Attentional Scene Classification with Human Eye Movements

Alex Kuefler
Department of Symbolic Systems
Stanford University
akuefler@stanford.edu

Abstract

Attentional models of computer vision present a biologically plausible and computationally inexpensive alternative to the deep learning pipeline popularized for image recognition in recent years. Although drawing inspiration from perceptual psychology, little research has attempted to leverage attentional policies inferred from human data in order to improve image classification for artificial intelligence. The purpose of this work is to compare the performance of a sequential classification model that learns a multi-class scene classification task, after training on sequences of observations generated by humans, saliency-maps, and reinforcement learning.

1. Introduction

Computer vision has experienced a period of rapid growth in the past four years, as the introduction of deep convolutional neural networks (CNN) has advanced the state of the art in tasks such as object recognition [1]. However, the computational expense of such methods scales with the dimensionality of input images [2] and can become prohibitive for image processing with high-resolution or large datasets. Recent work has focused on attentional models of visual recognition as an alternative to processing entire images in parallel [2, 3].

Attentional models recast computer vision as a sequential decision making problem, allowing an agent to deploy a sensor (i.e., an attentional window) to image data across multiple time-steps. This approach bears a resemblance to perceptual psychology, where a large amount of data has been gathered on human eye movements in visual search [4, 5]. Despite the parallels between attentional computer vision models and cognitive psychology, there is little past work that attempts to apply human attentional policies directly to image classification.

This work treats human subject data as demonstrations from domain experts, which a visual agent can use to excel in image classification tasks, while viewing only partial images.

A model is presented that takes inputs consisting of observation sequences generated from one of four policies (human observation, reinforcement learning, random sampling, or saliency maps). An observation sequence is defined as an array of 40x40 grayscale regions of interest (ROI), each centered at a fixation point given by the policy. The agent must then output one of 20 labels, identifying the category of the image from which the observation sequence was drawn (e.g., object, affective, outdoor, sketch, etc.)

Because this model has much more limited information about the images it classifies, it is reasonable to expect that it will underperform the classic approach in terms of pure classification accuracy. But by processing only critical sub-regions of its inputs, it is hypothesized that the model will trade off accuracy for training and test-time efficiency.

There are however some theoretical reasons to expect a qualitative difference in classification accuracy favoring the attentional model. For example, through experimentation on the salamander retinal ganglion cell, [28] demonstrated that frequent, fixational eye movements might decorrelate the retina's response to visual inputs. By reducing the redundancy in data processed prior to classification, it is conceivable that the attentional model may come up with a more compact representation of the training data and outperform the baseline. The key is to produce an observation-selecting policy that picks out image sub-regions that are most critical to the experimental task (i.e., classification), while managing to decorrelate each observation in the sequence.

The effects of the four policies on image classification are compared and recommendations are made for future work.

2. Related Work

Saliency maps have long found use in the cognitive science community as models of selective attention. Such models were first formalized by Niebur & Koch [6], who described saliency maps as topographically organized fields of visual processing, where low-level stimulus

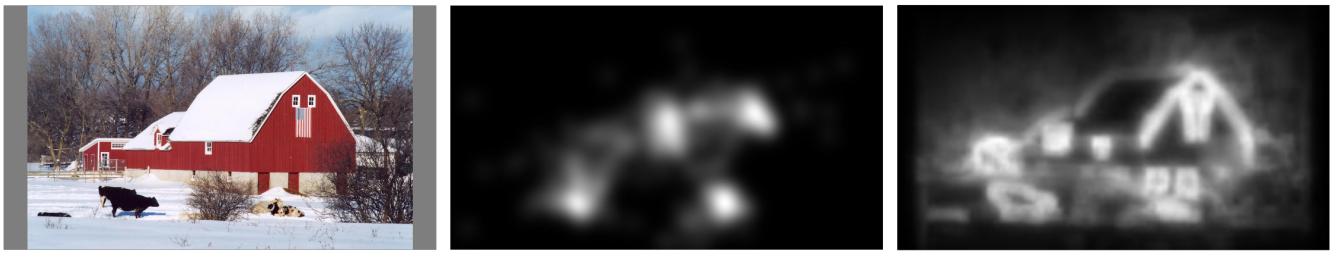

Figure 1. From left to right: A typical image from CAT2000, a fixation density may, and corresponding saliency map.

features (e.g., colors, gradients, intensities, orientations) drive heightened neural responses. They have been used fruitfully in both basic research [7] in addition to technical applications, such as improving visual displays with variable resolution [8, 9].

Although predictive of where people look, saliency maps fail to capture the temporal order in which visual attention is deployed. Recent work suggests reinforcement learning as a viable alternative for creating attentional models that extend over time periods. Butko & Movellan [10] model visual search as a discrete action-space, partially observable Markov Decision Process, which accumulates reward by minimizing the entropy of a belief vector (i.e., a probability distribution over regions of the visual field indicating where a target might be). Similarly, Bandera et al. employ residual Q-learning to an object classification task in creating biologically plausible model of foveal vision [11].

