Response to Reviewer 1

Strengths:

Comment 1-1: The paper is overall well-written, with solid analysis and contribution.

Response 1-1: Thanks for your appreciation.

Weaknesses:

Comment 1-2: Offloading the task to the edge seems little to no benefit but has negative implications.

Response 1-2: Thanks for your comment. In the following, we will first illustrate the benefits of our system design (*i.e.*, prediction with the aid of edge computing) with respect to prediction accuracy and response latency. Next, we will show an example to justify that the negative implication (*i.e.*, security/privacy issues) can be solved with the integration of our proposed scheme with the existing techniques (*e.g.*, Rivest-Shamir-Adleman (RSA) for encryption and decryption from "A performance analysis of DES and RSA cryptography" by Singh Sombir *et al.* published in *International Journal of Emerging Trends & Technology in Computer Science,* vol. 2, no. 3, pp. 418–423 in 2013).

First, regarding prediction accuracy, since edge has more powerful computing resources than smartphone, the smartphone under our system design can leverage more powerful neural models on the edge to provide more accurate results than local prediction.

Second, as for response latency, we compare our system design with local prediction and cloud-assisted prediction, respectively.

- The comparison between our system design and local prediction. The existing study (from "Powers of 10: Time Scales in User Experience" by Nielsen *et al.* in *Nielsen Norman Group*) shows that "0.1 second is the response time limit if you want users to feel like their actions are directly causing something to happen on the screen". In this way, the designed system needs to provide emoji prediction faster than 0.1 seconds. Under our system design (*i.e.*, edge-assisted prediction), the response latency can be less than tens of milliseconds (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017) which is in the range of the response time limit. Consequently, compared with local prediction, although the wireless communication between edge and the smartphone brings extra latency, the latency do not influence the user experience.
- The comparison between our system design and cloud-assisted prediction. Compared with cloud computing, edge computing brings computing resources closer to the end-user to provide lower latency and higher bandwidth communication (from "Latency comparison of cloud datacenters and edge servers" by Charyyev *et al.* in *Proceedings of IEEE GLOBECOM* 2020). In this way, our system design can provide emoji prediction services of lower latency than the existing cloud-assisted prediction (*e.g.*,

Sogou Keyboard from "Smartphone background activities in the wild: origin, energy Drain, and optimization" by Chen *et al.* in *Proceedings of MobiCom* 2015).

Third, we show an example to justify that the integration of our proposed scheme with the existing techniques of ensuring security and privacy, *e.g.*, Rivest-Shamir-Adleman (RSA) for encryption and decryption (from "A performance analysis of DES and RSA cryptography" by Singh Sombir *et al.* published in *International Journal of Emerging Trends & Technology in Computer Science,* vol. 2, no. 3, pp. 418–423 in 2013), is a straightforward and promising way. The following is the workflow while applying encryption and decryption technologies.

1) The user's input is encrypted as ciphertext by the smartphone.
2) The smartphone selects models on the edge servers and sends the ciphertext to each selected edge server.
3) Each selected edge server decrypts the received ciphertext as plaintext.
4) The model on each selected edge server conducts the inference process given the input of plaintext.
5) With the inference process completed, each selected model outputs a predicted emoji.
6) The predicted emoji of each selected model is encrypted as ciphertext by the corresponding edge server.
7) Each selected edge server sends back the ciphertext to the smartphone.
8) The smartphone decrypts the received ciphertext as emojis and presents the emojis to the user.
9) The user selects the desired predicted emoji.

In summary, our proposed scheme has benefits of prediction accuracy and response latency. Moreover, although the security and privacy issues are not considered in our work, our proposed scheme can still work while applying techniques of ensuring security and privacy issues.

Comment 1-3: The authors say security is not their focus but it should be properly discussed. With local prediction there is no security/privacy concern at all, but with edge intelligence there is a whole lot of concern.

