Embracing New Techniques in Deep Learning for Estimating Image Memorability

Coen D. Needell[1] and Wilma A. Bainbridge[1]

[1]University of Chicago

Abstract

Various work has suggested that the memorability of an image is consistent across people, and thus can be treated as an intrinsic property of an image. Using computer vision models, we can make specific predictions about what people will remember or forget. While older work has used now-outdated deep learning architectures to predict image memorability, innovations in the field have given us new techniques to apply to this problem. Here, we propose and evaluate five alternative deep learning models which exploit developments in the field from the last five years, largely the introduction of residual neural networks, which are intended to allow the model to use semantic information in the memorability estimation process. These new models were tested against the prior state of the art with a combined dataset built to optimize both within-category and across-category predictions. Our findings suggest that the key prior memorability network had overstated its generalizability and was overfit on its training set. Our new models outperform this prior model, leading us to conclude that Residual Networks outperform simpler convolutional neural networks in memorability regression. We make our new state-of-the-art model readily available to the research community, allowing memory researchers to make predictions about memorability on a wider range of images.

Background

A person's ability to remember is thought of as something that varies from person to person. Some people have "good" memories, others forget images easily. While this is certainly true, recent research has found that the things which are being remembered have a certain power over our memories; images can be intrinsically memorable and forgettable. The images that are remembered and forgotten are highly consistent across people, and each image can be assigned a memorability score (Bainbridge 2019). These "memorability scores" are determined experimentally by testing how well a large group of people remember an image. This is an expensive and time-consuming process, especially at scale. In this paper, we propose a new method for evaluating the memorability of images.

Memorability scores are conventionally generated using a memory test. A subject is given a series of images and is asked whether or not they have seen them before. By aggregating how human participants score on each image, researchers can generate an estimate for the ground truth memorability of that image. Memorability scores generated from a subset of the participants are a good estimator of the memorability scores from the complement of that subset (measured by a high Spearman rank correlation between them) (Goetschalckx and Wagemans 2019). The question then is how do these memorability scores arise, and why are they so consistent across people?

High-level visual areas and memory areas in the brain have been shown to compose into a representational space related to memorability. In this space, memorable items are clustered together, while forgettable items have more disparate representations (Bainbridge and Rissman 2018; Bainbridge, Dilks, and Oliva 2017). This representational space hypothetically reflects how the brain prioritizes information for memory. Memorable items are given high priority, and forgettable items are given low priority (Xie et al. 2020).

Knowing that memorability is intrinsic to the image, then we should be able to derive the memorability directly from the image. The earliest work in applying computer vision to memorability used hand-engineered Histograms of Oriented Gradients (HOG), and dense Scale Invariant Feature Transforms (dense-SIFT) (Khosla, Bainbridge, et al. 2013). While these methods can

be very effective, they are hand-engineered, they need to be tuned by a human, instead of being determined algorithmically and thus could have unintended biases. On the other hand, they are easy to interpret, and can be used to modify existing images for memorability. As end-to-end trained vision techniques like convolutional neural networks (CNNs) became available, the problems associated with HOGs and dense-SIFT models could be avoided, giving us better performing estimators at the cost of interpretability.

Previous methods for estimating memorability have been proposed using convolutional neural network regressions. The convolutional neural network is a deep learning technique that scans over an image in small regions, and optimizes the process of analyzing the regions, rather than optimizing over every data dimension (every pixel and channel). This generally yields good performance while lowering training costs and preventing vanishing gradients. The most well-known of these is MemNet (Khosla, Raju, et al. 2015), although it does have a few challengers which have employed attention models (Fajtl et al. 2018), image captioning features (Squalli-Houssaini et al. 2018), and transfer learning (Basavaraju, Gaj, and Sur 2019). MemNet's memorability estimations had a rank correlation of 0.57 with held-out ground truth scores from its training dataset and is the most commonly used neural network regression for this purpose, and has been used and cited in several research papers since publication. However, some roadblocks have emerged for researchers trying to use MemNet. MemNet was built in Caffe, a deep learning framework which has been defunct since shortly after MemNet's publication. MemNet was trained on LaMem (Khosla, Raju, et al. 2015), a dataset to which public access has been restricted. In addition to these factors, MemNet uses techniques which are no longer preferred for computer vision regression. Techniques like Residual Neural Networks, Semantic Segmentation, and new optimization methods have been developed and become widespread in the meantime. This project is focused on constructing a new state-of-the-art model for predicting image memorability based on recent advances in computer vision and our understanding of memorability itself. Using these new techniques, as well as a combination dataset, this new model can also be more generalizable to the image-space.

First, we will reproduce MemNet on a modern framework, and explore its statistical properties in a controlled environment. Then, we will outline deep learning methods that utilize these new

techniques for the same problem, and show that their use has resulted in large performance increases. The techniques we will discuss are based around residual neural networks (ResNets), a specialization of convolutional neural networks. Finally, we will explore these new models, and show that the addition of a residual neural network gives the model access to higher level conceptual features.

Models

We will compare three architectures, two of which are novel, that serve as the backbone for five models. First we will examine the MemNet architecture, for which we will compare MemNet as downloaded in its original CaffeModel form and we also reimplemented its architecture using the more modern PyTorch (P. Team n.d.) framework. Second we will describe our new ResMem architecture, an architecture that uses residual neural networks as a secondary feature. It was used to create two models, called ResMem and ResMemRetrain (more on the differences between them later). The third architecture we implement is M3M, which introduces a tertiary feature, semantic segmentation. Our new architectures also employ relatively simplified preprocessing steps, which avoid the assumption present in prior models that memorability is preserved across cropped and normalized versions of images.

MemNet

First, we will look at MemNet's architecture (Khosla, Raju, et al. 2015). The main innovation in MemNet was the use of convolutional neural networks for feature decomposition. Convolutional neural networks were first created in the 1980s (Fukushima 1980), but did not see widespread use until the 2000s when GPUs became widespread, and a GPU implementation was written for CNNs (Chellapilla, Puri, and Simard n.d.). Then, the final piece needed to use CNNs at the modern level was to implement backpropagation (Cireşan et al. 2010). Backpropagation speeds up neural network training by taking the derivative of the network's output with respect to each of the parameters of the model, and uses that to decide which parameters to try in the next step. This innovation was shortly followed up by AlexNet (Krizhevsky, Sutskever, and Hinton 2012), a model that used these techniques for image classification.

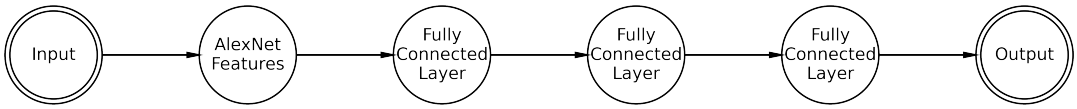


Figure 1: A skeleton diagram showing MemNet's architecture. It is characterized by a deep CNN followed by three fully connected layers.

MemNet is designed after AlexNet (fig. 1), which consisted of 5 convolutional layers (some of which are pooled) followed by 3 fully connected layers. In reality, MemNet's convolutional layers are not quite the same as AlexNet's. The number of filters and channels in each layer are different, and the number of neurons in the fully connected layers are different as well, but the overall architecture is the same. The intention in building MemNet was to leverage AlexNet's unparalleled efficiency (for the time) at extracting features from images.

ResMem

Historically, the next big advancement in using neural networks for computer vision was residual neural networks (ResNets). ResNets are built by taking convolutional layers and connecting them across levels with a "residual connection." The design builds on constructs observed in pyramidal cells in the cerebral cortex. In reality, this is implemented as adding the previous layer (possibly scaled to the proper size) to the convolution at each step (He et al. 2015).

The advent of ResNets allowed for convolutional neural networks to be built deeper, with more layers, without the gradient vanishing. We will take take a pre-trained ResNet from the `torchvision` model zoo (P. Team n.d.), specifically, the whole model and not just the convolutional features, and add it as an input feature for our regression step. The model we use is called "ResNet-150" and was originally trained for image classification using ImageNet, an industry standard dataset for image classification (Deng et al. 2009). This model consists of 150 layers of residual neural networks, followed by a short fully connected section that produces a 1000-length feature vector. Since the ResNet was previously trained on a semantic classification problem, this means that the ResNet features we are using are a semantic representation of the input image. We combine this ResNet feature with AlexNet features to create ResMem (fig. 2).

