The Journal of Open Source Software

Logodetect: One-shot detection of logos in image and video data

Jorge Davila-Chacon[1] **and Max Pumperla**[2, 3]

1 Heldenkombinat Technologies GmbH **2** IUBH Internationale Hochschule **3** Pathmind Inc.

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

`logodetect` allows the detection of logos in images and videos after being trained with just one example of a logo. One-shot learning systems are not only great because they require little data, but also because they practically remove the need for specialists whenever the user wants to extend or re-purpose their functionality. This means that the benefits are manifold: the user requires little effort to collect and label training data, there is practically no time or economic costs for the training procedure, and it also provides a strategic benefit for business as they become self-sufficient for a large part of the system maintenance. `logodetect` comes with pretrained models, data and an interface that allows users to detect logos in images and videos with a single command.

Statement of need

There is plenty of literature on the use of deep-learning for detecting logos, so, additionally to sharing pretrained algorithms with the community to get started with one-shot logo detection and a simple interface to use such detectors, the aim of `logodetect` is to provide a flexible architecture to facilitate the comparison of different algorithms for one-shot object recognition. While (Hsieh et al., 2019) published a reference implementation with their paper, there are generally few high-quality object recognizer architectures available to researchers.

`logodetect` works by first performing *object detection* on input images and then running *object recognition* on detection results. The idea is that the user can use a generic detector for a single class of objects (e.g. logos, traffic signs or faces) and then compare each of its detections with the exemplar, i.e., the sub-class that the user is trying to recognize, to determine if both belong to the same sub-class (e.g. a concrete brand, a stop sign or the face of a loved one).

Architecture

The inference pipeline of `logodetect` supports architectures with one or two stages. One-stage architectures directly perform object recognition on the raw images, and two-stage architectures first perform object detection and then object recognition in a second stage.

To get started, we include one `Detector` based on the Faster-RCNN architecture (Ren et al., 2015) for the object-detection phase, and two `Classifiers` for the object-recognition phase.

As a baseline for recognition, in the first `Classifier` provided by `logodetect` we embed the provided exemplars and the detection results from the detection stage into the same latent space, and then simply measure the Euclidean or Cosine distance between the two embeddings. Both inputs are considered to belong to the same sub-class if their distance is below a threshold, determined in a preliminary analysis of the training dataset. The properties of the embedding can be modified by the user.

As a first reference against the baseline, in the second of `logodetect`'s recognizers, we provide a modified ResNet (He et al., 2016) for object-recognition that directly takes the exemplars and the detections from the first stage and predicts if both belong to the same sub-class. Similarly to (Hsieh et al., 2019), this network infers a distance metric after being trained with examples of different sub-classes, but instead of sharing the same weights and processing each input in a separate pass as in (Koch et al., 2015), it concatenates both inputs and processes them in one pass. This concept follows more closely the architecture proposed by (Bhunia et al., 2019), where the assumption is that the exemplars often have more high-frequency components than the detections, and therefore the model can increase its accuracy by learning a separate set of weights for each input.

The code also provides functionality to add various transformations, so the user has the option to augment each exemplar with different transformations if desired. The user simply adds one or more exemplars and is good to go.

Acknowledgements

We would like to thank *NullConvergence* and all the contributors to the open-source framework `mtorch` (NullConvergence, 2018) which serves as the foundation of our training pipeline.

References

Bhunia, A. K., Bhunia, A. K., Ghose, S., Das, A., Roy, P. P., & Pal, U. (2019). A deep one-shot network for query-based logo retrieval. *Pattern Recognition*, *96*, 106965. https://doi.org/10.1016/j.patcog.2019.106965

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Hsieh, T.-I., Lo, Y.-C., Chen, H.-T., & Liu, T.-L. (2019). One-shot object detection with co-attention and co-excitation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32, pp. 2725–2734). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/92af93f73faf3cefc129b6bc55a748a9-Paper.pdf

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, *37*. https://www.cs.cmu.edu/%C2%A0rsalakhu/papers/oneshot1.pdf

NullConvergence. (2018). *Mtorch*. Open-source software under MIT License. https://github.com/NullConvergence/mtorch

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-CNN: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 28, pp. 91–99). Curran Associates, Inc. https://doi.org/10.1109/tpami.2016.2577031