Response 1-3: Thanks for your comment. In fact, although the security and privacy issues are not considered in our work, the integration of our proposed scheme with the existing techniques of ensuring security and privacy, *e.g.*, Rivest-Shamir-Adleman (RSA) for encryption and decryption (from "A performance analysis of DES and RSA cryptography" by Singh Sombir *et al.* published in *International Journal of Emerging Trends & Technology in Computer Science,* vol. 2, no. 3, pp. 418–423 in 2013), is a straightforward and promising way. Here we show an example of workflow while applying encryption and decryption technologies to avoid security and privacy issues.

10) The user's input is encrypted as ciphertext by the smartphone.
11) The smartphone selects models on the edge servers and sends the ciphertext to each selected edge server.
12) Each selected edge server decrypts the received ciphertext as plaintext.

13) The model on each selected edge server conducts the inference process given the input of plaintext.
14) With the inference process completed, each selected model outputs a predicted emoji.
15) The predicted emoji of each selected model is encrypted as ciphertext by the corresponding edge server.
16) Each selected edge server sends back the ciphertext to the smartphone.
17) The smartphone decrypts the received ciphertext as emojis and presents the emojis to the user.
18) The user selects the desired predicted emoji.

In summary, our proposed scheme can still work while applying techniques of ensuring security and privacy issues.

Comment 1-4: On the other hand, utilizing the edge to deploy "more powerful models" does not have demonstrated improvement over existing local on-device models, as no comparison is done to the Samsung Keyboard or other methods in the literature.

Response 1-4: Thanks for your comment. The evaluations of comparisons between our proposed method (*i.e.,* edge-assisted prediction) and existing methods (*i.e.,* local prediction and cloud-assisted prediction) in real scenarios are important and can be our future work. In this work, we focus on the modelling and analysis of mobile keyboard emoji prediction with the aid of edge computing, which provides insights for the deployment in real scenarios in the future.

Comment 1-5: Simply saying that "compressed models are not accuracy" is not convincing given the extensive machine learning literature on model compression.

Response 1-5: Thanks for your comment. The state-of-the-art (SOTA) benchmarks (from "Emoji prediction: Extensions and benchmarking" by Ma *et al.* in *Proceedings of EMNLP* 2020) show that some existing neural models (*e.g.*, BERT from "Bert: Pre-training of deep bidirectional transformers for language understanding" by Devlin *et al.* submitted to *arXiv preprint arXiv: 1810.04805* in 2018) largely outperform the other existing architectures. However, due to constrained on-device memory capacity and response latency (from "On-device neural language model based word prediction" by Yu *et al.* in *Proceedings of IEEE COLING* 2018), such models cannot be deployed directly on the smartphone. In this way, a series of recent researches in industry about mobile emoji prediction consider compressing the model, *e.g.*, Samsung Keyboard (from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021). Such a compression way reduces the parameters which would influence the expressive capacity of the model, incurring an insufficient prediction accuracy compared with uncompressed model (*e.g.*, BERT from "Bert: Pre-training of deep bidirectional transformers for language understanding" by Devlin *et al.* submitted to *arXiv preprint arXiv: 1810.04805* in 2018).

Comment 1-6: There is also the issue of communications in edge computing that is not considered at all. Edge communication incurs extra delay, is unstable due to wireless, and has limited bandwidth. A simple task like emoji prediction offloaded to the edge seems an overkill and a waste of the scarce edge bandwidth.

Response 1-6: Thanks for your comment. In the following, we first illustrate the motivation and impact of mobile keyboard emoji prediction to show that emoji prediction is a meaningful task. Next, we justify that offloading the emoji prediction tasks to the edge is not an overkill and a waste of the scarce edge bandwidth.

First, we illustrate the motivation and impact of mobile keyboard emoji prediction. In recent years, emojis have become one of the primary elements of instant messaging (from "Incorporating emoji descriptions improves tweet classification" by Singh *et al.* in *Proceedings of NAACL* 2019). In particular, emojis are usually highly abstractive, leading to more stylistic features and richer contexts than plain-text messages. To improve the user experience in terms of input efficiency, a series of recent researches about mobile emoji prediction in industry (*e.g.*, Gboard from "Federated learning for emoji prediction in a mobile keyboard" by Ramaswamy *et al.* submitted to *arXiv preprint arXiv:1906.04329* in 2019, Samsung Keyboard from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021) have emerged out.

