The Political Visual Literacy App: Real-Time Symbol Recognition for Field Reporting

Ishaan Jhaveri
Tow Center for Digital Journalism, Columbia University
iaj2107@columbia.edu

Svebor Karaman
School of Engineering and Applied Science, Columbia University
sk4089@columbia.edu

Xu Zhang
School of Engineering and Applied Science, Columbia University
xz2437@columbia.edu

Bhaskar Ghosh
Graduate School of Journalism, Columbia University
bg2625@columbia.edu

Nina Berman
Graduate School of Journalism, Columbia University
njb22@columbia.edu

Susan McGregor
Tow Center for Digital Journalism, Columbia University
sem2196@columbia.edu

Shih-Fu Chang
School of Engineering and Applied Science, Columbia University
sc250@columbia.edu

ABSTRACT

The Political Visual Literacy (PVL) app is a graphical-recognition system designed to support and inform the work of journalists as they encounter unfamiliar and evolving graphical imagery in the field. Using a combination of computer vision machine-learning techniques and user feedback, the mobile application allows users to select symbols within photographs on their mobile device and receive information about the meaning of those symbols in real-time. A web-based version of the tool allows newsroom editors and others the use the system on desktop devices. The goal is to provide journalists with more information about rapidly-evolving political symbols in a way that can be seamlessly integrated into their existing workflows.

CCS CONCEPTS

• **Information systems** → **Image search**; • **Applied computing** → **Computer-assisted instruction**.

KEYWORDS

symbol recognition, reverse image search

C+J 2020, March 20–21, 2020, Northeastern University, Boston, Massachusetts

ACM ISBN 978-1-4503-XXXX-X/20/03…$15.00
https://doi.org/10.XXXX/1122334.4556677X

ACM Reference Format:
Ishaan Jhaveri, Svebor Karaman, Xu Zhang, Bhaskar Ghosh, Nina Berman, Susan McGregor, and Shih-Fu Chang. 2020. The Political Visual Literacy App: Real-Time Symbol Recognition for Field Reporting. In *Computation + Journalism 2020, March 20–21, 2020, Northeastern University, Boston, Massachusetts.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.XXXX/1122334.4556677X

1 INTRODUCTION

Effective journalism depends on the ability of journalists to accurately perceive and identify the individuals, actions and symbols they encounter. Since 2015, journalists covering political movements have had increasing difficulty identifying many of the graphical symbols being used by political groups, such as the many "alt-right" groups that use logos and symbols that originate and evolve in niche online communities. These are often unfamiliar to mainstream political journalists, a suggested by news reports that appear to conflate or confuse political groups using similar symbols [4]. This may arise, in part, from less-than-optimal literacy about graphical political symbols among both journalists and photo editors who tag images within commercial databases like Getty and Redux. Compounding this problem is the fact that individuals bearing these symbols are often evasive or misleading about the symbols' connotations when interviewed directly [8]. Both the media [1] and activist groups (Fig 1), have at times attempted to educate the public about political symbols and the groups behind them. Given dramatic

Figure 1: Portland Flyer

staff reductions in local newsrooms in recent years, however, the human network that might previously have provided background expertise for identifying the symbols of regional groups has eroded substantially. As such, there is an urgent need for a tool that a) provides journalists at all levels with information about the connotations of a particular symbol, and b) can be efficiently updated as the form and meaning of these symbols evolves. The Political Visual Literacy app addresses this gap through a) key advancements in machine learning-based image-recognition of these symbols, using both original and synthesized images, and b) a trusted human-in-the-loop structure of image submission and review to continually update the database as symbols evolve and emerge. This project therefore contributes both key advancements in computer-vision around graphical symbols and a functional tool that can improve the work of journalists in the field.

