AutoCaption: Automatic Caption Generation for Personal Photos

Krishnan Ramnath, Simon Baker, Lucy Vanderwende, Motaz El-Saban, Sudipta Sinha,
Anitha Kannan, Noran Hassan, and Michel Galley
Microsoft Research, Microsoft Corporation

Yi Yang and Deva Ramanan
Dept. of Computer Science, UC Irvine

Alessandro Bergamo and Lorrenzo Torresani
Dept. of Computer Science, Dartmouth University

Abstract

AutoCaption is a system that helps a smartphone user generate a caption for their photos. It operates by uploading the photo to a cloud service where a number of parallel modules are applied to recognize a variety of entities and relations. The outputs of the modules are combined to generate a large set of candidate captions, which are returned to the phone. The phone client includes a convenient user interface that allows users to select their favorite caption, reorder, add, or delete words to obtain the grammatical style they prefer. The user can also select from multiple candidates returned by the recognition modules.

1. Introduction

The first thing people often do right after taking a photo on their phone is to share it. Most sharing mechanisms (MMS, email, Facebook, Twitter, cloud storage) allow the user to enter a caption describing their photo. It is rare, however, for users to actually enter a caption. This is a shame because if they did, it would help systems better organize their photos, for example by keyword based search.

We present AutoCaption, a system to help users with this process. By automatically suggesting a caption to the user, we argue that the user is much more likely to engage with the system. They are much more likely to either accept the caption or perform some simple edits, than they would have been to enter a caption from scratch.

Our system operates by first uploading the photo to a cloud service. A number of processing modules are then run in parallel. (See Figure 1.) Currently our system runs face detection [26, 20], face recognition [4], proxemics detection [24], scene classification [23] and landmarks recognition [2]. Meta-data such as GPS and timestamp information is also processed. Our system is modular and extensible with all the modules sharing the same output representation. This common representation allows future modules such as weather recognition, activity recognition, clothing recognition, etc, to be added with minimal development effort. The results of the processing modules are fused and used to generate a set of possible captions. Where possible, these captions are personalized to the photographer. The final set of captions are then returned to the phone.

Generating exactly the caption that the user wants is very difficult. Even if all the modules operated correctly, the first caption returned by the system may not be exactly what the user wanted. Our system has three user-interface constructs to help the user obtain a caption they are happy with. First, the text generator creates a large number of captions. Tapping on the caption on the phone cycles through the various options. Second, we allow the user to interactively edit the caption. They can drag a word or phrase from any part of the caption to any other position, or to trash. As the system has knowledge of a large number of captions, it can then “snap” to the closest valid caption. Third, we allow the user to select from multiple candidates using a construct that is similar to a keyboard auto-complete function. For example, if the face recognition module returns the top 3 candidates, then the user can select from those candidates.

The addition of the user interface to our system has an important effect. Edits made to a caption can be propagated back to the system so that it can learn to do better. For example, the selection of one of the multiple candidates, or the entry of a new candidate, can be treated as labeled training data. Because the user edits a word in a computer generated caption, we know what they are editing in the photo. There is no need to parse the caption, as there would be if we tried to infer tags from a freeform user input caption. When they edit a caption, a user is actually performing face, proxemics, scene, location, and landmarks tagging all at the same time. A user is unlikely to perform all this tagging if they need to do them one at a time. Furthermore, reordering edits could be used to learn user stylistic preferences.

1.1. Related Work

The generation of a natural language description of an image is a relatively new field, yet has received a great deal

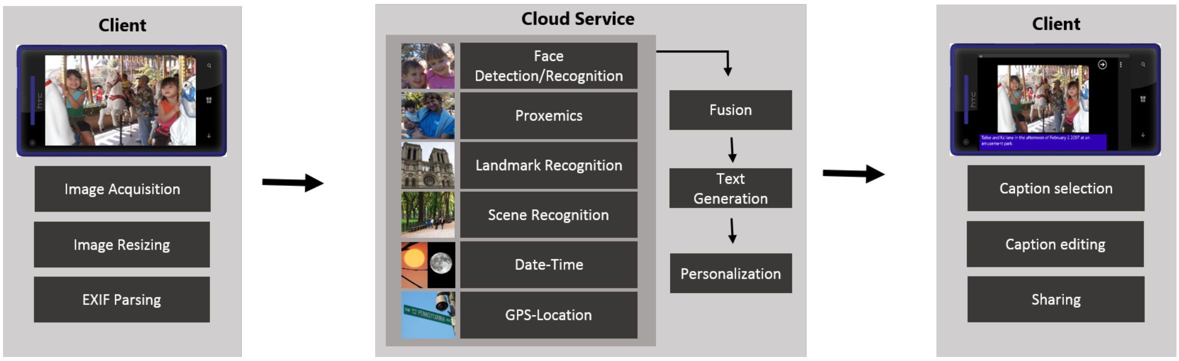


Figure 1: System architecture. (Left) The smartphone client captures the photo and uploads it along with associated meta-data to the cloud service. (Middle) The cloud service runs a number of processing modules in parallel. The outputs of the processing modules are passed through a fusion step and then to the text generator. The generated captions are then personalized. (Right) The client receives the captions and allows the user to pick and edit them before sharing.