We can use this structure to make two models. In one, the optimization algorithm will not have

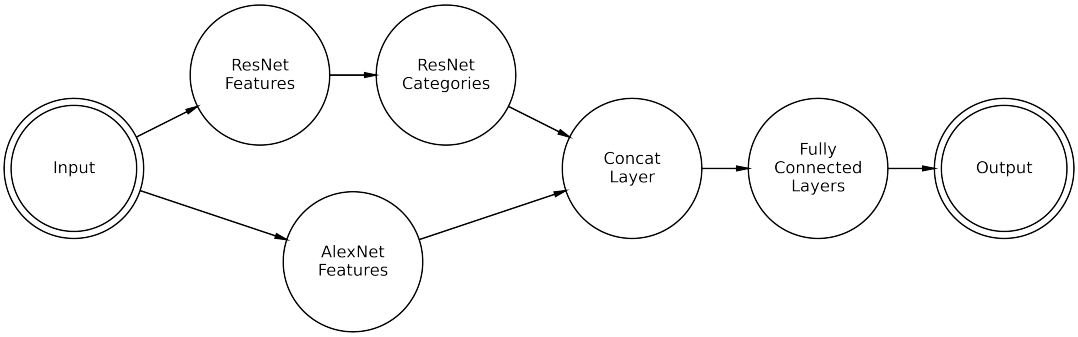


Figure 2: A skeleton diagram for ResMem. The input image is run through a deep residual neural network, and an AlexNet feature architecture, and the two features are combined, then run through a fully connected deep neural network.

access to the ResNet-150 parameters. This means that the intermediate ResNet features will be categorical data as the original was trained on ImageNet (Deng et al. 2009). In the other, which we will call ResMemRetrain, we will allow the ResNet part of the model to retrain for the memorability task. This will allow the model to specialize the ResNet features for this task, but we will lose the assumption that the ResNet features are semantically in line with ImageNet.

M3M

Now, we try to add a third feature to our model based on Semantic Segmentation (fig. 3). Semantic Segmentation is a common computer vision task where a model takes an image and classifies the object represented in each pixel. The segmentation model we are using is called fcn_ResNet-50, and also comes from the PyTorch model zoo (P. C. Team n.d.). Generally this will be built without any retraining of the two pretrained models, as semantic segmentation is a very high memory usage task, and would overload standard graphics cards.

The hypothesis is that by adding another semantic feature to the model, the overall accuracy of the model will improve. Where the classification model ResNet-150 returns a 1000-vector with estimated probabilities for each category, fcn_ResNet-50 returns an array the same size as the input image, but instead of having three channels (Red, green, and blue) it has 21 channels, with estimated probabilities for each category for each pixel. This is considerably more semantic information, and also contains coupled spatial-semantic information.

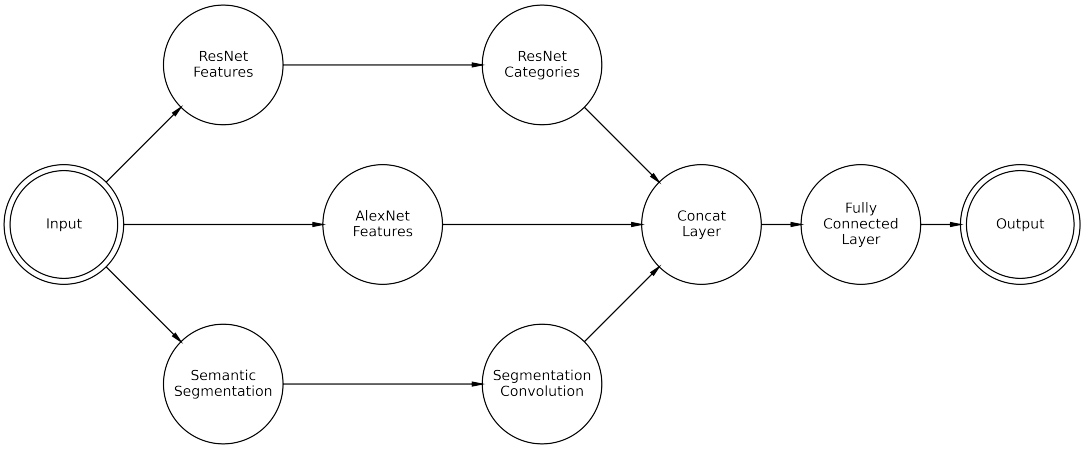


Figure 3: A skeleton diagram for M3M. In addition to the ResNet and CNN, a Semantic Segmentation feature is also sent to the fully connected deep neural network. Since semantic segmentation produces very high dimensional outputs (on the order of one million dimensions), this feature must also be passed through a convolutional neural network to down-scale it.

Of important note, Semantic Segmentation outputs a data array with more channels than a normal image, so it cannot be reshaped directly into a standard linear layer. To solve this we will use a small convolutional neural network to process the segmentation into more simple convolutional features.

Data Sharing

One impetus for this project is that MemNet is no longer accessible. The online demonstration server went down sometime in 2019, and at the time of writing has not been restored. You can still download the raw data needed to reconstruct it, but the code for implementing MemNet and full methodological information is inaccessible. To reproduce the results here, we had to rely on contacts associated with the original project to get full methodological information.

Many researchers used MemNet in their research, either to make predictions about memorability, or to control for memorability when studying other phenomena. Part of the success that MemNet had was tied to its availability. The pre-trained model was available online, and although there were limited instructions on how to use it, many researchers incorporated it into their work. Since we feel that having MemNet available was a boon to the visual cognition community, from the

beginning of this project we have had the goal of making an easy-to-use distribution of whatever model we create. A Python package that implements ResMem can be downloaded from PyPi using the line:

```
pip install resmem
```

It requires Numpy (Harris et al. 2020), Torch (P. Team n.d.), and Torchvision (P. C. Team n.d.) to run, which `pip` will download automatically. It requires minimal Python experience to operate, and we have also created a web application at The Brain Bridge Lab Website (https://brainbridgelab.uchicago.edu/resmem/), where you can upload an image and quickly get back its estimated memorability score.

Data Sources

The original MemNet was trained on a dataset called LaMem (Khosla, Raju, et al. 2015), the Large Scale Memorability Dataset. LaMem contains 58,741 images, all labeled with the average memorability for each image. If you split all of the human participants into two groups, and take the Spearman rank correlation between each group's average memory performance for a given image, you get 0.67 suggesting that people are indeed highly consistent in the images they remember and forget. The images in LaMem are a compilation of a few more well-known datasets, the MIR Flickr Image Set (Huiskes and Lew 2008), the AVA (Aesthetic Visual Analysis image set) (Murray, Marchesotti, and Perronnin 2012), the Affective Image Set (Machajdik and Hanbury 2010), a couple of saliency image sets (Judd et al. 2009), (Ramanathan et al. 2010), and (Xiao et al. 2010), an image popularity dataset constructed using Flickr (Khosla, Das Sarma, and Hamid 2014), an anomaly detection dataset (Saleh, Farhadi, and Elgammal 2013), and a dataset designed for attribute-based object description (Farhadi et al. 2009). Khosla et al. chose these image sets because they have other metadata associated with them, like popularity on Flickr, or eye tracking data, or aesthetic ratings. All together, LaMem also has the benefit of being relatively naturalistic and diverse. The dataset was compiled with the intention of looking for correlations between memorability and these other data points. LaMem is a well suited dataset for this task because of its size and relative quality.

Our reimplementation of MemNet, as well as all of our models, were trained on a mixture of

LaMem and MemCat (Goetschalckx and Wagemans 2019). MemCat is a smaller, but higher quality dataset of images and memorability scores. It has 10,000 images, and its inter-human rank correlation is 0.78. This value generally increases with the amount of samples the dataset has per image, and it scales with the certainty of each value. Taken together, this implies that the memorability scores in MemCat are better estimations of human performance. MemCat was designed to help understand how memorability changes across and within categories of images.

