KDSalBox: A toolbox of efficient knowledge-distilled saliency models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Dozens of saliency models have been designed over the last few decades, targeted at diverse applications ranging from image compression and retargeting to robot navigation, surveillance, and distractor detection. Barriers to their use include the different and often incompatible software environments that they rely on, as well as the computational inefficiency of older implementations. For application-purposes models are then frequently chosen based on convenience and efficiency, at the expense of optimizing for task performance. To facilitate the evaluation and selection of saliency models for different applications, we present KDSalBox - a toolbox of 10 knowledge-distilled saliency models. Using the original model implementations available in their native environments, we produce saliency training data for efficient MobileNet-based architectures, that are identical in their architecture but differ in how they learn to distribute saliency over an image. The resulting toolbox allows these 10 models to be efficiently run, compared, and be practically applied.

1 Introduction

At the speed at which technology and software libraries change, image processing applications like segmentation, stylization, etc., become quickly outdated. New implementations replace older ones, in some cases re-inventing prior solutions, and in others supplanting older algorithms with newer, data-driven ones. In the case of saliency modeling, the top performing models on saliency benchmarks (Bylinskii et al. (2015); Yu et al. (2017)) are deep neural networks, all trained on the same dataset of visual attention data - the only one of its size (Jiang et al. (2015)). As a result, the top models all fall prey to similar biases and have similar limitations (Bruce et al. (2015); Bylinskii et al. (2016)). At the same time, older models including the one by Itti et al. (1998b), based on low-level image features rather than being directly trained on eye movement data fare better for certain applications (Bruce et al. (2015)). For these reasons, they continue to serve as comparisons in saliency papers today.

On the other hand, the increasing availability of constrained computing environments (e.g., mobile phones) have led to increased focus on the design of efficient models. A number of compact architectures have been proposed recently (Sandler et al. (2018); Tan & Le (2019)). These models are carefully designed with components such as inverted bottleneck units, depthwise separability and pointwise convolutions with batch normalization. Key techniques that have been used to train these networks include model compression and knowledge distillation (Caruana et al. (2006); Hinton et al. (2015)), where a student (the efficient network) uses the teacher's (the complex network) supervision during the training process. Knowledge distillation of neural networks commonly uses the intermediate activations as hints (Romero et al. (2014)) for better transfer learning. However, the nature of the student-teacher paradigm allows to use the teacher in a black-box manner where the cross-entropy of only the output of the teacher and the student is backpropagated as a signal to

Submitted to 3rd Workshop on Shared Visual Representations in Human and Machine Intelligence (SVRHM 2021) of the Neural Information Processing Systems (NeurIPS) conference.

update the weights (Hinton et al. (2015)). Recent work in saliency has shown that efficient models for saliency prediction can perform on par with the more complex models while being several orders of magnitude faster (Hu & McGuinness (2020); Zhang et al. (2019)).

The sheer quantity and diversity of the available computational saliency models makes it difficult to evaluate what ideas or mechanisms are generalizable, scalable or better suited for a given application. To this end, the MIT Saliency Benchmark (Bylinskii et al. (2015)) was introduced and has shown to be beneficial for driving progress in saliency research. The MIT Saliency Benchmark provides for each model only a coarse summary of performance and a model ranking based on a set of standardized metrics. However, beyond comparing summary numbers, re-implementing or re-running these models to evaluate them on new datasets and applications often proves difficult due to different (and occasionally deprecated or outdated) technical requirements, environments, and dependencies. A few saliency frameworks (Kuemmerer et al. (2015), Wloka et al. (2018)) have recently been proposed to reduce this effort and create a standardized interface to many saliency models at once. These frameworks not only help researchers explore these models but also facilitate reproducibility. While these frameworks can facilitate saliency research, continuous maintenance of these frameworks and dependencies is required as the number of models continues to grow. Wloka et al. (2018) tackles this issue by providing containerized images for each model. However, practical technical barriers remain, such as that all Matlab-based models require a licence and are typically not efficient to run. Given that saliency is often used as a preprocessing step for downstream image processing tasks, the traditional (non-deep) models are often rendered impractical, even though the predictions they make may be better suited for some applications.

