A Picture is Worth a Thousand Words: Improving Mobile Messaging with Real-time Autonomous Image Suggestion

Joon-Gyum Kim
Korea Advanced Institute of Science and Technology
kjkpoi@kaist.ac.kr

Chia-Wei Wu
University of Technology of Belfort-Montbéliard
chia-wei.wu@utbm.fr

Alvin Chiang
National Taiwan University of Science and Technology
M10115088@mail.ntust.edu.tw

JeongGil Ko
Ajou University
jgko@ajou.ac.kr

Sung-Ju Lee
Korea Advanced Institute of Science and Technology
sjlee@cs.kaist.ac.kr

ABSTRACT

Text messaging on smartphones has become one of the most popular communication methods. With many smartphone chat applications, text messaging no longer is only "text"; users send emoticons to express emotions or share pictures stored on their phones. We believe that providing more visuals in chat applications by autonomously suggesting proper images from the Internet (i.e., "auto complete" with images), based on the chat content, is the next evolution of mobile messaging. Realizing this simple vision however, is a difficult task due to the intrinsic nature of mobile chat and resource limitations of smartphones. We identify these challenges and to overcome them, we suggest integrating solutions from the field of mobile computing, natural language processing, sentiment analysis, machine learning, storage, human computer interaction, networking, and systems. We present *MilliCat*, a lightweight mobile messaging service that autonomously suggests images based on chat context to improve emotion expression, nuance delivery, and information delivery of a conversation. Experimental results from our preliminary prototype implementation show promises that real-time autonomous image suggestion can provide timely, proper images while only incurring manageable networking and energy overhead.

Keywords

Mobile messaging; Image suggestion

1. INTRODUCTION

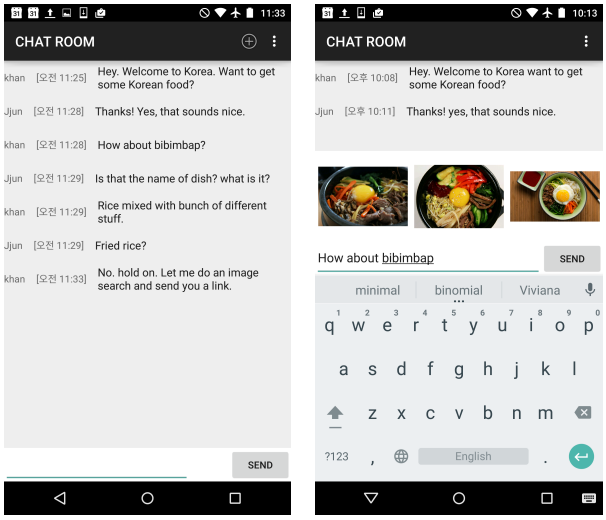
Rapid advancements in wireless communications and the wide popularity of smartphones have diversified how everyday users communicate with each other from voice-based phone calls to text messaging services and even video chat. Unlike short messaging services (SMS) over cellular networks, now with smartphones, using wireless data networks and a variety of text messaging applications, users can freely exchange text (and image) messages with no additional cost beyond their data plans. Such financial factors, combined with the changes in cultural aspects, have catalyzed the usage of smartphone messaging services. A recent report on the smartphone usage patterns of U.S. users show that text messaging is now the most widely used smartphone feature, exceeding the use of Internet access and voice/video calls [8].

However, when analyzing the functionalities of many messaging applications, their features are only slightly advanced than the basic SMS features of low-end feature phones. Specifically, these services are mostly based on a texting screen with minimal multimedia support. In a sense, while these applications use the wireless network to exchange text message data, they still lack the capability of connecting the users with the large amount of data available on the Internet as they exchange messages with their peers. Improved hardware resources of the smartphone allows for a user to not only "use" the wireless networks (as a communication medium), but also allows for a number of interactive services to be combined with the texting environment. As an example, based on the text message inputs of the users, a service can potentially provide real-time image suggestions to provide users with a chance to better express their feelings and deliver more visible information.