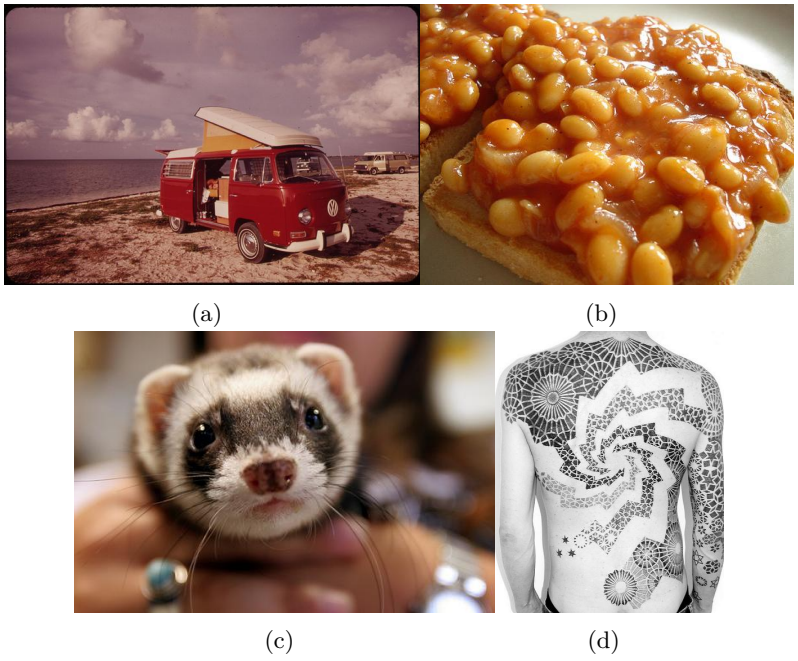


Figure 4: Some samples from LaMem.

These two datasets have some key features that can influence the behavior of our deep learning models. LaMem was compiled from a group of other datasets, which in turn were compiled to study secondary metadata of that image, like popularity on Flickr, aesthetic rating, etc. Considering the data sources for LaMem, we can notice that they are not a general sampling of naturalistic images. Flickr, for example, is an online community focused around sharing interesting images, thus, datasets from Flickr have a selection bias toward less naturalistic images. Among the other data sources, we have images that are designed to trigger eye movement, images that are designed to be anomalous, images that are designed to be affective, and so on. For example, many of the images from Flickr are intended as artistic renderings, and not necessarily

true reflections of reality. On the other hand, LaMem is very generalizable across image types. LaMem contains images of scenes, objects, people, and even a few images of text and iconography (fig. 4). The vast majority of LaMem's images are scenes, but the fact that it contains images of other types will improve the generalizability of the model across those types.

MemCat, on the other hand, was designed to describe several categories of object-focused images. Each image fits neatly into a category, like "cat" or "snowscape" and as such, every image is an object image, where the objects are largely unoccluded, and generally presented in context, and serve as the focus of each image (fig. 5). This contrasts with LaMem whose images are in a less natural context, and less focused on presenting a single object per image. However, what MemCat sacrifices in type-generalizability, it earns back in within-object generalizability. There are roughly a hundred images of deciduous forests and a hundred more of coniferous forests.

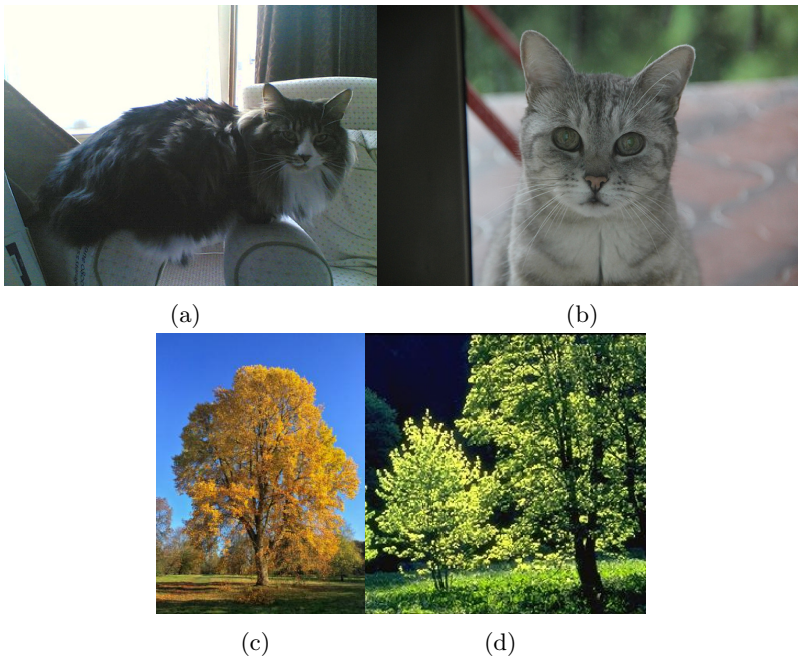


Figure 5: Some samples from MemCat, drawn from the categories "animal/cat" and "landscape/forest(broadleaf)"

Now, when you mix these two datasets together, then you end up with a larger dataset, with a mixture of objects, scenes, and miscellaneous images. The trends and features found in the larger, but lower certainty dataset get reinforced by the smaller, more precise dataset, and ideally, you

end up with a model that is powerful and generalizable on multiple axes.

Model Analysis

Model	MSE Loss ↓	Spearman Rank Correlation ↑	Approx. Train Time
MemNet	0.012	0.55	2500s
ResMem	0.008	0.66	8900s
ResMemRetrain	0.008	0.67	24000s
M3M	0.009	0.68	24600s

Table 1: A summary of results across models. Training time is for 20 epochs on an NVIDIA 1080TI running CUDA natively on Arch Linux. MSE loss is the average square distance between an estimation and the ground truth. Spearman rank correlation is a measure of how well preserved the order of scores is. The MemNet shown in this table is the model created with pyTorch. Train time is generally correlated with inference time, but ResMemRetrain is an exception, it takes the same amount of time to estimate scores as ResMem.

Overall our new models outperform MemNet, while still maintaining reasonable train times (Table 1). Adding a Residual Neural Network step to the model improves predictions by a considerable margin. Since ResMem employs both conceptual and perceptual features in its estimation process instead of just perceptual features, this implies that both types of features contribute to memorability. While the improvement on the general performance metrics shows that our multi-feature models outperform MemNet, it can also be useful to examine the other statistical properties of the model's estimations.

MemNet

Testing the publicly available version of MemNet yields a Spearman rank correlation of about 0.565, which is within expected limits of the 0.57 reported in the paper (Khosla, Raju, et al. 2015). Beyond just the rank correlation we can examine the other statistical properties of the predictions.

Specifically, we can plot the distribution of ground truth memorability scores and compare them to the distribution of estimations (fig. 6). An optimal model would produce the same distribution as the ground truths, and these visualizations can give us a deeper understanding of the model's statistical behavior. Note that passing this test does not show that the model is correct, since the distributions could be could be the same and have mismatched scores, giving a high loss and a low

rank correlation. Failing the test, however, is sufficient to show that the model under-performs. A common issue with complex estimation models is regression towards the mean. This is when a model optimizes loss by predicting the mean value in the input data with some small variation. If that small variation is correlated with the ground truth, then the model will achieve a good Spearman rank correlation, but a mediocre loss.

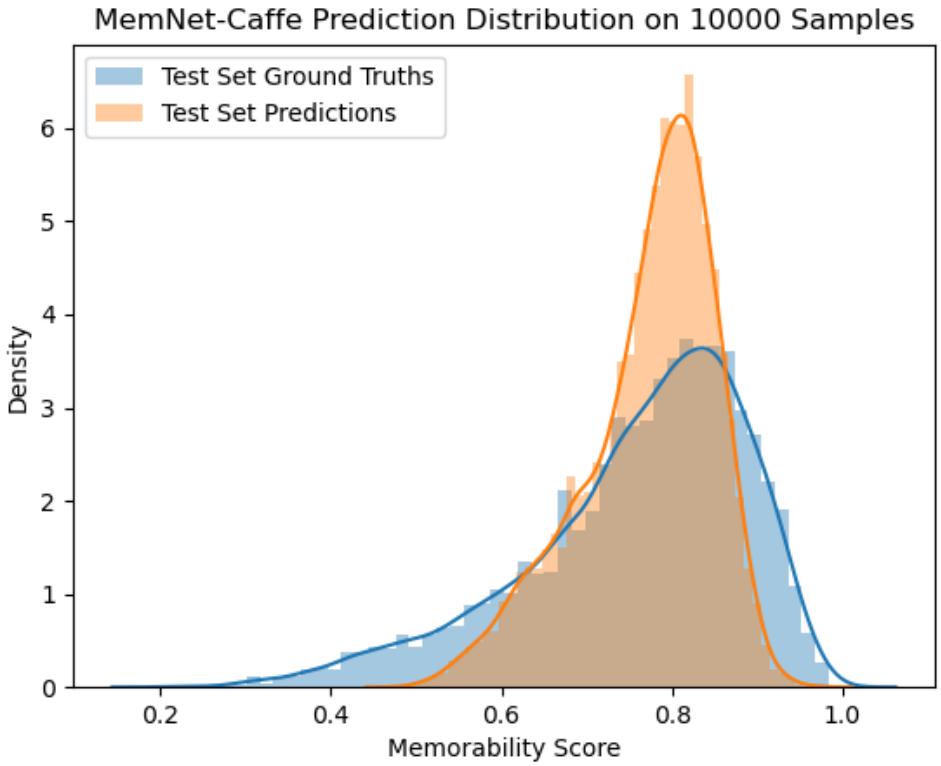


Figure 6: The Distribution of scores generated by running MemNet in Caffe, as specified by Khosla, Raju, et al. 2015. Predictions are shown in orange, and ground truths are shown in blue. These plots are created by estimating the Probability Density Function of the memorability scores, which range from 0 to 1. These plots are drawn such that the area under the curve between two x-values is the probability of randomly drawing an x-value in that range.