2 The KDSalBox

Our toolbox[1] consists of ten knowledge-distilled models that try to mimic as closely as possible the original implementations. This approach provides a unified framework for saliency research that is easy to maintain, facilitates the exploration, evaluation and selection of saliency models, and levels the playing field for different types of models (e.g., including Matlab-based libraries with bulky wrappers). All the knowledged-distilled models are identical in their architecture but only differ in how they learn to distribute saliency over an image. This makes the framework very easy to use, and we show later that despite the simplicity, the knowledge-distilled models learn to behave very closely to the original counterparts.

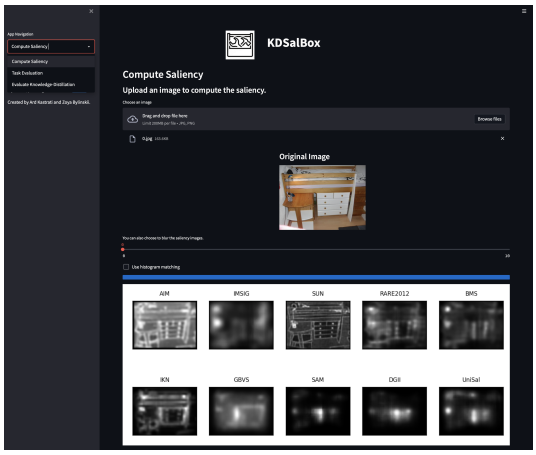

Figure 1: A screenshot of the Streamlit user interface for accessing the KDSalBox.

2.1 Interface of KDSalBox

Our toolbox offers a similar interface as Wloka et al. (2018). This includes pre- and postprocessing steps such as center prior, blurring, scaling as well as standardized evaluation using the metrics NSS, CC, KL, SIM (Bylinskii et al. (2018)). In addition, due to the simplicity and unified design of the framework, it also provides the infrastructure to distill the knowledge of new models with minimal effort. Finally, we also implemented a graphical user interface to plot the saliency maps (see Figure 1) with the goal to facilitate the evaluation and selection of saliency models for different applications.

2.2 Ten saliency models

The models that the toolbox currently includes are the following: AIM (Bruce & Tsotsos (2009)), IKN (Itti et al. (1998a)), GBVS (Schölkopf et al. (2007)), BMS (Zhang & Sclaroff (2013)), IMSIG (Hou et al. (2012)), RARE2012 (Riche et al. (2013)), SUN (Zhang et al. (2008)), UniSal (Droste

[1] Available at: URL anonymized.

et al. (2020)), SAM (Cornia et al. (2018)), DGII (Kümmerer et al. (2016)). The first seven models are originally implemented in Matlab and mostly focus on lower level image features. The remaining three are deep neural networks that capture higher level object-based features (such as understanding of objects, faces, etc.) and perform the best on current saliency benchmarks.

2.3 Knowledge distillation

To perform knowledge distillation, we generated saliency maps from each original model on the SALICON dataset (Jiang et al. (2015)). The generated saliency maps were then used for supervision during the training procedure. Note that, we used *only* the generated saliency images. In particular, we didn't use the ground truth (eye movements) of the SALICON dataset as input during training of our ten models, since our goal was to mimic as closely as possible the original models, no matter how they perform on other benchmarks.

Model architecture. Our model design follows the encoder-decoder pattern. We used the MobileNetV2 (Sandler et al. (2018)) for the encoder and a simple convolutional neural network (composed of seven convolutional layers with batch normalization) as a decoder. The MobileNet-based encoder serves as a lightweight backbone for image feature computation, while the decoder generates the final saliency map. Through some pilot testing, we converged on a decoder that was simple and compact while performing well i knowledge distilling the ten different saliency models.

Training procedure. For our training and evaluation we used the generated saliency images of each model on the SALICON training and evaluation datasets, respectively. First the encoder was initialized with the pretrained weights on ImageNet. Then for the first three epochs we froze the encoder, and trained the decoder with our generated images. After three epochs the encoder was unfrozen and the training proceeded normally. We used a learning rate of 0.1. During training we used a binary cross entropy loss and the Adam optimizer. We also experimented with other custom losses (such as a weighted combination of KL-divergence, Correlation Coefficient, etc.) but we didn't observe a noticeable improvement. All our models were trained on a TITAN Xp (12G) GPU.