There are many text messaging applications on smartphones (e.g., native apps, WhatsApp, Google Hangouts, Facebook Messenger, Skype, SnapChat, and popular apps in the Asian market such as WeChat, Line, and Kakao Talk) but their additional features, besides texting, mostly focus on enabling video/voice chat and customizable skins. While users can share images, it's largely for pictures already stored locally on their phones. Recently, some services added new features for image sharing. Facebook Messenger for example, allows users to send trending animated memes using external applications such as GIPHY, but mostly focuses on "funny situations" and may not reflect emotions during conversations. Kakao Talk has a search feature initiated by typing "#". This feature searches the Internet for a wide variety of contents, including dictionary, music, blog, twitter, and application search. While providing some useful information, it can potentially waste energy and bandwidth

HotMobile '16, February 26-27, 2016, St. Augustine, FL, USA

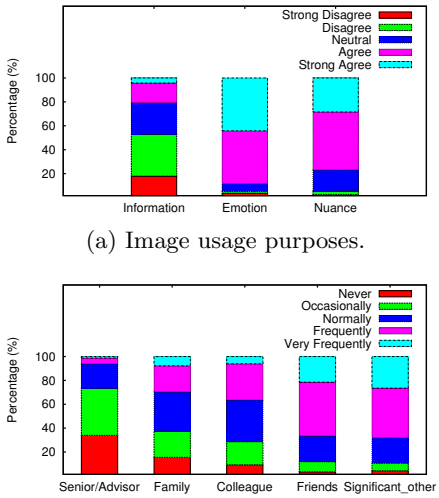
DOI: http://dx.doi.org/10.1145/2873587.2873602



(a) Typical messaging applications usage.

(b) MilliCat usage.

Figure 1: Illustration of MilliCat operations in use compared with traditional smartphone messaging.



(a) Image usage purposes.

(b) Image exchanging peers.

Figure 2: Purposes of using images on smartphone messaging applications and peers to whom images are typically used.

resources by overloading users with too much information, of which, mostly, a user may not be interested in.

In this work, we present *MilliCat*, a smartphone messaging service that analyzes the messages being entered by the user to *automatically* identify a proper image on the Internet and provides the search results as real-time suggestions to the user. We can think of this as an "auto complete" with images. As we illustrate in Figure 1, with MilliCat, users get real-time image suggestions on their input text.[1] For many conversations, such a feature can benefit the overall flow of the conversation and save the user from manual external application interaction.

We start this work with a user survey conducted with 250+ active smartphone users around the world on their preferences of image usage while using text messaging applications on smartphones. Our survey reveals that regardless of the gender and age groups, users agree that images can play a key role in their conversations. Many users concur that images from the Internet can be used especially to better express their intentions and relieve any tension that pure text-based messaging might introduce.

MilliCat combines sub-modules that (i) perform sentiment analysis to identify the opinions of users' input data, (ii) process the input data using natural language processing techniques to identify the proper phrase to perform the search, and (iii) interconnect the smartphone application with prefetching and caching techniques implemented at an external server dedicated for image search and analysis. More importantly, we designed MilliCat so that it can be easily adopted and implemented as a plug-in layer for many pre-existing text messaging applications. Using the preliminary MilliCat prototype, we perform an empirical study on the additional bandwidth and battery usage on the smartphone as it performs real-time image request and fetch operations with our dedicated server. Our results show that with filtering options and text processing schemes, MilliCat provides real-time image suggestions with latencies < 100 $msec$ and < 50 KB of packet overhead per conversation.

The contributions of this work are three-fold.

- We perform a large-scale user survey to identify the needs and requirements for providing a real-time autonomous image suggestion service on mobile chat applications.
- We identify a set of technical challenges in addressing such application requirements, and list potential solutions and tools to improve the user experience in the domain of real-time autonomous image suggestion.
- Our observations lead to the design of MilliCat, a prototype implementation of a real-time autonomous image suggestion service for smartphone messaging applications, which we use to perform preliminary studies on the latency and overhead performance, and suggest guidelines for future research.