Second, the existing keyboard, *i.e.*, Sogou Keyboard (from "Smartphone background activities in the wild: origin, energy Drain, and optimization" by Chen *et al.* in *Proceedings of MobiCom* 2015), leverages cloud-assisted prediction, *i.e.*, prediction using models deployed on the smartphone and the cloud. It offloads the emoji prediction task on the cloud and has the advantage of high prediction accuracy since the smartphone leverages powerful neural models on the cloud to conduct emoji prediction. However, under such a deployment scheme, the unstable network conditions affect the efficiency of wireless transmission between the smartphone and the cloud which incurs a high response latency. Consequently, it deteriorates the user experience of the emoji prediction services. Based on the above considerations, we considering conducting emoji prediction with models deployed on the smartphone and the edge. Compared with cloud computing, edge computing brings computing resources closer to the end-user to provide lower latency and higher bandwidth communication (from "Latency comparison of cloud datacenters and edge servers" by Charyyev *et al.* in *Proceedings of IEEE GLOBECOM* 2020). Meanwhile, edge computing has advantage of saving energy for smartphone when providing emoji prediction services (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017). In this way, under our system design, the smartphone can provide emoji prediction services with lower latency and lower energy consumption than the existing cloud-assisted prediction.

Comment 1-7: Given the 20ms bound on response latency as the authors point out, how much slack can be left to computing at edge given a communication latency of perhaps 10ms or even more?

Response 1-7: Thanks for your comment. The research about edge computing (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017) shows that with the aid of edge computing, the response latency in our system design can be less than tens of milliseconds. Specifically, the smartphone with the aid of edge computing can support the low-latency services, whose requirement is not smaller than 17ms (from "Mobile edge computing and field trial results for 5G low latency scenario" by Zhang *et al.* published in *China Communication*, vol. 13, no. 2, pp. 174–182 in 2016). In this way, our designed system is applicable to emoji prediction whose response latency is limited to 20ms.

Comment 1-8: There is no evaluation on how the edge latency/bandwidth affects the edge intelligence.

Response 1-8: Thanks for your comment. The evaluations on how the edge latency/bandwidth affects the edge intelligence in real scenarios are important and can be our future work. In this work, we focus on the modelling and analysis of mobile keyboard emoji prediction with the aid of edge computing, which provides insights for the deployment in real scenarios in the future.

Comment 1-9: Likewise, the energy consumption for wireless communication is not considered despite the work claiming "energy-awareness".

Response 1-9: Thanks for your comment. First, we consider the energy consumption of the smartphone accessing prediction models in **Section 3.5**, including local energy consumption and transmission energy consumption due to wireless communication. Specifically, for the prediction model on the smartphone, the energy consumption on the smartphone is caused by local computing (*i.e.*, local energy consumption), while for each prediction model on the edge server, the energy consumption on the smartphone is mainly caused by transmission (*i.e.*, the communication energy consumption of the smartphone sending the user's input to the prediction model).

Second, our target is not to minimize the energy consumption on the smartphone. Instead, our work focuses on improving user experience (including user satisfaction and response latency) via model selection given the energy budget. If the user has a high demand for energy consumption, *i.e.*, a low energy budget, our algorithm would make the smartphone select the models with low energy consumptions to constrain the energy.

Comment 1-10: How is the gain of predicting a little bit more accurate what a user wants to input compared to the loss of energy for transmitting everything to the edge and waiting for the result?

Response 1-10: Thanks for your comment. In the following, we will demonstrate the influences of response latency and energy consumption to the user experience, respectively, while conducting emoji prediction with the aid of edge computing.