3 Evaluation

Metrics. Following Bylinskii et al. (2018), we report the distribution-based metrics CC and KL for evaluating saliency models. **CC** measures Pearson's Correlation Coefficient between two saliency maps as a ratio of their covariance to the product of their variances. **KL** is the Kullback-Leibler divergence between two saliency maps, evaluating the loss of information when one of the maps is used to approximate the other. From their definitions, CC is a symmetric metric, while KL is not. Two maps are similar as CC approaches 1 (higher correlation) and KL approaches 0 (lower divergence).

How well have the students learned? We evaluate the ability of each of the knowledge distilled student models to approximate their teachers - the original saliency models run in their native code bases. We report the similarity between the saliency maps produced by the teacher and student models on the SALICON validation data of 5000 images, in Table 1. The trained models perform relatively close compared to their original counterparts. Interestingly, the Matlab-based ones are easier to reproduce, compared to the deep learning ones. We hypothesize that this is due to the limited dataset size during training making it difficult to capture high-level features, as well as the simple model architecture we used. Using the intermediate features of the deep learning models (*hints*) could have helped during training, but would have resulted in different training procedures for different models (since we can not adapt *hints* directly for the Matlab-based models).

Can the students generalize? Our student models were trained on saliency maps computed on the SALICON training images with the intention of capturing saliency behavior on a general sample of natural images. Therefore by design, our student models are intended to mimic the behavior of the original models on natural images, without guarantees about other image types. In Table 1, we also report the similarity of our student models to the original models on non-natural images, specifically the pattern images from the CAT2000 dataset by Borji & Itti (2015). The difference in their performance compared to the setting with natural images is considerable (especially the deep models). For example, the SAM model's CC score decreases to 0.48 from 0.82 when changing