Generating these scores from the Caffe model available online faces some challenges. In addition to issues with getting Caffe running, it was difficult to reverse engineer the pre-processing steps from Khosla, Raju, et al. 2015 alone. By referencing backend code from the MemNet web demonstration, we were able to reconstruct MemNet's original preprocessing steps. First, an

input image is offset by a preset image, which is purported to be an image mean. Second, the image is subjected to a Ten Crop Transformation. An image is cropped five times to a certain size, once with the crop aligned to each corner, and once aligned to the center of the image. The image is then mirrored, and the process is repeated to yield ten crops in total. These ten crops have their memorabilities estimated by the neural network, and their estimations are averaged to get one score. These two transformations are common, and were individually considered good practices in 2015, but are two separate approaches to the same problem. After this, the score is transformed using a standard scaling method. According to the server code, the parameters for this scaling were hard-coded, and were not part of the training process. This hard-coded scaling is not considered best practice, and for this reason, we also constructed a model using a modern framework, employing best practices, while using an identical architecture to MemNet.

Our MemNet Reimplementation

Since we are using a mixed dataset to train our models, it would be difficult to compare performance against the Caffe version of MemNet. We have no way of knowing what train-test split that model was trained on, so we cannot test our models on the same data. We need to reverse-engineer MemNet from what is available online, and train it on the same dataset as our new models to control these variables. Since Caffe is largely deprecated, we will do this in PyTorch (P. Team n.d.).

We need to reconstruct this from what can be gleaned from the `.caffemodel` that is provided on the MemNet website. We will also need to conduct our own hyperparameter tuning using Weights and Biases (Biases n.d.) (fig. 7). Each of these lines represents a separate training session of our reconstructed MemNet, recording the Spearman rank correlation of the model's predictions on a validation dataset after each epoch. Each attempt has different hyperparameters, or the training/testing environment was changed in some minor way.

The reverse engineered MemNet has a pretty good distribution of predictions, better than the Caffe model. However, it still holds too much probability mass around the mean, and has a hard cutoff on the high end (fig. 8). When trained on the combination dataset, with early stopping determined by rank correlation on a validation set, this remade MemNet has a rank correlation

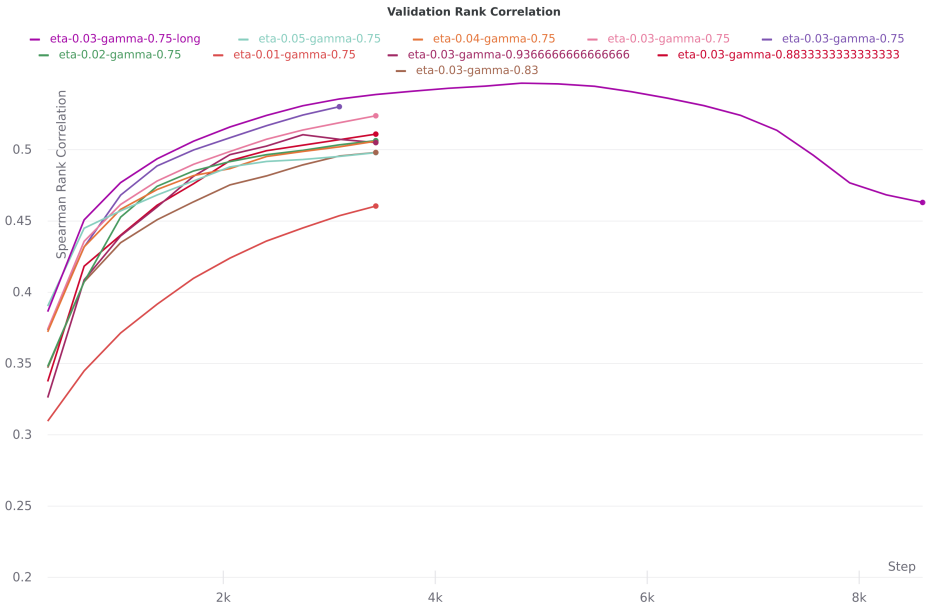


Figure 7: The MemNet tuning process. In this figure, hyperparameters were generated by hand to explore the behavior of the model in training. One run is longer than the others to examine how the model behaves in the overfitting regime. Eta(η) represents the learning rate, and gamma(γ) represents the momentum of the optimization method. It is clear that MemNet enters the overfitting regime fairly early, around training step 6000.

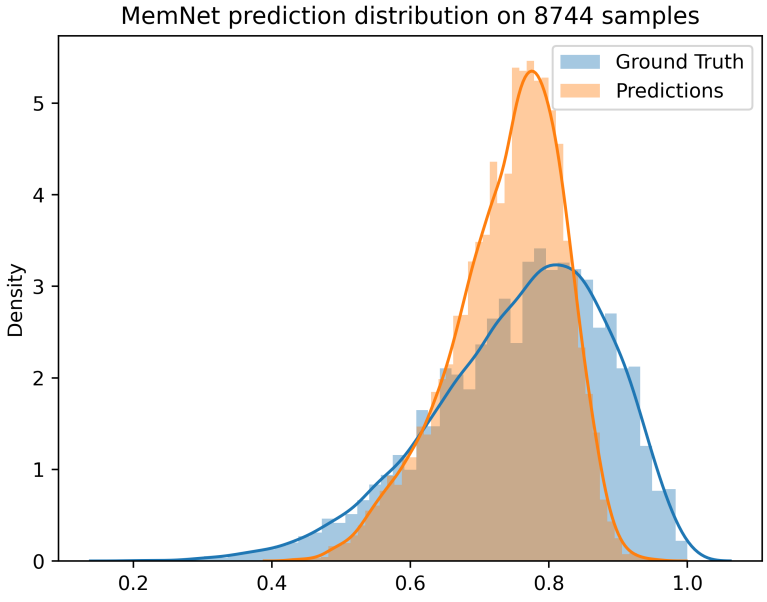


Figure 8: MemNet Distribution

0.55 on a held out test set, and an MSE Loss of 0.012. This implies that, on average, the memorability scores are accurate to 0.11.

ResMem

Using the ResMem architecture, we built two models using different training methods. The first model, called ResMem, does not change any of the ResNet-150 parameters during the training process. The second model, named ResMemRetrain, did retrain ResNet-150 parameters for the memorability task. By allowing ResNet-150 to retrain, the model gains specificity at the expense of a much longer training time. By examining the hyperparameter tuning process (fig. 9), we can see that this larger architecture enters the overfitting regime much later.