Background & Literature Review

The Leafsnap [11] app exploits computer vision to enable tree leaves recognition and has been exploited for educational purposes. Similarly, Merlin Bird Photo ID [16], allows bird watching enthusiasts to leverage computer vision techniques for automatic bird images recognition. The users can also contribute to the database of bird imagery. The PVL project builds on both the use of computer vision-based image-recognition for informational purposes and the human-in-the-loop approach. The latter feature is particularly well-suited to the case of political imagery, where similar variants of a symbol with distinct connotations may exist, and because symbols can evolve relatively quickly. Given the controversial nature of political symbols and the limits of technical measures against manipulation of the machine-learning model, however, the PVL app only allows image contributions from "trusted" users.

In computer vision, the most closely-related work to our application is general object detection [13], logo detection [7] and recognition [5]. Our machine-learning system builds on many of the approaches used to address these more general problems, though a specific challenge our system must accommodate is the limited amount of high-quality training data that we can ethically source for each symbol. As we will detail below, we are able to substantially overcome this limitation through the creation of synthetic training data in order to improve performance.

In the journalism and photography spaces, only two systems appear to address problems related to those of the PVL project. Specifically, the News Provenance Project [10] makes use of machine-learning and perceptual-hash techniques to determine the similarity of two published images; this system, however, does not address symbols in particular and is used to detect multiple publication instances, rather than provide information or context on a given symbol.

The Four Corners Project [3] allows photographers to embed additional metadata in their images in order to provide more robust and authoritative authorship and contextual information. However, there is no specific requirement that photographers include information about the political symbols that may appear in their images, nor does this provide any particular method for end-users to match or identify symbols in their own photographs.

2 METHODOLOGY

Creating an image-recognition app like PVL generally requires 1) sourcing relevant training images, 2) training and optimizing the image-recognition machine-learning model, and 3) designing, implementing, and testing the mobile and web applications. Below we outline how we approached these processes.

Knowledge base construction

We selected an initial symbol set for the system by reviewing existing news images of recent, high-profile political events, such as the 2017 Unite the Right rally in Charlottesville [14]. We also conducted pilot interviews with photojournalists and reporters who routinely cover these events to complement our initial list. We then prioritized symbols which appeared more frequently in our sample imagery or appeared in multiple locations as a starting point for populating the initial database of symbols.

Initially, we attempted to build our training set using the Google Image search results from a keyword search for the name of each symbol, but the resulting images were too low-quality (low-resolution or computer-modified) to train our model effectively. Moreover, despite using available search constraints, this approach did not allow us to control sufficiently for potential intellectual property or privacy concerns. Rights challenges also prevented us from using imagery from editorial databases such as a Getty Images and Redux.

Ultimately, we chose to work directly with individual photographers, whom we compensated for use of their images for model training and testing purposes.

Detection and Recognition of Political Symbols

To automatically detect and recognize political symbols in an image, we designed and built two machine learning "pipelines": one to identify flags in particular, and a second to recognize an individual symbol.

The first system automatically detects flags in the input image and then, for each detected flag, determines whether it includes any of the political symbols in the database. The second system identifies the symbols within a user-designated subregion of the image. This subregion is expected to be

tightly focus on the location (e.g. a hat, t-shirt, or area of skin) on which the image appears.

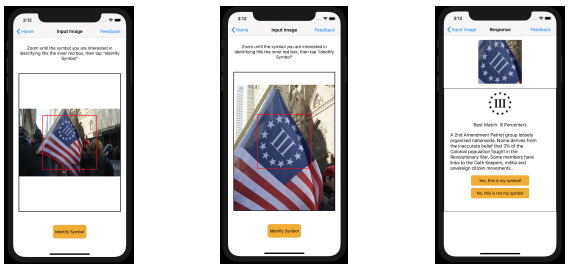
Flag detection. Flags are highly deformable objects that can contain several self-occlusions that therefore require a high volume of training data. Given the above-mentioned challenges to obtaining original imagery, we relied on *synthetic* data produced by incorporating computationally-generated flags into real protest images using the 3D software Blender. We then used these synthesized images to train our "flag detection" pipeline.