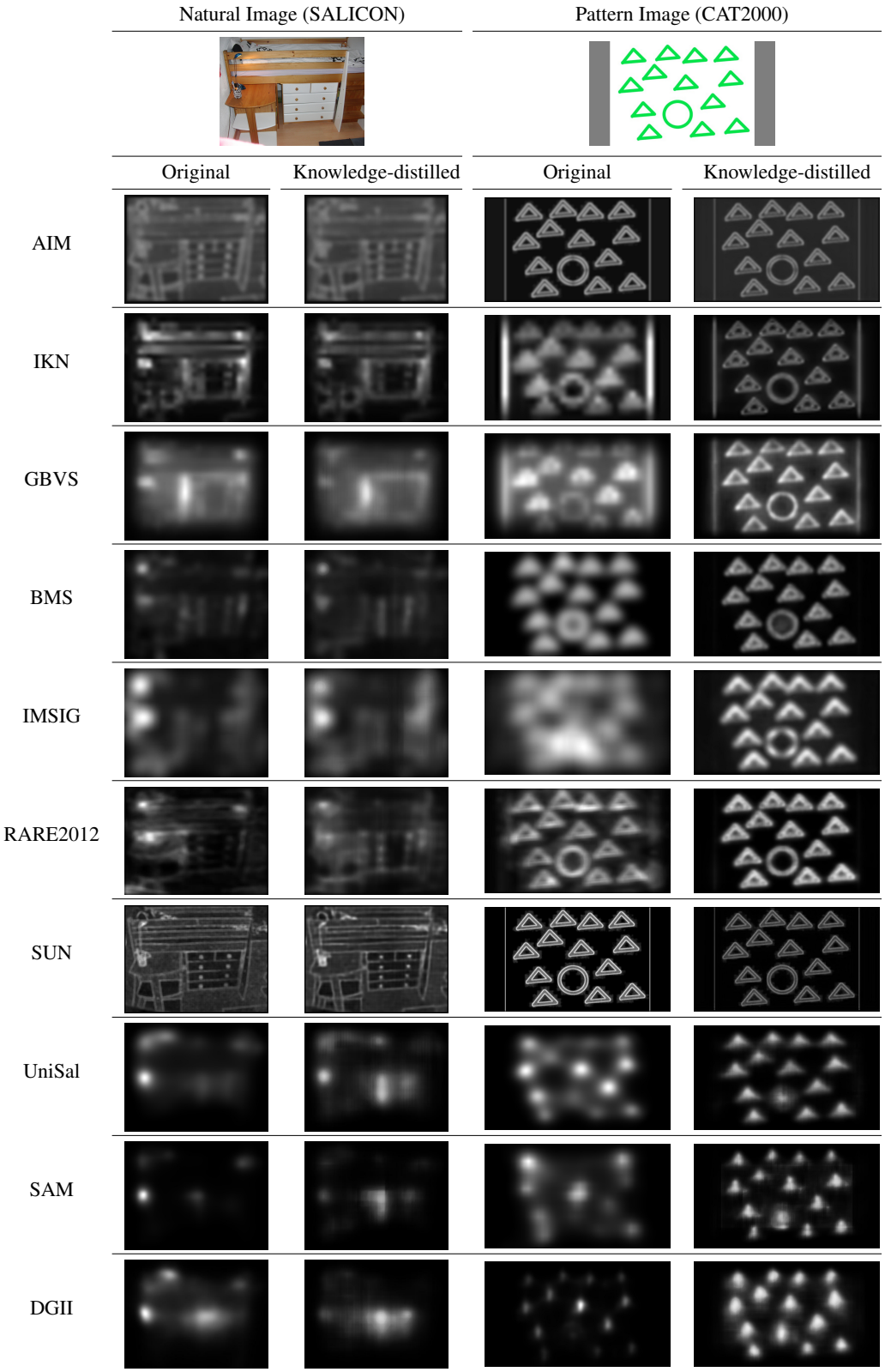

Figure 2: The performance of ten knowledged-distilled models in our KDSalBox shown for a sample image from both datasets: Natural Images (SALICON) and Patterns (CAT2000).

Table 1: The ten models currently part of the KDSalBox. We include scores comparing the knowledge-distilled student models to the original saliency model implementations. We report the average inference time by using the SMILER (Wloka et al. (2018)) framework and 640×480 sample images. We note that in comparison to the original implementations, all our resulting student models are 41.2 MB in size with an average inference time of 0.4 sec on CPU and 0.01 sec on GPU.

Model	Original implementation			(Dis)similarity between student & original models			
				Natural Images (SALICON)		Patterns (CAT2000)	
	Codebase	Size	Time (CPU)	CC ↑	KL ↓	CC ↑	KL ↓
AIM	Matlab	24MB	10sec	0.99 ±0.00	0.00 ±0.00	0.96 ±0.04	0.18 ±0.23
IKN	Matlab	1MB	6.0sec	0.97 ±0.01	0.02 ±0.01	0.78 ±0.09	0.14 ±0.06
GBVS	Matlab	1MB	6.1sec	0.98 ±0.01	0.01 ±0.01	0.75 ±0.09	0.17 ±0.07
BMS	Matlab	5MB	0.4sec	0.89 ±0.07	0.05 ±0.02	0.78 ±0.11	0.30 ±0.26
IMSIG	Matlab	20KB	5.7sec	0.96 ±0.04	0.02 ±0.02	0.71 ±0.15	0.16 ±0.13
RARE2012	Matlab	20KB	6.3sec	0.87 ±0.10	0.10 ±0.06	0.76 ±0.12	0.30 ±0.16
SUN	Matlab	1MB	12sec	0.94 ±0.04	0.02 ±0.01	0.95 ±0.02	0.21 ±0.25
UniSal	Python	30MB	0.4sec	0.92 ±0.05	0.11 ±0.06	0.68 ±0.14	0.46 ±0.36
SAM	Python	561MB	7.3sec	0.82 ±0.13	0.28 ±0.17	0.48 ±0.15	0.87 ±0.40
DGII	Python	330MB	4.8sec	0.90 ±0.08	0.14 ±0.09	0.68 ±0.16	0.48 ±0.42

domains from natural images to patterns. In Figure 2 we show how each model performs for a sample image in each dataset (SALICON and CAT2000). For the sample pattern image shown, all knowledge-distilled models output saliency along the edges of the triangles and the circle, whereas their original counterparts (especially the deep neural networks) predict differently, utilizing higher level features for the saliency maps. On the other hand, the original counterpart of the low level models (such as AIM, SUN, etc.) fail to capture these features as well, which explains why the knowledge-distilled models generalize well in this domain (i.e., reproducing their failures).

