Learn visual components in logos for different industries through deep learning

Mads Emil Marker Jungersen
Aarhus Universitet
Seoul National University
Department of Computer Science
madsemil.mj@gmail.com

Abstract

This study examines the possibility to infer visual components dominating the different logos in different industries. More specifically the concept of transfer learning is used to build a classifying network, where a pre-trained ResNet50 [6] network is fine-tuned to the task of industry recognition based on the Logo2K+ dataset [21]. To infer the visual clues in each industry different methods of Activation Maximization (AM) techniques are implemented, to generate images that maximize the output neuron in each industry. The fine-tuned ResNet50 achieves a test accuracy of 57.91%*, and this relatively low accuracy compared to other image classification tasks entails, that the images generated by the different AM techniques are very difficult to interpret making it hard to infer the visual components dominating each industry. The code for the implementations done in this research is made publically available at https://github.com/madsemilmj/*

1. Introduction

The logo of a company is usually the first contact the consumer has with the company and therefore makes an important impression shaping the consumer's expectations. Understanding this impression is important for managers/companies when marketing their brand, as well as for new start-ups competing for the consumer's attention. A recent study [10] discovered that *".. descriptive logos (those that include visual design elements that communicate the type of product) more favorably affect consumers' brand perceptions than nondescriptive ones."* and hence designing a logo that successfully depicts the industry or service sold in the company is essential.

Designing a logo for a new brand usually is an extensive and challenging process, and by bridging the gap between design and deep learning, I hope to use a large dataset to understand the secrets of visual design. I propose a method deriving a relationship between logos and their industry automatically. The goal is to determine if the company's industry can be learned based on the visual aspects of their logo, as well as learn what these visual aspects are in each industry. This research can aid the design process by revealing underlying information, and visual components dominating different industries.

1.1. Related work

Logo classification aims to classify the company name of an input logo and can be viewed as a special case of image recognition from a computer vision perspective. The task of logo classification has been widely studied due to its various applications, such as copyright infringement detection, product recommendation, and contextual advertising. Traditionally logo classification has been relying on hand-crafted features [16], but different deep learning methods have been used and shown superior performance with end-to-end pipeline automation [1, 4, 21]. Very much related to the logo classification task is the problem of logo detection where also localization of the logo in the input image is addressed. Also in this area, deep-learning methods have shown promising results [2, 8]. Because of the large logo-related research there, exist quite a few popular public available logo-datasets such as FlickrLogos-32 [17] and LOGO-Net [7]. In 2019 [21] introduced a new logo-dataset called Logo-2K+ with $167,140$ logos from more than $2,000$ companies. They introduced a Discriminative Region Navigation and Augmentation Network (DRNA-Net) receiving state-of-the-art accuracies on both their new dataset, as well as on existing popular logo datasets. This dataset also contains information about the industry of the company, and hence this is the dataset used in this research.

To accommodate the lengthy and tedious task of designing a new logo, different types of Conditional Generative Adversarial Networks (GAN) [5] have been proposed to synthesis plausible logos [11, 19]. These models use auxiliary labels (based on clustering) as well as conditioning on colors to synthesize logos and hence are not built to

generate logos conditioned on the industry. This is however an interesting area for future research and will be slightly touched later in this paper.

The contribution of this paper is to examine the possibility to infer visual components dominating the logos in different industries. To solve this task, I used the concept of transfer learning and developed a CNN-based model for industry classification based on company logos. ResNet [6] pre-trained on ImageNet [18] is adapted for the task of industry recognition. I benchmark our accuracy against the results obtained by [21], however since they classified the company and not the industry, these results are not comparable one-to-one. To inspect and infer the visual components dominating in each sector, different Activation Maximization (AM) visualization techniques are implemented.

AM is a visualization tool developed to visualize and understand the neurons of deep CNN [3, 20, 22], by generating/finding an input that maximizes the activation of a neuron. Mathematically it can be expressed as the following optimization problem:

$$x^* = \arg\max_x \Big(a_i(x) - R_\theta(x) \Big) \tag{1}$$

where R_θ is parameterized regularization function that penalizes input in various ways to enforce image-like outputs, and $a_i(x)$ is the activation of neuron i given input x. The above optimization problem is solved using gradient ascent. A lot of research has been done to modify and improve the AM by introducing different priors (i.e. different $R_\theta(x)$) as well as trying different start-images for the optimization process [12, 13, 15]. In this paper, I will be focusing on the work by [12, 13].