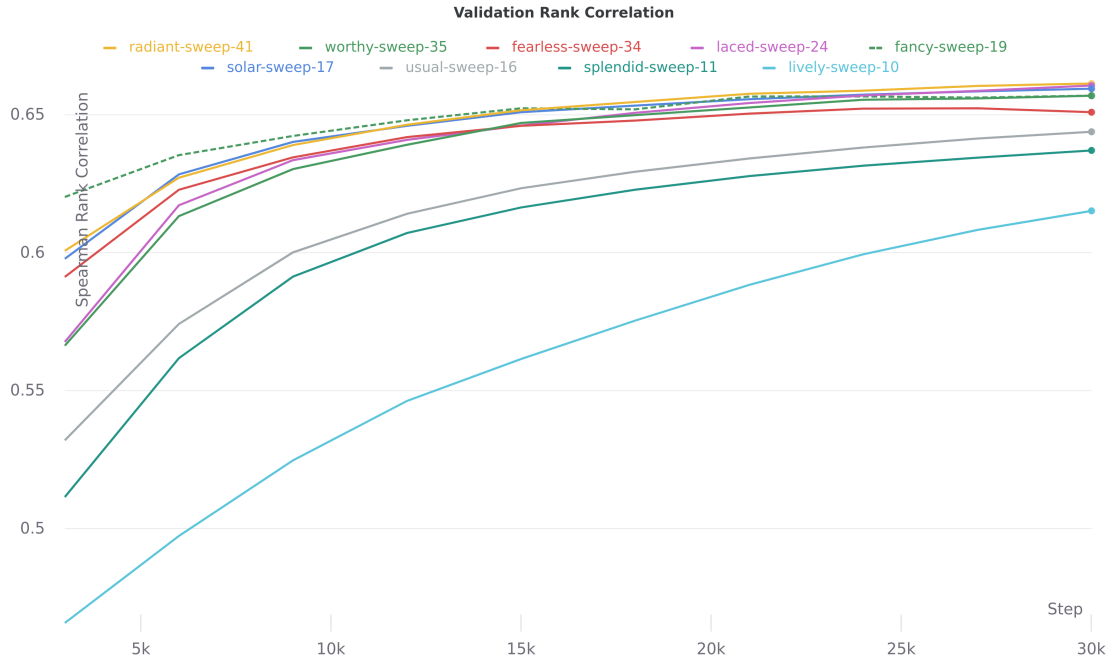


Figure 9: Validation performance from ResMem's hyperparameter tuning phase. Each line represents a model with slightly different hyperparameters (training rates, momenta). As the training phase goes on, for the most part, performance improves on the validation set.

By looking at the prediction distribution for ResMem (fig. 10), we see that the shape is roughly correct for most of the range, but there are sharp cutoffs at the high and low end. This model does, however, have a rank correlation of 0.68, and a loss of 0.0091 on the test set. This corresponds to being accurate within 0.095 on average. Thus, we can infer that the model is highly accurate

within the 0.6-0.9 range, although it struggles outside this region.

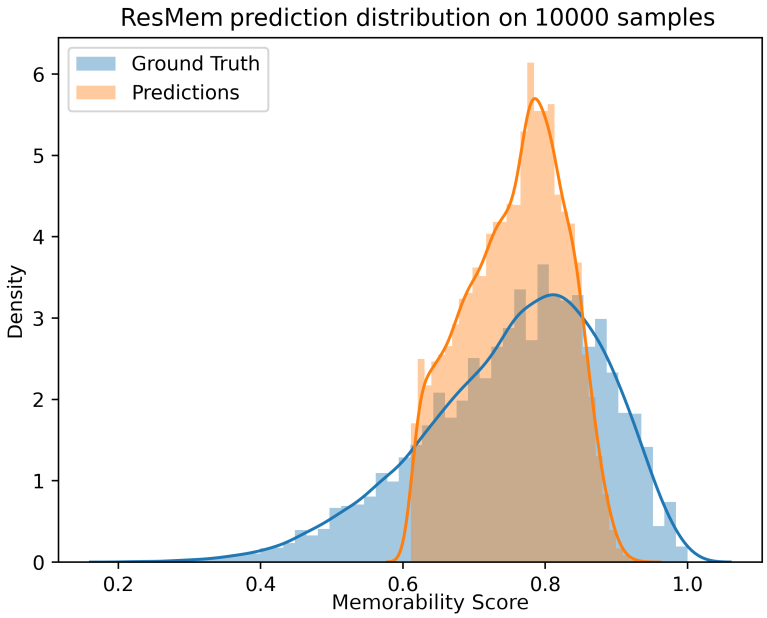


Figure 10: ResMem Distribution

Now, the version of ResMem with a retrained ResNet feature (ResMemRetrain) is interesting. Because we now include the model parameters in the training process, the intermediate features from ResNet will be re-optimized to estimate memorability, rather than to classify images. While we can no longer interpret the activation of those classification neurons as corresponding with ImageNet categories, we do gain significant predictive power. This is the version of ResMem that has been published on the Brain Bridge Lab website and on PyPI as `resmem`.

Compared to the others, the performance for ResMemRetrain (fig. 11) is astounding. We do see some cutting off at the low end at 0.411, but overall this is much closer to the ground truth distribution. Furthermore, this model achieves a rank correlation of 0.68, similar to its counterpart, but achieves tests losses as low as 0.0047. It is, on average, accurate to within 0.05 for all images. The distribution plot shows that some images are completely cut off of the prediction set because on the low only a few images have that low of a memorability score. The lowest score in the dataset is 0.2, and the lowest score that ResMemRetrain will assign is 0.411. We anticipate this

is not an important issue since only 0.6% of images have a memorability score below 0.411.

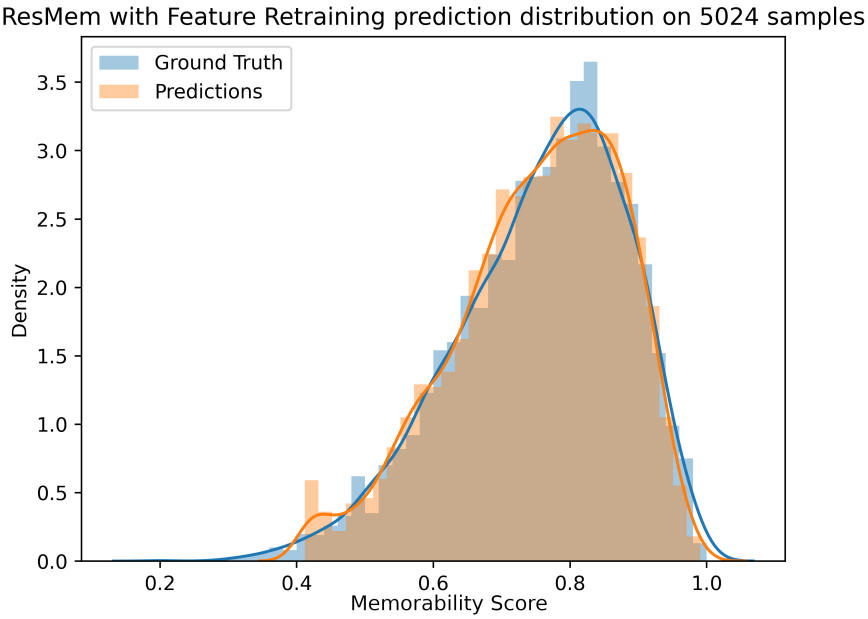


Figure 11: ResMemRetrain's predictions compared to ground truths.

M3M

Testing with M3M has yielded mixed results. While adding the semantic segmentation features to the model does increase rank correlations to around 0.68, this increase is very small compared to the heavy increase in complexity. This tells us that the semantic segmentation information does not contribute to memorability any more than the categorization data from ResMem does. While this is a null result, it is a null result that tells us quite a bit about the state of memorability: the exact location of semantic objects in the image does not matter as much as the simple existence of the semantic objects.

The distribution plot for M3M displays similar behavior to ResMem (fig. 12). This also supports our earlier discussion that M3M's performance increases in terms of rank correlation and loss are not worth the added costs. One training run of M3M took 30 hours to train on an Nvidia 1080TI. For comparison, ResMemRetrain takes 15 hours to run through the same amount of training steps, and ResMem without retraining and MemNet take 80 minutes.