of attention in the last few years [7, 19, 25, 13, 14, 10, 18]. On a high level there are two main approaches. The first is the **retrieval** approach of trying to match the image to a database of already captioned images and using elements of those captions [7, 19]. The second approach is the **generative** approach of using computer vision to recognize entities or relationships in the image and then combining the outputs to generate a sentence [13, 25, 10, 18]. Note that [14] uses a combination of these two techniques.

Whereas most prior work has focused on generating better and better sentences in various respects, we investigate what it takes to build a practical system.

Extensibility: Our system architecture and internal representation allows the easy addition of new processing modules as they become available. For example, the addition of the landmarks module to the already working system took around 1 hour of coding.

Error Tolerance and User Interface: Our system does not just generate the single best caption. It generates multiple captions, retaining multiple candidates whenever the processing modules output them. These generalizations allow the development of an engaging user-interface that allows the user to quickly obtain the caption they wanted.

Personalization: Our system knows who the user is and can personalize entities in the generated captions, leading to a more satisfying user experience.

2. System Architecture

AutoCaption consists of a thin, light-weight smartphone client and a cloud service. (See Figure 1.) A photo chosen by the user is sent to the cloud service for processing and the resulting captions are sent back to the client for viewing and editing. We now discuss each step of this workflow.

Capture: The photo can come either from the camera or from storage. The client first resizes the photo to a fixed maximum resolution to optimize data transfer to the cloud. The client also extracts meta-data such as time, date, and GPS location (from the OS for photos captured by the camera or from the EXIF meta-data for photos in storage.) The results are sent to the cloud service.

Cloud-Service Processing: The cloud service first runs a number of processing modules in parallel. We currently perform face detection [26, 20] and recognition [4], proxemics detection [24], scene classification [23], landmarks detection [2], process the date and time meta-data, and analyze the GPS information. Our system is modular and extensible. Future modules such as weather recognition, activity recognition, and clothing recognition could easily be added. This is enabled by the modules sharing the same output representation. (See Section 2.1.) The outputs of the modules are passed to a fusion module which integrates them, removes conflicting results, and passes the results to the text generator. The text generator creates a large set of possible captions which are personalized and sent back to the client.

We designed the cloud service to be both fault tolerant and efficient. We used Windows Azure [17] which provides fault tolerance. Our service still operates if one or more of the processing modules fails to return a result for whatever reason. The text generator can operate with any subset of partial information. For computational efficiency we run the processing modules in parallel, which are often multi-threaded and call additional services as needed.

View, Edit, and Share User Interface: Once the client receives the results, it first displays the top scored caption. It also allows the user to select from the entire set of results and to edit the caption interactively. (See Section 4.) Once the user is happy with their caption, the results are saved to the cloud service for retrieval at a later time. The user can also share the photo and caption to a social network.

2.1. Internal Representation

The inputs to the processing modules are an image and the associated meta-data. To build a modular system where we can easily add new modules, we need a representation of the output that is common across the modules and is rich enough to represent all possible cases. We use an entity-relationship representation [5]. The objects and places that are detected or recognized are entities. Examples include *Face*, *Beach*, and *Landmark*. We allow unary relationships that provide a property of an entity. Examples include *NameOf* and *CityAddressOf*. We also allow n-ary relationships (where in most cases n=2). Examples include *ArmAroundShoulder* and *HoldingHands*. Relationships are directed so the order of the entities in the relationship specifies, for example, who has their arm around whom.

3. Cloud Service

We now describe the cloud service in Figure 1(middle). We first describe the processing modules, then the fuser, the text generator and the personalization step.

3.1. Face Detection and Recognition

We used a multiple model [26] variant of Viola-Jones [20] to perform face detection. For face recognition, we use the learning-based descriptor of [4].

Two important questions for face recognition are: (a) how does the system protect users' privacy and (b) how does the face recognition module obtain a set of labeled training for each user. In terms of protecting privacy, the system requires each user to create an account and login. The face recognition module maintains separate trained models for each user. A face in one person's photos cannot mistakenly be labeled as someone outside their database of tagged friends and family.

In terms of adding labeled training data to the system, there are two ways. The first method is to side-load the tags using a web interface that is also exposed by the system. The user logs into the system website, tags a set of their photos using a simple desktop graphical user-interface (GUI), and uploads the generated face tag database to the system. The second method of adding face tags is within the phone client. When a face is detected and no name is found, or if the name is wrong, then the user can edit the text using the user interface on the client. Whenever the user performs such an edit, the new label for that face is passed back to the face recognition module to be stored in the face tag database. If the photo contains two or more faces, the system highlights the face that is being edited when the user enters edit mode so they know which face they are tagging.

