Development and Deployment of a Large-Scale Flower Recognition Mobile App

Anelia Angelova[*]
NEC Labs America
10080 North Wolfe Road
Cupertino, CA, USA
anelia@nec-labs.com

Shenghuo Zhu
NEC Labs America
10080 North Wolfe Road
Cupertino, CA, USA
zsh@nec-labs.com

Yuanqing Lin
NEC Labs America
10080 North Wolfe Road
Cupertino, CA, USA
ylin@nec-labs.com

Josephine Wong
Bio-Rad Laboratories
Pleasanton CA, USA
josephine_wong@bio-rad.com

Chelsea Specht
Department of Plant and Microbial Biology, UC Berkeley
Berkeley, CA, USA
cdspecht@berkeley.edu

ABSTRACT

Today's major image search engines, which focus on search-by-image-content, work by matching and retrieving of images that are already available on the web. With the proliferation of user generated content, especially from mobile devices, there is a need to develop applications which are more content-aware, i.e. can understand and recognize what is in the image, and which can handle the deteriorated quality inherent to user images uploaded from mobile devices.

In this paper we describe a mobile phone application intended to automatically recognize flower images taken by users and classify them into a set of flower categories. The app is served by a web-scale image search engine and relies on computer vision recognition technology. The mobile phone app is available free to users, and as a web interface.

We share experiences about designing, developing, and deploying such a system in practice. We discuss practical aspects of applying the object recognition technology, including runtime, scalability, and data collection which is crucial for such data-driven application. More specifically, we describe a large-scale dataset which was collected in a number of stages to be as close as possible to the data distribution the actual users of the app will encounter. We further describe our strategy for collecting user feedback, which we view as an opportunity to improve the server-side algorithms and datasets.

We envision that these issues will be encountered in similar applications, e.g. for recognition of plants, mushrooms, or bird species, all of which have practical importance for users, and we hope this work will be also useful for developing other types of applications.

To our knowledge, this is the first mobile app which can automatically recognize as many as 578 species of flowers, and which is available for free to users. The flower dataset that serves the application, specifically collected for this recognition task, is the first and largest in its scale and scope.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Online Information Services; I.4 [**Image Processing and Computer Vision**]: Applications; I.5 [**Pattern Recognition**]: Implementation

General Terms

Experimentation, Performance, Algorithms

Keywords

Fine-grained categorization, visual object recognition, large-scale datasets, crowd-sourcing, mobile applications

[*]Contact author

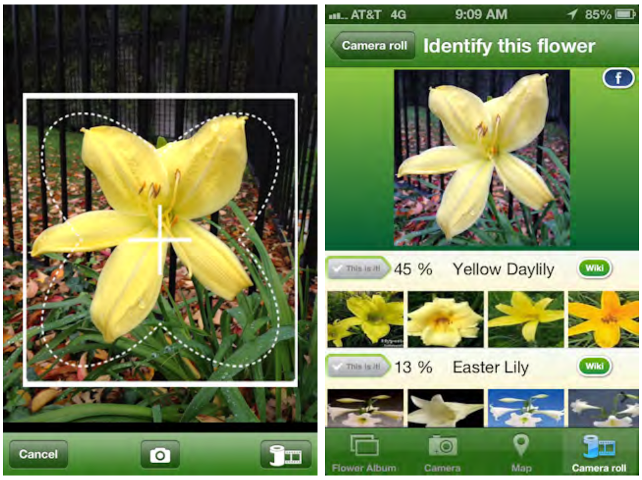


Figure 1: Snapshots from the flower recognition app for iPhone. An image of a flower is taken (left), our recognition engine processes the image and returns top five choices of possible flower classes that match the input flower best (right). The user can then make the best selection.

1. INTRODUCTION

This paper presents an end-to-end visual recognition system whose task is to automatically identify the species of the flower in an input image. It is available as a mobile