2. Method

This section is split up into two parts focusing on the two main tasks of this research, namely the classification of the industry based on a company logo, and the AM exploring the visual components dominant in each industry.

2.1. Classification

I used the concept of transfer learning, fine-tuning a pre-trained ResNet50 model. The authors of ResNet tackled the known issue of the fact that deeper models can have higher training error than shallower models. They overcome this issue by using "residual" mappings from shallower layers, allowing them to take advantage of the deeper structure and additional nonlinearities. The ResNet model has become one of the fundamental CNN for image classification and is used as the baseline for many new image classification models. The ResNet50 model was pre-trained on the ImageNet dataset, and different fine-tuning strategies were tested i.e. training only the classifier or training a larger number of layers.

The ResNet model was fine-tuned on the Logo-2K+ [21] where the $167,140$ logos was split into $116,958$ training images and 50182 test images $(70/30)$ where the training images were further split into a validation and training set $(80/20)$. Simple data augmentation was conducted on the training samples such as random cropping (75% of the original image) and horizontal flipping. Also due to computational limitations, small downsampling of both training and test images was performed. The original size of the logos in Logo-2K+ is 256×256 which I resized to 224×224. The dataset contains logos from 10 different industries, and the specific industries, as well as the distribution of the training data in each industry, is displayed in 3.

2.2. Activation Maximization (AM)

Once the classifier is trained, different AM methods were implemented in order to explore the visual components dominating each industry. The first approach generates images by trying different priors (different regularizes $R_\theta(x)$). This paper only focuses on generating images that maximize the activation of the output neuron for each industry/label, which I define as $S_c(x)$ for industry c. In practice, I use a slightly different formulation of the before-mentioned optimization problem. Because I search for x^* by starting at some x_0 and taking gradient steps, I instead define the regularization via an operator $r_\theta(\cdot)$ that maps x to a slightly more regularized version of itself. The gradient ascent algorithm applies the following update:

$$x \leftarrow r_\theta \left(x + \alpha \frac{\partial S_c(x)}{\partial x} \right) \tag{2}$$

where α defines the step-size. More specifically the following priors was used:

- L2 decay: $r_\theta(x) = (1 - \theta_{\text{decay}})$
- Gaussian blur: $r_\theta(x) = \text{GaussianBlur}(x, \theta_\sigma)$
- Gradient norm clipping

The L2 decay introduces a hyper-parameter θ_{decay} and it tends to prevent a small number of extreme pixel values from dominating the output image, which does not occur frequently in natural images. The gaussian blur is used to penalize high-frequency information by convolving the image with a blur kernel with width θ_σ. This operation is expensive and hence I chose to only blur every θ_n optimization step, using the fact that blurring an image multiple times

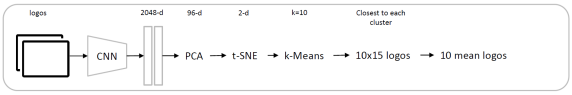

Figure 1. Illustration of the algorithm used to generate mean images for each class used as input to the AM technique. The trained ResNet50 is used to extract features for each image/logo. Then Principle Component Analysis (PCA) is used to reduce each feature vector x_i such that $x_i \in \mathbb{R}^{96}$. The t-SNE algorithm is used to reduce dimension to 2, followed by k-means clustering to generate $k = 10$ centroids. For each centroid, I calculate the mean image of the 15 closest images/logos in 2d space.

with a small width Gaussian kernel is equivalent to blurring once with a higher kernel. The gradient norm clipping is implemented to prevent the output image to contain non-zero pixel values everywhere, and hence trying to only maintain the main object, letting other regions be exactly zero if they are not needed. This introduces the hyper-parameter θ_{clip}.

The first approach of implementing the AM described above constructed images without regard for the multiple different facets of the output score, creating inappropriate mixes of colors, parts of objects, etc. The fact that the logo dataset has a high inter-class variability an algorithm that tries to account for this multi-modality within each class was implemented as the second approach. The algorithm closely follows the work of [13] and is summarized in 1. The mean images for each class computed by the algorithm illustrated in 1 are used as the start-images (x_0) for the AM technique described earlier. This should take into account the different modes within each class of the dataset.

