# Predicting a politician's party affiliation from a photo

Diego Garcia-Olano
UT Austin

Amin Anvari
UT Austin

Farzan Memarian
UT Austin

## Abstract

*We have assembled a dataset of around 11 thousand color photos of US state level congressmen from 2011 onward which includes their name, state, and party affiliation among other attributes. In this project we seek to predict a politician's party affiliation based solely on their image. To obtain a baseline of how well people can perform this task on average, we developed an online quiz and after 5000 responses observed an accuracy rate of 65%. Extensive hyper parameter testing over various Convolutional Neural Networks was conducted and our final model is able to obtain an accuracy of 72% underlining the difficulty of the task. We discuss the construction of our model and provide an analysis of our findings. To better understand our results, we ran object detection on our data using models pre-trained on various label sets to see if there exist correlations with the predictions made on our test set. Finally, we implement a boundary equilibrium generative adversarial network (GAN) to create "new" politician images and present our results for various classes.*

## 1. Introduction

A paradigm shift in Computer Vision tasks has occurred in the past few years moving from systems based on hand generated features to ones where features are learned through the use of Deep Convolutional Neural Networks. The process of determining the architecture for these models, the number and types of layers they contain, their activation functions, the associated hyper-parameters, methods to avoid over fitting, and the subsequent training and fitting can be quite time and labor intensive. One of the most powerful ideas to come from deep learning is that its possible to transfer knowledge from a network that has been trained on a given task and apply that knowledge towards a separate task. This process is called transfer learning[10]. It is then beneficial to fine tune the weights of this combined network for the task at hand. This process cuts down dramatically on the amount of time needed to train a model.

In the following project, we seek to utilize these computer vision techniques to see if we can determine a politician's party affiliation based solely on their photo. To that end, we constructed a dataset of US state level politicians based on information available at the Open States Project[1], which was previously run by the Sunlight Foundation and is now available under Creative Commons attribution. The dataset contains images and meta data for both current and inactive politicians from 2011 onward. We discuss the creation of the dataset along with the process of cleaning it and filling in gaps in the next section.

Figure 1: US state level Congressional politicians

We seek to accomplish our main objective by taking a neural network which is trained for image classification on the ImageNet[6] dataset or on the Deep Face dataset [9] and then adapt it for our task. While ImageNet has the advantage of containing over a million labeled images of 1000 different classes, we want to assess whether a network pre-trained on faces may work better for our task. Figure 1 shows sample images from the dataset.

To this end, we use various pre-trained networks that are available in open source deep learning frameworks such as tensorflow and keras [5]. A pre-trained network is simply a saved network previously trained on a large dataset, typically on a large-scale image classification task. If this original dataset is large enough and general enough, then the spatial feature hierarchy learned by the pre-trained network can effectively act as a generic model of our visual world, and hence its features can prove useful for other computer vision problems, even though these new problems might in-

[1] www.openstates.org

volve completely different classes from those of the original task.

We experimented with the following different architectures which are pre-trained on ImageNet: VGG16 [14], VGG19[14], InceptionV3[16], Xception [4], ResNet[7], and InceptionResNetV2 [15]. Additionally, we run experiments with the VGG-Face[9] architecture which is trained on 2.6 million face images. Models based on ResNet and InceptionV3 gave the best single model accuracy of around 69% while those based on VGG19 and VGG-Face gave 67% accuracy and Xception gave accuracy of 65%. Our final model is an ensemble of the 4 ImageNet based classifiers and is the best performing architecture for our party affiliation prediction task giving 72% accuracy on the test set. The rest of the paper is structured in the following manner. In Section 2, we discuss prior works along with the construction of our dataset and the establishing of a human level baseline. In Section 3, we detail our experiments and findings for the party affiliation prediction task. In Section 4, we discuss experiments on augmenting the analysis of our final test results through the use of various object detection mechanisms. In Section 5, we provide details and findings from our implementation of the Boundary Equilibrium Generative Adversarial Network[3] to generate "new" politician images for three groupings present in our dataset, all politicians, republicans only, and democrats only. In Section 6, we present our conclusions.