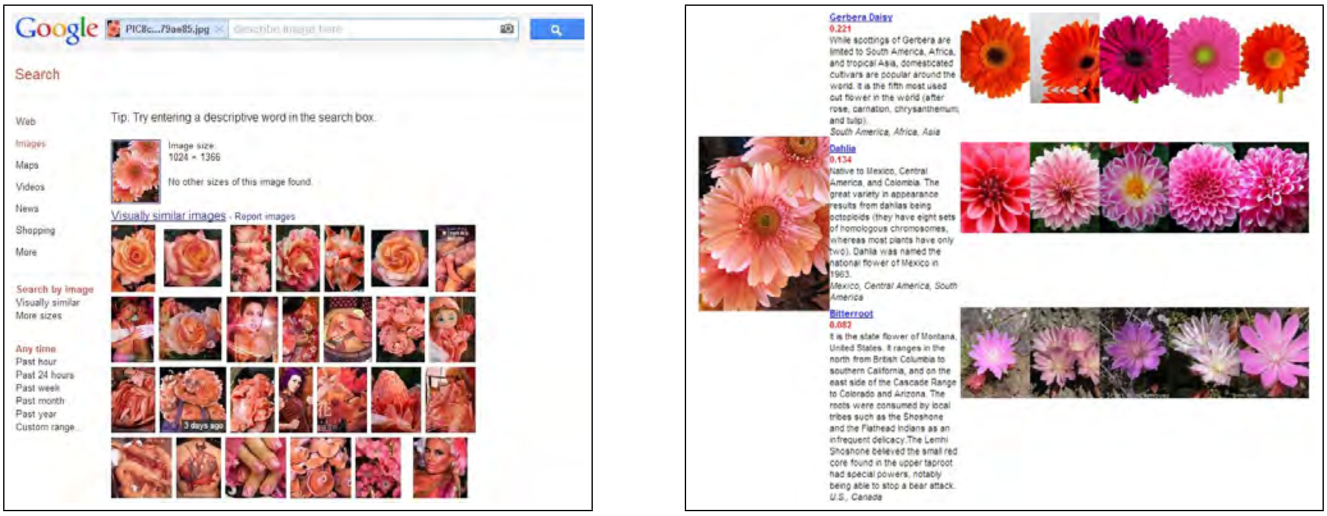


Figure 2: One of the discriminating characteristics of our application is that it can generalize within a class and can ignore irrelevant characteristics, such as color. Results of recognizing an image containing a Gerbera flower by Google Search-By-Image service (left), in which all returned images match the color distribution but no Gerbera flower is retrieved, and our system (right) which correctly identifies the flower species of the query image and also returns somewhat similar categories as choices 2 and 3. The web-based version of our application is shown here.

phone app[1] (Figure 1) and as a web-based service[2]. While some reference applications for flowers already exist (e.g. Flower Pedia, Audubon Wildflowers, What Flower, etc), our app has the unique capability to automatically recognize the flower from an image. This is of much value to interested users because no previous application can accomplish such a task, especially from a mobile device. The system is capable of recognizing 578 different flower species and relies on web-scale image dataset that has been collected from various sources and has been carefully labeled by a team of expert botanists. Our recognition system is specifically designed to handle user-generated images collected 'in the field' by mobile phone users of various devices. The latter is important since user-generated imagery may significantly vary in quality and will differ from the images of flowers that are encountered on web-search engines. We describe details of designing the system, developing the algorithms, as well as, collecting the dataset that can support this capability.

Recognizing among different types within the same general class of objects is called fine-grained classification [13, 22, 28], and is one of the active areas of research in computer vision. Unlike the so called basic-level recognition, which refers to recognizing whether an object is a flower, or a cat, or a chair, fine-grained recognition works within a single base-level category and aims to recognize among its subcategories, such as different species of flowers or different species of birds, plants, etc. Fine-grained classification requires knowledge and skills that only trained experts can provide. In the case of flower recognition, only an expert botanist, or a person specifically trained to identify and discriminate between flower species can do the task, and sometimes not without using further references [11], or following a complex identification procedure (see Section 4.3). Since very few people can successfully recognize among a large number of species, this makes the automatic fine-grained recognition application very valuable to non-experts.

General object recognition, and more specifically, fine-grained recognition, are still problems that are far from being solved algorithmically. Large-scale image retrieval systems [3, 4, 7, 25, 29] have been quite successful at retrieving a duplicate or near-duplicate images [7, 4], similar images [4], or extracting images 'by sketch' [25]. However, they do not have the capability to generalize within a class, e.g. to recognize that some object is a vacuum cleaner, for example, and do not have the capabilities to discriminate between very related but different classes of the same super-class, as is the case with flowers. Furthermore, the computer vision recognition technologies that are available on the web are mostly targetted towards retrieval of web content, rather than user generated 'field' images. With the advent of social media and mobile phones, naturally, the demand for recognition of user-generated images is increasing.

Another differentiating capability between our flower recognition technology and other available image search or image retrieval systems, e.g. Google's Search-By-Image service [4] is that our system is specifically targeted towards recognition, in our case of flower species. That is, our system is trained to identify features that generalize within a class, whereas other systems are more general and very often return irrelevant results (i.e. non-flowers) or match the input flower image by color distribution or other general visual features, rather than retrieving flowers from the same class. Figure 2 shows an example search of an image of a Gerbera flower by Google Search-By-Image service [4] and by our system (which can be found at http://mobileflora.heroku.com). As seen, although many images of Gerbera flowers are available in Google's image corpus, not a single Gerbera flower was retrieved. Instead our system is capable of recognizing

[1]The app is developed for iPhone and is called 'Mobile Flora'. It is currently under invitation-only beta testing, and will soon be available for general use. Please email the contact authors to send you an invitation.

[2]The web-based version of the app is available at http://mobileflora.heroku.com

the type of flower and providing related images. We note here that this example shows typical retrieval differences, especially for flower images that do not have a near-duplicate counterpart on the web, e.g. images in personal collections, or obtained by mobile devices, etc.

2. FINE-GRAINED RECOGNITION CHALLENGES

Fine-grained classification has its unique challenges because it aims to recognize among classes that will typically be considered as a part of one large class in standard visual object recognition tasks.

The main inherent challenge is that some species have very fine differences which are hard to notice for the common eye. For example, the categories that need to be discriminated among each other can be very similar and would require careful examination to determine the correct class. Figure 3 shows examples from three different flower species: *Matricaria chamomilla* (chamomile), *Bellis perennis* (common daisy), and *Leucanthemum vulgare* (ox-eye daisy). As seen these three categories are very similar to each other.