First, regarding response latency, the existing study (from "Powers of 10: Time Scales in User Experience" by Nielsen *et al.* in *Nielsen Norman Group*) shows that "0.1 second is the response time limit if you want users to feel like their actions are directly causing something to happen on the screen". In this way, we need to design a system to provide emoji prediction faster than 0.1 seconds. Under our system design (*i.e.*, edge-assisted prediction), the response latency can be less than tens of milliseconds (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017) that will not influence the user experience. However, for cloud-assisted prediction, the response latency is larger than 100 milliseconds (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017) that deteriorates the user experience. Consequently, compared with local prediction, although the wireless communication between the edge and the smartphone brings extra latency, the latency does not influence the user experience. Meanwhile, compared with cloud-assisted prediction, our system design (*i.e.*, edge-assisted prediction) can largely improve user experience in terms of response latency.

Second, as for energy consumption, compared with existing cloud-assisted emoji prediction (*e.g.*, Sogou Keyboard from "Smartphone background activities in the wild: origin, energy Drain, and optimization" by Chen *et al.* in *Proceedings of MobiCom* 2015), edge computing has the advantage of saving energy for the smartphone when providing emoji prediction services (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017). Moreover, our target is not to minimize the energy consumption on smartphone. Instead, our work focuses on improving user experience (including user satisfaction and response latency) via model selection given the energy budget. In this way, users can set any appropriate energy budget as they want. Consequently, the energy consumption on smartphone will not influence the user experience (including user satisfaction and response latency).

Comment 1-11: Overall, I feel the paper's theoretical aspect is strong, but the motivation is not well justified. Perhaps any other task (like video analytics or whatsoever) can be more suitable for emoji prediction given the little impact such a task can make versus the non-negligible overhead for edge offloading.

Response 1-11: Thanks for your comment. In the following, we first illustrate the motivation and impact of mobile keyboard emoji prediction to show that emoji prediction is a meaningful task. Then we will demonstrate the motivation of the need for mobile emoji prediction with the aid of edge computing. Next, we justify that our system design will not lead to non-negligible overhead for edge offloading in terms of latency and energy consumption.

First, we illustrate the motivation and impact of mobile keyboard emoji prediction. In recent years, emojis have become one of the primary elements of instant messaging (from "Incorporating emoji descriptions improves tweet classification" by Singh *et al.* in *Proceedings of NAACL* 2019). In particular, emojis are usually highly abstractive, leading to more stylistic features and richer contexts than plain-text messages. To improve the user experience in terms

of input efficiency, a series of recent researches about mobile emoji prediction in industry (*e.g.*, Gboard from "Federated learning for emoji prediction in a mobile keyboard" by Ramaswamy *et al.* submitted to *arXiv preprint arXiv:1906.04329* in 2019, Samsung Keyboard from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021) have emerged out.

Second, we demonstrate a convincing case to motivate the need for mobile keyboard emoji prediction with edge computing. In this case, one user utilizes a mobile keyboard to send instant messages to his/her friends with plain-text and emojis. Such a mobile keyboard can provide emoji prediction services. Suppose that the user is active in an environment with unstable network conditions. In the following, we compare three ways for mobile keyboard emoji prediction.

- Local prediction, *i.e.*, prediction using the model deployed on the smartphone. It has the advantage of low response latency regardless of network conditions. However, due to the constrained on-device memory capacity (*e.g.*, 10MB from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021) and response latency (*e.g.*, 20ms from "Federated learning for mobile keyboard prediction" by Hard *et al.* submitted to *arXiv preprint arXiv:1906.04329* in 2019) of the keyboards on the smartphones, it is infeasible to deploy sophisticated models of high expressive capacity with demanding resource requirements (*e.g.*, BERT from "Bert: Pre-training of deep bidirectional transformers for language understanding" by Devlin *et al.* submitted to *arXiv preprint arXiv: 1810.04805* in 2018 requires huge memory capacity and incurs high response latency for prediction) on the smartphone. In this way, the keyboard applications (*e.g.*, Samsung Keyboard from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021) are deployed on smartphones by utilizing models with limited expressive capacity, such as compressed neural models. Such a deployment scheme would incur insufficient prediction accuracy.
- Cloud-assisted prediction, *i.e.*, prediction using models deployed on the smartphone and the cloud (*e.g.*, Sogou Keyboard from "Smartphone background activities in the wild: origin, energy Drain, and optimization" by Chen *et al.* in *Proceedings of MobiCom* 2015). It has the advantage of high prediction accuracy since the smartphone leverages powerful neural models on the cloud to conduct emoji prediction. However, under such a deployment scheme, the unstable network conditions affect the efficiency of wireless transmission between the smartphone and the cloud which incurs a high response latency. Consequently, it deteriorates the user experience of the emoji prediction services.
- Edge-assisted prediction, *i.e.*, prediction using models deployed on the smartphone and multiple edge servers (*our solution in this manuscript*). Suppose that the pre-trained prediction models with high performances can be deployed on heterogeneous edge servers in proximity to the user. In this way, smartphone can leverage such models to provide emoji prediction services in an accurate and responsive fashion.