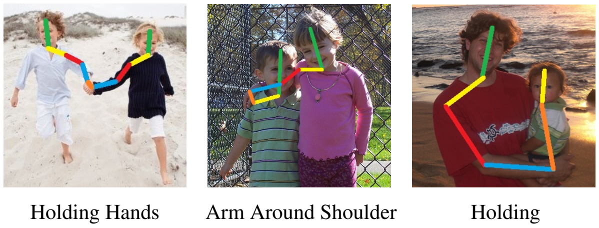


Figure 2: Our proxemics module recognizes the three most common proxemics from the set of six stuided in [24].

3.2. Proxemics

One of the most interesting properties of any photograph containing people is how the people are interacting. The field of understanding interpersonal interaction is known as *proxemics* and can be be traced to the work of Hall [11]. Yang *et al.* [24] recently presented a computational theory of proxemics and showed how to recognize six proxemics. The essence of the approach in [24] is to build a pictorial structure [8] consisting of two people plus a spring connecting them. The addition of the spring couples what would otherwise be two independent body pose estimates. This coupling makes the joint proxemic and pose estimation problem more robust than a single body pose estimate.

The location of the spring connecting the two people depends on the proxemic to be recognized. In our proxemics module, we implement three proxemics from [24]: (1) hand-hand (holding hands), (2) hand-shoulder (arm around shoulder), and (3) holding (hand-torso). In Figure 2 we include one example image for each proxemic overlayed with the estimated pictorial model. As explained in [24], irrelevant parts of the two people (e.g. the legs and the non-touching arms) can be dropped from the model.

3.3. Scene Classification

Automatic scene recognition has received a great deal of interest in the computer vision community. One of the most important papers is that of Xiao et al. [23] which introduced a large set of 908 scene classes, as well as presenting a comparative performance study for a set of popular visual features. Our scene classifier draws heavily on [23]. In our experiments, we found that using HoG along with color histogram features usually achieves the highest accuracy. Whereas the work in [23] used an SVM, we obtained the best performance using a random forest classifier with 100 trees and a maximum depth of 40. Based on the most commonly found scene categories in typical personal photos we built two versions of the scene classifier. The first classifier has 18 classes {*kitchen, bedroom, restaurant, lake/natural, living room, stadium, snowfield, desert, mountain, playground, rainforest, swimming pool, yard, street, beach, park, amusement park, office*}. The second has just four classes {*beach, park, amusement park, office*}.

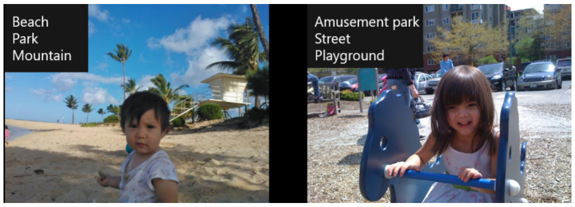


Figure 3: Example results from our scene classifier.

We train and test our classifier using the data splits presented in [23]. We use 672 training images per class and an average of 50 test images per class for the 18-class version and an average of 506 training images per class for the 4-class version. In our experiments we found that top 3 classification accuracy for the 18 classes is around 85%, whereas the top 1 classification accuracy is around 50%. Our scene classification module therefore returns the top 3 candidates so that the user can choose from them in the UI. Figure 3 includes two example images with an ordered list of predicted scene classes. The image on the left shows a correct result. The one on the right shows an error for top 1 classification, but for which top 3 classification is correct.

3.4. Landmarks

Although the scene classifier provides a general sense of where the photo was taken, recognizing specific landmarks allows for more informative and interesting image captions.

We treat each pre-determined landmark as a different class [16]. We use the approach proposed in [2] which is based on the popular bag of visual words approach [6] for image classification. DAISY feature descriptors [21] are extracted locally at keypoints in the image which are then efficiently aggregated into a global feature vector using a discriminative codebook and a VLAD-based encoding technique [12]. Finally, a set of one-vs-all linear SVM are used to predict the presence of a landmark in the image.

To train the codebook and the linear classifier as described in [2], 2.4 million images of 620 landmarks were downloaded from the Internet using text queries. The set of images of each landmark was processed using structure from motion to remove irrelevant images after which 614K images were retained. To evaluate the classifier, we tested it separately on held-out test sets of 62K images of 620 landmarks and 5K images of 25 famous landmarks where GPS tags were not used. The top-1 classification accuracy of our classifier in these two evaluations was 71.3 % and 98.4 % respectively. Figure 4 shows a few example results.