# 2. Background, Dataset Construction, and Human Baseline findings

To the best of our knowledge this is the first paper which uses only a politician's image to predict their party affiliation. Other similar work focuses on tracking the emotions of politicians during a debate in real time [12], and using face tracking to understand political interactions and derive multiple networks[2]. The work[1] most similar in nature to this one, in preprint as of one month, involves doing object detection using ResNet trained on ImageNet on the Facebook photos of current US Senators and Representatives, of which there are 535, to assess what sorts of objects (ie, labels) each congressmen is associated with. The work then uses these labels to create a document-term matrix where each row has the individual label counts for a politician along with their political party, and then runs a random forest over that to gather what are the most important features (ie, labels) associated with each party. The use of object detection is similar in nature to an extension used here though the other paper does not try different models which would be useful since ResNet is not the current state of the art nor does it include analysis or results for individual politicians. The dataset they provide is impressive for the number of images it contains for each politician, 296 thousand total, however it contains only 535 different politicians total.

## 2.1. Dataset Construction

The OpenStates Project is a wonderful resource for data relating to US politicians particularly how they've voted on bills, bill text, party affiliation, which chamber they serve in, their state, and an accompanying profile image. The site itself utilizes a series of web scrapers which regularly scrape the official state government websites of each state to keep its data up to date. The site itself provides a thumbnail version of each politician image it processes, but additionally provides the url to the original full size image link itself. OpenStates allows for bulk downloads of its text information, but not images, and also distinguishes between "active", ie. politicians currently in office, and "inactive" politicians which affects our purposes since "inactive" politicians have no "party affiliation" data associated with them, since it is possible that a politician may have changed parties during the span of their careers.

Our process for constructing our data set is as follows: We first downloaded the legislator csv files for each state from openstates.org and combined them into a single csv file. This file contained 13994 legislators total of which 7405 are active and 6289 are inactive. The active legislators contain party affiliation and the breakdown is 3156 democrats, and 4249 republicans. For each legislator we then attempted to download their original profile image to get an bigger image than the thumbnail versions available on the site. Some legislator rows did not contain original image urls and for broken image links, we attempted to scrape the thumbnail version. We then wrote scrapers to attempt to get the party affiliations based on their last held position for the 6289 inactive members from the openstates profile pages. At this point we had images and party affiliation for around 12,500 politicians. We then removed third party candidates, politicians from US territories, and black and white and gray scale images. Finally we only included images with widths of at least 100 pixels. After this final processing step, we had a dataset of 11090 color images with party affiliations of which 6159 were republican (55%), 4931 were democrat, and 6463 were active.

## 2.2. Human Baseline

As the final dataset contains images of which 55% are republican, any classifier simply always guessing republican could get that level of accuracy on average. In order to see how good people are at predicting a politician's party affiliation from an image we constructed an online[2] web quiz and after 5000 responses, we observed that on average people obtained an accuracy of 65%.

[2] http://www.diegoolano.com/demorepu/

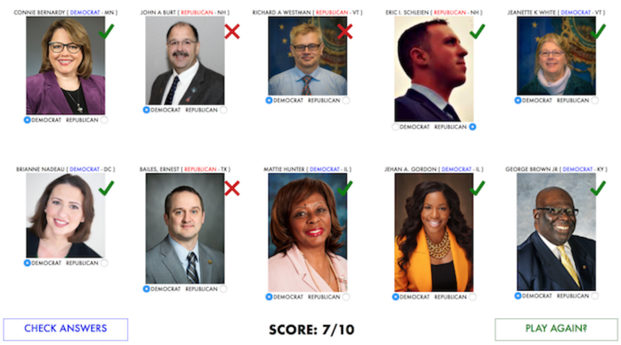



Figure 2: Quiz to determine baseline for predicting party

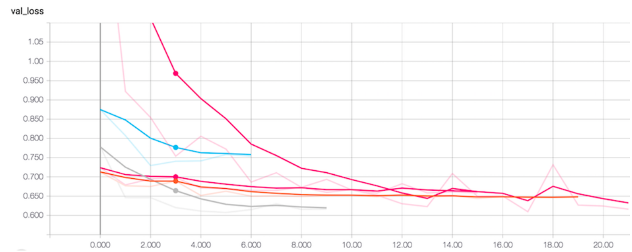



Figure 3: Validation loss: We see as the model is being trained, it is going down.