2. USER SURVEY

We start with a survey from 250+ users ranging in age groups from teens to over 50, with 11 nationalities, and different professions, including students, engineers, artists, scientists, salesmen, doctors, game designers, architects, chefs, housewives, etc. Without introducing details on the MilliCat design, we asked the participants, whom all are active smartphone users, to present their perspectives on image usage for mobile messaging. Specifically, our survey consisted of 11 questions, focusing on identifying the daily usage of smartphone messaging applications and their usage patterns on emoticons or images (typically already stored on the users' smartphones). Based on these results, we asked whether and how they thought automatically suggesting images from the Internet, based on what their input messages are, would benefit their conversations. Furthermore, we asked users on the types of images that they would like to be suggested and also the situations (e.g., conversation

[1]While this example is for a simple word, the core functionalities of MilliCat can be extended to more complex word phrases.

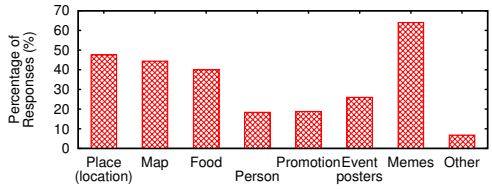


Figure 3: Preferred types of images for autonomous suggestion.

type, relationships with peer, etc.) they would prefer sharing these images in. While we omit the graphs due to space limitations, 95.7% of the survey participants answered that they often use smartphone-based messaging applications and 90% replied that they frequently use emoticons and images in their chat.

Figure 2 presents a summary of why and with whom users share images with during a smartphone messaging conversation. Notice in Figure 2(a) that most users agree that images are useful for expressing an appropriate nuance or emotions. Given that it is difficult to realize each others' "feelings" during *text* messages, users find this as an effective use-case for images. Furthermore, 20% of the participants express that images can be useful for information exchange, and the majority of these replies relate to exchanging map information or images of an unfamiliar conversation topic.

As Figure 2(b) shows, there is a trend in "when" the users prefer to use images. We notice a distinction in the usage patterns of images and emoticons with respect to the conversation peer. While widely used with friends or a significant other, the usage rate decreases significantly for colleagues, other family members (besides significant others), and senior colleagues (e.g., boss at work). This is an interesting result, and opens up opportunities for context-aware, adaptive image suggestion. Since an analysis on the contact groups or the chat history can provide hints on what category each contact belongs to, we can use this data to suggest images only when with high confidence that the images will be used when chatting with the contact.

Survey results also indicate that a high percentage (81.57%) of the participants would prefer to be autonomously suggested images from the Internet, based on the text being entered on the chat application. These results together motivate the need for further research in the domain of autonomous image suggestions for smartphone messaging applications as a way to improve the user experience.

We asked additional questions on what types of images (based on the typed text) would be beneficial if suggested autonomously during conversations. Figure 3 reveals that memes or funny images to go with the text were the most popular, followed by images of target locations, maps and food. We also notice here that there is no dominant answer to this question given that the maximum rate is ∼65%. We note that the participants were asked to select all that apply in the survey. This result suggests that an autonomous image suggesting service should not focus on a single category of images but diversify its search options.

18.43% of survey participants indicated they would rather *not* use images during their chats. Their quotes include "would take too long to find an image when connection is slow," "would it waste my data and slow down the speed?" Hence, such a new service, while providing real-time autonomous image suggestions, must be lightweight enough to not excessively consume power and wireless bandwidth.

In summary, the take away messages from our user survey are the following:

- Many mobile users believe it would be useful to be able to use various images from the Internet during the chat
- Mobile users would use various images such as memes, locations, food, maps, etc., within their chat
- Users are concerned however that this service might incur excessive power and data overhead, and hence our service must be energy and data efficient.

3. SYSTEM REQUIREMENTS AND TECHNICAL CHALLENGES

Based on the observations from our user survey and prior experiences with resource limited mobile platforms, we identify a set of system-level requirements and a list of technical challenges in realizing an autonomous image suggestion module for smartphone-based messaging applications.

Below is a short-list of core system-level requirements.