Based on the above comparisons among three ways for mobile keyboard emoji prediction, edge-assisted prediction is promising to improve user experience in terms of prediction accuracy and response latency. Consequently, how to improve the prediction performances under edge-assisted prediction is an essential problem, which is the focus of our work.

Third, we justify that our system design will not lead to non-negligible overhead for edge offloading in terms of latency and energy consumption. Compared with cloud computing, edge computing brings computing resources closer to the end-user to provide lower latency and higher bandwidth communication (from "Latency comparison of cloud datacenters and edge servers" by Charyyev *et al.* in *Proceedings of IEEE GLOBECOM* 2020). Meanwhile, edge computing has the advantage of saving energy for the smartphone when providing emoji prediction services (from "A survey on mobile edge computing: The communication perspective" by Mao *et al.* published in *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358 in 2017). In this way, under our system design, the smartphone can provide emoji prediction services with lower latency and lower energy consumption than cloud-assisted prediction.

Response to Reviewer 2

Comment 2-1: The authors mentioned that "Moreover, we assume all the prediction models have been pre-trained." This assumption is too strong, and not timely.

Response 2-1: Thanks for your comment. In the following, we will elaborate that such an assumption is reasonable and the prediction is timely. Specifically, we focus on how to optimize the *inference* process for emoji prediction instead of the *training* process. Before the inference process, the models are pre-trained on other servers. For a given input, the pre-trained model would conduct the inference process and timely output the prediction.

In our scenario, with the aid of edge computing, the pre-trained prediction models with high performances can be deployed on heterogeneous edge servers so that the smartphone can leverage such models to conduct the *inference* process for providing emoji prediction services in a more accurate and responsive way. Notably, such models will not be redeployed in the future.

With the aid of models deployed across distributed edge servers, the smartphone and edge servers process the workflow shown in **Fig.1**. In particular, the smartphone sends the user's input to each selected model. Each selected model (on smartphone or edge server) conducts the *inference* process and outputs a predicted emoji, which is timely sent back to the smartphone. The user clicks on one desired emoji among the aggregated emojis on the smartphone.

Comment 2-2: There are many studies about the use of federated learning in virtual keyboard applications, including the Google's one. The federated learning based approaches can provide an evolutionary approach by aggregating the knowledge from multiple devices while protecting user privacy. So, it is difficult to understand the novelty of this paper.

Response 2-2: Thanks for your comment. Most existing works on emoji prediction focus on how to design a suitable architecture to optimize the performances of model *training* (*e.g.*, Gboard from "Federated learning for emoji prediction in a mobile keyboard" by Ramaswamy *et al.* submitted to *arXiv preprint arXiv:1906.04329* in 2019, Samsung Keyboard from "Voicemoji: A novel on-device pipeline for seamless emoji insertion in dictation" by Kumar *et al.* in *Proceedings of IEEE INDICON* 2021). However, our work focuses on how to optimize the *inference* process for emoji prediction. In the following, we first justify the differences between training and inference. Next, we elaborate the novelty of our work.

First, we justify the differences between training and inference.

- *Training*. During the *training* process, a known dataset is put through an *untrained* neural network. The framework's results are compared with the known results of such a dataset. Then the framework re-evaluates the error value and updates the weight of the layers in the neural network based on how correct or incorrect the value is. For example, if we want to train a neural model to distinguish the orange and apple, it is required to collect the images of oranges and apples as a training set where each image is labeled with either "orange" or "apple". In the beginning, it is possible that half of

the images of apples are mistaken for oranges. With the neural network optimizing its parameters, the results of mistaken images are corrected. Such a process is known as *training*.