Only very recently have reinforcement learning models of attention begin to enjoy popularity in general purpose, prescriptive, artificial intelligence tasks such as object tracking [12], digit classification [2], and image captioning [13].

Nevertheless, myriad approaches exist for representing hard attention in computer vision systems. Recently introduced spatial transformer networks present a fully-differentiable method of giving deep networks sequential control over focal regions [14]. Saliency maps have also adapted well to the deep learning paradigm, where some researchers now use internal activation patterns of CNNs in place of theory-driven models of saliency [15].

Visual attention remains an open research question where reinforcement learning, saliency, and a wealth of human subject data all provide unique insights. This paper attempts to reconcile these approaches in a comparative analysis.

3. Data

The CAT2000 [5] dataset was used to train and evaluate all models. It includes 2000 images spread across 20 object categories. Each image was viewed by up to 24 human observers, whose eye movement patterns were used to generate the approximately 30,000 observation sequences that form each model's training, testing, and validation sets. Furthermore, the CIFAR-10 [16] dataset was used to benchmark the baseline model, in order to provide a basis of comparison for the difficulty of CAT2000. To my knowledge, CAT2000 has never before been used for scene classification.

CAT2000 also includes corresponding fixation density and saliency maps for each image. Fixation density maps are the smoothed locations of all 24 observers' fixations on each image, which is used to induce a reward signal for reinforcement learning, described in 3.2. Prior to experimentation and observation-sequence extraction, all images were downsampled to 256x256 and converted to grayscale.

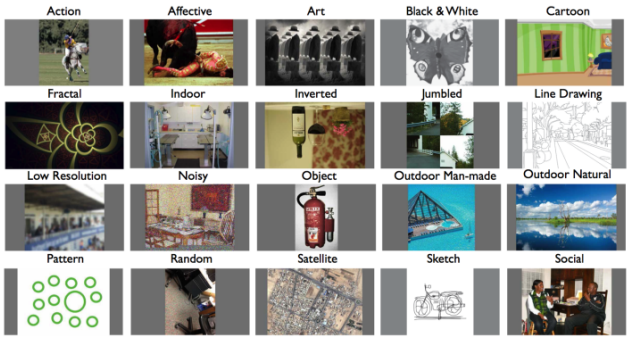

Figure 2. The 20 scene categories of CAT2000 [17].

4. Methods

Three neural network models were first implemented using the deep learning library Lasagne [18]. Two networks perform classification of the CAT2000 dataset, using either observation sequences or by processing whole images (to establish a baseline for comparison). The third network is trained on a regression task using reinforcement learning to generate sequences of observations to be used by the classifier.

In addition to the reinforcement learning agent, three other methods of extracting observation sequences from static images were also compared, and described below.

4.1. Observation sequence extraction

For the purposes of this work, an observation sequence is defined as a tensor in $\mathbb{R}^{9\times40\times40}$, where each submatrix is a 40x40 map of grayscale pixel intensities centered at some fixation point. The minimum number of fixation points generated by each human view per trial is close to 9 elements. This value was chosen as the horizon of each

observation in order to maximize the usable amount of human subject data.

Four methods of extracting fixation points were tested:

Human fixation points were collected from CAT2000 individual participant trials. Trajectories exceeding the horizon were clipped so that only the first 9 saccades of each trial were used. A participant's trial was discarded if at any point the tracking went out of bounds of the image. After all invalid data were removed, approximately 16.5 valid trials remained per image, leaving a final training set size of 16,550 sequences and 7,921 sequences for validation and testing.

Random observation sequences were generated by uniform sampling of each image. A random sequence was created for each valid human trajectory in order to keep training and testing set sizes even between the experimental conditions.

Saliency maps were also used to produce fixation points. Each CAT2000 stimulus image comes with the output image of a theory-driven model of visual salience, which was then treated as a landscape of candidate fixation points. 9 local maxima in the saliency map were sampled uniformly at random to serve as the center points for each observation in a training example sequence. If fewer than 9 maxima exist in the saliency map, the points are sampled with repetition.

Reinforcement learning was also used to train an Attentional Agent to generate observation sequences. A population of 10 agents with the same neural network architecture was trained on a small subset of 60 CAT2000 training images (2 examples from each of the 20 scene categories). During training, each agent learned to maximize reward received for attending to sub-regions of the image featuring high inter-observer fixation density. In this sense, the agents attempt to reproduce the spatial properties of human attention, but are free to learn their own temporal ordering of observations. After training to optimize reward, the agents' function mapping image inputs to fixation points was used like any of the other methods in order to generate training, validation, and testing sets for classification. It should be noted that because the agent population consisted of only 10 models, the final dataset sizes were smaller (10,000 for training, 5,000 for validation-training).

Details of the reinforcement learning agents' architecture and training algorithm are included below.