Symbol recognition. We faced similar training data limitations when developing the "symbol-recognition" pipeline, but found success using similar synthetic image-generation techniques to apply random image transformations to each symbol and incorporate it into different background images.

Implementation Detail

For the flag pipeline, we selected 15 symbols that are depicted on flags more than they are depicted on any other surface in the images we obtained from photographers. We will call these "symbol flags". We then collected one clean graphic image of each symbol flag from a Google Image search. To develop our synthesized data, we created "textures" from these symbol flags in the 3D software Blender and generated 32 images of each flag fluttering under various different wind conditions and camera views. We then randomly sampled (with replacement) 15,000 protest images that do *not* contain flags from the open-source UCLA protest dataset (the dataset contains metadata that notes whether a given image contains a flag) [17]. Into each sampled protest image, we spliced 1-3 synthesized symbol flag images at a random position, and with a random size and light level. We then ran a faster R-CNN [13] based flag detector trained on the Open Images dataset [12] over all these images. If any of the detected bounding boxes had an Intersection-Over-Union (IOU) larger than 0.5 of the bounding box of the spliced symbol flag, we considered the detected bounding box to be a training sample of that flag. This gave us a dataset that contained a total of 6,543 flag samples. Depending on the performance of the flag detector on different flags, the number of training samples for each flag varies between 90 and 1,029. We also ran the flag detector over the real-world images sourced from photographers to collect a test set of 186 flag samples.

We trained the symbol flag classifier with the image classification pipeline described in [18]. The base network was chosen to be the GoogLeNet [15]. We trained the model for a total of 100 epochs with a batch size of 128, the ADAM optimizer [9] and a learning rate of 0.001. We finally got an average accuracy of 81% on the test set over the 15 symbol flags. For comparison, we also collected a dataset of 425



(a) Input Image standard view

(b) Input Image after zooming into symbol of interest

(c) Example response

Figure 2: Prototype App Core Interface. Fig 2(a) shows the view when a user uploads or takes a photo, 2(b) shows the view after the user has zoomed into the symbol they are interested in identifying and 2(c) shows an example of the app's response. *Photos: Nina Berman*

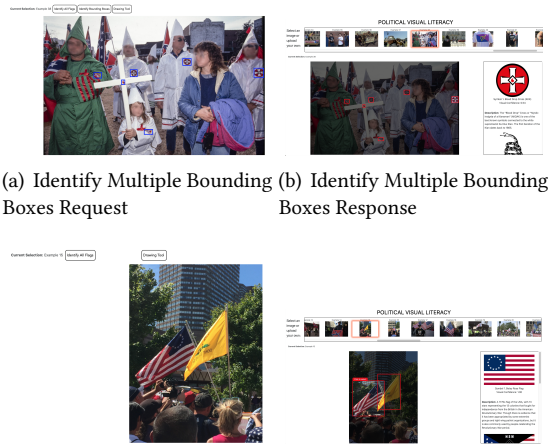
symbol flag samples from images obtained through extensive programmatic searches on Google Images and trained a CNN classifier with the same settings as the one above. This model only got an accuracy of 64% on the test set. This experiment justified to us the use of synthesized images to train the symbol flag recognition model.

For symbol recognition, we started by collecting several clean graphic images of each variation of each of the 52 symbols that we had seen when researching real-world examples of the symbols. If we found a variation on a Google search that we *h*adn't seen in a real-world image, we included it as long as it was a variation of the symbol, and not an entirely new symbol. This distinction was judged subjectively. We then generated 500 training examples by randomly selecting a variation and applying random image transformations (brightness, contrast, noise, transparency, geometrical transformations, morphological operations, etc.) for each symbol and spliced the symbol images onto a combination of textured backgrounds [6] and plain colored images. We cropped the bounding box of the symbol out and used the cropped image to train our symbol recognition model. Note that we did not rely on Blender in this pipeline as we only considered rigid transformations here. This final model achieved a balanced top-1 accuracy of 83% over 52 symbol classes on the photographers images we have annotated, which again proves the feasibility of using synthesized samples to train our political symbol recognition system.