Another challenge in a fine-grained recognition task is the variability in viewpoints, colors, and scales, that are encountered within the same class. These variabilities are mostly problematic to an automatic system rather than a human observer. Many flowers can have different colors but a person can very easily factor out that invariance, which is not necessarily true for an algorithm. Similarly, it is very easy for a person to understand if the image contains a close-up shot of a single flower or multiple small flowers of the same type, but this problem can be quite challenging for an automatic vision system.

Other potential issues specific to an automatic recognition system is the background, which can serve as a distractor, e.g. when the system assumes it is part of the flower, occlusions over the flowers, and others.

Figure 3: One of the challenges for fine-grained recognition is that some classes can be very similar to one another and only experts can discriminate them. This makes an automatic recognition system very valuable to non-experts. Each row shows a different flower species.

3. FLOWER RECOGNITION APPLICATION

Our system and the mobile phone app are intended to recognize the image of a flower of a blooming plant or tree. Figure 7 of the Appendix shows the main functionalities of the app.

The entry point of the app is to take a picture of a flower and submit it for recognition (Figure 7a, and also Figure 1). While taking the picture, a flower contour is overlayed to indicate that the flower is expected to take the full image and be within the boundaries of the image. These are only guidelines and, as our users will discover, the app is much more powerful and is capable of recognizing a flower that takes significantly more or less of the area that is desirable. Multiple flowers of the same type (also called 'cluster flowers') are also not a problem for recognition.

Results of the top five choices of potential classes that the input image can belong to are shown to the user (Figure 7b). Since some classes may be very similar, as seen in Figure 3, we found it much more useful to provide several choices, instead of the top one only.

After the user is offered a set of options of what the input flower could be, they can examine and compare the images from the proposed categories (Figure 7c) and make a decision which class the flower belongs to (Figure 7d). Here, in our application, we have taken advantage of the fact that people may not be very good at recognizing a particular flower, especially among such large variety of 578 different species, but when presented with a set of options they can be pretty good at verifying which one it is.

More results are available upon request, but we provide top ten choices at most to avoid showing too many irrelevant results, since the more results shown the more likely they will be very different from the input category. If the top ten results do not provide satisfactory choice, the users can give a name to the flower, if they like (Figure 7e). Otherwise the flower is stored in their album as an 'unknown' category. An example user album is shown in Figure 7f.

For each flower, which has been identified, we provided its common name and a short description about this flower, e.g. where it originated from, its uses, and some trivia information (Figure 7g). We also provide wikipedia link that describes this flower. Additionally we provide information about regions world-wide where the flower grows (Figure 7h). All this information has been compiled by our team of expert biologists. We further plan to provide information about how to grow the flower for interested users, as well as, more refined information about the regions where it grows. We also show the images that the user has collected for a flower category (Figure 7i).

4. FLOWER DATASET

For the purposes of developing the flower recognition application, we have collected a large-scale flower dataset. The dataset contains 578 different flower species and over 250,000 images. This is the only web-scale flowers dataset we are aware of. While we have compiled a seemingly modest number of images (250,000 compared to billions of images for general image search engines), our dataset has exhausted the

Figure 4: Example images from different classes from the large-scale 578 flower species dataset. A large variety of flower classes, as well as, intra-class variabilities, inter-class similarities, and changes in flower scales are available in this dataset.

web resources of available flower images, and for some rare species we could not find a sufficient number of images. Furthermore, this is the largest dataset for flower recognition, and also the largest of its kind among any other fine-grained recognition datasets we are aware of [13, 20, 22, 24, 28]. All previous datasets contain at most 200 different species [28] and the largest known flower dataset contains 102 classes and 8189 images [22]. Also compared to previous flower datasets, our dataset is targeted to be used 'in the field' by actual users, rather than for experimentation only. Thus our dataset contains a lot of intra-class variability, changes to scale, and quality variability than any previous dataset. Figure 4 shows example images from different classes, and as seen the images are much more realistic, which makes the recognition task harder. On the other hand, this data is much closer to the data distribution an actual user will encounter as they collect images by their phones and submit them to the app.

Collecting, defining and organizing the dataset for this application was a very challenging task, which is why we describe those efforts here.

4.1 Defining the set of classes

The first important question to address is to determine which set of classes the system is going to recognize. While there are many flowering plants that potentially exist worldwide [6], the largest number of species that an expert botanist is able to recognize is estimated to be about 30,000. Naturally, here we are striving towards recognizing as many flower species as possible, so as to achieve a maximal coverage of flowers that users may encounter. However, some real-life limitations exist. One issue is that collecting and curating images from such a dataset would be prohibitive, especially since only flower experts can do the identification and curation (i.e. labeling, or placing into categories, and giving the correct scientific name). A more serious issue is that there are no sufficient number of images for the less common flower categories. Furthermore, such a large number of classes is also pushing the boundaries of what we can do technologically to recognize automatically.

