Recurrent GlimpseNet

Isaac Supeene
Department of Computer Science
Stanford University
isupeene@stanford.edu

Kaiyue Wang
Department of Computer Science
Stanford University
garywang@stanford.edu

Abstract

Current approaches to image classification 'brute force' the problem by passing convolution filters over every nook and cranny of an image. Humans and other animals, however, take a much more parsimonious approach by intelligently directing their gaze towards regions of interest. Our work is built on an existing implementation of this concept: MGNet introduced by Tan et al. [4]. We explore the capabilities and limitations of MGNet and shed some light on the root cause of these limitations through model inspection and visualization. Additionally, we show a minor improvement in performance by replacing their spatial clue mechanism with a 2d positional encoding.

1. Introduction

Our main idea is to develop a more efficient model for image classification inspired by animal visual behavior. Rather than running a deep CNN over a full-resolution image, we approach the problem by letting the model see a low-resolution version of the full image, followed by a sequence of smaller, higher-resolution glimpses.

The state-of-the-art in models of this type is the Multi-Glimpse Network (MGNet) by Tan et al. [4], which achieves slight performance improvements when compared head-to-head against various flavours of ResNet. As a concrete example, MGNet using ResNet18 as the backbone achieves better accuracy with less latency and computation compared to a vanilla ResNet18. Additionally, MGNet shows improved robustness against adversarial attacks.

Our aim is to try to improve on the shortcomings of MGNet, particularly the slow training speed of the localization network, which the authors of [4] cite as the main barrier to further development, and the main factor forcing them to train on a smaller 100-class version of ImageNet [2].

We achieved moderate success in improving on MGNet by introducing a 2d spatial encoding and freezing the feature extraction backbone weights, which slightly increased the accuracy of the network. Additionally, we employed a number of model inspection techniques to gain insight into the nature of the localization network woes. These inspections, combined with the results from a variety of experiments, demonstrate that unreliable gradient signal to the localization network is the main culprit, which suggests directions for future research.

2. Related Work

The original GlimpseNet was introduced by Hang et al. [1], in which they applied the technique to the analysis of mammogram images. In their work, the localization network was trained separately from the classifier, using ground truth pixel-wise salience labels.

MGNet [4] represented a new approach to the problem by using spatial transformers [3], which allows backpropagation through the affine transform, thus making it possible to train the entire network end-to-end, including the localization network that creates the glimpses.

Although MGNet achieved remarkable success on the ImageNet100 dataset [2], they were unable to make the training process efficient enough to make it feasible to train on the full ImageNet dataset. The proximal cause of the extremely slow training of the localization net is the gradient rescaling that is applied to the gradients received by the localization net - the gradients are scaled down by a factor of 50. The reason that Tan et al. give for this rescaling factor is an exploding gradient problem in the Localization Network. We will shed further light on this issue in the present work.

3. Data

We use the same ImageNet100 [2] used by [4], in order to get comparable results. The classes comprise mainly birds, lizards, bugs, and snakes, in addition to triceratops and trilobyte.

4. Methods

Our work involved two separate types of experiment: model training, and model inspection.

4.1. Model Training Experiments

Our model was based on the MGNet architecture described in [4]. We will first briefly review the MGNet architecture, then describe our additions.

4.1.1 MGNet

The MGNet can be broadly understood in terms of 5 key components, and the recurrent architecture in which they are embedded. 1

The first component is the Convolution Backbone, which is generally some variant of ResNet. The classification head is removed, and the downstream components consume the output of the final average-pooling layer. The next component is the Feature Fusion layer, which uses a self-attention block to combine the backbone outputs from all glimpses that have been taken so far. The output of the Feature Fusion layer is consumed by the Classification Head and the Localization Network, which generates the scale and translation parameters (s, tx, and ty) for the next glimpse. Finally, the Glimpse Generator performs the differentiable affine transform given the affine parameters from the Localization Network. The resulting glimpse of the image is then provided to the backbone, and the cycle can continue indefinitely, typically for a pre-set number of glimpses.

