Detecting Faces in Animes Using Supervised Domain Adaptation

Can Koc Cem Koc Brian Su

Electrical Engineering and Computer Sciences, UC Berkeley
{cankoc, cemkoc, bsu}@berkeley.edu

Abstract

In this paper we explored detecting faces in anime images using pre-trained deep convolutional neural networks trained on ImageNet, YouTube faces and Labeled Faces in the Wild datasets. We constructed a new fully annotated dataset by sampling images from 11 different animes. Extracting features from the layers of pre-trained models before their fully connected layers as inputs, we used various SVM kernels for classification. In this paper we showed which model performs best for this classification task.

1. Introduction

After *Krizhevsky et al.* [7] demonstrated the success of utilizing convolutional neural networks in image classification in 2012, researchers focused more on utilizing CNNs for various tasks such as: Image classification, object detection even depth estimation [7, 11, 6]. It is still very challenging for CNNs to do classification of images from different domains. However, since collecting data is a very demanding task, it is very important for learning algorithms that performed well on a specific task to generalize to different tasks.

In addition, as *Ginosar et al*. [14] puts it, human visual system is incredibly robust in identifying figures in abstract representations such as drawings, cubist paintings where artists intentionally push the envelope to points where familiar figures are deformed and maybe even unrecognizable. Similarly, we believe that in order to realize the full potential even shortcomings of CNNs, we must find new domains where the CNNs fail to recognize as opposed to humans. In this paper we present you our approach of detecting faces in a new domain consisting of anime images. We shall henceforth refer to this new domain as *anime domain* for the rest of this paper.

Compared to detection of faces in real world images, detection of faces in anime images is a harder task due to various differences in drawings (described in section 2.1). Domain adaptations including transfer-learning techniques are necessary when images in target domain differ significantly than images in source domain. In this section we will introduce our problem statement and discuss past work that have been done in this field.

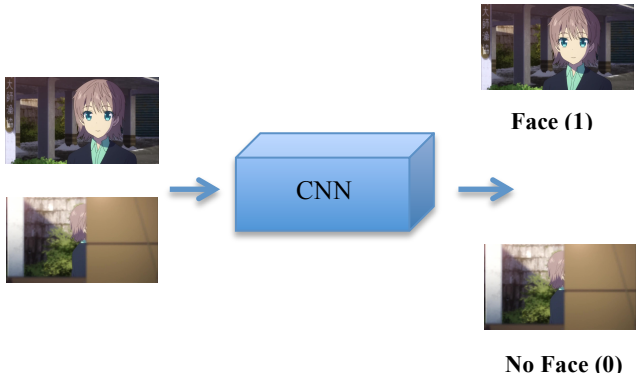

Figure 1: Example face detection process.

1.1. Problem Statement

Our problem is defined as follows: given an image from an anime, determine whether the image contains a face. We define a face in the next subsection.

1.2. Definition of a Face

In this section we describe choices we made in labeling anime images. Mainly we explain what consists of a face in an anime image. Anime images are very different from real world for various reasons. Every image is hand drawn and can contain partial faces or faces that are devoid of certain characteristics such as a clear nose line or lips. Images may also contain deformed animal faces depicting human emotions. In order to build a strong classifier, we need to have a clear definition of what we consider a face:

- Image contains 50% or more of the character's face.
- Can be profile of the character's
- Face doesn't belong to an animal and human facial features are visible such as eye(s), mouth, nose etc.

Please refer to figure 2 for examples.

Figure 2: Example frames and true labels. From left to right, (top): No face (0), No face (0), (bottom): Face (1), Face (1)

2. Data

Constructing a successful model for any new domain requires large amounts of labeled data. Large datasets of annotated images already exist with millions of images for object recognition [2] and for face detection/recognition tasks [12]. However we were unable to find a large dataset consisting anime frames with or without labels. Therefore we decided to construct our own anime faces dataset consisting of ~7k fully labeled images from 11 different Japanese animes. Full list of anime names and example images are shown in Figure 3**.** In this section we describe the intrinsics of our anime faces dataset and we will henceforth refer to it as *AnimeFaces*.