The first two approaches of generating AM images described above share the same issue of having quite a few hyper-parameters that have to be tuned in order to generate the best possible images. Random-search is used, and while this is a computationally heavy procedure, it is also not a very scientifically satisfactory approach. To overcome this issue I follow the idea of [12] and introduce a Generative Adversarial Networks (GAN) [5] as the third approach. While [12] uses a Generative network that directly inverts the output of an Encoder network, being the same model as the CNN generating the neurons to be visualized (referring to their research for further implementation details), this paper uses a slightly different approach. The main idea is however the same and is illustrated in 2. Instead of optimizing the maximum activation of the output neuron in high dimensional space, the idea is to optimize in low-dimensional latent space. To do this I train an Auxiliary Classifier GAN (ACGAN) [14]. The ACGAN consists of a generator (G) and a discriminator (D), as traditionally GAN networks. However, in the ACGAN, every generated sample has a corresponding class label c in addition to the noise z. This class label helps the model to synthesize the image based on the label passed. The generator G uses both the class label and the noise z to generate the image. The architecture of the generator network (G) consists of a few fully-connected layers using batch normalization [9] and ReLu activation functions. The fully connected layers are followed by transposed convolutional layers to generate the output image, again using batch normalization and ReLu activation function. The class label is simply concatenated with the input noise z, by generating a one-hot-vector of length 10 (10 classes/industries). The discriminator network (D) consists of a few convolutional layers followed by a few fully-connected layers. Between each layer, there is a batch normalization and leaky ReLu activation. The output activation functions are sigmoid and softmax for source and industry classification respectively. The objective function for the ACGAN has two parts; the log-likelihood of correct source(fake/real) and log-likelihood of correct class/label:

$$L_S = \mathbb{E}[\log P(S = \text{real}|X_{\text{real}})] + \mathbb{E}[\log P(S = \text{fake}|X_{\text{fake}})] \tag{3}$$

$$L_C = \mathbb{E}[\log P(C = c|X_{\text{real}})] + \mathbb{E}[\log P(C = c|X_{\text{fake}})] \tag{4}$$

While the discriminator (D) tries to maximize $L_S + L_C$ the generator (G) tries to maximize $L_C - L_S$. Once trained (on the Logo-2k+) I discard the discriminator and use the generator (G) to generate images based on the noise z. Since I now optimize in low-dimensional latent space, the AM problem becomes:

$$z^* = \arg\max_z \Big(S_c(G(z, c)) - R_\theta(z) \Big). \tag{5}$$

The prior $R_\theta(\cdot)$ is now over the latent code z, and I used a simple clipping to ensure that the values of $z \in (0, 1)$, since this is the distribution of z used in the training of the generator.

Since I try to identify the visual components dominating the logos in the different industries through the use of AM, I am in lack of any quantitative evaluation metric, and hence the generated output images from the different AM approaches will only be evaluated qualitatively. Nevertheless, I will evaluate our classifying network ResNet50 on its achieved accuracy on the set aside test dataset.

3. Experiments and results

This section is split up into two parts, where the first part discusses the implementation and training of the classifier as well as a short description of the practical usage of the

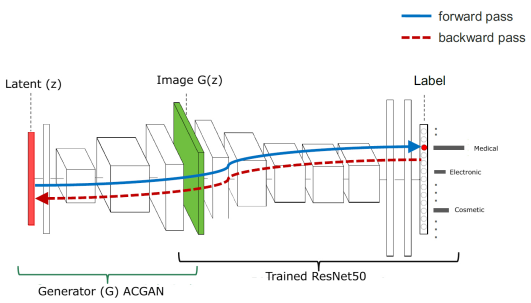

Figure 2. A trained generator (G) from the ACGAN model generates images from the noise-vector z. The generated image $G(z, c)$ (c for the given class/industry) is fed to the ResNet50 network. I use gradient ascent to maximize $S_c(G(z, c))$ which is the output score for class c from the ResNet50 model. (modified illustration from [12])