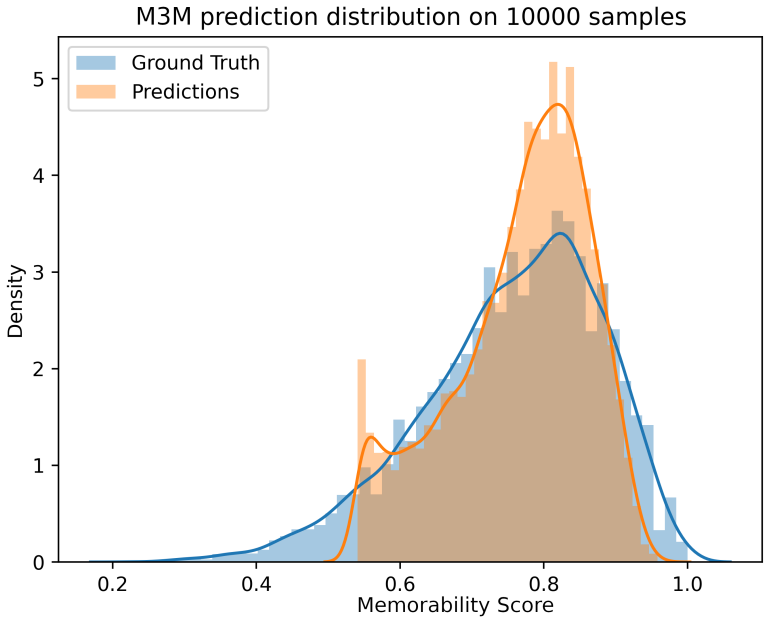


Figure 12: M3M's predictions compared to ground truths

Network Feature Analysis

Each layer of a convolutional neural network (CNN) or residual neural network (ResNet) does feature extraction. They extract features in the output of the previous layer, and the first layer extracts features from the input image. The "features" are patterns that appear in the data, and when optimized, the model picks out the patterns that are most useful in solving the problem. Because these features resist precise definition, it is difficult to tell what the model is latching on to. They can only be analyzed *post facto*. However, the consensus is that early layers are more perceptual and the later layers are more conceptual (Jozwik et al. 2018).

We can visualize the important features in the model by optimizing the inputs to maximally trigger a certain feature in the model. By running this process on every layer and feature, we can get a sense of what neural network features our model is hanging on to. When we apply a qualitative label to each feature, we can also see what images trigger which features, and get a better idea of how those features contribute to memorability in combination.

After running the activation optimization method on ResMem (fig. 13), we see more basic features

in the earlier layers, like edges and orientations (a-e). As the layers get deeper we see swoops and lines, branching patterns, and embossments, as opposed to the earlier layers that just show shapes arranged in lines (f-j). In the middle-to-late layers we see more obvious objects, like faces, eyes, and architectural features. Finally, the later layers should start to filter for more complicated and harder to interpret higher level conceptual features, and we begin to see "part" filters (Olah, Mordvintsev, and Schubert 2017). We can see what appear to be facial features in image (p), geometric structures in image (q), archways in image (r), what appears to be architectural glass ceilings in image (s), and eye-like patterns in image (t). As we get deeper we start to get into the conceptual space, and this method produces images that are harder to interpret. The repeated convolutions can make the features harder to optimize for, so by the time we get to the 139th layer, these images become more difficult to produce. In addition, the high level conceptual features are more complex, and thus are more difficult to represent in a single image. We see this reflected in images (u-y).

These feature visualizations help us understand what shapes that a filter is selecting for, but it doesn't tell us anything about what those shapes are. Now, instead of optimizing filter activation over the entire space of images, we can optimize filter activation over just the images in our dataset. Then, if we compare our feature visualizations with the maximally activating images (fig. 14), then we can more easily identify these features. We can see that the earlier layers (a-b) are maximally activated by images that are dominated by lines and simple shapes, and the later layers (c-d) are maximally activated by more complex forms, like the lower half of human faces, and the branching arch-like patterns that appear in plants and architecture.

Now, we can run the feature visualization analysis on our AlexNet features (fig. 15). All of the images are taken from the fourth out of five convolutional layers. While, in general, this technique is more difficult to perform on convolutional networks without residual connections, the images are clear enough to see what is going on in each feature. We see a couple filters selecting for orientation and edge information, especially (a), (c), (f), and (h), and all of them show evidence of selecting for textural information.

Even with the limitations on examining AlexNet features, when comparing them to the ResNet features there is considerable evidence that AlexNet is filtering for low level perceptual features,

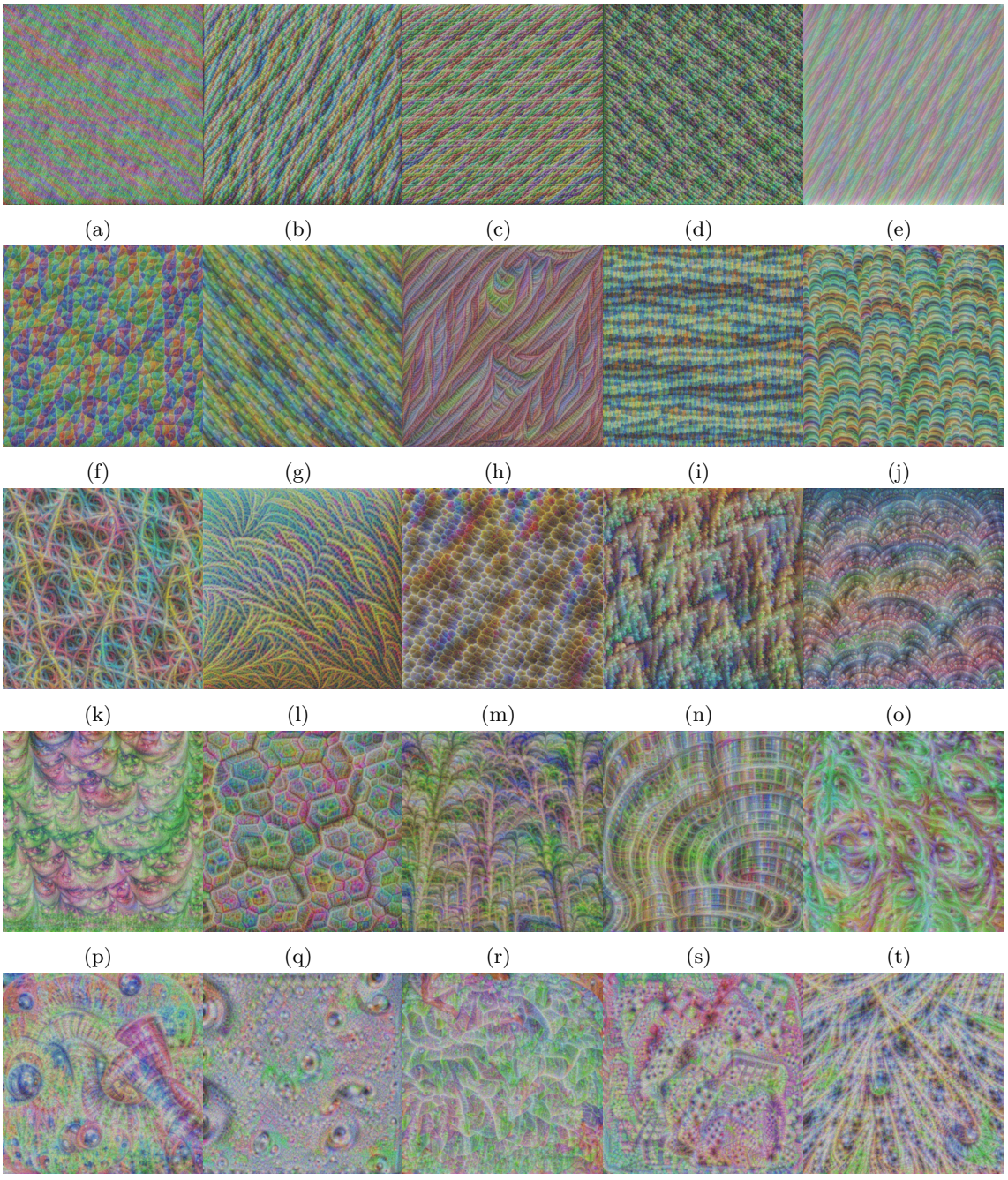


Figure 13: Feature visualizations from various layers of ResNet-150. The top row is from the 13th layer, the second is from the 35th layer, the third row is from the 81st layer. The fourth row is from a mixture of layers, (p) is from the 96th, (q-s) are from the 105th, and image (t) is from the 126th layer. The final row is from the 139th layer.