4.2. Attentional Agent

The first agent employs reinforcement learning to come up with an attentional policy, which is used to select ROIs to be processed by the downstream classifier. It is based on the Continuous Actor Critic Learning Automaton (CACLA) [19]. CACLA employs two function approximators: An "actor" π parameterized by ψ_t at timestep t that maps an observation of a system's state to a continuous action. And a "critic" V parameterized by θ_t that maps observations to the sum of expected future reward, weighted by a discount factor γ. CACLA learns by perturbing the actor's output action $\pi(s_t; \psi_t)$ each timestep with Gaussian noise, producing a candidate action a_t. If the temporal difference error between the critic's estimate of a current observation's value $V_t(s_t)$ and the estimate for an observation that proceed a_t is positive, then the actor network's parameters are updated using backpropagation.

Initialize θ_0 (below $V_t(s) = V(s;\theta_t)$), ψ_0, s_0.
for $t \in \{0, 1, 2, \ldots, 9\}$ **do**
  Choose $a_t \sim \pi(s_t; \psi_t)$
  Perform a_t, observe r_{t+1} and s_{t+1}
  $\delta_t = r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)$
  $\theta_{t+1} = \theta_t + \alpha_t(s_t)\delta_t \nabla_\theta V_t(s_t)$
    if $\delta_t > 0$ **then**
      $\psi_{t+1} = \psi_t + \beta_t(s_t)\big(a_t - Ac(s_t;\psi_t)\big)\nabla_\psi Ac(s_t;\psi_t)$
  if s_{t+1} is terminal **then**
    Reinitialize s_{t+1}

Figure 3. CACLA pseudocode. [20]

In our case, the observation is the 40x40 grayscale ROI taken from the window of attention the agent learns to control. The actions are 2D vectors whose components consist of values between 0 and 1, representing the percentage of the total height and width of an input image where the agent focuses its attentional window (e.g., <0.5, 0.5> represents image center, and <0.0, 1.0> represents the top left corner). The observation-value output by the critic is a scalar approximation of reward, which is defined as the mean human fixation density within the attentional window, after zero centering and division by its standard deviation. Reward is also "depleted" (pixels in the fixation density map are set to -1) in regions where the agent has already deployed its sensor, thus encouraging it to explore more of the image.

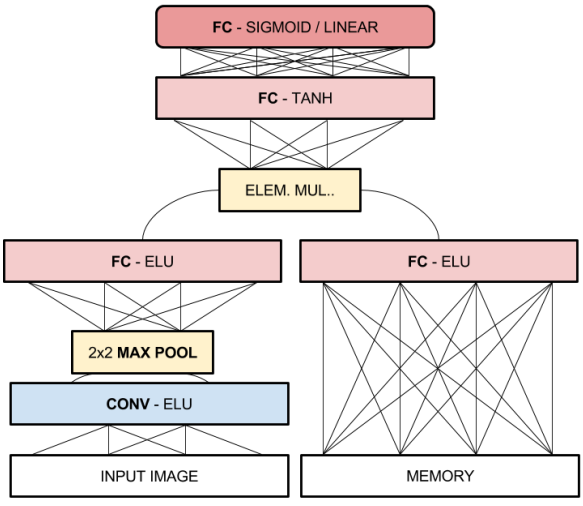

Figure 4. Factored architecture of attentional actor and critic models.

Both actor and critic are represented with factored regression networks, trained to minimize the mean-squared error loss function. Because the actor must output values between 0 and 1, its output layer consists of two sigmoid neurons, whereas *tanh* nonlinearities are used in the hidden layer to keep the distribution of the data in a scale close to the sigmoid outputs. The critic is instantiated with a single, linear output neuron, allowing it produce a scalar approximation of the reward given some observation, without constraint.

The factored design allows both actor and critic to use convolutions to process image-observations, while simultaneously accounting for flat "memory vectors". Memory vectors are simply histograms counting the number of times the actor already visited an image quadrant during a single episode. Image and memory information are merged through elementwise multiplication of their respective hidden layers [21].

4.3. Sequential Glimpse Classifier (SGC)

The second model is responsible for the actual classification of observation sequences. It accepts as input sequences generated by the *human*, *random*, *saliency*, or *reinforcement learning* methods.

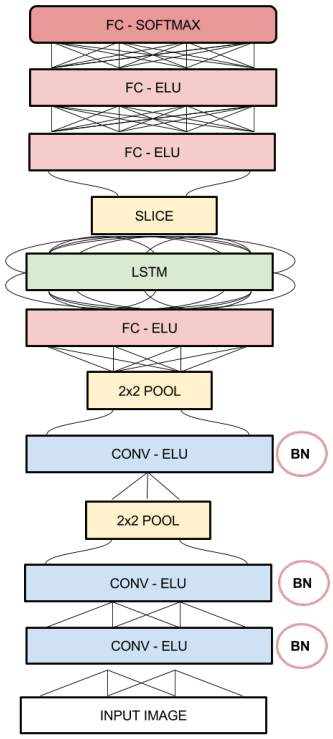

Figure 5. SGC Model Architecture.