The output of the final classification head is used as the classification for accuracy purposes, but a loss term is applied at each iteration. Additionally, a separate classification head is attached directly to the output of the backbone and used to provide a auxiliary cross-entropy loss term for each glimpse in isolation.

4.1.2 Our Contributions

We built several additions on top of MGNet.

The first (and most successful) addition we made was the replacement of the 'spatial clue' mechanism with a 2d positional encoding. Recall that the MGNet's spatial clue is simply to append the affine transform parameters to the output of the backbone. Instead of this, our model adds a positional encoding similar to that proposed by Vaswani et al. [5]. The distinction is that we devote half the dimensionality of the input vector to encoding the height dimension, and half to encoding the width dimension, so that full 2d spatial information is encoded. Additionally, the height and width encodings are interleaved so that when using multi-headed attention, each head doesn't simply see one dimension's encoding.

$$p(x,y) = \begin{bmatrix} \sin(\omega_1 x) \\ \cos(\omega_1 x) \\ \sin(\omega_1 y) \\ \cos(\omega_1 y) \\ \\ \sin(\omega_2 x) \\ \cos(\omega_2 x) \\ \sin(\omega_2 y) \\ \cos(\omega_2 y) \\ \vdots \\ \sin(\omega_{d/4} x) \\ \cos(\omega_{d/4} x) \\ \sin(\omega_{d/4} y) \\ \cos(\omega_{d/4} y) \end{bmatrix} \tag{1}$$

where

$$\omega_k = \frac{1}{10000^{4k/d}}$$

The second addition concerns the localization network. One of our theories was that the localization network was not getting sufficient spatial information to determine where to glimpse, since its only input was the output of the feature fusion layer, which consumes feature vectors from the backbone, which are ultimately the result of an average-pooling operation. This seems very likely to lose a lot of spatial information, so we developed a way for the localization network to access the spatial information more directly. Our new localization network consumes information directly from intermediate layers of the ResNet backbone, which go through a spatiotemporal attention layer using 2d positional encoding.

To explain this with a concrete example, suppose we have 224x224 images, scaled down by a factor of 3/7 to a size of 96x96. ResNet reduces the height and width dimensions each by 5 factors of 2, so after the 4th and final layer, we have a Cx3x3 tensor. This would give us 9 C-dimensional vectors to use self-attention on, which are augmented with the 2d positional encoding we describe above. Additionally, multiple glimpses may be taken in a single inference. At each timestep t, all 9t tensors from all prior glimpses are passed to the self-attention block. The output of the self-attention block is then passed through a fully-connected layer to produce the affine transform parameters.

The third addition we made was replacing the single-headed attention in the Feature Fusion layer with multi-headed attention. As will be further explained in section 5.2.3, one of our observations from inspecting the gradient flow in Localization Network was that glimpse generation is unintuitive and the resulting gradient direction is very noisy. One of our hypotheses was that the noise may come from the model not exploring enough using enough information from the prior glimpse features, which lead us to the idea