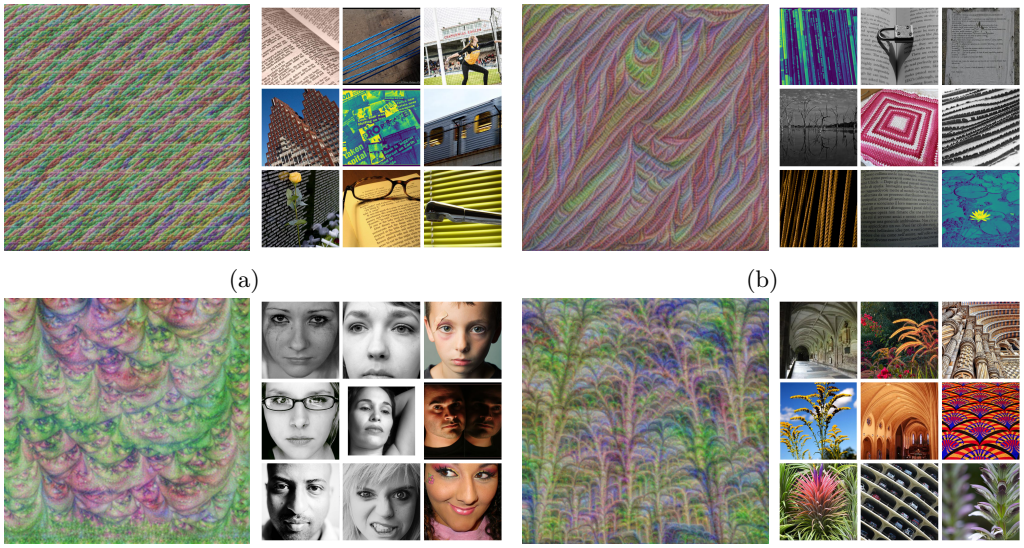
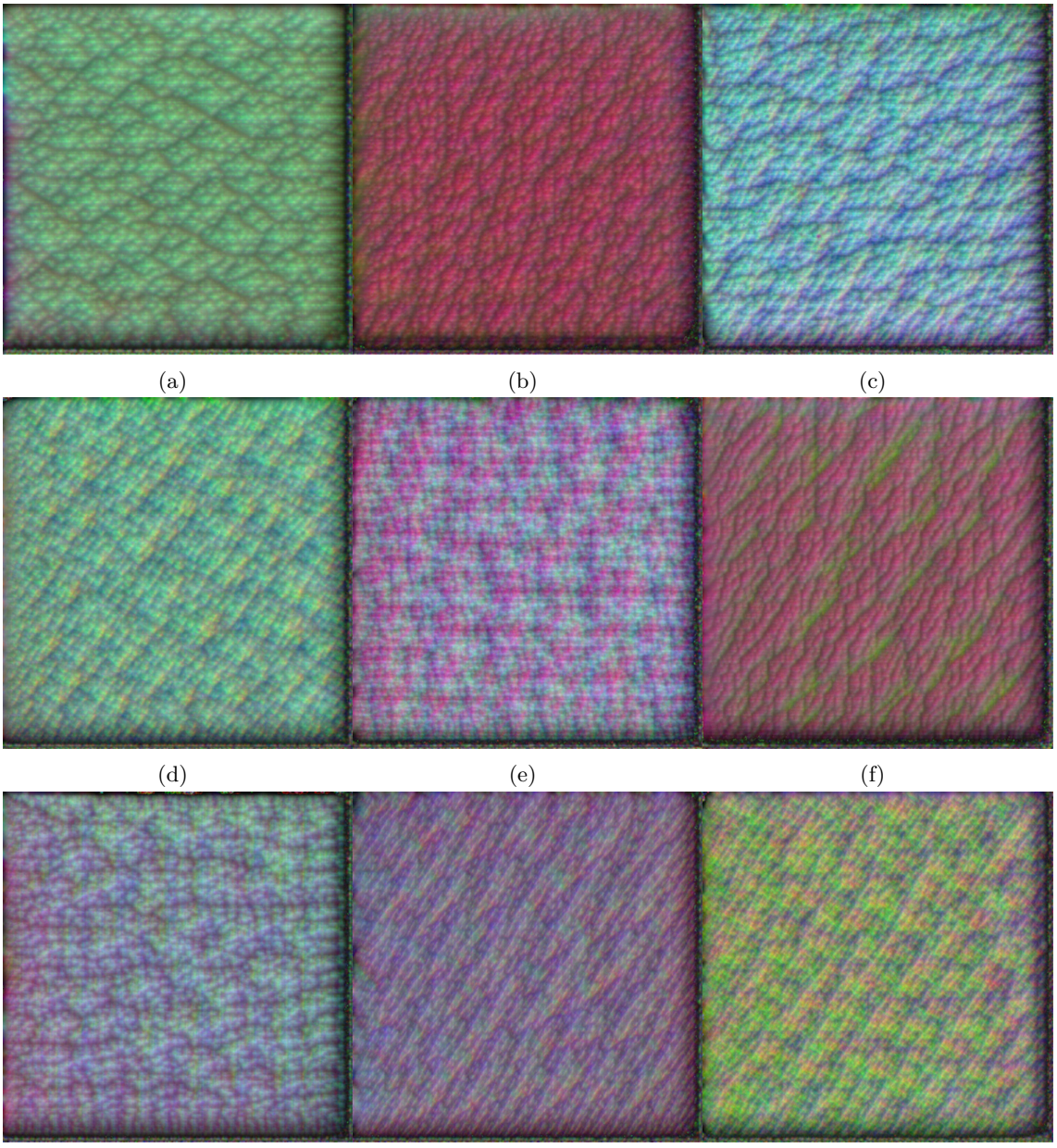


Figure 14: Feature Visualizations compared to the images that maximally activate the filter. This analysis helps us understand for what each filter selects. These examples show this analysis as done on both early layers (a-b), and late layers (c-d).

like textures and edges and in some cases simple patterns, while ResNet is able to filter for more conceptual objects.

Discussion

Using modern machine learning techniques can vastly improve our ability to estimate memorability, and it also opens up new avenues of analysis. ResMem has been made available to any scientist who wants to study what makes something more memorable. This will make it easier to control for memorability in other studies, and can make memorability-based analysis cheaper and easier to do; researchers can analyze data post-hoc to see if the intrinsic memorability of their images accounted for any of their effects. More broadly, this model can serve as a resource for the general public – allowing educators to identify memorable materials, helping the average person choose a memorable photo to post online, and aiding clinicians in creating memorable environments for those experiencing memory loss (Bainbridge, Berron, et al. 2019). To date, ResMem can be downloaded through the python package `resmem`, and a demonstration is available on the Brain Bridge Lab website (https://brainbridgelab.uchicago.edu/resmem).



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 15: AlexNet Features. Each of these features is taken from a layer 4 out of 5, and was generated by maximizing the activation of a filter with respect to the input image. You will notice that they are all very texture-driven, perceptual features. The most complex is in image (e) if we look closely. We can see a sort of flat hexagon structure forming. In general though, these are much less evocative than the ResNet features.

This research has shown that including high-level conceptual features in the memorability estimation process improves the accuracy of those estimations considerably. In comparison with MemNet which only decomposes images into perceptual features, ResMem achieves better predictive power in terms of generalizability, rank correlation, and estimation accuracy. This, when taken in concert with previous research, implies that memorability is related intimately to both the perceptual and conceptual features of an image (Kramer et al. 2021). Future work can examine the relationship between this model and the visual and memory systems in the brain, as prior work has done for visual areas (Cichy et al. 2016; Jaegle et al. 2019).

In addition to making memorability analysis simpler, models like ResMem can be leveraged to do analysis of high-level visual features, and how those features relate to memorability. As hardware technology continues to progress, more complicated models, such as graph-based neural networks, M3M without the deconvolution layers, and model-based algorithms can be employed to help understand memorability in those contexts. Carefully applying deep learning in the memorability field can help us better understand the phenomenon for years to come.

Acknowledgements

We would like to acknowledge Lore Goetschalckx for providing us with details on MemNet's implementation. We would also like to acknowledge Deepasri Prasad and Max Kramer for information sharing and general feedback.

References

Bainbridge, Wilma A. (2019). "Memorability: How What We See Influences What We Remember". en. In: p. 27.

Bainbridge, Wilma A., David Berron, et al. (Dec. 2019). "Memorability of Photographs in Subjective Cognitive Decline and Mild Cognitive Impairment: Implications for Cognitive Assessment". en. In: *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring* 11.1, pp. 610–618. ISSN: 2352-8729, 2352-8729. DOI: 10.1016/j.dadm.2019.07.005.

Bainbridge, Wilma A., Daniel D. Dilks, and Aude Oliva (Apr. 2017). "Memorability: A Stimulus-Driven Perceptual Neural Signature Distinctive from Memory". en. In: *NeuroImage* 149, pp. 141–152. ISSN: 10538119. DOI: 10.1016/j.neuroimage.2017.01.063.