Because the classifier performs sequential labeling of image data it consists of both convolutional and recurrent layers. Its architecture includes a series of convolutional layers followed by a single pooling layer. A fully-connected layer is used to map the pooled outputs to an activation vector that serves as input to the recurrent layer.

The design of separating convolutional inputs from a recurrent representation of the hidden state with a fully-connected layer is loosely inspired by [2]. The authors divide their model of visual attention into a "Glimpse Network" that maps image inputs to "glimpse vectors", and a recurrent layer that maps glimpse vectors to hidden states and outputs. In the SGC, the layers preceding the recurrent layer can be thought as forming a Glimpse Network, except convolutions, rather than a foveal sensor and rectified-affine transforms, are used to create glimpse vectors.

The recurrent layer itself is composed of 600 Long Short-Term Memory (LSTM) units. Like a traditional neuron, each unit of a LSTM stores some real-valued activation. However, incoming and outgoing connections to each LSTM unit is multiplied by the output of a weighted sigmoid function. Throughout backpropagation, weight configurations can be learned to "open" and "close" the cells by ensuring the values of the sigmoidal gates approach 1 or 0 respectively. As a result, each LSTM unit can perform three operations in response to every frame of a sequential input: They can *read* new activations into their memory, *write* their remembered value to the next processing layer, or *erase* their current activation [22].

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} sigm \\ sigm \\ sigm \\ tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

$$h_t^l = o \odot \tanh(c_t^l)$$

Figure 6. LSTM update equations. [23].

LSTMs are beneficial for sequential processing in that they allow the network to learn long-term dependencies between inputs at different timesteps. Because the 40x40 attentional window is too small to capture entire objects in the 256x256 CAT2000 images, the presence of LSTM units is motivated by the need to allow the SGC to integrate information from many glimpses.

Although the network inputs are naturally thought of as sequences, it must be noted that the actual classification task demands one label for the entire series of inputs, as opposed to individual labels per time-step. Two labeling schemes were considered. An early version of the SGC forward-propagated the outputs of the LSTM layer every timestep through the remaining fully-connected network, storing all 9 network output vectors. When the trial concluded, a global pooling layer was used to produce a final score for each class as the average value of each dimension of the 9 output vectors.

However, stockpiling observations before making a calculated decision at the end of each trial is

psychologically unrealistic [24] and appears to give empirically poor results. The final architecture follows the recommendation of [25], allowing only the final output of the LSTM to propagate to later layers.

The network's output layer features softmax nonlinearities and minimizes categorical cross-entropy loss.

	SGC Parameter Setting
No. Conv. Layers	3
No. Filters	10
No. FC Layers	3 (including glimpse layer)
No. Neurons per Layer	600
Pool size/stride	2
Regularization	1e-2
Batchsize	10

Table 1. SGC hyperparameter settings

4.4. All-Convolutional Baseline

The performance of the SGC was measured against a standard, deep CNN. The baseline model is structured similarly to *AlexNet* [1], featuring a variable number of alternating convolutional and pooling layers featuring dropout. Somewhat uniquely, this model includes only one fully-connected output layer and depends entirely on convolutional and pooling layers to create an internal representation of its inputs.

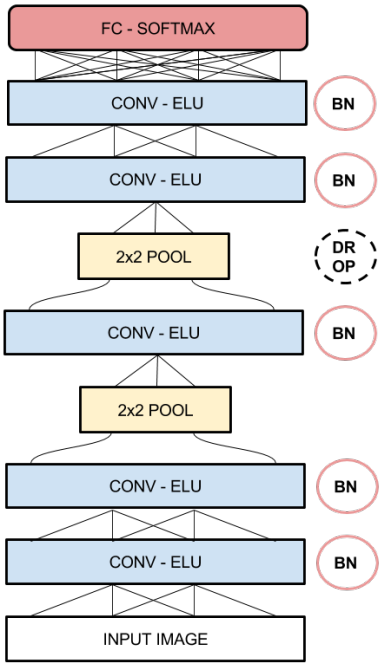

Figure 7. Baseline model architecture.

The baseline was first fine-tuned and evaluated on a well-understood dataset, CIFAR-10. After achieving satisfactory results, additional tuning of hyperparameters was performed for CAT2000. The major similarities and differences between the two baselines are reported below.

	CIFAR-10	CAT2000
No. Layers	**6**	**5**
Pool size/stride	2	2
Regularization	**1e-1**	**2e-1**
No. Filters	32	32
Dropout prob.	**0.5**	**0.7**
Batchsize	5	5

Table 2. Baseline hyperparameter settings.