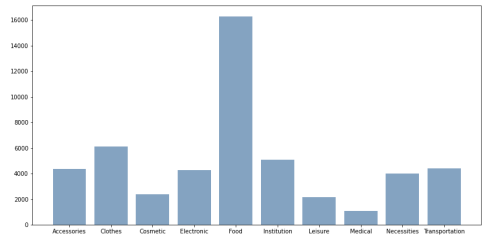

Figure 3. The distribution of the classes/industries in the Logo-2K+ dataset. From left to right the industries are as follows `Accessories`, `Clothes`, `Cosmetic`, `Electronic`, `Food`, `Institution`, `Leisure`, `Medical`, `Necessities`, `Transportation`

dataset Logo-2K+, while the second part elaborates on the implementation and setup of the Activation Maximization problem.

3.1. Implementing the classifier (ResNet50)

As mentioned earlier the experiments in this paper are conducted on the Logo-2K+ dataset, which contains $167,140$ logos from more than $2,000$ companies. The authors of the dataset implemented a model to classify the company name based on the logo, and small changes to the dataset were done in order to change the label from company to industry. The industry information was already a part of the dataset, and it contains logos from 10 different industries. The different industries and the distribution of the training logos in each industry can be found in 3.

The ResNet50 model is pre-trained on the ImageNet, and after running small introductory experiments, I seem to get better results by doing fine-tuning on all the weights, and hence this method is used for training the model. The model was trained with different optimizing algorithms, where again small introductory experiments were done to narrow down the hyper-parameter search. In all cases, a batch size of 20 seemed to be the optimal size. The following optimizing setups seemed to perform the best:

- SGD with learning rate of 0.01
- SGD with learning rate of 0.015
- SGD with learning rate of 0.02
- Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$ and with multi-step learning rate scheduler with initial learning rate of 0.015 reducing it by 10% after 6th and 10th epoch.

However, when performing the actual training, the vanilla SGD clearly outperformed the Adam optimizer. The result of the training can be seen in 4. The training loss and training accuracy for the SGD-optimizers does not seem to be converged, and they follow smooth descending and ascending curves respectively. The validation accuracy however contains a greater fluctuation and does not seem to improve much over the 18 epochs. I adopt the Pytorch framework to train and implement the network, using free GPU access through Google Colaboratory. The data is stored in Google Drive which generated a huge bottleneck in the I/O of the training data when training the network. To overcome this, the dataset was zip-compressed and imported into the main-memory at once, then unzipping the data when needed. This reduced the running time of one epoch by a factor of 3 going from approximately 3 hours for one epoch to approximately 1 hour pr. epoch. As seen in 4 the network trained with SGD and a learning rate $\alpha = 0.015$ has the highest validation accuracy (approx. 57%). This network archived an accuracy of 57.91% on the set-aside test data. This is slightly underperforming compared to the results obtained by [21], archiving test accuracy of 66.34% on their implementation of ResNet50, and test accuracy of $72,09\%$ on their own network implementation. These accuracies are obtained on the task of classifying the company name based on the logo, and hence these results are not one-to-one comparable.

3.2. Activation Maximization (AM) results

The 3 approaches of generating images with AM introduced in the method section, unfortunately, did not produce the expected or hoped results. In the first two approaches the random hyper-parameter search was conducted by pre-defining a set of possible values for each of the parameters (based on the setting that produced good results in [22]), and then trying different combinations of these. More concretely the following possible values was defined: $\theta_{\text{decay}} \in \{0.1, 0.05, 0.0005, 0.00001\}$, $\theta_n \in \{3, 4, 5, 6, 7\}$, $\theta_{\text{clip}} \in \{0.05, 0.1, 0.15\}$ and $\theta_\sigma = 1$. This generates 60 different combinations and running all 60 maximization

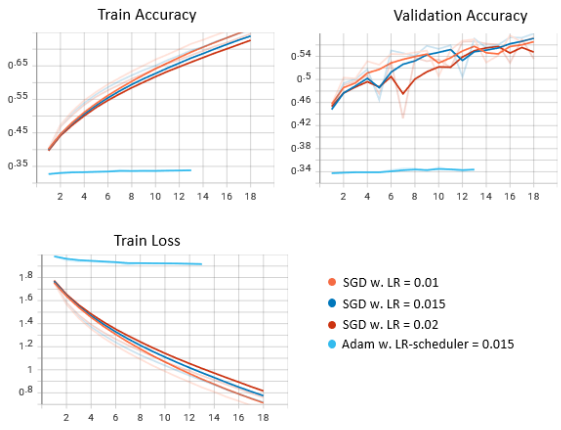

Figure 4. Training results of the 4 different approaches to optimize the ResNet50 network on Logo-2K+ dataset. It clearly shows that in this case, SGD outperforms the Adam optimizer with a learning rate schedule, and hence the training with that optimizer was stopped after epoch 13.