- *Inference*. During the *inference* process, it is not required to re-evaluate or adjust the layers of the neural network based on the results. Inference applies knowledge from a *trained* neural network model and uses it to infer a result. In this way, when a new unknown dataset is input through a trained neural network, it outputs the prediction results based on the predictive accuracy of the neural network. Inference comes after training as it requires a trained neural network model. For example, given an input of the new image of an apple, the trained neural network that distinguishes the orange and apple will output the result "apple". Such a process is known as *inference*.

Second, we elaborate the novelty of our work. Regarding the system designs of mobile keyboard emoji prediction, the existing works can be divided into two categories: *local prediction* and *cloud-assisted prediction*. Different from the existing works for mobile keyboard emoji prediction, we consider facilitating emoji prediction services with the aid of edge computing, *i.e.*, *edge-assisted prediction*. Specifically, the pre-trained prediction models with high performances can be deployed on heterogeneous edge servers so that the smartphone can leverage such models to conduct the *inference* process for providing emoji prediction services in a more accurate and responsive way. Under such a system design, the smartphone can leverage models with high performances on multiple edge servers to provide emoji prediction services of higher quality than local prediction. Meanwhile, the emoji prediction services can take advantage of edge computing in terms of lower latency than cloud-assisted prediction.

Comment 2-3: The authors mentioned that "However, due to the intermittent wireless connections between the smartphone and the cloud [10]". However, the proposed approach also include the wireless part, namely, between the user device and the edge server. So, it is not clear how the authors solves the problem.

Response 2-3: Thanks for your comment. Compared with cloud-assisted emoji prediction, although there exists wireless connection as well, our system design (i.e., *edge-assisted emoji prediction*) is more stable and robust. Specifically, our responses are given as follows.

First, compared with cloud computing, edge computing brings computing resources closer to the end-user to provide lower latency and higher bandwidth communication (from "Latency comparison of cloud datacenters and edge servers" by Charyyev *et al.* in *Proceedings of IEEE GLOBECOM* 2020). In this way, with the aid of edge computing, the emoji prediction service can be more stable and efficient.

Second, the pre-trained prediction models with high performances are deployed on heterogeneous edge servers in our system. Meanwhile, in each round, the smartphone can select multiple models to conduct inference for providing emoji prediction services. In this way, even if the end-to-end latency between one selected model (on the server) and the smartphone is high due to the intermittent wireless connections, the others can quickly send back the predicted

results for the user to choose from. Therefore, the smartphone can provide emoji prediction service via model selection in the next round without waiting for a long delay. Such a design can improve the robustness of emoji prediction services.

Comment 2-4: What is the main scientific contribution of this paper? A use of edge computing in Emoji Prediction? However, the edge computing part of this paper does not include a significant contribution. Moreover, more difficult tasks, such as predictive typing in Gboard, are widely discussed and implemented. So, what is the significance of discussing about a simpler problem using an older approach?

Response 2-4: Thanks for your comment. Recall that in **Response 2-2**, most existing works on emoji prediction focus on how to design a suitable architecture to optimize the performances of model *training*. However, our work focus on how to optimize the *inference* process for emoji prediction.

Regarding the system designs of mobile keyboard emoji prediction, the existing works can be divided into two categories: *local prediction* and *cloud-assisted prediction*. Different from the existing works for mobile keyboard emoji prediction, we consider facilitating emoji prediction services with the aid of edge computing, *i.e.*, *edge-assisted prediction*. Specifically, the pre-trained prediction models with high performances can be deployed on heterogeneous edge servers so that the smartphone can leverage such models to conduct the *inference* process for providing emoji prediction services in a more accurate and responsive way. Under such a system design, the smartphone can leverage models with high performances on multiple edge servers to provide emoji prediction services of higher quality than local prediction. Meanwhile, the emoji prediction services can take advantage of edge computing in terms of lower latency than cloud-assisted prediction.