2.1. Data Collection and Preprocessing

Our anime images are collected as high quality (PNG or JPEG) snapshots of video frames. We created a shell script that feeds a randomly chosen episode from the list of aforementioned animes as input to the software FFmpeg. FFmpeg then takes snapshots from the video with frequency 1 frame per second and saves it as a PNG or JPEG format. Most Japanese anime episodes are only ~25 minutes long, hence this process generates ~1500 snapshots per anime episode. Since this paper focuses on fully supervised domain adaptation we hand labeled all the images, splitting them into two categories: "face" (label: 1) and "No face" (label: 0). Former refers to anime frames that contain a distinguishable face and the latter refers to anime frames that don't. We made the clarification of what consists a face in an image and what doesn't in the later parts of this section.

After our collecting raw images from anime episodes and hand labeling them we convert all of them into same format. We choose JPEG as our image format since PNG format produces larger file sizes. Most of the architectures we used (i.e. GoogleNet, AlexNet) accept small image dimensions so our preprocessing stage also converts these images to dimensions accepted by each of these models. (i.e. 224x224 pixels for GoogleNet, 227x227 pixels for AlexNet, 224x224 for VGG16 and VGG19). Additional scaling and mean ImageNet pixel value subtraction are also done.

2.2. Domain Difference

Anime images can differ fundamentally from real world images and realistic paintings. Anime artists utilize different styles from plain caricature drawings and plain sketches, these different styles can focus on deformed facial figures, deformed head to body body ratio. Most of the difference between anime art and real world pictures or drawings is focused on the drawing of the head and face. Common character design convention is the exaggerated eye size in animes. The iconic disproportionately large eye size and exaggerated facial features allows anime faces to be able to depict entire range of human emotions solely through the eyes [3]. However not all animes have large eye sizes, in fact facial drawing styles in animes can change drastically from artist to artist. Colors and shading used in anime drawings are also very different such that, hair, eyes, skin color can be colored not only in different vibrant colors but also in different shades depending on the emphasis intended. In Figure 5 you can observe the most commonly used facial expression drawings used by anime artists. Figure 4 you can see the differences in artists' drawings of faces in different animes and their histograms in the RGB color space.

Domain adaptation problems often arise object detection due to the difference in the images depicting the same object. This difference can often come from a number of reasons including camera types, environment and background differences, light and shading (see [15] for a comprehensive overview). As Ginosar et al. [14] explains, it is very important to evaluate computers and learning algorithms when they are exposed to images that stretch limits of human vision. In this paper we chose to focus on anime faces and anime domain because of these reasons.

3. Our Approach

3.1. Caffe

Caffe is a deep learning framework developed by the Berkeley Vision and Learning Center (BVLC) at UC Berkeley. Caffe has many software packages including CUDA, BLAS and OpenCV and using these it provides high performance for neural network computations. In

particular, we use NVIDIA cudNN to speed up feature extraction from our caffe models.

Caffe also enables feature extraction from any given layer, you just need to provide the name of the layer from the prototxt file you are using to extract data from.

3.2. Keras

Keras is a higher-level abstraction to Theano or Tensorflow. Keras has simple APIs for defining layers in a neural network and has a method to load pre-trained weights for use in further training, or, in our case, domain adaptation. In this paper, we utilized the Theano backend, as it was provided by Keras by default and ready to run in a Stanford CS231 provided AMI image on AWS.

3.3. Feature Extraction

In practice, transfer learning has become a very common method for many computer vision tasks. Depending on the task and data available it has two main scenarios:

a) Using a pre-trained CNN as a fixed feature extractor.
b) Fine-tuning a CNN (as described in [8])

Our goal was to create a classifier that can separate anime images with faces from those without faces. Our target dataset had two key properties:

I. Images taken from different animes
II. ~7k anime images available.

In other words, we had very little data and very different datasets. Therefore we used our pre-trained CNNs as feature extractors and implemented an SVM for the classification task.