- **Appropriateness of the images:** The images should be suggested only in situations when an image helps the emotion expression or nuance delivery. The suggested images hence should match the context of the users' conversation. We need to understand *what kinds* of images to show, and believe our user survey provided us with hints. Image search quality [1] is also important, and we rely on available tools as it is not the focus of our work.
- **Timeliness:** Images should be suggested at proper times with respect to *when* the user might intend to use a suggested image. For this, the system should "learn" when to send queries for an image search. As an example, based on the input of the user, the system should know whether it should search for an image on a per-character basis or per-word basis.
- **Image suggestion latency:** No matter how "appropriate" the image is, on a user experience perspective, it is important that the suggested images appear within the duration of the topic conversation. Therefore, the *latency* of image suggestions, which includes the delay for querying, image processing, wireless transmissions, and display, should be minimal.
- **Energy and resource efficiency:** One of the major concerns from our survey participants was in the energy and resource usage of our image suggestion service. Mobile phones, although recently becoming more powerful, are still considered as resource-limited platforms given that they operate on battery and over-utilizing processing power would lead to energy drain. As a result, the image suggestion service should use only minimal computational and networking resources to conserve energy.

On a practical perspective, there are a number of technical obstacles to overcome in designing a service that satisfies the requirements above.

First, the process of suggesting an appropriate, proper image is a challenging task. Specifically, the complexity of chatting sentences and the diversity of emotions complicate

the estimation process in selecting the images to suggest given a specific word. We summarize some of the important tasks an image suggestion service should consider in selecting appropriate and proper images.

Sentiment analysis: Also known as opinion mining, sentiment analysis focuses on extracting underlying subjective information, such as emotions or opinions from a given text [3, 9]. It has been widely used, for instance, in analyzing consumer reviews on product websites, social media sites, blogs, and discussion forums. Existing tools such as Stanford CoreNLP [4] or the Natural Language Toolkit [5] can be integrated to provide sentiment analysis. Nevertheless, their computational complexity is high and accuracies are still in the 80% range (even with text longer than typical mobile chats) [4]. As an alternative, a list of emoticons can be used to catch first-stage opinions and extending this sentiment analysis feature remains as a major technical challenge.

Text partitioning: For proper image suggestion, the generation of a proper query message for image searching is extremely important. Given the text of a user, deciding at what point to partition the text and generate a query packet determines the quality of the image search. Partitioning can occur on a per-character basis, in words, word phrases, or in sentences. This design choice will not only impact the appropriateness of the suggested image, but also the bandwidth and energy usage of a mobile platform.

Word types: As related to text partitioning above, one could perform the query for each word. Moreover, one could limit the query to only nouns, instead of performing also for articles, adjectives, verbs, etc. However, recognizing whether a word is a noun could be difficult. For example, the word "love" in "I love you" is a verb while in "send my love" it is a noun. Also, within the nouns, there are *abstract* nouns that represent an intangible concept, such as emotions (e.g., happiness, anger), attributes and qualities (e.g., honesty, trust) that are difficult to have the right image. *Concrete* nouns on the other hand, are tangible, such as people (e.g., doctor), objects (e.g., cake), and places (e.g., island) and an image search on them would likely result in visuals that would improve the conversation. Note also that mobile users often use Internet slangs and acronyms (e.g., LOL) during smartphone chats, and are more tolerant to typos than in professional writing settings. The autonomous image suggestion service must understand this intrinsic nature of mobile chat culture to be effective.

The second technical challenge is in satisfying the latency requirement for image suggesting and display. Our experiences show that an intermediate server for image searching, fetching and caching helps ensure the fast display of images at the smartphone. However, the challenge here is in the management of this intermediate server's capabilities. Configuring the server to cache a large amount of data can benefit the latency experienced at the smartphone by minimizing the query latency, but the infrastructure cost may increase significantly. Schemes that allow the server to learn the chatting habits of individuals (and on a larger population scale) may help resolve this trade-off, but requires further research. In addition, the selection of a proper image search engine, with respect to the geographic location of the user and the server, can give a noticeable impact to overall latency performance.

Note that existing mobile prefetching techniques [2, 7] utilize users' smartphone usage pattern, for example, launching email and news applications in the morning, to prefetch contents and images. However, applying prefetching for mobile messaging services can be challenging due to the difficulties in predicting which words would be used at which instance.

