluation and compare the results

Keywords: Emoji, Computer Interaction, Input, Typing

Introduction

As a frequent user of Emojis, I have often found myself searching through a certain emoji through my keyboards, eventually giving up and using some less than ideal result. While Unicode has a standard for Emojis, this standard does not reciprocate well with users since the Unicode emoji names can in no way reflect all the possible ways an emoji is interpreted. This problem makes finding emojis inconvenient. To make this worse, many emoji programs have poorly designed search for emoji lookups. This is especially true on non-touch devices, where usage of emojis is infrequent. In this paper, I will present a new emoji input software called Emoji Tree. Emoji Tree is an emoji input software for both touch and keyboard based devices alike. It takes on the developments of prior emoji input programs and adds new improvements and advancements. The 3 main features of Emoji Tree are:

- Remember Emoji Use and perform Accurate Predictions
- Make Emoji Input personalized to each user, with the option to add aliases
- Emoji Input works just like text input, and you never have to leave the flow of typing when entering emojis.

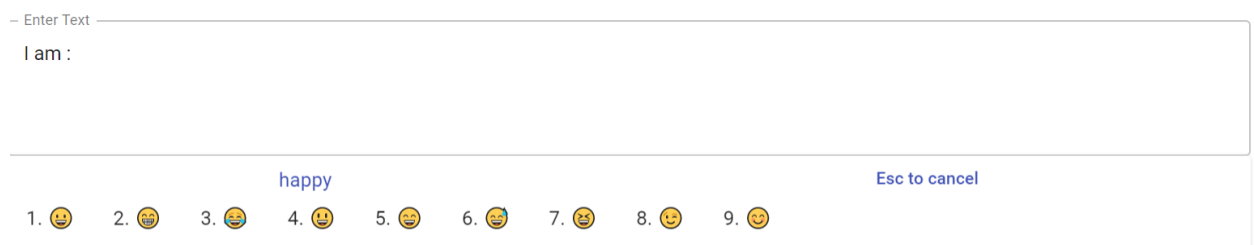


Figure 1: EmojiTree Proof Of Concept

Emoji Tree was developed after close analysis from 3 prominent emoji input mechanisms. In this paper, I will summarize the prior input mechanisms. First, I will look at the Google Keyboard, and see how a mobile device keyboard inputs emojis. Next, I will look at the Windows Emoji Keyboard focusing on how it uses the keyboard to allow for faster emoji input. Lastly, I will look at Discord studying how one of the largest communication platforms enhances user experience with their Emoji Input methods. I investigate how each of these methods approaches Emoji input and analyze the inconveniences and challenges that creates. Then, I will present reasons for the design of Emoji Tree, and how it improves upon the prior mechanisms. I perform an informal within-subject evaluation with 4 subjects to conclude and compare differences

Existing Emoji Input Software

The 3 main aspects of any emoji input software are very similar to that of a text input software: ease of use, accuracy and speed, and prediction. The 3 existing emoji input software that I will discuss each take a unique approach at providing for these aspects. Some, like the Google Keyboard, focus solely on mobile and touch devices with extreme accuracy and speed, while others like Discord's emoji input focuses on ease of use across device types, and offers a compelling alternative to google keyboard.



Figure 2: Google Emoji Keyboard

Ease of Use

The google keyboard offers a common way of emoji input on mobile devices, as a screen on the virtual keyboard that is accessible by pressing on one the emoji “key.” This is one of the most common ways that mobile devices provide to input emojis. On mobile devices, emoji input on google keyboard is extremely easy to use. It is baked into the virtual keyboard, which means that searching for emojis is as easy as typing a letter. There are no hotkeys involved, and the process for a user to switch to a list of emojis is simple and convenient. However, the method is not without its flaws. Since Google Keyboard is made for mobile devices, it does not translate well to desktops and PCs. In fact, such a mechanism becomes extremely painful to iterate through when using with a keyboard. The primary reason for this is because for most users, desktop keyboards act as a physical input that they can feel, while the monitor screen acts as the response to the physical input, thereby confirming the action the user took. However, with Emoji Inputs like this, users need to look at the screen to perform emoji entry, which drastically reduces the speed at which they can send such messages.