We extracted features from 5 different architectures:

1. AlexNet (trained on ILSVRC 14)
2. VGG-16 (trained on ILSVRC 12)
3. VGG-19 (trained on ILSVRC 12)
4. GoogleNet (trained on ILSVRC 14)
5. VGG-Face (trained on LFW Face [11] and YouTube Faces [12])

3.4. Reducing Feature Dimensions

Compared to our dataset size, the dimension output from many of our models would be more than the number of samples in all of our training validation and test sets combined. Therefore we needed to decrease the future dimension. In AlexNet and VGG models, Pool5 layer outputs 265, 6 by 6-feature weights that correspond to a total of 9216 dimension vectors per image. In GoogleNet pool5/7x7_s1 layer outputs 1024, 2 by 2 feature weights that correspond to 4096 dimension vectors (this is demonstrated by the rectified responses in Figure 3). Given our dataset size, we needed a way to decrease the dimension while keeping as much information about image as possible. A common way to do that is to choose the maximum activations per feature patch. This is essentially what max pooling layers do in CNNs. After this, we were able to decrease our feature vector dimensions to 512 for VGG models, 1024 for GoogleNet model and 256 for AlexNet.

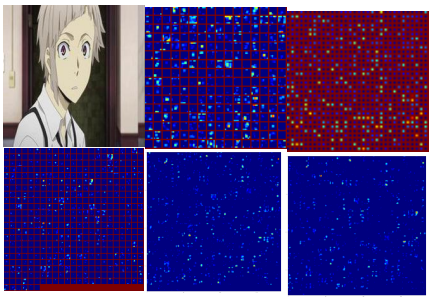

Figure 3: Rectified responses for the layers the features were extracted. Top row, left to right: Chosen Image, AlexNet, GoogleNet. Bottom row left to right: VGG-Face, VGG16, VGG19.

3.5. Classification

After extracting feature vectors for each image in our dataset, we use a Support Vector Machine to solve the classification task of faces and non-faces. Although a simple method, this is recognized by many researchers including [9] as a good starting point.

For the SVM classifier we followed a two-step procedure:

1. Shuffle the all the training set and then partition into validation set.
2. Train an SVM and then try different kernels to increase the accuracy.

Training / Validation / Test Accuracies					
	AlexNet	**GoogleNet**	**VGG 16**	**VGG 19**	**VGG-Face**
Linear SVC	.941 / .922 / .778	1 / .929 / .931	.997 / .891 / .904	.999 /. 901 / .907	.988 / .885 / .646
RBF SVC	1 / .67 / .455	1 / .595 / .607	1 / .615 / .583	1 / .615 / .583	1 / .67 / .455
Poly deg 3	1 / .97 / .787	1 / .956 / .96	1 / .938 / .953	1 / .945 / .95	1 / .958 / .688
Poly deg 4	1 / .969 / .785	1 / .953 / .960	1 / .930 / .94	1 / .935 / .946	1 / .961 / .698
Poly deg 5	1 / .966 / .778	1 / .952 / .961	1 / .923 / .933	1 / .927 / .94	1 / .958 / .726
Poly deg 6	1 / .961 / .770	1 / .953 / .955	1 / .917 / .916	1 / .92 / .924	1 / .956 / .738
Poly deg 10	1 / .93 / .753	1 / .923 / .936	1 / .868 / .853	1 / .867 / .873	1 / .912 / .742
Poly deg 20	.385 / .331 / .545	1 / .832 / .833	.428 / .404 / .439	.428 / .404 / .439	.386 / .331 / .545

Table 1: Each cell represents training, validation and test set accuracies with the respect to the type of SVM used.

This part is explained more comprehensively in the experiments and analysis section but on a high level we implemented our SVM using python scikit-learn package using following kernels.