## 3. Party Affiliation Experiments

The following section discusses how transfer-learning, fine-tuning, hyper parameter tuning, different optimizers, learning rates, regularization through data augmentation and batch normalization were used in the training and validation of our models.

### 3.1. Methods

We pose the task of predicting party affiliation as a binary classification task with republican and democrat classes. We set aside about 15% of the data as the test set and use the remaining 75% for training and 10% for validation. The training/validation data retained the full datasets distribution of being about 55% republican.

We use the Keras implementation of the VGG19, VGG16, inceptionV3, Xception, ResNet, and Inception-ResNetV2 network architectures that are pretrained on the ILSVRC dataset and additionally used a VGG-Face implementation in keras[3] that was pretrained on 2.6 million face images.

Since the size of the training dataset was moderately large, we decided to train the network in a pre-training and fine-tuning fashion. In short, we have taken the knowledge learned from an image recognition task on the ImageNet dataset or faces dataset and transferred it to the party affiliation prediction task. This is helpful because a lot of the low level features such as detecting edges and detecting curves, might help the learning algorithm do better in our task. We fine-tune the final fully connected layers of the networks on the politician image training data to predict party affiliation. We ran experiments training our networks for 50 to 100 epochs using the ADAM and RMSPprop optimizers with various learning rates ranging from .01 to .00001 in the pre-training step using categorical crossentropy loss. We set standard values for the momentum term $\beta = 0.9$ and Adam parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.

[3] https://github.com/rcmalli/keras-vggface

We additionally performed standard data augmentation of the images to help with our prediction task and to prevent overfitting. The learning rate was additionally reduced via keras' ReduceLROnPlateau callback function which monitors the model's validation loss and reduces the learning rate when that value has stayed constant for a set number of epochs.

We applied batch normalization using batches of size 128. Batch normalization makes the neural network much more robust to the choice of hyperparameters and it enables us to much more easily train even very deep networks. We did the batch normalization before the activation function meaning we normalized Z values rather than activations.

Due to sheer number of hyper-parameters, instead of doing a grid search, we decided to try to make a random sampling search to more richly explore the set of possible values and to make it more likely to find a value that works well. We also used a coarser to finer search scheme. So we started with a coarse random sampling first and then zoomed in and sampled more densely within that space.

### 3.2. Model Evaluations

Of the different models we ran the following table summarizes the best five in terms of test accuracy.

| model | learning rate | test accuracy |
|---|---|---|
| inceptionv3 | 0.0009 | 0.692 |
| resnet | 0.0001 | 0.691 |
| vgg19 | 0.001 | 0.670 |
| vggFace | 0.00009 | 0.677 |
| xception | 0.0009 | 0.657 |

The Adam optimizer generally worked better than RMSProp and was used in the five cases above, and each of the final experiments was allowed to run for 100 epochs. You can see plots of validation loss and validation accuracy in Figures 3 and 4 respectively.

### 3.3. Final Ensemble model

Ensembles of our best models were considered to see which obtained the best final results. The following ta-

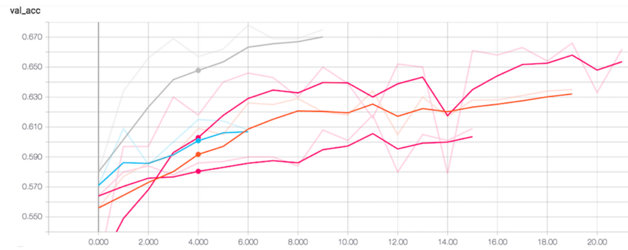



Figure 4: Validation accuracy: We see as the model is being trained, the accuracy is going up.

ble shows the combinations considered and how they performed in terms of overall test accuracy, and the test accuracy for the republican and democrat classes:

| models | test acc | repub acc | dem acc |
|---|---|---|---|
| inv3,res | 0.697 | 0.791 | 0.582 |
| inv3,res,v19 | 0.707 | 0.791 | 0.605 |
| inv3,res,v19,xcpt | 0.721 | 0.784 | 0.641 |
| inv3,res,v19,xcpt,vface | 0.722 | 0.798 | 0.629 |

Because the VGG-Face model requires a very small learning rate to perform well, its training time is substantial. This fact along with the .01 improvement in overall accuracy it provides does not justify its inclusion here. The final model thus is based on the ensemble of the 4 imagenet based models (incepctionv3,resnet,vgg19 and xception) using a soft majority vote prediction where the probability of each model's prediction was taken into account equally in determining the final prediction. This ensemble resulted in .72% test accuracy.