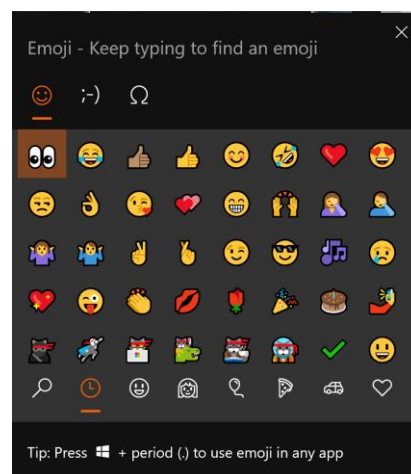


Figure 3: Windows Emoji Keyboard

On the other hand, the Windows Emoji Keyboard fixes this problem by letting users type to search for emojis. In order to open the keyboard, a user needs to press a hotkey (win + .).

Then, the user can simply type for the emoji they want, and the keyboard filters the list of emojis based on the query. In this case the ease of use on desktop devices is much higher because of the searching availability. By offering a mechanism to filter emojis by simply typing, the Windows Emoji Keyboard lets users at least partially continue the flow of typing text while also typing emojis.

Accuracy and Speed

Another big factor affecting Emoji input on google keyboard is the extensive effort required in searching. While Google Keyboard has good prediction, it shows emojis in categories as humungous lists, which are extremely hard to search through. The categories that are used to divide these emojis are not helpful since they do not have much correlation and lack any form of connection. Therefore, many users often resort to a handful of emojis that they use, which show up in the “Recent” section. Any emoji that has not been recently used takes an extremely long time to find and is a huge bottleneck for the entire process. On the Discord Emoji Input, there are no categories of Emoji input, rather, the only way of selecting an emoji is by searching for it. This results in faster speed compared to the Google Keyboard, however with the caveat of lower accuracy. Since the Google Keyboard shows all emojis as their corresponding pictorial representations, it is much more accurate. Discord has codes for each emoji that do not necessarily represent all ways that emoji can be used. Therefore, for a user to find an emoji, they need to remember some part of the emoji code at least partially. For example, searching “happy” on the discord emoji tool returns nothing, even though there are dozens of emojis that represent the “happy” feeling. This leads us to the issue with emoji input via keyboards: Emojis are used in dozens of different contexts and using a single phrase to represent an emoji will result in many

users unable to find the emoji. Categorizing them is one solution, but as we have seen, that drastically reduces the speed.