- LinearSVC (this is just a linear kernel)
- Polynomial Kernel with various degrees (degrees are in the next section)
- RBF Kernel

An important implementation detail is that we increased the default cache size to be 2000MB for training of SVM with polynomial and RBF kernels. This is done in order to speed up the training of our SVM.

4. Experiments

4.1. Different Test Sets

Convolutional Neural Nets that are trained in large datasets such as ImageNet, Pascal VOC have proven to be very successful in a number of tasks including object recognition. This classification accuracy is what convinced people about the power of CNNs [7]. We want models that can be robust in the face of new data from sub-domains where classification problems can stretch even human classifiers to their limits. Ginosar et al. [14] mentions that human visual system is amazingly robust to abstractness of object representations such as cubist art Szegedy et al. [8] discovered that it is very easy to confuse a CNN model with an adversarial example generated from the sign vector of the cost function of the image. For all these reasons we choose to further our experiments of our models by creating two separate test sets. One test set includes images from animes and artists that also appear in the training set. In other words, our test set is sampled from the same distribution of our training set. We shall henceforth refer to this test set as *derivative test set*. However, the other test set is generated by sampling from a different distribution of anime images, consisting of images from not only different animes but also different artists that don't appear on the training sets. We shall refer to this test set as *distinct test set.* In the upcoming sections we present the accuracy and robustness of our algorithms on these two different test sets. Furthermore, we present how our best model GoogleNet performs on the distinct test set.

4.2. Extracted Feature Vectors

There are 3091 train, 1267 validation, and 2099 test images that are randomly shuffled before training the models. For VGG16 and VGG19, we found pre-trained models for Keras and, using its simple API, snipped off the fully connected layers and then ran a forward pass using the train, test images after preprocessing the images by subtracting the means of ImageNet images. With the 512x7x7 output for each image for the VGG-based models, 1024x2x2 for GoogleNet, and 256x6x6 for AlexNet, we take the maximum of each of patches to reduce the dimensions of forward pass output. We then have a (number of images) x 512, 1024, or 256 dimension inputs, depending on the pre-trained model, to our anime classifiers.

	Linear SVM	RBF	Poly deg 2	Poly deg 3	Poly deg 4	Poly deg 5	Poly deg 10	Poly deg 50
Google Net	0.7951	0.4554	0.8494	0.8631	0.8575	0.8585	0.7775	0.4802

Table 2: Performance of GoogleNet on the distinct test set.

4.3. Performance of Different Architectures

For each model (alexnet , googlenet, vggface, vgg16, vgg19) make a table containing.

- LinearSVC (this is just a linear kernel)
- Polynomial Kernel with various degrees (degrees are in the next section)
- RBF Kernel

4.4. Performance on Distinct Test Set

As we have explained previously, we collected another test set called "distinct test set" consisting of anime images sampled from animes that were never included in the training set. Performance on this test set is more interesting because it shows how robust our best model's learning is when tested on different drawings of different artists. You can see this on Table 2.

5. Conclusion

In this paper, we followed a feature extraction approach to adapt pre-trained models on large image datasets to perform face detection in anime images. We created a dataset of several thousands of face/no-face images from several animes for this task. Because of our limited dataset, we took the activations of each pre-trained model before its fully connected layer as our input to our support vector models - RBF, linear, and polynomial models of various degrees. Our results were promising and our top SVM models were able to achieve at least 70% test accuracy for each pre-trained model, with our best accuracies of 95-96% obtained from GoogleNet, VGG16, and VGG19, using polynomial SVMs of degree 3 and 4. Furthermore, on our distinct test set where images sampled from animes that were not present in the training set, our best model GoogleNet achieves 86% accuracy using SVM using polynomial of degree 3.