problems for one class takes approx. 1.5 hours using free GPU on Google Colaboratory. This procedure was done for both of the two first approaches (in the second approach a few mean images was chosen due to limited computation time). Some of the generated images for the first and the second approach is displayed in figure 5 and figure 6, 7 respectively. The second approach differs from the first approach by starting the maximization process from different mean images (generated using the algorithm displayed in 1) instead of random noise. Examples of these mean images are also shown in figure 6 and 7.

In the third approach, I first trained the ACGAN[1] network on the Logo2K+ training dataset. Due to a memory constraint on the GPU provided by Google Colaboratory, the training images needed to be resized into a smaller size of 112×112, where it was possible to train the network using a batch size of 4. The ACGAN network was trained for 5 epochs, where the time for running one epoch using GPU was approximately 2 hours. The result of the images generated by the generator (G) in each epoch in the trained network is illustrated as an animation displayed in figure 9. Following the method illustrated in 2 the generator network (G) is now used to generate output images solving the optimization problem described in equation 5. The problem was, like in the other approaches, solved with gradient ascent running for 150 iterations. Some fine-tuning on the learning rate was done, and also I made sure that each value of the latent code z was between $(0, 1)$ since that was the distribution used for training the generator (G). The output images from the AM

[1]Modified the github implementation of ACGAN Github

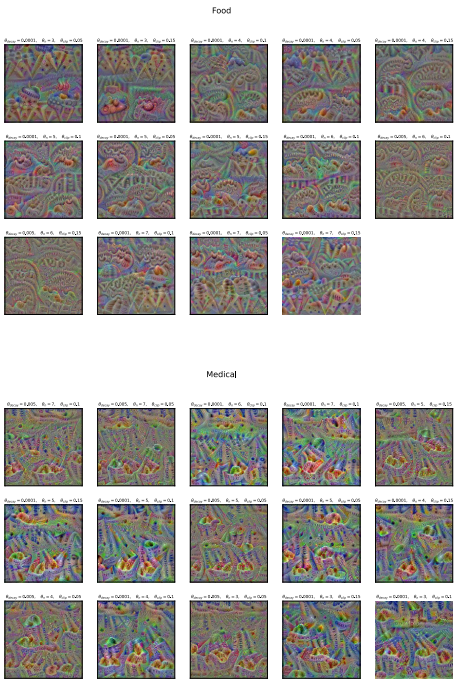

Figure 5. Images generated with the first approach of Activation Maximization (AM). Two industries (Food and Medical) are chosen at random for display. For each industry the "best" images of the total generated images are chosen, the parameter setting that generated the given image is written above each image. The image is in `svg` format so it is possible to zoom to get a closer look at each image. In all the above images $\theta_\sigma = 1$.

problem generated this way is illustrated in 8 and are quite different from the results for the two first approaches. The images generated in this approach are very similar to the actual output images from the generator network displayed in 9.

4. Discussion and conclusion

This paper tried to examine the visual components dominating the logos in different industries, and I hoped to learn the secrets of visual design by bridging the gap between design and deep learning by the use of a large dataset. The insight into the visual design clues was sought through the use of Activation Maximization for each of the industries/classes in the dataset. The output images generated by solving the different approaches of the AM problem shown in 5, 6, 7 and 8 are very hard to interpret. Comparing the results between the two different industries (Medical and Food) for the images generated by the first approach, does not provide any useful or distinctive differences regarding the industry. While the results of the first approach generate more dark/grayish images with some