3.5. Date and Time

Adding non-visual properties such as the time and date of capture can significantly enrich a caption. The Date-Time module parses the meta-data sent by the phone client and generates the time of day, month and year in phrases like “ the morning of December, 2012”.

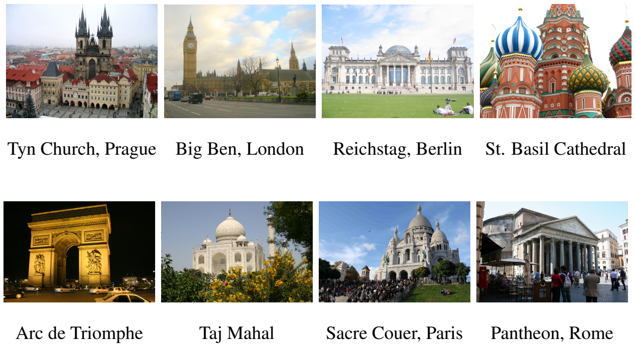

Tyn Church, Prague
Big Ben, London
Reichstag, Berlin
St. Basil Cathedral
Arc de Triomphe
Taj Mahal
Sacre Couer, Paris
Pantheon, Rome

Figure 4: Example landmark recognition results.

3.6. GPS-Location

Most photos taken on smartphones record the GPS location of the picture. The GPS-Location module takes the latitude-longitude coordinates from the GPS and queries a location service. We used the Bing Maps Locations Service [3] for this purpose. This service reverse geo-codes the coordinates and provides a street address. We filter the results and retain the city and state/country names.

3.7. Fusion Module

The fusion module integrates the outputs of the processing modules and passes them to the text generator. In the the process, it can remove conflicts and perform additional processing. For example, the fusion module could compare a detected landmark to its GPS, city, or scene type (as available) and removes false positives.

An example of additional processing is to combine the GPS information with the scene class to query a Points-of-Interest (POI) service. For example, if the photo is taken at Disneyland, the caption would be more engaging if the system generates the caption “Disneyland” rather than “an amusement park in Anaheim, CA”. In the fusion module, we query the Google Places POI Service [9] with the GPS location. To narrow down the search, we also pass the output of the scene classifier (amusement park, park, beach, etc) to the POI service. If the POI service returns any matches, we add the returned POI to the set of entities and relationships passed to the text generator.

3.8. Text Generation

The key requirement for our text generation algorithm is to support the following user-interface (UI) functions:

Selection: We must generate a quality ordered list of possible captions that the user can choose from.

Reordering: We must keep track of the recognized entities so that the user can reorder them in the UI.

Editing: We need to retain multiple candidates to allow users to choose from the top candidates in the UI.

We follow the “overgenerate-and-rank” paradigm common to much prior work in statistical text generation [15, *inter alia*]. We first construct a compact representation of many captions—in our case, the representation is a recursive transition network [22]—and then score and rank this representation with a linear model. Specifically, we first construct a recursive transition network (RTN) by combining 14 sentential templates, one example of which is:

PERSONPHRASE SCENEPHRASE? *in* LOCATIONPHRASE .

Each template consists of a sequence of words (e.g., *in*) intermixed with non-terminal symbols (e.g., PERSONPHRASE) to be later expanded. The example above contains three phrases (PERSONPHRASE, SCENEPHRASE, and LOCATIONPHRASE), where '?' indicates the phrase may be omitted.

The second part of RTN construction consists of defining each non-terminal using different possible realizations. For example, suppose two people are recognized in the photo, we define PERSONPHRASE as the set

$$\{ \text{“}\mathrm{N}_1 \textit{ and } \mathrm{N}_2\text{”}, \text{“}\mathrm{N}_2 \textit{ and } \mathrm{N}_1\text{”}, \text{“}\mathrm{N}_1\text{”}, \text{“}\mathrm{N}_2\text{”} \},$$

where the non-terminals N_1 and N_2 represent recognized entities taken from a list of names in the face recognition database.[1] For example, the realization of N_1 might be:

$$\{ \text{“Joe”}, \text{“Bob”}, \text{“Fred”} \}.$$

Finally, the RTN is scored with a linear model (which currently includes confidence and word count). We use dynamic programming to find the optimal caption and present it as the initial choice to the user. To let the user explore other caption realizations encoded in the RTN, a decision must be made whether to expand the RTN explicitly, or whether to expand it lazily in the UI. In our current implementation, we explicitly expand the combination of templates and phrase expansions in the cloud, but handle text realizations of the entities lazily in the phone client.