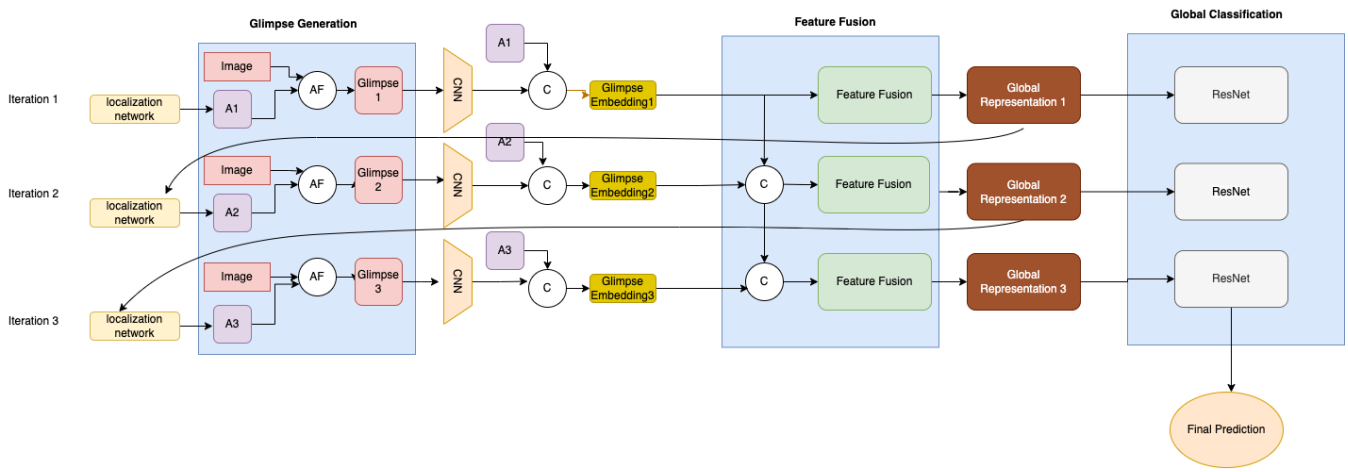


Figure 1. Recursive GlimpsNet architecture

of changing from single-headed to multi-headed attention. The intuition came from Transformer [5] where they repeat the attention module multiple times in parallel and combined the output from each attention head together to produce the final feature vector. This helps the transformer to capture richer interpretations of the sequence, and we hope it will help with capturing richer features for the localization model and reduce the noise.

The forth exploration was done on freezing the weight of pre-trained feature extractor. The intuition here is that glimpse generation process is noisy, If we train the backbone feature extractor along side with feature generation we might bias the feature extractor to pick up the noise generated by glimpse. To avoid this, Given that the existing pre-trained backbone model is already by itself a decent enough feature extractor we would want to freeze the weight and therefore prevent the feature extractor from biasing towards noisy glimpses.

4.2. Model Inspection Experiments

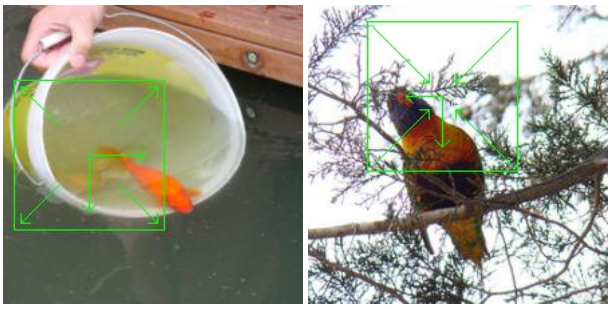


Figure 2. Example of affine transform gradient visualizations. Gradients of tx and ty are indicated by the two center arrows, and the gradient of s is indicated by the four corner arrows.

We used several different methods to inspect the model and the training procedure, leveraging the fact that the model produces an interpretable intermediate result.

Our first and simplest approach was to log certain statistics of the weights, gradients, and outputs of the Localization Network. Since there is allegedly an exploding gradient problem that necessitates the gradient rescaling factor, this seemed like a reasonable place to start investigating. We added code to output the max, min, mean, and standard deviation of weights and gradients in the Localization Network to verify the existence of the problem and get insight into how it manifests.

The next improvement we made was adding a visualization of the gradient with respect to the affine parameters. The MGNet codebase comes with code for visualizing individual glimpses, and we modified this code to visualize the gradients using arrows indicating which direction the gradient is trying to move the glimpse.

Measuring and visualizing the Localization Network in its natural habitat was quite informative, and the next logical step was to reach in and start manipulating the internals of the network. We added the capability to inject specific affine transform parameters to manipulate individual glimpses. This allowed us to sweep the glimpse across the image and visualize how the gradients of the loss with respect to the transform parameters change.

Lastly, we performed an experiment where we ran eval on a trained model, but replaced the Localization Network with code to generate a random glimpse. This allowed us to see how the learned glimpse generation compares against a random baseline.