The confusion matrix over the test set is shown in Figure 5. We can see the system does a much better job at predicting Republican's accurately (78%) as compared with Democrats (64%).

### 3.4. Test Set Findings

In the appendix 1, we show test result images pertaining to the four circumstances where the model has high confidence in its predictions and it correctly predicts democrat (top left), it correctly predicts republican (top right), it incorrectly predicts republican when the true class is democrat (bottom left) and it incorrectly predicts democrat when the true class is republican(bottom right). The top left result of democrats predicted correctly with high confidence appears more ethnically diverse and has a higher percentage of women, whereas the top right result of republicans predicted correctly with high confidence appears less diverse and less represented by women. The bottom left image set of democrats incorrectly predicted as republicans with high confidence is more diverse in gender and ethnicity than the top right result of images and also interestingly contains many female politicians in bright blue clothing, the color traditionally associated with the Democratic party. The right bottom result of republicans incorrectly predicted as democrats is also interesting in that it contains a large portion of women as well and additionally women in red clothing. From these results, we can posit that women are generally harder to classify and that at least for women politicians, the model associates red clothing with republicans and blue clothing with democrats. Its important to note that neither race nor gender was fed into the model.

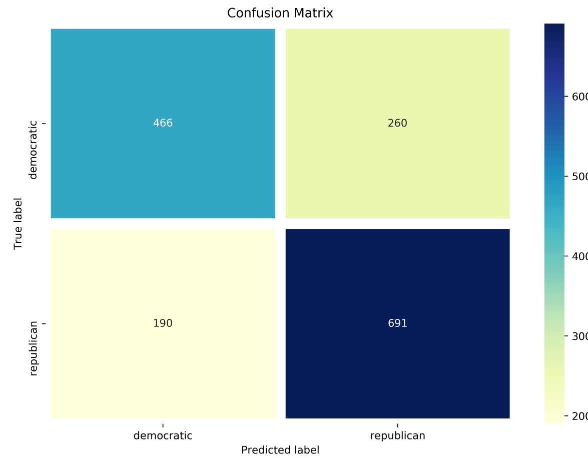



Figure 5: Confusion matrix on test data

### 3.5. Visualization of Network

In the following section, we catalog a method that can be used to determine what the model considers to be important regions in a given image using a technique known as Gradient-weighted Class Activation Mapping (Grad-CAM)[13]. This method can be used as a way of verifying what the model is basing its prediction on most heavily for a given image. For instance, Figure 6 shows an activation map for a random Democrat from the training set and we observe his tie and lower face seem to be important. Using this technique is useful however it nonetheless requires extensive manual inspection and is left for future work.

## 4. Object Detection

To gain a better understanding of whether there exist correlations between our dataset as a whole, though more precisely between our test predictions, and objects appearing in a politician's image, we decided to utilize a few object detection models which have been pre-trained on different image sets and labels. The idea is to first pull out high

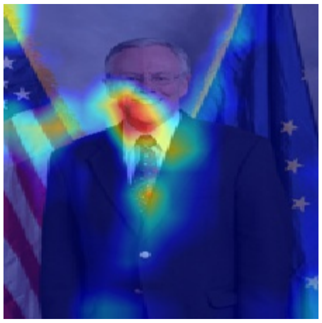


Figure 6: Gradient-weight CAM visualization

probability object labels and then see if any corresponded to higher rates of correct or incorrect classification. We utilized 3 systems: the YOLOv2 and YOLO9000[11] systems which are state-of-the-art, real-time object detection system trained on Microsoft's COCO dataset[8] with 80 labels and the YOLO9000 set with 9000 labels respectively, and ResNet which was trained on ImageNet with 1000 labels. With each system, we ran object detection over every image in our data set and then did manual inspection over pertinent labels to detect if the systems were doing a good job of correctly classifying relevant information.