3.9. Personalization

Personalization can dramatically improve the emotional appeal of a caption. We perform personalization with a set of simple transformations on the set of text realizations of the entities. If the user is logged in as "Joe", then the realization of N_1 above might be transformed to:

$$\{ \text{“Me”}, \text{“Joe”}, \text{“Bob”}, \text{“Fred”} \}.$$

(To avoid issues with people who share the same name, the system maintains the face recognition database ID of the person, so we can check that "Joe" is indeed the photographer before performing the personalization.) Besides the photographer, we also personalize dates that match a set of birthdays downloaded from the photographers Facebook account. For example "July 17th" might be transformed to "my birthday." Given the relevant information, a number of other personalizations are possible, such as social relationships ("my brother") or locations ("at my house.")

4. User-Interface

The user-interface (UI) is one of the most important aspects of AutoCaption. Even if all the computer vision modules produced accurate results, the first caption might still not be the one the user wants. It is therefore important to have an engaging UI so that the user can quickly perform edits such as switching the order of phrases, and adding or deleting words and phrases to change the caption to their liking. Our user interface has three editing modes:

Selection Mode: The text generator produces a large number of captions. (See Section 3.8.) The user is initially shown the first caption in the set. The user can then quickly cycle through the alternatives by tapping on the caption.

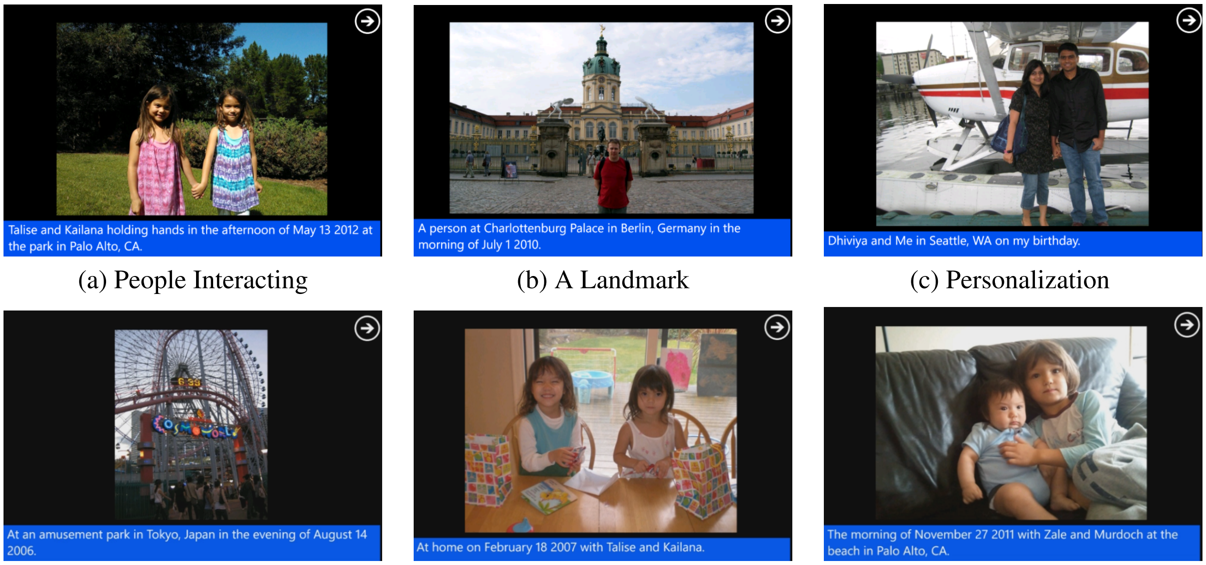
Reordering Mode: Often a user would just like to quickly rearrange the entities in the caption. For example, if the caption is "Talise and Kailana at the beach", the user might want to switch the ordering of the names to "Kailana and Talise at the beach" (without having to delete all the text and then re-enter it.) The user can trigger the reordering mode by tapping and holding on the caption in the UI. In this mode, the different entities that are recognized in the photo (for example, the place, people) are highlighted. The user can simply drag an entity to indicate where they would like it to appear in the caption. The UI then snaps to the closest valid caption. The user can also drag entities to and from a "trash" area. To choose the closest valid caption to snap to, we use a weighted combination of two score. The first score is given a much larger weight than the second.

Entity Overlap Score: We compare the set of entities in the edited caption with the set of entities in each possible caption. The overlap score is the number of entities in the intersection minus the number of entities in the caption that are not in the edited caption.

Distance Score: For each entity in the intersection between the re-arranged caption and the caption, we compute the distance between the position of the entity in the re-arranged caption and its position in the caption. We add these up, giving additional weight to the last moved entity. We also flip the sign to give a measure where more positive numbers are preferable.

Multiple Candidates Mode: The computer vision modules are unlikely to produce the correct results all the time. We therefore need to support user corrections in the UI. Our system supports modules that output the top-N candidates. The face recognition and scene classification modules cur-

[1] We also maintain confidences and ImageRegion information with this entity list (not shown). Confidences is used to order or truncate the list, and ImageRegion can be used to highlight the entity being edited in the UI.