Table 1. Experimental results of our method on MGNet Improvements showing accuracy on the *validation* split.

Method	ImageNet 100
	Accuracy
baseline	83.99
2d-spatial-clue	84.16
2d-spatial-loc	83.81
2d-spatial-loc-s7	62.00
2d-spatial-loc-s7-no-aux	67.45
multihead-attention	83.72
pretrained	85.14
pretrained-frozen	84.75

5. Experiments

5.1. Model Training Experiments

The baseline is the MGNet described in [4]. It uses a ResNet 18 backbone, image scaling factor of 2.33, gradient scaling factor of 0.02, and 4 glimpses per inference. It was trained for 400 epochs using SGD, with the OneCycleLR learning rate scheduler. The allowed scale range for each glimpse was [0.2, 0.5]. The auxiliary loss factor α was 0.6. All the other experiments use the same settings unless otherwise noted.

'2d-spatial-clue' is identical to baseline, except that it uses our 2d positional encoding.

'2d-spatial-loc' uses our updated Localization Network with spatiotemporal attention. The Localization Network uses the output from the 4th layer of the backbone.

'2d-spatial-loc-s7' uses our updated Localization Network, and also uses an image scaling factor of 7. Concretely, this means that the input to the backbone is 32x32. By the time we get to layer 4, we would be at 1x1, so for this experiment, we use the layer 3 output of the backbone as the input to the Localization Network, which is 2x2. Additionally, we change the allowable scale range to [0.143, 0.99]. 0.143, or $\frac{1}{7}$, is chosen so that the smallest possible glimpse permits a full-resolution view of part of the image. This experiment was chosen so that glimpses would be extremely impactful for model performance, to make it easier to learn interesting glimpses. Unfortunately, we observed exactly the opposite: the value of s averaged 0.96 after training converged, so all glimpses were essentially of the whole image.

'2d-spatial-loc-s7-no-aux' is like '2d-spatial-loc-s7', except the auxiliary loss has been removed. We hypothesized that the reason our other models tend to maximize glimpse size is that the auxiliary loss encourages each glimpse to be 'selfish' and attempt to maximize the classification accuracy for the individual glimpse. By removing the auxiliary loss, we found that the average glimpse scale became 0.55, and accuracy improved. Surprisingly, however, we found that the affine transform parameters converged to essentially constant values: every glimpse was a 0.55-scale window in the center of the image. Some examples are shown in Figure 3.

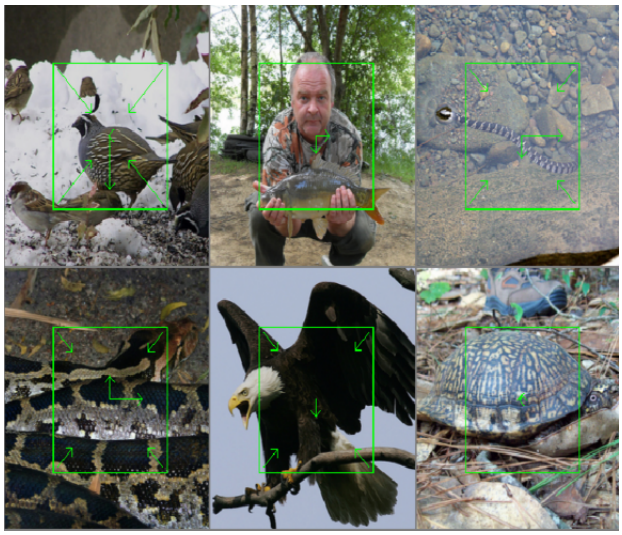


Figure 3. Glimpse visualizations for 2d-spatial-loc-s7-no-aux.

'multihead-attention' updated the single attention head in localization network to 4 attention head in order to capture richer interpretation of the feature. In our experiment we observe our model perform on-par with the baseline without significant improvement in accuracy. This tells us that richer interpretation might not be the main issue to the randomness.