The main differences between the CIFAR-10 and CAT2000 models occurred in attempting to combat overfitting on the smaller dataset. More aggressive regularization (both in dropout and L2 penalty) were applied to the CAT2000 learner, which also featured fewer convolutional layers. Because the CAT2000 training set was significantly smaller than CIFAR-10 (1,000 training examples versus 15,000 training examples), a less complex model performed better.

One of the largest challenges that emerged working with CAT2000 was the relatively small number of examples compared to the number of feature dimensions. This issue is explored further in the discussion.

4.5. Other Hyperparameters

Various other hyperparameter settings were found to be effective for all three models. All filters were sized 3x3, with a corresponding stride of 1, and pad 1. As a result, all image downsampling occurred during the dataset preprocessing stage or as the result of max pooling.

Based on preliminary results, the *elu* function contributed to model performance by keeping neuron activations from zeroing-out. It was used as the default nonlinearity for all layers except where otherwise noted (e.g., output layers, *tanh* squashing in actor-critic hidden layers, and LSTM outputs). Given that no *ReLU* units were used in the making of these models, the Xavier initialization was employed to determine weight-scales.

All updates were made using the Adam learning rule, which combines both momentum and gradient-scaling per feature. The learning rate α was set to 0.0001, the first moment exponential decay rate β_1 was set to 0.9, the second moment decay rate β_2 was set to 0.99, and the normalization constant ϵ was set to 0.00000001, per recommendations given in the original paper [26]. Furthermore, All convolutional layers in both the baseline and SGC underwent spatial batch normalization [27].

5. Results

Three experiments were performed in order to evaluate the difficulty of the CAT2000 scene classification task, the effect of different observation-extraction policies, and the role played by time information in observation-sequence classification. Each experiment took between 10-15 hours of training on a single-core machine.

5.1. Experiment 1

The All-Convolutional baseline was first fit to CIFAR-10 and CAT2000. Hyperparameters were chosen to improve validation accuracy during hold-out cross validation. In both cases, the training set consisted of half the total data and the validation and testing sets consisted of the remaining two quarters.

The baseline trained for 100 epochs on CIFAR-10 and achieved a validation accuracy of **69.8%** on the 60th epoch. CAT2000 consisted of much larger images (by a factor of 64) and was thus trained for only 10 epochs. On the 5th epoch, it achieved a validation accuracy of **31.6%**, but declined in performance with more training, as the model severely overfit the data.

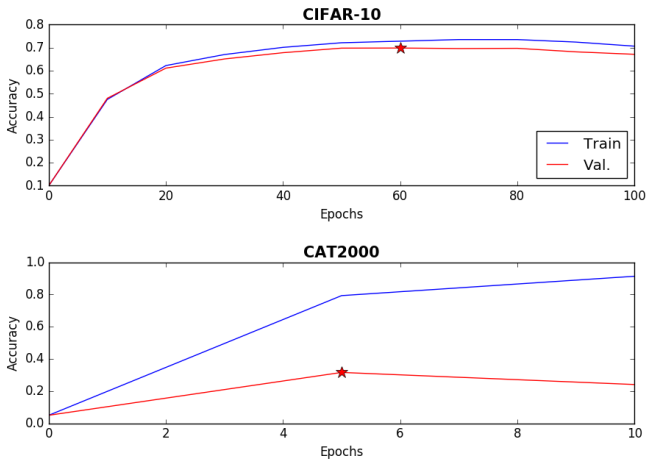

Figure 8. Baseline learning curves. Max marked with star marker.

5.2. Experiment 2

After establishing baseline accuracy for CAT2000, the four methods of observation-sequence extraction (human, saliency, random, reinforcement learning) were used to train and test the SGC.

Hold-out cross validation was again used to select hyperparameters to optimize performance on the sequences generated from human fixation points. Hyperparameters were not changed between experimental conditions. As in the first experiment, all learning was done on datasets split 50-25-25 between training, validation, and testing. However, for the *human*, *saliency*, and *random* conditions, the entire dataset consisted of approximately 32,000 examples. For the *agent* condition, there were 20,000 examples in the dataset. All four methods trained twice for 8 epochs. The mean of both runs and their standard deviations are shown below.

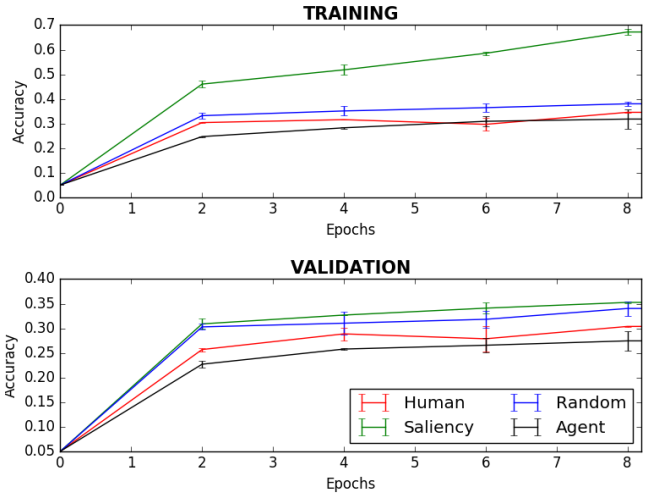

Figure 9. SGC learning curves with error bars.