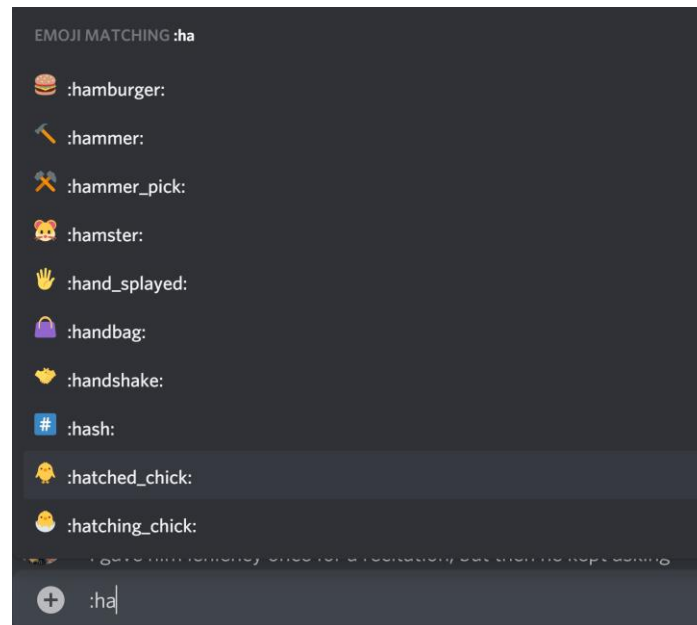


Figure 4: Discord Emoji Input

Prediction

Both Google and Windows offer one form of prediction: the recently used tab. Besides this, none of the three emoji input mechanisms I have mentioned perform any form of prediction. Without prediction, emoji input programs lack one of the key reasons text input is so fast. Windows does offer some query-based filtering that can serve as prediction. For example, searching for the term “happy” gives you a list of emojis that correspond to the feeling of being happy. However, this is nowhere near the level of prediction that text input performs.

Emoji Tree: Accurate and Personalized Emoji Input

Emoji Tree tries to solve many of the problems I have stated with existing emoji input mechanisms. It is designed with all kinds of devices in mind, which means it works well both on

Keyboard and on touch-based devices. The proof of concept has been made for keyboard devices; however, a similar design can be made for mobile devices as well.

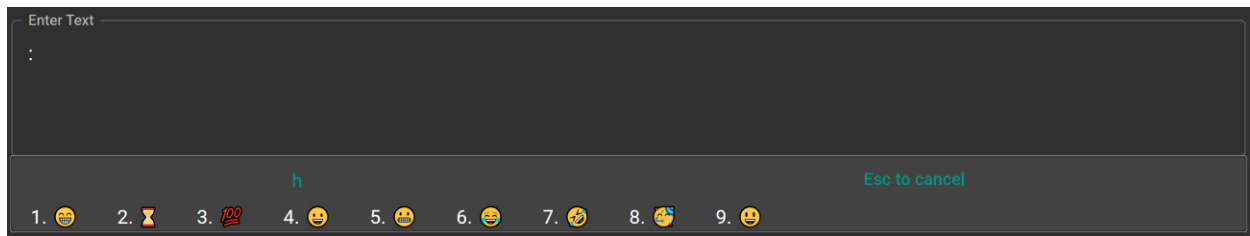


Figure 5: Emoji Tree Design

Emoji Tree is designed with core HCI principles in mind. As soon as the user types a colon (:), the Emoji Input modal pops up right below the text field. The user can now start typing their query which shows up on the top of the modal. On the bottom, there are at most 9 emojis that are placed in a vertical grid. These emojis are filtered through an algorithm that checks for the frequency of the emoji usage by the user, the recency as well as the past selections the user made with the given query. Instead of moving through these 9 emojis to select an emoji, the user simply needs to type the corresponding number on their keyboard. Once the user presses a number, the modal disappears, and the colon gets replaced with the emoji. The user can now continue typing. This is extremely important as we never disrupt the flow of typing when selecting an emoji. The user can continue to use their keyboard to not only search for the emoji, but also select it.

Unlike discord, Emoji Tree does not filter based on just one description. Each Emoji has a large array of keywords that have been collection from common use of the emojis through Emoji Lib, Twemoji, Emojipedia and the GitHub Emoji API. When a user searches, their query is searched through all keywords of an emoji. Thus, searching “happy” on Emoji Tree returns emojis that make sense. However, Emoji Tree cannot possibly cover all contexts an emoji is used in. After all, each user may have their own personal use for an emoji. Therefore, Emoji Tree

gives users the ability to add keywords for each emoji. As a user, I can add any keyword to an emoji once, and then search for my emoji using that keyword forever. This not only makes Emoji Tree extremely fast, but it also makes it personalized to each user, thereby increasing the accuracy of getting the right emoji when the user makes a query. Users can add keywords by clicking on “Add Aliases for Emojis” and then select any emoji that they would like to add a keyword to. All this information persists in the user’s browser, and they only have to do so once.