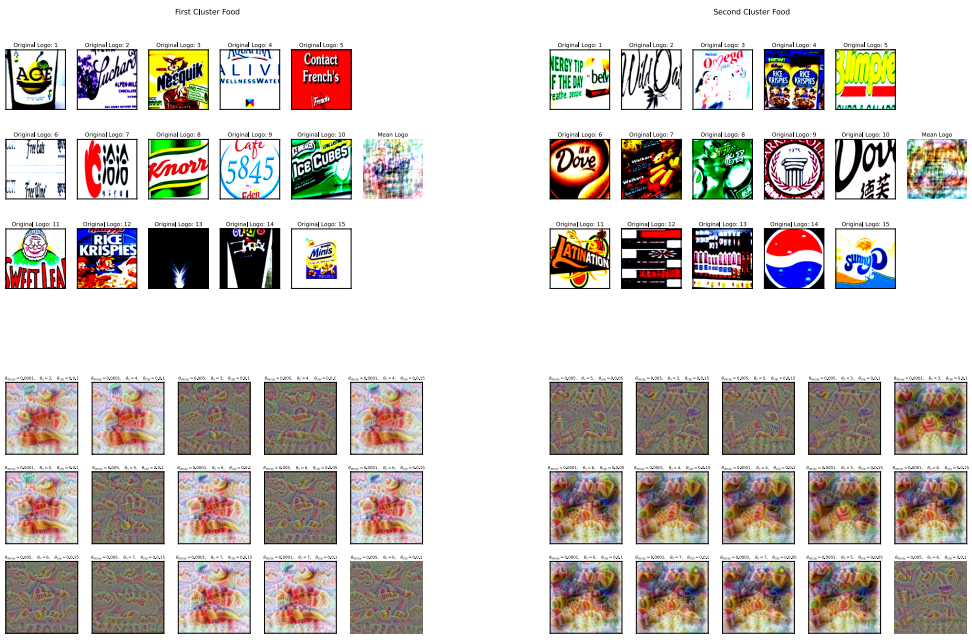

Figure 6. Displaying results from the second approach of Activation Maximization for industry `Food`. The first row shows 15 images and the corresponding mean image for two different clusters generated following algorithm displayed in figure 1. The second row shows 15 images generated when starting the gradient ascent optimization of AM from each of the two above mean images. The parameter setting that generated the given image is written above each image. In all the above images $\theta_\sigma = 1$. The image is in `svg` format so it is possible to zoom to get a closer look at each image.

unnatural colors dominating parts of the image, the two other approaches tend to have a lighter background, and these images might give clues into some dominating colors in the industry. For the second approach it can, for instance, be seen that the images generated for the medical industry tend to be dominated by a light-blue color while the images generated for the food industry tend to be dominated by darker and more red colors. Nonetheless, this difference might simply be due to the randomly chosen clusters and mean images generated hereof. The images generated by the third approach 8 are not really informative, but on the other hand, one might see small color differences between the industries. The image generated for the medical industry tends to have yellow, light-blue, and green colors, while for instance the image generated for the electronic industry has darker and more red colors.

The reason why I do not get more informative results from the output images for the different AM approaches should probably be found in the fact that our classifier network (ResNet50) achieves a relatively low accuracy on the test dataset (57.91%). The task of classifying an industry based on a logo is a difficult task, even for humans. The ResNet50 network is trained to find patterns in the data, and looking at the training loss and training accuracy it does seem to find patterns (in the training dataset) but the validation accuracy fluctuates during the epochs and does not seem to improve much. In 2017, [23] showed that modern deep neural networks can exactly memorize and interpolate training labels, even when the labels are completely randomized. This might as well be the case that is happening here. Looking deeper into the Logo2K+ dataset I, for instance, see that within the industry of Food I find both the brand "Aqua D'Or" and "Ben Jerry's" which indeed both belong to the food industry, but to what extent are these two brands comparable? Should these brands share the same visual components in their logo, and is there any pattern between the logos of these brands? This is just one of many cases, that occur within almost all of the 10 industries in the dataset, thus I am training our classifying network to detect patterns, that might not exist in the general sense. Another point related to the above is that logos across different industries share a high similarity in many cases, especially logos consisting of text. For instance, the brand Nicorette in the Medical industry and the brand Benelli in the transportation industry are quite alike 10 and unless you know the brand/company one would not really be able to tell the industry of these brands, purely based on their logos. Again this is just one out of many examples. Furthermore, when looking at the distribution of the images in each