'pretrained' used a pretrained backbone instead of a randomly initialized backbone. We also used the 2d spatial clue for this experiment. Unsurprisingly, we find that using a pretrained model with richer image representations built-in from the start outperforms a model trained from scratch.

'pretrained-frozen' used a pretrained backbone with frozen weights as the feature extractor. All weights except for the weights in Layer 4 were frozen. It saves 20 percent trainable parameter as well as getting a better validation accuracy. To explore its effect on glimpse generation, We plot out the gradient visualization of affine transformation parameter used to generate glimpses 4 and observed that the gradient slope using this method is somewhat less noisy and has more intuitive trend compare to that generated by trainable feature extractor models 5.

5.2. Model Inspection Experiments

5.2.1 Weight and Gradient Logging

By logging the weights and gradients of the Localization Network with the gradient rescaling disabled, we validated

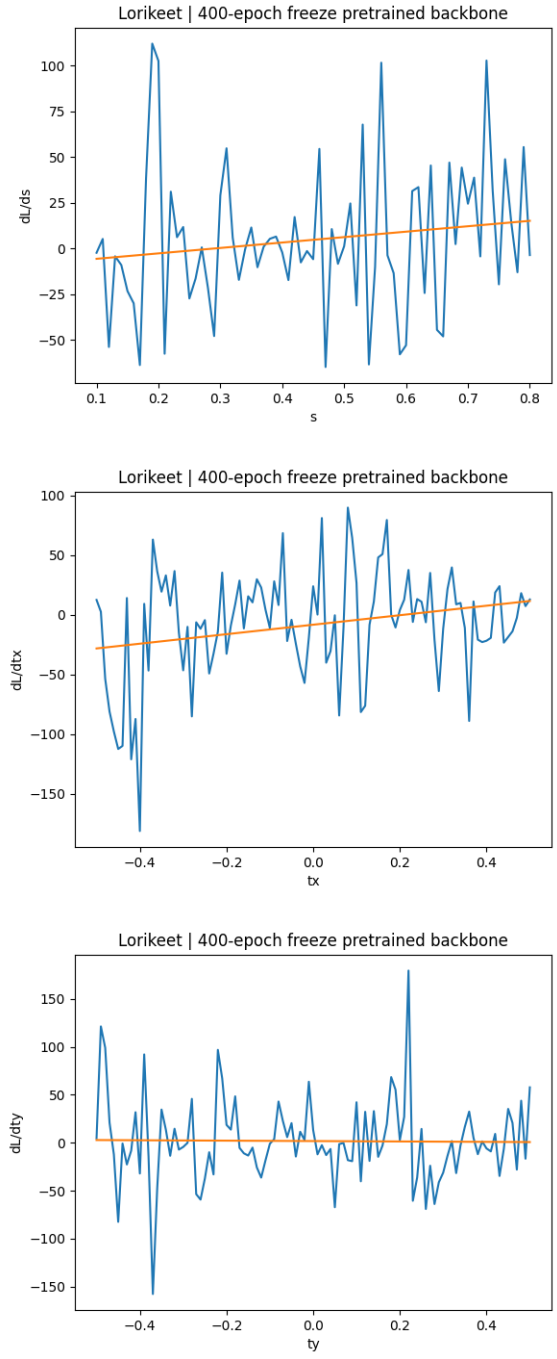


Figure 4. Affine transform parameter sweep over Lorikeet with freezing feature extractor.

that the weights would explode without the gradient rescaling. We also observed that the s parameter always converges to a constant value very close to the maximum allowable value. We confirmed that this is caused by the auxiliary loss by comparing 2d-spatial-loc-s7 with 2d-spatial-loc-s7-no-aux. The first one used the auxiliary loss and the second did not, and the result was that the in the first case, the average value of s was 0.96, whereas in the second, it was 0.55.