To address this problem, we focused on the most common flowers which have sufficiently large coverage over all possible flowers, and ignored the 'long tail' set of flowers. More specifically, our goal is to select the most frequent classes which cover 90% of all flowers world-wide. Since, it is not clear how to estimate such a set, we defined the classes in two steps. First, our team selected the set of top 500 most popular flowers in the world. We further collected images from the web when typing general flower-related terms such as 'flowers', 'desert flowers', 'mountain flowers', 'wedding flowers', 'spring flowers', 'exotic flowers', etc. The latter was an attempt to retrieve 'all possible' types of flowers and their natural distribution. In this way we could only capture the flower distribution that occurs on the web, which reflects areas of the world with at least some technology penetration, rather than the actual distribution which is unknown, but this was a good approximation, given that the users of our mobile app will also be representative of technologically advanced regions. After collecting such a set, the biologists on our team labeled each image and identified flowers that are not included in our pre-defined set of classes. We then included the next most common flower species, until their coverage reached 90%. As a result of that we obtained a set of 578 species of flowers.

This, of course, is only approximately capturing the full coverage of our application, and only after actual users start to use the application, will we know if our coverage is sufficient. Naturally, our set of classes can evolve with usage of the app, and this can improve the usefulness of the application, as well.

4.2 Data collection

Since we are developing user-based application which is going to work with user provided data, we do not necessarily have such data to begin with. Here we describe how we bootstrap the mechanism for data collection so that to imrove the data quality for our app. Initially we use images collected from the web (Section 4.2.1), then we collected user data by crowd-sourcing (Section 4.2.3), finally, after the application is launched we can collect user-uploaded images (Section 7). In a sense the dataset evolves until we can serve the users best.

4.2.1 Web crawling

Initially we collected data from the web, by automatically downloading images corresponding to each flower category by keyword search on Bing and Google search engines. We typically used the official name of the flower (e.g. 'matricaria recutita') and one or more of its common names (e.g. chamomile). Online search engines were very helpful to provide flower images, however since the search is done by keywords, a lot of the returned images are irrelevant to the flower category and need to be filtered out.

4.2.2 Near-duplicate elimination

Near-duplicate elimination is a process which removes examples that are visually very similar or almost identical. Typically near-duplicates are images of the same object but one of them has been modified in some way, e.g. by rescaling, compression, by adding noise, or by adding a logo or modifying a part of the image. Near-duplicate elimination is needed because our application shows a set of images for each proposed class and the user can look at them and decide if the class is the best match to the query flower. In this situation, we really want to show diverse images. Secondly, near-duplicate images have almost identical feature representations and are not useful for training or testing of our algorithm.

For near-duplicate elimination we have extracted HOG features [10] in the whole image. These features are then encoded in 8196-dimensional global feature dictionary using the Local Linear Coding method [26]. All the features in the image are combined by a global max-pooling [26] to obtain a final 8196-dimensional representation similar to [21]. This is essentially a signature-like summarization of the whole image, so two images which differ only due to rescaling, some noise or re-compression will likely have very similar signatures. Images whose signatures differ by less than a prespecified threshold are eliminated as duplicates and a representative single image is kept in the final collection. Note that the feature representation is derived from the more complex representation, presented later in Section 5 which is used for final classification into different classes.

Figure 5: Example images of a rose returned by Bing Search Engine (top panel), example images of a rose obtained by a mobile phone (bottom row). As seen, the quality of these images is very different, so images obtained online will not be sufficient to cover the test distribution of mobile images.

We are not aware of any other system using the method for near-duplicate detection described above and we chose it for simplicity and ease of application. There are other near-duplicate detection methods that have been proposed and used in practice [17, 19], but we found the method we have used as sufficient for our goals.

4.2.3 Collecting data via crowd-sourcing

Since our data were collected from web sources, naturally, their distribution is very different from the distribution of images provided by mobile phone users. Figure 5 shows example image for a 'rose' that are returned by Bing search engine [2] and other examples which are collected 'in the field' by a mobile device. As seen those are very different. Therefore, we needed to collect extra images which are closer to the user-generated content by mobile devices.

For that purpose we used a crowd-sourcing service for data collection. Apart from obtaining images closer to the images by our potential app user, we are aiming to get diversity both in geographical locations and also from user to user (e.g. a variety of user demographics, photography skill levels, etc.).

Mobile phone users on a crowd-sourcing service were instructed to capture flower images. We gave fairly loose guidelines so that they maximally match the users of the app, who will not be given instructions but will be guided by the UI to take flower images for recognition. Most of the users did what is expected, but for some users we had to refine and reinforce the instructions mostly for the sake of collecting useful data for our algorithms. In general, what we learned in the process was that we need to set up fairly simple and straightforward requirements for crowd-sourcing users.

Overall we are very pleased with the dataset collected from crowd-coursing: some of the agents paid to collect images were very industrious and creative and provided images of exotic and rare plants, which they have collected from vacations or by visiting botanical gardens and arboreta. Figure 6

Figure 6: Example images collected by different users via a crowd-sourcing service. As seen, the users provided very good variety of flower species from different geo-locations but also images vastly varying in quality. Each row corresponds to a separate user. Real mobile phone users were crucial in our experience, because their images reflect better the distribution of images that our app users are likely to upload.

shows images that the users collected, which shows us the diversity one can get thanks to crowd-sourcing. Naturally, we also have some images that will be considered of 'poor quality' in any other dataset, e.g. with shadows, blur, etc., but those are very useful for our application, because they represent images that mobile phone users may take.