Data preprocessing during training. Models differ in the ranges of their saliency maps (e.g., compare AIM and SAM in Figure 2). It is difficult to visually compare the suitability of models for an application by ignoring these differences in saliency values, independently of the actual locations the models predict. For this reason we provide histogram matching as a possible processing step to facilitate apples-to-apples model comparisons. For these reasons, we also trained histogram matched model versions, but did not find that they improved model performance (more details in Appendix A).

4 Discussion

Contributions. The toolbox presented in this work provides an approachable framework to conduct research on saliency with minimal effort. Our approach of offering a standardized but also very simple interface and environment for all saliency models, can be very useful in the research community and in the industry. Our toolbox composed of ten saliency models allows users to produce saliency images from a variety of saliency models with great efficiency to evaluate them for application purposes.

Limitations and Future Work. In our work, we have used a unified architecture for all models for simplicity and easier maintainability. Depending on the application and setting (e.g. on-device deployment), typically, a further reduction of the size or inference time might be needed in practice. Depending on the model, one can also design separate decoders that are more suitable for each model. This, however, comes at the expense of simplicity and generalizability. There are also other techniques, such as optimizing for the memory hierarchy, for special instruction sets, change of datatypes, etc., that can be used for even further reduction of the size and inference time of the models. Another current limitation is that our models are trained only on natural images of the SALICON dataset and there is little evidence how they will perform on other out-of-distribution datasets. We have provided evaluation metrics on the CAT2000 pattern dataset, however the training is done solely on the natural images. Extensions of this work call for training on other domains and with larger datasets. Finally, since all models share the same architecture and reside in the same framework, a natural extension of this work would be to look for ways to combine these models like building blocks to increase performance on saliency benchmarks.

References

Ali Borji and Laurent Itti. Cat2000: A large scale fixation dataset for boosting saliency research. *arXiv preprint arXiv:1505.03581*, 2015.

Neil D. B. Bruce and John K. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of Vision*, 9(3):5–5, 03 2009. ISSN 1534-7362. doi: 10.1167/9.3.5. URL `https://doi.org/10.1167/9.3.5`.

Neil DB Bruce, Calden Wloka, Nick Frosst, Shafin Rahman, and John K Tsotsos. On computational modeling of visual saliency: Examining what's right, and what's left. *Vision research*, 116:95–112, 2015.

Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark. 2015.

Zoya Bylinskii, Adrià Recasens, Ali Borji, Aude Oliva, Antonio Torralba, and Frédo Durand. Where should saliency models look next? In *European Conference on Computer Vision*, pp. 809–824. Springer, 2016.

Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 41(3):740–757, 2018.

Rich Caruana, Cristian Bucila, and Alexandru Niculescu-Mizil. Model compression, 2006.

Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. Predicting Human Eye Fixations via an LSTM-based Saliency Attentive Model. *IEEE Transactions on Image Processing*, 27(10):5142–5154, 2018.

Richard Droste, Jianbo Jiao, and J. Alison Noble. Unified Image and Video Saliency Modeling. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

Xiaodi Hou, Jonathan Harel, and Christof Koch. Image signature: Highlighting sparse salient regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):194–201, 2012. doi: 10.1109/TPAMI.2011.146.

Feiyan Hu and Kevin McGuinness. Fastsal: a computationally efficient network for visual saliency prediction. *CoRR*, abs/2008.11151, 2020. URL `https://arxiv.org/abs/2008.11151`.

L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998a. doi: 10.1109/34.730558.

Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11): 1254–1259, 1998b.

Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1072–1080, 2015.

M. Kuemmerer, T. Wallis, and M. Bethge. Information-theoretic model comparison unifies saliency metrics. *Proceedings of the National Academy of Science*, 112(52):16054–16059, Oct 2015. doi: 10.1073/pnas.1510393112. URL `http://www.pnas.org/content/112/52/16054.abstract`.

Matthias Kümmerer, Thomas S. A. Wallis, and Matthias Bethge. Deepgaze II: reading fixations from deep features trained on object recognition. *CoRR*, abs/1610.01563, 2016. URL `http://arxiv.org/abs/1610.01563`.