All four methods performed close to, and in some cases above, the accuracy established by the baseline. With the exception of human data, the accuracy of each method improved every epoch. The sequence-extractor to obtain the highest validation accuracy was the *saliency* technique, achieving **35.27%** validation accuracy, while experiencing some overfitting. The random extractor also performed well, achieving an accuracy of **34.03%**, but managed to maintain a training accuracy in that same range. This order was preserved in the final test set evaluation, reported below.

	Human	Saliency	CACLA	Random
Test Acc.	30.2%	**37.82%**	24.7%	33.1%

Table 3. Final SGC accuracy on test set.

5.3. Experiment 3

Both *human* and *reinforcement* learning policies encode a sequence of particularly ordered observations. In contrast, the *random* and *saliency* methods select some arbitrary ordering of their selected observations.

Given that *random* and *saliency* outperformed the order-sensitive methods, it was natural to ask whether the recurrent layer of the SGC (included to capture time information) was actually playing an important role in classification.

A third network, the Long-Convolutional Network (LCN) was implemented in an attempt to imitate the general architecture and representational capacity of the SGC, without employing LSTM units. Instead of rolling

over each observation in series, the LCN features a Reshape Layer that unfolds sequence inputs into one long, rectangular image. Inputs can then be treated as a normal, static image would.

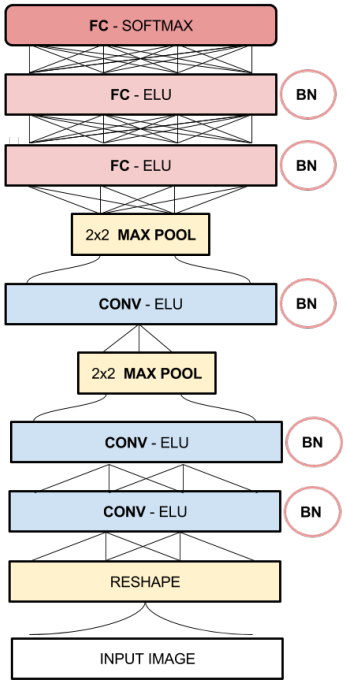

Figure 10. LCN Model Architecture.

The same sample of CAT2000 used for experiment 2 was again used to train and validate the LCN. In each condition, the LCN trained once for 8 epochs.

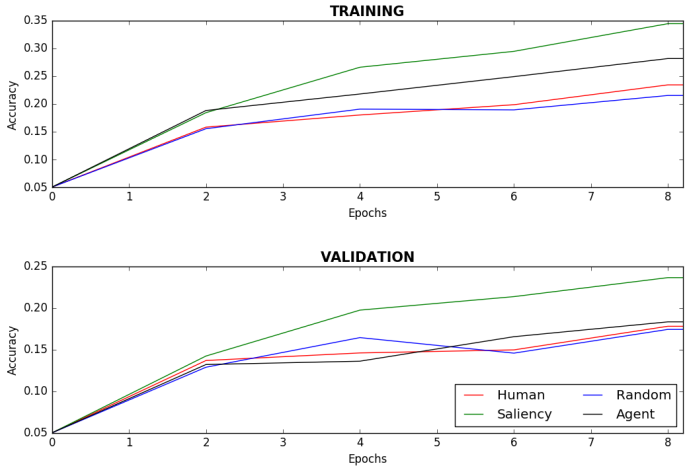

Figure 11. LCN learning curves.

On the validation set, the saliency method again emerged as the strongest approach by a considerable margin, achieving an accuracy of **23.63%**, with very little overfitting. No significant difference was observed between the performance of the three other methods. Final test set accuracies for the LCN are reported below.

	Human	Saliency	CACLA	Random
Test Acc.	18.73%	**23.57%**	18.95%	17.84%

Table 4. Final LCN accuracy on test set.

5.4. Qualitative findings

A confusion matrix for the best model tested (SGC with saliency extraction) was also generated in order to assess per-class performance [29].