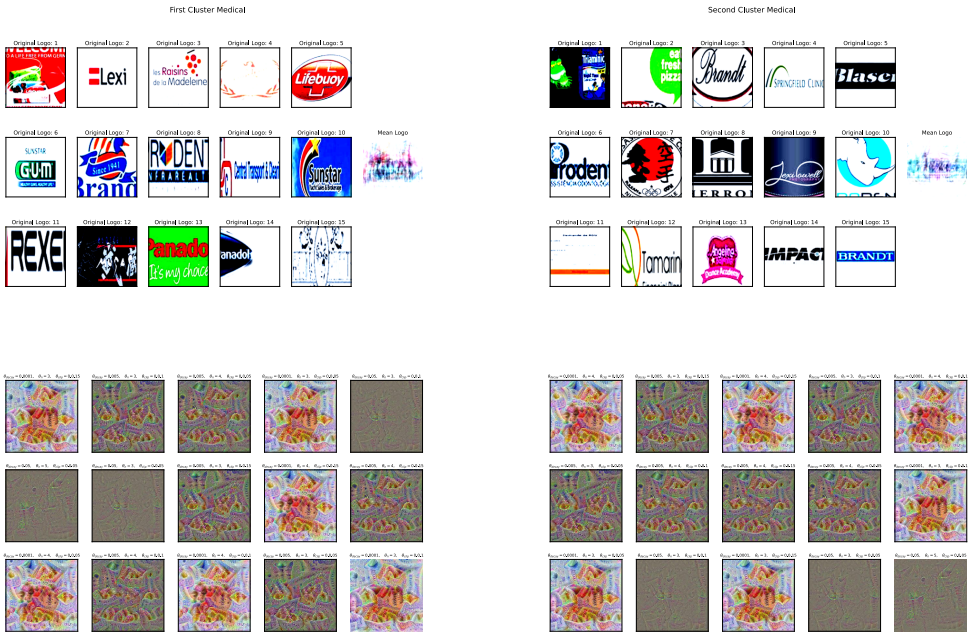

Figure 7. Displaying results from the second approach of Activation Maximization for industry `Medical`. The first row shows 15 images and the corresponding mean image for two different clusters generated following algorithm displayed in figure 1. The second row shows 15 images generated when starting the gradient ascent optimization of AM from each of the two above mean images. The parameter setting that generated the given image is written above each image. In all the above images $\theta_\sigma = 1$. The image is in `svg` format so it is possible to zoom to get a closer look at each image.

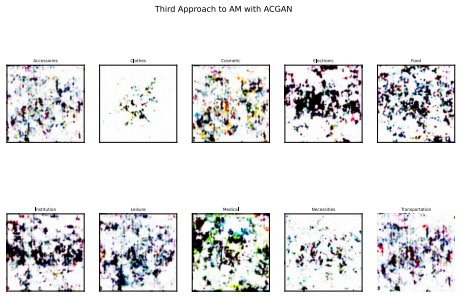

Figure 8. Images generated by the third approach of the AM problem. One image is generated for each class, running 150 iterations of gradient ascent. The image is in `svg` format so it is possible to zoom to get a closer look at each image.

class for the training dataset 3, there is a huge overweight of images/logos within the Food industry, and hence the classifier might tend to classify more logos as being in the Food industry just because of that overweight and not because of the visual clues. The confusion matrix shown in 11 provides clues that our model has a tendency to do just that.

Since all of our generated images from the different AM approaches highly depend on the classifying network, I simply can not expect to get really informative results due to the discussed matters above. On the last note it should also be mentioned that the images generated from the last approach to AM using the generator (G) of the trained ACGAN network, also suffered from the fact that the ACGAN network only was trained for 5 epochs, which definitely is not enough to generate good images. The lack of training of the ACGAN was simply due to computational limitations on the free GPU provided by Google Colaboratory.

4.1. Future research

As previously described the task of inferring visual components dominating logos in different industries, was in this paper sought to be solved by the use of different AM approaches. Since the goal was to learn the visual clues rather than actually generating new logos, this approach was chosen instead of the approach to generate new logos for each industry using Conditional Generative Adversarial Networks. On the other hand, the use of ACGAN was implemented in one of the AM approaches, and one might argue that the conditional Generative Adversarial Networks also captures the visual clues dominating in each industry when synthesiz-

Figure 9. Animated visualization of the images generated by the generator network (G) in each of the 5 epochs. Each column corresponds to one of the 10 industry classes, while each row represents different random noise vectors z generating the images. (Open in Adobe Reader to see the animation.)