Comment 2-5: The evaluation is unsatisfactory. The simulation setting, especially, the wireless communication part, is not realistic enough to represent a real resource limited wireless environment which is argued by the authors as one of the main problems.

Response 2-5: Thanks for your comment. Our settings can be modeled in more detail with the wireless communication part. Specifically, in each round t of the model selection process shown in **Fig.1**, the latency in each step can be modeled as follows.

- Step 1: The user's input is generated on the mobile keyboard.
- Step 2: The smartphone selects a subset of models and sends the current input to each of them. We define the transmission latency for the smartphone sending the current input to the model on the i-th edge server as

$$d_i^U(t) = \frac{\rho}{B\psi_i^U(t)},$$

where ρ is the size of input data, B is the bandwidth of one channel, and $\psi_i^U(t)$ is the uplink transmission rate of the i-th edge server in round t. Specifically,

$$\psi_i^U(t) = \log_2\left[1 + \frac{p_i^U \left|h_i^U(t)\right|^2}{\sigma^2}\right],$$

where p_i^U is the transmit power of the smartphone to the i-th edge server, σ^2 is the noise power, and $h_i^U(t)$ is the uplink channel gain of the i-th edge server in round t.

- Step 3: Each selected prediction model conducts the inference process and outputs a predicted emoji. We define the inference latency as

$$d_i^I(t) = \frac{\rho}{\Phi_i(t)},$$

where $\Phi_i(t)$ is the processing capacity of the i-th edge server in round t.

- Step 4: The predicted emojis are sent back to the smartphone. We define the transmission latency of the i-th edge server sending back the predicted emoji to the smartphone as

$$d_i^D(t) = \frac{\phi_i}{B\psi_i^D(t)},$$

where ϕ_i is the size of predicted results on the i-th edge server and $\psi_i^D(t)$ is the downlink transmission rate of the i-th edge server in round t. Specifically,

$$\psi_i^D(t) = \log_2\left[1 + \frac{p_i^D \left|h_i^D(t)\right|^2}{\sigma^2}\right],$$

where p_i^D is the transmit power of the i-th edge server to the smartphone and $h_i^D(t)$ is the downlink channel gain of the i-th edge server in round t.

- Step 5: The user clicks on one desired emoji among the aggregated emojis on the smartphone.

In this way, the end-to-end latency of the smartphone accessing prediction model i is denoted as

$$D_i(t) = d_i^U(t) + d_i^I(t) + d_i^D(t).$$

However, the details of the above modeling will only influence the distribution of the latency which will not affect the workflow of our algorithm since we only focus on the mean value of the latency. In this way, it is not necessary to model the latency in detail as above.

Comment 2-6: The details of the algorithm should be explained. Which part is conducted at the user devices? What part is conducted at the edge server? What kind of communication are required between the end device and edge server?

Response 2-6: Thanks for your comment. In this revision, we have added the details of the algorithm including the parts conducted by smartphone and edge servers, respectively, and the specific required communication between the smartphone and edge servers. Specifically, we give the corresponding responses in the following.

As shown in **Algorithm 1**, in each round, the smartphone first conducts the weight calculation in lines 2-10. According to the calculated weight, the smartphone processes model selection and determines the selected models in lines 11-23.

The smartphone and edge servers then conduct the workflow shown in **Fig.1**. In particular, the smartphone sends the user's input to each selected model. Each selected model (on smartphone or edge server) conducts the inference process and outputs a predicted emoji, which is sent back to the smartphone. The user clicks on one desired emoji among the aggregated emojis on the smartphone.

With the above workflow completed, the user satisfaction and response latency are known to the smartphone. In this way, the smartphone updates the statistics and virtual queues in lines 24-27.

Response to Reviewer 3

Comment 3-1: My earlier concerns were mainly about the motivation of the work and its Related Work section. The authors have adequately addressed my prior concerns, and the current version is in good shape. Therefore, I recommend accepting this submission for publication in its current form.

Response 3-1: Thanks for your appreciation.