### 4.1. YOLO

In our tests, YOLOv2 was very good at identifying "person" and "tie", but not much else since COCO's label are for the most part not of interest. 19,695 objects were detected in 10,663 out of our total 11 thousand images. Keeping only objects with probability greater than 30% we are left with 18,560 labels. Of these, 57% were "person" labels, and 41% were "tie" label. The remaining 2% (ie, 371 labels) are mainly composed of innocuous labels such as chair(50), dining table(30), bottle (24), book(21) cell phone(17), etc. What the system lacks in interest, it however makes up, for the most part, in precision. For instance, figure 7 shows an image which contains 4 "person" labels, which initially was counterintuitive since we were not expecting more than a single person in a given politician's image, but in this instance we see that 3 people are identified in the photos in the background of the image.

We have put up two web pages to allow readers to explore the objects detected over the images in the training/validation set[4] and the test set[5]. Although imperfect, in looking at our results from these pages, "tie" can be thought of as a rough proxy for "male" and "person" can be thought of as a proxy for overall accuracy.

[4]goo.gl/JrWmoX

[5]goo.gl/t16HCr


Figure 7: YOLO object detection

| label | all | republican | democrat |
|---|---|---|---|
| tie | 7646 (71.6%) | 4618 (78.3%) | 3028 (63.4%) |
| person | 10641 (99.7%) | 5884 (99.7%) | 4757(99.6%) |

Looking at the above table we can roughly say that republican's are 78% male while democrats are 63% male which seems to be in line with our findings about party diversity, and republicans and males being easier to classify as seen in section 3. In examining our test results, 70% of the images contained ties and of these our system was able to correctly predict 75% of them while it was only able to obtain 66% accuracy on images without ties.

Additionally, it turns out that the system agrees that most, but not all, politicians are human since we see that 99.7% of images were assigned the "person" label. We set up another page[6] to allow readers to explore these 35 images that were not assigned "person" labels with greater than 30% probability. When YOLOv2 fails, it fails spectacularly as seen by the "teddy bear" in figure 8 which is a part of the aforementioned group.

### 4.2. YOLO9000 and ResNet

In our experiments, the YOLO9000 object detection system was much more sensitive to probability cutoffs since it contains 9000 possible labels it could detect. It also predicted few classes like YOLOv2 however the added noise of many of the labels being of very low probability gave too many nonsensical results to be able to utilize without a lot of manual inspection. Most of the high probability labels were "whole", or "neckwear" followed by "object", "instrument", "worker", and "commodity" which were not of interest to our task so we did not explore YOLO900 further.

ResNet also suffers from probability cutoffs needing to

[6]https://goo.gl/gXbTRH


Figure 8: YOLO detecting a ”teddy bear”

Figure 9: RESNET detecting a ”boa constrictor”

be lowered significantly to obtain interesting results thus leading to highly noisy labels. Its also interesting to note that although some labels are always wrong no matter their probability, for instance ”bulletproof vest”, ”military uniform”, ”oboe”, ”wig” and ”bassoon” are always in the top 15 most detected object and were never correct when found, there do exist other labels which are almost always correct even if their probability is very low, such as ”bolo-tie”, ”cowboy hat”, and ”flag pole”. The issue with the very low probability labels outside of ”bolo-tie” which along with ”windsor tie” and ”bow tie” show up frequently, is that they rarely show up. For instance, using an extremely low cutoff of 1% (ie, use all label predictions greater than that thresh hold ) 31,829 labels are discovered of which only 61 are ”flagpole” even though its clear even from the example images in the appendix that flags are a common occurrence in this data set. Similarly only 21 ”cowboy” hats are discovered. ”Bolo tie” and ”Bow tie” occur 1105 and 926 times respectively and could be useful in prediction and that is left for future work. ResNet also makes some wonderfully bad detection predictions such as always predicting ”neck brace” for images where politicians are wearing turtle necks, and similarly predicting a scarf to be a ”boa constrictor” in figure 9.

## 5. GANs

As an extension to our main task of predicting party affiliation, we ran boundary equilibrium Generative Adversarial Networks (beGAN) over our dataset to construct ”new” fake politicians. GANs are notoriously finicky to hyper parmeter settings, and as such we mostly stuck to those used in the original paper since varying the learning rate or batch size gave wildly unstable results. We ran 4 different experiments, 3 using the same GAN settings for 64x64 images and one updated to use 128x128 images. Our code was mostly based on the following implementation [7] The 3 64x64 image GAN experiments are for all politicians[8], just republicans[9] and just democrats[10] . Each of these experiments ran for a day and selected results can be seen in the appendix, while the full results for selected epochs can be seeing in the links provided.