4.3 Labeling

A very demanding task is that of labeling the images with their correct flower class. We note here that only an expert botanist, or a person specifically trained to identify and discriminate between flower species can do this task. For example, Amazon's Mechanical Turk [1] users would generally not be able to provide such services because it is crucial to recognize subtle differences between species. Also note that labeling is a painstaking manual process, because each image has to be examined to verify it belongs to a specific category. In some cases the identification process can be very complicated in which the botanists answer a sequence of questions until the flower is identified [11]. While there is no established protocol, which makes the task so challenging, the botanists try to determine if it is a flower or an inflorescence (branch of flowers, e.g. the 'flower' we see in a Heliconia or a sunflower), the number of petals, stamens and carpels; whether the petals (stamens or carples) are fused or free, the type of symmetry of the floral whorls, etc. In order to label all images that belong to a certain class, the biology experts in our team followed this complex and demanding procedure for flower identification.

In addition to labeling, the dataset had to be cleaned to remove images of non-flowers or of artificial flowers (e.g. drawings, cartoons, clipart, etc). Although the process did not require expert knowledge, it is very tedious. Initially we have collected more than 3.5 million images which had to be filtered.

4.4 Discussion

As is clear from this section, collecting and labeling this data was a major accomplishment of our team, and is still a work in progress. This dataset, with its 250,000 images across 578 flower categories, is the largest we are aware of. For very common categories there are many images available, but we have limited the number we store so that the dataset is more balanced. For the categories which are very rare, although we attempted to collect several thousands images that contain the respective categories, much fewer results were available online. Moreover, the number of actually useful images (the ones that contain a flower or flowers of this category only) was very small. The images we collected, about 250,000 may seem relatively small compared to web image corpora, but with this dataset we have exhausted what search engines can return, e.g. for very rare categories. Future work needs to address collecting examples for these rare categories. User interactions with the app will serve as a means to redefine and expand the flower dataset.

5. RECOGNITION ALGORITHM

A key to the application presented is its recognition engine and algorithm, which we describe in this section.

The algorithm is based on the one of Lin et al. [21]. The algorithm starts by feature extraction on the input image: we first extract HOG features [10] at 4 different levels. These features are then encoded in 8196-dimensional global feature dictionary using the Local Linear Coding method of Wang et al. [26]. The global dictionary is created by clustering of low-level patches generated from a set of natural images which are unrelated to the classification task at hand. Encoding by using global dictionaries have been shown to be very advantageous for visual recognition [26]. After that, a global max pooling of the encoded features in the image is done, as well as, max poolings in a 3x3 grid of the image [21]. Combining all the poolings we obtain a vector representation which is 10 times the initial dictionary dimensionality, or about 80,000-dimensional feature vector.

This feature representation is then classified into individual 578 classes by a multi-class classification algorithm, such as Support Vector Machines (SVM). Our classification pipeline uses the 1-vs-all strategy of linear classification and we used a Stochastic Gradient Descent version of the algorithm [21] for these purposes since our dataset of 250,000 images does not fit on a single machine and standard SVM implementations [12] cannot be applied.

This recognition algorithm is selected because of its performance capabilities (see Section 6) and its runtime (for this app we need real-time performance because in addition to the recognition time, the user experiences additional time latencies for uploading the image and downloading and visualizing the results). We also preferred it for being general and easy to use.

Note that the recognition algorithm has been previously used for general object recognition (e.g. recognize a flower from a piano from a dog, etc), but here we have applied it for recognition into very closely related objects as is the case of different types of flowers.

6. TESTING AND DEPLOYMENT

We evaluate here the performance of our recognition algorithm. The results of the classification performance are shown in Table 1. We computed the accuracy of the test set for top 1, top 2, ..., top 10 returned results. As seen we can achieve 52.3% for top 1, 75.2% for top 5, and 82.3% for top 10 results. Translating this to the context of the functionality of the app, we found it useful to show top 5 suggestions per query image, and then allow the user to expand to top 10 suggestions. That is, we can hope to provide the correct answer among our suggestions in more than 82% of the time.

We tested our recognition on another flower dataset, Oxford 102 flowers [22], which has been widely used in the computer vision community. This is done as to see how the performance of our algorithm compares to other known methods in the literature, since we could not test all other algorithms on our large-scale dataset (since most algorithms are not publicly available, or they are not adequate for multi-machine parallel computations). As seen in Table 2, our algorithm achieves state-of-the-art performance, and is even a little better than the best known algorithm for this dataset. Apart from performing very well, it has other advantages, e.g. it is very general and easy to use. We want to note that better performances for the Oxford dataset have been reported [9], but they require segmentation of the image which is still computationally inefficient and cannot be used directly since our app's response should be real-time.

6.1 Practical considerations in deployment

Runtime. One important thing to consider is runtime. Apart from the time needed to upload the image and download the results, the most time is consumed in the recognition algorithm itself. Currently our recognition algorithm takes about 1-2 seconds (it typically works within 1 second and only very rarely needs more time). The upload and download times are relatively small since we resize the image before uploading it and download only image thumbnails initially. Larger size images are downloaded in the background to keep the latency small. The current response time is adequate for our application, we believe, but we continue to improve its efficiency.