5.2.2 Random Glimpse Generator

In order to determine if the Localization Network was learning anything useful at all, we decided to pit it against a baseline of simply generating random glimpses. Using our baseline model with random glimpses yielded a validation accuracy of **81.33%**, nearly 3% below the baseline accuracy using the trained Localization Network. Baseline accuracy with no glimpses at all is only 75.2%, so the Localization Network is responsible for about 30% of the improvement we get from glimpses. What this demonstrates is that the training of the Localization Network is not totally ineffective, which is something that we might have had serious questions about when looking at the results of the following model inspections.

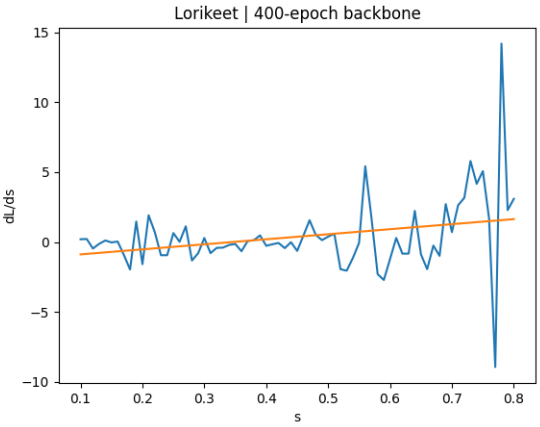
5.2.3 Transform Parameter Sweeps

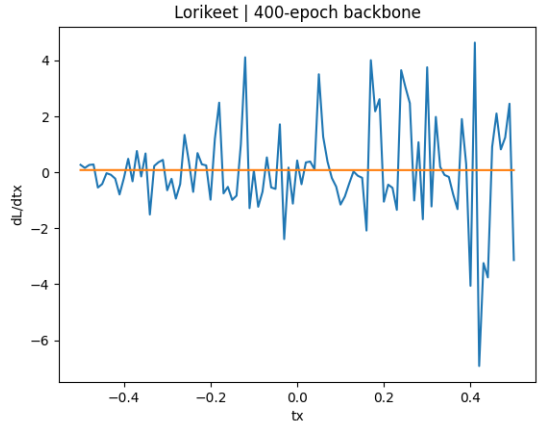
Using the following two images, we experimented with sweeping over the s, tx, and ty parameters of the affine transform.

Gifs of the gradient visualization are included in the Supplementary Materials.

For each of the three transform parameters, we plotted the gradient of the loss with respect to that parameter as we swept across a range of values. The graph we would expect to see is an approximately upward sloping graph. Using tx as an example to convey the intuition here, imagine we start with a glimpse on the left of the subject of the image, corresponding to a low value of tx. Ideally, the gradient would be pushing the glimpse to the right to encompass more of the subject. This corresponds to a negative gradient of the loss with respect to tx, since increasing tx should decrease the loss. As we sweep across the image, and start to move away from the subject as we move to the right, we should see a gradient trying to push the glimpse back to the left, corresponding to a lower value of tx. This would mean a positive gradient of the loss with respect to tx.

Figures 5 and 6 show the relation between s, tx, and ty for the goldfish image and the lorikeet image. It is abundantly clear that the noise substantially outweighs the signal, and in some cases, as shown by the line of best fit we include in the graph, the mean signal is the opposite of what we would expect, and we see a negative slope instead of a positive one.