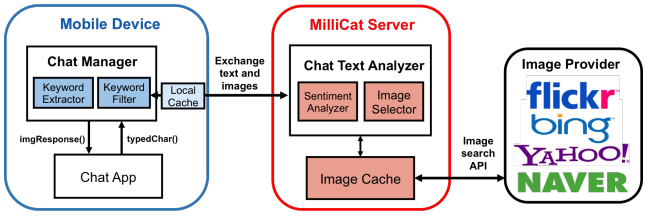


Figure 4: MilliCat system architecture.

Third, minimizing the energy and bandwidth consumption is a major requirement to satisfy, and at the same time a very challenging task. A system designed for image suggestions within mobile chatting applications should intelligently manage the queries it sends to the image server, since a query not only requires energy and bandwidth to *send* the packet, but the response packet will contain the resulting images, which will consume even more resources. A tradeoff between the latency and the resource consumption must be considered. Moreover, natural language processing (NLP) schemes that suppress unnecessary queries can help maximize the efficiency at the smartphone [6].

Lastly, the design of an efficient user interface is another significant challenge. Issues such as the number of images to suggest per query or the layout of presenting the images on the screen can have high impact on both the usability and system-level performance.

4. MILLICAT

4.1 System Architecture

As a preliminary prototype to evaluate the features needed in an image suggestion service for smartphone messaging applications, we design MilliCat. The overall system architecture of MilliCat is shown in Figure 4. On the smartphone, the **MilliCat chat manager** connects to a chat application, and interacts with the **keyword extractor** and **keyword filter** to identify core terms in a text and suppress queries for common words such as article terms. The smartphone also holds a small-sized **local cache** to minimize the number of external-bound queries. Once a query reaches the MilliCat server, the **chat text analyzer** uses tools such as the **sentiment analyzer** to extract a proper search keyword. If an associated image is not in the cache, the server interacts with external databases for a proper image. The retrieved image is then cached and resized before the **image selector** validates its usefulness and returns it to the smartphone.

When interconnecting with a chat application as a plug-in, MilliCat provides APIs such as `typedChar()` and `imgResponse()` to receive the input text from the chat application and send a sequence of suggested images for the application to display. Within these APIs, we ask the application to include its own application identification along with a unique ID for each conversation for MilliCat to distinguish between different apps and conversations within. This flexibility allows MilliCat to easily interact with and improve the original functionalities of existing chat applications.

Being in the early stages of development, MilliCat currently includes most of the core features for real-time image

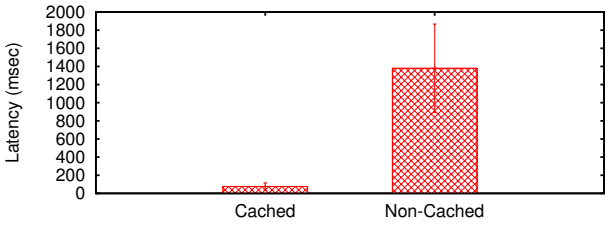


Figure 5: Image request latency with and without at-server image caching.

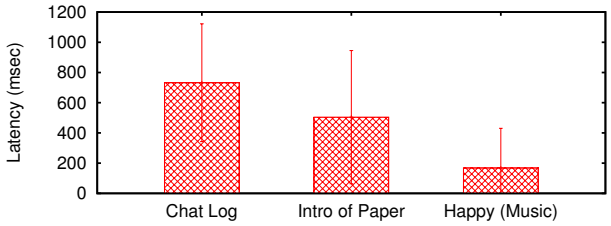


Figure 6: End-to-end image suggestion latency when queried on a per-word basis with different data sets.

suggestion as depicted in Figure 4. We are still undergoing research in providing suitable schemes for the sentiment analyzer and the keyword filter. Nevertheless, given that a major challenge is in ensuring that MilliCat does not excessively consume the smartphone's limited resources, using the current version of the system prototype, we perform a preliminary study on the bandwidth and energy overhead that real-time image suggestion introduces to traditional mobile messaging applications. Note that as our future work, we plan to evaluate the acceptability of MilliCat as part of a user study.