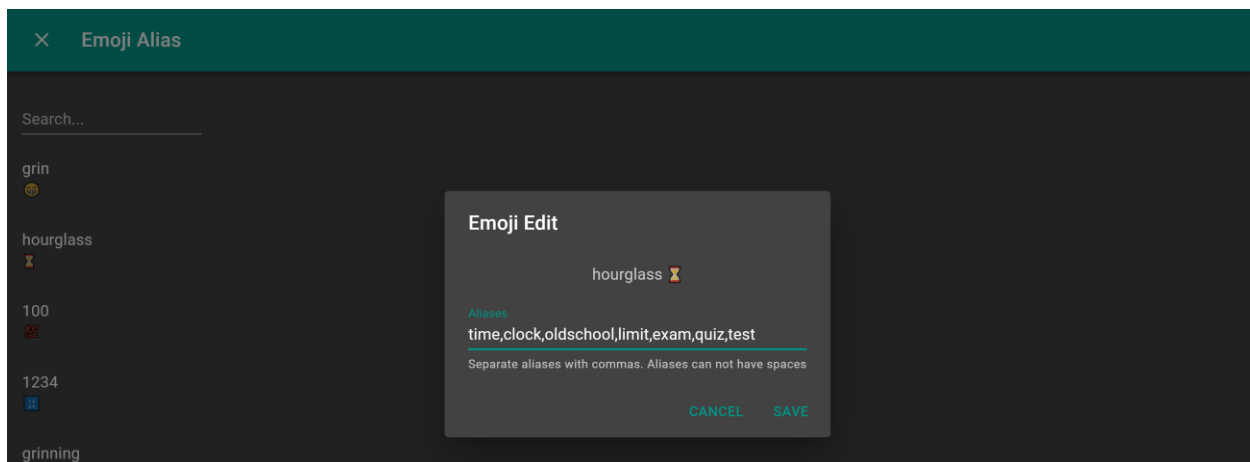


Figure 6: Emoji Keyword Addition

Evaluation

Design & Procedure

I performed a within-subject evaluation analyzing the effectiveness of Emoji Tree. For comparison, I chose the Discord Emoji Keyboard and the Google Keyboard. I recruited 6 participants (all male, ages 18-21). Note that all users had prior experience with the Google and Discord Keyboards, but none of them had experience with Emoji Tree. The evaluation was challenging to do since it is not feasible to ask each user to test searching for each emoji. I eventually decided to give the users five unique emojis and asked them to search for those using each of the input mechanisms while timing themselves. This evaluation compares the speed and accuracy at which participants are able to find emojis. Then, I asked each participant to find their

favorite emoji on each method. This evaluation compares the prediction of each method. Finally, each participant was asked to rate the ease of use for each method and answer some questions regarding Emoji Tree.

Results & Discussion









Emoji	Discord (μ)	Google (μ)	Emoji Tree (μ)
	12.13	9.22	11.61
	6.13	5.03	3.13
	26.43	6.80	3.03
	13.12	4.55	5.28
	3.49	7.12	4.22
Favorite	6.22	3.19	4.00

Table 1: Mean of participants times for each emoji

Looking at the mean of each emoji result, we can see that Emoji Tree performs at least as good as the other methods, if not better. Emoji Tree performs better than Google Keyboard and Discord Keyboard on 4/5 emojis. Emoji keyboard performs extremely well in less common emojis, like  (sweat_drops) and  (sweat_smile). This is most certainly because of how Emoji Tree uses keywords to search for Emojis, and therefore lowers the chance of making a mistake. Emoji Tree was extremely slow on the first emoji  (heart_eyes) which is a much more common and frequently used emoji. One explanation for this is that the participants were not aware of how Emoji Tree works and had trouble looking up the first emoji. The mean time for Emoji Tree inputs also proves this hypothesis since the first time is an extremely large outlier. One way to fix this would be to give users a precise tutorial before they begin using Emoji Tree, and such a tutorial should be considered for future versions.