Talise and Kailana holding hands in the afternoon of May 13 2012 at the park in Palo Alto, CA.

(a) People Interacting

A person at Charlottenburg Palace in Berlin, Germany in the morning of July 1 2010.

(b) A Landmark

Dhiviya and Me in Seattle, WA on my birthday.

(c) Personalization

At an amusement park in Tokyo, Japan in the evening of August 14 2006.

(d) No People

At home on February 18 2007 with Talise and Kailana.

(e) No GPS

The morning of November 27 2011 with Zale and Murdoch at the beach in Palo Alto, CA.

(f) Erroneous Scene

Figure 5: Example outputs: (a) An example with two people interacting. (b) An example with a prominent landmark. (c) An example where the caption is personalized to the user. (d) An example of a caption when no people are present. (e) An example of a caption for a photo with no GPS. (f) An example of a caption with a scene classification error.

rently output the top N=3 candidates. Tapping on an entity in the Reordering Mode enters the Multiple-Candidate Mode. In this mode, the top 3 candidate are displayed above the caption analogously to how suggestions are shown for the built in keyboard auto-correction mode. The user can choose the correct one from the list. There is also an option to "add new" which brings up the keyboard for the user to enter a new candidate. Whenever the user edits an entity in this manner, the details of the edit are passed back to the appropriate processing module. In the face recognition module, these provide face tags for that user. For modules such as scene classification, the edits provide more generic training data that can be used to retrain the module.

5. Results

AutoCaption integrates several computer vision algorithms. We focus on illustrating the end-to-end system (Section 5.1), a quantitative assessment of the quality of the captions (Section 5.2), and the user interface (Section 5.3). Readers are encouraged to look at the references in Section 3 for evaluation of the individual modules.

5.1. End-To-End System

The smartphone client is a Windows Phone 8 application running on the phone. The cloud service is a Windows Azure [17] system running two large instances of a web-role and worker-role. The webrole handles the web requests to/from the clients and the workerrole handles all the processing including the running of the modules. The cloud service is written in C# whereas the computer vision modules are written in C++. The whole process including uploading the photo and downloading results on the phone takes about 3-5 seconds with the phone on a WiFi network. Video *1_endtoend_system.mp4* [1] shows the entire workings of the end-to-end system including caption generation, selection, reordering and multi-candidate replacement.

Figure 5 includes a few outputs from our system. The supplemental video *2_caption_generation.mp4* [1] shows a few different examples of caption generation. Figure 5(a) contains two people holding hands. Looking at the caption, we can see the face recognizer correctly identifies the number of people and their identity. The scene classifier correctly classifies the scene as "park". The proxemics module is able to correctly identify the interaction as "holding hands". The timestamp and location information are also provided by the respective modules. Figure 5(b) is a photo of a person in front of the Charlottenburg Palace in Germany. The landmark recognizer correctly identifies the landmark. The person is not in the face recognition database for this user and so no good match is found. The caption therefore does not contain the name of the person. Figure 5(c) contains an example of "personalization." The name of the user is replaced with the pronoun "me" and the date (which is the birthday of the user, downloaded from their Facebook account) is replaced with the phrase "my birthday."

Figure 5(d) contains an example with no people. Even without any people, the caption is still interesting. Similarly, Figure 5(e) contains an example of a photo without any GPS. While the location information in GPS can en-

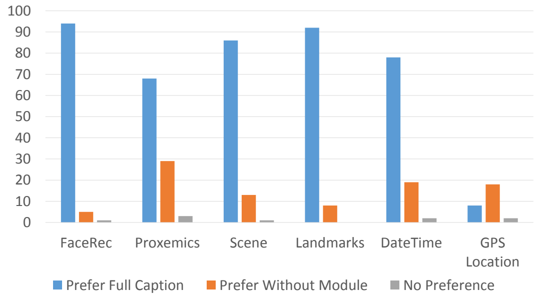


Figure 6: To assess the relative importance of the modules to the complete system, we used Amazon Mechanical Turk to assess whether users preferred the full caption or one with each module in turn removed. We plot the percentage of times that the turkers preferred the full caption, the caption with the module removed, or had no preference.

rich a caption, it it not essential for a compelling experience. Figure 5(f) contains an example with errors. The scene classifier returns the wrong class and the proxemics module misses "holding." The caption is still acceptable, and the user can quickly correct the scene error in the UI.

5.2. Quantitative Assessment of Captions

We assembled a test set of 159 images and used Amazon Mechanical Turk to rate the captions on a 4 point scale: "1=a very good caption," "2=a good caption," "3=an OK caption," and "4=a bad caption," We used captions which include results from all of the modules that returned results (the full captions), and had 15 turkers rate each image-caption pair. Overall, the turkers responded very positively. Of all $159 \times 15 = 2385$ ratings, 49.6% were "very good," 32.3% were "good," 12.6% were "ok," and 5.4% were "bad." The images with the most amount of "bad" ratings are mostly images with erroneous recognition results.