To verify the images produced from the first experiment were indeed unique ”new” images, we ran a nearest neighbor algorithm to determine the 3 closest images from the original dataset for each nearly generated ”fake” politician for one of the better epochs and the results can be viewed in Figure 10 or on the website[11]. We can see that the images produced by the GAN are indeed unique. In looking at the first ”all politicians” experiment, we can see that the images produced look as though they'll be diverse at the beginning, a few seem like they could be women or non-white, however as the epochs continue, their is a ”whitening male” effect which takes place and by 60 epochs all the politicians have become white males. This is a reflection of the distribution of images in our dataset being more white and male. At about 150 epochs the generator network begins to incorporate more variance and produce more degenerate results probably as a reflection of not being able to advance its results against the discriminator network. This phenomena can be seen in the ”failing” bottom right result. For the ”republican only” network, we observe that the generator network quickly converges to the all white males result from before whereas for the ”democrat only” network, we observe that the generator network has a harder time converging to clear results in general and the results look more diverse than in the ”all politicians” or ”republican only” images. This later finding is probably again due to the democrat image set being less homogeneous than the republican one.

[7] github.com/mokemokechicken/keras_BEGAN
[8] https://goo.gl/RRZnzz
[9] https://goo.gl/P6rDF2
[10] https://goo.gl/cwPY3B
[11] https://goo.gl/paGbJ4


Figure 10: Nearest neighbors to GAN output for all politicians

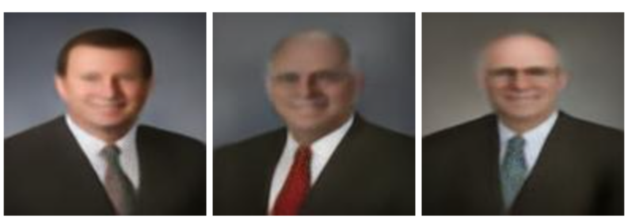


Figure 11: 128x128 GAN output for all politicians

For the 128x128 experiment, we see that the results are similar to those of the 64x64 experiment, except the imperfections are now more visible to the eye and the training time jumps up. This experiment was allowed to run for a little less than 4 days. Figure 11 shows a few of the more realistic fake politicians.

## 6. Conclusion

In this work, we constructed a dataset of 11 thousand color photos of US state level politicians along with associated meta data including party affiliation. We created a website to allow people to see how well they could perform the task of predicting party affiliation based on just a photo and obtained a baseline of 65% accuracy. We then ran experiments on several different deep convolutional neural network architectures to see which gave the best validation accuracy results. Instead of trying to build a model from scratch and training it on the limited amount of data we had access to, we took advantage of the availability of the most successful networks trained on large amounts of images, both ImageNet and the DeepFace dataset. We used transfer learning for this purpose, i.e. we took a trained ConvNet, froze the convolutional layers, replaced the fully connected layers and trained the model to learn the weights for the fully connected layers. We performed fine tuning on the last few convolutional layers. We then created an ensemble of our best models and reached our final best model which obtained an accuracy of 72% on the test set.

Additionally, we performed object detection as a way of further understanding our test results and showed findings for three different systems (YOLO, YOLO9000 and ResNet). Finally, we implemented boundary equilibrium generative adversarial networks (beGAN) on our full data set to construct ”new” politicians and showed our results.

## References

[1] L. J. Anastasopoulos, D. Badani, C. Lee, S. Ginosar, and J. R. Williams. Political image analysis with deep neural networks. 2017.

[2] D.-D. L. B. Renoust. Visual analytics of political networks from face-tracking of news video. *IEEE Trans. Multimedia.*, 2016.

[3] D. Berthelot, T. Schumm, and L. Metz. BEGAN: boundary equilibrium generative adversarial networks. *CoRR*, abs/1703.10717, 2017.

[4] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016.

[5] F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[9] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[10] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014.

[11] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[12] A. Ruiz-Garcia, M. Elshaw, A. Altahhan, and V. Palade. *Deep Learning for Emotion Recognition in Faces*, pages 38–46. Springer International Publishing, Cham, 2016.