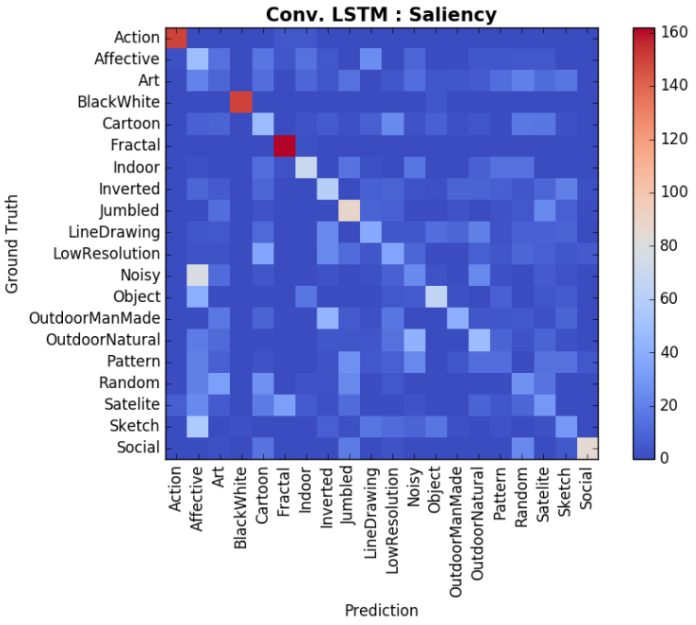

Figure 12. SGC confusion matrix using saliency maps.

The model proved to be most effective at classifying examples belonging to the **action** (93.16% accuracy), **fractal** (98.78%), **black-white** (96.77%), and to a lesser extent, the **social** (55.92%) categories. Good performance on the **fractal** category is unsurprising as each image is highly distinctive. Results with **black-white** are however, unexpected. Given that all images were converted to grayscale, the **black-white** images would seem to have little intra-group similarity, compared to other categories. One possible explanation for such high accuracy is that saliency maps were generated before conversion to grayscale, meaning that all observations in the **black-white** category must rely on contrast, rather than chromaticity. As such, the SGC may be learning to classify patches with significant changes between dark and light as belonging to **black-white**.

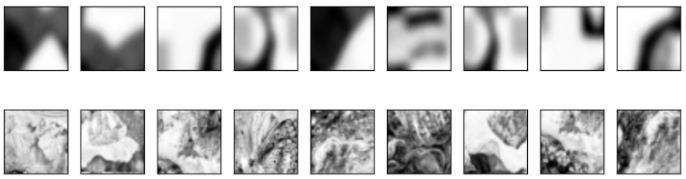

Figure 13. Observation sequence from black-white (top) and art (bottom) categories.

Art (5.98%) examples proved to be the most difficult to classify, and were often confused with **affective** scenes. **Pattern** (8.13%) examples also incurred high misclassification, being most typically labeled as **jumbled** or **noisy** images.

6. Discussion

These results demonstrate that carefully chosen models of visual attention can increase not only the efficiency, but also the accuracy of scene classification. The observation sequences used by the SGC consist of only 14,4000 pixels but appear to contain as much information relevant to image classification as the 65,536 pixels of a downsampled CAT2000 image.

Perhaps most valuably, the attentional approach allows for greatly increased sample efficiency. Whereas the deep CNN baseline overfitted the comparatively small CAT2000 training set, the attentional models were able to make the most of the 1,000 images by mining each for 10-16 training sequences. In this respect, attention can also be viewed as a form of data augmentation. Future work could focus on determining the maximum number of sequences that can be mined from training images, without introducing too much redundancy into the dataset.

Of the four attentional policies surveyed (human data, reinforcement learning, saliency maps, and random sampling), the simplest approaches proved the most effective. In particular, saliency maps appear to highlight important image sub-regions quickly, without the need to for lengthy model training (reinforcement learning) or human subject recruitment. Although the order for saliency-based observations was arbitrary, LSTMs still proved effective at modeling them in sequence. It is unclear if the SGC leverages time information specifically, or if LSTM units simply outperform the LCN due to having more representational "wiggle" than standard hidden units.

6.1. Directions for reinforcement models of attention

The CACLA-based attentional network succeeded in learning the reward signal from the human fixation data, but its final contribution to scene classification was disappointing. One possible explanation is that smoothed fixation density, unlike individual human trajectories, is strongly center-biased. The tendency for fixation density to occupy the middle of images is largely an experimental artifact caused by subjects instructed to begin each free-viewing trial by focusing on the middle of the display. As a result, the CACLA agent learns to sample a disproportionate number of observations from the middle of each image, making it less successful at decorrelating its inputs, or at least generating a more diverse sample.

However, CACLA may yet be a powerful algorithm for visual attention after adjustments have been made to the reward signal. Given the success of saliency-extraction, future work could train CACLA agents to optimize saliency instead of human fixation density. Using a reinforcement learning framework, rather than the simple saliency-maxima sampling presented here would have some advantages for artificial intelligence. In particular, an agent could be trained to attend to high-saliency areas, before being fine-tuned to a particular task (e.g., classification, object detection) using transfer learning.

Center-bias may also explain why the *human* extractor underperformed as well. Because human trajectories were clipped to their first 9 saccades, important attentional information from late in the trial has been discarded. Future models should allow for variable length sequences, or at the very least, samples should be taken from different points in the individual trajectories. Eye-movements recorded during free-viewing could also be qualitatively different than those driven by a vision task, such as search. Subsequent work should focus on saliency datasets in which subjects perform tasks similar to computer vision systems. But at present, very few sizable datasets exist.