Table 1: Accuracy of our recognition algorithm for top 1 to top 5, top 7 and top 10 retrieved classes. As seen for top 10 we achieve 82 percent correct.

Top 1	Top 2	Top 3	Top 4	Top 5	Top 7	Top 10
52.3	63.4	68.9	72.4	75.2	78.9	82.3

Table 2: Accuracy of our recognition algorithm, compared to other baseline methods on the smaller Oxford 102 flowers dataset.

Method	Accuracy (%)
Kanan and Cottrell [18]	72.0
Nilsback and Zisserman [22]	72.8
Ito and Cubota [16]	74.8
Nilsback and Zisserman [23]	76.3
Ours	76.7

Scalability. We have deployed our application on Salesforce's Heroku cloud platform [5]. It is available at http://mobileflora.heroku.com.

Logging. We also considered logging (see Section 7 for more details) so that we can track performance as the app is being used, and gauge whether users often encounter problems with flower misclassification or with the app itself.

Ease of use. We have limited the app functionality to a minimum so that it is easy and self-explanatory to use.

User perception. As any user-facing application, the actual performance numbers do not matter if the user perceives the results presented as inadequate. For example, we could guess correctly the class at position 5, but if the top 4 results are very different visually (e.g. returning a yellow dandelion for an input image containing a red rose), the user's perception will be that the application performance is poor. While our algorithm tends to return visually consistent results as top choices (see Figure 2) thanks to the feature representation we used, more can be done to enforce visual similarity.

7. COLLECTING USER FEEDBACK

We have designed our app to collect information which we are going to use for further improvement.

Usability. We enabled user feedback in which the users can provide free-form comments, asking for features, reporting issues, bugs etc.

Recognition accuracy. We provided the users an option to expand the top 5 choices provided to top 10 choices. As seen from our evaluation (Table 1) the accuracy among the top 10 is 82.3%. We observed that showing more options is counter-productive, because although the chances of showing the correct results increases, a lot more irrelevant results start to be shown. If the flower is not found among the provided choices, the user has the option to specify its name. Both events of expanding the top choices and entering a flower name are recorded and sent to our servers. This feedback is very useful to know for which examples our recognition engine did not do well. Furthermore, since the images that are submitted to our service are stored, we can further evaluate our performance on user supplied images.

Dataset coverage. As already mentioned, we provided the users an option to name a flower which our system does not identify correctly among the top 5 or top 10 choices. The users can give a common name of a flower (or whichever name they chose). This is reflected in their own image collection, e.g. if they take an image of a rose we fail to recognize, they can name the flower 'rose' and it will be assigned to an album with other roses. This provides us feedback that our recognition is incorrect, but also that a flower class may not exist in our dataset. This will help to expand the dataset with new flower species and thus to increase the coverage of flowers our application can provide.

Collecting the above mentioned information allows us to improve the application's functionality, increase the recognition performance, and expand the image corpus, all of which will improve the app's usability. In this way the application can evolve and improve with its use. Note that this is done without collecting any user-identifiable or private information.

8. RELATED WORK

There are several mobile apps that can help the users to identify flowers, e.g. Audubon Wildflowers, Flower Pedia, What Flower, etc., but these applications work as a reference guides without actually providing the capability to automatically recognize and identify the flower from an input image.

A recent mobile app for tree-leaf recognition has been developed. It is called Leafsnap [20] and can recognize among 184 different tree leaves. Although the recognition technology of LeafSnap is very different from ours, we share the common goal of bringing computer vision technology to an actual product that people can use for free and to allow them to do tasks that they would not be able to do without these technologies. The Leafsnap app has attracted a large number of users and has found a number of different applications [20]. We are not aware of any other mobile applications in the fine-grained recognition domain, but clearly they will be very useful, especially for mushroom or plant recognition [14].

Several algorithms for fine-grained recognition have been proposed and applied to various of datasets: e.g. for birds species recognition [8, 13], for flowers recognition [22], and for cats and dogs recognition [24]. These methods have shown great promise experimentally, but have yet to be deployed in practice.

Regarding the recognition technology, there are alternative solutions that achieve state-of-the-art recognition performance, e.g. by applying metric learning [27], exemplar-based matching and retrieval [20], trace-norm regularization optimization [15] and possibly others. We hope that some of these technologies will find applications and be deployed to more user-facing products.

9. CONCLUSIONS AND FUTURE WORK

Applying computer vision and machine learning technologies is one of the areas that provides enormous opportunities to make a difference in people's everyday lives.

In this paper we demonstrate how to apply state-of-the-art computer vision object recognition technology to discriminate between 578 different species of flowers and automatically recognize an unknown flower from an input image. We have developed a mobile phone app that is available for free to users. We discussed all aspects of development and deployment of our recognition technology in practice.

We hope that the experiences shared in this paper can generalize to other recognition tasks and their practical applications. For example, the recognition engine can be applied for recognition of species of birds [28] or mushrooms, or for one of the most challenging tasks, plant recognition [14]. All of these tasks have much practical importance because people often wonder whether a plant is dangerous, what is it, etc.