4.2 Preliminary Evaluations

4.2.1 Impact of Caching

First, we present in Figure 5 the latency of image suggestions with and without prior caching at the MilliCat server. Here we test for 100 different nouns and used Bing image search APIs for external image searching (e.g., non-cached images). For the cached case, we store all the images on our server and return these images directly without any external search. We see that as expected, image caching significantly reduces latency, from the query at the smartphone application to the image display, by an order of magnitude. This result shows the impact of at-server image caching for MilliCat, but we point out that image caching results in two main issues. First, as mentioned earlier, caching the images increases the storage overhead at the server. Second, without external searching, newly "trending" images may be difficult to suggest, since the server will return the images in its storage rather than performing a new search. An expiration timer for each image can resolve these issues, but the interval of image caching will become an important design choice leading to a major performance tradeoff.

Figure 6 presents the latency performance for the case where the server caches the images that were previously searched. Therefore, with commonly occurring words, the average latency of the query-reply process will reduce. We use three data sets for this experiment; the first data is a set of chat records from famous chat applications, as we detail later this section. The second data set is the first two paragraphs of this paper, and the third data set are lyrics from the song "Happy" by Pharrell Williams. Specifically, in the "Happy (Music)" data set, we see the word "happy" along with many phrases being repeated. As the results show, with more repeated data-based queries, the server gains the chance to cache related images, thus reducing the query-reply latency. Overall, these results show that with a well-configured caching server and a reasonable amount of history data, the latency of image suggestion can fall within practically usable range (<∼100 msec).

4.2.2 Networking & Energy Overhead

We now evaluate the packet transmission and reception overhead that real-time image suggestion systems, such as MilliCat, introduce. For our evaluations, we utilize 80 sample smartphone messaging conversations (e.g., WhatsApp, Facebook messenger, Android SMS, iOS SMS conversations) available on the Internet.[2] Each conversation has an average length of 5.95 lines with each line having an average length of 175.7 characters (or 42.175 words). Using this data, we test for five different test cases, each with different query issuing policies at the smartphone application. In the first case, a query is issued to search for a matching image at every keystroke. When typing the word "cat", in this scheme the application issues three queries; one for "c", another for "ca" and the third for "cat". The server checks if this word can be identified as a word through its local dictionary and makes a search query if the image data is not present in its cache. Potentially, we plan to use the Princeton WordNet search [10] for supporting a more robust and complete English word set. In the second case, a search query is issued for every `space` key entry. This case sends queries for each word typed in the text and was used for our experiments in Figure 6. Our third querying method combines the word-based querying method with the smartphone's keyword filter to suppress article terms such as "a" or "the" from being queried. In the keyword filter, we have a list of ∼80 words, which we are confident that a typical user would not request an image for. The fourth method utilizes the internal dictionary for noun-matching to only query the word if it is identified as a proper noun, and our fifth method combines this scheme with the keyword filter.

Figure 7 presents the average transmission overhead for each conversation trace. Note that when queries are sent on a per-click (i.e., per-character) basis, the transmission overhead is extremely high. As we perform word-level transmissions, local keyword filtering, and utilize the dictionary for typo filtering, the amount of transmissions drops dramatically. With the internal dictionary and keyword filter, only ∼2 KB of additional queries are generated on average for each conversation.

The result of the queries issued in Figure 7 are responses by the MilliCat server with image suggestions for the application. We plot this packet reception overhead in Figure 8. With more queries, comes more responses with images. As a result, for the per-click (per-character) case, an average of ∼9.1 MB are received for a single conversation. By changing

[2]We made this data available at https://goo.gl/O6k34x.

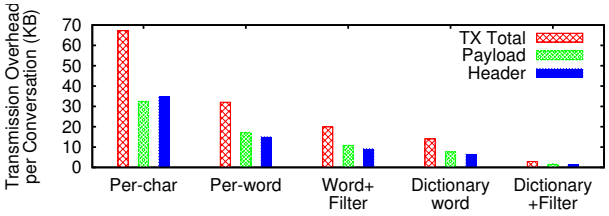


Figure 7: Chat trace transmission overhead for different querying methods.