3 MOBILE APP

Target Audience

The mobile app is targeted for political beat photojournalists and reporters wanting to gain knowledge about a symbol they have just observed in-the-field so as to inform their reporting, captioning and picture taking process. The app is also designed for writers and editors working from their



(a) Identify Multiple Bounding Boxes Request

(b) Identify Multiple Bounding Boxes Response

(c) Identify Flags and Recognize Symbol on Flags Request

(d) Identify Flags and Recognize Symbol on Flags Response

Figure 3: Web App Interface. Figs 3(a) and 3(b) show that the web app can handle multiple requests which correspond to regions within the same image. Figs 3(c) and 3(d) show that the webapp supports the flag pipeline too. *Photos: Nina Berman*

desk who encounter an image online which they need to identify, or editors who receive images from photographers in the field which require further annotation.

Mobile App Design

We first developed an iOS App because our early outreach with the journalism community confirmed the iOS preference especially among visual journalists. We are now working on an Android app to satisfy those on that platform.

The app allows a user to select an existing photo from their phone, and zoom in on the relevant symbol until it fills the in-app viewer. The user then clicks a button to send their selection to a server for identification. If the server finds a match or matches, it responds with a list of at most 3 symbols, organized in decreasing order of confidence. The user can then indicate the symbol that matches their image, or indicates that none are correct. In either case, the user can opt to upload their photo to improve or expand the image-recognition model. The threshold for returning a match is set to $1/N$ where N is the number of symbols the model is currently able to recognize (for the moment, $N = 52$), to ensure that the model returns only reasonable-quality matches. In either case the user can choose to upload metadata about the photo's location, date and event. This metadata may be useful for providing additional symbol context in later versions of the app.

4 WEB APP

Target Audience

We have also developed a web application that can be accessed on a desktop or laptop computer via a web browser. The web app is designed for photo editors or journalists working at their desk or doing research in online chat rooms where symbols are often used.

Web App Design

Given the larger screen space, the web app allows users to a) run the query the fully automatic pipeline flag detection and symbol recognition pipeline, or 2) draw one or more bounding boxes on a single image to obtain match results for all of them. Any of these recognition processes can be run on a picture uploaded by the user, and by clicking on each bounding box (either a flag detected or a bounding box drawn by the user) estimated most likely symbols and their short explanation are given in a sidebar. The web app does not contain a feature to upload anything back to the database at the moment.

5 DISCUSSION

We demonstrated the web app to NYC Media Lab's Demo Expo [2] in September 2019 to about 100 media and tech colleagues. Some questions that arose were a) how we curated the database, b) whether we would release the database publicly, c) how we would update the database to include new symbols and symbol variations, d) how we sourced the symbol definitions, and e) how we addressed privacy issues around the app and uploaded photos.

In December 2019, we held a closed door meeting with researchers, photographers, journalists and leaders in the photographic and media tech community to present the app and request feedback on user experience and concept. Participants uniformly requested clarity on what information the app is seeking when a user uploads a picture. They asked what metadata is captured and what can the user opt out of. While our goal is to have users contribute to the database as much as possible, we are sensitive to the needs of users who may not wish contribute their photos to database. As such, providing clear privacy and security assurances is paramount to encourage user participation and submission. Perhaps most significantly, we blur all faces that may appear in uploaded photos to avoid unintended facial-recognition.

Participants also requested a personal page on the app which would keep track of their submissions allowing them to create their own private log. Several participants wanted the database expanded to include more symbols across a wider range of political ideologies. Similarly, expressed interest in symbols appearing in countries outside of the US, but this presents regional interpretation challenges that we are currently leaving to future work.