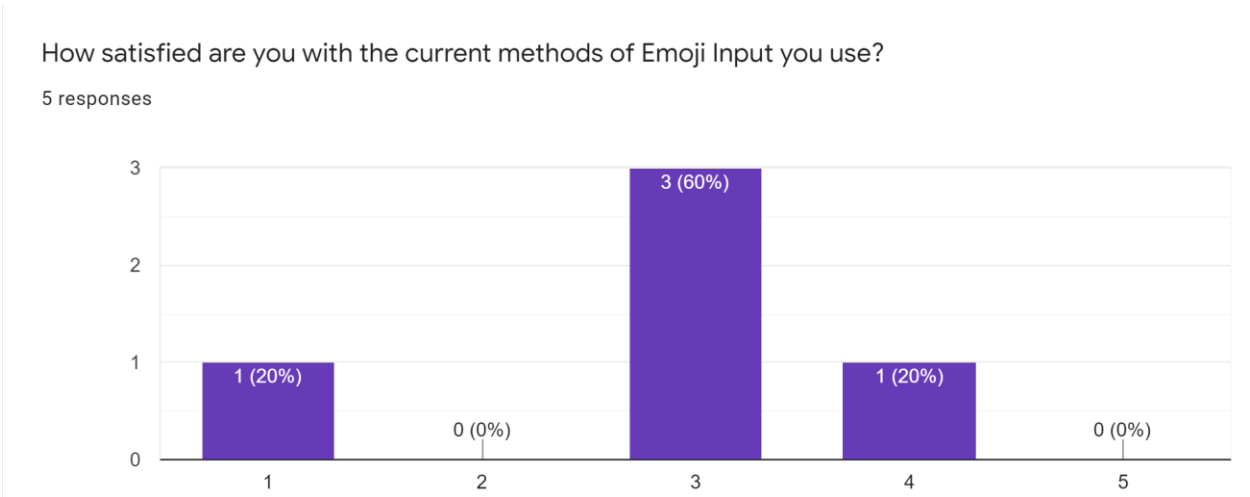


Table 2: How satisfied are you with the current methods of Emoji Input you use, rated on a 5-point Likert scale

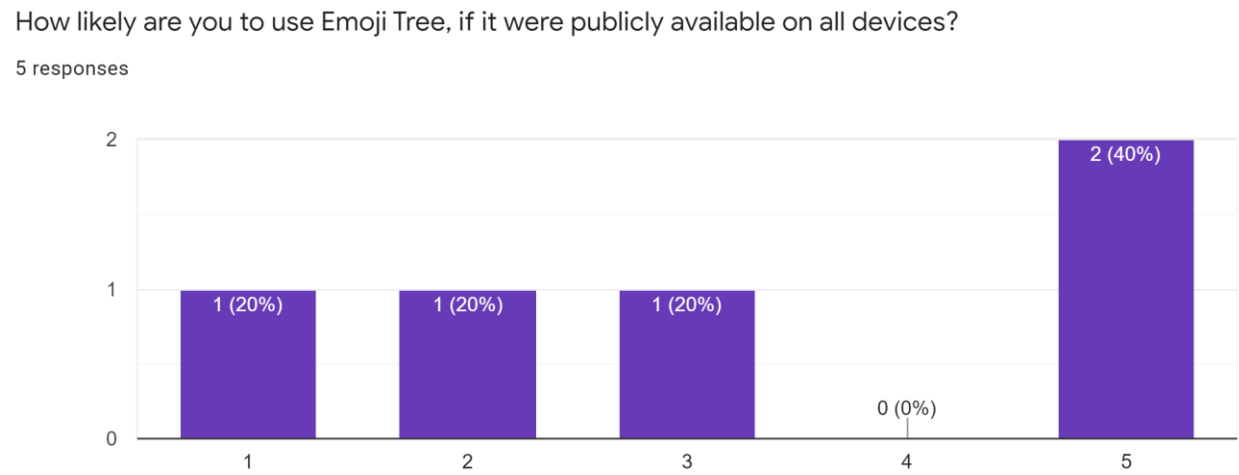


Table 3: How likely are you to use Emoji Tree, if it were publicly available on all devices, rated on a 5-point Likert scale