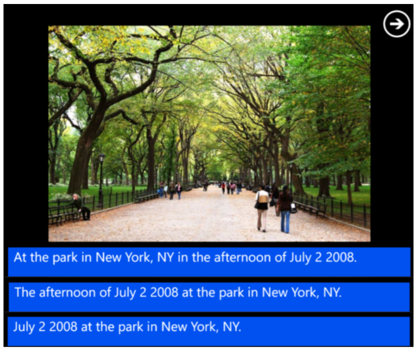
To assess the relative importance of the various modules, we also asked turkers to compare the full captions with ones obtained by removing one of the modules from the system. We had 12 turkers make an assesssment for each image and pair of captions. The results are included in Figure 6. The results show that the performance is degraded the most when the face recognition, landmarks, or scene classification module is removed.

5.3. Caption Selection and Editing

Figure 7 shows an input image with the first 3 captions that the system cycles through as the user taps on it. See the supplemental video *3_caption_selection.mp4* [1] for details.

Figure 8 shows the user moving entities around in the Reordering Mode. After each drag-and-drop operation, the caption caption snaps to the closest valid one. See the supplemental video *4_text_reordering.mp4* [1].

Figure 9 shows the user selecting one of the multiple candidates for the recognized scene. In this case, the point-of-interest (POI) query in the fusion module (see Section 3.7) returned the result "Disneyland Resort." See the supplemental video *5_multiple_candidates.mp4* [1] for an illustration.


At the park in New York, NY in the afternoon of July 2 2008.

The afternoon of July 2 2008 at the park in New York, NY.

July 2 2008 at the park in New York, NY.

Figure 7: The top 3 captions for an image that the phone client cycles through as the user taps on the caption.

6. Conclusion

We have presented a system that automatically generates captions for photos taken on a smartphone. The system operates by uploading the photo to a cloud service, which runs a number of processing modules in parallel. The results are fused, a large number of possible captions generated and then personalized. A key element in our system is the user interface which allows the user to obtain the caption they wanted. The user can (a) tap on the caption to iterate through the set of captions, (b) interactively edit the caption by reordering, adding, or removing phrases, and (c) select from multiple candidates to replace a word or phrase.

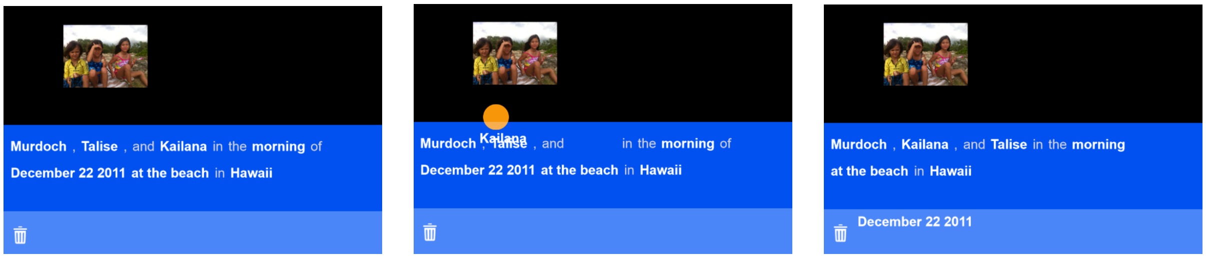
As future work, we wish to add more modules to expand the recognition capabilities of the system. These include weather, activity, and clothing recognition, etc. As the number of modules increases, the fuser becomes more important to facilitate more sophisticated reasoning. Our personalization module could also be extended to include social relationships, calendar information, and personal location information (e.g. "at home"). Another important future direction is to learn more from the feedback provided by a user as they edit a caption. We can learn user stylistic preferences or use the feedback to improve the modules.

References

[1] Project page. http://go.microsoft.com/fwlink?linkid=392018. 6, 7

[2] A. Bergamo, S. Sinha, and L. Torresani. Leveraging structure from motion to learn discriminative codebooks for scalable landmark classification. In *CVPR*, 2013. 1, 2, 4

[3] Bing. Bing Maps Locations REST Service. http://msdn.microsoft.com/en-us/library/ff701710.aspx. 4