Bainbridge, Wilma A. and Jesse Rissman (Dec. 2018). "Dissociating Neural Markers of Stimulus Memorability and Subjective Recognition during Episodic Retrieval". en. In: *Scientific Reports* 8.1, p. 8679. ISSN: 2045-2322. DOI: 10.1038/s41598-018-26467-5.

Basavaraju, Sathisha, Sibaji Gaj, and Arijit Sur (Feb. 2019). "Object Memorability Prediction Using Deep Learning: Location and Size Bias". en. In: *Journal of Visual Communication and Image Representation* 59, pp. 117–127. ISSN: 10473203. DOI: 10.1016/j.jvcir.2019.01.008.

Biases, Weights & (n.d.). *Wandb: A CLI and Library for Interacting with the Weights and Biases API.*

Chellapilla, Kumar, Sidd Puri, and Patrice Simard (n.d.). "High Performance Convolutional Neural Networks for Document Processing". en. In: (), p. 7.

Cichy, Radoslaw Martin et al. (Sept. 2016). "Comparison of Deep Neural Networks to Spatio-Temporal Cortical Dynamics of Human Visual Object Recognition Reveals Hierarchical Correspondence". en. In: *Scientific Reports* 6.1, p. 27755. ISSN: 2045-2322. DOI: 10.1038/srep27755.

Cireşan, Dan Claudiu et al. (Dec. 2010). "Deep, Big, Simple Neural Nets for Handwritten Digit Recognition". en. In: *Neural Computation* 22.12, pp. 3207–3220. ISSN: 0899-7667, 1530-888X. DOI: 10.1162/NECO_a_00052.

Clark, Alex (n.d.). *Pillow: Python Imaging Library (Fork).*

Deng, Jia et al. (2009). "Imagenet: A Large-Scale Hierarchical Image Database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, pp. 248–255.

Droettboom Michael, John D. Hunter (n.d.). *Matplotlib: Python Plotting Package*.

Fajtl, Jiri et al. (June 2018). "AMNet: Memorability Estimation with Attention". en. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, pp. 6363–6372. ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00666.

Farhadi, Ali et al. (June 2009). "Describing Objects by Their Attributes". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Miami, FL: IEEE, pp. 1778–1785. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206772.

Fukushima, Kunihiko (Apr. 1980). "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". en. In: *Biological Cybernetics* 36.4, pp. 193–202. ISSN: 0340-1200, 1432-0770. DOI: 10.1007/BF00344251.

Goetschalckx, Lore and Johan Wagemans (Dec. 2019). "MemCat: A New Category-Based Image Set Quantified on Memorability". en. In: *PeerJ* 7, e8169. ISSN: 2167-8359. DOI: 10.7717/peerj.8169.

Harris, Charles R. et al. (Sept. 2020). "Array Programming with NumPy". In: *Nature* 585.7825, pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

He, Kaiming et al. (Dec. 2015). "Deep Residual Learning for Image Recognition". en. In: *arXiv:1512.03385 [cs]*. arXiv: 1512.03385 [cs].

Huiskes, Mark J. and Michael S. Lew (2008). "The MIR Flickr Retrieval Evaluation". In: *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*. MIR '08. New York, NY, USA: Association for Computing Machinery, pp. 39–43. ISBN: 978-1-60558-312-9. DOI: 10.1145/1460096.1460104.

Jaegle, Andrew et al. (Aug. 2019). "Population Response Magnitude Variation in Inferotemporal Cortex Predicts Image Memorability". en. In: *eLife* 8, e47596. ISSN: 2050-084X. DOI: 10.7554/eLife.47596.

Jozwik, Kamila Maria et al. (2018). "Deep Convolutional Neural Networks, Features, and Categories Perform Similarly at Explaining Primate High-Level Visual Representations". en. In: *2018 Conference on Cognitive Computational Neuroscience*. Philadelphia, Pennsylvania, USA: Cognitive Computational Neuroscience. DOI: 10.32470/CCN.2018.1232-0.

Judd, Tilke et al. (2009). "Learning to Predict Where Humans Look". In: *IEEE International Conference on Computer Vision (ICCV)*.

Khosla, Aditya, Wilma A. Bainbridge, et al. (Dec. 2013). “Modifying the Memorability of Face Photographs”. en. In: *2013 IEEE International Conference on Computer Vision*. Sydney, Australia: IEEE, pp. 3200–3207. ISBN: 978-1-4799-2840-8. DOI: 10.1109/ICCV.2013.397.

Khosla, Aditya, Atish Das Sarma, and Raffay Hamid (2014). “What Makes an Image Popular?” en. In: *Proceedings of the 23rd International Conference on World Wide Web - WWW '14*. Seoul, Korea: ACM Press, pp. 867–876. ISBN: 978-1-4503-2744-2. DOI: 10.1145/2566486.2567996.

Khosla, Aditya, Akhil S. Raju, et al. (Dec. 2015). “Understanding and Predicting Image Memorability at a Large Scale”. en. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, pp. 2390–2398. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.275.

Kramer, Max et al. (2021). “Characterizing Memorability in Representational Space: Analyzing Relative Contributions of Perceptual and Conceptual Information”. In: *Vision Sciences Society*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). “ImageNet Classification with Deep Convolutional Neural Networks”. en. In: *Communications of the ACM* 60.6, pp. 84–90. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/3065386.

Machajdik, Jana and Allan Hanbury (2010). “Affective Image Classification Using Features Inspired by Psychology and Art Theory”. In: *Proceedings of the 18th ACM International Conference on Multimedia*. MM '10. New York, NY, USA: Association for Computing Machinery, pp. 83–92. ISBN: 978-1-60558-933-6. DOI: 10.1145/1873951.1873965.

Murray, N., L. Marchesotti, and F. Perronnin (June 2012). “AVA: A Large-Scale Database for Aesthetic Visual Analysis”. en. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Providence, RI: IEEE, pp. 2408–2415. ISBN: 978-1-4673-1228-8 978-1-4673-1226-4 978-1-4673-1227-1. DOI: 10.1109/CVPR.2012.6247954.

Olah, Chris, Alexander Mordvintsev, and Ludwig Schubert (Nov. 2017). “Feature Visualization”. In: *Distill* 2.11, 10.23915/distill.00007. ISSN: 2476-0757. DOI: 10.23915/distill.00007.

Ramanathan, Subramanian et al. (2010). “An Eye Fixation Database for Saliency Detection in Images”. en. In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Vol. 6314. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 30–43. ISBN: 978-3-642-15560-4 978-3-642-15561-1. DOI: 10.1007/978-3-642-15561-1_3.

Saleh, Babak, Ali Farhadi, and Ahmed Elgammal (June 2013). “Object-Centric Anomaly Detection by Attribute-Based Reasoning”. en. In: *2013 IEEE Conference on Computer Vision and*

Pattern Recognition. Portland, OR, USA: IEEE, pp. 787–794. ISBN: 978-0-7695-4989-7. DOI: 10.1109/CVPR.2013.107.

Scipy (n.d.). *Scipy: SciPy: Scientific Library for Python*.

Squalli-Houssaini, Hammad et al. (Apr. 2018). "Deep Learning for Predicting Image Memorability". en. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, AB: IEEE, pp. 2371–2375. ISBN: 978-1-5386-4658-8. DOI: 10.1109/ICASSP.2018.8462292.

Team, PyTorch (n.d.). *Torch: Tensors and Dynamic Neural Networks in Python with Strong GPU Acceleration*.

Team, PyTorch Core (n.d.). *Torchvision: Image and Video Datasets and Models for Torch Deep Learning*.

Tqdm (n.d.). *Tqdm: Fast, Extensible Progress Meter*.

Van Rossum, Guido and Fred L. Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace. ISBN: 1-4414-1269-7.

Waskom, Michael (n.d.). *Seaborn: Seaborn: Statistical Data Visualization*.

Xiao, Jianxiong et al. (June 2010). "SUN Database: Large-Scale Scene Recognition from Abbey to Zoo". en. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, pp. 3485–3492. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5539970.

Xie, Weizhen et al. (Sept. 2020). "Memorability of Words in Arbitrary Verbal Associations Modulates Memory Retrieval in the Anterior Temporal Lobe". en. In: *Nature Human Behaviour* 4.9, pp. 937–948. ISSN: 2397-3374. DOI: 10.1038/s41562-020-0901-2.