7. Conclusion

This work presents selective processing of image sub-regions as an alternative to observing whole images for classification. Four policies for selecting ROIs were surveyed and random sampling of saliency map maxima emerged as the best. A relatively shallow convolutional, recurrent network featuring LSTM units learned to classify a heterogeneous dataset above the level of a fully convolutional network.

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

[2] Mnih, V., Heess, N., & Graves, A. (2014). Recurrent models of visual attention. In *Advances in Neural Information Processing Systems* (pp. 2204-2212).

[3] Goodrich, B., & Arel, I. (2012, June). Reinforcement learning based visual attention with application to face detection. In *Computer Vision and Pattern Recognition*

Workshops (CVPRW), 2012 IEEE Computer Society Conference on (pp. 19-24). IEEE.

[4] Ehinger, K. A., Hidalgo-Sotelo, B., Torralba, A., & Oliva, A. (2009). Modelling search for people in 900 scenes: A combined source model of eye guidance. *Visual cognition*, *17*(6-7), 945-978.

[5] Borji, A., & Itti, L. (2015). CAT2000: A Large Scale Fixation Dataset for Boosting Saliency Research. *arXiv preprint arXiv:1505.03581*.

[6] Niebur, E., & Koch, C. (1996). Control of selective visual attention: Modeling the" where" pathway. *Advances in neural information processing systems*, 802-808.

[7] Borji, A., Sihite, D. N., & Itti, L. (2013). Quantitative analysis of human-model agreement in visual saliency modeling: a comparative study. *Image Processing, IEEE Transactions on*, *22*(1), 55-69.

[8] Parkhurst, D. J., & Niebur, E. (2002). Variable-resolution displays: A theoretical, practical, and behavioral evaluation. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *44*(4), 611-629.

[9] Niebur, E. (2007). Saliency Map. *Scholarpedia*, 2(8):2675. From http://www.scholarpedia.org/article/Saliency_maps.

[10] Butko, N. J., & Movellan, J. R. (2008, August). I-POMDP: An infomax model of eye movement. In *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on* (pp. 139-144). IEEE.

[11] Bandera, C., Vico, F. J., Bravo, J. M., Harmon, M. E., & Baird, L. C. (1996, July). Residual Q-learning applied to visual attention. In *ICML* (pp. 20-27).

[12] Denil, M., Bazzani, L., Larochelle, H., & de Freitas, N. (2012). Learning where to attend with deep architectures for image tracking. *Neural computation*, *24*(8), 2151-2184.

[13] Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*.

[14] Jaderberg, M., Simonyan, K., & Zisserman, A. (2015). Spatial transformer networks. In *Advances in Neural Information Processing Systems* (pp. 2008-2016).

[15] Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

[16] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[17] Bylinskii, Z., Judd, T., Borji, A., Itti, L., Durand, F., Oliva, A., & Torralba, A. MIT Saliency Benchmark. Retrieved from http://saliency.mit.edu/results_cat2000.html, Mar. 6, 2016.

[18] Battenberg, E., Dieleman, S., Nouri, D., Olson, E., van den Oord, A., Raffel, C., Schluter, J., Kaae Sonderby, S. (2015). Lasagne. From http://lasagne.readthedocs.org/en/latest/index.html.

[19] Van Hasselt, H., & Wiering, M. A. (2007). Reinforcement learning in continuous action spaces. In *Approximate Dynamic Programming and Reinforcement Learning*, 2007. ADPRL 2007. IEEE International Symposium, on (pp. 272-279). IEEE.

[20] Wiering, M., & Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, Learning, and Optimization*, *12*.

[21] Oh, J., Guo, X., Lee, H., Lewis, R. L., & Singh, S. (2015). Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems* (pp. 2845-2853).

[22] Graves, A. (2012). *Supervised sequence labelling* (pp. 5-13). Springer Berlin Heidelberg.

[23] Fei-Fei, L., Karpathy, A., & Johnson, J. (2016). Lecture 10: Recurrent neural networks. *CS231N: Convolutional Neural Networks for Visual Recognition*.

[24] Brockmole, J. R., & Irwin, D. E. (2005). Eye movements and the integration of visual memory and visual perception. *Perception & Psychophysics*, *67*(3), 495-512.

[25] Raffel, C. (2015). Recurrent networks in Lasagne. Mount Sinai Hammer Lab seminar. From http://colinraffel.com/talks/hammer2015recurrent.pdf.

[26] Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization.*arXiv preprint arXiv:1412.6980*.

[27] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

[28] Segal, I. Y., Giladi, C., Gedalin, M., Rucci, M., Ben-Tov, M., Kushinsky, Y., & Segev, R. (2015). Decorrelation of retinal response to natural scenes by fixational eye movements. *Proceedings of the National Academy of Sciences*, 112(10), 3110-3115.

[29] Confusion Matrix (2014). *Scikit-Learn*. From http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html.