Murdoch , Talise , and Kailana in the morning of

December 22 2011 at the beach in Hawaii

Murdoch , Kailana Talise , and in the morning of

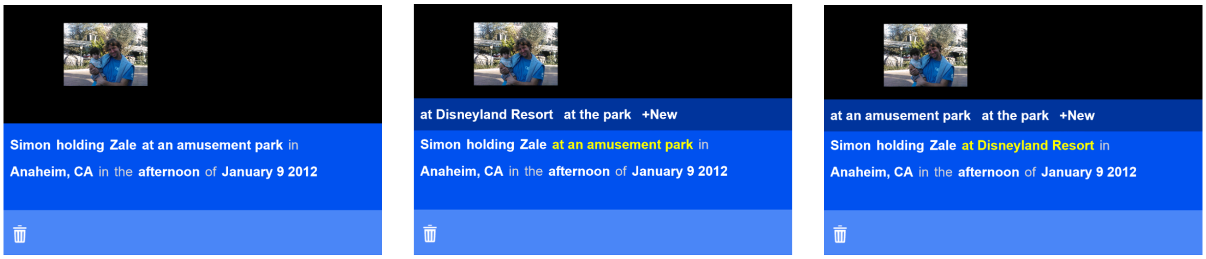
December 22 2011 at the beach in Hawaii

Murdoch , Kailana , and Talise in the morning

at the beach in Hawaii

December 22 2011

Figure 8: The Reordering Mode allows the user to quickly move around and delete entities. (Left) The draggable entities are highlighted when the user enters the mode. (Middle) The user drags the word "Talise" before "Kailana". (Right) The caption after the reordering and the phrase "December 22 2011" has been moved to trash.



Simon holding Zale at an amusement park in

Anaheim, CA in the afternoon of January 9 2012

at Disneyland Resort at the park +New

Simon holding Zale at an amusement park in

Anaheim, CA in the afternoon of January 9 2012

at an amusement park at the park +New

Simon holding Zale at Disneyland Resort in

Anaheim, CA in the afternoon of January 9 2012

Figure 9: Multiple Candidates Mode: (Left) The entities are highlighted as before, each of them being tap targets. (Middle) Tapping on the "at an amusement park" entity brings up the choices at the top. The point-of-interest (POI) query in the fusion module returned "Disneyland Resort". See Section 3.7. The scene classifier returned "at the park" as the second best choice. (Right) Tapping one of the choices swaps it out with the entity in the caption.

[4] Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning-based descriptor. In *CVPR*, 2010. 1, 2, 3

[5] P. Chen. The entity-relationship model: Toward a unified view of data. *ACM T. on Database Systems*, 1:9–36, 1976. 3

[6] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Statistical Learning in Computer Vision*, 2004. 4

[7] A. Farhadi, M. Hejrati, M. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *ECCV*, 2010. 2

[8] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005. 3

[9] Google. Places API. https://developers.google.com/places. 4

[10] A. Gupta and P. Mannem. From image annotation to image description. In *NIPS*, 2012. 2

[11] E. Hall. A system for the notation of proxemic behavior. *American anthropologist*, 65(5):1003–1026, 1963. 3

[12] H. Jegou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311, 2010. 4

[13] G. Kulkarni, V. Premraj, S. Shar, S. Li, Y. Choi, A. Berg, and T. Berg. Baby talk: Understanding and generating simple image descriptions. In *CVPR*, 2011. 2

[14] P. Kuznetzova, V. Ordonez, A. Berg, T. Berg, and Y. Choi. Collective generation of natural image descriptions. In *Proc. of the Association for Computational Linguistics*, 2012. 2

[15] I. Langkilde and K. Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the Association for Computational Linguistics*, 1998. 5

[16] Y. Li, D. Crandall, and D. Huttenlocher. Landmark classification in large-scale image collections. In *ICCV*, 2009. 4

[17] Microsoft. Windows Azure. http://www.windowsazure.com. 2, 6

[18] M. Mitchell, X. Han, J. Dodge, A. Mensch, A. Goyal, A. Berg, K. Yamaguchi, T. Berg, K. Stratos, and H. Daume. Midge: Generating image descriptions from computer vision detections. In *Proc. of the 13th Conf. of the European Chapter of the Assoc. for Comp. Linguistics*, 2012. 2

[19] V. Ordonez, G. Kulkarni, and T. Berg. Im2Text: Describing images using 1 million captioned photographs. In *NIPS*, 2011. 2

[20] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 1, 2, 3

[21] S. A. J. Winder, G. Hua, and M. Brown. Picking the best daisy. In *CVPR*, pages 178–185, 2009. 4

[22] W. A. Woods. Transition network grammars for natural language analysis. *CACM*, 13(10):591–606, Oct. 1970. 5

[23] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 1, 2, 3, 4

[24] Y. Yang, S. Baker, A. Kannan, and D. Ramanan. Recognizing proxemics in personal photos. In *CVPR*, 2010. 1, 2, 3

[25] Y. Yang, C. Teo, H. Daume, and Y. Aloimonos. Corpus-guided sentence generation in natural images. In *Empirical Methods in Natural Language Processing*, 2011. 2

[26] C. Zhang and P. Viola. Multiple-instance pruning for learning efficient cascade detectors. In *NIPS*, 2007. 1, 2, 3