9.1 Future work

As future work we plan to improve the classification performance and runtime of our algorithm.

One way to improve performance is to move towards hierarchical representation so that the whole image collection is a *taxonomy*. Taxonomic representation can open the door to exploring classification algorithms or cost metrics that do not penalize as much misclassification among very similar classes (e.g. it may not be as problematic which orchid it is as long as we do not confuse it with other non-related flowers). Flower species are naturally organized in a taxonomy, so we can take advantage of that in the future.

Improvements to the dataset will also have impact of the overall quality of the results. As already mentioned, we plan to use user-uploaded images and images from crowdsourcing to improve the coverage of the dataset. Further analysis of our classification results revealed that some classes may be too broad or a number of classes may be too similar. For example, if a class contains too much variance it may make sense to split it and train it separately. And, conversely, if a several classes invoke too much confusion among each other, it may make sense to combine them for training and use a very specialized algorithm to recognize among them.

Another avenue for future work is how to utilize the feedback provided by users. One interesting machine learning problem is how to automatically clean or reconcile the labels assigned by users, given that some may be noisy or not trustworthy.

10. ACKNOWLEDGMENTS

We would like to thank the members of the Plant and Microbial Biology Department at UC Berkeley: Andrew Brown, Kelsie Morioka, Lily Liu, and Crystal Sun who provided the crucial curation and labeling of the final dataset, and Chodon Sass and Ana Maria Almeida who provided curation of an earlier version of the dataset. We thank Marco Gloria who implemented the app. We also thank the numerous participants in the crowdsourcing service who collected diverse flower images, and volunteers at NEC Labs America who helped us with the data collection.

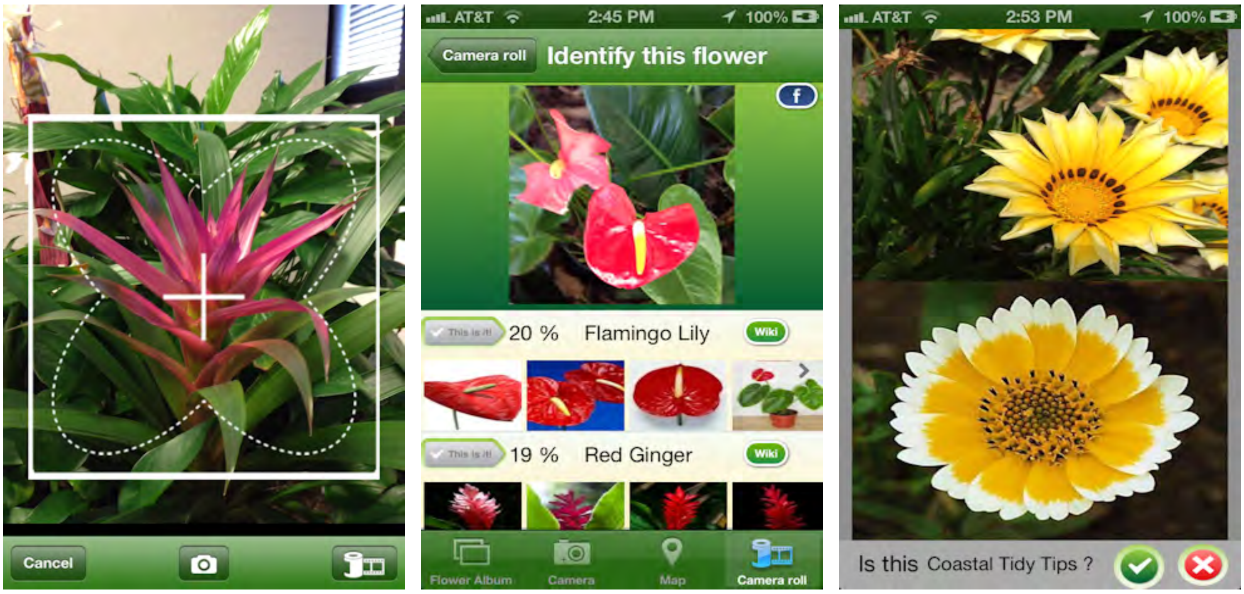
11. REFERENCES

[1] Amazon Mechanical Turk: https://www.mturk.com/mturk/welcome/.