Figure 10. The logos of two different brands belonging to two completely different industries.

ing new images. In future research it would be interesting to pursue the goal of synthesizing logos using Conditional Generative Adversarial Networks, conditioned on the industry instead of auxiliary clustering or conditioning on color as in [11, 19]. Also to accommodate some of the issues or challenges described in the discussion, it would be interesting to generate a more fine-grained dataset, with more specific industry labels, and see if this research would improve, or use that data to synthesize plausible logos with Conditional Generative Adversarial Networks.

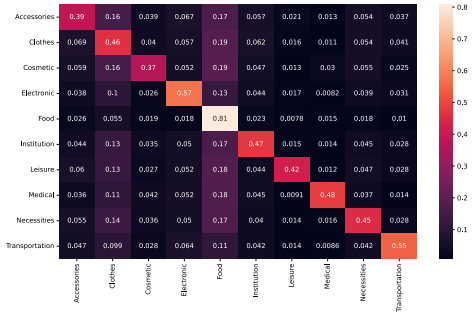

Figure 11. Confusion matrix of the ResNet50 network classifying the set aside Logo2K+ dataset. This provides a clue that our network has a tendency to classify logos as Food, because it is overrepresented. The image is in `svg` format so it is possible to zoom to get a closer look at the numbers.

References

[1] Simone Bianco, Marco Buzzelli, Davide Mazzini, and Raimondo Schettini (2017). Deep learning for logo recognition. *CoRR*, abs/1701.02620. 1

[2] Christian Eggert, Dan Zecha, Stephan Brehm, and Rainer(2017) Lienhart. Improving small object proposals for company logo detection. *Association for Computing Machinery*. 1

[3] Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, Univeristé de Montréal*, 01 2009. 2

[4] István Fehérvári and Srikar Appalaraju (2018). Scalable logo recognition using proxies. *CoRR*, abs/1811.08009. 1

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 1, 3

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1, 2

[7] Steven Hoi, Wu Xiongwei, Hantang Liu, Yue wu, Huiqiong Wang, Hui Xue, and Qiang Wu. Logo-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks. 11 2015. 1

[8] Forrest N. Iandola, Anting Shen, Peter Gao, and Kurt Keutzer (2015). Deeplogo: Hitting logo recognition with the deep neural network hammer. *CoRR*. 1

[9] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 3

[10] Jonathan Luffarelli, Mudra Mukesh, and Ammara Mahmood (2019). A study of 597 logos shows which kind is most effective. *Harvard Business Review*. 1

[11] Ajkel Mino and Gerasimos Spanakis. Logan: Generating logos with a generative adversarial neural network conditioned on color, 10 2018. 1, 8

[12] Anh Mai Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune (2016). Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *CoRR*. 2, 3, 4

[13] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune (2016). Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *CoRR*. 2, 3

[14] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans, 2017. 3

[15] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization. 2

[16] Stefan "Romberg and Rainer" (2013) Lienhart. Bundle min-hashing for logo recognition. pages 113–120. 1

[17] Stefan Romberg, Lluis Garcia Pueyo, Rainer Lienhart, and Roelof van Zwol. Scalable logo recognition in real-world images. 2011. 1

[18] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei (2014). Imagenet large scale visual recognition challenge. *CoRR*. 2

[19] Alexander Sage, Eirikur Agustsson, Radu Timofte, and Luc Van Gool. Logo synthesis and manipulation with clustered generative adversarial networks. *CoRR*, abs/1712.04407, 2017. 1, 8

[20] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014. 2

[21] Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, Haishuai Wang, and Shuqiang Jiang (2019). Logo-2k+: A large-scale logo dataset for scalable logo classification. *CoRR*, abs/1911.07924. 1, 2, 4

[22] Jason Yosinski, Jeff Clune, Anh Mai Nguyen, Thomas J. Fuchs, and Hod Lipson (2015). Understanding neural networks through deep visualization. *CoRR*. 2, 4

[23] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2017. 6