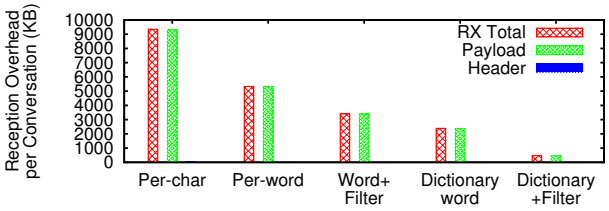


Figure 8: Chat trace packet reception overhead for different querying methods.

the querying method, we significantly reduce this overhead. This reduction also implies that the per-click case would request images less related to the actual word that the user is typing. For example, as the user types the word "catholic", the per-click case will return images for "cat" as well. Nevertheless, for the case with the internal dictionary and keyword filter, we are to expect ∼47 KB per conversation. We note that for each of the search query, our current system is configured to suggest three images. This, of course, is a design choice, and lowering this will naturally reduce the reception overhead.

Finally, in Figure 9 we plot the energy used to support the operations in Figures 7 and 8, using a Samsung Galaxy SIII smartphone with Wi-Fi connections. Here, we neglect the baseline operations of the smartphone and focus on the energy spent only for MilliCat packets (e.g., queries and replies). Naturally, the trends of energy consumption are similar to that of the packet exchange overhead. We point out that with the dictionary and local keyword filter, the power usage per conversation is ∼35 mW on average.

5. SUMMARY

We started this work asking ourselves, "Can we improve the user experience of smartphone messaging services using real-time autonomous image suggestions?" As of now, we have two different answers. First, our survey results show that users are willing to use such additional features as part of smartphone messaging, and our empirical results with MilliCat show that the latency and overhead of real-time image suggestions are within tolerable bounds. Therefore, real-time autonomous image suggestion for mobile chat applications holds the potential to improve mobile user experience.

On the other hand, on a technical perspective, there are still many challenges to overcome. One major challenge is in the fact that real-time autonomous image suggestions on smartphones require a combination of findings from diverse research fields: including areas such as mobile computing,

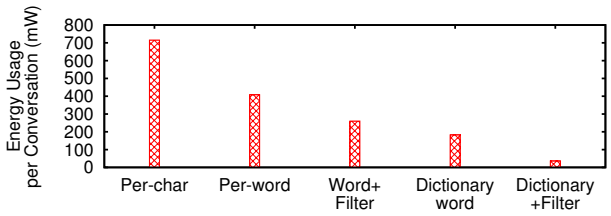


Figure 9: Overall energy overhead for different querying methods.

natural language processing, sentiment analysis, machine learning, storage, human computer interaction, networking, and systems. A key obstacle here is in compressing complex text analyzing algorithms to operate effectively with respect to the requirements of smartphone applications. For example, unlike how sentiment analysis and most text analysis tools are used today with a massive set of learning data from the Internet, chat messages are short and diverse in context. Therefore, taming existing text analyzing schemes to well-operate with smartphone chat messages is important in potentially initiating more precise queries and suppressing unnecessary queries from being sent. Nevertheless, we envision that our efforts in designing MilliCat will be the basis of realizing the next evolution of mobile messaging with more visual contents.

6. REFERENCES

[1] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. C. Jain, and C.-F. Shu. Virage image search engine: an open framework for image management. In *Proc. SPIE 2670, Storage and Retrieval for Still Image and Video Databases IV, 76*, 1996.

[2] B. D. Higgins, J. Flinn, T. J. Giuli, B. Noble, C. Peplin, and D. Watson. Informed mobile prefetching. In *Proceedings of the 10th International ACM Conference on Mobile Systems, Applications, and Services (MobiSys '12)*.

[3] B. Liu. Sentiment analysis: A multifaceted problem. *IEEE Intelligent Systems*, (3):76–80, 2010.

[4] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[5] NLTK Project. NLTK 3.0 documentation. Available at: http://www.nltk.org/, Oct. 2015.

[6] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Proc. of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*.

[7] A. Parate, M. Böhmer, D. Chu, D. Ganesan, and B. M. Marlin. Practical prediction and prefetch for faster access to applications on mobile phones. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*.

[8] A. Smith. U.S. Smartphone Use in 2015. PEW Research Center Report. Available at: http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/, Apr. 2015.

[9] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*.

[10] The Trustees of Princeton University. WordNet: A Lexical Database for English. . Available at: https://wordnet.princeton.edu/, Mar. 2015.