The feedback from both these events convinced us that a crowdsourcing approach is the best way to ensure wide coverage of new symbols, better coverage of diverse political symbols and to keep up to date with new and changing

symbols. Once users submit a new symbol, we will rely on manual review with to determine whether it should be added to the database.

Finally, these events made it clear how imperative it is to inform users that the app returns definitions only of the overtly political ways a given symbol is used, and that it is always entirely up to the user to determine whether, in context, the app-provided connotations match the context of a symbol in their particular image.

6 CONCLUSION AND FUTURE WORK

In addition to onboarding trusted users selected from news publications, we will soon expand our participant pool to freelance journalists and photographers. Once we understand the extent of engagement with the app from our preliminary users, we can begin to formally define processes for including their input into the database.

Our goal is to complete the Android version of the app in March 2020 so that the app can be use across platforms during the 2020 US general election.

7 ACKNOWLEDGEMENTS

The authors would like to thank Jessica Peng and Michael Deng Li for their tireless work searching for examples of the symbols, as well as annotating the photographs and helping with the machine learning code and data augmentation. We would also like to thank Sam Thielman for sharing his deep knowledge of political groups and helping us write descriptions for the symbols in our database. Finally, we are grateful to the photographers and other experts who have shared their images and feedback with us during the development of this project. This work was supported by Tow Center for Digital Journalism, the Digital Video and Multimedia Lab of Columbia Engineering School, and the Center for Data, Media & Society of Data Science Institute at Columbia University.

REFERENCES

[1] [n.d.]. Deconstructing the symbols and slogans spotted in Charlottesville. https://www.washingtonpost.com/graphics/2017/local/charlottesville-videos/

[2] [n.d.]. Demo Expo. https://summit.nycmedialab.org/demo-expo

[3] [n.d.]. Four Corners Project. https://fourcornersproject.org/en/

[4] 2020. US white supremacist propaganda incidents 'rose by 120% in 2019'. https://www.bbc.com/news/world-us-canada-51480500

[5] Simone Bianco, Marco Buzzelli, Davide Mazzini, and Raimondo Schettini. 2015. Logo recognition using cnn features. In *International Conference on Image Analysis and Processing*. Springer, 438–448.

[6] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. 2014. Describing Textures in the Wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[7] Christian Eggert, Dan Zecha, Stephan Brehm, and Rainer Lienhart. 2017. Improving small object proposals for company logo detection. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*. ACM, 167–174.

[8] Ishaan Jhaveri. [n.d.]. When is a frog not a frog? Building a new digital tool to track political symbols. https://www.cjr.org/tow_center/political-symbols-tow-center.php

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Sasha Koren. 2019. Introducing the News Provenance Project. https://open.nytimes.com/introducing-the-news-provenance-project-723dbaf07c44

[11] Neeraj Kumar, Peter N. Belhumeur, Arijit Biswas, David W. Jacobs, W. John Kress, Ida C. Lopez, and João V. B. Soares. 2012. Leafsnap: A Computer Vision System for Automatic Plant Species Identification.. In *ECCV (Lecture Notes in Computer Science)*, Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid (Eds.), Vol. 7573. Springer, 502–516. http://dblp.uni-trier.de/db/conf/eccv/eccv2012-2.html#KumarBBJKLS12

[12] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. 2018. The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982* (2018).

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.

[14] Hawes Spencer. 2017. A Far-Right Gathering Bursts Into Brawls. https://www.nytimes.com/2017/08/13/us/charlottesville-protests-unite-the-right.html

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[16] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. 2015. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Boston, MA, 595–604.

[17] Donghyeon Won, Zachary C Steinert-Threlkeld, and Jungseock Joo. 2017. Protest activity detection and perceived violence estimation from social media images. In *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 786–794.

[18] Xu Zhang, Felix Xinnan Yu, Svebor Karaman, Wei Zhang, and Shih-Fu Chang. 2018. Heated-up softmax embedding. *arXiv preprint arXiv:1809.04157* (2018).