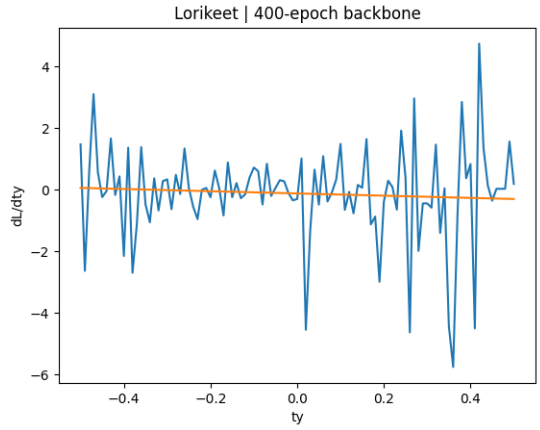
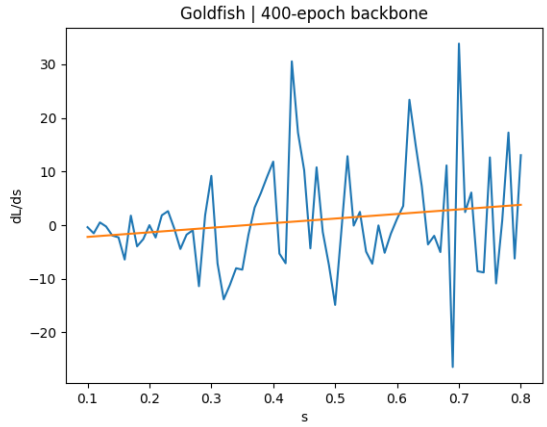
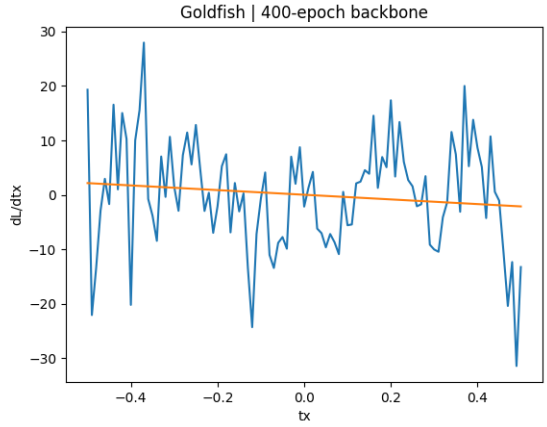


Figure 5. Affine transform parameter sweep over Lorikeet.





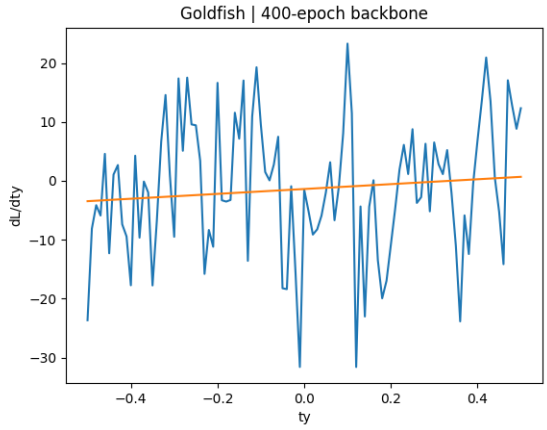


Figure 6. Affine transform parameter sweep over Goldfish.

6. Conclusion

In this paper, we explore additions to the MGNet architecture, and gain insight into the limitations of training the Localization Network by applying model inspection techniques.

We find that even very significant increases in the capacity of the Localization Network do not produce a meaningful improvement in accuracy. This suggests that there is a problem with the training signal that the Localization Network receives. Furthermore, in many cases, we find an extreme lack of variation in the size and location of glimpses, indicating that individual differences in images and the ap-

Figure 7. Images used for affine transform parameter sweep.

propriate glimpse for the image is drowned out by the noise, and only very general statistics of which glimpses are useful are being captured.

These results are consistent with our model inspection experiments, which show that there is a large amount of noise in the gradients of the loss with respect to the affine transform parameters. This suggests that follow-up work can improve upon the MGNet architecture by either overcoming the noisy gradients, or by using an alternate method of training the localization network, such as reinforcement learning.

References

[1] Glimpsenet. http://cs231n.stanford.edu/reports/2017/pdfs/517.pdf, 2017. 1

[2] Imagenet100. https://github.com/siahuat0727/MGNet/blob/main/tools/imagenet100.txt, 2021. 1

[3] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. 1

[4] Sia Huat Tan, Runpei Dong, and Kaisheng Ma. Multi-glimpse network: A robust and efficient classification architecture based on recurrent downsampled attention. 2021. 1, 2, 4

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 2, 3