Nicolas Riche, Matei Mancas, Matthieu Duvinage, Makiese Mibulumukini, Bernard Gosselin, and Thierry Dutoit. Rare2012: A multi-scale rarity-based saliency detection with its comparative statistical analysis. *Signal Processing: Image Communication*, 28(6):642–658, 2013. ISSN 0923-5965. doi: https://doi.org/10.1016/j.image.2013.03.009. URL `https://www.sciencedirect.com/science/article/pii/S0923596513000489`.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *arXiv*, 12 2014.

Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381, 2018. URL `http://arxiv.org/abs/1801.04381`.

Bernhard Schölkopf, John Platt, and Thomas Hofmann. *Graph-Based Visual Saliency*, pp. 545–552. 2007.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019. URL `http://arxiv.org/abs/1905.11946`.

Calden Wloka, Toni Kunic, Iuliia Kotseruba, Ramin Fahimi, Nicholas Frosst, Neil D. B. Bruce, and John K. Tsotsos. SMILER: saliency model implementation library for experimental research. *CoRR*, abs/1812.08848, 2018. URL `http://arxiv.org/abs/1812.08848`.

F. Yu, P. Kontschieder, S. Song, M. Jiang, Y. Zhang, C.Q. Zhao, T. Funkhouser, and J. Xiao. Large-scale scene understanding challenge. http://http://lsun.cs.princeton.edu/2017/, 2017.

Jianming Zhang and Stan Sclaroff. saliency detection: a Boolean map approach. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

Lingyun Zhang, Matthew H. Tong, Tim K. Marks, Honghao Shan, and Garrison W. Cottrell. SUN: A Bayesian framework for saliency using natural statistics. *Journal of Vision*, 8(7):32–32, 12 2008. ISSN 1534-7362. doi: 10.1167/8.7.32. URL `https://doi.org/10.1167/8.7.32`.

Peng Zhang, Li Su, Liang Li, BingKun Bao, Pamela Cosman, GuoRong Li, and Qingming Huang. Training efficient saliency prediction models with knowledge distillation. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, pp. 512–520, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450368896. doi: 10.1145/3343031.3351089. URL `https://doi.org/10.1145/3343031.3351089`.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] ans
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A] For anonymity purposes, this is not provided at the moment, but we plan to publish our toolbox.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Histogram Analysis

As in Table 1, we report our results in Table 2 for the case where the students are trained after all saliency maps are matched to the histogram of the ground truth of SALICON dataset. This makes it possible to compare the models with each other and facilitate the evaluation and selection of saliency models for different applications. As we can see the trained student models perform similarly as in the case without histogram matching in the SALICON dataset, but illustrate better where the saliency is distributed across the images. This is however not the case for the Pattern dataset, since the distribution in these images is quite different and reduces the ability of the student models to mimic their original counterparts. We also show in Figure 3 a sample image for both datasets for the histogram-matched case. Similar to our analysis in Section 3, the same conclusions apply here as well — i.e. the knowledge-distillation of the deep learning models generalize worse then the lower-level models. A potential way to improve this gap is to use a separate knowledge-distillation procedure for the deep learning models and use the intermediate layers to guide the training process (Romero et al. (2014)).