Emoji Tree was easier to use than other keyboards

5 responses

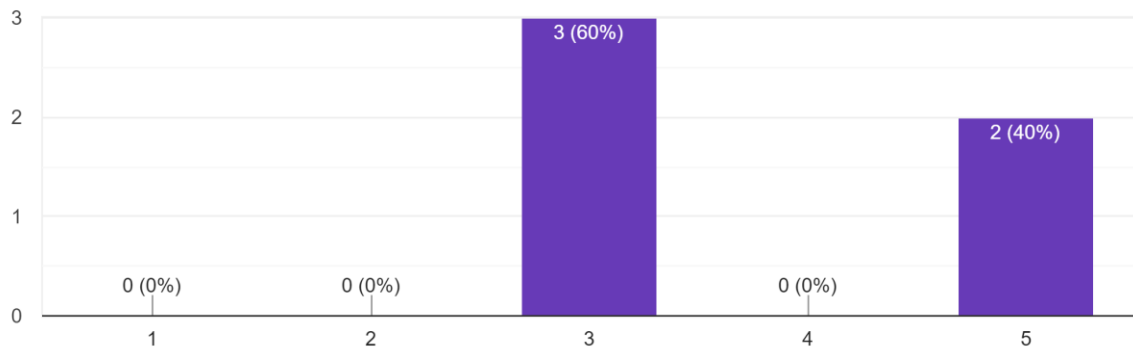


Table 4: Emoji Tree was easier to use than other keyboard rated on a 5-point Likert scale

Conclusion

Responses from the questions clearly seem to state that the satisfaction level with Emoji Input devices is lacking, and that Emoji Tree has an above-average performance when compared to Google and Discord Keyboards. I am satisfied with this response, as Emoji Tree is still in initial stages and there are some more developments that need to be done. One feature that I have gotten feedback on is the ability to analyze mistakes that users make. For example, if a user searches for “play,” but then does not select anything and then searches for “doll” and selects 🧸 (dolls), “play” should be added as a keyword to the dolls emoji. Such error fixes will lead in much higher accuracy and user satisfaction and will mean that Emoji Tree will learn how specific users interact with their emojis. Overall, I am very satisfied with the state of Emoji Tree. I believe that Emoji Tree offers a unique experience of Emoji Input that is fast, accurate and personalized to each user and flows extremely well with keyboard input. Such a mechanism is lacking in the current state of Emoji Software, and I will continue to work on developing this and making it publicly available in more accessible and usable formats.

References

- Chisholm, B., Hachey, H. B., & Radford, W. (2016, November). “Call me Ishmael” – How do you translate emoji? *Hugo.ai*. Retrieved from <https://arxiv.org/pdf/1611.02027.pdf>
- Full Emoji List*. (2020). Retrieved from <https://unicode.org/emoji/charts/full-emoji-list.html>
- Github Emoji API*. (n.d.). Retrieved from Github: <https://developer.github.com/v3/emojis/>
- Hancock, J. T., Silver, C., & Landrigan, C. (2007). Expressing emotion in text-based communication. *CHI 07*. doi:1240624.1240764
- Pohl, H., Stanke, D., & Rohs, M. (2016). EmojiZoom: Emoji Entry via Large Overview Maps. doi:2935334.2935382
- Sarah, W., & Sandy, G. J. (2018). Repurposing Emoji for Personalised Communication: Why 🍕 means “I love you”. *CHI 2018*. doi:3173574.3173726
- Twemoji*. (n.d.). Retrieved from <https://twemoji.twitter.com/>
- Wijeratne, S., Balasuriya, L., & Sheth, A. (n.d.). . A Semantics-Based Measure of Emoji Similarity. doi:3106426.3106490