The similarities and the differences between the real domain and the anime domain provide an interesting domain adaptation problem for the real world trained models. In this paper, we explored the problem of face detection. A natural extension is to perform face recognition - i.e. can we identify characters across various scenes. Not only this

There are much more domain adaptation problems that we'd like to explore beyond this paper. One is simultaneous localization and recognition of objects in the anime domain by using a RCNN. Second is to be able to recognize emotions in anime faces. Although depicted slightly differently than human emotions, a good start for this task may come from Figure 3, which provides a mapping of emotions to faces.

Acknowledgements.
Authors would like to thank Jeff Donahue, Deepak Pathak and Prof. Alyosha Efros for their patience and guidance. Authors would also like to thank Philz Coffee Berkeley for their delicious Philtered Soul.

References

[1] Jia, Yangqing and Shelhamer, Evan and Donahue, Jeff and Karayev, Sergey and Long, Jonathan and Girshick. Caffe: Convolutional Architecture for Fast Feature Embedding. arXiv preprint arXiv:1408.5093. 2014.

[2] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. *IJCV,* 2015.

[3] Poitras, Gilles (2000). *Anime Essentials: Every Thing a Fan Needs to Know*. Stone Bridge Press. ISBN 978-1-880656-53-2

[4] Girshick, R. B., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and

[5] semantic segmentation. CoRR, abs/1311.2524v5, 2014. Published in Proc. CVPR, 2014.

[6] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In Proc. IEEE Conf. Comp. Vis. Pattern Recogn., 2015.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012

[8] Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian J., and Fergus, Rob. Intriguing properties of neural networks. ICLR, abs/1312.6199, 2014b. URL http: //arxiv.org/abs/1312.6199.

[9] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional

activation feature for generic visual recognition. In ICML, 2014.
[10] Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. CNN Features off-the-shelf: an Astounding Baseline for Recognition. *CoRR*, abs/1403.6382, 2014.
[11] J. Donahue, J. Hoffman, E. Rodner, K. Saenko, and T. Darrell. Semi-supervised domain adaptation with instance constraints. In *Computer Vision and Pattern Recognition (CVPR)*, 2013.
[12] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller.
Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments.
University of Massachusetts, Amherst, Technical Report 07-49, October, 2007.
[13] Lior Wolf, Tal Hassner and Itay Maoz
Face Recognition in Unconstrained Videos with Matched Background Similarity.*IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2011.*
pdf
[14] S. Ginosar, D. Haas, T. Brown, and J. Malik. Detecting people in cubist art. In Computer Vision-ECCV 2014 Workshops, pages 101–116. Springer, 2014.
[15] J. Jiang. A literature survey on domain adaptation of statistical classifiers. http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/.
[16] Xiaogang Wang Ziwei Liu, Ping Luo and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), 2015.

Figure 4: Example images from different animes in the AnimeFaces dataset. Observe difference in drawings, exaggerated facial features, shadings and distortions.

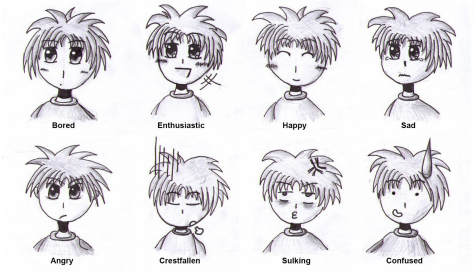

Figure 5: Commonly exaggerated facial features and emotions in Japanese anime drawings

6. Supplementary Material

6.1. Dataset Images

Here are you can find some example images from the Anime Dataset we created for this project. As you see from Figure 4, anime drawings are can be very different from each other and facial figures are generally exaggerated and distorted compared to real world faces.

6.2. List of Animes in AnimeFaces Dataset

1. Bakuman
2. Shinsekai Yori
3. Byousoku 5 Centimeters
4. Boku dake ga Inai Machi
5. Shingeki no Kyojin
6. Kumo no Mukou, Yakusoku no Basho
7. Bungou Stray Dogs
8. Steins Gate
9. Koutetsujou no Kabaneri
10. Nagi no Asukara
11. Ushinawareta Mirai o Motomete