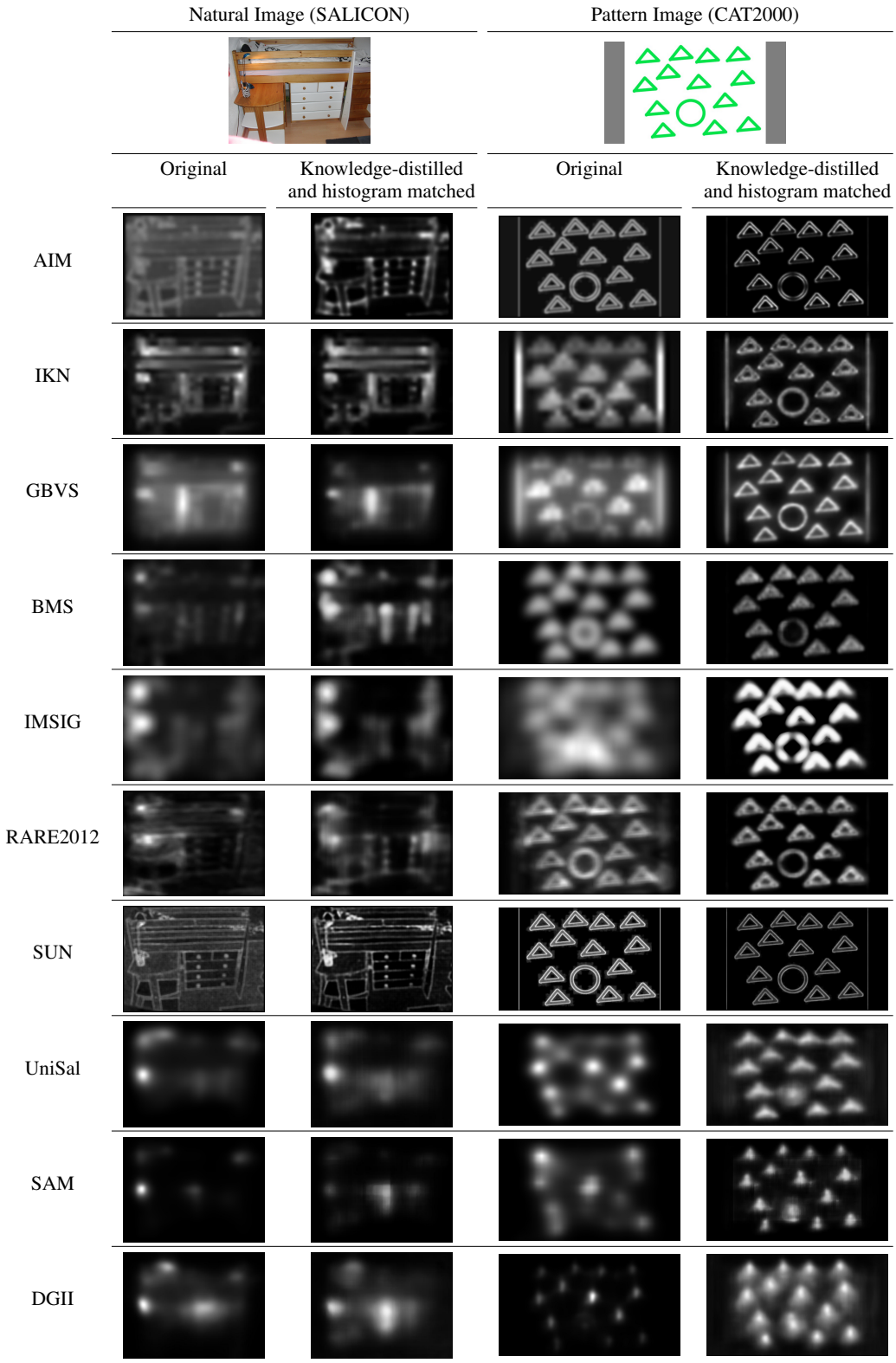

Figure 3: The performance of knowledged-distilled models after histogram-matching to the mean histogram of ground truth in the SALICON dataset shown for a sample image from both datasets: Natural Images (SALICON) and Patterns (CAT2000).

Table 2: The ten models currently part of the KDSalBox trained on the histogram-matched images.

Model	(Dis)similarity between student-teacher			
	Natural Images (SALICON)		Patterns (CAT2000)	
	CC ↑	KL ↓	CC ↑	KL ↓
AIM	0.98 ±0.01	0.04 ±0.10	0.87 ±0.10	3.03 ±3.00
IKN	0.97 ±0.02	0.04 ±0.03	0.73 ±0.10	0.29 ±0.12
GBVS	0.97 ±0.02	0.03 ±0.02	0.71 ±0.09	0.30 ±0.09
BMS	0.89 ±0.07	0.14 ±0.08	0.62 ±0.17	1.12 ±0.81
IMSIG	0.94 ±0.05	0.08 ±0.10	0.48 ±0.16	0.86 ±0.46
RARE2012	0.86 ±0.11	0.19 ±0.12	0.63 ±0.14	0.74 ±0.50
SUN	0.93 ±0.06	0.09 ±0.08	0.82 ±0.11	2.71 ±2.11
UniSal	0.93 ±0.05	0.07 ±0.04	0.57 ±0.12	0.35 ±0.09
SAM	0.80 ±0.09	0.86 ±0.43	0.48 ±0.14	0.90 ±0.37
DGII	0.95 ±0.04	0.06 ±0.04	0.73 ±0.12	0.21 ±0.08