[2] Bing: http://www.bing.com/.
[3] Google Goggles: http://www.google.com/mobile/goggles/.
[4] Google Search-By-Image service: http://www.images.google.com/.
[5] Heroku: http://www.heroku.com/.
[6] The Plant List: http://www.theplantlist.org/.
[7] Tineye Reverse Image Search: http://www.tineye.com/.
[8] S. Branson, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. *European Conference on Computer Vision*, 2010.
[9] Y. Chai, V. Lempitsky, and A. Zisserman. Bicos: A bi-level co-segmentation method for image classification. *International Conference on Computer Vision*, 2011.
[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition*, 2005.
[11] T. Elpel. *Botany in a Day.* HOPS Press, 2000.
[12] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 2008.
[13] R. Farrell, O. Oza, N. Zhang, V. Morariu, T. Darrell, and L. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. *International Conference on Computer Vision*, 2011.
[14] H. Goeau, P. Bonnet, A. Joly, N. Boujemaa, D. Barthelemy, J. Molino, P. Birnbaum, E. Mouysset, and M. Picard. The CLEF 2011 plant image classification task. *In CLEF 2011 working notes*, 2011.
[15] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick. Large-scale classification with trace-norm regularization. *Computer Vision and Pattern Recognition*, 2012.
[16] S. Ito and S. Kubota. Object classfication using heterogeneous co-occurrence features. *European Conference on Computer Vision*, 2010.
[17] A. Jaimes, S.-F. Chang, , and A. Loui. Detection of non-identical duplicate consumer photographs. *Proceedings of 4th IEEE Pacific-Rim Conference on Multimedia*, 2003.
[18] C. Kanan and G. Cottrell. Robust classification of objects, faces, and flowers using natural image statistics. *Computer Vision and Pattern Recognition*, 2010.
[19] Y. Ke, R. Sukthankar, and L. Huston. Efficient near-duplicate detection and sub-image retrieval. *ACM Int. Conf. on Multimedia*, 2004.
[20] N. Kumar, P. Belhumeur, A. Biswas, D. Jacobs, J. Kress, I. Lopez, and J. Soares. Leafsnap: A computer vision system for automatic plant species identification. *European Conference on Computer Vision*, 2012.
[21] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang. Large-scale image classification: fast feature extraction and SVM training. *Computer Vision and Pattern Recognition*, 2011.
[22] M.-E. Nilsback and A. Zisserman. A. automated flower classification over a large number of classes. *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.
[23] M.-E. Nilsback and A. Zisserman. An automatic visual flora - segmentation and classification of flower images. *DPhil Thesis, University of Oxford, UK*, 2009.
[24] O. Parkhi, A. Vedaldi, C. V. Jawahar, and A. Zisserman. The truth about cats and dogs. *International Conference on Computer Vision*, 2011.
[25] C. Wang, Z. Li, and L. Zhang. Mindfinder: Image search by interactive sketching and tagging. *international World Wide Web Conference*, 2010.
[26] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. *Computer Vision and Pattern Recognition*, 2010.
[27] K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Neural Information Processing Systems Conference*, 2006.
[28] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. *Technical Report CNS-TR-2010-001, California Institute of Technology*, 2010.
[29] H. Xu, J. Wang, and S. L. X.-S. Hua. Interactive image search by 2d semantic map. *international World Wide Web Conference*, 2010.

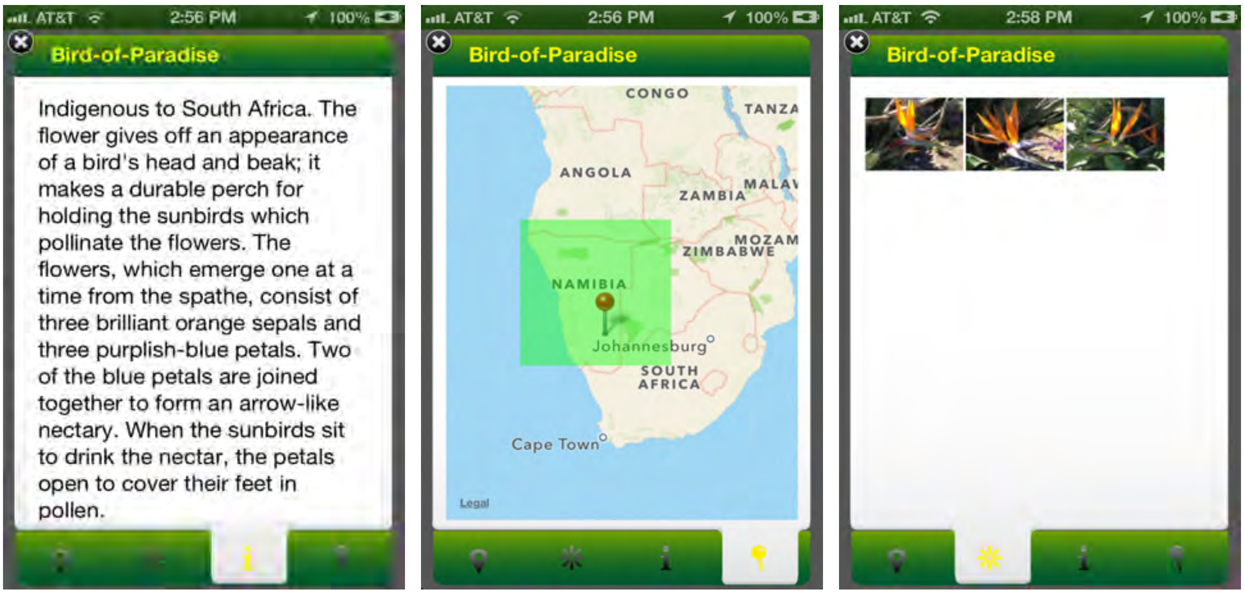
APPENDIX

We show the main functionalities of the app in Figure 7.



(a) Take a picture. (b) Returned suggested classes. (c) View images to compare.

(d) Select a class. (e) Suggest a label, if incorrect. (f) User's collection of flowers.



(g) Information about the flower. (h) Information about where it grows. (i) Images that belong to this class.

Figure 7: A walk-through of the app functionalities.