[13] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[15] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

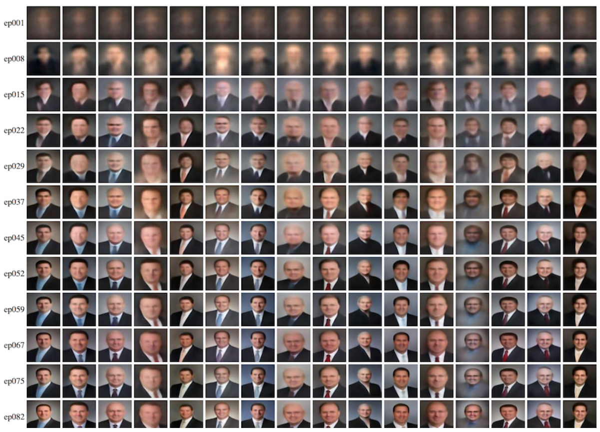
(a) Predicted as Democratic Correctly

(b) Predicted as Republican Correctly

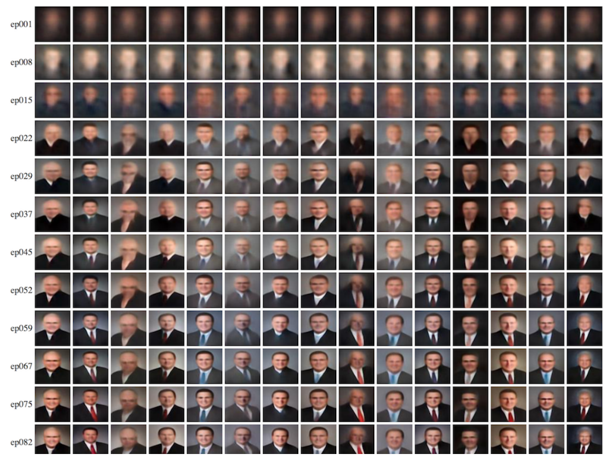
(c) Democrats incorrectly predicted as Republican

(d) Republicans incorrectly predicted as Democrat

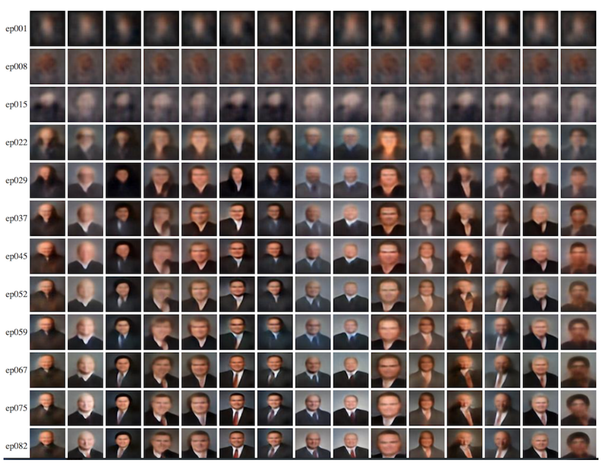
Figure 12: High Confidence Predictions



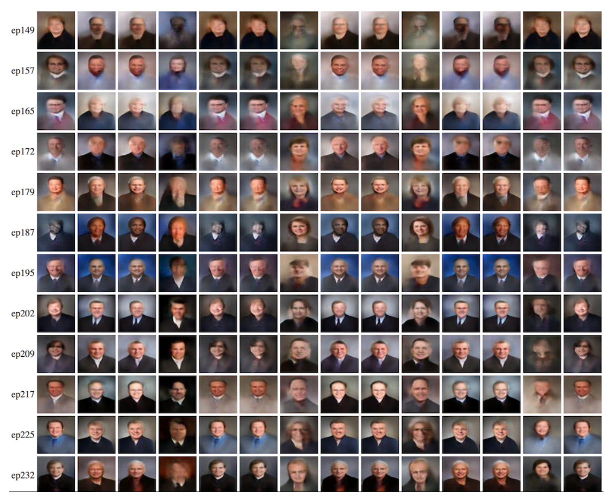

(a) All Politicians




(b) Republicans




(c) Democrats




(d) All Politicians failing

Figure 13: Fake Politicians Produced by beGAN

Figure 14: 128x128 GAN output for all politicians